LECTURE NOTES IN ELECTRICAL ENGINEERING | 8

Danica Kragic
Ville Kyrki
Editors

Unifying Perspectives in Computational and Robot Vision

Springer

Unifying Perspectives in Computational and Robot Vision

Lecture Notes in Electrical Engineering

Sensor and Ad-Hoc Networks
Makki, S.K.; Li, X.-Y.; Pissinou, N.; Makki, S.; Karimi, M.; Makki, K. (Eds.)
2008, Approx. 350 p. 20 illus., Hardcover
ISBN: 978-0-387-77319-3 , Vol. 7

Trends in Intelligent Systems and Computer Engineering
Castillo, Oscar; Xu, Li; Ao, Sio-Iong (Eds.)
2008, Approx. 750 p., Hardcover
ISBN: 978-0-387-74934-1, Vol. 6

Advances in Industrial Engineering and Operations Research
Chan, Alan H.S.; Ao, Sio-Iong (Eds.)
2008, XXVIII, 500 p., Hardcover
ISBN: 978-0-387-74903-7, Vol. 5

Advances in Communication Systems and Electrical Engineering
Huang, Xu; Chen, Yuh-Shyan; Ao, Sio-Iong (Eds.)
2008, Approx. 700 p., Hardcover
ISBN: 978-0-387-74937-2, Vol. 4

Time-Domain Beamforming and Blind Source Separation
Bourgeois, J.; Minker, W.
2009, approx. 200 p., Hardcover
ISBN: 978-0-387-68835-0, Vol. 3

Digital Noise Monitoring of Defect Origin
Aliev, T.
2007, XIV, 223 p. 15 illus., Hardcover
ISBN: 978-0-387-71753-1, Vol. 2

Multi-Carrier Spread Spectrum 2007
Plass, S.; Dammann, A.; Kaiser, S.; Fazel, K. (Eds.)
2007, X, 106 p., Hardcover
ISBN: 978-1-4020-6128-8, Vol. 1

Danica Kragic - Ville Kyrki
Editors

Unifying Perspectives in Computational and Robot Vision

Springer

Danica Kragic
Royal Institute of Technology (KTH)
CSC – CAS/ CVAP
SE-100 44 Stockholm
Sweden

Ville Kyrki
Lappeenranta University of Technology
Department of Information Technology
P.O. Box 20
53851 Lappeenranta
Finland

Library of Congress Control Number: 2007941797

ISBN 978-0-387-75521-2 e-ISBN 978-0-387- 75523-6

Printed on acid-free paper.

9 8 7 6 5 4 3 2 1

springer.com

To all people interested in visual algorithms that work.

Preface

The field of computer vision has developed significantly over the past few years and its importance for robotics is also growing. Due to many practical applications in a diverse variety of sub-disciplines, such as surveillance, robot control, and virtual environments, vision algorithms have a significant application impact. The robotics and computer vision communities have, however, remained largely separate. Dialog between the two communities would then contribute greatly to the progress in both disciplines. In many cases, methods available in the computer vision community are not in general use in the robotics community. On the other hand, the system level perspective is often neglected in the computer vision community, because the research is focused on individual problems and algorithms. There is consequently a need to bring the communities of robot and computer vision to a joint appreciation of the value of systems, where there is a need to consider all aspects from perception to action generation. A workshop "From Features to Actions: Unifying Perspectives in Computational and Robot Vision" was organized by the editors at the IEEE International Conference on Robotics and Automation (Rome, April 2007) with the objective of bringing together computer vision and robotics researchers. This book intends to continue exploring the benefits of dialog by highlighting current challenges and novel approaches to the above issues.

October 2007

Danica Kragic
Ville Kyrki

Contents

Contributors

Benedetto Allotta
Department of Energetics “Sergio Stecco”,
Universitaá degli Studi di Firenze,
Italy.

Tamim Asfour
Industrial Applications for Informatics and Microsystems (IAIM),
Institute for Computer Science and Engineering,
University of Karlsruhe,
P.O. Box 6980, D-76128 Karlsruhe, Germany.
e-mail: asfour@ira.uka.de

Pedram Azad
Industrial Applications for Informatics and Microsystems (IAIM),
Institute for Computer Science and Engineering,
University of Karlsruhe,
P.O. Box 6980, D-76128 Karlsruhe, Germany.
e-mail: azad@ira.uka.de

Cyrille Berger
LAAS-CNRS,
University of Toulouse,
France.
e-mail: Cyrille.Berger@laas.fr

Oliver Brock
Computer Science Department,
University of Massachusetts Amherst,
Amherst, MA, USA.
e-mail: oli@cs.umass.edu

Herman Bruyninckx
Department of Mechanical Engineering,
Katholieke Universiteit Leuven,
Celestijnenlaan 300B, 3001 Heverlee, Belgium.

Darius Burschka
Department of Computer Science,
Technische Universität München,
80333 München, Germany.
e-mail: burschka@cs.tum.edu

Tinne De Laet
Department of Mechanical Engineering,
Katholieke Universiteit Leuven,
Celestijnenlaan 300B, 3001 Heverlee, Belgium.

Joris De Schutter
Department of Mechanical Engineering,
Katholieke Universiteit Leuven,
Celestijnenlaan 300B, 3001 Heverlee, Belgium.

Rüdiger Dillmann
Industrial Applications for Informatics and Microsystems (IAIM),
Institute for Computer Science and Engineering,
University of Karlsruhe,
P.O. Box 6980, D-76128 Karlsruhe, Germany.
e-mail: dillmann@ira.uka.de

Christophe Doignon
Control, Vision and Robotics Team,
LSIIT (UMR ULP-CNRS 7005),
University of Strasbourg,
Boulevard Brant, 67412 Illkirch, France.
e-mail: doignon@lsiit.u-strasbg.fr

Duccio Fioravanti
Department of Energetics “Sergio Stecco”,
Universitaá degli Studi di Firenze,
Italy.

J. J. Guerrero
University of Zaragoza,
I3A - Department of Informatics and Systems Engineering,
50010 Zaragoza, Spain.

Dov Katz
Computer Science Department,
University of Massachusetts Amherst,
Amherst, MA, USA.
e-mail: dubik@cs.umass.edu

Danica Kragic
Centre for Autonomous Systems,
Computational Vision and Active Perception,
School of Computer Science and Communication,
Royal Institute of Technology,
10044 Stockholm, Sweden.
e-mail: dani@kth.se

Ben Kröse
ISLA Lab,
University of Amsterdam,
Kruislaan 403, 1098SJ Amsterdam, The Netherlands.
e-mail: krose@science.uva.nl

Alexandre Krupa
IRISA/INRIA Rennes (Lagadic Team),
Campus de Beaulieu,
35042 Rennes, France.
e-mail: alexandre.krupa@irisa.fr

Ville Kyrki
Department of Information Technology,
Lappeenranta University of Technology,
P.O. Box 20, 53851 Lappeenranta, Finland.
e-mail: kyrki@lut.fi

Simon Lacroix
LAAS-CNRS,
University of Toulouse,
France.
e-mail: Simon.Lacroix@laas.fr

Thomas Lemaire
LAAS-CNRS,
University of Toulouse,
France.
e-mail: Thomas.Lemaire@laas.fr

James Little
Department of Computer Science,
University of British Columbia,
ICCS 117, 2366 Main Mall, Vancouver, B.C., Canada V6T 1Z4.
e-mail: little@cs.ubc.ca

Jiri Matas
Center for Machine Perception,
Department of Cybernetics,
Faculty of Electrical Engineering,
Czech Technical University in Prague,
Karlovo nám. 13, 121 35 Prague, Czech Republic.
e-mail: matas@cmp.felk.cvut.cz

Larry Matthies
Jet Propulsion Laboratory,
California Institute of Technology,
Pasadena, California, USA.
e-mail: lhm@jpl.nasa.gov

Benjamin Maurin
Cerebellum Automation Company,
178, route de Cran Gevrier, 74650 Chavanod, France.
e-mail: maurin@eavr.u-strasbg.fr

A. C. Murillo
University of Zaragoza,
I3A - Department of Informatics and Systems Engineering,
50010 Zaragoza, Spain.
e-mail: acm@unizar.es

Florent Nageotte
Control, Vision and Robotics Team,
LSIIT (UMR ULP-CNRS 7005),
University of Strasbourg,
Boulevard Brant, 67412 Illkirch, France.
e-mail: nageotte@lsiit.u-strasbg.fr

C. Sagüés
University of Zaragoza,
I3A - Department of Informatics and Systems Engineering,
50010 Zaragoza, Spain.

Gabe Sibley
Robotic and Embedded Systems Laboratory,
University of Southern California,
Los Angeles, California, USA.
e-mail: gsibley@usc.edu

Ruben Smits
Department of Mechanical Engineering,
Katholieke Universiteit Leuven,
Celestijnenlaan 300B, 3001 Heverlee, Belgium.
e-mail: ruben.smits@mech.kuleuven.be

Jan Sochman
Center for Machine Perception,
Department of Cybernetics,
Faculty of Electrical Engineering,
Czech Technical University in Prague,
Karlovo nám. 13, 121 35 Prague, Czech Republic.
e-mail: sochmj1@cmp.felk.cvut.cz

Tristram Southey
Department of Computer Science,
University of British Columbia,
Vancouver, BC, Canada.
emailsouthey@cs.ubc.ca

Gaurav Sukhatme
Robotic and Embedded Systems Laboratory,
University of Southern California,
Los Angeles, California, USA.
e-mail: gaurav@usc.edu

Aleš Ude
* Dept. of Automatics, Biocybernetics, and Robotics,
Jožef Stefan Institute,
Jamova 39, 1000 Ljubljana, Slovenia.
e-mail: ales.ude@ijs.si

Kai Welke
Industrial Applications for Informatics and Microsystems (IAIM),
Institute for Computer Science and Engineering,
University of Karlsruhe,
P.O. Box 6980, D-76128 Karlsruhe, Germany.
e-mail: welke@ira.uka.de

Zoran Zivkovic
ISLA Lab,
University of Amsterdam,
Kruislaan 403, 1098SJ Amsterdam, The Netherlands.
e-mail: zivkovic@science.uva.nl

Chapter 1
Recent Trends in Computational and Robot Vision

Ville Kyrki and Danica Kragic

1.1 Introduction

There are many characteristics in common in computer vision research and vision research in robotics. For example, the Structure-and-Motion problem in vision has its analog of SLAM (Simultaneous Localization and Mapping) in robotics, visual SLAM being one of the current hot topics. Tracking is another area seeing great interest in both communities, in its many variations, such as 2-D and 3-D tracking, single and multi-object tracking, rigid and deformable object tracking. Other topics of interest for both communities are object and action recognition.

Despite having these common interests, however, "pure" computer vision has seen significant theoretical and methodological advances during the last decade which many of the robotics researchers are not fully aware of. On the other hand, the manipulation and control capabilities of robots as well as the range of application areas have developed greatly. In robotics, vision can not be considered an isolated component, but it is instead a part of a system resulting in an action. Thus, in robotics the vision research should include consideration of the control of the system, in other words, the entire perception-action loop. A holistic system approach would then be useful and could provide significant advances in this application domain.

We believe that although there have been good examples of robust vision systems, there is a gap between the research conducted in computer vision and robotics communities. In the following, we aim to identify some of the recent developments where we see great potential for the co-operation between the communities.

Ville Kyrki
Lappeenranta University of Technology, Department of Information Technology, P.O. Box 20, 53851 Lappeenranta, Finland, e-mail: kyrki@lut.fi

Danica Kragic
Centre for Autonomous Systems, Computational Vision and Active Perception, School of Computer Science and Communication, Royal Institute if Technology, 10044 Stockholm, Sweden, e-mail: dani@kth.se

1.2 Perception and Action

One of the main challenges in the field of robotics is to make robots ubiquitous. To intelligently interact with the world, robots need to perceive and interpret the environment and situations around them and react appropriately. In other words, they need context-awareness. But how to equip robots with capabilities of gathering and interpreting the necessary information for novel tasks through interaction with the environment and by providing some minimal knowledge in advance? This has been a longterm question and one of the main drives in the field of cognitive system development. For a service robot that is to perform tasks in a human environment, it has to be able to learn about objects and object categories. However, the robots will not be able to form useful categories or object representations by being only passive observers of the environment. They should, like humans, learn about objects and their representations through interaction.

The simplest type of interactions that can occur between a robot and an object may be to, for example, push an object in order to retrieve information about the size or weight of the object. Here, simple visual cues providing approximate 3D position of the object may be sufficient. A more complex interaction may be to grasp the object for the purpose of gaining the physical control over the object. Once the robot has the object in its hand, it can perform further actions on it, such as examining it from other views. Information obtained during interaction can be used to update the robots representations about objects and the world.

Such an approach is studied in Chapter 2 where a mobile manipulator system employs interactive perception to extract kinematic models from tools such as pliers or shears. The extracted models are then used to compute an action that transforms the kinematic attributes of the object to actions that can be performed on them thus mimicking tool use.

Another way of learning of how to interact with the environment is to observe humans or other agents and perform actions through a process of *imitation*, [4]. Imitation learning in robotics is therefore strongly related to action representation and recognition. The overall goal of imitation learning is to develop robots that are able to relate perceived actions of another agent to its own embodiment in order to learn, recognize and finally perform the demonstrated actions [8, 40, 41, 19, 29, 30, 22, 11, 6]. Most of the work work on imitation in robotics is motivated by human findings, where there is strong neurobiological evidence that human actions and activities are directly connected to the motor control of the human body [14, 36, 37].

Detection of human body and body part, recognition and interpretation of their actions has therefore during the past few years gained considerable amount of attention, [1, 2, 13, 48, 24, 47]. Main motivation is the large number of potential applications, e.g., in visual surveillance, entertainment industry, robot learning and control due to the ability to acquire and store a large amount of data that can be processed offline.

In visual surveillance applications, the work is mainly concentrated on classification of common versus uncommon actions. In the entertainment industry, the interest lies mainly in the field of motion capture and synthesis where precise motion capture

allows to replace an actor with a digital avatar. Ideally, the motion capture system should be non-intrusive thus making vision based techniques a natural solution.

Even if there are differences in the applications, the system design is very similar: the sensory input has to be acquired and represented so to enable i) the recognition of the observed actions and ii) understand the effects certain actions may have on the environment. In robotics, there is an additional dimension since the system also has to enable the robot to physically perform a certain action in order to cause the desired change in the environment. The last point point depends on the individual/robot under consideration: how to perform an action that causes a particular change in the environment for a human and a robot depends on their physical capabilities.

Chapter 3 presents an approach for people detection with a mobile robot through combination of visual and laser range scanner sensors. It is shown how a person may be represented as a constelation of body parts to facilitate the detection process.

Chapter 4 presents a system where the road towards developing cognitive capabilities in robots is followed by considering an interply between perception and action. A humanoid robot with perception, manipulation and communication skills is described with a special attention paid on the design of the robot head as the base for prividing the input to various visuo-motor behaviors. The presented work spans from human motion tracking and object representation to action imitation and adaptation.

One of the important aspects of integrating the results from the computer vision community on mobile robots that have constrained processing and storage capabilities, is to consider how to adapt complex processing methods for real-time applications. In Chapter 5, it is shown how time-constrained classification, detection and matching problems may be formalised in the framework of sequential decision-making. The work derives quasi-optimal time-constrained solutions for the three problems of major relevance for the community: feature detection, feature mathcing and face detection.

Chapter 6 presents an application where the real-time processing aspects are of utmost importance - a medical application intended for 3D positioning and guidance of surgical instruments in the human body. The work shows the importance of real-time visual processing for the purpose of vision guided control where model based techniques are adopted to facilitate the tracking process.

1.3 Mapping the Environment — SLAM and vSLAM

One of the essential capabilities of an autonomous mobile robot is to move around in its environment. To accomplish this in complex natural environments, the robot needs the ability to build maps of the environment using natural landmarks and to use them for localization [44, 7, 10, 43, 45]. One of the current research topics related to Simulatanous Localization and Mapping (SLAM) is the use of vision as the only exteroceptive sensor [9, 12, 15, 42, 27], due to its low cost. We adopt the term vSLAM[17] for visual SLAM. Currently, vSLAM solutions focus on accurate

localization, mostly based on the estimation of geometric visual features of the environment. Thus, the resultant map is useful for the localization of the robot, but its use for other purposes is often neglected. This section concentrates on the geometric mapping while in the next section, we try to take a look at some opportunities how to apply visual means for higher-level understanding of the robot's environment.

In mainstream computer vision research, the problem of 3D reconstruction from a sequence of images is termed structure from motion. There is a great similarity between the structure from motion (SfM)[1] and vSLAM. The essential problem of simultaneously estimating the structure of the environment and the motion of the observer is identical. In computer vision community, SfM is nowadays considered mostly a solved problem, as commercial solutions for SfM-based camera motion estimation have become available from companies such as 2d3[2] The state-of-art SfM solutions are mostly based on using projective geometry as the geometrical model [16] and bundle adjustment techniques (basically Levenberg-Marquardt minimization) for finding the maximum likelihood solution for the nonlinear optimization problem [16, 46]. In addition to the 3D reconstruction for calibrated cameras, the internal camera calibration can be estimated along with the structure (self calibration), thus eliminating the need to explicitly calibrate cameras (for example, [33, 21]).

However, there are some differences between SfM and vSLAM which are often overlooked. The main emphasis in SfM is to reconstruct the structure of the environment as accurately as possible. This is often termed "global bundle adjustment", emphasizing the fact that the maximum likelihood solution over the whole image sequence is sought. Thus, the solutions are essentially batch algorithms, requiring the whole data for processing. Also, the heavy computational load of the global optimization makes it impossible to run in real-time. One reason for this is that the intended application areas, such as camera tracking for the movie industry, do not require on-line estimation. In vSLAM, on-line operation is a fundamental requirement, as the information is often used for robot navigation. In practice, the computational complexity increases with respect to the number of landmarks and time. If the robot needs to operate in a large (or changing) environment for a long time, the vSLAM algorithm would need to be constant time (or at least sub-linear). Currently there is no vSLAM approach which would allow this, although there have been propositions of constant time SLAM algorithms using other sensors (for example, [20]) although the estimates given by the methods are conservative rather than optimal.

Another characteristic typical for vSLAM but not for SfM is that in addition to the estimate, its uncertainty needs to be characterized. Moreover, the algorithms need to be statistically consistent, that is, not giving overly confident estimates of the uncertainty. This characteristic can be exploited to build safeguards and allow the information to be safely used for controlling the robot.

Loop closing refers to identifying that a robot has returned to a previously visited area after touring in another one. The observation of previously known landmarks

[1] Nowadays often termed "structure and motion."

[2] See `http://www.2d3.com`.

allows to improve the estimate over the whole tour. Loop closing using the traditional SLAM sensors, for example, laser scanners, is hard due to the difficulties in correctly matching previously detected landmarks and current observations. Visual features are very powerful in this respect, and the use of visually salient features for loop closing has been recently proposed [26, 34]. However, there are still many unexplored possibilities in using image retrieval and matching approaches for loop closing.

In some cases, the maps built are not intended (or needed) to be used for localization at later time instants. In this case, the resulting map is only stored for a short period and the landmarks are removed from the map after they are no longer visible. This approach is usually called visual odometry [28, 3]. While this approach essentially solves the map complexity problem, it suffers from drift as the earlier landmarks are forgotten and loop closing is not possible. As a compromise between the accuracy of global bundle adjustment and the restrictions in the available processing capacity, it has been proposed to only optimize the pose of a small number of previous time instants [25]. This has the effect that the accuracy is increased, but unfortunately the local approach still can not remove drift and benefit from loop closing in contrast to the more global approaches.

Ideally, the robotics community would like to get a statistically consistent vSLAM algorithm that would perform constant time on-line estimation with the optimality of the global bundle adjustment, allowing efficient loop closing. This book contains three chapters which demonstrate some of the ideas and recent work towards this ideal goal. Chapter 7 presents the Sliding Window Filter, an approach for incremental SLAM using a sliding time window of most recent sensor measurements, thus attaining constant time complexity. Chapter 8 examines the joint use of object/scene recognition for coarse topological localization and more accurate local metric localization using 1D trifocal tensor. Chapter 9 discusses different types of visual landmarks which could be used for vSLAM. Furthermore, loop closing and future directions such as multi-robot SLAM and the use of geographical information systems (GIS) in SLAM are considered.

1.4 From Maps to Understanding the Environment

Robots of the future should be able to easily navigate in dynamic and crowded environments, detect as well as avoid obstacles, have a dialog with a user and manipulate objects. It has been widely recognized that, for such a system, different processes have to work in synergy: high-level cognitive processes for abstract reasoning and planning, low-level sensory-motor processes for data extraction and action execution, and mid-level processes mediating these two levels.

A successful coordination between these levels requires a well defined representation that facilitates anchoring of different processes. One of the proposed modeling approaches has been the use of *cognitive maps* [5]. The cognitive map is the body of knowledge a human or a robot has about the environment. In [5], it is ar-

gued that topological, semantic and geometrical aspects are important for representation of spatial knowledge. This approach is closely related to Human-Augmented mapping (HAM) where a human and a robot interact so to establish a correspondence between the human spatial representation of the environment and robot's autonomously learned one [18].

In addition, both during the mapping phase and during robot task execution, object detection can be used to augment the map of the environment with objects' locations, [39]. There are several scenarios here: while the robot is building the map it will add information to the map about the location of objects. Later, the robot will be able to assist the user when s/he wants to know where a certain object X is. As object detection might be time consuming, another scenario is that the robot builds a map of the environment first and then when no tasks are scheduled for execution, it moves around in the environment and searches for objects.

The same skill can also be used when the user instructs the robot to go to a certain area to fetch a particular object. If the robot has seen the object before and has it already in the map, the searching process is simplified to re-detection. By augmenting the map with the location of objects we also foresee a way of achieving place recognition. This provides valuable information to the localization system as well as it greatly reduces the problem with symmetries in a simple geometric map. This would be an alternative approach to the visual place recognition presented in [35] and the laser based system in [23]. Furthermore, along the way by building up statistics about what type of objects typically can be found in, for example, a kitchen the robot might not only be able to recognize a certain kitchen but also potentially generalize to recognize a room it has never seen before as probably being a kitchen.

However, although there exists a large body of work on mobile robots, there are still no fully operational systems that can operate robustly and long-term in everyday environments. The current trend in development of service robots is reductionistic in the sense that the overall problem is commonly divided into manageable subproblems.

Chapter 10 explores the relations between tasks, objects and contexts for robots using maps, in the context of visually guided robots. Task descriptions are necessary as they explain the structure of actions and objects the tasks require. Most tasks can only occur in prototypical places, which have a suitable arrangement of objects. The local set of objects and their configuration then determines the context. The context then allows to endow physical locations with semantic meaning.

A solution to the visual SLAM problem does not necessarily allow robot navigation. Specifically, the knowledge of the 3-D location of a robot is insufficient for solving the navigation in a general setting. On the other hand, navigation can also be possible using solely image-based information. Chapter 11 investigates the use of visual information for robot navigation, presenting both image-based and map-based approaches.

1.5 Future Directions

The recent trends and future directions in the computer vision community are not always well-known for robotics researchers. This section tries to outline some of them[3].

Benchmarking is an important issue, which has only lately gained notable interest in robotics[4]. Since early 1990s there has been a growing interest in the computer vision community on discovering ways to compare the performance of different methods. For example, the EU funded PCCV (Performance Characterization in Computer Vision) project produced tutorials and case-studies for benchmarking vision algorithms [31]. Based on these studies, important benchmarking techniques include common benchmark data, contests, and specialized benchmarking workshops. While all of these have begun to appear also in robotics [32], it seems that the robotics community could benefit from the issues learned. Another recent trend in validating methods in computer vision is to use huge data sets gathered from internet. For example, image search engines give a possibility to obtain thousands of labeled images for testing object and scene recognition approaches.

Machine learning methods are becoming more and more widely used in computer vision as the processing power of modern computers has reached the point to make this possible. Most importantly, they are not only used for traditional "recognition" applications, but they can be also used for constructing efficient and effective processing algorithms, for example, for image feature extraction. Chapter 5 of this book is a fine example of this development, as is Rosten's work in learning high-speed corner detection [38].

Robotics has for the past few decades evolved from an industrial, well-controlled environment, to our homes, medical/operating rooms and resulted in sending robots to Mars. Still, most of the existing robot systems are designed for specific purposes and preprogrammed to expected requirements posed by the task and the environment. As mentioned, the challenge for the future is to go beyond the current engineering paradigm and develop artificial cognitive robotic systems that can robustly perceive, interact, reason and cooperate with humans and each other in open-ended environments.

Compared to classical robot systems, where the main requirement of the system is to execute a predefined task, the problem of task learning and planning stands as an open problem. Easy and user-friendly programming of new tasks in robots is one of the integral problems that will have to be considered in the future service robot systems.

The work presented in Chapter 12 takes a step towards solving this problem by providing a programming support for the implementation of complex, sensor-based robotic tasks in the presence of geometric uncertainty. The application of the proposed framework is studied in image-based visual servoing tasks.

---

[3] Some of the following ideas originate from a panel discussion at the ICRA 2007 workshop, which was a source of inspiration for this book.

[4] A benchmarking initiative has been recently started in EURON, the European robotics network.

Acknowledgements The support to Ville Kyrki by Academy of Finland is gratefully acknowledged.

References

1. Aggarwal, J., Park, S.: Human Motion: Modeling and Recognition of Actions and Interactions. In: Second International Symposium on 3D Data Processing, Visualization and Transmission. Thessaloniki, Greece (2004)
2. Aggarwal, J.K., Cai, Q.: Human motion analysis: A review. Computer Vision and Image Understanding: CVIU **73**(3), 428–440 (1999)
3. Agrawal, M., Konolige, K.: Rough terrain visual odometry. In: International Conference on Advanced Robotics. Jeju, Korea (2007)
4. Billard, A.: Imitation: A review. Handbook of brain thory and neural network, M. Arbib (ed.) pp. 566–569 (2002)
5. B.J.Kuipers: The cognitive map: Could it have been any other way? In H. L. Pick, Jr. and L. P. Acredolo (Eds.), Spatial Orientation: Theory, Research, and Application, New York: Plenum Press, pp. 345–359 (1983)
6. Calinon, S., Guenter, F., Billard, A.: Goal-Directed Imitation in a Humanoid Robot. In: International Conference on Robotics and Automation. Barcelona, Spain, April 18-22 (2005)
7. Castellanos, J.A., Tardós, J.D.: Mobile Robot Localization and Map Building: A Multisensor Fusion Approach. Kluwer Academic Publishers (1999)
8. Dariush, B.: Human Motion Analysis for Biomechanics and Biomedicine. Machine Vision and Applications **14**, 202–205 (2003)
9. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: Internatinal Conference on Computer Vision (2003)
10. Dissanayake, G., Newman, P., Clark, S., Durrant-Whyte, H., Corba, M.: A solution to the slam building problem. IEEE Transactions on Robotics **17**(3), 229–241 (2001)
11. Ekvall, S., Kragic, D.: Grasp recognition for programming by demonstration tasks. In: IEEE International Conference on Robotics and Automation, ICRA'05, pp. 748 – 753 (2005)
12. Folkesson, J., Jensfelt, P., Christensen, H.: Vision slam in the measurement subspace. In: International Conference on Robotics and Automation (2005)
13. Gavrila, D.M.: The visual analysis of human movement: A survey. Computer Vision and Image Understanding: CVIU **73**(1), 82–98 (1999)
14. Giese, M., Poggio, T.: Neural Mechanisms for the Recognition of Biological Movements. Nature Reviews **4**, 179–192 (2003)
15. Goncavles, L., di Bernardo, E., Benson, D., Svedman, M., Ostrovski, J., Karlsson, N., Pirjanian, P.: A visual front-end for simultaneous localization and mapping. In: International Conference on Robotics and Automation, pp. 44–49 (2005)
16. Hartley, R., Zisserman, A.: Multiple View Geometry. Cambridge University Press (2003)
17. Karlsson, N., di Bernardo, E., Ostrowski, J., Goncalves, L., Pirjanian, P., Munich, M.: The vSLAM algorithm for robust localization and mapping. In: International Conference on Robotics and Automation, pp. 24–29. Barcelona, Spain, April 18-22 (2005)
18. Kruijff, G.J.M., Zender, H., Jensfelt, P., Christensen, H.I.: Clarification dialogues in human-augmented mapping. In: Proc. of the 1st Annual Conference on Human-Robot Interaction, HRI'06. Salt Lake City, UT (2006)
19. Kuniyoshi, Y., Inaba, M., Inoue, H.: Learning by watching, extracting reusable task knowledge from visual observation of human performance. In: IEEE Transactions on Robotics and Automation, vol. 10(6), pp. 799–822 (1994)
20. Leonard, J., Newman, P.: Consistent, convergent, and constant-time SLAM. In: International Joint Conference on Artificial Intelligence, pp. 1143–1150. Acapulco, Mexico (2003)

21. Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(3), 418–433 (2005)
22. Lopes, M.C., Victor, J.S.: Visual transformations in gesture imitation: What you see is what you do. In: International Conference on Robotics and Automation, pp. 2375– 2381 (2003)
23. Martínez Mozos, O., Stachniss, C., Burgard, W.: Supervised learning of places from range data using adaboost. In: Proc. of the IEEE International Conference on Robotics and Automation, ICRA'05, pp. 1742–1747. Barcelona, Spain (2005)
24. Moeslund, T., Hilton, A., Krueger, V.: A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding **104**(2-3), 90–127 (2006)
25. Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., Sayd, P.: Real-time localization and 3D reconstruction. In: Computer Vision and Pattern Recognition. New York City, New York, USA, June 17-22 (2006)
26. Newman, P., Ho, K.: Slam- loop closing with visually salient features. In: International Conference on Robotics and Automation. Barcelona, Spain, April 18-22 (2005)
27. Newman, P., Ho, K.: SLAM-loop closing with visually salient features. In: International Conference on Robotics and Automation, pp. 644–651 (2005)
28. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry. In: Computer Vision and Pattern Recognition, pp. 652–659. Washington DC, USA, June (2004)
29. Ogawara, K., Iba, S., Kimura, H., Ikeuchi, K.: Recognition of human task by attention point analysis. In: International Conference on Intelligent Robots and Systems, pp. 2121–2126 (2000)
30. Ogawara, K., Iba, S., Kimura, H., Ikeuchi, K.: Acquiring hand-action models by attention point analysis. In: International Conference on Robotics and Automation, pp. 465–470 (2001)
31. PCCV. Performance Characterization in Computer Vision website. Http://peipa.essex.ac.uk/benchmark/index.html
32. del Pobil, A.P. (ed.): Benchmarks in Robotics Research. IROS 2006 workshop (2006)
33. Pollefeys, M., Koch, R., Van Gool, L.: Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters. International Journal of Computer Vision **32**(1), 7–25 (1999)
34. Posner, I., Schroeter, D., Newman, P.: Using scene similarity for place labeling. In: International Symposium on Experimental Robotics. Rio de Janeiro, Brazil (2006)
35. Pronobis, A., Caputo, B., Jensfelt, P., Christensen, H.: A discriminative approach to robust visual place recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'06 (2006)
36. Rizzolatti, G., Fogassi, L., Gallese, V.: Parietal Cortex: from Sight to Action. Current Opinion in Neurobiology **7**, 562–567 (1997)
37. Rizzolatti, G., Fogassi, L., Gallese, V.: Neurophysiological Mechanisms Underlying the Understanding and Imitation of Action. Nature Reviews **2**, 661–670 (2001)
38. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: European Conference on Computer Vision, vol. 1, pp. 430–443 (2006)
39. S. Ekvall, P.J., Kragic, D.: Object detection and mapping for service robot tasks. Robotics **25(2)**, 175–188, (2007)
40. Schaal, S.: Is Imitation Learning the Route to Humanoid Robots? Trends in Cognitive Sciences **3**(6), 233–242 (1999)
41. Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. Philosophical transaction of the Royal Society of London, series B **358**(1431), 537–547 (2003)
42. Sim, R., Elinas, P., Griffin, M., Little, J.J.: Vision-based slam using the rao-blackwellised particle filter. In: IJCAI Workshop on Reasoning with Uncertainty in Robotics (2005)
43. Tardós, J., Neira, J., Newman, P., Leonard, J.: Robust mapping and localization in indoor environments using sonar data. International Journal of Robotics Research **4** (2002)
44. Thrun, S., Fox., D., Burgard, W.: A probalistic approach to concurrent mapping and localization for mobile robots. Autonomous Robots **5**, 253–271 (1998)

45. Thrun, S., Liu, Y., D.Koller, Ng, A., Ghahramani, Z., Durrant-White, H.: SLAM with sparse extended information filters. International Journal of Robotics Research **23**(8), 690–717 (2004)
46. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment – A modern synthesis. In: W. Triggs, A. Zisserman, R. Szeliski (eds.) Vision Algorithms: Theory and Practice, LNCS, pp. 298–375. Springer Verlag (2000)
47. Veeraranghavan, A., Chellappa, R., Roy-Chowdhury, A.: The Function Space of an Activity. In: Computer Vision and Pattern Recognition. New York City, New York, USA, June 17-22 (2006)
48. Wu, Y., Huang, T.S.: Vision-based gesture recognition: A review. Lecture Notes in Computer Science **1739**, 103+ (1999)

Chapter 2
Extracting Planar Kinematic Models Using Interactive Perception

Dov Katz and Oliver Brock

2.1 Introduction

Roboticists are working towards the deployment of autonomous mobile manipulators in unstructured and dynamic environments. Adequate autonomy and competency in unstructured environments would open up a variety of important applications for robotics, ranging from planetary exploration to elder care and from the disposal of improvised explosive devices to flexible manufacturing and construction in collaboration with human experts. Ongoing research efforts seek to enable the use of autonomous robots for these applications through the development of adequate hardware platforms [10, 26, 31], robust and task-oriented control strategies [19], and new learning frameworks [2, 5, 6, 27].

For unstructured and dynamic environments, it is not possible to provide the robot with a detailed a priori model of the world. Consequently, an autonomous robot has to continuously acquire perceptual information to successfully execute mobility and manipulation tasks [12, 17, 25, 29]. This extraction can be performed most effectively, if it occurs in the context of a specific task.

During task execution, the value of perceptual information can be maximized by interpreting sensor streams in a manner that is tailored to the task. Focusing on task-specific aspects during the interpretation of the sensor stream will reveal the most task-relevant information, while reducing the computational cost of perception. Both of these advantages can improve the robustness of task execution, particularly in the presence of significant uncertainty. In spite of the advantages

Dov Katz
University of Massachusetts Amherst, Computer Science Department, Amherst, MA, USA, e-mail: dubik@cs.umass.edu

Oliver Brock
University of Massachusetts Amherst, Computer Science Department, Amherst, MA, USA, e-mail: oli@cs.umass.edu

of integrating perception and manipulation, research towards integrated perceptual paradigms for autonomous manipulation is still in its early stages.

In this work, we argue that interactive perception, a framework that exploits forceful interactions with the environment to uncover perceptual information required for the robust execution of specific tasks, can serve as an adequate perceptual framework for autonomous manipulation. We will show that the combination of forceful interactions with visual perception reveals perceptual information unobtainable by forceful interactions or visual perception alone. Crossing the boundaries between manipulation and perception leads to novel perceptual capabilities, even when the manipulation and perception capabilities are very basic.

Fig. 2.1 Objects that possess inherent degrees of freedom; these degrees of freedom cannot be extracted from visual information alone, they have to be discovered through physical interaction

To illustrate the promise of interactive perception as a perceptual paradigm for autonomous robots operating in unstructured environments, we have developed a perceptual skill to extract kinematic models from unknown objects. Many objects in everyday environment possess inherent degrees of freedom that have to be actuated to perform their function. Such objects include doors, door handles, drawers, and a large number of tools such as scissors and pliers (Figure 2.1).

In our experiments, UMan (Figure 2.2), our experimental platform for autonomous mobile manipulation, employs interactive perception to extract kinematic models from tools such as pliers or shears. These models are then employed to compute an action that transforms the kinematic state of the tool into a desired goal state, mimicking tool use. Note that kinematic models of objects cannot be extracted using visual information alone. They are also very hard to obtain from tactile feedback. We believe that the relative ease with which we are able to address this task makes a convincing case for the use of interactive perception as a perceptual paradigm for autonomous robotics.

2.2 Related Work

In the absence of a model, autonomous manipulation in unstructured environments depends on sensor streams to assess the state of the world. The sensor streams should be interpreted and the resulting information can then be used to guide manipulation. In this section, we will discuss perceptual techniques that were developed independently of specific manipulation objectives as well as approaches that closely integrate perception and manipulation.

Computer vision researchers extensively explored object segmentation and labeling from static images [14]. These problems, which seem to be solved effortlessly by humans, were found to be quite challenging.

Active vision [1, 3, 4] represents a paradigm shift relative to computer vision based on static images. Now, the agent is no longer a passive observer but instead can control the motion of the sensor to actively extract relevant information. Active vision simplified the extraction of structure from visual input [23, 30] and facilitated depth estimation based on information about the camera's motion [20].

Visual servoing provides closed-loop position control for a robotic mechanism [15, 22]. It is an example of how position control, one of the fundamental primitives of manipulation, can be greatly improved through integration with vision.

Fig. 2.2 UMan (UMass Mobile Manipulator) consists of a holonomic mobile base with three degrees of freedom, a seven-degrees-of-freedom Barrett Technologies manipulator arm, and a four-degrees-of-freedom Barrett hand.

Although active vision greatly improves data acquisition, in some cases this process cannot generate the data required to support a specific task. For example, object segmentation and predicting kinematic and dynamic properties of rigid or articulated bodies remain great challenges even when the camera's position can be controlled. Prior work has shown that physical interaction with the world can remedy many of these difficulties.

Object segmentation can be solved by actively poking objects using a robotic manipulator [13, 24]. The generated optical flow allows the identification of moving objects and separates them from their background.

Tool use can be performed by treating tools in the context of their task. Instead of recovering the entire state of the world from the sensor stream, Edsinger and Kemp [11, 12] focus on information that is task-relevant. This simplifies the perceptual process, and allows successful operation in environments that were not adapted to the robot. The work of Fitzpatrick, Metta, Edsinger and Kemp can be characterized as interactive perception.

Predicting the movement of objects in the plane can also be simplified by interaction. Christiansen, Mason, and Mitchell addressed this problem by placing objects on a tray which could be tilted by a robotic manipulator [7]. The robot actively tilted the tray to increase its knowledge about the objects' motion. This knowledge then facilitated successful task execution which required object displacements. Stoytchev used a predefined set of interactions with rigid objects (tools) to explore their affordances [28]. He extracted the results of tool use by the robot by visually observing the motion of rigid bodies in the workspace of the robot. This knowledge was then applied during task execution by selecting the most appropriate tool.

The last three examples demonstrate the positive effects that deliberate action has on the successful completion of tasks and on the difficulty of the perception problem. They represent a natural development from the active vision paradigm towards the interactive perception paradigm, in which robots can actively change the world to increase sensor range. The following section presents this paradigm, and explains how it can dramatically improve the capabilities of robots in unstructured and dynamic environments.

2.3 Interactive Perception

A robot can enhance its perceptual capabilities by including physical interactions with the environment in its perceptual repertoire. Such interactions can remove obstructions, provide an easy and controlled way of exposing multiple views of an object, or can alleviate the negative effects of lighting conditions by moving objects into the field of view. Other perceptual tasks are difficult or even impossible to accomplish without interacting with the environment. For example, reading the text in a closed book, checking whether a door is locked, and finding out the purpose of a switch mounted on the wall. Physical interactions augment the sensor stream with force feedback and allow to evoke and observe behaviors in the world that can reveal physical properties of objects. Such information would otherwise remain inaccessible for non-interactive sensors. Physical interactions thus can make traditional perceptual tasks easier. Moreover, they make a new class of perceptual information accessible to a robotic agent.

The promise of interactive perception [17] is supported by examples from the development of physical and mental skills in humans. During the acquisition of physical skills by infants, for example, physical interactions with the environment are necessary to bootstrap the cognitive process of learning the connection between

action and effect, the kinematics of one's own body, and the properties and functions of objects in the environment.

Interacting with the environment as part of the perceptual process poses a challenge: selecting the most adequate interaction for a perceptual task. The need to choose the right exploratory skill while balancing between exploration and exploitation is not new. Active learning [8], a branch of machine learning, addresses the very same problem and has been shown to be highly effective. While in this work we will focus on a single perceptual primitive to demonstrate the effectiveness of interactive perception, our future work will integrate this and other primitives into a perceptual framework that can actively select when and which interactive perceptual primitive to invoke.

2.4 Obtaining Kinematic Models Through Forceful Interactions

In this section, we will present one instantiation of the interactive perception framework. We will demonstrate how a robotic manipulator can extract the kinematic properties of a tool lying on a table. No *a priori* knowledge about the tool is assumed. The robot can subsequently construct a model of the tool which will allow it to determine the appropriate interaction for using the tool. In this early work on interactive perception, we will restrict ourselves to revolute joints.

2.4.1 Algorithm

The key insight behind our algorithm is that the relative distance between two points on a rigid body does not change as the body is pushed. However, the distance between points on different rigid bodies connected by a revolute joint does change as the bodies rotate relative to each other.

First, we describe our algorithm for objects composed of two links connected by a single revolute joint. The robot interacts with a tool on the table by sweeping its end-effector across the surface. Tracking a set of features of the object throughout the interaction allows us to measure the distance between these features as the object is being moved. The features can be separated into three groups: features on the first link, features on the second link, and features on the joint connecting them. Features in the same group must maintain constant distance to each other, irrespective of the planar motion the object performs. However, the distance between features in the first group and features in the second group will change significantly as the object is being moved. The joint features are simply features that belong to both the first and the second group. This algorithm works also in the general case of multiple revolute joints. To identify the groups, a robotic manipulator interacts with the object to generate motions that will allow distinguishing between the different rigid bodies.

In order to determine the spatial extent of the links of the object, we construct a convex hull around the features in each group. Tracking enough features increases the match between the convex hull and the actual shape of the link. The length of each link is taken to be the distance between the furthest point in each group and the joint. We use this knowledge to create a kinematic model for planar kinematic chains. This model is later used to predict the actions required to manipulate the object in a meaningful fashion.

The following subsections describe in detail the implementation of the kinematic model building algorithm. It is worth noting that the specific way in which we choose features, track them or analyze their relative motion does not affect the algorithm.

2.4.2 Tracking Objects

Since our primary goal is to show the promise of interactive perception as a perceptual paradigm, we place objects on a plain white background, facilitating feature tracking. The white background assumption can be removed using ideas from active segmentation (similarly to [13] and [24]). This includes an initial random phase were the manipulator sweeps the environment in an attempt to segment objects. The interaction may provide interesting objects, for which we might want to construct a kinematic model.

We use the open source computer vision library OpenCV [16] to capture, record, and process images. OpenCV implements feature selection by finding corners with big eigenvalues, and feature tracking based on the optical flow algorithm of Lucas and Kanade [21]. We store the position of the automatically generated set of features in every frame during the interactive session.

The tracked features are selected before the interactive session begins. Some features may be obstructed by the manipulator's motion during the interaction. Those features will be very noisy, and therefore easily discarded. Moreover, no feature will be associated with the manipulator itself because all features are selected prior to the appearance of the arm in the scene.

2.4.3 Constructing a Graphical Representation

Every planar kinematic chain is composed of links and joints. Therefore, the first task we perform is joint and link identification. We build a graph based on the maximal change in distance between two features observed throughout the entire interaction. Every node $v \in V$ in the graph represents a tracked feature in the image. An edge $e \in E$ connects nodes (v_i, v_j) if and only if the distance between v_i and v_j remains constant (in practice, we allow the distance to vary up to a threshold). The resulting graph will be analyzed by the algorithm described in the following section.

2.4.4 Graph Analysis

Figure 2.3 shows a schematic depiction of a graph constructed for an object with two joints. We can detect in this graph three groups of nodes; each group is very highly interconnected. The groups represent links, and high interconnectivity is the result of no motion between features on the same rigid body (link). The nodes that connect two groups represent joints, and therefore are highly connected to two groups.

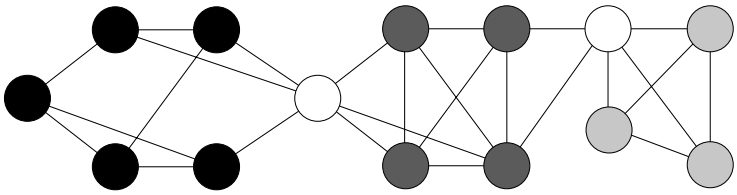


Fig. 2.3 Generated graph for an object with two degrees of freedom. Highly connected components (colored in shades of grey) represent the links. Nodes that connect between components represent joints (colored in white).

We use the min-cut algorithm [9] to identify different groups in a graph. Min-cut will separate a graph into two sub-graphs by removing as few edges as possible. In the simple case of one joint and two links, min-cut will remove the edges that connect between the two links, resulting in two highly connected sub-graphs, each representing a different link. In the general case, a graph may contain multiple highly connected components (each component represents a different link). Identifying components in this case is done simply by recursively breaking the graph into sub-graphs. The process stops when the input graph is highly connected, therefore representing one rigid component. Finally, nodes that belong to two different highly connected components are nodes that represent a joint connecting two links.

It should be noted that the above procedure automatically rejects errors in tracking features. If a feature "jumps" during tracking, the graph construction described above will lead to a disconnected component consisting of a single vertex. Disconnected nodes, and more generally small disconnected components, can be discarded during the graph analysis. The proposed procedure thus is inherently robust to errors in feature tracking.

2.4.5 Building a Kinematic Model

The graph provides us with information about the basic kinematic structure of the object. We can construct an approximate contour of the object by computing the convex hull of all features in a component of the graph. The extent of the visual hull gives us an approximate geometric description of each link. By combining all

the information, we construct a kinematic model of the kinematic chain. This model enables the robot to reason about the effects that its interactions with the object will have. This is a prerequisite for purposeful tool use.

2.5 Experimental Results

We validate the method described above in experiments on our robotic platform for autonomous mobile manipulation, called *UMan* (see Figure 2.2, [18]). *UMan* consists of a holonomic mobile base with three degrees of freedom, a seven-degrees-of-freedom Barrett Technologies manipulator arm, and a four-degrees-of-freedom Barrett hand. The vision system is an overhead web camera, mounted above a desk. The camera's resolution is 640X480. The platform provides adequate end-effector capabilities for a wide range of dexterous manipulation tasks.

UMan is tasked to extract a kinematic model of four different tools, shown in Figure 2.5. To demonstrate that *UMan* can use the kinematic model for purposeful interactions with those tools, it is required to push the tool until the two rigid links form a right angle. *UMan* first uses its end-effector to sweep the table in front of it, while observing the scene. Features are tracked in the resulting video sequence and the algorithm described above is used to extract a kinematic model of the tool. Using this model, *UMan* determines the appropriate pushing motion to achieve the desired angle between the two links and performs this motion. An example of such an experiment and the corresponding visual observation is shown in Figure 2.4.

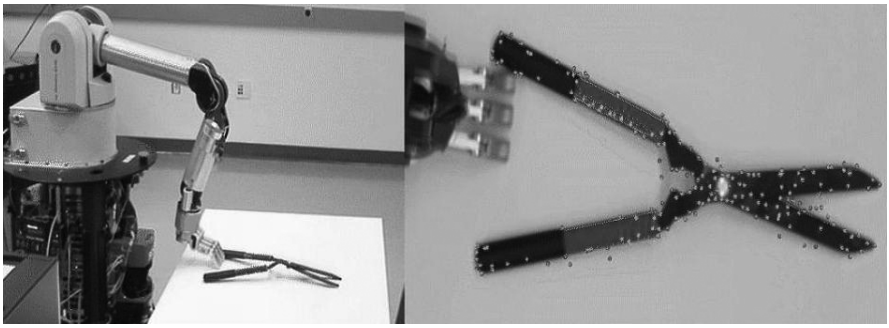


Fig. 2.4 UMan interacts with a tool by reaching its arm towards the tool. The right image shows the tool as seen by the robot, with dots marking the tracked features. The left image shows the experimental setting

Four tools were used in the experimental phase: scissors, shears, pliers, and a stapler. All four tools have a single revolute joint, with the exception of the pliers which also have a prismatic joint that was ignored. The tools are off-the-shelf products and have not been modified for our experiments. They vary in scale, shape, and

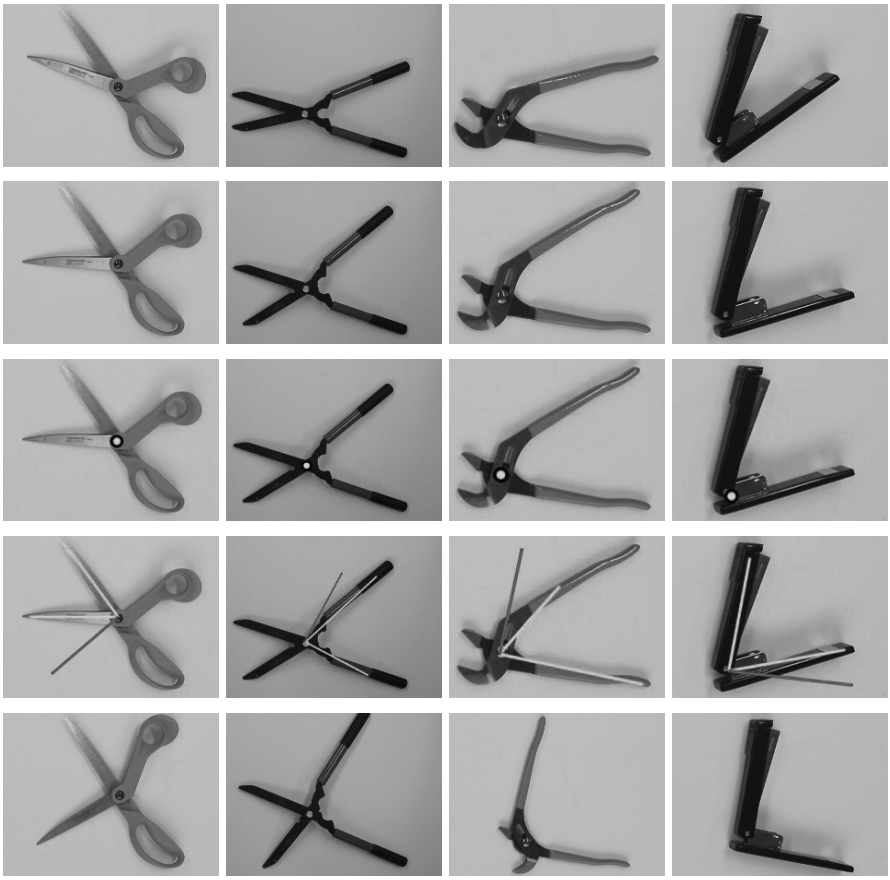


Fig. 2.5 Experimental results showing the use of interactive perception in extracting the kinematic properties of different objects. The first row of images shows the four objects (scissors, shears, pliers, and stapler) in their initial pose. The second row shows the final pose of the four objects after the robot has interacted with them. The third row shows the revolute joint that was detected using the described methods; the revolute joint is marked with a white disc. The fourth row of images shows the links of the obtained kinematic model and the manipulation plan to form a right angle between the two links of the tools. Putting the two links into a 90° angle here serves as an example of tool use. The links of the tools are shown as white lines, and the orientation of one of the links to achieve the goal configuration of the tool is marked by a black line. The last row of images shows the results of executing the manipulation plan as presented in the previous row: the two links of the tools have been arranged in a 90° angle.

color. Despite the differences in appearance, all four tool belong to the family of two-link kinematic chains with a single revolute joint.

Figure 2.5 shows in each row snapshots of the experiment with one of the four tools. Each row shows a particular phase of the experiment. First, the tools are in their initial pose (before the interaction begins). Next, we see the tools in their final pose (after the interaction). The third row shows the location of the joints, as detected by interacting with the tools. In the fourth row, two white lines mark the position of the parts of the links that will be used for the purposeful interaction. A third black line indicates where one of the links needs to be moved to in order to create a right angle between the links. Finally, the last row shows the tools after the execution of the plan from the previous row—each tool was manipulated to form an angle of 90°.

The experimental results show that the detection of the revolute joint is very accurate. Moreover, the length and position of the links are also discovered correctly. The algorithm uses the information collected in the interactive process to create kinematic models for the tools. The high accuracy and usefulness of these models is demonstrated by the successful manipulation of the tools to form an angle of 90° between the links.

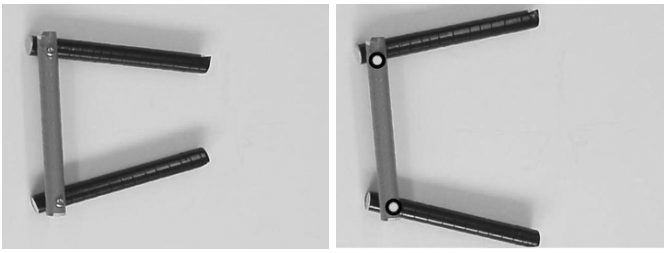


Fig. 2.6 Experimental results showing the use of interactive perception in extracting the kinematic properties of objects with two degrees of freedom.

Figure 2.6 shows an additional experiment with an object that possesses multiple degrees of freedom. Without any modification, the algorithm described above successfully identifies the two degrees of freedom. However, in our experiment, the first interaction only revealed a single degree of freedom. The second degree of freedom had to be extracted using another interaction. This illustrates the need to embed the perceptual primitive described here into a higher-level perceptual process.

In all of our experiments, the proposed algorithm was able to extract the kinematic axes of the tools with great precision, despite the cheap off-the-shelf web camera that was used. Additionally, only a small displacement of the object was required. The algorithm does not have any parameters that need to be tuned. The performance of the algorithm was extremely robust, all of our experiments "just worked." No changes were necessary to the algorithm to deal with the five objects, even though their size, visual appearance, and kinematic properties varied. Further-

more, our experiments have shown that the algorithm is insensitive to the distance of the camera to the object. The algorithm also performs without errors for a broad range of viewing angles. Even though we initially assumed that the viewing direction would be orthogonal to the surface of the table, the algorithm tolerates deviations of up to 30°. We have not explicitly tested this parameter but suspect that even higher deviations will continue to give good results.

The experiments discussed here demonstrate that the combination of two very fundamental capabilities, namely feature tracking and object pushing, yield a highly robust and accurate perceptual primitive. This primitive is able to extract perceptual information that neither of its two components could extract by themselves. The experiments thus demonstrate that interactive perception can increase the perceptual capabilities of a robot while at the same time improving the robustness of the perceptual process. As stated in the introduction, we believe that this is the consequence of combining manipulation and perception to develop a task-related perceptual process. We are convinced that interactive perception represents an important step towards the robust execution of autonomous manipulation tasks in unstructured environments.

2.6 Conclusion

This document explores interactive perception as an adequate perceptual paradigm for autonomous robots. Interactive perception tightly couples interaction and perception to enable the robust and efficient extraction of task-relevant information from sensor streams. The inclusion of interaction into the repertoire of perceptual primitives not only facilitates many conventional perception tasks, but also allows an autonomous agent to uncover information about the environment that would otherwise remain hidden. Such information includes, for example, the kinematic and dynamic properties of objects in the environment, or views of the environment that can only be obtained after visual obstructions have been removed.

We employed the principle of interactive perception to show that a robot can easily extract the kinematic properties of novel objects from a visual sensor stream if it is able to physically interact with these objects. We have further demonstrated how the extracted knowledge about the object can be used to determine appropriate use of the object. Our experimental results on a real-world platform for mobile manipulation show that interactive perception can result in highly robust and effective perceptual algorithms.

There are many possible directions for future research and extensions of the presented work. We plan to improve our feature tracking and contour detection algorithms by using active segmentation techniques [13, 24]. We also will generalize the types of kinematic properties that can be extracted. We would like to include other types of joints, such as prismatic or spherical joints, and joints that have joint axes with arbitrary orientations. Finally, we intend to extend our framework to support additional sensor modalities, such as force sensors and laser scanners.

Acknowledgements This work is supported by the National Science Foundation (NSF) under grants CNS-0454074, IIS-0545934, CNS-0552319 CNS-0647132, and by QNX Software Systems Ltd. in the form of a software grant. We are grateful for this support.

References

1. Aloimonos J. and Weiss I. and Bandyopadhyay A.: Active Vision. International Journal of Computer Vision **1**, 333–356 (1988)
2. Azad, P., Asfour, T., Dillmann, R.: Toward an Unified Representation for Imitation of Human Motion on Humanoids. In: International Conference on Robotics and Automation. Rome, Italy (2007)
3. Bajcsy, R.: Active Perception. IEEE Proceedings **76**(8), 996–1006 (1988)
4. Blake, A., Yuille, A.: Active Vision. MIT Press (1992)
5. Brock, O., Fagg, A., Grupen, R., Platt, R., Rosenstein, M., Sweeney, J.: A Framework for Learning and Control in Intelligent Humanoid Robots. International Journal of Humanoid Robotics **2**(3), 301–336 (2005)
6. Brooks, R., Aryananda, L., Edsinger, A., Fitzpatrick, P., Kemp, C., O'Reilly, U.M., Torres-Jara, E., Varshavskaya, P., Weber, J.: Sensing and manipulating built-for-human environments. International Journal of Humanoid Robotics **1**(1), 1–28 (2004)
7. Christiansen, A.D., Mason, M., Mitchell, T.: Learning reliable manipulation strategies without initial physical models. In: International Conference on Robotics and Automation, vol. 2, pp. 1224–1230. Cincinnati, Ohio, USA (1990)
8. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active Learning with Statistical Methods. Journal of AI Research **4**, 129–145 (1996)
9. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. MIT Press and McGraw-Hill (2001)
10. Deegan, P., Thibodeau, B., Grupen, R.: Designing a Self-Stabilizing Robot For Dynamic Mobile Manipulation. In: Robotics: Science and Systems - Workshop on Manipulation for Human Environments. Philadelphia, Pennsylvania, USA (2006)
11. Edsinger, A.: Robot Manipulation in Human Environments. Ph.D. thesis, Massachusetts Institute of Technology (2007)
12. Edsinger, A., Kemp, C.C.: Manipulation in Human Environments. In: IEEE/RSJ International Conference on Humanoid Robotics. Beijing, China (2006)
13. Fitzpatrick, P., Metta, G.: Grounding vision through experimental manipulation. Philosophical Transactions of the Royal Society: Mathematical, Physical, and Engineering Sciences **361**(1811), 2165–2185 (2003)
14. Forsyth, D.A., Ponce, J.: Computer Vision: A Modern Approach. Prentice Hall Professional Technical Reference (2002)
15. Hutchinson, S.A., Hager, G.D., Corke, P.I.: A tutorial on visual servo control. IEEE Transactions on Robotics and Automation **12**(5), 651–670 (1996)
16. Intel: http://www.intel.com/technology/computing/opencv/
17. Katz, D., Brock, O.: Interactive Perception: Closing the Gap Between Action and Perception. In: International Conference on Robotics and Automation Workshop: From features to actions - Unifying perspectives in computational and robot vision. Rome, Italy (2007)
18. Katz, D., Horrell, E., Yang, Y., Burns, B., Buckley, T., Grishkan, A., Zhylkovskyy, V., Brock, O., Learned-Miller, E.: The UMass Mobile Manipulator UMan: An Experimental Platform for Autonomous Mobile Manipulation. In: Workshop on Manipulation in Human Environments at Robotics: Science and Systems (2006)
19. Khatib, O., Yokoi, K., Brock, O., Chang, K.S., Casal, A.: Robots in Human Environments: Basic Autonomous Capabilities. International Journal of Robotics Research **18**(7), 684–696 (1999)

20. Koederink, J.J., Van Doorn, A.J.: Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. Optica Acta. **22**, 773–791 (1975)
21. Lucas, B.D., Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision (DARPA). In: Proceedings of the 1981 DARPA Image Understanding Workshop, pp. 121–130 (1981)
22. Martin Jägersand: On-line Estimation of Visual-Motor Models for Robot Control and Visual Simulation. Ph.D. thesis, University of Rochester (1997)
23. Maybank, S.: The Angular Velocity Associated with the Optical Flowfield Arising from Motion through a Rigid Environment. In: Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, vol. 401, pp. 317–326 (1985)
24. Metta, G., Fitzpatrick, P.: Early integration of vision and manipulation. Adaptive Behavior **11**(2), 109–128 (2003)
25. Neo, E.S., Sakaguchi, T., Yokoi, K., Kawai, Y., Maruyama, K.: Operating Humanoid Robots in Human Environments. In: Workshop on Manipulation for Human Environments, Robotics: Science and Systems (2006)
26. Nishiwaki, K., Kuffner, J., Kagami, S., Inaba, M., Inoue, H.: The experimental humanoid robot H7: a research platform for autonomous behaviour. Philosophical Transactions of the Royal Society **365**, 79–108 (2007)
27. Saxena, A., Driemeyer, J., Kearns, J., Ng, A.Y.: Robotic Grasping of Novel Objects. In: Neural Information Processing Systems (2006)
28. Stoytchev, A.: Behavior-Grounded Representation of Tool Affordances. In: International Conference on Robotics and Automation, pp. 3071–3076. Barcelona, Spain (2005)
29. Sutton, M., Stark, L., Bowyer, K.: Function from visual analysis and physical interaction: a methodology for recognition of generic classes of objects. Image and Vision Computing **16**, 746–763 (1998)
30. Waxman, A.M., Ullman, S.: Surface Structure and Three-Dimensional Motion from Image Flow Kinematics. The International Journal of Robotics Research **4**, 72–94 (1985)
31. Wimboeck, T., Ott, C., Hirzinger, G.: Impedance Behaviors for Two-Handed Manipulation: Design and Experiments. In: International Conference on Robotics and Automation. Rome, Italy (2007)

Chapter 3
People Detection Using Multiple Sensors on a Mobile Robot

Zoran Zivkovic and Ben Kröse

3.1 Introduction

Robots are moving out of laboratories into public places where the human beings have to be taken into account. Such robots should be able to interact with humans and show aspects of human style social intelligence. This also implies that in addition to the perception required for the conventional functions (localization, navigation, etc.), a "socially interactive" robot needs strong human oriented perceptual capabilities [1, 16]. For a start the robot should be able to accurately and robustly detect and localize the persons around it.

