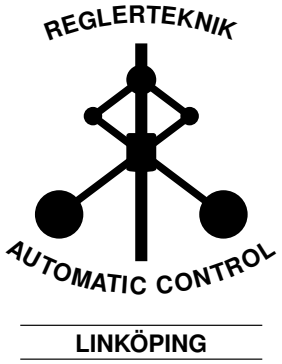
Linköping studies in science and technology. Thesis.
No. 1370

Pose Estimation and Calibration Algorithms for Vision and Inertial Sensors

Jeroen Hol



Division of Automatic Control
Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden
http://www.control.isy.liu.se
hol@isy.liu.se

Linköping 2008

This is a Swedish Licentiate's Thesis.

Swedish postgraduate education leads to a Doctor's degree and/or a Licentiate's degree.
A Doctor's Degree comprises 240 ECTS credits (4 years of full-time studies).
A Licentiate's degree comprises 120 ECTS credits,
of which at least 60 ECTS credits constitute a Licentiate's thesis.

Linköping studies in science and technology. Thesis.
No. 1370

Pose Estimation and Calibration Algorithms for Vision and Inertial Sensors

Jeroen Hol

hol@isy.liu.se
www.control.isy.liu.se
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping
Sweden

ISBN 978-91-7393-862-4 ISSN 0280-7971 LiU-TEK-LIC-2008:28

Printed by LiU-Tryck, Linköping, Sweden 2008

to Nantje

Abstract

This thesis deals with estimating position and orientation in real-time, using measurements from vision and inertial sensors. A system has been developed to solve this problem in unprepared environments, assuming that a map or scene model is available. Compared to 'camera-only' systems, the combination of the complementary sensors yields an accurate and robust system which can handle periods with uninformative or no vision data and reduces the need for high frequency vision updates.

The system achieves real-time pose estimation by fusing vision and inertial sensors using the framework of nonlinear state estimation for which state space models have been developed. The performance of the system has been evaluated using an augmented reality application where the output from the system is used to superimpose virtual graphics on the live video stream. Furthermore, experiments have been performed where an industrial robot providing ground truth data is used to move the sensor unit. In both cases the system performed well.

Calibration of the relative position and orientation of the camera and the inertial sensor turn out to be essential for proper operation of the system. A new and easy-to-use algorithm for estimating these has been developed using a gray-box system identification approach. Experimental results show that the algorithm works well in practice.

Acknowledgments

Good two-and-a-half years ago I started my PhD studies at the automatic control group in Linköping. I received a warm welcome from Fredrik Gustafsson and Thomas Schön, my supervisors, who invited me to Sweden. Thank you for doing so! I really appreciate our many discussions and the frequent application of red pencil and I hope to continue our fruitful collaboration in the future. Furthermore I would like to thank Lennart Ljung and Ulla Salaneck for enabling me to work in such a pleasant atmosphere.

Living in foreign country with a unknown language is always difficult in the beginning. I would like to thank the entire control group for making learning the Swedish language such fun. My special thanks go to David Törnqvist and Johan Sjöberg. You introduced me to the Swedish culture, listened to my stories and took care of distraction from work.

This thesis has been proofread by Linda Schipper, Christian Lundquist and Gustaf Hendeby. You kept an eye on the presentation of the material and improved the quality a lot, which is really appreciated.

Furthermore I would like to thank Per Slycke, Henk Luinge and the rest of the Xsens team for the collaboration over the years and for giving me the opportunity to finish this thesis at Xsens.

Parts of this work have been supported by the MATRIS project, a sixth framework research program within the European Union, which is hereby gratefully acknowledged.

Finally, I would like to thank Nantje for being patient and spontaneous. You bring love and joy to my life and make it all worthwhile.

Linköping, May 2008

Jeroen Hol

Contents

1 Introduction **1**
- 1.1 Problem formulation . . . 2
- 1.2 Contributions . . . 3
- 1.3 Thesis outline . . . 4

2 System overview **5**
- 2.1 Introduction . . . 7
- 2.2 Sensors . . . 8
- 2.3 Sensor fusion . . . 12
- 2.4 Implementation considerations . . . 17
- 2.5 Experiments . . . 19
- 2.6 Conclusion . . . 24
- References . . . 24

3 Sensors **29**
- 3.1 Inertial measurement unit . . . 29
  - 3.1.1 Sensor model . . . 30
  - 3.1.2 Calibration . . . 33
  - 3.1.3 Strapdown inertial navigation . . . 33
- 3.2 Vision . . . 34
  - 3.2.1 Sensor model . . . 35
  - 3.2.2 Calibration . . . 38
  - 3.2.3 Correspondence detection . . . 39

4 State space models **41**
- 4.1 Kinematics . . . 41
  - 4.1.1 Translation . . . 42

4.1.2 Rotation . . . 42
4.1.3 Time derivatives . . . 44
4.2 Continuous-time models . . . 45
4.3 Discrete-time models . . . 47

5 Calibration theory **51**
5.1 Kinematic relations . . . 52
5.1.1 Acceleration . . . 52
5.1.2 Angular velocity . . . 54
5.2 Geometric measurements . . . 58
5.2.1 Direction vectors . . . 58
5.2.2 Position and orientation . . . 59
5.3 Mixing kinematic and geometric measurements . . . 64

6 Calibration algorithms **67**
6.1 Internal calibration . . . 68
6.2 External calibration . . . 70
6.3 Experiments . . . 70

7 Application example **77**

8 Concluding remarks **81**
8.1 Conclusions . . . 81
8.2 Future work . . . 82

Bibliography **83**

A Quaternion preliminaries **89**
A.1 Operations and properties . . . 89
A.2 Exponential . . . 90
A.3 Matrix/vector notation . . . 90

B Conversions **93**
B.1 Rotation matrices . . . 93
B.2 Euler angles . . . 93
B.3 Rotation vector . . . 94

1

Introduction

Knowledge about *position and orientation* (pose) is a key ingredient in many applications. One such application can be found in the field of *augmented reality* (AR). Here, one of the main ideas is to overlay a real scene with computer generated graphics in real-time. This can be accomplished by showing the virtual objects on see-through head-mounted displays or superimposing them on live video imagery. Figure 1.1 illustrates the concept of AR with some examples. In order to have realistic augmentation it is essential to know the position and orientation of the camera with high accuracy and low latency. This knowledge is required to position and align the virtual objects correctly with the real world and they appear to stay in the same location regardless of the camera movement.

In this thesis the problem of pose estimation is approached using the combination of a camera and an *inertial measurement unit* (IMU). In theory, a 'vision only approach' suffices for pose estimation. Such an approach can give good absolute accuracy, but is difficult to run at high frame rate and is not robust during fast motions. The main justification for adding an IMU— by itself accurate for a short period, but drift-prone for longer timescales — is to obtain a robust system. This approach, partly inspired from the human sensory system, is becoming a promising solution as it is a self-contained system requiring no external infrastructure.

The combination of inertial and vision sensors has been previously used in literature, see e.g., Corke et al. (2007) for an introduction. Reported systems apply various methods: inertial measurements are used as backup (Aron et al., 2007), for short time pose prediction (Klein and Drummond, 2004), or depth map alignment (Lobo and Dias, 2004). Alternatively, vision and inertial subsystems are loosely coupled, using visual pose measurements (Ribo et al., 2004; Chroust and Vincze, 2004; Armesto et al., 2007). Vision relies either on specific targets, line contours or natural landmarks. Calibration of the sensors is discussed in e.g., (Lobo and Dias, 2007). Furthermore, the problem is closely related to the problem of *simultaneous localization and mapping* (SLAM) (Durrant-Whyte and Bailey, 2006; Thrun et al., 2005), where camera tracking and scene model reconstruction are performed simultaneously. Single camera SLAM is discussed in e.g., Davison (2003);

(a) Sport coverage using virtual annotations.

(b) Maintenance assistance.

(c) Visualization of virtual objects in TV shows.

(d) On-site virtual reconstruction.

Figure 1.1: *Examples of augmented reality applications. Courtesy of BBC R&D and Fraunhofer IGD.*

Davison et al. (2007); Klein and Murray (2007).

1.1 Problem formulation

The work in this thesis has been performed within the EU project MATRIS (MATRIS, 2008), where the objective is to develop a hybrid camera tracking system using vision and inertial sensors. By using a 3D scene model containing natural landmarks, there is no need for a prepared environment with artificial markers. This will remove the costly and time consuming procedure of preparing the environment, and allow for AR applications outside dedicated studios, for instance outdoors.

A schematic overview of the approach is shown in Figure 1.2. The inertial measurement unit provides rapid measurements of acceleration and angular velocity. The computer vision system generates correspondences between the camera image and the scene model. This 3D scene model contains positions of various natural markers and is generated offline using images and/or drawings of the scene. The inertial and vision measurements are combined in the sensor fusion model to obtain the camera pose. By using the pose estimates in the computer vision module, a tightly-coupled system is obtained.

The problem of estimating camera pose from inertial and visual measurements is formulated as a nonlinear state estimation problem. This thesis deals with the question of

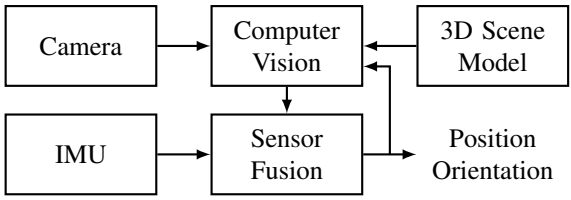


Figure 1.2: *Estimating camera pose by fusing measurements from an inertial measurement unit and a computer vision system.*

how to solve this nonlinear state estimation problem in real-time using the available sensor information. Furthermore, several issues, including calibration, are addressed in order to obtain a solution working in practice.

1.2 Contributions

The main contributions of the thesis are:

- The development, testing and evaluation of a real-time pose estimation system based on vision and inertial measurements.
- The derivation of process and measurements models for this system which can be used for nonlinear state estimation of position and orientation.
- The development of a new and easy-to-use calibration procedure to determine the relative position and orientation of a rigidly connected camera and IMU.

Some aspects have been previously published in

> F. Gustafsson, T. B. Schön, and J. D. Hol. Sensor fusion for augmented reality. In *Proceedings of 17th International Federation of Automatic Control World Congress*, Seoul, South Korea, July 2008. Accepted for publication.
>
> J. D. Hol, T. B. Schön, H. Luinge, P. J. Slycke, and F. Gustafsson. Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing*, 2(2):149–160, Nov. 2007. doi:10.1007/s11554-007-0040-2.
>
> J. D. Hol, T. B. Schön, F. Gustafsson, and P. J. Slycke. Sensor fusion for augmented reality. In *Proceedings of 9th International Conference on Information Fusion*, Florence, Italy, July 2006b. doi:10.1109/ICIF.2006.301604.

Outside the scope of this thesis fall the following conference papers

> J. D. Hol, T. B. Schön, and F. Gustafsson. On resampling algorithms for particle filters. In *Proceedings of Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, Sept. 2006a. doi:10.1109/NSSPW.2006.4378824.
>
> G. Hendeby, J. D. Hol, R. Karlsson, and F. Gustafsson. A graphics processing unit implementation of the particle filter. In *Proceedings of European Signal Processing Conference*, Poznań, Poland, Sept. 2007.

1.3 Thesis outline

This thesis is organized in the following way: Chapter 2 gives an overview of the developed pose estimation system. It is an edited version of the paper originally published as Hol et al. (2007) and discusses the setup, the sensor fusion algorithm and the performance evaluation of the system.

The sensor unit consisting of an IMU and a camera is the subject of Chapter 3. The operating principles, measurements and processing algorithms of these sensors are discussed. In Chapter 4 the process and measurement models of the sensor fusion algorithm for real-time camera pose estimation are derived.

Calibration of the relative position and orientation between the IMU and the camera is essential for proper operation of the pose estimation system. Similar types of problems occur when the estimated pose is compared to that of an external reference. Chapter 5 presents a theoretical framework for solving the relative pose calibration problem using various types of measurements. This theory is applied in Chapter 6 to develop a number of calibration algorithms.

The pose estimation system has been tested as an augmented reality application. The result of this experiment is the topic of Chapter 7. Finally, Chapter 8 concludes this thesis and gives suggestions for further work.

2

System overview

This chapter provides an overview of the developed pose estimation system. It is an edited version of the paper originally published as

> J. D. Hol, T. B. Schön, H. Luinge, P. J. Slycke, and F. Gustafsson. Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing*, 2(2):149–160, Nov. 2007. doi:10.1007/s11554-007-0040-2.

and discusses the setup, the sensor fusion algorithm and the performance evaluation of the system.

Robust real-time tracking by fusing measurements from inertial and vision sensors

J. D. Hol[a], T. B. Schön[a], H. Luinge[b], P. J. Slycke[b] and F. Gustafsson[a]

[a] *Linköping University, Division of Automatic Control, SE–581 83 Linköping, Sweden*

[b] *Xsens Technologies B.V., Pantheon 6a, Postbus 559, 7500 AN Enschede, The Netherlands*

Abstract

The problem of estimating and predicting position and orientation (pose) of a camera is approached by fusing measurements from inertial sensors (accelerometers and rate gyroscopes) and vision. The sensor fusion approach described in this contribution is based on non-linear filtering of these complementary sensors. This way, accurate and robust pose estimates are available for the primary purpose of augmented reality applications, but with the secondary effect of reducing computation time and improving the performance in vision processing.

A real-time implementation of a multi-rate extended Kalman filter is described, using a dynamic model with 22 states, where 100 Hz inertial measurements and 12.5 Hz correspondences from vision are processed. An example where an industrial robot is used to move the sensor unit is presented. The advantage with this configuration is that it provides ground truth for the pose, allowing for objective performance evaluation. The results show that we obtain an absolute accuracy of 2 cm in position and 1° in orientation.

2.1 Introduction

This paper deals with estimating the position and orientation (pose) of a camera in real-time, using measurements from inertial sensors (accelerometers and rate gyroscopes) and a camera. A system has been developed to solve this problem in unprepared environments, assuming that a map or scene model is available. For a more detailed description of the overall system and the construction of scene models we refer to Chandaria et al. (2007) and Koeser et al. (2007), respectively. In this paper, the sensor fusion part of the system is described, which is based upon a rather general framework for nonlinear state estimation available from the statistical signal processing community.

This problem can under ideal conditions be solved using only a camera. Hence, it might seem superfluous to introduce inertial sensors. However, the most important reasons justifying an *inertial measurement unit* (IMU) are:

- Producing more *robust* estimates. Any single camera system will experience problems during periods with uninformative or no vision data. This will occur, typically due to occlusion or fast motion. An IMU will help to bridge such gaps, which will be illustrated in the present paper.

- Reducing computational demands for image processing. Accurate short time pose estimates are available using the information from the IMU, reducing the need for fast vision updates.

The combination of vision and inertial sensors has been used previously in literature. Corke et al. (2007) give an introduction to this field and its applications. Reported systems apply various methods: inertial measurements are used as backup (Aron et al., 2007), for short time pose prediction (Klein and Drummond, 2004), or depth map alignment (Lobo and Dias, 2004). Alternatively, vision and inertial subsystems are loosely coupled, using visual pose measurements (Ribo et al., 2004; Chroust and Vincze, 2004; Armesto et al., 2007). Vision relies either on specific targets, line contours or natural landmarks. Calibration of the sensors is discussed in e.g., (Lobo and Dias, 2007). Furthermore, the problem is closely related to the problem of *simultaneous localization and mapping* (SLAM) (Durrant-Whyte and Bailey, 2006; Thrun et al., 2005), where camera tracking and scene model construction are performed simultaneously. Single camera SLAM is discussed in Davison (2003); Davison et al. (2007). In that context so called fast localization algorithms (Williams et al., 2007) are investigated as alternatives to inertial support (Pinies et al., 2007; Gemeiner et al., 2007).

In our approach, real-time camera pose estimation is achieved by fusing inertial and vision measurements using the framework of nonlinear state estimation, covering methods such as the *Extended Kalman Filter* (EKF), the *Unscented Kalman Filters* (UKF) and the *particle filter* (PF). This results in a tightly coupled system, naturally supporting multi-rate signals. The vision measurements are based on natural landmarks, which are detected guided by pose predictions. The measurements from the sensors are used directly rather than being processed to a vision based pose or an inertial based pose. The components of the system are well known. However, we believe that the way in which these components are assembled is novel and we show that the resulting system provides accurate and robust pose estimates.

The sensors generating the measurements y_t are described in Section 2.2. In Section 2.3, the framework for state estimation in nonlinear dynamic systems is introduced in more detail and used to solve the sensor fusion problem we are faced with in the present application. In implementing this, there are several practical issues that have to be solved. The overall performance of the system heavily relies on successful solutions to these matters, which is explained in Section 2.4. The performance of the implementation is evaluated in Section 2.5, and finally, the paper is concluded in Section 2.6.

2.2 Sensors

An IMU and a digital video camera are combined to provide measurements to the sensor fusion module, described in this paper. Both sensors are relatively small and unobtrusive and they can be conveniently integrated into a single *sensor unit*. An example of a proto-

type is shown in Figure 2.1. An on board digital signal processor containing calibration

Figure 2.1: *A prototype of the MATRIS project, integrating a camera and an IMU in a single housing. It provides a hardware synchronized stream of video and inertial data.*

parameters is used to calibrate and synchronize data from the different components.

Before discussing the inertial and vision sensors in the subsequent sections, the required coordinate systems are introduced.

2.2.1 Coordinate systems

When working with a sensor unit containing a camera and an IMU several coordinate systems have to be introduced:

- **Earth (e)**: The camera pose is estimated with respect to this coordinate system. It is fixed to earth and the features of the scene are modeled in this coordinate system. It can be aligned in any way; however, preferably it should be vertically aligned.
- **Camera (c)**: The coordinate system attached to the moving camera. Its origin is located in the optical center of the camera, with the z-axis pointing along the optical axis. The camera, a projective device, acquires its images in the **image plane (i)**. This plane is perpendicular to the optical axis and is located at an offset (focal length) from the optical center of the camera.
- **Body (b)**: This is the coordinate system of the IMU. Even though the camera and the IMU are rigidly attached to each other and contained within a single package, the body coordinate system does not coincide with the camera coordinate system. They are separated by a constant translation and rotation.

These coordinate systems are used to denote geometric quantities, for instance, $\boldsymbol{c}^e$ is the position of the camera coordinate system expressed in the earth system and R^{cb} is the rotation matrix from the body system to the camera system.

2.2.2 Inertial sensors

The sensor unit contains an IMU with three perpendicularly mounted $1200\ \mathrm{degree/s}$ ADXLXRS300 angular velocity sensors and two 5g 2D ADXL22293 accelerometers, which are mounted such that three of the sensitive axes are perpendicular to each other. MEMS rate gyroscopes are chosen because of their dramatically reduced size and low cost as compared to alternatives such as fiber optic angular velocity sensors.

The signals from the inertial components are synchronously measured at 100 Hz using a 16 bit A/D converter. A temperature sensor is added to compensate for the temperature dependency of the different sensing components.

The assembly containing the gyroscopes and accelerometers has been subjected to a calibration procedure to calibrate for the exact physical alignment of each component, the gains, the offsets and the temperature relations of the gains and offsets. With these a 3D angular velocity vector and a 3D accelerometer vector, both resolved in the body coordinate system, are computed using an on board processor. See e.g., Titterton and Weston (1997); Chatfield (1997) for suitable background material on inertial sensors and the associated signal processing.

The calibrated gyroscope signal $\boldsymbol{y}_{\omega,t}$ contains measurements of the angular velocity $\boldsymbol{\omega}^b_{eb,t}$ from body to earth $(_{eb})$ expressed in the body coordinate system $(^b)$:

$$\boldsymbol{y}_{\omega,t} = \boldsymbol{\omega}^b_{eb,t} + \boldsymbol{\delta}^b_{\omega,t} + \boldsymbol{e}^b_{\omega,t}. \tag{2.1}$$

Even though the gyroscope signal is corrected for temperature effects, some low-frequency offset fluctuations $\boldsymbol{\delta}_{\omega,t}$ remain, partly due to the unmodeled acceleration dependency. The remaining error $\boldsymbol{e}^b_{\omega,t}$ is assumed to be zero mean white noise. The measurements are not accurate enough to pick up the rotation of the earth. This implies that the earth coordinate system can be considered to be an inertial frame.

A change in orientation can be obtained by proper integration of the gyroscope signal. This orientation can be obtained even during fast and abrupt movements, not relying on any infrastructure other than the gyroscope itself. However, the accuracy in orientation will deteriorate for periods longer than a few seconds.

The calibrated accelerometer signal $\boldsymbol{y}_{a,t}$ contains measurements of the combination of the body acceleration vector $\ddot{\boldsymbol{b}}_t$ and the gravity vector $\boldsymbol{g}$, both expressed in the body coordinate system:

$$\boldsymbol{y}_{a,t} = \ddot{\boldsymbol{b}}^b_t - \boldsymbol{g}^b + \boldsymbol{\delta}^b_{a,t} + \boldsymbol{e}^b_{a,t}. \tag{2.2}$$

Even though the accelerometer measurement is corrected for temperature effects a small low-frequency offset $\boldsymbol{\delta}_{a,t}$ remains. The error $\boldsymbol{e}^b_{a,t}$ is assumed to be zero mean white noise.

Gravity is a constant vector in the earth coordinate system. However, expressed in body coordinates gravity depends on the orientation of the sensor unit. This means that once the orientation is known, the accelerometer signal can be used to estimate the acceleration, or alternatively, once the acceleration is known, the direction of the vertical can be estimated.

Accelerations can be integrated twice to obtain a change in position. This can be done during fast and abrupt motions as long as an accurate orientation estimate is available, for instance from the gyroscopes. However, the accuracy of the position change will

deteriorate quickly as a result of the double integration and the sensitivity with respect to orientation errors.

2.2.3 Monocular vision

Apart from the inertial sensors, the sensor unit is equipped with a ptGrey DragonFly CCD camera with a perspective lens with a focal length of $3.2\ \mathrm{mm}$. Color images with a resolution of 320x240 pixels at a frame rate of $12.5\ \mathrm{Hz}$ are streamed to a PC using a firewire connection. The camera is triggered by the IMU clock allowing for synchronized measurements.

This setup is one realization of monocular vision: cameras can vary in sensor type, resolution, frame rate and various lens types can be used, ranging from perspective to fish-eye. However, they remain projective devices, that is, they are bearings only sensors which do not provide distance directly.

Extracting camera position and orientation from images is a known and well studied problem in computer vision (Ma et al., 2006; Hartley and Zisserman, 2004). The key ingredient is to find correspondences, relations between features found in the image which correspond to an element in the scene model. All these are rather abstract concepts, which do have numerous implementations, ranging from Harris detectors (Harris and Stephens, 1988) and point clouds models to patches and textured free-form surfaces models (Koeser et al., 2007). The correspondences are the pieces of information which can be extracted from an image and they will be considered to be the vision measurements in this article.

Point correspondences $\boldsymbol{z}^c \leftrightarrow \boldsymbol{z}^i$ are the relation between 3D points $\boldsymbol{z}^c$ and 2D image points $\boldsymbol{z}^i$. For a perspective lens and a pinhole camera the correspondence relation is

$$\boldsymbol{z}^i = \begin{pmatrix} f z_x^c / z_z^c \\ f z_y^c / z_z^c \end{pmatrix} + \boldsymbol{e}^i, \tag{2.3a}$$

or equivalently,

$$\mathbf{0} \approx \begin{pmatrix} -fI_2 & \boldsymbol{z}_t^i \end{pmatrix} \boldsymbol{z}_t^c = \begin{pmatrix} -fI_2 & \boldsymbol{z}_t^i \end{pmatrix} R_t^{ce} (\boldsymbol{z}^e - \boldsymbol{c}_t^e), \tag{2.3b}$$

where f is the focal length and I_2 the 2×2 identity matrix. The error $\boldsymbol{e}_t^i$ is assumed to be a zero mean white noise. Here it is worth noting that this assumption is not that realistic, due to outliers, quantization effects etc. From (2.3b) it can be seen that the camera pose depends on the rotation matrix R^{ce} and the position $\boldsymbol{c}^e$. Hence, given sufficient correspondences and a calibrated camera the camera pose can be solved for. Similar relations can be derived for e.g., line correspondences which also provide information about the camera pose and optical velocity fields which provide information about the camera velocity (Corke et al., 2007).

Correspondences are bearings only measurements and as such they provide information about absolute position and orientation with respect to the earth coordinate system. Note that everything is determined up to a scale ambiguity; viewing a twice as large scene from double distance will yield an identical image. However, these vision measurements are available at a relatively low rate due to the trade off between exposure time and accuracy (pixel noise and motion blur) which is an important limit for small aperture cameras. Furthermore, processing capacity might constrain the frame rate. Hence, the observed

image can change drastically from frame to frame, which occurs already with normal human motion. This is the main cause for the limited robustness inherent in single camera systems.

The computer vision implementation used in the present implementation is based on a *sum of absolute difference* (SAD) block matcher in combination with a planar patch or free-form surface model of the scene. More details can be found in Chandaria et al. (2007); Koeser et al. (2007); Skoglund and Felsberg (2007). Both pixel data and 3D positions are stored for each feature. An example of a scene model is shown in Figure 2.2. While tracking, search templates are generated by warping the patches in the model ac-

Figure 2.2: *An example of a scene model consisting of planar patches (*lower right*) and the actual scene that is modeled (*upper left*).*

cording to homographies calculated from the latest prediction of the camera pose. These templates are then matched with the current calibrated camera image using the block matcher. In this way correspondences are generated.

2.3 Sensor fusion

The inertial and vision sensors contained in the sensor unit have complementary properties. Vision in combination with the map gives accurate absolute pose information at a low rate, but experiences problems during moderately fast motions. The IMU provides high rate relative pose information regardless of the motion speed, but becomes inaccurate after a short period of time. By fusing information from both sources it is possible to obtain robust camera pose estimates.

Combing inertial and vision sensors is possible in several ways. For instance, vision based methods might be extended by using pose predictions from the IMU. These pose predictions can be used to determine where in the image the features are to be expected. Once detected, the features can be used to calculate the pose and this pose is then used as a starting point for the next pose prediction by the IMU. Alternatively, the IMU can be considered to be the main sensor, which is quite common in the navigation industry. In that case, vision can be used for error correction, similar to how radio beacons or the *global positioning system* (GPS) are used to correct the drift in an *inertial navigation system* (INS).

Although the sensors have different properties, it is from a signal processing perspective not relevant to assign a 'main' sensor and an 'aiding' sensor. Both vision and inertial sensors are equivalent in the sense that they both provide information about the quantity of interest, the camera pose in this application. The objective is to extract as much information as possible from the measurements. More specifically, this amounts to finding the best possible estimate of the filtering *probability density function* (pdf) $p(x_t|y_{1:t})$, where $y_{1:t} \triangleq \{y_1, \ldots, y_t\}$. The topic of this section is to provide a solid framework for computing approximations of this type. First, a rather general introduction to this framework is given in Section 2.3.1. The rest of this section is devoted to explaining how this framework can be applied to handle the present application. The models are introduced in Section 2.3.2 and the fusion algorithm is discussed in Section 2.3.3.

2.3.1 Theoretical framework

The objective in sensor fusion is to recursively in time estimate the state in the dynamic model,

$$x_{t+1} = f_t(x_t, u_t, v_t), \tag{2.4a}$$
$$y_t = h_t(x_t, u_t, e_t), \tag{2.4b}$$

where $x_t \in \mathbb{R}^{n_x}$ denotes the state, $y_t \in \mathbb{R}^{n_y}$ denote the measurements from a set of sensors, v_t and e_t denote the stochastic process and measurement noise, respectively. The process model equations, describing the evolution of the states (pose etc.) over time are denoted by $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$. Furthermore, the measurement model is given by $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_e} \to \mathbb{R}^{n_y}$, describing how the measurements from the IMU and the camera relate to the state. The goal is to infer all the information from the measurements y_t onto the state x_t. The way of doing this is to compute the filtering pdf $p(x_t|y_{1:t})$. The filtering pdf contains everything there is to know about the state at time t, given the information in all the past measurements $y_{1:t}$. Once an approximation of $p(x_t|y_{1:t})$ is available it can be used to form many different (point) estimates, including maximum likelihood estimates, confidence intervals and the most common conditional expectation estimate

$$\hat{x}_t = E(x_t|y_{1:t}). \tag{2.5}$$

The key element in solving the nonlinear state estimation problem in real-time is the propagation of $p(x_t|y_{1:t})$ over time. It is well known (see e.g., Jazwinski, 1970) that a recursive solution can be obtained by applying Bayes' theorem, introducing model (2.4) in the iterations,

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t}, \tag{2.6a}$$
$$p(x_{t+1}|y_{1:t}) = \int p(x_{t+1}|x_t)p(x_t|y_{1:t})dx_t. \tag{2.6b}$$

Hence, the quality of the solution is inherently coupled to the models and hence good models are imperative. It is worth noticing that (2.6a) and (2.6b) are often referred to as

measurement update and time update, respectively. The sensor fusion problem has now been reduced to propagating (2.6) over time as new measurements arrive. The problem is that the multidimensional integrals present in (2.6) lack analytical solutions in all but a few special cases. The most common special case is when (2.4) is restricted to be a linear dynamic system, subject to additive Gaussian noise. Then all the involved densities will be Gaussian, implying that it is sufficient to propagate the mean and covariance. The recursions updating these are of course given by the Kalman filter (Kalman, 1960).

