Towards Visual-Inertial SLAM for Mobile Augmented Reality

Vom Fachbereich Informatik
der Technischen Universität Kaiserslautern
zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
(Dr.-Ing.)
genehmigte Dissertation von

Gabriele Bleser

Dekan:	Prof. Dr. Karsten Berns
Vorsitzender der Prüfungskomission:	Prof. Dr. Karsten Berns
Erster Berichterstatter:	Prof. Dr. Didier Stricker
Zweiter Berichterstatter:	Prof. Dr. Fredrik Gustafsson
Tag der wissenschaftlichen Aussprache:	24. März 2009

To my family

Abstract

The basic idea of augmented reality is to augment the view of a user or camera with virtual objects. Real-time camera tracking is an enabling technology for augmented reality. Besides the high estimation precision that is needed to allow for pixel accurate augmentations, mobile augmented reality applications impose further requirements on the tracking method. One important aspect is robustness in the presence of quick and erratic camera motions, which are typical for a handheld or head-mounted camera.

This thesis investigates robustness and accuracy of real-time markerless camera tracking for mobile augmented reality applications considering both known and unknown environments of different complexity. The major solution strategies are: visual-inertial sensor fusion, decoupling of pose and structure estimation and unification of computer vision and recursive filtering techniques.

First, a model-based camera tracking system that fuses visual and inertial measurements in the extended Kalman filter is developed. It uses an affine illumination invariant image processing method that exploits the pose prediction obtained from the sensor fusion algorithm and a textured CAD model of the environment to predict the appearances of corner features in the camera images. In several experiments, the system is demonstrated to work robustly in realistic environments of different complexity, under varying light conditions, fast and erratic camera motions and even short periods without visible features. Compared to vision-only tracking, the system shows less jitter and the computational costs are reduced both due to an accurate prediction of the feature appearances and locations in the images and a reduced demand on features in general.

Tracking in partially known environments is addressed by developing a vision-only system, which requires minimal pre-knowledge about the structure of the target scene and derives 3D information online. The system combines robust and efficient sequential structure from motion methods with a simplified stochastic model and recursive localisation of 3D point features. It scales with the size of the map, since pose and structure estimation are decoupled. Compared to an ordinary sequential structure from motion system, the developed method provides higher accuracy in both camera and feature localisation and significantly less drift. This is demonstrated in different experiments based on simulated data and in realistic mid-scale environments.

Based on these results, a conceptual solution for visual-inertial simultaneous localisation and mapping in large-scale environments is developed. The idea is to combine the marginalised particle filter for visual-inertial pose estimation with undelayed feature initialisation and localisation on a per-particle basis. The essential algorithms for pose and structure estimation are developed and a proof-of-concept implementation is evaluated in a simple test environment, demonstrating that the novel tracking strategy unifies and enhances the previous developments.

Acknowledgments

During my work with this thesis I have been supported, in one way or another, by many wonderful people. Some of the most important ones I want to mention here (in chronological order). I am very grateful towards Prof. Stefan Müller (University Koblenz), who paved the way for me to become a member of the Department for Virtual and Augmented Reality in Darmstadt, one of the greatest groups ever! The head of this department has been the most important person on my way. I want to express my deep thankfulness towards Prof. Didier Stricker, who has been my adviser during the past five years (starting with my diploma thesis). He has given me the opportunity to work in an excellent research environment and is a never-ending source of inspiration. Didier, thanks for always taking time for a discussion and for giving me invaluable advices, in business as well as in private life. In short: Thanks for encouraging me and bringing the best out of me! I am very grateful to all my colleagues, who have been cooperative, motivating, simply great, more than words can say. An explicit thank you to Mario and Yulian, for the shared debugging sessions to find the (rare) bugs in my code and for trying to teach me how to stay cool (not that I ever managed), to Harald — in my view the effortless superiority in person — for the fruitful discussions and common research projects, and to Renate, for her motherly manner, the pleasant talks in the morning and the support, when I found life tough!

During my work with this thesis, I have been involved in the European research project MATRIS and I would like to thank the partners, Linköping University, BBC R&D, Christian-Albrechts University and Xsens Technologies B.V. for the interesting discussions and common work. Most importantly among those, I want to thank the Automatic Control group in Linköping, where I had the pleasure to spend three month (thanks Thomas!). In the first place, I am very grateful that I met Prof. Fredrik Gustafsson, who co-advised my thesis, Dr. Thomas Schön and Jeroen Hol in the MATRIS project. They introduced me into their area of expertise, which turned out to be a watershed in my research and resulted in an important part of my thesis. Thank you for patiently answering all my questions, for listening to my ideas and sharing your ideas with me! I appreciate our collaboration and hope we can continue! Moreover, I want to thank my office and flat mate Jesica and all members of the Automatic Control group for making my stay in Linköping a very pleasant one and for really making me feel at home. The time there not only provided me with many interesting experiences, it also let me grow as a person. A special thank you to Dr. Gustaf Hendeby, who was a great support during finalising my thesis. Thank you for all the constructive discussions, for proof-reading my thesis and for helping me out in any situation, ranging from LaTeX problems to the preparation of my viva! I am looking forward to welcome you as a colleague in Germany!

Finally I want to express my deep gratitude towards my family and friends, most importantly, my love Christian, my parents Marlies and Karl-Heinz, my "second parents" Agnes and Winfried and my best friend Steffi. Thank you for the great support and understanding during the past years, for being sources of calm and for gently bringing me back to real life, when I was caught in augmented reality (at least for trying!). I couldn't have done it without you!

Contents

1

Introduction

Augmented reality is a growing field with many diverse applications ranging from TV and film production to industrial maintenance, medicine, education, prototyping, entertainment and immersive games. The basic idea is to insert virtual objects (augmentations) into a scene, either by displaying them in a see-through head-mounted display, or by superimposing them on an image captured by the camera. This is illustrated in Figure 1.1. Depending on the application, the inserted objects might be virtual characters in a TV or film production, instructions on how to repair a machine, or a reconstruction of an archaeological site. These and further

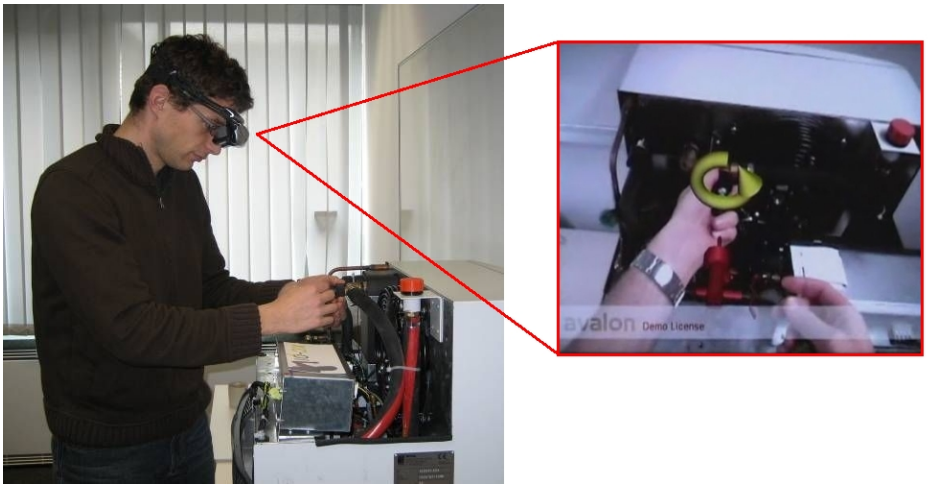


Figure 1.1: *Maintenance with augmented reality assistance: The user wears a head-mounted display with a camera attached. The picture on the right side shows what the user sees. Virtual arrows and tools are superimposed on the camera image to instruct the user while repairing the machine.*

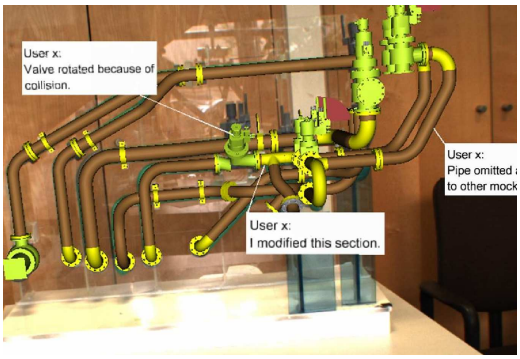
(a) Maintenance.

(b) Tied-to-pitch graphics for sport.

(c) Cultural heritage.

(d) Design review.

(e) Virtual broadcasting studio.



(f) Variance analysis.

Figure 1.2: *Augmented Reality applications.*

examples are illustrated in Figure 1.2. For the effects to be believable, the virtual objects must fit seamlessly into the real scene or the image of the scene. One important part in achieving this is to show them in the correct perspective, which depends on how the camera or user is viewing the scene. As the view is changing with each movement, it requires efficient algorithms to recalculate it constantly. This process is referred to as real-time camera tracking.

1.1 Motivation

Real-time camera tracking, *i.e.* the accurate estimation of the camera position and orientation in real-time, is an enabling technology for augmented reality. Deriving the camera pose from its images is referred to as vision-based or visual tracking. If no special infrastructure is needed in the environment, the method is called markerless. Hybrid tracking systems are those that exploit information from additional sensors in order to improve their performance. Besides the high estimation precision that is needed to allow for pixel accurate augmentations in real-time, mobile augmented reality applications put further requirements on the tracking method. Two important areas are:

Portability: The hardware setup needs to be portable, *i.e.* consist of a small camera, if applicable, small additional sensors and a mobile computer.

Robustness: In a mobile application, the camera is either head-mounted or handheld. Both implies erratic motions, which need to be handled by the tracking method. Moreover, the light conditions are basically uncontrolled. This must be taken into account when analysing the camera images.

Depending on how much information about the target environment is available, a camera tracking system — as modelled in this thesis — consists of up to three major building blocks: One building block, image processing, is concerned with searching the camera images for known structures in the environment. A second building block, pose estimation, is responsible for determining the camera pose from the established correspondences. Assuming that a 3D model of the target scene is known, these two building blocks make up a model-based camera tracking system. If no or not enough information about the target scene is available, a third building block, structure estimation, is required to derive additional 3D structure online. The scene structure is represented by a sparse 3D point cloud, which is the most common representation used in computer vision. A hybrid tracking system emerges when using additional sensors for pose estimation. In this thesis, visual-inertial sensor fusion is investigated. The small and low cost inertial sensors that are nowadays available are easily attached to a camera and thus provide a portable tracking setup. Moreover, inertial sensors offer complementary properties to the camera sensor, which allow for beneficial sensor fusion. Figure 1.3 gives a simplified conceptual illustration of a camera tracking system as it is modelled in this thesis. Note that pose and structure estimation are decoupled in this illustration.

Both the simultaneous estimation of pose and structure and the subproblem of pose estimation relative to a known scene model, are prominent research topics in two different communities: computer vision and robotics. The computer vision terms *structure from motion* and *model-based tracking* denote basically the same problem as *localisation* and *simultaneous localisation and mapping (SLAM)* in robotics. However, both communities have different strategies of posing and solving these estimation problems.