Person detection from images is a widely studied problem in the computer vision research area. Two types of applications can be distinguished. The first type is surveillance where usually much knowledge is available about the environment, camera position and camera parameters. This knowledge provides additional cues for person detection. For example in most man made environments people walk over a floor plane which leads to a limited set of possible person position in an image. Furthermore, the camera is often static and this can help to distinguish persons from the static background. The second type of application considers a more general and difficult problem where not much a priori knowledge is available about the images, e.g. images or videos from the internet. A common approach in such situations is to use the whole image to infer more about the environment and the camera which can then help to detect people, e.g. [10].

Typical robotics applications differ from the typical computer vision applications in a number of aspects. First, robotics systems are usually equipped with multiple sensors. For example 2D laser range scanner is often used to detect persons legs. Properly combining the information from different sensors can improve the detection results. Second, similar to the surveillance applications, camera and other sen-

Zoran Zivkovic and Ben Kröse
University of Amsterdam, Intelligent Systems Laboratory, Kruislaan 403, 1098SJ Amsterdam, The Netherlands e-mail: \{zivkovic,krose\}@science.uva.nl

sors positions and parameters are usually known. However, the sensors are not static since they are mounted on a moving platform, a mobile robot or a vehicle. Finally, the fact that robots move can be an advantage. Actively moving the sensors can improve the detection results, for example moving closer to the object or viewing it from another view point.

This chapter considers the important problem of dealing with multiple sensors. An approach for combining information from multiple sensors for people detection on a mobile robot is described. A person will be represented by a constellation of body parts. Person body parts are detected and the parts are constrained to be at certain positions with respect to each other. Similar part based representations are widely used in the computer vision area for describing objects in images. A probabilistic model is presented here to combine part detections from multiple sensors typical for mobile robots. For detecting the body parts specific detectors can be constructed in many ways. In this chapter the Ada-Boost [7] is used as a general "out of box" approach for building the part detectors.

The chapter starts with the related work which is presented in Section 3.2. Next, in Section 3.3 people detection using 2D laser range scanner is considered. Persons legs can be detected in the scans. A probabilistic part-based representation is presented that takes into account the spatial arrangement of the detected legs. The method is inspired by the latest results on the "part-based representations" from the computer vision area and the work of Weber, Perona and colleagues [21, 5]. The approach takes into account that the leg detector might produce false detections or fail to detect legs, for example because of partial occlusion. Section 3.4 describes a straightforward way to extend the presented probabilistic model to properly combine body parts detected using other sensors that might be present on the robot, a pan-tilt camera and an omnidirectional camera in our case, see Figure 3.1. Evaluation of the proposed model and some practical issues are discussed in Section 3.5. Finally, the conclusions are given in Section 3.6.

3.2 Related Work

A 2D laser range scanner is often used in robotics for detecting and tracking people [13, 11]. People are detected by finding their legs in the laser scans. Disadvantages of using the laser scans for people detection are: the persons can be detected only at limited distances from the robot, low detection rate in highly cluttered environments and that the methods fail when the person legs are occluded. Other sensors were also used like thermal vision [17], stereo vision [9] and regular cameras [23].

Person detection from images is a widely studied problem in the computer vision area. Many of the presented algorithms aim at the surveillance applications [6] and are not applicable to mobile platforms since they assume static camera. There is also a large number of papers considering the people detection without the static camera assumption, e.g. [8, 14, 22, 15].

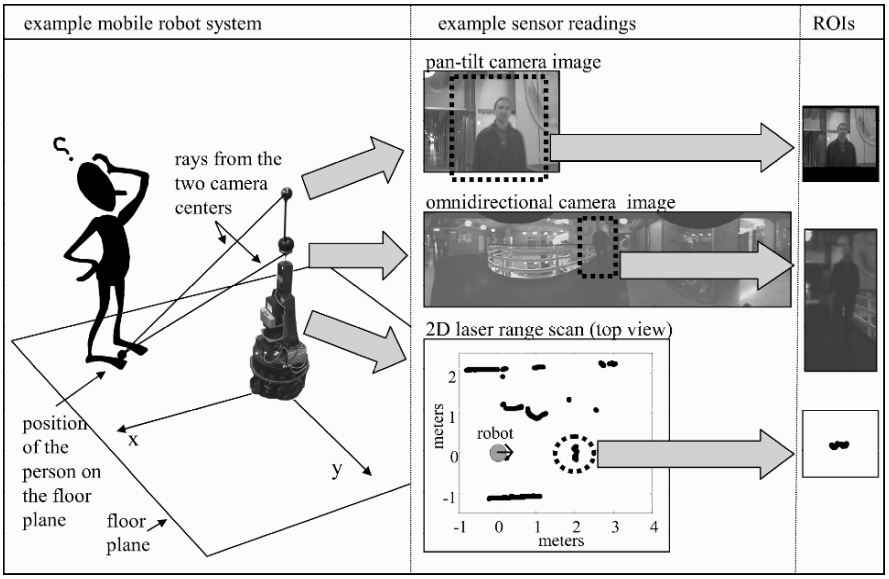


Fig. 3.1 Example moving robot platform equipped with three sensors: a 2D laser range scanner, a pan-tilt camera and an omnidirectional camera. The sensors are calibrated and their pose with respect to the floor plane is known. Given the typical size of a person we can define a region of interest (ROI) in each sensor corresponding a floor plane position as shown.

The people detection can be seen as a part of the more general problem of object detection. Many approaches were considered in the computer vision area. Recently it was shown that fast and reliable detection can be archived using "out of box" technique Ada-Boost to build classifiers [7]. For example Haar-like features with Ada-Boost were successively used for face detection by Viola and Jones [19]. Similar techniques were used for people detection [20, 18].

Another approach for object detection in images is the so called "part-based representation". Various part-based representations, e.g. [2, 5, 3, 4], are demonstrated to lead to high recognition rates. An important advantage of the part-based approach is it relies on object parts and therefore it is much more robust to partial occlusions than the standard approach considering the whole object.

The part-based people detection was considered a number of times. Seemann et al. [14] use SIFT based part detectors but do not model part occlusions. Wu and Nevatia [22] describe the part occlusions but the occlusion probabilities and part positions are learned in a supervised manner. We base our algorithm on a principled probabilistic model of the spatial arrangement of the parts similar to the work of Weber, Perona and colleagues [21, 5]. An advantage of having a proper probabilistic model is that, after constructing the part detectors, the part arrangement and occlusion probabilities can be automatically learned from unlabelled images. This chapter presents a part-based approach and shows how it can be used to prop-

erly combine information from multiple sensors on a mobile robot, 2D range data, omnidirectional camera and pan-tilt camera in our case.

3.3 Part-based Model

Legs of a person standing in front of a robot can be detected using a 2D laser range scanner [11]. A part-based model is presented that takes into account the possible distance between the detected persons legs. The fact that leg detector might produce false detections or fail to detect legs, for example because of partial occlusion, is taken into account. The model also presents the base for combining information from different sensors as described later.

3.3.1 Part Detection

A human is detected by detecting P human body parts, in this case $P = 2$ for the legs. The 2D position of a leg is $\mathbf{x}_p = (x_p, y_p)$. The Gaussian distribution is used as a simple model of the leg positions:

$$p_{shape}(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu, \Sigma) \tag{3.1}$$

where $\mathbf{x} = \begin{pmatrix}\mathbf{x}_1 & ... & \mathbf{x}_P\end{pmatrix}$ is a $2P$ long vector containing all the 2D part positions, μ is the mean and Σ is a $(2P) \times (2P)$ covariance matrix. If the covariance matrix is diagonal than this model can be seen as describing "string-like" constraints between the body-part positions [4]. The non-diagonal covariance matrix will express additional relations between the positions of the body parts.

A laser range scan is first divided into segments by detecting abrupt changes using the Canny edge detector. Reliable leg detection is performed using a set of geometric features and Ada-Boost classifier as described in [24]. Let N denote the number of segments classified as legs and let $\mathbf{x}_j$ denote the 2D position of the j-th detection. All leg detections from one scan are given by:

$$\mathscr{X} = \begin{pmatrix}\mathbf{x}_1 & \mathbf{x}_2 & ... & \mathbf{x}_N\end{pmatrix} \tag{3.2}$$

The 2D image position $\mathbf{x}_j = (x_j, y_j)$ of the j-th detection is calculated as the mean position of the 2D scan segment points. Note that sometimes the legs cannot be separated or their appearance in the scan might change drastically. Furthermore, in cluttered environments many other objects, e.g. chairs or tables, may produce 2D scan output similar to human legs and some detections might be false detections.

3.3.2 Missing Detections and Clutter

From a range scan the collection of person's leg candidates $\mathscr{X}$ is extracted but some of them are true and some false detections. To indicate which detections are correct a $P = 2$ element vector $\mathbf{h}$ is used with element $h_p = j$, $j > 0$, indicating that the j-th detection $\mathbf{x}_j$ belongs to the of the p-th body part (leg) and the other detections of that part are false detections. Given $\mathbf{h}$ the 2D positions of the person's legs are composed of the corresponding detections $\mathbf{x} = \left(\mathbf{x}_{h_1}\mathbf{x}_{h_2}\right)$. The set of all other detections that belong to the background clutter are denoted by $\mathbf{x}^{bg}$.

It is possible that a leg was not detected indicated using $h_p = 0$. The position of a not detected leg is considered as missing data. To make distinction between the missing and the observed parts the set of missing parts is denoted as $\mathbf{x}^m$ and the set of observed parts as $\mathbf{x}^o$. To indicate the fact that there can be missing parts, the probabilistic model of the arrangement of the body parts (3.1) will be written as: $p_{shape}(\mathbf{x}) = p_{shape}(\mathbf{x}^o, \mathbf{x}^m)$.

3.3.3 Probabilistic Model

The the possibility of part detector false alarms and missed detections of body parts of a person is determined by the unknown assignment hypotheses vector $\mathbf{h}$. The probabilistic model can be written as a joint distribution:

$$p(\mathscr{X}, \mathbf{x}^m, \mathbf{h}) = p(\mathscr{X}, \mathbf{x}^m|\mathbf{h})p(\mathbf{h}) \tag{3.3}$$

where both $\mathbf{x}^m$ and $\mathbf{h}$ are unknown missing data.

Two auxiliary variables $\mathbf{b}$ and $\mathbf{n}$ are used to further define $p(\mathbf{h})$. The variable $\mathbf{b} = sign(\mathbf{h})$ is a binary vector that denotes which parts have been detected and which not. The value of the element $n_p \leq N_p$ of the vector $\mathbf{n}$ represents the number of detections of part p that are assigned to the background clutter. The joint distribution (3.3) becomes:

$$p(\mathscr{X}, \mathbf{x}^m, \mathbf{h}, \mathbf{n}, \mathbf{b}) = p(\mathscr{X}, \mathbf{x}^m|\mathbf{h})p(\mathbf{h}|\mathbf{n}, \mathbf{b})p(\mathbf{n})p(\mathbf{b}) \tag{3.4}$$

where $\mathbf{b}$ and $\mathbf{n}$ are assumed to be independent and:

$$p(\mathscr{X}, \mathbf{x}^m|\mathbf{h}) = p_{shape}(\mathbf{x}^o, \mathbf{x}^m)p_{bg}(\mathbf{x}^{bg}) \tag{3.5}$$

where the observed parts $\mathbf{x}^o$, the missing parts $\mathbf{x}^m$ and the false detections from clutter $\mathbf{x}^{bg}$ correspond to the hypothesis $\mathbf{h}$. The $p_{bg}(\mathbf{x}^{bg})$ is the distribution of the false detections usually uniform or a wide Gaussian.

The probability $p(\mathbf{b})$ describing the presence or absence of parts is modelled as an explicit table of joint probabilities. Each part can be either detected or not, so there are in total 2^P possible combinations that are considered in $p(\mathbf{b})$.

The background part detections are assumed independent of each other and the number of detections $\mathbf{n}$ is modelled using Poisson distribution with mean M_p [21]. Different M_p-s for different parts admit different detector statistics. The Poisson parameter will be denoted by vector $\mathbf{M} = \begin{pmatrix} M_1 & \dots & M_P \end{pmatrix}$.

The density $p(\mathbf{h}|\mathbf{n},\mathbf{b})$ is defined as:

$$p(\mathbf{h}|\mathbf{n},\mathbf{b}) = \begin{cases} 1/|\mathscr{H}(\mathbf{b},\mathbf{n})| & \text{if } \mathbf{h} \in \mathscr{H}(\mathbf{b},\mathbf{n}), \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

where $\mathscr{H}(\mathbf{b},\mathbf{n})$ is the set of all hypotheses consistent with the values of $\mathbf{b}$ and $\mathbf{n}$. Here $|\mathscr{H}(\mathbf{b},\mathbf{n})|$ denotes the total number all consistent part assignment hypotheses. This expresses that these hypotheses are considered equally likely.

3.3.4 Learning Model Parameters

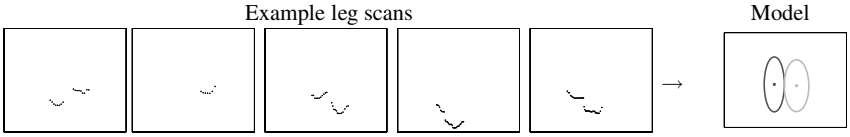


Fig. 3.2 Example person's legs scans from the data set used to train the probabilistic part-based model and the learned model parameters. For each part its mean position contained in the parameter μ is presented. The ellipse represents the 1-sigma uncertainty of the part position as described by the diagonal elements of the covariance matrix Σ.

The density distribution (3.4) will have the following set of parameters $\Omega = \{\mu, \Sigma, p(\mathbf{b}), \mathbf{M}\}$:

$$p(\mathscr{X}, \mathbf{x}^m, \mathbf{h}) = p(\mathscr{X}, \mathbf{x}^m, \mathbf{h}|\Omega) \tag{3.7}$$

The likelihood of a collection of detected parts $\mathscr{X}$ is obtained by integrating over the hidden hypotheses $\mathbf{h}$ and the missing parts:

$$p(\mathscr{X}|\Omega) = \sum_{\text{all possible } \mathbf{h}} \int_{\mathbf{x}^m} p(\mathscr{X}, \mathbf{x}^m, \mathbf{h}|\Omega). \tag{3.8}$$

Integrating over the missing parts $\mathbf{x}^m$ for the Gaussian distribution can be performed in closed form.

To estimate the parameters of the model a set of L aligned scans of persons is used. The collection of leg detections for i-th scan will be denoted as $\mathscr{X}_i$. The maximum likelihood estimate of the parameters Ω is computed by maximizing the likelihood of the data:

$$\prod_{i}^{L} p(\mathscr{X}_i|\Omega) \tag{3.9}$$

using expectation maximization algorithm, see [21] for details.

3.3.5 Detection

Let us denote the maximum likelihood parameters learned from a set of scans of persons as Ω_{person}. For a set of scans from the office clutter the $p_{bg}(\mathbf{x}^{bg})$ and other parameters can be estimated, denoted as Ω_{bg}. Given a new scan and extracted the set of detected parts $\mathscr{X}$. The scan is either a scan of a person or some background clutter:

$$p(\mathscr{X}) = p(\mathscr{X}|Person)p(Person) + p(\mathscr{X}|BG)p(BG) \tag{3.10}$$

where $p(Person)$ and $p(BG)$ are unknown a priori probabilities that the scan contains a person or background. The a posteriori probability that there is a person is:

$$p(Person|\mathscr{X}) = \frac{p(\mathscr{X}|Person)p(Person)}{p(\mathscr{X})} \approx \tag{3.11}$$

$$\frac{p(\mathscr{X}|\Omega_{person})p(Person)}{p(\mathscr{X}|\Omega_{person})p(Person) + p(\mathscr{X}|\Omega_{bg})p(BG)} \tag{3.12}$$

The last step above is an approximation since the maximum likelihood estimates for the model parameters Ω_{person} and Ω_{bg} are used instead of integrating over all possible parameter values. Calculating $p(\mathscr{X}|\Omega)$ is done using (3.8).

3.4 Combining Multiple Sensors

Robots are often equipped with multiple sensors. For example an omnidirectional and a pan-tilt camera as in Figure 3.1. In this section the part based model from the previous section is extended to include part detections from the corresponding images.

3.4.1 Part Detection in Images

Haar-like-feature classifiers are used to detect various human body parts in images. Each classifier is trained using Ada-Boost algorithm on a large set of example images of the corresponding body part [19]. Here the classifiers are trained on face, upper body, lower body and full body images. The part detectors can lead to many false alarms and missed detections [12], see Figure 3.3.

Fig. 3.3 Example body part detections with some false detections.

3.4.2 Extending the Part-based Model

The part based model from the previous section that was applied to the 2D range leg detections can be easily extended with the human body parts detected in the images. Instead of 2 parts there will be $P = 2+3+2 = 7$ body parts and $\mathbf{x} = \left(\mathbf{x}_1 \ \ldots \ \mathbf{x}_P\right)$ is a $2P$ long vector containing 2D leg positions, the 2D image positions for the upper body, lower body and full body detected in omnicam images and face and upper body detected positions from the pan-tilt camera.

The positions of all detected parts are summarized in a data structure:

$$\mathscr{X} = \begin{pmatrix} \mathbf{x}_{1,1} & \mathbf{x}_{1,2} & \cdots & \mathbf{x}_{1,N_{leg1}} & \\ \mathbf{x}_{2,1} & \mathbf{x}_{2,2} & \cdots & \mathbf{x}_{2,N_{leg2}} & \\ \mathbf{x}_{3,1} & \mathbf{x}_{3,2} & \cdots & \cdots & \mathbf{x}_{3,N_{up-body}} \\ \mathbf{x}_{4,1} & \mathbf{x}_{4,2} & \cdots & \mathbf{x}_{4,N_{low-body}} & \\ \mathbf{x}_{5,1} & \mathbf{x}_{5,2} & \cdots & \cdots & \mathbf{x}_{5,N_{full-body}} \\ \mathbf{x}_{6,1} & \mathbf{x}_{6,2} & \cdots & \cdots & \mathbf{x}_{6,N_{face-pan-tilt}} \\ \mathbf{x}_{7,1} & \mathbf{x}_{7,2} & \cdots & \mathbf{x}_{7,N_{up-body-pan-tilt}} & \end{pmatrix} \tag{3.13}$$

with one row per part and where each row contains information about the detections of the corresponding body part. The first two rows are repeated since the same detector is used for both legs detected in the range scans. The element $\mathbf{x}_{p,j}$ contains the 2D positions for the legs or the 2D image position for the parts detected in images of the j-th detection of the p-th part. The rows of $\mathscr{X}$ can have different lengths and some might be empty if that part was not detected.

Again the hidden P dimensional assignment vector $\mathbf{h}$ is used with element $h_p = j$, indicating that the j-th detection of the p-th part $\mathbf{x}_{p,j}$ belongs to the object and other detections of that part are false detections. Given $\mathbf{h}$ the shape of the object is composed of the corresponding detections $\mathbf{x} = \left(\mathbf{x}_{1,h_1} \ \ldots \ \mathbf{x}_{P,h_P}\right)$. Note that since the same detector is used for both legs, care should be taken not to select the same leg detection for both legs.

The other model equations remain the same and the same procedure can be used to learn now the part based model containing the part detectors from both sensors.

Example part based model learned from multiple sensors is presented in Figure 3.4.

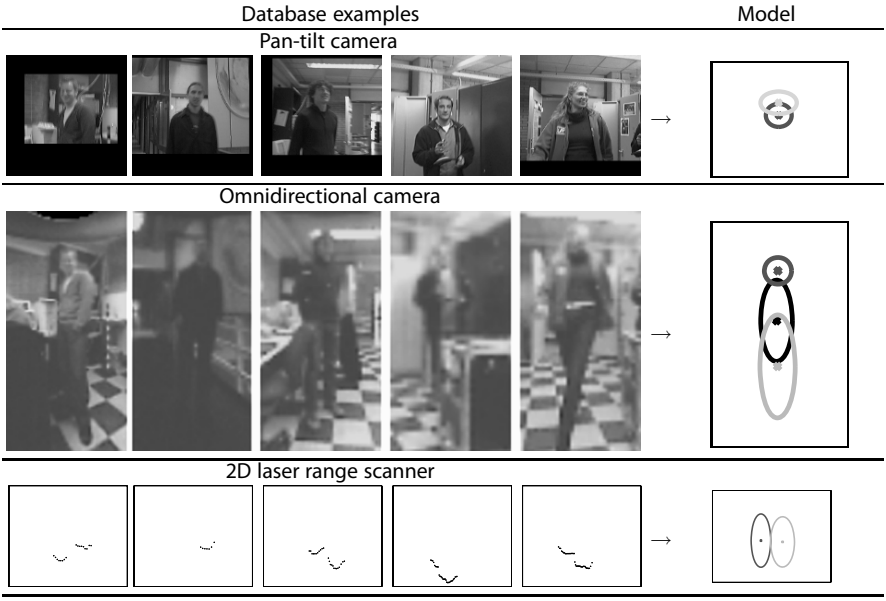


Fig. 3.4 Examples from the data set used to train the probabilistic part-based model and examples of learned part arrangement model parameters. For each part its mean position contained in the parameter μ is presented. The ellipse represents the 1-sigma uncertainty of the part position as described by the diagonal elements of the covariance matrix Σ.

3.5 Experiments

The presented method for combining information from multiple sensors is evaluated here. Building reliable part detectors for each sensor is considered first.

3.5.1 Part Detection

A data set of 2D range scans was recorded to build the leg detector. The URG-04LX 2D range scanner was mounted on our robot at 50cm above the floor. A set of 3530 scans was recorded while driving the robot through the corridors and cluttered offices in our building. This gives in total 4032 scan segments corresponding to person's legs and 14049 segments from the background clutter.

For each scan segment we extract the set of 12 geometric features, such as segment size, curvature, see [24] for details. The Gentle AdaBoost algorithm [7] then automatically selects the relevant features during training. Separate features are used to build simple linear classifiers, the so called "weak" classifiers. The classifiers are then combined to form the final classifier. The final classifier contains 25 classifiers. The number of classifiers was chosen to such that recognition results do not improve for adding more classifiers. Note that some features might be selected a number of times. The final classifier leads to a reliable leg detection. However, leg detection in cluttered office environments remains difficult since many object can produce range scan similar to legs. Details are given in [24].

The Haar-like-feature based image part detectors we used in our experiments were trained on the MIT pedestrian data set [12] and are available in the Intel OpenCV library.

3.5.2 Multiple Sensor People Detection

For evaluating the part-arrangement model, we collected a realistic data set simulating the scenario where a human is introducing the robot to a new environment. Five persons were asked to lead the robot around our office environment. The robot was teleoperated. During the teleoperation the movements of the robot were such as to try to keep the person in the field of view of the pan-tilt camera. The data set contains 3200 images from the both cameras captured at 4 frames/second and the corresponding laser scans. On average each person was leading the robot for $2-3$ minutes and there were around 600 images recorded for each person.

The calibrated omnidirectional camera images were used to manually select the ground truth person position on the floor-plane. The selected person position was used to cut out the corresponding regions from all three calibrated sensors, see Figure 3.1. The aligned images cut out of the omnicam images were 56×112 pixels and the corresponding images from the pan-tilt camera were 112×112 pixels, see Figure 3.4.

From the whole data set, 1000 randomly chosen images and scans were used to train our part based model and remaining part of the data set was used for testing. The automatically learned part based model parameters are presented in Figure 3.4. It can be observed that there is more uncertainty in the positions of the full and lower body than for the upper body region in the omnicam images. The pan-tilt camera was not very stable and it was shaking during the robot movements. This explains the larger uncertainty in the horizontal position of the detected face and upper body in the images from the pan-tilt camera.

In order to test recognition results, a set of 5000 no-person parts of the images and scans are selected from the data set. The recognition results on this data set containing aligned images of people and corresponding scan segments are summarized in Figure 3.5. The results are presented as recognition Receiver Operating Characteristic (ROC) curves. Changing the a priori chances $p(Person)$ and $p(BG)$ in (3.11),

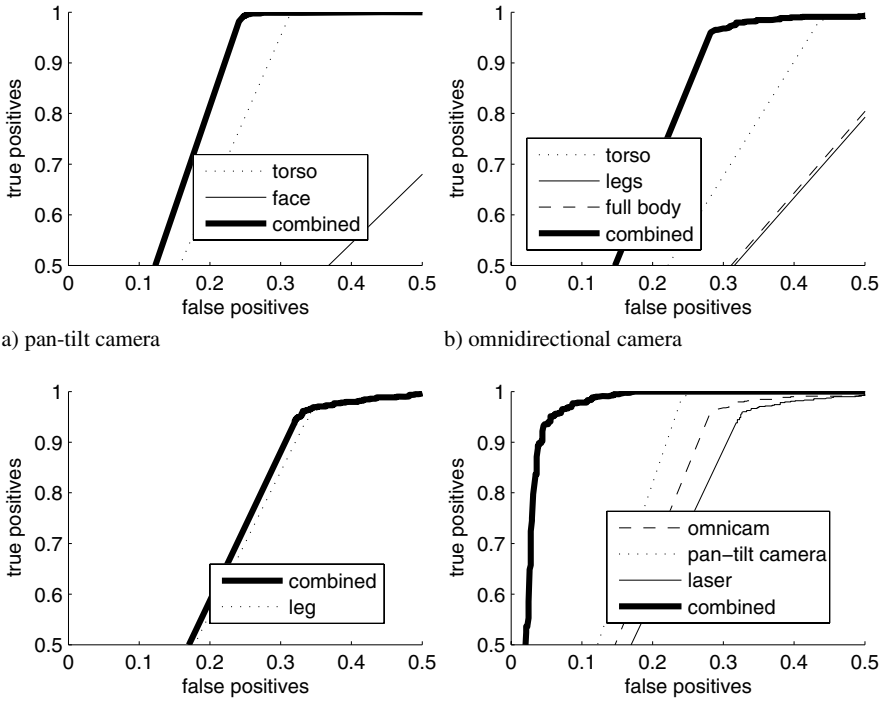


Fig. 3.5 Recognition Receiver Operating Characteristic (ROC) curves

various values are obtained for true positive (TP) and false positive (FP) detections used to plot the ROC curves. The ROC curves of the single part detectors for each sensor are also reported. Face detection in images is usually very reliable. However, for the pan-tilt camera images from our data set the face detector performs poorly, Figure 3.5a. This is mainly because the persons were not facing the robot very often during the trials where they were leading the robot around our office environment. The persons were facing the robot at the start of each trial and also later on from time to time when they turned around to check if the robot is following them. Furthermore, it can be observed from the ROC curves that the combination of parts leads to much better results. The improvement is small for the laser scanner, Figure 3.5c. The largest improvement can be noted when the different sensor are combined, Figure 3.5d.

Learning the part detectors using the Ada-Boost requires often long time and many training examples [15]. On the other hand, once the part detectors are available, learning the part arrangement model usually does not require many training examples [5]. Learning the part arrangement model parameters for the 7 parts takes around 2 minutes for 1000 images in our Matlab implementation. In Figure 3.6 the recognition accuracy for the various sizes of the data set used to training the part arrangement model is presented. For each size of the training data set the experi-

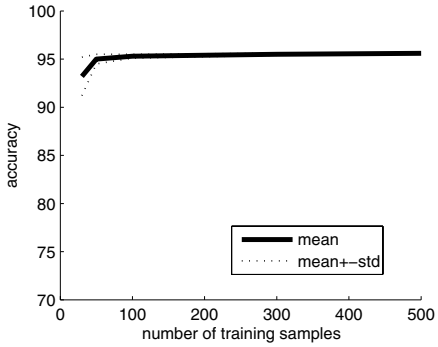


Fig. 3.6 Recognition accuracy for various sizes of the dataset used to training the part arrangement model. Mean results and the standard deviation from 10 random trials is presented.

ments are repeated by choosing the training data randomly 10 times. The mean and the standard deviation of the maximum accuracy is presented. It can be observed that consistent high accuracy recognition can be achieved even with only 50 training data samples. In practice this means that given reliable part detectors for each sensor, only a small additional effort needs to be made to construct the part-based combination of the detection results from multiple sensors.

3.5.3 Recognition from the Robot

The part based model detection is implemented on our robot, see Figure 3.1. The assumption is made that the people walk over a flat ground floor surface - true in most man-made environments. A set of possible 2D floor positions $\mathscr{T}_t$ is defined. In the experiments a $10m \times 10m$ area around the robot is used and a grid of possible positions at every $10cm$. This gives 10000 possible floor points $\mathscr{T}_t$ to evaluate the part based model. The sensors are calibrated and their pose with respect to the floor plane is known. Given the typical size of a person we can define a region of interest (ROI) in each sensor corresponding a floor plane position, see Figure 3.1. The data from the National Center for Health Statistics (*www.cdc.gov/ nchs/*) is used. For adult humans, the mean height is 1.7m with a standard deviation of 0.085m. The maximal height of a human is taken to be the mean plus three standard deviations and the width to be 1/2 of the height. For each floor position $\mathscr{T}_t$ we also extract the corresponding segments from the images and the range scan and use (3.11) to decide if there is a person at that floor position. Since (3.11) is computed at a dense grid of ground points, it often has large values for a number of ground points around the position where the person actually is. Therefore the persons are detected as the local maxima of (3.11).

The first stage of the algorithm where the body parts are detected in the omnidirectional images is the most computationally expensive. Running the three Haar-

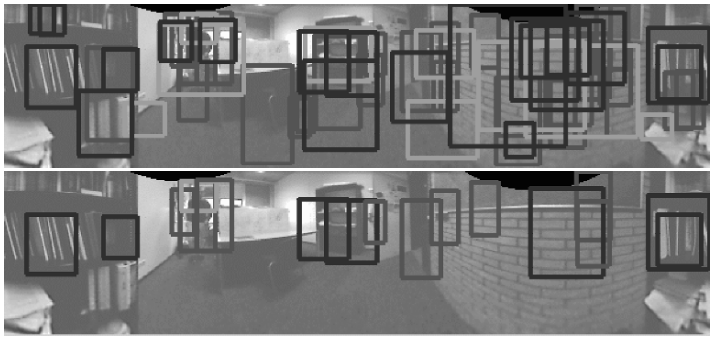


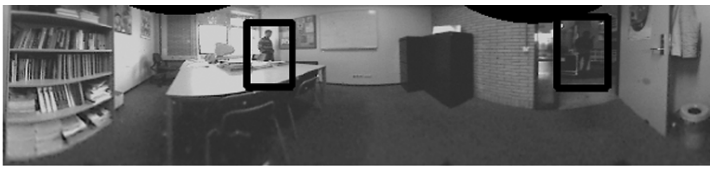
Fig. 3.7 Body part detection in omnicam images (top) and the heavily reduced set of detections when the floor plane constraint is used (below).

like-feature based part detectors on a 600×150 panoramic image takes on average 400ms on a 2GHz PC. This is the time needed for checking every image position and all possible part sizes. The possible part sizes start from the initial part size and then the part size is increased 1.1 times until it gets out of the image borders. The floor constraint can heavily reduce the number of positions and part sizes to search and detection can be done in around 100ms, see Figure 3.7. Once the parts are detected, detecting persons using our model takes around 25ms. Currently, the people detection with all three sensors and 7 detected parts can be performed 5 times/second in our implementation on a 2GHz single processor.

In Figure 3.8 a few panoramic images with the detection results are presented to illustrate the typical detection results. The data set from the human following trials was used to evaluate the actual detection performance on a mobile robot. A small subset of 100 annotated images and scans is used to train the model. The model is then applied using the floor constraint to detect people in the images and the range scans. The ground truth positions manually selected from the omnicam images were used to evaluate the performance. If a person was detected the corresponding rectangle ROI in the omnicam image was calculated, see Figure 3.8, and compared to the manually selected one using a relative overlap measure. Let R_{gt} be the image region defined by the ground truth bounding box. Let R_e be the estimated rectangle ROI (corresponding to the local maximum of (3.11)). The relative overlap is defined by:

$$overlap = \frac{R_e \cap R_{gt}}{R_e \cup R_{gt}} \tag{3.14}$$

where $R_e \cap R_{gt}$ is the intersection and $R_e \cup R_{gt}$ is the union of the two image regions. The relative overlap can have values between 0 and 1. A detection is considered to be true detection if the overlap was larger than 0.5. For the people following data set of 3200 sensor readings, there were 96% correctly detected people and only 120 false detections.



Two correct detections of partially occluded people.

Two correct detections. The persons are in the dark and hardly visible

One correct and one false detection.

Fig. 3.8 Example people detection results in panoramic images recorded from a moving robot.

3.6 Conclusions

Due to the large variability in shape and appearance of different people the problem of people detection in images remains difficult even after many years of research [15]. The detection results can be improved if multiple sensors are combined. This chapter presents a people detection approach that combines information from a set of calibrated sensors. Mobile robots that often have various senors are a typical example of a multisensory system. The approach is inspired by the part-based object representation from the computer vision area. A person is represented by a constellation of body parts. The person body parts are detected and the parts are constrained to be at certain positions with respect to each other. The presented probabilistic model combines the part detections from multiple sensors and can achieve person detection robust to partial occlusions, part detector false alarms and missed detections of body parts. The method is evaluated using a mobile test platform equipped with a pan-tilt camera, an omnidirectional camera and a 2D laser range scanner. The evaluation results show that highly reliable people detection can be achieved by properly combining the three sensors.

Acknowledgements This work has been sponsored by EU FP6-002020 COGNIRON ("The Cognitive Companion") project.

References

1. Breazeal, C.: Designing sociable robots. MIT Press, Cambridge (2002)
2. Burl, M., Leung, T., Perona, P.: Recognition of planar object classes. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (1996)
3. Fei-Fei, L., Perona, P.: A Bayesian hierarchical model for learning natural scene categories. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (2005)
4. Felzenszwalb, P., Huttenlocher, D.: Pictorial structures for object recognition. Intl. Journal of Computer Vision **61**(1), 55–79 (2005)
5. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (2003)
6. Ferryman, J., Crowley, J. (eds.): Proc. of the 9th IEEE Int. Workshop on Performance Evaluation of Tracking and Surveillance (2006)
7. Friedman, J.H., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Technical Report, Dept. of Statistics, Stanford University (1998)
8. Gavrila, D., Philomin, V.: Real-time object detection for smart vehicles. In Proc. of the Intl. Conf. on Computer Vision (1999)
9. Giebel, J., Gavrila, D., Schnrr, C.: A Bayesian framework for multi-cue 3D object tracking. In Proc. of the European Conf. on Computer Vision (2004)
10. Hoiem, D., Efros, A., Hebert, M.: Putting objects in perspective. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (2006)
11. Arras, K., Mozos, O., Burgard, W.: Using boosted features for detection of people in 2D range scans. In Proc. of the IEEE Intl. Conf. on Robotics and Automation (2007)
12. Kruppa, H., Castrillon-Santana, M., Schiele, B.: Fast and robust face finding via local context. In: Proc of the IEEE Intl. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (2003)
13. Schulz, D., Burgard, W., Fox, D., Cremers, A.: People tracking with a mobile robot using sample-based joint probabilistic data association filters. International Journal of Robotics Research **22**(2), 99–116 (2003)
14. Seemann, E., Leibe, B., Mikolajczyk, K., Schiele, B.: An evaluation of local shape-based features for pedestrian detection. In Proc. of the British Machine Vision Conference (2005)
15. Munder, S., Gavrila, D.M.: An experimental study on pedestrian classification. IEEE Trans. on Pattern Analysis and Machine Intelligence **28**(11), 1863–1868 (2006)
16. Fong, T., Nourbakhsh, I., Dautenhahn, K.: A survey of socially interactive robots. Robots and Autonomous Systems **42**, 143–166 (2003)
17. Treptow, A., Cielniak, G., Duckett, T.: Real-time people tracking for mobile robots using thermal vision. Robotics and Autonomous Systems **54**(9), 729–739 (2006)
18. Tuzel, O., Porikli, F., Meer, P.: Human detection via classification on riemannian manifolds. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (2007)
19. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (2001)
20. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In Proc. of the Intl. Conf. on Computer Vision (2003)
21. Weber, M., Welling, M., Perona, P.: Unsupervised learning of models for recognition. In Proc. of the European Conf. on Computer Vision (2000)
22. Wu, B., Nevatia, R.: Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In Proc. of the Intl. Conf. on Computer Vision (2005)
23. Zajdel, W., Zivkovic, Z., Kröse, B.: Keeping track of humans: have I seen this person before? In Proc. of the IEEE Intl. Conf. on Robotics and Automation (2005)
24. Zivkovic, Z., Kröse, B.: Part based people detection using 2D range data and images. In Proc. of the IEEE/RSJ Intl. Conf. Robots and Autonomous Systems (IROS) (2007)

Chapter 4
Perceiving Objects and Movements to Generate Actions on a Humanoid Robot

Tamim Asfour, Kai Welke, Aleš Ude, Pedram Azad and Rüdiger Dillmann

4.1 Introduction

To deal with problems in perception and action researchers in the late 80s introduced two new frameworks, one under the heading of active vision (animate, purposive, behavioral) originating in the field of computer vision and the other in AI/robotics under the heading of behavior-based robotics. In both formalisms, the old idea of conceiving an intelligent system as a set of modules (perception, action, reasoning) passing results to each other was replaced by a new way of thinking of the system as a set of behaviors. Behaviors are sequences of perceptual events and actions. These efforts still go on, but only with limited success up to now. One reason for this is that although it was expected that active vision would make many perceptual problems easier, machine perception still remains rather primitive when compared to human perception. A further reason for failure is that behaviors were often designed ad hoc without studying the interplay between objects and actions in depth, which is necessary to develop structures suitable for higher-level cognitive processes. A third reason was that no one succeeded in formulating a general enough theory for behavior-based robotics. Hence, it remains difficult or even impossible to predict how a newly designed behavior-based system will scale and deal with new situations.

In recent years there are renewed efforts to develop autonomous systems and especially humanoid robots (see [7, 1, 11, 14, 12, 3]), i.e. (embodied) robots that perceive, move and perform (simple) actions. The successful attempts in this area are still limited to simple scenarios, very much for the same reasons mentioned

Tamim Asfour, Kai Welke, Pedram Azad and Rüdiger Dillmann
University of Karlsruhe, Institute for Computer Science and Engineering, Industrial Applications for Informatics and Microsystems (IAIM)), P.O. Box 6980, D-76128 Karlsruhe, Germany, e-mail: \{asfour,welke,azad,dillmann\}@ira.uka.de

Aleš Ude
Jožef Stefan Institute, Dept. of Automatics, Biocybernetics, and Robotics, Jamova 39, 1000 Ljubljana, Slovenia, e-mail: ales.ude@ijs.si

above. One should also note the extensive research on visual recognition and categorization in computer vision, which resulted in quite advanced and efficient recognition methods, although seldom tested on real world scenes. Moreover, the relationship between this research and the development of cognitive systems is still weak; since these approaches usually assume that what constitutes objects and categories is given a priori by the external world, they do not pertain to seeing agents and their actions. Research into cognitive robots should combine the study of perceptual representations that facilitate motor control, motor representations that support perception, and learning based on actively exploring the environment and interacting with people that provides the constraints between perception and action. This will then allow, e.g., to learn the actions that can be carried out on and with objects, which leads to what we call Object-Action-Complexes (OAC).

The concept of Object-Action Complexes (OACs) has been introduced by the European PACO-PLUS consortium ([8]) to emphasize the notion that for a cognitive agent objects and actions are inseparably intertwined and that categories are therefore determined (and also limited) by the action an agent can perform and by the attributes of the world it can perceive. The resulting OACs are the entities on which cognition develops (action-centered cognition). Entities (*things*) in the world of a robot (or human) will only become semantically useful *objects* through the action that the agent can/will perform on them.

In this work we present a new humanoid active head which features human-like characteristics in motion and response and mimics the human visual system. We present algorithms that can be applied to perceive objects and movements, which form the basis for learning actions on the humanoid. For action representation we use an HMM-based approach to reproduce the observed movements and build an action library. Hidden Markov Models (HMM) are used to represent movements demonstrated to a robot multiple times. They are trained with the characteristic features (key points) of each demonstration. We propose strategies for adaptation of movements to the given situation and for the interpolation between movements stored in a movement library.

4.2 Active Humanoid Head

The humanoid robot ARMAR III has been designed under a comprehensive view so that it can perform a wide range of tasks and not only a particular task. In the design of the robot, we desire a humanoid that closely mimics the sensory and sensory-motor capabilities of the human. The robot should be able to deal with a household environment and the wide variety of objects and activities encountered in it.

To achieve the above goals, we use an integrated humanoid robot consisting of a humanoid head with seven degrees of freedom (DOF), two arms (seven DOF per arm) and five-finger hands (eight DOF per hand), a torso with three DOF, and a holonomic mobile platform. In designing the robot, we desire a humanoid that closely mimics the sensory and sensory-motor capabilities of the human. Therefore, the

Fig. 4.1 The humanoid robot ARMAR-III. The has 43 DOF. From the kinematics control point of view, the robot consists of seven subsystems: head, left arm, right arm, left hand, right hand, torso, and a mobile platform.

robot is equipped with manipulative, perceptive and communicative skills necessary for real-time interaction with the environment and humans.

4.2.1 System Requirements

We pay special attention to the design of the head since the head can provide rich perceptual input necessary to realize various visuo-motor behaviors, e.g. smooth pursuit and saccadic movements towards salient regions, and also more complex sensory-motor tasks such as hand-eye coordination, gesture identification, human motion perception and linking of visual representations to the motor representations. The major design criteria were as follows:

- The robot head should be of realistic human size and shape while modelling the major degrees of freedom (DOF) found in the human neck/eye system, incorporating the redundancy between the neck and eye DOF.
- The robot head should feature human-like characteristics in motion and response, that is, the velocity of eye movements and the range of motion will be similar to the velocity and range of human eyes.

- The robot head must enable saccadic motions, which are very fast eye movements allowing the robot to rapidly change the gaze direction, and smooth pursuit over a wide range of velocities.
- The optics should mimic the structure of the human eye, which has a higher resolution in the fovea.
- The vision system should mimic the human visual system while remaining easy to construct, easy to maintain and easy to control.

With this set of requirements, a first version of the head have been developed as part of a humanoid robot that will allow for the integration of motor control and perception. This is essential to enable explorative head, hand, and body movements for learning of OACs.

4.2.2 Head Motor System

The head has seven DOF and is equipped with two eyes. Each eye can independently rotate about a vertical axis (pan DOF), and the two eyes share a horizontal axis (tilt DOF). To approximate These two DOF allow for human-like eye movements1. The visual system is mounted on a neck mechanism [1] with four DOF organized as pitch-roll-yaw-pitch.

4.2.3 Head Sensory System

To start learning object-action complexes we must, firstly, identify regions that potentially contain objects of interest and secondly analyze these regions to build higher-level representations. While the first task is closely related to visual search and can benefit from a wide field of view, a narrower field of view resulting in higher-resolution images of objects is better suited for the second task. While the current technology does not allow us to exactly mimic the features of the human visual system and because camera systems that provide both peripheral and foveal vision from a single camera are still experimental, we decided for an alternative which allows to use commercially available camera systems that are less expensive and more reliable. Foveated vision was realized using two cameras per eye, one with wide-angle lens for peripheral vision and one with narrow-angle lens for foveal vision. We use the Point Grey Research Dragonfly IEEE-1394 camera in the extended version (www.ptgrey.com). The extended version allows the CCD to be up to 6 inches away from the camera interface board. This arrangement helps with accessing hard to reach places, and with placing the lens into a small volume. Since the cameras are very light and are extended from the interface board by a flexible extension cable, they can be moved with small and low-torque servos.

The cameras can capture colour images at a frame rate of up to 30 Hz. They implement the DCAM standard, and transmit a raw 8 bit Bayer Pattern with a res-

olution of 640x480, which is then converted on the PC to a 24 bit RGB image. The cameras have a FireWire interface, which is capable of delivering data rates of up to 400 Mbps. The benefit of transmitting the Bayer Pattern is that only a third of the bandwidth is needed for transmitting the colour image without loosing any information. Thus, it is possible to run one camera pair at a frame rate of 30 Hz and the other at a frame rate of 15 Hz, all being synchronized over the same FireWire bus, without any additional hardware or software effort. Running the foveal cameras, which have a smaller focal length and thus a narrower view angle, at a lower frame rate is not a drawback because these cameras are not crucial for time critical applications such as tracking, but are utilized for detailed scene analysis, which does not need to be performed at full frame rate in most cases anyway. The camera is delivered as a development kit with three micro lenses with the focal lengths 4, 6, and 8mm. In addition, one can use micro lenses with other focal lengths as well. We have chosen a 3 mm micro lens for the peripheral cameras and a 16 mm micro lens for the narrow angle cameras. Furthermore, the head is equipped with six microphones (SONY ECMC115.CE7): two in the ears, two in the front and two in the rear of the head. These microphones will be used in the later phase of the project to achieve a richer multi-sensory representation of objects and environment and to support the integration of speech components in order to provide an additional information for interaction and natural communication.

4.3 Perceiving Objects and Movements

4.3.1 Human Motion Tracking

For the tracking of human motion, an image-based markerless human motion capture system has been developed[6, 5]. The input of the system are stereo colour images of size 320×240 captured at 25 Hz, with two calibrated Dragonfly cameras built-in into the head of the humanoid robot ARMAR III. The input images are pre-processed, generating output for the gradient cue, the distance cue, and an optional region cue, as described in [5]. Based on the output of the image processing pipeline, a particle filter is used for tracking the movements in configuration space. The overall likelihood function to compute the a-posteriori probabilities is formulated as:

$$p(\mathbf{z}|\mathbf{s}) \propto \exp\left\{ -\frac{1}{2}\left(\frac{1}{\sigma_d^2}\sum_{i=1}^{3} d_i(\mathbf{s},\mathbf{c}_i) + \frac{1}{\sigma_g^2}(1 - \frac{1}{M_g}\sum_{m=1}^{M_g} g_m) \right) \right\}, \qquad (4.1)$$

where $\mathbf{s}$ is the configuration to be evaluated, $\mathbf{z}$ is a general denotation for the current observations i.e. the current input image pair, and $\mathbf{c}_i \in \mathbb{R}^3$ with $i \in \{1,2,3\}$ denotes the triangulated 3D position of the hands and the head. The function $d_i(\mathbf{s},\mathbf{c})$ is defined as:

Fig. 4.2 Illustration of the performance of the markerless human motion capture system. Left: projection of the estimated configuration into the left camera image. Right: 3D visualization of the estimated configuration with an articulated human model.

$$d_i(\mathbf{s},\mathbf{c}) := \begin{cases} |f_i(\mathbf{s}) - \mathbf{c}|^2 & : \quad \mathbf{c} \neq \mathbf{0} \\ 0 & : \quad \text{otherwise} \end{cases},$$

where $n := \dim(\mathbf{s})$ is the number of DOF of the human model. The transformation $f_i : R^n \rightarrow R^3$ transforms the n-dimensional configuration of the human model into the 3D position of the left hand, right hand or head respectively, using the forward kinematics of the human model. The g_m with $m \in \{1,2,...,M_g\}$ denote the intensity values in the gradient image (which is derived from the input images $\mathbf{z}$) at the M_g pixel coordinates of the projected contour of the human model for a given configuration $\mathbf{s}$. This process is performed for both input images using the calibration parameters of each camera. For each image pair of the input sequence the output of the system is the estimation of the particle filter, given by the weighted mean over all particles. A detailed description is given in [5].

In contrast to the acquisition method based on the magnetic tracking system, the joint angle values θ_3, θ_4, θ_5, and θ_6 are calculated *directly* and therefore the position of the elbow does not have to be approximated based on empirical studies but is determined explicitly.

4.3.2 Object Representations for Actions

Our scheme of object representation is driven by the conviction that objects and actions are inseparably intertwined. To facilitate the execution of complex actions in the currently perceived environment, we want our system to learn performing actions on objects in two ways: learning by demonstration and learning by exploration. In the following section we want to emphasize learning by exploration and the consequences for an action related object representation scheme.

While autonomously exploring possible actions on an object and finally associating successful actions with an object, the robot retrieves a set of object action relations. The related actions for an object can be considered object affordances[9]. Associating possible actions to prior percepts only will not result in a general repre-

sentation of object affordances. Therefore, a mechanisms is necessary which allows the determination of affordances for unknown percepts on the basis of previously experienced object action relations. Below we will introduce a conceptual way to generalize object action relations to a representation, which can be used to determine the affordance of an unknown percept.

Furthermore, to allow autonomous exploration of objects, a mechanism to measure success for an executed action is necessary. This ability can not be learned in a completely autonomous way. For a new action the system initially needs a feedback, whether an action executed on an object has been successful or not. This can be provided either by an assistant who judges the action after execution or by demonstrating successfully executed actions. Once the system learned this measure of success, it can judge itself if the execution of an action on an unknown object was successful or not.

A system which allows both, generalizing for object affordances and learning of how to measure success, has to rely on two distinct sets of object features: features which are stable and features which are varying during execution. If we consider the example of filling a cup, the shape and colour of the cup itself will be stable during action execution, while the fill level changes. In the following, we denote features of an object which are stable during action execution by $F(P) = (f_1, \ldots, f_{N_f})$ and features that are varying during execution with $G(P) = (g_1, \ldots, g_{N_g})$. Each feature vector component (f_n or g_n) will be called feature channel.

Considering the invariant feature vector $F(P)$ for an action A performed on an percept P, it is clear that the affordance $afford_A(P)$ has to be triggered by elements of $F(P)$. To determine, which feature channels are responsible for the affordance, the system has to acquire enough experience with the action A on different percepts $P_1, \ldots, P_N$ and to generalize over the resulting vectors $F(P_1), \ldots, F(P_N)$. During the generalization process, two things will be determined: the relevance R_n of each feature channel f_n for the affordance and the feature values for all channels, which frequently co-occur with the action and thus are strong indicators for the affordance. To determine significant domains in feature space, clustering is performed for each feature channel using all previous percepts which are associated with successful executions of the action. For feature channel n, the resulting clusters are combined in an extended signature containing the cluster centroid $c_{n,i}$, the number of samples $w_{n,i}$ associated to the cluster, and the distance from the cluster's centroid to the farthest cluster element $d_{n,i}$:

$$S_n = \{s_i = (c_{n,i}, w_{n,i}, d_{n,i})\} \tag{4.2}$$

For each cluster the probability $p_{n,i}$ is assigned which captures how probable an element which belongs to the cluster affords the action:

$$p_{n,i} = \frac{w_{n,i}}{N} \tag{4.3}$$

A cluster with high probability $p_{n,i}$ shows that the corresponding feature channel captures relevant information for the object affordance. We calculate the relevance

R_n of a feature channel n by:

$$R_n = p_{n,max} \tag{4.4}$$

where *max* is the index of the cluster center with largest number of elements $w_{n,i}$.

With the probabilities and the feature channel relevance, the affordance of an unknown percept X can be determined. First the invariant feature vector $F(X)$ is calculated. For each feature vector component the closest cluster $c_{n,r}$ is searched. The distance e_r to the centroid is calculated and used to determine if the component is significant for the cluster. To express this in our calculations, we define a binary function of significance:

$$k_n = \begin{cases} 1 \text{ if } e_r < \alpha d_{n,r} \\ 0 \text{ otherwise} \end{cases} \tag{4.5}$$

The affordance of the percept X for the action A can then be calculated by:

$$afford_A(X) = \frac{\sum_{n=0}^{N_f} k_n p_{n,r}}{\sum_{n=0}^{N_f} R_n} \tag{4.6}$$

The system has to hold an inner model which allows to determine affordances for percepts. For the calculations we only need the extended signatures S_n. To keep only relevant channels in the inner model, we threshold the relevance R_n of each channel and discard channels with low relevance. Channels with low relevance will usually have many small clusters and discarding them helps in keeping the inner model small. Thus as inner model for affordances *IA* for the action A we can write:

$$IA_A = \{(S_n) : R_n > minrelevance\} \tag{4.7}$$

Once an action has been performed on the object, the robot has to determine if the action was successful. For this, a measure of success which relates percepts prior to execution with percepts after execution is necessary. The action is considered as continuous process over time. Thus the change of a prior percept P_0 to a percept during action execution P_t can be written in the following way:

$$P_t = C_A(P_0, t) \tag{4.8}$$

where C_A describes the change of the percept when applying the action A. The success can be measured between two percepts $P_{t_i}, P_{t_{i+1}}$, where the interval $\Delta t = t_{i+1} - t_i$ is large enough to perceive the change triggered by the action. The invariant features $F(P)$ are not relevant for the measuring of success. We use the varying feature set $G(P)$ of previously perceived successful action executions as input to the measurement. Since we want to measure a relation of percepts between points in time, we observe the difference between the feature sets $g_n(P_{t+\Delta t}) - g_n(P_t) = d_n(t)$. The generalization of $d_n(t)$ is performed in a similar way as mentioned above. During the generalization process the expected values $E_n(t) = \{e_{n,i}(t)\}$ which correspond to cluster centers and relevances of feature channels R_n are determined. We assume, that all observed changes $d_n(t)$ for a relevant feature channel have the same course

in time for actions applied to different percepts. This has to be ensured in a normalization step where all $d_n(t)$ are mapped to a common time basis and has to be taken into account during generalization.

In the inner model for the measurement of success IM_A the expected values $E_n(t)$ are stored, if the corresponding relevance is above a threshold:

$$IM_A = \{(E_n(t)) : R_n > minrelevance\} \tag{4.9}$$

Critical in the realisation of the proposed scheme is the implementation of the feature extractors G and F. For the example of cup filling we use the problem specific features shape and colour as invariant features ($F(P) = (f_{shape}, f_{colour})$) and fill level as varying feature ($G(P) = g_{filllevel}$). Future challenges comprise the proposal of feature extraction methods, which follow the requirements formulated in this section for a broader range of problems. The proposed relevances allow to evaluate methods on their applicability for the extraction of affordances and the measurement of success for an action.

4.4 Action Representation

Our approach to generate, represent and reproduce actions makes use of Hidden Markov Models. We use three different HMM for each arm, one to encode the position of the TCP (Tool Center Point, a reference point on the hand), i.e. the hand path, with the Cartesian coordinates being represented by three-dimensional output distributions, one for the orientation of the TCP (described by three angles) and another one for the joint angle trajectories where the dimension of the output distributions is equal to the number of observed joint angles of the arm.

HMMs are used to generalize movements demonstrated to a robot multiple times [2]. Characteristic features of the perceived movement, so-called key points, are detected in a pre-processing stage and used to train the HMMs. By doing so, we avoid having a high number of states and facilitate the matching of (or between) multiple demonstrations. We use continuous HMMs and model the observations in each state with multivariate Gaussian density functions. Each HMM is trained with the key points of all demonstrations using the Baum-Welch algorithm for multiple observations. Each training sequence consists of the key points of the respective demonstration. For a given observation sequence, the Viterbi algorithm returns the optimal state sequence of the HMM with respect to that observation sequence, i.e. the sequence of states most likely to generate that observation sequence. For the reproduction of a perceived movement, key points that are common to all (or almost all) demonstrations, so-called common key points, are used. To determine the common key points across $d = 1, \ldots, D$ key point sequences $K_{d,1}, \ldots, K_{d,n(d)}$, where $n(d)$ denotes the number of key points for a demonstration d, we use the Viterbi algorithm D times to find sequences of HMM states that correspond best to these key point sequences. The common key points are determined by comparing these

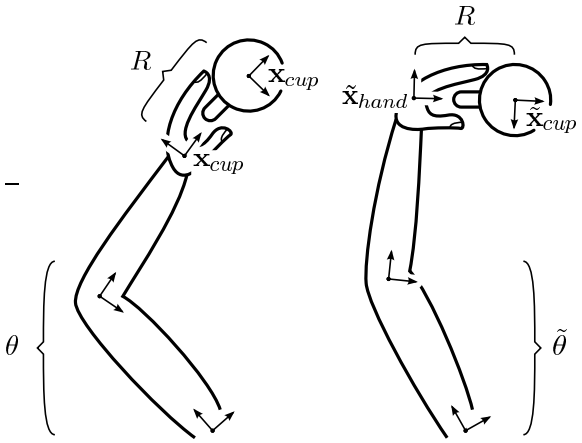


Fig. 4.3 Two arm postures grabbing a cup in the same way. It is obvious that the relation R to the object is much more important than the joint angles θ.

state sequences and selecting only those states that appear in every sequence. The actions that were considered initially are simple actions. In general, it is not possible to learn an HMM for all possible action imaginable. To treat a large set of complex actions, we will need to brake down the actions into very simple ones. These simple actions would define an alphabet based on which complex actions can be defined by concatenation.

4.5 Imitation on Objects

Learning trajectories both in the work space and in the joint space is one common approach in imitation learning approaches. Movement are often demonstrated to a robot by a human instructor, subsequently generalized (using the data from all demonstrations) and finally reproduced by the robot without trying to infer the goal of the movement. One main goal of imitation learning is to understand simple movements. Taking the example of grabbing a cup, the path has to be altered if the cup has a different position. Static learning does not fulfil these needs. In figure 4.3 one can see that for this example the exact arm posture is less important than the relation to the effected object. The joint trajectory is only a minor condition. It should not be used to calculate the hand position. Instead it could be used to solve the problem of the redundancy in computing the inverse kinematics.

In this section we present an novel way for imitation learning on objects.

4.5.1 Adaptation of Movements to the Given Situation

If an action is repeated in a different situation, the learned path has to be adjusted. Our idea is to learn paths only relative to the affected object (see Fig. 4.4). While a new trajectory is processed the linear path between $\mathbf{x}_{\text{start}}$ and $\mathbf{x}_{\text{end}}$ is calculated. For the adaptation, the system is trained with the difference $\Delta\mathbf{x}$ between the observed and the linear path. The reproduction is done by using the linear path between the new $\tilde{\mathbf{x}}_{\text{start}}$ and $\tilde{\mathbf{x}}_{\text{end}}$ and the learned difference. The result would be a similar path into a different direction.