However, in most cases there does not exist a closed form solution for (2.6), forcing the use of approximations of some sort. The literature is full of different ideas on how to perform these approximations. The most common being the EKF (Smith et al., 1962; Schmidt, 1966) where the model is linearized and the standard Kalman filter equations are used for this linearized model. A conceptually more appealing approximation is provided by the PF (Gordon et al., 1993; Isard and Blake, 1998; Kitagawa, 1996) which retains the model and approximates (2.6). Other popular approximations for the nonlinear state estimation problem are provided for example by the UKF (Julier and Uhlmann, 2004) and the point-mass filter (Bucy and Senne, 1971; Bergman, 1999). For a more complete account of the nonlinear state estimation problem, see e.g., Schön (2006).

2.3.2 Models

The probability density functions $p(x_{t+1}|x_t)$ and $p(y_t|x_t)$ are the key elements in the filter iterations (2.6). They are usually implicitly specified by the process model (2.4a) and the measurement model (2.4b). For most applications the model formulation given in (2.4) is too general. It is often sufficient to assume that the noise enters additively, according to

$$x_{t+1} = f_t(x_t) + v_t, \tag{2.7a}$$
$$y_t = h_t(x_t) + e_t. \tag{2.7b}$$

The fact that the noise is additive in (2.7) allows for explicit expressions for $p(x_{t+1}|x_t)$ and $p(y_t|x_t)$, according to

$$p(x_{t+1}|x_t) = p_{v_t}(x_{t+1} - f_t(x_t)), \tag{2.8a}$$
$$p(y_t|x_t) = p_{e_t}(y_t - h_t(x_t)), \tag{2.8b}$$

where $p_{v_t}(\cdot)$ and $p_{e_t}(\cdot)$ denote the pdf's for the noise v_t and e_t, respectively. Note that the input signal u_t has been dispensed with, since it does not exist in the present application. The rest of this section will discuss the model used in the current application.

First of all, the state vector has to include the position and the orientation, since they are the quantities of interest. However, in order to be able to use the IMU and provide predictions the state vector should also include their time derivatives, as well as sensor biases. The state vector is chosen to be

$$x_t = \begin{pmatrix} \boldsymbol{b}_t^e & \dot{\boldsymbol{b}}_t^e & \ddot{\boldsymbol{b}}_t^e & q_t^{be} & \boldsymbol{\omega}_{eb,t}^b & \boldsymbol{\delta}_{\omega,t}^b & \boldsymbol{\delta}_{a,t}^b \end{pmatrix}^T. \tag{2.9}$$

That is, the state vector consists of position of the IMU (the body coordinate system) expressed in the earth system $\boldsymbol{b}^e$, its velocity $\dot{\boldsymbol{b}}^e$ and acceleration $\ddot{\boldsymbol{b}}^e$, the orientation of the

body with respect to the earth system q^{be}, its angular velocity $\boldsymbol{\omega}^b_{eb}$, the gyroscope bias $\boldsymbol{\delta}^b_\omega$ and the accelerometer bias $\boldsymbol{\delta}^b_a$. All quantities are three dimensional vectors, except for the orientation which is described using a four dimensional unit quaternion q^{be}, resulting in a total state dimension of 22. Parameterization of a three dimensional orientation is in fact rather involved, see e.g., Shuster (1993) for a good account of several of the existing alternatives. The reason for using unit quaternions is that they offer a nonsingular parameterization with a rather simple dynamics. Using (2.9) as state vector, the process model is given by

$$\boldsymbol{b}^e_{t+1} = \boldsymbol{b}^e_t + T\dot{\boldsymbol{b}}^e_t + \tfrac{T^2}{2}\ddot{\boldsymbol{b}}^e_t, \tag{2.10a}$$
$$\dot{\boldsymbol{b}}^e_{t+1} = \dot{\boldsymbol{b}}^e_t + T\ddot{\boldsymbol{b}}^e_t, \tag{2.10b}$$
$$\ddot{\boldsymbol{b}}^e_{t+1} = \ddot{\boldsymbol{b}}^e_t + \boldsymbol{v}^e_{\ddot{b},t}, \tag{2.10c}$$
$$q^{be}_{t+1} = \exp(-\tfrac{T}{2}\boldsymbol{\omega}^b_{eb,t}) \odot q^{be}_t, \tag{2.10d}$$
$$\boldsymbol{\omega}^b_{eb,t+1} = \boldsymbol{\omega}^b_{eb,t} + \boldsymbol{v}^b_{\omega,t}, \tag{2.10e}$$
$$\boldsymbol{\delta}^b_{\omega,t+1} = \boldsymbol{\delta}^b_{\omega,t} + \boldsymbol{v}^b_{\delta_\omega,t}, \tag{2.10f}$$
$$\boldsymbol{\delta}^b_{a,t+1} = \boldsymbol{\delta}^b_{a,t} + \boldsymbol{v}^b_{\delta_a,t}, \tag{2.10g}$$

where the quaternion multiplication and exponential are defined according to

$$\begin{pmatrix} p_0 \\ \boldsymbol{p} \end{pmatrix} \odot \begin{pmatrix} q_0 \\ \boldsymbol{q} \end{pmatrix} \triangleq \begin{pmatrix} p_0 q_0 - \boldsymbol{p}\cdot\boldsymbol{q} \\ p_0\boldsymbol{q} + q_0\boldsymbol{p} + \boldsymbol{p}\times\boldsymbol{q} \end{pmatrix}, \tag{2.11a}$$
$$\exp(\boldsymbol{v}) \triangleq \begin{pmatrix} \cos\|\boldsymbol{v}\| \\ \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|}\sin\|\boldsymbol{v}\| \end{pmatrix}. \tag{2.11b}$$

A standard constant acceleration model (2.10a)– (2.10c) has been used to model the position, velocity and acceleration. Furthermore, the quaternion dynamics is standard, see e.g., Shuster (1993). Finally, the angular velocity and the bias terms are simply modeled as random walks, since there is no systematic knowledge available about these terms.

There is more than one sensor type available, implying that several measurement models are required. They have already been introduced in Section 2.2, but for convenience they are all collected here,

$$\boldsymbol{y}_{a,t} = R^{be}_t(\ddot{\boldsymbol{b}}^e_t - \boldsymbol{g}^e) + \boldsymbol{\delta}^b_{a,t} + \boldsymbol{e}^b_{a,t}, \tag{2.12a}$$
$$\boldsymbol{y}_{\omega,t} = \boldsymbol{\omega}^b_{eb,t} + \boldsymbol{\delta}^b_{\omega,t} + \boldsymbol{e}^b_{\omega,t}, \tag{2.12b}$$
$$\boldsymbol{y}_{c,t} = \begin{pmatrix} -fI_2 & \boldsymbol{z}^i_t \end{pmatrix} R^{cb}(R^{be}_t(\boldsymbol{z}^e_t - \boldsymbol{b}^e_t) - \boldsymbol{c}^b) + \boldsymbol{e}_{c,t}. \tag{2.12c}$$

Note that the rotation matrix R^{be}_t is constructed from q^{be}_t (Kuipers, 1999). The transformation from body to camera coordinate system is included in (2.12c), compared to (2.3b).

2.3.3 Fusion Algorithm

The nonlinear estimation framework discussed in Section 2.3.1 suggests Algorithm 2.1 to fuse the multi-rate information from the inertial and vision sensors. The algorithm uses

Algorithm 2.1 Recursive camera pose calculation

1. Perform an initialization and set initial state estimate and covariance.

$$x_0 \sim p(x_o)$$

2. Time update. Calculate $p(x_t|y_{1:t-1})$ by propagating $p(x_{t-1}|y_{1:t-1})$ through the process model (2.10).

3. Accelerometer and gyroscope measurement update using model (2.12a) and (2.12b).

$$x_t \sim p(x_t|y_{1:t})$$

4. If there is a new image from the camera,

   (a) Predict feature positions from the scene model using $\hat{x}_t = E(x_t|y_{1:t})$.

   (b) Detect the features in the image.

   (c) Measurement update with the found point correspondences using model (2.12c).

$$x_t \sim p(x_t|y_{1:t})$$

5. Set $t := t + 1$ and iterate from step 2.

the models (2.10) and (2.12) to perform the time and measurement update steps given in (2.6). Note that Algorithm 2.1 is generic in the sense that we have not specified which state estimation algorithm is used. Our implementation, which runs in real-time with 100 Hz inertial measurements and frame rates up to 25 Hz, uses the EKF to compute the estimates, implying that all involved pdf's are approximated by Gaussian densities. An UKF implementation was found to give similar accuracy at the cost of a higher computational burden (Pieper, 2007). This confirms the results from Armesto et al. (2007).

When the sensor unit is static during initialization, the IMU provides partial or full (using magnetometers) orientation estimates. This information can be used to constrain the search space when initializing from vision.

The high frequency inertial measurement updates in Algorithm 2.1 provide a rather accurate state estimate when a new image is acquired. This implies that the feature positions can be predicted with an improved accuracy, which in turn makes it possible to use a guided search in the image using reduced search regions. The algorithm can calculate the expected covariance of a measurement. This can be the basis for a temporal outlier removal as a complement to the spatial outlier removal provided by RANSAC methods (Fischler and Bolles, 1981). Alternatively it can be used to predict the amount of new information that a certain feature can contribute, which might be useful for task scheduling when the computational resources are limited (Davison, 2005).

The camera pose is estimated implicitly by Algorithm 2.1 rather than trying to determine it explicitly by inverting the measurement equations. Hence, when sufficient motion

is present, the system is able to continue tracking with a very low number of features and maintain full observability using temporal triangulation.

The information from the IMU makes Algorithm 2.1 robust for temporary absence of vision. Without vision measurements the estimates will eventually drift away. However, short periods without vision, for instance, due to motion blur, obstruction of the camera or an unmodeled scene, can be handled without problems.

Finally, Algorithm 2.1 is rather flexible. It can be rather straightforwardly extended to include other information sources. For instance, a GPS might be added to aid with outdoor applications.

2.4 Implementation considerations

When implementing Algorithm 2.1, several practical issues have to be solved. These turn out to be critical for a successful system, motivating their treatment in this section.

2.4.1 Metric scale

As mentioned in Section 2.2.3, vision-only methods suffer from a scale ambiguity, since projections, unit-less measurements, are used. Once the scale of the scene model is defined, camera pose can be determined explicitly using three or more correspondences in combination with a calibrated camera. However, changing the scale of a scene model will give scaled, but indistinguishable poses. Hence, for vision-only applications scene models can have an arbitrary scale; a standard choice is to define the unit length to be the distance between the first two cameras.

For the inertial-vision combination, the scale is relevant. Sensor fusion utilizes position information both from the camera and the IMU, which implies that these quantities must have identical units. Scale is also important when assumptions are made about the motions of the camera, for instance the type and parameters of a motion model (Davison et al., 2007).

Introducing a metric scale into the scene model solves this issue. An existing scene model with arbitrary scale can be converted by comparing it with a *Computer Aided Design* (CAD) model or measuring an object with known dimension. An interesting solution might be to include metric information, for instance using accelerometers, in the algorithms for building the scene models. However, this is still an open question.

2.4.2 Vertical alignment

Accelerometers cannot distinguish accelerations of the body from gravity, as previously discussed in Section 2.2.2. To separate the contributions in the measurement, the gravity vector can be rotated from the earth coordinate system to the body frame and then subtracted. Hence, the scene model should be vertically aligned, or equivalently the gravity vector should be known in the scene model. Typically, this is not the case.

The performance of the system is extremely sensitive to this alignment, since gravity is typically an order of magnitude larger than normal body accelerations. For example, a misalignment of 1° introduces an artificial acceleration of $0.17\ \mathrm{m/s^2}$ which gives rise to a

systematic position drift of $8.5\ \mathrm{cm}$ when integrated over $1\ \mathrm{s}$. Hence, even for small errors a systematic drift is introduced which causes the system to lose track without continuous corrections from correspondences. In this case the drift followed by a correction gives rise to a saw tooth pattern in the estimates, which deteriorates performance and will be visible as 'jitter'.

The gravity vector can be determined by averaging the accelerometer readings over some time, while the camera is stationary in a known pose. However, a preferable method is to record accelerometer measurements while scanning the scene and include this data in the model building procedure to align the scene model vertically.

2.4.3 Sensor pose calibration

The camera and the IMU both deliver measurements which are resolved in the camera and the body coordinate system, respectively. Typically, these do not coincide, since the sensors are physically translated and rotated with respect to each other. This rigid transformation should be taken into account while fusing the measurements.

The problem of determining the relative position and orientation is a well studied problem in robotics where it is known as hand-eye calibration, see for instance Strobl and Hirzinger (2006) for an introduction to this topic. However, most methods do not apply directly since the IMU does not provide an absolute position reference. Absolute orientation information is available since the accelerometers measure only gravity when the sensor unit is stationary.

The orientation part of the calibration is determined using a slight modification of standard camera calibration procedures (Zhang, 2000), where the calibration pattern is placed on a horizontal surface and accelerometer readings are taken in the various camera poses. The camera poses are determined in the camera calibration procedure, from which the vertical directions in the camera frame can be determined. The combination of these and the vertical directions in the body frame measured by the accelerometers allows for calculation of the rotation between the frames (Horn, 1987; Lobo and Dias, 2007). This method requires accurate positioning of the calibration pattern. As floors and desks in buildings are in practice better horizontally aligned than the walls are vertically aligned, it is recommended to use horizontal surfaces.

The translational part of the calibration is harder to estimate and a solid calibration method which does not require special hardware is an open issue. The translation should also be available from technical drawings of the sensor unit and a rough guess using a ruler gives a quite decent result in practice. However, with increasing angular velocity this parameter becomes more dominant and an accurate calibration is necessary.

2.4.4 Time synchronization

It is very important to know exactly when the different measurements are taken. Multiple sensors usually have multiple clocks and these have to be synchronized. This can be achieved for instance by starting them simultaneously. However, clocks tend to diverge after a while, which will introduce problems during long term operation. Hardware synchronization, i.e., one central clock is used to trigger the other sensors, solves this problem and this procedure has been applied in the sensor unit described in Section 2.2.

2.4.5 Filter tuning

The process and measurement models described in Section 2.3 have a number of stochastic components which are used to tune the filter. The settings used in the present setup are given in Table 2.1. The measurement noise typically depends on the sensors and should be experimentally determined. For the accelerometers and gyroscopes a measurement of a few seconds with a static pose was recorded to calculate an accurate noise covariance. Alternatively, the specification by the manufacturer can be used.

The noise acting on the vision measurements is harder to determine. The algorithms return a point estimate for the obtained matches, but typically there is no stochastic information available. The accuracy for each match is highly individual and can vary a lot depending on e.g., lighting conditions, local texture, viewing angle, distance and motion blur. These individual characteristics cannot be captured by a common noise setting. Hence, it would be beneficial to include accuracy estimation in the image processing algorithms. Although attempts are being made to solve this open issue, see e.g., Skoglund and Felsberg (2007), the current implementation uses a predefined noise covariance.

The process model currently used is a random walk in acceleration and angular velocity. This model is not so informative but is very general and is useful for tracking uncontrolled motions such as those generated by a human. The motion model is to be considered as a separate source of information, apart from the sensors. Hence, when more information is available in a certain application, for instance in the form of control signals, these should be included in the model to improve the filter performance. The covariances in the process model can be seen as tuning knobs, controlling the relative importance of the measurements and the process model and as such they are important parameters for stable tracking.

Valid models and parameters are imperative to obtain good estimates. The innovations, defined as the difference between a measurement and its expected value,

$$e_t = y_t - \hat{y}_t, \tag{2.13}$$

can be used to asses whether the models are correctly tuned. Under the model assumptions, the innovations should be normally distributed and the squared normalized innovations $e_t^T S_t^{-1} e_t$, where S_t is the predicted covariance of the measurement, should have a χ^2 distribution. It is highly recommendable to monitor these performance indicators, especially during testing, but also during normal operation.

2.5 Experiments

This section is concerned with an experiment where Algorithm 2.1 with an EKF is used to fuse the measurements from the sensor unit in order to compute estimates of its position and orientation. The experimental setup is discussed in Section 2.5.1 and the performance of the proposed inertial-vision combination provided by the sensor unit is assessed in Section 2.5.2.

2.5.1 Setup

The sensor unit is mounted onto a high precision 6 degrees of freedom (DoF) ABB IRB1440 industrial robot, see Figure 2.3. The reason for this is that the robot will allow

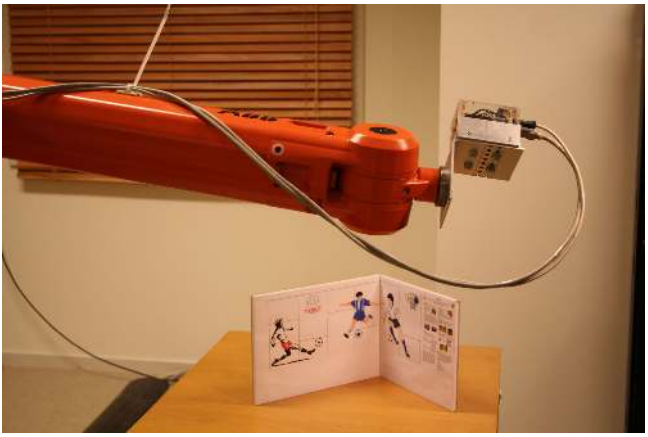


Figure 2.3: *The camera and the IMU are mounted onto an industrial robot. The* background *shows the scene that has been used in the experiments.*

us to make repeatable 6 DoF motions and it will provide the true position and orientation. The robot has an absolute accuracy of $2\ \mathrm{mm}$ and a repeatability of $0.2\ \mathrm{mm}$. This enables systematic and rather objective performance evaluation of various algorithms, based on absolute pose errors instead of the commonly used feature reprojection errors. The sensor unit provides $100\ \mathrm{Hz}$ inertial measurements synchronized with $12.5\ \mathrm{Hz}$ images. The complete specification is listed in Table 2.1. The scene used for the experiments consists of two orthogonal planar surfaces as shown in Figure 2.3. Because of the simple geometry, the scene model could be constructed from a textured CAD model. Its coordinate system is such that the x-axis points upward and that the y and z-axis span the horizontal plane. Although the scene was carefully positioned, it had to be calibrated w.r.t. gravity as described in Section 2.4.2. It should be emphasized that the scene has been kept simple for experimentation purposes only. The system itself can handle very general scenes and these are modeled using the methods described in Koeser et al. (2007).

With the setup several trajectories have been tested. In this paper, an eight-shaped trajectory, shown in Figure 2.4, will be discussed in detail. The sensor unit traverses this $2.6\ \mathrm{m}$ eight-shaped trajectory in $5.4\ \mathrm{s}$, keeping the scene in view at all times. The motion contains accelerations up to $4\ \mathrm{m/s^2}$ and angular velocities up to $1\ \mathrm{rad/s}$. Hence, the motion is quite aggressive and all six degrees of freedom are exited. As the displacement between images is limited to $15\ \mathrm{pixels}$ it is still possible to use vision-only tracking, which allows for a comparison between tracking with and without an IMU.

The experiment starts with a synchronization motion, which is used to synchronize the ground truth data from the industrial robot with the estimates from the system. Time synchronization is relevant, since a small time offset between the signals will result in a significant error. After the synchronization, the eight-shaped trajectory (see Figure 2.4)

Table 2.1: *Specifications for the sensor unit and the parameter values used for in the filter tuning. Note that the noise parameters specify the standard deviation.*

IMU	
gyroscope range	± 20.9 rad/s
gyroscope bandwidth	40 Hz
accelerometer range	± 17 m/s^2
accelerometer bandwidth	30 Hz
sample rate	100 Hz
Camera	
selected resolution	320×240 pixel
pixel size	7.4×7.4 µm/pixel
focal length	3.2 mm
sample rate	12.5 Hz
Filter settings	
gyroscope measurement noise	0.01 rad/s
accelerometer measurement noise	0.13 m/s^2
2D feature measurement noise	0.1 pixel
3D feature measurement noise	1 mm
angular velocity process noise	0.03 rad/s
acceleration process noise	0.1 m/s^2
gyroscope bias process noise	0.5 mrad/s
accelerometer bias process noise	0.5 mm/s^2

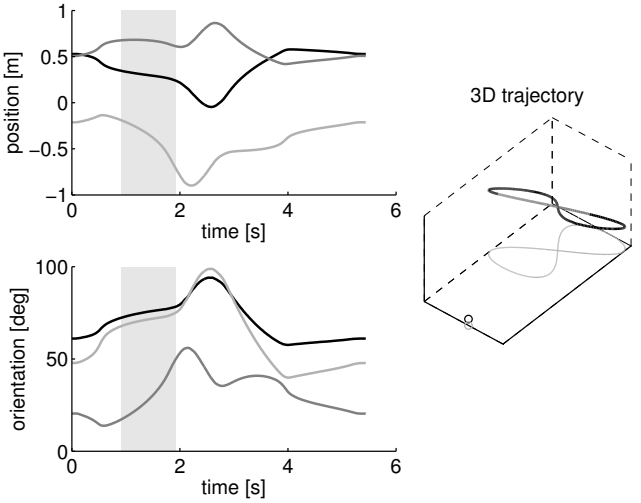


Figure 2.4: *The eight-shaped trajectory undertaken by the sensor unit. The* gray shaded parts *mark the interval where vision is deactivated. The* circle *indicates the origin of the scene model.*

is repeated several times, utilizing the accurate and repeatable motion provided by the industrial robot.

2.5.2 Results

The experimental setup described in the previous section is used to study several aspects of the combination of vision and inertial sensors. The quality of the camera pose estimates is investigated by comparing them to the ground truth data. Furthermore, the increased robustness of the system is illustrated by disabling the camera for $1\ \mathrm{s}$ during the second pass of the eight-shaped trajectory. Additionally, the feature predictions are shown to benefit from the inertial measurements. The findings will be discussed in the following paragraphs.

By comparing the estimates from the filter to the ground truth the tracking errors are determined. Examples of position and orientation errors (z, roll) are shown in Figure 2.5. The other positions (x, y) and orientations (yaw, pitch) exhibit similar behavior. The

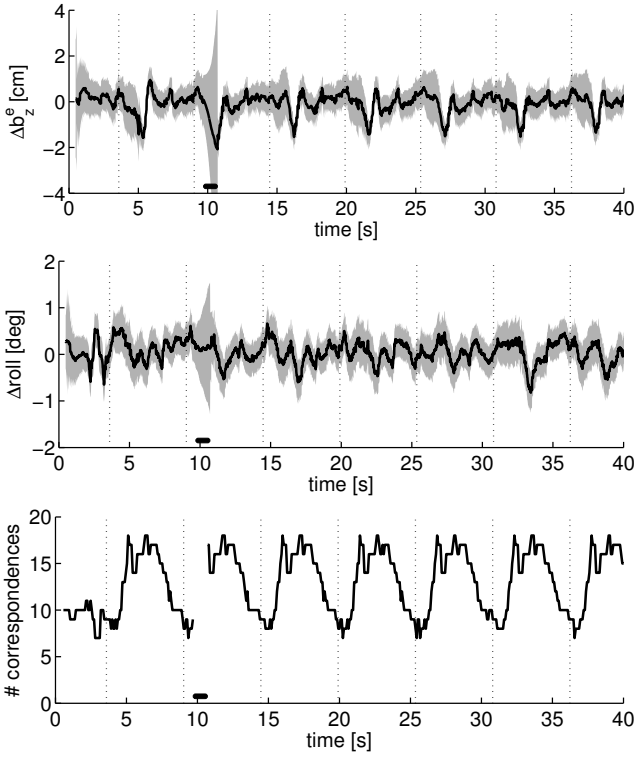


Figure 2.5: *Tracking error during multiple passes of the eight-shaped trajectory. The* black line *shows the position (z) and orientation (roll) errors, as well as the number of correspondences that were used. The* gray band *illustrates the 99% confidence intervals. Note that vision is deactivated from* $9.7\ \mathrm{s}$ *to* $10.7\ \mathrm{s}$. *The* vertical dotted lines *mark the repetition of the motion.*

absolute accuracy (with vision available) is below $2\ \mathrm{cm}$ for position and below 1° for orientation. These values turn out to be typical for the performance of the system in the setup described above. Furthermore, the accuracy of the IMU is not affected by the speed of motion, resulting in a tracking accuracy which is rather independent of velocity, as illustrated by Figure 2.6 which shows the tracking error of the eight-shaped trajectory executed at various speeds. Other experiments, not described here, show similar performance for

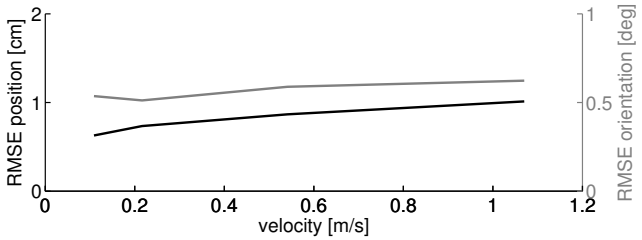


Figure 2.6: *Tracking error for several experiments executing the eight-shaped trajectory at different speeds.*

various trajectories.

A proper treatment of the implementation considerations as discussed in Section 2.4 is necessary in order to obtain good performance. Still, calibration errors and slight misalignments as well as scene model errors and other unmodeled effects are causes for non-white noise, which can deteriorate the performance. However, with the assumptions and models used, the system is shown to estimate the camera pose quite accurately using rather low-rate vision measurements. The estimated camera poses result in good and stable augmentation.

The system tracks the camera during the entire experiment, including the period where vision is deactivated. The motion during this period, indicated using gray segments in Figure 2.4, is actually quite significant. Vision-only tracking has no chance of dealing with such a gap and loses track. Indeed, such an extensive period where vision is deactivated is a little artificial. However, vision might be unavailable or corrupted, due to fast rotations, high velocity, motion blur, or simply too few visible features. These difficult, but commonly occurring, situations can be dealt with by using an IMU as well, clearly illustrating the benefits of having an IMU in the system. In this way, robust real-time tracking in realistic environments is made possible.

The measurements from the IMU will also result in better predictions of the feature positions in the acquired image. This effect is clearly illustrated in Figure 2.7, which provides a histogram of the feature prediction errors. The figure shows that the feature prediction errors are smaller and more concentrated in case the IMU measurement updates are used. This improvement is most significant when the camera is moving fast or at lower frame rates. At lower speeds, the vision based feature predictions will improve and the histograms will become more similar.

The improved feature predictions facilitate the use of smaller search regions to find the features. This implies that using an IMU more features can be detected, given a certain processing power. On the other hand, the improved feature predictions indicate that the IMU handles the fast motion and that the absolution pose information which

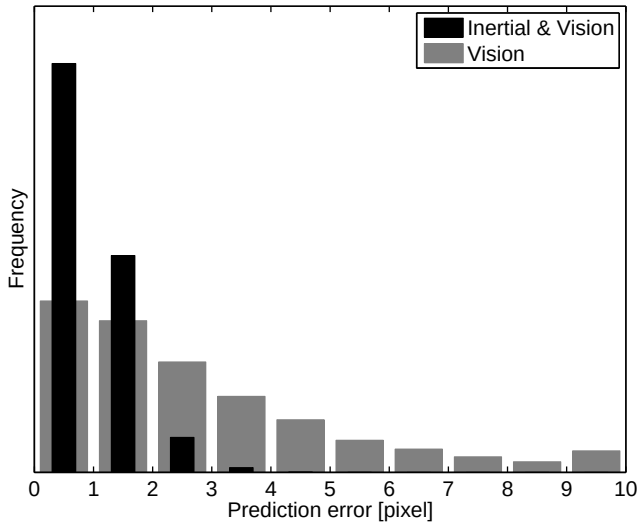


Figure 2.7: *Histogram of the prediction errors for the feature positions. The feature predictions are calculated using the latest vision pose and the most recent inertial pose, respectively.*

vision provides is required at a reduced rate.

2.6 Conclusion

Based on a framework for nonlinear state estimation, a system has been developed to obtain real-time camera pose estimates by fusing 100 Hz inertial measurements and 12.5 Hz vision measurements using an EKF. Experiments where an industrial robot is used to move the sensor unit show that this setup is able to track the camera pose with an absolute accuracy of 2 cm and 1°. The addition of an IMU yields a robust system which can handle periods with uninformative or no vision data and it reduces the need for high frequency vision updates.

Acknowledgments

This work has been performed within the MATRIS consortium, which is a sixth framework research program within the European Union (EU), contract number: IST-002013.

References

L. Armesto, J. Tornero, and M. Vincze. Fast ego-motion estimation with multi-rate fusion of inertial and vision. *International Journal of Robotics Research*, 26(6):577–589, 2007. doi:10.1177/0278364907079283.

