High-Speed Vision and Force Feedback for Motion-Controlled Industrial Manipulators

Olsson, Tomas

2007

Document Version:
Publisher's PDF, also known as Version of record

High-Speed Vision and Force Feedback for Motion-Controlled Industrial Manipulators

High-Speed Vision and Force Feedback for Motion-Controlled Industrial Manipulators

Tomas Olsson

Department of Automatic Control
Lund University
Lund, May 2007

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

ISSN 0280–5316
ISRN LUTFD2/TFRT--1078--SE

Printed in Sweden by Media-Tryck.
Lund 2007

Abstract

Over the last decades, both force sensors and cameras have emerged as useful sensors for different applications in robotics. This thesis considers a number of dynamic visual tracking and control problems, as well as the integration of these techniques with contact force control. Different topics ranging from basic theory to system implementation and applications are treated.

A new interface developed for external sensor control is presented, designed by making non-intrusive extensions to a standard industrial robot control system. The structure of these extensions are presented, the system properties are modeled and experimentally verified, and results from force-controlled stub grinding and deburring experiments are presented. A novel system for force-controlled drilling using a standard industrial robot is also demonstrated. The solution is based on the use of force feedback to control the contact forces and the sliding motions of the pressure foot, which would otherwise occur during the drilling phase.

Basic methods for feature-based tracking and servoing are presented, together with an extension for constrained motion estimation based on a dual quaternion pose parametrization. A method for multi-camera real-time rigid body tracking with time constraints is also presented, based on an optimal selection of the measured features. The developed tracking methods are used as the basis for two different approaches to vision/force control, which are illustrated in experiments.

Intensity-based techniques for tracking and vision-based control are also developed. A dynamic visual tracking technique based directly on the image intensity measurements is presented, together with new stability-based methods suitable for dynamic tracking and feedback problems. The stability-based methods outperform the previous methods in many situations, as shown in simulations and experiments.

Acknowledgments

First, I would like to thank my supervisor Rolf Johansson, who first gave me the opportunity to join the Robotics Group. His initial suggestion for a Master's Thesis regarding force control and camera feedback, became the foundation for the work described in this thesis. His many ideas, his feedback, and exceptional knowledge of many disciplines have been a great source of inspiration and guidance throughout my work. My co-supervisor Anders Robertsson is another source of ideas and inspiration. His willingness to help out with different problems—no matter how busy he is himself–should be greatly acknowledged, together with his expertise in many fields of robotics and control. Many thanks also to Klas Nilsson, Anders Blomdell, Mathias Haage, Isolde Dressler, and all other colleagues in the robot lab!

Many thanks also to all the people at the Department of Automatic Control, who make it such a wonderful and creative environment to work in. Leif Andersson and Anders Blomdell provide great support on all computer-related problems. The assistance from Rolf Braun has also been invaluable, especially his help with construction of all sorts of items that are necessary for robot experiments (sometimes from very loose specifications). The secretaries Agneta, Britt-Marie, and Eva provide great help in all sorts of administrative problems.

Thanks also to my fellow PhD students for many rewarding lunch time discussions, and different types of social events. I also greatly enjoyed my research collaborations with Johan Bengtsson and Dan Henriksson. The comments from Peter Alriksson and Anton Cervin on different parts of the manuscript have been most valuable. My roommate Staffan Haugwitz deserves a mention for his positive spirit. This has made it a pleasure to share office with him all these years, despite his nasty habit of always making me answer the phone for him. My *current* (sorry Dan!) teammates in PiHHP are acknowledged for their impressive physical, technical, and tactical skills, which made it possible for me to finally win the coveted

T-shirt. I could not have won it without you! Actually, on most days I did not really believe that I could win it *with* you, either.

The financial support from the EU 5th Framework Growth Project GRDI-2000-25135 *Affordable, flexible system for off-line automated fettling and finishing*, AUTOFETT, and the ProViking/SSF project *Flexible and Accurate Automation Using Robot Systems*, FlexAA, is gratefully acknowledged. Many thanks also to our colleagues at ABB Robotics, and to all the other competent and hard-working people that I have had the pleasure of working with in the above mentioned projects!

Finally I would like to thank my friends and family for their encouragement and support.

Tomas

Contents

1

Introduction

1.1 Motivation

Traditionally, industrial robot systems have been designed to achieve a high level of performance in free-motion tasks, in which the environment contacts are either non-existent or sufficiently soft to be neglected. Examples of tasks that fit this description are welding, gluing and sealing, as well as many operations that involve tooling with built-in passive compliance, such as Remote Center Compliance (RCC) devices in assembly. In such tasks, the control objective can be formulated as a problem of motion control, in which the goal is to track the desired path, as specified in the user program, as quickly and accurately as possible. Although feedback is necessary for achieving stability, and to compensate for modeling errors and disturbances, a large part of the performance is gained by other means than feedback. Examples in modern control systems include optimal trajectory and feedforward generation, and improved absolute calibration for better static positioning accuracy. In contrast, the feedback is customarily solved using simple and easily tuned controllers and encoder/resolver measurements of the motor positions. The performance and disturbance sensitivity of this type of motion control is sufficient for a large class of applications.

As examples of tasks which do not fit the above description, there exist numerous industrial tasks that require physical work to be performed on an object or component. In some cases, part of the environment in which the robot operates is unknown or non-stationary, and the interaction problem becomes more complex. Unknown or dynamic environments occur frequently in mobile, home and field robotics. In industrial robotics, uncertain and dynamic/non-stationary systems occur in operations where accurate fixturing can not be used, for operations on objects with (par-

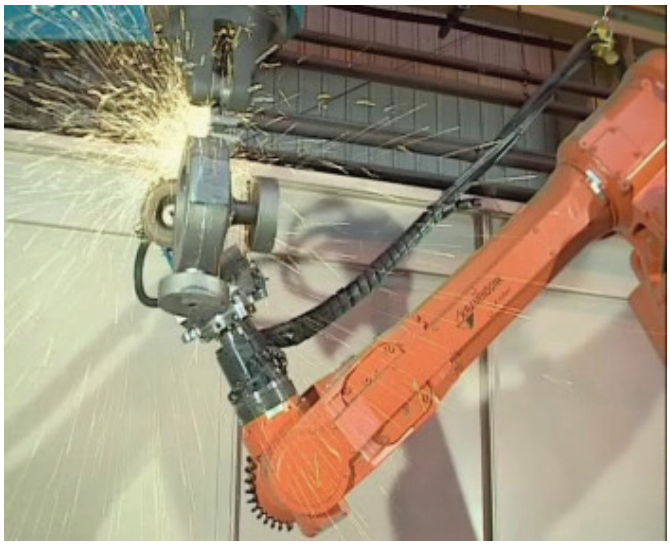

Figure 1.1 Stub grinding of castings using a force-controlled industrial robot system. The force control is necessary for maintaining a specified force of interaction between the robot and the casting, despite large uncertainties and variations in the geometry of the workpiece.

tially) unknown geometry, and for systems of uncalibrated cooperating robots. In general, these cases are more demanding from a control perspective, and may require high-bandwidth feedback. Depending on the task, different properties of the interaction may need to be controlled, such as the contact force magnitude or direction. For example, in grinding, the normal force needs to be accurately controlled, or otherwise too much (or too little) material is removed by the grinding tool. This is illustrated by Fig 1.1, in which a proper contact force should be maintained while the rotating grinding tool sweeps across the surface. In assembly and parts mating applications contact forces should also be controlled, as uncontrolled stiff contact may cause breakage of delicate parts.

When machines, such as industrial robots, are employed in order to automate such tasks, the physical contact gives rise to contact forces which act on both the robot and the manipulated workpiece. Thereby, a new feedback loop is created, where the robot motion depends on the contact force and vice versa. The nature of this feedback loop is determined by the dynamical properties of both the robot and its environment. The dynamics and feedback are also key issues in the similar problem of force-reflecting teleoperation [Hannaford and Anderson, 1988], where the contact dynamics and time delays represent some of the critical limiting factors for the

performance in practical implementations. For soft contact tasks, the low-gain environment feedback loop does not critically affect the stability or performance, and the problem of interaction control becomes dynamically similar to the motion control problem as described above. When the contact is stiff, however, the entire dynamics will change significantly. In general, the dynamic response of the robot to disturbances becomes more important in interaction control than in the motion control problem. While in simple cases the internal motion control may be reprogrammed to provide a softer and more compliant response in contact, this may conflict with the demands for performance and positioning accuracy.

The most flexible solution to the interaction control problem involves the use of external sensors and control loops. A primary example is the use of wrist-mounted force/torque sensors, which make it possible to measure and to directly control the contact forces. For reasons that will be described in more detail later, the external control loops in industrial robots are usually closed around the built-in velocity/position loops of the motion control. One reason is that most tasks include specifications not only on the force, but also on the motion trajectories of the robot. This has led to the development of several different concepts for combined control of force and motion, such as *impedance control* [Hogan, 1985] and *hybrid position/force control* [Raibert and Craig, 1981]. In addition, the properties of the inner-loop motion control will naturally also affect the stability and performance of the outer (external) control loops. Intuitively, the motion commands from the outer loop should ideally be followed as accurately as possible by the inner servo loop, even in the presence of disturbances. If the inner loop is unable to provide the desired performance, strong dynamic couplings will affect the outer loop, even to the point of causing instability. This illustrates the importance of considering also the motion control problem, as well as its effect on the interaction control.

This thesis considers vision-based position/force control, in which the motion control is complemented with camera feedback. Over the last two decades, cameras have emerged as useful sensors for measuring the positions of objects. In robotics, the use of cameras has become widespread, in the research community as well as in industry. Many commercial packages for vision-based calibration, positioning and inspection have been developed, and vision sensors have become crucial components for mobile and autonomous robot systems. Camera feedback makes it possible to handle unknown and dynamic environments, and to improve the position control and disturbance rejection of a robot. This makes the combination of force and vision an attractive option for accurate control of contact tasks.

Problem Illustration

In order to better illustrate the somewhat abstract discussion in the previous section, a series of short examples are presented. In these examples it is attempted to illustrate the dynamics, concepts and problems associated with interaction control, as well as the role of motion control in the force control problem. The examples contain highly idealized models in a single degree of freedom, and ignore important effects and modeling assumptions such as dynamically coupled, multi-dimensional robot dynamics, nonlinearities such as friction and backlash, sensor/actuator and environment dynamics, as well as issues concerning robustness.

Position Control. The simplest possible robot model is given by the linear or angular motion of a single unit mass/inertia with position or angle x, actuated by applying forces or torques, where the total torque is denoted τ_{tot}. The equations of motion are given in the frequency domain as the double integrator

$$x(s) = \frac{1}{s^2}\tau_{tot}(s) \tag{1.1}$$

Although the model (1.1) is very simple, rigid robot models have many dynamic properties, such as energy conservation, in common with the single-mass model. Furthermore, there exist feedback linearization techniques that transform an ideal n-degree-of-freedom (n-DoF) rigid robot model to a set of n decoupled double integrators [Khatib, 1987; Spong, 1989]. In the model (1.1), the input force contains both the control torque and any forces applied by external sources, for example through interaction with the environment. Hence, we decompose the total torque acting on the robot according to

$$\tau_{tot} = \tau + \tau_e \tag{1.2}$$

into the control torque τ and the external/environment torque τ_e. The control torque is computed from the current position x by a position control law with reference x_r, which in this case is assumed to be a PID-type[1] controller given by

$$\tau(s) = K_p\left(K_v + K_i\frac{1}{s}\right)\frac{1}{sT_{\text{filt}}+1}x_r(s) - \left(K_pK_v + K_i + K_pK_i\frac{1}{s} + K_vs\right)x(s) \tag{1.3}$$

[1]The controller is often implemented as a cascaded structure of velocity/position control. For instance, the PID-part of the controller (1.3) could be implemented as a combination of an inner velocity PI-controller with parameters K_v and K_i with an outer position P-controller with gain K_p. The advantages of such a structure include simplified controller tuning, as well as the potential for straightforward modification to include velocity/torque feedforward signals. This feedforward will be described in Chapter 3, where it is used in the implementation of the force controllers.

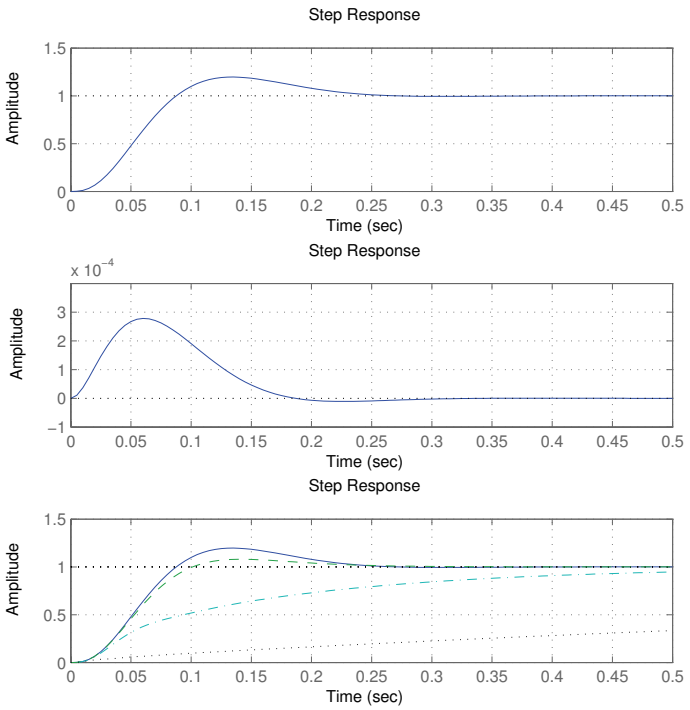

Figure 1.2 Step responses for the controlled rigid robot to position references (*top plot*), force step disturbances (*middle*), and responses to position references (*bottom plot*) when in contact with environments of stiffness of 0 N/m (*solid*), 500 N/m (*dashed*), 5000 N/m (*dash-dotted*) and 50000 N/m (*dotted*).

where the controller parameters have been tuned to create a fast and well-damped suppression of external disturbances. The feedforward low-pass filter with parameter T_{filt}, is used in this example to represent the highly advanced model-based feedforward generation algorithms of modern industrial robots, can be used to obtain a suitable position reference response. This response is shown in the the two upper plots of Fig. 1.2. This response represents a typical response of an (unconstrained) robot to raw reference changes, i.e. low-level references that have not passed through the high-level trajectory generation system. This type of fast reference changes are the main method of actuation for the robot control systems considered in this thesis.

When the robot comes into contact with an environment, the dynamics

of the problem changes, due to the new feedback loop from the robot motion to the external force. In the case when the environment can be modeled as a simple stiffness or linear spring, this contact feedback is characterized by

$$\tau_e = -kx. \tag{1.4}$$

In practice, the system may exhibit a switched behavior with jumps between contact and non-contact. If proper care is not taken in the design of the mechanical structure and the controller, undesired limit cycles and "bouncing" may occur in the contact transition zone, although such effects will be neglected in this example. The bottom plot in Fig. 1.2 illustrates how the behavior of the position control changes with the environment stiffness. We see that the step responses become slower, while stability is maintained. It can be shown, however, that there exist *passive* linear environments for which the system would become unstable. In order to avoid this, one must make sure that the robot end-point *impedance* relation from external torque to end-point velocity is passive, in which case the manipulator will be stable during contact with every possible linear passive environment [Colgate and Hogan, 1989]. This is the fundamental concept behind the technique of impedance control [Hogan, 1985], described in more detail in Chapter 2. In the above example, the position control using the PID-controller makes the end-point impedance non-passive, and thus there exist passive environments that would destabilize the system.

In cases when τ_e is measurable, for instance from a wrist-mounted force/torque sensor, complete cancellation of the disturbance τ_e can be achieved through feedforward. This eliminates the environment feedback loop, making the problem dynamically similar to a position control problem. While many force control methods rely on such feedforward, perfect cancellation of τ_e is possible only in case of perfect rigidity of the robot. Unfortunately, the rigid model is a rather poor approximation of a standard serial manipulator. In practice, flexibilities in the transmission and links will make the robot compliant, which complicates the control problem significantly. A flexible-joint robot can be modeled by two masses m_a and m_m connected through a spring-damper of stiffness k_r and damping d_r, as shown in Fig. 1.3. We use x_m and x_a to denote the positions of the two masses, representing the motor and arm positions, respectively. Note that the external torque τ_e and the control torque τ are no longer acting at the same point, i.e. the arm-side responses to τ_e and τ are no longer of the same form. The disturbance and the actuation are therefore said to be *dynamically non-collocated*. The ratio of the masses m_m and m_a is an important factor determining the achievable control performance of the system. For a high ratio m_a/m_m it may be difficult to achieve a satisfactory closed-loop bandwidth, while for low m_a/m_m it becomes very difficult

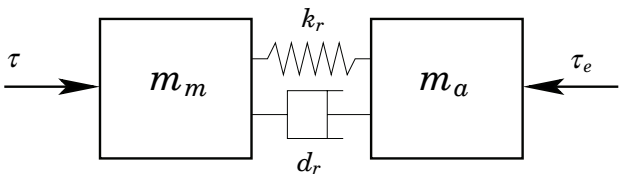

Figure 1.3 Two-mass model of a compliant robot, as used in the introductory example.

to control x_a from measurements of x_m only [Nordin and Gutman, 2002].

We modify our simple rigid robot model to include joint flexibility by setting $m_a = m_m = 0.5$ and adding a compliance, creating a resonance at 10 Hz with a relative damping of 0.3. The same PID controller as in the rigid case is used, with feedback from the motor side position x_m. In Fig. 1.4 we can see how the step responses change compared to the rigid case of Fig. 1.2. Due to the compliance and the motor-side feedback, external forces τ_e will result in stationary positioning errors. In particular, the system behavior changes when in contact with the environment, as seen in the bottom plot in Fig. 1.4. Although a more sophisticated controller, which would take the flexibility properly into account, could be designed in order to improve the trajectory tracking during free motion, it would be very difficult to achieve good rejection of the force disturbances without some feedback from measurements taken on the arm side. Two alternatives of such measurements would be to measure the contact force, and to use feedforward to suppress the disturbances, or to measure and control the position x_a directly, using for instance a camera. However, it is worth noting that the compliance of the robot also have favorable effects on the system properties. For example, due to the dominance of the mass-spring-damper dynamics on the end-point dynamics from τ_e to x_a, the system impedance is now in fact passive, and the manipulator is therefore guaranteed to be stable during contact with every linear passive environment [Colgate and Hogan, 1989]. When arm-side sensors are used to control the robot, particular care must be taken to ensure that the system remains stable also during contact. In Fig. 1.5 are shown two responses to step force disturbances for the flexible robot, one using the PID controller from the previous case, and the other for manually tuned PID controller using arm-side position feedback. Although the arm side control improves the asymptotic force rejection, the passivity property is lost. A practical consequence of the loss of passivity is that the system would become unstable when connected at its end-point to, for instance, an environment mass-spring-damper system with mass $m = 10$ kg, a resonance frequency of 1 Hz and relative damping of 0.2.

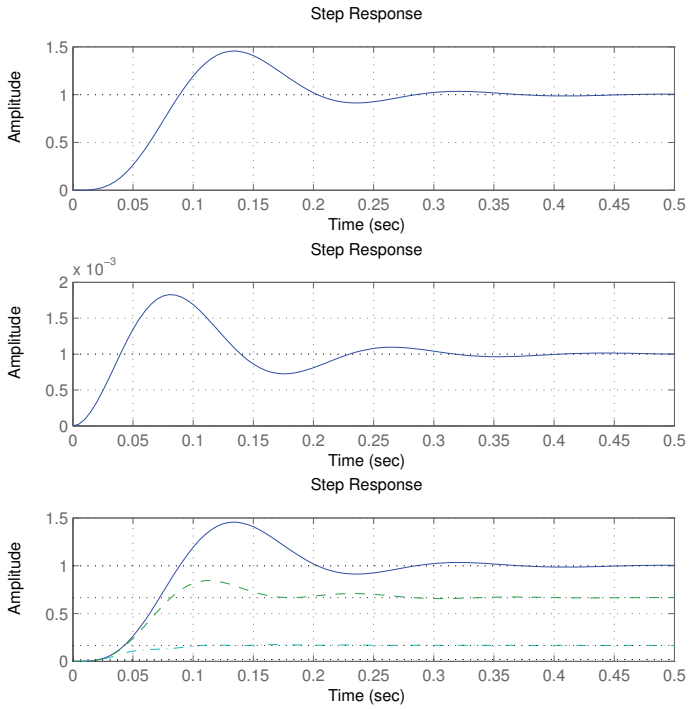

Figure 1.4 Step responses for the arm side position of the controlled flexible robot to position references (*top figure*), force step disturbances (*middle*) and responses to position references (*bottom*) when in contact with environments of stiffness of 0 N/m (*solid*), 500 N/m (*dashed*), 5000 N/m (*dash-dotted*) and 50000 N/m (*dotted*).

Force Control. The mechanical compliance of the robot may be insufficient for a given task in which contact is required, and small trajectory errors may result in damage to the workpiece and manipulator. Rather than artificially adding to the compliance, e.g. by using a passively compliant end-effector as in [De Schutter *et al.*, 1996; Robertsson *et al.*, 2006], force feedback provides a flexible way to design and control the interaction properties through software. Often, a force controller used for maintaining a desired contact force in some directions is combined with position control in other degrees of freedom. A structure suitable for implementation on industrial robots is to use an inner motion controller in the loop [De Schutter and Van Brussel, 1988b; Freund and Pesara, 1998; Siciliano and Villani, 1999], as seen in Fig. 1.6. This structure achieves some de-

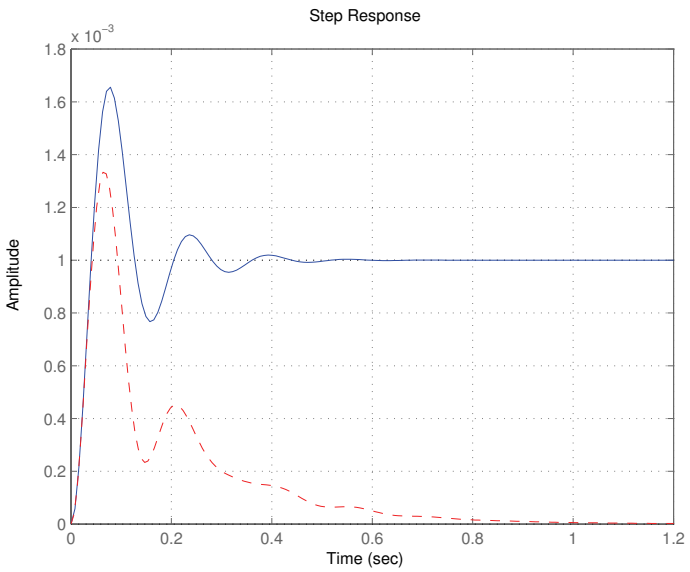

Figure 1.5 Arm-side responses of the controlled flexible robot to force step disturbances, with standard motor-side feedback controller (*solid*) and arm-side PID control (*dashed*).

gree of separation between the position- and force control, while being more robust than direct force control. This inner motion controller is used to track the desired motion commanded by an outer-loop force controller. In our case, we can achieve this by again using the PID controller from Eq. (1.3), where the reference x_r is obtained from the outer impedance controller

$$(m_i s^2 + d_i s + k_i)(x_r(s) - x_{des}(s)) = \tau_e(s), \tag{1.5}$$

where $x_{des}(s)$ is the nominal motion trajectory[2]. The resulting linear system can be represented on transfer function form as

$$x_a(s) = G_{c,p}(s)x_{des}(s) + (G_{c,p}(s)G_i(s) + G_{c,f}(s))\,\tau_e(s) \tag{1.6}$$

where $G_{c,p}(s)$ and $G_{c,f}(s)$ are the closed-loop transfer functions of the position-controlled system from x_r and τ_e to x_a, and

$$G_i(s) = \frac{1}{m_i s^2 + d_i s + k_i} \tag{1.7}$$

[2]If instead direct force control is desired, Eq. (1.5) can be modified with a force reference and integral action, as will be done in Chapter 3.

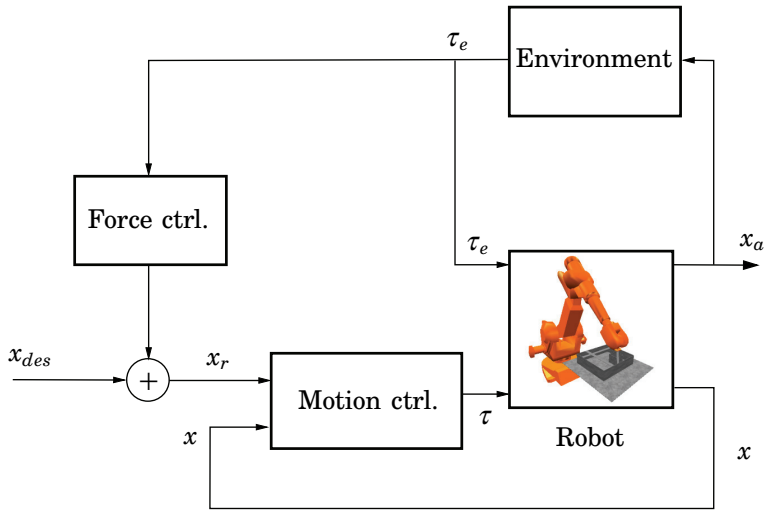

Figure 1.6 Illustration of typical structure used for force control with inner motion control.

is the transfer function representation of the desired impedance relation. As $G_{c,p}(j\omega) \approx 1$ and $G_{c,p}(j\omega)G_i(j\omega) \gg G_{c,f}(j\omega)$ for small frequencies ω, Eq. (1.6) will approximate the desired passive impedance relation $G_i(s)$ up to frequencies roughly corresponding to the bandwidth of the robot motion control. For high frequencies it holds approximately that $G_{c,p}(j\omega) \approx 0$ and $G_{c,p}(j\omega)G_i(j\omega) \ll G_{c,f}(j\omega)$, and the system impedance will approximate the impedance of the position-controlled robot. For a flexible robot, this will in turn approximate the (passive) mechanical compliance of the gear and arm dynamics, as described above. Therefore, the behavior of the transfer function $G_{c,p}(s)G_i(s) + G_{c,f}(s)$ does in general approximate the behavior of passive systems at high and low frequencies. However, for frequencies around the bandwidth of the system, the behavior is strongly influenced by the position controller response $G_{c,p}(s)$, and the overall system will in general not be passive. In effect, this limits the basic inner-motion impedance control techniques to low and medium bandwidth implementations. In order to increase the bandwidth, it becomes necessary to take all the dynamics of the robot, controller, and environment into account in the design. The requirements on modeling of the full system dynamics is a critical issue in practice, since detailed models of the environment or even the controlled robot may be very difficult to obtain. Returning to our example, Fig. 1.7 shows the relevant Bode diagram in the flexible-joint case, where the desired impedance was defined as $m_i = 1$, $d_i = 10$,

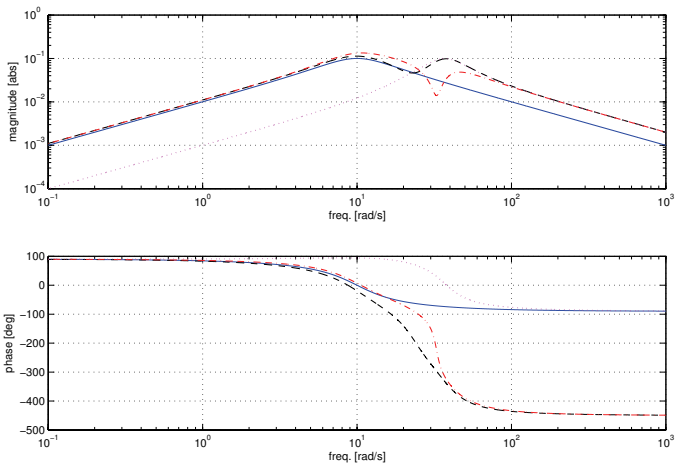

Figure 1.7 Bode diagram of a compliant robot for the transfer functions from τ_e to x_a. *Solid:* ideal (passive) impedance. *Dashed:* obtained closed-loop impedance using impedance control with inner position control. *Dash-dotted:* obtained closed-loop impedance using impedance control with extra phase-lead added in the impedance controller. *Dotted:* the obtained impedance under pure motion control as in Section 1.1.

$k_i = 100$. For the impedance controller, some additional phase advance was added by manual tuning, in order to partially compensate for the effects of the robot dynamics. As can be seen, the true impedance approximates the specified impedance only for frequencies up to 20 rad/s, roughly the bandwidth of the robot motion control system.

Discussion. Despite the drawbacks concerning the difficulty of system modeling and other limitations of the inner motion control approach, from a practical point of view there are also a number of significant advantages [Freund and Pesara, 1998]:

- The majority of industrial robot controllers do only provide an interface to the position control, through position/velocity references. Therefore, direct access to the driving torques is not possible.
- The availability of fast position controllers in robot systems significantly simplifies and speeds up the development of simple force controllers. In addition, for more advanced control the development of the position and force controllers can often be separated, which simplifies the control design problem.

- In most tasks, not all directions of the workspace are constrained. In addition to the force control, position control is required in some degrees of freedom. For a position-controlled manipulator, the combination of force/position control is conceptually straightforward.
- If fast joint-level position feedback loops are available, the force controller may be implemented at a significantly lower sampling rate than what would otherwise be necessary. This is an important advantage, since the external sensor and control interfaces—and the computationally expensive force control loops—would otherwise limit the achievable sampling rates and introduce undesired dead-time. Perhaps most important of all advantages is that inner high-gain position/velocity feedback loops can be used, which is necessary, e.g., in order to achieve good rejection of disturbances and effects such as Coulomb friction.

1.2 Outline, Contributions, and Related Publications

This section contains a brief outline of each chapter in the rest of the thesis, with a description of the contents, contributions, and references to related publications.

Chapter 2: Background

This chapter gives a short introduction to a number of subjects related to industrial robotics and control. Particular emphasis is put on topics related to visual sensing, visual estimation and visual servoing, force and interaction control, and combined force/vision control. Brief presentations of previous research in the relevant areas are given, together with discussions of alternative approaches.

Chapter 3: Experimental Industrial Robot System

This chapter presents the new interface developed for external sensor control, designed by making non-intrusive extensions to a standard industrial robot control system. The structure of these extensions are presented, and the dynamics of the system with its new interface are modeled and experimentally verified. Results from experiments, using the designed interface for force-controlled grinding and deburring, are presented.

Publications.

Blomdell, A., G. Bolmsjö, T. Brogårdh, P. Cederberg, M. Isaksson, R. Johansson, M. Haage, K. Nilsson, M. Olsson, T. Olsson, A. Roberts-

son, and J. Wang (2005): "Extending an industrial robot controller–Implementation and applications of a fast open sensor interface." *IEEE Robotics and Automation Magazine*, **12:3**, pp. 85–94.

Johansson, R., A. Robertsson, K. Nilsson, T. Brogårdh, P. Cederberg, M. Olsson, T. Olsson, and G. Bolmsjö (2004): "Sensor integration in task-level programming and industrial robotic task execution control." *Industrial Robot: An International Journal*, **31:3**, pp. 284–296.

Robertsson, A., T. Olsson, B. Lauwers, K. Nilsson, T. Brogårdh, A. Blomdell, H. De Baerdemaeker, M. Haage, R. Johansson, and H. Brantmark (2006): "Implementation of industrial robot force control—Case study: High power stub grinding and deburring." In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2743–2748. Beijing, China.

The interface described in this chapter is the result of the efforts of several people. Anders Robertsson and Klas Nilsson were responsible for the design and implementation of the interface, supported by Torgny Brogårdh and Mats Isaksson at ABB, and many others. Anders Blomdell, among many other contributions, developed the software for the Linux PowerPC platform on which the external controllers are run. Mathias Haage and Klas Nilsson designed and implemented the force control extensions to the robot programming language RAPID. J.J. Wang provided most of the code for the kinematics library, as well as other software. The dynamic models were developed by Tomas Olsson, who also designed and implemented the force controllers used in the experiments. Tomas Olsson, Klas Nilsson and Anders Robertsson carried out the grinding/deburring experiments, with contributions from Mathias Haage and Hans de Baerdemaeker.

Chapter 4: Feature-Based Visual Tracking and Force/Vision Control

In this chapter, the basic methods for visual- and force control, tracking and servoing are presented, together with some improvements of the tracking aimed at better robustness and performance. We present methods for real-time rigid body tracking with simultaneous calibration and tracking of intrinsic parameters, based on a dual quaternion parametrization of the object pose. It is shown that for a setup with a wrist-mounted camera, referred to as an eye-in-hand system, the rigid connection between the camera and the robot wrist can be expressed as two linear constraints. These constraints can be used to reduce the number of estimated motion parameters by two, resulting in improved robustness.

A method for multi-camera real-time rigid body tracking with time constraints is also presented in this chapter. The proposed method exploits the trade-off in a tracking algorithm between computing time and

the accuracy of the produced position/orientation estimates. An equation for the covariance of the estimation error is calculated, and an efficient algorithm for selection of a suitable subset of the available cameras is presented. This leads to a convex optimization problem for the distribution of computational resources over a given subset of cameras. The suggested strategy is compared to heuristic algorithms, and evaluated in simulations capturing the real-time properties of the tracking algorithm, and the effects of the timing on the performance of vision-based control systems.

Based on the feature-based motion estimation algorithms, methods and experiments using a position-based hybrid force/vision control algorithm are presented. The visual tracker is used to estimate the states of a linear system, and impedance control with inner motion control is used to obtain compliant behavior in the force-controlled directions. An image-based visual servoing technique is used together with force feedback to perform experiments with drawing on a planar surface, while the position of the surface is simultaneously estimated using the available sensor data.

Publications.

Olsson, T., J. Bengtsson, R. Johansson, and H. Malm (2002): "Force control and visual servoing using planar surface identification." In *IEEE Int. Conf. on Robotics and Automation*, pp. 4211–4216. Washington D.C., USA.

Olsson, T., R. Johansson, and A. Robertsson (2004): "Flexible force-vision control for surface following using multiple cameras." In *Proc. of 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 798–803. Sendai, Japan.

Olsson, T., J. Bengtsson, A. Robertsson, and R. Johansson (2003): "Visual position tracking using dual quaternions with hand-eye motion constraints." In *IEEE Int. Conf. on Robotics and Automation*, pp. 3491–3496. Taipei, Taiwan.

Henriksson, D. and T. Olsson (2004): "Maximizing the use of computational resources in multi-camera feedback control." In *10th IEEE Real-Time and Embedded Technology and Applications Symposium RTAS04*, pp. 360–367. Toronto, Canada.

The modification of the feature-based tracker for use with dual quaternions was originally developed together with Johan Bengtsson in [Olsson *et al.*, 2003]. The section on multi-camera tracking and resource constraints represents joint work with Dan Henriksson [Henriksson and Olsson, 2004]. Tomas Olsson developed the tracking algorithm and provided the tools used for simulation of the vision system. Dan Henriksson connected the vision simulation to the TrueTime tool [Henriksson *et al.*,

2002], in order to perform the real-time simulations. The resource allocation algorithm was developed in collaboration between the two authors.

Chapter 5: Intensity-Based High-Speed Tracking and Control

In this chapter, intensity-based approaches for tracking and vision based position control are presented. A dynamic visual tracking technique based directly on the image intensity measurements is used to obtain state estimates at a very high rate, and with very short input-output latency. Methods which take the stability of the resulting estimator explicitly into account are developed, suitable for dynamic tracking and feedback. By relaxing the traditional least-squares optimal solutions, significant performance improvements are demonstrated in several problems. It is shown how the linearization procedure can be modified to take suppression of measurement noise and illumination disturbances into account. The practical importance of the developments are illustrated in simulations and experiments. Experiments with 250 Hz image-based visual servoing are presented, as well as hybrid methods based on fusion of intensity/feature measurements. Methods for control of a system, consisting of a manipulator interacting with a poorly damped oscillatory environment using vision and force feedback, are also presented.

Publications.

Olsson, T., R. Johansson, and A. Robertsson (2006): "High-speed visual robot control using an optimal linearizing intensity-based filtering approach." In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1212–1217. Beijing, China.

Olsson, T., R. Johansson, and A. Robertsson (2005): "Force/vision based active damping control of contact transition in dynamic environments." In Vidal *et al.*, Eds., *ICCV 2005 Workshop on Dynamical Vision*, vol. 4358 of *Lecture Notes in Computer Science*. Springer.

Chapter 6: A Study on Force Control for Accurate Low-Cost Robot Drilling

This chapter presents a novel system for force-controlled drilling using a standard industrial robot. The solution presented is based on applying a constant pressure against the drilled surface with a pressure foot attached to the drilling tool, and to use force feedback to detect and control the sliding motions, which would otherwise occur during the drilling phase. Instead of controlling the position directly from arm-side position measurements, the controller attempts to achieve this task by making sure that the tangential interaction forces are always small enough to

keep the contact in the stiction regime. The friction contact will damp and suppress small disturbances, such as vibrations from the drill feed and spindle. The critical factor in the drilling system is to keep the tool stationary, so that no sliding occurs between the drilling tool and the surface. This application is different from most other applications of force control, where the force control is used to increase the compliance rather than to improve the stiffness to force disturbances, as is the case in this work. The force control and active sliding suppression takes care of large disturbances at lower frequencies, such as the slower variations of the cutting forces. Thereby, a system which is able to reject disturbances over a wide frequency range is obtained, at a very low cost.

Publications.

Olsson, T., A. Robertsson, and R. Johansson (2007): "Flexible force control for accurate low-cost robot drilling." In *IEEE Int. Conf. on Robotics and Automation.* Rome, Italy. To appear.

Appendix A: Vision System Modeling and Calibration Techniques

This appendix presents some technical details about the vision system used in the experiments presented in the thesis, as well as a description of the form of the pinhole camera projection model. The structure of the camera system and the processing platform are described in some detail. A multi-camera calibration algorithm and a simulation environment for visual servoing, which have both been used in the experiments and simulations, are presented. The multi-camera calibration procedure was originally presented in the report listed below.

Publications.

Olsson, T. (2001): "Vision guided force control in robotics." Master's Thesis ISRN LUTFD2/TFRT--5676--SE. Department of Automatic Control, Lund University, Sweden.

Other Publications

Olsson, T. (2004): "Feedback control and sensor fusion of vision and force." Licentiate Thesis ISRN LUTFD2/TFRT--3235--SE. Department of Automatic Control, Lund, Sweden.

2

Background

2.1 Introduction

In this chapter, we present some background material relevant to the work described in the rest of this thesis. The main part of the chapter concerns topics relevant to the use of external sensors in feedback control, with special focus on visual tracking and feedback, and combined vision/force control. A brief outline of previous research and the state-of-the-art in the relevant research areas is given, together with some presentation and discussion of alternative approaches and topics, that have not been treated in detail in this work. Additionally, a short introduction to industrial robots is given, with particular emphasis on the properties of current industrial robots that are important from a perspective of external sensor control. The description is focused on the class of non-redundant industrial arm-like serial manipulators with built-in position/velocity control.

2.2 Industrial Robotics

The industries in which most robots can be found are in the domain between small-scale short-series production and dedicated automation technologies developed for large-volume manufacturing. The demands on productivity in the latter type of automation requires costly and highly complex systems in order to be satisfied, while in the former case robot automation is not flexible enough to be able to provide a cost-efficient alternative to manual labor. The majority of industrial robots have traditionally been working with simple repetitive tasks such as spot welding. However, the domain of applications of industrial robots has grown to include tasks such as material handling, grinding, deburring, polishing,

gluing, and sealing. Currently, efforts are underway to introduce industrial robots into small and medium enterprises (SMEs), where the lack of flexibility and the high cost of training and deployment have previously made robot automation too costly, considering the small series production of typical such companies. This will increase the demands for concepts of programming and sensing with low cost, high reliability, and flexible usage.

Most current industrial robot systems are based on serial arm type manipulators with a number of joints. The robot is actuated through the use of a system consisting of a power supply and amplifiers, driving the motors for each joint. The motors are connected to the joints through the transmission or gearbox. The purpose of the gearbox is to reduce the high speed and low torque on the motor side to a desired lower speed with higher torque on the arm side. In addition, the use of gearboxes with high gear ratio helps to decrease the dynamic couplings and nonlinear effects from the moving links of the robot, simplifying the robot control problem. However, effects such as bending of the gear teeth introduce a significant flexibility which complicates the control, especially for control problems involving *non-collocated* sensors, i.e. sensors mounted on the arm side of the gears. Traditionally, the built-in motion control uses encoders or resolvers measuring only the position of the motors (i.e. before the transmission). These positions will differ from the arm side positions that are desired to control, due to flexibility in the transmission. This difference between measured and controlled positions is particularly important when external forces are applied to the robot.

Motion Control

The motion control problem for an industrial robot, illustrated in the diagram in Fig. 2.1, can be divided into motion/path planning and trajectory generation, and trajectory tracking. The motion planning and trajectory generation problem involves specifying a suitable path in the workspace, and generating a trajectory that leads the robot to follow this path. The motion planning is either specified by the robot operator using manual teaching, or by using off-line programming tools. From the trajectory generation the reference inputs to the trajectory following are obtained as timed trajectories of desired (joint-level) positions, velocities, and accelerations or torque feedforward signals.

For the trajectory following problem, many robot control techniques have been presented. In the disturbance-free case, using joint-level PD controllers with feedforward it is possible to achieve exponential trajectory tracking [Takegaki and Arimoto, 1981; Murray *et al.*, 1994]. In practice, extensions such as gravity compensation and integral action are usually necessary. Feedback linearization techniques, usually referred to as *in-*

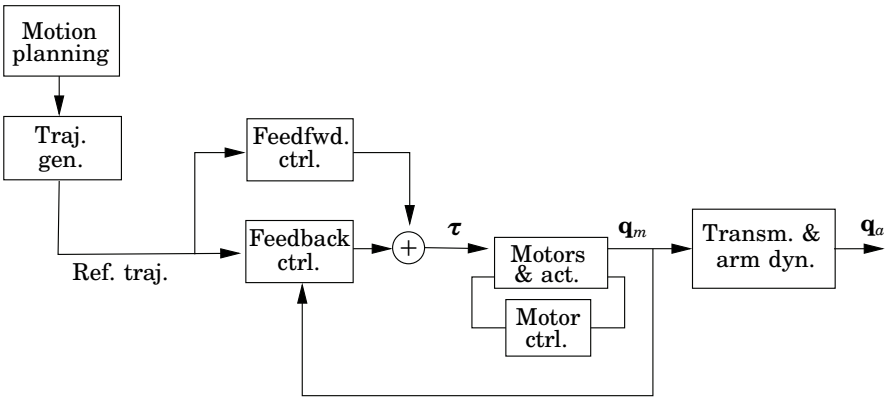

Figure 2.1 Block diagram giving a somewhat simplified description of a motion controller structure for an industrial robot system, with $\mathbf{q}_a$ and $\mathbf{q}_m$ representing arm and motor positions, respectively.

verse dynamics or *computed torque* control, are also frequently used in the literature [Craig, 1989; Murray *et al.*, 1994; Spong and Vidyasagar, 1989]. However, in practice the multi-variable robot motion control problem is very complex, as the full dynamics includes many other effects that need to be taken into account. Examples are the compliance in the transmission and links, sensor and actuator dynamics, external forces on the end-effector, and nonlinear effects such as friction and backlash. Therefore, standard computed torque control is generally not used in current industrial robot control systems. Instead, the control problem is divided into joint-wise controllers in combination with a multi-variable part. In this way, joint-level control problems such as motor torque control, and nonlinearities and disturbances in the joint dynamics such as friction and backlash, can all be handled locally in separate joint controllers. The multi-variable controller, or *arm control level* is then implemented on top of the joint-level motor control [Nilsson, 1996]. The joint-level controllers are usually executed at a higher sampling rate than the more complex arm controller. In addition, the control structure is often divided into a feedforward- and feedback structure as shown in Fig. 2.1. The trajectories obtained from the trajectory generation are used in the feedforward part of the controller, which can for instance be designed using computed torque methods. For improved performance, the feedforward generator may also be augmented with a model of the robot flexibilities, for which the model parameters may be found by system identification techniques [Wernholt, 2004]. As the feedforward part generally provides most of the performance,

the feedback controllers are often implemented as simple PID- or cascaded velocity- and position controllers, where the remaining dynamical couplings are treated as disturbances. More advanced methods based on multi-variable models including flexibilities and dynamical couplings, can be used to obtain improved feedback performance.

2.3 Robot Vision and Camera Feedback

Computer vision is the scientific discipline concerned with the design and analysis of artificial systems that are able to extract information from multi-dimensional data, usually two-dimensional images. In a robotics context, the term *robot vision* is often used when computer vision techniques are used in robotic applications. Cameras and robot vision techniques can provide a cost-effective way to obtain powerful workspace sensing capabilities. The location, structure and motion of objects in the environment of the robot can all be obtained using cameras, making vision a very powerful sensing modality. Computer vision and image processing techniques are frequently used in industry for monitoring and inspection of parts, or as a measurement device for detection, recognition, and measurement of objects to be manipulated. It is also possible to use vision for determining the position of the robot itself, relative to other objects in the environment. Making the robot act based on this information creates a feedback loop, for instance if the measured position is used for navigation in a mobile robot or for positioning of a robot arm. The field of *visual servoing* is concerned with systems which use the information extracted from vision systems for feedback control [Hutchinson *et al.*, 1996]. Visual servoing is strongly related to many other fields of computer vision. The research on *imaging and camera technology* has resulted in the availability of cheap and robust imaging devices of high quality. Systems for *image processing and image analysis* [Gonzalez and Woods, 1992], for instance for noise suppression, edge enhancement or color-space conversion, are usually necessary in the first step in the information extraction process. Methods from *feature extraction* are used for compressing the image data to a smaller number of features, characterized by some geometric property such as position, direction, or size. *Scene reconstruction* from images may be necessary for world modeling, if the overall structure of the observed scene and its objects is not known a priori. If the structure of the objects of interest is known, but not their position in the scene, *object recognition* and *pose estimation* techniques can be used [Trucco and Verri, 1998; Carceroni and Brown, 1998]. Camera *calibration* and other types of calibration techniques are used to obtain better models of the image formation process, for instance by obtaining the parameters describing the

inner structure of the camera/lens system [Zhang, 1999], or the transformation between a robot-mounted camera and the robot hand [Tsai and Lenz, 1989; Daniilidis, 1999]. Although a full review of all of these topics is beyond the scope of this work, a short background to the components of a vision-based control system is given below. The focus will be on topics and systems relevant for the methods used and developed in following chapters.

Imaging and Camera Technology

The function of a (digital) camera can be seen as taking the total light energy in a given frequency band that falls onto each element (pixel) of its light-sensitive sensor area during a certain integration time, and transform it into one or several digital values representing the *image intensity* (possibly at several different frequency bands or "colors") at this point. The light rays entering the camera are usually transmitted through a *lens system* in order to be concentrated onto the useful area of the small light-sensitive sensor. The intensity values for each pixel form a two-dimensional grid, a digital *image*, which is then transmitted from the camera using some type of data bus, into an image processing computer system. Many modern cameras also include some on-board high-level or low-level image processing, such as filtering or color conversion.

Most of the different steps of the image acquisition process described above are important for proper understanding and modeling of a camera system. The camera *projective* geometry, relating the directions of the incoming light rays to coordinates in the image, is important in order to relate positions of objects in the image to the corresponding positions of the objects in the world. The geometry and calibration of the most commonly used camera projection model, the *perspective* or *pinhole* camera model, is described in Appendix A. The literature on projective geometry in computer vision is huge, for practically oriented introductions to camera geometry see for instance [Trucco and Verri, 1998] or [Ma *et al.*, 2003].

The main types of light-sensitive sensor types for integration of the image intensity are Charge-Coupled Device (CCD) and Complementary Metal-Oxide Semiconductor (CMOS) sensors. Both sensors transform the light energy striking each pixel into electrons, and read out the accumulated charge from the light-sensitive area. In a CCD camera the charges are transported directly across the chip before being read out, while in a CMOS sensor transistors amplify and move the charge at each pixel. CMOS provides a more flexible read-out procedure with better control, where each pixel can be read individually. On the other hand, CCD cameras traditionally produce better quality images with lower noise levels than CMOS cameras. However, CMOS have a superior low power consumption, are comparatively easy to manufacture and tend to be signifi-

cantly less expensive compared to CCD sensors[1]. The image intensity can be influenced by the size of the aperture and the length of the exposure interval, determined by the shutter speed. The size of the aperture also determines the focal depth, meaning the range of distances over which objects in the scene will be in focus. As industrial cameras and lens systems do not in general have automatic focusing, there is often a trade-off to be made at setup, between image intensity, shutter speed, and focus. Artificial lighting systems are often necessary in order to achieve acceptable performance, especially if a fast shutter speed is required. Digital cameras use electronic shuttering, which can be divided into two different types. With a *global shutter* (also referred to as *frame shutter*), the shutter resets all sensor elements simultaneously, and all sensor elements start and stop accumulating charge at exactly the same time. Given a sufficiently short exposure time, the image of a moving object will be undistorted, as if frozen in time. With *rolling shutter* on the other hand, the sensor elements do not collect light at the same time, but at slightly different times for each row in the sensor grid. The resulting image of a moving object will look distorted, as the top and bottom parts will represent the position of the object at different times. Recently, methods have been suggested that exploit this distortion for velocity estimation [Ait-Aider *et al.*, 2006]. Traditionally, the global shutter was used by CCD sensors while CMOS sensors used a rolling shutter, although many modern CMOS cameras, such as the Basler A602fc used in the experiments in this thesis, use a global shutter.

Differently from consumer digital video (DV) cameras, special industrial cameras use communication structures and protocols with the ability to read images from the camera in real-time. Currently, the most common interfaces for cameras are CameraLink [Scheiber, 2006] and IEEE-1394 (FireWire) [FireWire, 2006]. CameraLink is a parallel interface with very high bandwidth, and is the more expensive solution due to the special cables and frame grabber hardware needed. In addition, due to the lack of a standard communication protocol, it is the most difficult solution to integrate. IEEE-1394, described in more detail below, is a serial interface which uses standardized hardware, software, and computer interfaces, and is therefore available at a significantly lower cost. However, standard IEEE-1394 is limited to a 400 Mbits/s data rate, which is several times slower than CameraLink interfaces, and cable length is limited to around 5 meters. A newer version of the standard exists—the IEEE-1394b specification—which avoids these drawbacks. An interesting alternative

[1]As an example, at the time of writing most commercial optical computer mice use a CMOS camera as the optical sensor for determining movement. The sensor generally has a resolution of approximately 30x30 pixels or less, with a sampling rate of 2.5 kHz or faster.

Figure 2.2 Firewire camera Basler A602fc and standard 6-pin connector.

is presented by Gigabit Ethernet, which would provide a very high bandwidth, the possibility for longer cables, and relatively low cost. A camera interface standard for Gigabit Ethernet, called GigE Vision [Scheiber, 2006], is emerging as the *de facto* standard for machine vision cameras using Gigabit Ethernet.

IEEE-1394 Camera Systems. IEEE-1394 is a serial communication interface and bus standard, which has found many uses in consumer products such as camcorders, hard drives and networking. As IEEE-1394 cameras are also still the most widely used type of industrial digital camera, and since all experiments in this work are based on IEEE-1394 camera systems, we will briefly describe the properties of a typical IEEE-1394 camera system. The description is based mainly on a Basler A602fc camera connected to a Linux-based platform, corresponding to the system used in most of the experiments. However, the functionality of other cameras is very similar, thanks to the standardization of the IEEE-1394 hardware and software interfaces.

The FireWire standard was invented by Apple Computer in the late 1980s, and adopted as the IEEE-1394 standard in 1995. Using FireWire up to 63 peripheral devices can be connected, allowing peer-to-peer device communication. This makes it possible for the communication to take place without use of the CPU, and devices may communicate by direct memory access enabling very high-speed and low-latency communication.

Flexibility of connections is provided by the ability for plug-and-play and hot swapping. Industrial IEEE-1394 cameras are connected to the host computer using cables with 6-pin connectors for data transfer and power supply, seen in Fig. 2.2. Data transfer rates up to 400 Mbit/s are specified and available in most cameras. Around 20% of the bandwidth is reserved for asynchronous communication, such as register reads/writes for camera status, control and command data. The IEEE-1394b amendment introduced in 2002 allows a transfer rate of 800 Mbit/s, with possibilities for future data rates up to 3.2 Gbit/s. IEEE-1394b uses a new 9-pin connector, but is backwards compatible with the slower rates and standard connectors used by IEEE-1394. It additionally supports optical connections up to 100 m in length. Digital cameras for the IEEE-1394b interface have relatively recently become available on the market.

IEEE-1394 cameras for real-time vision applications are usually run in *isochronous* (i.e. equidistant sampling) image capture mode, using direct memory access, where exposure and image readout are handled automatically by the camera. At the start of exposure, the light-sensitive elements begin to accumulate charges. When the exposure stops, either by external triggering or after a pre-programmed time, the accumulated charges are read out and converted to voltages. On many cameras, such as the Basler A602fc used in the experiments, the readout and voltage conversion first transfers the data into separate pixel memories, so that exposure of the next image can begin immediately, before the image has been read out from the sensor. Data is then read out from the pixel memories row-wise or column-wise using a data bus, amplified and analog-to-digital converted, and transferred into a camera-internal frame buffer. From the frame buffer, frame data is sent to a IEEE-1394 *link layer* controller which divides the data into packets for the *physical layer* controller, which transmits data to the IEEE-1394 card in the host computer. On the host computer, images are transferred into a ring buffer of frame buffers. The transfer is usually handled using direct memory access, thereby avoiding using valuable CPU resources and allowing significant parallelism between capture and processing.

The achievable frame rate and total delay is mainly limited by three factors; exposure, readout and transmission. Additional rate limitations are imposed by the time required for the image processing in the host computer. The effective delay due to the finite exposure time is often taken to be half the exposure time, although effects such as “smearing” and motion blur makes this delay difficult to model accurately, see for instance [Chen *et al.*, 1996]. The frame readout time is the time required for reading out image data from the sensor to the camera image buffer, and is roughly proportional to the image height (or width). The transmission time is the time required for all packets in a frame to be sent over the IEEE-1394 bus,

and is proportional to the number of data packets transmitted (and therefore to the number of pixels). A small delay is also introduced between the start of sensor readout and the start of IEEE-1394 transmission.

Calibration Methods for Camera Systems

As the properties of the camera and its environment frequently change due to camera and lens system reconfigurations, calibration procedures that are able to find reliable estimates of system parameters are essential for most vision systems. Although feedback methods are robust to modeling errors, calibration is used as a first step also in many applications of vision based control. The most common examples of calibration parameters are camera positions and internal camera parameters, often referred to as the *extrinsic* and *intrinsic* camera parameters. The relevant extrinsic parameters are related to some specific coordinate system. For robot-mounted (*eye-in-hand*) cameras this coordinate system is attached to the robot wrist, in which case the term *hand-eye calibration* is used.

Intrinsic Camera Parameters. The intrinsic (internal) camera parameters describe the internal structure of the camera and lens system. For the pinhole camera, as described in Appendix A, the intrinsic camera parameters are the focal length, skew, principal point, aspect ratio, and coefficients of some function describing the radial- and tangential distortion. The extrinsic (external) parameters describe the camera position/orientation with respect to the scene. Most methods for camera calibration use a special calibration object with accurately known geometry, covered with special markers for accurate detection and localization. Multiple images of the object are captured from several different positions and orientations, from which the calibration parameters can be computed. In [Zhang, 1999], a method is presented in which both intrinsic and extrinsic camera parameters are estimated, using an algorithm consisting of a linear initialization followed by a nonlinear least-squares optimization. Multiple images of a planar target are used, which highly simplifies the construction of accurate calibration objects. A detailed description of the methods used for calibration in this thesis are presented in Appendix A.

Extrinsic Camera Parameters and Hand-Eye Calibration. In addition to the internal camera parameters, in vision-based control it is necessary to establish the geometric relations between the sensor and actuator coordinate systems. The calculation of the relative position and orientation between the robot end-effector and a camera, which is mounted rigidly onto the end-effector, is referred to as hand-eye calibration. Finding this relationship between the positions of the sensor- and actuator

frames is a standard problem in vision guided robotics, and many different solution methods exist. The problem is usually formulated as finding the unknown transformation $\mathbf{X}$ from the hand-eye equation

$$\mathbf{AX} = \mathbf{XB}, \tag{2.1}$$

which is a special case of the Sylvester equation (as noted in [Andreff *et al.*, 2001]), where $\mathbf{A}$, $\mathbf{B}$ and the unknown $\mathbf{X}$ belong to the *Special Euclidean* group $SE(3)$ of rigid transformations. $\mathbf{A} = \mathbf{A}_2^{-1}\mathbf{A}_1$ and $\mathbf{B} = \mathbf{B}_2\mathbf{B}_1^{-1}$ are given by measurements of the position and orientation of the robot hand, and of the camera with respect to some object in the world, see Fig. 2.3. Typically a number of movements are performed to get measurements of different $\mathbf{A}$ and $\mathbf{B}$, which are used in order to solve for $\mathbf{X}$ in Eq. (2.1). Both linear [Tsai and Lenz, 1989; Daniilidis, 1999] and nonlinear methods [Horaud and Dornaika, 1995] have been suggested. In [Tsai and Lenz, 1989], it was shown that at least two motions with non-parallel rotation axes are required, and the problem was solved by dividing Eq. (2.1) into two separate equations for rotation and translation, respectively. The method of [Daniilidis, 1999] uses a dual quaternion representation of $\mathbf{A}$ and $\mathbf{B}$ to simultaneously solve for the rotation- and translation parts of $\mathbf{X}$ using linear methods. In the multi-camera calibration algorithm developed in Appendix A, a hand-eye calibration method is used to find the location of a robot-mounted calibration target.

Image Processing and Feature extraction

The raw image data is usually obtained in several channels, each consisting of an array of data of dimensions height×width. Processing the entire image at a sufficiently fast sample rate may require very large computational resources, and the raw image data generated by the cameras needs to be compressed into a more compact representation for the controller. The first steps are the image processing and feature extraction, where the image data is filtered and relevant image data is extracted into a measurement vector $\mathbf{y}$. Some classes of methods exist where no explicit feature extraction step is needed, such as methods based on optical flow [Allen *et al.*, 1991], or eigenspace methods [Schuurman and Capson, 2004; Deguchi and Noguchi, 1996]. To the class of feature-less methods can also be referred the intensity-based methods of Chapter 5. However, most methods operate by extracting the positions of image features, usually parts of the image with sharp changes in intensity, such as edges, corners or special markers. This class of methods is the focus of Chapter 4.

Feature Extraction. Edges and corners can be detected and localized using convolution with suitably chosen kernels, such as Sobel and Prewitt

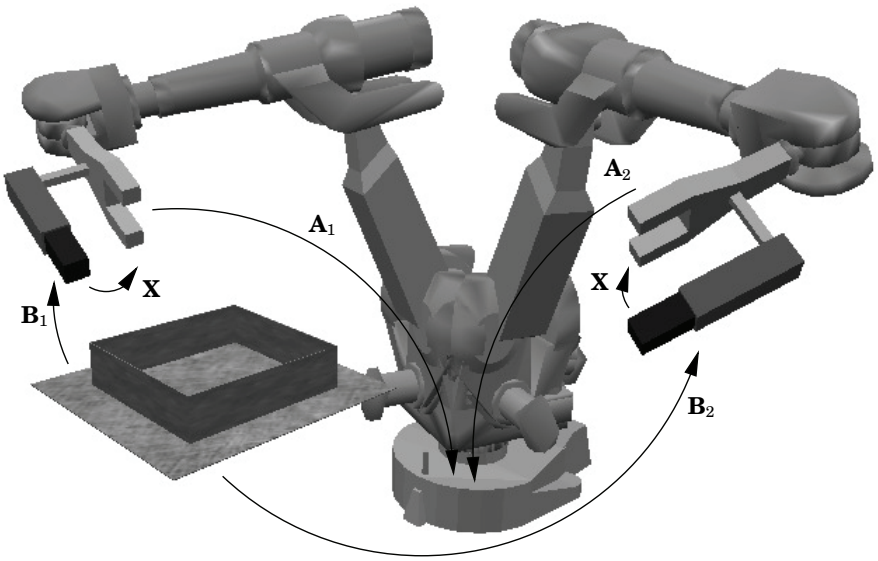

Figure 2.3 Two different positions of the robot, with the frames relevant to hand-eye calibration.

operators [Gonzalez and Woods, 1992]. In general, such methods detect edges as optima of an approximation of the spatial first derivatives in the image. A way to improve the robustness is to use a multi-scale approach, where features are first robustly detected and localized at a coarse scale, and then iteratively refined at finer scales, see Fig. 2.4 for an illustration. Another frequently used method is the Canny edge detector, which combines an optimal (with respect to a certain criterion) linear convolution with elimination of non-maxima and weak edges [Canny, 1986; Trucco and Verri, 1998], which makes it robust but time-consuming. The Harris corner detector uses the Taylor expansion of the sum of squares difference between image regions as a function of displacement, and an eigenvalue test can be used to detect corners [Harris and Stephens, 1988]. Interpolation can be used to achieve sub-pixel accuracy.