Computer vision methods are traditionally vision-only, *i.e.* they use the camera as single sensor. Moreover, they accentuate robustness — with respect to incorrect correspondences

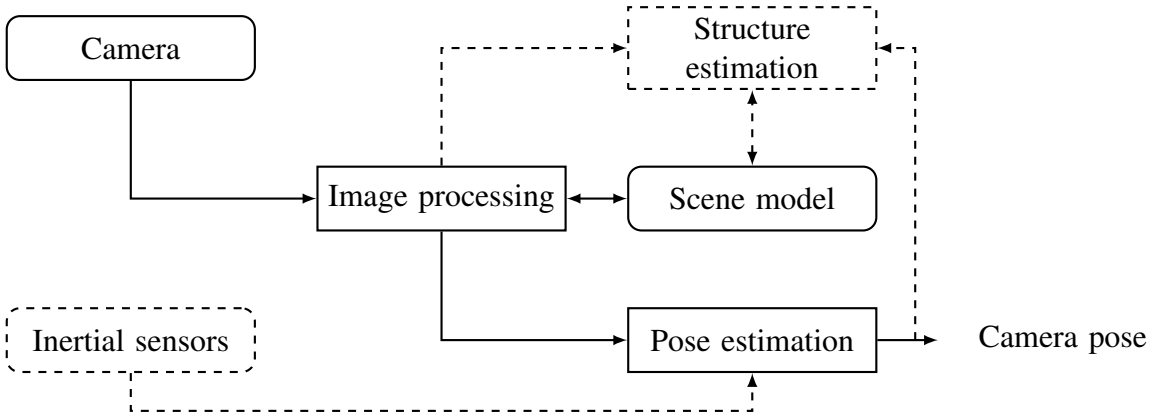


Figure 1.3: *Conceptual illustration of different occurrences of a camera tracking system as they appear in this thesis: Hardware and data are separated from the building blocks by rounded corners. The two building blocks image processing and pose estimation make up a model-based tracking system, implying that the scene model is given completely. By adding a third building block, structure estimation, the scene model can be learned online. Inertial sensors can be used to aid the vision-based pose estimation.*

— and accuracy. Structure from motion methods traditionally aim at obtaining an accurate estimate of the complete camera trajectory and scene structure. This is typically achieved by performing an offline batch optimisation, called bundle adjustment. Today, real-time capable sequential approaches are encountered in literature, too.

In robotics, measurements are typically combined from a number of sensors mounted to a robot, and recursive filters are used to perform sensor fusion. In a recursive scheme, rather than aiming at an accurate estimate of the complete trajectory, the goal is to obtain an estimate of the momentary pose — in case of SLAM along with the map — and a representation of the confidence in this estimate. Recursive techniques are typically less accurate and robust. However, they offer the possibility to improve the current estimate with additional information, without the need to store or reprocess previous estimates or measurements. The classical solution to SLAM is to estimate the trajectory and the map jointly in one extended Kalman filter (EKF). Since pose and structure estimation are performed at the same time, this solution does not fit into the conceptual illustration given in Figure 1.3. Another more recent approach to SLAM that scales with the size of the map and decouples pose and structure estimation in the fashion of sequential structure from motion is called FastSLAM. In FastSLAM, the pose is represented in a particle cloud and the map is estimated on a per-particle basis.

Nowadays, the computer vision and the robotics community are closing in on each other. The use of cameras is widespread in robotics and recursive filters and sensor fusion are frequently exploited concepts in computer vision. This thesis bridges a gap between both communities by pointing out the benefits of unifying their concepts.

1.2 Problem Formulation and Approach

The aim of this thesis is to develop solutions for real-time markerless camera tracking that:

- meet the requirements of mobile augmented reality applications as described in Section 1.1. In particular, some fundamental limitations of vision-only camera tracking are addressed: missing support for erratic and quick camera motions, missing robustness against occlusion, high demand on camera measurements (correspondences) and computational intensity of markerless image processing.

- address both known and unknown or partially known target scenes. In particular, the problem of drift, *i.e.* an accumulating error in the camera and structure registration, which is a notorious problem of sequential structure from motion, is addressed.

- have potentiality for large-scale environments. In the context of simultaneous estimation of pose and structure, this requires methods that scale with the size of the map, which is allowed for by decoupling both estimation processes as in Figure 1.3.

In order to achieve this, suitable algorithms for the three major building blocks shown in Figure 1.3 are developed. The major solution strategy is to unify concepts of computer vision, *i.e.* robust and precise estimation, decoupling of pose and structure estimation and markerless image processing, and robotics, *i.e.* visual-inertial sensor fusion, recursive filtering and confidence measures.

1.3 Contributions

The main contributions of this thesis are:

- A model-based real-time camera tracking system that fuses vision-based and inertial measurements in the extended Kalman filter to overcome some fundamental limitations of vision-only camera tracking, and the testing of this system in realistic environments of different complexity. Within this:

  - The derivation of an affine illumination invariant image processing approach based on the analysis-by-synthesis technique that can exploit an accurate pose prediction for increased computational efficiency.
  - The derivation of different nonlinear state-space models for position and orientation estimation based on camera and inertial measurements and the experimental evaluation of these models in the developed system under controlled conditions. In particular, the incorporation of accelerometer measurements is investigated.

  The presentation is based on the journal paper:

G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. *Computer & Graphics*, 2008. accepted.

and the conference papers:

G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. In *IEEE Virtual Reality Conference (VR)*, Reno, Nevada, March 2008.

G. Bleser, C. Wohleber, and D. Stricker. Fast and Stable Tracking for AR fusing Video and Inertial Sensor Data. In *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 109–115, Plzen, CZ, February 2006.

J. Chandaria, G. Thomas, B. Bartczak, K. Koeser, R. Koch, M. Becker, G. Bleser, D. Stricker, C. Wohlleber, M. Felsberg, J. Hol, T. Schön, J. Skoglund, P. Slycke, and S. Smeitz. Real-time Camera Tracking in the Matris Project. In *International Broadcasting Convention (IBC)*, pages 321–328, Amsterdam, Netherlands, September 2006.

G. Bleser, Y. Pastarmov, and D. Stricker. Real-time 3D Camera Tracking for Industrial Augmented Reality Applications. In *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 47–54, Plzen, CZ, February 2005.

- A vision-only system for real-time simultaneous localisation and mapping that unifies robust computer vision techniques and recursive filtering in order to reduce the drift in the camera pose and structure registration, and the experimental evaluation of this system on simulated data and in realistic environments of medium complexity. Within this:
  - A robust pose estimation technique that incorporates both correspondences between the scene model and the image and across images, and the uncertainties in these measurements.
  - Robust methods for the initialisation and recursive estimation of 3D point landmarks (features).
  - A measure of feature quality.

  These results are based on the journal paper:

  G. Bleser, M. Becker, and D. Stricker. Real-time vision-based tracking and reconstruction. *Journal of Real-Time Image Processing*, 2(2–3):161–175, November 2007.

and the conference papers:

> G. Bleser, H. Wuest, and D. Stricker. Online Camera Pose Estimation in Partially Known and Dynamic Scenes. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 56–65, Santa Barabara, CA, October 2006.
>
> M. Becker, G. Bleser, A. Pagani, Y. Pastarmov, D. Stricker, F. Vial, J. Weidenhausen, C. Wohlleber, and H. Wuest. Visual Tracking for Augmented Reality: No Universal Solution but Many Powerful Building Blocks. In *Virtuelle und Erweiterte Realität: 2. Workshop der GI-Fachgruppe VR/AR*, pages 107–118, 2005.
>
> M. Zoellner, D. Stricker, G. Bleser, and Y. Pastarmov. iTACITUS — Novel Interaction and Tracking Paradigms for Mobile AR. In *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, pages 110–117, Brighton, UK, October 2007.
>
> M. Becker, G. Bleser, A. Pagani, D. Stricker, and H. Wuest. An Architecture for Prototyping and Application Development of Visual Tracking Systems. In *IEEE 3DTV-Conference: Capture, Transmission and Display of 3D Video*, 2007.

- A conceptual solution for a visual-inertial SLAM system with increased potentiality for large-scale environments, a proof-of-concept demonstration in a simple test setup and a comparison with the previously developed visual SLAM system. Within this:
  - The derivation of a nonlinear state-space model with linear Gaussian substructure for position and orientation estimation based on camera and inertial measurements.
  - A novel marginalised particle filter algorithm for real-time visual-inertial pose estimation with increased robustness against erratic camera motions.
  - An experimental evaluation of this algorithm in a simple test setup and a comparison with other visual-inertial fusion strategies based upon the extended Kalman filter, the particle filter and the regular marginalised particle filter.

  The presentation is based on the conference paper:

  > G. Bleser and D. Stricker. Using the Marginalised Particle Filter for Real-Time Visual-Inertial Sensor Fusion. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Cambridge, UK, September 2008.

  Previously unpublished results are:

  - Undelayed initialisation and recursive estimation of 3D point landmarks on a per-particle basis.

1.4 Thesis Outline

This thesis is organised in the following way: Chapter 2 introduces the notation and gives the background for understanding this thesis. It first introduces the operating principles, mathematical models and measurements of cameras and inertial sensors and then proceeds with the fundamental estimation techniques in the area of computer vision and recursive filtering providing a brief review of the literature.

Chapter 3 is devoted to the development and experimental evaluation of a hybrid model-based camera tracking system, which fuses measurements from a camera and low cost inertial sensors in real-time. Efficient markerless image processing and visual-inertial sensor fusion in the extended Kalman filter are addressed.

Chapters 4 and 5 investigate camera tracking in partially known environments. Chapter 4 treats the development and experimental evaluation of a vision-only sequential structure from motion system. It addresses accurate and robust pose and structure estimation in a simplified stochastic framework.

Chapter 5 combines and extends the results of Chapters 3 and 4. It presents a conceptual solution for a visual-inertial SLAM system with increased potentiality for large-scale environments. Real-time visual-inertial pose estimation in the marginalised particle filter and per-particle structure estimation with undelayed feature initialisation are addressed. Moreover, a proof-of-concept is demonstrated in a simple test setup.

Finally, Chapter 6 concludes the thesis and gives suggestions for future work.

2

Background

This chapter introduces the notation used in this thesis and gives the background for understanding it. Sections 2.1 and 2.2 are devoted to sensors. The operating principles, mathematical models and measurements of cameras and camera *IMU (inertial measurement unit)* systems are discussed. Sections 2.3 and 2.4 introduce fundamental estimation techniques in the area of computer vision and recursive filtering that play an important role in this thesis.

2.1 Cameras

A digital camera consists of two functional parts: an optical system (objective, lens system) and an image sensor, either using CCD (charge-coupled device) or CMOS (complementary metal oxide semiconductor) technology.

The optical system projects incoming light onto the surface of the image sensor. Several types of lenses exist, for instance perspective lenses, macro lenses, telecentric lenses and fish-eye lenses. This thesis uses standard perspective lenses, which are the most commonly used in computer vision. From a physical point of view, the functionality of a lens system is extremely complex. However, ignoring effects such as focus and lens thickness, the refraction of light can be modelled as an ideal perspective projection — given by the *pinhole camera model* [12, 13] — combined with a *distortion* in the image plane.

The image sensor converts the incident light to a digital image (digitalisation). Digital images are typically indexed in terms of pixels. The relationship between the metric coordinate system of the image sensor and the pixel array is defined by the sensor geometry and its sampling characteristics. This information is contained in the *intrinsic camera parameters*. The distortion coefficients and intrinsic camera parameters are fixed for a given camera. The process of estimating those parameters is called *camera calibration*.