M. Aron, G. Simon, and M.-O. Berger. Use of inertial sensors to support video tracking. *Computer Animation and Virtual Worlds*, 18(1):57–68, 2007. doi:10.1002/cav.161.

N. Bergman. *Recursive Bayesian Estimation: Navigation and Tracking Applications*. Dissertations no 579, Linköping Studies in Science and Technology, SE-581 83 Linköping, Sweden, May 1999.

R. S. Bucy and K. D. Senne. Digital synthesis on nonlinear filters. *Automatica*, 7:287–298, 1971. doi:10.1016/0005-1098(71)90121-X.

J. Chandaria, G. A. Thomas, and D. Stricker. The MATRIS project: real-time markerless camera tracking for augmented reality and broadcast applications. *Journal of Real-Time Image Processing*, 2(2):69–79, Nov. 2007. doi:10.1007/s11554-007-0043-z.

A. Chatfield. *Fundamentals of High Accuracy Inertial Navigation*, volume 174. American Institute of Aeronautics and Astronautics, USA, 3rd edition, 1997. ISBN 1563472430.

S. G. Chroust and M. Vincze. Fusion of vision and inertial data for motion and structure estimation. *Journal of Robotics Systems*, 21(2):73–83, 2004. doi:10.1002/rob.10129.

P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *International Journal of Robotics Research*, 26(6):519–535, 2007. doi:10.1177/0278364907079279.

A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of 9th IEEE International Conference on Computer Vision*, volume 2, pages 1403–1410, Nice, France, Oct. 2003. doi:10.1109/ICCV.2003.1238654.

A. J. Davison. Active search for real-time vision. In *Proceedings of 10th IEEE International Conference on Computer Vision*, pages 66–73, Beijing, China, Oct. 2005. doi:10.1109/ICCV.2005.29.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (6):1052–1067, June 2007. doi:10.1109/TPAMI.2007.1049.

H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping (SLAM): Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, June 2006. doi:10.1109/MRA.2006.1638022.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. doi:10.1145/358669.358692.

P. Gemeiner, P. Einramhof, and M. Vincze. Simultaneous motion and structure estimation by fusion of inertial and vision data. *International Journal of Robotics Research*, 26(6): 591–605, 2007. doi:10.1177/0278364907080058.

N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, 140(2):107–113, Apr. 1993. ISSN 0956-375X.

C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, Manchester, UK, 1988.

R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004. ISBN 0521540518.

J. D. Hol, T. B. Schön, H. Luinge, P. J. Slycke, and F. Gustafsson. Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing*, 2(2):149–160, Nov. 2007. doi:10.1007/s11554-007-0040-2.

B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, Apr. 1987.

M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998. doi:10.1023/A:1008078328650.

A. H. Jazwinski. *Stochastic processes and filtering theory*. Mathematics in science and engineering. Academic Press, New York, USA, 1970. ISBN 978-0123815507.

S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, Mar. 2004. doi:10.1109/JPROC.2003.823141.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35–45, 1960.

G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, Mar. 1996.

G. S. W. Klein and T. W. Drummond. Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing*, 22(10):769–776, 2004. doi:10.1016/j.imavis.2004.02.007.

K. Koeser, B. Bartczak, and R. Koch. Robust GPU-assisted camera tracking using free-form surface models. *Journal of Real-Time Image Processing*, 2(2):133–147, Nov. 2007. doi:10.1007/s11554-007-0039-8.

J. B. Kuipers. *Quaternions and Rotation Sequences*. Princeton University Press, 1999. ISBN 0691102988.

J. Lobo and J. Dias. Inertial sensed ego-motion for 3D vision. *Journal of Robotics Systems*, 21(1):3–12, 2004. doi:10.1002/rob.10122.

J. Lobo and J. Dias. Relative pose calibration between visual and inertial sensors. *International Journal of Robotics Research*, 26(6):561–575, 2007. doi:10.1177/0278364907079276.

Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An invitation to 3-D vision – from images to geometric models*. Interdisciplinary Applied Mathematics. Springer-Verlag, 2006. ISBN 0387008934.

R. J. B. Pieper. Comparing estimation algorithms for camera position and orientation. Master's thesis, Department of Electrical Engineering, Linköping University, Sweden, 2007.

P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardos. Inertial aiding of inverse depth SLAM using a monocular camera. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2797–2802, Roma, Italy, Apr. 2007. doi:10.1109/ROBOT.2007.363895.

M. Ribo, M. Brandner, and A. Pinz. A flexible software architecture for hybrid tracking. *Journal of Robotics Systems*, 21(2):53–62, 2004. doi:10.1002/rob.10124.

S. F. Schmidt. Application of state-space methods to navigation problems. *Advances in Control Systems*, 3:293–340, 1966.

T. B. Schön. *Estimation of Nonlinear Dynamic Systems – Theory and Applications*. Dissertations no 998, Linköping Studies in Science and Technology, Department of Electrical Engineering, Linköping University, Sweden, Feb. 2006.

M. D. Shuster. A survey of attitude representations. *The Journal of the Astronautical Sciences*, 41(4):439–517, Oct. 1993.

J. Skoglund and M. Felsberg. Covariance estimation for SAD block matching. In *Proc. 15th Scandinavian Conference on Image Analysis*, pages 374–382, 2007. doi:10.1007/978-3-540-73040-8_38.

G. L. Smith, S. F. Schmidt, and L. A. McGee. Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle. Technical Report TR R-135, NASA, 1962.

K. H. Strobl and G. Hirzinger. Optimal hand-eye calibration. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4647–4653, Beijing, China, Oct. 2006. doi:10.1109/IROS.2006.282250.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. The MIT Press, Cambridge, MA, USA, 2005. ISBN 978-0-262-20162-9.

D. H. Titterton and J. L. Weston. *Strapdown inertial navigation technology*. IEE radar, sonar, navigation and avionics series. Peter Peregrinus Ltd., Stevenage, UK, 1997. ISBN 0863413587.

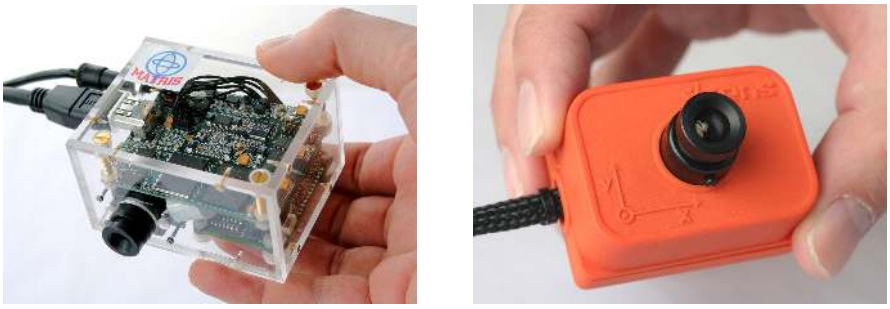
B. Williams, P. Smith, and I. Reid. Automatic relocalisation for a single-camera simultaneous localisation and mapping system. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2784–2790, Roma, Italy, Apr. 2007. doi:10.1109/ROBOT.2007.363893.

Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov. 2000. doi:10.1109/34.888718.

3

Sensors

Chapter 2 introduced the sensor unit and its application. This sensor unit consists of an inertial measurement unit (IMU) and a camera which are integrated in a singe small housing. The sensors are synchronized at hardware level, significantly simplifying the signal processing. Figure 3.1 shows two versions of the sensor unit. In the upcoming



(a) *2005* ***(b)*** *2007*

Figure 3.1: *Two version of the* sensor unit, *showing progressing product design and miniaturization.*

sections of this chapter, the operating principles, measurements and processing algorithms of the inertial measurement unit and the camera will be discussed in more detail.

3.1 Inertial measurement unit

The IMU within the sensor unit contains a 3D rate gyroscope and a 3D linear accelerometer. The gyroscope and accelerometer are based on *micro-machined electromechanical*

systems (MEMS) technology, see Figure 3.2. Compared to traditional technology, MEMS

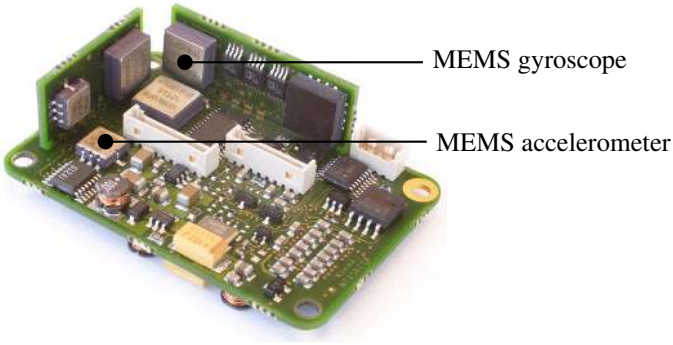


Figure 3.2: *The MEMS components are integrated into the circuit board of the IMU.*

devices are small, light, inexpensive, have low power consumption and short start-up times. Currently, their major disadvantage is the reduced performance in terms of accuracy and bias stability. This is the main cause for the drift in standalone MEMS inertial navigation systems (Woodman, 2007).

The functionality of the MEMS sensors are based upon simple mechanical principles. Angular velocity can be measured by exploiting the Coriolis effect of a vibrating structure. When a vibrating structure is being rotated, a secondary vibration is induced from which the angular velocity can be calculated. Acceleration can be measured with a spring suspended mass. When subjected to acceleration the mass will be displaced. Using MEMS technology the necessary mechanical structures can be manufactured on silicon chips in combination with capacitive displacement pickups and electronic circuitry (Analog Devices, 2008).

3.1.1 Sensor model

The MEMS accelerometer and gyroscope sensors have one or more sensitive axes along which a physical quantity (specific force and angular velocity, respectively) is converted to an output voltage. A typical sensor shows almost linear behavior in the working area as illustrated in Figure 3.3. Based on this linear behavior in a sensitive axis, the following relation between the output voltage $\boldsymbol{u}$ and the physical signal $\boldsymbol{y}$ is postulated for multiple sensors with their sensitive axis aligned in a suitable configuration,

$$\boldsymbol{u}_t = GR\boldsymbol{y}_t + \boldsymbol{b}. \tag{3.1}$$

Here G is a diagonal matrix containing the individual gains g, R is the alignment matrix specifying the direction of the sensitive axis w.r.t. the sensor housing and $\boldsymbol{b}$ is the offset vector. Note that the gain and the offset are typically temperature dependent. The calibrated measurement signal $\boldsymbol{y}_t$ is obtained from the measured voltages by inverting (3.1).

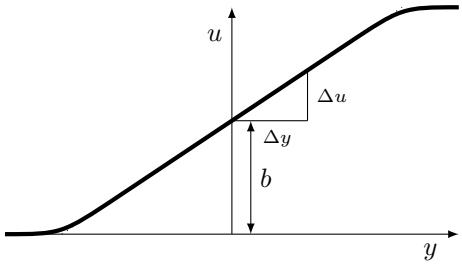


Figure 3.3: *Schematic behavior of an inertial sensor. The output voltage u depends almost linearly on the physical quantity y, where y denotes angular velocity or external specific force. This relation is parameterized by an offset b and a slope or gain $g = \frac{\Delta u}{\Delta y}$.*

Gyroscopes

The 3D rate gyroscope measures angular velocities resolved in the body coordinate frame, that is, with respect to (w.r.t.) the sensor housing. The sensor noise is characterized in Figure 3.4. The histogram of Figure 3.4a shows that the noise distribution is close to a Gaussian. However, the minima in the Allan deviation of Figure 3.4b indicate that even under constant conditions a slowly varying sensor bias is present (IEEE Std 952-1997, 1998). Additionally, calibration errors (errors in gain, alignment and linearity) as well as uncompensated temperature effects result in bias. The fluctuating behavior of the bias is usually approximated with a random walk.

Summarizing the previous discussion, the 3D rate gyroscope measurements $\boldsymbol{y}_\omega$ are modeled as

$$\boldsymbol{y}_\omega = \boldsymbol{\omega}_{eb}^b + \boldsymbol{\delta}_\omega^b + \boldsymbol{e}_\omega^b, \tag{3.2}$$

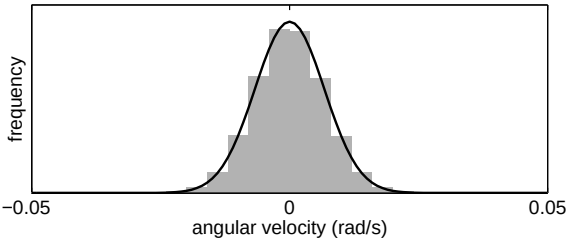
where $\boldsymbol{\omega}_{eb}^b$ is the angular velocity, body to earth, expressed in the body coordinate frame, $\boldsymbol{\delta}_\omega^b$ is a slowly varying sensor bias and $\boldsymbol{e}_\omega^b$ is white noise.

Accelerometers

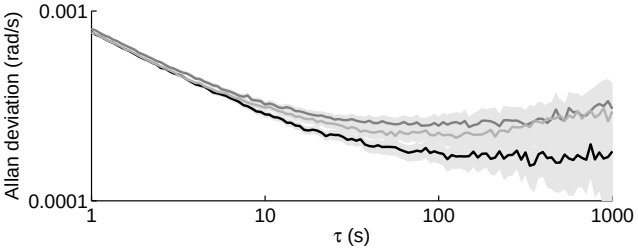
Contradictory to what their name implies, accelerometers do not measure accelerations. Instead, they measure the total external specific force acting on the sensor. Although acceleration is related to specific force by Newton's law, the two are not identical as shown in the following example: an accelerometer lying still on a table undergoes zero acceleration but will measure a force of $1\ \mathrm{g}$ pointing upward due to the earth's gravitational field. By subtracting gravity, acceleration can be recovered. Alternatively, the accelerometer can be used as an inclinometer when no acceleration is present.

Similar to the 3D gyroscope, the 3D accelerometer measurements suffer from white noise and a slowly varying bias, see Figure 3.5. With these, the accelerometer measurements $\boldsymbol{y}_a$ are modeled as

$$\boldsymbol{y}_a = \boldsymbol{f}^b + \boldsymbol{\delta}_a^b + \boldsymbol{e}_a^b = R^{be}(\ddot{\boldsymbol{b}}^e - \boldsymbol{g}^e) + \boldsymbol{\delta}_a^b + \boldsymbol{e}_a^b, \tag{3.3}$$

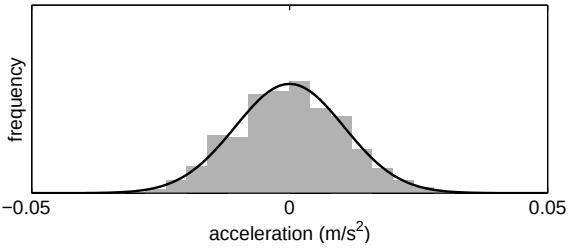


(a) *Histogram together with a Gaussian approximation.*

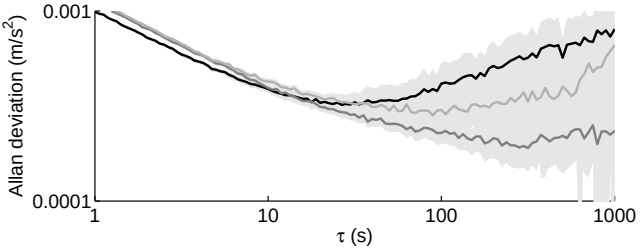


(b) *Allan deviations and their 99% confidence intervals.*

Figure 3.4: *Gyroscope noise characteristics.*



(a) *Histogram together with a Gaussian approximation.*



(b) *Allan deviations and their 99% confidence intervals.*

Figure 3.5: *Accelerometer noise characteristics.*

where $\boldsymbol{f}^b$ is the normalized external specific force in the body coordinate system, $\boldsymbol{\delta}_a^b$ is a slowly varying sensor bias and $\boldsymbol{e}_a^b$ is white noise. The second form the expression splits the specific force into its contributions from the acceleration of the sensor $\ddot{\boldsymbol{b}}^e$ and the gravity vector $\boldsymbol{g}^e$, both expressed in the earth coordinate frame. These vectors have been rotated to the body coordinate frame by the rotation matrix R^{be}.

3.1.2 Calibration

Using the discussion in the previous section, calibrating the IMU boils down to finding the gain G, alignment R and the offset $\boldsymbol{b}$ in (3.1) for the accelerometer and gyroscope. The calibration principle is to subject the IMU to a known acceleration or angular velocity and choose the calibration parameters such that the observed sensor output is as likely as possible. Ignoring the time variability of the biases and using the standard assumptions of independent, identically distributed Gaussian noise, this maximum likelihood optimization can be formulated as

$$\hat{\theta} = \arg\min_{\theta} \sum_{t} \frac{1}{2} \|\boldsymbol{u}_t - h(\boldsymbol{s}_t, \theta)\|^2, \tag{3.4}$$

where the parameter vector θ consists of G, R and $\boldsymbol{b}$. Traditionally, known excitations are obtained using special manipulators such as turntables. Alternatively, the IMU can be placed in several static orientations (Ferraris et al., 1995).

The sensor unit has been calibrated at production using a propriety calibration procedure. Besides gain, alignment and offset also temperature effects and g-sensitivity of the gyroscopes are accounted for (Xsens Motion Technologies, 2008). Recalibration is not necessary unless the housing is opened or the sensor is subjected to a large shock.

3.1.3 Strapdown inertial navigation

Inertial navigation is a technique to compute estimates of the position and orientation of an object relative to a known starting pose using inertial measurements from accelerometers and gyroscopes (Woodman, 2007; Chatfield, 1997; Titterton and Weston, 1997). In a strapdown configuration such as the sensor unit, the measurements are acquired in the body coordinate frame, rather than in an inertial reference frame. Hence, the orientation q^{eb} can be calculated by integrating the angular velocity $\boldsymbol{\omega}_{eb}^b$. The position $\boldsymbol{b}^e$ can be obtained by double integration of the external specific force $\boldsymbol{f}^b$ which has been rotated using the known orientation and corrected for gravity. This procedure is illustrated in Figure 3.6.

In practice, the angular velocity and the external specific force are replaced by the gyroscope and accelerometer measurements. These include bias and noise terms which cause errors in the calculated position and orientation, the integration drift. The gyroscope noise results in a random walk in orientation, whereas a constant gyroscope bias introduces orientation errors which grow linearly in time (Woodman, 2007). Similarly, the accelerometer noise results in a second order random walk in position and a constant accelerometer bias introduces position errors which grow quadratic in time. Note that in Figure 3.6 there is a coupling between position and orientation. Hence, any orientation error introduces an artificial acceleration as gravity is not correctly compensated for: a small,

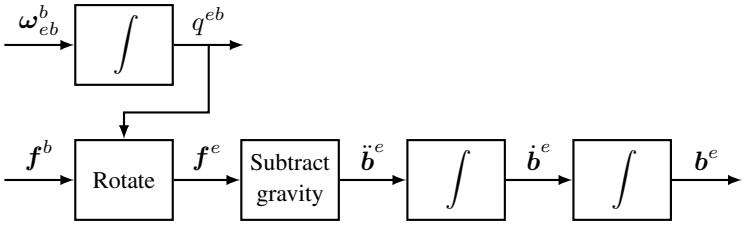


Figure 3.6: *Strapdown inertial navigation algorithm.*

but realistic inclination error of $\theta = 0.1^\circ$ already introduces a non-existing acceleration of $a = g \sin\theta = 0.0017\ \mathrm{m/s^2}$ which gives rise to a position error of $p = \frac{1}{2}at^2 = 3.1\ \mathrm{m}$ in $60\ \mathrm{s}$. For the used inertial sensor, this coupling turns out to be the dominant position error source (Woodman, 2007).

From the previous discussion it follows that any inertial navigation solution deteriorates with time. Using MEMS inertial sensors, the integration drift causes the orientation estimate, but especially the position estimate, to be accurate and reliable only for a short period of time.

3.2 Vision

Besides an IMU, the sensor unit contains a camera. This is a rather complex system which consists of two functional parts: an optical system (the so-called image formation system or objective) which collects light to form an image of an object, and an image sensor, usually a CCD or CMOS, which converts incident light into a digital image.

Various types of objectives exist, each with a specific application area. Examples include standard perspective lenses, wide angle lenses, zoom lenses, macro lenses and fish-eye lenses. In general they are rather complex composite devices composed of a number of functional elements, see Figure 3.7. The most important elements are lens

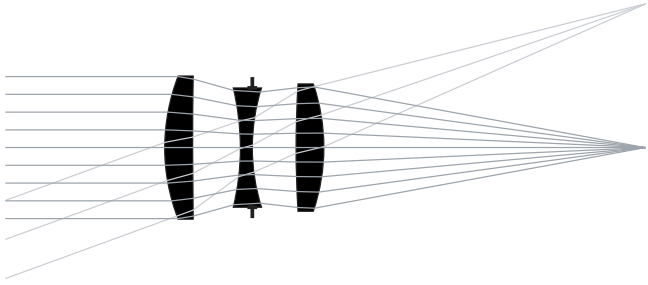


Figure 3.7: *Cross section of a low-cost objective. The triplet of lens elements bundle the parallel rays of light entering the system from the left and to form an image on the right.*

elements and stops. The lens elements have refractive surfaces which bend the light,

whereas the stops limit the bundle of light propagating through the system. Combining a number of elements an optical system can be designed in such a way that the desired image formation is achieved with minimal optical aberration.

From a pure geometric perspective, ignoring effects such as focus and lens thickness, the process of image formation can be described as a central projection (Hartley and Zisserman, 2004). In this projection, a ray is drawn from a point in space toward the camera center. This ray propagates through the optical system and intersects with the image plane where it forms an image of the point.

The perhaps best known example of a central projection is the pinhole camera, see Figure 3.8. Its widespread use in computer vision literature can be explained by noting

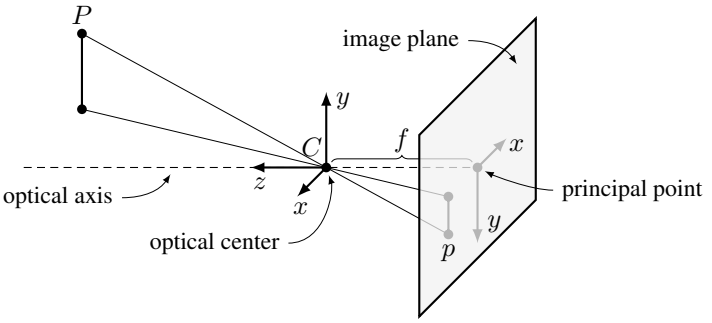


Figure 3.8: *Pinhole camera projection. The image p of a point P is the intersection point of the image plane and the line trough point P and the optical center C. Note that placing the image plane in front of the optical center yields an identical image.*

that a perfect perspective objective is equivalent to a pinhole camera. With this observation, the equivalence between the focal length f and the distance between the optical center and the image plane becomes clear.

Although the pinhole model is powerful model which is sufficient for many applications, it is a simplification of the imaging process. This simplification has its limitations. One of these is that it is unclear where the optical center is located physically in drawings such as Figure 3.7. Clearly, the optical center has to lie somewhere on the optical axis, but exactly where, or even whether it lies behind, inside, or in front of the objective depends highly on the typically unknown detailed design of all the elements in an objective. As discussed in Chapter 2, the location of the optical center is important when combining vision with inertial sensors as in the tracking application at hand. A calibration algorithm which can be used to determine the position of the optical center will be discussed in Chapter 5 and Chapter 6.

3.2.1 Sensor model

The image formation process of a camera consists of two stages: the objective projects incident rays of light to points in the sensor plane and these are converted to a digital image by the image sensor. The former is usually described using an ideal projection combined with a distortion accounting for deviations. The sensor model has to capture

all three phenomena — projection, distortion and digitalization — to describe the image formation of a real camera:

Normalized pinhole projection. Assume for now that the focal length of the pinhole model is unity, that is $f = 1$. In that case the image is called normalized. From similar triangles in Figure 3.8 it follows that the 2D normalized image $\boldsymbol{p}_n^i = (x_n, y_n)^T$ of a 3D point $\boldsymbol{p}^c = (X, Y, Z)^T$, resolved in the camera coordinate system, is given by

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} X \\ Y \end{pmatrix}. \tag{3.5a}$$

This elementary camera model can be classified as a 2D bearings only measurement: from image coordinates it is possible to find a line on which the corresponding 3D point lies. However, it provides no information about the distance or depth of the point. Hence, one cannot tell whether the size of an object is $1\ \mathrm{mm}$ or $1\ \mathrm{km}$. This property is the reason why it is by definition impossible to determine the scale factor in optical reconstruction or in structure from motion problems, where a model of the observed scene is constructed from optical measurements.

In computer vision literature, it is common to work with homogeneous coordinates which are elements of a projective space, see e.g., Hartley and Zisserman (2004). Homogeneous coordinates are obtained from Euclidean ones by augmenting them with an additional 1. Using the homogeneous vectors $\tilde{\boldsymbol{p}}_n^i = (x_n, y_n, 1)^T$ and $\tilde{\boldsymbol{p}}^c = (X, Y, Z, 1)^T$ the normalized pinhole projection (3.5a) can be written as

$$Z \begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\triangleq \Pi_0} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \tag{3.5b}$$

where Π_0 is known as the standard projection matrix. In this form the projection equation is linear, which is of course preferable in many situations. Note that since in projective space equivalence is defined up to scale, (3.5b) is usually written as

$$\lambda \tilde{\boldsymbol{p}}_n^i = \Pi_o \tilde{\boldsymbol{p}}^c, \tag{3.5c}$$

where $\lambda \in \mathbb{R}^+$ is an arbitrary scale factor.

Distortion. The normalized pinhole projection is an ideal projection and the image formation process of a real perspective objective will deviate from it, especially for low-quality objectives. An example is shown in Figure 3.9. The typical distortion encountered is dominated by radial distortion (Hartley and Zisserman, 2004; Zhang, 2000). A simple distortion model to account for the radial distortion expresses the distorted image coordinates $\boldsymbol{p}_d^i = (x_d, y_d)^T$ as a function of the normalized image coordinates $\boldsymbol{p}_n^i = (x_n, y_n)^T$,

$$\boldsymbol{p}_d^i = (1 + k_1 \|\boldsymbol{p}_n^i\|^2 + k_2 \|\boldsymbol{p}_n^i\|^4) \boldsymbol{p}_n^i, \tag{3.6}$$

(a) *Observed image.* ***(b)*** *Ideal image.*

Figure 3.9: *Camera images suffer from distortion.*

where the k_i are distortion coefficients. Several modifications and extensions of this distortion model, which include e.g., tangential distortion, are encountered in literature.

Digitalization. Digital cameras deliver images with coordinates typically specified in pixels and indexed from the top left. Furthermore, there is the possibility of non-square as well as non-orthogonal pixels. This introduces both (non-uniform) scaling and a principal point offset. Both effects, as well as focal lengths $f \neq 1$, can be accounted for by an affine transformation which transforms the distorted image coordinates $\boldsymbol{p}_d^i = (x_d, y_d)^T$ into pixel coordinates $\boldsymbol{p}^i = (x, y)^T$,

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} fs_x & fs_\theta & x_0 \\ 0 & fs_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\triangleq K} \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix}. \tag{3.7}$$

Here, the camera calibration matrix K is composed of the focal length f, the pixel sizes s_x, s_y, the principal point coordinates x_0, y_0 and a skew parameter s_θ.

Combining (3.5)–(3.7) in a single forward camera model, the image $\boldsymbol{p}^i = (x, y)^T$ of the 3D point $\boldsymbol{p}^c = (X, Y, Z)^T$ is given by

$$\boldsymbol{p}^i = \underbrace{(\mathcal{A} \circ \mathcal{D} \circ \mathcal{P}_n)}_{\triangleq \mathcal{P}}(\boldsymbol{p}^c), \tag{3.8}$$

where $\mathcal{P}$ is a composite function which consists of a normalized projection function $\mathcal{P}_n$, a distortion function $\mathcal{D}$ and an affine transformation $\mathcal{A}$. Here, $\mathcal{P}_n$ maps 3D points to normalized image coordinates as in (3.5b), $\mathcal{D}$ distorts the normalized image as in (3.6) and $\mathcal{A}$ transforms the distorted image to pixel coordinates, as in (3.7). That is, $\boldsymbol{p}^i = \mathcal{A}(\boldsymbol{p}_d^i)$, $\boldsymbol{p}_d^i = \mathcal{D}(\boldsymbol{p}_n^i)$ and $\boldsymbol{p}_n^i = \mathcal{P}_n(\boldsymbol{p}^c)$. The discussed relations hold only for a perspective camera, but the model structure is also applicable to omni-directional cameras and fish-eye lenses (Kannala and Brandt, 2006).