A different technique, which does not require the image derivatives and that can be used for both one- and two-dimensional features, is the SUSAN feature detector [Smith and Brady, 1997]. The method works by comparing the brightness of each pixel within a mask to the center (nucleus) of the mask. The size, centroid and second moments of the area with similar brightness are then used in order to detect and localize edges and corners. A method which similarly works by comparing the brightnesses

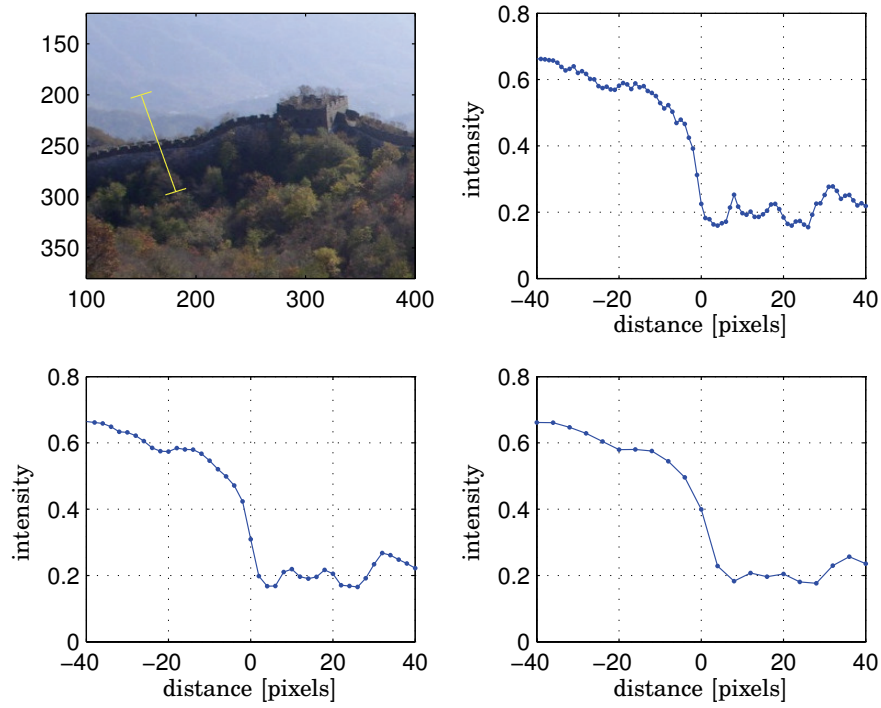

Figure 2.4 Illustration of edge detection along the line marked in the top left image. Intensity variations along the line can be seen in the top right figure. The bottom left and bottom right plots show the intensity variations at two coarser scales. The initial edge detection is performed faster and more reliably at coarser scales, followed by accurate localization at successively finer scales.

of pixels in a mask is the FAST feature detector presented in [Rosten and Drummond, 2005], which has been reported to give a significant improvement in performance.

A different kind of particularly distinctive features are the SIFT features introduced by [Lowe, 1999]. In SIFT, scale-space methods are used for detection of extrema and selection of suitable key-points, for which orientation assignment of gradients are used to compute 128-vector descriptors. The invariance of the SIFT feature representation with respect to image transformations, such as image rotation and scaling, has made it a powerful and frequently used matching technique in stereo vision and recognition problems.

Robust data correspondence. Although robust feature detection algorithms such as SIFT have been developed over the last decade, in all visual processing systems there exists a risk for false matches. In such cases, features are detected which do not correspond to the desired object or image structures. This is often caused by background clutter, specular highlights or other non-modeled effects. In order to minimize the number of such false matches, feature-based methods are often combined with window-based tracking, in which features are followed through the image sequence frame by frame. The position of the feature search windows can be obtained from the predicted position, computed based on the object position in the previous sample. By searching for the feature only in a small image window around the predicted position, the risk for false matches is decreased. In order to eliminate the effects of remaining matching errors, methods such as RANSAC (RANdom SAmpling Consensus) [Fischler and Bolles, 1981] or ICP (Iterated Closest Point) [Besl and McKay, 1992] can be used, as well as voting methods such as the Hough transform, see [Gonzalez and Woods, 1992]. Such methods can often be used to match features even in the presence of large numbers of outliers in the data sets. RANSAC attempts to find a small set of data to which a model can be fitted, by iteratively selecting a random subset of all data points. The number of random tests required depends on the (expected) fraction of outliers in the data set, and the desired probability of finding a match. Other options for handling outliers are presented by statistical methods such as M-estimators and iteratively re-weighted least-squares methods [Drummond and Cipolla, 2002]. In these methods, the relative weight of large errors is reduced as compared to standard least-squares methods, thereby decreasing the influence of large outliers on the resulting output. For a recent review and evaluation of a number of robust estimation techniques useful for real-time robot vision, see [Malis and Marchand, 2006].

Motion and State Estimation

The measurement and analysis of motion from 2D images is an important sub-field of computer vision. In the presence of motion it is possible to extract information that could not be obtained from a single image, such as the geometry of objects (*structure from motion*). However, this work is focused on the case where the structure of the objects are known, and our primary concern will be the estimation and control of the object motion. A brief review of visual motion tracking methods is given in [Blake, 2006]. In general, visual tracking problems can be formulated as estimation of the states in a discrete-time nonlinear system

$$\mathbf{x}(k+1) = \mathbf{f}_d\left(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k)\right) \tag{2.2}$$

$$\mathbf{y}(k) = \mathbf{h}_d\left(\mathbf{x}(k), \mathbf{e}(k)\right) \tag{2.3}$$

with state vector $\mathbf{x}$, (image space) output $\mathbf{y}$, control input $\mathbf{u}$ and affected by the disturbance $\mathbf{v}$ and noise $\mathbf{e}$. In systems with a large number of degrees of freedom, the output space can often be reduced to a vector of the most important modes, e.g., by eigenspace methods such as *active shape* models [Hill *et al.*, 1996]. Examples of such non-rigid applications include tracking in medical images, and tracking of complex hand, lip and facial motions.

The main difficulties in visual tracking problems are due to the nonlinearity of the system, and the complex disturbances with potentially large amounts of outliers in the measurements. Various estimation methods have been proposed in order to obtain better robustness to such disturbances.

Kalman Filtering. Traditionally, the most frequently used method for nonlinear estimation in dynamic vision problems has been the Extended Kalman Filter (EKF) [Gelb, 1974]. The EKF is obtained by using a standard Kalman Filter and by linearizing the process model around the current estimates. The equations for the EKF are given by the equations

$$\begin{aligned}
\hat{\mathbf{x}}(k+1|k) &= \mathbf{f}_d(\hat{\mathbf{x}}(k), \mathbf{u}(k), \mathbf{0}) && (2.4)\\
\mathbf{P}(k+1|k) &= \mathbf{A}_k \mathbf{P}(k) \mathbf{A}_k^T + \mathbf{W}_k \mathbf{Q}_k \mathbf{W}_k^T && (2.5)\\
\mathbf{K}_k &= \mathbf{P}(k|k\text{–}1)\mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}(k|k\text{–}1)\mathbf{H}_k^T + \mathbf{E}_k \mathbf{R}_k \mathbf{E}_k^T)^{-1} && (2.6)\\
\hat{\mathbf{x}}(k) &= \hat{\mathbf{x}}(k|k\text{–}1) + \mathbf{K}_k(\mathbf{y}(k) - \mathbf{h}_d(\hat{\mathbf{x}}(k|k\text{–}1), \mathbf{0})) && (2.7)\\
\mathbf{P}(k) &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}(k|k\text{–}1) && (2.8)
\end{aligned}$$

where $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the covariance matrices for $\mathbf{v}(k)$ and $\mathbf{e}(k)$, and where

$$\begin{aligned}
\mathbf{A}_k &= \frac{\partial \mathbf{f}_d}{\partial \mathbf{x}}(\hat{\mathbf{x}}(k), \mathbf{u}(k), \mathbf{0}) && (2.9)\\
\mathbf{W}_k &= \frac{\partial \mathbf{f}_d}{\partial \mathbf{v}}(\hat{\mathbf{x}}(k), \mathbf{u}(k), \mathbf{0}) && (2.10)\\
\mathbf{H}_k &= \frac{\partial \mathbf{h}_d}{\partial \mathbf{x}}(\hat{\mathbf{x}}(k), \mathbf{0}) && (2.11)\\
\mathbf{E}_k &= \frac{\partial \mathbf{h}_d}{\partial \mathbf{e}}(\hat{\mathbf{x}}(k), \mathbf{0}) && (2.12)
\end{aligned}$$

are the Jacobians of $\mathbf{f}_d$ and $\mathbf{h}_d$ from the process model in (2.2)–(2.3). Although the EKF works well in many vision applications, it has several important problems. In addition to the well-known difficulty of proving even local stability of the estimator, a practical problem is that in many applications the measurement $\mathbf{y} \in \mathbb{R}^n$ is a very high-dimensional vector. The most computationally expensive part of the EKF computations is then

the update of the Kalman gain $\mathbf{K}_k$ in Eq. (2.6), which will have a time complexity of $O(n^3)$. In [Wunsch and Hirzinger, 1997] it was suggested to use the EKF with the measurement defined in task space instead of image space, thereby decreasing the dimension of $\mathbf{y}$. For systems with linear state dynamics but a nonlinear measurement equation $\mathbf{h}$, it is often possible to use a Kalman filter with complexity $O(n)$ that linearizes only the measurement equation and uses a covariance estimate $(\mathbf{H}_k^T\mathbf{H}_k)^{-1}\sigma^2$, as shown in Chapter 4.

The linearized and approximate nature of the EKF is also a problem in vision applications, where both the state dynamics and the measurement equations are usually highly nonlinear. The so called *unscented transformation* has been introduced as a method to propagate the mean and covariance information through nonlinear transformations [Julier and Uhlmann, 2004]. The resulting Unscented Kalman Filter (UKF) has been reported to significantly outperform the EKF in many applications. The results of a simulation comparison of 6-degree-of-freedom motion tracking, with strong nonlinear perspective effects in the observations, can be seen in Fig. 2.5. In this simple example the UKF improves slightly on the EKF performance, especially during fast motions when the effects of the nonlinear camera projection are apparent. The UKF can be interpreted as a special case of the so called *Linear Regression Kalman Filter* (LRKF), in which the nonlinear process and measurement functions $\mathbf{f}$ and $\mathbf{h}$ are linearized by statistical linear regression, see the note by [Lefebvre *et al.*, 2002].

Sequential Monte Carlo Methods (Particle Filtering). The underlying assumption of the Kalman filters, that the noise can be modeled as uncorrelated and Gaussian, does not hold well for systems with large fractions of measurement outliers and complex spatial correlations, properties that characterize many vision-based systems. If a Kalman filter is to be used in such situations, approximations and extra outlier-removal steps should be employed. Particle filtering was presented in [Gordon *et al.*, 1993] as an alternative to the EKF. The key idea in particle filtering is to use a sample-based representation of the conditional probability density function $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ of the state given the available measurements. This pdf is represented by a set of random samples or particles with associated weights, which are updated iteratively using the conditional probabilities of the measurements and Bayes' law. As the number of particles increases, this characterization approaches that of the optimal Bayesian estimate [Arulampalam *et al.*, 2002].

The main advantage of particle filters is their ability to handle non-Gaussian, nonlinear systems without explicit linearization. This makes them suitable for problems in computer vision, where nonlinear motion

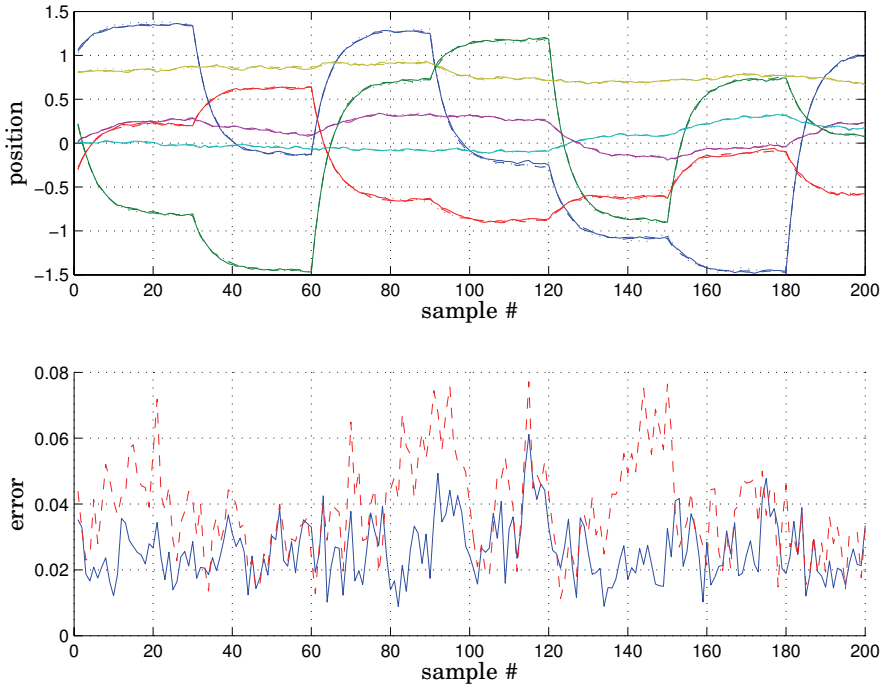

Figure 2.5 Comparison between UKF and EKF for estimation of 6-degree-of-freedom motion with camera measurements of 8 corner features, disturbed by Gaussian white noise. Top plot: True states (*solid*), UKF estimate (*dashed*) and EKF estimate(*dotted*). Bottom plot: norm of the estimation error for UKF estimate (*solid*) and EKF estimate(*dashed*).

models and complex state and position representations are common. Handling multi-modal probability distributions is especially important for robustness in visual tracking. In many situations, severe background clutter can easily cause Kalman filter based solutions to lose track, since the unimodal nature of the probability distributions makes the tracker "lock on" to the wrong image structures [Isard and Blake, 1998]. A drawback is the large computational power required, as the number of particles needed in practice increases rapidly with the dimension of the state space. Fewer particles may be used if particles are chosen according to some suitable *proposal distribution* [Arulampalam *et al.*, 2002]. Better proposal distributions may be generated by combining the particle filter with Kalman filters or unscented filters into a *Kalman Particle Filter* or *Unscented Particle Filter*, resulting in improved performance at the price of increased

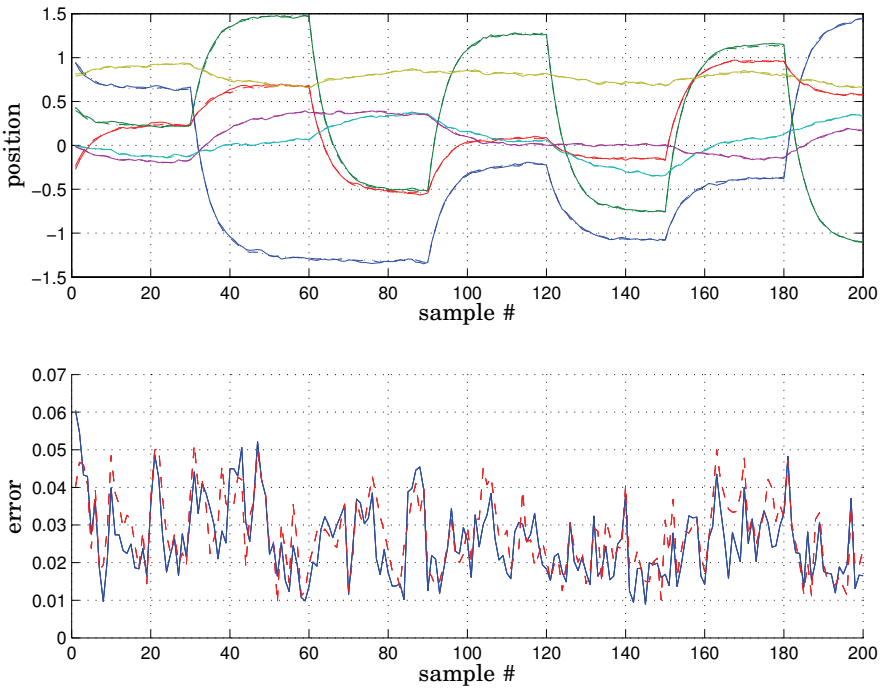

Figure 2.6 Comparison between UKF and Particle (SIR) filtering for estimation of 6-degree-of-freedom motion with camera measurements of 8 corner features, disturbed by Gaussian white noise. Top plot: True states (*solid*), UKF estimate (*dashed*) and SIR estimate(*dotted*). Bottom plot: norm of the estimation error for UKF estimate (*solid*) and SIR estimate(*dashed*).

computational complexity [Li *et al.*, 2003].

The results of a comparison between a *Sampling Importance Resampling (SIR)* or CONDENSATION-type particle filter [Isard and Blake, 1998; Arulampalam *et al.*, 2002] and an UKF can be seen in Fig. 2.6, in a simulated 6-degree-of-freedom (6-DoF) motion tracking experiment. The comparatively low number $N = 2000$ of particles, in combination with the six degrees of freedom of the motion, makes the SIR filter perform similarly to the UKF in this case. It should be noted, however, that compared to the UKF and EKF the particle filter is able to handle more general models with non-Gaussian noise. The price is a significantly larger computational cost, which will prevent real-time implementations at high frame rate, see Fig. 2.7.

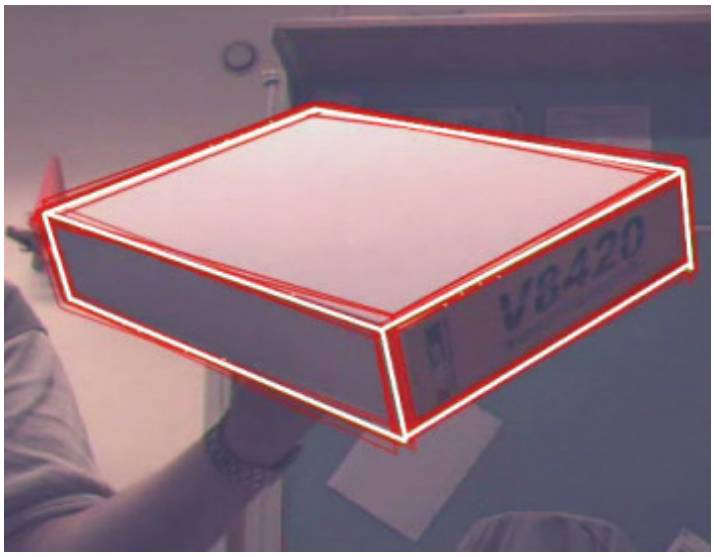

Figure 2.7 Six degree-of-freedom tracking an object at 30 images/second, using a Sampling Importance Resampling (SIR) particle filter. The object model is superimposed as a wireframe object, together with the positions of some of the particles. More cluttered scenes can be tracked by using well-tuned models of system dynamics and disturbances, or by increasing the number of particles.

Feedback Control and Visual Servoing

During the 1970s, it was realized that camera feedback could be used to correct the trajectories of a robot, thereby increasing the task accuracy. The term *visual servoing* was introduced for the technique of using cameras for feedback. In later years, the emergence of cheap, powerful computing power has increased the potential of visual servoing systems dramatically. A review of the research and history of visual servoing from the start until the mid 1990s can be found in [Hutchinson *et al.*, 1996].

In order to achieve both satisfactory performance and a manageable system complexity, the overwhelming majority of practical visual servoing controllers use a cascaded structure, with the vision-based feedback loop closed around the existing inner motion control loop. The inner control loop usually has a significantly higher bandwidth than the outer loop, greatly simplifying the design of the visual servo controllers. In practice, the inner loop is often modeled as a velocity control loop, providing ideal velocity control, and the visual servo will close the outer position/vision loop. An issue of great importance for the visual servo control design is

whether the control objective, and therefore the controller, is defined in task space or in image space. In the former case, referred to as *position-based* visual servoing (PBVS), the task space coordinates of the robot are estimated and controlled to their desired values, again defined in task space. This reduces the design problem for the visual servo loop to the problem of designing a standard position control with arm side measurements. For *image-based* visual servoing (IBVS), no explicit pose estimation is performed, and instead the errors are defined directly in (image) feature space. This control problem is in general more difficult, since the nonlinear camera projection function must be included in the model, and the motions in different degrees of freedom in feature space are highly dynamically coupled [Hutchinson *et al.*, 1996]. As a simplification in standard image based techniques, the inner (velocity) dynamics is assumed to be considerably faster than the closed-loop system. In this case, a model

$$\dot{\mathbf{z}} = \mathbf{f}(\mathbf{u}) \tag{2.13}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{z}) + \mathbf{e} \tag{2.14}$$

is often assumed, where $\mathbf{z}$ is the task space position, $\mathbf{u}$ is a commanded velocity, and $\mathbf{y}$ is a vector containing the image-space output. In many cases we can choose a model and transformation of $\mathbf{u}$ such that the model holds with $\mathbf{f}$ as the identity function. A controller

$$\mathbf{u} = \mathbf{K}\mathbf{J}^{\dagger}(\mathbf{z})(\mathbf{y}_r - \mathbf{y}) \tag{2.15}$$

where the matrix $\mathbf{J}$ is the *image Jacobian* or *interaction matrix* [Hutchinson *et al.*, 1996], and where $\mathbf{K}$ is a positive definite matrix, will in general drive the system such that $\mathbf{y} \to \mathbf{y}_r$. This approach is often referred to as *resolved rate* control. In the *task function* approach of [Espiau *et al.*, 1992], it is suggested to design the visual controllers to control an objective- or task function to zero in order to achieve convergence to the desired position, as well as to satisfy certain secondary objectives.

The advantage of image-based techniques is that the static positioning accuracy is independent of the calibration accuracy, especially if the image-space reference trajectory has been defined in a teach-by-showing approach [Hutchinson *et al.*, 1996]. Therefore, image-based techniques are suitable in situations where less information about the system parameters and the environment is available. It should be noted however, that the image Jacobian $\mathbf{J}(\mathbf{z})$ is in general a function of the task-space coordinates $\mathbf{z}$, specifically the depth (defined as the Z-coordinate as shown in Fig. A.1) of the features with respect to the camera. Therefore, some task-space quantities need to be estimated or approximated, just as for the position-based methods described in the next section. Additionally, in more complex servoing tasks the desired trajectories are often more naturally defined in

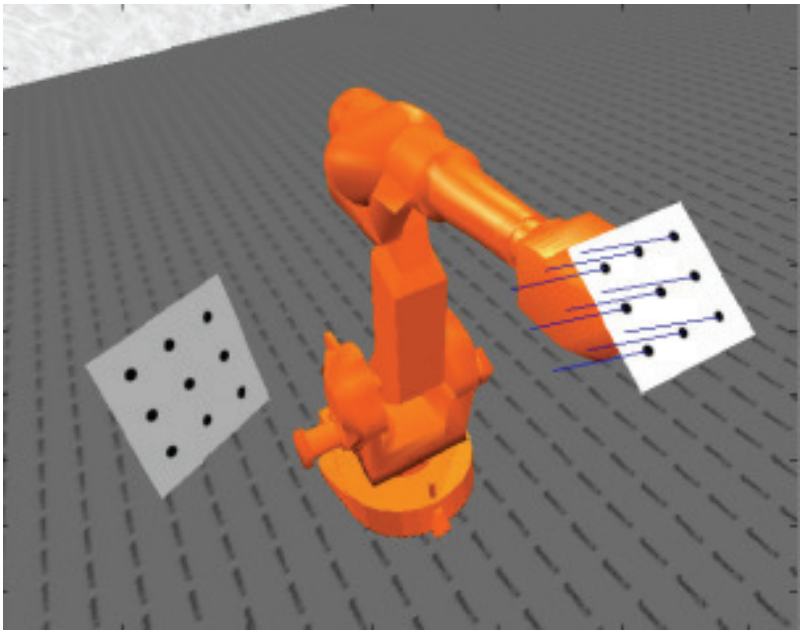

Figure 2.8 Illustration of motion trajectories generated by image-based visual servoing with a single set-point. The control system will attempt to impose trajectories that are as close as possible to straight lines in the image.

task space. Image-based methods require more elaborate path planning algorithms to be employed, in order to avoid image motions that correspond to infeasible task space trajectories. Hybrid methods that attempt to handle these issues by explicitly combining elements from image- and position-based methods have been developed [Deguchi, 1998; Chaumette and Malis, 2000; Corke and Hutchinson, 2001].

In Fig. 2.8, a simple case of image-based set-point control is illustrated. The robot is commanded to align the end-effector with the virtual target object to the left. The generated trajectories will correspond to almost straight lines in the image, but very little control over task space motion is provided.

In position-based techniques, the control signal is computed based on a value of $\mathbf{z}$, computed from the image data $\mathbf{y}$ by *pose estimation*. In some situations, such as for a number $n \geq 3$ of point features in a perspective camera, there exist analytic solutions to the pose estimation problem, often based on solving polynomial equations of relatively low order. Research on how to solve such analytical perspective-n-point problems have

a history since at least the mid 1800s, although many new solutions were developed in the computer vision field throughout the 80s and 90s. The review in [Carceroni and Brown, 1998] describes a number of the most frequently encountered analytical point- and line based pose estimation methods. However, since modern computer vision problems often involve hundreds or thousands of measurements with varying noise levels and degrees of reliability, iterative numerical and optimization-based methods are more relevant than the simple analytical solutions. Usually, the control algorithms are implemented in discrete time, and the pose estimation problem can be treated as a standard nonlinear optimization problem [Lowe, 1991], where the estimated value of $\mathbf{z}$ from the previous step may be used as a starting point for the iteration. In order to remove the requirement of a good initial estimate, other classes of methods employ iterative refinements to the solution obtained from a simplified pose estimation problem. Examples of such algorithms are the solutions based on the weak perspective and para-perspective projection models in [Dementhon and Davis, 1995; Horaud *et al.*, 1997]. In these algorithms, the system is initialized using the *linear* solution to the simplified problem, and successively refined until the full perspective solution is obtained.

For control purposes, or if predictive capabilities are desired for increased reliability of feature tracking, dynamic tracking approaches are useful. Traditionally, such methods are based on Kalman filters, although a vast number of different approaches have been proposed in the literature. The input to the estimator can be either the image-space measurements, or the resulting position from the pose estimation algorithm. The latter approach, although conservative in the general case, results in a modular structure with some degree of separation between the projection nonlinearity and the dynamics of the system.

The main advantage of position-based methods is that the reference trajectory, the measured position, as well as the control signal, can all be defined in task-space coordinates. This simplifies the control problem, as the dynamic model is usually expressed in task space. In Fig. 2.9, setpoint control for position-based servoing is illustrated. Compared to IBVS, using PBVS better control of the task space trajectories is obtained. The main drawback of position-based control is a decrease in the robustness to calibration errors. Errors in camera calibration parameters will lead to errors in estimated pose, and consequent errors in trajectories [Hutchinson *et al.*, 1996].

Dynamical Effects in Visual Servoing. Attempts to improve performance have led to different approaches in which the robot dynamics is taken into account in the design and analysis of visual servo controllers. Also when fast inner motion control loops are present, there are dynamic

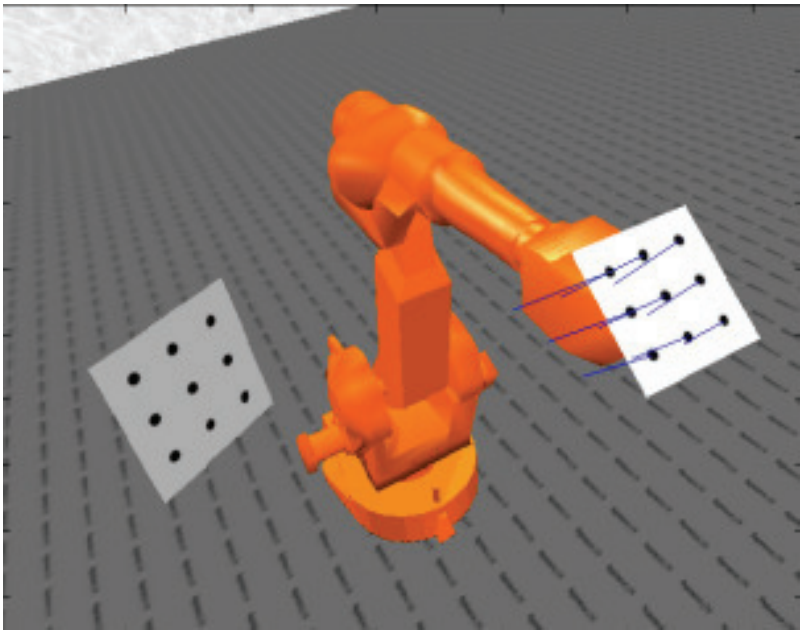

Figure 2.9 Illustration of motion trajectories generated by position-based visual servoing with a single set-point. Compared to Fig. 2.8, the control system will instead generate straight trajectories in task space, in this case parametrized using Euler angles.

effects that should be considered in the design and analysis. Among the most important dynamic effects of vision-based control systems are long and time-varying time delays, caused by the image capture, sensor read-out, image transfer, and image processing. Additionally, except for very high-speed vision systems, the sampled-data nature of visual measurements usually needs to be taken into account. It is also important to note that measurements from cameras are non-collocated with respect to the actuators. The effects of the non-collocation is a particularly crucial issue in control of flexible robots, as it increases the risk for instabilities and limit cycles [Nordin and Gutman, 2002]. However, using arm side measurements can also improve the tracking performance on the arm side, and makes it possible to obtain superior control performance.

In an early paper, [Corke and Good, 1992] analyzed the effects on closed loop performance of time delays, mechanical compliance, and gain variations from the perspective camera mapping. The tracking performance of visual servoing systems with inner velocity control was analyzed in

[Vincze, 2000], with respect to different controllers and models of the input-output latency from image capture and processing. It was recommended that a parallel processing structure, in which image processing and acquisition of the next frame are performed in parallel, should be used in combination with a fast frame rate for the best performance. In [Gangloff and de Mathelin, 2000] an approach based on Multi-Input-Multi-Output Predictive Control was used to control a 6 degree-of-freedom industrial robot, showing significant improvement over classical controllers. Predictive Control and H_∞-control have been proposed and evaluated for control of medical robots in surgical applications in [Cuvillon *et al.*, 2005]. Some researchers have also considered direct visual control of (rigid) robots without an inner motion control structure, see for example [Kelly, 1996; Deng *et al.*, 2002].

2.4 Force- and Interaction Control

Although the majority of industrial and other robots operate under pure position control, the problem of force control has drawn attention since the early days of robotics. In industrial robotics, force control is a crucial component in many industrial systems for polishing, deburring, stub grinding (Fig. 2.10), flexible assembly, and many other applications requiring controlled contact between the robot and its environment. In addition, force control concepts are needed in order to open up new application areas in non-standard environments such as in home, medical, and service robotics. Despite the number of potential applications, as well as several decades of research and successful laboratory experiments, successful industrial implementations of force control have previously been very few. Standard industrial robots have only recently been extended with functionality for high-bandwidth force control [Born and Bunsendal, 2001; Blomdell *et al.*, 2005].

The most important basic approaches to force control are hybrid control [Raibert and Craig, 1981] and impedance control [Hogan, 1985], although there exist methods which are not easily classified into either group. Surveys on methods and systems for robotic force control can be found in [De Schutter *et al.*, 1997; Chiaverini *et al.*, 1999; Yoshikawa, 2000]. In hybrid control, different directions are selected as either controlled by position control or force control, and the robot is controlled in order to simultaneously track both the desired position and the desired force. In impedance control, feedback is used for adjusting the apparent mechanical impedance of the robot tool with respect to external forces, such as forces arising from environment contact. Impedance control has the advantage of not requiring switching control laws with different behavior in

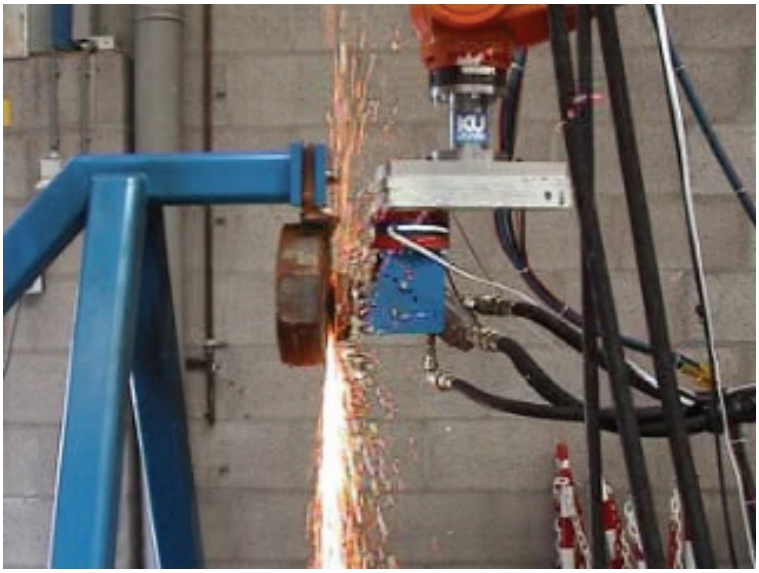

Figure 2.10 Force-controlled stub grinding using an ABB Irb 6400 industrial robot.

contact and non-contact, or accurate models of environment geometry or dynamics. The hybrid control framework on the other hand may be more suitable if accurate control of forces and/or positions is necessary. For both of the basic approaches, many applications require extensions in order to take robot dynamics more rigorously into account. Examples of such dynamical effects are the full rigid body dynamics of the manipulator, as well as the dynamics of the motors and amplifiers. Mechanical flexibility, as caused by bending of gear teeth or robot links, or by compliance in force sensors and robot mountings, is another important issue. In addition, mechanically compliant end-effectors are sometimes used in order soften the contacts and to decouple the robot dynamics from the interaction dynamics with the environment [De Schutter *et al.*, 1996]. Among the compliances in the robot system, particularly dynamically non-collocated modes such as transmission and link flexibility can cause serious performance and stability problems, and limit the achievable performance [Colgate and Hogan, 1989; Eppinger and Seering, 1992].

Within the literature on modeling and control of constrained systems, there is also a large class of methods that describe contact situations by geometric constraints, that is, constraints on the robot configuration. From a practical point of view, this would correspond to environments with

infinite stiffness. In such systems it is possible to eliminate a number of degrees of freedom, and instead work with a system of lower dimension. Most modeling methods for constrained robots take a Lagrangian approach, using d'Alembert's principle and Lagrangian multipliers to express the contact forces [Murray *et al.*, 1994; Spong and Vidyasagar, 1989].

Impedance Control Techniques

Assuming a rigid robot model of the form

$$\mathbf{M}(\mathbf{x})\ddot{\mathbf{x}} + \mathbf{N}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{u} + \mathbf{f}, \tag{2.16}$$

the classical dynamic impedance control can be used to give the robot a desired passive impedance

$$\mathbf{M}_I\ddot{\mathbf{x}} + \mathbf{D}_I\dot{\mathbf{x}}_e + \mathbf{K}_I\mathbf{x}_e = \mathbf{f}, \tag{2.17}$$

with positive definite $\mathbf{M}_I$, $\mathbf{D}_I$ and $\mathbf{K}_I$, and $\mathbf{x}_e = \mathbf{x} - \mathbf{x}_r$ the deviation from the position reference $\mathbf{x}_r$. This will cause the manipulator to respond to external forces $\mathbf{f}$ like a passive mass-spring-damper combination. The control law that achieves this behavior can be seen as an inner feedback linearizing control law, which makes the robot behave as a decoupled set of double integrators, combined with an outer impedance control law [Spong, 1989]. From a practical point of view, the basic impedance controller suffers from a number of problems. One problem is that a soft impedance also increases the effect of disturbances and modeling errors on the motion tracking [Siciliano and Villani, 1999]. Additionally, in industrial robots direct access to the motor torques is generally not available, and the rigid model (2.16) does not take important dynamic effects such as transmission compliance into account [Freund and Pesara, 1998; Ferretti *et al.*, 2004]. An alternative solution to the impedance control problem is to design an inner position control for the robot, and set this motion controller to track a reference position provided by the outer impedance controller. The reference position is given by an impedance relation such as Eq. (2.17). The inner position controller can either be chosen as the built-in motion controller, or a new motion controller could be designed for improved arm-side motion tracking and end-point stiffness, as described in Chapter 6. Ideally, the motion controller should make the response to motion commands as fast as possible, and simultaneously suppress the effects of external forces as quickly as possible, in order to make the robot behave as closely as possible to the desired impedance. In any real system, the range of achievable impedances is always limited by the controller bandwidth and stiffness of the robot, as well as by sensor noise and other disturbances.

Impedance control is a very general concept for control of interaction, and is not limited to any specific implementation [Won *et al.*, 1997].

Many different types of control structures, manipulators and forms of the impedance relation (2.17) can be used, such as the natural admittance control of [Dohring and Newman, 2003]. The common factor is that all impedance control methods attempt to control the dynamic behavior of the manipulator at the point of interaction with the environment, in particular the response of the tool motion to external forces. It is this dynamic behavior that determines the main properties of the interaction. For instance, a manipulator with linear dynamics is stable in contact with arbitrary linear passive environments if and only if its impedance is passive [Colgate and Hogan, 1988].

Direct Force Control and Hybrid Position/Force Control

In classical hybrid force/position control, different directions are designated for either position control or force control, and position and force are controlled simultaneously. This formulation requires a detailed description of the geometry of the workpiece. For a rigid robot, one way to implement the hybrid control is to design an inner feedback linearizing control as for the basic impedance control described above, giving a decoupled set of double integrators [Khatib, 1987]. For the outer loop, force control and position control laws are designed for each degree of freedom, and division of the workspace directions into force and position control is traditionally taken care of using special *selection* or *weighting* matrices [Spong, 1989; De Schutter *et al.*, 1997]. By using this inner/outer controller structure, significant freedom is provided in the design of the outer position and force control laws. Thanks to the linearization performed in the inner loop, the outer controller can in some cases be designed using linear methods. Similarly, by designing fast inner motion control loops, outer-loop force controllers can be designed based on decoupled linear models.

Another approach is the *parallel* force/position control approach, in which the desired position is modified by a *compliant* position obtained from the force controller, leading to a new reference position to be tracked by the inner motion controller [Siciliano and Villani, 1999]. In general, the force control is chosen to include integral action, in order to make the force control action dominate the position control. One advantage of the parallel formulation is that it provides some robustness to uncertainties, and the control can be designed using simplified dynamical and geometric models of the environment.

In the direct force control methods, the performance and force tracking properties, and even the stability, are strongly influenced by the properties of the environment, such as the stiffness. Improved force tracking can be achieved by using the stiffness as a known parameter in the force control design [Siciliano and Villani, 1999]. Since accurate models of the

environment are rarely available, the environment dynamics is frequently compensated for by inclusion of a variable gain, or other easily tunable parameters. In [Natale *et al.*, 2000], an automatic design procedure for force controllers was presented, using a robot model of first order with time delay and a rough estimate of the surface stiffness. Adaptive control algorithms, such as the method of [Roy and Whitcomb, 2002], can be used to find estimates of the stiffness and to adapt the control to unknown or time-varying environment properties.

In addition to the force control itself, special care may need to be taken in the transition between free motion and contact. Models of contact transition and impact, and studies of techniques for impact control, were discussed in [Wu *et al.*, 1995] using simple Hertzian models of impact, and in [Brogliato and Orhant, 1994] using models from distribution theory. Calibration of contact impedance parameters was investigated in [Diolaiti *et al.*, 2005], and the geometry, kinematics, and dynamics of contact in [Featherstone *et al.*, 1999].

Force Control of Flexible Manipulators

The development to make industrial robots lighter, faster and cheaper has increased the mechanical weakness of links and transmissions. For most industrial robots, the most important source of mechanical flexibility is the joint elasticity. The presence of joint elasticity introduces a non-collocation between the wrist-mounted force sensor and the actuation, which could seriously affect performance. In [Goldsmith *et al.*, 1999], it was shown that some common force control methods for rigid robots, such as the *resolved acceleration* or *operational space* formulations of hybrid control [Khatib, 1987], are destabilized by joint flexibility. In [Spong, 1989], a singular perturbation model for a flexible joint robot was given, and an inner control law which linearized the system restricted to an (attractive) integral manifold was derived. This makes it, at least theoretically, possible to design the outer control in the same way as in the rigid case.

For industrial robotics with built-in motion control, different techniques for handling flexibility must be employed. The force control problem for such controlled robots has received significantly less attention in the literature. The dynamic properties of a controlled robot are strongly affected by the dynamics and structure of the motion controllers. Approximations of the closed-loop transfer functions, and guidelines for selection of parameters for an impedance controller, were discussed in [Ferretti *et al.*, 2004], although the discussion was limited to certain classes of controllers and choices of parameters. In Chapter 6 a robot drilling application is presented, in which proper modeling of the joint flexibility was a crucial factor for the controller design.

2.5 Sensor Fusion and Combined Force/Vision Control

In many classes of control and estimation problems, improved performance can be achieved by combining information from several sensors. The combination can sometimes be performed using sensors of a similar type, such as in multi-camera systems, while in other cases the fusion is based on sensors measuring very different types of physical quantities. Sensor fusion, often referred to as *multi-sensor (data) fusion* or just *data fusion*, concerns the combination of data from several different sensors in such a way that the resulting information is "better" (in some sense) than the information which could be extracted from the individual sensors. In addition to sensor data, fusion with other types of data sources, such as databases and user input, may be included in the definition. The motivations for using sensor fusion are many, for instance to achieve redundancy and fault-tolerance of a sensory system, to cope with limited spatial coverage as in the multi-camera system in Chapter 4, to achieve higher temporal sampling rates [Schuurman and Capson, 2004], to obtain measurements at several different frequency ranges, or to handle imprecision in the sensor and uncertainty in the observed object. An introduction to the terminology and concepts of sensor fusion is given in [Elmenreich, 2001].

One classification of fusion methods is based on the type and configuration of the sensor [Durrant-Whyte, 1988; Elmenreich, 2001]. *Complementary* sensor configurations involve measurements of different, independent quantities. In this case, fusion can be performed by simply appending the data, in order to give a more complete picture of the observed phenomenon. The fusion of visual and contact force measurements would in most cases be considered to be complementary, since the sensors provide very different information on the current state of the system. *Competitive* sensors provide independent, redundant measurements of the same physical property. An example is given by multiple cameras which are all measuring the same degrees of freedom of the object motion. *Cooperative* sensors are combined in order to obtain information that could not be obtained from each individual sensor alone. Stereo cameras and multi-camera systems, which are able to obtain depth information from an unmodeled scene, are examples of this type of fusion. Obviously, the classification into complementary, competitive and cooperative sensors systems is not exact. In many systems aspects of more than one of the types can be found, such as in the multi-camera system in Chapter 4 which contains elements of both a competitive and a complementary nature.

Sensor fusion can be handled at several different levels, depending on the application in question. On the low/medium level fusion is performed on raw sensor data, while high-level or *decision fusion* involves decision

systems based on statistical methods or fuzzy logic. Many of the existing techniques for low-level fusion are related to the previously described state estimation techniques, such as Kalman filters, and sequential Monte Carlo methods. For competitive sensors, the fusion itself can be performed either directly on the raw data in a centralized fashion, or on the level of the estimated state vectors from individual estimators. The latter approach may be more suitable if a distributed implementation is desired. In addition to the estimation, separate preceding steps of data association and correlation are generally necessary, in order to group the available observations with respect to the different physical phenomena observed (e.g. feature matching in a real-time vision system).

Force/Vision Control

A very natural approach to manipulation, in both natural, home and industrial environments, is to combine force sensing and control with high-level guidance from a vision system. The potential advantage is that the limited accuracy and slow speed of the vision system is complemented with accurate force data upon contact with the environment. The desire to combine force and vision leads to approaches based on feedback from cameras and force sensors. The majority of research work presented on the combination of vision and force control has used some variation of hybrid force/position control or impedance control. In most cases the focus of this research has been on the vision systems, and how its geometry and kinematics influences the properties of the feedback. The treatment of the force control and dynamics problems are often less well developed, with purely kinematic or simple rigid robot models often used in the design and analysis. Using these standard control structures, many results and analysis tools from visual servoing and force control can be used and adapted also to the case of force/vision control.

In [Nelson *et al.*, 1995] three different basic strategies or control structures were presented, referred to as *hybrid* control, *traded* control and *shared* control. An obvious extension of standard hybrid force/position control, hybrid force/vision control divides the workspace into orthogonal force- and vision-controlled directions. In traded control, each degree of freedom is controlled by both force- and visual control, with switching between the sensors based on the sensor signals. Finally, in shared control both sensors are used simultaneously in each degree of freedom. In this case, the stability analysis must take coupling effects between the force control and vision control actions into account.

In the last decade, a large number of different approaches to combined force/vision control have been presented in the literature. The method presented in [Baeten *et al.*, 1999] used Mason's task frame and a high level task description to determine how to use each sensor in a structure

similar to hybrid force/vision control. A hybrid and adaptive vision/force control technique was developed in [Pichler and Jägersand, 2000]. The paper of [Zhao *et al.*, 2006] presented dynamic image-based control for a constrained rigid robot actuated directly through the motor torques, and Lyapunov stability proofs and conditions for the feedback gains were given. [Hosoda *et al.*, 1996] presented a shared adaptive technique, where the image Jacobian and the slope of the constraint surface were estimated online from sensor data. The method in [Pomares and Torres, 2005] used a weighted shared strategy for fusing visual- and force information. The weighting factors were determined based on an estimate of the current tracking performance, obtained from a special filter for surface change detection. The force controller part was a proportional controller with inner velocity control, while the visual controller was an image-based controller based on what was termed a *movement flow* vector field for improved trajectory following. In [Dean-Leon *et al.*, 2005] hybrid vision/force control for a constrained rigid manipulator model was described, together with dynamic friction compensation based on the *LuGre* model [de Wit *et al.*, 1995]. Experimental results with a hybrid technique which takes the robot dynamics into account were presented in [Xiao *et al.*, 2000]. In [Leite *et al.*, 2006], another hybrid technique was developed, where the force control was used to reorient the robot end-effector based on direction of the the measured normal force. The visual servoing law was an image-based resolved-rate control law, with adaptation of the uncertain parameters of the image Jacobian for a planar two-link robot. Hybrid position/force control techniques were also used for planar contour following in [Chang, 2004], where the robot dynamics was assumed to be given by a simple kinematic model with perfect velocity control. Similar assumptions on the robot dynamics were made in [Morel *et al.*, 1998; Malis *et al.*, 2001], where the use of visual servoing together with purely damping impedance control was proposed and demonstrated in a peg-in-the-hole insertion task. The target impedance was given by a pure damping, and it was shown that under these assumptions a separation property holds, so that overall stability is guaranteed if the force- and impedance control systems are stable separately. A similar technique using internal motion impedance control was suggested in [Carelli *et al.*, 2004], and used together with the concept of "fictitious forces" in obstacle-avoidance tasks. In addition, control structures for hybrid position/force control using vision were presented. In [von Collani *et al.*, 2000], a neuro-fuzzy force/vision control approach based on eigenspace methods was used for aligning and fastening a screw into a nut using collaborating robot arms. In [Lippiello *et al.*, 2006], an example of a position-based visual impedance control structure with inner motion control was presented. The EKF-based pose estimation algorithm was based on measurements of image features, joint positions and contact

forces, and used in an experiment with 6-DoF impedance control.

A number of application-oriented articles on force/vision control have been published in recent years. In [Zhou *et al.*, 1998] an application of image-based vision/force control in micro-manipulation was demonstrated. A traded control law was used, which switched between proportional force control and an image-based visual servoing control law, which was based on optimal control techniques. In [Ferreira *et al.*, 2004], another switching force/vision controller for an automatic micro-assembly system was presented. An application of position-based force and vision control in flexible assembly was presented in [Jörg *et al.*, 2000], with a demonstration of mating of moving parts. The pose estimation was based on line features extracted with the Hough Transform [Trucco and Verri, 1998], using a nonlinear Kalman filter for estimation of the circular motion of the moving target. [Krupa *et al.*, 2004] presented a system for hybrid force- and visual feedback, to be used in laparoscopy. Traditional image-based visual servoing was used together with proportional force control, in order to track and guide a surgical instrument. The combination of force/vision control is also a very natural approach for guidance of grasping systems [Hashimoto *et al.*, 2001].

3

Experimental Industrial Robot System

3.1 Introduction

In general, industrial robots are designed to handle repetitive tasks in well-known, well-structured environments. There are many types of tasks however, in which time-consuming calibration procedures or large workpiece variations make it difficult for standard industrial robots to operate efficiently. In particular, this is the case in tasks where well-defined and accurate contact with the workpiece is required, as described in Chapter 2. In addition to well-established force control applications such as assembly, applications such as stub grinding (see Fig. 3.1), fettling[1], and deburring represent examples of such tasks, which would be very relevant to automate. Not only are manual fettling and finishing major cost elements in the production process (representing up to 40% of total costs) which often lead to inconsistency in quality and delivery delays, there are also severe health aspects related to this process in the foundry industry.

For automation of these tasks using industrial robots, active force sensing and feedback gives the possibility to accurately control the interaction between the robot and the workpiece. Unfortunately, even relatively modern robot control systems provide no interfaces for external sensor feedback with sufficient performance. In order for an industrial robot to be able to satisfy the requirements of processes such as stub grinding and deburring, functionality for high-bandwidth contact force control needs to

[1] *Fettling* is the process of removing sand that is adhering to castings, and is traditionally performed manually by hammering or blast finishing. It is also used to refer to the process of removing excess material from the edges of a casting using fettling tools.

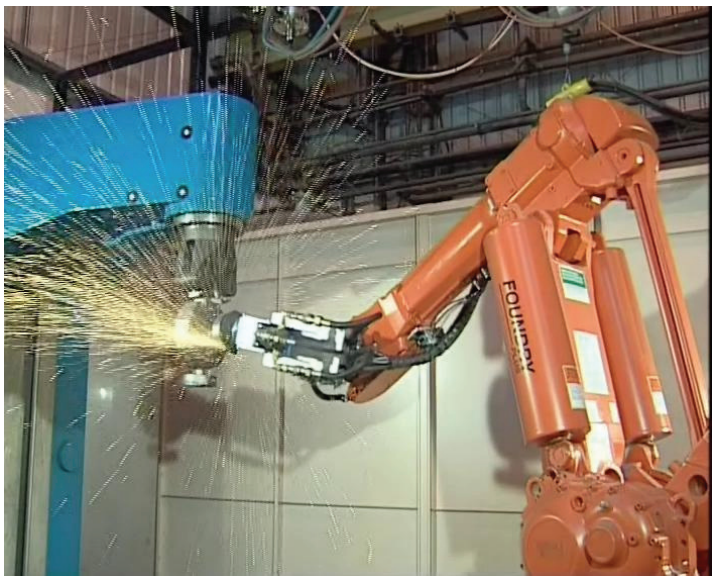

Figure 3.1 Force-controlled stub grinding using an ABB Irb6400 industrial robot with an extended ABB S4CPlus control system.

be implemented in current industrial robot systems. Robotic force control can be carried out on two levels:

- Forces are controlled by a separate external tool, meaning that the adaptation of the force occurs without intervention of the robot. The robot executes a pre-programmed path, and the changes in the tool position to achieve a constant force are carried out by the end-effector integrated with some additional external axis.
- Forces are controlled by using the robot motion, corresponding to the solution described in this chapter.

The advantages of the robot with integrated force control include the possibility of obtaining higher stiffness at a considerably lower weight, as well as increased flexibility in mounting and accessibility due to the six degrees of freedom available. The implementation of such flexible motion control using external sensors is particularly difficult, since the user or system integrator would need to influence the core real-time software functions that are critical for the performance and safe operation of the system. Therefore, techniques that permit real-time motion controllers to be extended for new demanding application areas need to be developed. A crucial issue is the achievable bandwidth for the external sensor feedback.

In many applications, the effects of the bandwidth limitations are evident in the form of longer duty cycles, whereas for some applications—for instance when contact force control is used—stability problems and severe performance degradation may result.

An examination of five major European robot brands (ABB, Comau, Kuka, Reis, Stäubli) shows that they all, to some extent, provide support for application-specific motion control. Some are fully open systems, but only if all original safety and programming features are disabled. The ABB S4CPlus used in this work is not an open system on the level of built-in motion control, but its internal design provides some features for development of open control. Systems and results similar to the system described in this chapter have been reported also for other systems [Born and Bunsendal, 2001].

In this chapter, the extension of an industrial robot system with functionality for control using external sensors is described, using the ABB S4CPlus controller as an example. The dynamic properties of the resulting robot system are described and analyzed. Results from experiments, where the designed interface was used for force-controlled grinding and deburring, are presented.

3.2 Structure of S4CPlus Extensions

In this section, a brief description of the structure of the extended ABB S4CPlus control system is given. A more detailed description of the complete system, together with a discussion of other issues considered in the design and implementation, is given in [Blomdell *et al.*, 2005].

The structure of the extended ABB S4CPlus control system is illustrated in Fig. 3.2. In standard S4CPlus, the high-level task description is first converted into program code written in the robot programming language RAPID. From the instructions in the RAPID program, the trajectories are generated for the internal arm-level and motor controllers responsible for the low-level motion control of each joint. Although the standard S4CPlus controller does not permit low-level external sensor feedback, on a high level there is a possibility to read sensors via customer I/O, and to influence the robot task according to instructions written in RAPID. However, since the sampling frequency is limited to around 10 Hz, and since significant time delays are introduced by the trajectory generation step, the achievable bandwidth on the RAPID level is far too low for most applications of force control.

In order to obtain a sufficient sampling bandwidth, the extensions were instead implemented by modifying the references on the arm control level with a 4 ms sampling interval, which gave a suitable trade-off between

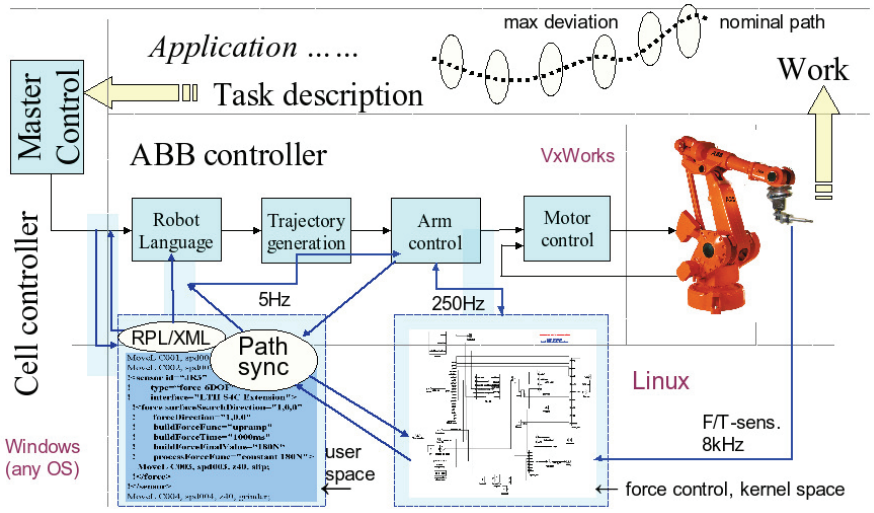

Figure 3.2 Extension of industrial robot controller with sensor interface and support for external computations and synchronization, using a Motorola PPC-G4 PrPMC-800 processor board mounted on a Alpha-Data PMC-to-PCI carrier board with a local PCI bus.

engineering effort, preserved safety, and performance. In order to accomplish interrupt-driven hard real-time execution, the force controller was run as a Linux kernel module on an additional external Motorola PPC-G4 processor board mounted on a PMC-to-PCI carrier board with a local PCI bus. In each sample, the references and parameters necessary for the external control were copied from the S4CPlus to the external controller board over the PCI bus, using a shared memory interface between the the built-in motion control and the external controller. The external controller modified the references for position, velocity and torque according to sensor data and the active control algorithm, and values were copied back to the S4CPlus system where the built-in safety functions performed checking of the updated references. The force controllers—represented by the force control block in Fig. 3.2—were implemented as block diagrams in Matlab/Simulink, and converted into C code using Real-Time Workshop. The code was cross-compiled to the target computer and finally linked to form the Linux kernel module.

On the user level, the RAPID program was extended with instructions for the external control such as references, parameters, activation and deactivation commands for the force controllers. The external instructions were encoded as XML-style tags and added as comments to create the ex-

```
MoveL C001, spd001, z40, grinder;
MoveL C002, spd002, z40, grinder;
!<sensor id="optidrive"
!        type="force"
!        interface="LTH+ABB S4C Extension">
  !<force surfaceSearchDirection="1,0,0"
  !       forceDirection="1,0,0"
  !       buildForceFunc="upramp"
  !       buildForceTime="1000ms"
  !       buildForceFinalValue="150N"
  !       processForceFunc="constant 150N">
  MoveL C003, spd003, z40, grinder;
  !</force>
!</sensor>
MoveL C004, spd004, z40, grinder;
```

Figure 3.3 An example of an ExtRAPID program for a simple 1-DoF force control task, the task consisting of establishing contact with a surface located in a specified direction (relative to the tool frame). The extended language constructs are located in RAPID comments and are modeled as XML tags in order to be easily modifiable. In addition, the nominal program can be executed in its standard form by the ordinary S4CPlus system.

tended code in a defined format called ExtRAPID, see Fig. 3.3. The part of the RAPID program without the extra instructions was then sent to and interpreted by the S4CPlus system. The extended instructions were processed by a Master PC, responsible for communication and for synchronizing the execution of the external controller with the S4CPlus system.

Preserved Safety for External Sensor Feedback. Open systems require careful engineering not to exhibit unpredictable or even unsafe behavior when extended with novel features at the customer site, and confronted with potentially inexperienced users. Installing third-party hardware means that there is an additional risk for system failures, despite high and ensured quality of the basic robot system. Importantly, sensor failures are inevitable and have since long been an important obstacle in applications. In cost-efficient production, solutions based on redundant sensors are often unacceptable because of the extra cost and the increased risk for system overload/failure. Instead, a combination of system structure, proper interface design, testing methodology, and well-defined fallback control is needed. Perhaps the most important part of safety is the ability to keep the internal safety functions activated (possibly which adjusted tolerances) even during sensor-based motions. Although this prob-

lem is possible to solve, the difficult part is to combine safety with performance. In the design presented in this work, modifying the references for the joint- or arm-level control causes the modified sensor-based references to be subject to the same safety logic as ordinary trajectories, thereby retaining a large part of the safety of the original system.

3.3 Programming, Control, and Modeling Issues

As mentioned previously, the fact that the extended control interface system is designed as an extension to the standard S4CPlus robot system makes it possible to reuse a large part of the user-level functionality. An example is the programming of the nominal motion component of a given task, which is still expressed in the standard RAPID language, the instructions of which are executed by the standard robot system. This facilitates the use of available CAD and off-line programming tools for task specification and programming. As the methodology for task specification and programming are not the main focus of this work, these aspects are not covered in detail here. For a more detailed discussion on these and related topics, see [Johansson *et al.*, 2004; Blomdell *et al.*, 2005; Robertsson *et al.*, 2006].

This reliance on the built-in robot functionality also puts a number of constraints on the usage of the system. Most importantly, the above mentioned standard task specification methodology results in sensor-based control actions which are modeled as corrections to the nominal, programmed path or trajectory[2]. The parameters determining the nature of the sensor-based corrections, as given in the extensions in the ExtRAPID language, describe a number of important issues:

Task-DoF selection. The specification for multi-DoF sensor-based correction, such as in 6-DoF force control, require careful consideration of several aspects related to the geometry of the task [De Schutter and Van Brussel, 1988a; De Schutter *et al.*, 1997]. In the specification, a task-related coordinate system is defined implicitly by specifying the directions in which force/torque control is to be applied, with standard motion control assumed in the remaining directions. In each direction specified, the velocity (or angular velocity) is allowed to be controlled using external sensor feedback. The directions, which are defined relative to the tool frame, may be updated online according to some model-based description. In this way it is possible

[2]It should be noted that, for experimental purposes, the corrections are not restricted by the system to be small, providing the possibility for unrestricted control of the robot trajectories in for instance a visual servoing system.

to handle complex scenarios where the orientation of the tool and the sensor-controlled directions vary independently. One example is force-controlled motion around a corner, where the desired force may be applied perpendicularly to the surface, without reorientation of the tool. This way of specifying the task geometry can be compared to the well-known *task frame* formalism [Mason, 1981; Baeten *et al.*, 1999].

Controller parameters. Although parameters for general sensor-based control could be included in the ExtRAPID specification, currently only parameters for force control have been implemented. On the ExtRAPID level, the desired behavior of the force controllers described in Section 3.4 is specified as the impedance parameters (mass/inertia, stiffness, and damping) in each force-controlled direction.

Reference signals. This specification include all desired sensor values, as well as their desired variations as a function of the coordinates along the path. Currently, only force references which are affine functions of the path coordinates are specified in the ExtRAPID language, making it possible to change and ramp up the force in a controlled manner.

Termination criteria. The criteria for termination of a phase are specified as upper/lower thresholds on a sensor value, or as a maximum desired time allowed for the phase. This allows to handle smooth transitions from a phase to another, as well as error handling (if, for instance, a surface is not found within a given time by a search in the specified direction).

CAD tools with partial support for automatic generation of the ExtRAPID-level specifications have also been developed [Robertsson *et al.*, 2006]. Such CAD tools provide support on the user level for the required geometrical specifications. Potentially, such tools could also be used to simplify the difficult problem of controller parameter selection by translation from more task-oriented specifications, for instance as in [Natale *et al.*, 2000].

Dynamic Modeling

For the purposes of simulation and model-based control design, a dynamic model of the system responses to external forces and motion references is required. In order to illustrate the expected properties of such a model, a local model is assumed in the form

$$\mathbf{M}_a\ddot{\mathbf{p}}_a + \mathbf{D}_1\dot{\mathbf{p}}_a + \mathbf{K}\mathbf{p}_a = \mathbf{K}\mathbf{p}_m + \mathbf{D}_2\dot{\mathbf{p}}_m + \mathbf{f}_e \tag{3.1}$$

$$\mathbf{M}_m\ddot{\mathbf{p}}_m + \mathbf{D}_3\dot{\mathbf{p}}_m + \mathbf{K}\mathbf{p}_m = \mathbf{K}\mathbf{p}_a + \mathbf{D}_4\dot{\mathbf{p}}_a + \mathbf{f}_c \tag{3.2}$$

where $\mathbf{p}_m$ and $\mathbf{p}_a$ are the motor and arm side positions in local Cartesian coordinates, and $\mathbf{f}_c$ and $\mathbf{f}_e$ are the (transformed) control torque and external force on the tool, respectively. $\mathbf{M}_a$ and $\mathbf{M}_m$ represent the arm and motor inertias, $\mathbf{D}_i$ are damping matrices, and $\mathbf{K}$ represent elasticity in the transmission and links. Together with a feedback/feedforward type motion controller

$$\mathbf{f}_c = -\mathbf{f}_{\text{fb}}(\mathbf{p}_m, \dot{\mathbf{p}}_m) + \mathbf{f}_{\text{ff}}(\mathbf{p}_r, \dot{\mathbf{p}}_r) + \mathbf{f}_I \left(\int \mathbf{p}_r - \mathbf{p}_m dt \right) \tag{3.3}$$

this leads to a full model of the motion-controlled robot. The position and velocity signals $\mathbf{p}_r$ and $\dot{\mathbf{p}}_r$, corresponding to the references for the inner motion controllers, are the control inputs to be used by the external control. In the presence of a constant external disturbance force $\mathbf{f}_e = \mathbf{f}_0$ and zero reference $\mathbf{p}_r = 0$, in stationarity we obtain the equilibrium

$$\mathbf{p}_m = \mathbf{0} \tag{3.4}$$

$$\mathbf{p}_a = \mathbf{K}^{-1}\mathbf{f}_0. \tag{3.5}$$

As the Cartesian stiffness matrix $\mathbf{K}$ will in general not be diagonal, this means that the deflection in the tool position will not be in the direction of the external force. An example of the consequences of this effect is given by a drilling system, in which motion tangential to the surface will result when axial cutting forces are applied to the drill. This would cause undesired sliding of the drill tip on the surface, which needs to be controlled as described in Chapter 6.