The *extrinsic camera parameters* describe the position and orientation of the camera in a

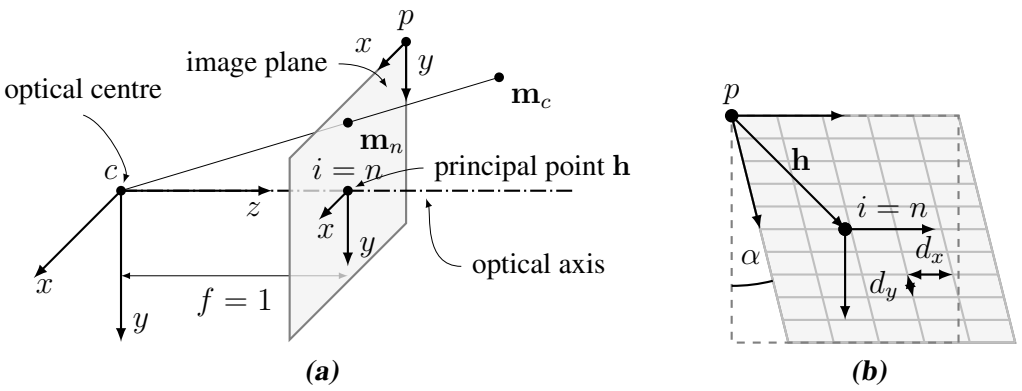


Figure 2.1: *From camera to pixel coordinates: (a) illustrates the normalised pinhole camera model with focal length $f = 1$. Note that the image plane is placed in front of the camera. (b) outlines the relation between the normalised image frame n and the pixel frame p with non-square and non-rectangular pixels. As distortions are neglected here $i = n$.*

fixed reference frame. These parameters change, when the camera moves, and the purpose of *camera tracking* or *camera pose estimation* is to estimate them.

Section 2.1.1 formalises the image formation process as described above. The camera calibration procedure is outlined in Section 2.1.2. Section 2.1.3 introduces geometric constraints that can be used to reconstruct 3D geometry from the camera images and Section 2.1.4 is concerned with the image processing aspects of this problem.

2.1.1 Camera Geometry

This section introduces the coordinate systems and transformations needed to obtain a mathematical model of the image formation process. As conclusion, the complete chain of transformations that allows for predicting the pixel location of each 3D point in space is summarised in Section 2.1.1.5.

2.1.1.1 Normalised Pinhole Camera Model

An illustration of the normalised pinhole camera model is given in Figure 2.1(a). Under the pinhole model, the camera is represented by a 2D *image plane* and a 3D point called the *centre of projection* or the *optical centre*. The shortest distance between the optical centre and the image plane is the *focal length* of the camera denoted f. The focal length is specified in terms of metric units. A point in 3D space is projected onto the image plane by drawing a ray from the 3D point towards the centre of projection. The intersection between this ray and the image plane represents the image of the 3D point. To obtain a mathematical representation, a 3D coordinate system, the *camera coordinate system* c, is introduced. Its origin, the *camera*

centre, coincides with the optical centre and its z-axis, also known as the *optical axis* or the *principal axis*, is orthogonal to the image plane. The principal axis meets the image plane in the *principal point* $\mathbf{h}$, which is the origin of the *image coordinate system* i. Under the normalised pinhole model, with focal length $f = 1$, the image frame coincides with the *normalised image coordinate system* n. The mapping of a point $\mathbf{m}_c = [x_c, y_c, z_c]^T$, expressed in the camera frame, to a point $\mathbf{m}_n = [x_n, y_n]^T$, expressed in the normalised image frame, is:

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \frac{1}{z_c} \begin{bmatrix} x_c \\ y_c \end{bmatrix}. \tag{2.1a}$$

By introducing homogeneous coordinates, (2.1a) can be written as a matrix multiplication:

$$\begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \propto z_c \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}. \tag{2.1b}$$

Equations (2.1a) and (2.1b) denote the normalised pinhole projection.

2.1.1.2 Distortion

The normalised pinhole projection is an ideal projection. However, real — especially low-cost and wide-angle — objectives introduce distortion. This is most visible at the border of the images. An example is shown in Figure 2.2(a). Several distortion models are encountered in literature. In this thesis, the distortion model of [14] is used. It accounts for radial and tangential (decentering) distortion. The distorted image coordinates $\mathbf{m}_d = [x_d, y_d]^T$ are expressed as a function of the normalised image coordinates $\mathbf{m}_n = [x_n, y_n]^T$, the distance to the principal point $r = \|\mathbf{m}_n\|$ and the polynomial coefficients $\kappa_1, \ldots, \kappa_5$:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \underbrace{(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \begin{bmatrix} x_n \\ y_n \end{bmatrix}}_{\text{radial distortion}} + \underbrace{\begin{bmatrix} 2\kappa_4 x_n y_n + \kappa_5 (r^2 + 2x_n^2) \\ \kappa_4 (r^2 + 2y_n^2) + 2\kappa_5 x_n y_n \end{bmatrix}}_{\text{tangential distortion}}. \tag{2.2}$$

Typical distortions are dominated by the coefficients κ_1 and κ_2. The appropriate number of coefficients depends on the camera objective and the accuracy required by the practical application. In many cases $\kappa_3, \ldots, \kappa_5$ are assumed zero.

Equation (2.2) can be used to undistort a whole image by sampling in the ideal image domain, computing the distorted coordinates and interpolating in the distorted image domain (resampling). An example is shown in Figure 2.2(b). In order to undistort single coordinates, the inverse function of (2.2) is needed. An analytical solution exists only, if $\kappa_2, \ldots, \kappa_5$ are all assumed zero [15]. However, the inverse mapping of (2.2) can be found numerically by applying Newton-Raphson iterations with the distorted coordinates as initial guess.

(a) Distorted image. *(b) Corrected image.*

Figure 2.2: *Image Distortion: as can be seen from the curved edges, (a) suffers from severe distortions. This image originates from a wide-angle firewire camera with focal length $f = 2.1$ mm. The polynomial coefficients for this camera are $\kappa_1 = -0.24$, $\kappa_2 = 0.06$ and $\kappa_3 = -0.01$. The tangential distortion coefficients are assumed zero. (b) shows the undistorted image using (2.2) with the given coefficients. Note that the field of view is narrowed by the resampling.*

2.1.1.3 Intrinsic Camera Parameters

The coordinates of a digital image are typically specified in pixels indexed from the top left corner (*cf.* Figure 2.1). In order to describe a projected point in the *pixel coordinate system* p, the intrinsic or internal camera parameters, as illustrated in Figure 2.1(b), have to be account for: d_x and d_y denote the width and height, respectively, of a sensor pixel in metric units. $f_x = f/d_x$ and $f_y = f/d_y$ describe the focal length in x- and y-direction, respectively, in pixels per metric unit. $\mathbf{h} = [h_x, h_y]^T$ denotes the principal point in pixels. $s_\alpha = f_x \tan\alpha$ accounts for non-rectangular sensor pixels. The angle α describes the skew of the sensor axis in terms of the deviation from the right angle. s_α is also specified in pixels per metric unit. The intrinsic parameters are used to compose the affine transformation matrix K, which describes the mapping from the *distorted image coordinate system* d to the pixel coordinate system p:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s_\alpha & h_x \\ 0 & f_y & h_y \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix}. \tag{2.3a}$$

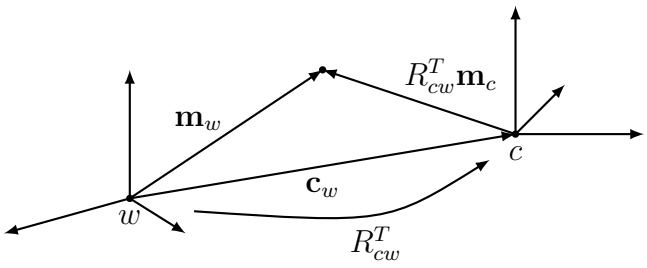


Figure 2.3: *Relation between the camera and the world coordinate system.*

In practical applications it is common to assume that $\alpha = 0$. The inverse mapping of (2.3a) is:

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{1}{f_x} & \frac{-s_\alpha}{f_x f_y} & \frac{-c_x f_y + c_y s_\alpha}{f_x f_y} \\ 0 & \frac{1}{f_y} & \frac{-c_y}{f_y} \\ 0 & 0 & 1 \end{bmatrix}}_{K^{-1}} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}. \tag{2.3b}$$

This can be used to normalise the pixel coordinates, once the camera is calibrated (*cf.* Section 2.1.2).

2.1.1.4 Extrinsic Camera Parameters

3D points are in general not expressed in the moving camera frame c but in a fixed reference frame, the *object* or *world coordinate system* w. The relation between those coordinate systems is given by a rigid transformation consisting of a rotation R_{cw} and a translation $\mathbf{c}_w$ called the *extrinsic camera parameters* or the *camera pose*. This is illustrated in Figure 2.3. The rotation R_{cw}^T can be interpreted as the camera orientation and the translation $\mathbf{c}_w$ as the position of the camera in the world frame. The mapping of a point $\mathbf{m}_w$ expressed in the world frame to a point $\mathbf{m}_c$ expressed in the camera frame is:

$$\mathbf{m}_c = R_{cw}(\mathbf{m}_w - \mathbf{c}_w). \tag{2.4a}$$

This equation can be restated to another commonly used form:

$$\mathbf{m}_c = R_{cw}\mathbf{m}_w - R_{cw}\mathbf{c}_w = R_{cw}\mathbf{m}_w + \mathbf{w}_c, \tag{2.4b}$$

where the rotation is applied first and where $\mathbf{w}_c = -R_{cw}\mathbf{c}_w$ is the origin of the world coordinate system in the camera frame. By introducing homogeneous coordinates, (2.4b) can be expressed as a linear operation:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{cw} & \mathbf{w}_c \\ \mathbf{0}_3^T & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}. \tag{2.4c}$$

The inverse mapping of (2.4a) and (2.4b), respectively, is:

$$\mathbf{m}_w = R_{cw}^T \mathbf{m}_c + \mathbf{c}_w \tag{2.4d}$$
$$= R_{cw}^T (\mathbf{m}_c - \mathbf{w}_c). \tag{2.4e}$$

The camera orientation is here given as a rotation matrix R, as matrices have the power to transform vectors from one coordinate frame to another. However, a rotation matrix has nine parameters, whereat a 3D rotation has only *three degrees of freedom (DOF)*. Many alternative ways exist to represent a 3D rotation with less parameters, each suitable for special applications. In this thesis, three different representations are used: rotation matrices R, axis angles $\mathbf{r}$ and unit quaternions $\mathbf{q}$. These are defined in Appendix B.1.