In general, 3D points will not be expressed in the camera coordinate frame as $\boldsymbol{p}^c$ since this coordinate frame is moving. Instead, they are expressed in the fixed earth coordinate system as $\boldsymbol{p}^e$. These two coordinate frames are related by a rotation R^{ce} and a translation $\boldsymbol{c}^e$, which can be used to obtain $\boldsymbol{p}^c$ from $\boldsymbol{p}^e$,

$$\boldsymbol{p}^c = R^{ce}(\boldsymbol{p}^e - \boldsymbol{c}^e). \tag{3.9}$$

The parameters R^{ce} and $\boldsymbol{c}^e$ parameterize the position and orientation of the camera and are called the extrinsic parameters. In contrast, the parameters involved in (3.8) are the so-called intrinsic or internal parameters.

The most direct way to obtain a measurement model for a 3D point $\boldsymbol{p}^e$ and its corresponding 2D image point $\boldsymbol{p}^i$ is the combination of (3.8) and (3.9). However, this would yield an unnecessary complex equation. Instead, it is advantageous to apply preprocessing and work with normalized image coordinates $\boldsymbol{p}_n^i = (\mathcal{D}^{-1} \circ \mathcal{A}^{-1})(\boldsymbol{p}^i)$. Then, the measurement model is based on the normalized camera projection $\mathcal{P}_n$, which in case of a perspective projection can be written in a particularly simple linear form,

$$\boldsymbol{y}_{c,k} = \begin{bmatrix} -I_2 & \boldsymbol{p}_{n,k}^i \end{bmatrix} R^{ce}(\boldsymbol{p}_k^e - \boldsymbol{c}^e) + \boldsymbol{e}_{c,k}. \tag{3.10}$$

Here $\boldsymbol{y}_{c,k}$ is a measurement constructed from the k-th measured 2D/3D correspondence $\boldsymbol{p}_k^i \leftrightarrow \boldsymbol{p}_k^e$, $\boldsymbol{c}^e$ is the position of the camera in the earth coordinate frame, R^{ce} is the rotation matrix which gives the orientation of the camera coordinate system w.r.t. the earth coordinate frame and $\boldsymbol{e}_{c,k}$ is white noise. Note that the prediction $\hat{\boldsymbol{y}}_{c,k,t|t-1} = 0$.

3.2.2 Calibration

The goal of a camera calibration procedure is to find the intrinsic parameters of the camera. These are the parameters involved in (3.8) and include the camera calibration matrix and the distortion parameters. Camera calibration is a well-known problem in computer vision and several approaches and accompanying toolboxes are available, see e.g., Bouguet (2003); Zhang (2000); Kannala and Brandt (2006). These procedures typically require images at several angles and distances of a known calibration object. A planar checkerboard pattern is a frequently used calibration object because it is very simple to produce, it can be printed with a standard printer, and has distinctive corners which are easy to detect. An example image involving such a pattern is shown in Figure 3.10. From the images of the calibration pattern 2D/3D correspondences $\boldsymbol{p}_k^i \leftrightarrow \boldsymbol{p}_k^e$ are constructed. In general this is a difficult problem, but exploiting the simple structure of the calibration pattern it is a relatively simple task.

The calibration problem is to choose the intrinsic parameters such that the obtained correspondences are as likely as possible. This cannot be done without determining the extrinsic parameters of the calibration images as well. Under the standard assumptions of i.i.d. Gaussian noise, this maximum likelihood optimization can be formulated as

$$\hat{\theta} = \arg\min_{\theta} \sum_{k=1}^{N} \frac{1}{2} \|\boldsymbol{p}_k^i - \mathcal{P}(\boldsymbol{p}_k^e, \theta)\|^2, \tag{3.11}$$

where the parameter vector θ consists of the intrinsic and extrinsic parameters. This is a nonlinear least squares problem which can be solved using standard algorithms (Nocedal

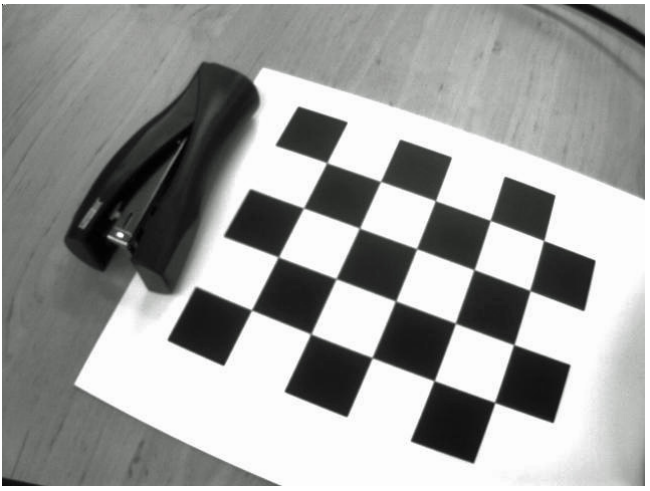


Figure 3.10: *An example of an image used for camera calibration. The calibration object is a planar checkerboard pattern.*

and Wright, 2006). These algorithms require an initial guess of the parameters which can be found from the homographies, the one-to-one relations that exist between the images and the planar calibration pattern (Zhang, 2000).

3.2.3 Correspondence detection

Computer vision techniques can be used to determine the position and orientation of the camera from the images it takes. The key ingredients for doing so are the 2D/3D correspondences, the corner stones in many computer vision applications. To obtain these correspondences typically two tasks have to be solved, which are extensively studied in literature:

Feature detection. The first task consists of detecting points of interest or features in the image. Here, features are distinctive elements of the camera image, for instance, corners, edges, or textured areas. Common algorithms include the gradient based Harris detector and the Laplace detector (Harris and Stephens, 1988; Mikolajczyk et al., 2005), and the correlation based Kanade-Lucas-Tomasi tracker (Shi and Tomasi, 1994).

Data association. Once a feature has been found, it needs to be associated to a 3D point to form a correspondence. This is the second task, which can be solved using probabilistic methods such as RANSAC (Fischler and Bolles, 1981). However, it can be drastically simplified by making use of some kind of descriptor of the feature which uniquely identifies it by providing information of the local image such as image patches or local histograms. This descriptor should preferably be invariant to scale changes and affine transformations. Common examples are SIFT (Lowe, 2004) and SURF (Bay et al., 2008). Other detectors as well as performance overviews are given in Mikolajczyk and Schmid (2005); Mikolajczyk et al. (2005).

Once three or more correspondences have been obtained in a single image, they can be used to calculate the position and orientation of the camera, see e.g., Hartley and Zisserman (2004); Ma et al. (2006). This is actually a reduced version of the camera calibration problem of Section 3.2.2, where in this case only the extrinsic parameters are sought as the intrinsic parameters are already known. Minimizing the prediction errors of the correspondences $\boldsymbol{p}_k^i - \mathcal{P}(\boldsymbol{p}_k^e, \theta)$ using nonlinear least squares yields the camera pose.

Correspondences can also be used to find the 3D position of a feature. In the simplest case this can be done using the epipolar geometry of a correspondence which is observed in two images taken from different locations. Extensions to multiple images exist as well (Hartley and Zisserman, 2004). These are the basis for structure from motion algorithms in which a model of the environment is computed from a sequence of images, see for instance Bartczak et al. (2007).

Implementation

In the setting of the MATRIS project, the basic assumption is that a textured 3D model of the tracking environment is available. Such a model can be obtained from e.g., CAD drawings or from structure from motion algorithms. Given a reasonably accurate prediction of the camera pose, e.g., from inertial navigation, an artificial image can be obtained by projecting the 3D model. This artificial image will resemble the camera image and is used to construct 2D search templates which are matched against the camera image, see Figure 3.11. For a successful match the association problem is already solved and a correspondence is obtained directly.

Figure 3.11: *Correspondences are generated by comparing the 3D scene model viewed from the predicted camera pose (left) to the camera image (right).*

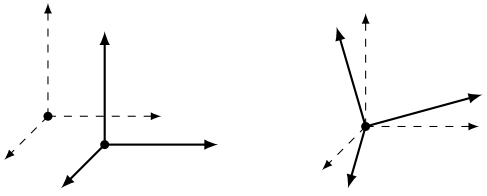
4

State space models

Chapter 2 introduced a sensor fusion algorithm for real-time camera pose estimation. The key components of this algorithm are the process and measurement models. In the upcoming sections these models are derived from the equations in Chapter 3 in combination with kinematics.

4.1 Kinematics

Kinematics deals with aspects of motion in absence of considerations of mass and force. It assigns coordinate frames to a rigid body and describes how these move over time. A general, length preserving transformation between two Cartesian coordinate frames consists of a translation and/or a rotation. Both transformations are illustrated in Figure 4.1. A translation is defined as a displacement of the origin, while keeping the axes aligned,



(a) *Translation.* ***(b)*** *Rotation.*

Figure 4.1: *The elementary transformations.*

whereas a rotation is a change in axes, while keeping the origins coincident. These transformations and their properties are the topic of the following sections.

4.1.1 Translation

A translation of a coordinate frame corresponds to a displacement of its origin and is parameterized using a displacement vector. In the translated frame a point x has a new coordinate vector, see Figure 4.2. Mathematically, expressing a point $\boldsymbol{x}$ resolved in the b

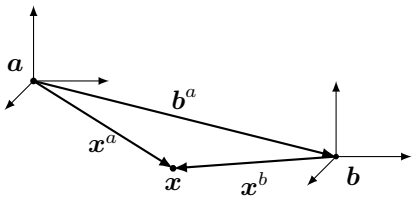


Figure 4.2: *The point $\boldsymbol{x}$ is expressed in the b frame and in the translated a frame.*

frame in the translated a frame is defined as

$$\boldsymbol{x}^a \triangleq \boldsymbol{x}^b + \boldsymbol{b}^a. \tag{4.1}$$

Here, $\boldsymbol{x}^a$ denotes the position of the point $\boldsymbol{x}$ w.r.t. the a frame whose origin is the point $\boldsymbol{a}$. Solving for $\boldsymbol{x}^b$ gives the inverse transformation,

$$\boldsymbol{x}^b = \boldsymbol{x}^a - \boldsymbol{b}^a \triangleq \boldsymbol{x}^a + \boldsymbol{a}^b. \tag{4.2}$$

Hence, $\boldsymbol{a}^b = -\boldsymbol{b}^a$.

4.1.2 Rotation

A rotation of a coordinate frame corresponds to changing direction of coordinate axis, while the origin remains where it is. Rotations can be described using a number of different parameterizations, see e.g., Shuster (1993) for an overview. Commonly encountered parameterizations include rotation matrices, Euler angles and unit quaternions.

A geometric interpretation of vector rotation is given in Figure 4.3. The rotation the

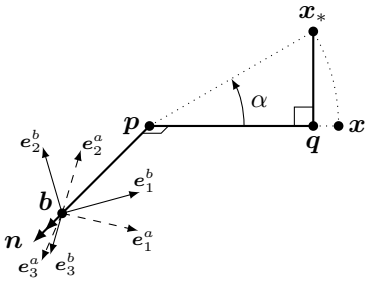


Figure 4.3: *The rotation of $\boldsymbol{x}$ around axis $\boldsymbol{n}$ with angle α.*

point $\boldsymbol{x}$ to $\boldsymbol{x}_*$, that is a rotation around the unit axis $\boldsymbol{n}$ by an angle α, can decomposed as

$$\begin{aligned}\boldsymbol{x}_*^b &= \boldsymbol{p}^b + (\boldsymbol{q}-\boldsymbol{p})^b + (\boldsymbol{x}_*-\boldsymbol{q})^b \\ &= (\boldsymbol{x}^b\cdot\boldsymbol{n}^b)\boldsymbol{n}^b + (\boldsymbol{x}^b - (\boldsymbol{x}^b\cdot\boldsymbol{n}^b)\boldsymbol{n}^b)\cos\alpha + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha \\ &= \boldsymbol{x}^b\cos\alpha + \boldsymbol{n}^b(\boldsymbol{x}^b\cdot\boldsymbol{n}^b)(1-\cos\alpha) + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha.\end{aligned}$$

Here all quantities are resolved in the b frame. Note that this 'clockwise' vector rotation corresponds to an 'anti-clockwise' rotation of the coordinate frame. Hence, expressing a point $\boldsymbol{x}$ resolved in the b frame in the rotated a coordinate frame is defined as

$$\boldsymbol{x}^a \triangleq \boldsymbol{x}^b\cos\alpha + \boldsymbol{n}^b(\boldsymbol{x}^b\cdot\boldsymbol{n}^b)(1-\cos\alpha) + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha. \tag{4.3}$$

This equation is commonly referred to as the rotation formula.

The cross product has the property,

$$\boldsymbol{u}\times\boldsymbol{v}\times\boldsymbol{w} = \boldsymbol{v}(\boldsymbol{w}\cdot\boldsymbol{u}) - \boldsymbol{w}(\boldsymbol{u}\cdot\boldsymbol{v}),$$

and can be written as a matrix-vector multiplication, $\boldsymbol{a}\times\boldsymbol{b} = S(\boldsymbol{a})\boldsymbol{b}$ with

$$S(\boldsymbol{a}) \triangleq \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{4.4}$$

Using these relations, (4.3) can be rewritten as

$$\begin{aligned}\boldsymbol{x}^a &= \boldsymbol{x}^b\cos\alpha + \boldsymbol{n}^b(\boldsymbol{x}^b\cdot\boldsymbol{n}^b)(1-\cos\alpha) + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha \\ &= \boldsymbol{x}^b\cos\alpha + (\boldsymbol{n}^b\times\boldsymbol{n}^b\times\boldsymbol{x}^b + \boldsymbol{x}^b)(1-\cos\alpha) + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha \\ &= \underbrace{\left[I + (\sin\alpha)S(\boldsymbol{n}^b) + (1-\cos\alpha)S^2(\boldsymbol{n}^b)\right]}_{\triangleq R^{ab}}\boldsymbol{x}^b.\end{aligned} \tag{4.5}$$

Hence, a rotation of the coordinate frame can be defined using a rotation matrix R,

$$\boldsymbol{x}^a \triangleq R^{ab}\boldsymbol{x}^b. \tag{4.6}$$

The rotation matrix R^{ab} is a member of the special orthogonal group, $\{R\in\mathbb{R}^{3\times3} : RR^T = I, \det R = +1\}$. Solving for $\boldsymbol{x}^b$ gives the inverse transformation

$$\boldsymbol{x}^b = (R^{ab})^T\boldsymbol{x}^a \triangleq R^{ba}\boldsymbol{x}^a. \tag{4.7}$$

It follows that $R^{ba} = (R^{ab})^T$.

Alternatively, a rotation of the coordinate frame can be defined using unit quaternions,

$$x^a \triangleq q^{ab}\odot x^b\odot(q^{ab})^c. \tag{4.8}$$

Here, $\{x^a, x^b\}\in\mathcal{Q}_v$ are the quaternion equivalents of $\{\boldsymbol{x}^a, \boldsymbol{x}^b\}$, $q^{ab}\in\mathcal{Q}_1$ is a unit quaternion describing the rotation from the b to the a coordinate frame and $\odot$, $\cdot^c$ denote

quaternion multiplication and conjugation respectively. Details on quaternions and their properties can be found in Appendix A. Solving for x^b gives the inverse transformation

$$x^b = (q^{ab})^c \odot x^a \odot q^{ab} \triangleq q^{ba} \odot x^a \odot (q^{ba})^c. \tag{4.9}$$

Hence, $q^{ba} = (q^{ab})^c$. The connection to (4.3) becomes clear by expanding the quaternion products in (4.8) and substituting $q^{ab} = (q_0, \boldsymbol{q}) = (\cos\frac{\alpha}{2}, \boldsymbol{n}^b \sin\frac{\alpha}{2})$:

$$\begin{aligned}(0, \boldsymbol{x}^a) &= (0, (q_0^2 - \boldsymbol{q}\cdot\boldsymbol{q})\boldsymbol{x}^b + 2(\boldsymbol{x}^b\cdot\boldsymbol{q})\boldsymbol{q} + 2q_0(\boldsymbol{x}^b \times \boldsymbol{q})) \\ &= (0, \boldsymbol{x}^b\cos\alpha + \boldsymbol{n}^b(\boldsymbol{x}^b\cdot\boldsymbol{n}^b)(1-\cos\alpha) + (\boldsymbol{n}^b\times\boldsymbol{x}^b)\sin\alpha).\end{aligned}$$

All the rotation parameterizations are similar and can be interchanged. However, they differ in the number of parameters, singularity, global representation and the difficulty of the differential equations. The reason for using unit quaternions is that they offer a nonsingular parameterization with a rather simple, bilinear differential equation which can be integrated analytically and have only four parameters. In contrast, Euler angles have only three parameters, but suffer from singularities and have a nonlinear differential equation. Furthermore, rotation matrices have at least 6 parameters.

Although quaternions are used for all the calculations, rotation matrices are occasionally used to simplify notation. Furthermore, Euler angles provide an intuitive representation which is used for visualizing a trajectory. All these parameterizations represent the same quantity and can be converted to each other. These conversions are given in Appendix B.

4.1.3 Time derivatives

Straightforward differentiation of (4.1) w.r.t. time gives the translational transformation for velocities

$$\dot{\boldsymbol{x}}^a = \dot{\boldsymbol{x}}^b + \dot{\boldsymbol{b}}^a. \tag{4.10}$$

The time derivative for rotation cannot be derived that easily. Since rotations can be decomposed in incremental rotations, see e.g., Shuster (1993), it holds that

$$q^{ab}(t+\delta t) = \delta q \odot q^{ab}(t) = \left(\cos\tfrac{\delta\theta}{2}, \boldsymbol{n}^a \sin\tfrac{\delta\theta}{2}\right) \odot q^{ab}(t),$$

for some rotation axis $\boldsymbol{n}^a$ and angle $\delta\theta$. Then, the formal definition of differentiation yields

$$\begin{aligned}\dot{q}^{ab}(t) &= \lim_{\delta t\to 0} \frac{q^{ab}(t+\delta t) - q^{ab}(t)}{\delta t} = \lim_{\delta t\to 0} \frac{\delta q \odot q^{ab}(t) - q^{ab}(t)}{\delta t}, \\ &= \lim_{\delta t\to 0} \frac{\left(\cos\frac{\delta\theta}{2} - 1, \boldsymbol{n}^a \sin\frac{\delta\theta}{2}\right)}{\delta t} \odot q^{ab}(t) = \tfrac{1}{2}\omega_{ab}^a(t) \odot q^{ab}(t).\end{aligned} \tag{4.11}$$

Here, the instantaneous angular velocity ω_{ab}^a from b to a and resolved a is defined as

$$\omega_{ab}^a \triangleq \lim_{\delta t\to 0} 2\frac{\left(1-\cos\frac{\delta\theta}{2}, \boldsymbol{n}^a \sin\frac{\delta\theta}{2}\right)}{\delta t} = \left(0, \lim_{\delta t\to 0} \boldsymbol{n}^a \frac{\delta\theta}{\delta t}\right). \tag{4.12}$$

Note that the angular velocity is a vector and conforms to the standard definition. Furthermore, (4.11) can be written in several ways, for instance,

$$\begin{aligned}\dot{q}^{ba} &= (\dot{q}^{ab})^c = q^{ba} \odot (\tfrac{1}{2}\omega^a_{ab})^c = -q^{ba} \odot \tfrac{1}{2}\omega^a_{ab} = q^{ba} \odot \tfrac{1}{2}\omega^a_{ba} \\ &= q^{ba} \odot \tfrac{1}{2}\omega^a_{ba} \odot q^{ab} \odot q^{ba} = \tfrac{1}{2}\omega^b_{ba} \odot q^{ba}\end{aligned}$$

This implies that the angular velocity vector can be transformed according to

$$\omega^a_{ab} = -\omega^a_{ba}, \tag{4.13a}$$

$$\omega^a_{ab} = q^{ab} \odot \omega^b_{ab} \odot q^{ba}. \tag{4.13b}$$

Using the above results (4.8) can be differentiated to obtain

$$\begin{aligned}\dot{x}^a &= \dot{q}^{ab} \odot x^b \odot q^{ba} + q^{ab} \odot x^b \odot \dot{q}^{ba} + q^{ab} \odot \dot{x}^b \odot q^{ba} \\ &= \tfrac{1}{2}\omega^a_{ab} \odot q^{ab} \odot x^b \odot q^{ba} - q^{ab} \odot x^b \odot q^{ba} \odot \tfrac{1}{2}\omega^a_{ab} + q^{ab} \odot \dot{x}^b \odot q^{ba} \\ &= \omega^a_{ab} \otimes x^a + q^{ab} \odot \dot{x}^b \odot q^{ba},\end{aligned} \tag{4.14}$$

where $\otimes$ is the quaternion cross product, see Appendix A.1. Note that (4.14) is equivalent to the perhaps more commonly used notation

$$\dot{\boldsymbol{x}}^a = \boldsymbol{\omega}^a_{ab} \times \boldsymbol{x}^a + R^{ab}\dot{\boldsymbol{x}}^b.$$

4.2 Continuous-time models

In Chapter 2 sensor fusion is used to combine the inertial and vision measurements to obtain a real-time camera pose estimate. The key components are the process and measurement models. In deriving these, the choice of which state vector to use is essential. For this a number of issues are considered, including whether or not to treat the inertial measurements as control inputs and which coordinates (body or camera) to use.

The process model consists of a constant acceleration model and a constant angular velocity model, implying that acceleration and angular velocity are included in the state vector. This model is mainly motivated by the intended application of real-time AR, where accurate pose predictions are required to compensate for the processing lag. The alternative of considering the inertial measurements as control inputs reduces the state vector dimension, but lacks an easy possibility for prediction.

The IMU provides kinematic quantities measured in the body frame, whereas the vision measurements relates to the camera frame. These two frames are rigidly connected, i.e., $\boldsymbol{c}^b$ and q^{bc} are constant. Hence, the camera and sensor poses w.r.t. the earth frame are related to each other according to

$$\boldsymbol{c}^e(t) = \boldsymbol{b}^e(t) + R^{eb}(t)\boldsymbol{c}^b, \tag{4.15a}$$

$$q^{ce}(t) = q^{cb} \odot q^{be}(t). \tag{4.15b}$$

Differentiating these equations w.r.t. to time according to Section 4.1.3 yields

$$\dot{\boldsymbol{c}}^e(t) = \dot{\boldsymbol{b}}^e(t) + \boldsymbol{\omega}^e_{eb}(t) \times R^{eb}(t)\boldsymbol{c}^b, \tag{4.16a}$$

$$\ddot{\boldsymbol{c}}^e(t) = \ddot{\boldsymbol{b}}^e(t) + \dot{\boldsymbol{\omega}}^e_{eb}(t) \times R^{eb}(t)\boldsymbol{c}^b + \boldsymbol{\omega}^e_{eb}(t) \times \boldsymbol{\omega}^e_{eb}(t) \times R^{eb}(t)\boldsymbol{c}^b, \tag{4.16b}$$

as well as

$$\dot{q}^{ce}(t) = q^{cb} \odot \dot{q}^{be}(t). \tag{4.16c}$$

This implies that a transition from body to camera or vice versa can occur on multiple locations in the process and measurement models resulting in slightly different state vectors. Some examples are given below.

Body based state vector. In this case the state vector contains position, velocity, acceleration, orientation and angular velocity of the body frame. This yields a relatively straightforward process model,

$$\tfrac{\partial}{\partial \mathrm{t}}\, \boldsymbol{b}^e = \dot{\boldsymbol{b}}^e, \tag{4.17a}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \dot{\boldsymbol{b}}^e = \ddot{\boldsymbol{b}}^e, \tag{4.17b}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \ddot{\boldsymbol{b}}^e = \boldsymbol{v}_a^e, \tag{4.17c}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, q^{be} = -\tfrac{1}{2}\boldsymbol{\omega}_{eb}^b \odot q^{be}, \tag{4.17d}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \boldsymbol{\omega}_{eb}^b = \boldsymbol{v}_\omega^b, \tag{4.17e}$$

where the time-dependence has been suppressed. It is driven by the process noises $\boldsymbol{v}_a^e$ and $\boldsymbol{v}_\omega^b$. The state vector implies that the measurement model is given by

$$\boldsymbol{y}_a = R^{be}(\ddot{\boldsymbol{b}}^e - \boldsymbol{g}^e) + \boldsymbol{\delta}_a^b + \boldsymbol{e}_a^b, \tag{4.17f}$$
$$\boldsymbol{y}_\omega = \boldsymbol{\omega}_{eb}^b + \boldsymbol{\delta}_\omega^b + \boldsymbol{e}_\omega^b, \tag{4.17g}$$
$$\boldsymbol{y}_{c,k} = \begin{bmatrix} -I_2 & \boldsymbol{p}_{n,k}^i \end{bmatrix} R^{cb}(R^{be}(\boldsymbol{p}_k^e - \boldsymbol{b}^e) - \boldsymbol{c}^b) + \boldsymbol{e}_{c,k}. \tag{4.17h}$$

The inertial measurement models (3.2) and (3.3) are used directly. The correspondence measurement model (3.10) is adapted using (4.15) to incorporate the transition from the body frame to the camera frame.

Camera based state vector. Here, the state vector contains position, velocity, acceleration, orientation and angular velocity of the camera frame. This results in

$$\tfrac{\partial}{\partial \mathrm{t}}\, \boldsymbol{c}^e = \dot{\boldsymbol{c}}^e, \tag{4.18a}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \dot{\boldsymbol{c}}^e = \ddot{\boldsymbol{c}}^e, \tag{4.18b}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \ddot{\boldsymbol{c}}^e = \boldsymbol{v}_a^e, \tag{4.18c}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, q^{ce} = -\tfrac{1}{2}\boldsymbol{\omega}_{ec}^c \odot q^{ce}, \tag{4.18d}$$
$$\tfrac{\partial}{\partial \mathrm{t}}\, \boldsymbol{\omega}_{ec}^c = \boldsymbol{v}_\omega^c, \tag{4.18e}$$

and

$$\boldsymbol{y}_a = R^{bc}(R^{ce}(\ddot{\boldsymbol{c}}^e - \boldsymbol{g}^e) + \dot{\boldsymbol{\omega}}_{ec}^c \times \boldsymbol{b}^c + \boldsymbol{\omega}_{ec}^c \times \boldsymbol{\omega}_{ec}^c \times \boldsymbol{b}^c) + \boldsymbol{\delta}_a^b + \boldsymbol{e}_a^b, \tag{4.18f}$$
$$\boldsymbol{y}_\omega = R^{bc}\boldsymbol{\omega}_{ec}^c + \boldsymbol{\delta}_\omega^b + \boldsymbol{e}_\omega^b, \tag{4.18g}$$
$$\boldsymbol{y}_{c,k} = \begin{bmatrix} -I_2 & \boldsymbol{p}_{n,k}^i \end{bmatrix} R^{ce}(\boldsymbol{p}_k^e - \boldsymbol{c}^e) + \boldsymbol{e}_{c,k}. \tag{4.18h}$$

Note that the process model is very similar to (4.17). However, using the camera based state vector the correspondence measurement model (3.10) remains unmodified and the inertial measurement models (3.2) and (3.3) are adapted using (4.16) to account for the transition from the camera frame to the body frame. This results in additional nonlinear terms and requires the introduction of the angular acceleration $\dot{\boldsymbol{\omega}}^c_{ec}$ in the state.

Mixed state vector. Using a mixed state vector, i.e., the state contains position and orientation of the camera frame and velocity, acceleration and angular velocity of the body frame, the process model is given by

$$\frac{\partial}{\partial \mathrm{t}} \boldsymbol{c}^e = \dot{\boldsymbol{b}}^e + R^{ec} R^{cb} (\boldsymbol{\omega}^b_{eb} \times \boldsymbol{c}^b), \tag{4.19a}$$
$$\frac{\partial}{\partial \mathrm{t}} \dot{\boldsymbol{b}}^e = \ddot{\boldsymbol{b}}^e, \tag{4.19b}$$
$$\frac{\partial}{\partial \mathrm{t}} \ddot{\boldsymbol{b}}^e = \boldsymbol{v}^e_a, \tag{4.19c}$$
$$\frac{\partial}{\partial \mathrm{t}} q^{ce} = -\tfrac{1}{2}(R^{cb}\boldsymbol{\omega}^b_{eb}) \odot q^{ce}, \tag{4.19d}$$
$$\frac{\partial}{\partial \mathrm{t}} \boldsymbol{\omega}^b_{eb} = \boldsymbol{v}^b_\omega. \tag{4.19e}$$

Note that this process model contains the transition from the body frame to the camera frame (4.16), resulting in additional coupling between the states. The associated measurement model is

$$\boldsymbol{y}_a = R^{bc} R^{ce} (\ddot{\boldsymbol{b}}^e - \boldsymbol{g}^e) + \boldsymbol{\delta}^b_a + \boldsymbol{e}^b_a, \tag{4.19f}$$
$$\boldsymbol{y}_\omega = \boldsymbol{\omega}^b_{eb} + \boldsymbol{\delta}^b_\omega + \boldsymbol{e}^b_\omega, \tag{4.19g}$$
$$\boldsymbol{y}_{c,k} = \begin{bmatrix} -I_2 & \boldsymbol{p}^i_{n,k} \end{bmatrix} R^{ce} (\boldsymbol{p}^e_k - \boldsymbol{c}^e) + \boldsymbol{e}_{c,k}. \tag{4.19h}$$

That is, the measurement models (3.2), (3.3) and (3.10) are unmodified.