Model tuning and identification. The model in Eqs. (3.1)–(3.3) was the basis for a tuned model used for control design. By exploiting the special structures of Eqs. (3.1)–(3.3) and a typical robot motion controller, models that captured the behavior of the controlled robot were obtained from experimental data. A structure of the form

$$\sum_{j=0}^{n_a} \bar{\mathbf{A}}_j \mathbf{p}_a(k–j) = \bar{\mathbf{B}}\mathbf{p}_m(k–1) + \bar{\mathbf{B}}\mathbf{K}^{-1}\mathbf{f}_e(k–1) \tag{3.6}$$

$$\sum_{j=0}^{n_m} \bar{\mathbf{F}}_j \mathbf{p}_m(k–j) = \sum_{i=1}^{z_m+1} \bar{\mathbf{G}}_i \mathbf{p}_a(k–i) + \bar{\mathbf{H}}\mathbf{p}_r(k–1) \tag{3.7}$$

was assumed for the tuned dynamic models. In order to obtain better reliability, the tuning procedure was divided into a static and a dynamic step. In the first stage, an algorithm for static calibration was used to

find the stiffness matrix $\mathbf{K}$. The algorithm was based on the application of known forces $\mathbf{f}_e$ against a surface and measurements of the resulting arm-side deformations, followed by a least-squares solution for the stiffness matrix inverse $\mathbf{K}^{-1}$. In the second stage of the tuning, a dynamic model including the motion-controlled rigid robot dynamics, resonances, and dynamic couplings was tuned, given motion references $\mathbf{p}_r$ as inputs. A pseudo-random binary excitation signal was sent as a motion reference to the unconstrained robot, and both motor and arm side positions were measured. The small arm side motions were measured in 5 degrees of freedom, using two high-speed cameras. A discrete-time input-output model of the arm side motion was found by minimizing the quadratic criterion

$$J(\bar{\mathbf{A}},\bar{\mathbf{B}},\bar{\mathbf{F}},\bar{\mathbf{G}},\bar{\mathbf{H}}) = \sum_{k=n_a}^{N} \left[\sum_{j=0}^{n_a} \left(\bar{\mathbf{A}}_j \mathbf{p}_a(k\text{–}j)\right) - \bar{\mathbf{B}}\mathbf{p}_m(k\text{–}1) + \right.$$
$$\left. + \sum_{j=0}^{n_m} \left(\bar{\mathbf{F}}_j \mathbf{p}_m(k\text{–}j)\right) - \sum_{i=1}^{z_m+1} \left(\bar{\mathbf{G}}_i \mathbf{p}_a(k\text{–}i)\right) - \bar{\mathbf{H}}\mathbf{p}_r(k\text{–}1) \right]^2 \tag{3.8}$$

subject to the constraints

$$\bar{\mathbf{B}} = \sum_{j=0}^{n_a} \bar{\mathbf{A}}_j, \quad \sum_{j=0}^{n_m} \bar{\mathbf{F}}_j = \bar{\mathbf{H}}, \quad \sum_{j=1}^{z_m+1} \bar{\mathbf{G}}_j = \mathbf{0} \tag{3.9}$$

stating the physical property that the system was statically decoupled by the integral action, giving a system with unit static gain in each degree of freedom. For validation of the obtained model, the true reference step responses were compared to the step responses predicted by the obtained model. The resulting reference step responses for a 3-degree-of-freedom model of translation only can be seen in Fig. 3.4, and the true and simulated arm side responses to an external force $\mathbf{f}_e$ can be seen in Fig. 3.5. A good fit was obtained both statically and around the resonance frequencies.

3.4 Case Studies and Experiments

As a case study, the external control system was used in a stub grinding application. Additionally, the system and software tools could be used for other applications, such as deburring and polishing.

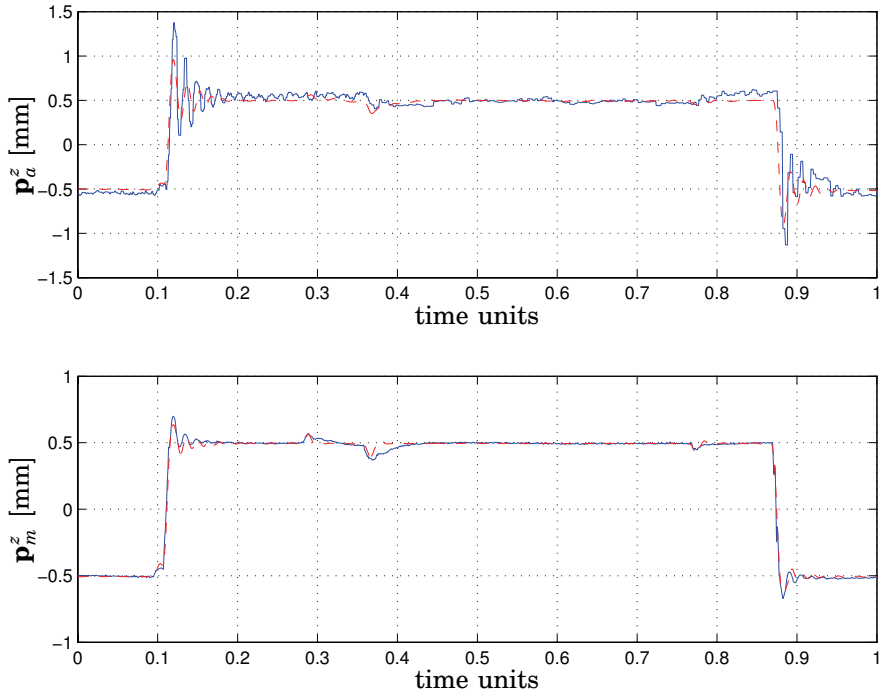

Figure 3.4 True arm side (top) and motor side (bottom) reference step responses (solid lines) in the z-direction, and the corresponding step responses of the model obtained by the tuning procedure (dashed lines). A modified and aggressively tuned velocity feedforward, in combination with the increased flexibility represented by the drilling tool, gave a faster and more resonant response to motion references than the built-in controllers.

Basic Force Control

In the applications considered in this chapter, only the motion perpendicular to the surface of the workpiece was required to be force-controlled, and therefore a force/position control strategy was employed. Since the built-in inner motion control in the robot control system was already available, this type of control corresponded to a suitable update of the motion references in the force-controlled directions. This reference update was performed by the force controller, which used a dynamical relation between position and external force to determine the motion in the force-controlled directions. The geometry of the problem is illustrated in Fig. 3.6. The nominal path was specified in the standard RAPID program, as the desired pose $\mathbf{T}_{\text{nom}}$ of the TCP-frame relative to some fixed spatial coordinate system. The figure illustrates the case where the orientation of the tool is fixed along

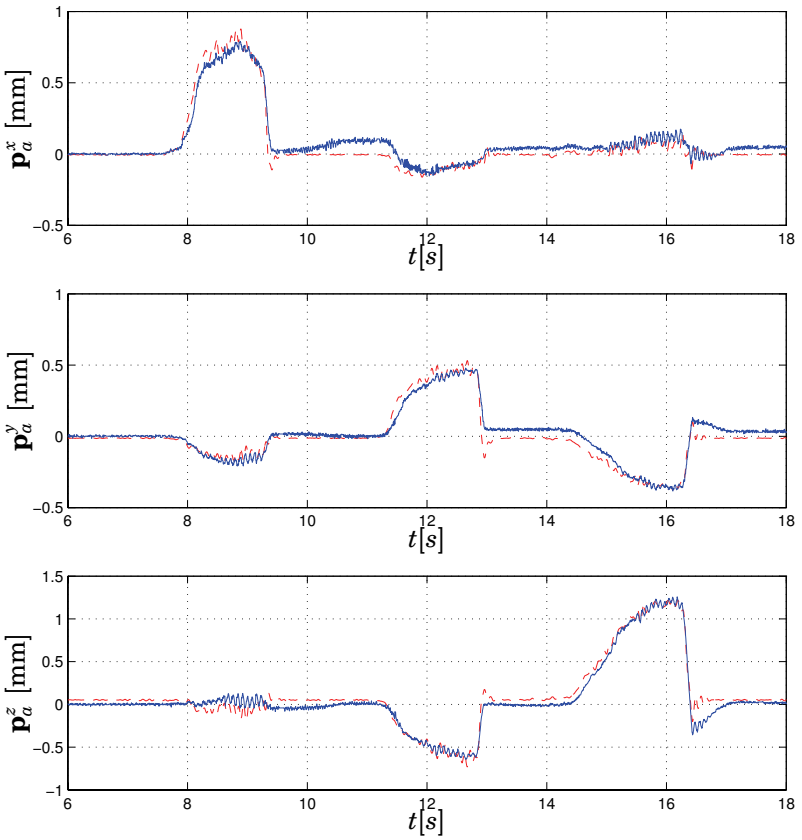

Figure 3.5 True arm side external force responses (solid lines) in the x-, y- and z-directions, and the corresponding responses of the model obtained by the tuning procedure (dashed lines).

the trajectory, although in the general case the TCP-trajectory included both the translation and the orientation of the tool. Additionally, a task description frame $\mathbf{T}_\mathrm{d}$ was specified in the extended RAPID instructions[3], together with information on which coordinate directions of $\mathbf{T}_\mathrm{d}$ were to be force-controlled. By continuously rotating frame $\mathbf{T}_\mathrm{d}$ complex behaviors could be achieved, such as force control along a curved surface without physically reorienting the tool (see Fig. 3.6).

The compliant frame $\mathbf{T}_\mathrm{c}$, defining the position to be tracked by the motion control, was defined relative to $\mathbf{T}_\mathrm{nom}$. The velocity $\mathbf{v}_c$ and angular

[3]Since the origin of frame $\mathbf{T}_\mathrm{d}$ was arbitrary, only the orientation of the frame (or just the force-controlled directions) needed to be specified in the ExtRAPID program.

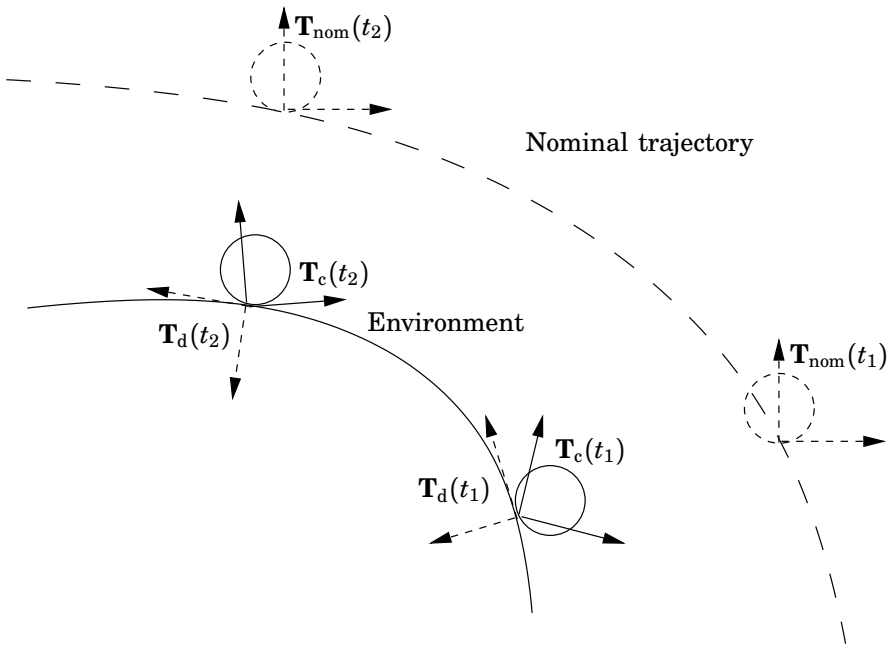

Figure 3.6 Geometrical description of the functionality of the basic force controller. The nominal path of the TCP-frame $\mathbf{T}_{\text{nom}}$ was specified as usual in the standard RAPID program. The orientation of the task description frame $\mathbf{T}_{\text{d}}$ was specified relative to the TCP frame in the extended RAPID instructions, together with a selection of which of its coordinate directions were to be force-controlled. The compliant frame $\mathbf{T}_{\text{c}}$, defined relative to $\mathbf{T}_{\text{nom}}$, was computed through the specified controller dynamics, given the measured contact force as input.

velocity $\boldsymbol{\omega}_c$ of frame $\mathbf{T}_{\text{c}}$, expressed in $\mathbf{T}_{\text{d}}$-coordinates, were obtained from the specified controller dynamics. This dynamic relation was described by rotational and translational motion relations

$$\mathbf{M}_{I_{\text{trans}}} \frac{d\mathbf{v}_c}{dt} + \mathbf{D}_{I_{\text{trans}}} \mathbf{v}_c = \mathbf{S}_{\text{trans}}(\mathbf{f} - \mathbf{f}_r) - \alpha(\mathbf{I} - \mathbf{S}_{\text{trans}})\mathbf{t}_c \tag{3.10}$$

$$\mathbf{M}_{I_{\text{rot}}} \frac{d\boldsymbol{\omega}_c}{dt} + \mathbf{D}_{I_{\text{rot}}} \boldsymbol{\omega}_c = \mathbf{S}_{\text{rot}}(\boldsymbol{\tau} - \boldsymbol{\tau}_r) - \alpha(\mathbf{I} - \mathbf{S}_{\text{rot}})\mathbf{r}_c \tag{3.11}$$

where $\mathbf{f}$ and $\boldsymbol{\tau}$ were the measured force and torque vectors acting at the TCP point, expressed in $\mathbf{T}_{\text{d}}$-coordinates. The vectors $\mathbf{f}_r$ and $\boldsymbol{\tau}_r$ were the corresponding reference force/torque vectors. The matrices $\mathbf{M}_I$ and $\mathbf{D}_I$ of desired inertia and damping were chosen to be diagonal. By setting the diagonal elements of the selection matrices $\mathbf{S}_{\text{trans}}$ and $\mathbf{S}_{\text{rot}}$ to one, the corresponding degrees of freedom became force-controlled. The last terms

of (3.10) and (3.11), with parameter $\alpha > 0$, were used for dynamically resetting the commanded motion in purely position-controlled directions to the nominal trajectory. This was needed after a change in the selection matrices or when rotating $\mathbf{T}_\mathrm{d}$, or otherwise positioning errors would occur in the motion-controlled directions. The vectors $\mathbf{t}_c$ and $\mathbf{r}_c$ were the translation and angle/axis representations of the relative pose $\mathbf{T}_\mathrm{c}$, also expressed in $\mathbf{T}_\mathrm{d}$-coordinates. At the beginning of each sample, the contact force and torque were read from the force/torque sensor, and compensated for the effects of gravity. The TCP-relative position and velocity of the compliant frame were obtained from (3.10)–(3.11). The updated joint servo position/velocity references were computed from the inverse kinematics function $\mathbf{f}_k^{-1}$ and manipulator Jacobian $\mathbf{J}(\mathbf{q})$, according to

$$\dot{\mathbf{q}}_r = \mathbf{J}^{-1}(\mathbf{q}) \left(\begin{bmatrix} \mathbf{R}_\mathrm{cd}\mathbf{v}_c \\ \mathbf{R}_\mathrm{cd}\boldsymbol{\omega}_c \end{bmatrix} \right) + \dot{\mathbf{q}}_\mathrm{nom} \tag{3.12}$$

$$\mathbf{q}_r = \mathbf{f}_k^{-1}(\mathbf{T}_\mathrm{nom}\mathbf{T}_\mathrm{c}), \tag{3.13}$$

where $\dot{\mathbf{q}}_\mathrm{nom}$ was the vector of joint velocities corresponding to the nominal trajectory, and the rotation matrix $\mathbf{R}_\mathrm{cd}$ described the current orientation of frame $\mathbf{T}_\mathrm{d}$ relative to $\mathbf{T}_\mathrm{c}$. In practice, the inverse kinematics $\mathbf{f}_k^{-1}$ was computed using a linearization around the nominal trajectory, since the deviations from this trajectory were assumed to be small.

Force-Controlled Stub Grinding

Final stub grinding experiments were performed in a specially developed work cell. The cell consisted of the robot with end-effector, a flexible clamping unit for the workpieces, the control system, and the Master PC. For the stub grinding process, a hydraulically driven end-effector developed within the AUTOFETT project was used [Robertsson *et al.*, 2006]. The main components of the end-effector were the hydraulic motor including the grinding stone (Fig. 3.7), and the force sensor consisting of a spring with a displacement sensor. The stiffness of the spring was set to 60 N/mm, the stroke being 20 mm. The natural system compliance was sufficient to react on irregularities of the casting product and surface roughness, which pass at high frequency. Bigger burrs, calibration errors and slow changes of the contour were handled by the force control system. An scheme based on CAD/CAM programming was used to generate the nominal tool trajectories, since the small series and complex geometry of the castings made traditional teach-in programming infeasible.

Images from the stub-grinding experiments, as well as finishing grinding experiments on a propeller blade, are shown in Fig. 3.9. The resulting contact force from a typical stub grinding experiment with reference

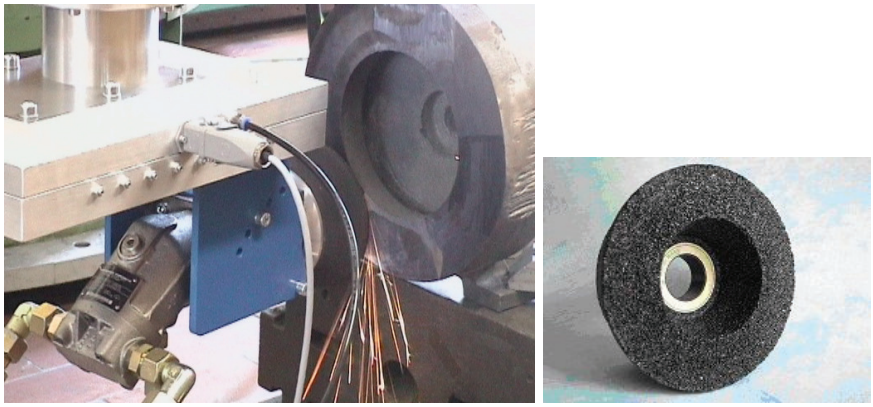

Figure 3.7 The compliant grinding tool (left) and cup stone (right) for the stub grinding application.

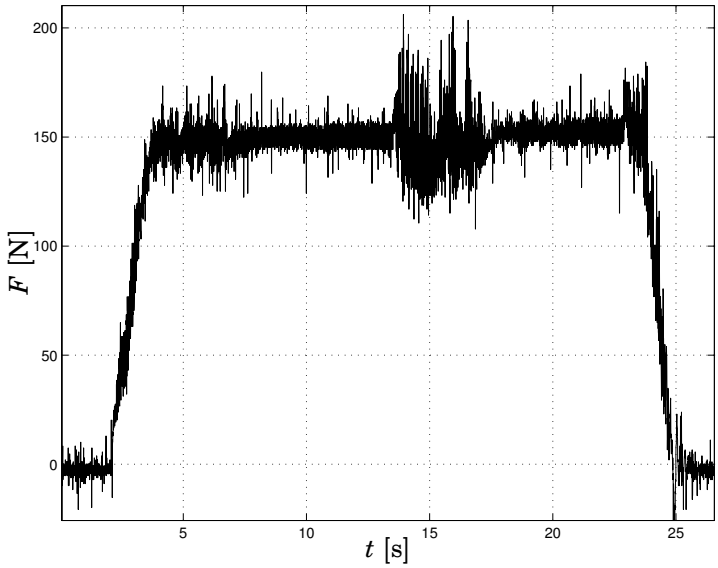

Figure 3.8 Contact force during grinding experiment with reference 150 N.

$F_r = 150$ N is shown in Fig. 3.8. The force remained close to the desired value during the entire operation, despite surface irregularities.

Force-Controlled Deburring

Deburring of workpieces with complex geometry, as shown in Fig. 3.10, is an example of the advantage of using six-degree-of-freedom force control.

Figure 3.9 Stub grinding (*left*) and finishing on a propeller blade (*right*).

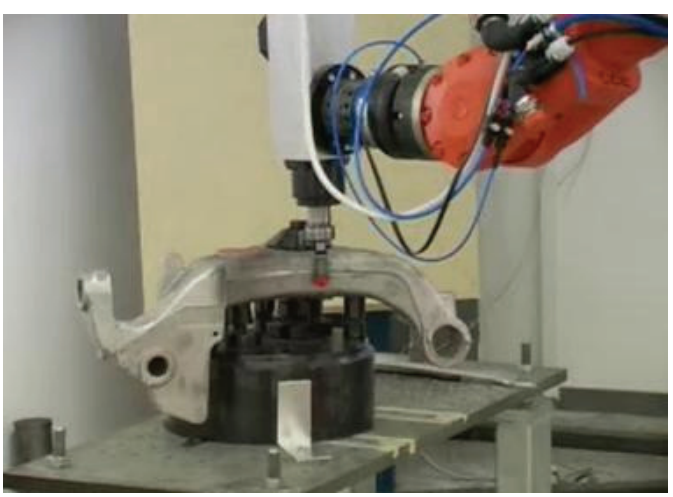

Figure 3.10 Removal of burrs for casted aluminum (car suspension) with milling motion around corner without tool reorientation. A stiff milling tool was used together with a wrist mounted (stiff) force/torque sensor (JR3) for programmable 6-axis compliance/force-controlled motion.

By programming a dynamically changing compliance in the different directions during the deburring process, extra flexibility was achieved in the trajectory generation process. One example is allowing stable contact to be maintained while moving across corners of the workpiece, without reorienting the tool. Another benefit is the possibility to do a rapid (rough)

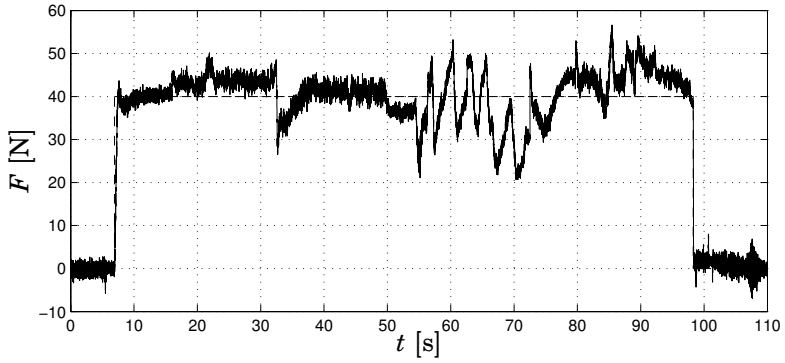

Figure 3.11 Contact force and during deburring experiment, with force reference 40 N.

teach-in procedure for a nominal trajectory along the surface. As long as the true profile of the workpiece is within the safety-zone of the programmed trajectory, and the bandwidth of the force control is sufficient, controlled contact will be maintained.

The experiments were performed using an Irb 2400 industrial robot equipped with a stiff 6-DoF JR3 force/torque sensor. Fig. 3.11 shows the achieved contact force during deburring of the workpiece in Fig. 3.10. An approximate trajectory was programmed along the burrs on the surface of the workpiece, and the force control was programmed to maintain a desired contact force of 40 N in the normal direction of the surface. In order to follow the trajectory, both the tool orientation and the force-controlled direction had to be changed abruptly at certain points along the trajectory. The re-orientations, together with the high stiffness of the contact and the roughness of the surface on which the milling tool was rolling, caused the variations in the measured contact force. Nevertheless, the bearing roller of the milling tool maintained contact with the surface during the whole constrained motion along the surface, and the burrs were successfully removed from the casting.

3.5 Summary and Concluding Remarks

In this chapter an interface for external control has been described, designed by extending an ABB S4CPlus industrial robot control system. The system extensions make it possible to modify the references of the internal motion control at a 4 ms sampling rate. To our knowledge, there are no other systems that provide the same high sampling rate and low

input-output latency together with all of the user-programming features, preserved safety, and supervisory functions. Results from experiments, where force- and impedance control were used for force-controlled grinding and deburring, were presented. Suitable dynamic model structures and tuning procedures for the extended robot system have been proposed. This will be used for designing model-based controllers for the robotic drilling system described in Chapter 6.

4

Feature-Based Visual Tracking and Force/Vision Control

4.1 Introduction

The majority of visual tracking algorithms use a separate feature extraction step, where the image data is compressed into a smaller number of image features. Image features are either structures with large intensity gradients—such as corners, lines, or edges—or image areas that can be distinguished by a similar color, intensity, or texture. The image features are characterized by some feature-dependent geometric properties, such as their image position, direction, or size. Following the feature extraction step, the motion is estimated from the extracted feature data. In the following, we will refer to such methods as *feature-based*[1], as opposed to the direct *intensity-based* methods which will be the topic of Chapter 5. From a perspective of feedback, feature-based methods make it possible to avoid the complex nonlinear relationship between motion and changes in image intensities. Instead, state-space modeling and control techniques can be used, in which the measurement equation is given by the (much less complicated) camera projection equation. Two fundamentally different approaches to control of such systems are possible, referred to as position-based and image-based visual servoing [Hutchinson *et al.*, 1996].

[1]In the literature, the term *feature-based tracking* is sometimes used with a different meaning, as a description for algorithms where the geometrical image feature itself, rather than the workspace position, is the structure to be tracked. This particular definition will not be used in this work.

In position-based visual servoing, pose estimation is used together with a feedback law defined in the workspace of the robot. In image-based visual servoing, the control law is based directly on the image feature data, without any explicit pose estimation. Image-based techniques are useful when the calibration accuracy of the camera system is insufficient for reliable pose estimation, or when the positioning task is specified more naturally in image space.

In Section 4.2 the basic version of a motion tracker is presented, using measurements of points and edges. As this motion estimation method suffers from a number of potential problems regarding robustness, we address two different problems related to the robustness and reliability of the edge-based tracker. In the first problem, a parametrization based on dual quaternions is used to provide additional linear constraints on the estimated state vector. This leads to improved robustness when tracking using a hand-eye camera, which is demonstrated in experiments with simultaneous tracking of motion and changing intrinsic camera parameters. In the second problem, a method for multi-camera visual tracking is presented, which aims at maximizing the accuracy of the estimate given a maximum allowed computation time. The suggested algorithm is evaluated in an extensive simulation study using a setup consisting of six cameras, with respect to both estimation variance and the resulting control performance. Finally, two basic approaches to force/vision control are presented. The first is a position-based technique, based on the edge-based motion tracker previously described. The second method is an image-based algorithm based on a hybrid force/vision control structure, where one degree of freedom is force-controlled, and the remaining degrees of freedom are controlled using a constrained visual servoing algorithm. The parameters describing the constraint equations can be estimated recursively from the sensor data.

4.2 Feature-Based Visual Tracking

A pose or motion estimator is a necessary component in any position-based visual control system. As mentioned above, in feature-based tracking methods an intermediate feature extraction step is included, such that the workspace position is estimated from the vector of feature coordinates.

Problem Formulation and State-of-the-Art. At least since the early 1980s, model-based visual tracking and estimation of the position of rigid objects have been active research topics. Traditionally, methods using non-linear minimization of the image-space errors [Lowe, 1991; Drummond and Cipolla, 2002; Martin and Horaud, 2002] or Kalman filtering techniques [Lippiello *et al.*, 2002] are employed. Other methods make use of the

geometrical structure of the problem, for instance by using a simplification of the perspective projection to find an approximate solution, followed by successive iterative refinements [Dementhon and Davis, 1995; Horaud *et al.*, 1997]. Such methods are applicable also in situations where no reliable initial value for the iteration can be found.

The main objective of recent research has been to improve robustness of tracking systems, with respect to error sources such as occlusions and reflections, unpredicted motions, data outliers and feature matching errors. Solutions include the use of data pre-processing with outlier elimination [Fischler and Bolles, 1981], applying robust statistical methods to the image measurements [Drummond and Cipolla, 2002; Comport *et al.*, 2005; Malis and Marchand, 2006], data fusion with multiple cameras [Martin and Horaud, 2002], and combining different types of image measurements with complementary properties regarding robustness and accuracy [Pressigout and Marchand, 2005; Rosten and Drummond, 2005; Kyrki and Kragic, 2006]. For motions described by dynamical systems, the robustness also depends strongly on the estimation algorithm. Particularly in highly cluttered environments with non-Gaussian disturbances, stochastic estimators capable of handling multi-modal probability distributions have proved useful. The primary examples are the CONDENSATION tracker and other types of Sequential Monte Carlo methods [Isard and Blake, 1998; Li *et al.*, 2003].

The question of real-time performance of visual tracking and servoing has also been considered by several researchers, with development of special-purpose hardware [Nakabo *et al.*, 2000] as well as efficient algorithms for feature extraction [Smith and Brady, 1997; Rosten and Drummond, 2005] and feature selection [Davison, 2005]. Apart from the performance and robustness issues, other researchers have focused on geometrical and representation problems, such as formulations of pose estimation problems using screw theory, geometric algebra [Rosenhahn *et al.*, 2005], or Lie algebra [Drummond and Cipolla, 2002]. The aim of such formulations is to obtain a consistent way to mathematically describe the output and state spaces, with respect to simplicity, compactness, and numerical efficiency and conditioning.

In this section, we present the feature-based framework used in this chapter. We present a singularity-free extension to the basic formulation, based on a parametrization using dual quaternions. Further advantages of the singularity-free representation include that the conditions of known angle and pitch of a screw motion can be translated into *linear* constraints on the state. This makes inclusion of the motion constraints in the estimation straightforward, and can be used to effectively reduce the number of estimated degrees of freedom by two.

Modeling and Motion Estimation

Assume that M cameras are placed in fixed locations, viewing a target object whose position and orientation with respect to some fixed (world) coordinate system should be estimated. The output of the pose estimation, i.e. the object position and orientation, can be parametrized in different ways, such as roll-pitch-yaw angles [Lippiello *et al.*, 2002], quaternions or dual quaternions [Olsson *et al.*, 2003]. We assume that the position and orientation are parametrized as a vector $\mathbf{z} \in \mathbb{R}^n$, where typically $n = 6$ or $n = 7$. The image data is compressed into a vector $\mathbf{y} \in \mathbb{R}^N$, usually representing the image space coordinates of corners, edges and other features. If the geometry of the target is known, $\mathbf{z}$ and $\mathbf{y}$ are related by the projection equations of the cameras

$$\mathbf{y} = \mathbf{h}(\mathbf{z}) \tag{4.1}$$

which is a nonlinear function, as described below. The task space position $\mathbf{z}$ could be obtained from a pose estimation, conceptually expressed as

$$\mathbf{z} = \mathbf{h}^{-1}(\mathbf{y}), \tag{4.2}$$

and used in a feedback control law in order to control the task space position. The pose estimation can be performed using some type of iterative least-squares optimization algorithm such as Gauss-Newton or Levenberg-Marquardt, using the previous position or prediction as a starting point for the iteration. Since this position should ideally be close to the true position, one or two iterations are usually sufficient in practice.

The most commonly used camera model is the homogeneous form pinhole camera projection equation, given by

$$\begin{pmatrix} \bar{u}_i \\ \bar{v}_i \\ w_i \end{pmatrix} = \mathbf{K}\mathbf{T}_{cw}\mathbf{T}_{wo}(\mathbf{z}) \begin{pmatrix} \mathbf{X}_i \\ 1 \end{pmatrix} \tag{4.3}$$

$$\mathbf{y_i} = \begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \bar{u}_i/w_i \\ \bar{v}_i/w_i \end{pmatrix}, \quad i \in [1, N] \tag{4.4}$$

where

$$\mathbf{K} = \begin{pmatrix} f & 0 & u_0 & 0 \\ 0 & \gamma f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.5}$$

is a matrix of intrinsic camera parameters as described in Appendix A, $\mathbf{X}_i$ is an object point expressed in the coordinate system of the object,

$w_i = w_i(\mathbf{z})$ is the depth of the point in the camera, and $\mathbf{T}_{cw}$ and $\mathbf{T}_{wo}(\mathbf{z})$ are the homogeneous coordinate transformation matrices between the world coordinate system and the camera, and between the target object and the world coordinate system, respectively. The parametrization $\mathbf{z}$ of $\mathbf{T}_{wo}$ corresponds to the unknown position/orientation to be estimated, while the camera position $\mathbf{T}_{cw}$ is assumed to be known. In practice, the relative positions of the cameras need to be accurately calibrated, in order to be able to relate measurements from the different cameras.

In order to linearize the projection equations (4.3)–(4.4), the image Jacobian or *interaction matrix*—describing the relation between workspace and feature velocities—must be computed. For many types of features, the Jacobian may be computed analytically, see for instance [Espiau *et al.*, 1992; Hutchinson *et al.*, 1996]. The Jacobian for a single point feature can be calculated using the equations

$$\begin{pmatrix} \dot{u}_i \\ \dot{v}_i \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{f}{w_i(\mathbf{z})} & 0 & -\frac{u_i}{w_i(\mathbf{z})} & -\frac{u_i v_i}{f} & \frac{f^2+u_i^2}{f} & -v \\ 0 & \frac{f}{w_i(\mathbf{z})} & -\frac{v_i}{w_i(\mathbf{z})} & -\frac{f^2+v_i^2}{f} & \frac{u_i v_i}{f} & -u \end{pmatrix}}_{\mathbf{J}_f^{(i)}(\mathbf{z})} \begin{pmatrix} \mathbf{v}_{oc} \\ \boldsymbol{\omega}_{oc} \end{pmatrix} \tag{4.6}$$

$$\begin{pmatrix} \mathbf{v}_{oc} \\ \boldsymbol{\omega}_{oc} \end{pmatrix} = \mathbf{J}_\omega(\mathbf{z})\dot{\mathbf{z}} \tag{4.7}$$

where f is the focal length of the corresponding camera[2], and $\mathbf{v}_{oc}$ and $\boldsymbol{\omega}_{oc}$ represent the velocity and angular velocity of the object with respect to the camera coordinate system, and where the Jacobian $\mathbf{J}_\omega(\mathbf{z})$ relates these velocities to the velocity in configuration coordinates $\mathbf{z}$. This results in the relation

$$\dot{\mathbf{y}}_i = \begin{pmatrix} \dot{u}_i \\ \dot{v}_i \end{pmatrix} = \mathbf{J}_f^{(i)}(\mathbf{z})\mathbf{J}_\omega(\mathbf{z})\dot{\mathbf{z}} \stackrel{\text{def}}{=} \mathbf{J}_v^{(i)}(\mathbf{z})\dot{\mathbf{z}}, \tag{4.8}$$

and the full Jacobian $\mathbf{J}_v(\mathbf{z})$ can be obtained by computing and stacking $\mathbf{J}_v^{(i)}(\mathbf{z})$ for each point $i \in [1, N]$ in each camera as

$$\dot{\mathbf{y}} = \underbrace{\begin{pmatrix} \mathbf{J}_v^{(1)}(\mathbf{z}) \\ \vdots \\ \mathbf{J}_v^{(N)}(\mathbf{z}) \end{pmatrix}}_{\mathbf{J}_v(\mathbf{z})} \dot{\mathbf{z}}. \tag{4.9}$$

[2]In the equations, it is assumed that the camera coordinates have been normalized such that the aspect ratio can be taken to be $\gamma = 1$. This simplification will be assumed throughout this work, unless otherwise indicated.

The Jacobian can be used to linearize the measurement equation Eq. (4.1) according to

$$\Delta\mathbf{y} = \mathbf{J}_v(\mathbf{z})\Delta\mathbf{z}. \tag{4.10}$$

Near singular configurations where $\mathbf{J}_v(\mathbf{z})$ loses rank, the pose estimation becomes very inaccurate. An example of such a situation is when the relative depth of the object points is small. In such cases a very accurate depth estimation may be required for each point in order to obtain convergence [Malis and Rives, 2003].

Instead of using measurements of the positions of point features [Lippiello *et al.*, 2002], it is possible to use features such as lines [Wunsch and Hirzinger, 1997] or edges [Drummond and Cipolla, 2002; Martin and Horaud, 2002]. In many cases, edges are the only structures that can be extracted reliably from the image data. Another major advantage of such measurements is that exact matching of features is not required, but only the error in the normal direction at a number of points on an edge [Drummond and Cipolla, 2002; Martin and Horaud, 2002]. This requires only a one-dimensional search for features (edges), which can be performed extremely quickly. However, this means that due to the so called *aperture problem*, only local image motion normal to the edge is measurable, while motion parallel to the edge becomes unobservable. This can be accounted for by modifying Eq. (4.8) by projecting the motion onto the normal direction as

$$\mathbf{n}_i^T(\mathbf{z})\dot{\mathbf{y}}_i = \mathbf{n}_i^T(\mathbf{z})\mathbf{J}_v^{(i)}(\mathbf{z})\dot{\mathbf{z}} \tag{4.11}$$

where $\mathbf{n}_i$ is an edge normal vector at point i along the edge [Drummond and Cipolla, 2002; Martin and Horaud, 2002]. This gives an equation analogous to Eq. (4.10) given by

$$\Delta\mathbf{e} \approx \mathbf{N}^T(\mathbf{z})\Delta\mathbf{y} = \mathbf{N}^T(\mathbf{z})\mathbf{J}_v(\mathbf{z})\Delta\mathbf{z} \stackrel{\text{def}}{=} \mathbf{J}_{v,\mathbf{N}}(\mathbf{z})\Delta\mathbf{z} \tag{4.12}$$

where $\mathbf{N}$ is a block diagonal matrix of the vectors $\mathbf{n}_i$, and $\Delta\mathbf{e}$ is the vector of measured normal distances between the measured and predicted edges, see Fig. 4.1 for an illustration. The small error in (4.12) is caused by approximating the measured normal distances with the edge normal projections of the point-to-point errors (Fig. 4.1). The approximation can be shown to result in an extra nearly quadratic error term in the estimation error $\tilde{\mathbf{z}}$. In most cases, this error term is negligible compared to the higher-order error terms from the linearization, and using (4.12) works well in practice. However, in some particular situations, such as for fast image-plane rotations near singular configurations, the effects of the error term become apparent. If the local orientations θ_i of each measured edge relative to the predicted edge are measurable, it is possible to directly

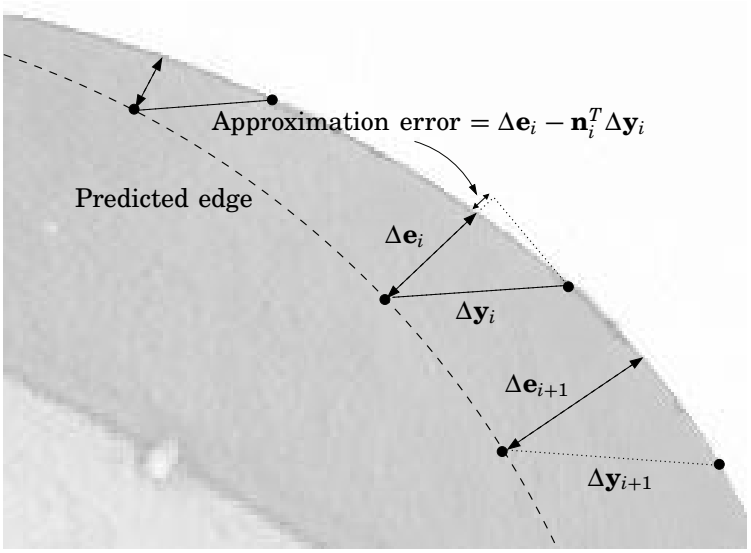

Figure 4.1 Edge detection in the normal direction of the predicted edges, illustrating the difference between the point-based measurements $\Delta \mathbf{y}$ and the edge-based measurements $\Delta \mathbf{e}$.

compensate for this error by modifying (4.12) to

$$\Delta \mathbf{e} \approx \left(\mathbf{N}^T(\mathbf{z}) + \boldsymbol{\Theta} \mathbf{T}^T(\mathbf{z}) \right) \mathbf{J}_v(\mathbf{z}) \Delta \mathbf{z} \stackrel{\text{def}}{=} \mathbf{J}_{v,\boldsymbol{\Theta}}(\mathbf{z}) \Delta \mathbf{z}, \tag{4.13}$$

where $\boldsymbol{\Theta}$ is a diagonal matrix of the image edge orientation angles. The tangential projection matrix $\mathbf{T}$ is a block diagonal matrix of the same form as $\mathbf{N}$, but with its block-diagonals given by the *tangential* vectors at each point along the predicted edge. The effects of the quadratic compensation in (4.13) can be seen in Fig. 4.2, in which the measured edge distance $\Delta \mathbf{e}_i$ for one measurement point is shown, for a simple setup of a planar quadratic object rotating and translating in the image plane. In this case, the compensation in (4.13) captures the nonlinear behavior for this particular measurement more accurately than the simple linearization (4.12).

Dynamic tracking. If we assume that the task space dynamics of the motion-controlled manipulator can be modeled as a linear system, we obtain the Wiener-type model

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{u} \\ \mathbf{z} = \mathbf{C}\mathbf{x} \\ \mathbf{y} = \mathbf{h}(\mathbf{z}) + \boldsymbol{\epsilon} \end{cases} \tag{4.14}$$

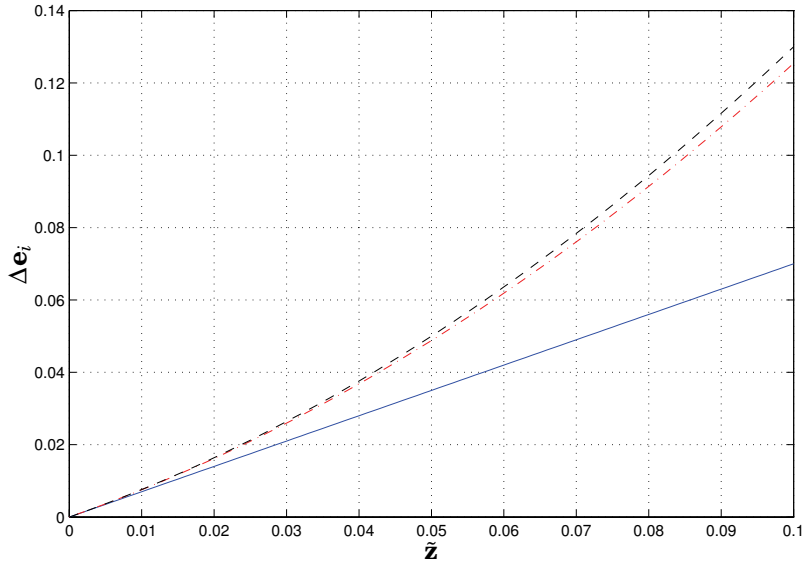

Figure 4.2 Measured edge distance $\Delta \mathbf{e}_i$ (*dashed*) as a function of the workspace estimation error $\tilde{\mathbf{z}}$, together with the linearization $\mathbf{J}_{v,\mathbf{N}}(\mathbf{z})\tilde{\mathbf{z}}$ (*solid*) of Eq. (4.12) and the compensated linearization $\mathbf{J}_{v,\boldsymbol{\Theta}}(\mathbf{z})\tilde{\mathbf{z}}$ (*dash-dotted*) of Eq. (4.13).

where $\mathbf{u}$ is the input, $\mathbf{x}$ is the state vector, and $\boldsymbol{\epsilon}$ denotes a measurement disturbance or noise sequence. For relatively low bandwidth systems, such as normal vision-based controllers, the approximation of the complex closed loop robot dynamics with a linear system of relatively low order may often be reasonable. A state estimator, using a correction term $\Delta \mathbf{z} = \mathbf{J}_v^{\dagger} \Delta \mathbf{y}$ obtained from the linearization, is given by

$$\frac{d\hat{\mathbf{x}}}{dt} = \mathbf{F}\hat{\mathbf{x}} + \mathbf{G}\mathbf{u} + \bar{\mathbf{K}}\Delta \mathbf{z} \stackrel{\text{def.}}{=} \mathbf{F}\hat{\mathbf{x}} + \mathbf{G}\mathbf{u} + \bar{\mathbf{K}}\mathbf{J}_v^{\dagger}(\mathbf{C}\hat{\mathbf{x}})\Delta \mathbf{y}, \tag{4.15}$$

or equivalently for edge measurements

$$\frac{d\hat{\mathbf{x}}}{dt} = \mathbf{F}\hat{\mathbf{x}} + \mathbf{G}\mathbf{u} + \bar{\mathbf{K}}\mathbf{J}_{v,\mathbf{N}}^{\dagger}(\mathbf{C}\hat{\mathbf{x}})\Delta \mathbf{e}, \tag{4.16}$$

where $\mathbf{J}^{\dagger}$ is the pseudo inverse of $\mathbf{J}$. The error dynamics is approximated by the system

$$\frac{d\tilde{\mathbf{x}}}{dt} = \left(\mathbf{F} - \bar{\mathbf{K}}\mathbf{C}\right)\tilde{\mathbf{x}} - \bar{\mathbf{K}}\mathbf{J}_v^{\dagger}(\mathbf{C}\hat{\mathbf{x}})\boldsymbol{\epsilon}, \tag{4.17}$$

with an equivalent expression in the edge-based case. In practice, the estimator is implemented in discrete time as a Kalman filter

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}(k|k\text{–}1) + \mathbf{K}_k \underbrace{\mathbf{J}_{v,\mathbf{N}}^{\dagger}(\mathbf{C}_d\hat{\mathbf{x}}(k|k\text{–}1))\,\Delta\mathbf{e}(k)}_{\Delta\mathbf{z}(k)} \tag{4.18}$$

$$\hat{\mathbf{x}}(k|k\text{–}1) = \mathbf{\Phi}_d\hat{\mathbf{x}}(k\text{–}1) + \mathbf{\Gamma}_d\mathbf{u}(k\text{–}1) \tag{4.19}$$

$$\mathbf{K}_k = \mathbf{P}(k|k\text{–}1)\mathbf{C}_d^T(\mathbf{C}_d\mathbf{P}(k|k\text{–}1)\mathbf{C}_d^T + \mathbf{R}_2(k))^{-1} \tag{4.20}$$

$$\mathbf{P}(k{+}1|k) = \mathbf{\Phi}_d(\mathbf{I} - \mathbf{K}_k\mathbf{C}_d)\mathbf{P}(k|k\text{–}1)\mathbf{\Phi}_d^T + \mathbf{R}_1 \tag{4.21}$$

where $(\mathbf{\Phi}_d, \mathbf{\Gamma}_d, \mathbf{C}_d)$ are the system matrices of the corresponding discrete-time system model, $\mathbf{R}_2(k)$ is the time-varying covariance matrix of the white (discrete-time) measurement noise $\boldsymbol{\epsilon}_k$, and $\mathbf{R}_1$ the covariance matrix of a modeled input noise sequence. Due to the shape of the correction term and Jacobian in (4.18), the accuracy of the estimation will improve with the number of image measurements N. If we assume that the measurement errors $\boldsymbol{\epsilon}$ can be modeled as Gaussian, spatially uncorrelated white noise with variance σ^2, a useful approximation of the covariance $\mathbf{R}_2$ of the effective measurement error $\mathbf{J}_v^{\dagger}\boldsymbol{\epsilon}$ can be obtained as

$$\mathbf{R}_2 = \mathrm{E}[\mathbf{J}_v^{\dagger}\boldsymbol{\epsilon}(\mathbf{J}_v^{\dagger}\boldsymbol{\epsilon})^T] = (\mathbf{J}_v^T\mathbf{J}_v)^{-1}\sigma^2 = (\sum_{i=1}^{M}\mathbf{J}_{v,i}^T\mathbf{J}_{v,i})^{-1}\sigma^2 \tag{4.22}$$

where the Jacobian has been partitioned into the individual Jacobians for each of the M cameras as $\mathbf{J}_v^T = [\mathbf{J}_{v,1}^T, \mathbf{J}_{v,2}^T, \cdots \mathbf{J}_{v,M}^T]^T$. In Section 4.3 a method which attempts to minimize the measurement error covariance in Eq. (4.22) by a proper selection of active cameras will be presented.

Robust Estimation and Implementation. In practice, the estimator (4.15) or (4.16) suffers from problems with outliers in the data, due to their similarity to least-squares approximation methods. The problem of outliers is traditionally solved by removing the outliers using, for instance, RANSAC [Fischler and Bolles, 1981], or by using other criteria than the ℓ_2-norm in the minimization. Here, we use an algorithm based on iterative re-weighted least-squares, similar to [Drummond and Cipolla, 2002]. In each sample two iterations are performed, where after the first iteration the weights $s_\sigma(\mathbf{d}_i)$ are applied to each measurement. The weights (Fig. 4.3) are given by the Huber penalty function [Boyd and Vandenberghe, 2004], defined by the function

$$s_\sigma(\mathbf{d}_i) = \begin{cases} \sigma\frac{\mathrm{sgn}(\mathbf{d}_i)}{\mathbf{d}_i}, & |\mathbf{d}_i| > \sigma \\ 1, & |\mathbf{d}_i| \leq \sigma \end{cases} \tag{4.23}$$

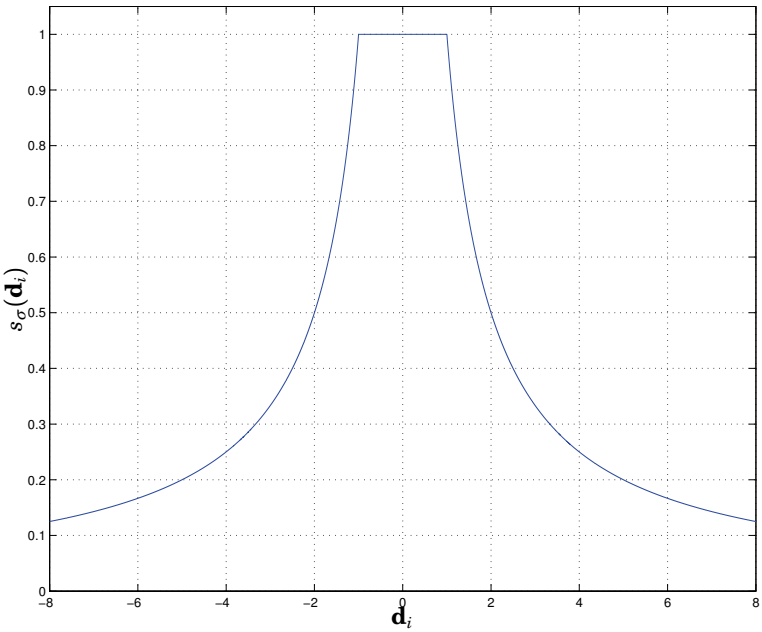

Figure 4.3 The weights $s_\sigma(\mathbf{d}_i)$ (for $\sigma = 1$) used in the robust version of the edge-based visual tracker, based on the Huber penalty function.

of the remaining image-space errors $\mathbf{d}$. Mathematically, the procedure for computing the correction term $\Delta\mathbf{z}$ can be described by

$$\mathbf{d} = \Delta\mathbf{e} - \mathbf{J}_{v,\mathbf{N}}\mathbf{J}^{\dagger}_{v,\mathbf{N}}\Delta\mathbf{e} \tag{4.24}$$

$$\Delta\mathbf{z} = \left[\mathbf{S}_\sigma(\mathbf{d})\mathbf{J}_{v,\mathbf{N}}\right]^{\dagger}\mathbf{S}_\sigma(\mathbf{d})\Delta\mathbf{e} \tag{4.25}$$

where $\mathbf{S}_\sigma(\mathbf{d})$ is a diagonal matrix with the weights $s_\sigma(\mathbf{d}_i)$ as diagonal elements. The parameter σ is usually set to 1–3 pixels, depending on the expected image noise level.

In order to be able to execute the tracker using a large number of features at frame rate, while minimizing the input-output latency, a number of issues related to the program structure must be considered. The basic algorithm contains a number of steps to be carried out in sequence during each sample k.

ALGORITHM 4.1—REAL-TIME DYNAMIC EDGE-BASED RIGID BODY TRACKING

1. Wait for the next set of images from the cameras. The capture is performed in hardware with images being transferred to memory through direct memory access, making complete overlap between capture and processing possible.

2. Perform image pre-processing, in the form of color-space conversion and region-of-interest rescaling for the multi-scale edge detection algorithm described below.

3. Based on the predicted position, measure distances $\Delta\mathbf{e}(k)$ between the predicted edge positions and the image edges as illustrated in Fig. 4.1. Remove measurements and rows of the Jacobians corresponding to non-detected features.

4. Compute the correction term and the remaining errors $\mathbf{d}$ in (4.24). Re-weight the measurements according to (4.23), and update the state estimate by (4.18), using the correction term $\Delta\mathbf{z}$ from (4.25).

5. In case of feedback, compute the control signal and actuate the process accordingly.

6. Compute the one-step prediction $\hat{\mathbf{x}}(k+1|k)$ of the state from (4.19).

7. Using the state prediction, predict which features will be visible in the next set of images. Using the predicted features, pre-calculate Jacobians $\mathbf{J}^{\dagger}_{v,\mathbf{N}}(\mathbf{C}_d\hat{\mathbf{x}}(k+1|k))$ and other data for the next sample.

□

The visibility of individual features is determined based on the predicted object pose and a pre-generated Binary Search Partitioning (BSP) tree description of the object [van Dam *et al.*, 1991]. The BSP tree representation can be computed off-line based on the object model, resulting in fast online computations. See Fig. 4.4 for an illustration of the BSP tree algorithm. Off-line the algorithm recursively arranges all polygons in the object model into a binary tree according to which polygons are "in front" and "behind", starting with an arbitrary surface and viewpoint. In this way, a perfect front-to-back ordering of the entire object model is obtained. Online, a recursive inorder traversal of the BSP tree is performed, with surfaces processed in front-to-back order *with respect to the current viewpoint*. Each surface is clipped directly in the image against all surfaces previously drawn, resulting in an extremely fast algorithm for determining which edges and surface patches are visible. Along these visible edges, the image edge measurements are then obtained from a one-dimensional edge search at each point. The edges are found from a fast convolution with a differentiated Gauss kernel at three different scales, where the rough initial localization is iteratively refined at finer scales in order to achieve a robust detection/localization.

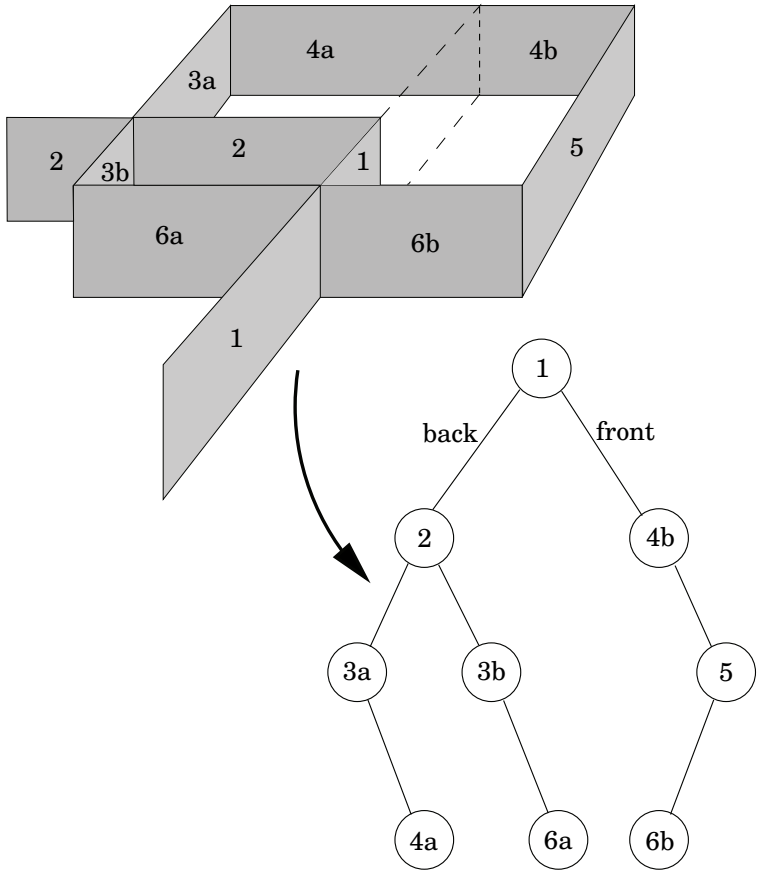

Figure 4.4 Object and corresponding BSP tree with respect to the given viewpoint, with tree built given the polygon ordering and clipping shown. Quadrilaterals number 3, 4 and 6 are automatically split during the construction of the BSP tree. A front-to-back ordering given this viewpoint would correspond to the sequence of polygons $5 \rightarrow 6b \rightarrow 4b \rightarrow 1 \rightarrow 6a \rightarrow 3b \rightarrow 2 \rightarrow 4a \rightarrow 3a$. A different choice for the numbering of the polygons would lead to a different BSP tree, with different balancing properties and different quadrilaterals being split.

EKF Tracking Using Dual Quaternions

One issue of some practical importance concerns the choice of representation for the space of configurations. Representation singularities will occur in certain configurations when using a representation of $SE(3)$ with a number of parameters less than or equal to six, with the Euler angles representation of orientation being a well-known example. Using a suit-

able higher-order representation of the configuration space, the problem can be avoided. A more serious problem is related to the singularities and conditioning of the projection mapping itself. For certain configurations and sets of visible features, the image Jacobian $\mathbf{J}_v$ loses rank or becomes very poorly conditioned. In such configurations, the sensitivity to errors is increased, frequently leading to breakdown and loss of tracking. The tracking problem is particularly difficult when many parameters are unknown, and therefore need to be estimated simultaneously. An example was given by [Drummond and Cipolla, 2002], in which a method was presented to recursively estimate not only the position and orientation, but also the (possibly varying) intrinsic parameters of the camera. The problem of simultaneously tracking position and intrinsic parameters is poorly conditioned in situations where the relative depth of all feature points in the image is small [Martin and Horaud, 2002]. In such cases, the only solution may be to reduce the number of estimated parameters, by using some extra information about the system.

One example of such information is if partial information about the object motion is available. If the motion in one or several degrees of freedom can be measured by other devices, for instance by a laser tracker or range sensor, the motion parameters for these degrees of freedom may just be treated as known parameters in the estimation. This would reduce the number of estimated motion parameters, increasing the accuracy and robustness. Here, we consider the less straightforward case which occurs if the object (or camera) is known to be rigidly attached to a robot hand, the motion of which can be measured. This corresponds to the setup illustrated in Fig. 2.3 of Chapter 2, repeated in Fig. 4.5 using a different notation. Assume that the objective is to accurately position the camera with respect to a stationary object in the world. The initial pose $\mathbf{A}(t_0)$ of the camera with respect to this object is assumed to be known. If the hand-eye calibration parameters of the system were perfectly known, accurate positioning of the camera could be achieved without camera feedback, instead using the robot hand motion $\mathbf{B}(t)$ with $\mathbf{X}$ to compute the current camera pose $\mathbf{A}(t)$. In practice, even small calibration errors in $\mathbf{X}$ would cause unacceptable positioning errors, and measurements from the camera would be needed in order to correct for these errors. At first view, it may seem as if full 6-DoF tracking would be necessary for this compensation, as there is no straightforward way to account for the effects of the calibration errors. However, although the hand-eye calibration can not be relied upon for positioning, some useful information can be obtained even from a completely uncalibrated eye-in-hand camera system. This information comes from the knowledge of the angle and pitch of the camera screw motion, as shown in the following section. This allows the number of estimated degrees of freedom to be decreased by two, thereby increasing the

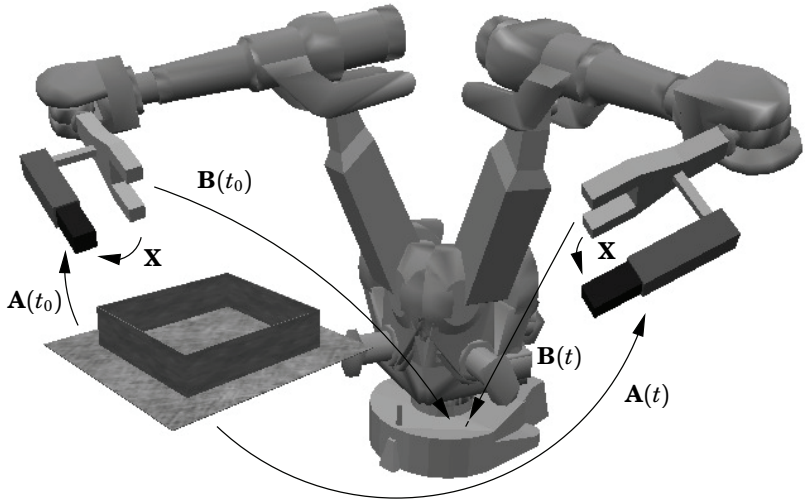

Figure 4.5 Tracking and positioning of a robot-mounted camera with respect to a stationary object, shown at two different times t_0 and t.

robustness of the system.

A Comment on Notation. Due to the nature of the dual quaternion representation used in this section, a detailed notational system is required. The notation for quantities related to quaternions and dual quaternions in this section follows the system described below.

$\mathbf{q}$ denotes a quaternion.

$\check{q}$ denotes a dual quantity, for example a dual number $\check{q}$, a dual vector $\check{\vec{\mathbf{q}}}$ or dual quaternion $\check{\mathbf{q}}$.

q^0 denotes the scalar part of the quantity $\mathbf{q}$, where $\mathbf{q}$ is a quaternion or dual quaternion (in which case the scalar part is a dual number denoted $\check{q}^0$).

$\vec{\mathbf{q}}$ is used to denote a vector, or the vector part of a quaternion or dual quaternion (in which case it is a dual vector denoted $\check{\vec{\mathbf{q}}}$).

q' denotes the dual part of the dual quantity $\check{q}$, and will be a (real) scalar q', a vector $\vec{\mathbf{q}}'$, or a quaternion $\mathbf{q}'$, depending on the context.

$\bar{\mathbf{q}}$ denotes the conjugate of $\mathbf{q}$.