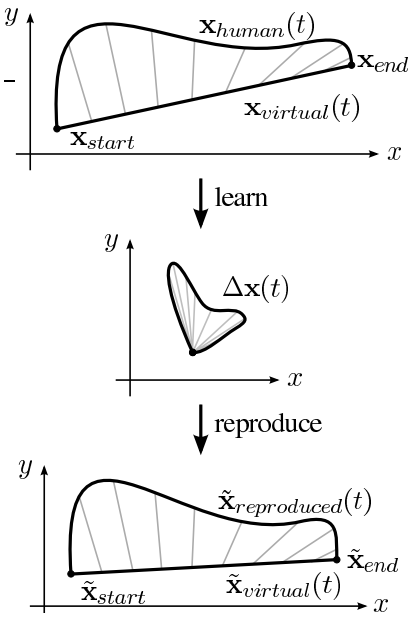


Fig. 4.4 Adaptation of an observed movement to the given new situation. Instead of the original sequence, the gray indicated difference to the linear path is learned. The normalized sequence is extracted to new parameters in the lower diagram.

Using the linear path as reference is the easiest and fastest possibility. The quality will be increased if the calculated path is already human-like. A system like the VITE model created by Hersch and Billard [10] seems to be convenient. The joint angle information of such a system would be ignored. Only the hand path would be used as a basis of the difference.

4.5.2 Generalization Across Movements

The methodology described in Section 4.4 allows us to reproduce the observed movements by a humanoid robot and build a library specifying a complete set of the desired trajectories for an observed action class. Motion capture has been used successfully to reproduce motions that may require a lot of skill and practicing, but do not need to fulfill a specific goal, such as for example dancing [15]. However, in tasks involving the manipulation of objects, it is often necessary to adapt the observed trajectories with respect to the current state of the 3-D world. It is highly unlikely that an appropriate movement would be observed a priori and included in the library. Hence it is necessary to generalize over the movements stored in the library and generate a new movement that can attain the goal of an action. With this in mind we designed a strategy to interpolate between movements stored in the library, with the goal of generating appropriate new movements that were not recorded in the data collection phase, but need to be executed to attain the goal of an action.

Let each example motion $\mathbf{M}_i,\ i = 1,\ldots,NumEx$, be given by key points $\mathbf{p}_{ij}$ at times $t_{ij},\ j = 1,\ldots,n_i,$. With each movement we also store the duration of motion T_i. Such data can be collected by the proposed motion capture system. The key points can be specified in various ways, for example as joint space postures or as end-effector poses in 3-D. We start by time normalizing the captured movements to an interval $[0,1]$. Similarly to Rose et al. [13], we encode the example trajectories using uniform cubic B-splines

$$\mathbf{M}_i(t) = \sum_{k=1}^{N} \mathbf{b}_{ik} \mathrm{B}_k(t), \tag{4.10}$$

where N is the number of spline basis functions. Linear least squares approximation can be used to approximate all trajectories with the same number of splines. The optimal number of splines N can be determined experimentally, but more sophisticated methods are also possible (see for example [15]).

In the following we propose a method for the generation of goal-directed arm reaching movements. The method is, however, much more general and we discuss how to apply it to other actions at the end of the section.

In the case of arm reaching movements, the start point $\mathbf{x}_{\text{start}}$ and the end point $\mathbf{x}_{\text{end}}$ of the end-effector in Cartesian space are very important. Therefore it makes sense to represent goal-directed arm reaching movements as end-effector trajectories $\mathbf{M}_i$ in a 3-D space. Another important factor is the duration of movement T. We use this information as a query point $\mathbf{q}$ into the database when generating new movements from example movements

$$\mathbf{q} = \left[\mathbf{x}_{\text{start}}^T, \mathbf{x}_{\text{end}}^T, T\right]^T. \tag{4.11}$$

Given a query point $\mathbf{q}$, we would like to determine movement $\mathbf{M}(\mathbf{q};t)$ defined as

$$\mathbf{M}(\mathbf{q};t) = \sum_{k=1}^{N} \mathbf{b}_k(\mathbf{q}) \mathrm{B}_k(t), \tag{4.12}$$

which starts at $\mathbf{x}_{\text{start}}$ and ends at $\mathbf{x}_{\text{end}}$. For each of the example trajectories $\mathbf{M}_i$, we calculate its start point $\mathbf{x}_{i,\text{start}}$ and its end point $\mathbf{x}_{i,\text{end}}$. We apply locally weighted regression [4] to generate new reaching movements. This results in the following optimization problem

$$\min_{\mathbf{b}} \mathrm{C}(\mathbf{q}) = \sum_{i=1}^{NumEx} \mathrm{L}\left(\mathbf{M}_i, \mathbf{M}(\mathbf{q})\right) \mathrm{K}\left(\mathrm{d}_i(\mathbf{q}_i, \mathbf{q})\right), \tag{4.13}$$

subject to

$$\mathbf{M}(\mathbf{q};0) = \mathbf{x}_{\text{start}},\ \mathbf{M}(\mathbf{q};1) = \mathbf{x}_{\text{end}}. \tag{4.14}$$

Here L is the loss function, K is the weighting function, and d_i are the distance functions between the query point and the data points $\mathbf{q}_i = \left[\mathbf{x}_{i,\text{start}}^T, \mathbf{x}_{i,\text{end}}^T, T_i\right]^T$. The unknown parameters we minimize over are $\mathbf{b} = \{\mathbf{b}_k(q)\}$.

We define the loss function by the Euclidean distance between the spline coefficients

$$\mathrm{L}\left((\mathbf{M}_i, \mathbf{M}(\mathbf{q})\right) = \sum_{k=1}^{N} \|\mathbf{b}_{ik} - \mathbf{b}_k(\mathbf{q})\|^2. \tag{4.15}$$

Distance function d_i is given as the weighted Euclidean distance between the data points, i. e.

$$\mathrm{d}(\mathbf{q}, \mathbf{q}_i) = \frac{1}{a_i}\|\mathbf{q} - \mathbf{q}_i\|,\ a_i > 0. \tag{4.16}$$

There are many possibilities to define the weighting function K [4]. We chose the tricube kernel

$$\mathrm{K}(d) = \begin{cases} (1 - |d|^3)^3 & \text{if} |d| < 1 \\ 0 & \text{otherwise} \end{cases}. \tag{4.17}$$

This kernel has finite extent and continuous first and second derivative. Combined with distance (4.16), these two functions determine how much influence each of the movements $\mathbf{M}_i$ has as the query point $\mathbf{q}$ moves away from the data point $\mathbf{q}_i$. It is best to select a_i so that there is some overlap between the neighboring query points. One possibility is

$$a_i = \min_j \|\mathbf{q}_i - \mathbf{q}_j\| \tag{4.18}$$

By selecting a_i in this way we ensure that the influence of neighboring movements in (4.13) overlaps, that $\mathbf{M}(\mathbf{q}_i) = \mathbf{M}_i$, and that as the query point transitions from one data point to the other, the generated movement also transitions between movements associated with data points.

Our choice of L, K, and d_i makes the optimization problem (4.13) a weighted linear least-squares problem with equality constraints, which can be solved using standard approaches. In this way we can generate new arm reaching movement that were not observed in the data collection phase. It can be clear from the above explanation that the method is not limited to arm reaching movements. For a given collection of movements, it is only necessary to specify reasonable query points

and impose any constraints that are necessary to achieve the goal of an action. The proposed movement interpolation technique can then be applied.

4.6 Discussion and Conclusions

As the goal of an action changes it is necessary to adapt the captured movements to new situations. Sometimes it is possible to attain the goal of an action by moving and scaling the desired trajectories in space and time. For this purpose we propose a method for adaptation of movements to the given situation (Sec. 4.5.1). In some situations, however, the movement changes more substantially depending on the goal of an action, e.g. the amplitude could increase or the frequency of oscillation could change. Such modification cannot be captured by the first approach, therefore we introduce a strategy to interpolate between movements in stored in the movement library (Sec. 4.5.2).

Future work will concentrate on both the evaluation of the proposed methods for the generation of actions and a complete implementation of a real-time imitation learning system using the active humanoid head.

Acknowledgements The work described in this paper was partially conducted within the EU Cognitive Systems project PACO-PLUS (FP6-2004-IST-4-027657) and funded by the European Commission and the German Humanoid Research project SFB588 funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft).

References

1. Akachi, K., Kaneko, K., Kanehira, N., Ota, S., Miyamori, G., Hirata, M., Kajita, S., Kanehiro, F.: Development of humanoid robot HRP-3. In: IEEE/RAS International Conference on Humanoid Robots (2005)
2. Asfour, T., Gyarfas, F., Azad, P., Dillmann, R.: Imitation learning of dual-arm manipulation tasks in humanoid robots. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids 2006). Genoa, Italy (2006)
3. Asfour, T., Regenstein, K., Azad, P., Schröder, J., Bierbaum, A., Vahrenkamp, N., Dillmann, R.: ARMAR-III: An integrated humanoid platform for sensory-motor control. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids 2006). Genoa, Italy (2006)
4. Atkeson, C.G., Moore, A., Schaal, S.: Locally weighted learning. AI Review **11**, 11–73 (1997)
5. Azad, P., Ude, A., Asfour, T., Cheng, G., Dillmann, R.: Image-based Markerless 3D Human Motion Capture using Multiple Cues. In: International Workshop on Vision Based Human-Robot Interaction. Palermo, Italy (2006)
6. Azad, P., Ude, A., Dillmann, R., Cheng, G.: A Full Body Human Motion Capture System using Particle Filtering and On-The-Fly Edge Detection. In: International Conference on Humanoid Robots (Humanoids). Santa Monica, USA (2004)
7. Cheng, G., Hyon, S., Morimoto, J., Ude, A., Jacobsen, S.: CB: a humanoid research platform for exploring neuroscience. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids 2006). Genoa, Italy (2006)

8. The PACO-PLUS project: Perception, Action and Cognition through Learning of Object-Action Complexes. European Cognitive Systems project, www.paco-plus.org.
9. Gibson, J.: The ecological approach to visual perception. Houghton Mifflin, Boston (1979)
10. Hersch, M., Billard, A.: A biologically-inspired model of reaching movements. In: IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics (2006)
11. I.W. Park J.Y. Kim, J.L., Oh, J.: Mechanical design of humanoid robot platform KHR-3 (kaist humanoid robot-3: Hubo). In: IEEE/RAS International Conference on Humanoid Robots (2005)
12. Nishiwaki, K., Sugihara, T., Kagami, S., Kanehiro, F., Inaba, M., Inoue, H.: Design and development of research platform for perception-action integration in humanoid robots: H6. In: IEEE/RSJ International. Conference on Intelligent Robots and Systems, pp. 1559–1564 (2000)
13. Rose, C., Bodenheimer, B., Cohen, M.F.: Verbs and adverbs: Multidimensional motion interpolation using radial basis functions. IEEE Computer Graphics and Applications **18**(5) (1998)
14. Sakagami, S., Watanabe, T., Aoyama, C., Matsunage, S., Higaki, N., Fujimura, K.: The intelligent ASIMO: System overview and integration. In: IEEE/RSJ International. Conference on Intelligent Robots and Systems, pp. 2478–2483 (2002)
15. Ude, A., Atkeson, C.G., Riley, M.: Programming full-body movements for humanoid robots by observation. Robotics and Autonomous Systems **47**(2-3), 93–108 (2004)

Chapter 5
Wald's Sequential Analysis for Time-constrained Vision Problems

Jiří Matas and Jan Šochman

5.1 Introduction

In many decision problems in computer vision, both classification errors and time to decision characterise the quality of an algorithmic solution. This is especially true for applications of vision to robotics where real-time response is typically required.

Time-constrained classification, detection and matching problems can be often formalised in the framework of sequential decision-making. We show how to derive quasi-optimal time-constrained solutions for three different vision problems by applying Wald's sequential analysis. In particular, we adapt and generalise Wald's sequential probability ratio test (SPRT) and apply it to the three vision problems: (i) face detection, (ii) real-time detection of distinguished regions (interest points) and (iii) establishing correspondences by the RANSAC algorithm with application e.g. in SLAM, 3D reconstruction and object recognition.

In the face detection problem, we are interested in learning the fastest detector satisfying constraints on false positive and false negative rates. We solve the problem by WaldBoost [15], a combination of Wald's sequential probability ratio test and AdaBoost learning [2]. The solution can be viewed as a principled way to build a close-to-optimal "cascade of classifiers" [22]. Naturally, the approach is applicable to other classes of objects.

In the interest point detection emulation, we show how a fast (real-time) implementation of the Hessian-Laplace detector [9] is obtained by WaldBoost [16]. The emulated detector provides a training set of positive and negative examples of interest points. WaldBoost finds an approximation to the detector output in terms of a linear combination of efficiently computable filter responses. The trained detector output differs from the "teacher" detector only at a small, controllable fraction of locations and yet is significantly faster.

Center for Machine Perception, Dept. of Cybernetics, Faculty of Elec. Eng. Czech Technical University in Prague, Karlovo nám. 13, 121 35 Prague, Czech Rep. e-mail: \{matas,sochmj1\}@cmp.felk.cvut.cz

RANSAC (**RAN**dom **SA**mple **C**onsensus) is a robust estimator that has been used in many computer vision algorithms e.g. for short and wide baseline stereo matching and structure and motion estimation. In the time-optimal RANSAC , we derive the fastest randomised strategy for hypothesis verification satisfying a constraint on the probability that the returned solution is correct. The optimal strategy is found again with the help of Wald's SPRT test.

The rest of the paper is structured as follows. First, we formally define the time-constrained detection problem and present the relevant parts of Wald's theory in Section 5.2. Next, the WaldBoost algorithm for sequential decision making is presented in Section 5.3. A face detector trained by the WaldBoost procedure is presented in Section 5.4. A similar methodology is applied in Section 5.5 to the problem of fast approximation of a repeatable interest point detector. In Section 5.6, Wald's SPRT test is combined with RANSAC and a very fast method for robust estimation of geometric relations and model parameters in general is obtained.

5.2 The Two-class Sequential Decision-making Problem

Let x be an object belonging to one of two classes $\{-1,+1\}$, and let an ordering on the set of measurements $\{x_1,\ldots,x_m\}$ on x be given. A sequential decision strategy is a set of decision functions $S=\{S_1,\ldots,S_m\}$, where $S_i:\{x_1,\ldots,x_i\}\rightarrow\{-1,+1,\sharp\}$. The strategy S takes the measurements one at a time and at time i makes a decision based on S_i. The '$\sharp$' sign stands for a "continue" (do not decide yet) decision[1]. If a decision is '$\sharp$', x_{i+1} is measured and S_{i+1} is evaluated. Otherwise, the output of S is the class returned by S_i.

In other words, a sequential strategy takes one measurement at a time. After the i-th measurement, it either terminates by classifying the object to one of the classes $+1$ or -1, or continues by taking the next measurement.

In two-class classification problems, errors of two kinds can be made by strategy S. Let us denote by α_S the probability of error of the first kind (x belongs to $+1$ but is classified as -1) and by β_S the probability of error of the second kind (x belongs to -1 but is classified as $+1$).

A sequential strategy S is characterised by its error rates α_S and β_S and its average evaluation time

$$\bar{T}_S = E(T_S(x)) \,, \tag{5.1}$$

where the expectation E is over $p(x)$ and $T_S(x)$ is the expected evaluation time (or time-to-decision) for strategy

$$T_S(x) = \arg\min_i (S_i(x) \neq \sharp) \,. \tag{5.2}$$

[1] In pattern recognition, this is called "the rejection option"

An optimal strategy for the sequential decision making problem is then defined as

$$S^* = \arg\min_S \bar{T}_S \tag{5.3}$$

$$\text{s.t.} \quad \beta_S \leq \beta \,, \tag{5.4}$$

$$\alpha_S \leq \alpha \tag{5.5}$$

for specified α and β.

The sequential decision-making theory was developed by Wald [23], who proved that the solution of the optimisation problem (5.3) is the *Sequential Probability Ratio Test (SPRT)*.

5.2.1 Sequential Probability Ratio Test

Let x be an object characterised by its hidden state (class) $y \in \{-1,+1\}$. This hidden state is not observable and has to be determined based on successive measurements $x_1, x_2, \ldots$. Let the joint conditional density $p(x_1,\ldots,x_m|y=c)$ of the measurements $x_1,\ldots,x_m$ be known for $c \in \{-1,+1\}$ and for all m.

SPRT is a sequential strategy S^*, which is defined as:

$$S^*_m = \begin{cases} +1, & R_m \leq B \\ -1, & R_m \geq A \\ \sharp, & B < R_m < A \end{cases} \tag{5.6}$$

where R_m is the likelihood ratio

$$R_m = \frac{p(x_1,\ldots,x_m|y=-1)}{p(x_1,\ldots,x_m|y=+1)} \,. \tag{5.7}$$

The constants A and B are set according to the required error of the first kind α and error of the second kind β. Optimal A and B are difficult to compute in practice, but tight bounds are easily derived.

Theorem 5.1 (Wald). *A is upper bounded by $(1-\beta)/\alpha$ and B is lower bounded by $\beta/(1-\alpha)$.*

Proof. For each sample $\{x_1,\ldots,x_m\}$, for which SPRT returns the class -1 we get from (5.6)

$$p(x_1,\ldots,x_m|y=-1) \geq A \cdot p(x_1,\ldots,x_m|y=+1) \,. \tag{5.8}$$

Since this holds for all samples classified to the class -1

$$P\{S^*=-1|y=-1\} \geq A \cdot P\{S^*=-1|y=+1\} \,. \tag{5.9}$$

The term on the left is the probability of correct classification of an object from the class -1 and is therefore $1-\beta$. The term on the right is the probability of incorrect classification of an object to the class $+1$, and is equal to α. After this substitution and rearranging, we get the upper bound on A. Repeating this derivation with the samples classified by SPRT to the class $+1$ the lower bound on B is derived. □

In practical applications, Wald suggests to set the thresholds A and B to their upper and lower bound respectively

$$A' = \frac{1-\beta}{\alpha}, \qquad B' = \frac{\beta}{1-\alpha} \; . \tag{5.10}$$

The effect of this approximation on the test error rates was summarised by Wald in the following theorem.

Theorem 5.2 (Wald). *When A' and B' defined in (5.10) are used instead of the optimal A and B, the real error probabilities of the test change to α' and β' for which*

$$\alpha' + \beta' \leq \alpha + \beta \; . \tag{5.11}$$

Proof. From Theorem 5.1 it follows that

$$\frac{\alpha'}{1-\beta'} \leq \frac{1}{A'} = \frac{\alpha}{1-\beta} \; , \text{and} \tag{5.12}$$

$$\frac{\beta'}{1-\alpha'} \leq \frac{1}{B'} = \frac{\beta}{1-\alpha} \; . \tag{5.13}$$

Multiplying the first inequality by $(1-\beta')(1-\beta)$ and the second by $(1-\alpha')(1-\alpha)$ and summing both inequalities, the result follows. □

This result shows that at most one of the probabilities α and β can be increased and the other has to be decreased by the approximation.

Theorem 5.3 (Wald). *SPRT (with optimal A and B) is an optimal sequential test in a sense of the optimisation problem (5.3).*

Proof. The proof is complex. We refer interested reader to [23]. □

Wald analysed SPRT behaviour when the upper bound A' and B' is used instead of the optimal A and B. He showed that the effect on the speed of evaluation is negligible.

However, Wald did not consider the problem of optimal ordering of measurements, since in all of his applications the measurements were i.i.d and the order did not matter. Secondly, Wald was not concerned with the problem of estimating (5.7) from a training set, since in the i.i.d case

$$p(x_1, \ldots, x_m | y = c) = \prod_{i=1}^{m} p(x_i | y = c) \tag{5.14}$$

and thus R_m can be computed incrementally from a one dimensional probability density function.

5.3 WaldBoost

For dependent measurements, which is the case in many computer vision tasks, SPRT can still be used if the likelihood ratio can be estimated. However, that usually encompasses many-dimensional density estimation, which becomes infeasible even for a moderate number of measurements.

In [15], it was suggested to use the AdaBoost algorithm for measurement selection and ordering and we review the relevant results in this section. The section is structured as follows. First, the AdaBoost learning algorithm is reviewed in Section 5.3.1. In Section 5.3.2, an approximation for the likelihood ratio estimation is proposed for such (statistically dependent) measurements. The WaldBoost algorithm combining SPRT and AdaBoost is described in Section 5.3.3.

5.3.1 AdaBoost

The AdaBoost algorithm [13, 2][2] is a greedy learning algorithm. Given a labelled training set $\mathscr{T} = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, where $y_i \in \{-1, +1\}$, and a class of weak classifiers $\mathscr{H}$, the AdaBoost produces a classifier of the form

$$H_T(x) = \sum_{t=1}^{T} h^{(t)}(x) \,, \tag{5.15}$$

where $h^{(t)} \in \mathscr{H}$ and usually $T \ll |\mathscr{H}|$. Weak classifiers can be of an arbitrary complexity but are often chosen to be very simple. The final classifier then boosts their performance by combining them into a strong classifier H_T.

The outputs of selected weak classifiers will be taken as measurements used in SPRT.

In AdaBoost training, an upper bound on the training error is minimised instead of the error itself. The upper bound has an exponential form

$$J(H_T) = \sum_i e^{-y_i H_T(x_i)} = \sum_i e^{-y_i \sum_{t=1}^{T} h^{(t)}(x_i)} \,. \tag{5.16}$$

Training of the strong classifier runs in a loop. One weak classifier is selected and added to the sum at each loop cycle. A selected weak classifier is the one which minimises the exponential loss function (5.16)

[2] The real valued version is used.

$$h_{T+1} = \arg\min_h J(H_T + h)\,, \tag{5.17}$$

It has been shown [13, 3] that the weak classifier minimising (5.17) is

$$h_{T+1} = \frac{1}{2}\log\frac{P(y=+1|x, w^{(T)}(x,y))}{P(y=-1|x, w^{(T)}(x,y))}\,, \tag{5.18}$$

where $w^{(T)}(x,y) = e^{-yH_T(x)}$ is a weight of a sample (x,y) at cycle T.

As shown in [3], choosing a weak classifier according to (5.18) in each cycle of the AdaBoost learning converges asymptotically to

$$\lim_{T\to\infty} H_T(x) = \tilde{H}(x) = \frac{1}{2}\log\frac{P(y=+1|x)}{P(y=-1|x)}\,. \tag{5.19}$$

This result will be used in the following section.

5.3.2 Likelihood Ratio Estimation with AdaBoost

The likelihood ratio (5.7) computed on the outputs of weak classifiers found by AdaBoost has the form

$$R_t(x) = \frac{p(h^{(1)}(x),...,h^{(t)}(x)|y=-1)}{p(h^{(1)}(x),...,h^{(t)}(x)|y=+1)}\,, \tag{5.20}$$

where the outputs of the weak classifiers cannot be treated as statistically independent.

Since the computation of $R_t(x)$ involves a high dimensional density estimation, it is approximated so that this task simplifies to a one dimensional likelihood ratio estimation. The t-dimensional space is projected into a one dimensional space by the strong classifier function H_t (see equation (5.15)). Hence, all points $(h^{(1)},...,h^{(t)})$ are projected to a value given by the sum of their individual coordinates. Using this projection, the ratio (5.20) is estimated by

$$\hat{R}_t(x) = \frac{p(H_t(x)|y=-1)}{p(H_t(x)|y=+1)}\,. \tag{5.21}$$

Justification of this approximation can be seen from equation (5.19) which can be reformulated using Bayes formula to the form

$$\tilde{H}(x) = -\frac{1}{2}\log R(x) + \frac{1}{2}\log\frac{P(+1)}{P(-1)}\,. \tag{5.22}$$

Thus, in an asymptotic case, the strong classifier is related directly to the likelihood ratio. In particular, it maps all points with the same likelihood ratio to the same value. Consequently, it makes sense to estimate the likelihood ratio for every value

Algorithm 1 WaldBoost Learning

Input: $(x_1, y_1), ..., (x_l, y_l)$; $x_i \in \mathcal{X}, y_i \in \{-1, 1\}$, desired final false negative rate α and false positive rate β.

Initialise weights $w_1(x_i, y_i) = 1/l$
Set $A = (1-\beta)/\alpha$ and $B = \beta/(1-\alpha)$
For $t = 1, ..., T$

1. Choose h_t according to equation (5.18),
2. Estimate the likelihood ratio R_t according to Eq. (5.21)
3. Find thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$
4. Throw away samples from training set for which $H_t \geq \theta_B^{(t)}$ or $H_t \leq \theta_A^{(t)}$
5. Sample new data into the training set

end
Output: Strong classifier H_T and thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$.

of $\tilde{H}(x)$ and the estimate (5.21) is then exactly equal to $R(x)$. For a non-asymptotic case we take an assumption that the same relation holds between $H_t(x)$ and $\hat{R}_t(x)$ as well.

Several methods can be used to estimate $\hat{R}_t(x)$, like logistic regression for direct ratio estimation or the class densities can be estimated instead and the ratio can be calculated based on these density estimates. The method used in our implementation is described in Section 5.3.6.

Having the likelihood ratio estimate $\hat{R}_t$, the SPRT can be applied directly. Assuming monotonicity of the likelihood ratio, only two thresholds are needed on H_t values. These two thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$, each one corresponding to one of the conditions in (5.6), are determined uniquely by the bounds A and B.

5.3.3 The WaldBoost Algorithm

The analysis given above allows us to define the WaldBoost algorithm. The WaldBoost learning phase is summarised in Algorithm 1 and described in Section 5.3.4. A WaldBoost classifier evaluation is explained in next Section 5.3.5 and summarised in Algorithm 2. Finally, a discussion of the algorithm details is given in Section 5.3.6.

5.3.4 Learning

WaldBoost requires, in addition to the usual AdaBoost initialisation by a labelled training set, two additional parameters specifying desired final false negative rate

Algorithm 2 WaldBoost Classification

Given: H_T, $\theta_A^{(t)}$, $\theta_B^{(t)}$, γ.
Input: a classified object x.
For $t = 1, \ldots, T$ *(SPRT execution)*
  If $H_t(x) \geq \theta_B^{(t)}$, classify x to the class $+1$ and terminate
  If $H_t(x) \leq \theta_A^{(t)}$, classify x to the class -1 and terminate
end
If $H_T(x) > \gamma$, classify x as $+1$. Classify x as -1 otherwise.

α and false positive rate β of the output classifier. These rates are used to compute the two thresholds A and B according to equation (5.10). The training runs in a loop, where the first step is a standard AdaBoost search for the best weak classifier (Step 1), as described in Section 5.3.1. Then, the likelihood ratio is estimated (Step 2) and the thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$ are found (Step 3), as described in Section 5.3.2. Based on the thresholds, the training set is pruned (Step 4). Finally, a new training set is created by a random sampling over the samples, which have not been decided yet (Step 5). The steps 4 and 5 are discussed in more detail below.

Pruning of the training set (Step 4) is necessary to keep the final false negative and false positive rate under the specified values α and β. SPRT requires the likelihood ratio R_m to be estimated only over the samples which have passed undecided through all pruning steps up to the current learning cycle. The samples already classified as positive or negative class samples are removed from the training set.

For the **new data collection** (Step 5), a random sampling is performed over those data samples, which have not been assigned to any class yet. The number of newly sampled samples depends on the previous pruning step.

These two steps are similar to the bootstrapping technique [17] except that the samples are not collected only but thrown away in Step 4 as well. Another close approach is the cascade building procedure [22] with the substantial difference that the pruning and new data collection in the WaldBoost learning are run after every weak classifier is trained.

5.3.5 Classification

The structure of the WaldBoost classifier is summarised in Algorithm 2. The classification executes the SPRT test on the trained strong classifier H_T with thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$. If H_t exceeds the respective threshold, a decision is made. Otherwise, next weak classifier is taken. If a decision is not made within T cycles, the input is classified by thresholding H_T on a value γ specified by the user.

5.3.6 Algorithm Details

Two parts of WaldBoost have not been fully specified. First, the exact likelihood ratio $R_t(x)$ is not know. Only its approximation $\hat{R}_t$ is used. Although this estimate is approaching the correct value with onward training, wrong and irreversible decisions can be made easily in early evaluation cycles. Hence, an inaccurate likelihood ratio estimation can affect performance of the whole classifier.

To reduce this effect, we estimate the likelihood ratio in the following way. The densities $p(H_t(x)|y=+1)$ and $p(H_t(x)|y=-1)$ are estimated not from the training set directly, but from an independent validation set to get an unbiased estimate. Moreover, the estimation uses the Parzen windows technique with the kernel width set according to the *oversmoothing rule* for the Normal kernel [14]

$$h_{OS} = 1.144\sigma n^{-1/5}, \tag{5.23}$$

where σ is the sample standard deviation and n the number of samples. The h_{OS} is an upper bound on an optimal kernel width and thus, the density estimate is smoother than necessary for an optimal density estimation. Due to this conservative strategy, the evaluation time can be prolonged but the danger of wrong and irreversible decisions is reduced.

Second important aspect of the WaldBoost learning is the stopping criterion. For practical reasons, only limited number of weak classifiers is found, which implies truncation of the sequential test during strong classifier evaluation. Wald [23] studies the effect of truncation of the sequential test procedure, however, his derivations hold only for cases where independent identically distributed measurements are taken. For that case, he suggests to threshold the final likelihood ratio at zero and analyses the effect of such method on the false negative and false positive rates of the test.

In our implementation, the final threshold is left unspecified. It can be used to regulate a false positive and a false negative rate in the application. It is also used in a ROC curve generation in the experiment section.

Generally, the more training cycles are allowed, the more precise is the likelihood ratio estimation and the better is the separation of the classes, but the slower is the classifier evaluation. For an analysis of the effect of truncation on WaldBoost performance see Section 5.4.1.

5.4 WaldBoost Applied to Face Detection

The WaldBoost algorithm is applicable to any time-constrained classification task. In this section, we show how to apply WaldBoost to face detection. The face detection problem has two specific features: (i) highly unbalanced class sizes and complexities, and (ii) particular requirements on error of the first and the second kind.

The object class size (the face class in our case) is usually relatively small and compact compared to the non-object class. The object class samples are difficult to collect and too much pruning can reduce the size of the object training set irreversibly. The non-object class, on the other hand, consists of all images except the images of an object itself. Such a huge and complex class cannot be represented by a small training set sufficiently. So, the goal of the learning is to explore the largest possible subspace of the non-object class while keeping most of the object samples during the learning process.

The second specific of the object detection is that error of the first kind (missed object) is considered as more serious than error of the second kind (falsely detected object). An ideal way of training a classifier would be to require a zero false negative rate and the smallest possible false positive rate.

Having the above specifics in mind, WaldBoost can be initialised in the following way. Let the required false positive rate β is set to zero and the required false negative rate α to some small constant (note the inverse initialisation compared to the above reasoning). In this setting, equations (5.10) reduce to

$$A = \frac{1-0}{\alpha} = \frac{1}{\alpha}\,, \qquad B = \frac{0}{1-\alpha} = 0 \tag{5.24}$$

and the SPRT strategy (5.6) becomes

$$S_m^* = \begin{cases} +1, & R_m \leq 0 \\ -1, & R_m \geq 1/\alpha \\ \sharp, & 0 < R_m < 1/\alpha \end{cases} \tag{5.25}$$

Since R_m is always positive, the algorithm will never classify a sample to the object class. The only allowed decision is the classification to the non-object class. Hence, the learning process will never prune the object part of the training set while pruning the non-object part. Such initialisation thus leads to an exploration of the non-object class (by pruning and new sample collection) while working with a small and unchanging object training set. Moreover, the detection rate of the final classifier is assured to be $1-\alpha$ while the false positive rate is progressively reduced by each training cycle.

5.4.1 Experiments

The proposed WaldBoost algorithm was tested on the frontal face detection problem. The classifier was trained on 6350 face images divided into a training and a validation set. In each training cycle, the non-face part of the training and the validation set included 5000 non-face samples sampled randomly from a pool of subwindows from more than 3000 non-face images. The weak classifier set $\mathscr{H}$ used in training is the same as in [22] but WaldBoost is not feature-specific and any other weak classifiers can be used. Unlike [22], the weak classifiers are real valued (de-

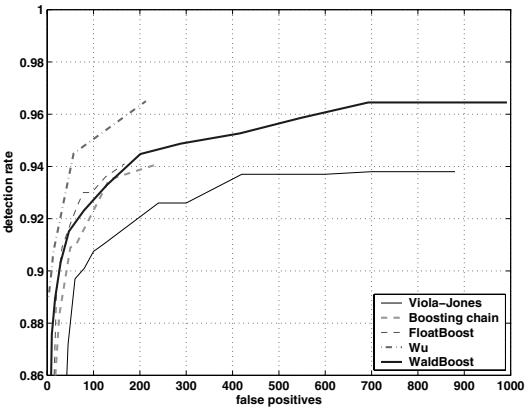


Fig. 5.1 ROC curve comparison of the WaldBoost algorithm with the state-of-the-art methods.

fined by equation (5.18)) and implemented as in [6]. The allowed false negative rate α was set to $5 \cdot 10^{-4}$. The training was run with $T = 600$, i.e. till the strong classifier consisted of 600 weak classifiers.

The WaldBoost classifier was tested on the MIT+CMU dataset [12] consisting of 130 images containing 507 labelled faces. A direct comparison with the methods reported in literature is difficult since they use different subsets of this dataset with the most difficult faces removed (about 5% in [6, 25]!). Nevertheless, we tested the WaldBoost classifier on both full and reduced test sets with similar results, so we report the results on the full dataset and plot them in one graph with the other methods (see Figure 5.1). However, the results of the other methods are not necessarily mutually comparable.

The speed and the error rates of a WaldBoost classifier are influenced by the classifier length. To examine this effect, four classifiers of different lengths (300, 400, 500 and 600 weak classifiers) were compared. The average evaluation time $\bar{T}_S$ (for definition see (5.1)) for these four classifiers is reported in Table 5.1. As expected, the average evaluation time decreases when less weak classifiers are used. However, shortening of the classifier affects the detection rates as well. The ROC curves for the four classifiers are depicted in Figure 5.2. Detection rates are comparable for the classifiers consisting of 400, 500 and 600 weak classifiers but the detection rate drops significantly when only 300 weak classifiers are used. Thus, using the classifier consisting of 400 weak classifiers only may be preferred for its faster evaluation. However, further reducing the classifier length leads to a substantial detection results degradation.

For a comparison of the WaldBoost classifier length with the other methods see Table 5.2. From the compared methods, the WaldBoost classifier needs the least number of weak classifiers, or in other words it produces the most compact classifier.

The bottom row of Table 5.2 shows the average evaluation times to decision $\bar{T}_S$ (sometimes referred to as the average number of weak classifiers evaluated) for the

#wc	600	500	400	300
$\bar{T}_S$	13.92	12.46	10.84	9.57

Table 5.1 Speed for different length WaldBoost classifiers.

Method	WB	VJ[22]	Li[6]	Xiao[25]	Wu[24]
#wc	400	4297	2546	700	756
$\bar{T}_S$	10.84	8	(18.9)	18.1	N/A

Table 5.2 The number of weak classifiers used and a speed comparison with the state-of-the-art methods. The parentheses around $\bar{T}_S$ of Li's method indicate that this result was not reported by the authors but in [25].

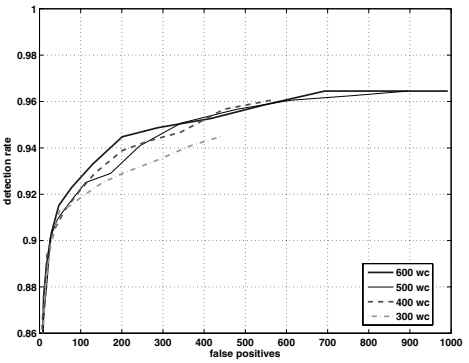


Fig. 5.2 The effect of reducing the number of weak classifiers in WaldBoost classifier on the detection rate.

compared methods. The WaldBoost learning results in the fastest classifier among the compared methods except for the Viola-Jones method which, despite its high speed, gains significantly worse detection results.

To conclude the experiments, the WaldBoost algorithm applied to the face detection problem proved its near optimality in the number of measurements needed for a reliable classification. The detection rates reached by the proposed algorithm are comparable to the state-of-the-art methods. The only method outperforming the proposed algorithm in the quality of detection is the "nesting-structured cascade" approach by Wu [24]. This can be caused by different features used, different subset of the MIT+CMU dataset used or any other implementation details.

5.5 WaldBoost Trained Fast Interest Region Detection

Learning a sequential classifier implementing a face detector as described in Section 5.3 can be viewed as a process of a fast minimum error approximation of the face detector. If suitable computation elements are available, many other binary functions can be approximated in the same way.

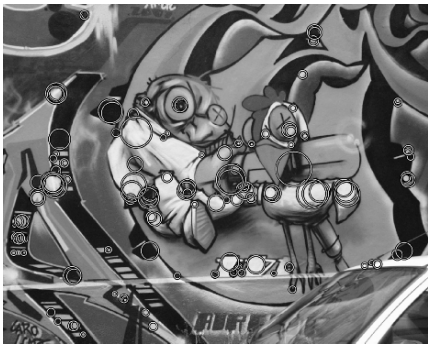


Fig. 5.3 The strongest responses of Mikolajczyk Hessian-Laplace region detector (threshold 3500).

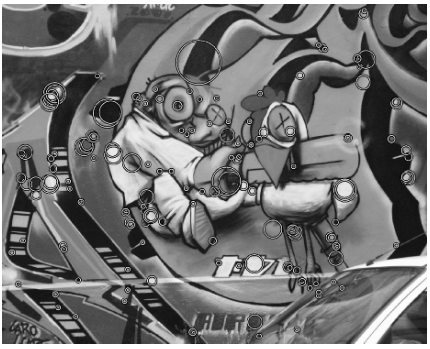


Fig. 5.4 The strongest responses of WaldBoost Hessian-Laplace region detector (same number of responses as in Figure 5.3). $\bar{T}_S = 2.07$.

This section shows how to train a sequential detector approximating the behaviour of an interest region detector – the non-affine Hessian-Laplace detector of Mikolajczyk [9]. The detector has been shown to be widely useful in applications like wide baseline matching or image retrieval.

5.5.1 Hessian-Laplace WaldBoost Classifier

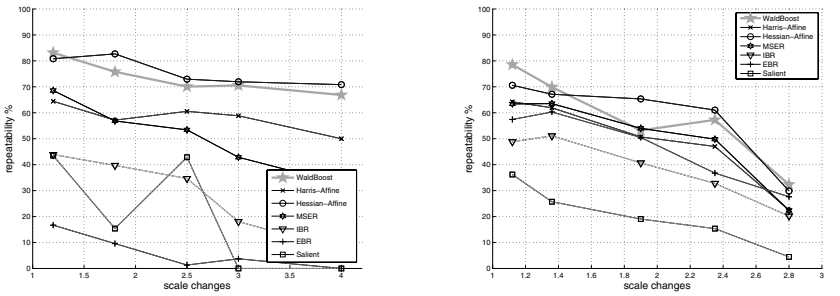
In the task of Hessian-Laplace region detector[3] approximation the positive examples correspond to the regions found by the detector and all other regions in the image are used as negative examples. The approximation error is determined by the agreement of the detectors outputs.

An example of output of a non-affine Hessian-Laplace detector is shown in the Figure 5.3. Only the strongest responses corresponding to threshold 3500 are shown. Training a WaldBoost classifier approximating the behaviour of the Hessian-Laplace detector entails several important differences in the training settings compared to the face detection.

First, positive examples are collected as an output of *rotationally invariant* Hessian-Laplace detector. To mimic this quality, the training set has to include rotated versions of positive examples. Nevertheless, the Haar-like filters are inherently axis parallel and thus the final rotation invariance will be weakened.

An important property of the interest regions detection task is that the *positive examples are very easy to collect*. Running the original detector on any image gives a new set of positive examples. The situation is similar to the problem of very huge negative examples set in the face detection problem. To process as many positive

[3] Available from `http://www.robots.ox.ac.uk/~vgg/research/affine/`



(a) (b)

Fig. 5.5 Repeatability score of the WaldBoost detector compared to the state-of-the-art methods for **(a)** Bark sequence **(b)** Boat sequence. Overlap 40%, norm. size = 30 pixels.

examples during training as possible, the positive examples set can be bootstrapped as well (i.e. β is set to a non-zero value).

Another difference is that there are *no images without positive samples* for negative examples collection by random sampling. Negative examples are not taken from an indifference region in the vicinity of a positive sample.

Finally, *the positive and negative classes are highly overlapping*. The interest region detectors have usually one parameter regulating the amount of returned positive samples. However, the more regions are returned, the less reliable the regions are and thus changing this parameter, the difficulty of the training set is controlled.

Another consequence of positive and negative class overlapping is that the WaldBoost classifier will give responses on many positions and scales. One way of removing the less trusty detections is to threshold the final classifier response. However, a better option is to set α to a higher value and let the training to concentrate on the most representative part of the positive class. This leads to much faster classifier, since the less trusty detections are decided earlier and do not need full classifier evaluation.

5.5.2 Experiments

The Hessian-Laplace detector has been approximated by a sequential WaldBoost classifier consisting of 20 weak classifiers (see [16] for more details). The strongest responses of the WaldBoost classifier are shown in Figure 5.4. Note that the detections are not exactly the same as the strongest responses of the Hessian-Laplace detector (Figure 5.3). The WaldBoost training does not retain the quality ordering of the detections of the original detector. Nevertheless, similar and sometimes the same structures are detected.

To evaluate the detector, the same tests as in [10] have been run to test the repeatability rate. The result on Bark and Boat sequences is shown in Figure 5.5. The

sequences contain scale and rotation changes. The repeatability of the WaldBoost detector is similar to the *affine* version of the Hessian-Laplace detector. The important property of the trained detector is its speed of 2 weak classifiers evaluated per window on average. The WaldBoost detector thus gains the speed of the original manually tuned algorithm with 1.3s per 850×680 image.

5.6 Robust Estimation of Model Parameters - RANSAC with Optimal Sequential Verification

RANSAC (**RAN**dom **SA**mple **C**onsensus) is a widely used robust estimator that has become a de facto standard in the field of computer vision. RANSAC has been applied to many vision problems: short baseline stereo [20, 19], wide baseline stereo matching, motion segmentation [20], mosaicing, detection of geometric primitives , robust eigenimage matching, structure and motion estimation [11, 18], object recognition and elsewhere.

In this section, we show how RANSAC speed can be improved by application of Wald's theory. We first briefly review a model verification strategy for RANSAC based on Wald's SPRT test. The resulting method [8] finds, like RANSAC, a solution that is optimal with user-specified probability. The solution is found in time that is (i) close to the shortest possible and (ii) superior to any deterministic verification strategy.

The RANSAC algorithm proceeds as follows. Repeatedly, subsets of the input data (e.g. a set of tentative correspondences are randomly selected and model parameters fitting the sample are computed. In a second step, the quality of the parameters is evaluated on the input data. Different cost functions have been proposed [21], the standard being the number of inliers, i.e. the number of data points consistent with the model. The process is terminated when the probability of finding a better model becomes lower than a user-specified probability η_0. The $1-\eta_0$ confidence in the solution holds for all levels of contamination of the input data, i.e. for any number of outliers within the input data.

The speed of standard RANSAC depends on two factors: the number of random samples and the number N of the input data points. In all common settings where RANSAC is applied, almost all models whose quality is verified are incorrect with arbitrary parameters originating from contaminated samples. Such models are consistent with only a small number of the data points.

A provably fastest model verification strategy is designed for the (theoretical) situation when the contamination of data by outliers is known. In this case, the algorithm is the fastest possible (on average) of all randomised RANSAC algorithms guaranteeing a given confidence in the solution. The derivation of the optimality property is based on Wald's theory of sequential decision making, in particular a modified sequential probability ratio test (SPRT). In application, the requirement of a priori knowledge of the fraction of outliers is unrealistic and the quantity must be estimated online.

The speed of RANSAC depends on two factors. First, the percentage of outliers determines the number of random samples needed to guarantee the a given confidence in the optimality of the solution. Second, the time needed to assess the quality of a hypothesised model parameters is proportional to the number N of input data points. The total running time t of RANSAC can be expressed as

$$t = k(t_M + \overline{m}_S\, t_V)\ , \tag{5.26}$$

where k is the number of samples drawn, t_M is time needed to instantiate a model hypotheses given a sample, $\overline{m}_S$ is an average number of models per sample and t_V is average time needed to evaluate the quality of the sample. We choose the time needed to verify a single correspondence as the unit of time for t_M, t_V and t. Note that in standard RANSAC $t_V = N$.

The core idea of the Randomised (hypothesis evaluation) RANSAC, i.e. RANSAC with sequential hypothesis testing, is that most evaluated model hypotheses are influenced by outliers. To reject such erroneous models, it is sufficient to perform a statistical test on only a small number of data points. The test can be formulated as follows. The hypothesis generation step proposes a model. It is either 'good', i.e. it is uncontaminated with outliers and leads to the optimal solution (the solution with maximal support), or it is 'bad' (or contaminated), i.e. at least one of the data points in the sample is an outlier The property 'good' is a hidden state that is not directly observable but is statistically linked to observable events. The observable events are "data point (correspondence) is/is-not consistent with the model".

The statistical test has two effects on RANSAC behaviour: it (i) reduces the number of verified correspondences (and thus time complexity of the verification step) and (ii) introduces the possibility of rejecting (overlooking) a good sample. The probability α of rejecting a good sample is the significance of the test and it increases the number of samples drawn before the $1-\eta_0$ confidence is ensured. The correct model parameters are recovered if an uncontaminated sample is drawn and passes the test. This happens with probability

$$P = P_g(1-\alpha)\ .$$

The problem is to find a test that balances the number of correspondences needed for model verification and the increase in the number of samples induced by false rejections so that the total running time t Eq. (5.26) is minimised. Since the average time to draw an uncontaminated model that passes the test is $\bar{k} = 1/(P_g(1-\alpha))$, we have

$$t = \frac{1}{P_g(1-\alpha)}(t_M + \overline{m}_S\, t_V)\ . \tag{5.27}$$

5.6.1 The Optimal Sequential Test

In sequential testing, as applied e.g. in industrial inspection, the problem is to decide whether a model (or the batch of products) is 'good' or 'bad' in the shortest possible time (i.e. making the smallest number of observations) and yet satisfying the predefined bounds on the probabilities of the two possible errors – accepting a 'bad' model as 'good' and vice versa. Wald's SPRT test is a solution of this *constrained optimisation* problem. The user supplies the acceptable probabilities of the errors of the first and the second kind and the resulting optimal test is a trade-off between time to decision (or cost of observations) and the errors committed.

However, when evaluating RANSAC, the situation is different. First of all, a 'good' model is always evaluated for all data points (correspondences) since the number of inliers is one of the outputs of the algorithms. So the only error that can be committed is an early rejection of a 'good' model (error of the first kind). But this only means that more samples have to be drawn to achieve the required confidence $1-\eta_0$ of finding the optimal solution. So unlike in the classical setting, we are solving a *global optimisation* problem, minimising a single real number – the time to decision, since the consequence of an error is also a loss of time.

The model evaluation step of the optimal R-RANSAC proceeds as Wald's sequential probability ratio test (SPRT) with the probability α of rejecting a 'good' sample set to achieve maximum speed of the whole RANSAC process.

In the model evaluation step, our objective is to decide between the hypothesis H_g that model is 'good' and the alternative hypothesis H_b that the model is 'bad'. A 'good' model is computed from an all-inlier sample. The Wald's SPRT is based on the likelihood ratio [23]

$$\lambda_j = \prod_{r=1}^{j} \frac{p(x_r|H_b)}{p(x_r|H_g)} = \lambda_{j-1} \cdot \frac{p(x_j|H_b)}{p(x_j|H_g)} \,, \tag{5.28}$$

a ratio of two conditional probabilities of the observation x_r under the assumptions of H_g and H_b respectively. Not that here, unlike in the case of face and interest point detection, observations are independent since we are sampling at random and the product rule applies. In RANSAC, x_r is equal to 1 if the r-th data point is consistent with a model with parameters θ and 0 otherwise. For example, a correspondence is consistent with (i.e. supporting) an epipolar geometry represented by a fundamental matrix F if its Sampson's error is smaller than some predefined threshold [4]. The probability $p(1|H_g)$ that any randomly chosen data point is consistent with a 'good' model is approximated by the fraction of inliers ε among the data points[4]. The probability of a data point being consistent with a 'bad' model is modelled as a probability of a random event with Bernoulli distribution with parameter δ: $p(1|H_b) = \delta$. The process of estimation of δ and ε is discussed in Section 5.6.2.

[4] The probability ε would be exact if the data points were selected with replacement. Since the objective of the verification is to count the size of the support of the model, the correspondences are drawn without replacement. However, the approximation is close.

Algorithm 3 The Adapted Sequential Probability Ratio Test (Adapted SPRT).

Output: model accepted/rejected, number of tested data points j, a fraction of data points consistent with the model

Set $j = 1$
1. Check whether j-th data point is consistent with the model
2. Compute the likelihood ratio λ_j Eq. (5.28)
3. If $\lambda_j > A$, decide the model is 'bad' (model "rejected"), else increment j
4. If $j > N$, where N is the number of correspondences, decide model "accepted" else go to Step 1.

Algorithm 4 The Structure of R-RANSAC with SPRT.

Initialise ε_0, δ_0, calculate A_0 and set $i = 0$.

Repeat until the probability η of finding a model with support larger than $\hat{\varepsilon}$ falls under a user defined value η_0:

1. Hypothesis generation
- Select a random sample of minimum size m from the set of data points.
- Estimate model parameters θ fitting the sample.

2. Verification

1mm=3pt Execute the SPRT (Alg. 3) and update the estimates if

a. Model rejected: re-estimate δ. If the estimate $\hat{\delta}$ differs from δ_i by more than 5% design (i+1)-th test ($\varepsilon_{i+1} = \varepsilon_i$, $\delta_{i+1} = \hat{\delta}$, $i = i+1$)

b. Model accepted and the largest support so far: design (i+1)-th test ($\varepsilon_{i+1} = \hat{\varepsilon}$, $\delta_{i+1} = \hat{\delta}$, $i = i+1$). Store the current model parameters θ.

After each observation the standard Wald's SPRT makes one of three decisions: accept a 'good' model, reject a 'bad' model, or continue testing. Since in RANSAC the total number of inliers is needed to decide on termination, nothing is gained by an early decision in favour of a 'good' model. Therefore the option of an early acceptance of the model has been removed in the Adapted SPRT (Alg. 3). The full SPRT is described e.g. in Wald [23] and, in a more accessible form, in Lee [5].

The Optimal Value of the Decision Threshold The decision threshold A is the only parameter of the Adapted SPRT. In [8], Chum and Matas show show how to set it to achieve optimal performance. The total expected time of RANSAC is expressed as a function of A: The average time to the solution expressed as a function of A is

$$t(A) = \frac{1}{P_g(1-1/A)} \left(t_M + \overline{m}_S \frac{\log A}{\mathbb{E}\left(\log \frac{p(x|H_b)}{p(x|H_g)}\right)} \right) . \tag{5.29}$$

The minimum of $t(A)$ is found iteratively by process with fast convergence.

The R-RANSAC with SPRT algorithm is outlined in Alg. 4. To fully specify details of the algorithm, two issues have to be addressed. First, the estimation of parameters δ and ε; second, the termination criterion guaranteeing $1-\eta_0$ confidence in the solution has to be derived.

Algorithm 4 proceeds like standard RANSAC [1, 4], only instead of checking all data points in the model verification step, the data points are evaluated sequentially and hypotheses with low support are rejected early. After a hypothesis is rejected, δ is re-estimated (Alg. 4, step 2a). Accepted hypotheses are candidates for the RANSAC outcome (see below). The overhead of the evaluation of the likelihood ratio λ_j Eq. (5.28) is negligible compared to the evaluation of the model versus data point error function.

5.6.2 Estimation of δ and ε

The optimal test derived in Section 5.6.1 requires the knowledge of two parameters, ε and δ. These probabilities are different for different data sets and we assume they are unknown. The proposed algorithm uses values of ε and δ that are estimated during the sampling process and the test is adjusted to reflect the current estimates.

If the probabilities ε and δ are available a-priori, e.g. in some standard setting where the algorithm is run repeatedly, they can be used in the initialisation of the algorithm.

Estimation of δ. Since almost all tested models are 'bad'[5], the probability δ can be estimated as the average fraction of consistent data points in rejected models. When current estimate δ differs from the estimate used to design the SPRT (by more than 5%, for example), new (i+1)-th test is designed. The initial estimate δ_0 is obtained by geometric considerations, i.e. as a fraction of the area that supports a hypothesised model (a strip around an epipolar line in case of epipolar geometry) to the area of possible appearance of outlier data (the area of the search window). Alternatively, a few models can be evaluated without applying SPRT in order to obtain an initial estimate of δ.

Estimation of ε. In general, it is not possible to obtain an unbiased estimate of ε, since this would require the knowledge of the solution to the optimisation problem we are solving. The tightest lower bound on ε is provided by the size of the largest support so far. It was shown in [7] that a sample with the largest support so far appears $\log k$ times, where k is the number of samples drawn. When such a sample (with support of size I_{i+1}) appears, new test is designed for $\varepsilon_{i+1} = I_{i+1}/N$. Throughout the course of the algorithm, a series of different tests with

$$\varepsilon_0 < \cdots < \varepsilon_i < \cdots < \varepsilon$$

are performed. The initial value of ε_0 can be derived from the maximum time the user is willing to wait for the algorithm to terminate.

The properties of R-RANSAC with SPRT were tested on a wide range of standard data and a two to tenfold speed up of the algorithm was observed [8]. Tests

[5] RANSAC verifies, on average, $-\log(\eta_0)$ 'good' models, e.g. for the typical $\eta_0 = 0.05$ a 'good' model is hypothesised three times prior to termination of the algorithm.

included epipolar geometry estimation in both wide and narrow baseline settings and homography estimation.

5.7 Conclusions

A framework exploiting Wald's sequential analysis for designing time-efficient two-class detection and matching algorithms was presented. Besides Wald's Sequential Probability Ratio Test, we relied on WaldBoost, a method that allows learning sequential classifiers in the case of non-i.i.d. features.

The WaldBoost algorithm was applied to the problems of face and interest point detection. Error rates of the face detector proposed algorithm were comparable to the state-of-the-art methods. In the interest point application, the output of the Hessian-Laplace detector [9] was approximated by a sequential WaldBoost classifier consisting of 20 weak classifiers. The detector was evaluated according to the standard testing protocol on reference images [10] and its repeatability was similar to the *affine* version of the Hessian-Laplace detector. The WaldBoost detector gains the speed of the original manually tuned Hessian-Laplace algorithm — only about 2 weak classifiers are evaluated per window on average, which means that about eight additions are needed on average to decide a window corresponds to an interest point. Finally, we have presented a sequential strategy based on Wald's SPRT for evaluation of model quality in RANSAC. The resulting RANSAC with SPRT is significantly faster (2 to 10 times) than its deterministic counterpart.

Acknowledgements The authors were supported by EC projects FP6-IST-004176 COSPAL (JM), FP6-IST-027113 eTRIMS (JŠ) and by the Grant Agency of the Czech Republic project 201/06/1821.

References

1. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. CACM **24**(6), 381–395 (1981)
2. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences **55**(1), 119–139 (1997)
3. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Tech. rep., Department of Statistics, Sequoia Hall, Stanford Univerity (1998)
4. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision, 2nd edn. Cambridge University, Cambridge (2003)
5. Lee, P.M.: Sequential probability ratio test. University of York. www.york.ac.uk/depts/maths/teaching/pml/ais/sprt.ps
6. Li, S., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.: Statistical learning of multi-view face detection. In: ECCV, p. IV: 67 ff. (2002)
7. Matas, J., Chum, O.: Randomized RANSAC with $T_{d,d}$ test. Image and Vision Computing **22**(10), 837–842 (2004)

8. Matas, J., Chum, O.: Randomized ransac with sequential probability ratio test. In: S. Ma, H.Y. Shum (eds.) Proc. IEEE International Conference on Computer Vision (ICCV), vol. II, pp. 1727–1732. IEEE Computer Society Press, New York, USA (2005)
9. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. International Journal of Computer Vision **60**(1), 63–86 (2004). URL `http://lear.inrialpes.fr/pubs/2004/MS04`
10. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.: A Comparison of Affine Region Detectors. IJCV (2005). To appear
11. Nister, D.: Preemptive RANSAC for live structure and motion estimation. In: Proc. ICCV03, vol. I, pp. 199–206 (2003)
12. Rowley, H., Baluja, S., Kanade, T.: Neural network-based face detection. PAMI **20**(1), 23–38 (1998)
13. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning pp. 37(3): 297–336 (1999)
14. Scott, D.W.: Multivariate Density Estimation : Theory, Practice, and Visualization. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons (1992)
15. Šochman, J., Matas, J.: WaldBoost - learning for time constrained sequential detection. In: C. Schmid, S. Soatto, C. Tomasi (eds.) Proc. of Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 150–157. IEEE Computer Society, Los Alamitos, California, USA (2005)
16. Šochman, J., Matas, J.: Learning a fast emulator of a binary decision process. In: ACCV (2007)
17. Sung, K.K., Poggio, T.: Learning human face detection in cluttered scenes. In: Computer Analysis of Images and Patterns, pp. 432–439 (1995). URL `citeseer.nj.nec.com/sung95learning.html`
18. Tordoff, B., Murray, D.: Guided sampling and consensus for motion estimation. In: Proc. 7th ECCV, vol. 1, pp. 82–96. Springer-Verlag (2002)
19. Torr, P., Zisserman, A., Maybank, S.: Robust detection of degenerate configurations while estimating the fundamental matrix. CVIU **71**(3), 312–333 (1998)
20. Torr, P.H.S.: Outlier detection and motion segmentation. Ph.D. thesis, Dept. of Engineering Science, University of Oxford (1995)
21. Torr, P.H.S., Zisserman, A.: MLESAC: A new robust estimator with application to estimating image geometry. CVIU **78**, 138–156 (2000). URL `http://www.robots.ox.ac.uk/~vgg/vggpapers/Torr2000.ps.gz`
22. Viola, P., Jones, M.: Robust real time object detection. In: SCTV. Vancouver, Canada (2001)
23. Wald, A.: Sequential analysis. Dover, New York (1947)
24. Wu, B., AI, H., Huang, C., Lao, S.: Fast rotation invariant multi-view face detection based on real adaboost. In: FGR (2004)
25. Xiao, R., Zhu, L., Zhang, H.: Boosting chain learning for object detection. In: ICCV, pp. 709–715 (2003)

Chapter 6
Pose Estimation and Feature Tracking for Robot Assisted Surgery with Medical Imaging

Christophe Doignon, Florent Nageotte, Benjamin Maurin and Alexandre Krupa

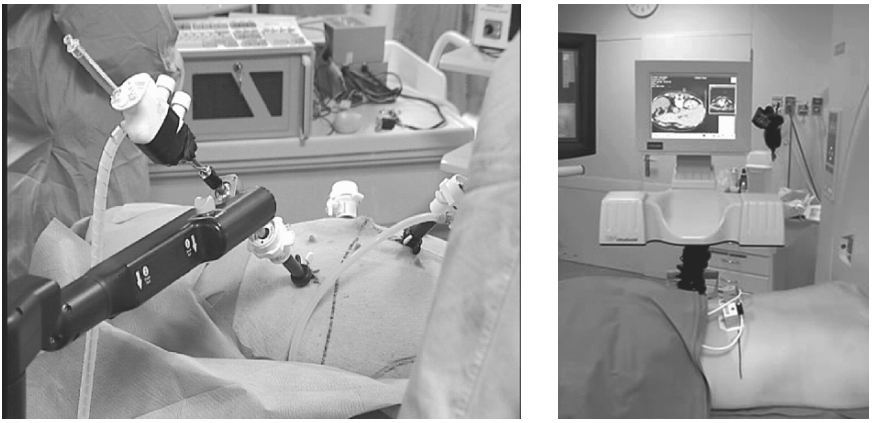
6.1 Introduction

The field of vision-based robotics has been widely growing for more than three decades, and more and more complex 3-D scenes are within robot vision capabilities thanks to better understanding of the scenes, improvement of computer capabilities and control theory. The achievement of applications like medical robotics, mobile robotics, micro-robotic manipulation, agricultural automation or the observation by aerial or underwater robots needs the integration of several research areas in computer vision and automatic control ([32, 19]).
For the past two decades, medical robot and computer-assisted surgery have gained increasing popularity. They have expanded the capabilities and comfort for both patients and surgeons in many kinds of interventions such as local therapy, biopsies, tumors detection and removal with techniques like multi-modal registration, online visualization, simulators for specific interventions and tracking. Medical robots provide a significant help in surgery, mainly for the improvement of positioning accuracy and particularly for intra-operative image guidance [36]. The main challenge in visual 3-D tracking for medical robotic purposes is to catch the relevant video information from images acquired with endoscopes [5], ultra-sound probes [17, 21] or scanners [35, 26] so as to evaluate the position and the velocity of objects of interest

Christophe Doignon and Florent Nageotte
University of Strasbourg, LSIIT (UMR ULP-CNRS 7005), Control, Vision and Robotics Team, Boulevard Brant, 67412 Illkirch, France. e-mail: \{doignon,nageotte\}@lsiit.u-strasbg.fr

Benjamin Maurin
Cerebellum Automation Company, 178, route de Cran Gevrier, 74650 Chavanod, France.
e-mail: maurin@eavr.u-strasbg.fr

Alexandre Krupa
IRISA/INRIA Rennes (Lagadic Team), Campus de Beaulieu, 35042 Rennes, France.
e-mail: alexandre.krupa@irisa.fr



(a) (b)

Fig. 6.1 (a) The laparoscopic experimental setup. The instrument is mounted on the end effector of a surgical robot and inserted through the abdominal wall while the laparoscope is inserted through another insertion point. (b) An image-guidance with CT scanners feedback control during the percutaneous insertion of a radio-frequency needle.

which usually are natural or artificial landmarks attached to a surgical instrument.