Varying the representation of the rotation and the order of the rigid transformations results in different parametrisations, $\mathbf{s}$, of the camera pose. Two representations are used explicitly in this thesis: the seven-dimensional vector $\mathbf{s}^q = [\mathbf{q}_{cw}^T, \mathbf{c}_w^T]^T$ comprises the camera orientation $\mathbf{q}_{cw}$ in the quaternion form, with $R_{cw} = \mathrm{rot}(\mathbf{q}_{cw})$, and the position $\mathbf{c}_w$, as used in (2.4a). The six-dimensional vector $\mathbf{s}^r = [\mathbf{r}_{cw}^T, \mathbf{w}_c^T]^T$ comprises the camera orientation $\mathbf{r}_{cw}$ in the axis angle form, with $R_{cw} = \mathrm{rot}(\mathbf{r}_{cw})$, and the position $\mathbf{w}_c$, as used in (2.4b). Note the difference between c, $\mathbf{c}$ and $\mathbf{s}$: c is used to name a camera, a camera pose or a camera coordinate system, while $\mathbf{c}$ denotes the optical centre of c, that is the 3D point that coincides with the origin of this coordinate system. In general, $\mathbf{a}_b$ is used to denote the origin of coordinate system a expressed in coordinate system b. Finally, $\mathbf{s}$ denotes a vector that contains the camera pose parameters and can be used, for instance, as argument to a cost function.

2.1.1.5 From World to Pixel Coordinates

By combining the forward transformations given in (2.1)–(2.4) to a single function $\mathcal{P}$, the expected pixel location $\mathbf{m}_p$ can be computed for each point $\mathbf{m}_w$:

$$\mathbf{m}_p = \mathcal{P}(\mathbf{m}_w) = (\mathcal{K} \circ \mathcal{D} \circ \mathcal{P}_n \circ \mathcal{T})(\mathbf{m}_w). \tag{2.5a}$$

Here, $\mathbf{m}_c = \mathcal{T}(\mathbf{m}_w)$ denotes the mapping from the world to the camera frame given in (2.4a) and (2.4b), $\mathbf{m}_n = \mathcal{P}_n(\mathbf{m}_c)$ maps $\mathbf{m}_c$ from the camera to the normalised image frame according to (2.1a), $\mathbf{m}_d = \mathcal{D}(\mathbf{m}_n)$ corresponds to the distortion function (2.2) and $\mathbf{m}_p = \mathcal{K}(\mathbf{m}_d)$ denotes the affine transformation from the distorted frame to the pixel frame as in (2.3a).

By using homogeneous coordinates and assuming an ideal normalised pinhole projection, that is $\mathcal{D} = \mathrm{I}$, $\mathcal{P}$ can be written as a 3×4 matrix P:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \propto z_c \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s_\alpha & h_x \\ 0 & f_y & h_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_{cw} & \mathbf{w}_c \\ \mathbf{0}_3^T & 1 \end{bmatrix}}_{P} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}. \tag{2.5b}$$

The matrix P is called the *projection matrix*. Note that the functions $\mathcal{K}$, $\mathcal{D}$ and $\mathcal{P}$ depend on the intrinsic camera parameters, the distortion coefficients and the camera pose, respectively. This

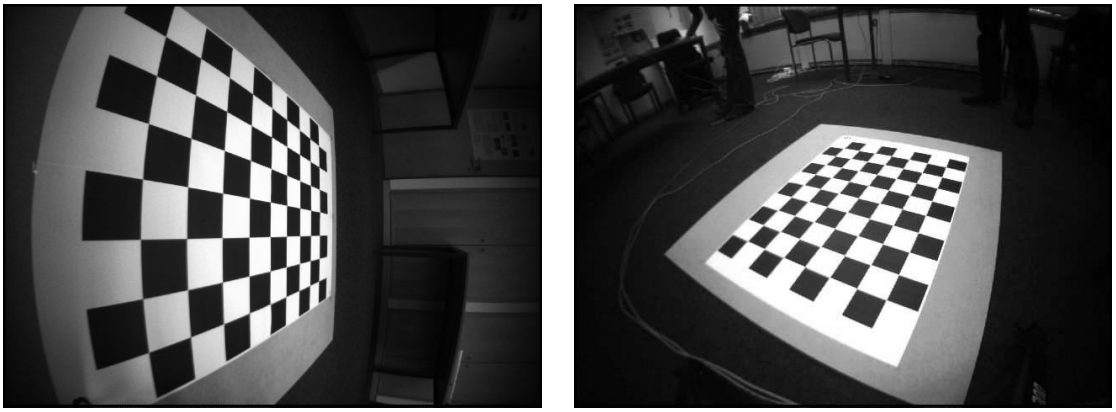


Figure 2.4: *Example images of a planar checkerboard pattern used for camera calibration. The camera is the same as in Figure 2.2.*

dependency will be expressed explicitly where required by adding the appropriate parameters, *i.e.* $\mathcal{T}(\mathbf{m}_w, \mathbf{s})$, where $\mathbf{s}$ denotes the camera pose.

2.1.2 Camera Calibration

The aim of the camera calibration procedure is to find the distortion coefficients and intrinsic parameters introduced in Section 2.1.1.2 and 2.1.1.3, respectively. The calibration typically requires images of a calibration object from different distances and angles. Figure 2.4 shows example images of a planar checkerboard pattern. This is a commonly used calibration object, as it is simple to detect the corner points j in the images and thus to construct sets of 2D/3D point correspondences $(\mathbf{m}_p^{(ij)}, \mathbf{m}_w^{(i)})$ for each image i. The calibration requires the extrinsic camera parameters to be estimated, too.

The procedure consists of several steps. The intrinsic and all the extrinsic camera parameters are initialised from the homographies between the calibration object plane and its images by ignoring distortions [16]. The homographies are obtained from (2.5b) by assuming that $z_w = 0$. The initial solution is then refined by solving the nonlinear least squares (LS) problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} \sum_{j=1}^{n} r_p^{(ij)LS} = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} \sum_{j=1}^{n} \|\mathbf{m}_p^{(ij)} - \mathcal{P}(\mathbf{m}_w^{(i)}, \mathbf{x})\|^2, \tag{2.6}$$

where m is the size of the image set, n is the number of corner points on the calibration pattern, $\mathbf{x}$ comprises the distortion coefficients, the intrinsic and all the extrinsic parameters, $\hat{\mathbf{x}}$ is the estimate of $\mathbf{x}$ and $\mathcal{P}(\mathbf{m}_w^{(i)}, \mathbf{x})$ corresponds to (2.5a). The residual $r_p^{(ij)LS}$ is referred to as the *reprojection error* and is based on one possible formulation of the *collinearity constraint* defined by the 2D/3D correspondence $(\mathbf{m}_p^{(ij)}, \mathbf{m}_w^{(i)})$. The collinearity constraint is treated in

Section 2.1.3.1. Standard algorithms such as the *Levenberg-Marquardt (LM)* algorithm [17] can be used to perform the minimisation of (2.6). Camera calibration is a well-known problem and many toolboxes are freely available online, see for instance [18] and the links given therein.

Camera measurements (*cf.* Section 2.1.4), *i.e.* the observed locations of landmarks in the images, are initially obtained in terms of pixels. Once the camera is calibrated, it is advantageous to preprocess the data and work in normalised image coordinates $\mathbf{m}_n$ instead. The normalisation step simplifies the equations for the geometric constraints induced by the camera measurements (*cf.* Section 2.1.3.1), as it decouples those from the fixed camera parameters and the lens model used. In this thesis, the camera is assumed to be calibrated and the camera measurements are assumed to be given in normalised image coordinates henceforth.

2.1.3 Geometric Constraints

This section describes geometric constraints induced by point correspondences. The collinearity constraint, defined by a 2D/3D correspondence, and the *epipolar constraint*, defined by a 2D/2D correspondence resulting from two camera views, are introduced.

2.1.3.1 Collinearity Constraint

The collinearity constraint means that a 3D point given in the camera frame, its projection on the image plane and the optical camera centre are collinear (*cf.* Figure 2.1(a)). The ray from the optical centre towards the 3D point is referred to as *view ray*. This geometric relation is a necessary condition for a point observed in an image and a world point to represent a real correspondence given the camera pose. The collinearity constraint can be used to either estimate the camera pose from 2D/3D point correspondences (*cf.* Section 2.3.1) or to *triangulate* a 3D point from 2D image locations and corresponding camera poses (*cf.* Section 2.3.2). Moreover, it can be used to validate point correspondences. This process is referred to as *outlier rejection* (*cf.* Section 2.3.3).

Different mathematical formulations of the collinearity constraint are encountered in literature [19]. In this thesis, two implicit formulations are used. The *image collinearity constraint* formulates the error in the (normalised) image space:

$$\mathbf{0}_2 = d_n(\mathbf{m}_n, \mathbf{m}_w, \mathbf{s}) = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \frac{1}{z_c} \begin{bmatrix} x_c \\ y_c \end{bmatrix} = \mathbf{m}_n - (\mathcal{P}_n \circ \mathcal{T})(\mathbf{m}_w, \mathbf{s}), \tag{2.7a}$$

where $\mathbf{s}$ denotes an appropriate parametrisation of the camera pose (*cf.* Section 2.1.1.4). This constraint is obtained from (2.1a) by subtracting $\mathbf{m}_n$ from both sides of the equation. The reprojection error as introduced in (2.6) is based on the image collinearity constraint, however, given in the pixel coordinate system, as it is used for camera calibration. The *homogeneous collinearity constraint* formulates the error in homogeneous space:

$$\mathbf{0}_2 = d_h(\mathbf{m}_n, \mathbf{m}_w, \mathbf{s}) = z_c \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} -\mathrm{I}_2 & \mathbf{m}_n \end{bmatrix} \mathcal{T}(\mathbf{m}_w, \mathbf{s}), \tag{2.7b}$$

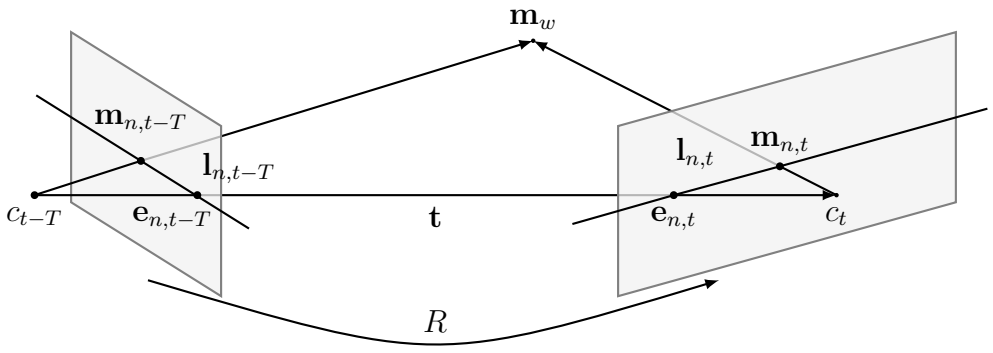


Figure 2.5: *Epipolar geometry:* $\mathbf{e}_{n,t-T}$ *and* $\mathbf{e}_{n,t}$ *are the epipoles and the lines through* $\mathbf{m}_n$ *and* $\mathbf{e}_n$ *are the epipolar lines* $\mathbf{l}_n$*. The vector* $\mathbf{t}$ *between the camera positions is called the baseline.*

where I_2 is the 2×2 identity matrix. It is obtained from (2.1a) by multiplying both sides of the equation with z_c. From a probabilistic point of view (2.7b) is preferred, as it avoids the division with z_c. The division of two normal variables results in a Cauchy distribution, which has infinite second and higher order moments [20]. However, the residual $d_h(\mathbf{m}_n, \mathbf{m}_w, \mathbf{s})$ depends on z_c, implying that distant points produce larger residuals in general. In contrast, $d_n(\mathbf{m}_n, \mathbf{m}_w, \mathbf{s})$ is not affected by the distance of the 3D points. Hence, the image collinearity constraint is preferred as a cost function for nonlinear estimation and outlier rejection.