Quaternions and Dual Quaternions. In this section we briefly introduce the properties of dual quaternions used in this chapter. For a more detailed description and introduction to the theory and properties of quaternions and screws, see for instance [Murray *et al.*, 1994; Daniilidis, 1999; Goddard, 1997].

Quaternions were first invented by Hamilton [Hamilton, 1853] as a non-commutative extension to complex numbers. The usefulness of (unit) quaternions in robotics comes from their use for representing and computing with rotations in three dimensions. Quaternions can be represented as a pair $\mathbf{q} = (q^0, \vec{\mathbf{q}})$, where $q^0 \in \mathbb{R}$ and $\vec{\mathbf{q}} \in \mathbb{R}^3$, with the operations

$$\mathbf{q}_1 + \mathbf{q}_2 = (q_1^0 + q_2^0, \vec{\mathbf{q}}_1 + \vec{\mathbf{q}}_2) \tag{4.26}$$

$$k\mathbf{q} = (kq^0, k\vec{\mathbf{q}}) \tag{4.27}$$

$$\mathbf{q}_1\mathbf{q}_2 = (q_1^0 q_2^0 - \vec{\mathbf{q}}_1^T\vec{\mathbf{q}}_2, q_1^0\vec{\mathbf{q}}_2 + q_2^0\vec{\mathbf{q}}_1 + \vec{\mathbf{q}}_1 \times \vec{\mathbf{q}}_2) \tag{4.28}$$

where $k \in \mathbb{R}$. A quaternion has a norm which is given by $\|\mathbf{q}\|^2 = \mathbf{q}\bar{\mathbf{q}}$, where $\bar{\mathbf{q}} = (q^0, -\vec{\mathbf{q}})$ is the conjugate quaternion. It is well known that every rigid rotation (element of the special orthogonal group SO(3)) with angle θ about an axis $\vec{\mathbf{n}}$ with $\|\vec{\mathbf{n}}\| = 1$ can be represented as a unit quaternion

$$\mathbf{q} = (\cos(\theta/2), \sin(\theta/2)\vec{\mathbf{n}}). \tag{4.29}$$

The quaternion $\mathbf{q}$ rotates a vector $\vec{\mathbf{x}} \in \mathbb{R}^3$ to the vector represented by the quaternion $\mathbf{q}(0, \vec{\mathbf{x}})\bar{\mathbf{q}}$.

Similarly to real quaternions, dual quaternions are defined according to $\check{\mathbf{q}} = (\check{q}^0, \check{\vec{\mathbf{q}}})$, where $\check{q}^0 = q^0 + \epsilon q'^0$ is a dual number[3], and where $\check{\vec{\mathbf{q}}} = \vec{\mathbf{q}} + \epsilon\vec{\mathbf{q}}'$ is a dual vector. Analogously to ordinary quaternions, the dual quaternion operations are

$$\check{\mathbf{q}}_1 + \check{\mathbf{q}}_2 = (\check{q}_1^0 + \check{q}_2^0, \check{\vec{\mathbf{q}}}_1 + \check{\vec{\mathbf{q}}}_2) \tag{4.30}$$

$$k\check{\mathbf{q}} = (k\check{q}^0, k\check{\vec{\mathbf{q}}}) \tag{4.31}$$

$$\check{\mathbf{q}}_1\check{\mathbf{q}}_2 = (\check{q}_1^0\check{q}_2^0 - \check{\vec{\mathbf{q}}}_1^T\check{\vec{\mathbf{q}}}_2, \check{q}_1^0\check{\vec{\mathbf{q}}}_2 + \check{q}_2^0\check{\vec{\mathbf{q}}}_1 + \check{\vec{\mathbf{q}}}_1 \times \check{\vec{\mathbf{q}}}_2). \tag{4.32}$$

We will often write a dual quaternion as the sum of its real and dual parts $\mathbf{q} + \epsilon\mathbf{q}'$. Its norm is given by $\|\check{\mathbf{q}}\|^2 = \check{\mathbf{q}}\bar{\check{\mathbf{q}}}$ with $\bar{\check{\mathbf{q}}} = \bar{\mathbf{q}} + \epsilon\bar{\mathbf{q}}'$, and the unity conditions become

$$\mathbf{q}\bar{\mathbf{q}} = 1 \tag{4.33}$$

$$\bar{\mathbf{q}}\mathbf{q}' + \bar{\mathbf{q}}'\mathbf{q} = 0. \tag{4.34}$$

[3]Dual numbers were invented by Clifford [Clifford, 1873]. Complementary to the complex numbers, a dual number is defined as $\check{z} = a + \epsilon b$ with a and b real numbers, and where the dual element ϵ satisfies the property $\epsilon^2 = 0$.

Unit dual quaternions can be used to represent general rigid transformations including translations, similarly to the way rotations can be represented by real quaternions. The rigid transformation of a line passing through the point $\vec{\mathbf{p}}$, represented by its direction $\vec{\mathbf{n}}$ and moment $\vec{\mathbf{m}} = \vec{\mathbf{p}} \times \vec{\mathbf{n}}$, is given by $\check{\mathbf{q}}(\mathbf{n} + \epsilon\mathbf{m})\bar{\check{\mathbf{q}}}$, where $\vec{\mathbf{n}}$ and $\vec{\mathbf{m}}$ are expressed as quaternions $\mathbf{n} = (0, \vec{\mathbf{n}})$ and $\mathbf{m} = (0, \vec{\mathbf{m}})$, respectively [Daniilidis, 1999]. The dual quaternion itself may be expressed as $\mathbf{q} + \epsilon\mathbf{q}'$, where $\mathbf{q}$ is the quaternion describing the rotation, and where $\mathbf{q}' = \mathbf{t}\mathbf{q}/2$ with $\mathbf{t} = (0, \vec{\mathbf{t}})$ is the translation. According to Chasles' theorem, a general rigid transformation can be modeled as a rotation about an axis (not necessarily through the origin) and a translation along the same axis [Murray *et al.*, 1994]. The parameters of the screw are the direction $\vec{\mathbf{n}}$ and the moment $\vec{\mathbf{m}}$ of the screw axis line, the rotation angle θ, and the translation (pitch) d along $\vec{\mathbf{n}}$. Together with the constraints $\vec{\mathbf{n}}^T\vec{\mathbf{n}} = 1$ and $\vec{\mathbf{n}}^T\vec{\mathbf{m}} = 0$ these parameters constitute the six degrees of freedom of a rigid transformation. It can be shown that the dual quaternion corresponding to the screw with parameters $\vec{\mathbf{n}}$, $\vec{\mathbf{m}}$, θ, and d can be written as

$$\check{\mathbf{q}} = (\cos(\check{\theta}/2), \sin(\check{\theta}/2)\check{\vec{\mathbf{l}}}), \tag{4.35}$$

where the dual angle is $\check{\theta} = \theta + \epsilon d$, and the line is given by $\check{\vec{\mathbf{l}}} = \vec{\mathbf{n}} + \epsilon\vec{\mathbf{m}}$. Note the similarity between the representation of rigid transformations in Eq. (4.35) to the case of rotations and ordinary quaternions in Eq. (4.29).

The hand-eye equation (2.1) can be written using dual quaternions as

$$\check{\mathbf{q}}_A = \check{\mathbf{q}}_X \check{\mathbf{q}}_B \bar{\check{\mathbf{q}}}_X \tag{4.36}$$

where $\check{\mathbf{q}}_A$, $\check{\mathbf{q}}_B$ and $\check{\mathbf{q}}_X$ are dual quaternion representations of the rigid transformations $\mathbf{A}(t)\mathbf{A}(t_0)^{-1}$, $\mathbf{B}(t)^{-1}\mathbf{B}(t_0)$ and $\mathbf{X}$ in Fig. 4.5, respectively. A consequence of Eq. (4.36) is that the scalar parts of $\check{\mathbf{q}}_A$ and $\check{\mathbf{q}}_B$ are equal, which follows directly from the equation [Daniilidis, 1999]

$$\begin{aligned} \mathrm{Sc}(\check{\mathbf{q}}_A) &= \frac{1}{2}(\check{\mathbf{q}}_A + \bar{\check{\mathbf{q}}}_A) = \frac{1}{2}(\check{\mathbf{q}}_X \check{\mathbf{q}}_B \bar{\check{\mathbf{q}}}_X + \check{\mathbf{q}}_X \bar{\check{\mathbf{q}}}_B \bar{\check{\mathbf{q}}}_X) = \\ &= \frac{1}{2}\check{\mathbf{q}}_X(\check{\mathbf{q}}_B + \bar{\check{\mathbf{q}}}_B)\bar{\check{\mathbf{q}}}_X = \mathrm{Sc}(\check{\mathbf{q}}_B)\check{\mathbf{q}}_X \bar{\check{\mathbf{q}}}_X = \mathrm{Sc}(\check{\mathbf{q}}_B). \end{aligned} \tag{4.37}$$

From the expression for the dual quaternion in Eq. (4.35), and using that a continuous and differentiable function of a dual number can be rewritten using Taylor expansion as

$$f(a + \epsilon b) = f(a) + \epsilon b f'(a), \tag{4.38}$$

we can write Eq. (4.37) as

$$\underbrace{\cos\frac{\theta_a}{2} - \epsilon\frac{d_a}{2}\sin\frac{\theta_a}{2}}_{\cos(\check{\theta}_a/2)} = \underbrace{\cos\frac{\theta_b}{2} - \epsilon\frac{d_b}{2}\sin\frac{\theta_b}{2}}_{\cos(\check{\theta}_b/2)}. \tag{4.39}$$

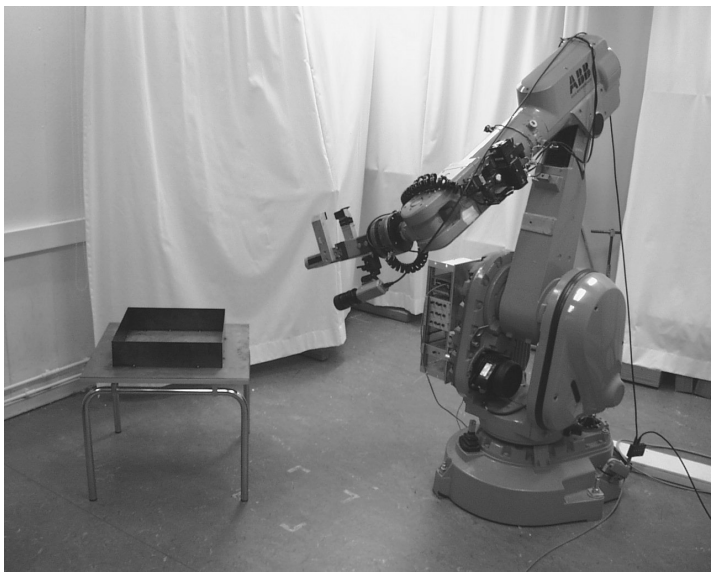

Figure 4.6 ABB Irb2000 industrial robot with robot-mounted digital camera used in the experiments.

Dividing the equality in (4.39) into real and dual parts, we can see that the angle and pitch of the camera screw and the robot end-effector screw must be equal. This is known as the Screw Congruence Theorem, see [Chen, 1991]. In the hand-eye calibration method of [Daniilidis, 1999], this equality is used to rewrite the hand-eye equation using only the vector parts of $\check{\mathbf{q}}_A$ and $\check{\mathbf{q}}_B$. Each motion of the robot and camera will provide six linear equations in the unknowns $\mathbf{q}_X$ and $\mathbf{q}'_X$, the real and dual parts of the unknown hand-eye dual quaternion. A minimum of two motions together with the constraints from Eqs. (4.33)–(4.34) are generally enough to solve for the eight unknowns. The solution is obtained by finding the vectors spanning the null space of the linear system using SVD, and then finding the particular linear combination which satisfies the unity conditions (4.33)–(4.34), see [Daniilidis, 1999] for details.

Modeling and Assumptions. We assume a setup with a single camera, viewing a stationary rigid object. The camera is attached to the hand of a robot, as described by Fig. 4.5. The camera is modeled as a four parameter pinhole camera as in Appendix A, where the parameters in the camera model are assumed to be the focal length f, aspect ratio γ, and principal point coordinates u_0, v_0. The object geometry and the initial rigid transformation between the object and the camera are assumed to

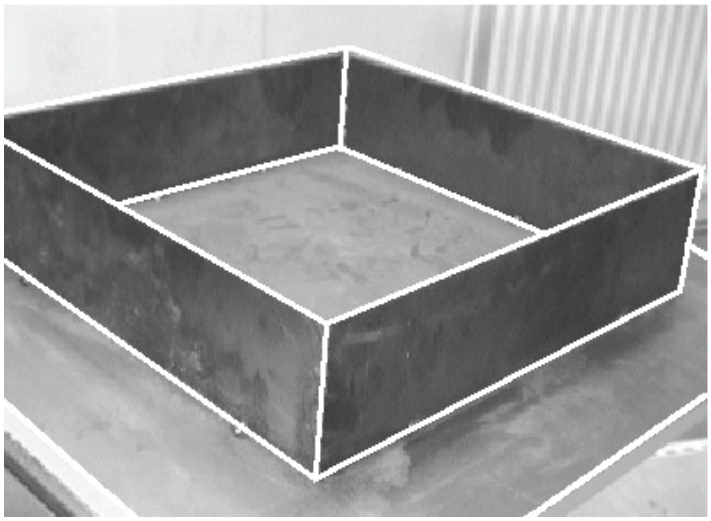

Figure 4.7 Example of image with superimposed object model, where hidden object edges have been removed.

be known. The object motion relative to the camera should be estimated, together with the intrinsic camera parameters. The motion of the system is assumed to be modeled as a nonlinear discrete-time dynamic system

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) \tag{4.40}$$

$$\mathbf{z}_k = \mathbf{C}\mathbf{x}_k \tag{4.41}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{z}_k) \tag{4.42}$$

with $\mathbf{x}_k \in \mathbb{R}^n$ the state of the system, $\mathbf{z}_k \in \mathbb{R}^{8+n_K}$ containing the pose expressed as a dual quaternion in addition to n_K intrinsic camera parameters, and $\mathbf{y}_k \in \mathbb{R}^m$ a vector of measured outputs. The projection function $\mathbf{h}$ relates the pose parameters to the output, as previously described. As shown above, the state vector contains a parametrization of the position/orientation of the tracked object given by $\mathbf{q}$, $\mathbf{q}' \in \mathbb{R}^4$. These are the vector representations of the parameters of the dual quaternion $\check{\mathbf{q}} = \mathbf{q} + \epsilon\mathbf{q}'$, which represents the rigid transformation $\mathbf{A}(t)$ in Fig. 4.5. As before, the projection equation can be expressed on linearized form by use of the image Jacobian for the dual quaternion parametrization. The image Jacobian with respect to the pose and camera parameters

$$\mathbf{J}_v(\hat{\mathbf{z}}_k) = \frac{\partial \mathbf{h}}{\partial \mathbf{z}}(\hat{\mathbf{z}}_k) \tag{4.43}$$

can in this case be calculated directly by analytical differentiation of

Eqs. (4.3)–(4.4) with respect to the elements of $\mathbf{z}$. In order to express the projection equation as a function of the dual quaternion parameters, the rotation matrix can be calculated directly from the unit quaternion $\mathbf{q}$, and the translation vector can be obtained from $\breve{\mathbf{q}}$ as

$$\mathbf{t} = 2\mathbf{q}'\bar{\mathbf{q}}. \tag{4.44}$$

Including the State Constraints. The Jacobian $\mathbf{J}_v$ in (4.43) contains the differential relationship between the eight parameters of the dual quaternion and the image-space error vector. Due to the constraint that the dual quaternion should have unit norm, the two quadratic constraints (4.33)–(4.34) should also be included, thereby reducing the number of degrees of freedom to six. The price to pay for this reduction is that special care is needed in order to handle the nonlinear constraints (4.33)–(4.34). In contrast, inclusion of the hand-eye constraints from Eq. (4.39) is considerably easier, and can be performed without approximation or linearization. Consider the known dual quaternion $\breve{\mathbf{q}}_B$, representing the current measured pose of the robot hand relative to its initial pose. The corresponding object-camera transformations are $\breve{\mathbf{q}}(t_0)$, corresponding to the known initial pose, and $\breve{\mathbf{q}}(t)$, which is the current pose to be tracked. From the assumptions and Eq. (4.37), we know that the scalar parts of $\breve{\mathbf{q}}(t)\bar{\breve{\mathbf{q}}}(t_0)$ and $\breve{\mathbf{q}}_B$ must be equal. Define the scalar part of $\breve{\mathbf{q}}_B$ as

$$\breve{q}_B^{(0)} = q_B^{(0)} + \epsilon q_B'^{(0)}. \tag{4.45}$$

The scalar part of $\breve{\mathbf{q}}(t)\bar{\breve{\mathbf{q}}}(t_0)$ can be seen from Eq. (4.32) to be

$$\mathrm{Sc}\left(\breve{\mathbf{q}}(t)\bar{\breve{\mathbf{q}}}(t_0)\right) = \mathbf{q}^T(t_0)\mathbf{q}(t) + \epsilon(\mathbf{q}'^T(t_0)\mathbf{q}(t) + \mathbf{q}^T(t_0)\mathbf{q}'(t)), \tag{4.46}$$

with the quaternions written on 4-vector form. Setting the scalar parts equal provides two more linear equations in the states, given by

$$\mathbf{q}^T(t_0)\mathbf{q}(t) = q_B^{(0)} \tag{4.47}$$

$$\mathbf{q}'^T(t_0)\mathbf{q}(t) + \mathbf{q}^T(t_0)\mathbf{q}'(t) = q_B'^{(0)}. \tag{4.48}$$

The set of equations to be solved for the error $\Delta\mathbf{z}_k$ can now be formulated as

$$\mathbf{h}(\hat{\mathbf{z}}_k + \Delta\mathbf{z}_k) = \mathbf{y}_k \tag{4.49}$$

$$\mathbf{a}_0\Delta\mathbf{z}_k = \mathbf{b}_k \tag{4.50}$$

$$(\hat{\mathbf{z}}_k + \Delta\mathbf{z}_k)^T\mathbf{Q}_n(\hat{\mathbf{z}}_k + \Delta\mathbf{z}_k) = 1. \tag{4.51}$$

where the matrices $\mathbf{a}_0$ and $\mathbf{b}_k$, containing the parameters of $\check{\mathbf{q}}(t_0)$ and $\check{q}_B^{(0)}$, represent the linear constraints of (4.47)–(4.48), and (4.51) correspond to the quadratic norm constraints (4.33)–(4.34). The resulting equations (4.49)–(4.51) could either be solved directly, for instance by (iterative) linearization as previously described, or be linearized and included in a nonlinear estimator, such as an EKF. In either case, any number of motion constraints of the type (4.50) could be added to the measurement equation, with each measured $\check{\mathbf{q}}_B$ providing two independent constraints. This means that three known positions are sufficient to completely constrain the estimated pose. This can be compared to the problem of hand-eye calibration, where two motions are necessary for the calculation of the hand-eye transformation [Tsai and Lenz, 1989].

Dynamic Models for the State Estimation. We chose to investigate two different versions of the function $\mathbf{f}$ in the state equation (4.40). First, we assumed the dynamics

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \boldsymbol{\epsilon}_k, \tag{4.52}$$

where $\boldsymbol{\epsilon}_k$ was assumed to be an uncorrelated Gaussian noise sequence, and where the state vector was

$$\mathbf{x} = \begin{pmatrix} \mathbf{q} & \mathbf{q}' & f & \gamma & u_0 & v_0 \end{pmatrix}^T. \tag{4.53}$$

The second version was obtained by extending the state vector $\mathbf{x}_k$ with velocity $\vec{\mathbf{v}}_k \in \mathbb{R}^3$ and angular velocity $\vec{\boldsymbol{\omega}}_k \in \mathbb{R}^3$ to second-order dynamics, leading to the equations of motion

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega}\mathbf{q} = \frac{1}{2}(0, \vec{\boldsymbol{\omega}})\mathbf{q} \tag{4.54}$$

$$\dot{\mathbf{q}}' = \frac{1}{2}\dot{\mathbf{t}}\mathbf{q} + \frac{1}{2}\mathbf{t}\dot{\mathbf{q}} = \frac{1}{2}\mathbf{v}\mathbf{q} + \frac{1}{4}\mathbf{t}\boldsymbol{\omega}\mathbf{q}. \tag{4.55}$$

By discretizing Eqs. (4.54) and (4.55) using sample time h, we obtained the equation

$$\begin{pmatrix} \mathbf{q}_{k+1} \\ \mathbf{q}'_{k+1} \\ \mathbf{k}_{k+1} \\ \vec{\boldsymbol{\omega}}_{k+1} \\ \vec{\mathbf{v}}_{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \frac{h}{2}\mathbf{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \frac{h}{4}\mathbf{T}_k\mathbf{Q}_k & \frac{h}{2}\mathbf{Q}_k \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{q}_k \\ \mathbf{q}'_k \\ \mathbf{k}_k \\ \vec{\boldsymbol{\omega}}_k \\ \vec{\mathbf{v}}_k \end{pmatrix} \tag{4.56}$$

where $\mathbf{k} = (f, \gamma, u_0, v_0)^T$ was a vector of intrinsic camera parameters, and where the matrices

$$\mathbf{Q}_k = \begin{pmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & q_3 & -q_2 \\ -q_3 & q_0 & q_1 \\ q_2 & -q_1 & q_0 \end{pmatrix} \tag{4.57}$$

and

$$\mathbf{T}_k = \begin{pmatrix} 0 & -t_x & -t_y & -t_z \\ t_x & 0 & -t_z & t_y \\ t_y & t_z & 0 & -t_x \\ t_z & -t_y & t_x & 0 \end{pmatrix} \tag{4.58}$$

corresponded to the quaternion products with $\mathbf{q}_k = (q_0, q_1, q_2, q_3)$ and $\mathbf{t}_k = (0, t_x, t_y, t_z) = 2\mathbf{q}'_k\bar{\mathbf{q}}_k$ in Eqs. (4.54) and (4.55). With noise, Eq. (4.56) could be written as

$$\mathbf{x}_{k+1} = \mathbf{A}_k\mathbf{x}_k + \boldsymbol{\epsilon}_k. \tag{4.59}$$

The linearized system equations were used in an Extended Kalman Filter, together with the linearized measurement and constraint equations obtained from (4.49)–(4.51). The constraint equations were linearized around the predicted state, and included as (nearly) perfect measurements.

Experiments

The edge-based tracking algorithm, based on an EKF for the system models described above, was evaluated in experiments using the image generation software described in Appendix A.4. The simulated camera was mounted in an uncalibrated eye-in-hand configuration, viewing the stationary object. The object model consisted of a number of planar surfaces connected at their edges, as shown in Fig. 4.7. At each time step, visible object edges were selected using the BSP-tree algorithm and image edge positions were measured.

A comparison study was performed to investigate the difference between estimators with and without the constraints of Eqs. (4.47)–(4.48). The two cases are referred to as the *constrained case* and the *unconstrained case*, respectively.

Motion Tracking. Figures 4.8 and 4.9 show the results from a sequence, where tracking of the translation and orientation of the object was performed. The difference between the constrained and the unconstrained methods are more clearly seen from Table 4.1, which shows the

resulting estimation errors for a number of different cases. For Case 1 no extra noise was added to the measured output, apart from the naturally occurring noise from the image processing. For Case 2 extra noise $\delta \in N(0,3)$ was added in order to investigate the effects of poor image quality. Case 3 was the same as Case 1, but using the second-order dynamical model in Eq. (4.56) instead of the stationary model in Eq. (4.52). Case 4 was the same as Case 2, but again using the model (4.56) instead of model (4.52). The initial state covariance, $\mathbf{P}_0$, and state noise covariance, $\mathbf{Q}$, for Cases 1 and 2 were set to

$$\begin{aligned} \mathbf{P}_0 &= \text{diag}(0.1^2 \cdot \mathbf{1}_8, 50^2, 0.1^2, 40^2, 40^2) \\ \mathbf{Q} &= \text{diag}(0.1^2 \cdot \mathbf{1}_8, 3^2, 0.01^2, 0.2^2, 0.2^2). \end{aligned}$$

For Cases 3 and 4, the initial values were set to

$$\begin{aligned} \mathbf{P}_0 &= \text{diag}(0.1^2 \cdot \mathbf{1}_8, 50^2, 0.1^2, 40^2, 40^2, \mathbf{0}_6) \\ \mathbf{Q} &= \text{diag}(\mathbf{0}_8, 3^2, 0.01^2, 0.2^2, 0.2^2, 0.1^2 \cdot \mathbf{1}_6). \end{aligned}$$

The noise variance for the image measurements used in the EKF design was set to $E(\boldsymbol{\delta}_k^2) = \mathbf{I}$ in Cases 1 and 3 and to $E(\boldsymbol{\delta}_k^2) = 9\mathbf{I}$ in Cases 2 and 4. As seen from Table 4.1, the variance of the estimation error was reduced when using the constraints, due to the number of degrees of freedom being effectively reduced to four.

The results from a simulation, where the focal length was varied linearly from 250 to 150 during the motion sequence, can be seen in Fig. 4.10.

Table 4.1 Comparison between the estimation errors obtained through constrained (C) and unconstrained (U) estimation, for the described four cases with different dynamic models and measurement noise.

Case	1-C	1-U	2-C	2-U	3-C	4-C
$\Delta\theta_z$ (°)	0.122	0.216	0.292	0.541	0.120	0.184
$\Delta\theta_y$ (°)	0.317	0.313	0.529	0.632	0.335	0.346
$\Delta\theta_x$ (°)	0.130	0.381	0.221	0.777	0.140	0.136
Δt_x (mm)	2.105	3.406	2.434	3.310	2.456	2.086
Δt_y (mm)	2.390	2.093	2.322	2.878	2.456	1.867
Δt_z (mm)	4.857	4.926	5.704	7.228	5.148	5.178
$\|\Delta\boldsymbol{\theta}\|$ (°)	0.364	0.538	0.643	1.138	0.382	0.415
$\|\Delta\mathbf{t}\|$ (mm)	5.808	6.345	6.622	8.455	6.210	5.887

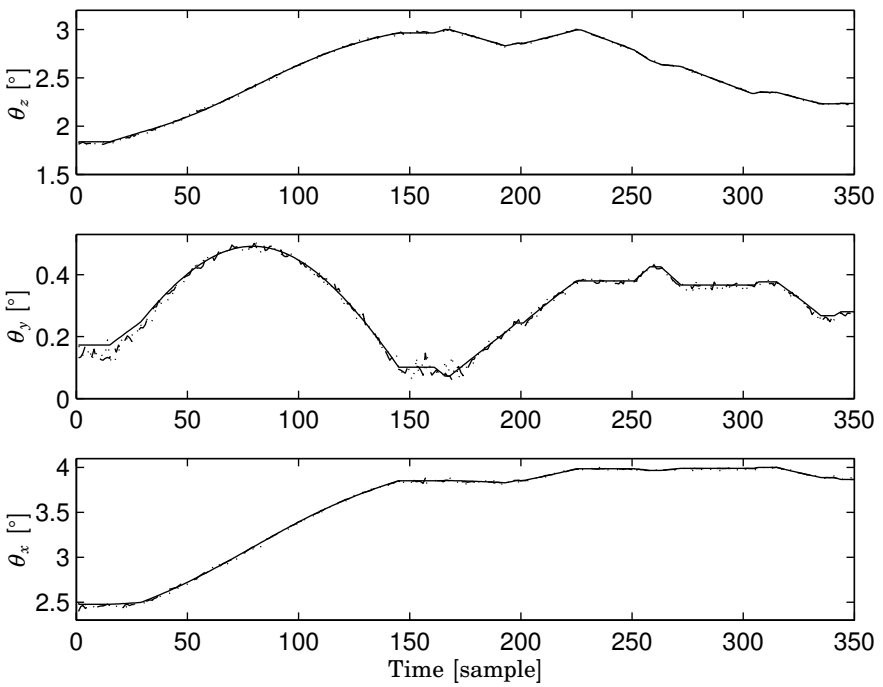

Figure 4.8 Tracking of the orientation, illustrated using the Euler angles $\boldsymbol{\theta}$. The diagram shows the real orientation (*solid*), estimated orientation using the constrained estimation (*dashed*), and estimated orientation without constraints (*dotted*).

The figure illustrates the tracking of the varying focal length (due to zoom), as well as the effect on the depth estimation. Using the constraints, the average errors in the estimated focal length and z-position during the motion sequence were reduced by more than 50%.

The use of the hand-eye constraints resulted in a small improvement in the estimation of the parameters, even though the hand-eye transformation was unknown. Additionally, the extra constraints improved the robustness of the tracking against other error sources, such as errors due to the edge detector locking on to false edges. The system was also capable of performing a total calibration of all relevant parameters, based on only rough initial values. The fact that accurate estimation of motion is possible also during changes in the intrinsic parameters is an advantage in vision-based control, since it allows the vision system to dynamically

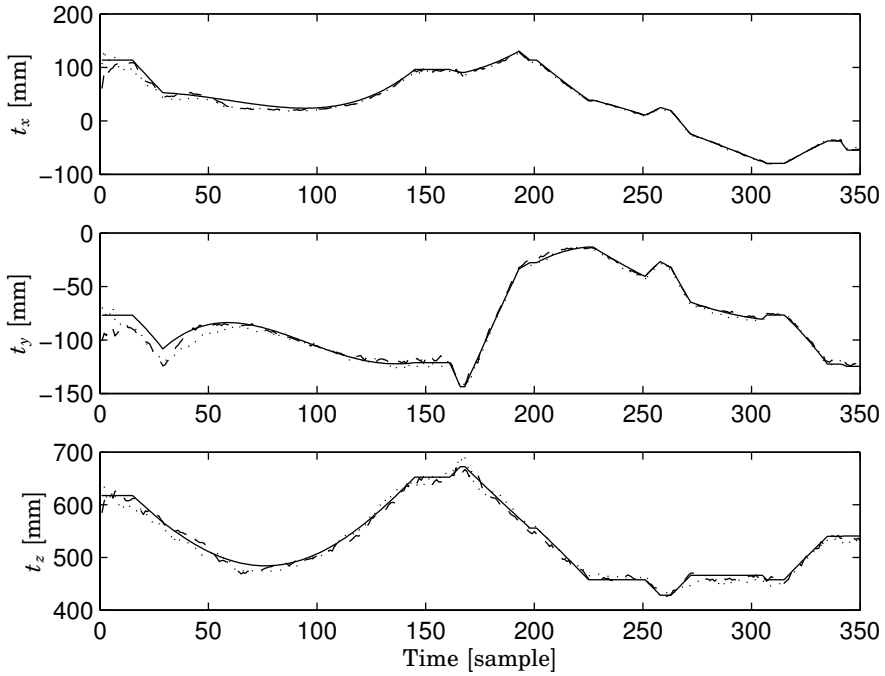

Figure 4.9 Tracking of the translation **t**. The diagram shows the real translation (*solid*), estimated translation using the constrained estimation (*dashed*), and estimated translation without constraints (*dotted*).

change its field of view by zooming in and out, allowing a wider range of motions.

Real World Experiments. Figure 4.11 shows the results of an experiment using images from a Sony DFW-V300 640x480 pixels digital camera mounted on an ABB Irb2000 industrial robot in an eye-in-hand configuration. The setup is shown in Fig. 4.6 and a camera image from the experiment can be seen in Fig. 4.7. The top figures in Fig. 4.11 show the estimated focal length and principal point, which should be compared to the values $f = 1020$, $u_0 = 344$ and $v_0 = 215$ obtained from an accurate off-line camera calibration. The lower figure shows the estimated position of the camera, where the lines indicate the direction of the camera optical axis.

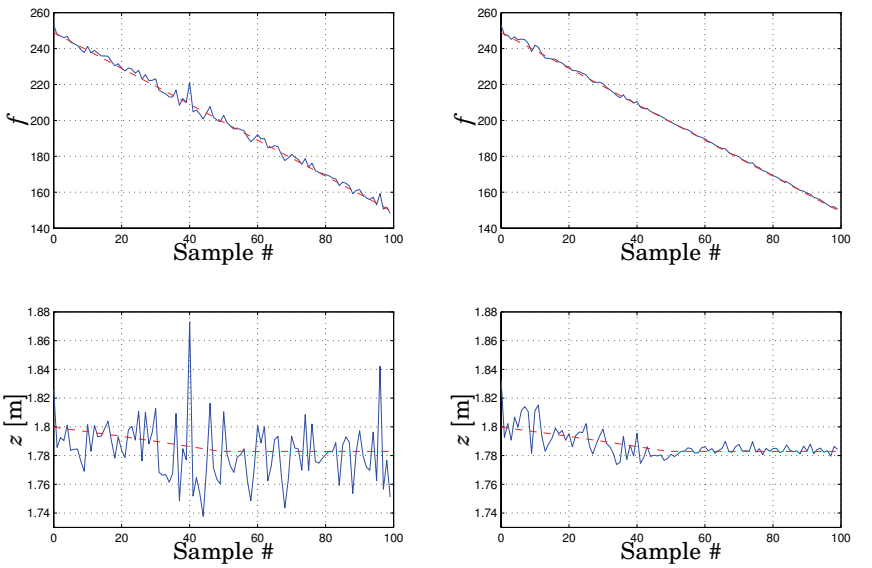

Figure 4.10 Experiment where the focal length f was varying between 250 and 150 during motion. True values are indicated by (*dotted*) lines, while estimated values are shown as (*solid*) lines. The left two plots show the resulting depth (z-motion) and focal length for the estimation without hand-eye constraints, and the right two plots show the corresponding results for the constrained estimation.

4.3 Multi-Camera Tracking with Resource Constraints

If the purpose of the visual motion estimation is feedback control, the timing properties of the algorithm become important. For real-time control applications in general, the importance of minimizing the input-output latency—the delay from the reading of the sensors to the actuation of the control output—is well-known. Unless compensated for, the input-output latency will compromise the performance of the control system, even to the point of causing instability. In most vision-based control systems the latency is dominated by the time required for image processing and estimation. This delay is therefore important to minimize, by the use of efficient algorithms as well as a proper choice of measurements.

Problem Formulation and State-of-the-Art. In many feature-based tracking algorithms, the number and character of the measured features are allowed to vary during the execution. The number of features used at a given time can depend directly on the number of features available, or be chosen freely by the algorithm itself. An example of the latter case

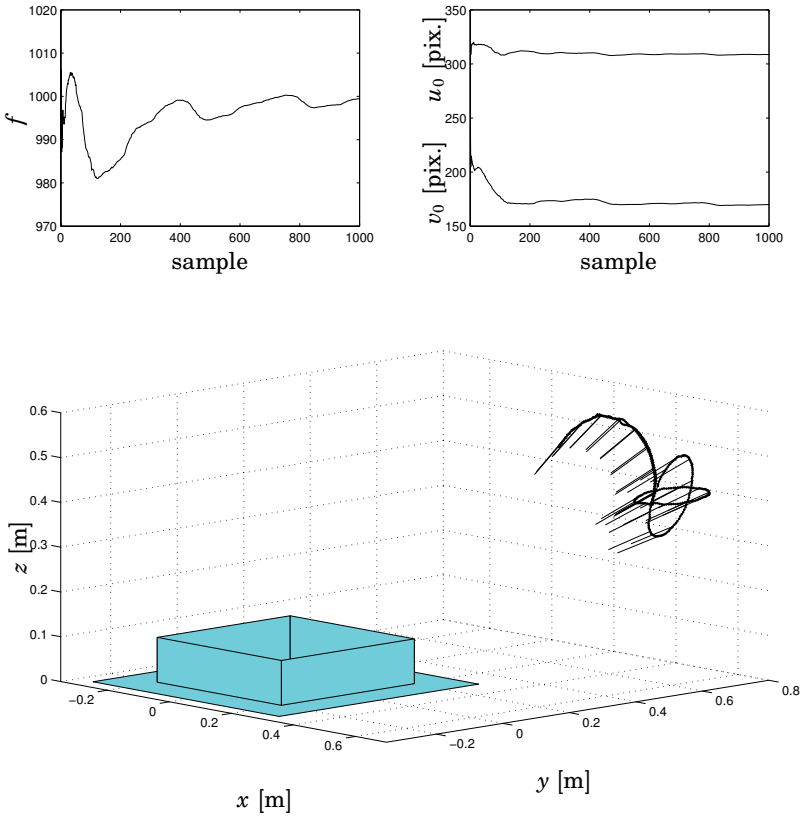

Figure 4.11 Estimated focal length, principal point, and trajectory in the real world experiment. A movie showing the camera motion can be found at http://www.control.lth.se/database/publications/article.pike?artkey=ols07dis.

is the edge-based tracker described in Section 4.2, where any number of edge searches can be performed along the edges of the object. This class of visual tracking algorithms are examples of *anytime algorithms* or imprecise computation algorithms [Henriksson, 2006]. In a visual tracking context, the consequence is that the trade-off between computation time and estimation accuracy may be influenced online, through some suitable algorithm. As an example of a vision application, a schedulability problem for a multi-camera system was treated in [Caccamo *et al.*, 2000]. By using several cameras with different settings and distance to the tracked target, it was suggested that the likelihood of the object moving out of the tracking region could be reduced. Another problem related to anytime algorithms, is that of active search for suitable image measurements in [Davison, 2005]. Methods based on information theory were used to guide a vision

system to the best point and edge features to measure, thereby tightly integrating the pose estimation and the image processing.

In this section, we consider the problem of online resource allocation for real-time tracking using several cameras. We consider the problem of optimizing the accuracy of the estimation obtained from a system of multiple cameras, given a maximum allowed computation time. This is obtained by a proper choice of active cameras and distribution of the feature points between these cameras. The proposed algorithm is based on the solution to a convex minimization problem, finding the optimal distribution of image processing resources over a heuristically chosen subset of all available cameras.

Estimation of Object Position and Orientation

We assume that M cameras are placed in fixed locations, viewing a target object whose position and orientation with respect to some fixed (world) coordinate system should be estimated. The pose is parametrized as an n-vector $\mathbf{z}$ as in Section 4.2. The pose $\mathbf{z}$ and the image-space feature position vector $\mathbf{y}$ are related by the projection equations of the cameras

$$\mathbf{y} = \mathbf{h}(\mathbf{z}), \tag{4.60}$$

in our case given by the homogeneous form pinhole camera projection equations (4.3)–(4.4) for the cameras, which can be stacked to form the projection equations for the multi-camera system. The measurement correction term $\Delta\mathbf{z}_k$ at sample k is computed from the image-space errors $\Delta\mathbf{y}_k$ by

$$\Delta\mathbf{z}_k = \mathbf{J}_v^\dagger(\hat{\mathbf{z}}_{k|k-1})\Delta\mathbf{y}_k \tag{4.61}$$

where $\hat{\mathbf{z}}_{k|k-1}$ is the prediction from the previous sample, and the matrix $\mathbf{J}_v^\dagger = (\mathbf{J}_v^T\mathbf{J}_v)^{-1}\mathbf{J}_v^T$ is the pseudo-inverse of the Jacobian $\mathbf{J}_v$ described in Section 4.2. Since this work primarily concerns edge features, the modified correction term becomes

$$\Delta\mathbf{z}_k = (\mathbf{N}^T\mathbf{J}_v(\hat{\mathbf{z}}_{k|k-1}))^\dagger\Delta\mathbf{e}_k = \mathbf{J}_{v,\mathbf{N}}^\dagger(\hat{\mathbf{z}}_{k|k-1})\Delta\mathbf{e}_k \tag{4.62}$$

where $\Delta\mathbf{e}_k$ are the edge distance measurements, and $\mathbf{N} \in \mathbb{R}^{N\times 2N}$ is a sparse matrix of normal directions at the N different measurement points along the edge, as illustrated in Fig. 4.1. When using edge features, there is a significant freedom in how to choose which features to measure, since any number of edge searches can be performed anywhere along any object edge in each camera. Finding the 'best' set of features to measure is a very difficult problem, and some simplifying assumptions are needed in order to solve the optimal feature selection problem in real-time.

Algorithm and Timing

The method for rigid body tracking from Section 4.2 was summarized in Algorithm 4.1. The image pre-processing step involves all necessary image conversions and spatial image filtering necessary for each camera. Based on the previously predicted position, the errors in edge positions are measured, and used together with the pre-calculated Jacobian to update the state estimation according to the correction in Eq. (4.62). The position for the next sample is predicted, and the predicted position is used to determine where interesting image features will be visible in the next sample. Visible features are determined, and a large number of search points are divided between the cameras using the algorithm described below, and placed along the predicted edges of the object. Finally, the Jacobian for each camera is computed.

The total computation time required in each sample depends on the number of cameras, M, and the total number of feature search points, N. The time required for pre-processing all images is proportional to the number of cameras used, whereas the total time for finding edges, placing search points, updating the estimation and calculating the Jacobians, is proportional to the total number of search points. The total time T_{tot} from sampling the cameras until the new estimation is obtained can therefore be modeled by the equation

$$T_{tot} = T_0 + T_c M + T_f N \tag{4.63}$$

where T_0 is a constant time required for image capture and image data transfer. The values of the time coefficients depends on many factors, such as camera sensor type and interface, camera shutter speed, platform, and implementation.

The implications of the timing model are twofold. First, as the computation time is deterministic, it can be compensated for by the control algorithm. Second, the relation between T_c and T_f shows a potential of gaining accuracy by switching off the processing of some cameras, thereby allowing a larger total number of feature search points to be distributed throughout the remaining active cameras. Assuming a desired computational delay T_{comp}, and M_k active cameras at sample k, the number of feature search points to distribute between the M_k cameras is given by

$$N_k = \frac{T_{comp} - T_o - T_c M_k}{T_f} \tag{4.64}$$

The delay T_{comp} should be chosen in relation to the dynamics of the controlled system and the closed-loop bandwidth. With a camera frame rate of 30 Hz, corresponding to a sample period of 0.033 s, simple rules-of-thumb

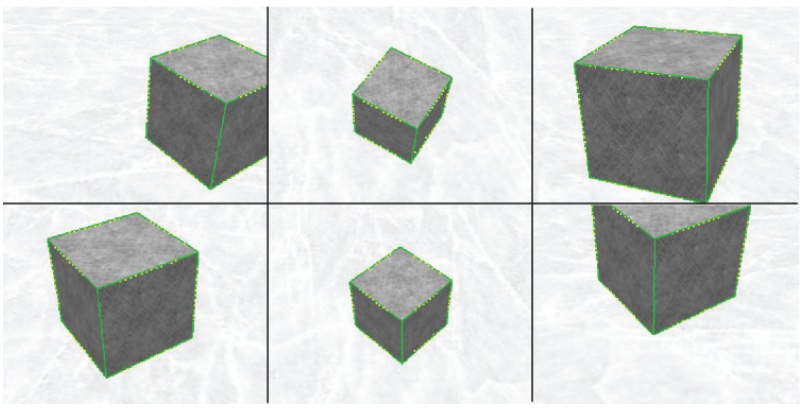

Figure 4.12 Example of six images from the simulated cameras with wireframe object superimposed, and search point locations indicated.

[Åström and Wittenmark, 1997] give that a realistic closed-loop bandwidth should lie between 6 and 18 rad/s. A delay of 15 ms would in that case correspond to a phase lag of 5–15 degrees, which can be compensated for without too much performance degradation. Using the estimated camera timing parameters in our implementation, $T_{comp} = 15$ ms corresponds to a total of 300 feature search points when using $M_k = 6$ active cameras, and 800 points when using only one camera. Thus, depending on the complexity of the scene and the dynamics of the control system, the relation between computational delay and the number of feature search points can be determined off-line.

Estimation Accuracy

In general, it is clear that the accuracy of the estimation will improve with the number of image measurements N. The estimation error depends on the current configuration and the distribution of the measurement errors in a complicated way. However, when N is large, the estimation error has a simpler, approximately Gaussian structure, as illustrated in Fig. 4.13. The figure was generated by creating synthetic images of 1500 small motions around a nominal position, and computing the errors in one rotational degree of freedom for each motion. Apart from a small configuration-dependent bias, caused by the image-processing, the non-biased Gaussian approximation is accurate. Accordingly, by assuming that the errors on the image measurements $\Delta\mathbf{e}_k$ are unbiased and can be modeled by Gaussian, independent noise with variance σ^2, the covariance of the effective

measurement error

$$\mathbf{z}_e = \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1} - \Delta\mathbf{z}_k \tag{4.65}$$

can be approximated as in Section 4.2 by

$$E[\mathbf{z}_e\mathbf{z}_e^T] = \sigma^2(\mathbf{J}_{v,\mathbf{N}}^T\mathbf{J}_{v,\mathbf{N}})^{-1} = \sigma^2(\sum_{i=1}^{M}\mathbf{J}_i^T\mathbf{J}_i)^{-1} \tag{4.66}$$

where the Jacobian has been partitioned into M individual Jacobians for each camera as $\mathbf{J}_{v,\mathbf{N}}^T = [\mathbf{J}_1^T\ \mathbf{J}_2^T\ \cdots\ \mathbf{J}_M^T]^T$. The Jacobian for each camera is a function of the current estimated position $\hat{\mathbf{z}}$, and depends on the number of search points N_i for that camera and their distribution in the image. Empirically, it has been found that if search points are distributed evenly along the visible edges of the object, the approximation

$$\mathbf{J}_i^T\mathbf{J}_i \approx N_i\mathbf{\Phi}_i(\hat{\mathbf{z}}) \tag{4.67}$$

holds with good accuracy, where $\mathbf{\Phi}_i$ is a positive semi-definite $n \times n$ matrix independent of N_i. This is illustrated in Fig. 4.14, where the diagonal elements of the resulting matrix $\mathbf{J}^T\mathbf{J}$ have been plotted for different numbers N of features, evenly distributed along the edges of a cubic object. Using (4.67) in (4.66) we get

$$E[\mathbf{z}_e\mathbf{z}_e^T] = \sigma^2\left(\sum_{i=1}^{M}N_i\mathbf{\Phi}_i(\hat{\mathbf{z}})\right)^{-1} \tag{4.68}$$

which shows that the covariance of the measurement error is a function of the number N_i of search points placed in each camera. As can be seen from Eq. (4.68), the estimation error is also a function the estimated object position $\hat{\mathbf{z}}$. In Eq. (4.68), the expression for the estimation accuracy has been explicitly separated into the dependence on the number of search points N_i, and the dependence on configuration dependant factors, as expressed by the positive semi-definite matrices $\mathbf{\Phi}_i(\hat{\mathbf{z}})$. For the single camera i, a criterion for the accuracy that can be obtained is given by the smallest eigenvalue $\underline{\lambda}(\mathbf{\Phi}_i)$ of $\mathbf{\Phi}_i(\hat{\mathbf{z}})$, which should ideally be as large as possible. In general, this is most likely to be true when many features of the object are visible, and when the object is close to the camera. Conversely, poorly conditioned situations occur when only part of the object is visible, or when the object is very far from the camera, when $\underline{\lambda}(\mathbf{\Phi}_i)$ becomes small or zero. One example is when all visible image features lie on a straight line, causing rotations of the object around this line to become unobservable from the image feature data. Frequently, no single camera provides sufficient information for a reliable estimate to be obtained, and suitably chosen measurements from several cameras must be combined, as described in the following sections.

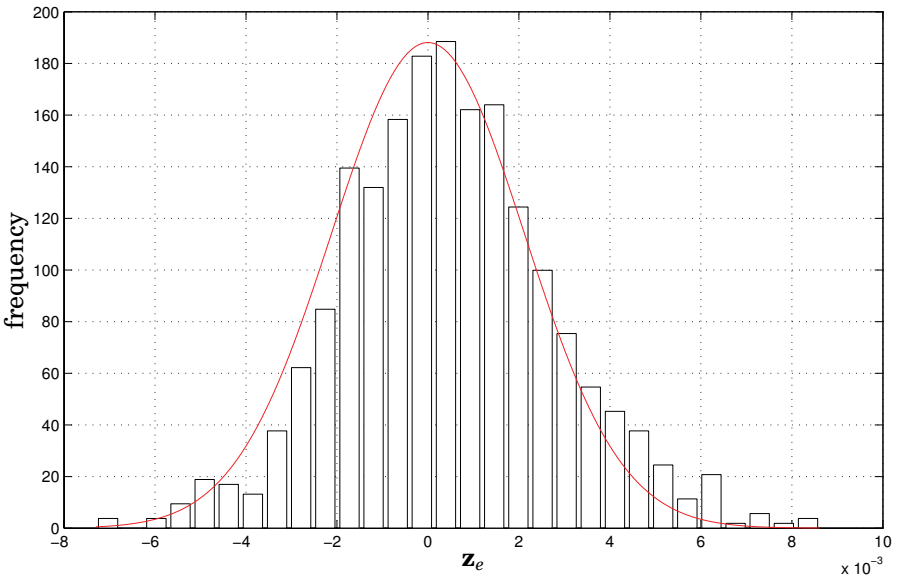

Figure 4.13 Histogram of the obtained effective measurement errors $\mathbf{z}_e$ in one degree of freedom, compared the histogram predicted by a Gaussian distribution corresponding to Gaussian image measurement noise with a standard deviation of 1.35 pixels.

Resource Allocation

In addition to the requirements for tracking accuracy, cameras need to be distributed to provide coverage of the entire workspace. Because of the limited resolution and field of view of each camera, it is usually beneficial to place the cameras so that each camera covers only a part of the available workspace. Some cameras may be positioned to cover a large part of the workspace, providing rough information on the location of the object, while other cameras cover only part of the workspace for a more accurate localization. If the object is moving, different cameras will provide more or less useful or accurate information at different times, depending on the current object position.

In general, the most accurate estimation of the position is obtained when using a subset of the available cameras. When timing is important, for instance when the estimated position is to be used for feedback control, it would be an advantage to use only the 'best' subset of cameras. The reason is the extra processing time required for each camera due to image pre-processing and other factors, giving less time for the feature extraction and other operations. In addition, each edge detection takes time,

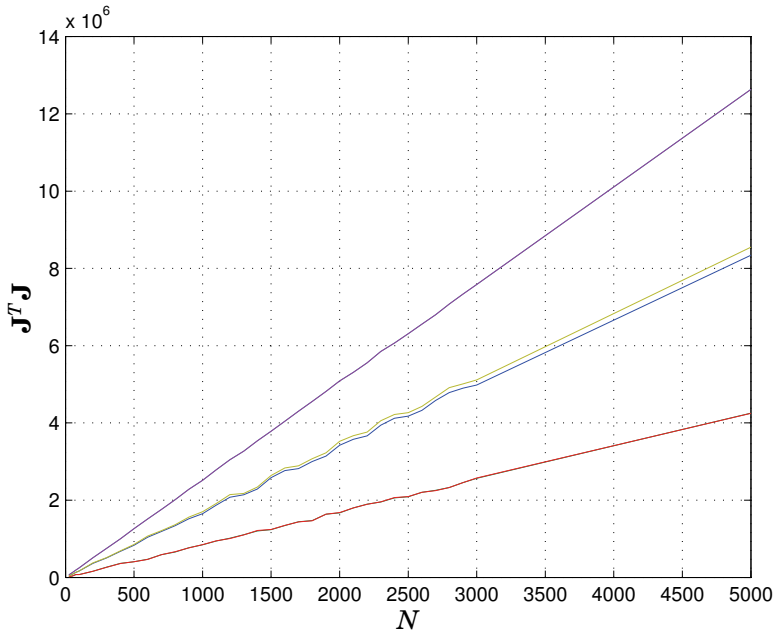

Figure 4.14 Diagonal elements of the matrix $\mathbf{J}^T\mathbf{J}$ as a function of N, with features evenly distributed along the edges of cubic object. The approximately linear dependence of the diagonal elements in the number of features N can be clearly seen, and the linearity holds well also for the non-diagonal elements.

and therefore the edge search point should be distributed only among the cameras in this 'optimal' set $\{C_k\}$. Using the covariance of the estimation error as a measure of the estimation accuracy, we see from Eq. (4.68) that for a given estimated object position $\hat{\mathbf{z}}$, we must choose the numbers of feature search points N_i for each camera in the set $\{C_k\}$ such that

$$E[\mathbf{z}_e\mathbf{z}_e^T] = \sigma^2 \left(\sum_{i \in \{C_k\}} N_i \mathbf{\Phi}_i(\hat{\mathbf{z}}) \right)^{-1} \tag{4.69}$$

is minimized with respect to some criterion. In general, the covariance decreases with increasing N_i, although finding the optimal camera set and search point distribution from Eq. (4.69) is a non-trivial task. Heuristic choices are possible—such as using the best individual camera or using all cameras—but can be arbitrarily far from the optimum, or may not even be feasible due to the timing constraints.

Optimization Problem. The solution to the problem of achieving the best possible measurement error covariance given a desired maximum latency T_{tot} can be expressed as an optimization problem

$$\text{minimize} \quad f_o(\mathbf{r}) = \bar{f}_o\left(\left(\sum_{i\in\{C_k\}} r_i\mathbf{\Phi}_i\right)^{-1}\right) \tag{4.70}$$

$$\text{subject to} \quad \mathbf{r} \geq \mathbf{0} \tag{4.71}$$

$$T_f\mathbf{1}^T\mathbf{r} + T_c\mathbf{1}^T\theta(\mathbf{r}) \leq T_{tot} - T_0 \tag{4.72}$$

in the numbers $\mathbf{r}$ of feature points in each camera, where $\bar{f}_o$ is a function expressing the desired properties of the covariance matrix to be optimized. $\theta(\mathbf{r})$ is the step function, defined element-wise by the property

$$\begin{cases} \theta_i(\mathbf{r}) = 1, & r_i \geq 0 \\ \theta_i(\mathbf{r}) = 0, & r_i < 0. \end{cases} \tag{4.73}$$

In the case when the function $\bar{f}_o$ is the matrix 2-norm, the function

$$f_o(\mathbf{r}) = \left\| \left(\sum_{i\in\{C_k\}} r_i\mathbf{\Phi}_i)\right)^{-1} \right\| \tag{4.74}$$

can be proved to be convex, that is, that for all $\mathbf{x} \geq \mathbf{0}$, $\mathbf{y} \geq \mathbf{0}$ and $\alpha \in [0,1]$ it holds that

$$f_o(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f_o(\mathbf{x}) + (1-\alpha) f_o(\mathbf{y}). \tag{4.75}$$

The convexity can be proved by first defining the positive definite matrices

$$\mathbf{X} = \sum_{i\in\{C_k\}} x_i\mathbf{\Phi}_i \tag{4.76}$$

$$\mathbf{Y} = \sum_{i\in\{C_k\}} y_i\mathbf{\Phi}_i \tag{4.77}$$

which can be assumed to be positive definite. Then, the inequality in Eq. (4.75) can be written

$$\left\|(\alpha\mathbf{X} + (1-\alpha)\mathbf{Y}))^{-1}\right\| \leq \alpha \left\|\mathbf{X}^{-1}\right\| + (1-\alpha)\left\|\mathbf{Y}^{-1}\right\| \tag{4.78}$$

and since for a positive definite matrix $\mathbf{Z}$ is holds that

$$\left\|\mathbf{Z}^{-1}\right\| = \frac{1}{\underline{\lambda}(\mathbf{Z})} \tag{4.79}$$

with $\underline{\lambda}(\mathbf{Z})$ denoting the smallest eigenvalue of $\mathbf{Z}$, Eq. (4.78) can be rewritten as the equivalent condition

$$\underline{\lambda}\left(\alpha\mathbf{X} + (1-\alpha)\mathbf{Y}\right) \geq \frac{1}{\frac{\alpha}{\underline{\lambda}(\mathbf{X})} + \frac{1-\alpha}{\underline{\lambda}(\mathbf{Y})}} = \frac{\underline{\lambda}(\mathbf{X})\underline{\lambda}(\mathbf{Y})}{\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X})} \tag{4.80}$$

which can be proved by

$$\begin{aligned}
&\underline{\lambda}\left(\alpha\mathbf{X} + (1-\alpha)\mathbf{Y}\right) \geq \alpha\underline{\lambda}(\mathbf{X}) + (1-\alpha)\underline{\lambda}(\mathbf{Y}) = \\
&= \frac{(\alpha\underline{\lambda}(\mathbf{X}) + (1-\alpha)\underline{\lambda}(\mathbf{Y}))(\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X}))}{\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X})} = \\
&= \frac{\alpha(1-\alpha)(\underline{\lambda}(\mathbf{X})^2 + \underline{\lambda}(\mathbf{Y})^2 - 2\underline{\lambda}(\mathbf{X})\underline{\lambda}(\mathbf{Y})) + \underline{\lambda}(\mathbf{X})\underline{\lambda}(\mathbf{Y})}{\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X})} = \\
&= \frac{\alpha(1-\alpha)(\underline{\lambda}(\mathbf{X}) - \underline{\lambda}(\mathbf{Y}))^2 + \underline{\lambda}(\mathbf{X})\underline{\lambda}(\mathbf{Y})}{\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X})} \geq \\
&\geq \frac{\underline{\lambda}(\mathbf{X})\underline{\lambda}(\mathbf{Y})}{\alpha\underline{\lambda}(\mathbf{Y}) + (1-\alpha)\underline{\lambda}(\mathbf{X})}.
\end{aligned} \tag{4.81}$$

A small technical issue in the proof concerns the assumption of positive definiteness of $\mathbf{X}$ and $\mathbf{Y}$ in Eq. (4.76)–(4.77), although each $\mathbf{\Phi}_i$ is only guaranteed to be positive *semi-definite* and $x_i \geq 0$. However, as long as the camera set $\{C_k\}$ is chosen such the full image Jacobian has full rank (i.e., all motions can be observed from the image data), the problem can be avoided by imposing the extra condition that $x_i \geq x_\delta$ for all cameras in $\{C_k\}$, for some small $x_\delta > 0$. The problem of finding such a set $\{C_k\}$ of cameras is discussed below.

Although the objective function $f_o(\mathbf{r})$ is convex, the presence of the non-convex constraint function $\theta(\mathbf{r})$ makes the optimization problem given by Eqs. (4.70)–(4.72) non-convex. A practical solution to the problem could therefore be divided into two parts. In the first part, a suitable set of active cameras is determined, based on the structure of the matrices $\mathbf{\Phi}_i$, and knowledge of the individual image noise properties. When M is large, a solution sufficiently close to the optimum can usually be found by only considering active camera sets with a small number of cameras, which sometimes makes it possible to find the active camera set by an exhaustive search through all possible such combinations. An improved algorithm

for finding the active camera set is described below. Once the set has been found, the second problem corresponds to finding a distribution of features between the cameras, which solves the resulting convex optimization problem. Since the camera set is given by the camera selection algorithm, the exact value of the non-convex function $\theta(\mathbf{r})$ in Eq. (4.72) is known and can be treated as a constant.

Algorithm for Real-Time Resource Allocation. Since timing is important, a fast algorithm for selecting a suitable camera set and feature distribution has been developed. The algorithm is outlined in Algorithm 4.2.

ALGORITHM 4.2—REAL-TIME RESOURCE ALLOCATION

```
N_feat[N_cam] = number of edge detection points using N_cam cameras;
for i = all cameras
  Φ[i] = transpose(J[i])*J[i]/N[i];
clear set sel of selected cameras and distribution of edge detection points;
set Φ_max = 0, best_val = inf, N_cam = 1, stop_flag = false;
while stop_flag == false {
  for k = all cameras not in sel
    find the α∈[0,1] minimizing new_val[k] = inv(α*Φ_max+(1-α)*Φ[k]);
  set best_new_cam = the camera cam with smallest new_val[cam];
  set best_new_ratio = the minimizing α for camera best_new_cam;
  set Φ_new = best_new_ratio*Φ_max + (1-best_new_ratio)*Φ[best_new_cam];
  if (inverse(N_feat[N_cam]*Φ_new) < best_val) {
    add best_new_cam to sel;
    redistribute edge detection points according to best_new_ratio;
    set Φ_max = Φ_new, best_val = inv(N_feat[N_cam]*Φ_new);
    set N_cam = N_cam + 1;
  } else stop_flag = true;
}
Solve the convex optimization problem (4.70)-(4.72) for {C_k}=sel;
```

□

The algorithm updates the active set of cameras and the preliminary distribution of edge search points among the active cameras. This is done by testing if it is possible to decrease the estimation error covariance by adding a camera, and redistributing the search points between the current active set and the added camera. If the covariance can be decreased[4], the active set and distributions are updated with the new camera, and the

[4]In the comparisons between covariance matrices, the criterion chosen for determining which matrix is largest is the ordinary matrix 2-norm.

algorithm tries to decrease the covariance iteratively by adding another camera. If nothing can be gained by adding another camera to the active camera set, the algorithm will find the final feature distribution by solving the convex optimization problem in Eqs. (4.70)–(4.72). The optimization is performed with $\{C_k\}$ as the calculated active camera set, and using the preliminary feature distribution as the starting point for the optimization. In practice, the preliminary active camera set and calculated feature distribution from the camera selection algorithm can also be used directly without the final optimization, which was the solution used in the simulations.

The algorithm is very robust and easy to implement. It takes negligible time to execute, since all information about the relative accuracy of the cameras is compressed into the small $n \times n$-matrices $\mathbf{\Phi}_i$. Due to the sub-optimal selection algorithm for the active camera set, the algorithm is not guaranteed to achieve the optimal covariance. However, in the majority of cases it will find a small subset of cameras which results in a significantly lower covariance than for the heuristic choices, as seen from the experiments.

Simulations

The algorithm is evaluated in simulations using six cameras, using the image generation software described in Appendix A.4. The object being tracked is a textured box, which is moved around in front of a textured background. Fig. 4.12 shows example images taken from an experiment sequence. We have assumed the timing model of Eq. (4.63), the sampling period $h = 33$ ms and a maximum desired control delay $T_{comp} = 15$ ms.

Tracking Accuracy. The tracking accuracy using a stationary target was evaluated by measuring the estimation error variance for different image sequences, taken from different camera positions. Three different algorithms for resource allocation between the cameras were investigated.

1. Choosing the best single camera, i.e., the camera i for which $\underline{\lambda}(\mathbf{\Phi}_i)$ is largest.
2. Using all cameras, with an equal distribution of edge search points.
3. Choosing the best set of cameras, using the algorithm in Fig. 4.2.