This chapter presents several 3-D pose estimation algorithms and visual servoing-based tracking with monocular vision systems such as endoscopes and CT scanners (see Fig. 6.1) developped in an attempt to improve the guidance accuracy. These are intended for the 3-D positioning and guidance of surgical instruments in the human body. The efficiency of most of model-based visual servoing approaches relies on correspondences between the position of tracked visual features in the current image and their 3-D attitude in the world space. If these correspondences contain errors then the servoing usually fails or converges towards a wrong position. Overcoming these errors is often achieved by improving the quality of tracking algorithms and features selection methods ([37, 20]). Following this purpose, the work integrates several issues where computational vision can play a role:

1. estimating the distance between the tip of a laparoscopic instrument and the targeted organ with projected collinear feature points,
2. estimating the 3-D pose of an instrument using a multiple features tracking and a virtual visual servoing,
3. positioning a cylindrical-shaped instrument,
4. registering the instantaneous position of a robot using stereotaxy.

The chapter is organized as follows. In the next Section, the problem of the pose estimation of surgical instruments with markers is stated and solved for some degrees of freedom. In Section 3, we focus on the positioning of the symmetry axis of a cylindrical-shaped instrument. Applications of both Sections use endoscopic vision

in laparoscopy. The stereotactic registration with a single view (2-D/3-D registration) is studied as a pose estimation problem in Section 4. Finally, a conclusion with some perspectives is drawn in Section 5.

6.2 Pose Estimation of a Laparoscopic Instrument with Landmarks

6.2.1 Pose Estimation with Collinear Markers

There exist several difficulties when tackling the problem of estimating the 3-D position of a laparoscopic surgical instruments with a single endoscopic view. One difficulty is the use of monocular vision which gives poor depth information. Another one relies on the highly unstructured nature of the scene with varying lighting conditions and with a background moving due to breathing or heart beating. To solve these problems, we conceived five years ago a special instrument which projects a laser pattern onto the organ surface in order to provide the relative orientation of the instrument with respect to the organ, even if the instrument is not in the camera field of view. Optical markers have been added on the tip of the surgical instrument. These markers (composed of three circular LEDs) were directly projected onto the image and in conjunction with images of the laser pattern, they were used to guide the instrument (see Fig. 6.2). We combined image feature points (spots center coordinates) and depth information for positioning the instrument with respect to the pointed organ [22].

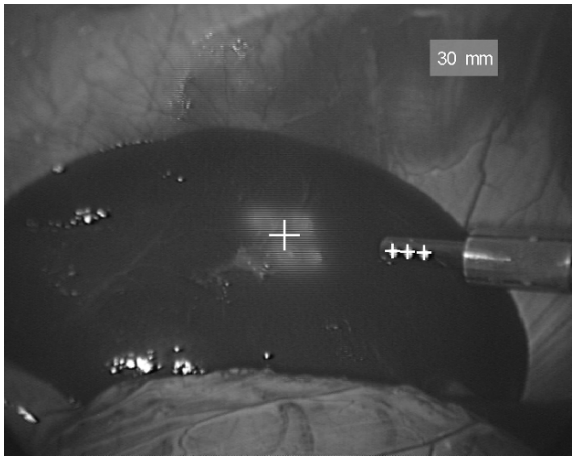


Fig. 6.2 A surgical instrument with a laser pointing device (laser beam - big cross) and three optical markers (three small crosses). The cross-ratio is computed with crosses' centres and controlled so as to estimate the distance and its variations between the tip and the pointed organ surface.

6.2.1.1 Pose Estimation with a Calibrated Endoscope

Recovering the relative orientation (2 degrees of freedom - a unit vector $\mathbf{r}$) and position (3 degrees of freedom - a vector $\mathbf{t}$) of a set of n collinear points such as the optical markers and laser projections in Fig. 6.2 with respect to the camera has been previously investigated by Haralick fifteen years ago [15]. The interpoint distances (structure) and a focal length f of the camera are assumed to be known. Haralick solved this problem with a linear algorithm. Let $P_0 = \mathbf{t}$, $P_1 = \mathbf{t} + \lambda_1 \mathbf{r}$,..., $P_{n-1} = \mathbf{t} + \lambda_{n-1}\mathbf{r}$ be n discriminated points where λ_i represents the distance between the $(i+1)$th and ith points. The first point P_0 is arbitrarily chosen as the origin ($\lambda_0 = 0$), hence the perspective projection $Q_i = (u_i, v_i, 1)^\mathsf{T}$ of the ith point is given by

$$[0\ 0\ 1]\ (\lambda_i \mathbf{r} + \mathbf{t}) \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \mathtt{K}^c\ (\lambda_i \mathbf{r} + \mathbf{t}) \tag{6.1}$$

where $\mathtt{K}^c$ is a (3×3) upper diagonal matrix containing the internal parameters of the camera. From the above equation, Haralick built a homogeneous linear system with a uni-variate matrix $\mathtt{K}^c = \mathtt{diag}(f, f, 1)$ and vectors $\mathbf{t}$ and $\mathbf{r}$ as unknowns

$$\underbrace{\begin{bmatrix} \mathbf{A}_r & \mathbf{A}_t \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \mathbf{r} \\ \mathbf{t} \end{bmatrix} = \mathbf{0}\ . \tag{6.2}$$

$\mathbf{A}$ is a $(2n \times 6)$ real matrix and a closed-form solution can be found with $n \geq 3$ discriminated points. This system may be reformulated as a classical optimization problem with an equality constraint:

$$\min \|\mathbf{A}_r\ \mathbf{r}\ +\ \mathbf{A}_t\ \mathbf{t}\| \quad \text{subject to} \quad \mathbf{r}^\mathsf{T}\mathbf{r}\ = 1\ , \tag{6.3}$$

where $\mathbf{A}_r$ and $\mathbf{A}_t$ are two $(2n \times 3)$ real matrices defined as:

$$\mathbf{A}_r = \underbrace{\begin{bmatrix} \lambda_0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & \lambda_0 & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \lambda_{n-1} & 0 \\ 0 & 0 & \cdots & \cdots & 0 & \lambda_{n-1} \end{bmatrix}}_{\Lambda} \mathbf{A}_t\ ,\ \mathbf{A}_t = \begin{bmatrix} \begin{bmatrix} 1\ 0\ 0 \\ 0\ 1\ 0 \end{bmatrix} \left(\mathtt{K}^c - \begin{bmatrix} u_0 \\ v_0 \\ 1 \end{bmatrix} [0\ 0\ 1] \right) \\ \vdots \\ \begin{bmatrix} 1\ 0\ 0 \\ 0\ 1\ 0 \end{bmatrix} \left(\mathtt{K}^c - \begin{bmatrix} u_{n-1} \\ v_{n-1} \\ 1 \end{bmatrix} [0\ 0\ 1] \right) \end{bmatrix} \tag{6.4}$$

The solution for $\mathbf{r}$ is given by the eigenvector associated with the smallest eigenvalue of the following symmetric matrix

$$\mathbf{E} = \mathbf{A}_r^{\mathsf{T}} \left(\mathbf{I} - \mathbf{A}_t \left(\mathbf{A}_t^{\mathsf{T}} \mathbf{A}_t \right)^{-1} \mathbf{A}_t^{\mathsf{T}} \right) \mathbf{A}_r \tag{6.5}$$

and the position vector $\mathbf{t}$ is given by $\mathbf{t} = -\left(\mathbf{A}_t^{\mathsf{T}} \mathbf{A}_t\right)^{-1} \mathbf{A}_t^{\mathsf{T}} \mathbf{A}_r \mathbf{r}$. We end up with two different estimates for $\mathbf{r}$ (a twofold ambiguity in the sign of $\mathbf{r}$). However, for real objects placed in front of the camera, the third component of vector $\mathbf{t}$ must be strictly positive assuming that the camera z-axis (usually, the optical axis) is pointed towards the scene. This leads to the uniqueness of the solution for the pose.
It is worth pointing out that collinearity is a projective invariant property which is not fully exploited in this technique for pose recovery. Moreover, in presence of both noisy data and close points in the object pattern, matrices $\mathbf{A}_r$ and $\mathbf{A}_t$ are ill-conditioned, which introduces some significant bias in the results. The use of the least mean squares and the lack of data normalization in the original algorithm tend the solution to be sensitive to the condition number. One has to pay attention to data normalization since the pose estimation may be computed with points not always well scattered. This may also lead to numerical problems. To lower the condition number, it seems advisable to normalize data coordinates with an affine transformation as in [16].

6.2.1.2 Distance with Collinear Landmarks

To perform 3-D positioning of an instrument with respect to an organ [22], we need to estimate the distance between the instrument and the targeted organ (depth d_0 in Fig. 6.3). Since the three optical markers centers P_1, P_2 and P_3 are placed along the instrument axis, we assumed they are collinear with the laser spot's barycentre P.

Under this assumption, a cross-ratio can be computed from these four points [28]. This projective invariant can also be computed in the image using their respective projections p_1, p_2, p_3 and p (see Fig. 6.2 and 6.3) and can be used to estimate the depth d_0. In other words, since a 1-D projective basis can be defined either with $\{P_1, P_2, P_3\}$ or their respective images $\{p_1, p_2, p_3\}$, the-cross ratio is a projective invariant built with the fourth point (P or p). Consequently, a 1-D homography H exists between these two bases, so that the straight line Δ corresponding to the instrument axis is transformed, in the image, into a line $\delta = H(\Delta)$ as shown in Fig. 6.3. The cross-ratio τ is given by:

$$\tau = \frac{\left(\frac{\overline{pp_2}}{\overline{p_1p_2}}\right)}{\left(\frac{\overline{pp_3}}{\overline{p_1p_3}}\right)} = \frac{\left(\frac{\overline{PP_2}}{\overline{P_1P_2}}\right)}{\left(\frac{\overline{PP_3}}{\overline{P_1P_3}}\right)} \tag{6.6}$$

and d_0 is obtained as

$$d_0 = \overline{PP_1} = (1-\tau)\frac{\overline{P_1P_3}}{\tau - \frac{\overline{P_1P_3}}{\overline{P_1P_2}}} = \alpha \, \frac{1-\tau}{\tau - \beta} \tag{6.7}$$

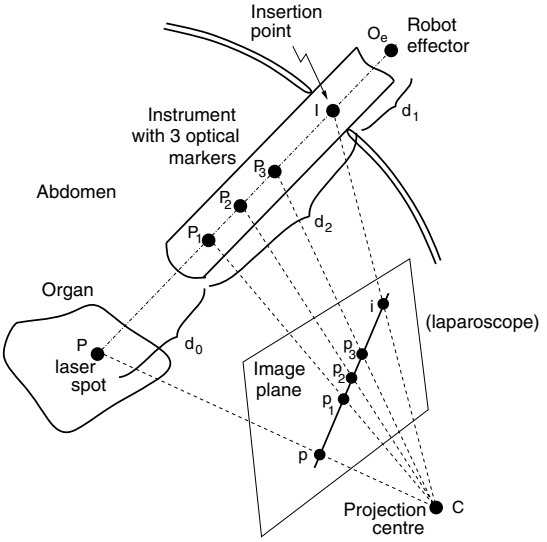


Fig. 6.3 The basic geometry involved for the relative instrument positioning with the laser spot aligned with three collinear LED centres.

where α and β depend only on the known relative position of P_1, P_2 and P_3. To simplify the computation of the cross-ratio in the image plane, it's necessary to characterize the straight line δ in order to relate the pixels coordinates of an image point $p = (u, v, 1)^\mathsf{T}$ and its projective coordinates $(s\lambda, s)^\mathsf{T}$ on δ. Let $(-b, a)^\mathsf{T}$ be the normalized cosine direction of δ and $p_k = (u_k, v_k, 1)^\mathsf{T}$ a point on δ. This gives:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} -b & u_k \\ a & v_k \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}} \begin{bmatrix} \lambda \\ 1 \end{bmatrix} \tag{6.8}$$

or

$$\lambda = \begin{bmatrix} -b & a \end{bmatrix} \begin{bmatrix} u - u_k \\ v - v_k \end{bmatrix} \tag{6.9}$$

with :

$$\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0 \tag{6.10}$$

where $(-c)$ is the orthogonal distance from δ to the image origin. The computation of the cross-ratio is then:

$$\tau = \frac{\lambda_0 + \overline{p_1 p_2}}{\overline{p_1 p_2}} \frac{\overline{p_1 p_3}}{\lambda_0 + \overline{p_1 p_3}} \tag{6.11}$$

From equation (6.7), it is straightforward that d_0 is a function of τ which, in turn, is a function of λ_0, $\overline{p_1p_2}$ and $\overline{p_1p_3}$. Similar computations lead to the same relationship between d_2 and another cross-ratio μ defined with the points P_1, P_2, P_3, I and their respective projections provided that i, the perspective projection of the incision point I (see Fig. 6.4), can be recovered [10]. Since I is generally not in the camera field of view, this can be achieved by considering a displacement of the surgical instrument between two configurations yielding straight lines δ and δ' in the image. Then, i is the intersection of these lines given that I is invariant. Finally :

$$\mu = \frac{\left(\frac{\overline{p_1p_3}}{\overline{p_2p_3}}\right)}{\left(\frac{\overline{p_1i}}{\overline{p_2i}}\right)} = \frac{\left(\frac{\overline{P_1P_3}}{\overline{P_2P_3}}\right)}{\left(\frac{\overline{P_1I}}{\overline{P_2I}}\right)}\,, \tag{6.12}$$

$$d_2 = \overline{P_1I} = \frac{\frac{\alpha}{1-\beta}}{\mu + \frac{\beta}{1-\beta}}\,. \tag{6.13}$$

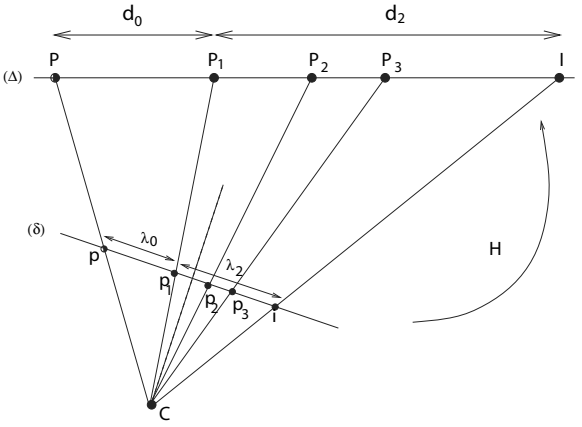


Fig. 6.4 Markers P_1,P_2,P_3 on the tool axis Δ and their images p_1, p_2, p_3 on line δ. Note that $i = H(I)$ is invariant during surgical procedures.

6.2.2 Pose Estimation with Multiple Features

6.2.2.1 Objectives and Related work

In many early works, the images of laparoscopic instruments are segmented, in order to control the position of the endoscopic camera. These methods are based on the structure, mainly the apparent lines of the instrument [1, 6], or on its frequential features [8, 41]. In order to make the detection more robust and accurate, instru-

ments can be marked with structuring markers as described before ([1, 43, 22]) or frequential (color) markers ([42, 38]). Most of these works use only the 2-D position of the instrument in the image and the accuracy of the features extraction is not so important for aimed applications. On the contrary, the 3-D pose estimation requires a very accurate feature extraction step.

None of the earlier works have focused on the complete 6 degrees of freedom detection. To determine the six degrees of freedom, the non symmetric part of the instrument has to be used or the instrument has to be marked. The second solution avoids using a CAD model and can be applied to any kind of instrument. We use the marker presented in Fig. 6.5, which is composed of twelve black spots on a white area, building four "marker lines" which can be discriminated from each other by using the cross-ratio invariant. For more information on the choice of this marker, the interested reader can refer to [30]. The detection of this marker is mainly based on the intensity of the white area and the black spots in the endoscopic images. It can thus be used in grey level images as well as color endoscopic images. The results of the detection technique are shown in Fig. 6.5 for laboratory endoscopic images. The image features noted $\mathbf{s}$ are n points corresponding to the centres of the visible spots (generally $n \in [0,6]$) and two lines $\mathbf{l}^+$ and $\mathbf{l}^-$ corresponding to the apparent contours of the shaft in the endoscopic image and represented by their distance ρ to the origin of the image and their orientation θ:

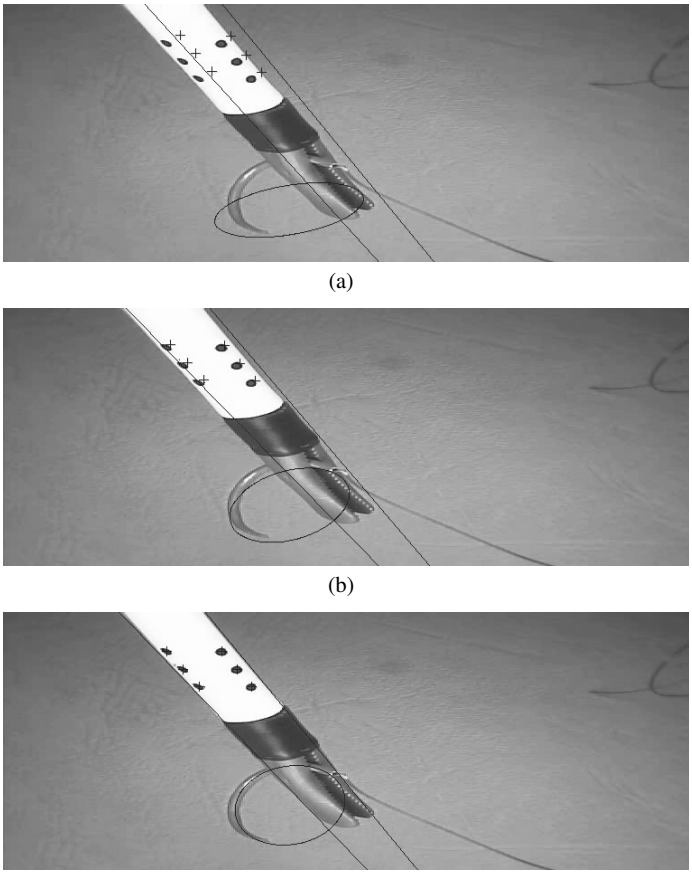
$$\mathbf{s} = \left(\mathbf{l}^+, \mathbf{l}^-, p_1, \cdots, p_n\right)^\mathsf{T} . \tag{6.14}$$

Depending on the size of the white marker area in the image, the complete extraction process can take up to 200 ms. In order to track the instrument at higher rate, we have developed techniques based on the moving edges method due to Bouthemy [3]. The main difficulty is to track the black spots which can appear and disappear, due to the rotation of the instrument around its own axis, and possibly due to occlusions. We have proposed a method based on the prediction of the spots appearance and disappearance which allows to track the markers without the need to register the images of the spots with the real positions [31]. Thus, the tracking of the instrument can be handled at a rate of 20 Hz as long as at least one spot is visible in the endoscopic images.

6.2.2.2 Pose Estimation of a Tagged Instrument

Our model-based pose estimation process requires a calibrated camera. Endoscopic cameras have a large field of view and include large radial distortion. As a consequence, the calibration method must estimate the distortion parameter [2, 39].

Only four degrees of freedom are necessary to estimate the attitude of the instrument axis (see next Section). Theoretically, the 4 degrees of freedom of the pose can be determined using the contour generator and its image (the apparent contour) of the



(a)

(b)

(c)

Fig. 6.5 (a-c) The pose estimation as a virtual visual servoing process with multiple geometric features (apparent lines, marker-lines and eventually the circular needle). (a) The straight lines are the projections with the initial virtual camera position. (c) The projections when the error vector $\mathbf{s} - \mathbf{s}_d$ tends to 0.

cylinder [9]. However, the positions of the marking spots not only define the proper rotations and translations, but also give information on the orientation and position of the axis of the shaft. We then chose to estimate all the degrees of freedom of the instrument. This can be done with analytical methods using both the apparent contours and one known point at the cylinder's surface [29]. Other methods, like those proposed by Horaud [18], Haralick [14], DeMenthon [7] or Quan [33] can also be used. However, the full pose estimation is interesting for robustness considerations only if all the available information given by the apparent lines and all the spots is used. To this purpose, the Virtual Visual Servoing (VVS) due to Marchand and Sundareswaran ([24, 34]) may handle the information redundancy. VVS is a numerical iterative method for minimizing the error between the extracted features and the

forward projection of the object in the images, based on the image-based visual servoing (IBVS) schemes. This process needs the computation of an interaction matrix which relates the variations of each image feature and the the camera velocity screw τ. With the image features we use, the interaction matrix L_s has the following form:

$$\begin{pmatrix} \dot{\mathbf{l}}^+ \\ \dot{\mathbf{l}}^- \\ \dot{p}_1 \\ \vdots \\ \dot{p}_n \end{pmatrix} = \underbrace{\begin{pmatrix} L_{line}(\mathbf{l}^+) \\ L_{line}(\mathbf{l}^-) \\ L_{pt}(p_1) \\ \vdots \\ L_{pt}(p_n) \end{pmatrix}}_{L_s} \underbrace{\begin{pmatrix} {}^cV^C_{c/i} \\ {}^c\boldsymbol{\omega}_{c/i} \end{pmatrix}}_{\tau}. \tag{6.15}$$

The interaction matrices associated to a point L_{pt} and to a line L_{line} can be found in the works of Chaumette [11]. In order to guarantee a fast convergence and a good stability of the VVS, it is useful to initialize the algorithm close enough to the real pose (from which a desired feature vector $\mathbf{s}_d$ can be defined). To this purpose, we use either the DeMenthon iterative method when at least four points are visible. From the obtained initial estimate, the following control law is applied to the virtual camera

$$\tau = -\lambda \tilde{L_s}^+ (\mathbf{s} - \mathbf{s}_d) \tag{6.16}$$

until the control vector becomes smaller than a specified value. The process converges quickly towards the real pose of the camera (see Fig. 6.5-c).

6.3 Pose Estimation of a Laparoscopic Instrument without Landmarks

6.3.1 Problem Statement and Perspective Projection

The aim of this section is to briefly present a new algorithm for the determination of the pose of a straight homogeneous circular cylinders (SHCC) without markers, that is to say directly from the apparent contour. More details are provided in [9]. The apparent contour (γ) of a cylinder is a set of points which intersect the viewline and the image plane. It is the projection of a 3-D curve on the cylinder's surface referred to as the contour generator (Γ).

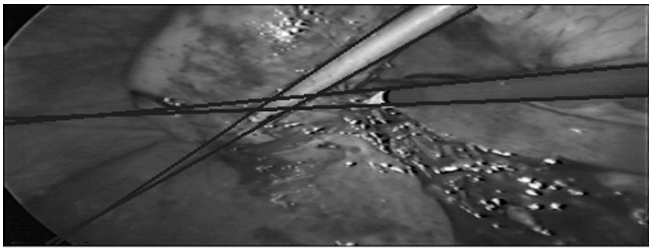
Given the matrix $\mathtt{K}^c$ of camera intrinsic parameters, the cylinder radius r_c and the apparent lines $\{\mathbf{l}^-, \mathbf{l}^+\}$, we look for the estimation of the Plücker coordinates $(\mathbf{r}, \mathbf{w})$ of the cylinder axis (see Fig. 6.7) satisfying the non-linear equation $\mathbf{r}^\mathsf{T}\mathbf{w} = 0$. This means that one has to solve a polynomial equation for a unique (double) solution (see [9]) that is, for a null discriminant. This one equals $B^2 - AC = 0$ with

$$\begin{cases} A = m^{\mathsf{T}}(\mathtt{K}^c)^{-\mathsf{T}}[\mathbf{r}]_\times\,[\mathbf{r}]_\times^{\mathsf{T}} m \\ B = \quad m^{\mathsf{T}}(\mathtt{K}^c)^{-\mathsf{T}}[\mathbf{r}]_\times \mathbf{w} \\ C = \qquad \|\mathbf{w}\|^2 - r_c^2 \end{cases} \tag{6.17}$$

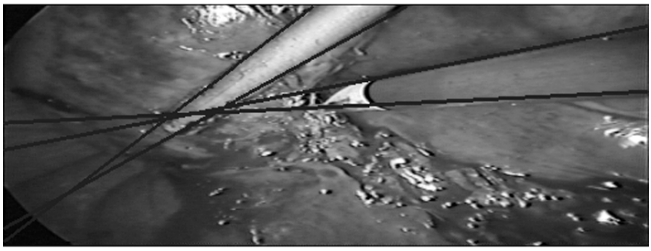
and after some computations, it can be expressed as

$$(\frac{r_c\,B}{\sqrt{C}} + \mathbf{w}^{\mathsf{T}}(\mathtt{K}^c)^{-1}m)^{\mathsf{T}}\,(\frac{r_c\,B}{\sqrt{C}} - \mathbf{w}^{\mathsf{T}}(\mathtt{K}^c)^{-1}m) \;=\; 0\,. \tag{6.18}$$

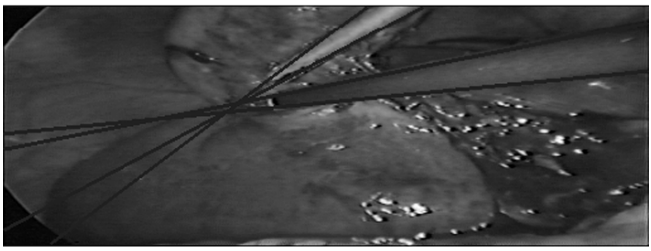
If the scalar $C \leq 0$, the projection center $(0,0,0,1)^{\mathsf{T}}$ is located inside the cylinder (or on its surface, if $C = 0$) and the above equation yields no real solutions. We do not consider these special cases in the remainder, we rather focus this work on the more practical situation with real solutions. In the case of a circular cylinder with infinite



(a)



(b)



(c)

Fig. 6.6 (a-c) Segmentation of three endoscopic images of surgical instruments in the abdominal cavity. The degenerate conic fitting superimposed with the overall apparent contour of the two detected cylindrical instruments.

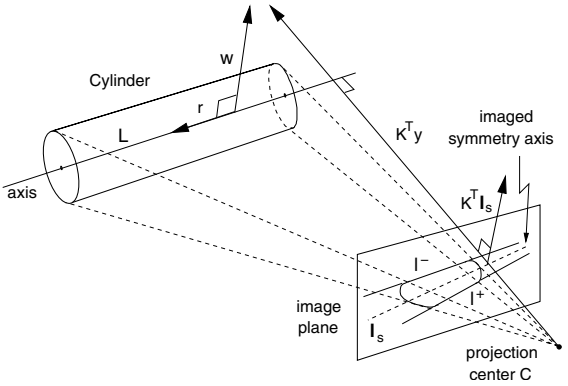


Fig. 6.7 A cylinder and its image with the perspective projection P. The backprojection of the apparent lines $(\mathbf{l}^-,\mathbf{l}^+)$ is a pair of planes $\mathtt{P}^\mathsf{T}\mathbf{l}^-$ and $\mathtt{P}^\mathsf{T}\mathbf{l}^+$, passing through the projection center. The image of the cylinder axis Δ is the axis $\mathbf{l}_s$ of the harmonic homology H relating the apparent lines.

height and a constant radius, equation (6.18) shows that the apparent contour is a set of two straight lines represented either with the pair of vectors $\mathbf{l}^-$ and $\mathbf{l}^+$ satisfying

$$\begin{cases} (\mathbf{l}^-)^\mathsf{T}\, m \equiv \{(\mathtt{K}^c)^{-\mathsf{T}}\,(\mathtt{I}-\alpha[\mathbf{r}]_\times)\,\mathbf{w}\}^\mathsf{T}\, m = 0 \\ (\mathbf{l}^+)^\mathsf{T}\, m \equiv \{(\mathtt{K}^c)^{-\mathsf{T}}\,(\mathtt{I}+\alpha[\mathbf{r}]_\times)\,\mathbf{w}\}^\mathsf{T}\, m = 0 \end{cases} \tag{6.19}$$

with $\alpha = r_c/\sqrt{\|\mathbf{w}\|^2 - r_c^2}$, or alternatively with the (3×3) matrix $\mathtt{C} = \mathbf{l}^-\,\mathbf{l}^{+\mathsf{T}} + \mathbf{l}^+\,\mathbf{l}^{-\mathsf{T}}$ satisfying $m^\mathsf{T}\,\mathtt{C}\,m = 0$. It is a rank-2 symmetrical matrix defined up to a scale and both representations are equivalent to model the apparent contour with 4 parameters.

6.3.2 Direct Pose Computation

In this paragraph we present a linear algorithm for the pose estimation. Starting from (6.19), the matrix C can be related to the pose parameters since we have

$$\mathtt{K}^{c\mathsf{T}}\,\mathtt{C}\,\mathtt{K}^c \equiv \mathtt{K}^{c\mathsf{T}}\,\left(\mathbf{l}^-\,\mathbf{l}^{+\mathsf{T}} + \mathbf{l}^+\,\mathbf{l}^{-\mathsf{T}}\right)\,\mathtt{K}^c \tag{6.20}$$

$$\equiv (\alpha[\mathbf{r}]_\times - \mathtt{I})\,\mathbf{w}\mathbf{w}^\mathsf{T}\,(\alpha[\mathbf{r}]_\times^\mathsf{T} + \mathtt{I}) + (\alpha[\mathbf{r}]_\times + \mathtt{I})\,\mathbf{w}\mathbf{w}^\mathsf{T}\,(\alpha[\mathbf{r}]_\times^\mathsf{T} - \mathtt{I}) \tag{6.21}$$

$$\equiv \alpha^2[\mathbf{r}]_\times\mathbf{w}\mathbf{w}^\mathsf{T}[\mathbf{r}]_\times + \mathbf{w}\mathbf{w}^\mathsf{T} \tag{6.22}$$

$$\equiv [\mathbf{r}]_\times(\alpha^2\frac{\mathbf{w}\mathbf{w}^\mathsf{T}}{\|\mathbf{w}\|^2} + \frac{[\mathbf{w}]_\times[\mathbf{w}]_\times^\mathsf{T}}{\|\mathbf{w}\|^2})[\mathbf{r}]_\times^\mathsf{T} = [\mathbf{r}]_\times(\mathtt{I} - (1-\alpha^2)\frac{\mathbf{w}\mathbf{w}^\mathsf{T}}{\|\mathbf{w}\|^2})[\mathbf{r}]_\times^\mathsf{T} \tag{6.23}$$

$$\equiv \left(\mathtt{I} - \mathbf{r}\mathbf{r}^\mathsf{T} - \mathbf{z}\mathbf{z}^\mathsf{T}\right) = \begin{bmatrix}\mathbf{a} & \mathbf{z}_u & \mathbf{r}\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & 1-\sigma^2 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{a}^\mathsf{T}\\ \mathbf{z}_u^\mathsf{T}\\ \mathbf{r}^\mathsf{T}\end{bmatrix} \tag{6.24}$$

with $\mathbf{z} = \frac{\sqrt{1-\alpha^2}}{\|\mathbf{w}\|}\,[\mathbf{r}]_\times \mathbf{w}$ and the unit vector $\mathbf{z}_u = \mathbf{z}/\sigma$. On the other hand, the SVD has the following expression ${\mathtt{K}^c}^\mathsf{T}\,\mathtt{C}\,\mathtt{K}^c = \mathbf{U}\,\mathbf{D}\,\mathbf{U}^\mathsf{T} = \mathbf{U}\,\mathrm{diag}(\lambda_1,\lambda_2,0)\,\mathbf{U}^\mathsf{T}$. Then, it is easy to see that $\mathbf{U} = \begin{bmatrix}\mathbf{a} & \mathbf{z}_u & \mathbf{r}\end{bmatrix}$ and

$$\sigma = \|\mathbf{z}\| = \sqrt{1-\alpha^2} = \sqrt{1-\frac{\lambda_2}{\lambda_1}}\,.$$

Finally, $\mathbf{w} \equiv \mathbf{z}_u \times \mathbf{r} = \mathbf{a}$ and

$$\|\mathbf{z}\| = \sqrt{1-\frac{r_c^2}{\|\mathbf{w}\|^2 - r_c^2}} \;\Rightarrow \|\mathbf{w}\| = r_c\,\sqrt{1+\frac{\lambda_1}{\lambda_2}}\,. \tag{6.25}$$

Many results are provided in [9]. In particular, we have compared the pose computation from the apparent contours of cylinders and the Haralick's method for the pose of a set of collinear points [15] with some artificial markers attached to the cylinder's surface. Here, we rather focus the discussion on the application of concern which is the image-guidance for intra-operative procedures in minimally invasive surgery (MIS). In laparoscopic surgery, most of surgical instruments have cylindrical parts and are metallic (see Fig. 6.6) leading to grey regions with many specularities in the image. Prior researchs involving such endoscopic images have been conducted in the field of color image segmentation [8]. Once regions have been segmented, the region boundaries are ordered and used to perform a degenerate conic-based contours fitting. With the calibrated and distortion-corrected endoscope used in the experiments, the 3-D localization of the two moving surgical instruments in Fig. 6.6 has been done with success for more than 300 successive images of the abdominal cavity of a pig.

With the proposed method, the location of each insertion point in laparoscopy can be recovered, on-line, with no marker, without any knowledge of robot kinematics and without an external measurement device [10]. Since any laparoscopic instrument is passsing through this point, the motion constraint in MIS can be expressed as the intersection of multiple convergent 3-D straight lines. Since any (homogeneous) point $\mathbf{X}$ is on L if $\mathtt{L}^\star\mathbf{X} = \mathbf{0}$, given n positions corresponding to the set of dual Plücker matrices $\{\mathtt{L}_1^\star, \mathtt{L}_2^\star, ..., \mathtt{L}_n^\star\}$, the intersection of lines is obtained with a rank-3 $(4n \times 4)$ matrix $\mathtt{G}_n^\mathsf{T}$ such that

$$\mathtt{G}_n = [\mathtt{L}_1^\star, \mathtt{L}_2^\star, ..., \mathtt{L}_n^\star]\,. \tag{6.26}$$

That is to say the null-space of $\mathtt{G}_n^\mathsf{T}$ must be a one-dimensional subspace and the intersection may be computed with n $(n \geq 2)$ 3-D positions. By computing the SVD of $\mathtt{G}_n^\mathsf{T}$, one obtains the common intersection with the singular vector associated to the null singular value (or the smallest one in presence of noisy data). Moreover, the perspective projection of the 3-D line L_j is the image line $\mathbf{l}_j$ defined by

$$[\mathbf{l}_j]_\times = \mathtt{K}^c\mathtt{P}^c\,\mathtt{L}_j\,(\mathtt{K}^c\mathtt{P}^c)^\mathsf{T} = [(\mathtt{K}^c)^{-\mathsf{T}}\,\mathbf{w}_j]_\times \;\Rightarrow\; \mathbf{l}_j \equiv (\mathtt{K}^c)^{-\mathsf{T}}\,\mathbf{w}_j\,, \tag{6.27}$$

where $\mathtt{P}^c$ is the projection matrix. Since vector $\mathbf{l}_j$ is defined up to a scale, it does not depend on the magnitude of vector $\mathbf{w}_j$, hence the n convergent image lines $\mathbf{l}_1, \mathbf{l}_2, ..., \mathbf{l}_n$ must satisfy

$$\left(\mathbf{l}_1 \ \ldots \ \mathbf{l}_n\right)^{\mathsf{T}} \ i \ = \underbrace{\left(\mathbf{w}_1 \ \ldots \ \mathbf{w}_n\right)^{\mathsf{T}}}_{\mathtt{W}_n} \ (\mathtt{K}^c)^{-1} \ i \ = \ \mathbf{0} \tag{6.28}$$

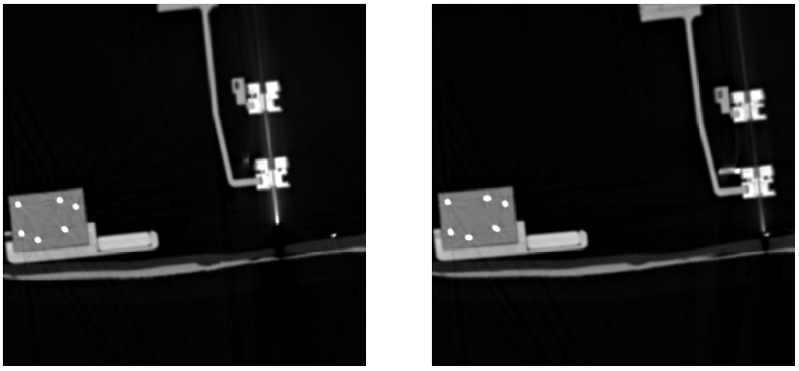
where i is the image of the insertion point I. It follows that a set of n 3-D straight lines is projected to n convergent image lines if the above $(n \times 3)$ matrix $\mathtt{W}_n$ is of rank 2. It is only a necessary condition which does not ensure the convergence of the 3-D lines, but it makes very important the accurate estimation of the imaged cylinder axis (lines $\mathbf{l}_j$), hence the estimation of its Plücker coordinates.

6.4 Pose Estimation of Stereotactic Landmarks

This Section deals with the 2-D/3-D registration of a stereotactic frame from a single slice captured with a computed tomography (CT) scanner. A registration with a single image is very well suited for CT-guided robotic systems in interventional radiology, particularly to quickly correct the needle positioning (see Fig. 6.8) during percutaneous procedures [27].
In stereotaxy, line fiducials are usually used to produce a set of image points that are further employed in a pose estimation algorithm (see Fig. 6.8). To achieve the registration, the matching and pose estimation processes need to be robust and fast enough so as to be convenient in clinical conditions. To this end, a new formulation of the patient-to-modality stereotactic registration with a single image and for any arrangement of the fiducials has been proposed. It is worth pointing out that our solution requires very few fiducials in comparison with previous techniques.

6.4.1 The Imaging Model

Since most CT imaging devices execute some proprietary algorithms to generate image slices and since these algorithms usually are not in the public domain, we consider the imaging device as a black box. In other words, this work is focused on the delivery of a general framework for 2-D/3-D registration rather than a study of the physical properties of each step of image formation. A CT scanner provides slices of objects. It has internal parameters such as the thickness of a slice and scaling parameters that influence the tomographic reconstruction process from the projection measurements. To take care of them, we propose an imaging model composed of an affine transformation accounting for intrinsic parameters, an Euclidean one for the rigid-body transformation that relates the scanner to the stereotactic frame and an orthographic projection that expresses the projection of a thin slice onto the image.



(a) (b)

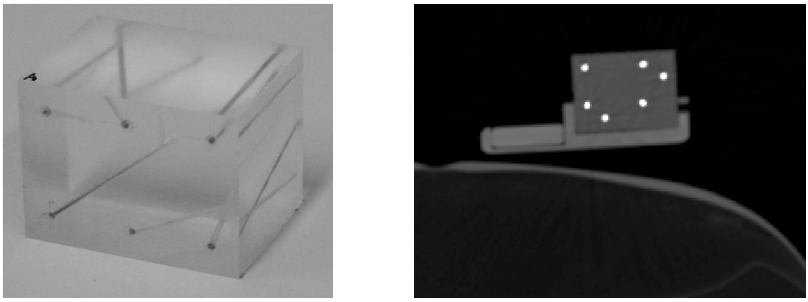
Fig. 6.8 "Look an move" with CT scanners. (a) The needle is maintained in the needle-holder jaws of a lightweight parallel robotic plateform (CT-Bot), which has to be moved to the target point (right). (b) A new acquisition to check for the final positioning.

To formulate the registration, we denote with $\mathscr{F}_0$ the reference frame attached to the fiducials and with $\mathscr{F}_{ct}$ the frame attached to the scanner. A scaled frame $\mathscr{F}_I$ is also attached to the CT image with pixel units instead of millimeters. A point in space like the origin of a reference frame is written in bold as $\mathbf{O}$. The imaging model relates the coordinates of a 3-D point $\mathbf{P}_j$ expressed in $\mathscr{F}_{ct}$ and coordinates of the corresponding point ${}^I\mathbf{Q}_j = \begin{bmatrix} u_j & v_j \end{bmatrix}^\mathsf{T}$ in the image as

$$
\begin{aligned}
{}^{ct}\mathbf{OP}_j = \begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} &= \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}}_{\text{orthographic}} \underbrace{\begin{bmatrix} s_x & g \\ 0 & s_y \end{bmatrix}}_{\text{intrinsic parameters}} \underbrace{\begin{bmatrix} u_j \\ v_j \end{bmatrix}}_{\text{pixel}} \\
&= {}^{ct}\Pi_\pi \, {}^\pi S_I \, {}^I\mathbf{Q}_j
\end{aligned}
\tag{6.29}
$$

where ${}^{ct}\Pi_\pi$ is a (3×2) matrix accounting for the orthographic projection onto the cutting plane (π) (see Fig. 6.10) and the non-null entries of matrix ${}^\pi S_I$ are the intrinsic parameters of the scanner. They consist of two scaling factors s_x and s_y and a shearing parameter g accounting for a gantry tilt angle error or table bending during the scan. Usually, this parameter is very small and it is often neglected. However, it may be identified, since in some circumstances, it may decrease the registration accuracy as it is for MRI [4]. In the rest of the paper, this parameter will be neglected. Since there exists a rigid-body transformation between $\mathscr{F}_0$ and $\mathscr{F}_{ct}$, the expression for the vector $\mathbf{OP}_j$ in $\mathscr{F}_0$ is given by ${}^0\mathbf{OP}_j = R \, {}^{ct}\mathbf{OP}_j + \mathbf{t}$ where R is a rotation matrix and $\mathbf{t}$ is a position vector. Then, one may see the following expression

$$
{}^0\mathbf{OP}_j = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 \end{bmatrix} \, {}^\pi S_I \, {}^I\mathbf{Q}_j + \mathbf{t} \tag{6.30}
$$



(a) (b)

Fig. 6.9 (a) The plastic cube with the line fiducials used for experiments. - (b) A CT scanner image (magnified) when the cube is placed on a phantom.

as a compact representation for the transformation $\mathscr{F}_I \rightarrow \mathscr{F}_0$ including the orthographic projection, where $\mathbf{r}_k$ is the k^{th} column of R. Therefore, the following (3×2) real matrix

$$ {}^0L_I = [\mathbf{r}_1 \ \mathbf{r}_2] \ {}^{\pi}S_I = [\mathbf{l}_1 \ \ \mathbf{l}_2] \tag{6.31} $$

must satisfy the quadratic constraints coming from the orthonormality of any rotation matrix [1] :

$$ \mathbf{l}_1{}^{\top}\mathbf{l}_1 = s_x^2 \ , \ \ \mathbf{l}_2{}^{\top}\mathbf{l}_2 = s_y^2 \ \text{ and } \ \mathbf{l}_1{}^{\top}\mathbf{l}_2 = 0 \, . \tag{6.32} $$

Finally, considering the notations for homogenous coordinates of ${}^I\mathbf{Q}_j$ as ${}^I\underline{\mathbf{Q}}_j = (u_j, v_j, 1)^{\top}$, equation (6.30) is rewritten as

$$ {}^0\mathbf{OP}_j = \left[{}^0L_I \ \mathbf{t} \right] \ {}^I\underline{\mathbf{Q}}_j \, . \tag{6.33} $$

6.4.2 Modeling the Fiducials

Fiducials used in stereotaxy are usually composed of rods (see Fig. 6.9-a) and are represented with straight lines [13, 35, 23]. Let Δ_j be the j^{th} line. This line may be represented with the origin $\mathbf{O}_j$ (3 dof) and a unit vector $\mathbf{y}_j$ (2 dof). Its intersection with the scanner plane (π) is (generally) a point $\mathbf{P}_j = \Delta_j \cap \pi$ (see Fig. 6.9-b), and substituting the expression of ${}^0\mathbf{OP}_j$ in (6.33), it can be expressed with

$$ {}^0\mathbf{OP}_j \ = \ {}^0\mathbf{OO}_j + \lambda_j \, {}^0\mathbf{y}_j \ = \ \left[{}^0L_I \ \mathbf{t} \right] \ {}^I\underline{\mathbf{Q}}_j \ , \ \ \lambda_j \in \mathbb{R} \tag{6.34} $$

[1] In accordance with (6.29), if the shearing parameter g is significant, equation (6.32) should be replaced by $\mathbf{l}_1^{\mathsf{T}}\mathbf{l}_1 = s_x^2$, $\mathbf{l}_1^{\mathsf{T}}\mathbf{l}_2 = s_x g$ and $\mathbf{l}_2^{\mathsf{T}}\mathbf{l}_2 = g^2 + s_y^2$.

where $\mathbf{O}_j$ is the orthogonal projection of the origin of the frame $\mathscr{F}_0$ onto Δ_j, thus satisfying $\mathbf{O}_j\mathbf{P}_j \times \mathbf{y}_j = \mathbf{0}$. Therefore, to achieve the registration, one must solve (6.34) for ${}^0L_I$, $\mathbf{t}$ and the $\{\lambda_j\}$'s, that is for $(9+n)$ unknowns with n lines. Consequently, the size of the system to solve increases with the number of rods, leading to large matrices which must be precessed with many numerical operations [26]. To reduce the number of unknowns, we introduce the Plückerian representation [16]. The Plückerian coordinates of a 3-D line Δ_j are the pair of orthogonal vectors $(\mathbf{y}_j, \mathbf{w}_j)$ (see Fig. 6.10) where $\mathbf{w}_j$ is defined by $\mathbf{w}_j = \mathbf{y}_j \times \mathbf{OP}_j$. With this representation, the origin of the line as well as the $\{\lambda_j\}$'s are removed from the system. The above definition and the latter expression for ${}^0\mathbf{OP}_j$ can be gathered in the following equation, expressed in $\mathscr{F}_0$:

$$[{}^0\mathbf{y}_j]_\times \begin{bmatrix} {}^0L_I & \mathbf{t} \end{bmatrix} {}^I\underline{\mathbf{Q}}_j = {}^0\mathbf{w}_j \,, \tag{6.35}$$

where $[\mathbf{y}_j]_\times$ is the (3×3) skew-symmetric (singular) matrix associated to $\mathbf{y}_j$. Equation (6.35) is the basis for our registration approach.

Generally, intersections of straight lines with the cutting plane should provide as many spots as there are lines (see Fig. 6.9-b). In practice, several spots may be missing in the image or in contrary some artifacts may appear [23]. In practice, line fiducials are bounded ($\lambda_j^{min} \leq \lambda_j \leq \lambda_j^{max}$). It is easy to compute these extremal values for any displacement (R,$\mathbf{t}$) and to check the relevance of the corresponding spot. To do so, a pre-multiplication with a unit vector ${}^0\mathbf{y}_j^{\mathsf{T}}$ in (6.34) gives the following expression

$$\lambda_j = {}^0\mathbf{y}_j{}^{\mathsf{T}} \begin{bmatrix} {}^0L_I & \mathbf{t} \end{bmatrix} {}^I\underline{\mathbf{Q}}_j \,. \tag{6.36}$$

Given n lines/points correspondences, (6.35) can be expressed as a minimization problem with equality constraints:

$$\min_{\mathbf{x}} || A\,\mathbf{x} - \mathbf{b} ||^2 \quad \text{subject to} \quad \mathbf{x}^{\mathsf{T}}\, C\, \mathbf{x} = 0 \tag{6.37}$$

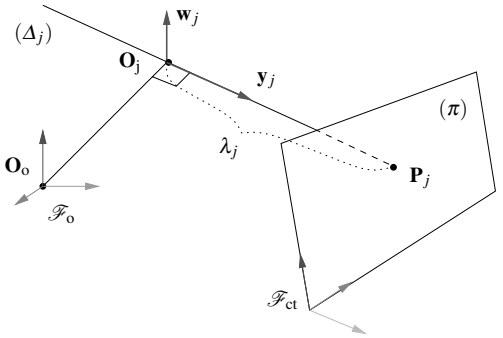


Fig. 6.10 A 3-D line Δ_j crossing the cutting plane (π). The pair of vectors $(\mathbf{y}_j, \mathbf{w}_j)$ is the Plückerian representation of the line.

where $\mathbf{x} = \left[\mathbf{l}_1{}^\mathsf{T} \ \mathbf{l}_2{}^\mathsf{T} \ \mathbf{t}^\mathsf{T} \right]^\mathsf{T}$, C is a (9×9) symmetrical matrix with null entries except for $C_{14} = C_{25} = C_{36} = C_{41} = C_{52} = C_{63}$, A is a $(3n \times 9)$ and $\mathbf{b}$ is a $(3n \times 1)$ matrix, respectively defined as:

$$A = \begin{bmatrix} {}^I\underline{\mathbf{Q}}_1{}^\mathsf{T} \otimes [{}^0\mathbf{y}_1]_\times \\ \vdots \\ {}^I\underline{\mathbf{Q}}_n{}^\mathsf{T} \otimes [{}^0\mathbf{y}_n]_\times \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} {}^0\mathbf{w}_1{}^\mathsf{T} \\ \vdots \\ {}^0\mathbf{w}_n{}^\mathsf{T} \end{bmatrix}. \tag{6.38}$$

6.4.3 Registration as a Pose Estimation Problem

This section aims at designing fast algorithms for estimating the parameters of the rigid registration, assuming a calibrated scanner is available (see [26] for uncalibrated scanners). We tackle this rigid registration problem (recovery of R and $\mathbf{t}$) with the minimum number of fiducials needed and by means of a linear algorithm. Given a single image, and considering the unknown vector $\xi = \left[\mathbf{r}_1{}^\mathsf{T} \ \mathbf{r}_2{}^\mathsf{T} \ \mathbf{t}^\mathsf{T} \right]^\mathsf{T}$, (6.35) becomes

$$\left[{}^I\underline{\mathbf{Q}}_j{}^\mathsf{T} \otimes [{}^0\mathbf{y}_j]_\times \right] S_9 \, \xi = {}^0\mathbf{w}_j, \tag{6.39}$$

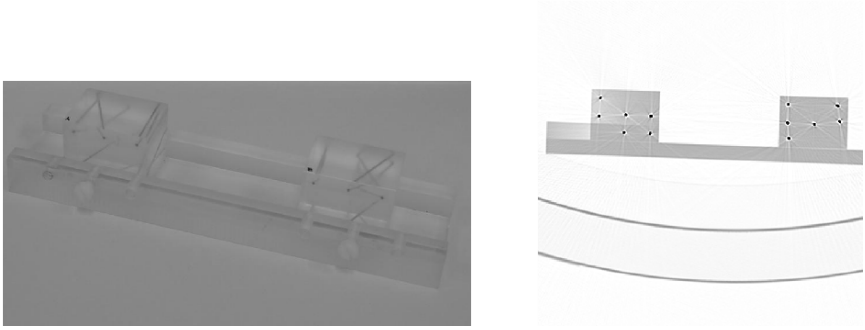
where $S_9 = \begin{bmatrix} {}^\pi S_I{}^\mathsf{T} \otimes I_3 & 0 \\ 0 & I_3 \end{bmatrix}$. With exactly 4 lines/points correspondences, (6.39) is a deficient-rank system which can be solved thanks to rotations properties. Except for some arrangements of the fiducials enumerated in [25], the matrix A has rank 8 when components contain uncorrupted data. However, with noisy data, the rank may be greater than 8. Hence, we wish to enforce the rank value because of the matrix structure (it is built with singular matrices). Therefore, there is a one-parameter family of solutions and (6.39) may be solved with the the Singular Value Decomposition (SVD). We summarize it as follows:

1. Find the SVD of A: $A = UDV^\mathsf{T}$, where the diagonal entries d_i of D are in descending numerical order,
2. Set $\mathbf{b}' = \left[b'_1 \ b'_2 \ \ldots \ b'_9 \right]^\mathsf{T} = U^\mathsf{T} \, \mathbf{b}$ (see (6.38)),
3. Build the vector $\mathbf{z}$ defined by $z_i = b'_i / d_i$, for $i = 1, \cdots, 8$ and $z_9 = 0$,
4. The general solution is $\xi = S_9^{-1} \ (V\mathbf{z} \ + \ \gamma \, \mathbf{v}_9)$, where $\mathbf{v}_9$ is the last (rightmost) column of V.
5. Compute the value of scalar γ with the quadratic relations between $\mathbf{r}_1 = \left[\xi_1 \ \xi_2 \ \xi_3 \right]^\mathsf{T}$ and $\mathbf{r}_2 = \left[\xi_4 \ \xi_5 \ \xi_6 \right]^\mathsf{T}$.

Equation (6.39) can be solved provided that all combinations of triplets verify the conditions mentioned in previous section. Because of the presence of noise, R is not exactly a rotation matrix. One may enforce R to be a rotation by computing the SVD, $R = U\Sigma V^\mathsf{T}$ and by setting singular values to 1. If R' is the corrected rotation matrix, it is given by $R' = U \ \mathrm{diag}(1, 1, \det(UV)) \ V^\mathsf{T}$ [12, 40].

A Newton-Raphson (N-R) numerical approach has also been carried out. It uses the initial guesses provided by the above least-squares method (LS) but we do not describe it here (see [26] for details).

6.4.4 Experimental Validation



(a) (b)

Fig. 6.11 (a) The two cubes used for the experiments. - (b) image picked up from a CT helicoidal sequence with image size of (512×512) pixel.

Experiments were conducted with a Siemens Somatom Plus CT scanner and with fiducials composed of two cubes with six rods each (see Fig. 6.11). The relative position of the cubes is constrained to by a guide rail on which the cubes are screwed. Each cube has been calibrated as well as their relative position with a Mitutoyo measuring machine which can achieve a precision better than 10 μm.
We have assessed the accuracy of the relative pose recovery by registering only one fiducial cube at a time, and by computing the relative position and rotation. By doing so, it is possible to verify the consistency of the pose estimation between two coordinate frames with a single image. To this end, a helicoidal sequence has been captured while a constant translation of the table is performed. In Fig. 6.12, we present the estimated position vector between the two cubes, as the orientation is approximately the identity matrix (it differs to the identity matrix by less than 10^{-5} on each component). The registration has been executed for each cube independently and once it has been done for both (with the LS method and the N-R minimization), the relative position and orientation have been computed. The position $\mathbf{T} = \begin{bmatrix} 0 & -118.29 & 0 \end{bmatrix}$ mm was measured during a calibration procedure (orientations are equal for the two cubes' reference frames). Thus, this value (dotted line in Fig. 6.12) can be compared to the estimations.

As shown in Fig. 6.13, the 3-D pose algorithm works very well on experimental data, since the registration of each cube can serve to predict the position of the rods of the other cube. The estimated error bounding-boxes with an assumed spot location error of 0.25 pixel are also represented. As illustrated, all the detected spots are inside a box with boundaries corresponding to 3-D position errors always less than 2 mm.

6.5 Conclusion

In this chapter, we have described some pose estimation problems by means of intra-operative images. We focused the works on endoscopic views for assisted laparoscopy and CT images slices with X-ray scanners for the image guidance in interventional radiology. For vision-based integrated systems used in minimally invasive surgery, we have developed a set of techniques for assisting surgeons in navigating and manipulating the three-dimensional space within the human body. To that purpose, simple geometrical features have been attached to surgical instruments. Alternatively, when the task is sufficiently constrained by the shape of object of interest, we directly solve the pose without artificial markers: it is the case for the 4 degrees of freedom of a cylindrical needle-holder inside the human abdomen.

One path toward safety and reliability is to incorporate all the available video information. Following this issue, the virtual visual servoing has been used to combine both the apparent contour of the instrument and artificial markers in a numerical iterative process.

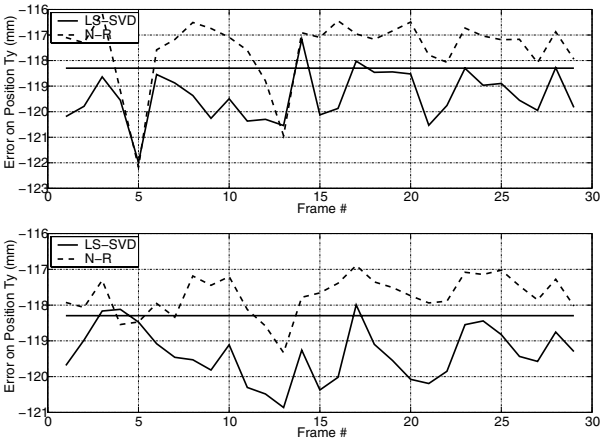


Fig. 6.12 Relative positions between the cubes during the acquisition (slice thickness is 0.5 mm) while translating the table. The first plot (up) is with 5 fiducials for the estimation while the second (down) is with 6 fiducials.

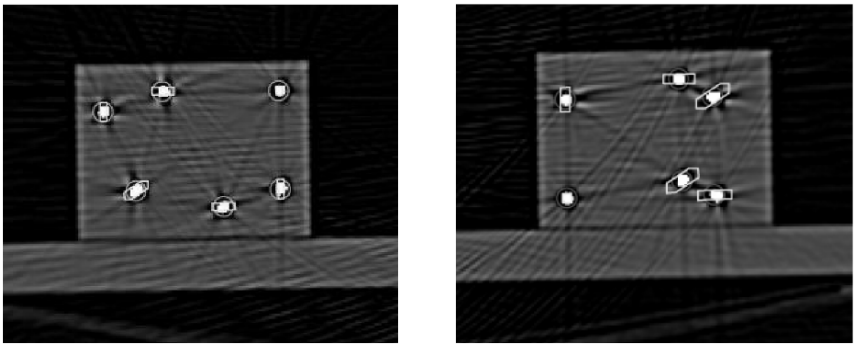


Fig. 6.13 CT images of the cubes. Each drawing box in one cube is corresponding to 2 mm of error bounds computed with the registration of the other cube.

The recovery of out-of-field of view instrument in laparoscopy, the automatic suturing intervention demonstrated in vitro [31] and the positioning of a radio-frequency needle with CT scanners [25] are some applications we contributed for the aforementioned pose problems.

Finally, we believe that significant advances are possible when the geometric information is fused across time and across modality. Furthermore, pre-operative information like the insertion point's localization, the CAD model of instruments, the eye-to-hand calibration or the availability of several statistical atlases of organs can provide some strong constraints on the vision problem. These are crucial factors to achieve reliable dedicated vision systems while compensating small displacements due to patient breathing or any small disturbances which may occur during an image-guided surgical procedure.

Acknowledgements This work has been partially sponsored by the Region Alsace Council. The experimental part of this work has been made possible thanks to the collaboration with the Department of Radiology at the Strasbourg Hospital and the "Institut de Recherche contre les Cancers de l'Appareil Digestif". In particular, we would like to thank Prof. Marescaux, Leroy, Soler and Gangi for their advices, as well as for the use of their facilities.