2.1.3.2 Epipolar Constraint

The epipolar constraint results from the geometry of two camera views that are coupled by a relative rotation R and translation $\mathbf{t}$, the *epipolar geometry*. This is shown in Figure 2.5. The images may be acquired simultaneously. However, in this thesis, they are assumed to be acquired sequentially as c_{t-T} and c_t by one camera moving relative to a 3D point in space $\mathbf{m}_w$, where T denotes its sample time. The epipolar constraint means that $\mathbf{m}_{n,t}$, that is the projection of $\mathbf{m}_w$ in one image, defines an epipolar line $\mathbf{l}_{n,t-T}$ in the other image, on which the corresponding projection $\mathbf{m}_{n,t-T}$ lies. This geometric relation is a necessary condition for a 2D/2D point correspondence to be valid given the camera poses.

Assume that $\mathbf{s}_{t-T}$ and $\mathbf{s}_t$, as before, denote vectors containing the extrinsic parameters R_{cw} and $\mathbf{w}_c$ of both camera views. The mathematical formulation of the epipolar constraint is based on homogeneous coordinates:

$$0 = d_e(\mathbf{m}_{n,t-T}, \mathbf{m}_{n,t}, \mathbf{s}_{t-T}, \mathbf{s}_t) = \begin{bmatrix} x_{n,t} & y_{n,t} & 1 \end{bmatrix} \underbrace{\mathrm{S}(\mathbf{t})R}_{E} \begin{bmatrix} x_{n,t-T} \\ y_{n,t-T} \\ 1 \end{bmatrix}, \tag{2.8a}$$

where $\mathrm{S}(\mathbf{t})$ is given in (B.5) in Appendix B.1 and the relative rotation R and translation $\mathbf{t}$ are

defined as:

$$R = R_{cw,t} R_{cw,t-T}^T \tag{2.8b}$$

$$\mathbf{t} = R\mathbf{w}_{c,t-T} - \mathbf{w}_{c,t}. \tag{2.8c}$$

The 3×3 matrix $E = \mathrm{S}(\mathbf{t})R$ is called the *essential matrix*. The epipolar lines are:

$$\mathbf{l}_{n,t-T} = E \begin{bmatrix} x_{n,t} \\ y_{n,t} \\ 1 \end{bmatrix} \tag{2.9a}$$

and

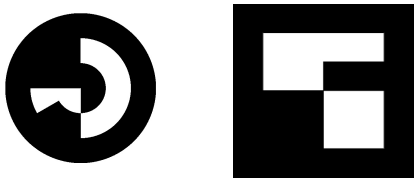
$$\mathbf{l}_{n,t} = E^T \begin{bmatrix} x_{n,t-T} \\ y_{n,t-T} \\ 1 \end{bmatrix}, \tag{2.9b}$$

respectively. The epipolar constraint can be used to reconstruct the essential matrix E from a set of 2D/2D correspondences (*cf.* Section 2.3.1.2). Note that in this application it is required to parametrise the epipolar constraint $d_e(\mathbf{m}_{n,t-T}, \mathbf{m}_{n,t}, E)$ by E instead of $\mathbf{s}_{t-T}$ and $\mathbf{s}_t$.

2.1.4 Camera Measurements

The previous sections describe the mathematical model of the image formation process and the geometric constraints resulting from this model given point correspondences, either between the 3D world and one camera image (*cf.* Section 2.1.3.1) or between two camera images (*cf.* Section 2.1.3.2). This section treats the image processing aspects of the *correspondence problem*. The correspondence problem can consist in deciding either, which primitive in an image corresponds to which landmark in 3D space, or, which primitive in one image corresponds to which primitive in another image in the sense that they are both images of the same landmark in 3D space.

In the first approaches to camera tracking, artificial landmarks, called *markers* or *fiducials*, have been used. A marker is a pattern that is easily detected in an image and contains a code for unique identification. Examples of different fiducial designs are shown in Figure 2.6. By placing markers at measured 3D positions in the world coordinate system, reliable 2D/3D correspondences can be obtained. Marker-based tracking technology as well as applications using it are described, for instance, in [21–24]. The freely available software library ARToolKit [25] is extensively used, especially for building augmented reality applications. Marker-based commercial tracking systems are, for instance, InterSense VIS-Tracker [26] and BBC FreeD [27]. Today, markers are still frequently used due to their reliability of detection. In this thesis, rectangular markers are used for the experimental evaluation of the visual-inertial tracking method developed in Chapter 5. However, as it is inconvenient and time-consuming to place special infrastructure in the target scene, in particular in large-scale environments, extensive research is undertaken in the computer vision community on *markerless image processing*.



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

Figure 2.6: *Different fiducial designs: circular fiducials (a) usually define one 3D point, given by the centre, while rectangular fiducials (b) define four points, given by the four corners. The detection of one rectangular marker is sufficient to compute the camera pose uniquely (*cf. *Section 2.3.1.1), as it yields four 2D/3D correspondences. The camera pose can also be estimated uniquely from the contour and code of one circular marker.*

Figure 2.7: *Corresponding point features in two camera images. Note how the periodic appearance of the feature marked by the circle and the significant change in the local appearance of the feature marked by the rectangle complicate the correspondence problem.*

The remainder of this section is concerned with this research area and the different approaches encountered in literature. Figure 2.7 illustrates the correspondence problem in the area of markerless image processing.

Similar to the scan matching technique in robotics [28], some recent image processing approaches solve the correspondence problem by performing a global image analysis, see for instance [29, 30]. However, the approach most commonly used in computer vision is based on local image analysis and *features*. A feature is a primitive combined with a descriptor describing its appearance. It is associated to a natural or synthetic landmark in 3D space. Features can, for instance, be corners [31–34], regions [35–38], lines [39–42], edges [43–45] or curves [46, 47]. The descriptor of a corner feature can be as simple as a vector containing the pixel intensities of a local window centred around the point, or more complex such as an oriented gradient histogram [48]. The feature-based approach is prone to ambiguities, which

can result in false correspondences (outliers). However, compared to the global approach, it is in general less computationally intense and more robust against occlusions. In this thesis, corner features are used to establish correspondences: they need no special segmentation, are robust to occlusions and provide distinctive descriptors. Furthermore, they provide point correspondences that can be used in the geometric constraints.

In order to match corner features across different views, two major approaches exist, both of which are suited for different applications and assumptions about the camera motion: *feature matching* and *feature tracking*. The properties of these techniques are described in the sequel.

2.1.4.1 Feature Matching

This approach is suitable for establishing correspondences between two images separated by a wide baseline, as those given in Figure 2.7, where no prior information about the camera pose is available. This configuration occurs, for instance, during *system initialisation* [8] or during *loop closing* [49]. Features are *detected* in both images and then matched against each other. This approach is also referred to as the *bottom up* approach, as no information is used for the matching other than the image. An outline and a performance analysis of corner detectors and descriptors suitable for wide baseline matching are given in [50, 51]. Another solution to wide baseline matching is to handle the problem as a classification problem, see for instance [34, 52–54]. The feature matching approach is in general computationally intense and tends to produce outliers.

2.1.4.2 Feature Tracking

This approach is suited for establishing correspondences within a continuous image sequence, or when a good prediction of the camera pose is available. Features are detected once and then tracked directly in subsequent images. This approach is also referred to as the *top down* approach, as prior information or assumptions about the camera pose are used to guide the feature tracking.

Two prominent techniques for solving the feature registration process are encountered in literature: one group of researchers uses the *block matching* method, see for instance [55–57]. The other group uses advanced techniques developed upon the well-known *Kanade-Lucas tracker (KLT)*, the seminal work of Lucas and Kanade in [58]. Important advancements are: introducing an affine transformation and illumination model [32, 35, 59], detecting outliers [59], improving the convergence of the registration [60, 61] and minimising the drift in the 2D feature locations [38]. Drift is a general problem of the feature tracking approach.

A common method to handle larger feature displacements between two images is to generate *image pyramids* and to perform the registration recursively, starting at the coarsest pyramid levels and descending to the finest, the original images, while propagating the respective registration result as initial guess. This *coarse-to-fine* strategy can be applied to both registration techniques mentioned above. An image pyramid is obtained by repeatedly subsampling and

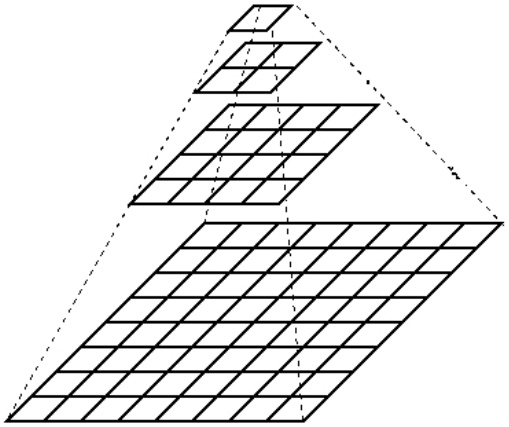


Figure 2.8: *Schematic representation of an image pyramid.*

lowpass filtering the image. The idea is shown in Figure 2.8. A pyramidal implementation of the basic Kanade-Lucas tracker is described in [62].

2.2 Camera IMU Systems

This section introduces the properties, measurement models and processing algorithms of inertial sensors or inertial measurement units and describes the synchronisation and calibration issues that arise when combining cameras with such sensors. The specific devices used in this thesis are presented as well.

2.2.1 Inertial Sensors

Inertial sensors measure translational and rotational body kinematics without the need for an external reference. In this thesis, the inertial measurement unit MT9 from XSens technologies [63] is used. It combines gyroscopes, accelerometers and magnetometers in a small sensor package and provides calibrated sensor readings given in physical units as well as a 3D orientation. All local computations are performed in an embedded digital signal processor (DSP).

The sensor readings — 3D angular velocity (rate of turn), 3D linear acceleration and 3D earth-magnetic field — are expressed in the *sensor coordinate system* s, which is aligned to the external housing. The readings are compensated for temperature, 3D misalignment and cross-sensitivity. However, the small and low cost microelectromechanical systems (MEMS) provide limited performance in terms of accuracy and bias stability, which is the main reason for the drift in a *strapdown inertial navigation system* [64].

The 3D orientation is obtained from the calibrated sensor readings using the accelerometers and magnetometers to compensate for the drift in the integrated gyroscope data. Note that large

amounts of ferrous material cause magnetic disturbances that deteriorate the accuracy of the estimate. The integration of angular velocities to an absolute orientation is given in (B.15) in Appendix B.3. Such a system is called *attitude and heading reference system (AHRS)*. The 3D orientation $R_{gs,t}$ is defined as the rotation between the sensor coordinate system s and the *global coordinate system* g. The global frame is an inertial reference frame, which is fixed to earth with the x-axis pointing to local magnetic north and the z-axis pointing opposite gravity. The IMU can output orientation as unit quaternions, Euler angles and rotation matrices (*cf.* Appendix B.1).

The sensor model of the 3D gyroscope and the 3D linear accelerometer as well as the principles of strapdown inertial navigation are essential for this thesis and are therefore discussed in the sequel.