The accuracy was measured for image sequences taken with several different camera configurations as given by Table 4.2. The resulting standard deviations for the error in the estimated orientation and translation is shown in Table 4.3. The estimation using the single-camera method did not converge for Sequence 2, since the problem became very poorly conditioned for any choice of a single camera. In Sequence 1, the minimum

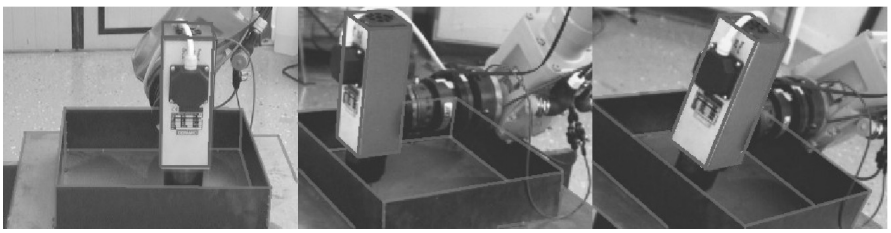

Figure 4.15 Simultaneous resource-optimal tracking of tool and workpiece, using two Sony DFW-V300 and one Basler A602fc digital cameras. A movie showing the motion, selected cameras, and features can be found at http://www.control.lth.se/database/publications/article.pike?artkey=ols07dis.

translation error was obtained by using all cameras, but the price was a significantly larger orientation error.

The algorithm was also tested in real-time on real image data from three cameras, as seen in Fig. 4.15. Despite a somewhat cluttered scene with multiple parallel edges, the tracking is successful.

Control Performance. The visual feedback was applied in a feedback control setting, where the textured box was controlled one-dimensionally along the y coordinate axis. The estimated y-position was used as feedback information to the controller. The simulated dynamics was described by a second order system, which after discretization with the sampling interval

Table 4.2 Camera configurations, with each row corresponding to one numbered scenario. Shown are the distances z (in mm unless otherwise indicated) from camera to target, and the rough percentage (%) of the object which was visible in each of the six cameras.

#	Camera 1		Camera 2		Camera 3		Camera 4		Camera 5		Camera 6	
	z	%	z	%	z	%	z	%	z	%	z	%
1	840	100	840	100	880	100	890	100	870	100	910	100
2	320	20	4 m	100	4 m	100	445	20	5 m	100	4 m	100
3	510	100	4 m	100	5 m	100	500	0	6 m	100	7 m	100
4	550	30	360	10	330	5	450	10	380	5	280	5
5	600	100	600	100	400	0	600	100	1 m	0	1 m	0
6	650	100	280	30	450	0	700	0	900	0	940	0
7	300	30	330	35	500	70	700	0	1 m	0	1 m	0
8	400	30	350	30	280	20	400	40	370	15	300	25

$h = 33$ ms was given by

$$\begin{aligned} \mathbf{x}_{k+1} &= \begin{bmatrix} 1 & 0.033 \\ 0 & 0.97 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} 0.55 \\ 32.8 \end{bmatrix} \mathbf{u}_k \\ \mathbf{z}_k &= \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}_k \end{aligned} \tag{4.82}$$

The controller was an LQG-controller, designed to maximize a continuous-time cost function

$$J(u) = \int_0^\infty \left(\begin{bmatrix} \mathbf{x}^T(t) & \mathbf{u}^T(t) \end{bmatrix} \mathbf{Q} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{bmatrix} \right) dt \tag{4.83}$$

with

$$\mathbf{Q} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.001 \end{bmatrix} \tag{4.84}$$

As seen by the process model in Eq. (4.82), only the first state was measurable. Therefore a Kalman filter was designed to reconstruct the state vector. The state and output noise variances used in the design of the Kalman filter were chosen as $\mathbf{R}_1 = 10\mathbf{I}$ and $\mathbf{R}_2 = 0.01$. The real-time simulations of the control system were performed in MATLAB/Simulink using the TrueTime simulator [Henriksson *et al.*, 2002]. This simulator allows for co-simulation of continuous plant dynamics and discrete controllers

Table 4.3 Tracking error standard deviations, orientation and translation, when using the best single camera, all cameras and the best selection of cameras, respectively.

#	Orientation error [°]			Translation error [mm]		
	Single	All cam.	Best sel.	Single	All cam.	Best sel.
1	0.18	0.36	0.18	0.75	0.49	0.68
2	∞	0.49	0.17	∞	0.37	0.25
3	0.12	1.23	0.10	0.26	1.64	0.21
4	0.22	0.20	0.16	0.97	0.28	0.26
5	0.12	0.29	0.10	0.35	0.36	0.31
6	0.14	0.20	0.08	0.36	0.46	0.17
7	0.22	0.25	0.07	1.10	0.28	0.16
8	0.08	0.09	0.05	0.24	0.14	0.12

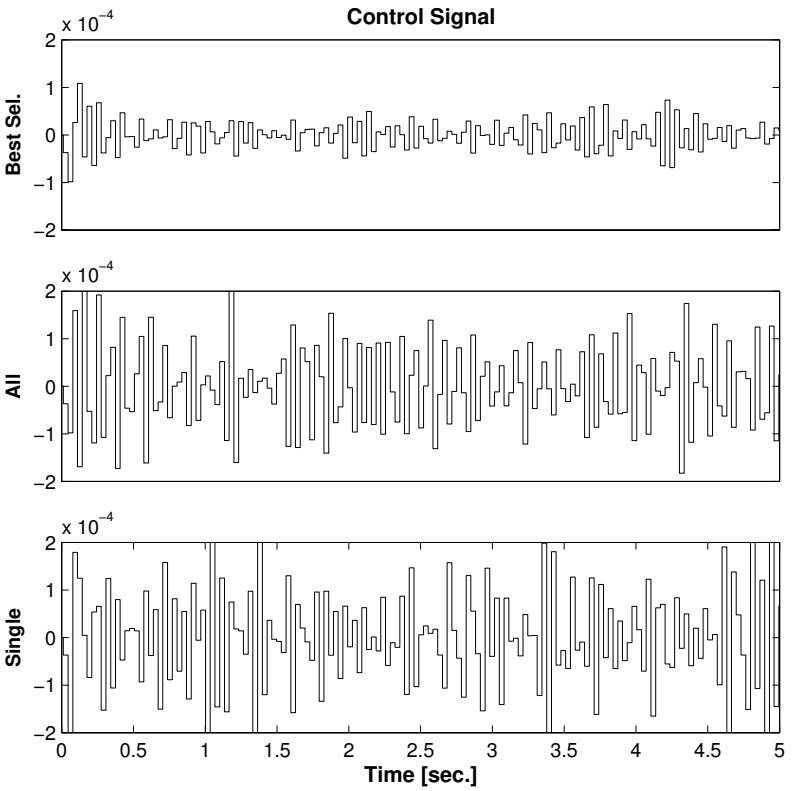

Figure 4.16 Control signal using the three different tracking algorithms in scenario #6 of Table 4.2.

implemented as tasks in a computer. The controller was implemented as a periodic task with the sampling interval 33 ms. In the beginning of the sample, only the images from the cameras in the active set were read and processed. The position estimate was then fed into the LQG-control algorithm, after which the computed control signal was actuated. A simulation corresponding to the configuration on scenario #6 in Table 4.2 was performed, and the objective of the control was steady-state regulation of the position around $\mathbf{z} = 0$, with no external load disturbances. Simulation results are shown in Figs. 4.16–4.17, showing the control signal and the controlled position. It is seen that the suggested resource allocation algorithm results in better control performance than the heuristic choices for this scenario. Similar results were obtained for other scenarios, although

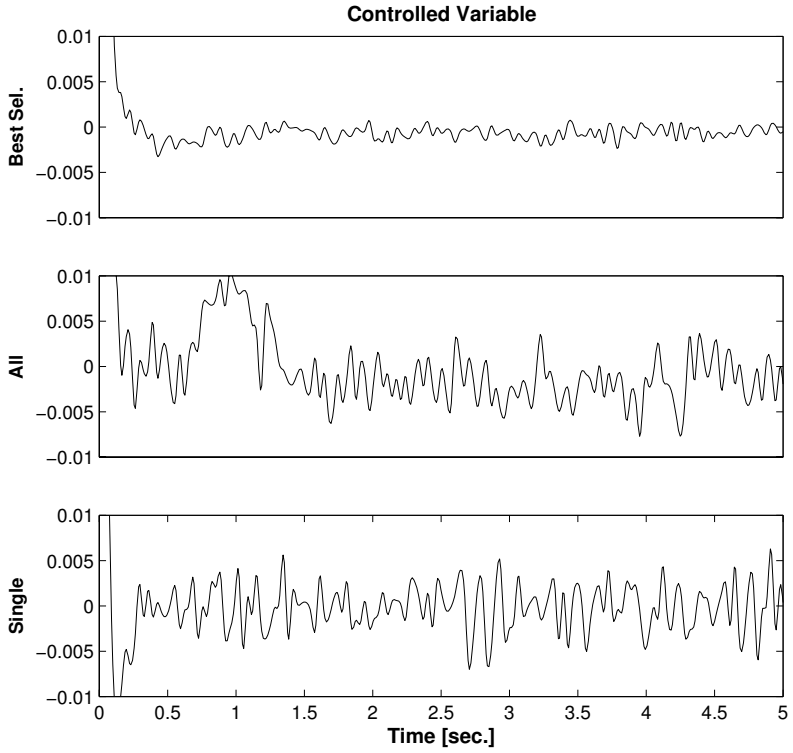

Figure 4.17 The controlled output using the three different tracking algorithms in scenario #6 of Table 4.2.

in certain scenarios one of the heuristic methods came close to the performance of the resource allocation, as indicated by the tracking accuracies in Table 4.3.

Discussion. As can be seen from Table 4.3, the proposed method for choosing the best camera set works well for all camera configurations in Table 4.2. The heuristic selection of the best single camera works well in situations where we can find a single camera that gives sufficient information, but will fail in many common configurations such as the one in Sequence 2. Using all cameras works well in many situations, but this is rarely necessary and it may not even be feasible if there are a large number of available cameras. The best selection algorithm will find a small camera set, typically consisting of 1–3 cameras, that will give sufficient information to accurately and robustly estimate the position and orienta-

tion.

The assumed model of the image noise as independent and Gaussian is usually not realistic, since there is a spatial correlation between the measurements, particularly for large N_i when the distance between search points will be small. In addition, outliers are an inevitable result of the image processing, and should be removed or re-weighted as discussed in Section 4.2. Other approximations involve the timing model in Eq. (4.63) and the approximation of the covariance given from Eq. (4.67). Despite the presence of such phenomena, the resource allocation algorithm will still generally outperform the heuristic methods.

4.4 Position-Based Force/Vision Control

By combining a version of the force controller described in Section 3.4 with feedback from the motion tracker in Section 4.2, a straightforward way to implement a force/vision control was obtained. The block diagram for the system under force/vision control was shown in Fig. 4.18. The desired trajectory of the tool was defined relative to the target object, whose position was estimated from the image data. The velocity-controlled robot system from Chapter 3 was controlled using the hybrid control law

$$\mathbf{v}_c = \mathbf{S}_v \mathbf{L}(\mathbf{x}_r - \hat{\mathbf{x}}) + \mathbf{S}_f \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{x}_I, \tag{4.85}$$

where $\mathbf{v}_c$ was the Cartesian velocity reference to the built-in robot motion control. $\mathbf{S}_v$ and $\mathbf{S}_f$ were selection matrices for the force/vision control, whose diagonal elements were set to 1 or 0. The state feedback $\mathbf{L}$ for the reference following was designed as a stationary LQ controller. The force controller dynamics were given by

$$\begin{aligned} \frac{d\mathbf{x}_I}{dt} &= \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & -\mathbf{M}_I^{-1}\mathbf{D}_I \end{bmatrix} \mathbf{x}_I + \begin{bmatrix} \mathbf{0} \\ \mathbf{M}_I^{-1} \end{bmatrix} \mathbf{S}_f(\mathbf{f} - \mathbf{f}_r) = \\ &= \mathbf{F}_I \mathbf{x}_I + \mathbf{G}_I \mathbf{S}_f(\mathbf{f} - \mathbf{f}_r), \end{aligned} \tag{4.86}$$

with $\mathbf{M}_I$ and $\mathbf{D}_I$ the desired inertia and damping matrices. The state estimate $\hat{\mathbf{x}}$ from the visual tracker was based on edge measurements and a discretized decoupled dynamic model with system matrices

$$\mathbf{F} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & -\omega\mathbf{I} \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \mathbf{0} \\ \omega\mathbf{I} \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}. \tag{4.87}$$

The estimator was designed as a Kalman filter, using a noise covariance matrix of the form given by Eq. (4.22), with $\sigma = 1$ pixel.

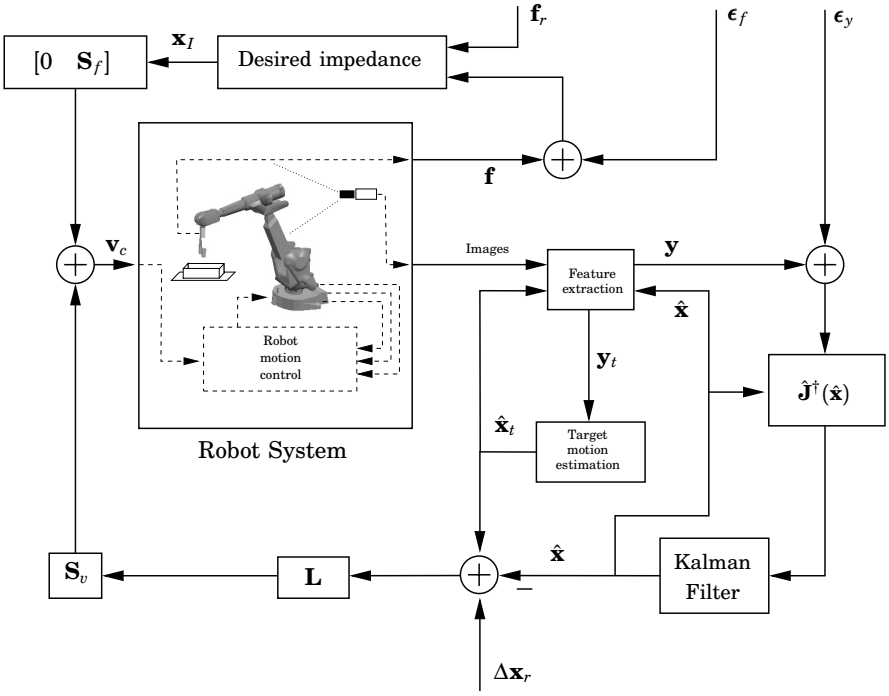

Figure 4.18 Block diagram showing the structure of the combined force/vision control system. Inputs were the reference signals $\Delta\mathbf{x}_r$ and $\mathbf{f}_r$ and the noises $\boldsymbol{\epsilon}_y$ and $\boldsymbol{\epsilon}_f$. The motion of the target was estimated using a nonlinear least-squares estimator.

Experiments were performed using an ABB Irb2400 industrial robot equipped with a rolling tool, in contact with a metal box with dimensions $40 \times 40 \times 10$ cm. Experiments were first performed using only two Sony DFW-V300 digital cameras, and later repeated with an additional camera. The setup used was nearly identical to the experimental setup in Section 4.3, illustrated in Fig. 4.15. In the experiments, the visual tracker was sampled and executed at 33 ms, while the force controller in Eq. (4.86) was discretized at a sampling period of 4 ms. The force controller was executed on the external PowerPC G4 processor described in Chapter 3. The image processing, and calculation and inversion of the image Jacobian were handled by a separate 2 GHz Pentium 4 computer, which communicated with the controller on the PowerPC using standard Ethernet. The BSP tree hidden-line removal technique was used to predict locations of visible edges in the next set of images, using an object model consisting of a number of planar surfaces connected at their edges.

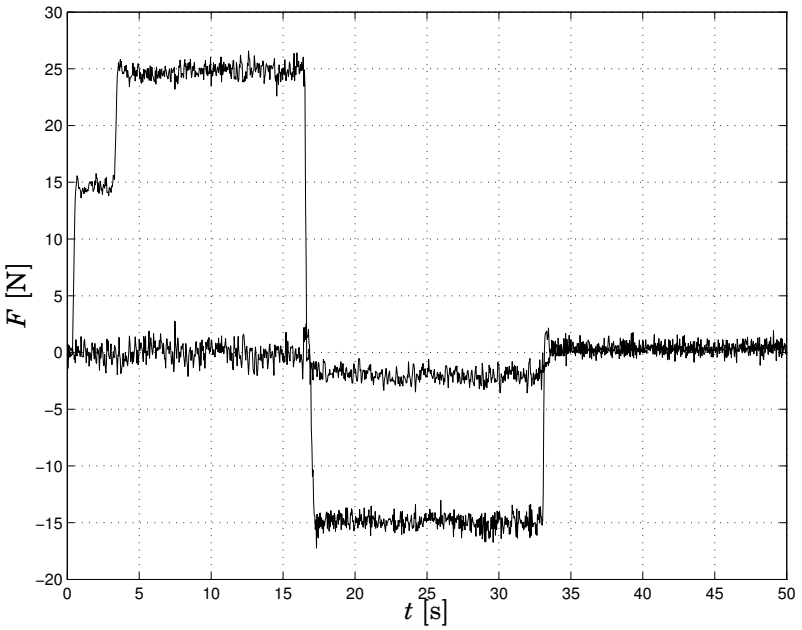

Figure 4.19 Measured contact force during vision-guided force control. The force reference was first changed from 15 N to 25 N in the x-direction, and finally to 15 N in the negative y-direction.

In the experiments, the robot made stable contact with the workpiece under vision-guided impedance control. When contact had been established, the system was switched to force/vision control where the tool was programmed to move across the surface with a speed of 10 mm/s. The resulting contact force can be seen in Fig. 4.19. At time $t = 3$ s the force reference was immediately changed from 15 N to 25 N in the x-direction of the tool. At time $t = 17$ s the tool reached the corner in the workpiece, and the force reference changed to 15 N in the negative y-direction. The combined stiffness of the robot and surface was approximately 10 N/mm, and the controller parameters were chosen as $M_I = 0.1$, $D_I = 1.5$ and $K_I = 0$. Fig. 4.20 shows the estimated position of the tool in the target frame, and Fig. 4.21 shows the corresponding estimated velocities. The system was able to follow low-speed trajectories with an accuracy of around 1 mm, while accurately controlling the contact force. The force controller achieved force tracking with rise times faster than 0.2 s, which meant that the force controller could quickly compensate for deviations from the nominal geometry.

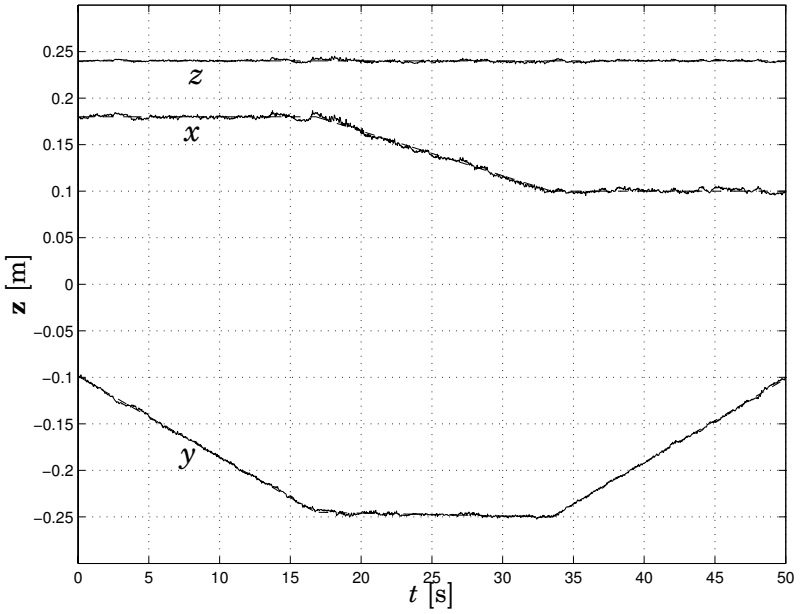

Figure 4.20 Estimated tool position (solid) and reference trajectory (dashed) during vision-guided force control.

4.5 Image-Based Force/Vision Control

When removing the pose estimation step, and instead closing the feedback loop from the image features directly, an image-based force/vision approach results. To this purpose, we assume a dynamic model for a velocity-controlled robot

$$\dot{\mathbf{z}} = \mathbf{f}(\mathbf{u}) \tag{4.88}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{z}) + \boldsymbol{\epsilon} \tag{4.89}$$

where $\mathbf{z}$ is a parametrization of the robot end-effector position, $\mathbf{u}$ is a vector of the desired translational and angular velocity (a *velocity screw*), and $\mathbf{y}$ is a vector of image feature measurements. In an image-based visual servo, the control error is defined directly in image space as $\mathbf{y}_r - \mathbf{y}$, where $\mathbf{y}_r$ is the desired position of the image features. A simple control law that would drive $\mathbf{y} \rightarrow \mathbf{y}_r$ is given by

$$\mathbf{u} = k_v \mathbf{J}_f(\mathbf{z})^{\dagger}(\mathbf{y}_r - \mathbf{y}), \tag{4.90}$$

where k_v is a constant gain, and $\mathbf{J}_f(\mathbf{z})^{\dagger}$ is the pseudo-inverse of the stereo

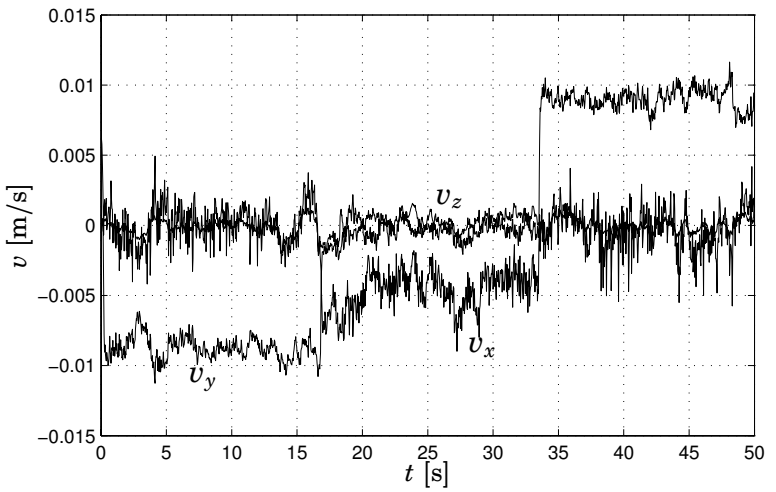

Figure 4.21 Estimated tool velocity during vision-guided force control, with respect to the target frame.

image Jacobian computed from Eq. (4.6). This Jacobian relates the image-space velocities $\dot{\mathbf{y}}$ to the corresponding end-effector velocity in Cartesian space

$$\dot{\mathbf{y}} = \mathbf{J}_f(\mathbf{z})\mathbf{u}. \tag{4.91}$$

From the form of Eq. (4.6) it can be seen that the image Jacobian is a function of the Cartesian coordinates $\mathbf{z}$, or more specifically the depth of each point in the camera. The depth information could be obtained from the stereo images. Some care needs to be taken however, as there are situations where the robustness to depth estimation errors becomes extremely poor [Malis and Rives, 2003]. When using stereo (or multiple) cameras, the combined Jacobian for the stereo system is obtained by stacking the Jacobians for the individual cameras [Martinet and Cervera, 2001]

$$\mathbf{J}_f(\mathbf{z}) = \begin{bmatrix} \mathbf{J}_f^{(l)}(\mathbf{z})\mathbf{M}_{bl} \\ \mathbf{J}_f^{(r)}(\mathbf{z})\mathbf{M}_{br} \end{bmatrix} \tag{4.92}$$

where $\mathbf{M}_{bl}$ and $\mathbf{M}_{br}$ are the transformation matrices for the velocity screw to the respective camera coordinate systems, and $\mathbf{J}_f^{(l)}(\mathbf{z})$ and $\mathbf{J}_f^{(r)}(\mathbf{z})$ are the Jacobians for the left and right cameras. The screw transformation matrix for the cameras is given by

$$\mathbf{M}_{bi} = \begin{bmatrix} \mathbf{R}_{bi} & [\mathbf{t}_{bi}]_\times \mathbf{R}_{bi} \\ \mathbf{0}_{3\times 3} & \mathbf{R}_{bi} \end{bmatrix}, \quad i \in \{l, r\} \tag{4.93}$$

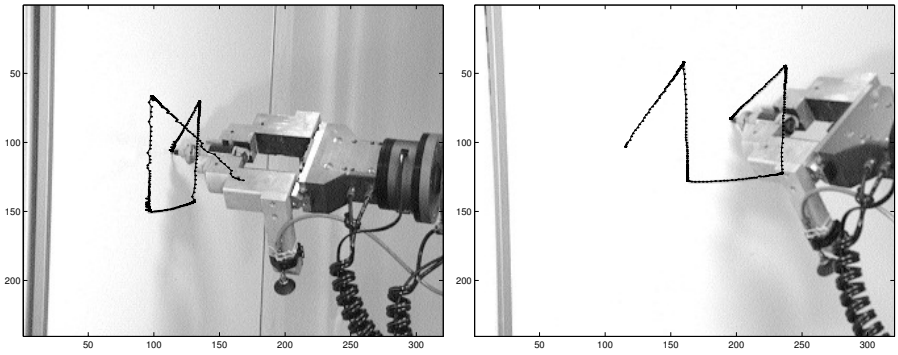

Figure 4.22 Pen trajectories in the image planes, during the approach and drawing phases.

where rotation and translation $\mathbf{R}_{bi}$ and $\mathbf{t}_{bi}$ represent the pose of camera i in world coordinates, and $[\cdot]_\times$ denotes the skew-symmetric matrix corresponding to the cross-product.

Example—Vision-Guided Drawing

As an example of image-based force/vision control, a simple and illustrative task was chosen. Two objects were placed in the field of view of both cameras, a pen and a white-board. The exact position and orientation of the objects were unknown. On the white-board, a number of dots were drawn in random positions. The objective of the control was to align the end-effector with the pen and grasp it, and use it to connect the dots on the white-board with lines as in Fig. 4.22. The force control was designed to keep the contact force constant during the drawing phase. A proportional force controller in the z-direction was combined with a 3-DoF visual servoing controller for translation only, into the hybrid control law

$$\mathbf{u} = \begin{bmatrix} \mathbf{0}_{2\times 1} \\ 1 \end{bmatrix} k_F(F_r - F) + k_v \mathbf{P}\left[\mathbf{J}_f(\mathbf{z})\mathbf{P}\right]^\dagger (\mathbf{y}_r - \mathbf{y}) \tag{4.94}$$

where the matrix

$$\mathbf{P} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ p_1 & p_2 \end{bmatrix} \tag{4.95}$$

depends on the constraint plane represented by the board. The parameters p_1 and p_2 were estimated online from sensor data using a recursive least-squares method, as described in [Olsson *et al.*, 2002].

The experimental system consisted of a 6-DoF ABB Irb 2000 robot equipped with a 6-DoF JR3 force/torque sensor, and two Sony DFW-V300

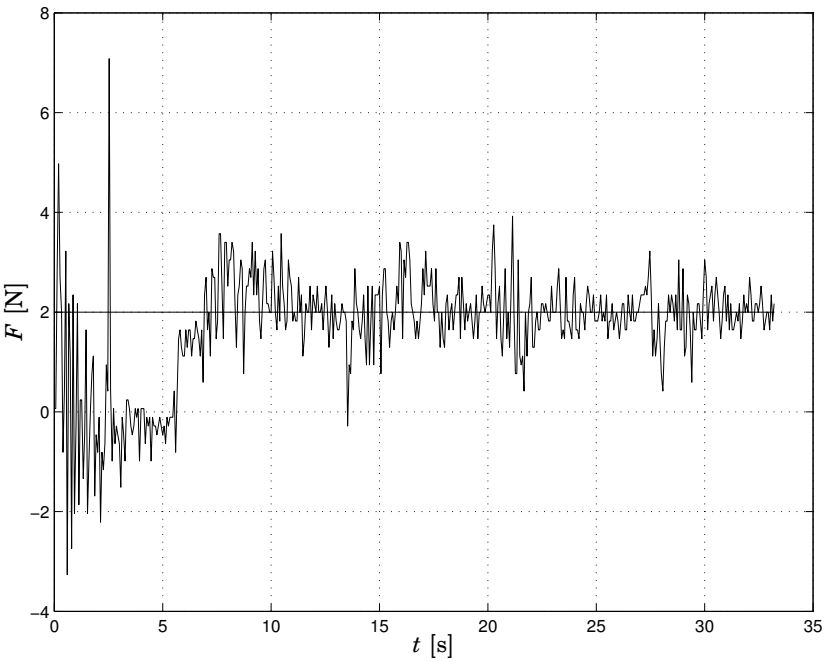

Figure 4.23 Measured force F and reference $F_r = 2$ N during the vision/force-controlled drawing experiment.

digital cameras working at a frame rate of 30 images/second. The cameras sent image data through a 400 Mbit/s IEEE-1394 connection to a standard 450 Mhz Windows PC where the image processing was performed. The feature point locations extracted were then sent to a Sun Ultra60 workstation running a Matlab/Simulink version of the force/vision controller. The sampling period of the controller was 67 ms. The low-level joint position control was handled by an open robot control system described in [Nilsson, 1996], which obtained its motion references from the Matlab/Simulink controller.

In Fig. 4.23 the measured force in the force-controlled direction can be seen. The force control was switched on at $t = 5.3$ s, and contact was achieved at $t = 5.5$ s. The total stiffness of the board and pen in the experiment were 400 N/m. The disturbance at $t < 3$ s was caused by the inertial forces during the acceleration in the approach phase. The measured trajectory of the tip of the pen in Cartesian space can be seen in Fig. 4.24, and the corresponding image-space trajectories in Fig. 4.22.

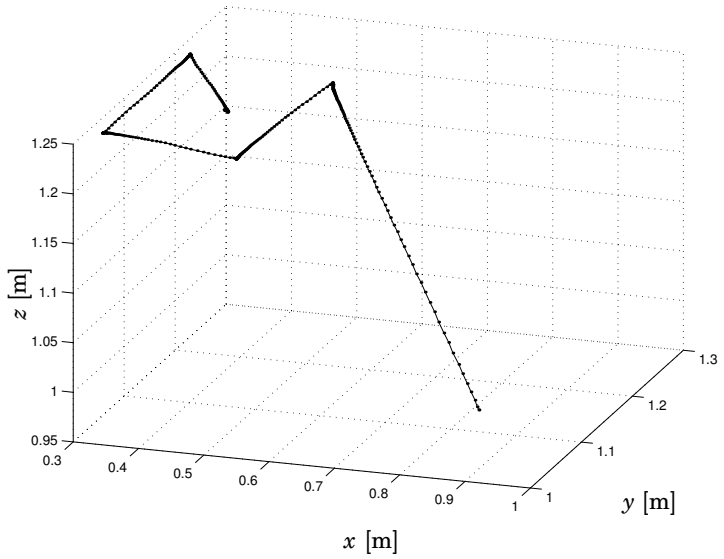

Figure 4.24 Trajectory of the pen during the approach and drawing phases, as shown in in Cartesian space.

4.6 Summary and Concluding Remarks

In this chapter, basic visual pose estimation and motion tracking algorithms were presented. Real-time rigid body tracking with simultaneous tracking of intrinsic parameters was demonstrated, in which a dual quaternion parametrization and linear hand-eye constraints help to reduce the tracking error. An extension to multi-camera visual tracking was presented, which aims at achieving optimal accuracy of the estimate given a maximum computation time, by distribution of image processing resources over the available cameras. Further, two approaches to position-based and image-based force/vision control were presented. Experiments with 6-DoF position-based force/vision control and 3-DoF image-based vision-guided drawing were used to validate the approaches.

For the problem of multi-camera based feedback control, it is shown how the covariance of the position estimation error depends on the set of cameras used, and the number of feature search points in each camera image. These parameters in turn affect the timing properties of the tracking algorithm. An experimentally verified timing model was used to quantify the relation between the number of active cameras and the number of possible features. Using this timing model, it was possible to distribute

image processing resources between cameras to obtain a nearly constant input-output latency of the control loop, which could then be compensated for by the controller. The objective of the resource allocation algorithm was to minimize the variance of the position estimate within the time specified by this desired input-output latency. Real-time simulations were performed in order to demonstrate the effectiveness of the algorithm.

The most important drawback of the edge-based tracker is the difficulty in distinguishing the object edges from nearby parallel edges on the object itself or in the background. Particle filters, such as the well known CONDENSATION algorithm [Isard and Blake, 1998], have proved to be effective in cluttered scenes, due to their ability to incorporate multi-modal probability distributions. However, the effectiveness of particle filters are strongly dependant on the accuracy of the dynamic model. For tracking of motions with several degrees of freedom, real-time implementation of particle filters is problematic, because of the large number of particles required. Alternatively, combining several different types of features—such as edges and point features—may improve robustness considerably [Rosten and Drummond, 2005; Pressigout and Marchand, 2005; Kyrki and Kragic, 2006]. Another method for combining different image measurements will be presented in Chapter 5.

5

Intensity-Based High-Speed Tracking and Control

5.1 Introduction

As in the feature-based method of the previous chapter, the research field of *dynamic vision* is concerned with computer vision for analysis of dynamic scenes. The term *dynamic* in this context is often taken to mean *changing*, as when the scene in question contains moving objects, observed from an image sequence. Although many researchers have attempted to improve robustness and performance of vision systems by exploiting dynamic models for this motion, such as the pioneering work of [Dickmanns, 1988] on dynamic vision for control of autonomous vehicles, many methods referred to as dynamic vision do not make full use of such models. From the point of view of control theory, the dynamic models are of course of central importance. The properties of the feedback depend strongly on the dynamics of the system, as well as on the dynamics of the system model used for estimation of the controlled motion. In particular, dynamic models are important for the performance of high-speed vision systems, where cameras are used to measure and estimate high-speed dynamic phenomena such as robot motion.

In a state-space formulation, the majority of all solutions to dynamical visual tracking and feedback problems share a common structure.

Assuming a nonlinear system

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{u}) \\ \mathbf{z} = \mathbf{C}\mathbf{x} \\ \Delta\mathbf{y} = \mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}), \end{cases} \tag{5.1}$$

where $\Delta\mathbf{y}$ are the image-space errors between predictions and measurements, the estimation of the state $\mathbf{x}$ is handled by including a correction term in the estimator. The correction term is a function of the measured error $\tilde{\mathbf{z}}$ between the true and the predicted workspace positions, and the estimator[1] becomes

$$\dot{\hat{\mathbf{x}}} = \mathbf{f}(\hat{\mathbf{x}}) + \mathbf{g}(\mathbf{u}) + \mathbf{k}(\tilde{\mathbf{z}}, t), \tag{5.2}$$

where the nonlinear and possible time-varying function $\mathbf{k}(\tilde{\mathbf{z}}, t)$ is a function of the error $\tilde{\mathbf{z}} = \mathbf{z} - \hat{\mathbf{z}}$. As this error is not directly measurable, it must be approximated from the image-space error $\Delta\mathbf{y}$. Classical solutions, as described in Chapter 4, are based on modular structures with pose estimation and output injection. In such structures, the function $\mathbf{k}(\tilde{\mathbf{z}}, t)$ is composed of a pose estimation, which can be seen as inverting the function $\mathbf{h}$ to obtain $\tilde{\mathbf{z}}$, and a time-varying (often linear) function of $\tilde{\mathbf{z}}$ to inject the system with the desired error dynamics. Similarly, many visual feedback techniques use a control law of the form

$$\mathbf{u} = \mathbf{l}(\tilde{\mathbf{z}}, t) \tag{5.3}$$

with the control error $\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}_r$. In position-based methods, the function $\mathbf{l}$ is again composed of a pose estimation and a control law expressed as a function of $\tilde{\mathbf{z}}$. In addition to the explicitly modular estimation and control approaches, in many methods an *implicit* pose estimation or linearization of the projection function $\mathbf{h}$ is performed. Examples are the inverse Jacobian control laws of image-based visual servoing, and the linearization of the projection equation used in an EKF, as described in Chapter 4.

A radically different approach is to exclude the pose estimation step, and instead include the structure of the function $\mathbf{h}$ directly in the feedback or correction term. In principle, by composition with a function $\mathbf{K}$, it would be possible to synthesize a correction/feedback term $\mathbf{k}(\tilde{\mathbf{z}}, t)$ or $\mathbf{l}(\tilde{\mathbf{z}}, t)$ of the form

$$\mathbf{k}(\tilde{\mathbf{z}}, t) = \mathbf{K}(\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z})), \tag{5.4}$$

[1]Given the sampled-data nature of image data, the estimators are almost always implemented in discrete time. However, as the work in this chapter is focused on high-speed algorithms, we will assume that the sampling rates are sufficiently fast to make the sampled-data systems possible to approximate by continuous-time models.

designed to satisfy the desired properties with respect to stability and disturbance sensitivity. Then, the correction term in Eq. (5.2) or the feedback in Eq. (5.3) could be computed directly from the image-space errors as $\mathbf{K}(\Delta\mathbf{y})$. The solution to the problem of how to best design the function $\mathbf{K}$ in Eq. (5.4), however, is not obvious in the general case. Instead, it depends on the type of image measurements, and thereby on the structure of the function $\mathbf{h}$.

Problem Formulation and State-of-the-Art. The class of intensity-based visual tracking algorithms, where no explicit geometric feature extraction step is performed, represents one of the traditional methods for visual tracking. One early example is the image registration (or optical flow) method of [Lucas and Kanade, 1981], which used the image intensity gradients directly to update a position estimate by (local) least-squares optimization. More recently, other intensity/color-based methods have been developed for object tracking, such as the *mean shift* tracker of [Comaniciu *et al.*, 2000]. In this chapter we are considering the general class of methods represented by the works of [Gleicher, 1997; Hager and Belhumeur, 1998; Jurie and Dhome, 2001; Malis and Benhimane, 2006]. Such methods are also very closely related to the *active blobs* approach of [Sclaroff and Isidoro, 2003] and the *active appearance* model described in [Cootes *et al.*, 2001]. The common factor in all such methods is that they attempt to minimize, with respect to a number of configuration parameters, the difference between the intensities in some part of the current image and a stored reference image. This approach has been frequently used in visual tracking and the closely related field of image registration, using different optimization methods and expressions of the relationship between the intensities and the configuration parameters.

In [Hager and Belhumeur, 1998], it was suggested to (locally) linearize the relationship between task space motions and the corresponding intensity variations, through a numerical Jacobian computation. The linearization was used to find the position in each image using least-squares optimization. In order to avoid time-consuming online computation of the image gradients, the Jacobian was decomposed into a product of a constant matrix of image gradients at the linearization point, and a configuration-dependent matrix accounting for the image deformation caused by the motion. Illumination variations were modeled by a reduced-order model for illumination of Lambertian surfaces, where the basis functions of the illumination model were computed off-line from training data. However, direct linearization based on the image gradients is not applicable for tracking of fast motions, since the linear approximation will only be accurate in a very narrow region around the linearization point. Higher-order approximations, such as the second-order optimization of [Malis and Ben-

himane, 2006], have been proposed as a way to extend this region, but are unable to extend beyond local convergence.

A very interesting solution to this problem, especially in light of (5.4), is given by the methods of [Gleicher, 1997; Jurie and Dhome, 2001]. In [Gleicher, 1997], it was suggested to generate the correction term from a decomposition of the measured intensity error into a sum of *difference templates* (basis functions). The relation of these templates to the configuration error could be computed off-line. The number of difference templates was implicitly assumed to be "small", although it was pointed out that the number of templates (and thereby the computational complexity) could be increased in order to improve the approximation. As an extension to the method of difference templates, the paper by [Jurie and Dhome, 2001] showed that the performance and region of attraction of the optimization could be increased significantly by using a larger number of templates, and computing off-line a least-squares optimal gain matrix from the intensity error to the applied correction term. This corresponds to composition with a *linear* function in Eq. (5.4). Real-time tracking of the full pose of a rigid object was demonstrated, using a model-based pose estimation technique for compensation of the image deformations caused by the perspective effects.

A major advantage of the approaches of [Gleicher, 1997; Jurie and Dhome, 2001] is that the complex relationship between motion and intensity variations can be compressed *off-line* into a comparatively small matrix. The online computations are then essentially reduced to a matrix multiplication performed at each sample. This, in combination with the very fast direct measurements of image intensities, makes the algorithm possible to run at extremely high frame rates. The purpose of this chapter is to generalize and formalize the above mentioned methods of intensity-based tracking. While the focus of previous work has been on achieving optimal approximation quality for an accurate (static) pose estimation in each time step, the methods in this chapter are focused on problems relevant for dynamic tracking and feedback. In principle, this is achieved by designing the function $\mathbf{K}$ in Eq. (5.4) to give the resulting dynamic system the desired properties with respect to stability, performance and disturbance sensitivity. The capabilities of the methods are illustrated in experiments with 6-DoF tracking and high-speed planar visual servoing at 250 Hz frame rate. Another application of high-speed vision is force/vision control in situations where the most important dynamics is the dynamics of the environment itself. In the final example, methods for force/vision control of a rigid manipulator interacting with a compliant and poorly damped environment are presented, and illustrated in simulations.

Modeling and Basic Intensity-Based Tracking

In visual tracking, a general form of the motion dynamics is given by a nonlinear system of differential equations as in Eq. (5.1). Although the methods of this chapter are applicable for a broader class of systems, we will assume that the motion is governed by a linear dynamical system of the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{5.5}$$

$$\mathbf{z} = \mathbf{C}\mathbf{x} \tag{5.6}$$

where the position $\mathbf{z}$ represents the current pose or general configuration of the target in some well-defined coordinate system. In the following, for a given object point $\mathbf{X}$ these coordinates will be assumed to be related to the camera image coordinates $\mathbf{i}$ by a known transformation

$$\mathbf{i} = \mathbf{f}(\mathbf{z}, \mathbf{X}), \tag{5.7}$$

which could, for instance, represent the projection equation of a pinhole camera. Further, we define a set of N object points $\mathcal{X}_o = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N\}$, located on the visible parts of the object surface. Assume a time-varying intensity image $I(\mathbf{i}, \mathbf{z})$, where the dependence of the image intensity on the object position has been made explicit. Then, the vector of measured image intensities in these points at time t is given by

$$\mathbf{I}(\mathbf{z}, \hat{\mathbf{z}}) = [I(\mathbf{f}(\hat{\mathbf{z}}, \mathbf{X}_1), \mathbf{z}), \cdots, I(\mathbf{f}(\hat{\mathbf{z}}, \mathbf{X}_N), \mathbf{z})] \tag{5.8}$$

where $\mathbf{I}(\mathbf{z}, \hat{\mathbf{z}}) \in \mathbb{R}^{n_c N}$ with n_c denoting the number of image data channels. The objective is to recursively update the current state and position estimate, based on a reference vector $\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0)$ computed with the object in a known initial position $\mathbf{z}_0 = \mathbf{z}(t_0)$ at time t_0, and $\mathbf{I}(\mathbf{z}(t), \hat{\mathbf{z}})$ measured at the estimated position $\hat{\mathbf{z}}$ in the current image.

Previous Approaches. In the previous work of [Gleicher, 1997; Hager and Belhumeur, 1998; Jurie and Dhome, 2001], the dynamics generating the motion were not considered explicitly, and the tracking was expressed in discrete time as iterative corrections to the position estimate. In [Hager and Belhumeur, 1998], the corrections were generated by using a linearized representation of the relationship between motion and the corresponding change in the image intensity at a number of points in the image. By Taylor expansion it can be found that for a small position error $\tilde{\mathbf{z}} = \mathbf{z}(t+h) - \hat{\mathbf{z}}(t)$ and sampling interval h, it holds that

$$\mathbf{I}(\mathbf{z}(t+h), \mathbf{z}(t+h)) = \mathbf{I}(\mathbf{z}(t), \hat{\mathbf{z}}(t)) + \mathbf{I}_{\hat{z}}(\mathbf{z}(t), \hat{\mathbf{z}}(t))\tilde{\mathbf{z}} + \mathbf{I}_t(\mathbf{z}(t), \hat{\mathbf{z}}(t))h + \\ + h.o.t. \tag{5.9}$$

where $\mathbf{I}_{\hat{z}}$ and $\mathbf{I}_t$ denote partial derivatives of $\mathbf{I}$ with respect to estimated position and time, respectively. In order to reduce this expression further, three assumptions are necessary:

1. $\mathbf{I}_t(\mathbf{z}(t), \hat{\mathbf{z}})h = \mathbf{I}(\mathbf{z}(t+h), \hat{\mathbf{z}}) - \mathbf{I}(\mathbf{z}(t), \hat{\mathbf{z}})$
2. $\mathbf{I}(\mathbf{z}(t+h), \mathbf{z}(t+h)) = \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0)$
3. The higher order terms can be neglected.

The first assumption is reasonable as long as the sampling time h is short, which is the case in the high-speed vision applications considered in this chapter. The second assumption expresses the so called *image brightness constancy* assumption, which is in practice only satisfied approximately due to effects such as specular (direction-dependant) reflections and time-variation of light source intensities. Due to the linearization, as expressed by $\mathbf{I}_{\hat{z}}$, the third assumption is unfortunately only approximately valid for small $\tilde{\mathbf{z}}$, and extending the approximation to hold in larger regions of the state space is necessary if tracking of high-speed motion is desired. Still, given the assumptions it is possible to rewrite Eq. (5.9) as

$$\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}(t+h), \hat{\mathbf{z}}(t)) = \mathbf{I}_{\hat{z}}(\mathbf{z}(t), \hat{\mathbf{z}}(t))\tilde{\mathbf{z}} \tag{5.10}$$

which represents a linear relationship between the position error $\tilde{\mathbf{z}}$ and the intensity difference, i.e. the difference between the stored reference intensity vector measured at the initial position $\mathbf{z}_0$, and the measured intensity given by $\mathbf{I}(\mathbf{z}(t+h), \hat{\mathbf{z}}(t))$. The numerically computed Jacobian matrix $\mathbf{I}_{\hat{z}}(\mathbf{z}(t), \hat{\mathbf{z}}(t))$ could be used directly to update the current position estimate as in [Hager and Belhumeur, 1998]

$$\hat{\mathbf{z}}(t+h) = \hat{\mathbf{z}}(t) + \mathbf{I}_{\hat{z}}^{\dagger}(\mathbf{z}(t), \hat{\mathbf{z}}(t))\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}(t+h)) \tag{5.11}$$

where in accordance with Eq. (5.1) we have defined $\mathbf{h}$ and the output error $\Delta\mathbf{y}$ according to

$$\Delta\mathbf{y} = \mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}) = \mathbf{I}(\mathbf{z}, \mathbf{z}) - \mathbf{I}(\mathbf{z}, \underbrace{\mathbf{z} - \tilde{\mathbf{z}}}_{\hat{\mathbf{z}}}) = \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}, \mathbf{z} - \tilde{\mathbf{z}}). \tag{5.12}$$

The accuracy and the region of convergence when using the linear approximation are strongly dependant on the shape and texture of the object around the points of measurement. Using the update equation (5.11) and denoting $\mathbf{I}_{\hat{z}}^{\dagger} = \mathbf{K}_{\mathbf{h}}$, a nonlinear error term

$$\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}) = \tilde{\mathbf{z}} - \mathbf{K}_{\mathbf{h}}\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}) \tag{5.13}$$

is introduced, for which bounds can usually not be given due to the complex and irregular form of the texture dependency. In [Gleicher, 1997; Jurie and Dhome, 2001], it was proposed to directly estimate a matrix $\mathbf{K_h}$ to be used instead of $\mathbf{I}_{\hat{\mathbf{z}}}^{\dagger}$ in Eq. (5.11), a technique referred to as a *hyperplane representation* in [Jurie and Dhome, 2001] or a *difference decomposition* in [Gleicher, 1997]. The matrix $\mathbf{K_h}$ was chosen such that it minimized the differences

$$\boldsymbol{\Delta}(\tilde{\mathbf{z}}_i, \mathbf{z}_0) = \tilde{\mathbf{z}}_i - \mathbf{K_h}\mathbf{h}(\tilde{\mathbf{z}}_i, \mathbf{z}_0) \tag{5.14}$$

in the least-squares sense for some set of training data, consisting of a number of motions $\tilde{\mathbf{z}}_i = \mathbf{z}_0 - \hat{\mathbf{z}}_i$ and the corresponding measured intensity changes $\Delta\mathbf{y}_i = \mathbf{h}(\tilde{\mathbf{z}}_i, \mathbf{z}_0)$ around some linearization position $\mathbf{z}_0$. It is important to note that no physical motions of the object need to be performed in the linearization, since the intensity changes for a number of different *estimated* positions $\hat{\mathbf{z}}_i$ can be calculated from a single image. If the training data matrices are defined as

$$\mathbf{Z} = [\, \tilde{\mathbf{z}}_1 \quad \tilde{\mathbf{z}}_2 \quad \cdots \quad \tilde{\mathbf{z}}_{N_m} \,] \tag{5.15}$$

and

$$\mathbf{Y} = [\, \Delta\mathbf{y}_1 \quad \Delta\mathbf{y}_2 \quad \cdots \quad \Delta\mathbf{y}_{N_m} \,] \tag{5.16}$$

where $N_m > N$ is the number of sample perturbations in the training set, $\mathbf{K_h}$ could be found by taking

$$\mathbf{K_h} = \mathbf{Z}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}. \tag{5.17}$$

As $\mathbf{h}$ is a function of $\mathbf{z}$, the matrix $\mathbf{K_h}$ will depend on the choice of linearization position $\mathbf{z}_0$, and the relationship $\tilde{\mathbf{z}} = \mathbf{K_h}\Delta\mathbf{y}$ can therefore only be guaranteed to provide a good approximation for motions around $\mathbf{z}_0$. Rather than recomputing $\mathbf{K_h}$ at each new position using the time-consuming computations involved in Eq. (5.17), it is possible to compensate for the changes in configuration and viewpoint by pose estimation techniques [Jurie and Dhome, 2001]. In [Hager and Belhumeur, 1998], the Jacobian $\mathbf{I}_{\hat{\mathbf{z}}}$ is updated using a decomposition into configuration-independent and configuration-dependent factors. Another related technique for updating $\mathbf{K_h}$ will be presented later in this chapter.

Graphical Illustration

The multiplication of the measured intensity errors at N points with a matrix $\mathbf{K_h}$ represents a linear combination (in each degree of freedom) of N highly nonlinear functions

$$h_j(\tilde{\mathbf{z}}) \stackrel{\text{def.}}{=} I\left(\mathbf{f}(\mathbf{z}_0, \mathbf{X}_j), \mathbf{z}_0\right) - I\left(\mathbf{f}(\mathbf{z}_0 - \tilde{\mathbf{z}}, \mathbf{X}_j), \mathbf{z}_0\right) \tag{5.18}$$

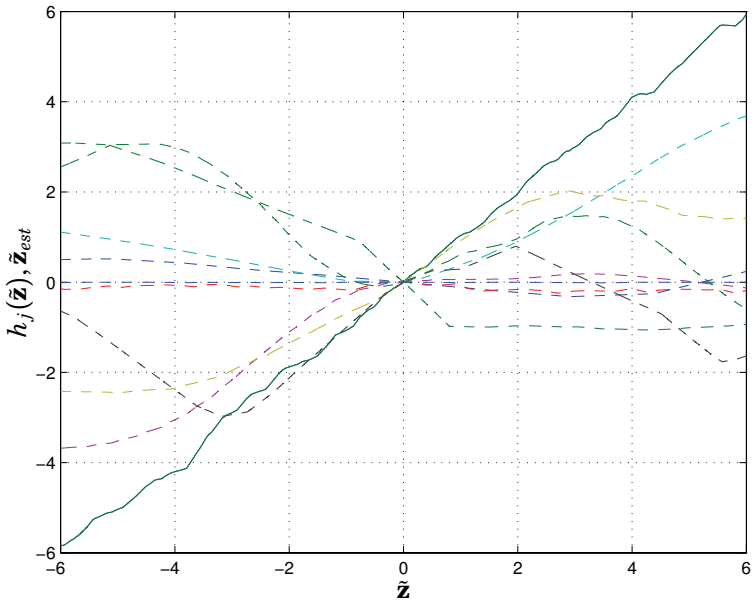

Figure 5.1 A total of $N = 10$ functions h_j (*dashed lines*) and the linear combination (*solid line*) of these N functions, corresponding to the least-squares solution of Eq. (5.17).

of $\tilde{\mathbf{z}}$, where each function h_j corresponds to one element of the vector-valued function $\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}_0)$. The linearization corresponds to finding the linear combination which "best" approximates the linear function $\tilde{\mathbf{z}}$ inside the relevant region, according to a chosen criterion. In the previously described solution, a least-squares criterion was used. In this approximation problem, the basis functions h_j are sampled at N_m points through the creation of the training data set. If we have exactly $N = N_m$ (linearly independent) basis functions, the obtained approximation will be exact (at the sampling points). If $N > N_m$, extra conditions on $\mathbf{K_h}$ must be added in order to obtain a unique solution to the least-squares problem, such as minimum norm. For most practical cases we will have $N < N_m$, and the obtained solution will only be approximate at the sampling points. As N increases, more basis functions will become available, and the approximation error converges to zero.

The approximation of a linear function is illustrated in Fig. 5.1, which shows $N = 10$ nonlinear functions h_j and the "optimal" $\tilde{\mathbf{z}}$-approximating

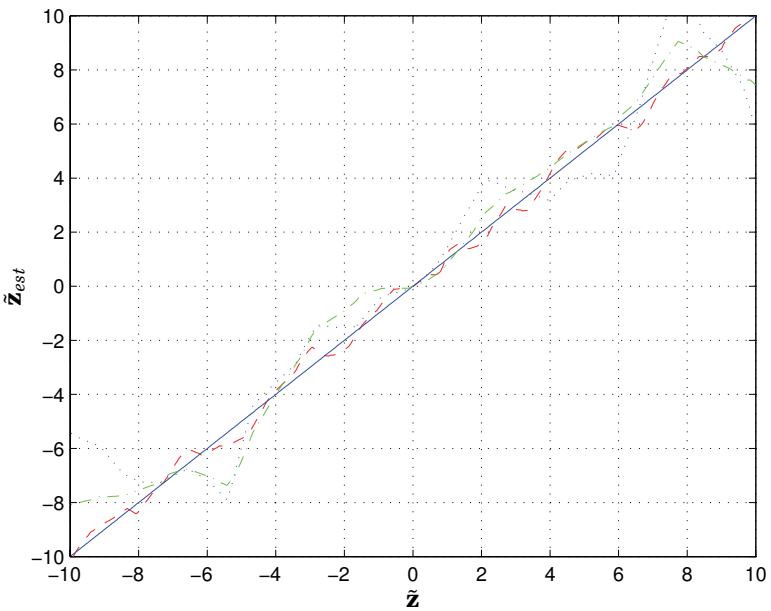

Figure 5.2 Approximations for different numbers of intensity measurement points N, with fixed $N_m = 120$. Shown are $N = 100$ (*solid line*), $N = 15$ (*dashed line*), $N = 5$ (*dash-dotted line*) and $N = 2$ (*dotted line*).

linear combination of these N functions. Despite the low number N of measurement points, the approximation quality is reasonable over the entire region of linearization. Increasing N, the quality can be improved significantly as illustrated in Fig. 5.2. The linearization approximation works well also for multi-dimensional problems, as is shown for a 2-DoF system in Fig. 5.3.

A more critical point, related to the approximation quality, is the number N_m of perturbations used to build the training set. This number determines how densely in $\tilde{\mathbf{z}}$ the continuous functions $h_j(\tilde{\mathbf{z}})$ are sampled, and thereby how well the training set approximates the true behavior around the relevant point. Using too sparse spatial sampling, the approximation between the sampling points may become very poor, as illustrated in Fig. 5.4. This effect becomes even more critical as the number of degrees of freedom increases, and a very large N_m may then be necessary. The required number N_m may be decreased significantly by using a multi-scale approximation technique, as described later in this chapter.

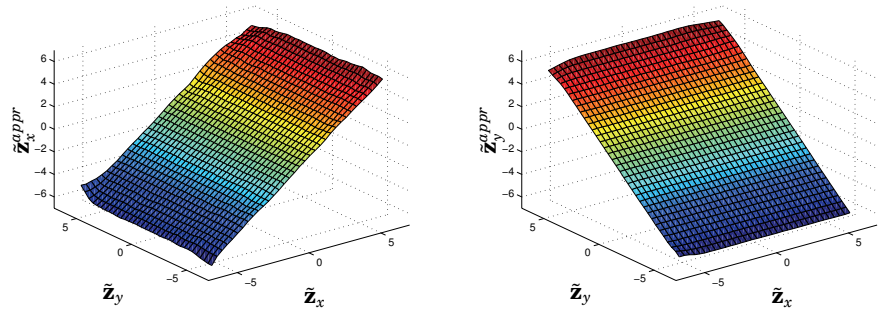

Figure 5.3 Approximation for 2-DoF system using $N = 100$ intensity measurement points, with $N_m = 300$. The approximation is accurate inside the entire region considered in the linearization, with some small errors occurring mainly near the boundaries of the linearization region.

A final important point to note is that the approximation quality depends strongly on the size of the region of approximation, i.e., the region in which the points $\tilde{\mathbf{z}}_i$ of the learning data set are distributed. Intuitively, it is reasonable to assume that it is easier to obtain a good approximation for only small $\tilde{\mathbf{z}}$, for which the functions h_j can be expected to be nearly linear in $\tilde{\mathbf{z}}$, than it is to obtain good behavior for a very wide range of errors $\tilde{\mathbf{z}}$. This is illustrated for a 1-DoF problem in Fig. 5.5, which shows the structure of the functions h_j and the obtained approximations for three different scales, corresponding to rotations of up to 2°, 10°, and 50° around the optical axis. The resulting approximations for both the least-squares approximation of Eq. (5.17) and the local linearization of [Hager and Belhumeur, 1998] are shown for comparison. The local linearization gives acceptable local results, but breaks down outside the region where h_j are nearly linear, while the least-squares approximation works well for a large range of errors. As the dimension n of the problem increases, this range will become smaller. For problems such as 6-DoF rigid body tracking, the basic least-squares approximation of Eq. (5.17) will frequently break down, due to effects such as weak textures, large motions, reflections, and noisy images.

5.2 Approximation-Based Approach

In the solution presented above, the goal is to design $\mathbf{K_h}$ to obtain an approximation of $\tilde{\mathbf{z}}$ which is as close as possible, in the least-squares sense. In the dynamic case, this $\mathbf{K_h}$ can also be used in a continuous-time state

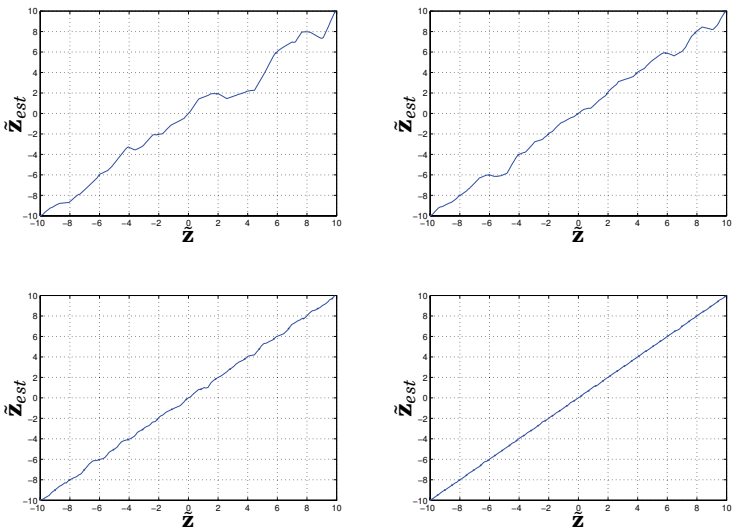

Figure 5.4 Approximations for different numbers of test perturbations N_m in the training set, with $N = 30$ intensity measurement points. Shown are the cases (top to bottom, left to right) $N_m = 6$, $N_m = 11$, $N_m = 21$ and $N_m = 42$.

estimator

$$\begin{aligned}\frac{d\hat{\mathbf{x}}}{dt} &= \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{K}\mathbf{K}_\mathbf{h}\left(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}(t), \hat{\mathbf{z}})\right) = \\ &= \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{K}\mathbf{K}_\mathbf{h}\mathbf{h}(\tilde{\mathbf{z}}(t), \mathbf{z}(t)).\end{aligned} \tag{5.19}$$

Using Eq. (5.13), the estimation error $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ is given by the nonlinear system

$$\begin{cases} \dot{\tilde{\mathbf{x}}} = (\mathbf{A} - \mathbf{K}\mathbf{C})\tilde{\mathbf{x}} + \mathbf{K}\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}) \\ \tilde{\mathbf{z}} = \mathbf{C}\tilde{\mathbf{x}} \end{cases} \tag{5.20}$$

where the stability now depends on the nonlinear function $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$ defined in (5.13). The assumption in this design is that the error term $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$ is sufficiently small to be neglected. In the general case, however, this may not be true, and the effect of the error term on performance and stability must be analyzed. Such methods could for instance be based on establishing bounds[2] for the gain of the nonlinearity $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$, seen as a

[2]In order to establish such bounds from the sampled training data, it is helpful to note that the image intensity can be considered a continuous function of image coordinates. The reason is the image is usually low-pass filtered during pre-processing of the image.

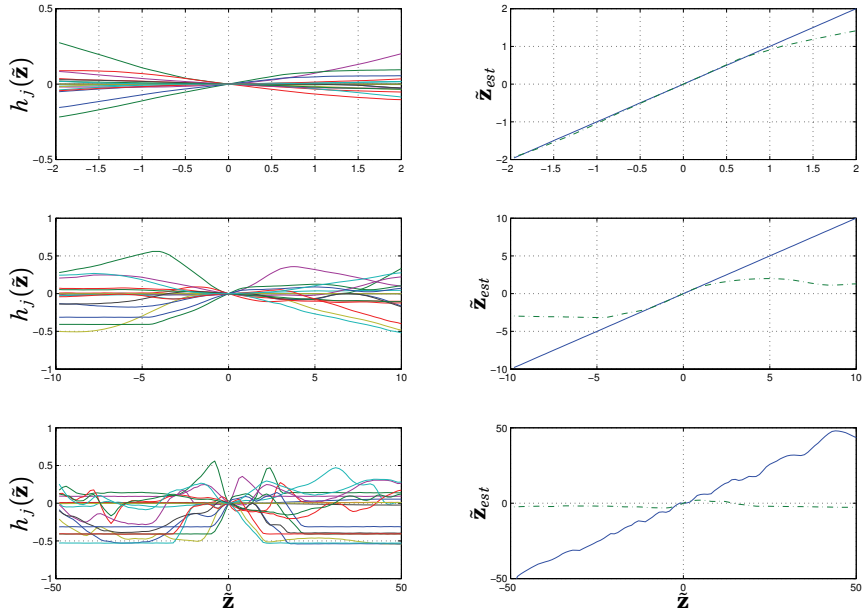

Figure 5.5 *Left, top to bottom:* A total of $N = 18$ functions h_j shown at three different scales. *Right, top to bottom:* Approximations obtained at these scales, for the least-squares solution of Eq. (5.17) (*solid*) and the local linearization as in [Hager and Belhumeur, 1998] (*dashed*).

function of the position estimation error $\tilde{\mathbf{z}}$. However, such bounds are not explicitly considered or derived by the optimization process, and the real bounds may in certain cases be very poor.