In view of the framework for nonlinear estimation introduced in Chapter 2, linear models are favorable. Compared to (4.17), both (4.18) and (4.19) introduce nonlinear terms at the high sampling rate of the IMU. Hence the body based state vector (4.17) has been developed further, resulting in the discrete time models discussed in the upcoming section.

4.3 Discrete-time models

Integrating (4.17) — extended with the gyroscope and acceleration biases $\boldsymbol{\delta}^b_{\omega,t}$ and $\boldsymbol{\delta}^b_{a,t}$ discussed in Section 3.1.1 — w.r.t. time results in the discrete-time model of Chapter 2. For convenience it is repeated here.

$$\boldsymbol{b}^e_{t+1} = \boldsymbol{b}^e_t + T\dot{\boldsymbol{b}}^e_t + \tfrac{T^2}{2}\ddot{\boldsymbol{b}}^e_t, \tag{4.20a}$$
$$\dot{\boldsymbol{b}}^e_{t+1} = \dot{\boldsymbol{b}}^e_t + T\ddot{\boldsymbol{b}}^e_t, \tag{4.20b}$$
$$\ddot{\boldsymbol{b}}^e_{t+1} = \ddot{\boldsymbol{b}}^e_t + \boldsymbol{v}^e_{\ddot{b},t}, \tag{4.20c}$$
$$q^{be}_{t+1} = \exp(-\tfrac{T}{2}\boldsymbol{\omega}^b_{eb,t}) \odot q^{be}_t, \tag{4.20d}$$
$$\boldsymbol{\omega}^b_{eb,t+1} = \boldsymbol{\omega}^b_{eb,t} + \boldsymbol{v}^b_{\omega,t}, \tag{4.20e}$$

$$\boldsymbol{\delta}^b_{\omega,t+1} = \boldsymbol{\delta}^b_{\omega,t} + \boldsymbol{v}^b_{\delta_\omega,t}, \tag{4.20f}$$
$$\boldsymbol{\delta}^b_{a,t+1} = \boldsymbol{\delta}^b_{a,t} + \boldsymbol{v}^b_{\delta_a,t}, \tag{4.20g}$$

where T is the integration interval and $\exp$ denotes the quaternion exponential defined in Appendix A.2. Furthermore, the Jacobian of (4.20) w.r.t. to the state vector is given by

$$F = \begin{bmatrix} I_3 & TI_3 & \frac{T^2}{2}I_3 & 0 & 0 & 0 & 0 \\ 0 & I_3 & TI_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \exp(-\frac{T}{2}\boldsymbol{\omega}^b_{eb})_L & F_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I_3 \end{bmatrix}. \tag{4.21}$$

Here, the quaternion operator $\cdot_L$ ($\cdot_R$ similarly) is defined in Appendix A.3 and

$$\begin{aligned} F_1 &\triangleq \frac{\partial}{\partial \boldsymbol{\omega}^b_{eb}} \left(\exp(-\tfrac{T}{2}\boldsymbol{\omega}^b_{eb}) \odot q^{be}\right) \\ &= -\tfrac{T}{2}(q^{be})_R \begin{bmatrix} -\frac{\boldsymbol{v}^T}{\|\boldsymbol{v}\|}\sin\|\boldsymbol{v}\| \\ \frac{1}{\|\boldsymbol{v}\|}\left[I - \frac{\boldsymbol{v}\boldsymbol{v}^T}{\|\boldsymbol{v}\|^2}\right]\sin\|\boldsymbol{v}\| + \frac{\boldsymbol{v}\boldsymbol{v}^T}{\|\boldsymbol{v}\|^2}\cos\|\boldsymbol{v}\| \end{bmatrix}_{\boldsymbol{v}=-\frac{T}{2}\boldsymbol{\omega}^b_{eb}}. \end{aligned}$$

Using the small angle approximation, i.e., $\cos x = 1$ and $\sin x = x$, this expression can be simplified to

$$F_1 \approx -\tfrac{T}{2}(q^{be})_R \begin{bmatrix} \frac{T}{2}\boldsymbol{\omega}^{b,T}_{eb} \\ I_3 \end{bmatrix}.$$

This approximation is valid when the angular acceleration signal is sampled fast enough, for instance in case of human motion and sample rates of 100 Hz and above.

The measurement models of (4.17) remain unchanged, but are mentioned again for completeness,

$$\boldsymbol{y}_a = R^{be}(\ddot{\boldsymbol{b}}^e - \boldsymbol{g}^e) + \boldsymbol{\delta}^b_a + \boldsymbol{e}^b_a, \tag{4.22a}$$
$$\boldsymbol{y}_\omega = \boldsymbol{\omega}^b_{eb} + \boldsymbol{\delta}^b_\omega + \boldsymbol{e}^b_\omega, \tag{4.22b}$$
$$\boldsymbol{y}_{c,k} = \begin{bmatrix} -I_2 & \boldsymbol{p}^i_{n,k} \end{bmatrix} R^{cb}(R^{be}(\boldsymbol{p}^e_k - \boldsymbol{b}^e) - \boldsymbol{c}^b) + \boldsymbol{e}_{c,k}. \tag{4.22c}$$

The Jacobian of (4.22) w.r.t. to the state vector is given by

$$H = \begin{bmatrix} 0 & 0 & R^{be} & H_2(\ddot{\boldsymbol{b}}^e - \boldsymbol{g}^e) & 0 & I_3 & 0 \\ 0 & 0 & 0 & 0 & I_3 & 0 & I_3 \\ H_1 R^{be} & 0 & 0 & H_1 H_2(\boldsymbol{p}^e_k - \boldsymbol{b}^e) & 0 & 0 & 0 \end{bmatrix}, \tag{4.23}$$

where

$$H_1 \triangleq \begin{bmatrix} -I_2 & \boldsymbol{p}^i_n \end{bmatrix} R^{cb},$$
$$H_2(\boldsymbol{v}^e) \triangleq \frac{\partial}{\partial q^{be}}(q^{be} \odot \boldsymbol{v}^e \odot q^{eb}) = (q^{be})^T_R(\boldsymbol{v}^e)_R + (q^{be})_L(\boldsymbol{v}^e)_L \begin{bmatrix} 1 & 0 \\ 0 & -I_3 \end{bmatrix}.$$

The discussed process and measurement models and their Jacobians are used in an EKF to fuse the visual and inertial measurements using Algorithm 2.1. Note that these models depend on the relative pose between the camera and the IMU, $\boldsymbol{c}^b, q^{cb}$, for which accurate calibration values need to be found.

5

Calibration theory

Calibration refers to the process of determining the output relation of a measurement device as well as applying suitable correction factors to it in order to obtain desired behavior. Examples of calibration procedures are IMU calibration and camera calibration, see Section 3.1.2 and Section 3.2.2. Camera calibration deals with estimation of the intrinsic and extrinsic parameters explaining the observed projection. With IMU calibration the raw sensor readings are converted into relevant physical units, while correcting for several undesired effects such as non-orthogonality and temperature dependence.

A calibration method proposes an appropriate model structure to describe the measurements and determines the model parameters which give the best match between the prediction and the measured output. This can be formulated as a nonlinear least squares minimization problem,

$$\hat{\theta} = \arg\min_{\theta} V(\theta) \tag{5.1a}$$

$$V(\theta) = \sum_{t=1}^{N} \|e_t\|^2 = \sum_{t=1}^{N} \|y_t - \hat{y}(\theta, u_t)\|^2, \tag{5.1b}$$

where V is the cost function, e_t are the prediction errors, y_t are the measurements, u_t the inputs, θ are the model parameters to be estimated and $\hat{y}(\,\cdot\,,\,\cdot\,)$ is the predicted measurement according to the model. This formulation is a special case of the prediction error framework (Ljung, 1999) used in system identification. Depending on the structure of the problem and the available measurements, various calibration methods can be designed.

The sensor unit consists of an IMU and a camera. Obviously, these sensors have to be calibrated individually. However, as stressed in Chapter 2 and Section 4.3, calibration of the relative position and orientation between the sensors is essential for proper operation of the present application. A similar type of problem occurs in assessing the performance of Algorithm 2.1 as discussed in Chapter 2. There the estimated pose is compared to an external reference, in our case an industrial robot. Here, the positions and orientations

between the involved coordinate systems have to be taken into account as well. Hence, the relative pose calibration problem has to be solved for several coordinate frames, see Figure 5.1. Details about the coordinate frames can be found in Chapter 2.

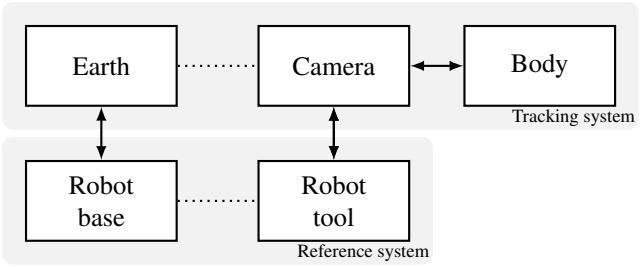


Figure 5.1: *Relations between the coordinate frames.* Solid lines *stand for rigid connections which have to be calibrated,* dotted lines *for non-rigid, time varying relations.*

The topic of this chapter is to provide a theoretical background on the various approaches which can be used to calibrate the relative position and orientation of the rigid connections. The starting point for relative pose calibration are the geometric relations between three coordinate frames, denoted a, b and c, see (4.15)

$$q^{ca} = q^{cb} \odot q^{ba}, \tag{5.2a}$$

$$\boldsymbol{c}^a = \boldsymbol{b}^a + R^{ab}\boldsymbol{c}^b. \tag{5.2b}$$

These equations can be used in various ways, depending on which measurements are available. Several possibilities will be discussed in the upcoming sections.

5.1 Kinematic relations

An IMU typically provides measurements of acceleration and angular velocity. This section deals with the problem of finding the relative position and orientation from these kinematic quantities.

5.1.1 Acceleration

A rigid connection between the b and the c system implies that q^{cb} and $\boldsymbol{c}^b$ are constants. Hence, (5.2b) reduces to

$$\boldsymbol{c}^a(t) = \boldsymbol{b}^a(t) + R^{ab}(t)\boldsymbol{c}^b.$$

Differentiating twice w.r.t. time and applying the transformation rules for angular velocity and angular acceleration (4.13a) yields the following relation between the accelerations $\ddot{\boldsymbol{c}}^a(t)$ and $\ddot{\boldsymbol{b}}^a(t)$,

$$\begin{aligned}\ddot{\boldsymbol{c}}^a(t) &= \ddot{\boldsymbol{b}}^a(t) + \dot{\boldsymbol{\omega}}^a_{ab}(t) \times R^{ab}(t)\boldsymbol{c}^b + \boldsymbol{\omega}^a_{ab}(t) \times \boldsymbol{\omega}^a_{ab}(t) \times R^{ab}(t)\boldsymbol{c}^b \\ &= \ddot{\boldsymbol{b}}^a(t) + R^{ab}(t)(\dot{\boldsymbol{\omega}}^b_{ab}(t) \times \boldsymbol{c}^b + \boldsymbol{\omega}^b_{ab}(t) \times \boldsymbol{\omega}^b_{ab}(t) \times \boldsymbol{c}^b).\end{aligned}$$

This expression can be manipulated according to

$$\begin{aligned} R^{ca}(t)(\ddot{\boldsymbol{c}}^a(t) - \ddot{\boldsymbol{b}}^a(t)) &= R^{cb}(\dot{\boldsymbol{\omega}}^b_{ab}(t) \times \boldsymbol{c}^b + \boldsymbol{\omega}^b_{ab}(t) \times \boldsymbol{\omega}^b_{ab}(t) \times \boldsymbol{c}^b) \\ &= R^{cb}[S(\dot{\boldsymbol{\omega}}^b_{ab}(t)) + S(\boldsymbol{\omega}^b_{ab}(t))^2]\boldsymbol{c}^b, \end{aligned} \tag{5.3}$$

where S has been defined in (4.4). This equation is linear in $\boldsymbol{c}^b$ and can be written as $y_t = H_t x$. Combining several time instants, a large system of linear equations can be constructed from which $\boldsymbol{c}^b$ can be solved using the following theorem.

Theorem 5.1 (Generalized least squares)
Suppose $\{y_t\}_{i=1}^N$ are measurements satisfying $y_t = H_t x$. Then the sum of the squared residuals, weighted according to Σ_t^{-1},

$$V(x) = \sum_{t=1}^{N} \|e_t\|^2_{\Sigma_t} = \sum_{t=1}^{N} (y_t - H_t x)^T \Sigma_t^{-1} (y_t - H_t x), \tag{5.4}$$

equivalently formulated using stacked matrices,

$$V(x) = \|e\|^2_{\Sigma} = (y - Hx)^T \Sigma^{-1} (y - Hx), \tag{5.5}$$

is minimized by

$$\hat{x} = (H^T \Sigma^{-1} H)^{-1} H^T \Sigma^{-1} y. \tag{5.6}$$

Proof: At its minimum, the gradient of V is zero,

$$\frac{\mathrm{d}\, V}{\mathrm{d}\, x} = -2H^T \Sigma^{-1}(y - Hx) = 0.$$

Solving for x yields (5.6), see e.g., Nocedal and Wright (2006). □

Note that methods based on the QR-factorization or the singular value decomposition (SVD) are numerically superior to the analytical expression in Theorem 5.1 and should be used to perform the actual computation.

Theorem 5.1 defines a mapping f from the measurements y to the estimate $\hat{x}$, i.e., $\hat{x} = f(y)$ according to (5.6). Introducing noise e in the model, that is, $y = Hx + e$, where e has zero mean and covariance $\sigma^2\Sigma$, the covariance of the estimate can be approximated as

$$\operatorname{Cov} \hat{x} \approx [\mathrm{D}_y\, \hat{x}] \operatorname{Cov} y [\mathrm{D}_y\, \hat{x}]^T, \tag{5.7}$$

where $\mathrm{D}_y\, \hat{x}$ is the Jacobian of the estimator $\hat{x}$ w.r.t. the measurements y. This expression is known as Gauss' approximation formula (Ljung, 1999) and is exact for linear estimators. Application to Theorem 5.1 yields

$$\operatorname{Cov} \hat{x} = \sigma^2 (H^T \Sigma^{-1} H)^{-1}, \tag{5.8}$$

and Theorem 5.1 returns the best linear unbiased estimate.

Assuming Gaussian noise, i.e., $y = Hx + e$ with $e \sim \mathcal{N}(0, \sigma^2\Sigma)$, the probability density function (PDF) of the measurements y is given by

$$p(y) = \frac{1}{(2\pi)^{\frac{n_e}{2}}\sigma^{n_e}\sqrt{\det\Sigma}} e^{-\frac{1}{2\sigma^2}(y-Hx)^T\Sigma^{-1}(y-Hx)}, \tag{5.9}$$

where n_e is the dimension of e. Maximizing $p(y)$, or equivalently $\log p(y)$, w.r.t. x results in (5.6). That is, the result of Theorem 5.1 can be interpreted as the maximum likelihood (ML) estimate. Furthermore, maximizing $\log p(y)$ w.r.t. σ^2 yields the following ML estimate for the covariance scale factor,

$$\hat{\sigma}^2 = \frac{V(\hat{x})}{n_e}. \tag{5.10}$$

Introducing the notation $P \triangleq H(H^T\Sigma^{-1}H)^{-1}H^T\Sigma^{-1}$, the expected value of V can be shown to be

$$\begin{aligned}
\mathrm{E}\, V(\hat{x}) &= \mathrm{E}(y - H\hat{x})^T\Sigma^{-1}(y - Hx) = \mathrm{E}\, e^T(I - P^T)\Sigma^{-1}(I - P)e \\
&= \mathrm{E}\, e^T\Sigma^{-1}(I - P)e = \mathrm{E}\,\mathrm{tr}[(I - P)\Sigma^{-1}ee^T] = \mathrm{tr}[(I - P)\Sigma^{-1}\,\mathrm{Cov}\, e] \\
&= \sigma^2\,\mathrm{tr}(I - P) = \sigma^2(n_e - n_x).
\end{aligned}$$

Hence, the ML estimate $\hat{\sigma}^2$ (5.10) is biased. Correcting for this bias results in

$$\hat{\sigma}^2 = \frac{V(\hat{x})}{n_e - n_x}. \tag{5.11}$$

5.1.2 Angular velocity

Switching focus from translation to orientation, the rigid connection b–c implies that (5.2a) reduces to

$$q^{ca}(t) = q^{cb} \odot q^{ba}(t).$$

Differentiating left and right hand sides w.r.t. time, see (4.12), yields

$$\tfrac{1}{2}\boldsymbol{\omega}^c_{ca}(t) \odot q^{ca}(t) = q^{cb} \odot \tfrac{1}{2}\boldsymbol{\omega}^b_{ba}(t) \odot q^{ba}(t).$$

Hence, the angular velocities for rigid connections are related by

$$\boldsymbol{\omega}^c_{ca}(t) = q^{cb} \odot \boldsymbol{\omega}^b_{ba}(t) \odot q^{bc}. \tag{5.12}$$

This expression can be used to solve for the relative orientation q^{cb}. A one-step solution is provided by Horn (1987). Theorem 5.2 contains a simplified proof and extends the original theorem with expressions for the Jacobians of the estimate w.r.t. the measurements, thus allowing analytic expressions for the covariance of estimate.

Before Theorem 5.2 is stated, some notation has to be introduced. In order to avoid ambiguities with partitioning of Jacobian matrices, they are defined as (Magnus and Neudecker, 1999)

$$\mathrm{D}_x f = \frac{\partial\,\mathrm{vec}\, f(x)}{\partial(\mathrm{vec}\, x)^T}, \tag{5.13}$$

where $\mathrm{vec}(\,\cdot\,)$ is the vectorization operator. With this definition, even matrix functions of matrix variables are well defined. As an alternative to evaluating each partial derivative in (5.13), the Jacobian $\mathrm{D}_x f$ can be identified from the (vectorized) differential of f:

$$\mathrm{d}\,\mathrm{vec}\, f(x) = A(x)\,\mathrm{d}\,\mathrm{vec}\, x \quad \Leftrightarrow \quad \mathrm{D}_x f = A(x). \tag{5.14}$$

Since computations with differentials are relatively easy, this is a rather useful and powerful approach.

Theorem 5.2 (Rotation A)
Suppose $\{\boldsymbol{v}_t^a\}_{t=1}^N$ and $\{\boldsymbol{v}_t^b\}_{t=1}^N$ are measurements satisfying $v_t^a = q^{ab} \odot v_t^b \odot q^{ba}$. Then the sum of the squared residuals,

$$V(q^{ab}) = \sum_{t=1}^{N} \|e_t\|^2 = \sum_{t=1}^{N} \|v_t^a - q^{ab} \odot v_t^b \odot q^{ba}\|^2, \tag{5.15}$$

is minimized by $\hat{q}^{ab} = x_1$, where x_1 is the eigenvector corresponding to the largest eigenvalue λ_1 of the system $Ax = \lambda x$ with

$$A = -\sum_{t=1}^{N} (v_t^a)_L (v_t^b)_R. \tag{5.16}$$

Furthermore, the Jacobians of $\hat{q}^{ab}$ w.r.t. the measurements are given by

$$\mathrm{D}_{v_t^a}\, \hat{q}^{ab} = -[(\hat{q}^{ab})^T \otimes (\lambda_1 I_4 - A)^\dagger][I_4 \otimes (v_i^b)_R][\mathrm{D}_v\, v_L], \tag{5.17a}$$

$$\mathrm{D}_{v_t^b}\, \hat{q}^{ab} = -[(\hat{q}^{ab})^T \otimes (\lambda_1 I_4 - A)^\dagger][I_4 \otimes (v_t^a)_L][\mathrm{D}_v\, v_R], \tag{5.17b}$$

where $\otimes$ is the Kronecker product and † is the Moore-Penrose pseudo inverse. The Jacobians $\mathrm{D}_v\, v_L$ and $\mathrm{D}_v\, v_R$ are defined as

$$\mathrm{D}_v\, v_L = \begin{bmatrix} e_R^0 \\ e_R^1 \\ e_R^2 \\ e_R^3 \end{bmatrix} \begin{bmatrix} 0 \\ I_3 \end{bmatrix}, \qquad \mathrm{D}_v\, v_R = \begin{bmatrix} e_L^0 \\ e_L^1 \\ e_L^2 \\ e_L^3 \end{bmatrix} \begin{bmatrix} 0 \\ I_3 \end{bmatrix},$$

where $\{e^i\}_{i=1}^4$ is the standard basis in $\mathbb{R}^4$.

Proof: The squared residuals can be written as

$$\|e_t\|^2 = \|v_t^a\|^2 - 2v_t^a \cdot (q^{ab} \odot v_t^b \odot q^{ba}) + \|v_t^b\|^2.$$

Minimization only affects the middle term, which can be simplified to

$$\begin{aligned} v_t^a \cdot (q^{ab} \odot v_t^b \odot q^{ba}) &= -(v_t^a \odot (q^{ab} \odot v_t^b \odot q^{ba}))_0 \\ &= -(v_t^a \odot q^{ab})^T (v_t^b \odot q^{ba})^c \\ &= -(q^{ab})^T (v_t^a)_L (v_t^b)_R q^{ab}, \end{aligned}$$

using the relation $(a \odot b)_0 = a^T b^c$ for the scalar part of quaternion multiplication. The minimization problem can now be restated as

$$\arg\min_{\|q^{ab}\|=1} \sum_{t=1}^{N} \|e_t\|^2 = \arg\max_{\|q^{ab}\|=1} (q^{ab})^T A q^{ab},$$

where A is defined in (5.16). Note that the matrices $\cdot_L$ and $\cdot_R$ commute, i.e., $a_L b_R = b_R a_L$, since $a_L b_R x = a \odot x \odot b = b_R a_L x$ for all x. Additionally, $\cdot_L$ and $\cdot_R$ are skew symmetric for vectors. This implies that

$$(v_t^a)_L (v_t^b)_R = [-(v_t^a)_L^T][-(v_t^b)_R^T] = [(v_t^b)_R (v_t^a)_L]^T = [(v_t^a)_L (v_t^b)_R]^T,$$

from which can be concluded that A is a real symmetric matrix.

Let $q^{ab} = X\alpha$ with $\|\alpha\| = 1$, where X is an orthonormal basis obtained from the symmetric eigenvalue decomposition of $A = X\Sigma X^T$. Then,

$$(q^{ab})^T A q^{ab} = \alpha^T X^T X \Sigma X^T X \alpha = \sum_{i=1}^{4} \alpha_i^2 \lambda_i \leq \lambda_1,$$

where λ_1 is the largest eigenvalue. Equality is obtained for $\alpha = (1, 0, 0, 0)^T$, that is, $\hat{q}^{ab} = x_1$.

The sensitivity of the solution can be determined based on an analysis of the real symmetric eigenvalue equation, $Ax = \lambda x$. The Jacobian of the eigenvector $x(A)$ is given by

$$\mathrm{D}_A\, x = x^T \otimes (\lambda_1 I_4 - A)^\dagger$$

as derived by Magnus and Neudecker (1999). Furthermore, writing $A_t = -R_t L_t = -L_t R_t$ and applying (5.14), yields

$$\begin{aligned} \mathrm{d}\, A_t(L_t) = -R_t(\mathrm{d}\, L_t) \quad &\Leftrightarrow \quad \mathrm{D}_{L_t}\, A_t = -I_4 \otimes R_t \\ \mathrm{d}\, A_t(R_t) = -L_t(\mathrm{d}\, R_t) \quad &\Leftrightarrow \quad \mathrm{D}_{R_t}\, A_t = -I_4 \otimes L_t \end{aligned}$$

Straightforward application of the chain rule results in

$$\mathrm{D}_{v_t^a}\, \hat{q}^{ab} = [\mathrm{D}_A\, x][\mathrm{D}_{L_t}\, A][\mathrm{D}_{v_t^a}\, L_t], \qquad \mathrm{D}_{v_t^b}\, \hat{q}^{ab} = [\mathrm{D}_A\, x][\mathrm{D}_{R_t}\, A][\mathrm{D}_{v_t^b}\, R_t].$$

Evaluating this expression gives (5.17). □

To incorporate measurement noise, the model is extended to

$$\boldsymbol{v}_t^a = \boldsymbol{v}_{t,0}^a + \boldsymbol{e}_t^a, \qquad \boldsymbol{v}_t^b = \boldsymbol{v}_{t,0}^b + \boldsymbol{e}_t^b, \qquad v_{t,0}^a = q^{ab} \odot v_{t,0}^b \odot q^{ba}, \tag{5.18}$$

where $\boldsymbol{e}_t^a$ and $\boldsymbol{e}_t^b$ are mutually independent, zero mean noises with covariance $\Sigma_{v_t^a}$ and $\Sigma_{v_t^b}$. Application of Gauss' approximation formula (5.7) yields the following covariance expression for the estimate produced by Theorem 5.2

$$\mathrm{Cov}\, \hat{q}^{ab} = \sum_{t=1}^{N} [\mathrm{D}_{v_t^a}\, \hat{q}^{ab}] \Sigma_{v_t^a} [\mathrm{D}_{v_t^a}\, \hat{q}^{ab}]^T + [\mathrm{D}_{v_t^b}\, \hat{q}^{ab}] \Sigma_{v_t^b} [\mathrm{D}_{v_t^b}\, \hat{q}^{ab}]^T. \tag{5.19}$$

Assuming independent identically distributed Gaussian noise, $\boldsymbol{e}_t^a, \boldsymbol{e}_t^b \sim \mathcal{N}(0, \sigma^2 I_3)$, the residuals $\boldsymbol{e}_t$ are distributed according to

$$\boldsymbol{e}_t \triangleq v_t^a - q^{ab} \odot v_t^b \odot q^{ba} = e_t^a - q^{ab} \odot e_t^b \odot q^{ba} \sim \mathcal{N}(0, 2\sigma^2 I_3), \tag{5.20}$$

and the PDF for all the residuals is given by

$$p(e) = \prod_{t=1}^{N} \frac{1}{(4\pi)^{\frac{3}{2}}\sigma^3} e^{-\frac{1}{4\sigma^2}(e_t^a - q^{ab}\odot e_t^b \odot q^{ba})^T (e_t^a - q^{ab}\odot e_t^b \odot q^{ba})}. \tag{5.21}$$

Maximizing $p(e)$, or equivalently $\log p(e)$, w.r.t. q^{ab} results in Theorem 5.2. That is, the result of Theorem 5.2 can be interpreted as the maximum likelihood (ML) estimate. Furthermore, maximizing $\log p(e)$ w.r.t. σ^2 yields the following ML estimate for the covariance scale factor,

$$\hat{\sigma}^2 = \frac{V(\hat{q}^{ab})}{6N}. \tag{5.22}$$

The expected value of $V(\hat{q}^{ab})$ is given by

$$\begin{aligned}
\mathrm{E}\, V(\hat{q}^{ab}) &= \mathrm{E} \sum_{t=1}^{N} (\boldsymbol{e}_t^a - \hat{R}^{ab}\boldsymbol{e}_t^b)^T (\boldsymbol{e}_t^a - \hat{R}^{ab}\boldsymbol{e}_t^b) \\
&= \mathrm{E} \sum_{t=1}^{N} \operatorname{tr} \boldsymbol{e}_t^a \boldsymbol{e}_t^{a,T} + \operatorname{tr} \boldsymbol{e}_t^b \boldsymbol{e}_t^{b,T} - 2(\boldsymbol{e}_t^a)^T \hat{R}^{ab} \boldsymbol{e}_t^b \approx (6N - 6)\sigma^2,
\end{aligned}$$

where a second order Taylor expansion of $\mathrm{E}(\boldsymbol{e}_t^a)^T \hat{R}^{ab} \boldsymbol{e}_t^b$ has been used. Hence, the ML estimate $\hat{\sigma}^2$ (5.22) is biased. Correcting for this bias results in

$$\hat{\sigma}^2 = \frac{V(\hat{x})}{6(N-1)}. \tag{5.23}$$

Validation

Theorem 5.2 and its associated covariance expressions have been validated using *Monte Carlo* (MC) simulations. The scenario defined by orientation $q^{ab} = 2^{-\frac{1}{2}}(1, 1, 0, 0)$ and measurements $\{\boldsymbol{v}_t^b\} = \{\boldsymbol{e}^1, \boldsymbol{e}^2, \boldsymbol{e}^3, -\boldsymbol{e}^1, -\boldsymbol{e}^2, -\boldsymbol{e}^3\}$ where $\{\boldsymbol{e}^i\}_{i=1}^3$ is the standard basis in $\mathbb{R}^3$ will be used as an example. Measurements are generated by adding Gaussian noise with $\Sigma = 10^{-4} I_3$ to $\{\boldsymbol{v}_t^a\}$ and $\{\boldsymbol{v}_t^b\}$.