2.2.1.1 Gyroscopes

The 3D gyroscope measures the angular velocities $\boldsymbol{\omega}_{s,t}^{gs}$ expressed in the sensor frame s. The superscript indicates the direction, that is from the sensor frame s to the global frame g. For the sake of better readability, the direction is suppressed in Chapter 3 and 5, respectively, where the notation is unambiguous. The largest error remaining after the calibration is a slowly varying bias $\mathbf{b}_{s,t}^{\omega}$. All other errors are modelled as zero mean white noise $\mathbf{e}_{s,t}^{\omega}$. The calibrated gyroscope signal $\mathbf{y}_{s,t}^{\omega}$ is:

$$\mathbf{y}_{s,t}^{\omega} = \boldsymbol{\omega}_{s,t}^{gs} + \mathbf{b}_{s,t}^{\omega} + \mathbf{e}_{s,t}^{\omega}. \tag{2.10}$$

The bias contains a static part, the startup bias, and a dynamic part, which is in part the result of calibration errors and unmodelled acceleration dependencies (g-sensitivity). The rate of change is referred to as the *bias stability* and it determines the worst case drift when integrating the gyroscope signal.

2.2.1.2 Accelerometers

The 3D accelerometer measures all linear accelerations affecting the IMU, that is body acceleration $\ddot{\mathbf{s}}_{s,t}$, sometimes referred to as free acceleration, and acceleration due to gravity $(-\mathbf{g}_s)$, both expressed in the sensor frame s. Similar to the gyroscope signal, the calibrated accelerometer signal $y_{s,t}^{a}$ is corrupted by a slowly varying bias $\mathbf{b}_{s,t}^{a}$ and zero mean white noise $\mathbf{e}_{s,t}^{a}$:

$$\mathbf{y}_{s,t}^{a} = \ddot{\mathbf{s}}_{s,t} - \mathbf{g}_s + \mathbf{b}_{s,t}^{a} + \mathbf{e}_{s,t}^{a}. \tag{2.11a}$$

The linear accelerations can be expressed in the global frame g, where the gravity vector $\mathbf{g}_g$ is a constant (*cf.* Figure 2.11 in the sequel):

$$\mathbf{y}_{s,t}^{a} = R_{sg,t}(\ddot{\mathbf{s}}_{g,t} - \mathbf{g}_g) + \mathbf{b}_{s,t}^{a} + \mathbf{e}_{s,t}^{a}. \tag{2.11b}$$

The above equations show that once the body acceleration $\ddot{\mathbf{s}}_{g,t}$ is known, the accelerometer signal can be used to estimate the direction of gravity, or alternatively, once the orientation $R_{sg,t}$ is known, the body acceleration $\ddot{\mathbf{s}}_{g,t}$ can be estimated.

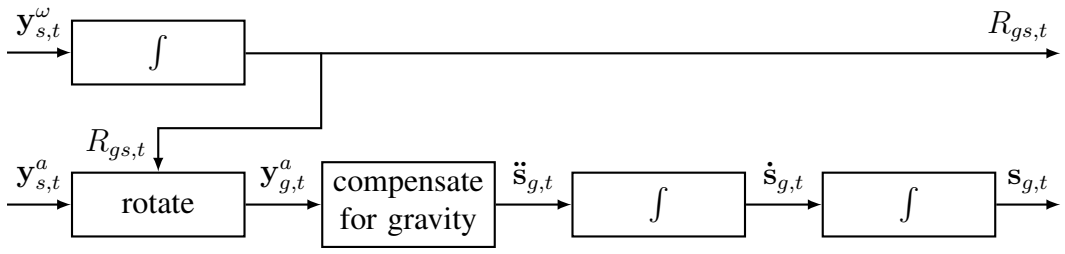


Figure 2.9: *The workflow of a strapdown inertial navigation system. Note that the sensor readings are assumed to be ideal,* i.e. *bias and noise terms are neglected.*

2.2.1.3 Strapdown Inertial Navigation

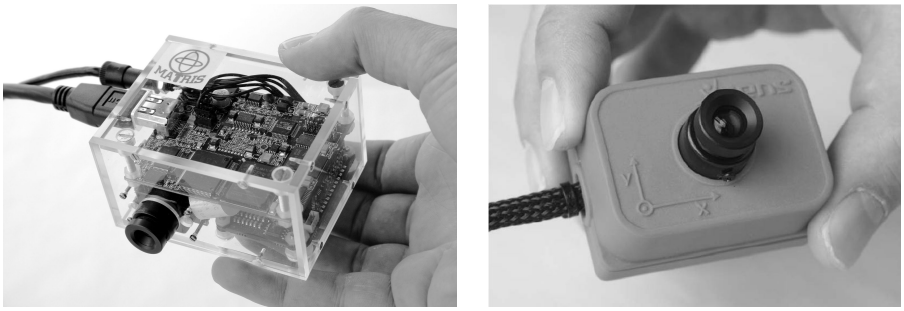
The technique of estimating the orientation and position of a moving body relative to a starting pose using only gyroscopes and accelerometers is called inertial navigation [64, 65]. In a strapdown configuration, the sensors are affixed to the body. The orientation of the body with respect to the inertial reference frame g, $R_{gs,t}$, is estimated by integrating the measured angular velocities $\mathbf{y}^{\omega}_{s,t}$. The position $\mathbf{s}_{g,t}$ is obtained by using $R_{gs,t}$ to subtract the acceleration due to gravity from the accelerometer readings and integrating the resulting 3D body acceleration $\ddot{\mathbf{s}}_{g,t}$ twice. The workflow of such a system, assuming ideal sensor readings, is outlined in Figure 2.9.

In practice, the inertial measurements are corrupted by noise and bias terms. These cause a drift in the calculated orientation and position. For instance, a constant gyroscope bias introduces a linearly growing orientation error. If the initial bias can be measured, the drift rate is determined by the bias stability. A constant accelerometer bias introduces a position error, which grows quadratically in time. The predominant error in the position is, however, caused by the coupling to the orientation. As can be seen in (2.11b), an error in the estimated orientation $R_{gs,t}$ results in an imaginary body acceleration, since gravity is not correctly compensated for. This erroneous body acceleration propagates quadratically into the position. Using small and low cost MEMS, the accuracy of the orientation and most notable the position estimate deteriorate quickly.

2.2.2 Combining Cameras and Inertial Sensors

Each physical sensor provides measurements in its own spatial and time reference. In order to fuse the information from different sensors, their measurements have to be synchronised, both in space and in time.

For the special case of visual-inertial *sensor fusion*, an integrated camera IMU device, the CamIMU, has been developed by XSens technologies [63] in the European project MATRIS [66]. The first and the most recent generation of the CamIMU are shown in Figure 2.10. These prototypes have been used during this thesis. The CamIMU integrates a firewire camera and an IMU — model MT9 as outlined in Section 2.2.1 — in one housing. It provides a time



(a) *(b)*

Figure 2.10: *The first (a) and the most recent (b) generation of the integrated camera IMU device, the CamIMU, developed by XSens technologies [63] in the European project MATRIS [66]. The CamIMU combines a camera and an IMU in one housing and provides a hardware synchronised stream of images and inertial data. Prototype (a) integrates a PtGrey DragonFly CCD colour camera and (b) a monochrome PtGrey Firefly MV CMOS camera.*

synchronised stream of images and IMU readings. The sample rate of the IMU is 100 Hz. The camera can be sampled at 12.5 or at 25 Hz. The spatial relations between both sensors and their frames of reference are given by rigid transformations and are outlined in the sequel. The 3D coordinate systems connected to the camera and the IMU have been introduced separately in Section 2.1.1.4 and 2.2.1, respectively. Figure 2.11 illustrates, how those coordinate systems are related. It outlines the complete geometric configuration of visual-inertial sensor fusion. To summarise, the following 3D coordinate systems are used:

World w**:** the world frame is typically attached to a real object in the target scene and is therefore often called object frame.

Camera c**:** the camera frame is fixed to the moving camera. The camera pose — comprising the rotation $R_{cw,t}$ and the translation $\mathbf{c}_{w,t}$ — is estimated with respect to the world frame.

Global g**:** the global frame is fixed to earth with the x-axis pointing to local magnetic north and the z-axis pointing opposite gravity.

Sensor s**:** the sensor frame is fixed to the moving IMU. The calibrated sensor readings as well as the orientation $R_{gs,t}$ obtained from the IMU are given with respect to the global frame. The sensor pose, however, is within this thesis assumed to be the rotation $R_{sw,t}$ and the translation $\mathbf{s}_{w,t}$ with respect to the world frame.

The camera and the sensor frame are rigidly attached. They are related by the *hand-eye transformation* comprising a hand-eye rotation R_{cs} and translation $\mathbf{c}_s$, where the latter denotes the origin of the camera frame expressed in the sensor frame. This rigid transformation can be

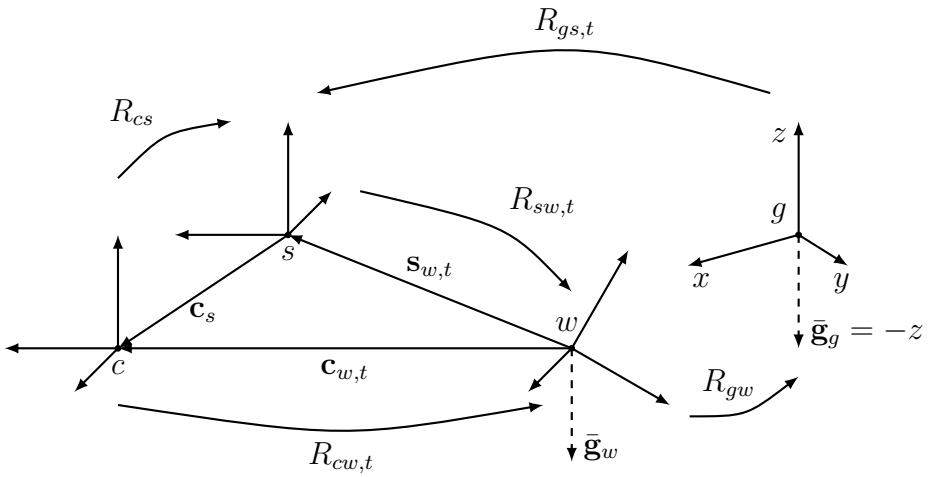


Figure 2.11: *Illustration of the 3D coordinate systems involved in visual-inertial sensor fusion and how they are related. The non-rigid transformations are indicated by a time subscript t. The dashed lines denote the direction of gravity $\bar{\mathbf{g}} = \frac{\mathbf{g}}{\|\mathbf{g}\|}$ expressed in the different coordinate systems. Note that $\mathbf{c}_s$ is given in the sensor frame, while $\mathbf{c}_{w,t}$ and $\mathbf{s}_{w,t}$ are expressed in the world frame. The general notation used in this thesis is: $\mathbf{a}_b$ denotes the origin of coordinate system a expressed in coordinate system b.*

used to convert the sensor pose to the camera pose:

$$R_{cw,t} = R_{cs} R_{sw,t} \tag{2.12a}$$
$$\mathbf{c}_{w,t} = (\mathbf{s}_{w,t} + R_{ws,t}\mathbf{c}_s) \tag{2.12b}$$

and vice versa:

$$R_{sw,t} = R_{cs}^T R_{cw,t} \tag{2.12c}$$
$$\mathbf{s}_{w,t} = (\mathbf{c}_{w,t} - R_{sw,t}^T \mathbf{c}_s). \tag{2.12d}$$

By inserting (2.12a) and (2.12b) into (2.4a), the hand-eye transformation can be further used to express the mapping of points from the world to the camera frame using the sensor instead of the camera pose:

$$\mathbf{m}_c = R_{cs}\big(R_{sw}(\mathbf{m}_w - \mathbf{s}_w) - \mathbf{c}_s\big). \tag{2.13}$$

This relation is the basis for integrating camera measurements into the visual-inertial tracking method as developed in Chapter 3. The procedure used to calibrate the hand-eye transformation is outlined in Section 2.2.2.1.