Another problem is the sensitivity of the intensity measurements to illumination changes and reflections. Although it is possible to cope with a small number of outliers in the measured intensities, for improved performance the intensity disturbances should be compensated for by model-based disturbance rejection. To this purpose, the model is augmented with a disturbance term $\mathbf{I}_\epsilon(t)$ in the intensity measurement. The estimation error of Eq. (5.20) becomes

$$\dot{\tilde{\mathbf{x}}} = (\mathbf{A} - \mathbf{KC})\tilde{\mathbf{x}} + \mathbf{K}\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}) + \mathbf{K}\mathbf{K}_\mathbf{h}\mathbf{I}_\epsilon(t) \tag{5.21}$$

which indicates that the choice of $\mathbf{K}_\mathbf{h}$ will strongly influence the disturbance sensitivity of the system. If $\mathbf{I}_\epsilon$ is zero-mean white noise with covariance $\mathbf{R}_\epsilon$, the covariance of the noise $\mathbf{K}_\mathbf{h}\mathbf{I}_\epsilon$ in Eq. (5.21) will be equal to $\mathbf{K}_\mathbf{h}\mathbf{R}_\epsilon\mathbf{K}_\mathbf{h}^T$. For spatially uncorrelated noise with $\mathbf{R}_\epsilon = \mathbf{I}$, making the covari-

ance small corresponds to making the sum-of-squares of the elements of the matrix $\mathbf{K_h}$ small. Similarly, by different choices of $\mathbf{R}_\epsilon$ it is possible to model the effects of certain spatially correlated disturbances. For example, a global additive disturbance can be modeled by using $\mathbf{R}_\epsilon = \hat{\mathbf{I}}_\epsilon \hat{\mathbf{I}}_\epsilon^T$ with $\hat{\mathbf{I}}_\epsilon = \mathbf{1}_N$, and the disturbance can be completely suppressed by imposing the constraint

$$\mathbf{K_h}\hat{\mathbf{I}}_\epsilon = 0. \tag{5.22}$$

This would be useful for suppression of varying global illumination or rapidly time-varying artificial lighting. Another condition on $\mathbf{K_h}$, which could be used to ensure local stability, would be to impose that

$$\frac{\partial}{\partial \tilde{\mathbf{z}}}\boldsymbol{\Delta}(\mathbf{0}, \mathbf{z}_0) = \mathbf{I} - \mathbf{K_h}\frac{\partial}{\partial \hat{\mathbf{z}}}\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) = \mathbf{0} \tag{5.23}$$

at $\tilde{\mathbf{z}} = \mathbf{0}$, which is the condition for exact local linearization. A new solution, which attempts to satisfy each of these requirements, is given by minimization of a quadratic cost function in $\mathbf{K_h}$ of the form

$$J(\mathbf{K_h}) = \sum_{i=1}^{N_m} \|\tilde{\mathbf{z}}_i - \mathbf{K_h}\Delta\mathbf{y}_i\|^2 + \gamma^2 \text{trace}(\mathbf{K_h}\mathbf{K_h}^T) \tag{5.24}$$

subject to the linear constraints

$$\begin{cases} \mathbf{K_h}\frac{\partial}{\partial \tilde{\mathbf{z}}}\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I} = \mathbf{0} \\ \mathbf{K_h}\hat{\mathbf{I}}_\epsilon^{(j)} = 0, \quad j = 1, \cdots, N_c \end{cases} \tag{5.25}$$

where $\hat{\mathbf{I}}_\epsilon^{(j)}$ represent vectors or patterns of the illumination disturbances to be suppressed. The constant weight γ can be used to tune the relative importance of data fitting and noise sensitivity.

Multi-Scale Approximation. The accuracy, as well as the stability and region of convergence, can be improved dramatically by using a multi-scale approach, in which a series of matrices $\mathbf{K}_\mathbf{h}^{(j)}$, $j = 1, \cdots, N_s$ are found by linearization using consecutively larger motions. This makes it possible to create a series of approximations at different scales, making it possible to obtain good accuracy both for small and large inter-frame motions. Similar hierarchical solutions with approximation at consecutively finer scales and less blurred images are frequently used in image registration [Gleicher, 1997]. The state estimator in Eq. (5.19) is modified to

$$\frac{d\hat{\mathbf{x}}}{dt} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{K}\mathbf{k}_0(\tilde{\mathbf{z}}, \mathbf{z}) \tag{5.26}$$

where the function $\mathbf{k}_0(\tilde{\mathbf{z}}, \mathbf{z})$ is given recursively by

$$\mathbf{k}_{j-1}(\tilde{\mathbf{z}}, \mathbf{z}) = \mathbf{K}_{\mathbf{h}}^{(j)} \left(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}, \hat{\mathbf{z}} + \mathbf{k}_j(\tilde{\mathbf{z}}, \mathbf{z})) \right) + \mathbf{k}_j(\tilde{\mathbf{z}}, \mathbf{z}) \tag{5.27}$$

$$\mathbf{k}_{N_s}(\tilde{\mathbf{z}}, \mathbf{z}) = \mathbf{0} \tag{5.28}$$

where each step gives a successively better approximation $\mathbf{k}_{j-1}(\tilde{\mathbf{z}}, \mathbf{z})$ of the output error $\tilde{\mathbf{z}} = \mathbf{z} - \hat{\mathbf{z}}$. The advantage of using the multi-scale approach is most easily seen by analyzing the norm of the resulting nonlinear term $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$. If the linearization in each step provides a norm bound

$$\|\boldsymbol{\Delta}_{j-1}(\tilde{\mathbf{z}}, \mathbf{z})\| \stackrel{\text{def.}}{=} \|\tilde{\mathbf{z}} - \mathbf{k}_{j-1}(\tilde{\mathbf{z}}, \mathbf{z})\| \leq \bar{k}_j \|\tilde{\mathbf{z}} - \mathbf{k}_j(\tilde{\mathbf{z}}, \mathbf{z})\| \tag{5.29}$$

on each nonlinearity $\boldsymbol{\Delta}_j$, the total norm bound of the resulting nonlinearity $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}) = \boldsymbol{\Delta}_0(\tilde{\mathbf{z}}, \mathbf{z})$ is given by

$$\|\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})\| \leq \prod_{j=1}^{N_s} \bar{k}_j \|\tilde{\mathbf{z}}\|. \tag{5.30}$$

The tightening of the bounds described by Eq. (5.30) gives a significantly improved performance compared to the single-scale case. The price is an increase in the time complexity of the algorithm, since the image measurement step is repeated N_s times. In many cases, this increase is negligible compared to the time for image capture, transfer and pre-processing.

Compensation for Changes in Configuration. The dependence of $\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z})$ on the current configuration $\mathbf{z}$ causes the linearization to be valid only around $\mathbf{z}_0$. In order for the linearization to be usable in larger regions of the configuration space, it is necessary to compensate for configuration-dependent deformations of the images. We make the assumption that there exists a function $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$, such that for all positions $\mathbf{z}$ and all $\tilde{\mathbf{z}}$

$$\mathbf{I}(\mathbf{z}, \mathbf{z}) - \mathbf{I}(\mathbf{z}, \mathbf{z} - \mathbf{T}(\mathbf{z}, \mathbf{z}_0)\tilde{\mathbf{z}}) = \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0 - \tilde{\mathbf{z}}) \tag{5.31}$$

$$\mathbf{T}(\mathbf{z}_0, \mathbf{z}_0) = \mathbf{I} \tag{5.32}$$

$$\|\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\| \cdot \|\mathbf{T}^{-1}(\mathbf{z}, \mathbf{z}_0)\| < \bar{\mathbf{T}} \tag{5.33}$$

where the matrix $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$ is invertible for all $\mathbf{z}$. The defined function $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$ describes how the relationship between motion and image intensities changes when the surface patches around each measurement point are deformed, through the change in viewpoint associated with the motion from $\mathbf{z}$ to $\mathbf{z}_0$. There are many simple cases in which Eq. (5.31) holds with $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$ as the identity, in which case any bounds for $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$ established

around the initial position $\mathbf{z}_0$, would hold also around any other position $\mathbf{z}$. This removes the configuration dependency of the error term, and $\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$ becomes a function of the error $\tilde{\mathbf{z}}$ only, given by

$$\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}) = \tilde{\mathbf{z}} - \mathbf{K_h} \left(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0 - \tilde{\mathbf{z}}) \right). \tag{5.34}$$

An important example where this is true is when the object undergoes only pure image plane translations or rotations. For more general motions, it is possible to recompute $\mathbf{K_h}$ based on the measured position $\mathbf{z} = \hat{\mathbf{z}} + \tilde{\mathbf{z}}$ as

$$\mathbf{K_h}(\mathbf{z}) = \mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{K_h}(\mathbf{z}_0) \tag{5.35}$$

where $\mathbf{K_h} = \mathbf{K_h}(\mathbf{z}_0)$ corresponds to the original matrix calculated at the linearization position. Using the recomputed $\mathbf{K_h}(\mathbf{z})$ in the estimator in Eq. (5.19) and using assumption (5.31), the estimation error can be written and simplified as

$$\begin{aligned}
\dot{\tilde{\mathbf{x}}} &= \mathbf{A}\tilde{\mathbf{x}} - \mathbf{K}\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{K_h}(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}, \hat{\mathbf{z}})) = \\
&= \mathbf{A}\tilde{\mathbf{x}} - \mathbf{K}\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{K_h}(\mathbf{I}(\mathbf{z}, \mathbf{z}) - \mathbf{I}(\mathbf{z}, \mathbf{z} - \tilde{\mathbf{z}})) = \\
&= \mathbf{A}\tilde{\mathbf{x}} - \mathbf{K}\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{K_h}(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0 - \mathbf{T}(\mathbf{z}, \mathbf{z}_0)^{-1}\tilde{\mathbf{z}})) = \\
&= \mathbf{A}\tilde{\mathbf{x}} - \mathbf{K}\mathbf{T}(\mathbf{z}, \mathbf{z}_0)(\mathbf{T}(\mathbf{z}, \mathbf{z}_0)^{-1}\tilde{\mathbf{z}} - \mathbf{\Delta}(\mathbf{T}(\mathbf{z}, \mathbf{z}_0)^{-1}\tilde{\mathbf{z}}, \mathbf{z}_0)) = \\
&= (\mathbf{A} - \mathbf{K}\mathbf{C})\tilde{\mathbf{x}} + \mathbf{K}\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{\Delta}(\mathbf{T}(\mathbf{z}, \mathbf{z}_0)^{-1}\tilde{\mathbf{z}}, \mathbf{z}_0).
\end{aligned} \tag{5.36}$$

The form of Eq. (5.36) is equivalent to Eq. (5.20), except that the error term $\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$ is transformed by $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$. Together with the bounds (5.33), this can be used to find stability bounds that hold in a larger region of the configuration space[3].

The form of the function $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$ can be very complex, and the computation of $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)$ can be handled in different ways, either explicitly or implicitly. In [Hager and Belhumeur, 1998], the effects of changing configuration were explicitly separated from the factors dependant on the local texture at the measurement points, and used to recompute $\mathbf{K_h}$. In [Jurie and Dhome, 2001], an implicit viewpoint correction of $\mathbf{K_h}$ was proposed, based on standard pose estimation techniques. However, in real-time high-speed applications, the viewpoint compensation algorithms should have a low time complexity, and the computations involved must be possible to distribute over several consecutive samples. In our proposed solution, we choose to update $\mathbf{K_h}$ using the image Jacobian $\mathbf{J}_v$ of image feature motion with respect to position $\mathbf{z}$. The updated matrix $\mathbf{K_h}$ at any estimated position $\hat{\mathbf{z}}$ is calculated from the original $\mathbf{K_h}$ at position $\mathbf{z}_0$ as

$$\mathbf{K_h}(\hat{\mathbf{z}}) = \mathbf{T}(\hat{\mathbf{z}}, \mathbf{z}_0)\mathbf{K_h}(\mathbf{z}_0) = \mathbf{J}_t(\hat{\mathbf{z}})^{\dagger}\mathbf{J}_t(\mathbf{z}_0)\mathbf{K_h}(\mathbf{z}_0) \tag{5.37}$$

[3] For instance, if it is known that $\|\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)\| \leq \bar{k}\|\tilde{\mathbf{z}}\|$, from (5.33) it follows that the transformed nonlinearity satisfies $\mathbf{T}(\mathbf{z}, \mathbf{z}_0)\mathbf{\Delta}(\mathbf{T}(\mathbf{z}, \mathbf{z}_0)^{-1}\tilde{\mathbf{z}}, \mathbf{z}_0) \leq \bar{k}\bar{\mathbf{T}}\|\tilde{\mathbf{z}}\|$.

where $\mathbf{T}(\hat{\mathbf{z}}, \mathbf{z}_0)$ has been approximated by the linearization $\mathbf{J}_t(\hat{\mathbf{z}})^\dagger \mathbf{J}_t(\mathbf{z}_0)$. The Jacobian $\mathbf{J}_t$ relates infinitesimal motions of the object to the corresponding infinitesimal motion of the projection of all measurement points onto the object surface, using a pinhole camera with focal length f. The surface motion is expressed in a local 2D coordinate system attached to a point $(x_o, y_o, z_o)^T$, with basis vectors $(x_1, y_1, z_1)^T$ and $(x_2, y_2, z_2)^T$ tangential to the surface. $\mathbf{J}_t$ can be computed efficiently from the Jacobian $\mathbf{J}_v(\mathbf{z})$ of the projection equation using the equation

$$\mathbf{J}_t(\mathbf{z}) = \mathrm{diag}(\mathbf{M}_{t,1}, \cdots \mathbf{M}_{t,N}) \mathbf{J}_v(\mathbf{z}) \tag{5.38}$$

where the matrices $\mathbf{M}_t$ are matrices in the form

$$\mathbf{M}_t = \frac{1}{f} \begin{bmatrix} (x_1 z_o - z_1 x_o)/z_o^2 & (x_2 z_o - z_2 x_o)/z_o^2 \\ (y_1 z_o - z_1 y_o)/z_o^2 & (y_2 z_o - z_2 y_o)/z_o^2 \end{bmatrix}^{-1}. \tag{5.39}$$

As the update is based on a local linearization, in practice the approximation is most accurate for nearly flat surfaces and motions where the perspective effects are moderate.

5.3 Stability-Based Approach

Apart from the modular, approximation-based approach of Section 5.2, many other approaches to the design of the correction or feedback terms in (5.2)–(5.3) are possible. For systems with linear dynamics, Eq. (5.2) reduces to

$$\frac{d\hat{\mathbf{x}}}{dt} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{k}(\tilde{\mathbf{z}}, t), \tag{5.40}$$

where the stability and other properties of the system are determined by both the system matrices and the term $\mathbf{k}(\tilde{\mathbf{z}}, t)$. The method for analyzing stability outlined in Section 5.2 can be seen as performing a loop transformation with gain $\mathbf{K}$ to achieve a nominally stable system, and establishing a bound for the norm of the resulting nonlinear term $\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z})$. Stability can then be analyzed using standard methods from robust control, such as the small-gain theorem [Khalil, 1996; Zhou and Doyle, 1998]. The problem is that no such bounds are explicitly considered by the optimization process, and in practice the bounds obtained may be arbitrarily poor.

A more general view of properties of the system in Eq. (5.40), is achieved by analyzing the nonlinear part in terms of a sector condition[4]

$$(\mathbf{k}(\tilde{\mathbf{z}}, t) - \mathbf{K}_{\min}\tilde{\mathbf{z}})^T (\mathbf{k}(\tilde{\mathbf{z}}, t) - \mathbf{K}_{\max}\tilde{\mathbf{z}}) \leq 0 \tag{5.41}$$

[4]Note that this formulation includes the previous case of gain bounds, since the sector condition on $\mathbf{k}(\tilde{\mathbf{z}}, t)$ with $\mathbf{K}_{\min} = \mathbf{K} - \bar{k}\mathbf{I}$ and $\mathbf{K}_{\max} = \mathbf{K} + \bar{k}\mathbf{I}$ can be transformed into an equivalent gain condition $\bar{k}$ on the nonlinearity $\mathbf{k}(\tilde{\mathbf{z}}, t) - \mathbf{K}\tilde{\mathbf{z}}$.

for some real matrices $\mathbf{K}_{\max}$ and $\mathbf{K}_{\min}$ with $\mathbf{K}_{\max} - \mathbf{K}_{\min} > 0$ [Khalil, 1996]. Usually, the controller design or system structure will impose *known* sector conditions that the linearization must satisfy, which leads to extra constraints in the problem. Assuming that $\mathbf{k}(\tilde{\mathbf{z}}, t)$ is composed as before of a linear function of $\Delta\mathbf{y} = \mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z})$ as

$$\mathbf{k}(\tilde{\mathbf{z}}, t) = \mathbf{K_h}\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}), \tag{5.42}$$

a general formulation for this case is given by the minimization of the convex cost function (based on the Frobenius norm $\|\mathbf{K_h}\|_F$)

$$J(\mathbf{K_h}) = \text{trace}(\mathbf{K_h}\mathbf{K_h}^T) = \|\mathbf{K_h}\|_F^2 \tag{5.43}$$

with respect to the unknown $\mathbf{K_h}$, subject to the linear and convex quadratic constraints

$$\begin{cases} (\mathbf{K_h}\Delta\mathbf{y}_i - \mathbf{K}_{\min}\tilde{\mathbf{z}}_i)^T(\mathbf{K_h}\Delta\mathbf{y}_i - \mathbf{K}_{\max}\tilde{\mathbf{z}}_i) \leq 0, & i = 1, \cdots, N_m \\ \mathbf{K_h}\hat{\mathbf{I}}_\epsilon^{(j)} = 0, \quad j = 1, \cdots, N_c. \end{cases} \tag{5.44}$$

For some practically important classes of problems, the quadratic constraints (5.44) can be reformulated into linear constraints. For such problems (possibly conservative or sub-optimal) solutions can be found by quadratic programming [Boyd and Vandenberghe, 2004], for which very effective algorithms and software exist. Two such classes of problems are described in the following sections.

General Sector Conditions. In the case of general sector conditions, in order to avoid solving the quadratically constrained problem defined by (5.43)–(5.44), sub-optimal solutions must be sought. The new solution is based on rewriting the sector condition with matrices $\mathbf{K}_{\min} = \mathbf{K} - \mathbf{D}$ and $\mathbf{K}_{\max} = \mathbf{K} + \mathbf{D}$, with $\mathbf{D}$ positive definite, as an equivalent norm condition

$$\|\mathbf{K_h}\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}) - \mathbf{K}\tilde{\mathbf{z}}\|_2 \leq \|\mathbf{D}\tilde{\mathbf{z}}\|_2. \tag{5.45}$$

To proceed, a straightforward solution is to instead use the standard matrix ∞-norm, and to replace the norm condition in Eq. (5.45) with

$$\|\mathbf{K_h}\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}) - \mathbf{K}\tilde{\mathbf{z}}\|_\infty \leq n^{-1/2}\|\mathbf{D}\tilde{\mathbf{z}}\|_\infty \tag{5.46}$$

where $n = \dim(\mathbf{z})$. Due to the properties of the ∞-norm, condition (5.45) is satisfied if (5.46) holds. The opposite is not true, a fact that leads to some degree of conservatism in the solution. Using the condition of Eq. (5.46),

it is possible to formulate a quadratic program (QP), which optimizes the cost

$$J(\mathbf{K_h}) = \text{trace}(\mathbf{K_h}\mathbf{K_h}^T) \tag{5.47}$$

subject to the linear constraints[5]

$$\begin{cases} -n^{-1/2}\|\mathbf{D}\tilde{\mathbf{z}}_i\|_\infty \preceq \mathbf{K_h}\Delta\mathbf{y}_i - \mathbf{K}\tilde{\mathbf{z}}_i \preceq n^{-1/2}\|\mathbf{D}\tilde{\mathbf{z}}_i\|_\infty, \quad i = 1, \cdots, N_m \\ \mathbf{K_h}\frac{\partial}{\partial \tilde{\mathbf{z}}}\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{K} = \mathbf{0} \\ \mathbf{K_h}\hat{\mathbf{I}}_c^{(j)} = 0, \quad j = 1, \cdots, N_c \end{cases} \tag{5.48}$$

where the criterion is minimized subject to the constraint that all motions and corresponding intensity variations in the test set satisfy the desired sector bound. For certain classes of sector conditions, such as the previously discussed gain conditions where $\mathbf{D} = \bar{k}\mathbf{I}$, "optimal" bounds may be found by considering $\bar{k}$ as a parameter to be included in the criterion to be optimized, rather than as a known constant.

When the number of degrees of freedom is large, a very large number of test motions N_m is required, or otherwise the approximation may not satisfy the sector condition between the sampling points. In practice, the solution to the quadratic program (5.47)–(5.48) may still give improved stability, due to the better approximation of the nonlinearity around the origin. The drawback is that the accuracy is in general decreased when tracking faster motions. This drawback is avoided by using a multi-scale estimator structure as in Section 5.3.

EXAMPLE 5.1—SINGLE-DOF SYSTEM

A planar textured object, with a single rotational degree of freedom around the camera optical axis, was placed in the linearization position shown in Fig. 5.6. On the planar surface, $N = 20$ positions for the intensity measurements were placed in a rectangular grid. The solution to a problem with $\mathbf{K}_{\min} = (1 - \bar{k}_j)\mathbf{I}$ and $\mathbf{K}_{\max} = (1 + \bar{k}_j)\mathbf{I}$ was computed from the quadratic program (5.47)–(5.48) for $N_m = 200$ reference motions, where the optimization criterion (5.47) was modified to include a quadratic term in the parameter $\bar{k}_j$. Fig. 5.7 shows the resulting $\mathbf{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)$ and the obtained sectors corresponding to $\bar{k}_j$. The linearization was computed at three different scales, corresponding to maximum rotations of 30°, 8°, and 2°, respectively. The final resulting approximation error for the multi-scale method is also shown, indicating the significant improvement of the multi-scale algorithm. □

[5]In Eq. (5.48), $\preceq$ is used to denote element-wise inequality.

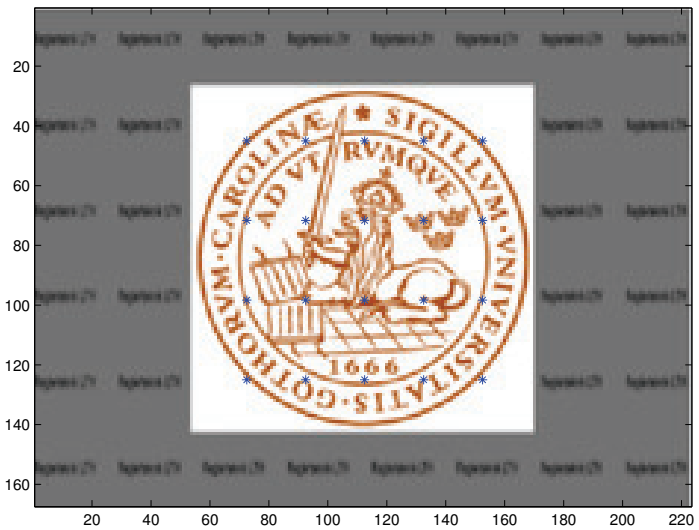

Figure 5.6 The object used in Example 5.1 shown in the linearization position, with the stars marking the $N = 20$ intensity measurement positions. In the example, a smoothed gray-scale version of the image was used.

EXAMPLE 5.2—MULTI-DOF SYSTEM

As an example of a system with multiple degrees of freedom, the system in Example 5.1 was extended to a 3-DoF system with rotation and translation in a plane parallel to the image plane. A total of $N = 100$ intensity measurement points were placed in a rectangular grid on the planar surface, and the quadratic program (5.47)–(5.48) was solved for $N_m = 1000$ reference motions. The results can be seen in Fig. 5.8, where $\|\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)\|$ is plotted against $\|\tilde{\mathbf{z}}\|$ for all points in a large validation data set[6]. Due to the spatial sampling introduced by the finite number of test motions, the computed bounds did not hold exactly, and a larger N_m would be necessary to provide more accurate bounds. As a comparison, the results for the least-squares solution of Eqs. (5.24)–(5.25) can be seen in Fig. 5.9. The parameter γ was tuned such that the norms of $\mathbf{K_h}$ were comparable for the two solutions. While the least-squares solution provided a slightly better overall approximation of the nonlinearity, it could not guarantee any bounds on the norm of the error term $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)$, as for the sector-based solution. For both types of solutions, significantly better approximations were obtained with a multi-scale approach, as shown in the bottom plots of Figs. 5.8–5.9. □

[6]More specifically, rather than showing each point in the (large) validation data set, the figure displays a computed upper bounding curve for all points in the set.

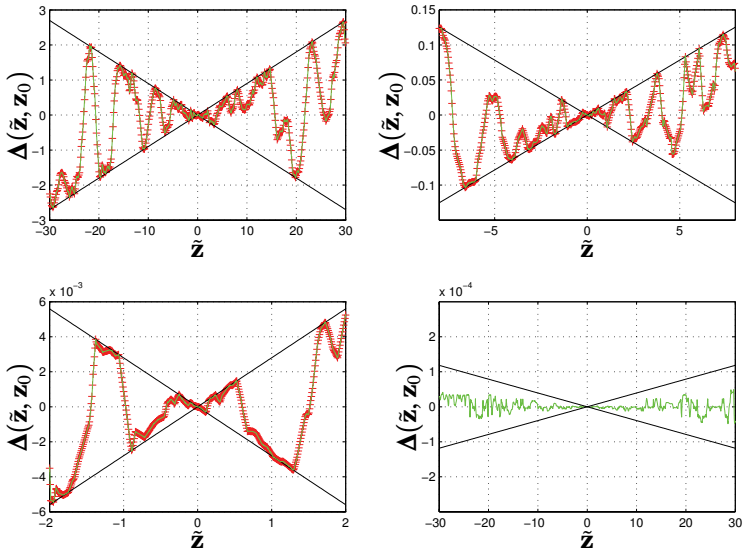

Figure 5.7 Estimation error nonlinearities $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)$ and computed sector/norm bounds for 1-DoF rotational example. *Top left, top right, bottom left*: Approximation errors and computed bounds $\bar{k}_j$ at three consecutively finer scales of 30°, 8°, 2° rotation, respectively. *Bottom right*: Resulting approximation error for the multi-scale method, with the sector for the finest scale approximation shown as comparison. Comparison between top left and bottom right shows an improvement in the accuracy and computed bounds with a factor of over 25000.

Upper/Lower Sector Bounds. An important case which can be handled without introducing conservatism in the solution, is when the control design imposes a single-sided (lower) sector bound in the form

$$(\mathbf{k}(\tilde{\mathbf{z}}, t) - \mathbf{K}_{\min}\tilde{\mathbf{z}})^T \tilde{\mathbf{z}} \geq 0. \tag{5.49}$$

Such bounds occur, for instance, when trying to estimate the state of a (stable or unstable) linear system with first-order dynamics. The condition in Eq. (5.49) can be imposed by solving the QP given by minimizing

$$J(\mathbf{K}_\mathbf{h}) = \text{trace}(\mathbf{K}_\mathbf{h}\mathbf{K}_\mathbf{h}^T) \tag{5.50}$$

subject to the linear constraints

$$\begin{cases} \tilde{\mathbf{z}}_i^T \mathbf{K}_\mathbf{h} \Delta\mathbf{y}_i \geq \tilde{\mathbf{z}}_i^T \mathbf{K}_{\min} \tilde{\mathbf{z}}_i, & i = 1, \cdots, N_m \\ \mathbf{K}_\mathbf{h} \hat{\mathbf{I}}_\epsilon^{(j)} = 0, & j = 1, \cdots, N_c \end{cases} \tag{5.51}$$

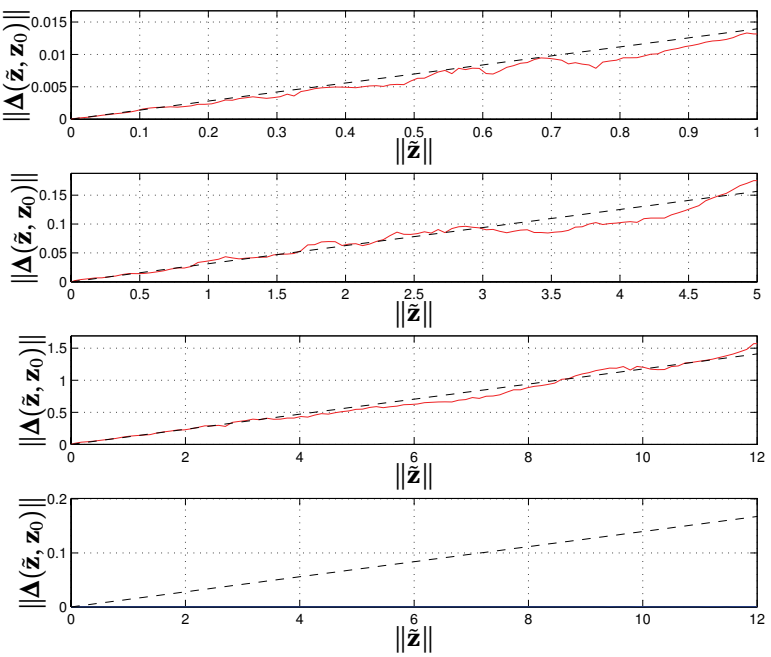

Figure 5.8 Norm $\|\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)\|_2$ of estimation error nonlinearities $\boldsymbol{\Delta}(\tilde{\mathbf{z}}, \mathbf{z}_0)$ and computed norm bounds for 3-DoF example, given by solution to quadratic program (5.47)–(5.48). *Top 3 plots*: Errors and computed bounds $\bar{k}_j$ at three consecutively coarser scales. *Bottom plot*: Resulting error for the multi-scale method (almost equal to zero everywhere).

where $\mathbf{K}_{\min}$ is assumed to be part of the given specification. The equations for the case in which an upper bound $\mathbf{K}_{\max}$ is desired are completely analogous.

EXAMPLE 5.3—SINGLE-SIDED SECTOR CONDITION

Using the same one-dimensional problem as in Example 5.1, Fig. 5.10 illustrates the advantage of exploiting the extra freedom of the asymmetrical sector condition approach over the approximation-based approach represented by the least-squares solution of Eqs. (5.24)–(5.25). The noise sensitivity of the improved solution, as measured by $\|\mathbf{K}_\mathbf{h}\|_F$, was decreased by a factor of 14, while also improving the robustness and providing guarantees of stability. While in the least-squares case the parameter γ could be tuned for a better trade-off between approximation error and noise

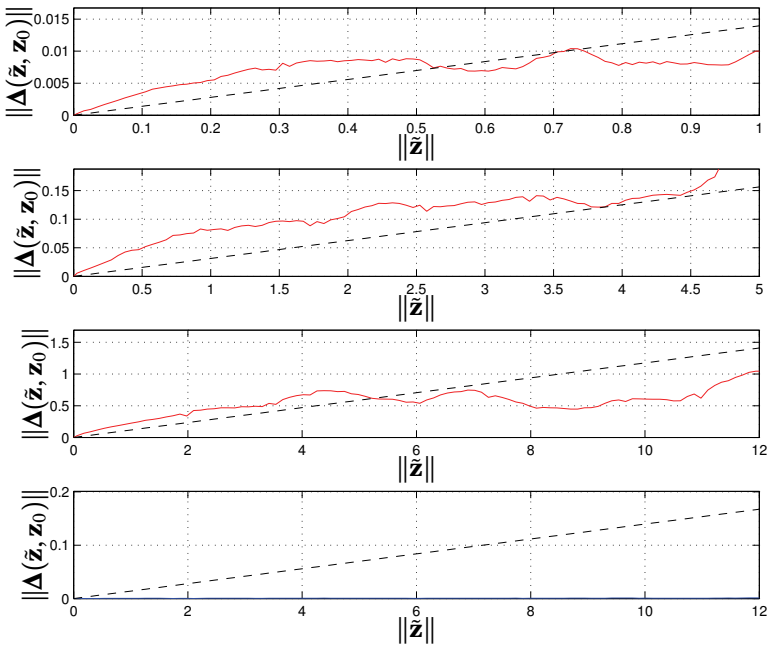

Figure 5.9 Norm $\|\Delta(\tilde{\mathbf{z}}, \mathbf{z}_0)\|_2$ of estimation error nonlinearities $\Delta(\tilde{\mathbf{z}}, \mathbf{z}_0)$ for 3-DoF example, given by least-squares problem from Eqs. (5.24)–(5.25). *Top 3 plots*: Errors at three consecutively coarser scales. The best possible norm bounds $\bar{k}_j$ obtained from (5.47)–(5.48) are shown for comparison. *Bottom plot*: Resulting error for the multi-scale method (almost equal to zero everywhere).

sensitivity, the improved QP-solution of the problem in Eqs. (5.50)–(5.51) would directly provide the solution with the best noise sensitivity among those satisfying the specified bounds. □

EXAMPLE 5.4—GENERALIZED INTENSITY-BASED IBVS/PBVS
Another example in which techniques similar to Example 5.3 can be used, is provided by a generalization of the simple visual servoing dynamic model Eq. (2.13)–(2.14). We assume that a 1-DoF system is given by

$$\dot{z} = f(z) + u \tag{5.52}$$

$$\Delta\mathbf{y} = \mathbf{I}(z, z_0) - \mathbf{I}(z_0, z_0) \stackrel{\text{def}}{=} \mathbf{h}(z) \tag{5.53}$$

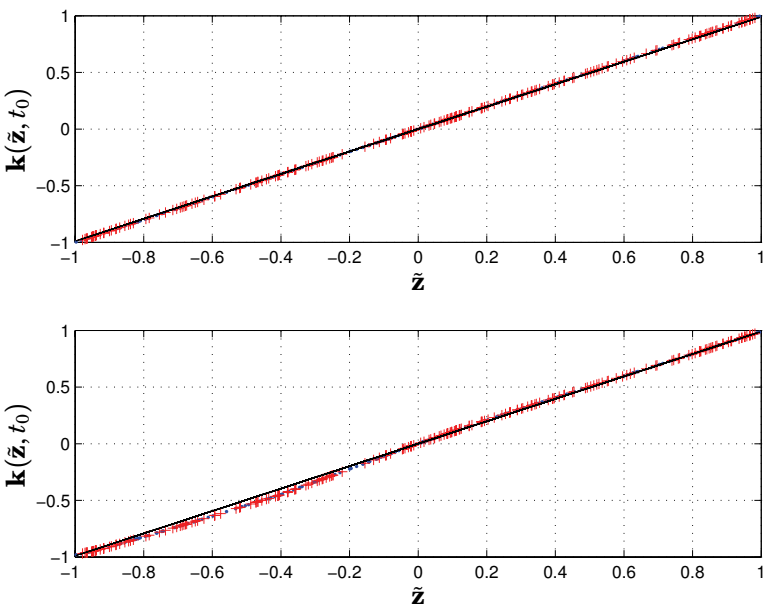

Figure 5.10 Function $\mathbf{k}(\tilde{\mathbf{z}}, t_0) = \mathbf{K_h}\mathbf{h}(\tilde{\mathbf{z}}, \mathbf{z}_0)$ and desired (lower) sector bound for the 1-DoF system in Example 5.3. *Top plot*: Least-squares solution given by solving the problem in Eqs. (5.24)–(5.25). *Bottom plot*: The solution given by the quadratic program (5.50)–(5.51). Compared to the least-squares solution, the norm of the linearization matrix $\mathbf{K_h}$ was reduced by a factor of 14.

where $f(z)$ is a scalar nonlinear function with $f(0) = 0$, and the measurement/intensity vector $\mathbf{I}(z, z_0)$ is always taken at measurement points corresponding to the *fixed* location $z_0 = 0$. As before, the dependence of the image and output intensity change $\Delta\mathbf{y}$ on z has been made explicit through the function $\mathbf{h}(z)$. Assuming a control law $u = K(\Delta\mathbf{y})$ and using a quadratic Lyapunov function $V(z) = z^2$, the system can be shown to be asymptotically stable if and only if the condition

$$f(z)z + \underbrace{K(\mathbf{h}(z))}_{l(z)} z < 0, \quad \forall z \neq 0 \tag{5.54}$$

holds. If the control law is chosen to be linear in the measured intensities

$$u = l(z) = \mathbf{K_h}\Delta\mathbf{y} = \mathbf{K_h}\mathbf{h}(z), \tag{5.55}$$

a suitable controller gain is found by minimizing

$$J(\mathbf{K_h}) = \text{trace}(\mathbf{K_h}\mathbf{K_h}^T) \tag{5.56}$$

as before, subject to the constraints

$$\begin{cases} z_i \left(f(z_i) + k_{\min} z_i + \mathbf{K_h}\mathbf{h}(z_i) \right) \leq 0, & i = 1, \cdots, N_m \\ \mathbf{K_h}\hat{\mathbf{I}}_\epsilon^{(j)} = 0, \quad j = 1, \cdots, N_c \end{cases} \tag{5.57}$$

where $k_{\min} > 0$ characterizes an optional sector condition, used to obtain exponential stability. In Fig. 5.11 we can see the results of using a controller (5.55), compared to an approximately linearizing controller

$$u = -f\left(\mathbf{K}_\mathbf{h}^{(a)} \Delta\mathbf{y} \right) - k_{\min} \Delta\mathbf{y} \tag{5.58}$$

where $\mathbf{K}_\mathbf{h}^{(a)}$ was computed using Eqs. (5.24)–(5.25), by tuning the parameter γ in order to provide a suitable trade-off between approximation $z \approx \mathbf{K}_\mathbf{h}^{(a)}\mathbf{h}(z)$ and noise rejection. The controller (5.58) represents a position-based visual servoing control law, where the feedback contains an explicit computation of the work-space position z, while Eq. (5.55) corresponds to purely image-based visual servoing. As can be seen in Fig. 5.11, the image-based controller (5.55) resulted in both faster convergence and improved noise sensitivity. In fact, the system controlled using the position-based controller (5.58) had multiple stable equilibria near z_0, as indicated by the behavior in the top plot of Fig. 5.11. Using the image-based controller, exponential convergence to the single stable equilibrium z_0 was obtained. □

EXAMPLE 5.5—MULTI-DOF IBVS AND DYNAMIC PBVS
As a direct multi-DoF extension to the IBVS controller in Example 5.4, we designed a controller for the decoupled 3-DoF integrator system

$$\dot{\mathbf{z}} = \mathbf{u} \tag{5.59}$$

$$\Delta\mathbf{y} = \mathbf{I}(\mathbf{z}, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) \stackrel{\text{def}}{=} \mathbf{h}(\mathbf{z}), \tag{5.60}$$

where $\mathbf{z}$ was a vector of the rotation angles of a textured cubic object around position $\mathbf{z}_0$, as shown in Fig. 5.12. Using $N_m = 300$ test motions, with $N = 90$ measurement points, and designing a control law

$$\mathbf{u} = \mathbf{l}(\mathbf{z}) = \mathbf{K_h}\Delta\mathbf{y} = \mathbf{K_h}\mathbf{h}(\mathbf{z}), \tag{5.61}$$

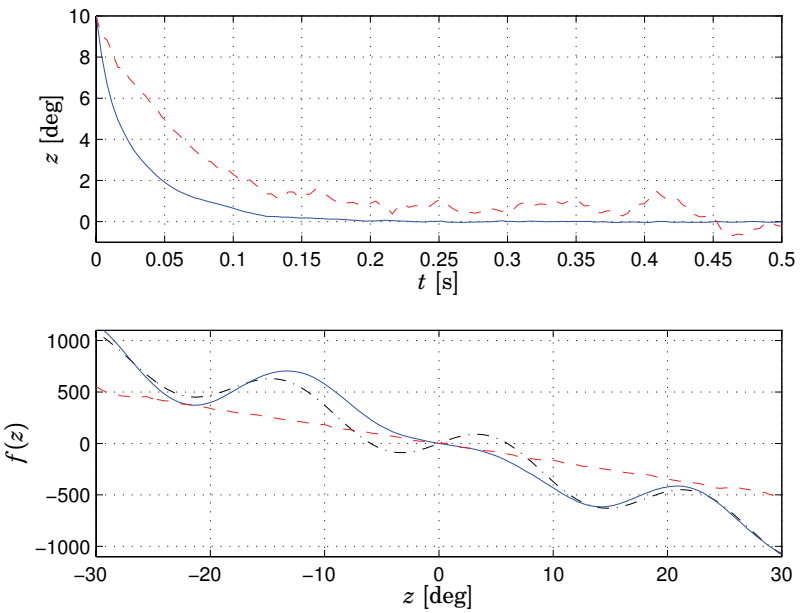

Figure 5.11 *Top plot*: Convergence of simulated IBVS in Example 5.4, using the image-based controller (5.55) (*solid*) and the linearizing controller (5.58) (*dashed*). *Bottom plot*: function $f(z)$ (*dash-dotted*) and resulting right hand side $f(z) + l(z)$ when using the one-sided criterion (*solid*) and approximately linearizing controller (*dashed*). The approximately linearizing controller gives several local equilibria $f(z) + l(z) = 0$ near the origin.

by solving the optimization problem of Eqs. (5.50)–(5.51) for a sector condition (5.49) with $\mathbf{K}_{\min} = 20\mathbf{I}$, a control resulting in exponential stability of $\mathbf{z}_0$ was found, as seen in Fig. 5.13.

As an alternative approach, the exact same matrix $\mathbf{K_h}$ was used for dynamic position-based visual servoing, by using a state estimator and controller structure in the form

$$\dot{\hat{\mathbf{z}}} = \mathbf{u} + \mathbf{L}_1\mathbf{K_h}(\mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}, \hat{\mathbf{z}})) \tag{5.62}$$

$$\mathbf{u} = \mathbf{L}_2(\mathbf{z}_0 - \hat{\mathbf{z}}), \tag{5.63}$$

where $\mathbf{L}_1$ and $\mathbf{L}_2$ were linear gains, used to obtain the desired dynamic properties. The results, using the same setup as before with $\mathbf{L}_1 = 2.5\mathbf{I}$ and $\mathbf{L}_2 = 20\mathbf{I}$, can be seen in Figs. 5.14–5.15. A comparison with Figs. 5.12–5.13 shows that the convergence in the position-based case was smoother and

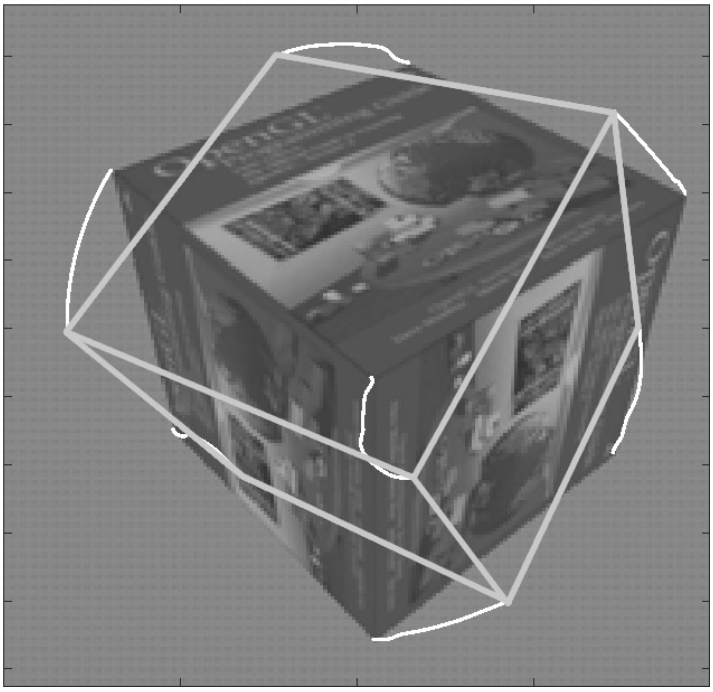

Figure 5.12 Convergence in image-space for the simulated 3-DoF IBVS in Example 5.5.

more controlled, thanks to the fast state estimator which would effectively "linearize" the system, decreasing the effect of the nonlinearity $\mathbf{h}(\mathbf{z})$ on the output. □

EXAMPLE 5.6—CONTROL OF A MECHANICAL SYSTEM
We consider the (rigid) mechanical dynamics in the form

$$\dot{\mathbf{x}}_1 = \mathbf{x}_2 \tag{5.64}$$

$$\dot{\mathbf{x}}_2 = -\mathbf{f}_1(\mathbf{x}_1) - \mathbf{f}_2(\mathbf{x}_2) + \mathbf{u} \tag{5.65}$$

$$\mathbf{z} = \mathbf{x}_1 \tag{5.66}$$

$$\Delta\mathbf{y} = \mathbf{I}(\mathbf{z}, \mathbf{z}_0) - \mathbf{I}(\mathbf{z}_0, \mathbf{z}_0) \stackrel{\text{def}}{=} \mathbf{h}(\mathbf{z}) \tag{5.67}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ correspond to position and velocity, $\mathbf{f}_1(\mathbf{x}_1)$ with $\mathbf{f}_1(\mathbf{0}) = \mathbf{0}$ is a potential vector field, and $\mathbf{f}_2(\mathbf{x}_2)$ represents the natural dissipation of

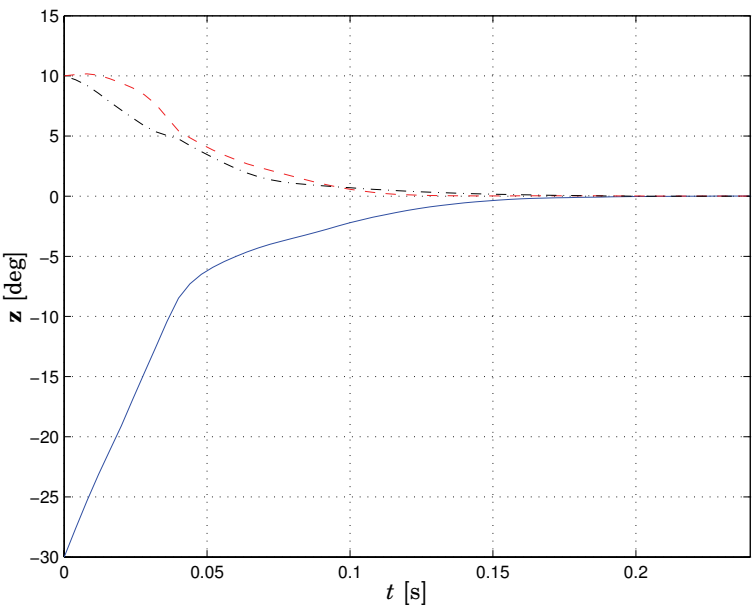

Figure 5.13 Convergence of simulated 3-DoF IBVS in Example 5.5, shown as the control error $\mathbf{z}$ in the rotation angles around the z-axis (*solid*), y-axis (*dashed*) and x-axis (*dash-dotted*), respectively.

the system, with $\mathbf{x}_2^T \mathbf{f}_2(\mathbf{x}_2) \geq 0$. Using a control law in the form

$$\mathbf{u} = -\mathbf{u}_1(\mathbf{z}) - \frac{d}{dt}[\mathbf{u}_2(\mathbf{z})] \tag{5.68}$$

with the nonlinear functions $\mathbf{u}_i$

$$\mathbf{u}_i(\mathbf{z}) = \mathbf{K}_{h,i}\Delta\mathbf{y} = \mathbf{K}_{h,i}\mathbf{h}(\mathbf{z}), \tag{5.69}$$

the stability could be analyzed using energy-based Lyapunov functions. As an example, we take the control of a one-link planar robot or pendulum given by

$$\dot{x}_1 = x_2 \tag{5.70}$$

$$\dot{x}_2 = -f_1(x_1) - f_2(x_2) + u = \omega_0^2 \sin(x_1) - dx_2 + u \tag{5.71}$$

$$z = x_1 \tag{5.72}$$

$$\Delta\mathbf{y} = \mathbf{I}(z, z_0) - \mathbf{I}(z_0, z_0) \stackrel{\text{def}}{=} \mathbf{h}(z) \tag{5.73}$$

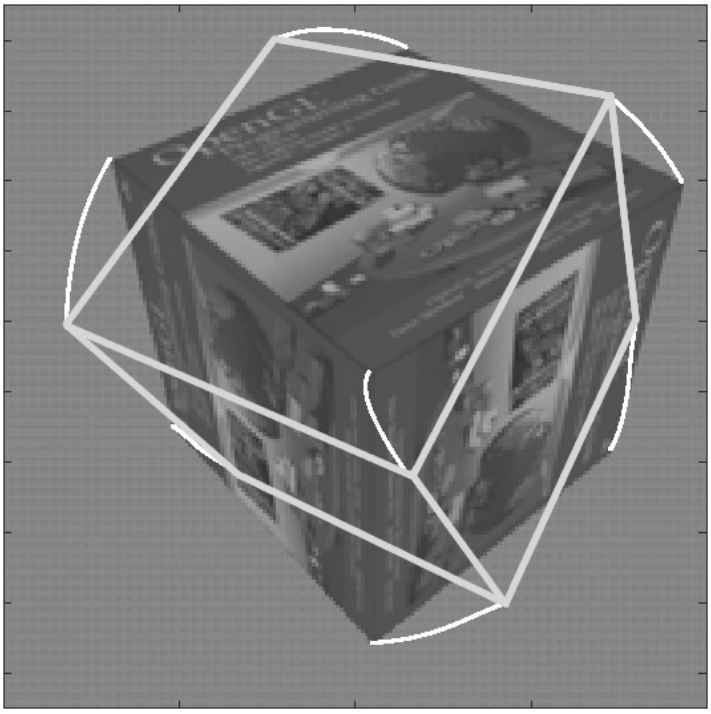

Figure 5.14 Convergence in image-space for the simulated 3-DoF PBVS in Example 5.5.

around the unstable upper equilibrium, corresponding to $x_1 = z_0 = 0$. The control signal u was the normalized torque applied to the arm, $d > 0$ was a damping coefficient, and $\omega_0^2 = mgl/I$ depended on the mass m, effective length l and moment of inertia I of the arm. Using the control law

$$u = -u_1(z) - \frac{d}{dt}[u_2(z)] = -\mathbf{K}_{h,1}\mathbf{h}(z) - \frac{d}{dt}\left[\mathbf{K}_{h,2}\mathbf{h}(z)\right] \tag{5.74}$$

together with the total mechanical energy

$$V(x_1, x_2) = \int_0^{x_1} f_1(x) + u_1(x)dx + \frac{1}{2}x_2^T x_2. \tag{5.75}$$

as a Lyapunov equation, we obtained

$$\dot{V} = -x_2 f_2(x_2) - \frac{du_2}{dx_1}(x_1)x_2^2. \tag{5.76}$$

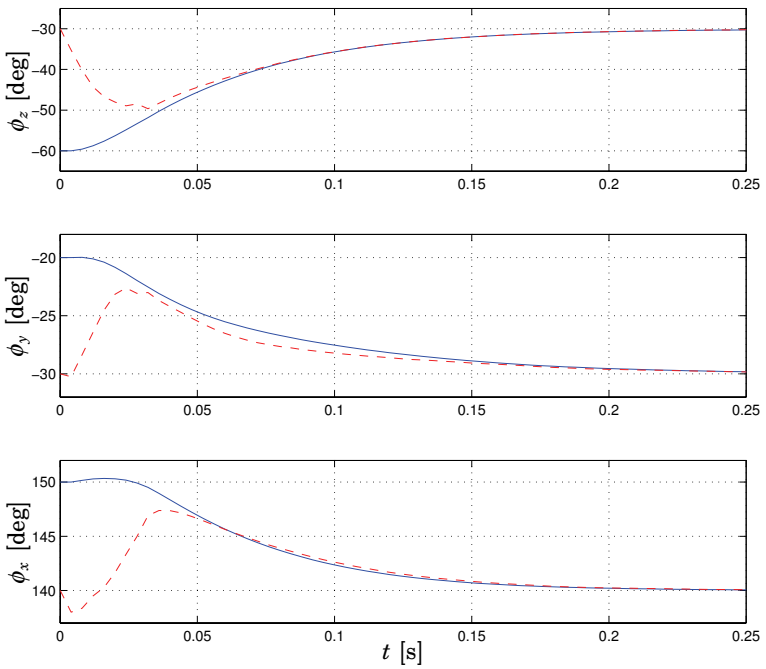

Figure 5.15 Convergence of simulated 3-DoF PBVS in Example 5.5, shown as the true position $\mathbf{z}$ (*solid*) and estimated position error $\hat{\mathbf{z}}$ (*dashed*) for rotations around the x,y, and z-axes respectively.

Negative semi-definiteness of $\dot{V}$ and positive definiteness of V was assured by imposing the conditions

$$\frac{du_2}{dx_1}(x_1) > 0, \qquad \forall x_1 \tag{5.77}$$

$$x_1(f_1(x_1) + u_1(x_1)) > 0, \qquad \forall x_1 \neq 0, \tag{5.78}$$

which led to a convex problem equivalent to (5.56)–(5.57) for $\mathbf{K}_{h,1}$. The matrix $\mathbf{K}_{h,2}$ was given by minimizing a criterion

$$J(\mathbf{K}_{h,2}) = \text{trace}(\mathbf{K}_{h,2}\mathbf{K}_{h,2}^T) \tag{5.79}$$

subject to the constraints

$$\begin{cases} \mathbf{K}_{h,2}\frac{\partial}{\partial z}\mathbf{h}(z_i) \geq d_2, & i = 1, \cdots, N_m \\ \mathbf{K}_{h,2}\hat{\mathbf{I}}_\epsilon^{(j)} = 0, & j = 1, \cdots, N_c \end{cases} \tag{5.80}$$

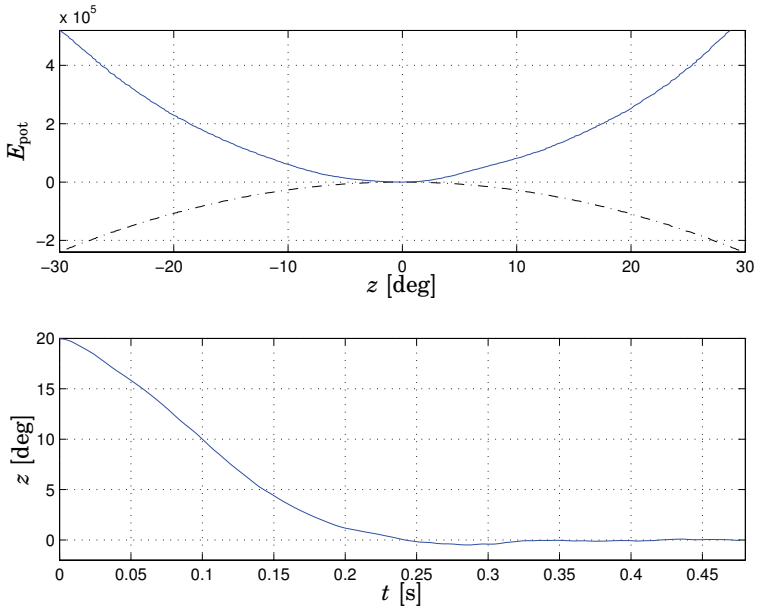

Figure 5.16 *Top plot*: Potential energy as a function of position z for the original system (*dashed*) and the stabilized, reshaped system (*solid*). *Bottom plot*: Simulation results for the stabilized system.

for some finite $d_2 > 0$ used to increase the damping in the system. The approach described above can be seen as a variation of the *energy shaping plus damping injection* approach [Takegaki and Arimoto, 1981; Santibanez and Kelly, 1997]. The potential energy of the system, as a function of x_1, was reshaped by the feedback in order to obtain a unique minimum at the desired equilibrium position, while extra damping was injected through the velocity feedback term $-\dot{u}_2$. In Fig. 5.16 we can see the results of a simulation using such a controller, as well as the potential energy functions for the uncontrolled and the stabilized systems. The system and controller parameters were set to $\omega_0 = 10$, $d = 2$ and $d_2 = 18$, and the lower sector bound for u_1 was given by $k_{\min} = 14$. The controller was sampled at a 4 ms rate, and Gaussian noise with a standard deviation of 5% of the full dynamic intensity range was added to the synthetic images. In practice, the direct differentiation of u_2 in the control law could be replaced by the corresponding approximate ("dirty") derivatives, as is often done in control of elastic joint robots [Kelly *et al.*, 1994]. □

5.4 Experiments and Simulations

In order to validate the intensity-based approach, and to illustrate some possible applications of high-speed visual motion estimation, we have performed a number of experiments and simulations. The examples presented are a simulation study on rigid-body tracking and visual servoing, experiments with hybrid intensity/feature-based tracking, high-speed image-based visual servoing for an industrial manipulator, and an active damping control approach for poorly damped dynamic environments.

Tracking and Servoing of 6-DoF Rigid Motion. The difficulty in obtaining a useful approximation for large motions increases rapidly with the number of degrees of freedom. The reason is that a high-quality approximation in multi-DoF problems requires a very dense sampling of the output space, through the computation and measurement of a very large number of learning data $\tilde{\mathbf{z}}_i$ and $\Delta\mathbf{y}_i$. However, the relatively slow variations of the image intensity variations with $\tilde{\mathbf{z}}$ makes the intensity-based methods feasible also for relatively sparse sampling. In addition, when a multi-scale approach is used, the required number of sampling points is decreased further. In order to investigate the capability to handle full 6-DoF rigid body tracking, a series of simulations were performed. Synthetic 512×400 pixels camera images were generated from virtual stereo cameras. The motion of a cubic box was controlled in 6 DoF as described in Example 5.5, using the controller (5.62)–(5.63) with the same parameters as the example. In all simulations, the total number of measurement points were set to $N = 72$. Measurement noise, implemented as Gaussian, spatially uncorrelated white noise with an amplitude of 10% of the full image intensity range, was added to the images. The convergence and performance of the tracking was investigated with respect to a number of parameters:

- The number N_m of motions $\tilde{\mathbf{z}}_i$ in the learning data set.
- The maximum of $\|\tilde{\mathbf{z}}_i\|$ for the uniformly randomly generated sampling of the output space in the learning data, illustrating the effect of the size of the desired region of convergence.
- The fraction of (random) outliers in the intensity measurements, representing occlusions or specular reflections. It was assumed that each outlier could be represented as a missing measurement, i.e., that elements of $\Delta\mathbf{y}$ corresponding to outliers were replaced with zeros.
- The algorithm used for computing $\mathbf{K_h}$, represented by the stability-based solution of (5.50)–(5.51) or the approximation-based solution

of (5.24)–(5.25). To facilitate comparisons between the solutions, in the latter case the parameter γ was tuned such that the same value (within 5 %) of $\|\mathbf{K_h}\|_F$ was achieved as for the stability-based solution.

To this purpose, a nominal set of parameters was defined. The nominal case was defined by using the stability-based solution with a learning set of $N_m = 1200$ motions, a maximum for $\|\tilde{\mathbf{z}}_i\|$ of 7° and 7 mm, and no measurement outliers. The result of a simulation for the nominal case can be seen in Figs. 5.17–5.18. For the nominal values, the solution of the quadratic optimization problem for $\mathbf{K_h}$ could be computed in 1.6 seconds, using the Matlab Optimization Toolbox on a 1.7 GHz computer. For applied variations to the nominal parameters, the following observations could be made:

- For the given value of $\max(\|\tilde{\mathbf{z}}_i\|) = 7$, the servoing converged for all starting points with $\|\tilde{\mathbf{z}}\| < 7$, even when N_m was decreased as low as $N_m = 48$. However, for convergence when $\max(\|\tilde{\mathbf{z}}_i\|) = 15$ it was necessary to increase the number of sampling points to $N_m > 400$ for convergence.
- With all other parameters as in the nominal case, $\max(\|\tilde{\mathbf{z}}_i\|)$ could be increased to as high as 15° and 15 mm, while guaranteeing convergence for all starting points with $\|\tilde{\mathbf{z}}\| < 15$. For $\max(\|\tilde{\mathbf{z}}_i\|) > 20$ the estimator became unstable due to the poor behavior for small $\|\tilde{\mathbf{z}}\|$, and instability could not be avoided even by increasing N_m to 5000.
- For the nominal case, the fraction of random measurement outliers could be increased to approximately 25-30 % before the estimation error diverged. For constant outliers, i.e., outliers affecting the *same* elements of $\Delta\mathbf{y}$ during each sample, the maximum acceptable fraction was even lower, at roughly 20 %.
- A comparison between the convergence of the estimation errors for the stability-based and the approximation-based solutions can be seen in Fig. 5.19. The stability-based solution results in a both faster and smoother convergence, and is also less sensitive to disturbances. The advantages of the approximation-based solution, on the other hand, are related to its simplicity and low computational complexity. Since the solution of (5.24)–(5.25) requires only the solution of a standard least-squares problem, much larger problems can be solved efficiently than for the stability-based solutions. In the example discussed above, both the number of measurements N and learning motions N_m could be increased by a factor of 5, while giving the same total computation time as for the stability-based solution.