References

1. A. Casals, J.A., Laporte, E.: Automatic guidance of an assistant robot in laparoscopic surgery. In: IEEE Int'l Conf. on Robotics and Automation, pp. 895–900. Minneapolis, USA (1996)
2. Bouguet, J.Y.: Camera calibration matlab toolbox. MRL - Intel Corp. URL `http:$//www.vision.caltech.edu/bouguetj/calib-doc/$`
3. Bouthemy, P.: A maximum likelihood framework for determining moving edges. IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(5), 499–511 (1989)
4. Breeuwer, M., Zylka, W., Wadley, J., Falk, A.: Detection and correction of geometric distortion in 3D CT/MR images. In: Proc. of Computer Assisted Radiology and Surgery, pp. 11–23. Paris

(2002)
5. Burschka, D., Corso, J.J., Dewan, M., Hager, G., Lau, W., Li, M., Lin, H., Marayong, P., Ramey, N.: Navigating inner space: 3-d assistance for minimally invasive surgery. In: Workshop Advances in Robot Vision, in conjunction with the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 67–78. Sendai, Japan (2004)
6. Climent, J., Mars, P.: Automatic instrument localization in laparoscopic surgery. Electronic Letters on Computer Vision and Image Analysis **4**(1), 21–31 (2004)
7. Dementhon, D., Davis, L.S.: Model-based object pose in 25 lines of code. IJCV **15**(1), 123–141 (1995)
8. Doignon, C., Graebling, P., de Mathelin, M.: Real-time segmentation of surgical instruments inside the abdominal cavity using a joint hue saturation color feature. Real-Time Imaging **11**, 429–442 (2005)
9. Doignon, C., de Mathelin, M.: A degenerate conic-based method for a direct fitting and 3-d pose of cylinders with a single perspective view. In: IEEE Int'l Conf. on Robotics and Automation, pp. 4220–4225. Roma, Italy (2007)
10. Doignon, C., Nageotte, F., de Mathelin, M.: The role of insertion points in the detection and positioning of instruments in laparoscopy for robotic tasks. In: In Proceedings of the Int'l. Conference on Medical Image Computing and Computer-Assisted Intervention - MICCAI, pp. 527–534 (Part I). Copenhaguen, Denmark (2006)
11. Espiau, B., Chaumette, F., Rives, P.: A new approach to visual servoing in robotics. IEEE Trans. Robotics and Automation **8**(3), 313–326 (1992)
12. Goryn, D., Hein, S.: On the estimation of rigid body rotation from noisy data. IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(12), 1219–1220 (1995)
13. Grunert, P., Maürer, J., Müller-Forell, W.: Accuracy of stereotactic coordinate transformation using a localisation frame and computed tomographic imaging. Neurosurgery **22**, 173–203 (1999)
14. Haralick, R., Lee, C., Ottenberg, K., Nlle, M.: Analysis and solutions of the three-point perspective pose estimation problem. In: IEEE Conf. Computer Vision and Pattern Recognition, pp. 592–598. Maui, Hawai, USA (1991)
15. Haralick, R.M., Shapiro, L.G.: Computer and Robot Vision, vol. 1. Addison-Wesley Publishing (1992)
16. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge Univ. Press (2000)
17. Hong, J., Dohi, T., Hashizume, M., Konishi, K., Hata, N.: An ultrasound-driven needle insertion robot for percutaneous cholecystostomy. Journal of Physics in Med. and Bio. pp. 441–455 (2004)
18. Horaud, R., Conio, B., Leboulleux, O., Lacolle, B.: An analytic solution for the perspective 4-point problem. Computer Vision, Graphics, and Image Processing **47**, 33–44 (1989)
19. Hutchinson, S., Hager, G., Corke, P.: A tutorial on visual servo control. IEEE Trans. Robotics and Automation **12**(5), 651–670 (1996)
20. Kragic, D., Christensen, C.: Cue integration for visual servoing. IEEE Trans. on Robotics and Autom. **17**(1), 19–26 (2001)
21. Krupa, A., Chaumette, F.: Control of an ultrasound probe by adaptive visual servoing. In: IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems, vol. 2, pp. 2007–2012. Edmonton, Canada (2005)
22. Krupa, A., Gangloff, J., Doignon, C., de Mathelin, M., Morel, G., Leroy, J., Soler, L., Marescaux, J.: Autonomous 3-d positioning of surgical instruments in robotized laparoscopic surgery using visual servoing. IEEE Trans. on Robotics and Automation, special issue on Medical Robotics **19**(5), 842–853 (2003)
23. Lee, S., Fichtinger, G., Chirikjian, G.S.: Numerical algorithms for spatial registration of line fiducials from cross-sectional images. American Ass. of Physicists in Medicine **29**(8), 1881–1891 (2002)
24. Marchand, E., Chaumette, F.: Virtual visual servoing: a framework for real-time augmented reality. In: Proceedings of the EUROGRAPHICS Conference, vol. 21 (3 of Computer Graphics Forum), pp. 289–298. Saarbrücken, Germany (2002)

25. Maurin, B.: Conception et réalisation d'un robot d'insertion d'aiguille pour les procédures percutanées sous imageur scanner. Ph.D. thesis, Louis Pasteur University, France (2005)
26. Maurin, B., Doignon, C., de Mathelin, M., Gangi, A.: Pose reconstruction from an uncalibrated computed tomography imaging device. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Madison, WN, U.S.A. (2003)
27. Maurin, B., Gangloff, J., Bayle, B., de Mathelin, M., Piccin, O., Zanne, P., Doignon, C., Soler, L., Gangi, A.: A parallel robotic system with force sensors for percutaneous procedures under ct-guidance. In: Int'l Conf. on Medical Image Computing and Computer-Assisted Intervention. St Malo, France (2004)
28. Maybank, S.: The cross-ratio and the j-invariant. Geometric invariance in computer vision pp. 107–109 (1992)
29. Nageotte, F.: Contributions à la suture assistée par ordinateur en chirurgie mini-invasive. Ph.D. thesis, Louis Pasteur University, France (2005)
30. Nageotte, F., Doignon, C., de Mathelin, M., Zanne, P., Soler, L.: Circular needle and needle-holder localization for computer-aided suturing in laparoscopic surgery. In: SPIE Medical Imaging, pp. 87–98. San Diego, USA (2005)
31. Nageotte, F., Zanne, P., Doignon, C., de Mathelin, M.: Visual servoing-based endoscopic path following for robot-assisted laparoscopic surgery. In: IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems, pp. 2364–2369. Beijin, China (2006)
32. Papanikolopoulos, N., Khosla, P., Kanade, T.: Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. IEEE Trans. Rob. and Autom. **9**(1), 14–35 (1993)
33. Quan, L., Lan, Z.: Linear n-point camera pose determination. IEEE Transactions on PAMI **21**(8) (1999)
34. Sundareswaran, V., Behringer, R.: Visual-servoing-based augmented reality. In: In Proceedings of the IEEE Int'l. Workshop on Augmented Reality. San Francisco, USA (1998)
35. Susil, R.C., Anderson, J.H., Taylor, R.H.: A single image registration method for CT guided interventions. In: Proceedings of the Second Int'l Conf. on Medical Image Computing and Computer-Assisted Intervention, pp. 798–808. Cambridge, UK (1999)
36. Taylor, R., Funda, J., LaRose, D., Treat, M.: A telerobotic system for augmentation of endoscopic surgery. In: IEEE Int'l Conf. on Engineering in Med. and Bio., pp. 1054–1056. Paris, France (1992)
37. Tommasini, T., Fusiello, A., Trucco, E., Roberto, V.: Making good features track better. In: IEEE Int'l Conf. on Computer Vision and Pattern Recognition, pp. 178–183. Santa Barbara, USA (1998)
38. Tonet, O., Ramesh, T., Megali, G., Dario, P.: Image analysis-based approach for localization of endoscopic tools. In: Surgetica'04, pp. 221–228. Chambery, France (2005)
39. Tsai, R.: A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the shelf tv cameras and lenses. IEEE Journal of Robotics and Automation **3**(4), 323–344 (1987)
40. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on PAMI **13**(4), 376–380 (1991)
41. Wang, Y.F., Uecker, D.R., Wang, Y.: A new framework for vision-enabled and robotically assisted minimally invasive surgery. Journal of Computerized Medical Imaging and Graphics **22**, 429–437 (1998)
42. Wei, G.Q., Arbter, K., Hirzinger, G.: Automatic tracking of laparoscopic instruments by color-coding. In: S. Verlag (ed.) Proc. First Int. Joint Conf. CRVMed-MRCAS'97, pp. 357–366. Grenoble, France (1997)
43. Zhang, X., Payandeh, S.: Application of visual tracking for robot-assisted laparoscopic surgery. Journal of Robotic systems **19**(7), 315–328 (2002)

Chapter 7
A Sliding Window Filter for Incremental SLAM

Gabe Sibley, Larry Matthies and Gaurav Sukhatme

7.1 Introduction

This work develops a sliding window filter for incremental simultaneous localization and mapping (SLAM) that focuses computational resources on accurately estimating the immediate spatial surroundings using a sliding time window of the most recent sensor measurements. Ideally, we would like a constant time algorithm that closely approximates the all-time maximum-likelihood estimate as well as the minimum variance Cramer Rao lower bound (CRLB) - that is we would like an estimator that achieves some notion of statistical optimality (quickly converges), efficiency (quickly reduces uncertainty) and consistency (avoids over-confidence). To this end we give a derivation of the SLAM problem from the Gaussian non-linear least squares optimization perspective. We find that this results in a simple, yet general, take on the SLAM problem; we think this is a useful contribution.

Our approach is inspired by the results from the photogrammetry community, dating back to the late 1950's [1], and later derivatives like Mikhail's least squares treatment [6], the Variable state dimension filter(VSDF) [5], visual odometry(VO) [4], modern bundle adjustment(BA) [10, 3] and of course extended Kalman filter (EKF) SLAM [9].

We apply the sliding window filter to SLAM with stereo vision and inertial measurements. Experiments show that the best approximate method comes close to matching the performance of the optimal estimator while attaining constant time complexity - empirically, it is often the case that the difference in their performance is indistinguishable.

Gabe Sibley and Gaurav Sukhatme
Robotic and Embedded Systems Laboratory, University of Southern California, Los Angeles California e-mail: gsibley$|$gaurav@usc.edu

Gabe Sibley and Larry Matthies
Jet Propulsion Laboratory, California Institute of Technology in Pasadena California e-mail: gsibley$|$lhm@jpl.nasa.gov

7.2 Non-linear Least Squares SLAM

Standard statistical point estimation is a useful tool for understanding the basic structure of the SLAM problem. Non-linear least squares is appealing for a number of reasons. First, because it emphasizes the fundamental minimization principle at work in least squares, which, we would argue, is a principle that is more difficult to see from the recursive estimation perspective. Second, starting with the underlying probability density functions that describe our problem, it clearly shows the basic probabilistic nature of SLAM - that is, SLAM is simply tracking a normal distribution through a large state space; a state space that changes dimension as we undertake the fundamental probabilistic operations of removing parameters via marginalization, and adding parameters via error propagation and conditioning. Another reason to derive SLAM via statistical point estimation is because it exposes a rich body of theory about the convergence of non-linear least squares estimators. With this in mind, we carry forward in the usual way, by describing the system state vector, process model, measurement model and how we incorporate prior information.

7.2.1 Parameterization

The parameter vector is a temporal sequence of robot poses $\mathbf{x}_{\mathbf{p}_j}$, $1 \leq j \leq m$, and 3D landmark positions $\mathbf{x}_{\mathbf{m}_i}$, $1 \leq i \leq n$.

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_\mathbf{p} \\ \mathbf{x}_\mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{\mathbf{p}_1} \\ \vdots \\ \mathbf{x}_{\mathbf{p}_m} \\ \mathbf{x}_{\mathbf{m}_1} \\ \vdots \\ \mathbf{x}_{\mathbf{m}_n} \end{bmatrix}$$

A pose $\mathbf{x}_{\mathbf{p}_j}$ at time index j is represented by a six parameter column vector comprised of a 3D point and an Euler angle $\mathbf{x}_{\mathbf{p}_j} = [x_{\mathbf{p}_j}\, y_{\mathbf{p}_j}\, z_{\mathbf{p}_j}\, r_{\mathbf{p}_j}\, p_{\mathbf{p}_j}\, q_{\mathbf{p}_j}]^T$. Map landmarks are represented by their 3D position, $\mathbf{x}_{\mathbf{m}_i} = [x_{\mathbf{m}_i}\, y_{\mathbf{m}_i}\, z_{\mathbf{m}_i}]^T$. The state dimension is thus $|x| = (6m + 3n)$ and grows as the robot path increases and as new landmarks are observed.

7.2.2 Kinematic Process Model

The process model $f_j : \mathbb{R}^6 \rightarrow \mathbb{R}^6$ for a single step describes each pose in terms of the previous pose

$$\mathbf{x}_{\mathbf{p}_j} = f_j(\mathbf{x}_{\mathbf{p}_{j-1}}, \mathbf{u}_j) + \mathbf{w}_j \tag{7.1}$$

where $\mathbf{u}_j$ is an input command to the robot. The noise vector $\mathbf{w}_j$ is additive and follows a normal distribution $\mathbf{w}_j \sim \mathcal{N}(0, \mathbf{Q}_j)$. We also assume it is reasonable to have $\mathbf{x}_{\mathbf{p}_j} \sim \mathcal{N}(f_j(\mathbf{x}_{\mathbf{p}_{j-1}}, \mathbf{u}_j), \mathbf{Q}_j)$. A simple and useful kinematic process model for f_j is the compound operation, $\oplus$, which is described in [9]. The 6×6 Jacobian of f_j, $\mathbf{F}_j = \left.\frac{\partial f_j}{\partial \mathbf{x}_{\mathbf{p}_j}}\right|_{\mathbf{x}_{\mathbf{p}_j}, \mathbf{u}_{j+1}}$, which we will need in a moment, is also derived in [9]. Concatenating individual process models together, the p.d.f. describing the robot path, $\mathbf{x}_\mathbf{p} = [\mathbf{x}_{\mathbf{p}_1}^T, ..., \mathbf{x}_{\mathbf{p}_m}^T]^T$, is $p(\mathbf{x}_\mathbf{p}) = \mathcal{N}(\mu_\mathbf{p}, \mathbf{Q})$, where

$$\mu_\mathbf{p} = f(\mathbf{x}) = \begin{bmatrix} \mathbf{x}_{\mathbf{p}_1} \\ f_1(\mathbf{x}_{\mathbf{p}_1}, \mathbf{u}_2) \\ \vdots \\ f_m(\mathbf{x}_{\mathbf{p}_{m-1}}, \mathbf{u}_m) \end{bmatrix}, \; \mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 & & \\ & \ddots & \\ & & \mathbf{Q}_m \end{bmatrix}.$$

In practice, one usually extends this basic model to also estimate other quantities, such as linear and angular velocities; for clarity, we will stick with this basic kinematic formulation.

7.2.3 Sensor Model

We say a measurement of the i^{th} landmark taken from the j^{th} pose is related to the state vector by the sensor model $h_{ij} : \mathbb{R}^{|\mathbf{x}_{\mathbf{m}_i}|+|\mathbf{x}_{\mathbf{p}_j}|} \rightarrow \mathbb{R}^{|\mathbf{z}_{ij}|}$

$$\mathbf{z}_{ij} = h_{ij}(\mathbf{x}_{\mathbf{m}_i}, \mathbf{x}_{\mathbf{p}_j}) + \mathbf{v}_{ij} \tag{7.2}$$

which generates the expected value the sensor will return when landmark i is observed from pose j. We assume $\mathbf{v}_{ij} \sim \mathcal{N}(0, \mathbf{R}_{ij})$ so that $\mathbf{z}_{ij} \sim \mathcal{N}(h_{ij}, \mathbf{R}_{ij})$, where $\mathbf{R}_{ij}$ is the observation error covariance matrix.

Lumping all the observations, measurement functions and measurement covariances together we write $\mathbf{z}$, h, and $\mathbf{R}$ as

$$\mathbf{z} = \begin{bmatrix} \mathbf{z}_{11} \\ \vdots \\ \mathbf{z}_{1m} \\ \vdots \\ \mathbf{z}_{nm} \end{bmatrix}, \; h(\mathbf{x}) = \begin{bmatrix} h_{11} \\ \vdots \\ h_{1m} \\ \vdots \\ h_{nm} \end{bmatrix}, \; \mathbf{R} = \begin{bmatrix} \mathbf{R}_{11} & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ & & \mathbf{R}_{1m} & \\ \vdots & & \ddots & \\ 0 & \dots & & \mathbf{R}_{nm} \end{bmatrix}.$$

Treating the process information as observations, we get the measurement likelihood $p(\mathbf{z}, \mathbf{u}|\mathbf{x}) = \mathcal{N}(\mu_z, \Sigma_z)$, where

$$\mu_z = \begin{bmatrix} h(\mathbf{x}) \\ f(\mathbf{x}) \end{bmatrix}, \; \Sigma_z = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & \mathbf{Q} \end{bmatrix}.$$

7.2.4 Point Estimation

Suppose we are also given *prior information* about the first pose and the map, $p(\mathbf{x}_\pi) = \mathcal{N}\left(\mu_\pi, \Pi^{-1}\right)$, where

$$\mathbf{x}_\pi = \begin{bmatrix} \mathbf{x}_{\mathbf{p}_1} \\ \mathbf{x}_{\mathbf{m}} \end{bmatrix}, \; \mu_\pi = \begin{bmatrix} \hat{\mathbf{x}}_{\mathbf{p}_1} \\ \hat{\mathbf{x}}_{\mathbf{m}} \end{bmatrix}, \; \Pi = \begin{bmatrix} \Pi_{\mathbf{p}_1} & \Pi_{\mathbf{pm}} \\ \Pi_{\mathbf{pm}}^T & \Pi_{\mathbf{m}} \end{bmatrix}.$$

This prior encodes information about a single starting pose, about some previously known map of n landmarks, and about the relationships between the starting pose and the map; $\Pi_{\mathbf{p}}$ is the 6×6 initial pose information matrix, $\Pi_{\mathbf{m}}$ is $3n \times 3n$ map prior information matrix, and $\Pi_{\mathbf{pm}}$ is the $6 \times 3n$ pose-map information matrix.

Armed with the above we can now write the posterior probability of the system,

$$p(\mathbf{x}|\mathbf{z}, \mathbf{u}) = p(\mathbf{z}, \mathbf{u}|\mathbf{x}) p(\mathbf{x}). \tag{7.3}$$

We wish to compute the *maximum a posteriori* estimate of $\mathbf{x}$ which maximizes this density. First, it helps if we lump the sensor model, process model, and prior information terms together by defining the function $g(\mathbf{x})$ and matrix $\mathbf{C}$ as

$$g(\mathbf{x}) = \begin{bmatrix} g_z(\mathbf{x}) \\ g_f(\mathbf{x}) \\ g_\pi(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \mathbf{z} - h(\mathbf{x}) \\ \mathbf{x}_{\mathbf{p}} - f(\mathbf{x}_{\mathbf{p}}) \\ \begin{bmatrix} \hat{\mathbf{x}}_{\mathbf{p}_1} \\ \hat{\mathbf{x}}_{\mathbf{m}} \end{bmatrix} - \begin{bmatrix} \mathbf{x}_{\mathbf{p}_1} \\ \mathbf{x}_{\mathbf{m}} \end{bmatrix} \end{bmatrix}, \qquad \mathbf{C}^{-1} = \begin{bmatrix} \mathbf{R}^{-1} & 0 & 0 \\ 0 & \mathbf{Q}^{-1} & 0 \\ 0 & 0 & \Pi \end{bmatrix};$$

then by taking the negative logarithm of (7.3) we get a proportional non-linear least squares problem

$$\ell(\mathbf{x}) = \frac{1}{2} \left(g(\mathbf{x})^T \mathbf{C}^{-1} g(\mathbf{x}) \right). \tag{7.4}$$

Letting $\mathbf{S}^T\mathbf{S} = \mathbf{C}^{-1}$ and $r(\mathbf{x}) = \mathbf{S}g(\mathbf{x})$ then (7.4) is clearly a non-linear least squares problem of the form

$$\ell(\mathbf{x}) = \frac{1}{2}||r(\mathbf{x})||^2. \tag{7.5}$$

Newton's solution to such optimization problems is the iterative sequence

$$\mathbf{x}_{i+1} = \mathbf{x}_i - (\nabla^2\ell(\mathbf{x}_i))^{-1}\nabla\ell(\mathbf{x}_i). \tag{7.6}$$

For small residual problems a useful approximation to (7.6) is the Gauss-Newton method, which approximates the Hessian $\nabla^2\ell(\mathbf{x}_i)$ by $r'(\mathbf{x}_i)^T r'(\mathbf{x}_i)$. Thus, since the gradient of (7.5) is $\nabla\ell(\mathbf{x}_i) = r'(\mathbf{x}_i)^T r(\mathbf{x}_i)$, the Gauss-Newton method defines the sequence of iterates [2]

$$\mathbf{x}_{i+1} = \mathbf{x}_i - (r'(\mathbf{x}_i)^T r'(\mathbf{x}_i))^{-1} r'(\mathbf{x}_i)^T r(\mathbf{x}_i) \tag{7.7}$$

Noting that $r'(\mathbf{x}_i) = \mathbf{S}\mathbf{G}_i$ where $\mathbf{G}_i$ is the Jacobian of $g(\mathbf{x}_i)$, (7.7) becomes

$$\delta\mathbf{x}_i = (\mathbf{G}_i^T\mathbf{C}^{-1}\mathbf{G}_i)^{-1}\mathbf{G}_i^T\mathbf{C}^{-1}g(\mathbf{x}_i). \tag{7.8}$$

such that $\mathbf{x}_{i+1} = \mathbf{x}_i + \delta\mathbf{x}_i$. When iterated, this sequence is locally q-quadratically convergent to the MAP estimate for near zero-residual problems [2]. The system of linear equations

$$\mathbf{G}_i^T\mathbf{C}^{-1}\mathbf{G}_i\delta\mathbf{x}_i = \mathbf{G}_i^T\mathbf{C}^{-1}g(\mathbf{x}_i) \tag{7.9}$$

is the essential least squares form of the SLAM problem (we will often omit the iteration index). The difference between many SLAM algorithms can be boiled down to differences in how these equations are solved. It is also interesting to note here that for many problems the Gauss-Newton method is algebraically identical to the iterated extended Kalman filter (IEKF).

7.3 Sparsity in the System Equations

Before describing the sliding window filter it is useful to take a look at the overall structure of the SLAM least squares equations, and to study how this structure lends itself to various algebraic solutions.

Expanding the Jacobian $\mathbf{G}$,

$$\mathbf{G} = \begin{bmatrix} \frac{\partial g_z}{\partial \mathbf{x}} \\ \frac{\partial g_f}{\partial \mathbf{x}} \\ \frac{\partial g_\pi}{\partial \mathbf{x}} \end{bmatrix} = - \begin{bmatrix} \mathbf{H} \\ \mathbf{D} \\ \mathbf{L} \end{bmatrix},$$

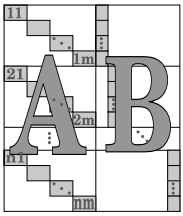


Fig. 7.1 Basic structure of the sensor model Jacobian, $\mathbf{H}$.

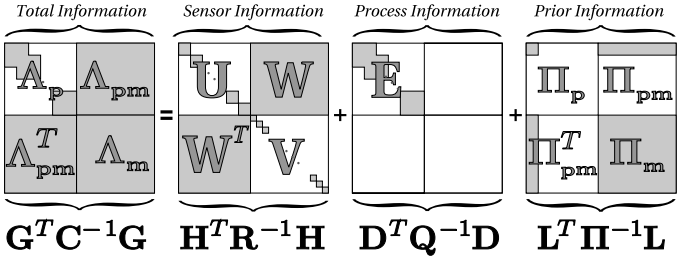


Fig. 7.2 The sparse structure of least squares SLAM system matrix is due to contributions from three components: the measurement block $\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}$, the process block $\mathbf{D}^T\mathbf{Q}^{-1}\mathbf{D}$, and the prior information block $\mathbf{L}^T\Pi\mathbf{L}$.

we see that the system matrix, $\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G} = \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \mathbf{D}^T\mathbf{Q}^{-1}\mathbf{D} + \mathbf{L}^T\Pi\mathbf{L}$, has a sparse structure. The structure of $\mathbf{H}$ is shown in Fig. 7.1. The sparsity pattern of least squares SLAM system matrix is due to contributions from the three components

$$\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} = \begin{bmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V} \end{bmatrix}, \; \mathbf{D}^T\mathbf{Q}^{-1}\mathbf{D} = \begin{bmatrix} \mathbf{E} & 0 \\ 0 & 0 \end{bmatrix}, \text{ and } \mathbf{L}^T\Pi\mathbf{L} = \begin{bmatrix} \Pi_{\mathbf{p}} & 0 & \dots & \Pi_{\mathbf{pm}} \\ 0 & 0 & & 0 \\ \vdots & 0 & \ddots & \vdots \\ \Pi_{\mathbf{pm}}{}^T & 0 & \dots & \Pi_{\mathbf{m}} \end{bmatrix}$$

where $\mathbf{U} = \mathbf{A}^T\mathbf{R}^{-1}\mathbf{A}$, $\mathbf{W} = \mathbf{A}^T\mathbf{R}^{-1}\mathbf{B}$ and $\mathbf{V} = \mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}$. The block tri-diagonal process matrix is

$$\mathbf{E} = \begin{bmatrix} \mathbf{Q}_1^{-1} + \mathbf{F}_1\mathbf{Q}_2^{-1}\mathbf{F}_1^T & -\mathbf{F}_1^T\mathbf{Q}_2^{-1} & 0 & \dots & 0 \\ -\mathbf{Q}_2^{-1}\mathbf{F}_1 & \mathbf{Q}_2^{-1} + \mathbf{F}_2\mathbf{Q}_3^{-1}\mathbf{F}_2^T & \ddots & & \vdots \\ 0 & \ddots & \ddots & & 0 \\ \vdots & & & \mathbf{Q}_{m-1}^{-1} + \mathbf{F}_{m-1}\mathbf{Q}_m^{-1}\mathbf{F}_{m-1}^T & -\mathbf{F}_{m-1}^T\mathbf{Q}_m^{-1} \\ 0 & \dots & 0 & -\mathbf{Q}_m^{-1}\mathbf{F}_{m-1} & \mathbf{Q}_m^{-1} \end{bmatrix}.$$

This structure is also depicted graphically in Fig. 7.2. The task is to solve the system of normal equations 7.7 which expand to

$$\begin{bmatrix} \Lambda_p & \Lambda_{pm} \\ \Lambda_{pm}{}^T & \Lambda_m \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_{\mathbf{p}} \\ \delta\mathbf{x}_{\mathbf{m}} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\mathbf{p}} \\ \mathbf{g}_{\mathbf{m}} \end{bmatrix}$$

where $\mathbf{g}_{\mathbf{p}}$ and $\mathbf{g}_{\mathbf{m}}$ are the least squares RHS vector corresponding to the robot path and map, respectively. We solve this system of equations using elementary matrix operations - for example the Schur complement - to *reduce* the lower right map block Λ_m onto the upper left process block Λ_p

$$\begin{bmatrix} \Lambda_p - \Lambda_{pm}(\Lambda_m)^{-1}\Lambda_{pm}{}^T & 0 \\ \Lambda_{pm}{}^T & \Lambda_m \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_{\mathbf{p}} \\ \delta\mathbf{x}_{\mathbf{m}} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\mathbf{p}} - \Lambda_{pm}(\Lambda_m)^{-1}\mathbf{g}_{\mathbf{m}} \\ \mathbf{g}_{\mathbf{m}} \end{bmatrix}$$

which is solved directly for $\delta\mathbf{x_p}$ and then for $\delta\mathbf{x_m}$ by back-substitution:

$$\begin{aligned}\delta\mathbf{x_p} &= (\Lambda_p - \Lambda_{pm}(\Lambda_m)^{-1}{\Lambda_{pm}}^T)^{-1}(g_p - \Lambda_{pm}(\Lambda_m)^{-1}g_\mathbf{m}) \\ \delta\mathbf{x_m} &= (\Lambda_m)^{-1}(g_\mathbf{m} - {\Lambda_{pm}}^T\delta\mathbf{x_p})\end{aligned}$$

Alternately, we can also reduce the upper left process block Λ_p onto the lower right map block Λ_m

$$\begin{bmatrix}\Lambda_p & \Lambda_{pm} \\ 0 & \Lambda_m - {\Lambda_{pm}}^T(\Lambda_p)^{-1}\Lambda_{pm}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x_p} \\ \delta\mathbf{x_m}\end{bmatrix} = \begin{bmatrix}\mathbf{g_p} \\ \mathbf{g_m} - {\Lambda_{pm}}^T(\Lambda_p)^{-1}\mathbf{g_p}\end{bmatrix}$$

giving the solution

$$\begin{aligned}\delta\mathbf{x_m} &= (\Lambda_m - {\Lambda_{pm}}^T(\Lambda_p)^{-1}\Lambda_{pm})^{-1}(g_\mathbf{m} - {\Lambda_{pm}}^T(\Lambda_p)^{-1}g_p) \\ \delta\mathbf{x_p} &= (\Lambda_{pm})^{-1}(g_p - \Lambda_{pm}\delta\mathbf{x_m})\end{aligned}$$

Depending on the process noise and the prior, the system matrix $\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G}$ can take on different sparsity patterns that affect the complexity of finding a solution. In the field, the problem at hand will define the sparsity pattern, which will influence the choice of which algorithm to use.

7.4 The Sliding Window Filter

To keep the complexity of the filter constant with the number of landmarks it is necessary to reduce the size of the state vector. This is accomplished by removing the oldest pose parameters and distant landmark parameters. If we directly remove parameters from the system equation however, we can lose information about how the parameters interact. The right way to remove parameters from a multi-dimensional normal distribution is to marginalize them out.

7.4.1 The Effects of Marginalizing Out Parameters

Marginalizing out a set of pose parameters will add cross-information terms in the SLAM least squares system matrix (that is the Hessian, or information matrix) between all the landmarks that were conditionally dependent on those parameters. This is depicted graphically in Fig. 7.3 for a system that starts *without* any prior information. Studying this structure we see that downdating the oldest pose causes fill-in in three places: 1) between any landmarks that were visible from the downdated pose, 2) between the parameters of the next-oldest-pose (the pose one time step after the pose being downdated), and 3) between the next-oldest-pose and all landmarks seen by the downdated pose. Interestingly, Π is the only place that ever suffers from fill-in. Because of this structure, when solving we can still take advantage of any

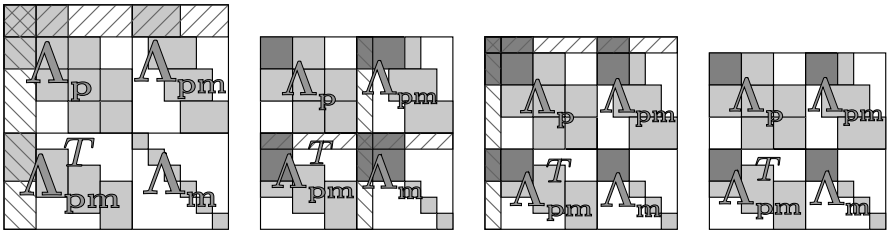


Fig. 7.3 Information matrix evolution for an example problem with 4 poses and 6 landmarks. The left image is after measuring landmarks 1, 2, 3 from pose 1, landmarks 2, 3, and 4 from pose 2, landmarks 3, 4, and, 5 from pose 3 and 4, 5, and 6 at pose 4. In the second image from left we see that marginalizing out pose 1 induces conditional dependencies (fill-in) in three places: 1) the top left 6×6 of the process-block, 2) the prior map-block Π_m between landmarks that were visible from pose 1, and 3) the prior pose-to-map block Π_{pm} between landmarks that were visible from pose 1. These places are shaded in darker grey. At this point (second from right image) downdating landmark 1, which is not visible from any of the remaining poses, will induce no extra fill-in in Π (right image).

sparsity patterns in Λ_{pm} and Λ_p. It is important to note that the Π term catches all the prior information as we "roll" up old state parameters. If we were to ignore Π, we would not benefit from past measurements. Marginalizing out landmarks that are not visible from any active pose will also only ever cause fill in Π.

Marginalizing out poses at a fixed rate and landmarks when they lose support results in a constant time complexity incremental SLAM estimation algorithm. By choosing when to downdate poses and landmarks sliding window SLAM can scale from the full batch solution, to the extended Kalman filter solution. That these algorithms are subsumed within one framework testifies to the generality of the simple least squares approach.

It is interesting to note what happens if we simply delete parameters from the estimator instead of marginalizing them out. For a sliding window of size k, the error converges like $1/k$ just as we would expect the batch estimator to do. However, after k steps, the error stops converging as we delete information from the back of the filter. With such deleting and a sliding window of $k = 2$ it is interesting to note that we end up with a solution that is nearly identical to previous forms of Visual Odometry [4, 7, 8]. The graph in Fig. 7.4 shows the average RMS mapping error for this type of Visual Odometry compared to the batch solution, as well as the sliding window filter solution.

7.5 Conclusions

This chapter describes a SLAM solution that concentrates computational resources on accurately estimating the immediate spatial surroundings by using a sliding time window of the most recent sensor measurements. Focusing computation on improving the local result is crucial for applications that wish to fuse spatially high-

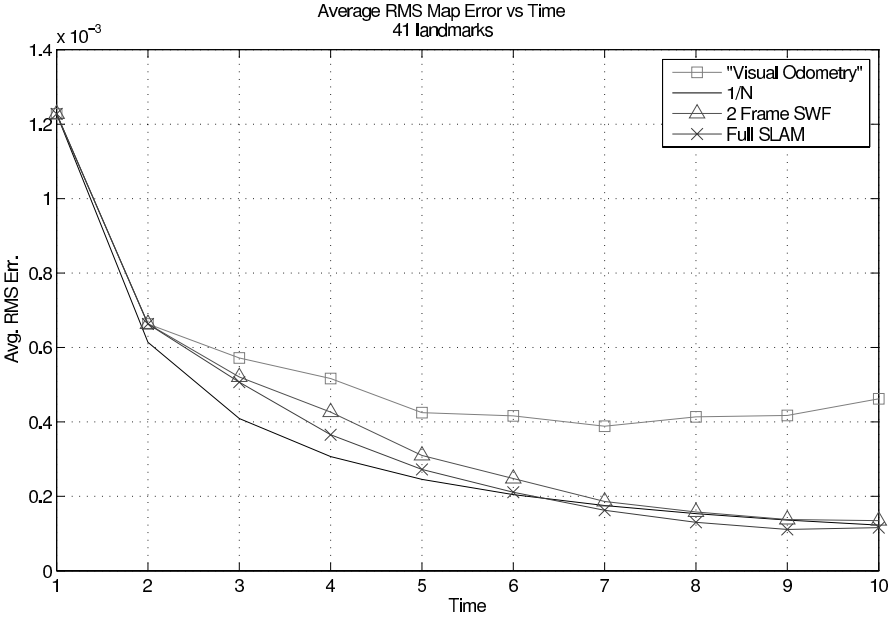


Fig. 7.4 Graph showing average RMS mapping error. Each curve is a trial for different size time window, averaged over 50 Monte-Carlo trials, with 0.1 pixel std. dev. measurement noise. 1.0m std. dev. process noise. Note that the sliding window filter comes close to the full SLAM solution. A sliding window of 2 is close to optimal $1/k$ full batch curve. Further, because VO does not combine information over time, it does not reduce uncertainty as time passes.

resolution, dense structure estimates. With high bandwidth sensors (like cameras) this is clearly beneficial for computational reasons, and it especially true if we wish to fuse all of the sensor data (or a significant portion thereof).

By tuning a few parameters, the sliding window algorithm can scale from exhaustive batch solutions to fast incremental solutions. Ideally, we would like a constant time algorithm that closely approximates the all-time maximum-likelihood estimate as well as the minimum variance Cramer Rao Lower Bound - that is, we would like an estimator that achieves some notion of statistical optimality (quickly converges), efficiency (quickly reduces uncertainty) and consistency (avoids over-confidence). We find that approaching this problem from the statistical point estimation point of view results in a simple, yet general, take on the SLAM problem; we think this is a useful contribution. Data-fusion is fundamental for improving a robot's metric estimation of the world. Doing it quickly and with large amounts of data is a challenging task. Ultimately, some form of dense data-fusion will enable accurate high-resolution spatial perception for autonomous robots.

Acknowledgements This work is supported in part by Caltech/JPL under contract 1277958, and by NSF grants IIS-0133947 and CCR-0120778.

References

1. Brown, D.: A solution to the general problem of multiple station analytical stereotriangulation. Tech. rep., RCP-MTP Data Reduction Technical Report No. 43, Patrick Air Force Base, Flordia (also designated as AFMTC 58-8) (1958)
2. Dennis, J.J., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Society for Industrial & Applied Mathematics (1996)
3. Engels, C., Stewenius, H., Nister, D.: Bundle adjustment rules. In: Photogrammetric Computer Vision (2006)
4. Matthies, L., Shafer, S.: Error modelling in stereo navigation. IEEE Journal of Robotics and Automation **3**(3), 239–248 (1987)
5. McLauchlan, P.F.: The variable state dimension filter applied to surface-based structure from motion. Tech. rep., University of Surrey (1999)
6. Mikhail, E.M.: Observations and Least Squares. Rowman & Littlefield (1983)
7. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–659. Washington, DC (2004)
8. Olson, C.F., Matthies, L.H., Schoppers, M., Maimone, M.W.: Stereo ego-motion improvements for robust rover navigation. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 1099–1104. Washington, DC (2001)
9. Smith, R.C., Self, M., Cheeseman, P.: Estimating uncertain spatial relationships in robotics. In: I.J. Cox, G.T. Wilfong (eds.) Autonomous Robot Vehicles, pp. 167–193. Springer-Verlag (1990)
10. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment – A modern synthesis. In: W. Triggs, A. Zisserman, R. Szeliski (eds.) Vision Algorithms: Theory and Practice, LNCS, pp. 298–375. Springer Verlag (2000)

Chapter 8
Topological and Metric Robot Localization through Computer Vision Techniques

A. C. Murillo, J. J. Guerrero and C. Sagüés

8.1 Introduction

Nowadays, robotic applications based on vision sensors have become widespread, but there is still a gap between these applications and the pure computer vision developments. Sometimes this separation can be due to the lack of communication between both research communities or to the divergence in their objectives. Other times this difference is due to the inadequacy of the methods for certain tasks, e.g., there are computer vision methods which can not be applied for robotic tasks due to their computational complexity. However, this can be solved many times just with a slight adaptation of the techniques.

Many works during the last years have developed vision based methods for robotic tasks such as control [5], automatic topological map building [23], topological localization [10], or Simultaneous Localization and Mapping [3]. This work is focused on the application of computer vision techniques for robot global self-localization, a fundamental issue for any autonomous device. Both topological and metric localization are taken into account, as the two of them have huge similarities with computer vision applications. On the one hand, topological localization usually consists of identifying the current location of our mobile device in a higher cognitive level than just metric units, for example identifying the room where the robot currently is. This could also be named room/scene identification. Object recognition is an important issue in computer vision research, with many works and important results in the previous years, e.g. [11], [8] or [19], that could be adapted for scene recognition. For instance, in [12] a room identification technique was presented, that mixes range and camera information and is based on a learning method typically used for object classification/recognition (AdaBoost). On the other hand, the metric localization as well as the Simultaneous Localization and Mapping (SLAM)

A. C. Murillo · J. J. Guerrero · C. Sagüés
University of Zaragoza, I3A - Department of Informatics and Systems Engineering, 50010 Zaragoza, Spain, e-mail: acm@unizar.es

are very similar to the classical computer vision problem of Structure from Motion (SFM). The SFM algorithms provide the camera (or robot) and landmarks location from the required minimum number of multi-view correspondences. Thus, they have the same goal as the SLAM. This has been studied in previous works, e.g, SFM from the 1D trifocal tensor has been proved to improve bearing only SLAM initialization [4], and more recently it has been shown also the utility of SFM methods for the always difficult problem of loop closing [18], in this case using the 2D geometry for image pairs.

This paper explains a vision-based method to obtain both topological and metric localization through a hierarchical process, presented in our previous work [17]. There, global localization is obtained with respect to a visual memory (a topological map built with sorted reference images). The global localization, sometimes known as the "kidnapped robot problem", intends to localize the robot only with the current acquisition of the sensors, without any knowledge of previous measurements, oppositely to the continuous localization tasks. The aforementioned localization hierarchy consists of an initial less accurate localization result, in terms of topological information (room identification), which applies object recognition techniques. The second localization result of the hierarchy is a more accurate metric localization. It is obtained through a SFM algorithm for 1D bearing only data [1], [4] based on the 1D trifocal tensor [6]. This kind of data is intuitively extracted from images, Fig. 8.1 shows two examples of 1D bearing only data. On the left, the orientation of point features in omnidirectional images, that is the more stable cue in that kind of images; on the right, another situation where using only 1D is convenient, the horizontal coordinate of vertical lines in conventional images, as these line segments usually have a clear orientation (x-coordinate) but they do not have accurate tips (y-coordinate).

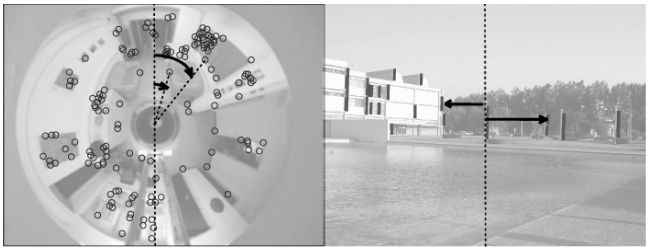


Fig. 8.1 Two examples of 1D bearing only data extracted from images.

The outline of this paper is as follows. Next section 8.2 is divided in two parts: subsection 8.2.1 details the process used to perform the room/scene recognition and subsection 8.2.2 explains the 1D trifocal tensor estimation and its SFM algorithms. In section 8.3, a brief description of the features that have been studied in our examples is given, followed by section 8.4 with several examples of localization results obtained applying the explained techniques. Finally section 8.5 concludes the work.

8.2 Vision-based Hierarchical Localization

This section summarizes the hierarchical localization process developed in [17], including some small improvements in the process and emphasizing the similarities between well-known computer vision tasks and some robotic ones, as well as how these computer vision methods are applied to robot localization.

To perform both topological and metric localization in the same process has several advantages. First of all, both kinds of information are usually necessary, e.g., the topological one is more suitable to interact with users but the metric one is more accurate. The fact of designing a hierarchical process, leaving the computationally expensive steps at the end (those needed for a metric localization), helps to deal efficiently with a big amount of reference images. The diagram in Fig. 8.2 summarizes the hierarchical localization process, whose two main stages are detailed in next subsections.

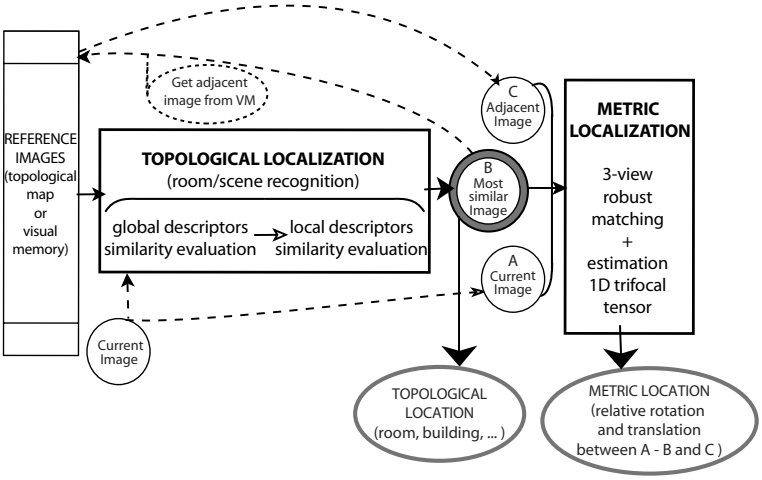


Fig. 8.2 Diagram of the hierarchical localization process.

8.2.1 Object Recognition ⇒ Room/Scene Recognition

Firstly, let us focus in the topological localization, which corresponds to the first stage of the hierarchical process. In our case of study, this topological localization consists of room identification indoors and of building recognition outdoors. The goal is to localize the robot in the available topological map (or visual memory of reference images). In practice, this means to identify which image from the reference set is the most similar to the current view, which will point the topologi-

cal location. In order to obtain this, a similarity evaluation algorithm is run in two stages: a first one using simple global image descriptors, and a second one using local image features.

8.2.1.1 Global Image Descriptors Evaluation

First, a pre-filtering is carried out evaluating a global descriptor, such as color histograms or color invariant moments, which is computed for each image over all its pixels. Then, all reference images are compared with the current view with regard to the chosen global descriptor, and a probability to be the current room/location is estimated for each reference image. Images with lower probability than the established threshold are discarded. This step intends to reject in a fast way as many wrong candidates as possible, with a rough but quick global evaluation of the appearance of the images.

8.2.1.2 Local Image Descriptors Evaluation

After the rough initial step to discard reference images which are unprovable to match the current one, a more detailed similarity measure is obtained evaluating local image features descriptors. Two approaches were studied in our previous work [17] for this task, a typical nearest neighbour (*NN*) based matching and the pyramidal matching method developed in [8], that approximates the optimal correspondences between two given feature sets in linear time with the number of features.

NN-based approach. The first approach for image similarity evaluation is based on a local feature nearest neighbour matching. This process has quadratic (n^2) or $n\ log(n)$ computational cost with the number of features (n), depending on the implementation chosen. We use a typical approximate nearest neighbour implementation that has the lower complexity. After the matching, a probability P_v is estimated for each reference image (v) processed at this stage, which depends on the similarity S_{cv} between the reference image and the current view (c):

$$P_v = e^{\frac{-(1-S_{cv})}{\sigma_s}}, \tag{8.1}$$

being σ_s the variance among the similarity values between all reference images and the current one. In this approach, the similarity measure S_{cv} is obtained from the distance between the two views d_{cv}: $S_{cv} = \frac{1}{d_{cv}+1}$, being $d_{cv} = md_e + Fz$, with m the number of matches, d_e the average Euclidean distance between each pair of matched features, F the number of non matched features and z a penalty for it.

Pyramid-based approach. In the second image similarity evaluation approach based on local features studied, the descriptor sets of all features are used to implement a *pyramid matching kernel* [8]. This implementation consists of building for each image several multi-dimensional histograms (each dimension corresponds to one descriptor), where each feature occupies one of the histogram bins. The value of

each feature descriptor is rounded to the corresponding histogram resolution, which gives the coordinates of the bin corresponding to that feature. Several levels of histograms are defined, and in each level, the size of the bins is increased by powers of two until all the features fall into one bin. The histograms of each image are stored in a vector (pyramid) ψ with different levels of resolution. Once these pyramids are built, the similarity S between two images, the current one (c) and a reference image (v), is obtained from the intersection of the two pyramids of histograms as explained in next eq. (8.2). This operation is quite efficient, with linear complexity in the number of features:

$$S(\psi(c),\psi(v)) = \sum_{i=0}^{L} w_i N_i(\psi(c),\psi(v)) \,, \tag{8.2}$$

with N_i the number of matches between images c and v in level i of the pyramid (features that fall in the same bin in level i of the histograms, see Fig. 8.3). w_i is the weight for the matches in level i and is the inverse of the current bin size (2^i). L is the level where all features fall in the same bin, e.g., Fig. 8.3 example has $L = 3$. This similarity measure is divided by a factor determined by the self-similarity score of each image, in order to avoid giving advantage to images with bigger feature sets, so the *normalized* similarity measure obtained with this approach is

$$S_{cv} = \frac{S(\psi(c),\psi(v))}{\sqrt{S(\psi(c),\psi(c))\; S(\psi(v),\psi(v))}} \,. \tag{8.3}$$

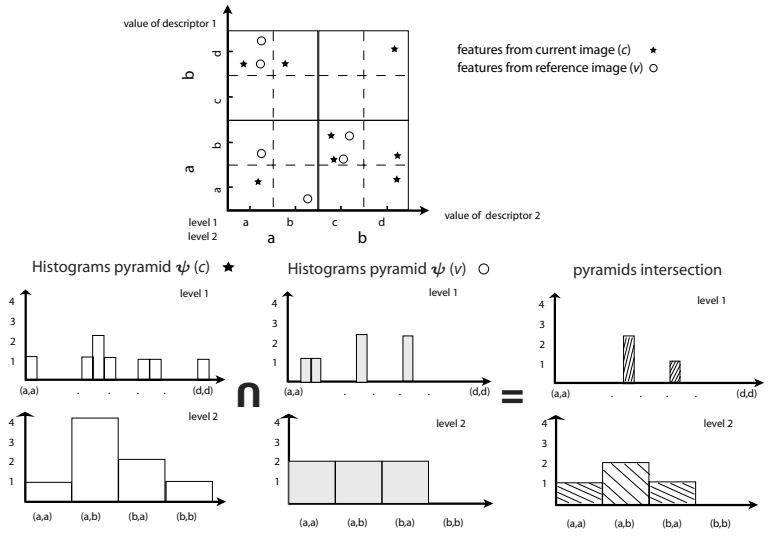


Fig. 8.3 Local Features Pyramids construction and matching (intersection). Top: plot with the features from two images (c and v). Bottom: histograms pyramids (ψ) of both images and their intersection (for graphic simplification, only two levels evaluation and feature descriptor of 2 dimensions).

Similarly to the previous studied approach, a probability for each reference image being the current location is estimated based on this S_{cv} (see eq. 8.1).

Notice that the matches found with this approach are not always individual feature-to-feature matches, as the method just counts how many features fall in the same bin. The more levels we check in the pyramid the bigger the bins are, so the easier it is to get multiple coincidences in the same bin (as it can be seen in Fig. 8.3). Although this matching method can be less accurate, it can also be faster than typical matching methods based on nearest neighbour approaches, so it is very convenient for the current task when it is necessary to deal with big amounts of reference images.

Finally, to obtain the topological localization result, using either similarity evaluation approach shown in this section, we select the reference location (image) with higher probability (P_v) as the current topological location. If our process would finish with this topological localization, it would be convenient to make a robust selection from the x most probable locations, for example imposing a multi-view geometry constraint [9] to theirs sets of correspondences or making a voting process with these images. However, if we carry on the whole hierarchical localization process, the method continues with a last step based pursuing a more accurate metric localization, which is based on a robust estimation of geometry constraints that will be explained in next subsection. Then, if this following robust estimation fails, the process could go back and pick up the reference image with second higher probability.

8.2.2 Structure From Motion (SFM) ⇒ Metric Localization

As previously mentioned, the methods known in computer vision as SFM provide the simultaneous recovery of the robot and landmarks locations from feature correspondences in multiple views [9], i.e., similar goals as in the SLAM problem. The difference could be noticed in the fact that the SLAM methods are continuous processes where the robot integrates the sensor measurements along the time, in order to obtain an accurate metric map of the environment at the end together with the robot current location with regard to that map. However, SFM algorithms are a more instantaneous procedure that gives robot and landmarks location at a certain moment. It does not use any a priori information, therefore it is very convenient for obtaining a global localization, or recovering a lost robot. Applications based on two view geometry have been more frequently studied in computer vision than the case of three views, which could be convenient for robotics for example in the case of using 1D bearing only data. This situation is the subject of this section and is described in Fig. 8.4.

To obtain the metric localization in the case of study, the 1D three view geometry constraint (1D trifocal tensor) has to be computed. This tensor is robustly estimated simultaneously to a robust set of three view feature correspondences, as explained

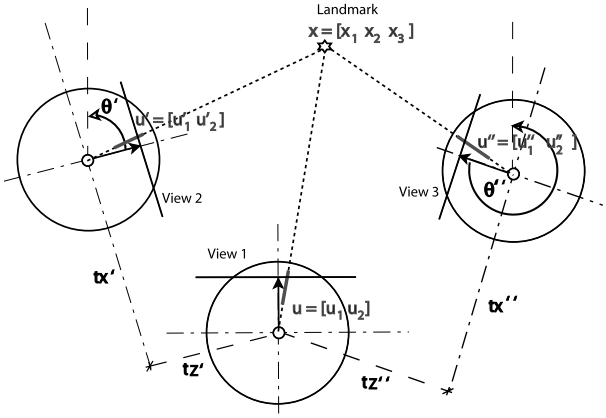


Fig. 8.4 SFM with three-view 1D geometry (the 1D trifocal tensor): with at least five correspondences of bearing-only observations in three views ($\mathbf{u}$, $\mathbf{u}'$, $\mathbf{u}''$), we can estimate the 1D tensor and extract from it the relative location of the robot (θ', θ'', $\mathbf{t}' = [t'_x t'_z]$, $\mathbf{t}'' = [t''_x t''_z]$) and the position of the landmarks $\mathbf{x}$.

in next section 8.2.2.1. Afterwards, the robot and landmarks locations are recovered from the tensor as shown in section 8.2.2.2.

8.2.2.1 Automatic Robust Matching and 1D Trifocal Tensor Computation

The 1D trifocal tensor, $\mathbf{T}$, can be computed as explained in the literature, using the trilinear constraint [6], that relates observations of a landmark in three views (u,u',u''):

$$\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} T_{ijk} u_i u'_j u''_k = 0. \tag{8.4}$$

where T_{ijk} $(i,j,k = 1,2)$ are the eight elements of the $2 \times 2 \times 2$ 1D trifocal tensor.

The minimal number of correspondences varies in different situations. In a general case, at least seven correspondences are required, but if the two calibration constraints from [1] are included in the computations only five matches are needed. A deeper study about the tensor estimation options, and about their performance in robot applications can be found in [20] and [15].

With more matches than the minimum number required, the SVD procedure gives the least squares solution, which assumes that all the measurements can be interpreted with the same model. This is very sensitive to outliers, then robust estimation methods are necessary to avoid those outliers in the process, such as the well known RANSAC [7], which makes a search in the space of solutions obtained from subsets of minimum number of matches. This robust estimation allows to obtain simultaneously the tensor and a robust set of correspondences. It consists of the following steps:

- Extract relevant features in the three views, and perform an automatic matching process to firstly obtain a putative set of matches (*basic matching*), based on the appearance of the features in the image.
- Afterwards, the geometrical constraint imposed by the tensor is included to obtain a *robust matching* set using a RANSAC voting approach. This robust estimation efficiently rejects the outliers from the *basic matching*.
- Optionally, the tensor constraint can help to grow the final set of matches, obtaining new ones with weaker appearance-based similarity but fitting well the geometric constraint.

8.2.2.2 SFM from the 1D Trifocal Tensor

The camera and landmarks location parameters can be computed from the 1D trifocal tensor in a closed form. These parameters can be related to the components of the tensor by developing the elements of the projection matrixes ($\mathbf{M}, \mathbf{M}', \mathbf{M}''$). These matrixes project a 2D feature in homogeneous 2D coordinates, $\mathbf{x} = [x_1, x_2, x_3]^T$), in the $\mathscr{P}^1$ projective space, 1D images, as $\mathbf{u} = [u_1, u_2]^T$:

$$\lambda \mathbf{u} = \mathbf{M}\mathbf{x}, \qquad \lambda' \mathbf{u}' = \mathbf{M}'\mathbf{x}, \qquad \lambda'' \mathbf{u}'' = \mathbf{M}''\mathbf{x}, \tag{8.5}$$

where λ, λ' and λ'' are scale factors.

If we suppose all the 2D features in a common reference frame placed in the first robot location, the projection matrixes relating the scene and the image features are $\mathbf{M} = [\mathbf{I}|\mathbf{0}]$, $\mathbf{M}' = [\mathbf{R}'|\mathbf{t}']$ and $\mathbf{M}'' = [\mathbf{R}''|\mathbf{t}'']$ for the first, second and third location respectively. Here, $\mathbf{R}' = \begin{bmatrix} \cos\theta' & \sin\theta' \\ -\sin\theta' & \cos\theta' \end{bmatrix}$ and $\mathbf{R}'' = \begin{bmatrix} \cos\theta'' & \sin\theta'' \\ -\sin\theta'' & \cos\theta'' \end{bmatrix}$ are the rotations, and $\mathbf{t}' = [t'_x, t'_z]^T$ and $\mathbf{t}'' = [t''_x, t''_z]^T$ are the translations (Fig. 8.4).

We have studied two methods to recover the robot and landmarks locations from these relations: the algorithm presented in [4], which is based on the decomposition of the tensor into two intrinsic homographies [21], and the method from [1]. Both methods give almost identical results, but the SFM algorithm from [4] is a little easier to implement (see Algorithm 8.1). They both provide two symmetric solutions for the location parameters, defined up to a scale for the translations. This twofold ambiguity [1] is one of the drawbacks of using only three views to solve this problem, it can be solved using a fourth view or some additional information such as odometry. Once the relative location of the sensor has been estimated, the location of the landmarks can be obtained by solving the projection equations (8.5) for each landmark [4].

Algorithm 8.1 (Robot Motion from the 1D Trifocal Tensor [4]).

1. Decompose the trifocal tensor (computed for images 1, 2 and 3) into its intrinsic homographies. We get 6 of those homographies, but we need just three to find the epipoles, for example $\mathbf{H}^X_{32}$, $\mathbf{H}^Z_{32}$ and $\mathbf{H}^X_{12}$:

$$\mathbf{H}^X_{32} = \begin{bmatrix} -T_{112} & -T_{122} \\ T_{111} & T_{121} \end{bmatrix} \quad \mathbf{H}^Z_{32} = \begin{bmatrix} -T_{212} & -T_{222} \\ T_{211} & T_{221} \end{bmatrix} \quad \mathbf{H}^X_{12} = \begin{bmatrix} -T_{211} & -T_{221} \\ T_{111} & T_{121} \end{bmatrix}$$

2. Compose an homology (**H**), to reproject the points of one image to the same image. The only points that will stay invariant under this reprojection are the epipoles ($e = \mathbf{H}e$), as they are the eigenvectors of **H**.

$$\mathbf{H} = (\mathbf{H}_{32}^{Z})^{-1} * \mathbf{H}_{32}^{X}$$
$$[\mathbf{e}_{21} \quad \mathbf{e}_{23}] = eigenVectors(\mathbf{H})$$

with $[\mathbf{e}_{21} \quad \mathbf{e}_{23}]$ being the epipoles in the image 2 of the camera 1 and 3 respectively. A second solution will be obtained swapping both epipoles.

3. Project the epipoles in the image 2 to the other cameras using any of the intrinsic homographies

$$\mathbf{e}_{31} = \mathbf{H}_{32}^{X} * \mathbf{e}_{21} \; ; \; \mathbf{e}_{32} = \mathbf{H}_{32}^{X} * \mathbf{e}_{23}$$
$$\mathbf{e}_{12} = \mathbf{H}_{12}^{X} * \mathbf{e}_{21} \; ; \; \mathbf{e}_{13} = \mathbf{H}_{12}^{X} * \mathbf{e}_{23}$$

4. Compute the camera motion from the epipoles as

$$\theta' = \arctan(\tfrac{e_{12}(2)}{e_{12}(1)}) - \arctan(\tfrac{e_{21}(2)}{e_{21}(1)})$$
$$[t_x' \quad t_z'] = scale * [e_{12}(1) \quad e_{12}(2)]^T$$

Those are the motion parameters from image 2 to 1. The parameters from image 3 to 1 (θ'',t_x'' and t_z'') are computed in a similar way, substituting in the expressions above the subindex 2 by 3.

5. Recover landmarks location from the projection equations (8.5) for each landmark $\mathbf{x} = (x_1, x_2, x_3)^T$:

$$\mathbf{u} \times [\mathbf{I}|\mathbf{0}]\mathbf{x} = 0$$
$$\mathbf{u}' \times [\mathbf{R}'|\mathbf{t}']\mathbf{x} = 0$$
$$\mathbf{u}'' \times [\mathbf{R}''|\mathbf{t}'']\mathbf{x} = 0$$

where $\times$ indicates the cross product. They can be explicitly developed to solve the position of the landmarks **x** defined up to an overall scale factor.

8.3 Local Image Features

The main parts of the localization processes explained in previous section are based on the analysis and matching of local image features. Choosing the feature to use is a very important practical issue, the purpose is to find the simplest and fastest feature that provides all the invariant properties required. There are many local features developed in the last years for image analysis, with the outstanding SIFT [11] as the most popular. In the literature, there are several works studying the different features and their descriptors, for instance [14] evaluates the performance of the state of the art in local descriptors, and [13] shows an study on the performance of different features for object recognition.

We have used different features with the developed algorithms in our previous works, to try to evaluate their efficiency for the aimed robotic tasks. The three kind of features used in the experiments in next section are

- Line segments, with their line support regions. We used the extraction method and descriptors explained in [17].

- SIFT (Scale Invariant Feature Transform). The original code provided by D. Lowe [11] was used.
- SURF (Speeded Up Robust Features), a recently developed local feature, whose original extraction method provided by the authors [2] was used as well, which allows a flexible descriptor length. We will use the simpler descriptor, SURF-36, (36 descriptors per feature) for the topological localization, since at that stage speed is more important. However for the metric localization, where higher accuracy is necessary, we will extract the 64-descriptors features.