The world coordinate system is the reference frame of the camera in the same way as the global coordinate system is the reference frame of the IMU. The world frame and the global frame are in general not identical. They are assumed to be rigidly coupled. As a visual-inertial tracking system can use only one frame of reference, typically the world frame, some

knowledge about this coupling is required. The quantities that have to be accounted for are treated in Sections 2.2.2.2 through 2.2.2.4.

2.2.2.1 Hand-Eye Calibration

The hand-eye rotation is calculated from a set of gravity vectors expressed in both, the camera frame and the sensor frame, using a method based on unit quaternions [67]. Let $\mathbf{g}_{c,t}$ and $\mathbf{g}_{s,t}$, with $t = 1, \ldots, n$, be a set of corresponding gravity vectors satisfying $\mathbf{g}_{c,t} = \mathbf{q}_{cs} \odot \mathbf{g}_{s,t} \odot \mathbf{q}_{sc}$, where $\odot$ denotes the quaternion rotation. The sum of squared residuals:

$$\sum_{t=1}^{n} \|\mathbf{g}_{c,t} - \mathbf{q}_{cs} \odot \mathbf{g}_{s,t} \odot \mathbf{q}_{sc}\|^2 \tag{2.14a}$$

is then minimised by $\hat{\mathbf{q}}_{cs} = \mathbf{x}_1$, where $\mathbf{x}_1$ is the eigenvector corresponding to the largest eigenvalue $\lambda_{\max}$ of:

$$A = -\sum_{t=1}^{n} (\mathbf{g}_{c,t})^L (\mathbf{g}_{s,t})^R. \tag{2.14b}$$

The matrices $(\cdot)^L$ and $(\cdot)^R$ are given in (B.6b) in Appendix B.1. The gravity vectors can be obtained by reusing the camera calibration setup as described in Section 2.1.2. Provided that the calibration object is placed on a horizontal surface and that accelerometer readings $\mathbf{y}^a_{s,t}$, with $t = 1, \ldots, n$, are recorded for each calibration image while holding the CamIMU statically, the gravity vectors are given by $\mathbf{g}_{c,t} = R_{cw,t}\mathbf{g}_w$, where $R_{cw,t}$ is a byproduct of the camera calibration, and $\mathbf{g}_{s,t} = -\mathbf{y}^a_{s,t}$. Note that by placing the calibration object horizontally, the z-axis of the world frame and the global frame are aligned and $\mathbf{g}_w = \mathbf{g}_g$. The hand-eye translation $\mathbf{c}_s$ is computed from the location of the centre of the CCD Chip with respect to the origin of the sensor coordinate system — this measurement is provided by the manufacturer — and the internal camera parameters. A recent method for determining the hand-eye translation automatically as a part of the calibration process above can be found in [68]. In practice, the hand-eye calibration has to be repeated each time the objective is changed or adjusted, since this changes the camera coordinate system.

2.2.2.2 Direction of Gravity

In order to use the accelerometer readings $\mathbf{y}^a_{s,t}$ to estimate the body acceleration $\ddot{\mathbf{s}}_{w,t}$ with respect to the world frame, (2.11b) can be reformulated as:

$$\mathbf{y}^a_{s,t} = R_{sw,t}(\ddot{\mathbf{s}}_{w,t} - \mathbf{g}_w) + \mathbf{b}^a_{s,t} + \mathbf{e}^a_{s,t}. \tag{2.15}$$

This implies that the gravity vector $\mathbf{g}_w$ is required instead of $\mathbf{g}_g$. In other words, the z-axis of the global coordinate system has to be expressed in the world frame.

As the angular velocities are identical in each point of a rigid body implying that $\boldsymbol{\omega}^{gs}_{s,t} = \boldsymbol{\omega}^{ws}_{s,t}$, (2.10) can be restated as:

$$\mathbf{y}^{\omega}_{s,t} = \boldsymbol{\omega}^{ws}_{s,t} + \mathbf{b}^{\omega}_{s,t} + \mathbf{e}^{\omega}_{s,t} \tag{2.16}$$

and $\boldsymbol{\omega}_{s,t}^{ws}$ can be integrated to obtain the sensor orientation R_{sw} with respect to the world frame. The relation $\boldsymbol{\omega}_{s,t}^{gs} = \boldsymbol{\omega}_{s,t}^{ws}$ is derived in (B.18) in Appendix B.3.

2.2.2.3 Metric Scale

In a vision-only tracking system, the scale of the world frame is arbitrary. This results from the ambiguity between the size of the target scene and the distance of the camera to the scene: changing the scale of the target scene results in scaled camera positions. This ambiguity is introduced by the normalised pinhole projection given in (2.1). In an inertial navigation system, however, the scale of the global frame is relevant, as the accelerometer readings are obtained in absolute metric units. Thus, in order to apply sensor fusion, the relative scale of the world frame with respect to the global frame must be known and accounted for. Alternatively, both coordinate systems can be specified in identical units. The latter is assumed within this thesis, implying that the camera and IMU poses are obtained in identical units as well.

2.2.2.4 Relative Orientation

In order to utilise the orientation $R_{gs,t}$ provided by the IMU in a visual-inertial tracking system, knowledge about the relative rotation R_{gw} is required. Assuming that the hand-eye rotation R_{cs} is already known, this rigid rotation can be calculated directly from a sequence of corresponding camera and IMU orientations $R_{cw,t}$ and $R_{gs,t}$ by computing the average of $R_{gs,t}R_{cs}^{T}R_{cw,t}$ [69]. As within this thesis $R_{gs,t}$ is only used for assisting the initialisation of the visual-inertial tracking system, the accuracy of R_{gw} is less critical than the accuracy of the hand-eye rotation R_{cs}. However, if both quantities are equally important, *e.g.* in case $R_{gs,t}$ is used as a measurement rather than the calibrated sensor readings $\mathbf{y}_{s,t}^{\omega}$ and $\mathbf{y}_{s,t}^{a}$, the rotations R_{cs} and R_{gw} should be calibrated simultaneously. Such a method is given, for instance, in [70].

2.3 Computer Vision Techniques

This section introduces elementary computer vision techniques for performing 3D estimation based on camera measurements, or more specifically, based on point correspondences. Camera pose estimation and triangulation are treated in Section 2.3.1 and 2.3.2, respectively, and robust estimation is discussed in Section 2.3.3. As a summary, Sections 2.3.4 and 2.3.5 show, how these basic algorithms are applied to the fundamental computer vision problems of *structure from motion* and *model-based camera tracking* and give a brief review of the literature in these areas.

2.3.1 Camera Pose Estimation

If the structure of the target scene is known, the extrinsic camera parameters can be reconstructed from a set of 2D/3D corresponding points using the collinearity constraint as intro-

duced in Section 2.1.3.1. Three correspondences give a finite number of solutions. A unique solution can be obtained from four and more points. If no 3D information about the target scene is available, the camera pose can be estimated up to an unknown scale from a set of 2D/2D correspondences. In both cases, the common procedure is to compute a linear estimate and to use this as starting point for a nonlinear optimisation as outlined in the sequel.

2.3.1.1 Estimation from 2D/3D Correspondences

The camera pose has six degrees of freedom (DOF), three rotation angles and three position coordinates. In order to estimate the parameters, a minimal set of three 2D/3D correspondences is required. Linear methods based on three points are outlined, for instance, in [71]. Note that they yield six possible solutions.

A unique solution can be obtained from at least four point correspondences. The iterative methods based on four points are most relevant in practical applications, see for instance [72]. In this thesis, the POSIT (pose from orthography and scaling with iterations) algorithm [73] is used. An extension to planar point configurations is provided in [74].

With six or more points the standard direct linear transform (DLT) method can be used. It formulates a linear equation system $A\mathbf{x} = \mathbf{0}$ from the observed point correspondences using (2.5b) and solves this with a singular value decomposition (SVD) [17]. Assuming that $n \geq 6$ is the number of data points, A is a $2n \times 12$ matrix and $\mathbf{x} = (p_{11}, p_{12}, \ldots, p_{34})^T$, where p_{ij} denotes the element in the i^{th} row and j^{th} column of the 3×4 matrix P. Since the camera is assumed to be calibrated, the pose can easily be extracted from the estimate of P by multiplying with K^{-1} (2.3b), or, by using normalised point coordinates.

Planar point configurations require special treatment. As outlined in Section 2.1.2, the camera parameters are obtained from the homographies induced by the 2D/3D correspondences.

A solution from one of the methods above can be used as starting point for nonlinear optimisation. Assume that a set of correspondences $(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)})$ with $i = 1, \ldots, n \geq 3$ is given. The formulation of the nonlinear LS problem then becomes:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \sum_{i=1}^{n} r_n^{(i)LS} = \arg\min_{\mathbf{s}} \sum_{i=1}^{n} \| \, d_n(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)}, \mathbf{s})\|^2, \tag{2.17}$$

where d_n corresponds to (2.7a) and $\mathbf{s}$ is a minimal parametrisation of the camera pose, *e.g.* as $\mathbf{s}^r$ (*cf.* Section 2.1.1.4). If the point correspondences are *heteroscedastic*, that is if they have different covariances $P_n(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)})$, a nonlinear weighted least squares (WLS) problem can be formulated:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \sum_{i=1}^{n} r_n^{(i)WLS} = \arg\min_{\mathbf{s}} \sum_{i=1}^{n} d_n(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)}, \mathbf{s})^T P_n^{-1}(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)}) d_n(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)}, \mathbf{s}). \tag{2.18}$$

The quantities $r_n^{(i)WLS}$ are also referred to as *Mahalanobis distances*. The nonlinear optimisation can be performed using standard methods such as the Levenberg-Marquardt algorithm.

Both the LS and the WLS estimator are sensitive to outliers. Even a single incorrect correspondence can have a large effect on the resulting pose. In order to reduce the influence of outliers, many different robust estimation techniques have been proposed. The most popular one is the *M-estimation* method suggested in [75]. M-estimators minimise the sum of a function of residuals:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \sum_{i=1}^{n} \rho(r^{(i)}). \tag{2.19}$$

The derivative of ρ is called the *influence function*. It appears as a weighting function in the minimisation. In this thesis, the *TUKEY* influence function is used. This influence function is zero for residuals $r^{(i)}$ larger than a threshold. Other, less restrictive, influence functions are the *Cauchy* and the *Huber* influence function, see for instance [76]. M-estimators are suitable, when there are few outliers. A method that handles much higher outlier rates ($\gg 20\%$) is the *random sampling consensus (RANSAC)* method. This is introduced in Section 2.3.3.