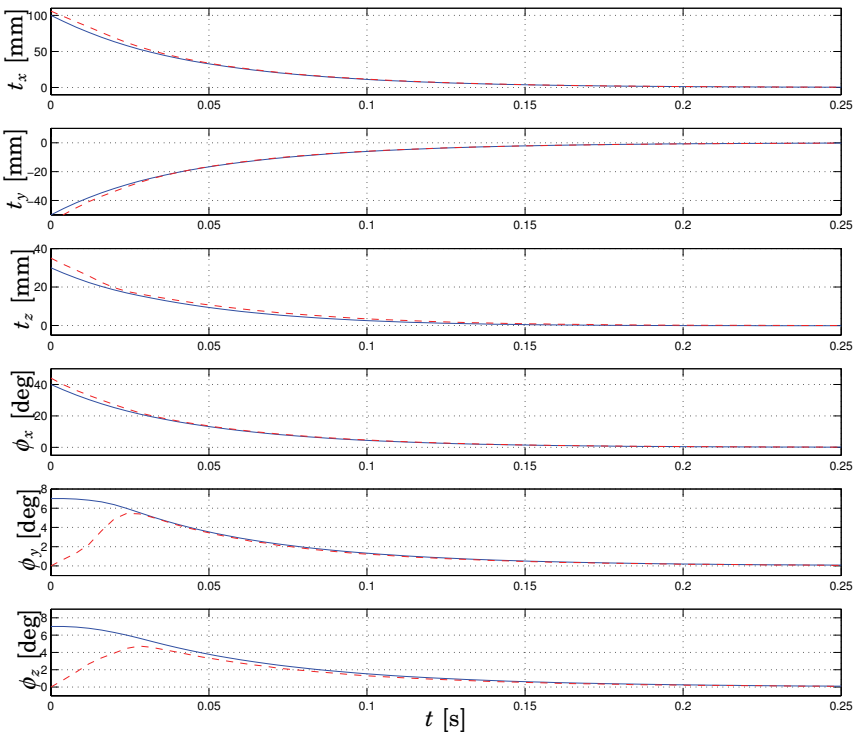

Figure 5.17 Estimated position (*dashed*) and true position (*solid*) in 6-DoF visual servoing simulation, expressed as translations and Euler angles.

Discrete-Time and Hybrid Intensity/Feature-Based Tracking

In many practical situations, relying purely on intensity-based methods may lead to errors, since slow variations in the illumination conditions may introduce undesired drift in the estimates. The feature-based methods of Chapter 4, on the other hand, are known to provide long-term stable estimates [Rosten and Drummond, 2005]. However, many feature-based methods suffer from short-term robustness problems, due to outliers caused by failures in the feature matching process. This is a particularly serious problem for methods using edges or lines, in which nearby and almost parallel edges frequently lead to failures in feature matching, often with loss of tracking as a consequence. Using a combination of intensity-based methods with edge based methods, many of these drawbacks may be avoided [Rosten and Drummond, 2005; Pressigout and Marchand, 2005]. In this way, the robustness and short-term predictive capability of the

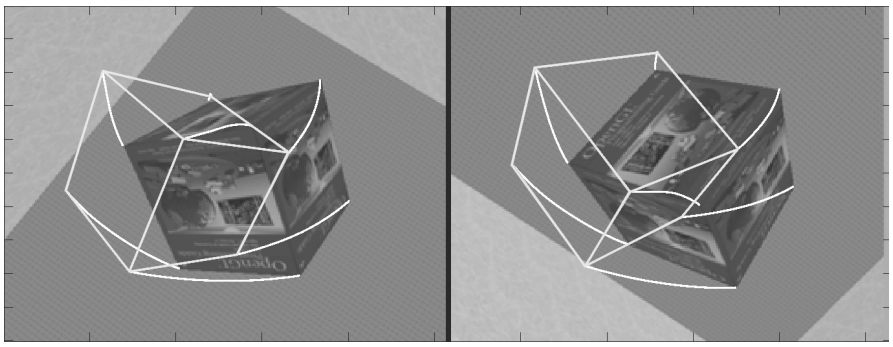

Figure 5.18 Convergence in image-space for the simulated 6-DoF visual servoing in the first and second camera, respectively.

intensity-based method is combined with the drift-free accuracy of feature-based methods. Using a feature-based measurement $\Delta\mathbf{e}_k$ as in Chapter 4 in addition to the intensity-based measurement $\Delta\mathbf{y}_k$, a discrete-time estimator which approximates the behavior of the optimal stationary Kalman filter for the system

$$\mathbf{x}_{k+1} = \mathbf{\Phi}\mathbf{x}_k + \mathbf{\Gamma}\mathbf{u}_k \tag{5.81}$$

$$\mathbf{z}_k = \mathbf{C}\mathbf{x}_k \tag{5.82}$$

is given by

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{\Phi}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{\Gamma}\mathbf{u}_{k-1} \tag{5.83}$$

$$\bar{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_I\mathbf{K}_\mathbf{h}\Delta\mathbf{y}_k \tag{5.84}$$

$$\hat{\mathbf{x}}_{k|k} = \bar{\mathbf{x}}_{k|k} + \mathbf{K}_F\left(\mathbf{h}_e^{-1}(\Delta\mathbf{e}_k, \mathbf{C}\bar{\mathbf{x}}_{k|k}) + \mathbf{C}\bar{\mathbf{x}}_{k|k} - \mathbf{C}\hat{\mathbf{x}}_{k|k-1}\right) \tag{5.85}$$

where the feature-based measurement $\Delta\mathbf{e}_k$ is related to $\mathbf{z}_k$ and the prediction by

$$\Delta\mathbf{e}_k = \mathbf{h}_e(\mathbf{z}_k - \bar{\mathbf{z}}_{k|k}, \bar{\mathbf{z}}_{k|k}) \tag{5.86}$$

is a vector of measured normal distances between the image edges and the predicted edges at position $\bar{\mathbf{z}}_{k|k} = \mathbf{C}\bar{\mathbf{x}}_{k|k}$. As in Chapter 4, the function

$$\mathbf{h}_e^{-1}(\Delta\mathbf{e}_k, \mathbf{C}\bar{\mathbf{x}}_{k|k}) = \mathbf{z}_k - \bar{\mathbf{z}}_{k|k} \tag{5.87}$$

can be computed efficiently from the linearization

$$\mathbf{h}_e^{-1}(\Delta\mathbf{e}_k, \bar{\mathbf{z}}_{k|k}) \approx \mathbf{J}_{v,N}^{-1}(\bar{\mathbf{z}}_{k|k})\Delta\mathbf{e}_k, \tag{5.88}$$

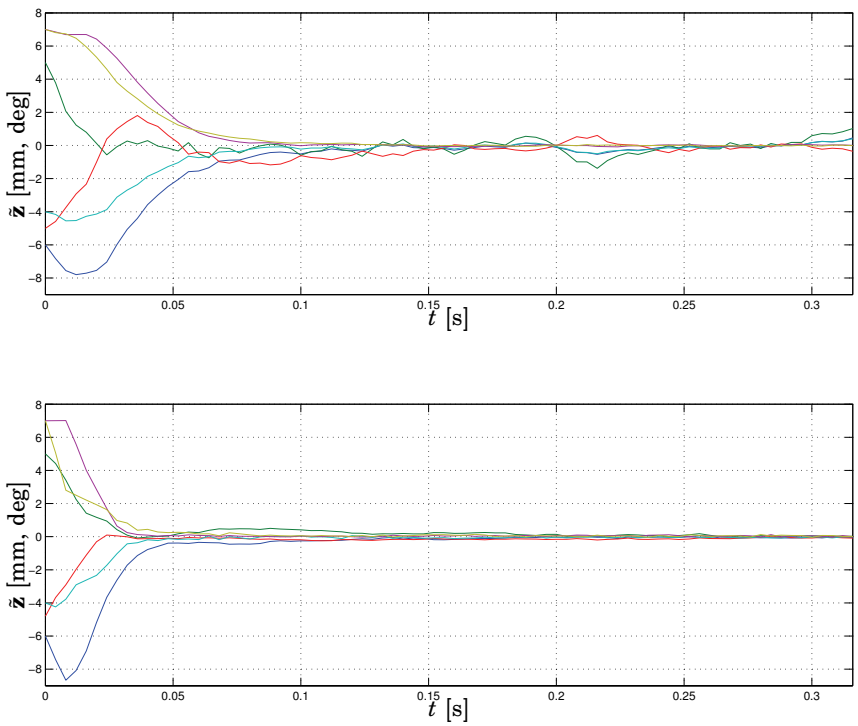

Figure 5.19 Estimation error in the visual servoing simulation, for the approximation-based solution (*top plot*) and stability-based solution (*bottom plot*), respectively.

using the analytically computed Jacobian $\mathbf{J}_{v,N}^{-1}(\bar{\mathbf{z}}_{k|k})$. Because of the comparatively low algorithmic complexity of the intensity-based method presented above, which requires just N simple direct measurements in the image followed by a matrix multiplication, it is ideal for use in a multi-rate estimator. An object tracked using such a multi-rate Kalman filter can be seen in Fig. 5.20, where motion in six degrees of freedom of a three-dimensional target was tracked at 250 images/second. The feature-based correction/measurement update step was executed at a lower rate of 50 Hz. With this estimator it was possible to track edges translating and rotating up to 25 pixels and 7° per frame, even for sequences of irregular motions. For this particular motion sequence, this was roughly twice as fast as what could be achieved using an edge-based tracker.

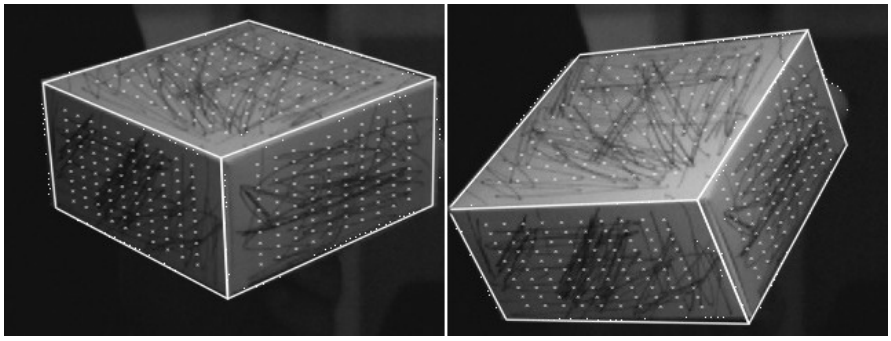

Figure 5.20 Two images from an image sequence showing a weakly textured moving object tracked at a frame rate of 250 frames/second using a hybrid multi-rate tracker. A full video of the sequence can be found at the address http://www.control.lth.se/database/publications/article.pike?artkey=ols07dis.

High-Speed Image-Based Visual Servoing

As an illustration of multiple-DoF control, a setup for a 3-DoF image-based visual servoing experiment was created. A textured planar quadratic object with 13 cm side was attached to the end-effector of an ABB Irb 2400 industrial robot. The Basler A602fc digital camera used in the experiments was intrinsically uncalibrated, and the camera- and robot tool coordinate systems were slightly misaligned, causing a model error which meant that visual feedback was required. A typical image from the camera can be seen in Fig. 5.21.

The robot system was equipped with the external sensor interface described in Chapter 3, and actuated by updating the position/velocity references. A sampled implementation of the continuous-time state estimator (5.19) was used to obtain estimates of the image coordinates of the center point and the image-plane rotation angle of the planar target. The estimator was designed using the approximation-based approach in (5.24)–(5.25), using a multi-scale structure at three different scales. The dynamic model used in the tracker was given by a simple decoupled integrator model from velocity reference to position output, as is commonly assumed in image-based visual servoing systems. The output from the estimator was filtered through a notch filter, in order to remove small remaining effects of the 50 Hz frequency of the indoor illumination. The estimated position was used in an outer 250 Hz proportional visual servo controller, designed to follow a linear trajectory through a number of predefined positions. The results of the image trajectory tracking can be seen in Fig. 5.22. In stationarity, the standard deviation of the noise in the estimated position corresponded to 10–15 μm and 0.01° in Cartesian space,

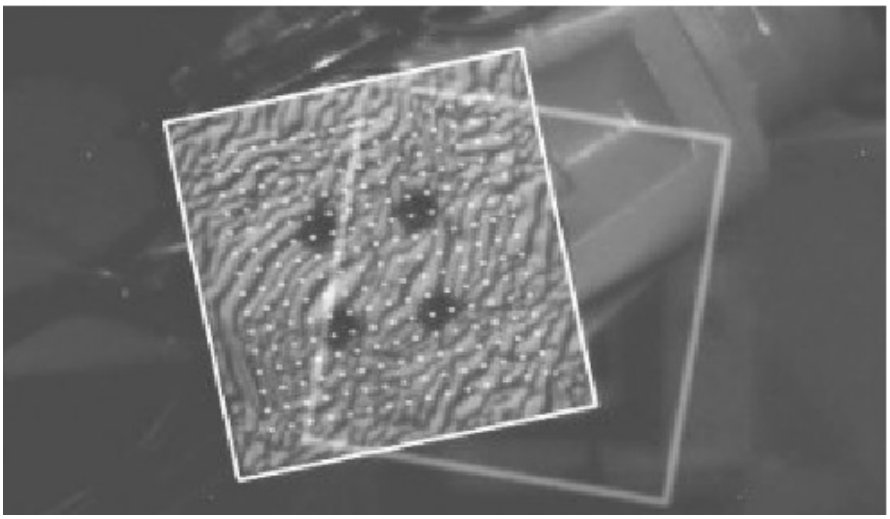

Figure 5.21 Example of an image used for feedback in the image-based visual servoing experiment, with the current reference position superimposed on the image. The full video showing the experiment can be found at the address http://www.control.lth.se/database/publications/article.pike?artkey=ols07dis.

which was 5–7 times more accurate than the guaranteed repeatability of the ABB Irb 2400 robot used in the experiment.

Dynamic Force/Vision Control

Most force control methods focus on the robot dynamics, assuming that the environment can be modeled by ideal constraints, or as simple mass-less (linear or nonlinear) spring-damper systems [Diolaiti *et al.*, 2005]. However, there exist force control applications where more detailed models of dynamic environments are required [Vukobratovic *et al.*, 2004]. Typically, this would be the case when the environment compliance is significantly greater than that of the manipulator, while the inertia or general dynamics of the environment are also non-negligible. Examples from industrial applications are different types of compliant devices, which are mounted between a rigid workpiece and the fixture, serving as an extra mechanical compliance in polishing, grinding and other contact tasks. Other examples are cooperating interacting robots, and control of other types of "weak" mechanical structures with low-frequency resonances. The interaction control problem in dynamic environments is in general more complex than for traditional force control. The manipulator and environment dynamics are coupled through the forces of interaction, which affects the stability and performance of the system in a complex way. A purely force/impedance-

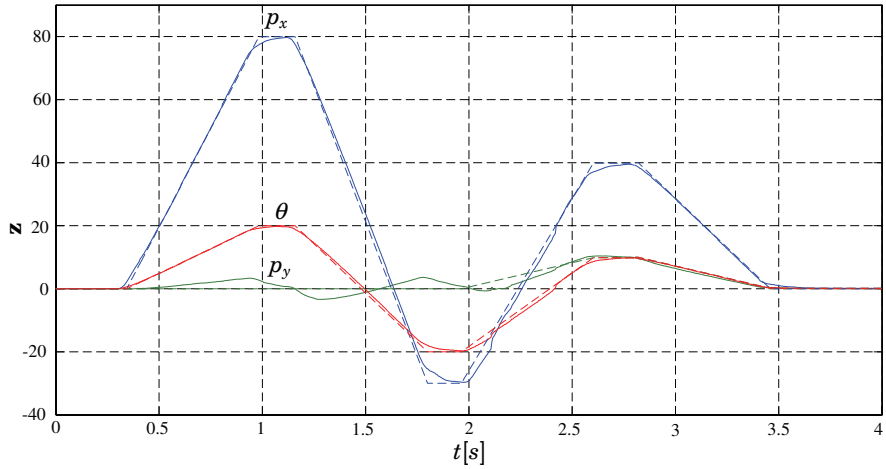

Figure 5.22 Position (*solid*) and position reference (*dashed*) in the image-based visual servoing experiment. The units of translation (p_x and p_y) and rotation (θ) correspond roughly to 1 mm and 1°, respectively.

based approach may not be sufficient for efficient control, since the range of achievable impedances is always limited by force sensor noise and transmission elasticity, and the natural damping of the environment may not be sufficient. Therefore, the dynamic state of the environment must also be included in the modeling and controller design. In the following example, we illustrate the combination of high-speed vision and force control in a simple setup, where it is attempted to actively damp the oscillations of a poorly damped mass-spring-damper system.

Modeling and Controller Design. We assume the setup shown in Fig. 5.23. A robot, considered to be completely rigid in comparison with the environment stiffness, is in contact with a dynamic environment, where the interaction is modeled by contact forces in the surface normal direction at the contact points. To illustrate the main idea, and to simplify the presentation, we will consider the force control problem in each degree of freedom separately. We assume that the robot is internally motion-controlled, where the motion in the direction considered is approximated by a second-order system, with the states being the position z_3 and velocity z_4. The environment dynamics in one direction is modeled as a stable linear second order system with position z_1 and velocity z_2, and we can

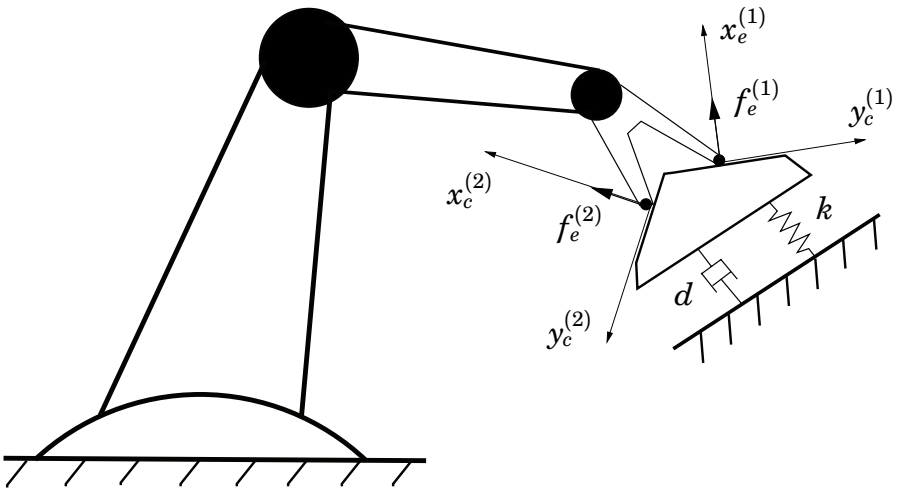

Figure 5.23 Setup with multiple point contacts between robot and environment. Local coordinates $(x_c^{(i)}, y_c^{(i)}, z_c^{(i)})$ are attached to the workpiece at each contact point, while the dynamics of the environment is modeled by a linear mass-spring-damper system.

write the coupled dynamics of the robot and environment as

$$\dot{z}_1 = z_2 \tag{5.89}$$
$$\dot{z}_2 = -kz_1 - dz_2 + \Psi(z_3 - z_1) \tag{5.90}$$
$$\dot{z}_3 = z_4 \tag{5.91}$$
$$\dot{z}_4 = u - k_p z_3 - k_d z_4 \tag{5.92}$$

where the scalar contact force is

$$f_c = \Psi(\delta) \stackrel{\text{def}}{=} K(\delta)\delta, \tag{5.93}$$

where for convenience we have defined

$$\delta \stackrel{\text{def}}{=} z_3 - z_1, \tag{5.94}$$

and where the contact stiffness $K(\delta)$ is assumed to be a differentiable function of δ, satisfying the properties

$$K(\delta) \geq 0, \quad \forall \delta \tag{5.95}$$
$$K(\delta) = 0, \quad \delta < 0 \tag{5.96}$$
$$K'(\delta)\delta + K(\delta) > 0, \quad \delta > 0. \tag{5.97}$$

Eq. (5.93) together with properties (5.95)–(5.97) can be used as an approximate global model for a linear spring mechanism, or a local model for a higher-order (e.g. cubic or Hertzian) contact model. The final inequality (5.97) expresses the physically reasonable assumption that the contact force $f_c = K(\delta)\delta$ increases with an increasing deformation δ. Mathematically, this implies that the "inverse" $\hat{\Psi}^{-1}(f_c)$ of $\Psi(\delta)$, as defined by

$$\hat{\Psi}^{-1}(f_c) = \Psi^{-1}(f_c), \quad f_c > 0 \tag{5.98}$$

$$\hat{\Psi}^{-1}(f_c) = 0, \quad f_c = 0 \tag{5.99}$$

is well-defined. This implies that the contact stiffness function $K(\delta)$ and its derivative can (with some abuse of notation) be expressed as

$$K(f_c) \stackrel{\text{def}}{=} K\left(\hat{\Psi}^{-1}(f_c)\right) \equiv K(\delta) \tag{5.100}$$

$$K'(f_c) \stackrel{\text{def}}{=} K'\left(\hat{\Psi}^{-1}(f_c)\right) \equiv K'(\delta) \tag{5.101}$$

which are defined for all f_c. These functions will be required for the implementation of the controller.

The aim of the controller is to obtain a sufficiently damped impact transition, which can be achieved by controlling the interaction forces suitably. The form of the system in Eq. (5.89)–(5.92) is similar to the so called strict feedback form (or triangular form) [Krstić *et al.*, 1995]. For such systems, the backstepping design method can be used to find a control law and a Lyapunov function in a recursive fashion. If direct control of the interaction force f_c was possible, we could introduce extra damping into the environment by choosing a "virtual" control signal of the form

$$f_c = K(\alpha_1(z_1, z_2) - z_1) \cdot (\alpha_1(z_1, z_2) - z_1) \tag{5.102}$$

with

$$\alpha_1(z_1, z_2) = z_1 + h(z_2) \tag{5.103}$$

where the *damping function* $h(z_2)$ is twice continuously differentiable and chosen to satisfy the properties

$$h(z_2) \geq 0, \quad \forall z_2 \tag{5.104}$$

$$h(z_2) = 0, \quad z_2 > 0. \tag{5.105}$$

In this way, extra damping is introduced by a suitable dissipation of energy by application of a contact force in the opposite direction of motion during the part of the motion when the contact point velocity $z_2 < 0$. This can be seen by introducing the energy-based Lyapunov function

$$V_1(z_1, z_2) = \frac{1}{2}kz_1^2 + \frac{1}{2}z_2^2 \tag{5.106}$$

which gives

$$\begin{aligned}\dot{V}_1 &= kz_1z_2 + z_2(-kz_1 - dz_2 + K(\delta)\,(\epsilon_3 + h(z_2)) = \\ &= \underbrace{-dz_2^2 + z_2K(\delta)h(z_2)}_{\stackrel{\text{def}}{=}-W(z_2,\epsilon_3)\leq 0} + z_2K(\delta)\epsilon_3 \end{aligned} \tag{5.107}$$

where the term $z_2K(\delta)h(z_2) \leq 0$ introduces extra damping due to properties (5.95), (5.104)–(5.105), and where

$$\epsilon_3 = z_3 - \alpha_1(z_1, z_2) = z_3 - z_1 - h(z_2), \tag{5.108}$$

is interpreted as the error between the tool tip position corresponding to the "virtual" control signal and the true position. By defining

$$f_2(z_1, z_2, \epsilon_3) \stackrel{\text{def}}{=} -kz_1 - dz_2 + K(\delta)h(z_2) + K(\delta)\epsilon_3 \tag{5.109}$$

and augmenting V_1 as

$$V_2(z_1, z_2, \epsilon_3) = V_1(z_1, z_2) + \frac{1}{2}p_3\epsilon_3^2, \quad p_3 > 0 \tag{5.110}$$

with a quadratic term in ϵ_3, we obtain

$$\dot{V}_2 = -W(z_2, \epsilon_3) - p_3k_3\epsilon_3^2 + p_3\epsilon_3\epsilon_4 \tag{5.111}$$

with $k_3 > 0$ and the new error signal ϵ_4 given by

$$\epsilon_4 = z_4 - z_2 - h'(z_2)\,f_2(z_1, z_2, \epsilon_3) + k_3\epsilon_3 + p_3^{-1}K(\delta)z_2. \tag{5.112}$$

We now augment V_2 with a quadratic term in the error ϵ_4

$$V_3(z_1, z_2, \epsilon_3, \epsilon_4) = V_2(z_1, z_2, \epsilon_3) + \frac{1}{2}p_4\epsilon_4^2, \quad p_4 > 0, \tag{5.113}$$

and choose the control signal

$$\begin{aligned} u = u_{\text{ffw}} &- (k_3 + k_4)\epsilon_4 + k_3^2\epsilon_3 + k_3p_3^{-1}K(f_c)z_2 - p_3^{-1}\left(K(f_c)f_2(\mathbf{z}, \epsilon_3) + \right. \\ &\left. + K'(f_c)\left(-k_3\epsilon_3 + \epsilon_4 - p_3^{-1}K(f_c)z_2 + h'(z_2)f_2(z_1, z_2, \epsilon_3)\right)z_2\right) \end{aligned} \tag{5.114}$$

with $k_4 > 0$ and the acceleration feedforward term according to

$$\begin{aligned} u_{\text{ffw}} = {} & f_2(\mathbf{z}, \epsilon_3) + h''(z_2)\,f_2(\mathbf{z}, \epsilon_3)^2 + k_pz_3 + k_dz_4 + \\ & + h'(z_2)\left(-kz_2 - df_2(\mathbf{z}, \epsilon_3) + (K'(f_c)(\epsilon_3 + h(z_2)) + K(f_c))\,f_3(\mathbf{z}, \epsilon)\right) \end{aligned} \tag{5.115}$$

where we have used the function

$$f_3(\mathbf{z}, \epsilon) \stackrel{\text{def}}{=} -k_3\epsilon_3 + \epsilon_4 - p_3^{-1}K(f_c)z_2, \tag{5.116}$$

to obtain

$$\dot{V}_3 = -W(z_2, \epsilon_3) - k_3 p_3 \epsilon_3^2 - k_4 p_4 \epsilon_4^2 + p_3 \epsilon_3 \epsilon_4. \tag{5.117}$$

This expression can be made negative semi-definite since p_4 can be chosen arbitrarily. Asymptotic stability of the origin $z_1 = z_2 = \epsilon_3 = \epsilon_4 = 0$ follows from LaSalle's theorem [Krstić *et al.*, 1995], and the fact that the largest invariant set in $\{(z, \epsilon) | \dot{V}_3 = 0\}$ is the origin.

The control law in Eqs. (5.114)–(5.115) requires state feedback, not only from the robot states, but also from the position/velocity of the environment. The multi-scale version of the high-speed intensity-based tracking method was used to provide measurements with sufficient accuracy through

$$\hat{z}_1(t_{k+1}) = \hat{z}_1(t_k) + k_0(\tilde{z}_1(t_k), z_1(t_k)) \tag{5.118}$$

$$\hat{z}_2(t_{k+1}) = (\hat{z}_1(t_{k+1}) - \hat{z}_1(t_k))/(t_{k+1} - t_k) \tag{5.119}$$

with the functions k_j defined recursively according to

$$\begin{aligned} k_{j-1}(\tilde{z}_1(t_k), z_1(t_k)) = \mathbf{K}_{\mathbf{h}}^{(j)} \left(\mathbf{I}(z_0, z_0) - \mathbf{I}(\hat{z}_1(t_k) + k_j(\tilde{z}_1(t_k), z_1(t_k)))\right) + \\ + k_j(\tilde{z}_1(t_k), z_1(t_k)) \end{aligned} \tag{5.120}$$

$$k_{N_s}(\tilde{z}_1(t_k), z_1(t_k)) = 0. \tag{5.121}$$

Each $\mathbf{K}_{\mathbf{h}}^{(j)}$ was found by solution of the quadratic program (5.47)–(5.48), in order to make sure that the resulting nonlinear error terms Δ were norm-bounded.

Simulation. In the experiments, the environment dynamics was given by a poorly damped mass-spring-damper system with mass 20 kg, stiffness 4000 N/m, and linear damping of 20 N/(m/s). The contact stiffness $K(\delta)$ was set to 40000 N/m. The damping function $h(\hat{z}_2)$ was composed of piecewise second order polynomials, as shown in Fig. 5.24, and chosen to correspond roughly to a damping of $d_{\text{active}} = 400$ N/(m/s) in order to obtain a critically damped response. An additional constant term was added to the control signal to obtain a contact force $f_c = 100$ N in stationarity. The intensity-based filter was set to track the translation of the workpiece, using measurements on one of its planar surfaces, from synthetic 320×240 pixels camera images generated with the image generation software described in Appendix A.4. Both the controller and the tracker were

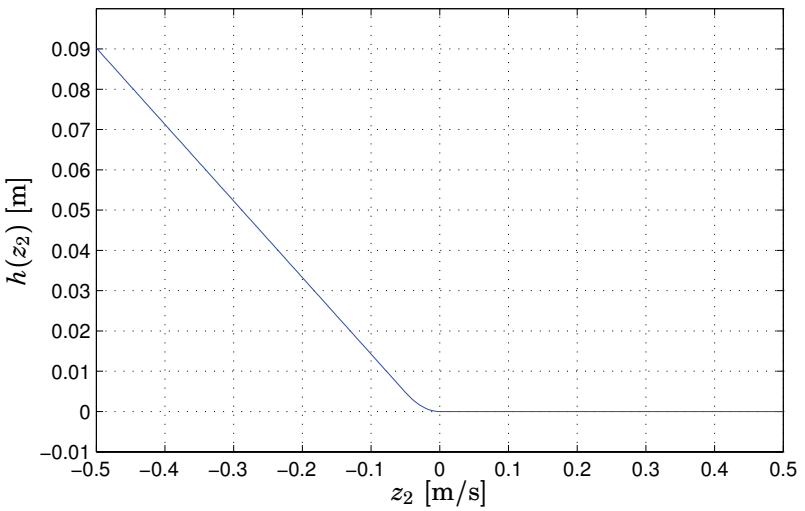

Figure 5.24 Damping function $h(\hat{z}_2)$ composed of piecewise second order polynomials, as used in the simulations.

sampled at 4 ms, and an unmodeled time delay of 8 ms for image capture and transmission was added to the simulation. The controller parameters were set to to $k_3 = k_4 = 20$, $p_3 = 50000$, giving a trade-off between fast convergence and robustness. The robustness was assured by iteratively tuning the parameters such that the controller for the nominal case was able to cope with parameter variations within a certain range, without any serious performance degradation. The range of variations was assumed to be between 50%–200% of the nominal value for the contact stiffness $K(\delta)$, 80%–120% in the environment stiffness k, and 30%–300% in the damping d. Additional image noise with a standard deviation corresponding to 5% of the total intensity range was added to the synthetic images, and white noise of standard deviation 2 N was added to the measured force.

The environment oscillation mode was excited, and at time $t = 1$ s the damping controller was started. Fig. 5.25 shows the resulting robot- and environment positions, while Fig. 5.26 shows the resulting contact force, and the corresponding control signal can be seen in Fig. 5.27. As can be seen, the force during the damping phase in Fig. 5.26 was reduced when the controller was started, while the initial oscillations were damped quickly.

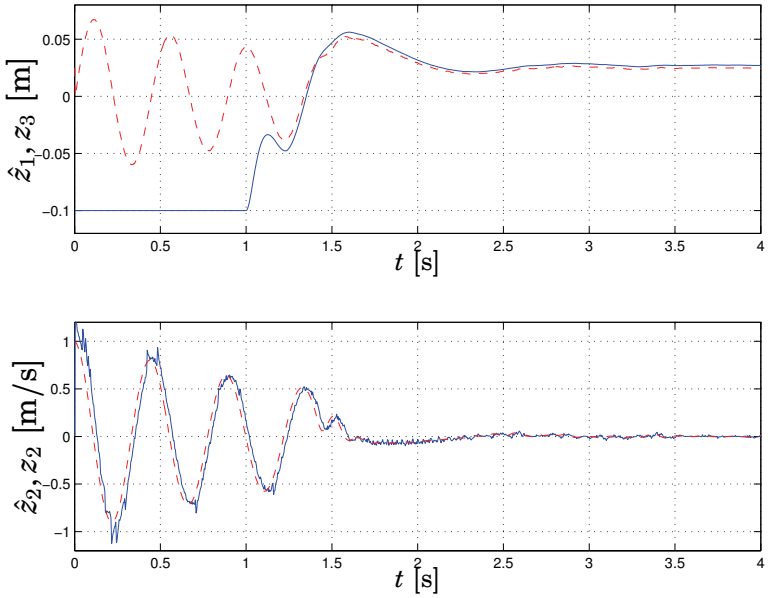

Figure 5.25 *Top:* The estimated environment position $\hat{z}_1$ (*dashed line*) and robot tool tip position z_3 (*solid line*). *Bottom:* True (*dashed line*) and estimated (*solid line*) environment velocities z_2 and $\hat{z}_2$. The controller was switched on at time $t = 1$ s.

5.5 Summary and Concluding Remarks

In this chapter a dynamic visual tracking technique based directly on image intensity measurements was developed and investigated. The use of such methods could be used to obtain state estimates at a very high rate and with very short input-output latency. Such high-speed vision techniques is an interesting approach for real-time measurements and feedback control of complex motions in multiple degrees of freedom. The main advantages of the intensity-based approach are the low time complexity of the online part of the computations, and that robust tracking is possible also when a sufficient number of well-defined and localizable features can not be found in the scene. The drawbacks come from the reliance on the image brightness constancy assumption, although greatly improved performance can be achieved by compensating for simple disturbances in the design.

The methods are based on sampling the motion-intensity relationship in order to build a test set of data, for which the desired properties are enforced in the form of convex optimization criteria and constraints. As

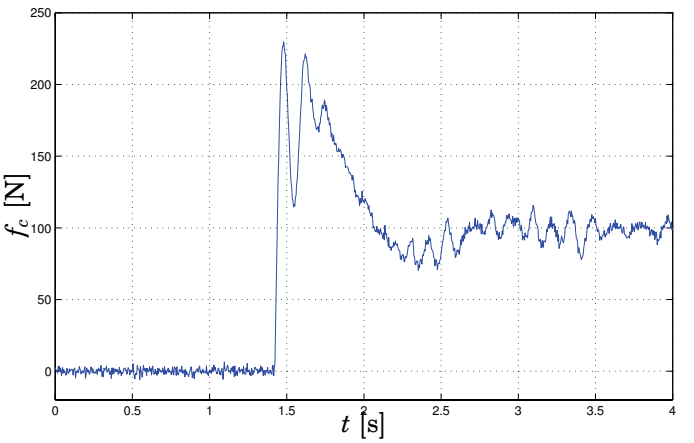

Figure 5.26 Contact force f_c during simulated contact transition and active damping. The controller was switched on at time $t = 1$ s.

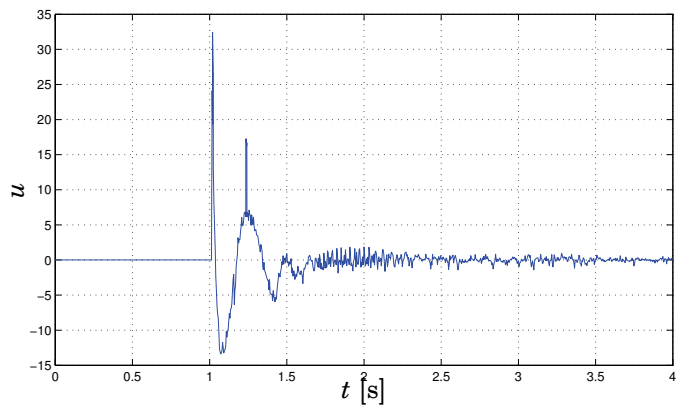

Figure 5.27 Control signal u during simulated contact transition and active damping. The controller was switched on at time $t = 1$ s.

an extension to the standard least-squares solutions, methods for explicitly considering and optimizing suitable stability bounds—such as norm bounds and sector conditions for the approximation errors—have been presented. Such methods frequently outperform the least-squares solutions in dynamic tracking and control problems, such as visual servoing. Better approximations can be achieved using a multi-scale implementation, in which a number of linear approximations of the motion-intensity relationship are computed at consecutively finer scales. This results in a

potentially very accurate approximation over a wide range of motions.

The usefulness of the intensity-based tracking approach, and the improvements presented in this chapter, have been demonstrated in a number of experiments, such as high-speed hybrid intensity/feature-based 6-DoF motion tracking, and 250 Hz image-based visual servoing. Further, a method for force/vision control of contact with poorly damped environments has been presented.

6

A Study on Force Control for Accurate Low-Cost Robot Drilling

6.1 Introduction

Systems for automatic drilling have a long history both in industry and in the research community. In particular, the use of industrial robots for drilling is interesting due to their flexible programming and the comparatively low cost of industrial robot systems. However, robot drilling is a very challenging task due to the poor mechanical stiffness of the typical serial industrial robots in use today. This compliance makes the robot deflect due to the cutting forces, with poor hole quality as a result. Nevertheless, a number of industrial robot systems for drilling exist. Traditionally, such systems have been based on mechanical solutions, using large-size robots and customized, high-cost drilling end-effectors. In addition, different devices for rigidly attaching the drilling tool to the surface are commercially available, for instance, based on vacuum suction or electromagnetic devices. In many situation, such devices provide very robust solutions, although somewhat inflexible.

In many drilling tasks, for example in aircraft assembly, components consisting of several layers of material are drilled. In such cases, it is important to simultaneously apply pressure to the surface in order to make sure that no chips or other material from the drilling are lodged between the layers, in which case the entire structure would have to be manually disassembled and cleaned. The pressure force which is applied must therefore be controlled during the entire drilling phase, which makes

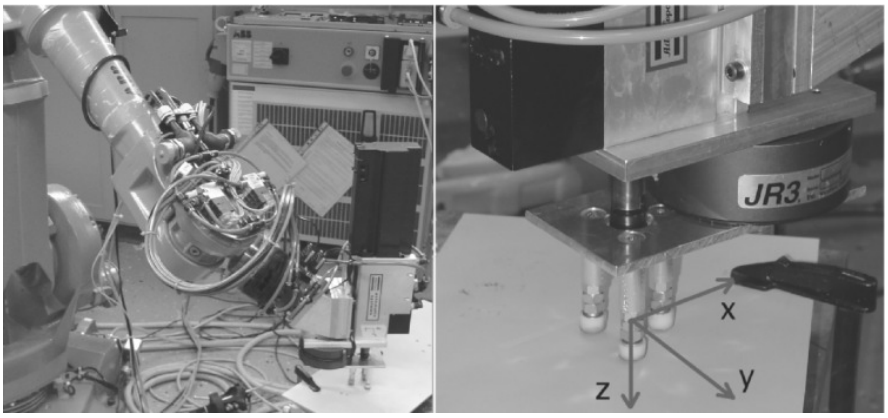

Figure 6.1 Robot, drilling tool and tripod, with JR3 force sensor mounted on the drilling tool. As seen in the figure, the axial direction is denoted z, while the x and y-directions are tangential to the surface during drilling.

high-bandwidth feedback techniques an attractive alternative to mechanical solutions. Research and development on force-controlled drilling has not received as much attention as many other applications of industrial force control, such as assembly, deburring, milling, or polishing. The reason is probably the difficulties involved in robotic drilling, as well as the lack of available industrial robot systems with capacity for sufficiently high-bandwidth force control. Some results on force control for special drilling machines have been reported in [Kawaji *et al.*, 2001]. Experimental systems for force-controlled robot drilling have been presented in [Alici, 1999], where a force controller with inner-loop position control was used for the drilling thrust force control, and in [Lee and Shih, 2006], where an application to bone drilling in orthopedic surgery was presented. In addition, numerous research papers and patents related to robot drilling exist, which are based on mechanical solutions or special-purpose end-effectors rather than force control.

Problem Description

The purpose of this chapter is to develop and evaluate techniques for force control using an industrial robot setup, shown in Fig. 6.1. The setup consisted of a robot holding a pneumatic drilling tool with a linear feed mechanism. A pressure foot in the form of a tripod was mounted on the drilling tool, as shown in the right part of the figure. The goal was to apply a constant normal force to the drilled surface with the tripod prior to drilling, and to keep the tangential forces small enough to avoid sliding of the drilling tool on the surface during drilling. The undesired sliding was

due to the compliance in the robot transmission, links and environment, and could be up to several millimeters without compensation, seriously degrading both quality and positioning of drilled holes. The proposed solution was based on a combination of high-bandwidth control of the axial forces applied to the workpiece, and active suppression of the sliding forces through a model-based force control scheme. The forces were measured using a stiff six-axis force/torque sensor, mounted directly between the drilling machine and the tripod, see Fig. 6.1. A separate, pneumatically driven axis of the drilling machine was feeding the drill along the tripod central axis and into the material. The feasibility of the proposed method was demonstrated in drilling experiments using an industrial robot system.

6.2 Modeling and Control

The force control structure was based on the extended robot systems presented in Chapter 3. In order to use the extended robot system in applications requiring model-based control techniques, a dynamic model of the system responses to external forces and motion references was needed. For these purposes, the modeling and tuning techniques in Section 3.3 was used, resulting in a model for the robot motion responses to both control actions and external forces.

Environment properties

During stiction contact between the tripod and the drilled component, the contact behavior was similar to a very stiff and poorly damped spring, as predicted by many friction models such as the LuGre model [Canudas de Wit *et al.*, 1995]. When the tangential forces became larger than the breakaway forces of the stiction, the tripod started to slide across the surface, with poor hole quality and positioning as a result. Therefore, it was important both to control the tangential forces so that sliding was avoided, and to control the moments to keep the tripod in contact with the surface at each of the three contact points. Once such a contact had been achieved, the dependence of the contact force $\mathbf{f}_e$ on the tool position $\mathbf{p}_a$ was expressed through the environmental dynamics. The resulting high-gain feedback loop affected the stability and performance of the manipulator. For a point contact, the environment dynamics can often be approximated by a local stiffness, or as a (nonlinear) spring-damper [Diolaiti *et al.*, 2005]. For the tripod contact of the drilling tool used in this work, it was necessary to take also the geometry of the contact into account. The contact was considered as a combination of three-point contacts, where the

force acting at each point contributed to the effective force and moment acting at the robot TCP point. General frameworks for multi-contact situations have previously been demonstrated, e.g., in [Park and Khatib, 2005] using an operational space formulation. In the drilling application, using a position/velocity-controlled robot, the contact properties of the small tripod could be expressed as a (non-diagonal) stiffness matrix. For the model of the 5-DoF system used in the experiments, the tool position $\mathbf{p}_a = [\,\mathbf{t}_a^T \quad \boldsymbol{\varphi}_{a,x} \quad \boldsymbol{\varphi}_{a,y}\,]^T$ was represented by the three translations $\mathbf{t}_a$ and the two rotation angles $\boldsymbol{\varphi}_{a,x}$ and $\boldsymbol{\varphi}_{a,y}$ around the x and y-axes (see Fig. 6.1), expressed in a fixed coordinate system which was taken to coincide with the initial position of the tool, $\mathbf{p}_a = \mathbf{0}$. The coordinates $\mathbf{X}_i^w$ of each contact point in the world coordinate system were related to $\mathbf{p}_a$ by

$$\begin{aligned}\mathbf{X}_i^w &= \mathbf{R}_x(\boldsymbol{\varphi}_{a,x})\mathbf{R}_y(\boldsymbol{\varphi}_{a,y})\mathbf{X}_i^{TCP} + \mathbf{t}_a \approx \\ &\approx \begin{bmatrix} 1 & 0 & 0 & 0 & \mathbf{z}_i^{TCP} \\ 0 & 1 & 0 & -\mathbf{z}_i^{TCP} & 0 \\ 0 & 0 & 1 & \mathbf{y}_i^{TCP} & -\mathbf{x}_i^{TCP} \end{bmatrix} \mathbf{p}_a \stackrel{\text{def.}}{=} \mathcal{X}_i \mathbf{p}_a \end{aligned} \tag{6.1}$$

where $\mathbf{X}_i^{TCP} = [\,\mathbf{x}_i^{TCP} \quad \mathbf{y}_i^{TCP} \quad \mathbf{z}_i^{TCP}\,]$ described the TCP-coordinates of the contact points of the tripod. Assuming a linear stiffness $\mathbf{f}_i = \mathbf{K}_i \mathbf{X}_i^w$ at each contact point, the full contact stiffness model was given by

$$\mathbf{f}_e = \sum_{i=1}^{3} \left(\begin{bmatrix} \mathbf{K}_i \mathcal{X}_i \\ [\mathbf{X}_i^{TCP}]_\times \mathbf{K}_i \mathcal{X}_i \end{bmatrix} \right) \mathbf{p}_a \stackrel{\text{def.}}{=} \mathbf{K}_e \mathbf{p}_a. \tag{6.2}$$

For the setup described in this work, where the contact points were placed symmetrically on a circle in the xy-plane with radius r around the origin, and where each point stiffness matrix $\mathbf{K}_i = \text{diag}(k_x, k_y, k_z)$ was assumed to be completely decoupled, the contact model was also decoupled with stiffness matrix

$$\mathbf{K}_e = \text{diag}(3k_x, 3k_y, 3k_z, 1.5r^2 k_z, 1.5r^2 k_z). \tag{6.3}$$

The contact model in Eq. (6.2) provided a useful local approximation during stiction, and the objective of the control was to keep the system in this stiction regime. This resulted in a multi-DoF force control problem in a stiff environment, which is known to be a very challenging problem [Colgate and Hogan, 1989; De Schutter and Van Brussel, 1988b; Siciliano and Villani, 1999].

Control Design

In the control design it is necessary to take both robot dynamics and environment properties into account. Therefore, in addition to the model-based control, the option to tune controllers manually in order to account for poorly modeled or varying environment parameters is desired. Automatic design procedures for force controllers have previously been presented [Natale *et al.*, 2000], but they are not suitable for the drilling application due to the significantly higher contact stiffness and special control objectives. Instead, we propose a control strategy based on an easily tunable force controller using an inner motion controller with model-based disturbance rejection and decoupling. In inner-motion force control, the measured contact force and force reference are used as inputs to a motion- or impedance equation. This relation is often chosen as a passive second-order system in order to emulate the behavior of a passive mass-spring-damper. The robot motion controller is set to track the output position from the impedance equation. Because of the limited bandwidth of the motion control system and the deformations of the robot caused by external forces, the tracking of the desired motion may be poor when the robot is in contact with a stiff environment.

In order to improve the tracking performance, the inner motion control should be redesigned to include external force compensation. This can be seen as trying to increase the "stiffness" of the robot as seen from the tool, which improves the ability to control contact forces and moments. To this purpose, a controller structure which includes this inner loop compensation as in Fig. 6.2 was used. In order to track the desired position obtained by integrating the impedance relation, the inner motion controller should have both a fast arm side response to motion commands, and good suppression of external forces up to the desired bandwidth of the system. In addition to force sensors, which can be used to obtain improved disturbance suppression through feedforward, feedback from arm side position measurements could be used in the inner controller to improve the accuracy of the positioning. Such measurements could be obtained from, e.g., cameras or laser trackers. Here, force measurements and an H_∞-optimal inner controller were used to give a faster and more decoupled response in contact. The discrete-time robot model obtained in Chapter 3, together with a force sensor low-pass filter, can be written in input-output form according to

$$\mathbf{p}_a(z) = \mathbf{G}_{ar}(z)\mathbf{p}_r(z) + \mathbf{G}_{af}(z)\mathbf{f}_e(z) \tag{6.4}$$

$$\mathbf{p}_m(z) = \mathbf{G}_{mr}(z)\mathbf{p}_r(z) + \mathbf{G}_{mf}(z)\mathbf{f}_e(z) \tag{6.5}$$

$$\mathbf{f}_{e,f}(z) = \mathbf{G}_{LP}(z)\mathbf{f}_e(z). \tag{6.6}$$

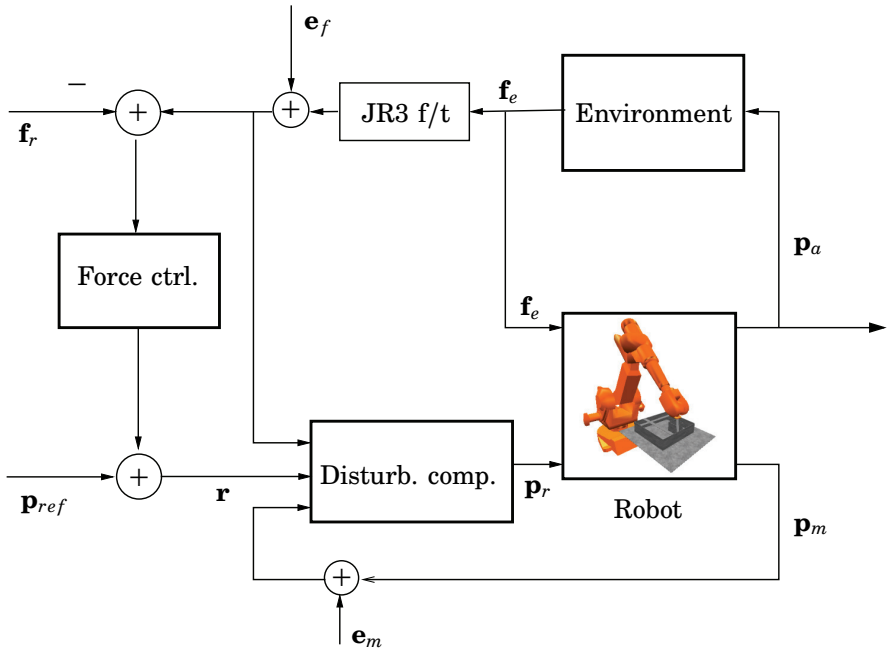

Figure 6.2 Simplified structure of the control system for drilling, with an outer force control loop and inner-loop disturbance compensation.

Using the controller

$$\mathbf{p}_r(z) = \mathbf{r}(z) - \underbrace{(\mathbf{C}_m(z)\mathbf{p}_m(z) + \mathbf{C}_f(z)\mathbf{f}_{e,f}(z))}_{\mathbf{v}(z)} \tag{6.7}$$

the model of the inner loop system was given by

$$\begin{aligned}\mathbf{v} &= (\mathbf{I} + \mathbf{C}_m\mathbf{G}_{mr})^{-1}(\mathbf{C}_m\mathbf{G}_{mf}\mathbf{W}_f\mathbf{f}_e + \mathbf{C}_f\mathbf{G}_{LP}\mathbf{W}_f\mathbf{f}_e + \\ &\quad + \mathbf{C}_m\mathbf{W}_{dm}\mathbf{e}_m + \mathbf{C}_f\mathbf{W}_{df}\mathbf{e}_f) \triangleq \mathbf{G}_v\,[\,\mathbf{f}_e^T \quad \mathbf{e}_m^T \quad \mathbf{e}_f^T\,]^T \qquad (6.8)\\ \mathbf{p}_a &= \mathbf{G}_{af}\mathbf{W}_f\mathbf{f}_e - \mathbf{G}_{ar}\mathbf{v} \triangleq \mathbf{G}_a\,[\,\mathbf{f}_e^T \quad \mathbf{e}_m^T \quad \mathbf{e}_f^T\,]^T \qquad (6.9)\end{aligned}$$

where $\mathbf{e}_f$ and $\mathbf{e}_m$ modeled measurement noise, and the weighting transfer functions $\mathbf{W}_i$ were chosen to give a proper suppression of disturbances $\mathbf{f}_e$ at the arm side position $\mathbf{p}_a$, for frequencies up to approximately 25% of the mechanical bandwidth. This lead to the optimization problem

$$\min_{\mathbf{C}_m,\mathbf{C}_f} \left\| \begin{bmatrix} \mathbf{W}_v\mathbf{G}_v \\ \mathbf{W}_a\mathbf{G}_a \end{bmatrix} \right\|_\infty \tag{6.10}$$

which was solved to give the controller transfer functions $\mathbf{C}_m(z)$ and $\mathbf{C}_f(z)$, using standard H_∞-optimization methods [Zhou and Doyle, 1998]. In order to simplify real-time implementation, the high-order controllers obtained were reduced to order 15 using balanced model reduction [Zhou and Doyle, 1998].

The force controller was designed by manual tuning of the parameters in a desired decoupled impedance in the form

$$\mathbf{M}_I \frac{d^2}{dt^2}\Delta\mathbf{p} + \mathbf{D}_I \frac{d}{dt}\Delta\mathbf{p} = \mathbf{f}_{e,f} - \mathbf{f}_r \tag{6.11}$$

$$\mathbf{r} = \mathbf{p}_{ref} + \mathbf{K}_{dc}\Delta\mathbf{p} \tag{6.12}$$

with $\mathbf{M}_I$ and $\mathbf{D}_I$ diagonal matrices. Since the inner loop design was based on a 3-DoF model with translation only, a static decoupling matrix $\mathbf{K}_{dc}$ was included for improved decoupling between the control of xy-torques and xy-forces. A proper choice for $\mathbf{K}_{dc}$ could be found from the static calibration data in Section 3.3. Here, $\mathbf{K}_{dc}$ was chosen such that for all unit basis vectors $\mathbf{e}_i$

$$\Delta\mathbf{p}_{m,i} = \mathbf{K}_{dc}(\mathbf{e}_i^T \Delta\mathbf{p}_{m,i})\mathbf{e}_i \tag{6.13}$$

where $\Delta\mathbf{p}_{m,i}$ was the motor side motion required for a force change $\Delta f_e \mathbf{e}_i$ in stationarity. The arm side response of the full system to external forces in Fig. 6.2 was described by the transfer matrix

$$\mathbf{G}_{\text{tot}}(z) = \mathbf{G}_{c,af}(z) + \mathbf{G}_{c,ar}(z)\mathbf{G}_I(z) \tag{6.14}$$

where $\mathbf{G}_{c,af}(z)$ and $\mathbf{G}_{c,ar}(z)$ were the responses in the tool position $\mathbf{p}_a$ of the closed inner loop to forces $\mathbf{f}_e$ and motion references $\mathbf{r}$, and $\mathbf{G}_I(z)$ represented the discretized dynamics in Eqs. (6.11)–(6.12) from applied force to $\mathbf{r}$. The stability of the resulting system could be analyzed by considering a system with $\mathbf{G}_{\text{tot}}(z)$ connected to the environment dynamics in a simple feedback loop.

6.3 Experiments

The drilling experiments were carried out on an ABB Irb 2400 industrial robot using the external sensor interface described in Chapter 3. The robot was equipped with a pneumatic Atlas Copco LBL25 drilling machine with 4 mm drill diameter. A number of experiments were performed using different robot configurations. The contact forces were measured using a JR3 force/torque sensor, and the workpiece was a 3.5 mm thick plate of high-strength aluminum.

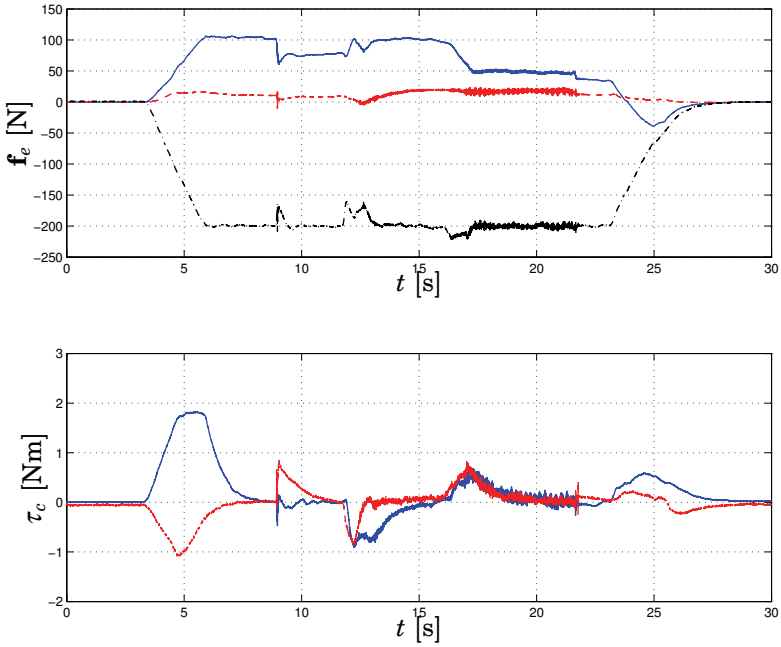

Figure 6.3 The forces during a drilling experiment using the built-in motion controllers for the inner-loop control, with no control of the sliding forces. Top: Sliding forces in the x-direction (*solid*) and in the y-direction (*dashed*), and normal (axial) forces (*dash-dotted*). Bottom: Contact moments acting on the TCP point around the x-axis (*solid*) and y-axis (*dashed*). In this case, large tangential forces were built up in the uncontrolled x-direction. For purposes of illustration, the signals have been filtered by a narrow-band notch filter in order to remove the effect of the drill rotation at a frequency of 2100 RPM.

3-Degree-of-Freedom Control

In the first set of experiments, the built-in motion control of the robot was used without force compensation in the inner loop. The axial environment stiffness was approximately 150 N/mm. The approximate stiffness of the robot and tool, with respect to forces applied at the drill tip, was 160 N/mm in the axial z-direction, and 100 N/mm and 50 N/mm in the tangential x and y-directions. There was also a significant static coupling, resulting in a tangential deflection when axial forces were applied to the drill. In Figs. 6.3–6.4 the results from one of the drilling experiments with 3-DoF force/torque control is shown. The axial z-force and the moments around the x and y-axes were controlled such that a stable contact was achieved with a total axial force of 200 N. However, although the axial

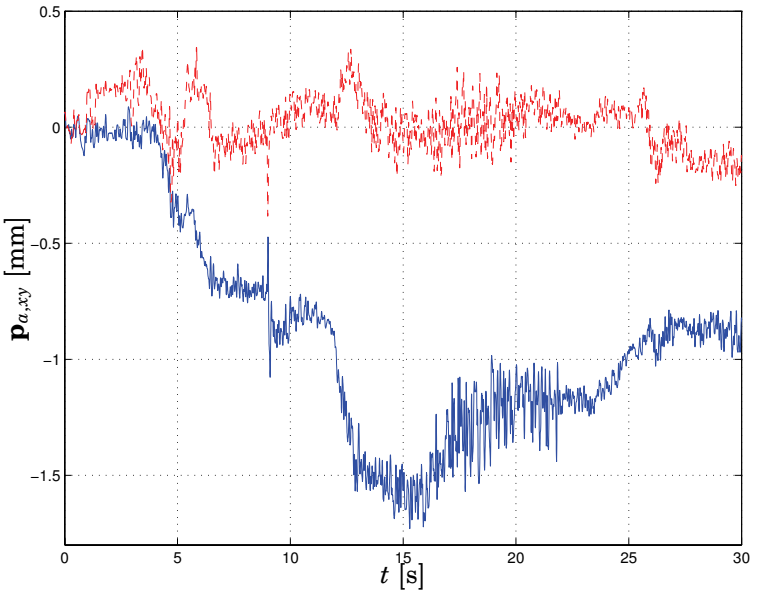

Figure 6.4 The linear motion of the tool in the x and y-directions during a drilling experiment using the built-in motion controllers for the inner-loop control, with no control of the sliding forces. Time $t = 4$ s corresponded to the start of the force build-up phase, and the start and end times of the drilling phase occurred at $t = 12$ s and $t = 16$ s. Undesired sliding of approximately 1.6 mm occurred in the tangential x-direction, caused by the variations in the axial pressure forces.

force in Fig. 6.3 was accurately controlled to the desired value, the friction forces between the tripod and the surface were not sufficient to be able to suppress the sliding motion of the tool. This can be seen in Fig. 6.4, as sliding occurred primarily in the x-direction, both during the application of the pressure foot onto the surface, and when the cutting forces were applied during the drilling. When forces were applied, the tripod contact switched from stiction to slip and back again several times. This behavior is also indicated in Fig. 6.3, which shows the presence of large x-forces with discontinuities at transitions between stiction and slip. The total deflection in the experiment was approximately 1.6 mm, of which 0.8 mm occurred during the drilling phase. Both the positioning and quality of the resulting holes were unsatisfactory.

5-Degree-of-Freedom Control

In the next set of experiments a 5-DoF force/torque control with an inner-loop force compensation was used, in which the forces in the x- and y-

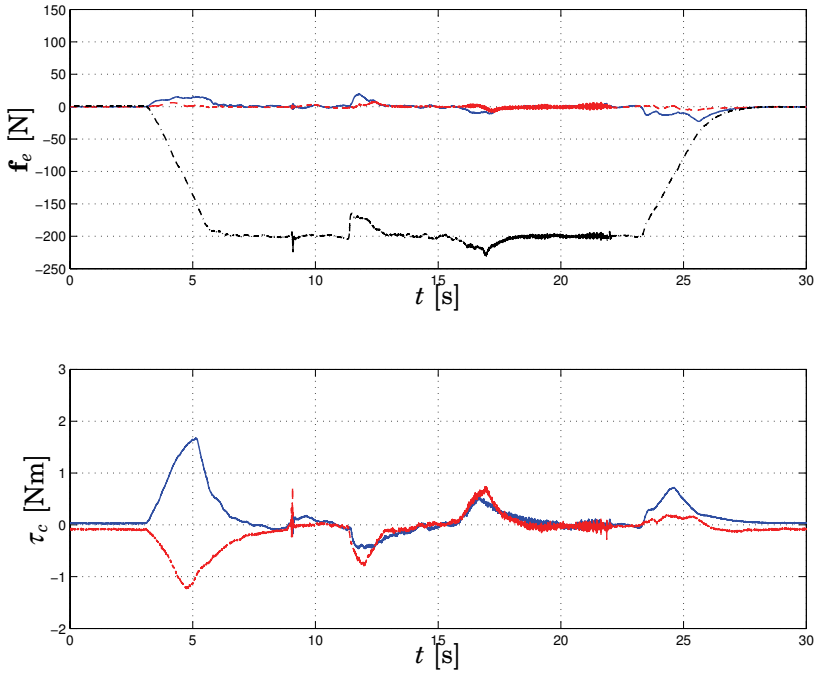

Figure 6.5 The forces during a drilling experiment using using an inner-loop controller with compensation for the robot compliance, and with active control of the sliding forces. Top: Sliding forces in the x-direction (*solid*) and in the y-direction (*dashed*), and normal (axial) forces (*dash-dotted*). Bottom: Contact moments acting on the TCP point around the x-axis (*solid*) and y-axis (*dashed*). The tangential forces were controlled to keep the friction contact in the stiction regime.

directions were controlled in order to suppress sliding. Figs. 6.5–6.6 show the results of a drilling experiment where this controller was used. Except for this change of controller, all other parameters were identical to the previous set of experiments. In Fig. 6.6 it can be seen that the tool deflection when forces were applied during the force build-up was reduced to approximately 0.1 mm, and sliding during the drilling phase was reduced to 0.1 mm. A spectrogram plot of the forces can be seen in Fig. 6.7, showing the spectral characteristics of the disturbance forces from the drill. Having performed a number of experiments in different configurations, the model-based force controller was always able to control the sliding forces so that the tripod contact remained in the stiction regime during the entire drilling operation, and the tangential deformation was reduced significantly.