From the measurements a set ($M = 10^4$) of orientation estimates $\{\hat{q}_k^{ab}\}_{k=1}^M$ and covariance estimates $\{\hat{Q}_k\}_{k=1}^M$ have been generated using Theorem 5.2 and (5.19). Figure 5.2 shows the distribution of the orientation error vectors $\boldsymbol{e}_k \triangleq 2\log(\hat{q}_k^{ab} q^{ba})$, where the quaternion logarithm is defined in Appendix A. The orientation errors have zero mean, implying that the orientation estimates are unbiased. Furthermore, the empirical distribution of the orientation errors is consistent with the theoretical distribution derived using the covariance estimates of (5.19). A comparison between the MC covariance

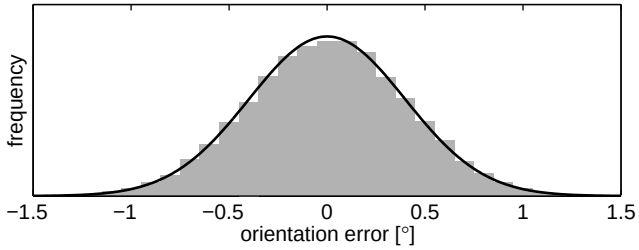


Figure 5.2: *Histogram of the orientation errors. Both the empirical distribution (*gray bar*) as well as the theoretical distribution (*black line*) are shown.*

$Q_{\text{MC}} = \operatorname{Cov} \hat{q}^{ab}$ and the theoretical covariance $Q_{\text{th}} = \operatorname{E} \hat{Q}$ shows also a very good match:

$$Q_{\text{MC}} = 10^{-5} \begin{pmatrix} 0.62 & -0.62 & -0.00 & -0.00 \\ -0.62 & 0.62 & 0.00 & 0.00 \\ -0.00 & 0.00 & 1.24 & -0.01 \\ -0.00 & 0.00 & -0.01 & 1.24 \end{pmatrix},$$

$$Q_{\text{th}} = 10^{-5} \begin{pmatrix} 0.62 & -0.62 & 0.00 & -0.00 \\ -0.62 & 0.62 & -0.00 & -0.00 \\ 0.00 & -0.00 & 1.25 & 0.00 \\ -0.00 & -0.00 & 0.00 & 1.25 \end{pmatrix}.$$

5.2 Geometric measurements

Besides measuring kinematic quantities, an IMU can also measure direction vectors such as gravity and the magnetic field. These vectors are geometric measurements, as well as direct position and orientation measurements from for instance an external reference system or computer vision. In this section it is discussed how relative position and orientation can be determined using these geometric measurements.

5.2.1 Direction vectors

Directional vectors evaluated in different coordinate frames are related by

$$\boldsymbol{v}_t^a = q^{ab} \odot \boldsymbol{v}_t^b \odot q^{ba}. \tag{5.24}$$

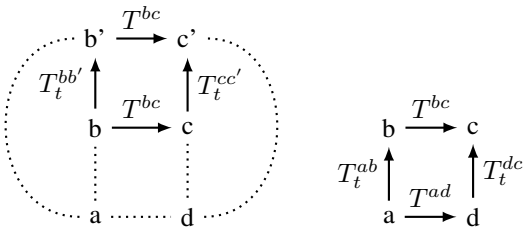
This is the same relation that holds for angular velocities (5.12). Hence, q^{ab} can be determined by direct application of Theorem 5.2.

5.2.2 Position and orientation

The transformations (5.2) can be written using homogeneous transformation matrices,

$$T^{ac} \triangleq \begin{bmatrix} R^{ac} & \boldsymbol{c}^a \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R^{ab}R^{bc} & \boldsymbol{b}^a + R^{ab}\boldsymbol{c}^b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R^{ab} & \boldsymbol{b}^a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R^{cb} & \boldsymbol{c}^b \\ 0 & 1 \end{bmatrix} = T^{ab}T^{bc}. \tag{5.25}$$

These transformation matrices are useful for evaluating paths between coordinate frames. Comparing multiple paths between two systems yields relations between the intermediate coordinate transformations. Figure 5.3 gives two examples of such relations. These



(a) Relative pose. **(b)** Absolute pose.

Figure 5.3: *Two examples of geometric relations between 4 coordinate systems. Multiple paths exist between $a-c$ and $b-c'$. The connections $a-d$ and $b-c$ are rigid, implying that T^{ad}, T^{bc} are constant. All other transformations vary between measurements, denoted with T_t.*

geometric relations have been studied extensively in the robotics literature and are there known as hand-eye calibration, see for instance Tsai and Lenz (1989); Strobl and Hirzinger (2006).

The classical hand-eye calibration scenario in robotics is to move a manipulator (hand) from b to b' and observe the position change of a sensor (eye) from c to c'. Figure 5.3a illustrates this scenario. Evaluating the two routes between b and c' yields the relation $T_t^{bc'} = T_t^{bb'}T^{bc} = T^{bc}T_t^{cc'}$, typically written as $AX = XB$, from which the unknown transformation T^{bc} can be solved given a number of relative poses $\{T_t^{bb'}, T_t^{cc'}\}_{t=1}^N$. The relative poses are usually not available since both the manipulator and the sensor give the tool - robot base transformation respectively the camera - world transformation. Hence, the relative poses are constructed from two absolute poses at different times, e.g., $T_t^{bb'} = T_{t_1}^{ba}T_{t_2}^{ab}$. These absolute poses are used directly in the slightly more general scenario given in Figure 5.3b. It yields the relation $T^{ac} = T_t^{ab}T^{bc} = T^{ad}T_t^{dc}$, typically written as $AX = ZB$, from which the unknown transformations T^{bc} and T^{ad} can be jointly solved for, given a number of poses $\{T_t^{ab}, T_t^{dc}\}_{t=1}^N$.

The relation of Figure 5.3b can be decomposed in its rotational and translational part,

$$T_t^{ab}T^{bc} = T^{ad}T_t^{dc} \Leftrightarrow \begin{cases} q_t^{ab}q^{bc} = q^{ad}q_t^{dc} \\ \boldsymbol{b}_t^a + R_t^{ab}\boldsymbol{c}^b = \boldsymbol{d}^a + R^{ad}\boldsymbol{c}_t^d \end{cases} \tag{5.26}$$

These parts are inherently coupled and many nonlinear optimization techniques have been proposed for solving it, see Strobl and Hirzinger (2006) for an overview. However, decoupling approaches are frequently employed with satisfactory results: assuming known rotations, the translational part is linear in the unknowns $\boldsymbol{c}^b, \boldsymbol{d}^a$ and can be solved using

$$\boldsymbol{b}_t^a - R^{ad}\boldsymbol{c}_t^d = \begin{bmatrix} R_t^{ab} & I \end{bmatrix} \begin{bmatrix} \boldsymbol{c}^b \\ \boldsymbol{d}^a \end{bmatrix} \tag{5.27}$$

in combination with Theorem 5.1. A one step solution to the rotational part is given by Theorem 5.3. It gives an explicit solution along with expressions for the Jacobians of the estimate.

Theorem 5.3 (Rotation B)
Suppose $\{q_t^{ab}\}_{t=1}^N$ and $\{q_t^{dc}\}_{t=1}^N$ are measurements satisfying $q_t^{ab} \odot q^{bc} = q^{ad} \odot q_t^{dc}$. Then the residual rotation error,

$$V(q^{ad}, q^{bc}) = \sum_{t=1}^{N} \|e_t\|^2 = \sum_{t=1}^{N} \|q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1\|^2, \tag{5.28}$$

is minimized by $\hat{q}^{ad} = v_1$ and $\hat{q}^{bc} = u_1$, the first right and left singular vectors of the matrix $A = U\Sigma V^T$, with

$$A = \sum_{t=1}^{N} (q_t^{ab})_L^T (q_t^{dc})_R. \tag{5.29}$$

Furthermore, the Jacobians of $\hat{q}^{dc}, \hat{q}^{ab}$ w.r.t. the measurements are given by

$$\mathrm{D}_{q_t^{ab}} \hat{x} = [\hat{x}^T \otimes (\sigma_1 I_{16} - B)^\dagger][\mathrm{D}_A\, B][K_{4,4}(I_4 \otimes (q_t^{dc})_R^T)][\mathrm{D}_q\, q_L] \tag{5.30a}$$

$$\mathrm{D}_{q_t^{dc}} \hat{x} = [\hat{x}^T \otimes (\sigma_1 I_{16} - B)^\dagger][\mathrm{D}_A\, B][I_4 \otimes (q_t^{ab})_L^T][\mathrm{D}_q\, q_R], \tag{5.30b}$$

where K, implicitly defined by $\operatorname{vec} A^T = K \operatorname{vec} A$, is the commutation matrix. Furthermore,

$$\hat{x} = \begin{pmatrix} \hat{q}^{ad} \\ \hat{q}^{bc} \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix}, \qquad \mathrm{D}_A\, B = [I_{64} + K_{8,8}] \left[\begin{pmatrix} I_4 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ I_4 \end{pmatrix} \right],$$

$$\mathrm{D}_q\, q_L = [(e_R^0)^T, (e_R^1)^T, (e_R^2)^T, (e_R^3)^T]^T, \quad \mathrm{D}_q\, q_R = [(e_L^0)^T, (e_L^1)^T, (e_L^2)^T, (e_L^3)^T]^T,$$

where $\{e^i\}_{i=1}^4$ is the standard basis in $\mathbb{R}^4$.

Proof: The residual orientation error can be rewritten as

$$\begin{aligned} \|e_t\|^2 &= \|q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1\|^2 \\ &= (q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1)(q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1)^c \\ &= 2 - (q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da}) - (q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da})^c. \end{aligned}$$

Using the quaternion properties, $q + q^c = 2q_0$ and $(a \odot b)_0 = a^T b^c$, the above expression can be simplified to

$$\|e_t\|^2 = 2 - 2(q_t^{ab} \odot q^{bc})^T (q_t^{cd} \odot q^{da})^c = 2 - 2(q^{bc})^T (q_t^{ab})_L^T (q_t^{dc})_R q^{ad}.$$

The minimization problem can now be restated as

$$\underset{\substack{\|q^{ad}\|=1\\ \|q^{bc}\|=1}}{\arg\min} \sum_{t=1}^{N} \|e_t\|^2 = \underset{\substack{\|q^{ad}\|=1\\ \|q^{bc}\|=1}}{\arg\max} (q^{bc})^T A q^{ad},$$

where A is defined in (5.29).

Let $q^{bc} = U\alpha$ and $q^{ad} = V\beta$ with $\|\alpha\| = 1$ and $\|\beta\| = 1$, where U and V are orthonormal bases obtained from the singular value decomposition of $A = U\Sigma V^T$. Then,

$$(q^{bc})^T A q^{ad} = \alpha^T U^T U \Sigma V^T V \beta = \sum_{i=1}^{4} \alpha_i \sigma_i \beta_i \leq \sigma_1,$$

where σ_1 is the largest singular value. Equality is obtained for $\alpha = \beta = (1, 0, 0, 0)^T$, that is, $\hat{q}^{bc} = u_1$ and $\hat{q}^{ad} = v_1$.

The sensitivity of the solution can be found by analyzing the differential of the SVD, analogous to Papadopoulo and Lourakis (2000). However, explicit expressions can be obtained by making use of the connection between the singular value decomposition of A and the eigenvalue decomposition of B, see Golub and Van Loan (1996). Indeed,

$$Bx = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ u_1 \end{pmatrix} = \begin{pmatrix} 0 & V\Sigma U^T \\ U\Sigma V^T & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ u_1 \end{pmatrix} = \sigma_1 \begin{pmatrix} v_1 \\ u_1 \end{pmatrix},$$

so a singular value σ_1 of A, with its singular vectors u_1 and v_1, is also an eigenvalue of B with eigenvector x. Hence, the sensitivity of the solution can be determined based on an analysis of the real symmetric eigenvalue equation, $Bx = \sigma x$. The Jacobian of the eigenvector $x(B)$ is given by

$$\mathrm{D}_B\, x = x^T \otimes (\sigma_1 I_8 - B)^\dagger$$

as derived by Magnus and Neudecker (1999). Now, notice that B can be decomposed as

$$B = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix} = \begin{pmatrix} I_4 \\ 0 \end{pmatrix} A^T \begin{pmatrix} 0 & I_4 \end{pmatrix} + \begin{pmatrix} 0 \\ I_4 \end{pmatrix} A \begin{pmatrix} I_4 & 0 \end{pmatrix}.$$

Taking the differential of the above equation yields

$$\begin{aligned} \mathrm{d}\, B &= \begin{pmatrix} I_4 \\ 0 \end{pmatrix} \mathrm{d}\, A^T \begin{pmatrix} 0 & I_4 \end{pmatrix} + \begin{pmatrix} 0 \\ I_4 \end{pmatrix} \mathrm{d}\, A \begin{pmatrix} I_4 & 0 \end{pmatrix}, \\ \mathrm{vec}(\mathrm{d}\, B) &= \left[\begin{pmatrix} 0 & I_4 \end{pmatrix}^T \otimes \begin{pmatrix} I_4 \\ 0 \end{pmatrix} \right] K_{4,4}\, \mathrm{vec}(\mathrm{d}\, A) + \left[\begin{pmatrix} I_4 & 0 \end{pmatrix}^T \otimes \begin{pmatrix} 0 \\ I_4 \end{pmatrix} \right] \mathrm{vec}(\mathrm{d}\, A) \\ &= \underbrace{[I_{64} + K_{8,8}] \left[\begin{pmatrix} I_4 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ I_4 \end{pmatrix} \right]}_{\mathrm{D}_A\, B} \mathrm{vec}(\mathrm{d}\, A). \end{aligned}$$

Furthermore, writing $A_t = L_t^T R_t$ and applying (5.14) once more, yields

$$\begin{aligned} \mathrm{d}\, A_t(L_t) = (\mathrm{d}\, L_t)^T R_t \quad &\Leftrightarrow \quad \mathrm{D}_{L_t}\, A_t = K_{4,4}(I_4 \otimes R_t^T), \\ \mathrm{d}\, A_t(R_t) = L_t^T (\mathrm{d}\, R_t) \quad &\Leftrightarrow \quad \mathrm{D}_{R_t}\, A_t = (I_4 \otimes L_t^T). \end{aligned}$$

Straightforward application of the chain rule results in

$$\mathrm{D}_{q_t^{ab}}\,\hat{x} = [\mathrm{D}_B\,x][\mathrm{D}_A\,B][\mathrm{D}_{L_t}\,A][\mathrm{D}_{q_t^{ab}}\,L_t],$$
$$\mathrm{D}_{q_t^{dc}}\,\hat{x} = [\mathrm{D}_B\,x][\mathrm{D}_A\,B][\mathrm{D}_{R_t}\,A][\mathrm{D}_{q_t^{ab}}\,R_t].$$

Evaluating these expressions gives (5.30). □

Notice that $q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} = 1 \odot \delta q_t$, where $\delta q_t = (\cos\frac{\alpha_t}{2}, \boldsymbol{n}_t^a \sin\frac{\alpha_t}{2}) = (1, \mathbf{0})$ in absence of errors. With this notation, the cost function (5.28) can be interpreted as

$$\begin{aligned} V(\hat{x}) &= \sum_{t=1}^{N} \|q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1\|^2 = \sum_{t=1}^{N} (\cos\tfrac{\alpha_t}{2} - 1, \boldsymbol{n}_t^a \sin\tfrac{\alpha_t}{2})^2 \\ &= \sum_{t=1}^{N} (\cos\tfrac{\alpha_t}{2} - 1)^2 + (\sin\tfrac{\alpha_t}{2})^2 = 2\sum_{t=1}^{N} (1 - \cos\tfrac{\alpha_t}{2}) \approx \frac{1}{4}\sum_{t=1}^{N} \alpha_t^2. \end{aligned}$$

That is, an intuitive interpretation of Theorem 5.3 is that it minimizes error angles, and the solution is physically relevant.

To incorporate measurement noise, the model is extended to

$$q_t^{ab} = q_{t,0}^{ab} \odot \delta q_t^{ab}, \qquad q_t^{dc} = q_{t,0}^{dc} \odot \delta q_t^{dc}, \qquad q_{t,0}^{ab} \odot q^{bc} \odot q_{t,0}^{cd} \odot q^{da} = 1. \tag{5.31}$$

Here, the quaternion errors $\delta q_t^{ab}, \delta q_t^{dc}$ are modeled as mutually independent random rotations about random vectors, that is, $\delta q = \exp\frac{1}{2}\boldsymbol{\theta}$ where $\boldsymbol{\theta}$ is zero mean noise and has covariance Σ_θ. This implies that

$$\mathrm{E}\,\delta q = \mathrm{E}\exp\tfrac{1}{2}\boldsymbol{\theta} \approx \begin{pmatrix} 1 - \mathrm{E}\,\|\boldsymbol{\theta}\|^2 \\ \mathbf{0} \end{pmatrix}, \tag{5.32a}$$

$$\begin{aligned} \mathrm{Cov}\,\delta q &= \mathrm{E}(\delta q - \mathrm{E}\,\delta q)(\delta q - \mathrm{E}\,\delta q)^T \\ &\approx \mathrm{E}\begin{pmatrix} \frac{1}{64}(\|\boldsymbol{\theta}\|^2 - \mathrm{E}\,\|\boldsymbol{\theta}\|^2)^2 & -\frac{1}{16}(\|\boldsymbol{\theta}\|^2 - \mathrm{E}\,\|\boldsymbol{\theta}\|^2)\boldsymbol{\theta}^T \\ -\frac{1}{16}(\|\boldsymbol{\theta}\|^2 - \mathrm{E}\,\|\boldsymbol{\theta}\|^2)\boldsymbol{\theta}^T & \frac{1}{4}\boldsymbol{\theta}\boldsymbol{\theta}^T \end{pmatrix}, \end{aligned} \tag{5.32b}$$

where the small angle approximation, i.e., $\cos x = 1 - \frac{1}{2}x^2$ and $\sin x = x$ have been used. Application of Gauss' approximation formula (5.7) yields the following covariance expression for the estimate of Theorem 5.3

$$\mathrm{Cov}\,\hat{x} = \sum_{t=1}^{N} [\mathrm{D}_{q_t^{ab}}\,\hat{x}]\Sigma_{q_t^{ab}}[\mathrm{D}_{q_t^{ab}}\,\hat{x}]^T + [\mathrm{D}_{q_t^{dc}}\,\hat{x}]\Sigma_{q_t^{dc}}[\mathrm{D}_{q_t^{dc}}\,\hat{x}]^T, \tag{5.33}$$

where the covariance $\Sigma_{q_t^{ab}} = (q_t^{ab})_L[\mathrm{Cov}\,\delta q_t^{ab}](q_t^{ab})_L^T$ (and $\Sigma_{q_t^{dc}}$ analogously).

Assuming independent identically distributed Gaussian noise $\boldsymbol{\theta}_t^{ab}, \boldsymbol{\theta}_t^{dc} \sim \mathcal{N}(0, \sigma^2 I_3)$, the quaternion covariance (5.32b) simplifies to

$$\mathrm{Cov}\,\delta q \approx \begin{pmatrix} \frac{6\sigma^4}{64} & 0 \\ 0 & \frac{\sigma^2}{4} I_3 \end{pmatrix}. \tag{5.34}$$

Hence, for small σ^2 the (1,1) entry is negligible and the residuals e_t are distributed according to

$$\boldsymbol{e}_t \triangleq q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1 \sim \mathcal{N}(0, \Sigma_{e_t}), \qquad \Sigma_{e_t} = \begin{bmatrix} 0 & 0 \\ 0 & \frac{\sigma^2}{2} I_3 \end{bmatrix}. \tag{5.35}$$

Hence, the PDF for all the residuals is given by

$$p(e) = \prod_{t=1}^{N} \frac{1}{(\pi/6)^{\frac{3}{2}}\sigma^3} e^{-\frac{1}{2}(q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1)^T \Sigma_{e_t}^{\dagger} (q_t^{ab} \odot q^{bc} \odot q_t^{cd} \odot q^{da} - 1)}. \tag{5.36}$$

Maximizing $p(e)$, or equivalently $\log p(e)$, w.r.t. x results in Theorem 5.2, only with a different weighting. That is, the result of Theorem 5.2 is very similar to the maximum likelihood (ML) estimate. Furthermore, maximizing $\log p(e)$ w.r.t. σ^2 yields the following ML estimate for the covariance scale factor,

$$\hat{\sigma}^2 = \frac{2V(\hat{x})}{3N}. \tag{5.37}$$

The expected value of $V(\hat{x})$ is given by

$$\begin{aligned} \mathrm{E}\, V(\hat{x}) &= \mathrm{E} \sum_{t=1}^{N} (q_t^{ab} \odot \hat{q}^{bc} \odot q_t^{cd} \odot \hat{q}^{da} - 1)^T (q_t^{ab} \odot \hat{q}^{bc} \odot q_t^{cd} \odot \hat{q}^{da} - 1) \\ &\approx \frac{3(N-2)}{2} \sigma^2, \end{aligned}$$

where a second order Taylor expansion w.r.t. q_t^{ab}, q^{dc} has been used. Hence, the ML estimate $\hat{\sigma}^2$ (5.37) is biased. Correcting for this bias results in

$$\hat{\sigma}^2 = \frac{2}{3(N-2)} V(\hat{x}). \tag{5.38}$$

Validation

Theorem 5.3 and its associated covariance expression have been validated using *Monte Carlo* (MC) simulations. The scenario defined by the orientations $q^{ad} = 2^{-\frac{1}{2}}(1, 1, 0, 0)$, $q^{bc} = 2^{-\frac{1}{2}}(0, 0, 1, 1)$ and measurements $\{q_t^{ab}\} = \{e^1, e^2, e^3, e^4, -e^1, -e^2, -e^3, -e^4\}$ where $\{e^i\}_{i=1}^4$ is the standard basis in $\mathbb{R}^4$ will be used as an example. Measurements are generated by adding Gaussian rotation vectors with $\Sigma = 10^{-4} I_3$ to $\{q_t^{ab}\}$ and $\{q_t^{dc}\}$.

From the measurements a set ($M = 10^4$) of orientation estimates $\{\hat{q}_k^{ad}, \hat{q}_k^{bc}\}_{k=1}^M$ and covariance estimates $\{\hat{Q}_k\}_{k=1}^M$ have been generated using Theorem 5.3 and (5.33). Figure 5.4 shows the distribution of the orientation error vectors $\boldsymbol{e}_k^a \triangleq 2\log(\hat{q}_k^{ad} q^{da})$ and $\boldsymbol{e}_k^b \triangleq 2\log(\hat{q}_k^{bc} q^{cb})$. The orientation errors have zero mean, implying that the orientation estimates are unbiased. Furthermore, the empirical distribution of the orientation errors is consistent with the theoretical distribution derived using the covariance estimates of (5.33). A comparison between the MC covariance $Q_{\mathrm{MC}} = \mathrm{Cov}\, \hat{x}$ and the theoretical

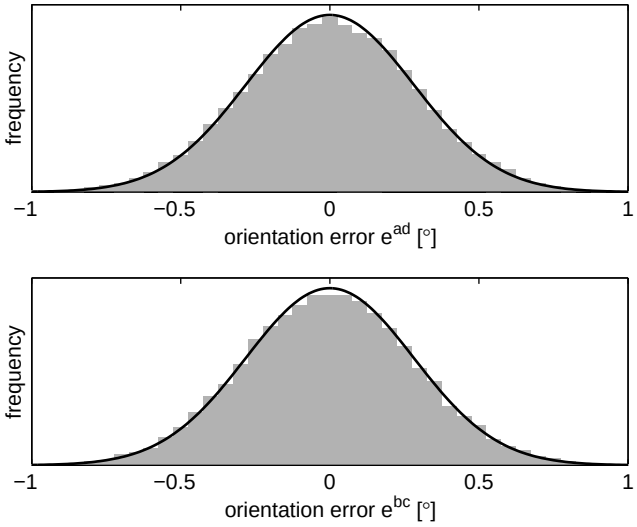


Figure 5.4: *Histogram of the orientation errors. Both the empirical distribution (*gray bar*) as well as the theoretical distribution (*black line*) are shown.*

covariance $Q_{\text{th}} = \mathrm{E}\,\hat{Q}$ shows also a very good match:

$$Q_{\text{MC}} = 10^{-5} \begin{pmatrix} 0.31 & 0.00 & -0.31 & -0.00 & 0.01 & 0.00 & -0.00 & -0.01 \\ 0.00 & 0.62 & -0.00 & -0.00 & -0.00 & -0.00 & -0.01 & 0.00 \\ -0.31 & -0.00 & 0.31 & 0.00 & -0.01 & -0.00 & 0.00 & 0.01 \\ -0.00 & -0.00 & 0.00 & 0.63 & 0.01 & 0.00 & 0.00 & -0.01 \\ 0.01 & -0.00 & -0.01 & 0.01 & 0.31 & -0.00 & -0.01 & -0.31 \\ 0.00 & -0.00 & -0.00 & 0.00 & -0.00 & 0.61 & -0.01 & 0.00 \\ -0.00 & -0.01 & 0.00 & 0.00 & -0.01 & -0.01 & 0.61 & 0.01 \\ -0.01 & 0.00 & 0.01 & -0.01 & -0.31 & 0.00 & 0.01 & 0.31 \end{pmatrix},$$

$$Q_{\text{þ}} = 10^{-5} \begin{pmatrix} 0.31 & 0.00 & -0.31 & 0.00 & 0.00 & -0.00 & 0.00 & -0.00 \\ 0.00 & 0.63 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -0.00 \\ -0.31 & 0.00 & 0.31 & 0.00 & -0.00 & 0.00 & -0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.63 & 0.00 & 0.00 & 0.00 & -0.00 \\ 0.00 & 0.00 & -0.00 & 0.00 & 0.31 & -0.00 & 0.00 & -0.31 \\ -0.00 & 0.00 & 0.00 & 0.00 & -0.00 & 0.63 & 0.00 & -0.00 \\ 0.00 & 0.00 & -0.00 & 0.00 & 0.00 & 0.00 & 0.63 & 0.00 \\ -0.00 & -0.00 & 0.00 & -0.00 & -0.31 & -0.00 & 0.00 & 0.31 \end{pmatrix}.$$

Note that the estimates $\hat{q}^{ad}$ and $\hat{q}^{bc}$ are uncorrelated.

5.3 Mixing kinematic and geometric measurements

The methods discussed in Section 5.1 and Section 5.2 are based on either kinematic or geometric measurements. Difficulties arise combining the two types of measurements,

for instance in case of an IMU measuring angular velocity and acceleration and a camera (indirectly) measuring position and orientation. Attempts using differentiated or integrated measurements are not successful. Integration of the kinematic measurements (dead-reckoning) suffers from severe drift after a short period of time. Differentiating the geometric measurements has problems due to noise amplification and low sampling frequencies.

In this section the prediction error method (Ljung, 1999) is used to combine measurements from both types. The idea used in the prediction error method is very simple, minimize the difference between the measurements and the predicted measurements obtained from a model of the system at hand. This prediction error is given by

$$e_t(\theta) = y_t - \hat{y}_{t|t-1}(\theta), \tag{5.39}$$

where $\hat{y}_{t|t-1}(\theta)$ is used to denote the one-step ahead prediction from the model. The parameters θ are now found by minimizing a norm $V(\theta, e)$ of the prediction errors,

$$\hat{\theta} = \arg\min_{\theta} V_N(\theta, e). \tag{5.40}$$

Obviously, a suitable predictor $\hat{y}_{t|t-1}(\theta)$ is needed to solve (5.40). The key idea is to realize that the state-space models derived in Chapter 4 describe the underlying process and that a EKF can be used to compute the one-step ahead prediction $\hat{y}_{t|t-1}(\theta)$, see Figure 5.5. The parameters in the process and measurements model have a clear physical

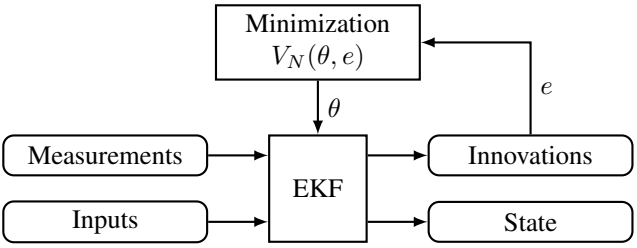


Figure 5.5: *Gray-box system identification using KF innovations as prediction errors. The parameter vector θ is adjusted to minimize the cost function $V_N(\theta, e)$.*

interpretation, allowing for gray-box identification where only the parameters of interest are estimated. The prediction errors, $e_t = y_t - \hat{y}_{t|t-1}(\theta)$, or innovations are already being computed in the KF iterations. This explains why the term innovation representation is used for KF-based model structures (Ljung, 1999).