The following section shows localization experiments using all these features, showing some advantages and disadvantages of using one or another.

8.4 Experiments

This section shows experimental results using the methods explained in this work for robot localization with different image data sets. The data sets used are *Almere* (publicly available [22]) and *data set LV*, which were acquired with omnidirectional vision sensors with hyperbolic mirror and were explained in more detail in [17], and another data set of conventional images, the *data set ZGZ*, that consists of 630 outdoor images from an urban environment.

8.4.1 Topological Localization: Room/Building Recognition

This section presents several results applying the explained methods for room recognition in case of indoor omnidirectional images, and for building/scene recognition in case of outdoor conventional images.

Room recognition with omnidirectional images. In a first topological localization experiment, robot localization is performed with respect to a reference topological map using omnidirectional images. Initially it is necessary to build the reference map, also named visual memory (VM). Here it was built manually, grouping the images in rooms, as its automatic construction was not the case of study. We used both data sets of omnidirectional images mentioned previously (*Almere* and *data set LV*), that where divided in reference images and test images. In case of the first data set, images were frames from robot tour videos, then every 5th even image was used as reference, and every 5th odd image was used as test. From the second data set, as images were already sparser, every image was localized with regard to the rest.

In this experiment, the global descriptor used is based on color invariant moments, similar to those used in [17]. A summary of the room recognition rates obtained for the different local features studied (Sec. 8.3) is shown in Table 8.1. Those results were studied in more detail in [16]. The time information in column Time/Time$_{surf}$ in this experiment is just a comparative of the relative speed of the

localization using each of the three evaluated features. It does not intend to evaluate their maximal speed, note that the experiments were run in Matlab and were not optimized for speed. Then, the surf execution time (Time_{surf}) is taken as reference and the others are relative to it. Column *% Ok* shows the percentage of tests where the image selected as most similar to the current one was correct, and the second column shows the matching approach used in that case. Both of them (NN-based and Pyramidal-based) performed similarly for lines and SURF. However, using SIFT much better results were obtained with the NN-based approach (only a 60% of correct classifications with the Pyramidal based approach while the 85% was obtained with the NN-based one). This result is not surprising, since the Pyramidal matching method is not convenient for features with very long descriptor sets.

Table 8.1 Room recognition results (the number after each feature type shows the length of its descriptor set).

feature used	matching approach	% Ok	$\text{Time}/\text{Time}_{surf}$
lines-22	Pyramidal-based	81%	0.1
surf-36	Pyramidal-based	96%	1
sift-128	NN-based	85%	3

These results are the average results using test images, from both *LV* and *Almere* data sets, that were obtained under similar conditions (illumination, noise, occlusions, ...) than the reference images and they do not include big baselines between images, therefore the performance is acceptable with the three studied features. However, when the test images have higher variances, as shown in some of the experiments in [16], radial lines performance decreases dramatically. As it was also concluded there, the best compromise between correctness in the results and efficiency in the localization process was obtained with the SURF features.

This topological localization approach is quite robust, as we reduced the size of the reference images to the half and the performance stayed similar to the presented results. So reducing the reference image set is not the main problem for the correctness in the topological localization, but it can be a problem for the accuracy in a next step towards the metric localization, because a big variation between reference and current images can makes the robust local feature matching fail, as it will be seen in next section 8.4.2 results.

Building recognition with conventional images. In a second topological localization experiment, we will use only SURF features, as they showed the best performance versus execution time trade-off. The data set used in this case consists of outdoors conventional images from a urban environment. Now the localization consists of recognizing the scene (building) where the current view is taken, as part of an autonomous urban guide. The visual memory contains 600 reference images obtained with a conventional camera, from which 100 images correspond to labelled buildings (10 buildings, both day and night images). The test set consist of other 30 images from the labelled buildings, obtained from different days.

In this experiment the global descriptor used was a histogram computed over the Hue color band of the images (in the HSV image color space). Table 8.2 shows the average correct recognition rates (*%Ok*) and execution times for all tests. Most failures were due to the pre-filtering, i.e., the global descriptor evaluation discarded too many or all images from the correct reference location. Then, skipping the global descriptor based pre-filtering increases the recognition rates close to 100%. However, as the reference set is quite big, it is necessary to include the pre-filtering step if we require an efficient answer. Note that although the matching time (column *matching time*) using the NN-based approach is higher than using the Pyramidal one, it requires being able to pre-load the pyramidal matching structures, otherwise the whole process execution time ($matching + structures\ build$) is much lower for the NN-based method. Then, the Pyramidal matching is convenient only if it is possible to load the Pyramid search structures in advance, otherwise the advantage of the linear complexity matching method is hidden by the high cost of building its data structures. In this experiment, we consider a urban tourist guide whose goal was to recognize the building in the current view, therefore we can stop at this point (topological localization) of the hierarchical process. As mentioned before, it is convenient to apply a geometric constraint to the reference images with higher probability of being the current location. Here we perform a RANSAC based estimation of a Fundamental matrix and robust set of matches [9] between the current view and the five reference images with higher P_v. Then, the P_v is re-computed only with the correspondences that passed this robust estimation [9] and based on this we choose the current location. Without this robust selection process the rate of correct recognitions shown in Table 8.2 decreased from 90% to 80%. Two examples of the reference image selected as most probable location with the two different approaches are shown in Fig. 8.5.

Table 8.2 Building recognition results using SURF features with a 600 reference images set. Both approaches using Hue histograms as global descriptor and two view geometric constraints to make a robust selection of the current location.

similarity evaluation approach	*%Ok*	*matching time*	*matching + structures build time*
NN-based	90 %	1.35 s.	
Pyramidal-based	90 %	0.2 s.	12.45 s.

8.4.2 Metric Localization

Other previous works, such as [20] and [15], contain extensive experiments with simulated data to evaluate more accurately the metric localization results obtained from the 1D trifocal tensor. This section shows an example of metric localization to remark some of the conclusions previously mentioned. In this experiment, the 1D trifocal tensor for omnidirectional images [20] was robustly estimated using the

Fig. 8.5 Two examples of building recognition. Top: an example where both approaches (NN-based and Pyramidal-based) succeed; Bottom: example with night images, where the Pyramidal-based approach failed.

bearing from local features correspondences. An example of the robust matching obtained with radial lines or with SURF is shown in Fig. 8.6, together with the 2D reconstruction of the scene (the matched features and the robot locations) obtained from the tensor estimated in each case. Results using SIFT, both for matching and reconstruction, were very similar to SURF's ones.

A more detailed evaluation of the same experiment is summarized in Table 8.3, with the localization errors for rotation, translation direction (parameters detailed in Fig. 8.4) and landmarks location. Any of the three local feature studied (lines, SURF and SIFT) provides accuracy enough for the metric localization, as long as the variance between the used images does not make its matching process fail. The less robust features are the radial lines, as they are not able to deal with as big image changes as SIFT or SURF. In this example we can observe the fact that radial lines provide a less stable matching. In some executions they give a good result (see results in Fig. 8.6), but other times its robust estimation process fails because there were too many outliers in the initial matching sets. Then, the process incorrectly includes some wrong correspondence in the final trifocal tensor estimation, what makes the accuracy of the corresponding metric localization decrease.

8.5 Conclusion

Some results in vision research are difficult to be used in robotic applications, probably due to the current divergence of computer vision and robotics communities.

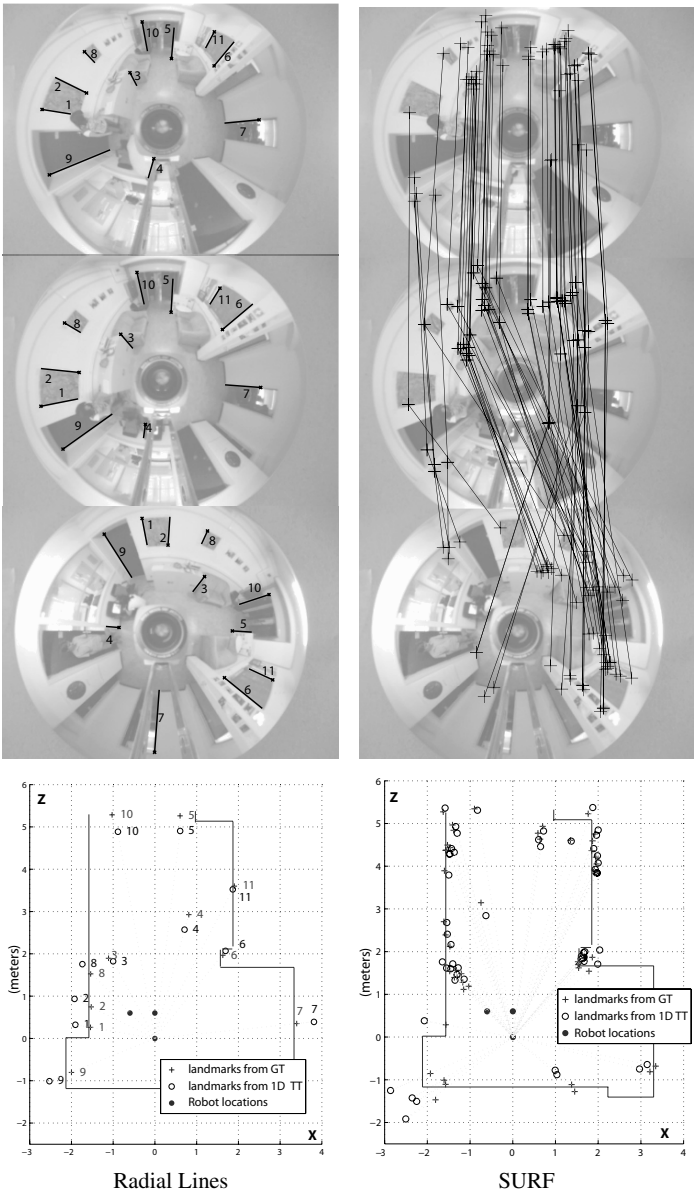


Fig. 8.6 Omnidirectional images (from *dataSet LV*) with robust matches and reconstruction of the scene from the motion parameters obtained with 1D tensor estimated with those matches (landmarks from 1D TT) or with the ground truth motion (landmarks from GT).

Here, we show experiments and results that intend to do accessible for robotic researchers some results in the frontier.

Table 8.3 Robot metric localization errors estimating the 1D tensor with different features (statistics from 50 executions). The number after each feature type shows the length of its descriptor vector.

Feature used	*Robot localization error*				*Landmarks reconstruction error (m)*	
	rotation		*transl. direction*		*mean*	*mean*
	θ' (std)	θ'' (std)	t' (std)	t'' (std)	in x-coord. (std)	in z-coord. (std)
lines-22	0.9^o (6)	1.8^o (11)	7.5^o (25)	6.5^o (13)	0.7 (1)	1.6 (2.8)
surf-64	1.1^o (0.1)	2.1^o (0.1)	0.4^o (0.4)	8.7^o (0.3)	0.2 (0.02)	0.2 (0.06)
sift-128	0.8^o (0.1)	1.9^o (0.1)	0.9^o (0.5)	8.7^o (0.3)	0.1 (0.01)	0.2 (0.01)

In the case of applying object recognition methods for scene identification, the adaptation is quite straightforward, maybe a more difficult decision is to find the most convenient kind of feature that finds a proper balance between invariant properties and fast computations.

In the case of Structure From Motion methods applied in robot localization, most of the mathematics can be recovered from computer vision papers, and in this work we summarized its particularization to the 1D bearing-only observations with planar sensor motion, which is useful in robotics. In the research areas of omnidirectional vision systems as well as bearing-only localization and mapping, navigation or visual servoing, two view relations like the fundamental matrix or the homography have been extensively used, but the use of other multi-views constraints, like the tensors, are yet poorly studied despite its attractive properties.

Acknowledgements Thanks to Oscar Calderón for his contribution with the building recognition experiments. This work was supported by projects DPI2003-07986, DPI2006-07928 and IST-1-045062-URUS-STP.

References

1. Åström, K., Oskarsson, M.: Solutions and ambiguities of the structure and motion problem for 1d retinal vision. Journal of Mathematical Imaging and Vision **12**(2), 121–135 (2000)
2. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: European Conference on Computer Vision, pp. 404–417 (2006). Http://www.vision.ee.ethz.ch/ surf/
3. Davison, A.J.: Real-time simultaneous localisation and mapping with a single camera. In: the IEEE Int. Conf. on Computer Vision, pp. 1403–1410 (2003)
4. Dellaert, F., Stroupe, A.: Linear 2d localization and mapping for single and multiple robots. In: IEEE Int. Conf. on Robotics and Automation, pp. 688–694 (2002)
5. DeSouza, G., Kak, A.C.: Vision for mobile robot navigation: A survey. IEEE Trans. on Patt. Analysis and Machine Intelligence **24**(2), 237–267 (2002)
6. Faugeras, O., Quan, L., Sturm, P.: Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. IEEE Trans. on Pattern Analysis and Machine Intelligence **22**(10), 1179–1185 (2000)
7. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM **24**, 381–395 (1981)
8. Grauman, K., Darrell, T.: The pyramid match kernels: Discriminative classification with sets of image features. In: IEEE Int. Conf. on Computer Vision, pp. 1458–1465 (2005)

9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
10. Košecká, J., Li, F.: Vision based topological Markov localization. In: IEEE Int. Conf. on Robotics and Automation, pp. 1481–1486 (2004)
11. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. Journal of Computer Vision **60**(2), 91–110 (2004). Http://www.cs.ubc.ca/ lowe/keypoints/
12. Martínez Mozos, O., Triebel, R., Jensfelt, P., Rottmann, A., Burgard, W.: Supervised semantic labeling of places using information extracted from sensor data. Robotics and Autonomous Systems **5**(55), 391–402
13. Mikolajczyk, K., Leibe, B., Schiele, B.: Local features for object class recognition. In: the IEEE Int. Conf. on Computer Vision, pp. 1792–1799 (2005)
14. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis & Machine Intelligence **27**(10), 1615–1630 (2005)
15. Murillo, A.C., Guerrero, J.J., Sagüés, C.: Robot and landmark localization using scene planes and the 1d trifocal tensor. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2070–2075 (2006)
16. Murillo, A.C., Guerrero, J.J., Sagüés, C.: Surf features for efficient robot localization with omnidirectional images. In: IEEE/RSJ Int. Conf. on Robotics and Automation, pp. 3901–3907 (2007)
17. Murillo, A.C., Sagüés, C., Guerrero, J.J., Goedemé, T., Tuytelaars, T., Van Gool, L.: From omnidirectional images to hierarchical localization. Robotics and Autonomous Systems **55**(5), 372–382 (2007)
18. Newman, P., Cole, D., Ho, K.: Outdoor slam using visual appearance and laser ranging. In: IEEE Int. Conf. on Robotics and Automation, pp. 1180–1187 (2006)
19. Opelt, A., Pinz, A., Fussenegger, M., Auer, P.: Generic object recognition with boosting. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(3), 416–431 (2006)
20. Sagüés, C., Murillo, A.C., Guerrero, J.J., Goedemé, T., Tuytelaars, T., Van Gool, L.: Localization with omnidirectional images using the 1d radial trifocal tensor. In: IEEE Int. Conf. on Robotics and Automation, pp. 551–556. Orlando, USA (2006)
21. Shashua, A., Werman, M.: Trilinearity of three perspective views and its associate tensor. In: IEEE Int. Conf. on Computer Vision, pp. 920–925 (1995)
22. Workshop-FS2HSC-data: IEEE/RSJ International Conference on Intelligent Robots and Systems (2006). Http://staff.science.uva.nl/ zivkovic/FS2HSC/dataset.html
23. Zivkovic, Z., Bakker, B., Kröse, B.: Hierarchical map building using visual landmarks and geometric constraints. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 7–12 (2005)

Chapter 9
More Vision for SLAM

Simon Lacroix, Thomas Lemaire and Cyrille Berger

9.1 Introduction

SLAM has been identified as a key problem in mobile robotics for over 20 years [4, 46], and has received much attention since, especially these last 10 years. An overview of the problem and the main proposed solutions can be found in [10, 11]. Dozens of robots now use on-board SLAM solutions on an everyday basis in laboratories.

First SLAM solutions concerned robots evolving on a 2D plane, that perceive the environment with a laser range finder. It is only quite recently that solutions to SLAM using vision have been proposed: first using stereovision [16, 41], and then with monocular cameras. A large amount of contributions to the latter problem have rapidly been proposed since the pioneer work of [7] (see for example [35, 6, 21, 13]), and a commercial software is available since 2005 [28] – though only applicable to robots evolving on a 2D plane.

There are many interests to use vision for SLAM. Besides the advantages of using a small, low cost and lightweight sensor, vision offers the benefit of perceiving the environment in a 3D volume, up to infinite distances, and of providing plenty of information relevant to analyze the perceived scenes. Last – but certainly not least, the computer vision community has provided numerous formalisms and algorithmic solutions to a collection of essential problems that are very relevant for the SLAM problem (feature detection and matching, structure from motion, image segmentation and classification, object recognition and scene interpretation, image indexing).

Simon Lacroix
LAAS-CNRS, University of Toulouse, France, e-mail: Simon.Lacroix@laas.fr

Thomas Lemaire
LAAS-CNRS, University of Toulouse, France, e-mail: Thomas.Lemaire@laas.fr

Cytille Berger
LAAS-CNRS, University of Toulouse, France, e-mail: Cyrille.Berger@laas.fr

Still, vision-based SLAM solutions do not exploit all the possibilities brought by vision. The goal of this work is to explore these possibilities. The next section briefly reviews the SLAM problem and presents the required functionalities, analyzing the ones that can benefit from vision. Relations between filtering solutions applied to SLAM and estimation techniques applied to the structure from motion problem are now clearly established, and will not be detailed here. Section 9.3 is the main of the paper: it describes how various vision algorithms can benefit to SLAM in robotics, mainly for the essentials problems of environment modeling and loop closing. Finally section 9.4

9.2 Overview of the SLAM

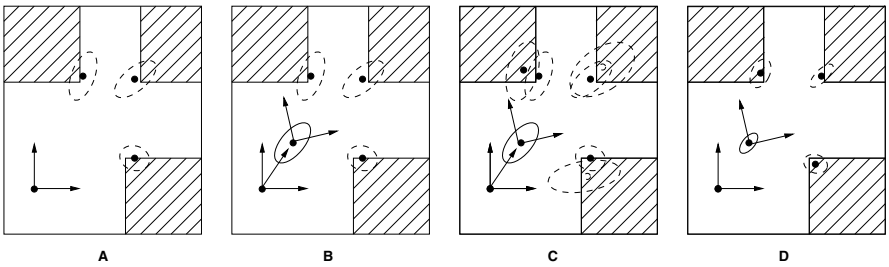


Fig. 9.1 Main steps of a SLAM process. With the first observations, the robot builds a map of 3 points (A), then it moves and computes an estimate of its position (B), 3 new observations (blue) are matched with the current map (C), and are fused to update the map and robot pose (D).

9.2.1 Functionalities Required by SLAM

A typical SLAM process is chronologically depicted in figure 9.1, in the case of a robot evolving in a 2D plane, where the landmarks are corners. The various functionalities involved in this process can be summarized as the following four ones:

- **Environment feature selection.** It consists of detecting landmarks in the perceived data, *i.e.* features of the environment that are salient, easily observable and whose relative position to the robot can be measured.
- **Relative measures estimation.** Two processes are involved here:
  + Estimation of the landmark location relatively to the robot pose from which it is observed: this is the *observation*.
  + Estimation of the robot motion between two landmark observations: this is the *prediction*. This estimate can be provided by sensors, by a dynamic model

of robot evolution fed with the motion control inputs, by assumptions on the robot motions – such as a constant velocity model.

- **Data association.** The observations of landmarks are useful to compute robot position estimates only if they are perceived from different positions: they must imperatively be properly associated (or matched), otherwise the robot position can become totally inconsistent.
- **Estimation.** This is the core of the solution to SLAM: it consists in integrating the various predictions and observations to estimate the robot and landmarks positions in a common global reference frame.

In the robotics community, the main effort has been put into the estimation functionality. Various stochastic estimation frameworks have been successfully applied [8, 49, 26], and important contributions deal with the definition of landmark map structures that lower the estimation computational complexity and allow to overcome the difficulties raised by the non-linearities of the problem [32, 33, 14].

However, most of the functionalities involved in SLAM are *perception processes*. This is obvious for the landmark detection functionality, that represents the landmark with a specific data structure, depending on the considered environment and on the sensors the robot is equipped with. Also, the relative position of the landmarks in the sensor frame (the observation) is estimated by processing the perceived data. As for the data association process, it can be achieved considering only the current estimated states of the world and the robot (that is positions and associated variances [29]). But it can be solved much more robustly and easily when tackled as a perception process, especially to establish loop closures, where perception can provide correct data associations regardless of the current estimated states.

9.2.2 Vision and SLAM

For all these perception functionalities, vision can obviously provide powerful solutions. Images indeed carry a vast amount of information on the perceived environment, and many algorithms that process these information can be very effective for SLAM:

- Detection and modeling of landmarks from images can be performed thanks to images features extraction processes for instance, as it has mainly been made up to now in vision-based SLAM approaches (for example using Harris or SIFT points). But some visual segmentation, classification or tracking processes can also be very useful for that purpose – not to mention the numerous higher level approaches to visual object recognition.
- Relative 3D coordinates of the detected landmarks are readily observable with multi-camera systems. For single cameras, the fact that angular observations are available has recently lead to the development of various partially observable SLAM solutions (“bearing-only”). Naturally, solutions to the classic structure

from motion problem (SFM) developed in the vision community have recently been successfully applied to vision-based SLAM [35, 27].
- Finally, the data association problem in SLAM is often ill-posed when only considered within the estimation framework. Although well founded approaches have been proposed (*e.g.* [29, 50]), they are hardly efficient when the estimated states become inconsistent, and they are still challenged by partially observable SLAM problems – namely by monocular vision SLAM. On the contrary, the vision literature provide with plenty of approaches that can robustly solve the data association problem, such as feature matching algorithms, object recognition or image indexing approaches.
  It is important to note here that the data association problem is very different when it comes to associate landmarks from two consecutive positions than when it comes to associate landmarks perceived from very different viewpoints, as it happens when closing loops for instance. In the first case, the problem is easily solved by *feature tracking* algorithms, whereas the second case calls for more complex *feature matching* algorithms[1].

9.3 Vision and Mapping in SLAM

9.3.1 Visual Landmarks for SLAM

In a SLAM context, landmark must satisfy the following two properties:

- They must be *detectable* in the perceived data,
- and some parameter of their position must be *observable*, so as to feed an estimation technique.

But to solve the data association problem independently of the robot and landmarks estimated positions, they must also be represented by *a model*, that contains the information required for the data association processes. A landmark model is defined by the specification of the actual landmark nature in the physical world, by the considered sensor model, and by the specification of the detection and association algorithms.

The key to a successful vision based SLAM approach relies then on the choice of the landmark representation (model) and corresponding algorithms that allow to detect, observe and associate landmarks.

Point Features

Up to now, most vision based SLAM solutions derive landmarks from point features detected in the images, be it for stereovision-based approaches [41, 17] or for

[1] Of course, using matching algorithms for the first case is often overkill.

monocular approaches [6, 21]. This is mainly due to the facts that there exist various algorithms that extract stable feature points, and that points landmarks are the simplest geometric objects to handle in a SLAM estimation framework.

Detection

The Harris points are often used, because they have good invariant properties with respect to image rotations and small scale changes [39]. More recently, SIFT features have become very popular: their detection is more scale independent, and the information associated to them (the local descriptor [22], a vector of scalar values) is very discriminant, thus allowing to successfully match them.

Geometric representation

The geometric representation of point landmarks is straightforward, their state being fully represented by the 3 Cartesian coordinates $X = (x, y, z)$. With stereovision, this state is fully observable: $(x, y, z) = h(\theta, \phi, d)$, where (θ, ϕ) are the angles at which the point is perceived, and d is the computed disparity – in the case of a rectified image pair. With monocular vision, the state is partially observable, and the observation function h is:

$$\begin{pmatrix} \theta \\ \phi \end{pmatrix} = \begin{pmatrix} \arctan(y/x) \\ -\arctan(z/\sqrt{x^2+y^2}) \end{pmatrix}$$

The observation error model is not straightforward to derive from the feature detection process. Most of the authors use a fixed estimate of the error on the observed measures (for example a standard deviation of 0.5 pixel for Harris points [39]), while some investigated the definition of more precise error models (see [24] for an error model of the disparity estimate in stereovision). There are probably some more work to be done regarding this, as good (accurate) error models are a prerequisite for any SLAM implementation.

Data association

The algorithms to find association between point features perceived vary a lot, depending on whether the viewpoints are close or not. In the second case, the choice and representation of the point features is essential, as their model must carry enough information to match them, whereas in the first case the problem is easily solved by frame-to-frame tracking algorithms, for example using simple correlation measures.

Various algorithms that match Harris points are available in the literature, some of them being able to deal with large scale changes. In [38], the Harris points are

modeled by a vector of local characteristics computed from the "local jet", a set of image derivatives. Matches are determined thanks to the computation of the Mahalanobis distance between their characteristics, and the geometric configurations between points is exploited to ensure the elimination of false matches. Extensions to large scale variations have been presented in [9]. The approach presented in [16], uses a combination of the points' signal information computed during their extraction and of the geometric constraints between detected points. Matches are established for groups of points, which allows the estimate of a local affine transformation between the images: this transformation is exploited to compute correlation scores between points, and to guide the search for matches (figure 9.2).

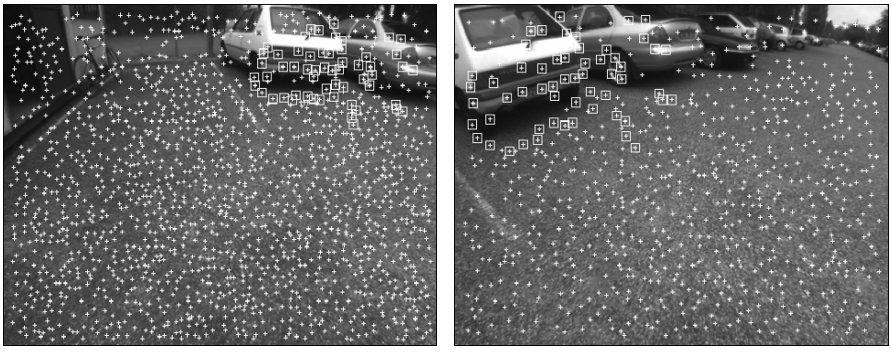


Fig. 9.2 Harris points matched with a 1.5 scale change by the algorithm presented in [16] (crosses shows all the detected points, and squares indicate successful matches).

SIFT features are intrinsically scale independent, and they can be modeled by a large vector of characteristics computed for their extraction, from which matches can be found. They are therefore well suited for a visual SLAM implementation [42, 28].

Nevertheless, these matching algorithms can be challenged by large variations of illumination conditions, and can be quite time consuming when the memorized landmarks become numerous, in particular if the robot position estimate is too coarse to focus the match search. Section 9.3.2 explains how place recognition algorithms can overcome this latter difficulty.

Map management and representation

In the literature, the maps resulting from a visual point-based SLAM approach are a set of localized 3D points landmarks with the associated variances (figure 9.3), to each of which are associated the landmark representation (for instance local image characteristics). Other useful information could however advantageously be memorized: for instance, the orientations from which points have been perceived during

the map building process could help to cast the set of landmarks to search within when closing loops.

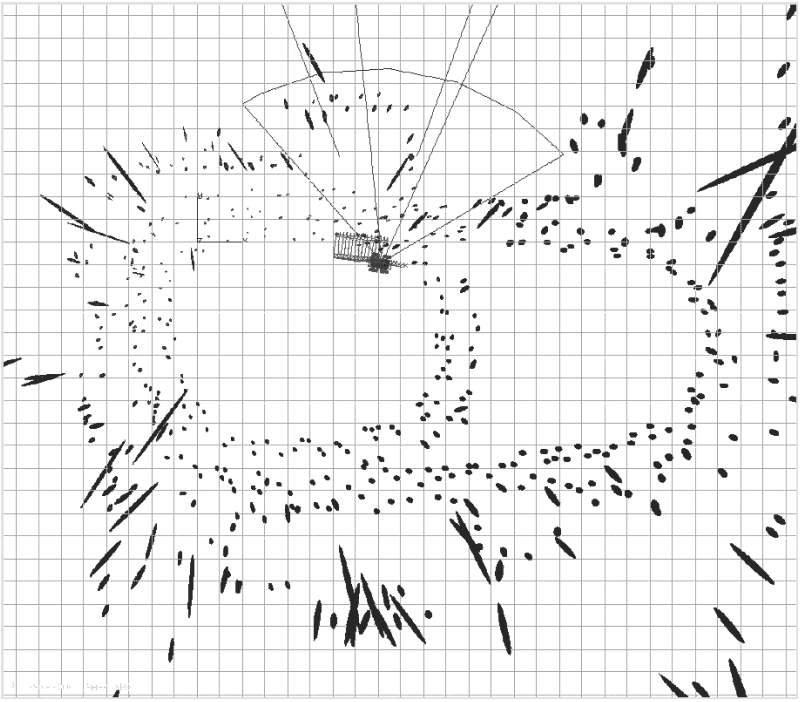


Fig. 9.3 2D projection of the map resulting from a bearing-only SLAM algorithm (the robot moved along two loops).

One of the interesting issues to deal with in the case of point-based visual SLAM is the *landmark selection* issue. Indeed feature points are so numerous in images that integrating all of them in the SLAM map rapidly yields huge maps, hardly tractable by the estimation and matching algorithms. Various simple criteria can be applied to select the points to memorize as landmarks. For instance, since a good landmark should easily be observable (matched), and landmarks should be regularly placed in the environment, the following strategy can be applied [19]: each acquired image is regularly sampled in cells. If there is at least one mapped landmark in a cell, no new landmark is selected; if not, the most salient feature point (for instance the one that has the highest Harris low eigenvalue) is selected as a landmark. This ensures a quite good regularity in the observation space (the image plane - figure 9.4). Furthermore, a simple selection in the 3D space, such as maintaining a maximum volumetric density of landmarks, can also help to get rid of useless landmarks.

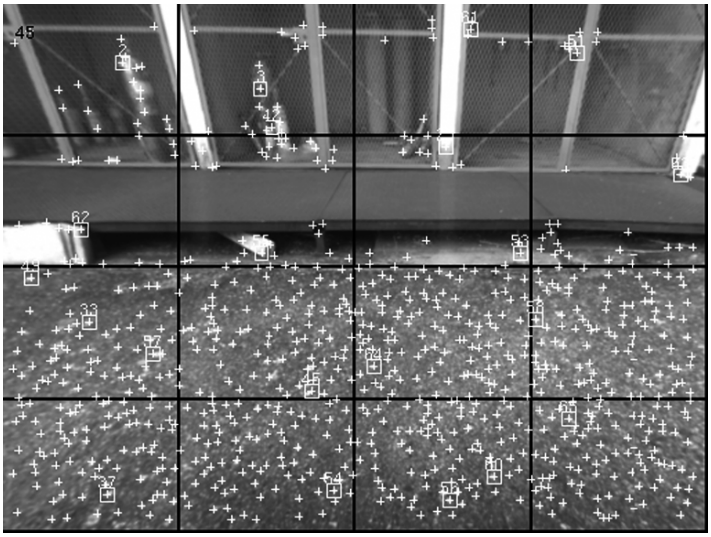


Fig. 9.4 Selection of the points that will be kept as landmarks (green squares). Some cells here contain more than one landmark: indeed, when a landmark leaves a cell, it can move to a cell where there are already landmarks (a new landmark is then generated in the old cell).

Line Features

If point features do yield successful SLAM solutions, the resulting map is however very poor, and actually only useful for the SLAM process. Various other higher level environment representations are required for autonomous mobility (by the trajectory planning processes for instance). Such representations can be built using dedicated data processing algorithms, their spatial consistency being ensured by the robot and landmark localization estimates provided by SLAM [34]. But this would be made simpler if higher level maps could be handled by SLAM: this is possible using higher level visual features.

Monocular visual SLAM using line features has only been tackled very recently. In [45], edges are defined by their two end-points, and the authors use the inverse depth parametrization [13]. A more convincing approach has been introduced in [12], and we also recently investigated the problem [20] (figure 9.5). Note that other approaches use edge information as landmarks, but do not maintain a 3D estimate of their position. For instance, in [18], vertical edges are used as 2D bearing measures.

Surprisingly, since pioneer work made in the 80's on SLAM with 3D lines extracted from stereovision [2], the problem did not retain much attention from the community (except in [5], where results are only provided in simulation).

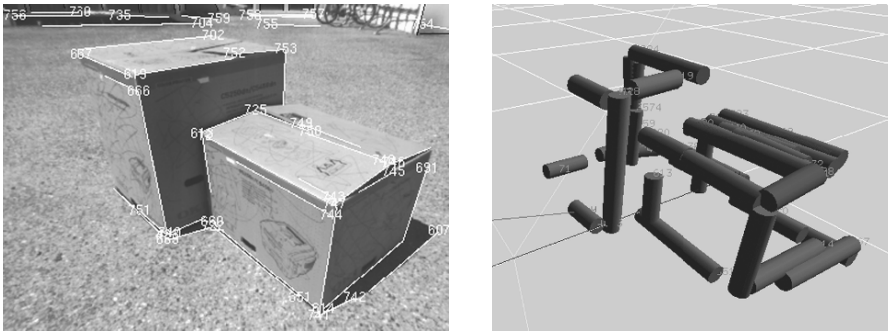


Fig. 9.5 Example of a 3D line segments model built from a sequence of monocular images (from [20]).

Detection

The vision literature provide with many approaches to extract line segments in images (such as Hough transform or contour images segmentation). But all these algorithms are very sensitive to noise and illumination: a precise estimate of the segment extremities is hard to obtain, a single line in the environment is often described as several line features, and two collinear distinct lines can be detected as a single line in the image (figure 9.6). Of course, many of these phenomena are caused by occlusions and some particular viewpoint conditions.

Fig. 9.6 A typical result of a line segment extraction algorithm: some segments are artifacts, and others are longer or smaller than the actual line in the environment.

Geometric representation

Because of the difficulties to perceive the segment extremities, it is much more reasonable to consider the parameters of the supporting lines as the state to be estimated in the SLAM process.

Several sets of parameters can be used to represent a 3D line L in Euclidean space. The minimal representation consists of 4 scalars: such a minimal representation is $(\mathbf{P_1}, \mathbf{P_2})$ where $\mathbf{P_1} = (x_1, y_1, 0)^t$ is the intersection of L with the plane $\Pi_1 (z = 0)$ and $\mathbf{P_2} = (x_2, y_2, 1)^t$ is the intersection of L with the plane $\Pi_2 (z = 1)$. Several conventions for (Π_1, Π_2) must be considered, so as to represent all possible lines with a satisfactory numerical precision (lines parallel to the planes (Π_1, Π_2) can indeed not be represented on the basis of these planes). A more intuitive but non minimal representation of L is $(\mathbf{A}, \underline{\mathbf{u}})$, where $\mathbf{A}$ is any point of L, and $\underline{\mathbf{u}}$ is a direction vector of L. In this representation, the choice of $\mathbf{A}$ is arbitrary and $\mathbf{A}$ is not observable since it cannot be distinguished on the line.

An other representation often used in the vision community is the Plücker coordinates [15]. The Euclidean Plücker coordinates are represented by the following 6-vector:

$$L_{(6\times 1)} = \begin{pmatrix} \mathbf{n} = h.\underline{\mathbf{n}} \\ \underline{\mathbf{u}} \end{pmatrix} \tag{9.1}$$

$\underline{\mathbf{n}}$ is the normal to the plane containing the line and the origin O of the reference frame, h is the distance between O and the line and $\underline{\mathbf{u}}$ is a unit vector which represents the direction of the line. The Plücker constraint has to be satisfied:

$$\mathbf{n} \cdot \underline{\mathbf{u}} = 0$$

This ensures that the representation is geometrically consistent. Any point P on the line satisfies the relation:

$$P \wedge \underline{\mathbf{u}} = \mathbf{n} \tag{9.2}$$

The advantage of the Plücker representation is that the projection of a 3D line L in an image is a 2D line l which is defined by the intersection of the image plane and the plane defined by $\mathbf{n}$: the canonical representation of l $(ax + by + c = 0)$ is exactly $\mathbf{n}$ expressed in image coordinates.

Data association

Although numerous line segment matching and tracking algorithms can be found in the vision literature [40, 37], the problem of finding outlier-free sets of matches remains a difficult one. Texture information and epipolar geometry allow to get rid of most of them, but in SLAM, one can also rely on the estimated states to filter out the remaining ones, by analyzing the difference between the predicted and observed states. Also, 3D model based object recognition algorithms can be exploited to deal with loop-closing.

Map management and representation

The resulting map of a 3D line segment SLAM approach that estimate the supporting line parameters is still far from a 3D wire model of the environment. In order to get a more precise description of the scene structure, the coordinates of the segment endpoints must be estimated. These coordinates can hardly be part of the estimated numerical state, as their observation is affected by noise, segmentation errors and occlusions, these errors being not at all Gaussians. A dedicated process must therefore be defined – for instance a simple heuristic can consist in updating their linear abscissas by considering the observations that yield the longest segments.

Planar Features

If a wire 3D model of the environment contains more structural information than a 3D points model, it can still hardly be exploited by other functionalities than localisation. A natural extension would be to add 3D planar patches and areas to the model. Again, numerous contributions and vision can be exploited for that purpose in SLAM, with the very interesting fact that planes carry more geometric information to be estimated. Indeed, if one is able to measure the normal of a plan patch (2 parameters) plus an orientation around this normal, a planar patch is described by 6 independent parameters: the observation of a single planar landmark provide enough information to estimate the 6 parameters of the robot position.

Detection

Of course, at least two images taken from different viewpoints are required to extract planar areas in the perceived scene. The most efficient way to detect such areas is to determine whether or not there exists a homography H that transforms the plane's projection from one image to the other. Considering a plane P and two images I_1 and I_2 taken from different viewpoints, for all points of P, the coordinates of the corresponding pixel in I_1 and I_2 are linked by a homography H. Two areas I_1^p from I_1 and I_2^p from I_2, correspond to a planar feature if there is a matrix H such that:

$$H * I_1^p = I_1^p \tag{9.3}$$

Moreover, the homography estimate can provide a measure of the planar patch surface, which is linked with the 3D transformation (R,t) between the two view points by the following relation:

$$H \approx R + t\frac{\mathbf{n}^T}{d} \tag{9.4}$$

where $\mathbf{n}$ is the normal vector of the plane and d its distance to the camera.

Homographies can be retrieved from two close viewpoints thanks to image alignment techniques (a review of image alignment algorithms can be found in [3], and a very efficient method to track large image patches has been proposed in [23]).

Such algorithms can be used for SLAM in the monocular case: in [25], a nice approach that also estimates the planar patches normal is presented. However, here the normal estimate is not precise enough to be part of the landmark states for SLAM: indeed, image alignment techniques require rather large patches to provide an accurate estimate of the plane normal. Nevertheless, in [25] the normal estimate is used to predict how the image patches should be warped to ease the point matching process in case of large viewpoint changes.

Planar patches can of course be more easily detected from stereovision images. A first natural idea is to use dense points pixel correspondences to detect them. But fast stereovision algorithms are quite noisy, and the normal vector estimates provided by plane fitting algorithms applied on sets of neighbouring 3D points are not reliable[2]. Finding the homography estimates using an image alignment algorithm, which is made easier thanks to the knowledge of the epipolar geometry, yields much better results. Nevertheless, their application on small areas (*e.g.* 20×20 pixels) can sometimes provide totally erroneous normal estimates.

Figure 9.7 shows the local planar patches ("facets") detected from a pair of stereovision images, centred on Harris points in the image pair. To eliminate facets whose normal estimate is erroneous, there are two possible solutions. The first one consists in exploiting texture attributes to determine whether the homography estimate is good or not, and the second is to rely on the behaviour of the data association process (see below) to discard wrong facets.

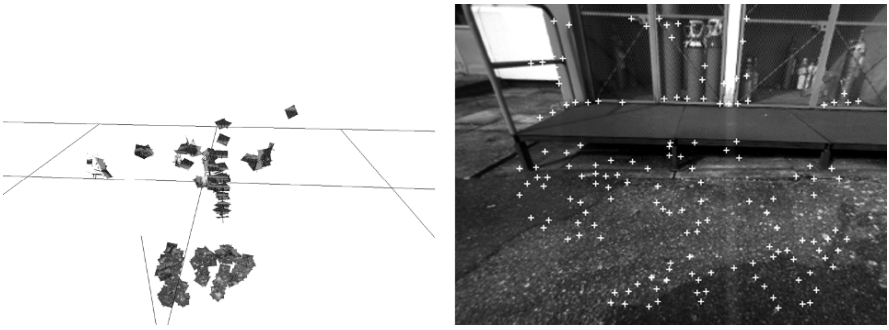


Fig. 9.7 Facets extracted from a single pair of stereovision images. Right: left camera image, with the matched Harris points on which the facets are centred.

[2] Various sophisticated dense stereovision algorithms that provide more precise results exist in the literature – but they are computationally much more expensive than the fast algorithms used in robotics.

Geometric representation

Facets extracted on matched Harris points are naturally described by the point local characteristics, but their description is extended by the estimate of their normal, which gives 2 additional positioning parameters (orientation). This description can be completed by a third orientation parameter, which can for instance be defined by local gradients computed on the facet pixels (figure 9.8). This additional orientation provides a full description of the facet position in 3D, which can be advantageously used for the data association and SLAM estimation processes.

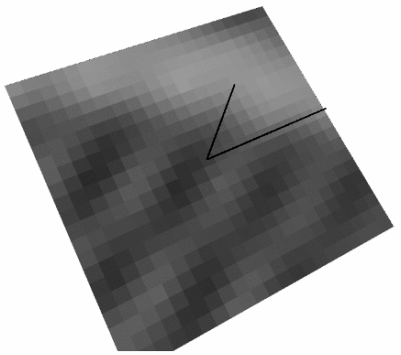


Fig. 9.8 Close view of an extracted facet, with its normal (red) and third orientation (green) estimates.

Data association

We are currently experimenting on an algorithm similar to the Harris point matching algorithm described in [16]: the principle is to generate match hypotheses on the basis of local information, and to confirm these hypotheses using geometric constraints between neighbouring facets. The geometric constraints are 3D constraints, and are very much discriminant. Figure 9.9 shows a map of facets built from several positions.

Higher Level Landmarks

Facets are only the first step toward a higher level of representation of the environment. In structured environments (that is urban-like and indoor), many objects can be described using first order geometric primitives. Algorithms that extract large planar areas from monocular image sequences are now becoming efficient and robust (see for instance [43, 44]): combined with facets and line segments representations, they can yield the building of high level maps in a SLAM context. Other approaches

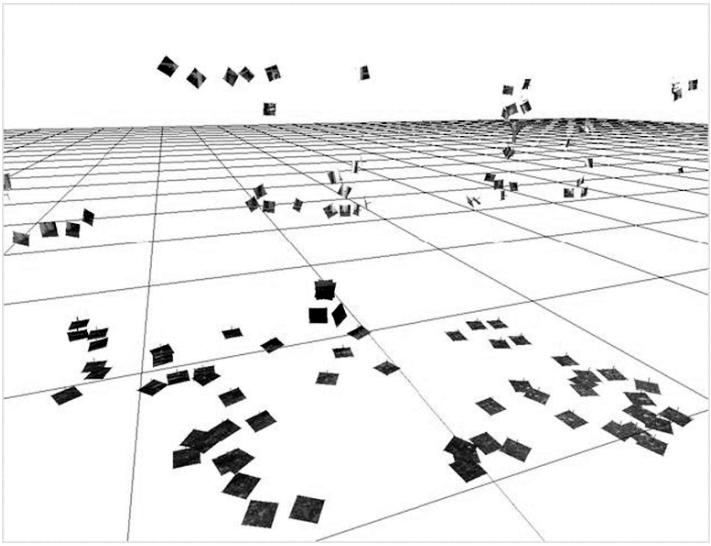


Fig. 9.9 Planar patches extracted from a set of stereoscopic pairs.

that model and detect non-structured objects can also be very helpful for SLAM (see for instance a method for detecting tree trunks in [1]).

9.3.2 Loop Closing

As the estimation framework can hardly solve the loop closing problem when the robot position estimate is poorly known, we rather consider the loop closing process as a perception process. This implies a re-definition of the loop-closing in SLAM: instead of defining it as a topological event corresponding to a loop trajectory, we rather consider that a loop-closure occurs when a mapped landmark that is currently not being tracked is re-observed – and associated thanks to a landmark matching algorithm.

We have seen that with good landmark visual representations, various matching algorithms could be used for the purpose of loop closing. However, relying only on landmark matching processes to *detect* loop closures can be an issue: with maps containing a large number of landmarks, the matching algorithms are challenged and can be quite time consuming. Image indexing techniques can be of a very good help here, and have already been successfully applied in various SLAM approaches [36], and naturally, the problem is made much easier when using panoramic cameras [19, 47]. The literature on "view-based navigation" or "appearance-based localization" in robotics is already abundant in the robotics community, and progresses in this domain in computer vision are definitely worth to be considered in visual SLAM

approaches. Thanks to these approaches, topological loop closures can be efficiently and robustly detected, which allows to focus landmark matching algorithms.

Using such techniques yields to the definition of a new kind of environment model, dedicated to loop closure detection, consisting in a database of image indexes and signatures.

9.4 Discussion

The next steps in vision-based SLAM approaches are certainly to focus on the development of rich maps, that exhibits the environment 3D structure and semantic information. We have seen that many vision tools are available for that purpose, and that some have already lead to interesting results. Much work remains however to be done to integrate those tools in SLAM solutions. One of the interesting issue is to focus on the synergies between these tools and the SLAM estimation process. Such developments also appear promising to tackle two difficult challenges for SLAM, namely multi-robot SLAM and the integration of SLAM approaches within Geographic Information Systems (GIS).

Multi Robot SLAM

From the estimation point of view, various contributions solve the multi-robot SLAM problem, in which robots can observe the position of other robots and of landmarks mapped by other robots (see for instance [30, 48, 31]). For this problem, data association between landmarks perceived by the different robots would of course greatly benefit from the building of high level landmark map representations – all the more when considering heterogeneous robots (figure 9.10).

SLAM, GIS and Vision

There is currently a tremendous development in the building and exploitation of Geographic Information Systems, that partly inherits from progresses in computer vision. Any operational robotic system, be it aerial, terrestrial or even maritime, should not ignore such initial information on the environment, as it can be of a very good help to perform SLAM. There are obvious similarities between GIS and robotic mapping, as the resulting environment models are organized in layers containing information relevant for different processes. Considering the various environment models previously sketched (planar regions, segment-based object descriptions, dense models), we end up with an environment model that has the same layered structure of a usual GIS. The bottom layer is made of the set of landmarks

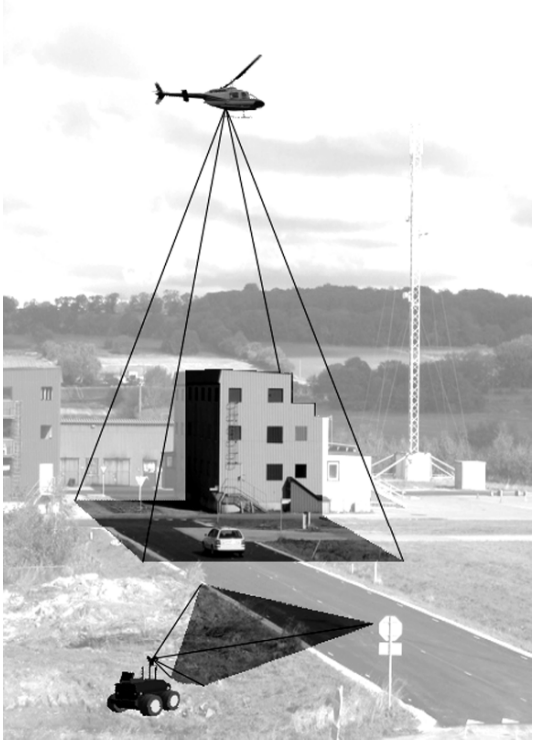


Fig. 9.10 Air/ground multi robot cooperation. What landmarks can be used to build a consistent environment model from the data perceived by the two kinds of robots ?

which are consistently estimated by SLAM. The upper layers are the maps containing dense data, or possibly other sparse information relevant for the robots or the mission (see figure 9.11). The only difference is that the layers of a GIS are defined in a single Earth centered reference frame, whereas the layers of the SLAM maps are made of local maps anchored in the bottom stochastic layer.

Note also that some visual SLAM approaches have been able to build environment models from aerial data (*e.g.* [17]) – a problem that had been exclusively considered in the GIS community so far.

Again, the problems to solve to integrate SLAM-built maps and GIS models rely essentially on the data association side. We believe that the development of higher environment models such as planar regions or segment-based descriptions on the basis of visual information is a promising way to tackle them.

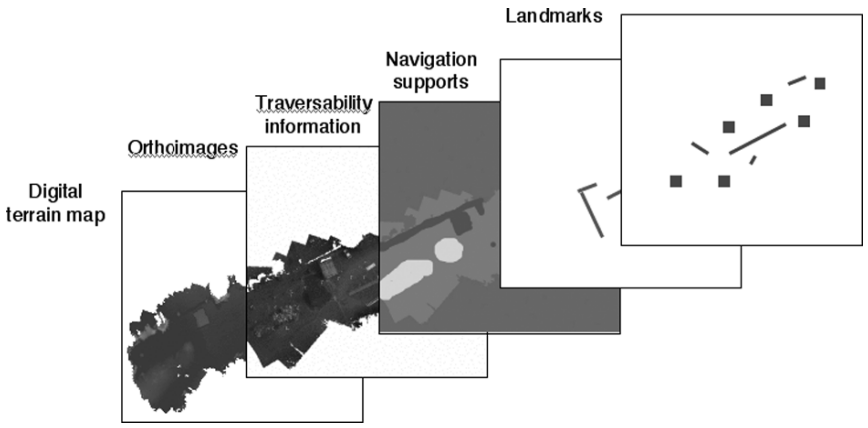


Fig. 9.11 The various environment models built by an autonomous robot have a layered structure akin to the one of a GIS.

References

1. Asmar, D.C., Zelek, J.S., Abdallah, S.M.: Tree trunks as landmarks for outdoor vision slam. Conference on Computer Vision and Pattern Recognition Workshop (2006). DOI http://doi.ieeecomputersociety.org/10.1109/CVPRW.2006.207
2. Ayache, N., Faugeras, O.: Building a consistent 3D representations of a mobile robot environment by combining multiple stereo views. In: 10th International Joint Conference on Artificial Intelligence, Milan (Italy), pp. 808–810 (1987)
3. Baker, S., Matthews, I.: Equivalence and efficiency of image alignment algorithms. In: Proceedings of the 2001 IEEE Conference on Computer Vision and Pattern Recognition, Kauai, USA (2001)
4. Chatila, R., Laumond, J.P.: Position referencing and consistent world modeling for mobile robots. In: IEEE International Conference on Robotics and Automation, St Louis (USA), pp. 138–145 (1985)
5. Dailey, M., Parnichkun, M.: Simultaneous localization and mapping with stereo vision. In: Proceedings of the International Conference on Automation, Robotics, and Computer Vision, Singapore (2006)
6. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: IEEE International Conference on Computer Vision, Nice (France), pp. 1403–1410 (2003)
7. Davison, A., Kita, N.: Sequential localisation and map-building for real-time computer vision and robotics. Robotics and Autonomous Systems **36**, 171–183 (2001)
8. Dissanayake, G., Newman, P.M., Durrant-Whyte, H.F., Clark, S., Csorba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotic and Automation **17**(3), 229–241 (2001)
9. Dufournaud, Y., Schmid, C., Horaud, R.: Matching images with different resolutions. In: International Conference on Computer Vision and Pattern Recognition, Hilton Head Island, SC (USA), pp. 612–618 (2000)
10. Durrant-Whyte, H., Bailey, T.: Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. IEEE Robotics and Automation Magazine **13**(2), 99–110 (2006)
11. Durrant-Whyte, H., Bailey, T.: Simultaneous Localisation and Mapping (SLAM): Part II State of the Art. IEEE Robotics and Automation Magazine **13**(3), 108–117 (2006)
12. Eade, E., Drummond, T.: Edge landmarks in monocular slam. In: British Machine Vision Conference, Edinburgh (UK) (2006)

13. Eade, E., Drummond, T.: Scalable monocular SLAM. In: Conference on Computer Vision and Pattern Recognition, New York (USA), pp. 469–476 (2006)
14. Estrada, C., Neira, J., Tards, J.: Hierarchical SLAM: real-time accurate mapping of large environments. IEEE Transactions on Robotics **21**(4), 588–596 (2005). URL http://webdiis.unizar.es/~jdtardos/papers/Estrada_TRO_2005.pdf
15. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2004)
16. Jung, I.K., Lacroix, S.: A robust interest point matching algorithm. In: 8th International Conference on Computer Vision, Vancouver (Canada) (2001)
17. Jung, I.K., Lacroix, S.: High resolution terrain mapping using low altitude aerial stereo imagery. In: International Conference on Computer Vision, Nice (France)) (2003)
18. Kwok, N., Dissanayake, G.: Bearing-only SLAM in indoor environments using a modified particle filter. In: Australasian Conference on Robotics and Automation, Brisbane (Australia) (2003)
19. Lemaire, T., Lacroix, S.: Long term SLAM with panoramic vision. Journal of Field Robotics **24**(1-2), 91–111 (2007)
20. Lemaire, T., Lacroix, S.: Monocular-vision based SLAM using line segments. In: IEEE International Conference on Robotics and Automation, Roma (Italy) (2007)
21. Lemaire, T., Lacroix, S., Sola, J.: A practicle bearing-only SLAM algorithm. In: IEEE International Conference on Intelligent Robots and Systems, Edmonton (Canada) (2005)
22. Lowe, D.: Distinctive features from scale-invariant keypoints. International Journal on Computer Vision **60**(2), 91–110 (2004)
23. Malis, E.: Improving vision-based control using efficient second-order minimization techniques. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, USA (2004)
24. Matthies, L.: Toward stochastic modeling of obstacle detectability in passive stereo range imagery. In: IEEE International Conference on Computer Vision and Pattern Recognition, Champaign, Illinois (USA), pp. 765–768 (1992)
25. Molton, N., Davison, A., Reid, I.: Locally planar patch features for real-time structure from motion. In: British Machine Vision Conference, London (UK) (2004)
26. Montemerlo, M., Thrun, S., Wegbreit, B.: Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: International Conference on Artificial Intelligence (AAAI) (2003). URL http://www-2.cs.cmu.edu/~mmde/mmdeijcai2003.pdf
27. Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., Sayd, P.: Monocular vision based SLAM for mobile robots. In: 18th International Conference on Pattern Recognition, Hongkong, China (2006)
28. Munich, M., Pirjanian, P., Bernardo, E.D., Goncalves, L., Karlsson, N., Lowe, D.: SIFT-ing through features with ViPR. IEEE Robotics and Automation Magazine **13**(3), 72–77 (2006)
29. Neira, J., Tardós, J.: Data association in stochastic mapping using the joint compatibility test. IEEE Transactions on Robotics **17**(6), 890–897 (2001)
30. Nettleton, E., Gibbens, P., Durrant-Whyte, H.: Closed form solutions to the multiple platform simultaneous localisation and map building (slam) problem. In: SPIE AeroSense conference, Orlando, Fl (USA) (2000)
31. Nettleton, E., Thrun, S., Sukkarieh, H.D.W.S.: Decentralised SLAM with low-bandwidth communication for teams of vehicles. In: International Conference on Field And Service Robotics , Yamanashi (Japan) (2003)
32. Newman, P.: On the structure and solution of the simultaneous localisation and map building problem. Ph.D. thesis, Australian Centre for Field Robotics - The University of Sydney (1999). URL http://oceanai.mit.edu/pnewman/papers/pmnthesis.pdf
33. Newman, P.M., Leonard, J.J.: Consistent convergent constant time SLAM. In: International Joint Conference on Artificial Intelligence, Acapulco (Mexico) (2003). URL http://www.robots.ox.ac.uk/~pnewman/papers/IJCAI2003.pdf
34. Nieto, J., Guivant, J., Nebot, E.: DenseSLAM: Simultaneous localisation and dense mapping. International Journal of Robotics Research **25**(8), 711–744 (2006)

35. Nister, D.: An efficient solution to the five-point relative pose problem. In: IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wi. (USA) (2003)
36. Posner, I., Schroeter, D., Newman, P.: Using scene similarity for place labeling. In: International Symposium on Experimental Robotics, Rio de Janeiro (Brazil) (2006)
37. Pressigout, M., Marchand, E.: Real-time hybrid tracking using edge and texture information. International Journal of Robotics Research **26**(7), 689–713 (2007)
38. Schmid, C., Mohr, R.: Local greyvalue invariants for image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(5) (1997)
39. Schmid, C., Mohr, R., Bauckhage, C.: Comparing and evaluating interest points. In: International Conference on Computer Vision, Bombay, India (1998)
40. Schmid, C., Zisserman, A.: Automatic line matching across views. In: IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Porto Rico (1997)
41. Se, S., Lowe, D., Little, J.: Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. International Journal of Robotics Research **21**(8), 735–758 (2002)
42. Se, S., Lowe, D., Little, J.: Vision based global localization and mapping for mobile robots. IEEE Transactions on Robotics **21**(3), 364–375 (2005)
43. Silveira, G., Malis, E., Rives, P.: Real-time robust detection of planar regions in a pair of images. In: IEEE/RSJ International Conference on Intelligent Robots Systems, Beijing (China) (2006)
44. Silveira, G., Malis, E., Rives, P.: An efficient direct method for improving visual SLAM. In: IEEE International Conference on Robotics and Automation, Roma (Italy) (2007)
45. Smith, P., Reid, I., Davison, A.: Real-time monocular SLAM with straight lines. In: British Machine Vision Conference, Edinburgh (UK) (2006)
46. Smith, R., Self, M., Cheeseman, P.: A stochastic map for uncertain spatial relationships. In: Robotics Research: The Fourth International Symposium, Santa Cruz (USA), pp. 468–474 (1987)
47. Tapus, A., Siegwart, R.: Bayesian sensory-motor models and programs, chap. Topological SLAM using Fingerprints of Places. Springer-Verlag (2007)
48. Thrun, S., Liu, Y.: Multi-robot slam with sparse extended information filters. In: International Symposium of Robotics Research, Sienna (Italy) (2003)
49. Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., Durrant-Whyte, H.: Simultaneous Localization and Mapping With Sparse Extended Information Filters. International Journal of Robotics Research **23**(7-8), 693–716 (2004)
50. Wijesoma, W.S., Perera, L., Adams, M.: Toward multidimensional assignment data association in robot localization and mapping. IEEE Transactions on Robotics **22**(2), 350–365 (2006)

Chapter 10
Maps, Objects and Contexts for Robots

James J. Little and Tristram Southey

10.1 Introduction

When you have brought your new Apple iRobot[1] home for the first time, you are faced with the challenging task of introducing the robot to its new home/workspace. Of course the robot knows about homes and typical tasks. That's why you bought the sleek, stylish robot, in addition to the fact that it promised a simple interface. It prompts you to take it on a tour of the house, naming the rooms, pointing out the appliances, and identifying the occupants of the house.

The promise of the now discontinued Aibo whose communication and basic behaviours show that even simple visual sensors using strong features (SIFT[17]) can enable visual tracking and recognition. Built in to the home robot will be the necessary concepts – tasks, objects, contexts, locations. Your home vacuum robot "knows" only about stairs, objects, infrared walls, and random search. Your iRobot knows about kitchens, doors, stairs, bedrooms, beer (you have the party version of the iRobot that can bring you beer in the entertainment room). How does the robot tie the sensory flow it receives to its plans, names, and goals in its repertoire?

This fanciful thought experiment is not so far in the future. For some applications such as assistive technologies[21, 1] which operate in contexts where rich visual sensing is deployed, the range of objects may be limited to a care facility where patients' rooms are typically constrained in their contents. Here it may be effective to learn the connection between features of the visual stream and the object in the scene, and how they influence the actions of the robot.