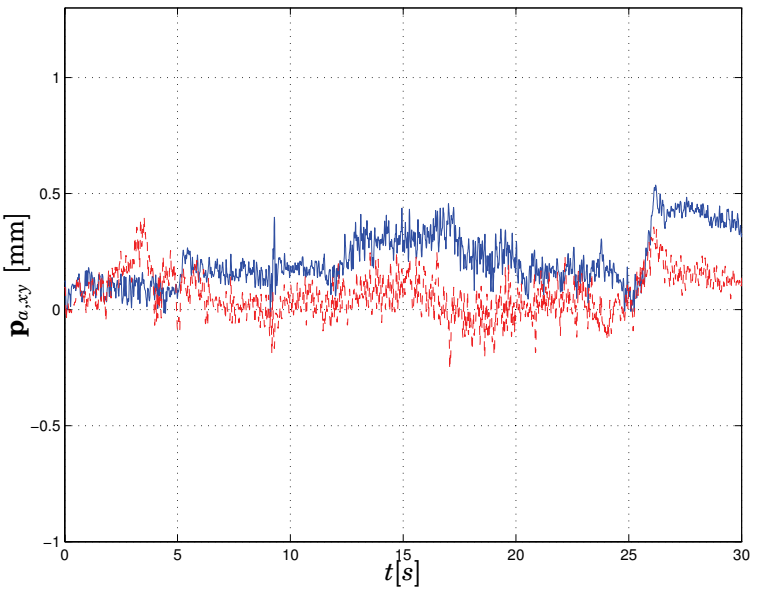

Figure 6.6 The linear motion of the tool in the x and y-directions during a drilling experiment using an inner-loop controller with compensation for the robot compliance, and with active control of the sliding forces. The drill sliding was reduced in the critical drilling phase by a factor of five as compared to the previous case.

In order to more accurately evaluate the motion during application of contact forces and drilling, measurements were also performed using a 3-DoF Leica laser tracker. The reflector prism of the tracker system was attached to the pressure foot, approximately 25 mm above the point of contact between the surface and the drill (Fig. 6.8). The resulting measured tool sliding can be seen in Fig. 6.9, where it can be seen that the maximum deflection was well below 0.2 mm, confirming the previous measurements.

6.4 Summary and Concluding Remarks

The experiments indicate that force control of the pressure forces is feasible for drilling tasks. The full 5-DoF force/torque control resulted in greatly improved mechanical stiffness and vibration suppression, leading to significant improvements in hole quality and positioning. The use of an industrial robot and a small tripod also provides good dexterity and flexible usage, making the system operable in a large workspace and complex structures. The use of a tripod limits the system to drilling perpendicu-

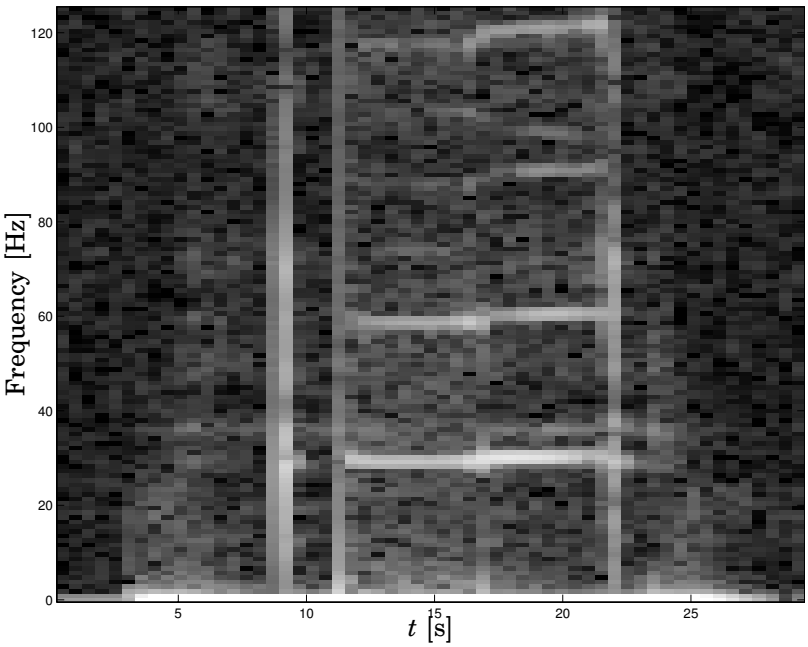

Figure 6.7 Spectrogram plot of the axial forces during a drilling experiment using an inner-loop controller with compensation for the robot compliance. Note the broadband impact phenomenon when the drill feed is started at $t = 9$ s, and the harmonics exhibited during the drill phase.

larly to the surface, but together with the torque control also helps obtain good normality of the drilled holes.

As an alternative to controlling the position using arm-side position feedback, the controller attempts to achieve sliding suppression by ensuring that the tangential interaction forces are always small enough to keep the interaction in the stiction regime. In practice, the achievable bandwidth of the force control is limited by the mechanical bandwidth of the robot, as well as by the bandwidth of the inner motion control. Instead, high-frequency disturbances are damped out by the tripod high-friction contact, providing extra mechanical stiffness against disturbances such as vibrations from the feeding and rotation of the drilling tool. The force control and active sliding suppression takes care of large disturbances at lower frequencies, such as the slower variations of the cutting forces. Thereby, a system which is able to reject disturbances over a wide frequency range is obtained, at a very low cost.

The drilling force control system differs from most other applications of force control, such as polishing, grinding, and assembly, where the force

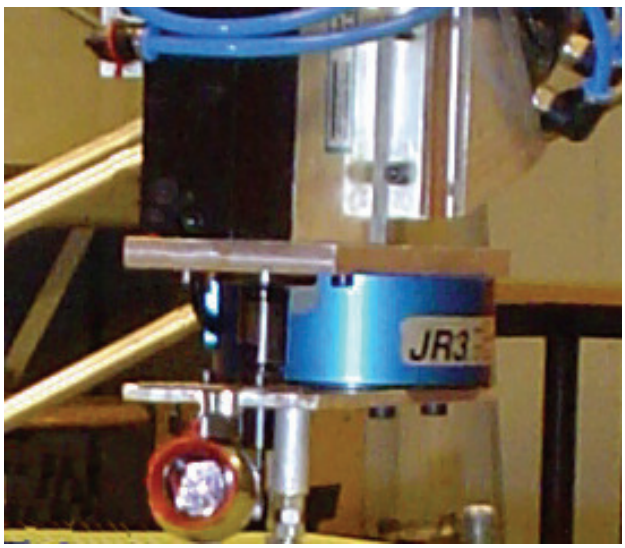

Figure 6.8 Drilling tool with attached reflector prism, used for measurement with the Leica laser tracker. The prism was attached to the tripod, giving accurate measurements of the translations at the drill tip.

control is used to increase the compliance rather than to improve the stiffness to force disturbances. In the drilling system, the model-based inner-loop compensation improves the stiffness, using one or several local models of the robot stiffness and dynamics. In order to experimentally obtain and tune such models, arm side position measurements must be available. In cases when such measurements are not available, similar results can in some configurations be obtained without inner loop compensation, using proper tuning of the outer force controller. However, the couplings between different degrees of freedom may lead to poor performance, and attempts to increase the bandwidth result in limit cycles and oscillations in the contact force.

The use of industrial robots in automatic drilling applications has been limited, mainly due to the presence of rapidly varying interaction forces in combination with compliance in gear boxes and links. Functionality for high-bandwidth force control in modern industrial robot control systems could potentially lead to robotic drilling systems with significantly improved performance, without the use of costly hardware modifications and calibration procedures. In this chapter, we have presented methods and systems for force-controlled robot drilling. Using a 6-DoF force/torque sensor, an outer force control loop and a model-based inner-loop disturbance compensation scheme have been designed, and used to control the axial

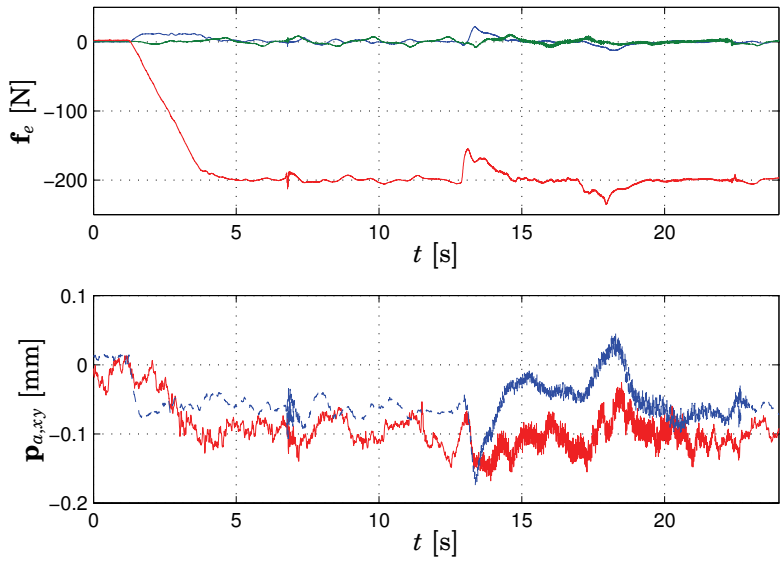

Figure 6.9 (*Top plot:*) Filtered contact forces during a drilling experiment using an inner-loop controller with compensation for the robot compliance, and with active control of the sliding forces. (*Bottom plot:*) Unfiltered linear deflection of the drilling tool in the x (*solid*) and y-directions (*dashed*), as measured by the Leica laser tracker.

contact force and suppress the sliding of a tripod contact while the drilling is performed. The advantage of the proposed controller is demonstrated in reproducible drilling experiments using a medium-sized industrial robot system.

7

Conclusion

7.1 Summary

This thesis has dealt with questions related to feedback from two very different types of sensors: digital cameras and force/torque sensors. Apart from a number of topics related to visual motion estimation and implementation of force control, several different approaches to combination of these two types of sensors have been considered. In the treatment of the vision-related topics it has been attempted to take a control-oriented approach, rather than a more vision-oriented approach based on for instance projective and epipolar geometry. In general, vision-oriented approaches are relevant in control problems where a modular structure of visual estimation and control is used, such as the position-based algorithm in Chapter 4. On the other hand, a good example of the advantages of a control-oriented approach is provided by the stability-oriented versions of the intensity-based tracking presented in Chapter 5. In such problems, exploiting the dynamic nature of the tracking problem makes it possible to relax the requirement of least-squares optimality of the approximation, thereby obtaining systems exhibiting greatly improved tracking performance, both in theory and in practice.

As a summary, the work, contributions and conclusions drawn can roughly be divided into the following categories.

Robot System Sensor Interface and Applications

A new interface for external sensor control for a standard industrial robot control system, designed at Lund University, has been used to demonstrate control using external sensors in several different applications. Questions related to geometry and force control task specification have been discussed. The dynamical properties of the system with its new in-

terface have been modeled and verified. Industrial applications of force control, such as grinding, deburring and drilling, have all been demonstrated. Additionally, the system has been used successfully as an experimental platform for high-speed visual servoing and force/vision control.

Robotic force control can be carried out on two levels, either with forces controlled by a separate external tool, or by integrating the force control with the control of the robot motion. The advantages of the robot with integrated force control include the possibility of obtaining higher stiffness at a considerably lower weight, as well as increased flexibility in mounting and accessibility, due to the six degrees of freedom available. As an example, as illustrated in Chapter 6, functionality for high-bandwidth force control in modern industrial robot control systems could potentially lead to usable robotic drilling systems, without the use of expensive special-purpose drilling end-effectors or calibration procedures.

Feature-Based Methods for Visual Tracking and Control

Most visual tracking algorithms use a separate feature extraction step, where the image data is compressed into a smaller number of image features, from which the motion is estimated. Methods such as [Drummond and Cipolla, 2002; Martin and Horaud, 2002] have been extended and reformulated to the case of dynamic tracking. For tasks where task-space specifications exist, methods using position-based hybrid force/vision control algorithms have been developed, using the motion estimator together with inner-motion impedance control to obtain compliant behavior in the force-controlled directions. For tasks defined directly in image space, an image-based technique for vision/force control has been presented, and used to perform drawing on a surface. Issues concerning robustness and real-time performance of the feature-based trackers have also been considered. Motion estimation for uncalibrated hand/eye camera systems has been demonstrated, where a dual quaternion representation of the pose is used to obtain linear constraints on the estimated parameters, thereby reducing the dimensionality of the problem. Further, a novel method for online minimization of the measurement error covariance for a multi-camera setup has been presented. The suggested strategy was compared to heuristic algorithms, and evaluated in simulations capturing the real-time properties and effects on a vision-based control system.

Feature-based methods in general suffer from problems relating to the feature matching step. If the matching step is always possible to perform reliably, the solution to the pose/motion estimation problem is generally more robust to illumination variations, occlusions and other disturbances and imperfections in the modeling. Satisfying the demands for reliable matching in a real-time context may be very challenging in the general case. In practice, it is necessary to integrate the tracking and match-

ing steps, such that information from the tracking is propagated to the matching algorithm, and vice versa. Examples of this integration are the tracking windows commonly used for the matching of features in image sequences, and the possibility for the estimation algorithm to modify the relative weight of different features depending on matching confidence.

Intensity-Based Methods for Visual Tracking and Control

Intensity-based approaches for tracking and vision based position control have also been presented. Dynamical visual tracking based directly on image intensity measurements results in very high processing rate, good accuracy and short input-output latency. This has been illustrated in several tracking and visual servoing experiments, using sampling rates of up to 250 Hz. In the experiments, the tracking frame rates were limited only by the capability of the camera and the IEEE-1394 bus, as the tracking algorithm itself consumed less than 10% of the CPU time on a 2.4 GHz computer. By focusing on problems related to stability and disturbance sensitivity in feedback systems, it was shown to be possible to relax previous approximation-oriented solutions, where an optimal least-squares fit resulting in high-gain feedback was required. In situations where this sub-optimal structure with pose estimation and feedback can be avoided, it is possible to improve performance several orders of magnitude in many problems. This was illustrated by simulations of visual servoing and control of simple mechanical systems. The effect of disturbances can be suppressed by modifying the state estimator to take the disturbance characteristics into account in the linearization.

The intensity-based methods are not limited only to simple tracking problems with few degrees of freedom, but are fully capable of tracking rigid motions in well-structured environments, in particular when using multi-scale techniques. Although illumination variations can be partially compensated for, the inherent sensitivity to the illumination conditions make robust implementation in natural environments a challenging topic for future research. Hybrid methods, in which the intensity-based methods are complemented by standard feature measurements for drift compensation, represent one possible extension to the basic methods.

A

Vision System Modeling and Calibration Techniques

In this appendix, some details on camera modeling are covered, as well as the details of the calibration techniques used. An introduction is given to the different coordinate systems, representations and notations which are used throughout this thesis. Some details on the hardware and software used for the experiments and simulations of real-time vision are also presented.

A.1 Frames, Poses and Notation

Models of robotic systems, and robotic vision systems in general, contain a large number of different coordinate systems. Coordinate systems are regularly attached to the robot base, wrist/flange and the tool held by the robot (at the so called Tool Center Point, TCP), to each camera, and to each object in the workcell/environment. Many methods, for instance in kinematics and visual pose estimation, introduce new, intermediate coordinate system on each robot link and on the image plane of each camera. The pose of each frame[1] (coordinate system) is described relative to another frame, often as an orthogonal rotation matrix $\mathbf{R}$ with $\det(\mathbf{R}) = 1$ and a translation vector $\mathbf{t}$. Rotation/translation $\mathbf{R}_{ab}$ and $\mathbf{t}_{ab}$ describe the pose of frame **a** relative to another frame **b**. Given a point in space with coordinates $\mathbf{X}_a$ and $\mathbf{X}_b$ in frame **a** and **b**, respectively, we can use

$$\mathbf{X}_a = \mathbf{R}_{ab}\mathbf{X}_b + \mathbf{t}_{ab} \tag{A.1}$$

[1]When in this thesis we talk about the pose (position/orientation) of an object, what is actually referred to is always the pose of its coordinate system relative to some other coordinate system.

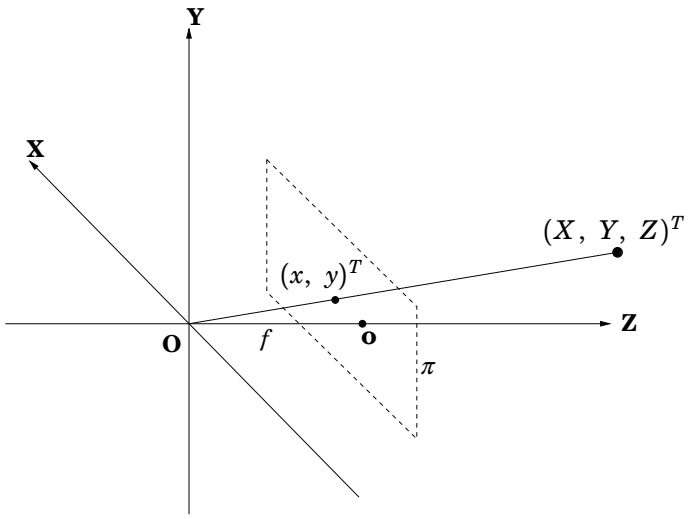

Figure A.1 The pinhole camera model, showing the coordinate system and optical Z-axis, the image plane π, focal length f, and principal point **o**.

to transfer the point coordinates from **b** to **a**. Geometrically, the columns of $\mathbf{R}_{ab}$ and the vector $\mathbf{t}_{ab}$ represent the three basis vectors and the origin of **b**, all expressed in the coordinates of frame **a**. Expression (A.1) is often written using *homogeneous coordinates* as

$$\begin{bmatrix} \mathbf{X}_a \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{ab} & \mathbf{t}_{ab} \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X}_b \\ 1 \end{bmatrix} \stackrel{\text{def.}}{=} \mathbf{T}_{ab} \begin{bmatrix} \mathbf{X}_b \\ 1 \end{bmatrix} \tag{A.2}$$

using the 4×4-matrix $\mathbf{T}_{ab}$. This matrix representation of the pose is highly over-parametrized, since 12 parameters are used to describe the six degrees of freedom of a rigid transformation. For purposes of calibration and motion estimation, in which the rigid transformation is calculated by solving a parameter optimization problem, representations using a lower number of parameters are required. Euler angles, angle/axis or quaternion parametrizations are the most frequently used pose representations for such purposes.

A.2 Camera Modeling

The most common camera model is the *pinhole* or *perspective camera*. For more information about other camera models, such as the *weak perspective* and *orthographic* approximations, see for instance [Trucco and Verri,

1998]. The perspective camera model consists of a point **O**, the center of projection, and a plane π, the image plane. The origin of the camera-centered coordinate system is located at **O**, see Fig. A.1. The distance between **O** and π is called the *focal length*, f. The line perpendicular to π that passes through **O** is the *optical axis*, and the intersection of this line with π is the origin **o** of the image coordinate system, the *principal point*. The projection equations for a point $(X\ Y\ Z)^T$ in Cartesian space in the perspective camera are given by

$$x = f\frac{X}{Z} \tag{A.3}$$

$$y = f\frac{Y}{Z}. \tag{A.4}$$

This can be written using homogeneous coordinates as

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{A.5}$$

where $\lambda = Z$ is the depth of the imaged point in the camera. To further model the internal geometric properties of the camera optics and sensor, it is convenient to introduce a number of intrinsic camera parameters. These parameters allow us to describe non-quadratic pixels (aspect ratio$\neq$1), skewed sensor pixel arrays, and translation of the principal point from the origin in the pixel grid. A camera model including these effects is given by

$$\begin{aligned} \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \begin{bmatrix} f & s & u_0 & 0 \\ 0 & \gamma f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \\ &= \underbrace{\begin{bmatrix} f & s & u_0 \\ 0 & \gamma f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{R}_{3\times 4}} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{\mathbf{X}}. \end{aligned} \tag{A.6}$$

The matrix **K** is the intrinsic camera matrix, and $\mathbf{R}_{3\times 4}$ is the extrinsic camera matrix. The intrinsic parameters f and γ describe the focal length

and aspect ratio, s is the skew (usually close to 0 in modern cameras), and $(u_0 \; v_0)^T$ is the position of the principal point in the image. The matrix $\mathbf{R}_{3\times4}$ can be used to change coordinate system in the world, usually to a coordinate system attached to some object in the scene

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \mathbf{X}^o \tag{A.7}$$

where $\mathbf{R}$ and $\mathbf{t}$ are the orthogonal rotation matrix and the vector describing the pose of the object with respect to the camera. $\mathbf{X}^o$ describes the object model in a local object coordinate system. In addition to these parameters, it is often necessary to model distortions in the camera and lens system. A commonly used technique is to use polynomial models for radial and tangential distortion [Zhang, 1999], in the form

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} p_r(r^2) \begin{bmatrix} x \\ y \end{bmatrix} + \mathbf{p}_t(r^2, x, y) \\ 1 \end{bmatrix} \tag{A.8}$$

with $x = X/Z$ and $y = Y/Z$ from Eqs. (A.3)–(A.4) (assuming $f = 1$) and $r^2 = x^2 + y^2$, and $\mathbf{K}$ the intrinsic camera matrix in Eq. (A.6). The radial distortion function $p_r(r^2)$ is usually a polynomial with $p_r(0) = 1$. In practice, using high-order polynomials to model radial distortion is only necessary when using wide-angle lenses. The tangential distortion vector $\mathbf{p}_t(r^2, x, y)$, modeling imperfect centering of the lens system by low-order polynomials, can often be neglected for modern lens systems.

The general camera model including distortion contains a large number of parameters, which makes reliable calibration difficult when only a few calibration images are available. Therefore, in most calibrations performed within this work only a first order symmetric radial distortion model was used. Another advantage is that this distortion model is easy to invert, which makes real-time distortion correction possible to implement without any significant decrease in performance. Distortion compensation was performed directly on the image data, thereby making it possible to use the distortion-free camera models for modeling of image projections. Additionally, the image skew parameter s was always assumed to be equal to zero, as is customary in modern calibration software. Another simplification, which was sometimes used, was to reject the principal point from the optimization. The reason is that the principal point is often very difficult to estimate accurately, and that good approximations can often be obtained by assuming that it is located at the center of the pixel grid.

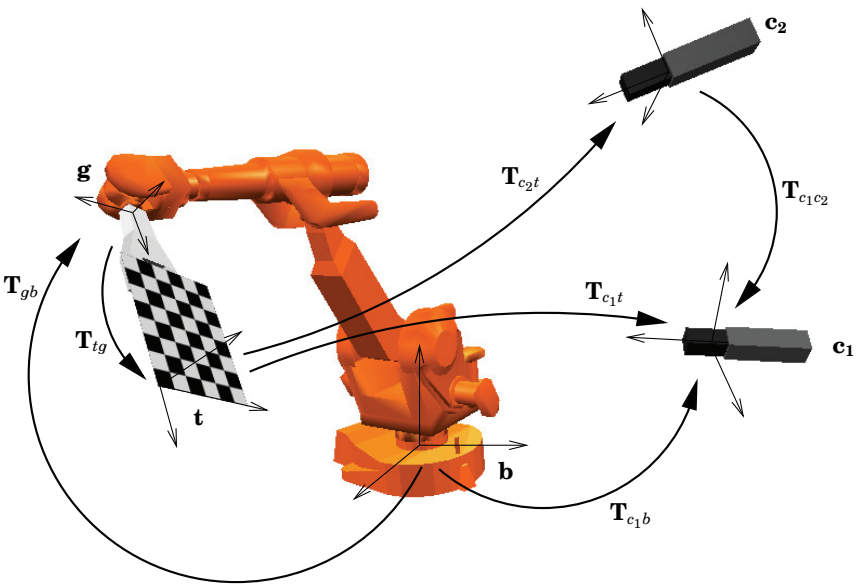

Figure A.2 The most important frames and transformations. Shown are the frames attached to the tool/gripper **g**, the calibration target **t**, cameras $\mathbf{c_1}$ and $\mathbf{c_2}$, and the robot base **b**.

A.3 Multi-Camera Calibration Algorithm

When a multi-camera system is used, both the intrinsic camera parameters and the relative positions of the sensors need to be determined. When the cameras are fixed in the workspace, as depicted in Fig. A.2 for the two-camera case, the calibration object can be attached to the robot end-effector. The measured joint positions and the kinematics of the robot provide accurate information on the (relative) movement of the end-effector between the images in the sequence. Therefore, the extrinsic parameters are partially known, and this information was used in the algorithm below in order to decrease the number of parameters that need to be estimated. In order to find all the geometric parameters illustrated in Fig. A.2, a calibration procedure in three steps was used:

1. Individual estimation of intrinsic and extrinsic camera parameters for each camera, using standard calibration algorithms without use of the robot position measurements.
2. Analytical computation of $\mathbf{T}_{tg}$ from calibration data and robot kinematics.

3. Nonlinear least-squares optimization of all parameters simultaneously.

The calibration procedure used in Step 1 was based on the method of [Zhang, 1999], although the final optimization step was not required in this step of the algorithm. The problem in Step 2 is mathematically equivalent to a standard hand-eye calibration problem. This is seen if we note that for two different end-effector positions, denoted by numbers p and q, and for each camera c_k, it holds that

$$\mathbf{T}_{c_k t}(q)^{-1}\,\mathbf{T}_{c_k t}(p)\,\mathbf{T}_{tg} = \mathbf{T}_{tg}\,\mathbf{T}_{gb}(q)\,\mathbf{T}_{gb}(p)^{-1} \tag{A.9}$$

which can be solved for the unknown constant $\mathbf{T}_{tg}$. Many different solutions to the hand-eye calibration problem exist, in this work the classical method of [Tsai and Lenz, 1989] was used. The final optimization in Step 3 was used to minimize, for all cameras, the errors between the m measured and reprojected image points in each of the n images. For a total number of cameras q, this error becomes

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{q}\left(\mathbf{y}_{ijk} - \hat{\mathbf{y}}_{ijk}(\mathbf{K}_1, \cdots, \mathbf{K}_q, \mathbf{T}_{tg}, \mathbf{T}_{c_1c_2}, \mathbf{T}_{c_1b})\right)^2 \tag{A.10}$$

where the reprojected image coordinates $\hat{\mathbf{y}}_{ijk}$ was given by the projection equations

$$\begin{cases} \lambda_{ijk}\hat{\mathbf{y}}_{ijk} = \mathbf{K}_k\mathbf{T}_{c_1b}\mathbf{T}_{gb}(i)^{-1}(\mathbf{T}_{tg})^{-1}\mathbf{X}_j, & k = 1 \\ \lambda_{ijk}\hat{\mathbf{y}}_{ijk} = \mathbf{K}_k(\mathbf{T}_{c_1c_k})^{-1}\mathbf{T}_{c_1b}\mathbf{T}_{gb}(i)^{-1}(\mathbf{T}_{tg})^{-1}\mathbf{X}_j, & k \geq 2 \end{cases} \tag{A.11}$$

where $\mathbf{X}_j$ were the model points of the calibration object in its local coordinate system, and $\mathbf{K}_k$ were the matrices of intrinsic camera parameters. The minimization was performed with respect to the parameters of $\mathbf{K}_k$, $\mathbf{T}_{c_1b}$, $\mathbf{T}_{c_1c_2}$ and $\mathbf{T}_{tg}$.

A.4 Simulated Real-Time Vision

In order to facilitate real-time simulations of robot vision algorithms, a simulation tool using the graphics API OpenGL was developed and used extensively in the simulations. In the program, the geometry of a scene can be specified and controlled from Matlab/Simulink, with object geometries specified either directly in Matlab or in XML files. The communication is handled through TCP/IP sockets and a shared memory interface for fast

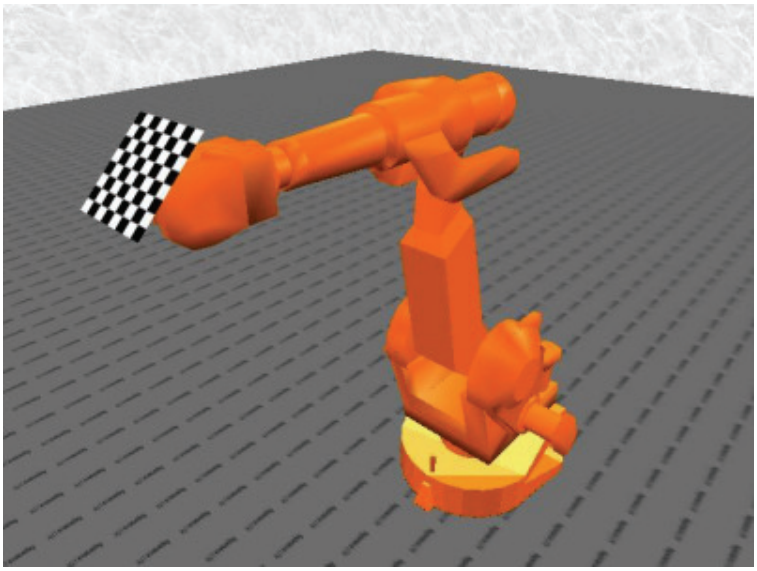

Figure A.3 Image from calibration experiment using the virtual camera/robot simulation tool.

readout of images. Using mid-end graphics hardware, simulated visual servoing at frame rates of several hundred Hz can be performed in real-time. The simulation tool makes it possible to perform rapid prototyping and testing of vision algorithms, using practically important effects and disturbance sources such as surfaces with irregular textures, background clutter, reflections, and varying illumination.

In order to use the tool for evaluation of estimation accuracy, the exact geometrical and image transformations of the graphics engine must be known exactly. A camera calibration was performed, using a distortion-free virtual camera with a specified camera matrix

$$\mathbf{K} = \begin{bmatrix} f & s & u_0 \\ 0 & \gamma f & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 500 & 0 & 320 \\ 0 & 500 & 240 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.12}$$

and a planar quadratic calibration pattern as shown in Fig. A.3. Using 6 images and the calibration algorithm in A.3 for a single camera with four intrinsic parameters (focal length, aspect ratio, and principal point), the

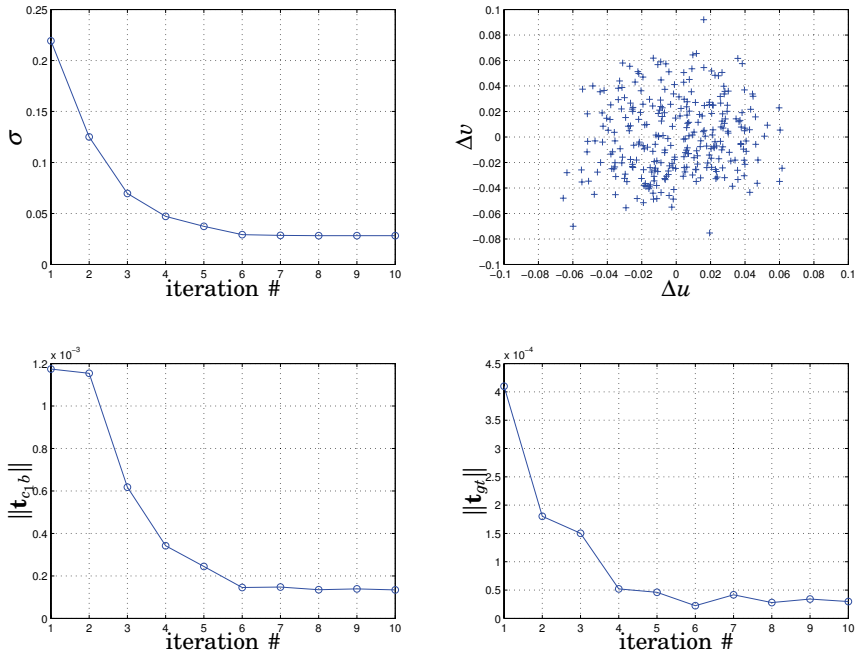

Figure A.4 Results from calibration. *Top left:* standard deviation of the reprojected image error after each iteration in the final Levenberg-Marquardt optimization step. *Top right:* the reprojected image errors at the optimum. *Bottom left:* the norm of the translation error in $\mathbf{T}_{c_1 b}$ after each iteration. *Bottom right:* the norm of the translation error in $\mathbf{T}_{gt}$ after each iteration.

estimated camera matrix was

$$\hat{\mathbf{K}} = \begin{bmatrix} 499.93 & 0.0 & 320.34 \\ 0 & 499.92 & 240.60 \\ 0 & 0 & 1.0000 \end{bmatrix}, \tag{A.13}$$

indicating that the projection equations of the simulation correspond well to the specified parameters. The results are illustrated in Fig. A.4, where the convergence of the errors in the reprojected image points and parameters are shown, as well as the reprojected 2D-errors in the images.

A.5 Experimental Vision System

The communication with IEEE-1394 cameras is in general handled using one of a number of available APIs, such as the Linux version of the dc1394

API [DC1394, 2006] used in this thesis. The APIs make it possible to control the setup by accessing the camera control registers over the IEEE-1394 bus, and to obtain image data from the camera. For IEEE-1394 cameras, exposure start and image readout can be handled automatically by the camera. The frame rate and exposure time are then set directly by writing suitable values to the camera control registers using the API, and the camera is started in isochronous (i.e. equidistant sampling) image capture mode. The camera will then start to capture and transfer images at equidistant capture times. Many cameras also include the possibility to control exposure start and stop by the use of an external trigger signal, providing the ability to synchronize the exposure of several cameras.

The vision system used in the experiments in this thesis was based on a number of Basler A602fc and Sony DFW-V300 digital cameras, connected to a Linux platform via standard IEEE-1394 connections. The Sony cameras have support for standard video modes with VGA resolution and frame rates of up to 60 images/second. The Basler cameras are CMOS cameras supporting global shuttering, making them suitable for high-speed vision applications. The Basler cameras additionally support non-standard region-of-interest based video modes, where the frame rates are limited only by shutter speed, image read-out and transfer delays. Full VGA resolution single-channel images can be obtained at 100 images/second, while at the external robot controller sampling frequency of 250 Hz, the maximum resolution is 512×256 pixels. Some low-level image processing and graphics operations were performed using the OpenCV library [Bradski, 2000]. Most non-real-time algorithms were implemented in Matlab, while real-time code such as visual trackers were implemented entirely in C/C++.

Appendix A. Vision System Modeling and Calibration Techniques

References

Ait-Aider, O., N. Andreff, P. Martinet, and J.-M. Lavest (2006): "Simultaneous pose and velocity measurement by vision for high-speed robots." In *Proc. Int. Conf. Robotics and Automation*, pp. 3742–3747.

Alici, G. (1999): "A systematic approach to develop a force control system for robotic drilling." *Industrial Robot: An International Journal*, **26:5**, pp. 389–397.

Allen, P., B. Yoshimi, and A. Timcenko (1991): "Real-time visual servoing." In *IEEE Int. Conf. Robotics and Automation*, pp. 851–856. Sacramento, CA.

Andreff, N., R. Horaud, and B. Espiau (2001): "Robot hand-eye calibration using structure from motion." *Int. J. Robotics Research*, **20:3**, pp. 228–248.

Arulampalam, M., S. Maskell, N. Gordon, and T. Clapp (2002): "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking." *IEEE Trans. Signal Processing*, **50:2**, pp. 174–188.

Åström, K. J. and B. Wittenmark (1997): *Computer-Controlled Systems*, 3rd edition. Prentice Hall, Upper Saddle River, NJ.

Baeten, J., H. Bruyninckx, and J. De Schutter (1999): "Combining eye-in-hand visual servoing and force control in robotic tasks using the task frame." In *Proc. IEEE Int. Conf. Multisensor Fusion*, pp. 141–146. Taipei, Taiwan.

Besl, P. and N. McKay (1992): "A method for registration of 3-d shapes." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **14:2**, pp. 239–256.

Blake, A. (2006): "Visual tracking: a very short research roadmap." *Electronics Letters*, **42:5**, pp. 254–256.

Blomdell, A., G. Bolmsjö, T. Brogårdh, P. Cederberg, M. Isaksson, R. Johansson, M. Haage, K. Nilsson, M. Olsson, T. Olsson, A. Robertsson, and J. Wang (2005): "Extending an industrial robot controller–Implementation and applications of a fast open sensor interface." *IEEE Robotics and Automation Magazine*, **12:3**, pp. 85–94.

Born, H. and J. Bunsendal (2001): "Programmable multi sensor interface for industrial applications." In *Proc. Int. Conf. Multisensor Fusion and Integration for Intelligent Systems*, pp. 189–194. Baden-Baden, Germany.

Boyd, S. and L. Vandenberghe (2004): *Convex Optimization*. Cambridge University Press.

Bradski, G. (2000): "The OpenCV library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, **25:11**, pp. 120–124.

Brogliato, B. and P. Orhant (1994): "On the transition phase in robotics: impact models, dynamics and control." In *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, pp. 346–351. San Diego, CA.

Caccamo, M., G. Buttazzo, and L. Sha (2000): "Elastic feedback control." In *Proc. Euromicro Conference on Real-Time Systems*, pp. 121–128. Stockholm, Sweden.

Canny, J. (1986): "A computational approach to edge detection." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **8:6**, pp. 679–698.

Canudas de Wit, C., H. Olsson, K. Åström, and P. Lischinsky (1995): "A new model for control of systems with friction." *IEEE Trans. Automatic Control*, **40:3**, pp. 419–425.

Carceroni, R. L. and C. M. Brown (1998): "Numerical methods for model-based pose recovery." Technical Report 659. Comp. Sci. Dept., Univ. of Rochester, Rochester, N.Y.

Carelli, R., E. Oliva, and C. Soria (2004): "Combined force and visual control of an industrial robot." *Robotica*, **22:2**, pp. 163–172.

Chang, W.-C. (2004): "Cartesian-based planar contour following with automatic hybrid force and visual feedback." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, vol. 3, pp. 3062–3067. Sendai, Japan.

Chaumette, F. and E. Malis (2000): "2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings." In *Proc. Int. Conf. Robotics and Automation*, vol. 1, pp. 630–635. San Francisco, CA.

Chen, H. (1991): "A screw motion approach to uniqueness analysis of head-eye geometry." In *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 145–151. Maui, HI.

Chen, W.-G., N. Nandhakumar, and W. Martin (1996): "Image motion estimation from motion smear - a new computational model." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **18:4**, pp. 412–425.

Chiaverini, S., B. Siciliano, and L. Villani (1999): "A survey of robot interaction control schemes with experimental comparison." *IEEE/ASME Trans. Mechatronics*, **4:3**, pp. 273–285.

Clifford, W. (1873): "Preliminary sketch of bi-quaternions." *Proc. London Math. Soc.*, **4**, pp. 381–395.

Colgate, E. and N. Hogan (1988): "Robust control of dynamically interacting systems." *Int. J. of Control*, **48:1**, pp. 65–88.

Colgate, E. and N. Hogan (1989): "An analysis of contact instability in terms of passive physical equivalents." In *Proc. Int. Conf. Robotics and Automation*, vol. 1, pp. 404–409. Scottsdale, AZ.

Comaniciu, D., V. Ramesh, and P. Meer (2000): "Real-time tracking of non-rigid objects using mean shift." *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, **2**, pp. 142–149.

Comport, A., D. Kragic, E. Marchand, and F. Chaumette (2005): "Robust real-time visual tracking: Comparison, theoretical analysis and performance evaluation." In *Proc. Int. Conf. Robotics and Automation*, pp. 2841–2846.

Cootes, T., G. Edwards, and C. Taylor (2001): "Active appearance models." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **23:6**, pp. 681–685.

Corke, P. and M. Good (1992): "Dynamic effects in high-performance visual servoing." In *Proc. Int. Conf. Robotics and Automation*, pp. 1838–1843. Nice, France.

Corke, P. and S. Hutchinson (2001): "A new partitioned approach to image-based visual servo control." *IEEE Trans. Robotics and Automation*, **17:4**, pp. 507–516.

Craig, J. J. (1989): *Introduction to Robotics: Mechanics and Control*. Addison-Wesley.

Cuvillon, L., E. Laroche, J. Gangloff, and M. de Mathelin (2005): "GPC versus H_∞ control for fast visual servoing of a medical manipulator including flexibilities." In *Proc. Int. Conf. Robotics and Automation*, pp. 4044–4049. Barcelona, Spain.

Daniilidis, K. (1999): "Hand-eye calibration using dual quaternions." *Int. J. Robotics Research*, **18:3**, pp. 286–298.

Davison, A. (2005): "Active search for real-time vision." In *IEEE Int. Conf. Computer Vision*, vol. 1, pp. 66–73. Beijing, China.

DC1394 (2006): `http://sourceforge.net/projects/libdc1394`.

De Schutter, J., H. Bruyninckx, W.-H. Zhu, and M. W. Spong (1997): "Force control: A bird's eye view." In *IEEE CSS/RAS Int. Workshop on Control Problems in Robotics and Automation: Future Directions*, pp. 1–17. San Diego, CA.

De Schutter, J., D. Torfs, and H. Bruyninckx (1996): "Robot force control with an actively damped flexible end effector." *Robotics and Autonomous Systems*, **19:2**, pp. 205–215.

De Schutter, J. and H. Van Brussel (1988a): "Compliant robot motion I. A formalism for specifying compliant motion tasks." *Int. J. Robotics Research*, **7:4**, pp. 3–17.

De Schutter, J. and H. Van Brussel (1988b): "Compliant robot motion II. A control approach based on external control loops." *Int. J. Robotics Research*, **7:4**, pp. 18–33.

de Wit, C. C., H. Olsson, K. J. Åström, and P. Lischinsky (1995): "A new model for control of systems with friction." *IEEE Transactions on Automatic Control*, **40:3**.

Dean-Leon, E., L. Garcia-Valdovinos, V. Parra-Vega, and A. Espinosa-Romero (2005): "Uncalibrated image-based position-force adaptive visual servoing for constrained robots under dynamic friction uncertainties." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 2013–2020. Edmonton, Canada.

Deguchi, K. (1998): "Optimal motion control for image-based visual servoing by decoupling translation and rotation." *Proc. IEEE/RSJ Int. Conf.Intelligent Robots and Systems*, **2**, pp. 705–711.

Deguchi, K. and T. Noguchi (1996): "Visual servoing using eigenspace method and dynamic calculation of interaction matrices." In *Proc. Int. Conf. Pattern Recognition*, vol. 1, pp. 302–306. Vienna, Austria.

Dementhon, D. and L. Davis (1995): "Model-based object pose in 25 lines of code." *Int. J. Computer Vision*, **15:1**, pp. 123–141.

Deng, L., F. Janabi-Sharifi, and W. Wilson (2002): "Stability and robustness of visual servoing methods." In *Proc. Int. Conf. Robotics and Automation*, vol. 2, pp. 1604–1609. Washington DC, USA.

Dickmanns, E. (1988): "An integrated approach to feature based dynamic vision." In *Proc. Comp. Soc. Conf. Computer Vision and Pattern Recognition*, pp. 820–825. Ann Arbor, MI.

Diolaiti, N., C. Melchiorri, and S. Stramigioli (2005): "Contact impedance estimation for robotic systems." *IEEE Trans. Robotics*, **21:5**, pp. 925–936.

Dohring, M. and W. Newman (2003): "The passivity of natural admittance control implementations." In *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 3, pp. 3710–3715. Taipei, Taiwan.

Drummond, T. and R. Cipolla (2002): "Real-time tracking of complex structures with on-line camera calibration." *Image and Vision Computing*, **20:5-6**, pp. 427–434.

Durrant-Whyte, H. (1988): "Sensor models and multisensor integration." *Int. J. Rob. Res.*, **7:6**, pp. 97–113.

Elmenreich, W. (2001): "An introduction to sensor fusion." Research Report 47/2001. Techn. Univ. Wien, Inst. Techn. Informatik.

Eppinger, S. and W. Seering (1992): "Three dynamic problems in robot force control." *IEEE Trans. Robotics and Automation*, **8:6**, pp. 751–758.

Espiau, B., F. Chaumette, and P. Rives (1992): "A new approach to visual servoing in robotics." *IEEE Trans. Robotics and Automation*, **8:3**, pp. 313–326.

Featherstone, R., S. Thiebaut, and O. Khatib (1999): "A general contact model for dynamically-decoupled force/motion control." In *IEEE Int. Conf. Robotics and Automation*, vol. 4, pp. 3281–3286. Detroit, MI.

Ferreira, A., C. Cassier, and S. Hirai (2004): "Automatic microassembly system assisted by vision servoing and virtual reality." *IEEE/ASME Trans. Mechatronics*, **9:2**, pp. 321–333.

Ferretti, G., G. Magnani, and P. Rocco (2004): "Impedance control for elastic joints industrial manipulators." *IEEE Trans. Robotics and Automation*, **20:3**, pp. 488–499.

FireWire (2006): `http://www.1394ta.org`.

Fischler, M. and R. Bolles (1981): "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography." *Communications of the ACM*, pp. 381–395.

Freund, E. and J. Pesara (1998): "High-bandwidth force and impedance control for industrial robots." *Robotica*, **16:1**, pp. 75–87.

Gangloff, J. and M. de Mathelin (2000): "High speed visual servoing of a 6 DoF manipulator using MIMO predictive control." In *Proc. Int. Conf. Robotics and Automation*, vol. 4, pp. 3751–3756. San Francisco, CA.

Gelb, A., Ed. (1974): *Applied Optimal Control*. The M.I.T. Press, Cambridge, Massachusetts.

Gleicher, M. (1997): "Projective registration with difference decomposition." In *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 331–337. San Juan, Puerto Rico.

Goddard, J. (1997): *Pose and motion estimation from vision using dual quaternion-based extended Kalman filtering*. PhD thesis, University of Tennessee, Knoxville, TN.

Goldsmith, P., B. Francis, and A. Goldenberg (1999): "Stability of hybrid position/force control applied to manipulators with flexible joints." *Int. J. of Robotics and Automation*, **14:4**, pp. 146–160.

Gonzalez, R. and R. Woods (1992): *Digital Image Processing*. Addison-Wesley, Boston, MA.

Gordon, N., D. Salmond, and A. Smith (1993): "Novel approach to nonlinear/non-Gaussian Bayesian state estimation." *IEE Proc. F Radar and Signal Processing*, **140:2**, pp. 107–113.

Hager, G. and P. Belhumeur (1998): "Efficient region tracking with parametric models of geometry and illumination." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **20:10**, pp. 1025–1039.

Hamilton, W. R. (1853): *Lectures on Quaternions*. Hodges Smith & Co., Dublin.

Hannaford, B. and R. Anderson (1988): "Experimental and simulation studies of hard contact in force reflecting teleoperation." In *IEEE Int. Conf. Robotics and Automation*, pp. 584–589.

Harris, C. and M. Stephens (1988): "A combined edge and corner detector." In *Proc. 4th Alvey Vision Conf.*, pp. 147–151.

Hashimoto, K., A. Namiki, and M. Ishikawa (2001): "A visuomotor control architecture for high-speed grasping." In *Proc. IEEE Conf. Decision and Control*, vol. 1, pp. 15–20. Orlando, FL.

Henriksson, D. (2006): *Resource-Constrained Embedded Control and Computing Systems*. PhD thesis ISRN LUTFD2/TFRT--1074--SE, Department of Automatic Control, Lund University, Sweden.

Henriksson, D., A. Cervin, and K.-E. Årzén (2002): "TrueTime: Simulation of control loops under shared computer resources." In *Proceedings of the 15th IFAC World Congress on Automatic Control*. Barcelona, Spain.

Henriksson, D. and T. Olsson (2004): "Maximizing the use of computational resources in multi-camera feedback control." In *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS04)*. Toronto, Canada.

Hill, A., T. Cootes, and C. Taylor (1996): "Active shape models and the shape approximation problem." *Image and Vision Computing*, **14:8**, pp. 601–607.

Hogan, N. (1985): "Impedance control: An approach to manipulations parts I, II, III." *ASME Journal of Dynamic Systems, Measurement, and Control*, **107:1**.

Horaud, R. and F. Dornaika (1995): "Hand-eye calibration." *Int. J. of Robotics Research*, **14:3**, pp. 195–210.

Horaud, R., F. Dornaika, and B. Lamiroy (1997): "Object pose: The link between weak perspective, paraperspective, and full perspective." *Int. J. Computer Vision*, **22:2**, pp. 173–189.

Hosoda, K., K. Igarashi, and M. Asada (1996): "Adaptive hybrid visual servoing/force control in unknown environment." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, vol. 3, pp. 1097–1103. Osaka, Japan.

Hutchinson, S., G. Hager, and P. Corke (1996): "A tutorial on visual servo control." *IEEE Trans. Robotics and Automation*, **12:5**, pp. 651–670.

Isard, M. and A. Blake (1998): "CONDENSATION-conditional density propagation for visual tracking." *Int. J. of Computer Vision*, **29:1**, pp. 5–28.

Johansson, R., A. Robertsson, K. Nilsson, T. Brogårdh, P. Cederberg, M. Olsson, T. Olsson, and G. Bolmsjö (2004): "Sensor integration in task-level programming and industrial robotic task execution control." *Industrial Robot: An International Journal*, **31:3**, pp. 284–296.

Jörg, S., J. Langwald, J. Stelter, G. Hirzinger, and C. Natale (2000): "Flexible robot-assembly using a multi-sensory approach." In *IEEE Int. Conf. Robotics and Automation*, pp. 3687–3694. San Francisco, CA.

Julier, S. and J. Uhlmann (2004): "Unscented filtering and nonlinear estimation." *Proceedings of the IEEE*, **92:3**, pp. 401–422.

Jurie, F. and M. Dhome (2001): "Real time 3D template matching." In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 1791–1799. Kauai, HI.

Kawaji, S., M. Arao, and Y. Chen (2001): "Thrust force control of drilling system using neural network." In *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, vol. 1, pp. 476–481. Como, Italy.

Kelly, R. (1996): "Robust asymptotically stable visual servoing of planar robots." *IEEE Trans. Robotics and Automation*, **12:5**, pp. 759–767.

Kelly, R., R. Ortega, A. Ailon, and A. Loria (1994): "Global regulation of flexible joint robots using approximate differentiation." *IEEE Trans. Robotics and Automation*, **39:6**, pp. 1222–1224.

Khalil, H. K. (1996): *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ.

Khatib, O. (1987): "A unified approach for motion and force control of robot manipulators: The operational space formulation." *IEEE J. of Robotics and Automation*, **3:1**.

Krstić, M., I. Kanellakopoulos, and P. V. Kokotović (1995): *Nonlinear and Adaptive Control Design*. Wiley, New York, NY.

Krupa, A., G. Morel, and M. De Mathelin (2004): "Achieving high-precision laparoscopic manipulation through adaptive force control." *Advanced Robotics*, **18:9**, pp. 905–927.

Kyrki, V. and D. Kragic (2006): "Tracking unobservable rotations by cue integration." In *Proc. Int. Conf. Robotics and Automation*, pp. 2744–2750.

Lee, W. and C. Shih (2006): "Control and breakthrough detection of a three-axis robotic bone drilling system." *Mechatronics*, **16:2**, pp. 73–84.

Lefebvre, T., H. Bruyninckx, and J. De Schutter (2002): "Comment on "A new method for the nonlinear transformation of means and covariances in filters and estimators" [with authors' reply]." *IEEE Trans. Automatic Control*, **47:8**, pp. 1406 – 1409.

Leite, A., F. Lizarralde, and L. Hsu (2006): "Hybrid vision-force robot control for tasks on unknown smooth surfaces." In *IEEE Int. Conf. Robotics and Automation*, pp. 2244–2249.

Li, P., T. Zhang, and A. Pece (2003): "Visual contour tracking based on particle filters." *Image and Vision Computing*, **21:1**, pp. 111–123.

Lippiello, V., B. Siciliano, and L. Villani (2002): "Objects motion estimation via BSP tree modeling and Kalman filtering of stereo images." In *IEEE Int. Conf. Robotics and Automation*, pp. 2968–2973. Washington D.C.

Lippiello, V., B. Siciliano, and L. Villani (2006): "Robot interaction control using force and vision." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1470–1475. Beijing, China.

Lowe, D. (1991): "Fitting parameterized three-dimensional models to images." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **13:5**, pp. 441–450.

Lowe, D. (1999): "Object recognition from local scale-invariant features." *Proc. 7th IEEE Int. Conf. Computer Vision*, **2**, pp. 1150–1157.

Lucas, B. and T. Kanade (1981): "An iterative image registration technique with an application to stereo vision." In *Int. Joint Conf. Artificial Intelligence*, pp. 674–679.

Ma, Y., S. Soatto, J. Kosecka, and S. Sastry (2003): *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer-Verlag.

Malis, E. and S. Benhimane (2006): "Integration of euclidean constraints in template based visual tracking of piecewise-planar scene." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1218–1223. Beijing, China.

Malis, E. and E. Marchand (2006): "Experiments with robust estimation techniques in real-time robot vision." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 223–228. Beijing, China.

Malis, E., G. Morel, and F. Chaumette (2001): "Robot control using disparate multiple sensors." *Int. J. Robotics Research*, **20:5**, pp. 364–377.

Malis, E. and P. Rives (2003): "Robustness of image-based visual servoing with respect to depth distribution errors." In *IEEE Int. Conf. Robotics and Automation*, pp. 1056–1061. Taipei, Taiwan.

Martin, F. and R. Horaud (2002): "Multiple camera tracking of rigid objects." *Int. J. of Robotics Research*, **21:2**, pp. 97–113.

Martinet, P. and E. Cervera (2001): "Stacking jacobians properly in stereo visual servoing." In *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 717–722. Seoul, Korea.

Mason, M. T. (1981): "Compliance and force control for computer controlled manipulators." *IEEE Trans. on Systems, Man, and Cybernetics*, **No 11**, pp. 418–432.

Morel, G., E. Malis, and S. Boudet (1998): "Impedance based combination of visual and force control." In *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 1743–1748. Leuven, Belgium.

Murray, R., Z. Li, and S. Sastry (1994): *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton, FL.

Nakabo, Y., M. Ishikawa, H. Toyoda, and S. Mizuno (2000): "1 ms column parallel vision system and its application of high speed target tracking." In *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 650–655.

Natale, C., R. Koeppe, and G. Hirzinger (2000): "A systematic design procedure of force controllers for industrial robots." *IEEE/ASME Trans. Mechatronics*, **5:2**, pp. 122–131.

Nelson, B., J. Morrow, and P. Khosla (1995): "Improved force control through visual servoing." In *Proc. American Control Conf.*, pp. 380–386. Seattle, WA.

Nilsson, K. (1996): *Industrial Robot Programming*. PhD thesis ISRN LUTFD2/TFRT--LUTFD2/TFRT–1046–SE--SE, Lund University, Lund, Sweden.

Nordin, M. and P.-O. Gutman (2002): "Controlling mechanical systems with backlash—a survey." *Automatica*, **38:10**, pp. 1633–1650.

Olsson, T. (2001): "Vision guided force control in robotics." Master's Thesis ISRN LUTFD2/TFRT--5676--SE. Department of Automatic Control, Lund University, Sweden.

Olsson, T. (2004): "Feedback control and sensor fusion of vision and force." Licentiate thesis ISRN LUTFD2/TFRT--3235--SE. Department of Automatic Control, Lund Institute of Technology, Sweden.

Olsson, T., J. Bengtsson, R. Johansson, and H. Malm (2002): "Force control and visual servoing using planar surface identification." In *IEEE Int. Conference on Robotics and Automation*, pp. 4211–4216. Washington D.C., USA.

Olsson, T., J. Bengtsson, A. Robertsson, and R. Johansson (2003): "Visual position tracking using dual quaternions with hand-eye motion constraints." In *IEEE Int. Conference on Robotics and Automation*, pp. 3491–3496. Taipei, Taiwan.

Olsson, T., R. Johansson, and A. Robertsson (2004): "Flexible force-vision control for surface following using multiple cameras." In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2004*, pp. 798–803. Sendai, Japan.

Olsson, T., R. Johansson, and A. Robertsson (2005): "Force/vision based active damping control of contact transition in dynamic environments." In Vidal *et al.*, Eds., *ICCV 2005 Workshop on Dynamical Vision*, vol. 4358 of *Lecture Notes in Computer Science*. Springer.

Olsson, T., R. Johansson, and A. Robertsson (2006): "High-speed visual robot control using an optimal linearizing intensity-based filtering approach." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1212–1217. Beijing, China.

Olsson, T., A. Robertsson, and R. Johansson (2007): "Flexible force control for accurate low-cost robot drilling." In *IEEE Int. Conf. Robotics and Automation*. Rome, Italy. To appear.

Park, J. and K. Khatib (2005): "Multi-link multi-contact force control for manipulators." In *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3624–3629. Barcelona, Spain.

Pichler, A. and M. Jägersand (2000): "Uncalibrated hybrid force-vision manipulation." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, vol. 3, pp. 1866–1871. Takamatsu, Japan.

Pomares, J. and F. Torres (2005): "Movement-flow-based visual servoing and force control fusion for manipulation tasks in unstructured environments." *IEEE Trans. Systems, Man and Cybernetics: Part C—Applications and Reviews*, **35:1**, pp. 4–16.

Pressigout, M. and E. Marchand (2005): "Real time planar structure tracking for visual servoing: a contour and texture approach." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1701–1706. Edmonton, Canada.

Raibert, M. H. and J. J. Craig (1981): "Hybrid position/force control of manipulators." *ASME Journal of Dynamic Systems, Measurement and Control*, **103:2**, pp. 126–133.

Robertsson, A., T. Olsson, B. Lauwers, K. Nilsson, T. Brogårdh, A. Blomdell, H. De Baerdemaeker, M. Haage, R. Johansson, and H. Brantmark (2006): "Implementation of industrial robot force control—Case study: High power stub grinding and deburring." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 2743–2748. Beijing, China.

Rosenhahn, B., C. Perwass, and G. Sommer (2005): "Pose estimation of 3d free-form contours." *Int. J. Computer Vision*, **62:3**, pp. 267–290.

Rosten, E. and T. Drummond (2005): "Fusing points and lines for high performance tracking." In *Proc. 10th IEEE Int. Conf. Computer Vision*, pp. 1508–1515. Beijing, China.

Roy, J. and L. L. Whitcomb (2002): "Adaptive force control of position/velocity controlled robots: Theory and experiment." *IEEE Trans. Robotics and Automation*, **18:2**, pp. 121–138.

Santibanez, V. and R. Kelly (1997): "Energy shaping based controllers for rigid and elastic joint robots: analysis via passivity theorems." In *IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 2225–2231. Albuquerque, NM.

Scheiber, S. (2006): "Camera Link and GigE improve image speeds." *Test and Measurement World*, **26:1**, pp. 52–57.

Schuurman, D. and D. Capson (2004): "Robust direct visual servo using network-synchronized cameras." *IEEE Trans. Robotics and Automation*, **20:2**, pp. 319–334.

Sclaroff, S. and J. Isidoro (2003): "Active blobs: region-based, deformable appearance models." *Computer Vision and Image Understanding*, **89:2-3**, pp. 197–225.

Siciliano, B. and L. Villani (1999): *Robot Force Control*. Kluwer Academic Publishers, Dordrecht, NL.

Smith, S. and J. Brady (1997): "SUSAN-a new approach to low level image processing." *Int. J. of Computer Vision*, **23:1**, pp. 45–78.

Spong, M. and M. Vidyasagar (1989): *Robot Dynamics and Control*. John Wiley & Sons.

Spong, M. W. (1989): "On the force control problem for flexible joint manipulators." *IEEE Trans. Automatic Control*, **34**, pp. 107–111.

Takegaki, M. and S. Arimoto (1981): "New feedback method for dynamic control of manipulators." *ASME J. Dynamic Systems, Measurement and Control*, **103:2**, pp. 119–125.

Trucco, E. and A. Verri (1998): *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, New Jersey.

Tsai, R. and R. Lenz (1989): "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration." *IEEE Trans. Robotics and Automation*, **5**, June, pp. 345–358.

van Dam, A., L. Foley, S. Feiner, and J. Hughes (1991): *Computer Graphics: Principles and Practice*, 2nd edition. Addison-Wesley, Boston, MA.

Vincze, M. (2000): "Dynamics and system performance of visual servoing." In *Proc. Int. Conf. Robotics and Automation*. San Francisco, CA.

von Collani, Y., M. Ferch, J. Zhang, and A. Knoll (2000): "A general learning approach to multisensor based control using statistic indices." In *IEEE Int. Conf. Robotics and Automation*, vol. 4, pp. 3221–3226. San Francisco, CA.

Vukobratovic, M., V. Potkonjak, and A. Rodic (2004): "Contribution to the dynamic study of humanoid robots interacting with dynamic environment." *Robotica*, **22:4**, pp. 439–447.

Wernholt, E. (2004): "On multivariable and nonlinear identification of industrial robots." Technical Report Licentiate Thesis no. 1131. Department of Electrical Engineering, Linköping University.

Won, J., S. Stramigioli, and N. Hogan (1997): "Comment on 'The equivalence of second-order impedance control and proportional gain explicit force control'." *Int. J. of Robotics Research*, **16:6**, pp. 873–876.

Wu, Y., T. Tarn, and N. Xi (1995): "Force and transition control with environmental uncertainties." In *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, pp. 899–904. Nagoya, Japan.

Wunsch, P. and G. Hirzinger (1997): "Real-time visual tracking of 3D objects with dynamic handling of occlusion." In *IEEE Int. Conf. Robotics and Automation*, pp. 2868–2873. Albuquerque, NM.

Xiao, D., B. Ghosh, N. Xi, and T. Tarn (2000): "Sensor-based hybrid position/force control of a robot manipulator in an uncalibrated environment." *IEEE Trans. Control Systems Technology*, **8:4**, pp. 635–645.

Yoshikawa, T. (2000): "Force control of robot manipulators." In *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 220–226. San Francisco, CA.

Zhang, Z. (1999): "Flexible camera calibration by viewing a plane from unknown orientations." In *Proc. IEEE Int. Conf. Computer Vision*, vol. 1, pp. 666–673. Corfu, Greece.

Zhao, Y., C. Cheah, and J. Slotine (2006): "Adaptive vision and force tracking control for constrained robots." In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1484–1489. Beijing, China.

Zhou, K. and J. Doyle (1998): *Essentials Of Robust Control*. Prentice Hall, Upper Saddle River, NJ.

Zhou, Y., B. Nelson, and B. Vikramaditya (1998): "Fusing force and vision feedback for micromanipulation." In *IEEE Int. Conf. on Robotics and Automation*, pp. 1220–1225. Leuven, Belgium.