Although the choice of the cost function $V_N(\theta, e)$ does not influence the limit of the estimate $\hat{\theta}$, it influences the covariance of the estimate (Ljung, 1999). The optimal, minimum variance estimate is obtained with the maximum likelihood cost function

$$V_N(\theta, e) = \sum_{t=1}^{N} -\log p_e(e_t, t). \tag{5.41}$$

Since for a correctly tuned filter, the innovations e_t are normal distributed with zero mean and covariance $S_t = C_t P_{t|t-1} C_t^T + R_t$, where the state covariance $P_{t|t-1}$, the measurement Jacobian C_t and the measurement covariance R_t are provided by the EKF, the cost

function (5.41) is equivalent to

$$V_N(\theta, e) = \frac{1}{2}\sum_{t=1}^{N} e_t^T S_t^{-1} e_t = \frac{1}{2}\epsilon^T \epsilon. \tag{5.42}$$

Here, the Nn_y-dimensional vector $\epsilon = (\epsilon_1^T, \ldots, \epsilon_N^T)^T$ is constructed by stacking the normalized innovations

$$\epsilon_t = S_t^{-1/2} e_t \tag{5.43}$$

on top of each other. With this choice of cost function the optimization problem boils down to a nonlinear least-squares problem. This kind of problem can be efficiently solved using Gauss-Newton or Levenberg-Marquardt methods, see e.g., Nocedal and Wright (2006). These methods require partial derivatives of the normalized innovations ϵ w.r.t. the parameter vector θ. Since the KF iterations do not allow simple analytic expression for these numerical differentiation is used. The covariance of the estimate $\hat{\theta}$ can be determined using

$$\operatorname{Cov}\hat{\theta} = \frac{\epsilon^T \epsilon}{Nn_y}\left([\mathrm{D}_\theta\, \epsilon][\mathrm{D}_\theta\, \epsilon]^T\right)^{-1}, \tag{5.44}$$

where the residuals ϵ and the Jacobians $[\mathrm{D}_\theta\, \epsilon]$ are evaluated at $\hat{\theta}$, see Ljung (1999). This expression can also be obtained by linearizing $\epsilon(\theta)$,

$$\epsilon(\theta) \approx \epsilon(\hat{\theta}) + [\mathrm{D}_\theta\, \epsilon](\theta - \hat{\theta}).$$

and applying Theorem 5.1.

The validation of this system identification approach to calibrate relative pose from mixed measurements is postponed to Section 6.3.

6

Calibration algorithms

The calibration theory discussed in Chapter 5 can be applied in a number of calibration methods for relative pose. Two groups are distinguished: internal calibration algorithms, which calibrate the coordinate frames within the sensor unit of the tracking system and external calibration algorithms for calibrating the tracking system with an external reference system. The classification and their coordinate frames are shown in Figure 6.1. These coordinates frames have already been introduced in the previous chapters, but their

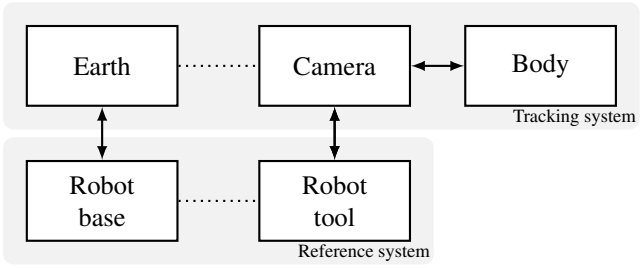


Figure 6.1: *coordinate frames of the tracking system and the reference system. Solid lines are rigid connections, dotted lines are non-rigid, varying connections.*

definition will be repeated here for convenience.

Earth (e): The camera pose is estimated with respect to this coordinate frame. It is fixed to earth and the features of the scene are modeled in this coordinate frame. It can be aligned in any way, however, preferably it should be vertically aligned.

Camera (c): The coordinate frame attached to the moving camera. Its origin is located in the optical center of the camera, with the z-axis pointing along the optical axis. The camera, a projective device, acquires its images in the **image plane (i)**. This

plane is perpendicular to the optical axis and is located at an offset (focal length) from the optical center of the camera.

Body (b): This is the coordinate frame of the IMU. Even though the camera and the IMU are rigidly attached to each other and contained within the sensor unit, the body coordinate frame does not coincide with the camera coordinate frame. They are separated by a constant translation and rotation.

Robot base (r): The robot positions its tool with respect to this coordinate frame. This unmovable frame is fixed to the earth, so it is rigidly connected to the earth frame.

Robot tool (t): This is the coordinate frame controlled by the robot. The sensor unit is mounted to the robot tool, hence the camera frame is rigidly connected to this frame.

Both the internal and external calibration methods will be discussed in the subsequent sections.

6.1 Internal calibration

Internal calibration algorithms determine the relative pose between the camera and the IMU in the sensor unit, see Chapter 3. It has to be repeated when the objective of the camera is adjusted or changed. Since in that case a camera calibration has to be performed anyway, it would be preferable when the sensor unit can be calibrated reusing the camera calibration setup and without additional hardware.

Lobo and Dias (2007) present a two-step algorithm to determine the relative pose between an IMU and a camera. First they determine the relative orientation comparing gravity measured by the IMU with gravity determined from the camera images, i.e., using directional vectors as discussed in Section 5.2.1. The optical vertical direction is calculated from vertical vanishing points, determined using vertical lines present in the scene or in a vertically aligned camera calibration pattern. However, it is quite difficult to align a calibration pattern perfectly with the vertical; it requires a vertical surface on which the pattern has to be aligned. It turns out that floors and desks are more horizontal than walls and edges are vertical. Additionally, levelness can be easily verified with a spirit level and the vertical is independent of the alignment in the horizontal plane. Taking this into account, a modified algorithm is presented in Algorithm 6.1. It is an extension to the camera calibration procedure, see Section 3.2.2, and facilitates calculation of the relative orientation between the IMU and the camera.

Once the relative orientation is known, Lobo and Dias (2007) describe a second algorithm to determine the relative position: the sensor unit has to be placed on a turntable and positioned such that the accelerometer reading stays constant for rapid rotations. That is, the accelerometers are positioned in the rotation center of the turntable and do not translate when the turntable is rotated. This geometric measurement is complemented with relative pose information from images of a calibration pattern before and after a turn. To calculate the relative position this procedure has to be repeated several times with different poses of the sensor unit on the turntable.

Algorithm 6.1 Orientation calibration (static)

1. Place a camera calibration pattern on a horizontal, level surface, e.g., a desk or the floor.
2. Acquire images $\{I_t\}_{t=1}^N$ of the pattern while holding the sensor unit static in various poses, simultaneously taking accelerometer readings $\{\boldsymbol{y}_{a,t}\}_{t=1}^N$.
3. Perform a camera calibration using the images $\{I_t\}_{t=1}^N$ to obtain the orientations $\{R_t^{ce}\}_{t=1}^N$.
4. Compute an estimate $\hat{q}^{cb}$ from $\{\boldsymbol{g}_t^c\} = \{R_t^{ce}\boldsymbol{g}^e\}$ and $\{\boldsymbol{g}_t^b\} = \{-\boldsymbol{y}_{a,t}\}$ using Theorem 5.2. Note that $\boldsymbol{g}^e = (0, 0, -g)^T$ since the calibration pattern is placed horizontally.
5. Determine the covariance of $\hat{q}^{cb}$ using (5.19).

This algorithm has two major drawbacks: not only does it require special hardware, i.e., a turntable, but it is also labor intensive as the positioning of the sensor unit is very sensitive. Using Section 5.3 a flexible algorithm has been derived for estimating the relative pose between the IMU and the camera which does not suffer from these drawbacks.

The system is modeled similar to Section 4.3 as

$$\boldsymbol{b}_{t+1}^e = \boldsymbol{b}_t^e + T\dot{\boldsymbol{b}}_t^e + \frac{T^2}{2}\ddot{\boldsymbol{b}}_t^e, \tag{6.1a}$$

$$\dot{\boldsymbol{b}}_{t+1}^e = \dot{\boldsymbol{b}}_t^e + T\ddot{\boldsymbol{b}}_t^e, \tag{6.1b}$$

$$q_{t+1}^{be} = e^{-\frac{T}{2}\boldsymbol{\omega}_{eb,t}^b} \odot q_t^{be}, \tag{6.1c}$$

where $\ddot{\boldsymbol{b}}_t^e$ and $\omega_{eb,t}^b$ are given by

$$\ddot{\boldsymbol{b}}_t^e = R_t^{eb}\boldsymbol{u}_{a,t} + \boldsymbol{g}^e - R_t^{eb}\boldsymbol{\delta}_a^b - R_t^{eb}\boldsymbol{e}_{a,t}^b, \tag{6.2a}$$

$$\boldsymbol{\omega}_{eb,t}^b = \boldsymbol{u}_{\omega,t} - \boldsymbol{\delta}_\omega^b - \boldsymbol{e}_{\omega,t}^b. \tag{6.2b}$$

Here $\boldsymbol{u}_{a,t}$ and $\boldsymbol{u}_{\omega,t}$ are the accelerometer signal and the gyroscope signal respectively. The associated measurements are modeled as

$$\boldsymbol{y}_c = \begin{bmatrix} -I_2 & \boldsymbol{p}_n^i \end{bmatrix} R^{cb}(R^{be}(\boldsymbol{p}^e - \boldsymbol{b}^e) - \boldsymbol{c}^b) + \boldsymbol{e}_c. \tag{6.3}$$

This is a standard discrete-time state-space model parameterized by

$$\theta = \left((\boldsymbol{\phi}^{cb})^T, (\boldsymbol{c}^b)^T, (\boldsymbol{\delta}_\omega^b)^T, (\boldsymbol{\delta}_a^b)^T, (\boldsymbol{g}^e)^T\right) \tag{6.4}$$

That is, the parameter vector θ consists of relative orientation as axis angle $\boldsymbol{\phi}^{cb}$, relative position $\boldsymbol{c}^b$, gyroscope bias ($\boldsymbol{\delta}_\omega^b$, accelerometer bias $\boldsymbol{\delta}_a^b$ and gravity $\boldsymbol{g}^e$.

Algorithm 6.2 applies the model (6.1)–(6.3) in the grey-box system identification approach discussed in Section 5.3 to estimate the relative pose. Besides relative position and

Algorithm 6.2 Pose calibration (dynamic)

1. Place a camera calibration pattern on a horizontal, level surface, e.g., a desk or the floor.
2. Acquire inertial measurements $\{\boldsymbol{y}_{a,t}\}_{t=1}^{M}, \{\boldsymbol{y}_{g,t}\}_{t=1}^{M}$ and images $\{I_t\}_{t=1}^{N}$.
   - Rotate around all 3 axes, with sufficiently exiting angular velocities.
   - Always keep the calibration pattern in view.
3. Obtain the point correspondences between the undistorted and normalized 2D feature locations $\boldsymbol{z}_{t,k}^{i,n}$ and the corresponding 3D grid coordinates $\boldsymbol{z}_{t,k}^{e}$ of the calibration pattern for all images $\{I_t\}$, see Section 3.2.2.
4. Compute an estimate $\hat{\theta}$ by solving (5.40), using $\theta_0 = \left((\boldsymbol{\phi}_0^{cb})^T, \mathbf{0}, \mathbf{0}, \mathbf{0}, (\boldsymbol{g}_0^e)\right)$ as a staring point for the optimization. Here, $\boldsymbol{g}_0^e = (0, 0, -g)^T$ since the calibration pattern is placed horizontally and $\boldsymbol{\phi}_0^{cb}$ can be obtained using Algorithm 6.1.
5. Determine the covariance of $\hat{\theta}$ using (5.44).

orientation, nuisance parameters like sensor biases and gravity are also determined. The algorithm requires a calibrated camera and, apart from a camera calibration pattern, no hardware is required. The data sequences can be short, a few seconds of data is sufficient. The algorithm is very flexible: the motion of the sensor unit can be arbitrary, provided it contains sufficient rotational excitation. A convenient setup for the data capture is to mount the sensor unit on a tripod and pan, tilt and roll it. However, hand-held sequences can be used equally well.

6.2 External calibration

In case the tracking system is to be compared to an external reference system a calibration has to be performed to determine the relative poses between the coordinate frames involved. Depending of the type of reference system different calibration methods have to be used. For a reference system providing pose measurements, e.g., an industrial robot as used in Chapter 2, the theory of Section 5.2 applies and Algorithm 6.3 can be used.

Alternatively, a high grade inertial navigation system, see Section 3.1.3, can be used to as an external reference system. Such systems also provide kinematic measurements, see Section 5.1, and Algorithm 6.4 can be used to determine the relative poses between the involved coordinate frames.

6.3 Experiments

The algorithms presented in the previous sections have been applied to obtain the results of Chapter 2. This section is devoted to show calibration results for internal calibration.

Algorithm 6.3 Reference system calibration (pose)

1. Acquire pose measurements $\{T_t^{ec}\}_{t=1}^N$ and $\{T_t^{rt}\}_{t=1}^N$ from the sensor unit and the reference system respectively.
2. Compute an estimate $\hat{q}^{rw}, \hat{q}^{tc}$ from $\{q_t^{rt}\}$ and $\{q_t^{ec}\}$ using Theorem 5.3.
3. Compute an estimate $\hat{\boldsymbol{e}}^r, \hat{\boldsymbol{c}}^t$ from $\{\boldsymbol{t}_t^r\}$ and $\{\boldsymbol{c}_t^e\}$ by applying Theorem 5.1 to (5.27).
4. Use (5.33) and (5.8) to determine the covariances of $\hat{q}^{rw}, \hat{q}^{tc}$ and $\hat{\boldsymbol{e}}^r, \hat{\boldsymbol{c}}^t$.

Algorithm 6.4 Reference system calibration (inertial)

1. Capture inertial measurements $\{\boldsymbol{y}_{a,t}\}_{t=1}^N, \{\boldsymbol{y}_{\omega,t}\}_{t=1}^N$ and $\{\boldsymbol{z}_{a,t}\}_{t=1}^N, \{\boldsymbol{z}_{\omega,t}\}_{t=1}^N$ from the sensor unit and the reference IMU respectively. Rotate around all 3 axes, with sufficiently exiting angular velocities.
2. Compute an estimate $\hat{q}^{bt}$ from $\{\boldsymbol{\omega}_t^b\} = \{\boldsymbol{y}_{\omega,t}\}$ and $\{\boldsymbol{\omega}_t^t\} = \{\boldsymbol{z}_{\omega,t}\}$ using Theorem 5.2.
3. Compute an estimate $\hat{\boldsymbol{b}}^t$ from $\{\boldsymbol{y}_{a,t}\}$ and $\{\boldsymbol{z}_{a,t}\}$ by application of Theorem 5.2 to the combination of (5.3) and
$$\boldsymbol{y}_{a,t} - R^{bt}\boldsymbol{z}_{a,t} = R_t^{be}(\ddot{\boldsymbol{b}}_t^e - \boldsymbol{g}^e) - R^{bt}R_t^{te}(\ddot{\boldsymbol{t}}_t^e - \boldsymbol{g}^e) = R_t^{be}(\ddot{\boldsymbol{b}}_t^e - \ddot{\boldsymbol{t}}_t^e).$$
4. Use (5.19) and (5.8) to determine the covariances of $\hat{q}^{bt}$ and $\hat{\boldsymbol{t}}^b$.

Algorithm 6.2 has been used to calibrate the sensor unit described in Chapter 3. This algorithm computes estimates of the relative position and orientation between the IMU and the camera, i.e., $\boldsymbol{c}^b$ and $\boldsymbol{\phi}^{cb}$, based on the motion of the sensor unit. This motion can be arbitrary, as long as it is sufficiently exiting in angular velocity and the calibration pattern always stays in view. The setup employed, shown in Figure 6.2, is identical to that of a typical camera calibration setup: the camera has been mounted on a tripod and a camera calibration pattern is placed on a desk.

A number of experiments have been performed. During such an experiment the sensor unit has been rotated around its three axes, see Figure 6.3 for an example. The measurements contains relatively small rotations as the calibration pattern has to stay in view. However, modest angular velocities are present, which turn out to provide sufficient excitation. The data is split into two parts, one estimation part and one validation part, see Figure 6.3. This facilitates cross-validation, where the parameters are estimated using the estimation data and the quality of the estimates can then be assessed using the validation data Ljung (1999).

In Table 6.1 the estimates produced by Algorithm 6.2 are given together with confidence intervals (99%). Note that the estimates are contained within the 99% confidence

Figure 6.2: *The* sensor unit *is mounted on a tripod for calibration. The* background *shows the camera calibration pattern that has been used in the experiments.*

intervals. Reference values are also given, these are taken as the result of Algorithm 6.1 (orientation) and from the technical drawing (position). Note that the drawing defines the position of the CCD, not the optical center. Hence, no height reference is available and some shifts can occur in the tangential directions. Table 6.1 indicates that the estimates are indeed rather good.

In order to further validate the estimates the normalized innovations (5.43) are studied. A histogram of the normalized innovations and their autocorrelations are given in Figure 6.4 and Figure 6.5, respectively. Both figures are generated using the validation data. The effect of using the wrong relative position and orientation is shown in Figure 6.4b and Figure 6.4c. From Figure 6.4a and Figure 6.5 it is clear that the normalized innovations are close to white noise using $\hat{\theta}$. This implies that the model with the estimated parameters and its assumptions appears to be correct, which in turn is a good indication that reliable estimates $\hat{\phi}^{cb}, \hat{c}^{b}$ have been obtained. The reliability and repeatability of the estimates has also been confirmed by additional experiments.

The experiments show that Algorithm 6.2 is an easy-to-use calibration method to determine the relative position and orientation between the IMU and the camera. Even small displacements and misalignments of the sensor unit can be accurately calibrated from short measurement sequences made using the standard camera calibration setup.

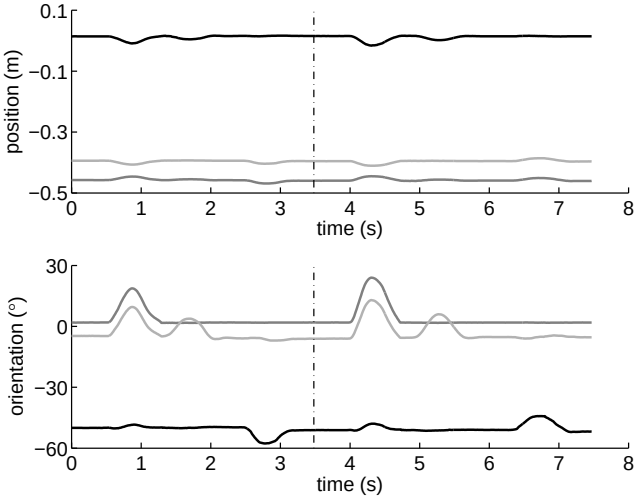


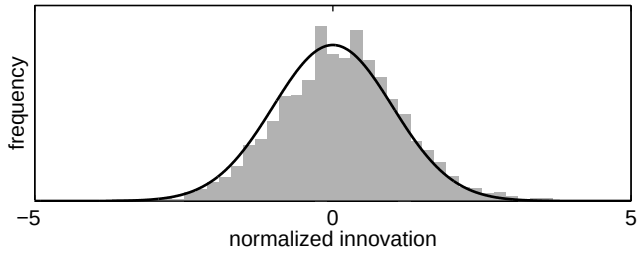
Figure 6.3: *A trajectory of the sensor unit used for calibration. It contains both estimation data ($t < 3.5$ s) and validation data ($t \geq 3.5$ s), as indicated by the dashed line.*

Table 6.1: *Calibration results for Algorithm 6.2. The obtained estimates and their 99% confidence intervals are listed for 3 trials. Reference values have been included for comparison.*

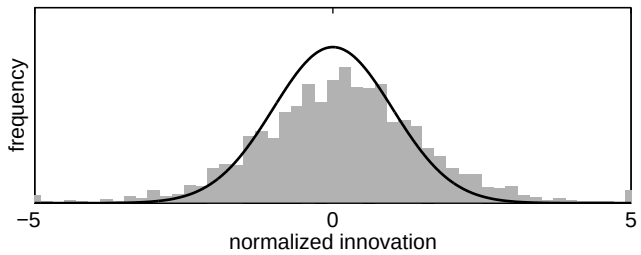
Orientation	ϕ_x^{cb} (°)	ϕ_y^{cb} (°)	ϕ_z^{cb} (°)
Trial 1	-0.06 [-0.28, 0.17]	0.84 [0.67, 1.01]	0.19 [-0.06, 0.44]
Trial 2	-0.19 [-0.36, -0.02]	0.75 [0.62, 0.88]	0.45 [0.23, 0.67]
Trial 3	-0.29 [-0.48, -0.10]	0.91 [0.76, 1.05]	0.08 [-0.11, 0.27]
Reference[a]	-0.23 [-0.29, -0.17]	0.80 [0.73, 0.87]	0.33 [0.22, 0.44]
Position	c_x^b (mm)	c_y^b (mm)	c_z^b (mm)
Trial 1	-13.5 [-15.2, -11.9]	-6.7 [-8.1, -5.2]	34.5 [31.0, 38.0]
Trial 2	-15.7 [-17.3, -14.2]	-8.8 [-10.1, -7.5]	33.2 [28.7, 37.7]
Trial 3	-13.5 [-14.9, -12.0]	-7.3 [-8.6, -6.0]	29.7 [26.8, 32.7]
Reference[b]	-14.5	-6.5	-

[a] using Algorithm 6.1.

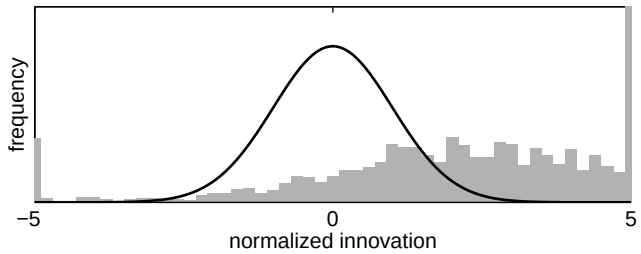
[b] using the CCD position of the technical drawing.



(a) using $\hat{\theta}$



(b) using $c^b = 0$



(c) using $\varphi^{cb} = 0$

Figure 6.4: *Histograms of the normalized innovations, for validation data. Both the empirical distribution (gray bar) as well as the theoretical distribution (black line) are shown for several parameter vectors.*

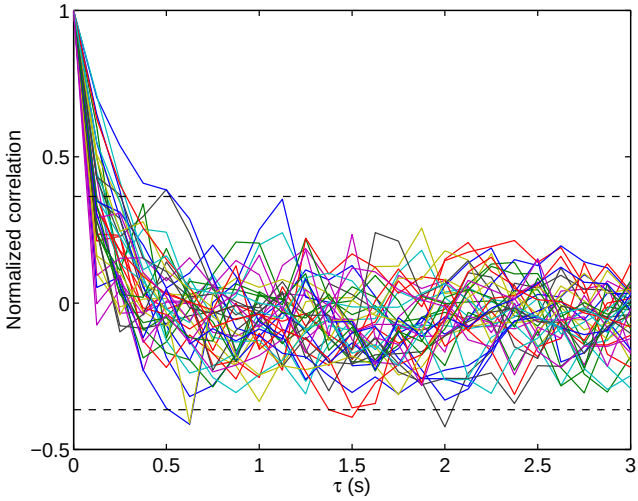


Figure 6.5: *Autocorrelation of the normalized innovations, for validation data. The horizontal lines indicate the* 99% *confidence interval.*

7

Application example

The pose estimation system of Chapter 2 has been tested in a number of scenarios. Its accuracy has been evaluated using an industrial robot as ground truth, discussed in Section 2.5. Furthermore, the system has been tested as an augmented reality application, also reported in Chandaria et al. (2007). The results of this experiment will be the topic of this chapter.

Example 7.1: An augmented reality application

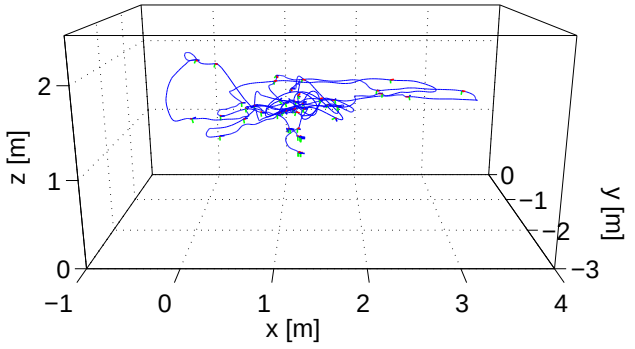
The system has been used to track the sensor unit in a relatively large room, approximately $5 \times 4 \times 2.5$ m in size, see Figure 7.1. The sensor unit is handheld and is allowed to

Figure 7.1: *The sensor unit is tracked in a large room. The monitor shows the live camera image augmented with a virtual character.*

move without constraints in this room, both close to and far away from the walls. The pose output of the pose estimation system is used to draw virtual graphics on top of the camera images in real-time. There is no ground-truth data available for this test, implying that the tracking performance has to be evaluated qualitatively from the quality of the augmentation.

The pose estimation system requires a 3D model of the environment. In this case, the model was not generated using the computer vision approaches described in Section 3.2.3, but created manually using a 3D modeling tool. This tool takes the geometry from a CAD model and uses digital photo's to obtain textures for the surfaces. The resulting model, shown in Figure 7.2a, consists of the three main walls. The floor and roof do

(a) *3D model of the room.*



(b) *Camera trajectory.*

Figure 7.2: *Overview of the test setup.*

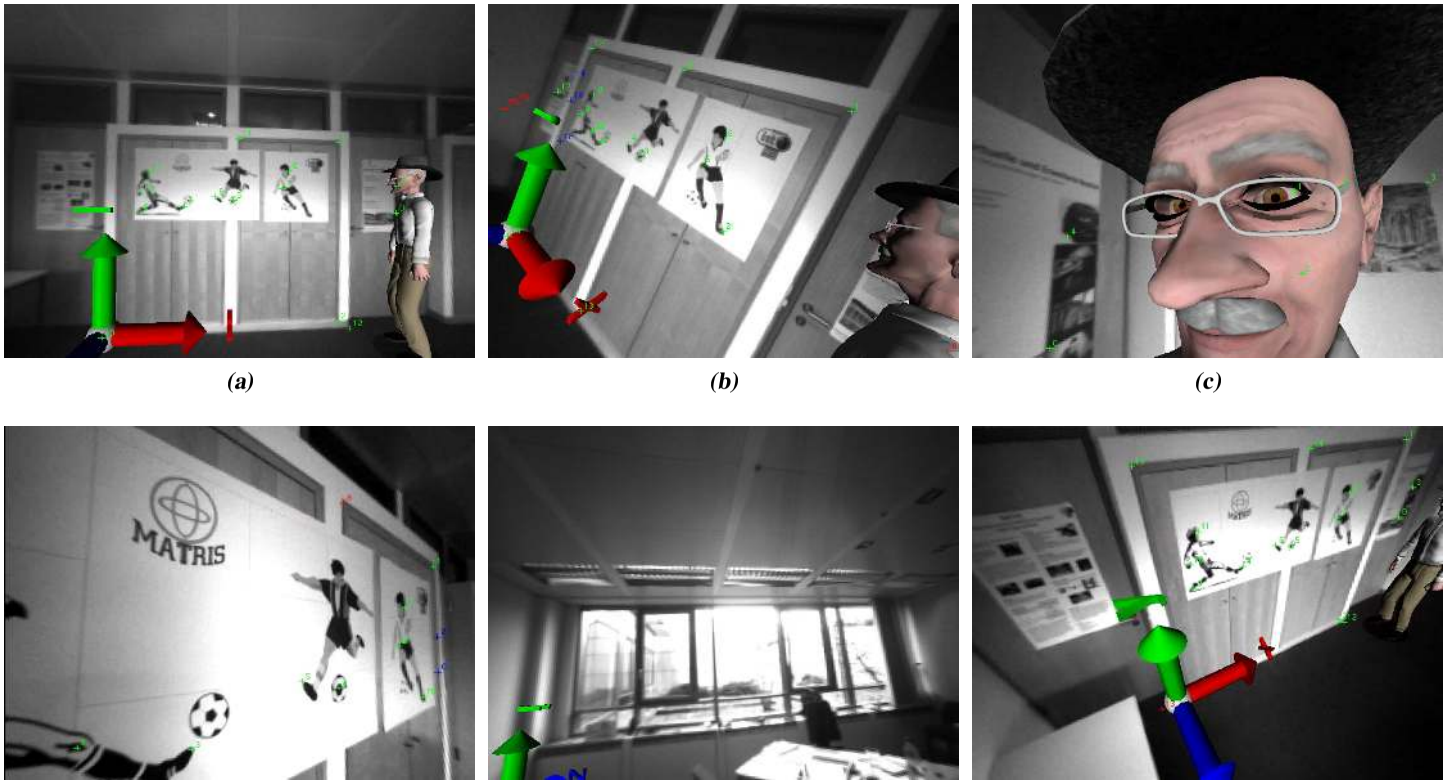
not contain sufficient features and are ignored, together with the fourth wall containing mostly windows.

The system worked very well for the described setup. The augmentations showed no visible jitter or drift, even during fast motion. Tracking continued for extensive periods of time without deterioration or divergence. Furthermore, the system is capable to handle periods with few or no features at all, which pose difficulties for pure computer vision approaches. These situations occur for instance when the camera is close to a wall or during a 360° revolution. A reinitialization was required after 2 s without visible features.

Beyond that period the predicted feature positions were to far off to enable detection.