James J. Little
Department of Computer Science, University of British Columbia, Vancouver, BC, Canada, e-mail: little@cs.ubc.ca

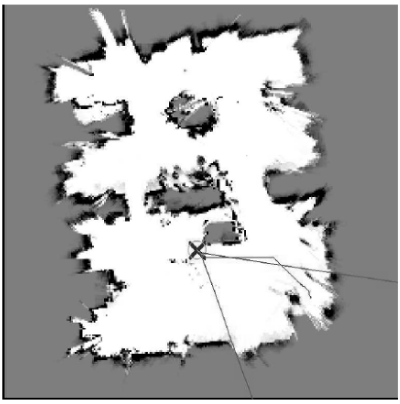
Tristram Southey
Department of Computer Science, University of British Columbia, Vancouver, BC, Canada, e-mail: southey@cs.ubc.ca

[1] Apologies to two major corporations that may be clashing over naming rights in the future.

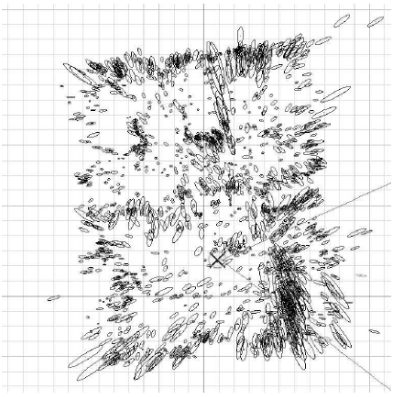
What is needed is knowledge of the structure of the environment: the objects in the world, their functional relations, their spatial layout and their role in tasks. Visual sensing offers some solutions. There are many ways to compute features or keypoints useful for localization and detection/recognition[17]. See Mikolajczyk and Schmid[22] for a discussion of the performance of feature detectors. Many techniques used in recognition, mapping and localization begin with these interest points.

Fig. 10.1 SIFT features found by our stereo-equipped robot. SIFT features in the three stereo images are matched; horizontal and vertical lines indicate the horizontal and vertical disparities respectively (from [16]).

Laser and other active sensors are not only becoming more compact, less power hungry, and less expensive, but also our techniques for solving mapping and localization (SLAM) have become increasingly powerful[35]. Solutions based on passive visual sensors, both monocular[2] and stereo[27, 29, 30], are capable also of delivering both excellent localization in and high-quality geometrical descriptions of unstructured environments. Figure 10.1 shows a set of features during a robot's tour of our lab, with their disparities. As the robot moves in the world, it aggregates local stereo information into an occupancy grid[5] where the robot plans paths and represents the structure of solids and voids. A stereo sensor delivers real-time data about the presence of obstacles and the full arsenal of probabilistic methods for localization and estimation of actions are available[29, 30]. Figure 10.2 shows the occupancy grid constructed for the maximum-likelihood sample at the end of exploration, and the landmark map constructed for the maximum-likelihood sample at the end of a traversal of the lab; error ellipses show the spatial uncertainty of the SIFT feature points (projected from 3D). Note that the stereo cameras provide a high density of image features which match between the separate camera views and as the robot moves.



(a)



(b)

Fig. 10.2 (a) Occupancy grid constructed for the maximum-likelihood sample at the end of exploration using a stereo camera. (b) Map of SIFT landmarks constructed for the maximum-likelihood sample at the end of exploration (from [30]).

During map-building the process of localizing the landmarks removes the dynamic scene elements (i.e. things that move). However, we intend that long-term use of the maps will enable us to track configurations of features that move rigidly and so label some scene elements as movable. We should then be able to enhance our robot with knowledge of the separable objects in its environment. Section 10.3 builds on this insight and close observation of moving objects to isolate them and build 3D model shape and appearance models.

Using these appearance models of objects, and recognizing their semantic type, we can find them again in the environment. But the robot needs more than geometric maps and objects. To perform its tasks, it needs a structure of its activities, their constituent actions, and the objects they require. Activities, e.g., cooking, oc-

cur in prototypical places, e.g., kitchens, which have an expected arrangement of the objects. The expected local set of objects and their configuration consitutes the context of an object. The remainder of this paper explores the relations between tasks, objects and contexts for robots using maps.

10.2 Structuring Space

Others have described the connection between metric conceptions and representations and topological representations that connect regions or places. In [14] Kuipers describes the Spatial Semantic Hierarchy that builds representations of the spatial domain at multiple levels: sensorimotor, control, procedures, topology, geometry. The base level present in many sensed representations is geometry, where metric quantified information about the position and orientation of objects is maintained. The more tractable topological representation captures paths and connections between distinguished locations.

We can look to research in ontological description of objects[34] to assist us. Researchers have recently explored two important aspects of increasing usability of systems – using external knowledge and the structured information about the world in the form of ontologies[34].

We were motivated to pursue this by the work of Kautz and his colleagues[24] on assistive technologies where they track people in their homes and observe their everyday activities using RFID tags to sense the people as they use tagged objects. In a sensored home with cameras the limitations of RFID tagging can be overcome[20]. The objects are labeled with their description, for example, paring knife, spoon, cup, jug, and so forth. But classification of activities is hampered by the level of detail of the labels. To improve classification, they use an ontology, a description of the objects in the world, and their properties. Ontologies are hierarchical, and provide a variety of levels of abstraction of labels (properties), so that paring knives and butter knifes are knives, while knives and spoons are both utensils, and whisks are utensils also, and so forth. By choosing the right level of abstraction, action classification improves.

Surprisingly it is more and more possible to find information supplying ontological description for informal situations and objects, amazingly available for many categories of objects[34]. Ontologies for more formal and technical subjects, such as geographical entities[18], have been extensively studied.

In order for a home robot to accomplish useful tasks, it must have a description of the semantic organization of the home. Tasks are completed by a sequences of *activities* or sequences of actions. Typical activities might be cooking or washing, and the actions are taken to be primitive elements like cutting food or heating water. We call a location in the home where a particular collection of activities occurs a *place*. A functional robot needs a map where places and activities are attached to locations in the map.

How will the robot acquire information about places and activities? By pointing, demonstration, download, exploration? If the information is from exploration, we may face several obstacles in sharing ontologies with the robot. Once the robot has learned the structure of the environment, will it partition the world into events homeomorphic to some of our concepts? There certainly will be hidden correlations between the objects and situations in our world and the actions they foster, and an exploring robot will uncover some novel dependencies.

We do not yet have hierarchical knowledge of categories of objects, nor their appearance, but extensive current work on representing the appearance and shape of objects[15] allows us to use these appearance models in recognition. These models need not be acquired from images, but their shape can be gotten from CAD models in the design process. Our work on object discovery (Section 10.3) goes part way to this goal in an unstructured environment. Object discovery is not recognition, as in [3], but creation of models of scene elements by identifying independently moving parts of the scene and then growing a 3D model over several observations.

10.3 Feature-based Object Discovery

Coherently moving groups of features in a map are an indicator of the presence of a moving rigid object. These features can provide information about both the appearance and shape of the object. However, because of the difficulty of successfully matching image features between different views of a moving object, they are typically sparse, resulting in imperfect information about the object. The goal of object discovery is to recover more complete object descriptions[33] from these coherent structures by isolating them from their surroundings using differences in depth or, more generally, parallax and appearance. This isolation or segmentation allows us to extract addition information from the input stream. Our work is similar in spirit to [31] where they build models from two types of features: interest point similar to SIFT, and MSER[19], tracking the features over long sequences, and then using the appearance models aggregated over the frames to match objects in other frames of the movie.

We examine the problem of *object discovery*, autonomous acquisition of object models, using a combination of shape, appearance and motion. Our approach is a multi-stage technique for detecting rigidly moving objects and modeling their appearance for recognition. First, the stereo camera is used to find a sequence of images and depth maps of a given scene (Figure 10.3). This is the same information that is used to construct our maps and so can be collected concurrently.

Then the scene is oversegmented using normalized cuts[28], a technique for segmenting a graph based on a weight function between nodes. NCuts can be applied to an image by treating the pixels as nodes. Our weight function is based on the differences in intensity, depth and 2D image position between pixels (Figure 10.3(c)). SIFT features[17] are matched between sequential pairs of images to identify groups of rigidly moving features; the 3D movement of these features determines which

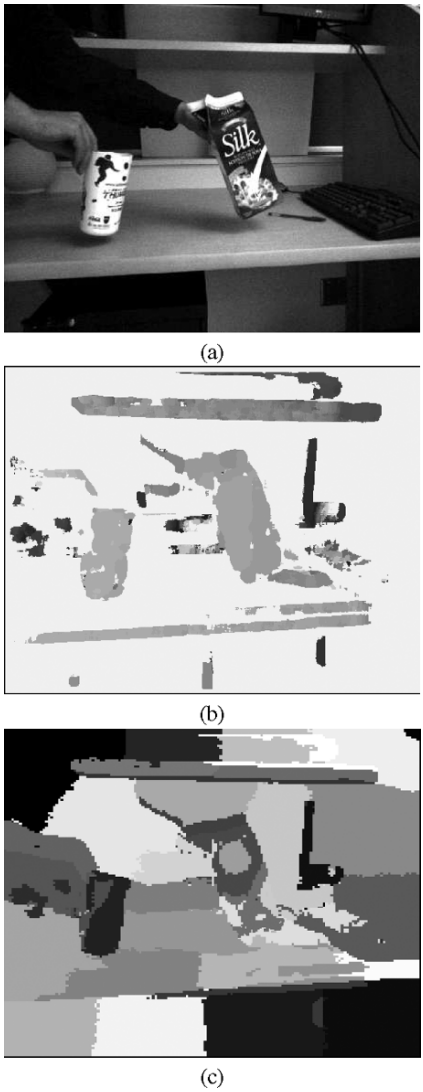


Fig. 10.3 (a) Input image for object discovery by motion, appearance, and shape (b) depth image from stereo (c) segmentation of the image and depth map using normalized cuts based on both the pixel intensity and depth.

regions in the segmentation of the scene correspond to rigid objects, grouping over-segmented regions as necessary (see Figure 10.4). Determining which segmented regions correspond to which rigid feature group (i.e., which rigidly moving object) is done through a voting process based on the number of features from each group in a region.

Additional features are then extracted from segmented regions and combined with the rigidly moving image features to create snapshots of the object's appearance (see Figure 10.4) and shape. Over time, even when objects have ceased moving, these snapshots are combined to produce models that are effective for recognition. The models contain shape information, since all feature points contain depth information, and appearance, from the aggregation of image regions.

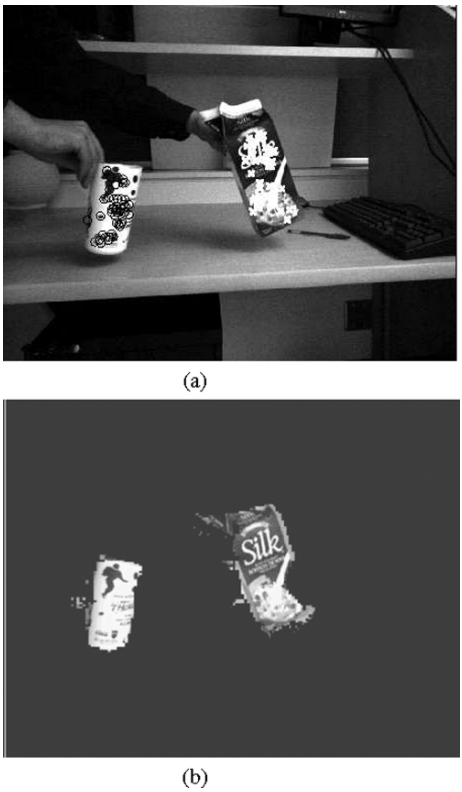


Fig. 10.4 (a) This image shows the position of every feature within the regions corresponding to the rigidly moving features in the input image, both core features and adjunct features. Features on cup on the left are indicated with Os while those on the milk box on the right are indicated with Xs. (b) Object snapshots of the cup and the milk box acquired through region voting.

Our object discovery method envisions a robot system seeing moving objects – we have ourselves moved the objects in our experiments, but in the future the robot will continually revisit the scene. Objects that have moved can be indexed through their SIFT features and our process applied over pairs of images where the robot has moved instead. Other methods (Ferrari et al.[9], for example) extend local groups of features to connect them into an appearance model that can be used in recognition.

10.4 Visual Context

Visual sensing can also situate the operation of a robot in a larger context: providing image and scene contexts, object categories, object recognition, actions, and intentions. Connecting the robot to a task and the elements of a scene simplifies the sensing requirements, by narrowing down the focus. We will see later how *José* used this aspect of vision.

Context tells the robot where to look and links with the task through the types of locations and objects involved in the task. Context directs visual processing by informing the robot "where to look". Completion of a task requires knowing "where" and "when to look" which is in turn dependent on a model of the task and environment.

The control of attention depends on the sequence of the task and exogenous events that may not be consistent with the normal staging of a robot's operations. Most models of visual attention[13, 38] depend on a model of the visual saliency or novelty of a visual feature or phenomenon, whatever its modality, and model the spatial structure of the visual attention path and its temporal course. These are essentially bottom-up. The active realization of this concept is shown in Elder [4] where the high-resolution camera is directed by low-resolution cues to salient regions.

Context can direct attention in a top-down fashion, moving from a global estimation of the category of the scene to relative spatial location of interesting objects in the scene. In pioneering work, Rimey and Brown[26] showed a selective vision system that used Bayesian estimation and decision-theoretic control to decide where and how to gather visual evidence. We will see later (Section 10.5) how POMDPs have become the model of choice for sensing and action. Recently Vogel and Murphy[40] have used learned information about the spatial arrangement of objects to speed up search in images. The innovation in Itti and Baldi's new model[12] is to draw the attention to spots where the visual event is unlikely given a model of the image. This introduces a component of the scene and its content via the prior model.

[37] introduces the concept of the "gist" of the scene, the visual characteristics (cues) that enable us without close inspection to determine the category of a location. They demonstrated that one could learn the gist, described in terms of responses to banks of filters, and then discriminate broad categories of scenes such as corridors, offices, and streets. Whereas localization and recognition using feature points employs specific discriminative features, gist identification targets descriptors that correspond more to spatial organization and structure, such as parallelism, rectangular-orientation, or clutter. It is important to note that gist and context are not the same. Gist is a quantitative feature of a image which can be used to identify the scene's type, while context encompasses information about object relative spatial structure, task dependent object knowledge, and scene type.

In the spirit of [3] we wish to connect semantic and spatial information. They use a representation of the local appearance of an object and tag the world map with that appearance representation so that it can be indexed later. Likewise Vasudevan

et al.[39] take the first steps toward building world models that connect spatial representations with the labeled content of the scene.

Relative spatial location of objects, their configuration, often depends on the relation of the objects to the tasks in which they participate. A space is well organized when the objects are at hand during the accomplishment of a task. It is relatively unlikely that a desk, for example, will be arranged so that the objects next to the keyboard and beside the monitor are automotive parts. There is a need for stronger perception of objects, stability, continuity, identity before identification. Low level processes can tease out identity and coherence, which precedes the process of categorization, but stronger spatial information can direct processing to appropriate locations. Context can also be used to aid in the classification of groups of uncertain objects by bootstrapping, with more easily recognizable objects aiding in the classification of difficult ones and reinforcing the original classification.

Our current research plans entertain the possibility of learning the spatial relations between objects in the scene. Constellation models[8] encode varying spatial configurations of model parts for recognition by Gaussian distributions of relative part location learned from examples. Conditional random fields now can describe spatial relations between regions[36, 41]. Both of these representations are 2D, where full object spatial relations would have to be expressed in 3D. Where do we get the information for learning spatial configurations of objects in a robot's world? Several possibilities arise: housing plans, virtual worlds as in electronic games, online contests such as the ESP game, and finally the world itself, via exploration, and object recognition and discovery.

By teaching a robot the categories of the objects in its environs we can avoid the difficult problem of categorization. But will RFID technology obviate this work entirely, by making object self-identifying? This is possible in some not too near future, when all legacy appliances have been abandoned, but the task will remain problematic for some while, and issues of privacy may prevent the wide adoption of self-identifying objects.

In [10] Bill Gates predicts that robotics will be an exciting field in the coming years, while promoting Microsoft's Robot Studio as the software substrate for solving the difficult control software problems such as concurrency and coordination. As mentioned in Gates' article, it can be assumed that the robot agent often works in an environment where networks of cameras observe the activity of the people and the robot. Many projects have constructed test homes rich with sensors and context aware components. It has become apparent that many of the techniques that have been developed with a view toward applications in surveillance, which often use networks of cameras, also apply to the world of service robots[20]. The robot can also be instrumental in recovering the relative geometry of the cameras as it explores its world[25].

10.5 Acting in Space

José [6] is a visually-guided mobile robot that uses our visual processes, localization, mapping, navigation and human-robot interaction, in the context of a particular robotic task: serving food to a gathering of people. To accomplish this task, *José* must reliably navigate around a room populated by groups of people, politely serving appetizers to humans. The robot must also monitor the food it has available to serve, and return to a home base location to refill when the food is depleted. Problems specific to the serving task are also solved using vision, including finding people to serve and monitoring food. As well, the robot's success depends significantly on its interaction with users, its "persona", and its ability to generate appropriate actions and responses. In *José*'s planner, we identified specific locations with their tasks such as refilling the food. Other locations of interest, such as where unserved people are standing, depend on the history of the actions of the robot. Similarly a service robot will include in its behaviours defined places where activities occur: the kitchen, the laundry, for example. Tying the location and its appearance to the sensory process is the problem of *grounding*, of determining the connection between beliefs and physical objects and situations, through the sensed data.

We argued in [16] that robots can be modeled as a set of capabilities, processes that enable behaviours. A collection of tasks can be performed in a role. For example, our robot enacted the *José* role of waiting as well as the Homer (Human Oriented MEssenger Robot)[7] role. It is tedious if not infeasible to engineer each role for the robot out of behaviours. It is more sensible to model the environment, the robot, and its actions by a POMDP[16], which has proven its worth in complex assistive applications[1].

10.5.1 Selecting Features for Actions

Tasks[16] organize a workspace into sequences of places, while actions and their effects determine the order of the places. The places then facilitate their associated actions by providing the necessary objects. When designing planners, whether based on traditional, though reactive planners, as in [23], or POMDPs, there is a classification or recognition step in which the fluent in the planner or the state variable in the POMDP estimates its discrete state; we envision this process as identifying the place where the robot is and its actions.

In [11] we demonstrated how to select visual features of the facial communication pattern in a game, essentially finding the correct aspect of the sensory stream to correspond with the action. It is important to link the actual elements of the visual stream that are relevant to a particular task, by representing the connections between the utility of the action and the sensed features, within the confines of a task. Reinforcement learning is the usual solution for learning such connections, but it is infeasible to test thoroughly the enormous state space.

The solution we suggest is to "teach" the robot, rather than have it learn by trial and error. That is, the robot must import the cultural descriptions that adhere to places, which take their meaning from the actions there. The form of these descriptions will be ontologies of objects and models of their dynamics under actions, expressed as Bayes nets, for example. For visually guided robots, we will need to ground the objects in appearances, and connect them with their spatial organization. Then we can sense, reason, decide, and act with them. Already we have some of the tools for connecting hierarchical spatial descriptions with probabilistic reasoning (Smyth and Poole, [32]). There, a system is developed for reasoning about general hierarchical models combined with qualitative information about the distribution of properties; the running example is rooms and types of rooms in a house.

10.6 Summary

The challenge of visually guided robotics, particularly as partners with humans, is to progress from the laboratory into the home. Within constrained applications such as assistive technologies, the more limited semantics of the scene and smaller range of activities will likely lead to more rapid progress. To perform tasks robots need more than the ability to construct maps and recognize objects. They must be able to identify the relationships and structures of the required objects and actions. Context then is a recognizable configuration of objects which is a product of the spatial requirements of the task that the objects can entail. Thus, by endowing locations and objects with semantic tags and context, this cultural information situates the embedded system within a richer context useful for performing activities.

The success of these approaches depends on the ability of the robot to assess the spatial priors over contexts and objects. In any particular application, there has to be a balance between narrow focus (feature detection for localization, for example) and broad support (context identification, categorization). The tradeoff between these two will be driven by the reliability of visual sensing.

The challenge at hand is to acquire the structured information about the world, the ontology of objects and their spatial organization and appearance, connect it to maps, and formulate robotic controllers to accomplish complex tasks. Exploration, recording of layout and appearance, and mining ontologies will deliver valuable maps, contexts, and theories of the world.

Acknowledgements The authors gratefully acknowledge the contribution of the National Science and Engineering Research Council of Canada and GEOIDE, a Canadian Network of Centres of Excellence. As well, many past and present members of our lab have contributed to the development of the ideas explored here.

References

1. Boger, J., Hoey, J., Poupart, P., Boutilier, C., Fernie, G., Mihailidis, A.: A planning system based on markov decision processes to guide people with dementia through activities of daily living. IEEE Transactions on Information Technology in BioMedicine **10**(2), 323–333 (2006)
2. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: Proceedings of the IEEE Int. Conf. on Computer Vision, pp. 1403–1410 (2003)
3. Ekvall, S., Jensfelt, P., Kragic, D.: Integrating active mobile robot object recognition and slam in natural environments. In: Proc. of the IEEE/RSJ International Conference on Robotics and Automation (IROS'06). Beijing, China (2006)
4. Elder, J., Prince, S., Hou, Y., Sizintsev, M., Olevskiy, E.: Pre-f and attentive detection of humans in wide-field scenes. International Journal of Computer Vision **72**(1), 47–66 (2007)
5. Elfes, A.: Using occupancy grids for mobile robot perception and navigation. IEEE Computer **22**(6), 46–57 (1990)
6. Elinas, P., Hoey, J., Lahey, D., Montgomery, J.D., Murray, D., Se, S., Little, J.J.: Waiting with jose, a vision-based mobile robot. IEEE International Conference on Robotics and Automation pp. 3698–3075 (2002)
7. Elinas, P., Hoey, J., Little, J.J.: Homer: Human oriented messenger robot. AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments pp. 45–51 (2003)
8. Fergus, R., Perona, P., Zisserman, A.: Weakly supervised scale-invariant learning of models for visual recognition. International Journal of Computer Vision **71**(3), 273–303 (2007)
9. Ferrari, V., Tuytelaars, T., Van Gool, L.: Simultaneous object recognition and segmentation from single or multiple model views. International Journal of Computer Vision **67**(2), 159–188 (2006)
10. Gates, B.: A robot in every home. Scientific American **296**(1), 58–65 (2007)
11. Hoey, J., Little, J.J.: Value-directed human behavior analysis from video using partially observable markov decision processes. IEEE Trans. Pattern Anal. Mach. Intell. **29**(7), 1118–1132 (2007)
12. Itti, L., Baldi, P.: A principled approach to detecting surprising events in video. In: CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, pp. 631–637. IEEE Computer Society, Washington, DC, USA (2005)
13. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Anal. Mach. Intell. **20**(11), 1254–1259 (1998)
14. Kuipers, B.: The spatial semantic hierarchy. Artificial Intelligence **119**, 191–233 (2000)
15. Lensch, H.P.A., Kautz, J., Goesele, M., Heidrich, W., Seidel, H.P.: Image-based reconstruction of spatial appearance and geometric detail. ACM Trans. Graph. **22**(2), 234–257 (2003). DOI http://doi.acm.org/10.1145/636886.636891
16. Little, J.J., Hoey, J., , Elinas, P.: Visual capabilities in an interactive autonomous robot. In: H.I. Christensen, H.H. Nagel (eds.) Cognitive vision systems : sampling the spectrum of approaches, 365. Springer Verlag, Berlin et al. (2006)
17. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)
18. Mark, D.M., Smith, B., Tversky, B.: Ontology and geographic objects: An empirical study of cognitive categorization. In: COSIT '99: Proceedings of the International Conference on Spatial Information Theory: Cognitive and Computational Foundations of Geographic Information Science, pp. 283–298. Springer-Verlag, London, UK (1999)
19. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. Proceedings of British Machine Vision Conference, September 2002 **I**, 384–393 (2002)
20. Medioni, G.G., Francois, A.R., Siddiqui, M., Kim, K., Yoon, H.: Robust real-time vision for a personal service robot. Computer Vision and Image Understanding ((to appear))

21. Mihailidis, A., Elinas, P., an Jesse Hoey, J.N.B.: Pervasive computing to enable the mobility of older adults with cognitive impairments: An anti-collision system for a powered wheelchair
22. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2005)
23. Mueller, A., Beetz, M.: Designing and implementing a plan library for a simulated household robot. In: AAAI Cognitive Robotics Workshop (2006)
24. Pentney, W., Popescu, A.M., Wang, S., Kautz, H.A., Philipose, M.: Sensor-based understanding of daily life via large-scale use of common sense. In: AAAI. AAAI Press (2006)
25. Rekleitis, I., Meger, D., Dudek, G.: Simultaneous planning, localization, and mapping in a camera sensor network. Robotics and Autonomous Systems **54**(11), 921–932 (2006)
26. Rimey, R.D., Brown, C.M.: Control of selective perception using bayes nets and decision theory. Int. J. Comput. Vision **12**(2-3), 173–207 (1994)
27. Se, S., Lowe, D.G., Little, J.J.: Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. Intl. Jour. Robotic Research **21**(8), 735–760 (2002)
28. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(8), 888–905 (2000)
29. Sim, R., Elinas, P., Little, J.: A study of the rao-blackwellised particle filter for efficient and accurate vision-based slam. International Journal of Computer Vision **74**(3), 303–318 (2007)
30. Sim, R., Little, J.J.: Autonomous vision-based exploration and mapping using hybrid maps and Rao-Blackwellised particle filters. In: Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS), pp. 2082–2089. IEEE/RSJ, IEEE Press, Beijing (2006)
31. Sivic, J., Schaffalitzky, F., Zisserman, A.: Object level grouping for video shots. International Journal of Computer Vision **67**(2), 189–210 (2006)
32. Smyth, C., Poole, D.: Qualitative probabilistic matching with hierarchical descriptions. In: KR, pp. 479–487 (2004)
33. Southey, T., Little, J.J.: Object discovery using motion, appearance and shape. In: AAAI Cognitive Robotics Workshop (2006)
34. Tapia, E.M., Choudhury, T., Philipose, M.: Building reliable activity models using hierarchical shrinkage and mined ontology. In: Pervasive, pp. 17–32 (2006)
35. Thrun, S., Fox, D., Burgard, W.: Monte carlo localization with mixture proposal distribution. In: Proceedings of the 2000 National Conference of the American Association for Artificial Intelligence (AAAI), pp. 859–865 (2000)
36. Torralba, A., Murphy, K., Freeman, W.: Sharing visual features for multiclass and multiview object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(5), 854–869 (2007)
37. Torralba, A., Murphy, K., Freeman, W., Rubin, M.: Context-based vision system for place and object recognition. In: Nice, France, Oct 13-16, pp. 273–280 (2003)
38. Tsotsos, J., Culhane, S., Wai, W., Lai, Y., Davis, N., Nuflo, F.: Modeling visual-attention via selective tuning. Artificial Intelligence **78**(1-2), 507–545 (1995)
39. Vasudevan, S., Gächter, S., Nguyen, V., Siegwart, R.: Cognitive maps for mobile robots-an object based approach. Robot. Auton. Syst. **55**(5), 359–371 (2007)
40. Vogel, J., Murphy, K.: A non-myopic approach to visual search. In: CRV '07: Proceedings of the Fourth Canadian Conference on Computer and Robot Vision, pp. 227–234. IEEE Computer Society, Washington, DC, USA (2007). DOI http://dx.doi.org/10.1109/CRV.2007.5
41. Winn, J., Shotton, J.: The layout consistent random field for recognizing and segmenting partially occluded objects. In: CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 37–44. IEEE Computer Society, Washington, DC, USA (2006)

Chapter 11
Vision-Based Navigation Strategies

Darius Burschka

11.1 Motivation

Navigation and localization are important capabilities of mobile systems allowing definition of mission goals. Only the knowledge about the absolute and relative position in an indoor or outdoor environment allows a free definition of mission goals and path planning in areas not previously traversed by the system. A typical concurrent goal is the reconstruction of coherent 3D geometric representations of arbitrary indoor or outdoor environments from a configurable set of sensors. This representation is typically used as a reference for localization. The sensor configuration is thereby defined by the required accuracy and system costs. We investigate monocular and binocular cameras, laser range finders, and inertial systems as input sources for this task. The minimal hardware configuration of such a system is a monocular camera that can be supported by additional sensors to enhance the quality of the reconstructed models. The goal is to replace expensive inertial systems with a set of low-cost sensors, like video cameras available on most current computer systems. The necessary accuracy is achieved through fusion of information over a sequence of images. The idea is to replace expensive hardware with appropriate algorithmic techniques to compensate for the imperfections of the low-cost sensors.

An important milestone towards a high accuracy reconstruction of the environment is an exact localization in an unknown or partially known environment. The reference model for localization needs often to be extracted in parallel to the actual localization task. This process is known in the literature as *Simultaneous Localization and Mapping* (SLAM). The localization is necessary to fuse the sensor readings from different positions to a consistent and complete 3D model.

In this chapter, we will focus on the localization task from a video camera. We assume a video camera mounted on a mobile system. The localization implicates several challenges. The first challenge is an accurate estimation of the 3D pose pa-

Technische Universität München, Department of Computer Science, 80333 München, Germany, e-mail: burschka@cs.tum.edu

rameters from the available sensor data. Another challenge is to perform the localization in situations, where the reference points or landmarks as we will refer to them in the following text are not known a-priori and need to be estimated in parallel to the localization process. We propose systems that are capable of simultaneous localization of the camera and navigation relative to obstacles in the world.

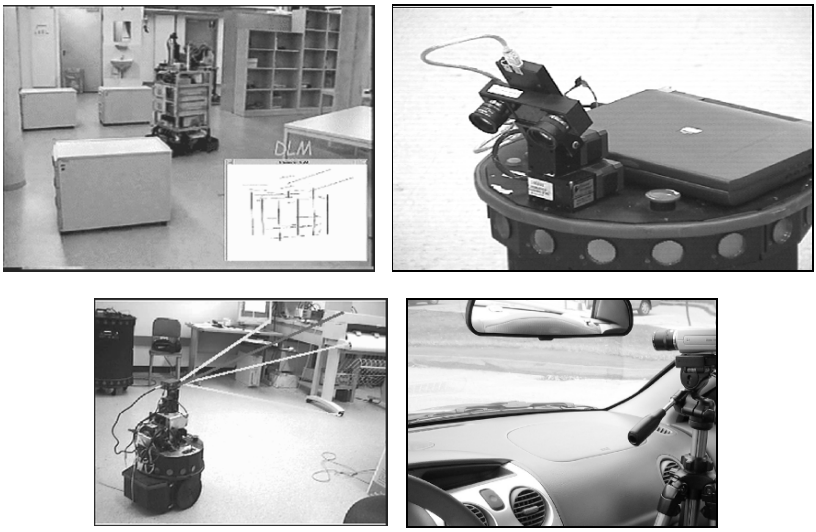


Fig. 11.1 Different types of navigation systems using video cameras: (Marvin) 3D exploration of indoor environments with stereo [4], (Speedy) obstacle avoidance from dense disparity maps [7], (Goomba) sentry robot using vision-based control for navigation [5], (car application) Visual SLAM for traffic sign detection [8].

The problem of *Simultaneous Localization and Mapping*, also known as SLAM, has attracted immense attraction especially in the mobile robotics literature. SLAM addresses the problem of building a map of an environment from a sequence of landmark measurements obtained from a moving system. Since the motion especially of hand-operated devices is unknown, the mapping problem induces a localization problem. The partial 3D reconstructions can only be fused to a complete model given an accurate relative localization between them. A solution to the SLAM problem using Kalman Filters was introduced in a paper by Smith, Self, and Cheeseman [23]. This paper proposed the use of the Extended Kalman Filter (EKF) for incremental estimation of the posterior distribution over the robot pose along with the positions of the landmarks. While many popular SLAM implementations use laser range information as input to the process to simplify the estimation to pure localization task, we present an extension to vision-based techniques. The challenge here is to obtain the necessary information for the SLAM process from a monocular camera as input source.

We developed a variety of navigation systems using different approaches ranging from mission planning based on explored 3D models from a binocular setup [4], over localization relative to obstacles in the world from stereo [7] to monocular approaches based on visual servoing [5] and a recently developed visual *simultaneous localization and mapping* system ((VGPS)SLAM) [10] (see Fig. 11.1). We will give a short evaluation of the advantages and disadvantages of these systems and discuss the next steps in our current research.

11.1.1 Related Work

The problem that we address here is a simultaneous estimation of the motion parameters R,T (rotation and translation) and the depth information as a metric distance to the observed points. We address the extension of typical SLAM based on laser range finders to monocular cameras.

There exist solutions to pose estimation for 3 point correspondences for most traditional camera models, such as for example orthographic, weak perspective [1], affine, projective [14, 18] and calibrated perspective [19]. These approaches constrain the possible poses of the camera to up to four pairs of solutions in the case of a calibrated perspective camera. At most one solution from each pair is valid according to the orientation constraints and the other solution is the reflection of the camera center across the plane of the three points.

Many localization approaches for indoor applications use simplifications like assumptions about planarity of the imaged objects in the scene or assume a restricted motion in the ground plane of the floor that allows to derive the metric navigation parameters from differences in the images using Image Jacobians in vision-based control approaches. A true 6DoF localization requires a significant computational effort to calculate the parameters while solving an octic polynomial equation [22] or estimating the pose with a Bayesian minimization approach utilizing intersections of uncertainty ellipsoids to find the true position of the imaged points from a longer sequence of images [12]. While the first solution still requires a sampling to find the true solution of the equation due to the high complexity of the problem, the second one can calculate the result only after a motion sequence with strongly varying direction of motion of the camera that helps to reduce the uncertainty about the position of the physical point. In the work of Nister [22], an approach sampling for the correct solution along the rays of projection solving an octic polynomial to find the actual camera pose is presented. It is limited to exactly 3 points neglecting any possible additional information. While it represents a direct solution to the problem, the high order of the polynomial and the typical noise in real images makes this solution still very complicated and sensitive to noise.

Our system is motivated by the same idea as the system presented in [17], where a tracking approach for "2.5D space" was proposed. The system is supposed to compensate for the drawbacks of classical position-based visual servoing. In the approach presented in [17], eight landmarks are necessary to estimate the pose of an

object in space. A reduction to four points is only possible in case that four co-planar points can be identified. The co-planarity constraint is a special case that is difficult to enforce in all situations. Additionally, a robust tracking of eight landmarks in the image is contradictory to our goal to build a compact system running on hardware with limited computational power that can usually be found on mobile systems. The smaller the number of landmarks that we need to track, the more processing power can be dedicated to other important tasks on the robot.

Our pose estimation is based on an image-based approach that compares the 2D projections of an *internal 3D model* between images. The *internal 3D model* is estimated up to scale due to the limitations in the perception of a monocular camera system (see Section 11.3.3.1). In [11] a recursive model-based object pose estimation is presented that is based on orthographic projection of points onto camera image. This approach is limited to configurations that can be projected onto a planar image. In our case, we propose a pose estimation method allowing robust pose verification from 3 tracked landmarks that can be placed anywhere around the sensor. Our approach operates in image coordinates of the camera using a novel representation for the 3D model that does not require any knowledge about the three-dimensional position in the world to register the reconstructions to each other.

We propose an approach that we validated in a wide range of applications ranging from reconstructions from endoscopic medical images to 3D scene reconstructions in outdoor environments.

We assume to know the initial 3D structure of at least 3 points in the world P_i with known correspondences in the image frame n_i. The system is initialized manually or automatically with an initial set of feature correspondences with a known metric relation and it maintains these correspondences through tracking in color or texture. It adds new features to the set to compensate for loss of features that become occluded or that disappear from the field of view. Further, we assume also a calibrated camera measuring directly the angles of incidence. In this chapter, we focus on strategies for depth recovery from spherical projection and the motion+structure update.

In the following Section 11.2, we give an overview of possible navigation approaches with a discussion of their advantages and disadvantages. In Section 11.3, we describe the way the information about the depth changes due to motion and how the motion itself is calculated. In Section 11.3.3, we discuss the open challenges for our monocular navigation system. The accuracy of the algorithm is evaluated in Section 11.4. We conclude in Section 11.5 with an overall evaluation of the presented system and present our future research goals.

11.2 Navigation Alternatives

In this section, we present an overview of navigation approaches that we implemented on our mobile systems in the course of the past years. They lead us to our fi-

nal approach based on monocular VSLAM. We also discuss lessons that we learned about the advantages and disadvantages of each of the approaches.

We distinguish between map-based systems that build an intern representation of the environment as a 2D- or 3D-model and image based approaches deriving the information directly from an error between an expected and an observed position of a physical point in the world.

11.2.1 Map-based Navigation

Map-based navigation systems represent an approach to global navigation. The navigation is based on 2.5D or 3D models of the environment. A 3D model as a global reference allows a localization relative to a specific physical reference point in the world in opposite to a relative localization between two sensor frames. Relative localization is common in image-based approaches.

The sensor information is abstracted to a 3D representation and fused from all sensor readings to a consistent global or local model of the environment. The model allows planning of arbitrary missions in the environment that may traverse locations which were only perceived by the sensors, but which were never actually passed in previous missions. It is possible, because the 3D model allows a prediction of any new sensor view in the world even for new locations. Their advantage is the flexibility allowing planning of arbitrary missions even in regions which were not traversed before, but a significant disadvantage is the necessity of fusion of information from the sensor readings requiring an exact localization over a long period of time to allow a correct registration of all sensor readings in an area. An additional disadvantage is the abstraction of the information from the direct sensor data to three-dimensional descriptions which are prone to errors due to calibration errors in the system.

11.2.1.1 Binocular 3D Reconstruction

At the Lab for Real-Time Computer Systems of the Technical University in Munich, we built a mobile robot *Marvin* that reconstructs the 3D world model from the perception of a binocular camera system [4]. The system is depicted in Fig. 11.2.

The system reconstructs three-dimensional line segments representing the boundaries of human made objects. The line segments are stored in a local map (DLM) fusing the consecutive sensor readings from the sensor in Fig. 11.3. The map is the central element of the navigation system. It decouples the three major information flow loops in the system marked as colored regions in Fig. 11.3. They all operate with different cycle times. There is a fast bidirectional information exchange between the local map DLM and the sensor system that predicts expected information for the current view based on the 3D map content and stores back the current reconstruction.

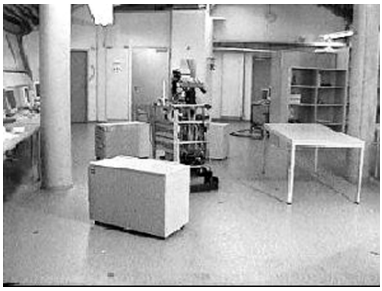


Fig. 11.2 Exploration system *Marvin* using binocular stereo.

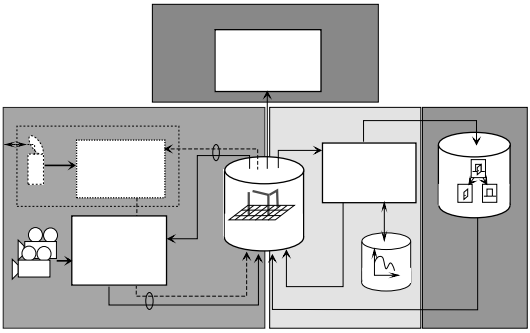


Fig. 11.3 The information flow in the map-based exploration system *Marvin*.

This loop helps to reduce false matches in stereo processing (3D reconstruction module), because expected information is predicted for each step increasing the matching score of correct matches between line segments in the stereo images. It helps also to filter out wrong matches that cannot be verified in a current frame in Fig. 11.4.

This filtering is based on the assumption that correct matches will always occur close to their true position with a small positional error due to localization and camera calibration errors. False matches move to varying locations depending on the viewing position. Correct entries in the map correspond to line segments that could be reconstructed from different viewing positions in a local environment.

The reconstructed line segments can be abstracted to polygons or even objects in the spatial prediction module in the right block in Fig. 11.3. This can provide additional hypotheses about missing lines based on assumptions about underlying structures that can be provided to the sensor system as local predictions to be verified or discarded in the current view. This module operates outside of the fast sensor loop and does not interfere with the sensor processing directly. The calculated information about missing parts of hypothetical objects identified in the data is inserted asynchronously to be verified in the sensor loop when the corresponding region comes into sensor view.

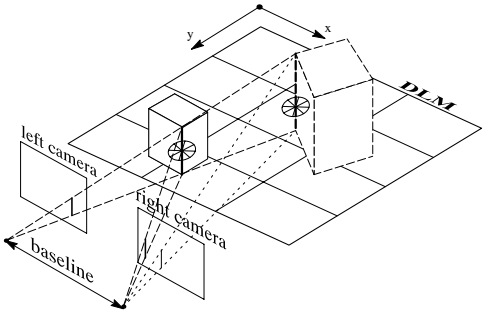


Fig. 11.4 Multiple matching candidates for a given segment can be stored in the local map DLR to be verified from a different location.

11.2.1.2 Monocular VSLAM Systems

In many cases, 3D reconstruction is necessary in large distances to the camera system to allow pre-selection of interesting objects for a mission far ahead before the system moves closer to them. A typical example is a car navigation system, where the high speed of motion requires analysis of objects in large distance to the car to give enough time for decision. An example can be a traffic sign detection system

Fig. 11.5 Large baseline can easily be constructed utilizing the motion of the vehicle with monocular systems.

that analyzes candidates for signs as soon as they become visible [10]. The relationship between the depth, z, of a scene point and its disparity, D, in two images separated by baseline B is given by [14]:

$$D = \frac{B \cdot f}{p} \cdot \frac{1}{z}, \tag{11.1}$$

where f is the focal length of the camera and p is the pixel-size on the camera chip. This relationship is shown in Fig. 11.6 for several values of the distance between the stereo cameras B. From the graph, it is clear that the larger the value D for a given landmark, the better the signal to noise ratio of the resulting reconstruction.

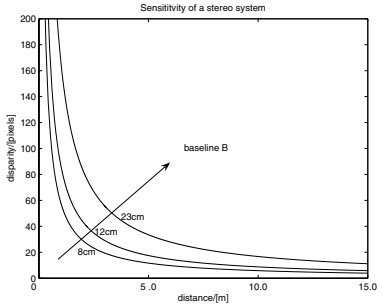


Fig. 11.6 The disparity value drops rapidly with the distance to the imaged landmark.

Some ways of increasing depth accuracy include increasing f (at the cost of field of view) or decreasing p (i.e. using a higher resolution camera). The former is generally limited by the need for a reasonably wide field of view; the latter is limited by the bandwidth and processing necessary to handle higher resolution images.

As a result, the only real flexibility is in the baseline. There are natural limits set on the maximum width of a binocular system. These limits can be defined, e.g., by the width of the car. Any further increase requires a change to a monocular reconstruction that uses the motion of the system as a baseline for reconstruction (see Fig. 11.7).

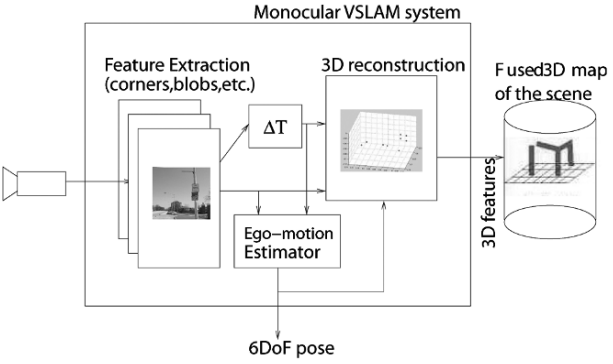


Fig. 11.7 Monocular VSLAM system tracking positions of point feature in a sequence of images to reconstruct their 3D position in parallel to estimation of the motion parameters of the camera system.

Feature points representing a specific 3D point in the world are tracked in a sequence of images to estimate both their 3D position and the relative motion of the camera to them in a SLAM *(simultaneous localization and mapping)* approach.

Like in the case of a binocular reconstruction, the resulting system constructs a global or local 3D model containing in our case points representing centers of unique patterns in the world. An accurate localization is necessary to fuse the single

reconstructions from consecutive steps making the processing more complex and sensitive to reconstruction and localization errors. In case that the system is used for absolute localization, special care needs to be taken to keep the resulting errors small. Relaxation techniques known from laser based SLAM approaches are applied to reconstructed data to minimize the error.

The monocular approach can also be used as a relative localization system providing just position changes between consecutive sensor readings. In this case, just the image position of corresponding feature points is analyzed in two image frames allowing an estimation of relative motion without a necessity of fusion with information from previous steps. Here, no global localization is performed. Our monocular SLAM implementation is a generalization of the Vision-Based Control (VBC) navigation described in Section 11.2.2.1 below. This generalization allows large displacements between the acquisition points of both images, because an analytic pose estimation is used instead of a local linearization used in the image Jacobian from VBC. In both cases, a displacement to a reference pose is calculated.

11.2.2 Image-Based Navigation

Many of the complicated house keeping methods to ensure correct global pose and exact 3D reconstruction that are necessary for correct data fusion can be avoided in image-based navigation approaches. Usually, these approaches do not provide an absolute localization relative to a physical reference point in the world. They calculate merely a relative motion between sensor readings instead. A fusion to an absolute pose can be done outside of the navigation module. This navigation method corresponds more to an inertial unit estimating just changes instead of integrating them to an absolute value. Possible localization errors appear as noise on the top of the relative pose estimation values.

11.2.2.1 Vision-Based Control

Many applications of mobile systems involve repeating tasks that require a robot to move along a pre-defined path. The system does not require significant flexibility in the choice of the paths, but high robustness is required for a long term operation. A typical task for this type of systems is a sentry robot or mail-delivery robot repeating the same paths in each mission. To avoid the localization problems of the map-based approaches that suffer from calibration errors which may occur due to vibrations during operation, these systems use directly the images as a ”model” to store the correct path. Changes in the projection between an expected and the actual position of a landmark feature are used to calculate the pose error. This is a relative localization error relative to the pose from where the reference image was taken. The system does not need to have any knowledge about the absolute pose in the world at any time.

We explain the method at an example of a robot moving in a plane of the floor that restricts its motion to the two dimensions of the plane (x,z) and the orientation Θ in Fig. 11.8.

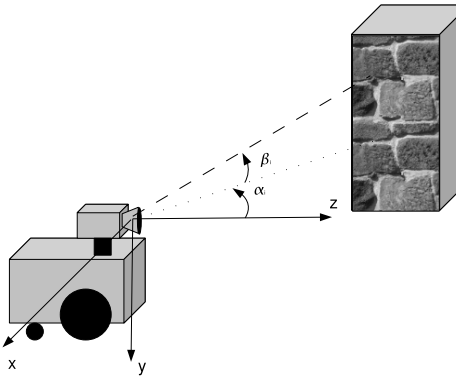


Fig. 11.8 The system estimates motion errors based on the difference in the observation between the expected and the actual position of an imaged point.

A camera system is an angle measuring device. It estimates the direction (α_i, β_i) from which a specific point in the world can be seen. The radial distance along the line of sight is lost in the projection. Each observation with the metric pixel coordinates (u_i, v_i) for a focal length f=1 can be converted into two angles (α_i, β_i) describing the azimuth and elevation values for a given observation to:

$$\alpha_i = \arctan u_i = \arctan\left(\frac{x_i}{z_i}\right), \quad \beta_i = \arctan\frac{v_i}{\sqrt{1+u_i^2}} = \arctan\frac{y_i}{\sqrt{x_i^2+z_i^2}}$$

Assuming motion in the plane, we can compute the following image Jacobian relating the change of angles in the observation, (α_i, β_i), due to changes in motion in the plane, (x_i, z_i) to:

$$\mathscr{J}^t = \begin{pmatrix} \frac{\partial\alpha_i}{\partial x_i} & \frac{\partial\alpha_i}{\partial z_i} & \frac{\partial\alpha_i}{\partial\Theta_i} \\ \frac{\partial\beta_i}{\partial x_i} & \frac{\partial\beta_i}{\partial z_i} & \frac{\partial\beta_i}{\partial\Theta_i} \end{pmatrix} = \begin{pmatrix} \frac{z_i}{x_i^2+z_i^2} & -\frac{x_i}{x_i^2+z_i^2} & -1 \\ -\frac{x_i y_i}{(x_i^2+y_i^2+z_i^2)\cdot\sqrt{x_i^2+z_i^2}} & -\frac{y_i z_i}{(x_i^2+y_i^2+z_i^2)\cdot\sqrt{x_i^2+z_i^2}} & 0 \end{pmatrix} \tag{11.2}$$

The dependency on the Cartesian coordinates can be avoided considering the geometry of the system to:

$$R_i = \sqrt{x_i^2 + y_i^2 + z_i^2} = \frac{y_i}{\sin\beta_i} \quad r_i = \sqrt{x_i^2 + z_i^2} = \frac{y_i}{\tan\beta_i}$$

$$\mathscr{J}_i^t = \begin{pmatrix} \frac{z_i}{r_i^2} & -\frac{x_i}{r_i^2} & -1 \\ -\frac{x_i y_i}{R_i^2 \cdot r_i} & -\frac{y_i z_i}{R_i^2 \cdot r_i} & 0 \end{pmatrix} = \begin{pmatrix} \frac{\tan\beta_i \cdot \cos\varphi_i}{y_i} & -\frac{\tan\beta_i \cdot \sin\varphi_i}{y_i} & -1 \\ -\frac{\sin^2\beta_i \cdot \sin\varphi_i}{y_i} & -\frac{\sin^2\beta_i \cdot \cos\varphi_i}{y_i} & 0 \end{pmatrix}$$

Note that the image Jacobian is a function of only one unobserved parameter, y_i, the height of the observed point. Furthermore, this value is *constant* for motion in the plane. Thus, instead of estimating a time-changing quantity as is the case in most vision-based control, we only need to solve a simpler static estimation problem for a constant value y_i in case of the motion in the floor plane.

This system is very robust to errors in the calibration, since the goal of the processing is to correct an image error to zero, which is independent of the estimates of the focal length and radial lens distortions. These errors usually just cause the system to assume a too large deviation. The correct alignment is still detected correctly.

11.2.2.2 Disparity-based Navigation

Obstacle avoidance systems are essential to protect robots from collisions with the environment or driving towards staircases or gaps (*negative obstacles*) while operating in unknown or partially known environments. Many obstacle avoidance systems are based on sensors that provide direct 3D measurements, such as laser range finders and sonar systems [3, 13]. In some cases, e.g. [16], cues from a monocular camera combined with prior knowledge of supporting surface geometry and appearance have been used. In contrast, our system relies completely on the data from a real-time stereo system with relative few prior assumptions.

Disparity images are pseudo-images, where each pixel value corresponds to the disparity D (reciprocal value to the depth distance z, see (11.1)). Two example of such images are depicted in Fig. 11.9 below. The goal of the binocular system is to recover all planar structures with a given size and position in space in the current camera view. In previous work [6], we describe a system that was able to recover supporting planes from binocular stereo images to detect obstacles in the scene. This approach relied on the fact that there is a homography between the (u,v,D) coordinates of a disparity image ([u,v]-image coordinates and disparity D) and the corresponding Cartesian coordinates from the 3D scene. Here we sketch how we use the idea to locate and estimate planar structures.

Following the derivation in [6], given a plane $\mathscr{P}_r$ in R^3,

$$\mathscr{P}_r : a_r x + b_r y + c_r z = d_r \tag{11.3}$$

the equivalent disparity plane is given by

$$\forall z \neq 0: \quad a_r \frac{x}{z} + b_r \frac{y}{z} + c_r = \frac{d_r}{z} \tag{11.4}$$

$$a_r u + b_r v + c_r = k \cdot D(u,v) \tag{11.5}$$

$$\text{with } u = \frac{x}{z},\ v = \frac{y}{z},\ k = \frac{d_r}{B}. \tag{11.6}$$

where $D(u,v)$ represents the disparity at image coordinates (u,v). Clearly, (11.4) describes a plane in UVD space. We can write (11.4) in the following form

$$D(u,v) = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \end{pmatrix} \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{n}_\mathbf{r}^* \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{11.7}$$

$$\text{with} \quad \rho_1 = \frac{a_r}{k},\ \rho_2 = \frac{b_r}{k},\ \rho_3 = \frac{c_r}{k} \tag{11.8}$$

All pixels in the disparity image that have the in (11.7) predicted disparity value are removed form the image and the remaining pixels are treated as obstacles. Ground suppression is fundamental for the entire process. An example of a suppression is shown in Fig. 11.9. It shows the resolution of the system, which is capable of distinguishing between the ground plane and objects as low as 1cm above the ground at a distance of up to 3m. The newspaper disappears as an obstacle as soon as it lays flat on the ground. Each image triple shows the real image in the upper left corner, the computed disparity image in the upper right corner and the detected obstacles at the bottom.

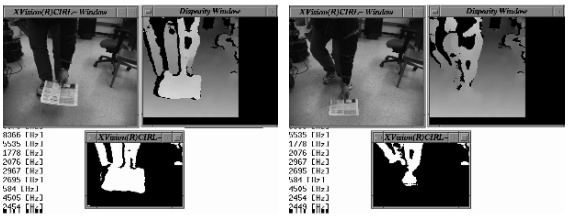


Fig. 11.9 The newspaper is classified as obstacle left, but it disappears in the right image.

The common feature of this navigation category is that all the necessary information is derived from the current sensor reading without any necessity of fusion between different readings. In this case, like already in section 11.2.2.1, we get the navigation information directly from the image itself.

11.3 Monocular VSLAM Approach

Since a typical video camera measures only the angle of incidence of the incoming rays of light, it is useful to remove the dependency on the physical imaging properties of the sensor and introduce a more generic sensor model. We decided to use a spherical projection model for our system, where every 3D-point P_i is represented as a unit vector n_i pointing in its direction:

$$n_i = \frac{P_i}{||P_i||} \quad \vee \quad n_i = \frac{(u\ v\ 1)^T}{||(u\ v\ 1)^T||} \tag{11.9}$$

We see in (11.9) that there is a simple relation between the uni-focal (focal length=1) image coordinates (u, v) and the projection on the sphere n_i.

In the remainder of this section we describe the way, how the motion parameters (R, T) and the changing 3D structure elements $\{D_i\}$ are recovered. The motion parameters represent a delta motion to the previous or a reference frame, but, for simplicity, we will omit the Δ expression in front of them. We use image-based tracking to maintain the correspondences between the image frames.

11.3.1 3D-Reconstruction

Analogous to the typical binocular approach, the 3D information is extracted using additional information from a second image or an initial 3D reference model.

11.3.1.1 Reconstruction of Unknown Points.

This processing step is necessary to recover the depth structure for new points that appeared in the camera images and need to be added to the tracking process. This processing can also be used for dual camera systems (e.g., two omnidirectional cameras) with known, calibrated displacement (R, T).

In a parallel binocular system with a distance B between the cameras, the normal distance Z to an imaged point P_i is estimated from a horizontal shift (metric disparity) d between both images [25] to

$$Z = \frac{B \cdot f}{d}, \quad \text{f - focal length of the cameras} \tag{11.10}$$

Since we deal here with a monocular system that reconstructs only sparse information about a few corresponding points, we want to avoid any warping operation to the parallel case. In typical structure-from-motion applications, the translation T is known only up to scale $\frac{T}{m}$ [25]. Therefore, we modified (11.14) in the following way:

$$\frac{D_i'}{m} n_i' = R \frac{D_i}{m} n_i + \frac{T}{m}$$

$$\begin{pmatrix} n_i' & -R \cdot n_i \end{pmatrix}^{-1} \cdot \frac{T}{m} = \begin{pmatrix} \frac{D_i'}{m} \\ \frac{D_i}{m} \end{pmatrix} \tag{11.11}$$

This is the *spherical disparity* equation with a similar structure to (11.10). The baseline B of the system is the distance $\frac{T}{m}$ traveled by the camera and it is "divided" by the *spherical disparity* s

$$s = \begin{pmatrix} n_i' & -R \cdot n_i \end{pmatrix}, \tag{11.12}$$

which represents a difference vector between the two projections (n_i', n_i) rotated to the coordinate frame of n_i' in which $\frac{T}{m}$ is defined. Since there is no significant plane as it is the case for the image plane of a coplanar binocular system, a normal distance definition of Z does not make any sense and it is replaced by the radial distance to the focal points of both projections $(\frac{D_i}{m}, \frac{D_i'}{m})$. The reconstructed depths are scaled down to the same scale as T. The scale m is preserved in the reconstruction. In the following text, we will assume m=1 to simplify the notation. It is easy to verify that all equation are true for any value of $m > 0$.

11.3.1.2 Update of Known Radial Distances.

The presented system maintains a set of known correspondences that was used to recover the motion. For these points, the depths $\{D_i\}$ in the previous frame or in the reference position at the origin are assumed to be known. The task is to update them to the current depth $\{D_i'\}$. Theoretically, the equation (11.11) can be used for this task, but since some of the data may represent very accurate model information, a different type of update equation is used. It takes the accurate depth information instead of the calculated motion information (R,T) into account.

Since we try to estimate the 6DoF motion, point and line features do not provide sufficient information to describe all 6 motion parameters. We have chosen a plane $\mathscr{E}$ spawned by 3 feature points $\{P_1, P_2, P_3\}$ and $\{P_1', P_2', P_3'\}$ in both images as a reference feature that is observed in both images of a sequence. The features must not be collinear. We construct two vectors $v_1 = P_2' - P_1' \wedge v_2 = P_3' - P_1'$ and describe the plane segment with the diagonal resulting from addition of these two vectors.

$$v_1 = P_2' - P_1' \wedge v_2 = P_3' - P_1'$$

$$\mathscr{D} = (D_1, D_2, D_3)^T$$

$$v_1 = D_2' n_2' - D_1' n_1' = R \cdot D_2 n_2 + T - R \cdot D_1 n_1 - T$$

$$v_2 = D_3' n_3' - D_1' n_1' = R \cdot D_3 n_3 - R \cdot D_1 n_1$$

$$v_1 + v_2 =$$

$$= (-2n_1' \, n_2' \, n_3') \cdot \mathscr{D}' = R \cdot (-2n_1 \, n_2 \, n_3) \cdot \mathscr{D}$$

$$F' \cdot \mathscr{D}' = R \cdot F \cdot \mathscr{D}$$

$$\Rightarrow \mathscr{D}' = F'^{-1} R \cdot F \cdot \mathscr{D} \tag{11.13}$$

The equation (11.13) introduces the projection matrix F that projects the *depth vector* $\mathscr{D}$ onto the diagonal vector $v_1 + v_2$ in the plane $\mathscr{E}$. It allows a recovery of the updated depth values $\mathscr{D}'$ based on the current image data that was used to construct F and F', and the known geometric structure $\mathscr{D}$ from the previous frame. The equation (11.13) shows that from a known set of relative distances $\mathscr{D}$ the new 3D structure D' after the motion can be reconstructed without any knowledge about the translation in the system T. It is an important property of this estimation system, since monocular systems are able to recover the rotation matrix R correctly, while the translation vector T is estimated only up to an unknown scale factor if there is no external metric reference in the world used.

11.3.2 Motion Recovery

The reconstruction approaches in the previous section (Section 11.3.1) assumed a knowledge of the motion parameters (R,T). In case of a monocular system, these parameters are unknown and need to be estimated in parallel to the reconstruction process. We mentioned already in the motivation section that there is no linear relation between the motion and structure parameters according to (11.14). The typical structure-from-motion approaches are able to reconstruct the motion from 5-8 point correspondences between the images. In our case, we assume to have additional information about the metric distances $\mathscr{D}$ to the imaged points that will allow us to reduce this number to a minimal set of 3 features. Our goal is to develop an algorithm that on one hand works with a minimum feature set but on the other hand accepts additional features if they are available. This is one of the important differences to the algorithm presented in [22] that operates on 3 points assuming their accurate detection. In real applications, the feature detection is error-prone and some of the

errors can be compensated by using additional features in over-determined systems of equations.

11.3.2.1 Motion-Induced Changes in the Feature Projections

The equation (11.11) describes completely the change in the projection $n_i \rightarrow n_i'$ due to arbitrary motion in all 6 degrees of freedom (R,T). The motion estimation needs to be separated from the reconstruction of the depth parameters $\{D_i, D_i'\}$.