A sample trajectory of about $90\ \mathrm{s}$ is shown in Figure 7.2b. It contains acceleration up to $12\ \mathrm{m/s^2}$ and angular velocity up to $9.5\ \mathrm{rad/s}$. Furthermore, the trajectory involves several $360°$ rotations which include several views where the camera only observes the unmodeled window wall. An impression of the augmentation result is given by Figure 7.3. The overlaid graphics stay on the same location, regardless of the position and orientation of the camera. This is also the case when no features are available, for instance when only the unmodeled wall is in view, see Figure 7.3e.

This example serves as a proof of concept for the performance of the developed pose estimation system in realistic environments. The potential of the system is high as this example is only one of many possible applications where it can be used.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 7.3: *Several frames taken from the sample trajectory. Shown are the camera image (gray), located feature positions (+) and overlaid graphics (avatar, coordinate frames).*

8

Concluding remarks

In this thesis the problem of pose estimation is approached using a combination of vision and inertial sensors. The aim has been to show how the associated nonlinear state estimation problem can be solved in real-time using the available sensor information and how a solution working in practice can be obtained. The conclusions are given in Section 8.1 and in Section 8.2 some ideas about future work are discussed.

8.1 Conclusions

This thesis deals with estimating position and orientation in real-time, using measurements from vision and inertial sensors. A system has been developed to solve this problem in unprepared environments, assuming that a map or scene model is available. Compared to 'camera-only' systems, the combination of the complementary sensors yields a robust system which can handle periods with uninformative or no vision data and reduces the need for high frequency vision updates.

The system is well suited for use in augmented reality applications. An application example is discussed where the augmentations based on the pose estimates from the system showed no visible jitter or drift, even during fast motion and tracking continued for extensive periods of time without deterioration or divergence. Furthermore, experiments where an industrial robot is used to move the sensor unit show that this setup is able to track the camera pose with an absolute accuracy of $2\ \mathrm{cm}$ and 1°.

The system achieves real-time pose estimation by fusing vision and inertial sensors using the framework of nonlinear state estimation. Accurate and realistic process and measurement models are required. Fur this reason, a detailed analysis of the sensors and their measurements has been performed.

Calibration of the relative position and orientation of the camera and the inertial sensor is essential for proper operation. A new algorithm for estimating these parameters has been developed, which does not require any additional hardware, except a piece of

paper with a checkerboard pattern on it. The key is to realize that this problem is in fact an instance of a standard problem within the area of system identification, referred to as a gray-box problem. The experimental results shows that the method works well in practice. Even small displacements and misalignments can be accurately calibrated from short measurement sequences made using the standard camera calibration setup.

8.2 Future work

Some suggestions for future research related to the work in this thesis are the following:

- **Sensor fusion**
  - Adapt the pose estimation system and its models to function in combination with spherical lenses.
  - Extend the scene model while tracking with newly observed features or generate it from scratch. That is, perform simultaneous localization and mapping (SLAM).
  - Investigate covariance estimation for feature detectors.
- **Calibration algorithms**
  - Extend the calibration method to determine the relative pose of the camera and the inertial sensor (Algorithm 6.2) for use with spherical lenses.
  - Apply the calibration theory of Chapter 5 to related problems, such as determining the distance between a GPS antenna and an IMU.

Bibliography

Analog Devices, 2008. URL http://www.analog.com/. Accessed April 2nd, 2008.

L. Armesto, J. Tornero, and M. Vincze. Fast ego-motion estimation with multi-rate fusion of inertial and vision. *International Journal of Robotics Research*, 26(6):577–589, 2007. doi:10.1177/0278364907079283.

M. Aron, G. Simon, and M.-O. Berger. Use of inertial sensors to support video tracking. *Computer Animation and Virtual Worlds*, 18(1):57–68, 2007. doi:10.1002/cav.161.

B. Bartczak, K. Koeser, F. Woelk, and R. Koch. Extraction of 3D freeform surfaces as visual landmarks for real-time tracking. *Journal of Real-Time Image Processing*, 2(2): 81–101, Nov. 2007. doi:10.1007/s11554-007-0042-0.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Journal of Computer Vision and Image Understanding*, 2008. doi:10.1016/j.cviu.2007.09.014. Accepted for publication.

N. Bergman. *Recursive Bayesian Estimation: Navigation and Tracking Applications*. Dissertations no 579, Linköping Studies in Science and Technology, SE-581 83 Linköping, Sweden, May 1999.

J.-Y. Bouguet. Camera calibration toolbox for matlab, 2003. URL http://www.vision.caltech.edu/bouguetj/calib_doc/. Accessed April 2nd, 2008.

R. S. Bucy and K. D. Senne. Digital synthesis on nonlinear filters. *Automatica*, 7:287–298, 1971. doi:10.1016/0005-1098(71)90121-X.

J. Chandaria, G. A. Thomas, and D. Stricker. The MATRIS project: real-time markerless camera tracking for augmented reality and broadcast applications. *Journal of Real-Time Image Processing*, 2(2):69–79, Nov. 2007. doi:10.1007/s11554-007-0043-z.

A. Chatfield. *Fundamentals of High Accuracy Inertial Navigation*, volume 174. American Institute of Aeronautics and Astronautics, USA, 3rd edition, 1997. ISBN 1563472430.

S. G. Chroust and M. Vincze. Fusion of vision and inertial data for motion and structure estimation. *Journal of Robotics Systems*, 21(2):73–83, 2004. doi:10.1002/rob.10129.

P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *International Journal of Robotics Research*, 26(6):519–535, 2007. doi:10.1177/0278364907079279.

A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of 9th IEEE International Conference on Computer Vision*, volume 2, pages 1403–1410, Nice, France, Oct. 2003. doi:10.1109/ICCV.2003.1238654.

A. J. Davison. Active search for real-time vision. In *Proceedings of 10th IEEE International Conference on Computer Vision*, pages 66–73, Beijing, China, Oct. 2005. doi:10.1109/ICCV.2005.29.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (6):1052–1067, June 2007. doi:10.1109/TPAMI.2007.1049.

H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping (SLAM): Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, June 2006. doi:10.1109/MRA.2006.1638022.

F. Ferraris, U. Grimaldi, and M. Parvis. Procedure for effortless in-field calibration of three-axial rate gyro and accelerometers. *Sensors and Materials*, 7(5):311–330, 1995. ISSN 0914-4935.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. doi:10.1145/358669.358692.

P. Gemeiner, P. Einramhof, and M. Vincze. Simultaneous motion and structure estimation by fusion of inertial and vision data. *International Journal of Robotics Research*, 26(6): 591–605, 2007. doi:10.1177/0278364907080058.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996. ISBN 0801854148.

N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, 140(2):107–113, Apr. 1993. ISSN 0956-375X.

F. Gustafsson, T. B. Schön, and J. D. Hol. Sensor fusion for augmented reality. In *Proceedings of 17th International Federation of Automatic Control World Congress*, Seoul, South Korea, July 2008. Accepted for publication.

W. R. Hamilton. On quaternions; or on a new system of imaginaries in algebra. *Philosophical Magazine*, xxv, 1844.

C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, Manchester, UK, 1988.

R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004. ISBN 0521540518.

G. Hendeby, J. D. Hol, R. Karlsson, and F. Gustafsson. A graphics processing unit implementation of the particle filter. In *Proceedings of European Signal Processing Conference*, Poznań, Poland, Sept. 2007.

J. D. Hol, T. B. Schön, and F. Gustafsson. On resampling algorithms for particle filters. In *Proceedings of Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, Sept. 2006a. doi:10.1109/NSSPW.2006.4378824.

J. D. Hol, T. B. Schön, F. Gustafsson, and P. J. Slycke. Sensor fusion for augmented reality. In *Proceedings of 9th International Conference on Information Fusion*, Florence, Italy, July 2006b. doi:10.1109/ICIF.2006.301604.

J. D. Hol, T. B. Schön, H. Luinge, P. J. Slycke, and F. Gustafsson. Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing*, 2(2):149–160, Nov. 2007. doi:10.1007/s11554-007-0040-2.

B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, Apr. 1987.

IEEE Std 952-1997. IEEE standard specifcation format guide and test procedure for single- axis interferometric fiber optic gyros. Technical report, IEEE, 1998. Annex C.

M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998. doi:10.1023/A:1008078328650.

A. H. Jazwinski. *Stochastic processes and filtering theory*. Mathematics in science and engineering. Academic Press, New York, USA, 1970. ISBN 978-0123815507.

S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, Mar. 2004. doi:10.1109/JPROC.2003.823141.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35–45, 1960.

J. Kannala and S. S. Brandt. Generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1335–1340, aug 2006. doi:10.1109/TPAMI.2006.153.

G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, Mar. 1996.

G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of 6thInternational Symposium on Mixed and Augmented Reality*, Nara, Japan, Nov. 2007.

G. S. W. Klein and T. W. Drummond. Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing*, 22(10):769–776, 2004. doi:10.1016/j.imavis.2004.02.007.

K. Koeser, B. Bartczak, and R. Koch. Robust GPU-assisted camera tracking using free-form surface models. *Journal of Real-Time Image Processing*, 2(2):133–147, Nov. 2007. doi:10.1007/s11554-007-0039-8.

J. B. Kuipers. *Quaternions and Rotation Sequences*. Princeton University Press, 1999. ISBN 0691102988.

L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, Inc, Upper Saddle River, NJ, USA, 2nd edition, 1999. ISBN 0-13-656695-2.

J. Lobo and J. Dias. Inertial sensed ego-motion for 3D vision. *Journal of Robotics Systems*, 21(1):3–12, 2004. doi:10.1002/rob.10122.

J. Lobo and J. Dias. Relative pose calibration between visual and inertial sensors. *International Journal of Robotics Research*, 26(6):561–575, 2007. doi:10.1177/0278364907079276.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004. doi:10.1023/B:VISI.0000029664.99615.94.

Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An invitation to 3-D vision – from images to geometric models*. Interdisciplinary Applied Mathematics. Springer-Verlag, 2006. ISBN 0387008934.

J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley & Sons, Ltd, 2nd edition, 1999. ISBN 978-0471986331.

MATRIS, 2008. URL `http://www.ist-matris.org/`. Accessed April 2nd, 2008.

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct. 2005. doi:10.1109/TPAMI.2005.188.

K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, Nov. 2005. doi:10.1007/s11263-005-3848-x.

J. Nocedal and S. J. Wright. *Numerical optimization*. Springer-Verlag, New York, 2006. ISBN 0387987932.

T. Papadopoulo and M. I. A. Lourakis. Estimating the jacobian of the singular value decomposition: Theory and applications. In *Proceedings of 6th European Conference on Computer Vision*, pages 554–570, Dublin, Ireland, June 2000. doi:10.1007/3-540-45054-8_36.

R. J. B. Pieper. Comparing estimation algorithms for camera position and orientation. Master's thesis, Department of Electrical Engineering, Linköping University, Sweden, 2007.

P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardos. Inertial aiding of inverse depth SLAM using a monocular camera. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2797–2802, Roma, Italy, Apr. 2007. doi:10.1109/ROBOT.2007.363895.

M. Ribo, M. Brandner, and A. Pinz. A flexible software architecture for hybrid tracking. *Journal of Robotics Systems*, 21(2):53–62, 2004. doi:10.1002/rob.10124.

S. F. Schmidt. Application of state-space methods to navigation problems. *Advances in Control Systems*, 3:293–340, 1966.

T. B. Schön. *Estimation of Nonlinear Dynamic Systems – Theory and Applications*. Dissertations no 998, Linköping Studies in Science and Technology, Department of Electrical Engineering, Linköping University, Sweden, Feb. 2006.

J. Shi and C. Tomasi. Good features to track. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, June 1994. doi:10.1109/CVPR.1994.323794.

M. D. Shuster. A survey of attitude representations. *The Journal of the Astronautical Sciences*, 41(4):439–517, Oct. 1993.

J. Skoglund and M. Felsberg. Covariance estimation for SAD block matching. In *Proc. 15th Scandinavian Conference on Image Analysis*, pages 374–382, 2007. doi:10.1007/978-3-540-73040-8_38.

G. L. Smith, S. F. Schmidt, and L. A. McGee. Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle. Technical Report TR R-135, NASA, 1962.

K. H. Strobl and G. Hirzinger. Optimal hand-eye calibration. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4647–4653, Beijing, China, Oct. 2006. doi:10.1109/IROS.2006.282250.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. The MIT Press, Cambridge, MA, USA, 2005. ISBN 978-0-262-20162-9.

D. H. Titterton and J. L. Weston. *Strapdown inertial navigation technology*. IEE radar, sonar, navigation and avionics series. Peter Peregrinus Ltd., Stevenage, UK, 1997. ISBN 0863413587.

R. Y. Tsai and R. K. Lenz. A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5(3):345–358, June 1989. doi:10.1109/70.34770.

B. Williams, P. Smith, and I. Reid. Automatic relocalisation for a single-camera simultaneous localisation and mapping system. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2784–2790, Roma, Italy, Apr. 2007. doi:10.1109/ROBOT.2007.363893.

O. J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, Aug. 2007.

Xsens Motion Technologies, 2008. URL `http://www.xsens.com/`. Accessed April 2nd, 2008.

Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov. 2000. doi:10.1109/34.888718.

A

Quaternion preliminaries

This appendix provides a very short introduction to quaternions and their properties. Only the most basic operations are stated, without proof. For more details, see e.g., Kuipers (1999); Hamilton (1844).

A.1 Operations and properties

A quaternion $q \in \mathbb{R}^4$ is a 4-tuple of real numbers is denoted by $q = (q_0, q_1, q_2, q_3)$. Alternatively it is denoted by $q = (q_0, \boldsymbol{q})$, where q_0 is called the scalar part and $\boldsymbol{q}$ the vector part of a quaternion. Special quaternions groups are $\mathcal{Q}_s = \{q \in \mathbb{R}^4 : \boldsymbol{q} = \boldsymbol{0}\}$, $\mathcal{Q}_v = \{q \in \mathbb{R}^4 : q_0 = 0\}$ and $\mathcal{Q}_1 = \{q \in \mathbb{R}^4 : \|q\| = 1\}$.

For quaternions the following operators are defined:

addition	$p + q \triangleq (p_0 + q_0, \boldsymbol{p} + \boldsymbol{q}),$	(A.1)
multiplication	$p \odot q \triangleq (p_0 q_0 - \boldsymbol{p} \cdot \boldsymbol{q}, p_0 \boldsymbol{q} + q_0 \boldsymbol{p} + \boldsymbol{p} \times \boldsymbol{q}),$	(A.2)
conjugation	$q^c \triangleq (q_0, -\boldsymbol{q}),$	(A.3)
norm	$\|q\| \triangleq (q_0^2 + \boldsymbol{q} \cdot \boldsymbol{q})^{\frac{1}{2}} = \sqrt{(q \odot q^c)_0}$	(A.4)
inverse	$q^{-1} \triangleq \|q\|^{-2} q^c,$	(A.5)
inner product	$p \cdot q \triangleq -\frac{1}{2}(p \odot q + q \odot p),$	(A.6)
cross product	$p \otimes q \triangleq \frac{1}{2}(p \odot q - q \odot p).$	(A.7)

Associative and distributive properties hold, but only additions are commutative. Multiplications do in general not commutate.

$$p + (q + r) = (p + q) + r,$$
$$p + q = q + p,$$

$$p \odot (q \odot r) = (p \odot q) \odot r,$$
$$p \odot (q + r) = p \odot q + p \odot r,$$
$$p \odot q \neq q \odot p.$$

An exception to this is scalar multiplication,

$$\lambda q = (\lambda, \mathbf{0}) \odot (q_0, \boldsymbol{q}) = (\lambda q_0, \lambda \boldsymbol{q}) = q\lambda.$$

Furthermore, the following properties are useful,

$$(p \odot q)^c = q^c \odot p^c,$$
$$(p \odot q)^{-1} = q^{-1} \odot p^{-1},$$
$$\|p \odot q\| = \|p\| \|q\|.$$

A.2 Exponential

The quaternion exponential is defined as a power series similar to the matrix exponential:

$$\exp q \triangleq \sum_{n=0}^{\infty} \frac{q^n}{n!}. \tag{A.8}$$

The quaternion exponential of a vector $v \in \mathcal{Q}_v$ is a special case, since $v = (0, \boldsymbol{v})$ and $v^2 \triangleq v \odot v = (0 \cdot 0 - \boldsymbol{v} \cdot \boldsymbol{v}, 0\boldsymbol{v} + 0\boldsymbol{v} + \boldsymbol{v} \times \boldsymbol{v}) = (-\|\boldsymbol{v}\|^2, \mathbf{0})$. Hence,

$$\begin{aligned}
\exp v &= \sum_{n=0}^{\infty} \frac{v^n}{n!} = \sum_{n=0}^{\infty} \frac{v^{2n}}{2n!} + \sum_{n=0}^{\infty} \frac{v^{2n+1}}{(2n+1)!} \\
&= \left(\sum_{n=0}^{\infty} (-1)^n \frac{\|\boldsymbol{v}\|^{2n}}{(2n)!}, \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|} \sum_{n=0}^{\infty} (-1)^n \frac{\|\boldsymbol{v}\|^{2n+1}}{(2n+1)!} \right) \\
&= \left(\cos \|\boldsymbol{v}\|, \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|} \sin \|\boldsymbol{v}\| \right).
\end{aligned} \tag{A.9}$$

The inverse operation $\log q$ is for unit quaternions $q = (q_0, \boldsymbol{q})$ given by

$$\log q = \frac{\boldsymbol{q}}{\|\boldsymbol{q}\|} \arccos q_0 \tag{A.10}$$

A.3 Matrix/vector notation

The multiplication of quaternions can also be written using matrix/vector notation:

$$\begin{aligned}
p \odot q &= (p_0 q_0 - \boldsymbol{p} \cdot \boldsymbol{q}, p_0 \boldsymbol{q} + q_0 \boldsymbol{p} + \boldsymbol{p} \times \boldsymbol{q}) \\
&= \underbrace{\begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix}}_{p_L} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \underbrace{\begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix}}_{q_R} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix},
\end{aligned} \tag{A.11}$$

where the left and right multiplication operators, $\cdot_L$, $\cdot_R$ have been introduced. Note that

$$(q^c)_L = q_L^T, \qquad\qquad (q^c)_R = q_R^T.$$

This notation turns out to be very useful in deriving various expressions, for instance,

$$\frac{\mathrm{d}}{\mathrm{d}\,p}(p \odot q) = \frac{\mathrm{d}}{\mathrm{d}\,p}(q_R p) = q_R, \qquad\qquad \frac{\mathrm{d}}{\mathrm{d}\,q}(p \odot q) = \frac{\mathrm{d}}{\mathrm{d}\,p}(p_L q) = p_L.$$

Furthermore, the Jacobians of the matrix operators have the following special structure

$$\mathrm{D}_q\, q_L = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{array}\right] = \begin{bmatrix} e_R^0 \\ e_R^1 \\ e_R^2 \\ e_R^3 \end{bmatrix}, \qquad \mathrm{D}_q\, q_R = \begin{bmatrix} e_L^0 \\ e_L^1 \\ e_L^2 \\ e_L^3 \end{bmatrix}, \tag{A.12}$$

where $\{e^i\}_{i=0}^4$ is the standard basis of $\mathbb{R}^4$.

B

Conversions

Orientations can be described with several interchangeable parameterizations. This appendix gives conversions between unit quaternions, rotation vectors, rotation matrices and Euler angles.

B.1 Rotation matrices

The rotation $x^a = q^{ab}x^b q^{ba}$ can also be written as $\boldsymbol{x}^a = R^{ab}\boldsymbol{x}^b$ with

$$R = \begin{bmatrix} 2q_0^2 + 2q_1^2 - 1 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & 2q_0^2 + 2q_2^2 - 1 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & 2q_0^2 + 2q_3^2 - 1 \end{bmatrix}, \tag{B.1}$$

where the annotation $_{ab}$ has been left out for readability.

B.2 Euler angles

The aerospace sequence – Euler angles $(\psi\theta\phi) \rightarrow (zyx)$ – yields the rotation matrix

$$\begin{aligned} R^{ab} &= R_\phi^x R_\theta^y R_\psi^z \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}. \end{aligned}$$

Solving for the angles gives

$$\psi^{ab} = \tan^{-1}\left(\frac{R_{12}}{R_{11}}\right) = \tan^{-1}\left(\frac{2q_1q_2 - 2q_0q_3}{2q_0^2 + 2q_1^2 - 1}\right), \tag{B.2a}$$

$$\theta^{ab} = -\sin^{-1}(R_{13}) = -\sin^{-1}(2q_1q_3 + 2q_0q_2), \tag{B.2b}$$

$$\phi^{ab} = \tan^{-1}\left(\frac{R_{23}}{R_{33}}\right) = \tan^{-1}\left(\frac{2q_2q_3 - 2q_0q_1}{2q_0^2 + 2q_3^2 - 1}\right). \tag{B.2c}$$

Here, the convention is to call ψ heading or yaw, θ elevation or pitch and ϕ bank or roll.

B.3 Rotation vector

A rotation around axis $\boldsymbol{n}$ by angle α has a rotation vector $\boldsymbol{e} \triangleq \alpha\boldsymbol{n}$. The conversion to and from a quaternion is given by

$$q^{ab} = \exp \tfrac{1}{2}\boldsymbol{e}^{ab}, \qquad \boldsymbol{e}^{ab} = 2\log q^{ab}, \tag{B.3}$$

with $\exp$ and $\log$ defined in Appendix A.2.

Licentiate Theses
Division of Automatic Control
Linköping University

P. Andersson: Adaptive Forgetting through Multiple Models and Adaptive Control of Car Dynamics. Thesis No. 15, 1983.
B. Wahlberg: On Model Simplification in System Identification. Thesis No. 47, 1985.
A. Isaksson: Identification of Time Varying Systems and Applications of System Identification to Signal Processing. Thesis No. 75, 1986.
G. Malmberg: A Study of Adaptive Control Missiles. Thesis No. 76, 1986.
S. Gunnarsson: On the Mean Square Error of Transfer Function Estimates with Applications to Control. Thesis No. 90, 1986.
M. Viberg: On the Adaptive Array Problem. Thesis No. 117, 1987.
K. Ståhl: On the Frequency Domain Analysis of Nonlinear Systems. Thesis No. 137, 1988.
A. Skeppstedt: Construction of Composite Models from Large Data-Sets. Thesis No. 149, 1988.
P. A. J. Nagy: MaMiS: A Programming Environment for Numeric/Symbolic Data Processing. Thesis No. 153, 1988.
K. Forsman: Applications of Constructive Algebra to Control Problems. Thesis No. 231, 1990.
I. Klein: Planning for a Class of Sequential Control Problems. Thesis No. 234, 1990.
F. Gustafsson: Optimal Segmentation of Linear Regression Parameters. Thesis No. 246, 1990.
H. Hjalmarsson: On Estimation of Model Quality in System Identification. Thesis No. 251, 1990.
S. Andersson: Sensor Array Processing; Application to Mobile Communication Systems and Dimension Reduction. Thesis No. 255, 1990.
K. Wang Chen: Observability and Invertibility of Nonlinear Systems: A Differential Algebraic Approach. Thesis No. 282, 1991.
J. Sjöberg: Regularization Issues in Neural Network Models of Dynamical Systems. Thesis No. 366, 1993.
P. Pucar: Segmentation of Laser Range Radar Images Using Hidden Markov Field Models. Thesis No. 403, 1993.
H. Fortell: Volterra and Algebraic Approaches to the Zero Dynamics. Thesis No. 438, 1994.
T. McKelvey: On State-Space Models in System Identification. Thesis No. 447, 1994.
T. Andersson: Concepts and Algorithms for Non-Linear System Identifiability. Thesis No. 448, 1994.
P. Lindskog: Algorithms and Tools for System Identification Using Prior Knowledge. Thesis No. 456, 1994.
J. Plantin: Algebraic Methods for Verification and Control of Discrete Event Dynamic Systems. Thesis No. 501, 1995.
J. Gunnarsson: On Modeling of Discrete Event Dynamic Systems, Using Symbolic Algebraic Methods. Thesis No. 502, 1995.
A. Ericsson: Fast Power Control to Counteract Rayleigh Fading in Cellular Radio Systems. Thesis No. 527, 1995.
M. Jirstrand: Algebraic Methods for Modeling and Design in Control. Thesis No. 540, 1996.
K. Edström: Simulation of Mode Switching Systems Using Switched Bond Graphs. Thesis No. 586, 1996.
J. Palmqvist: On Integrity Monitoring of Integrated Navigation Systems. Thesis No. 600, 1997.
A. Stenman: Just-in-Time Models with Applications to Dynamical Systems. Thesis No. 601, 1997.
M. Andersson: Experimental Design and Updating of Finite Element Models. Thesis No. 611, 1997.
U. Forssell: Properties and Usage of Closed-Loop Identification Methods. Thesis No. 641, 1997.

M. Larsson: On Modeling and Diagnosis of Discrete Event Dynamic systems. Thesis No. 648, 1997.

N. Bergman: Bayesian Inference in Terrain Navigation. Thesis No. 649, 1997.

V. Einarsson: On Verification of Switched Systems Using Abstractions. Thesis No. 705, 1998.

J. Blom, F. Gunnarsson: Power Control in Cellular Radio Systems. Thesis No. 706, 1998.

P. Spångéus: Hybrid Control using LP and LMI methods – Some Applications. Thesis No. 724, 1998.

M. Norrlöf: On Analysis and Implementation of Iterative Learning Control. Thesis No. 727, 1998.

A. Hagenblad: Aspects of the Identification of Wiener Models. Thesis No. 793, 1999.

F. Tjärnström: Quality Estimation of Approximate Models. Thesis No. 810, 2000.

C. Carlsson: Vehicle Size and Orientation Estimation Using Geometric Fitting. Thesis No. 840, 2000.

J. Löfberg: Linear Model Predictive Control: Stability and Robustness. Thesis No. 866, 2001.

O. Härkegård: Flight Control Design Using Backstepping. Thesis No. 875, 2001.

J. Elbornsson: Equalization of Distortion in A/D Converters. Thesis No. 883, 2001.

J. Roll: Robust Verification and Identification of Piecewise Affine Systems. Thesis No. 899, 2001.

I. Lind: Regressor Selection in System Identification using ANOVA. Thesis No. 921, 2001.

R. Karlsson: Simulation Based Methods for Target Tracking. Thesis No. 930, 2002.

P.-J. Nordlund: Sequential Monte Carlo Filters and Integrated Navigation. Thesis No. 945, 2002.

M. Östring: Identification, Diagnosis, and Control of a Flexible Robot Arm. Thesis No. 948, 2002.

C. Olsson: Active Engine Vibration Isolation using Feedback Control. Thesis No. 968, 2002.

J. Jansson: Tracking and Decision Making for Automotive Collision Avoidance. Thesis No. 965, 2002.

N. Persson: Event Based Sampling with Application to Spectral Estimation. Thesis No. 981, 2002.

D. Lindgren: Subspace Selection Techniques for Classification Problems. Thesis No. 995, 2002.

E. Geijer Lundin: Uplink Load in CDMA Cellular Systems. Thesis No. 1045, 2003.

M. Enqvist: Some Results on Linear Models of Nonlinear Systems. Thesis No. 1046, 2003.

T. Schön: On Computational Methods for Nonlinear Estimation. Thesis No. 1047, 2003.

F. Gunnarsson: On Modeling and Control of Network Queue Dynamics. Thesis No. 1048, 2003.

S. Björklund: A Survey and Comparison of Time-Delay Estimation Methods in Linear Systems. Thesis No. 1061, 2003.

M. Gerdin: Parameter Estimation in Linear Descriptor Systems. Thesis No. 1085, 2004.

A. Eidehall: An Automotive Lane Guidance System. Thesis No. 1122, 2004.

E. Wernholt: On Multivariable and Nonlinear Identification of Industrial Robots. Thesis No. 1131, 2004.

J. Gillberg: Methods for Frequency Domain Estimation of Continuous-Time Models. Thesis No. 1133, 2004.

G. Hendeby: Fundamental Estimation and Detection Limits in Linear Non-Gaussian Systems. Thesis No. 1199, 2005.

D. Axehill: Applications of Integer Quadratic Programming in Control and Communication. Thesis No. 1218, 2005.

J. Sjöberg: Some Results On Optimal Control for Nonlinear Descriptor Systems. Thesis No. 1227, 2006.

D. Törnqvist: Statistical Fault Detection with Applications to IMU Disturbances. Thesis No. 1258, 2006.

H. Tidefelt: Structural algorithms and perturbations in differential-algebraic equations. Thesis No. 1318, 2007.

S. Moberg: On Modeling and Control of Flexible Manipulators. Thesis No. 1336, 2007.

J. Wallén: On Kinematic Modelling and Iterative Learning Control of Industrial Robots. Thesis No. 1343, 2008.

J. Harju Johansson: A Structure Utilizing Inexact Primal-Dual Interior-Point Method for Analysis of Linear Differential Inclusions. Thesis No. 1367, 2008.