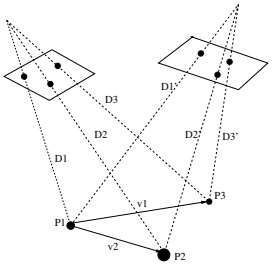


Fig. 11.10 Minimum set of three non-collinear points $\{P_1, P_2, P_3, \dots\}$ in 3D space is used to recover the motion parameters.

The equation (11.11) shows that any translation T changes the lengths $\{D_i\}$ of the associated rays of projection. On the other hand, motion is necessary for the depth reconstruction according to (11.11). We want to recover the motion from observations of a static set of points.

Recursive Algorithm for Simultaneous Motion and Structure Estimation

Since the influence of motion (R,T) and structure D_i is non-linear in

$$D_i' \mathbf{n_i'} = R \cdot D_i \mathbf{n_i} + T, \tag{11.14}$$

therefore, we need to estimate both in parallel. Instead of sampling the rays for the correct solution, we use an algorithm that we originally developed for small positional changes [9, 10], but that proves to be valid for large deviations as well.

The algorithm is based on the idea of alternating refinement of pose and structure information. For small movements in the scene, the assumption is valid that the changes are mostly in the pose parameters (especially rotation R) while the distances to the observed points remain almost the same. Therefore, for each new frame, we start with an initial guess for distances $\{\hat{D}_i^t\}$ that is chosen for each iteration step t as follows:

$$\hat{D}_i^t = \begin{cases} D_{init}, & t = 0 \\ D_i^{t-1}, & t \neq 0 \end{cases} \tag{11.15}$$

These depths are used to calculate guesses for the point positions to

$$\hat{P}'_i = \hat{D}'_i \cdot n'_i \tag{11.16}$$

The initial error for a significant change in pose is depicted in Fig. 11.11 as green(gray) lines and circles. We compute the pose change for these two initial point sets and use it to refine our guess about the depth structure $\{D'_i\}$.

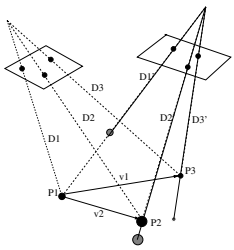


Fig. 11.11 The initial depth assumption for a very large deviation in position.

Computing the absolute orientation is the process of determining $\tilde{R}$ and $\mathbf{T}$ from corresponding pairs $\hat{P}'_i$ and P_i. With three or more non-collinear points, $\tilde{R}$ and $\mathbf{T}$ can be obtained as a solution to the following least-squares problem as described in [15].

$$\min_{R,\mathbf{T}} \sum_{i=1}^{n} \|RP_i + \mathbf{T} - \hat{P}'_i\|^2, \quad \text{subject to} R^T R = I. \tag{11.17}$$

Such a constrained least squares problem [14] can be solved in closed form using quaternions [21, 24], or singular value decomposition (SVD) [20, 2, 21, 24]. We use the SVD method to calculate the rotation matrix in our system as described in [10] in more detail. We see in (11.13) that the rotation alone is sufficient to estimate the changes in the distances to the tracked points. The corresponding translation T can be estimated from the pose change of the corresponding points (P_i, P'_i) assuming that the rotation matrix R is known.

This is an iterative approach, where the result of each iteration is used to estimate new improved guesses of the depth structure D_i.

11.3.3 Open Challenges

The presented VSLAM system was tested in different scales ranging from outdoor navigation down to navigation of endoscopes in medical applications. The system works reliably if the initial depth structure is known. The correct initialization is still an open challenge that we try to approach. The second challenge is a compensation of drifts due to accumulation of errors for the case that the presented system is used

for global localization and the noise values create an offset value deteriorating the localization quality.

11.3.3.1 Estimation of the Initial Depth Relations

The initial depth structure is usually initialized in two ways. In case that the system starts at a known location, like e.g. a landing place or a docking station, a known reference structure can be observed and a reference projection can be calculated from it. As an example, the reference structure can be a rectangle on the floor and the reference view can be a pose with an image plane coplanar to the rectangle with the focal point 1m above the center of the rectangle. Basic projection equations can be used to calculate this "virtual projection". The depth information for a current observation can now be estimated using our iterative approach (Section 11.3.2) in Fig. 11.12.

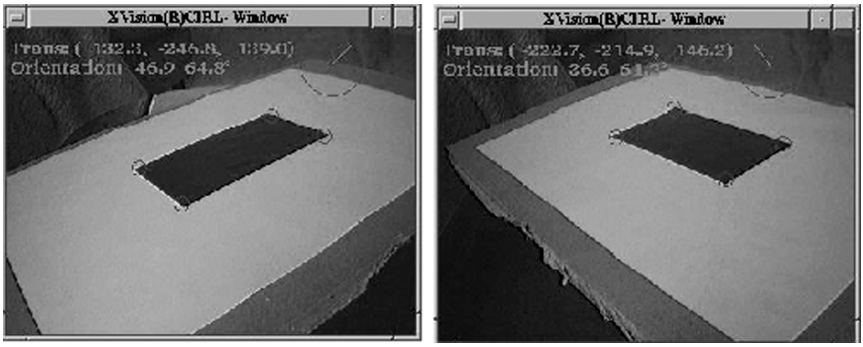


Fig. 11.12 Initialization from a known reference structure.

A second initialization method is based on the essential matrix computation [25]. The motion parameters can be estimated up to an unknown scale in the translation in Fig. 11.13. A relation between the projections p_i, p_i^* in two camera images with known internal parameters can be expressed with the Essential Matrix $\tilde{\mathbf{E}}$ [14] as

$$p_i^* \tilde{\mathbf{E}} p_i = 0 \tag{11.18}$$

The Essential Matrix $\tilde{\mathbf{E}}$ consists of a product of two matrices

$$\tilde{\mathbf{E}} = \tilde{\mathbf{R}} \cdot \mathrm{sk}(\mathbf{T}),$$

$$\text{with} \quad \mathrm{sk}(\mathbf{T}) = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix} \tag{11.19}$$

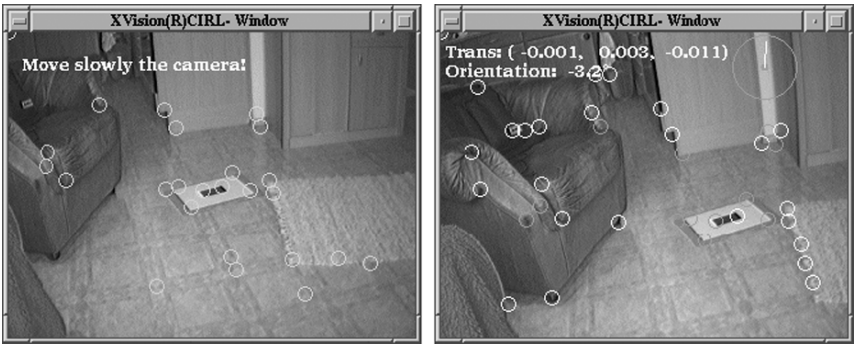


Fig. 11.13 Initialization from an initial motion. After the initial motion a subset of the initial points in the left image can still be tracked. It is used for the essential matrix method.

Note that, given a correspondence, we can form a linear constraint on $\tilde{\mathbf{E}}$. It is only unique up to scale, therefore, we need 8 matches, then we can form a system of the form $\tilde{\mathbf{C}} \cdot e = 0$ where e is the vector of the 9 values in $\tilde{\mathbf{E}}$.

The essential matrix solution gives a valid result only if the corresponding points did not lie on a super-quadric, like e.g. on a plane. This condition occurs unfortunately quite often in case of flying systems observing the ground. For such a configuration a homography matrix method needs to be used. An important decision is to recognize that a given feature set is on a plane without any knowledge about the environment. Our current solution uses the eigenvalues of the essential matrix which should be equal to (1,1,0) in the ideal case. Noise and detection errors cause them to deviate from this ideal case. A planar condition can be identified as a result with two non-zero eigenvalues with a ratio significantly larger than 1. In such a case, the homography solution is chosen.

11.3.3.2 Compensation of Drifts

A typical off-shelf perspective camera has only a limited field of view. Therefore, only a small set of landmarks is usually visible in the sensor cone with an opening angle defined by the focal length of the lens. The shorter the focal length the larger is the field of view of the camera. There are natural limits on the maximum size of the field of view. Fish-eye lenses with a wide field of view have usually significant radial distortions and do not focus in a single point, which deteriorates the quality of the navigation result that relies on the knowledge of the angle of incidence of the light rays.

Our camera model represents the imaged points on a sphere (see Fig. 11.14). They are represented by the normalized direction vectors n_i. This allows us to construct a reference view used for the localization in the local area that spans the entire space. In the initialization, only features contained in the current sensor view

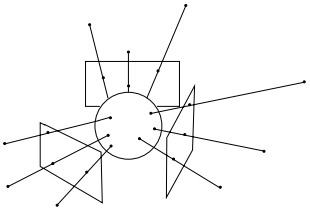


Fig. 11.14 Points in different directions around the origin of the local coordinate frame are getting projected on a sphere and represented as direction vectors n_i.

are used as the initial reference set, but this view gets extended with additional landmarks that are reconstructed using the current motion parameters between images containing a specific landmark. The images used for reconstruction of a specific landmark do not need to contain the initial reference view. The newly estimated landmark position gets transformed back to this initial frame. This allows a reconstruction of landmarks in all directions (360° field of view).

This extension of the field of view permits the usage of one unique set of reference features in a local area independent of the direction of motion. Any noise or error in the localization results in this case just in a noise in the resulting pose estimation. Since the localization is calculated always relative to the same reference structure in the world, we can avoid drifts in the localization that could be caused by continues integration of the relative motions between consecutive frames. The reference frames need to be changed because of the limited range in which a given set can be observed. This hand-off process is an interesting open issue. Interesting solutions can be used from laser based SLAM approaches.

11.4 Results

11.4.1 Convergence of the Pose Estimation

The presented system estimates the pose change between two frames. This can be an incremental change between two consecutive frames or the absolute difference to a reference frame. Depending on the requirements in the system, both modes are of interest. An important question here is the accuracy of the system for varying distances from the original configuration.

The number of iterations required to estimate the motion parameters with an accuracy below 1cm stays below 50 for most large indoor environments tested with this system. The proof of global convergence is mathematically derived in [15]. The number of iterations to find the best transformation explaining the changes between the reference and the current position of the projections varies depending on the initial differences between the reference model and the current pose.

Usually, we don not propagate the changes in the λ_i lengths between the steps. Instead of calculating the change from the reference position, a relative change to the previous step can be calculated which converges in very few ($le10$) iterations.

11.4.2 Reconstruction Results

11.4.2.1 Endoscope Navigation

The system was tested in micro-scale performing pose estimation of the endoscope camera in a phantom of a human skull. The experimental validation of our approach was carried out on the setup depicted in Fig. 11.15.

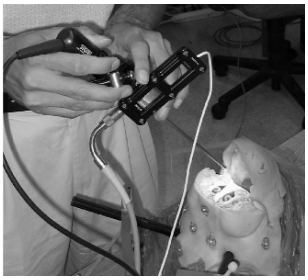


Fig. 11.15 Experimental setup for the validation of the accuracy of the endoscope navigation in a porcine cadaver head.

We tracked the position of the endoscope with the $OptoTrak^{TM}$ system in the background to verify the motion estimation results from our system. The resulting reconstruction errors had a standard deviation of (0.62, 0.3382) for each of the cases. The minimal rotational error expressed as Rodrigues vector was r=(0.0017, 0.0032, 0.0004), (-0.0123, -0.0117, -0.0052) for both cases. The error in the estimate of the translation vector was $\Delta T = (0.05, -0.398, 0.2172)^T, (-0.29, 0.423 - 0.4027)^T [mm]$

11.4.2.2 Outdoor Scene Reconstruction

We used the presented VSLAM system to classify geometric positions of regions in the image to filter candidates for traffic signs. The system was recovering the motion of the camera by tracking of features in the images and performing a 3D reconstruction of the position of the extracted color blobs. Candidates in the right geometric location relative to the road were additionally checked for planarity by adding additional points on the surface in Fig. 11.16.

Fig. 11.16 (Left) Outdoor scene, (right) 3D reconstruction.

11.5 Conclusions

We presented an overview of the navigation approaches tested on our mobile systems. A theoretical background for our approach for explicit recovery of structure and motion from a minimum set of 3 corresponding landmarks in spherical projections was discussed. The presented approach assumes the knowledge about the initial geometrical relation between the depths to the observed points, which may be obtained from a 3D model of the world or from more complex structure-from-motion approaches requiring more points. A good candidate for initialization is, e.g., the eight point algorithm [25] that delivers an initial guess for the depths to the points. This information is refined using the presented 3D-reconstruction. This initial information is updated in the system using a recursive algorithm updating the motion and depth parameters in parallel.

In opposite to other existing approaches, the presented system presents an explicit solution for an arbitrary number of point correspondences in monocular image sequences. We require a minimum of 3 landmarks for the structure and motion recovery, but the system scales easily to more corresponding points, which improve the error compensation capabilities of the system.

Our future work will focus on improvements in the convergence of the system by controlled fixation of parameters depending on the feature configuration and on solving the open challenges mentioned in section 11.3.3.

Acknowledgements The work was partially supported by the DFG Cluster of Excellence in Cognitive Technical Systems (CoTeSys). The author would like to thank Gregory Hager from the Johns Hopkins University, who helped to formulate many of the mathematical foundations for the initial implementation of the presented approaches.

References

1. T. Alter. 3D Pose from 3 Points Using Weak Perspective.. IEEE PAMI, Vol. 16, No. 8, pp. 802-808, 1994
2. K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pat. Anal. Machine Intell.*, vol. 9, pp. 698–700, 1987.

3. J. Borenstein and Y. Koren. Real-time Obstacle Avoidance for Fast Mobile Robots in Cluttered Environments. In *IEEE International Conference on Robotics and Automation*, pages 572 – 577, May 1990.
4. D. Burschka, C. Eberst, C. Robl, and G. Färber. Vision-Based Exploration of Indoor Environments. *Workshop on Robust Vision for Vision-Based Control of Motion at the IEEE International Conference on Robotics and Automation*, WS2, May 1998.
5. D. Burschka and G. Hager. Vision-based control of mobile robots. In *Proc. International Conference on Robotics and Automation*, pages 1707–1713, 2001.
6. D. Burschka and G. Hager. Scene Classification from Dense Disparity Maps in Indoor Environments. In *Proc. ICPR*, 2002.
7. D. Burschka and G. Hager. Stereo-Based Obstacle Avoidance in Indoor Environments with Active Sensor Re-Calibration. In *International Conference on Robotics and Automation*, pages 2066–2072, 2002.
8. D. Burschka and G.D. Hager. Vision-Based 3D Scene Analysis for Driver Assistance. *ICRA*, 2005.
9. D. Burschka and Gregory D. Hager. V-GPS – Image-Based Control for 3D Guidance Systems. In *Proc. of IROS*, pages 1789–1795, October 2003.
10. D. Burschka and Gregory D. Hager. V-GPS(SLAM): – Vision-Based Inertial System for Mobile Robots. In *Proc. of ICRA*, pages 409–415, April 2004.
11. D. F. DeMenthon and Larry S. Davis. Model-Based Object Pose in 25 Lines of Code. *International Journal of Computer Vision*, 15:123–141, June 1995.
12. A. J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. Proc Internation Conference on Computer Vision, Volume 2, pages 1403–1412, 2003.
13. G. Dudek, P. Freedman, and I. Rekleitis. Just-in-time sensing: efficiently combining sonar and laser range data for exploring unknown worlds. *In Proc. of ICRA*, pages 667–672, April 1996.
14. O. Faugeras, *Three-Dimensional Computer Vision*, The MIT Press, 1993.
15. G. Hager, C-P. Lu, and E. Mjolsness. Object pose from video images. *PAMI*, 22(6):610–622, 2000.
16. L.M. Lorigo, R.A. Brooks, and W.E.L. Grimson. Visually-Guided Obstacle Avoidance in Unstructured Environments. *IEEE Conference on Intelligent Robots and Systems*, pages 373–379, September 1997.
17. E. Malis, F. Chaumette, and S. Boudet. 2D 1/2 visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, April 1999.
18. R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2000.
19. R. Haralick, C. Lee, K. Ottenberg and M. Nölle. Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. International Journal on Computer Vision, 13(3), pages 331–356, 1994.
20. B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonomal matrices," *J. Opt. Soc. Amer.*, vol. A-5, pp. 1127–1135, 198.
21. B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternion," *J. Opt. Soc. Amer.*, vol. A-4, pp. 629–642, 1987.
22. D. Nister. A Minimal Solution to the Generalised 3-Point Pose Problem. CVPR 2004, 2004.
23. R. C. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, Springer-Verlag:167–193, 1990.
24. M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, 1991.
25. E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

Chapter 12
Image-Based Visual Servoing with Extra Task Related Constraints in a General Framework for Sensor-Based Robot Systems

Ruben Smits, Duccio Fioravanti, Tinne De Laet, Benedetto Allotta,
Herman Bruyninckx and Joris De Schutter

12.1 Introduction

Robotic tasks of limited complexity, such as simple positioning tasks, trajectory following or pick-and-place applications in well structured environments, are straightforward to program. For these kinds of tasks extensive programming support is available, as the specification primitives for these tasks are present in current commercial robot control software.

While these robot capabilities already fulfill some industrial needs, research focuses on specification and execution of much more complex tasks. The goal of our recent research is to open up new robot applications in industrial as well as domestic and service environments. Examples of complex tasks include sensor-based navigation, like visual servoing and 3D manipulation in partially or completely unknown environments, using redundant robotic systems such as mobile manipulator arms, cooperating robots, robotic hands or humanoid robots, and using multiple sensors such as vision, force, torque, tactile and distance sensors. Little programming support is available for these kinds of tasks.

As a result, the task programmer has to rely on extensive knowledge in multiple fields such as image processing, spatial kinematics, 3D modeling of objects, geometric uncertainty and sensor systems, dynamics and control, estimation, as well as resolution of redundancy and of conflicting constraints.

The goal of our recent research is to fill this gap. We want to develop programming support for the implementation of complex, sensor-based robotic tasks in the presence of geometric uncertainty. The foundation for this programming support is a

Ruben Smits, Tinne De Laet, Herman Bruyninckx, and Joris De Schutter
Katholieke Universiteit Leuven, Department of Mechanical Engineering, Celestijnenlaan 300B, 3001 Heverlee, Belgium, e-mail: ruben.smits@mech.kuleuven.be

Duccio Fioravanti and Benedetto Allotta
Universitaá degli Studi di Firenze, Department of Energetics “Sergio Stecco”,Italy,

generic and systematic approach to specify and control a task while dealing properly with geometric uncertainty [8].

Previous work on specification of sensor-based robot tasks, such as force controlled manipulation [10, 15, 16, 17] or force controlled compliant motion combined with visual servoing [3], was based on the concept of the *compliance frame* [19] or *task frame* [5]. In this frame, different control modes, such as trajectory following, force control, visual servoing or distance control, are assigned to each of the translational directions along the frame axes and to each of the rotational directions about the frame axes. The task frame concept has proven to be very useful for the specification of a variety of practical robot tasks. However, the drawback of the task frame approach is that it only applies to task geometries with limited complexity, that is, task geometries for which separate control modes can be assigned independently to three pure translational and three pure rotational directions along the axes of a *single* frame.

A more general approach is to assign control modes and corresponding constraints to *arbitrary* directions in the six dimensional manipulation space. This approach, known as *constraint-based programming*, opens up new applications involving a much more complex geometry and/or involving multiple sensors that control different directions in space simultaneously.

Seminal theoretical work on constraint-based programming of robot tasks was done by Ambler and Popplestone [2] and by Samson and coworkers [22]: this approach was first applied to vision-based control in [12]. Based on the same theoretical background a more recent work of Mezouar and Chaumette [20] considers the problem of image-based control with bounded field of view and robot joint limits constraints, while in [13] visual servoing in despite of change of visibility in image feature is addressed. Image-based visual servoing with visibility constraint is also discussed in [6]. Also motion planning research in configuration space methods (see [18] for an overview of the literature) specifies the desired relative poses as the result of applying several (possibly conflicting) constraints between object features.

Our own preliminary work on a task specification framework was presented in [9], while the mature framework is thoroughly discussed in [8].

This paper shows the application of this general framework to the example application of image-based visual servoing combined with extra task related constraints in the 3D Cartesian space, this way showing the power and the practical advantages of this systematic approach. Extension to other constraints on extra sensor measurements (like distance), other task spaces or joint space is possible.

The paper is structured as follows. Section 12.2 introduces the formulation of image-based visual servoing with extra task specific constraints in 3D Cartesian space inside the general framework. Section 12.3 defines the additional feature coordinates that are used to model task constraints. Subsequently, Section 12.4 details a velocity-based control scheme which uses these additional coordinates. Experimental results are presented in Section 12.5. Finally, Section 12.6 discusses the proposed approach and summarizes the main conclusions.

12.2 Application

This work presents the application of the general task specification framework to a visual servoing application with extra task constraints in Cartesian space . The goal is to track an object in the image or plan a motion in the image given a planar object and given a camera attached to the robot end effector (eye in hand configuration). To select the object, a number of points are selected in the image before execution. These are the desired positions of the respective image-points. The object is moved around by a human and the robot tracks the object using feedback control on each selected image-point. Robot motion is however constrained in Cartesian space: the robot end effector is not allowed to come too close to a vertical wall. A security border to the wall is defined for the robot end effector. An overview of the experimental setup is shown in Fig. 12.1.

Notice that although the image and Cartesian constraints are defined in different operational spaces they can be easily combined, as shown in the next section.

12.3 Modeling

The general framework introduces additional task related coordinates, denoted as *feature coordinates* χ_f, to facilitate the modeling of both constraints and measurements by the user. These coordinates are defined in *object* frames and *feature* frames that are chosen by the task programmer in a way that simplifies the specification of the task at hand. First the adopted camera projection model is shown, next frames and task coordinates assignments for the constrained visual servoing application are presented.

12.3.1 Camera Model

As for classical IBVS applications, the camera is modeled by its intrinsic matrix K (an upper triangular 3×3 matrix containing focal length, principal point and skew parameter information) and a frame attached to the sensor itself defined as *camera frame* (having its z axis coincident with the focal axis and origin in the projection center) [14]. In this work K is supposed to be known and assumed to be constant (no zoom). Let $P = [X_c\, Y_c\, Z_c]^T$ be the $3D$ coordinates of a generic target point expressed in the camera frame; in the *normalized image* the corresponding *normalized image point* homogeneous coordinates are defined by:

$$\tilde{m} = \begin{bmatrix} m^T & 1 \end{bmatrix}^T = \begin{bmatrix} m_x & m_y & 1 \end{bmatrix}^T = \frac{1}{Z_c} P. \tag{12.1}$$

If pixel coordinates are used, the same point is defined by the following equation:

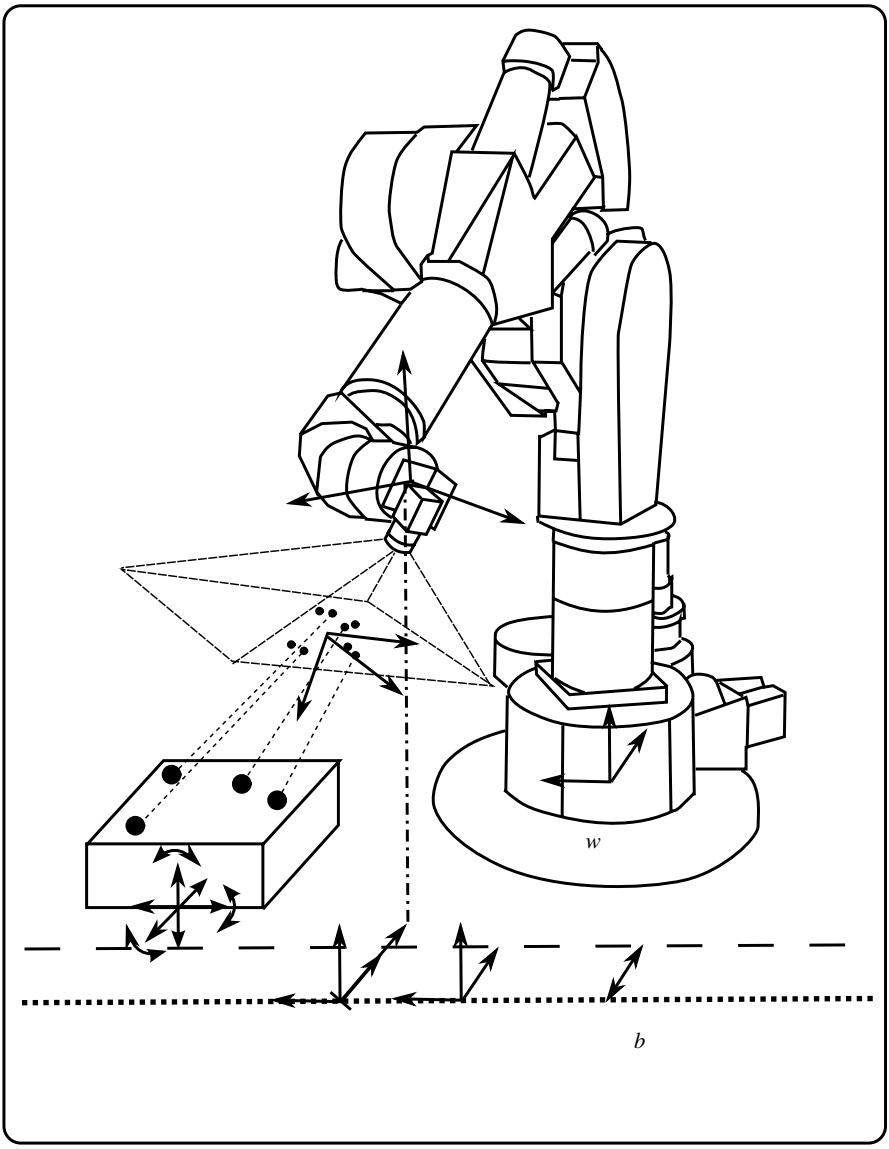


Fig. 12.1 Overview of the experiment: the camera is attached to the robot end effector; target points are projected by the camera in the image plane. The dotted line represents the vertical wall and the dashed line represents the security border.

$$\tilde{p} = \begin{bmatrix} u & v & 1 \end{bmatrix}^T = K\tilde{m} \,. \tag{12.2}$$

Since K is known, it is straightforward to find the normalized coordinates m from the known pixel coordinates $\tilde{p}$ by inverting (12.2) for each image target point of interest.

12.3.2 Object and Feature Frames

A typical robot task accomplishes a relative motion between objects[1]. A systematic procedure is presented to introduce a set of reference frames in which these relative motions are easily expressed.

The first frame is the "world" reference frame, denoted by w. In this application the world frame is placed at the base frame of the robot.

The other frames are *object and feature frames* that are relevant for the task. In the framework an *object* can be any rigid object in the robot system (for example a camera frame, robot end effector or a robot link) or in the robot environment.

In this application the following objects are relevant: the normalized camera-image and its tracked image-points, the robot end effector and the wall.

A *feature*, as defined in the framework, is linked to an *object*, and indicates a *physical entity* on that object (such as an image point, vertex, edge, face, surface), or an *abstract geometric property* of a physical entity (such as the symmetry axis of a hollow cylinder).

In this application relevant features are the tracked image-points, and the camera principal point (in the normalized camera-image) and the distance between the robot end effector and the wall.

The relative motion between two objects is *specified by imposing constraints* on the feasible relative motion between one feature on the first object and a corresponding feature on the second object. Each such constraint needs four frames: *two object frames* (called $o1$ and $o2$, each attached to one of the objects), and *two feature frames* (called $f1$ and $f2$, each attached to one of the corresponding features of the objects).

For an application in 3D space, there are in general six degrees of freedom between $o1$ and $o2$. The connection $o1 \rightarrow f1 \rightarrow f2 \rightarrow o2$ forms a *kinematic chain*, that is, the degrees of freedom between $o1$ and $o2$ are distributed over three sub-motions: the relative motion of $f1$ with respect to $o1$ (sub-motion I), the relative motion of $f2$ with respect to $f1$ (sub-motion II), and the relative motion of $o2$ with respect to $f2$ (sub-motion III), as shown in Figure 12.2.

Furthermore, two different kind of kinematic chains are recognized, one for each of the tracked image-points and one for the constraint on the distance between the robot end effector and the wall (respectively denoted by a and b primes in the following).

[1] In general also controlled dynamic interactions between objects can be taken into account, but the constraint visual servoing example does not involve such interaction.

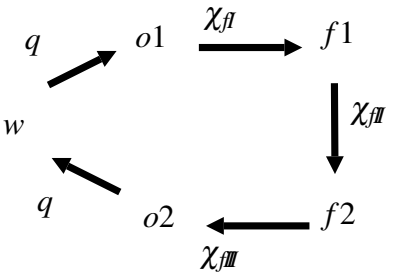


Fig. 12.2 Object and feature frames and feature coordinates.

Figure 12.3 shows the object and feature frames chosen for the constrained visual servoing example. For each kinematic chain of the first kind, corresponding to the tracked image-point i:

- frame $o1^a$ fixed to the principal point in the normalized camera-image with the x- and y-axes along the image axes, and and the z-axis along the focal axis,
- frame $f1^a$ is the same as $o1^a$,
- frames $o2_i^a$ have the same orientation as $o1^a$, but are located at each tracked point in the normalized camera-image,
- frames $f2_i^a$ are the same as the respective $o2_i^a$,

and for the kinematic chain of the second kind:

- frame $o2^b$ fixed to robot end effector,
- frame $o1^b$ fixed to the wall, with its y-axis along the wall,
- frame $f1^b$ is translated from $o1^b$ along the wall, with its origin corresponding to the projection of $f2^b$ onto the y-axis of $o1^b$, and its x-axis perpendicular to the wall.
- frame $f2^b$ is the same as $o2^b$.

The next paragraphs show how the choice of these frames simplifies the mathematical representation of the sub-motions, the system outputs and the measurements.

12.3.3 Feature Coordinates

Task related *feature coordinates* χ_f are introduced to facilitate the task specification by the user. These coordinates represent the sub-motions between $o1$ and $o2$.

For the motion of the tracked image-points in the camera-image the feature coordinates of interest, expressing the sub-motions for each tracked image-point are:

$$\chi_{f\,i}^{\,a} = \chi_{f3\,i}^{\,a} = \begin{pmatrix} m_{xi} & m_{yi} \end{pmatrix}^T, \tag{12.3}$$

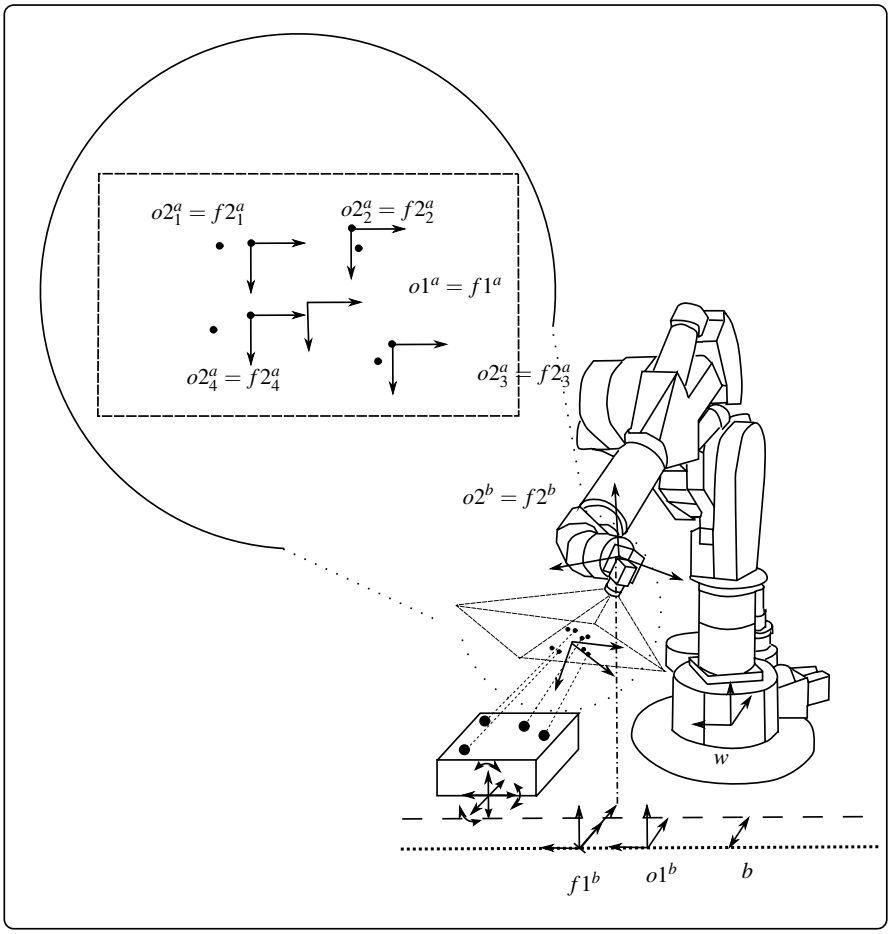


Fig. 12.3 Definition of the frames: $o1^a = f1^a$ is at the principal point of the normalized image, and $o2_i^a = f2_i^a$ are the tracked points in normalized image. $o1^b$ on the base of the wall $o2^b$ is on the robot end effector, $f2^b$ is the same as $o2_b$ and $f1^b$ is translated along the wall, with its origin corresponding to the projection of $f2^b$ onto the y-axis of $o1^b$. The isolated dots represent the desired positions of the respective image-points.

where m_{xi} and m_{yi} are expressed in $f1^a$ and represent the position of the tracked point i, $f2_i^a$ with respect to the principal point of the normalized image. These feature coordinates can be stored in the vector ${\chi_f}^a$:

$$ {\chi_f}^a = \left({\chi_f}_1^{a\,T} \cdots {\chi_f}_k^{a\,T} \right)^T, \tag{12.4} $$

where k represents the number of tracked image-points.

For constraint on the distance between the robot end effector and the wall, the feature coordinates expressing the sub-motions are:

$$\chi_{fI}{}^{b} = y^{b}, \tag{12.5}$$

$$\chi_{fII}{}^{b} = \left(x^{b}\ z^{b}\ \phi^{b}\ \theta^{b}\ \psi^{b} \right)^{T}, \tag{12.6}$$

$$\chi_{f}{}^{b} = \left(\chi_{fI}{}^{b}\ \chi_{fII}{}^{b\,T} \right)^{T}, \tag{12.7}$$

where y^b is the translation along the wall from $o1^b$ to $f1^b$. x^b, z^b are Cartesian coordinates expressed in $f1^b$, representing the x- and z- translation of the robot end effector from the wall to the robot end effector , while ϕ^b, θ^b, ψ^b represent Euler XYZ angles between the wall and the robot end effector.

All feature coordinates are grouped into a single vector χ_f embedding the complete system motion:

$$\chi_f = \left(\chi_f{}^{a\,T}\ \chi_f{}^{b\,T} \right)^{T}. \tag{12.8}$$

12.3.4 Task Specification

To define the desired task, constraints have to be specified on the set of feature coordinates. Constraints are specified as:

$$y_j(t) = y_{dj}(t), \tag{12.9}$$

where $y_j(t)$ represents a system output, and $y_{dj}(t)$ the desired output for constraint j.

To assign the desired position of tracked image-points (or to generate the desired path in the camera-image for an image-point), constraints have to be specified on the following outputs:

$$y_{2i-1} = m_{xi} \text{and} y_{2i} = m_{yi}, \tag{12.10}$$

for $i = 1 \cdots k$, while in Cartesian space, the constraint on the distance from the wall to the robot end effector, has to be specified on:

$$y_{2k+1} = x^{b}. \tag{12.11}$$

12.4 Control

In this section a velocity-based control law is derived for the constrained visual servoing example. The plant is assumed to be an ideal velocity controlled system, that is, the robot dynamics are neglected. Hence the *system equation* is given by:

$$\dot{q} = u = \dot{q}_d, \tag{12.12}$$

where the control input u corresponds to the desired joint velocities $\dot{q}_d$.

On the other hand, the *output equation* relates the system state to the the outputs y:

$$f\left(q, \chi_f\right) = y, \tag{12.13}$$

The system state, consisting of q and χ_f is non-minimal, because a dependency relation exists between q and χ_f. This dependency relation corresponds to the loop closure equations, and is expressed as:

$$l\left(q, \chi_f\right) = 0. \tag{12.14}$$

In the constrained visual servoing example a loop closure is defined for each of the kinematic chains defined in Section 12.3.2.

For the derivation of velocity-based control the output (12.13) and the loop closure equations (12.14) are differentiated with respect to time to obtain equations at velocity level. The output equation at velocity level is written as:

$$C_q \dot{q} + C_f \dot{\chi}_f = \dot{y}, \tag{12.15}$$

with $C_q = \frac{\partial f}{\partial q}$ and $C_f = \frac{\partial f}{\partial \chi_f}$. C_q is used for the constraints in joint space and is zero in our case. C_f is used for the constraints in the different feature spaces.

On the other hand, velocity loop closure for the constraint on the distance to the wall becomes:

$$_wJ_q\, \dot{q} + {}_wJ_f\, \dot{\chi}_f{}^b = 0, \tag{12.16}$$

where ${}_wJ_f = \frac{\partial l^b}{\partial \chi_f{}^b}$ and ${}_wJ_q = \frac{\partial l^b}{\partial q}$, the robot jacobian are expressed in the base frame, w.

For IBVS the velocity loop closure is projected in the reduced normalized image-space using the interaction matrix, J_I:

$$-J_I\, {}_cJ_q \dot{q} + \dot{\chi}_f{}^a = 0\,, \tag{12.17}$$

where ${}_cJ_q$ represents the robot jacobian giving the 6D camera velocity with respect to the base frame, centered in the camera origin and expressed in the camera frame. Because J_I is known to have singular configurations for 3 points, at least 4 points have to be tracked. The points do not have to be on the same object. Different objects could be tracked, the constraints however should be defined in such a way that the task still makes sense. Tracking independently moving objects only makes sense if the constraints are not to keep the image-points at the exact desired position in the image. For this application the image-points are chosen on the same object. Other visual servoing techniques can also be used if a corresponding jacobian is available. How to obtain J_I will be discussed in 12.4.2. Notice that the previous loop closure is only valid if the target object is fixed with respect to the world w; (12.17) neglects indeed image-point motion components given by the relative $3D$ target-world motion. However, if the control frequency is sufficiently high, according to the magnitude of the relative target-world velocity, equation (12.17) can be successfully

used to track a moving target, as shown in the following. To increase the tracking performance an estimator could be used to take into account the neglected image motion component.

From (12.16), (12.17) and (12.8) $\dot{\chi}_f$ can be solved, yielding:

$$\dot{\chi}_f = -{J_f}^{-1} J_q \dot{q}. \tag{12.18}$$

Note that ${J_f}^{-1}$ and J_q are constructed out of the two different velocity loop closures:

$${J_f}^{-1} = \begin{pmatrix} -J_I & 0_{2k\times 6} \\ 0_{6\times 6} & {}_w{J_f}^{-1} \end{pmatrix}, \tag{12.19}$$

$$J_q = \begin{pmatrix} {}_cJ_q \\ {}_wJ_q \end{pmatrix}, \tag{12.20}$$

where J_q is known since it is formed by two different known robot jacobians while J_f can be easily constructed as shown in the next paragraphs. Substituting (12.18) into (12.15) yields the modified output equation:

$$A\dot{q} = \dot{y}, \tag{12.21}$$

where $A = C_q - C_f {J_f}^{-1} J_q$ is introduced for simplicity of notation.

12.4.1 Definition of the Constraints

Constraint equation (12.9) is also expressed at velocity level. As a result, the constraint equation has to include feedback at position level for tracking or to compensate for drift, modeling errors and disturbances:

$$\dot{y} = \dot{y}_d + K_p (y_d - y) = \dot{y}_d^{\circ}, \tag{12.22}$$

with K_p a matrix of feedback constants (in this case a simple diagonal matrix) and $\dot{y}_d^{\circ}$ the modified constraint at velocity level. For the constraints on the tracked image-points (tracking case) $\dot{y}_d$ is set to zero and y_d is assigned equal to the desired position of each image-point. The visual servoing constraints at velocity level finally for the x- and y-position of the tracked image-points become:

$$\dot{y}_j = k_v (y_{dj} - y_j),$$

with k_v the respective diagonal element from K_p for constraint $j = 1 \rightarrow 2k$. In the case of planning for the image-points $\dot{y}_d$ and y_d for each image-point can be set as the result of the planning algorithm.

For the constraint on the distance to the wall:

$$\begin{aligned}\dot{y}_{d\,2k+1} &= 0,\\ y_{d\,2k+1} &= b,\\ \dot{y}_{2k+1} &= k_{cart}(y_{d\,2k+1} - y_{2k+1}),\end{aligned}$$

where b is a positive value representing a security distance to the wall and k_{cart} is the diagonal element from K_p for the constraint on the distance to the wall.

12.4.2 Obtaining the Constraint Matrices

In the constraint visual servoing example the matrices C_q and C_f are easily found by inspection. Since no constraints exist directly on the joint level in this example, $C_q = 0$. Since each constraint is directly expressed on one feature coordinate, C_f becomes a simple matrix selecting the appropriate components of χ_f (12.8):

$$C_f = \begin{pmatrix} I_{2k\times 2k} & 0_{2k\times 6} \\ 0_{1\times 2k} & 0\;1\;0\;0\;0\;0 \end{pmatrix} \tag{12.23}$$

12.4.3 Obtaining the Feature Jacobian

The feature jacobian J_f is composed using the jacobians of the two sub-motions. The jacobian ${}_wJ_f$ expresses the motion between the wall and robot end effector motion and for this application ${}_wJ_f = I_{6\times 6}$. For the camera-image points motion, J_I is obtained from the vision theory. Since we assume to track k image-points, J_I will be a $2k \times 6$ matrix structured as follows:

$$J_I = \begin{bmatrix} J_1^T & \dots & J_i^T & \dots & J_k^T \end{bmatrix}^T \quad , \tag{12.24}$$

where the generic entry J_i ($i = 1 \rightarrow k$) has the form:

$$\begin{bmatrix} -\dfrac{1}{Z_i} & 0 & \dfrac{m_{xi}}{Z_i} & m_{xi}m_{yi} & -(1+m_{xi}^2) & m_{yi} \\ 0 & -\dfrac{1}{Z_i} & \dfrac{m_{yi}}{Z_i} & (1+m_{yi}^2) & -m_{xi}m_{yi} & -m_{xi} \end{bmatrix}. \tag{12.25}$$

J_i is the well known interaction matrix [12] for a normalized image-point m_i, as defined in (12.1) : it expresses the relation between the $2D$ motion of the normalized image-point and the $6D$ camera velocity with respect to the world, expressed in the camera-frame. J_i is both function of the image-point coordinates and of the correspondent Cartesian $3D$ point depth Z_i with respect to the camera frame.

Notice that point depths are unknown: for this reason approximate fixed values or on-line updated estimates [7] must be used for Z_i to build J_I.

12.4.4 Obtaining the Weighting Matrix

In general the system to be solved (12.21) is over-constrained due to tracking k different points and an extra constraint on the distance to the wall. Therefore a weighted pseudo-inverse[11, 21] is used to calculate the desired joint velocities, $\dot{q}_d$:

$$\dot{q}_d = A_W^+ \dot{y}_d^\circ, \tag{12.26}$$

where W is a diagonal matrix which contains the weight of each constraint, $W = diag(w_j^2)$. Following the general framework:

$$w_j = \frac{1}{\Delta_j k_j},$$

where Δ_j denotes the tolerance on constraint j and k_j is the feedback constant from K_p for constraint j. Similarly, in this case, the weight of the constraint on the distance to the wall uses the following function for the weight:

$$g(x,L,l) = max\left(0,\ \frac{l}{|L-x|} - 1\right), \tag{12.27}$$

with L a limit value where the value x is not allowed to pass and l a value describing an activation band. Fig. 12.4 shows the value of $g(x,L,l)$ in function of x for $L = 0$ and $l = 0.05$. If x is out of the band the $g(x,L,l)$ becomes zero, if x enters the band $g(x,L,l)$ grows hyperbolically to infinity as x comes closer to the limit L. The weight of the constraint on the distance to the wall becomes:

$$w_{cart} = \frac{1}{k_{cart}\,\Delta_{cart}}\ g(x^b, 0, b). \tag{12.28}$$

By calculating the weight like this it's made sure that the robot end effector will never come too close to the wall since the weight goes to infinity as the robot end effector approaches the wall. This results in a bigger importance of the constraint on the distance of the wall when solving (12.26).

Because the weight is zero outside the activation band, $x^b > b$, the constraint on the distance to the wall only influences the object tracking if the distance between the robot end effector and the wall enters the security zone defined by b.

For the constraints on the image points also an activation band is used: the points should not leave the image when the Cartesian constraint is activated. It is used on all borders of the normalized image. Let $t_l = [l_x\ l_y]^T$ and $b_r = [L_x\ L_y]^T$ respectively be the top-left and the bottom-right limit corners in the normalized image and l be the security band in the normalize image. For each normalized tracked image-point m_i, as defined in (12.1), weights are calculated as follows:

$$w_{2i-1} = \frac{1}{k_v \Delta_v}\left[1 + g(m_{xi}, L_x, l) + g(m_{xi}, l_x, l)\right],$$

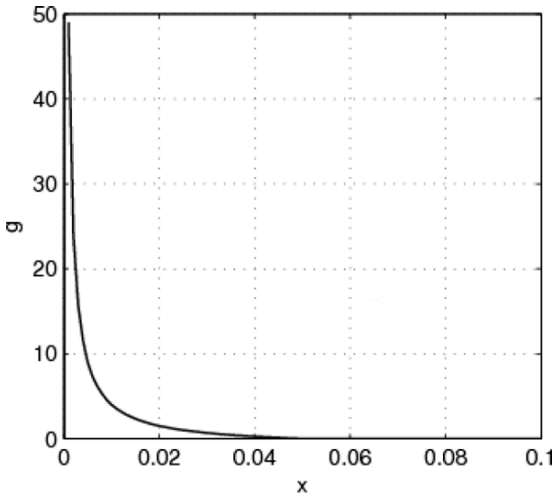


Fig. 12.4 The value of $g(x,L,l)$ with respect to x with $L=0$ and $l=0.05$.

$$w_{2i} = \frac{1}{k_v \Delta_v} \left[1 + g(m_{yi}, L_y, l) + g(m_{yi}, l_y, l)\right].$$

t_l and b_r are easily calculated by inversion of (12.2), applied to the correspondent pixel image corners.

Fig. 12.5 shows the weight w_{xi} of an $m_x i$-position constraint for an image-point in the normalized image in function of $m_x i$ with $k_v = 0.2$, $\Delta_v = 0.005m$ and $l = 0.0857m$, these values are used for the experiments.

12.5 Results

This task is executed on a velocity controlled industrial robot with a fire-wire camera attached to the end effector. The robot is connected to a PC, with a Linux-based operating system and a RTAI realtime execution extension. To control the robot the OROCOS-software[23, 4] is used. The image processing is done using the OpenCV[1] library.

In the experiment color images of pixel-size 640×480 are processed. Feedback constants $k_v = 0.2$ and $k_{cart} = 0.2$ are used. Four image points on the same object are selected for tracking before execution. These points should be stable on the object, this way motion of the points corresponds with motion of the object. The object is moved (by a human) towards the (virtual) wall which is located at $-1m$ of the base frame of the robot. The activation band is chosen $b = 0.05m$. This means that the constraint on the distance to the wall is only activated if the robot end effector x-position in the base frame passes $-0.95m$, which is clearly noticeable in Fig. 12.6.

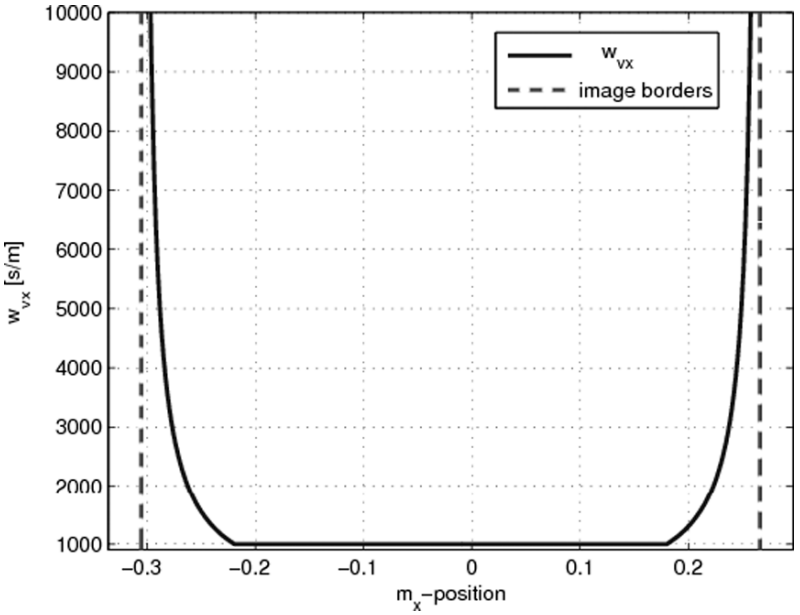


Fig. 12.5 Weight of an x-position constraint in the image in function of the point's x-position.

The figure shows the x-position of the robot end effector expressed in the base frame. As long as the distance of the robot end effector to the wall does not enter the activation band b, the robot is only tracking the selected image-points. If the robot enters the security zone, which is the case between $100s$ and $200s$, the control will try to keep the error for the tracked image-points as small as possible without coming closer to the wall.

Fig. 12.7 shows the positions of the tracked points in the image. Because the feedback constant, k_v is small the image-points do not stay exactly on the desired values, defined by the black crosses in Fig. 12.7, but even when the constraint on the distance of the wall is activated the error on the image points does not enlarge significantly. Note that the image is not centered in $(0,0)$, this is because the principal point is not in the center of the image. Fig 12.8 shows a different experiment with different selected image-points in a very extreme situation where the object has passed the security border very far, the robot does not come any closer but still tries to get the error in the image as small as possible.

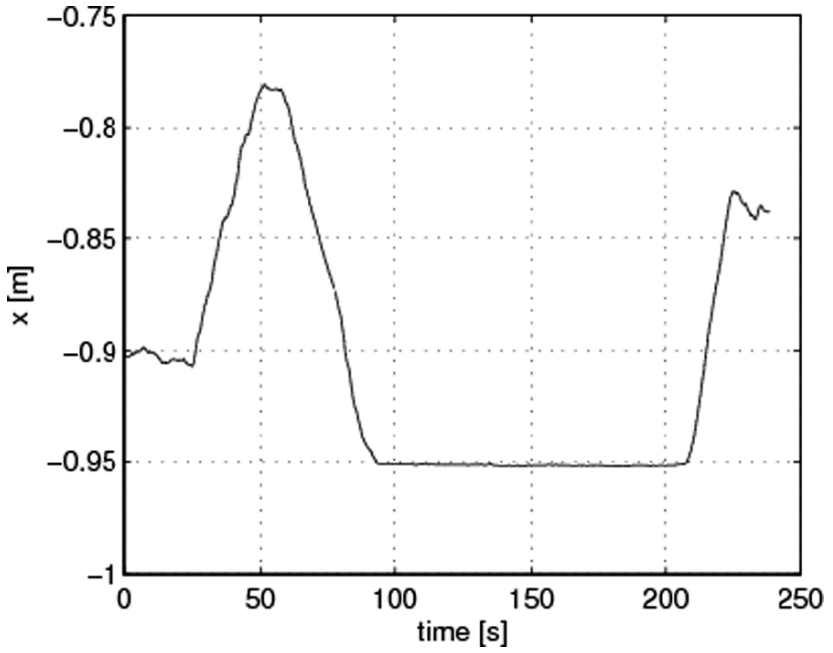


Fig. 12.6 X-position of the robot end effector.

12.6 Conclusion

This paper shows how control in different spaces, image space, Cartesian space, joint space or any other task space can easily be defined in a general task specification framework without adding complexity in the control of the robot. Over-constrained systems can be solved using weights between the different constraints. Inequality constraints are used by defining a hyperbolic function for the weight of the respective constraint. The experimental results show how the constraints in different spaces are correctly combined.

Acknowledgements Tinne De Laet is a Doctoral Fellow of the Fund for Scientific Research–Flanders (F.W.O.) in Belgium.
All authors gratefully acknowledge the financial support by K.U.Leuven's Concerted Research Action GOA/05/10.

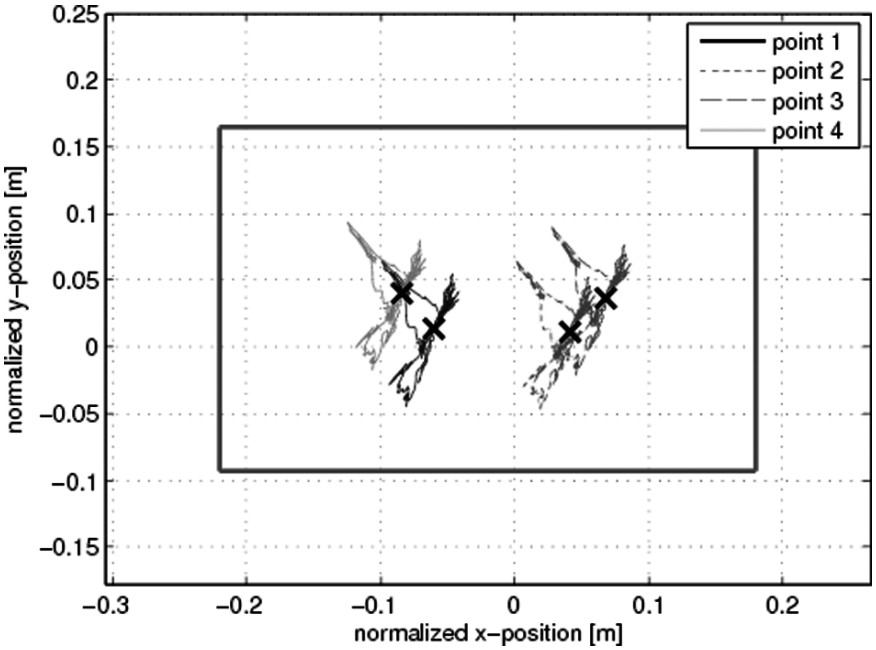


Fig. 12.7 Normalized position of the tracked points in the normalized camera-image, the crosses are the desired positions, the rectangle describes the security-band l for the tracking constraints.

References

1. Open computer vision (opencv). `http://www.intel.com/research/mrl/research/opencv/`
2. Ambler, A.P., Popplestone, R.J.: Inferring the positions of bodies from specified spatial relationships. Artificial Intelligence **6**, 157–174 (1975)
3. Baeten, J., Bruyninckx, H., De Schutter, J.: Integrated vision/force robotics servoing in the task frame formalism. The International Journal of Robotics Research **22**(10), 941–954 (2003)
4. Bruyninckx, H.: Open RObot COntrol Software. `http://www.orocos.org/` (2001)
5. Bruyninckx, H., De Schutter, J.: Specification of force-controlled actions in the "Task Frame Formalism": A survey. IEEE Transactions on Robotics and Automation **12**(5), 581–589 (1996)
6. Chesi, G., Prattichizzo, D., Vicino, A.: Visual servoing: Reaching the desired location following a straight line via polynomial parameterizations. In: Proc. IEEE International Conference on Robotics and Automation (ICRA'05). Barcelona, Spain (2005). On CD-ROM
7. Conticelli, F., Allotta, B.: Discrete-time robot visual feedback in 3D positioning tasks with depth adaptation. IEEE/ASME Transaction on Mechatronics **6**(3), 356–363 (2001)
8. De Schutter, J., De Laet, T., Rutgeerts, J., Decré, W., Smits, R., Aertbeliën, E., Claes, K., Bruyninckx, H.: Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. The International Journal of Robotics Research **26**(5), 433–455 (2007)
9. De Schutter, J., Rutgeerts, J., Aertbelien, E., De Groote, F., De Laet, T., Lefebvre, T., Verdonck, W., Bruyninckx, H.: Unified constraint-based task specification for complex sensor-based robot systems. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 3618–3623. Barcelona, Spain (2005)

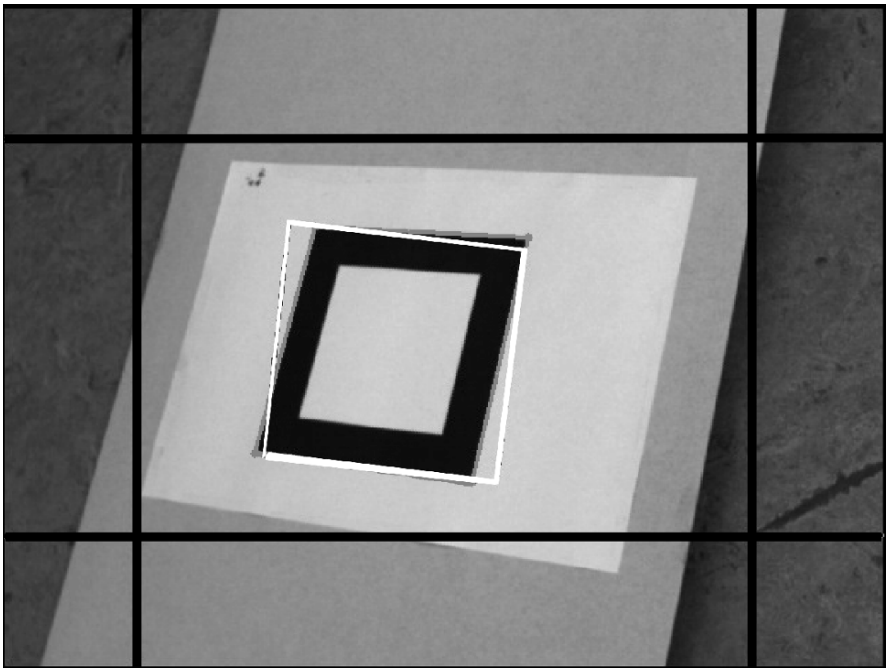


Fig. 12.8 The camera image with the four tracked image-points (the outside corners of the black rectangle), in gray the actual points and object, in white the desired position of the image-points and object and in black the security border.

10. De Schutter, J., Van Brussel, H.: Compliant Motion I, II. The International Journal of Robotics Research **7**(4), 3–33 (1988)
11. Doty, K.L., Melchiorri, C., Bonivento, C.: A theory of generalized inverses applied to robotics. The International Journal of Robotics Research **12**(1), 1–19 (1993)
12. Espiau, B., Chaumette, F., Rives, P.: A new approach to visual servoing in robotics. IEEE Transactions on Robotics and Automation **8**(3), 313–326 (1992)
13. García-Aracil, N., Malis, E., Aracil-Santonja, R., Pérez-Vidal, C.: Continuous visual servoing despite the changes of visibility in image features. IEEE Trans. Robot. **21**(6), 1214–1220 (2005)
14. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2003)
15. Hogan, N.: Impedance control: An approach to manipulation. Parts I-III. Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control **107**, 1–24 (1985)
16. Hogan, N.: Stable execution of contact tasks using impedance control. In: Proceedings of the 1987 IEEE International Conference on Robotics and Automation, pp. 1047–1054. Raleigh, NC (1987)
17. Kazerooni, H.: On the robot compliant motion control. Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control **111**, 416–425 (1989)
18. Latombe, J.C.: Robot motion planning, *Int. Series in Engineering and Computer Science*, vol. 124. Kluwer Academic Publishers, Boston, MA (1991)
19. Mason, M.T.: Compliance and force control for computer controlled manipulators. IEEE Transactions on Systems, Man, and Cybernetics **SMC-11**(6), 418–432 (1981)
20. Mezouar, Y., Chaumette, F.: Path planning for robust image-based control. IEEE Transactions on Robotics and Automation **18**(4), 534–549 (2002)

21. Nakamura, Y.: Advanced robotics: redundancy and optimization. Addison-Wesley, Reading, MA (1991)
22. Samson, C., Le Borgne, M., Espiau, B.: Robot Control, the Task Function Approach. Clarendon Press, Oxford, England (1991)
23. Soetens, P.: A software framework for real-time and distributed robot and machine control. Ph.D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium (2006)

Index