Note that the robust estimation techniques apply not only to the special problem considered within this section but to nonlinear optimisation in general.

2.3.1.2 Estimation from 2D/2D Correspondences

The epipolar constraint (2.8a) can be used to estimate the essential matrix $E = \mathrm{S}(\mathbf{t})R$ from a set of 2D/2D corresponding points $(\mathbf{m}_{n,t-T}^{(i)}, \mathbf{m}_{n,t}^{(i)})$ with $i = 1, \ldots, n \geq 5$. Once E is known, it can be factorised into R and $\mathbf{t}$ and — assuming that one camera frame $c_{t-T} = w$ defines the world frame — the camera pose parameters $\mathbf{s}_t$ as well as the point $\mathbf{m}_w$ can be computed except for an unknown scaling. The geometric relations are illustrated in Figure 2.5.

As E has five DOF and each 2D/2D correspondence yields one equation, a minimum set of five correspondences is needed. A solution based on five points is described in [12]. However, it yields comparatively unstable results. The well-known eight-point algorithm is most relevant in practical applications. Analogously to the DLT method, it formulates a linear equation system $A\mathbf{x} = \mathbf{0}$ from the observed point correspondences using (2.8a) and solves this with a singular value decomposition. The matrix A is a $n \times 9$ matrix, with $n \geq 8$, and $\mathbf{x} = (e_{11}, e_{12}, \ldots, e_{33})^T$, where e_{ij} denotes the element in the i^{th} row and j^{th} column of the 3×3 matrix E.

The linear estimate can be used as a starting point for a nonlinear optimisation. Many different cost functions have been suggested, see for instance [19]. In this thesis, the *symmetric epipolar distance* is used. For one 2D/2D correspondence $(\mathbf{m}_{n,t-T}, \mathbf{m}_{n,t})$ this is defined as:

$$d_s(\mathbf{m}_{n,t}, \mathbf{m}_{n,t-T}, E) = \frac{d_e(\mathbf{m}_{n,t-T}, \mathbf{m}_{n,t}, E)}{\sqrt{(\mathbf{l}_{n,t})_1^2 + (\mathbf{l}_{n,t})_2^2}} + \frac{d_e(\mathbf{m}_{n,t}, \mathbf{m}_{n,t-T}, E)}{\sqrt{(\mathbf{l}_{n,t-T})_1^2 + (\mathbf{l}_{n,t-T})_2^2}}, \tag{2.20}$$

where $\mathbf{l}_{n,t-T}$ and $\mathbf{l}_{n,t}$ are as in (2.9) and $(\mathbf{v})_i$ denotes the i^{th} element of vector $\mathbf{v}$. Assuming n such correspondences, the formulation of the nonlinear LS problem based on the symmetric

epipolar distance is:

$$\hat{E} = \arg\min_{E} \sum_{i=1}^{n} r_s^{(i)LS} = \arg\min_{E} \sum_{i=1}^{n} d_s^2(\mathbf{m}_{n,t}^{(i)}, \mathbf{m}_{n,t-T}^{(i)}, E), \tag{2.21}$$

where the squared distance can be simplified to:

$$d_s^2(\mathbf{m}_{n,t}^{(i)}, \mathbf{m}_{n,t-T}^{(i)}, E) = d_e^2(\mathbf{m}_{n,t-T}, \mathbf{m}_{n,t}, E) \left(\frac{1}{(\mathbf{l}_{n,t})_1^2 + (\mathbf{l}_{n,t})_2^2} + \frac{1}{(\mathbf{l}_{n,t-T})_1^2 + (\mathbf{l}_{n,t-T})_2^2} \right) \tag{2.22}$$

Note that it is essential to use a minimal parametrisation of E, one is given in [77]. The nonlinear WLS problem can be formulated analogously to (2.18).

Further details on the estimation of the matrix E, its factorisation into R and $\mathbf{t}$ and the handling of special cases such as planar configurations are given, for instance, in [13].

2.3.2 Triangulation

The triangulation problem is to find the position $\mathbf{m}_w$ of a point in 3D space given several camera views. A camera view is here defined as a tuple consisting of the normalised image coordinate $\mathbf{m}_{n,t}$ and the corresponding camera pose $\mathbf{s}_t$ at time t. Many different triangulation methods are encountered in literature. An experimental evaluation of the most popular techniques can be found in [78].

A linear LS estimate is obtained from the known camera views by formulating a linear equation system using the homogeneous collinearity constraint (2.7b) and solving it with pseudo-inverses [17]. As $\mathbf{m}_w$ has three DOF and each view yields two equations, a minimum set of two views is needed. Given $n \geq 2$ views, A is a $2n \times 3$ matrix, $\mathbf{b}$ is a vector of length $2n$ and $\mathbf{x} = \mathbf{m}_w$. The nonlinear LS and the WLS estimate can be formulated analogously to (2.17) and (2.18) by parametrising $\mathbf{m}_w$ instead of $\mathbf{s}$.

2.3.3 Random Sampling Consensus

Random sampling consensus (RANSAC) is an iterative technique for rejecting outliers from a set of observations, where the outlier rate is high ($\gg 20\%$). The general definition of an *outlier* is: an *observation* that is not consistent with the *pattern* given by the biggest consistent subset of observations. In the context of camera tracking, observations are typically corresponding primitives or camera views, the pattern is a mapping or a constraint induced by those correspondences such as (2.7) or (2.8), and outliers arise from poorly localised or incorrect correspondences. The standard RANSAC scheme [79] is given in Algorithm 2.1. Several advanced techniques have been developed upon this original method with improved sampling algorithms or better cost functions, see for instance [80, 81] and the references therein. The RANSAC method is a general *hypothesise-and-test* technique that can be used to increase the

Algorithm 2.1 RANSAC

Assume that p is the minimal number of observations required to estimate the pattern.

1. Randomly choose m sample sets of p observations.
2. For each sample set:
   (a) Estimate the pattern.
   (b) Determine the number of observations consistent with this pattern (inliers) by applying a user-defined outlier threshold.
3. Keep the largest set of consistent correspondences and estimate the pattern.
4. Eliminate all observations, which are not consistent with this pattern.

robustness of the camera pose estimation and triangulation as outlined in Section 2.3.1 and 2.3.2, respectively.

2.3.4 Structure From Motion

The structure from motion (SFM) problem consists in simultaneously estimating the camera trajectory and the scene structure from a continuous image sequence. A SFM system comprises three major building blocks: the image processing required to establish correspondences, the camera pose estimation and the structure estimation. In this thesis, the scene structure is represented by a sparse 3D point cloud, which is the most common representation used in computer vision, implying that a feature tracking approach based on corners can be used. Standard approaches with respect to all of the three building blocks are outlined in Sections 2.1.4 through 2.3.2. Robust estimation is an important topic in the context of SFM. The RANSAC algorithm or advanced hypothesise-and-test frameworks have become the standard way of handling outliers.

A notorious problem is the accumulating error in the camera and structure registration, the *drift*. In an offline system, the drift is typically corrected for by performing a batch optimisation involving the whole image sequence. This technique is called global *bundle adjustment* [82, 83]. Some recent contributions apply bundle adjustment also in a real-time system by either performing local optimisation, that is by processing only a subset of images [84], or by applying the optimisation as a parallel background process [57].

However, if real-time operation is required, the recursive techniques are better suited. In a recursive scheme, rather than aiming at an accurate estimate of the complete camera trajectory, the goal is to obtain an estimate for the current time instance and a representation of the confidence in this estimate. Recursive SFM, or — in the robotics jargon — *simultaneous localisation and mapping (SLAM)* is treated in Section 2.4.3 and in Chapters 4 and 5.

2.3.5 Model-Based Camera Tracking

Model-based camera tracking denotes the much simpler problem of estimating the camera pose relative to a known 3D model of the target scene. Different representations of such models have been proposed in literature, *e.g.* a database of features [52, 85, 86], a CAD (computer aided design) model [5, 42, 87] or a set of reference images [30, 31, 37]. Model-based tracking is nowadays a well-understood technique. A survey of different algorithms can be found, for instance, in [8].

2.4 Recursive Filtering Techniques

Filtering is the problem of estimating the unknown *state* of a system from a sequence of noisy *observations* or *measurements* made on the system and a sequence of known *control inputs* or *input signals* that carry information about the change of the system state. In a recursive filtering approach the observations and controls are processed sequentially rather than as a batch and a *Markov assumption* is made, according to which the past and the future data are independent, if the current state is known. When working with a dynamic system, the distribution of interest is:

$$p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t), \tag{2.23}$$

the *probability density function (pdf)* of state $\mathbf{x}$ at time t given the entire history of measurements $\mathbf{Y}_t = \{\mathbf{y}_i\}_{i=1}^t$ and inputs $\mathbf{U}_t = \{\mathbf{u}_i\}_{i=1}^t$. The state history is denoted $\mathbf{X}_t = \{\mathbf{x}_i\}_{i=0}^t$. The Markov assumption postulates that the current state is a complete summary of the past, implying that it is not necessary to store the entire data set nor to reprocess existing data, when a new measurement or input becomes available. The problem of real-time camera tracking fits perfectly into the recursive filtering framework. In this case the state vector contains information related to the pose and the kinematic characteristics of the camera and potentially to the scene structure as well.

Assuming that the a priori pdf $p(\mathbf{x}_{0|-1})$ is available, the density $p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t)$ is obtained recursively in two stages: prediction and measurement update. The *prediction* distribution follows from marginalisation:

$$p(\mathbf{x}_{t+T}|\mathbf{Y}_t, \mathbf{U}_{t+T}) = \int p(\mathbf{x}_{t+T}|\mathbf{x}_t, \mathbf{u}_{t+T}) p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t)\, d\mathbf{x}_t, \tag{2.24a}$$

where the integration is performed over the entire state space of $\mathbf{x}_t$ and T is the sample time. Note that in (2.24a) the Markov assumption $p(\mathbf{x}_{t+T}|\mathbf{x}_t, \mathbf{Y}_t, \mathbf{U}_{t+T}) = p(\mathbf{x}_{t+T}|\mathbf{x}_t, \mathbf{u}_t)$ is used. A new measurement $\mathbf{y}_t$ is incorporated into the state estimate using Bayes' rule:

$$p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t) = \frac{p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{Y}_{t-T}, \mathbf{U}_t)}{p(\mathbf{y}_t|\mathbf{Y}_{t-T}, \mathbf{U}_t)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{Y}_{t-T}, \mathbf{U}_t)}{\int p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{Y}_{t-T}, \mathbf{U}_t)\, d\mathbf{x}_t}, \tag{2.24b}$$

where $\int p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{Y}_{t-T}, \mathbf{U}_t)\, d\mathbf{x}_t$ is a normalising constant and $p(\mathbf{y}_t|\mathbf{x}_t)$ is referred to as measurement *likelihood*. The pdf $p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t)$ is referred to as *posterior* distribution. In

general, the integrals that are required to propagate the posterior distribution are intractable. The *Kalman filter (KF)* [88] is an important exception to this. It provides an optimal analytic solution to the linear Gaussian problem. In most other cases, deterministic approximations such as the *extended Kalman filter (EKF)* or stochastic approximations such as the *particle filter (PF)* [89, 90] are required.

The state-space approach is the most commonly used method to model a dynamic system. A *state-space model* consists of two components: a model describing the evolution of the state with time and subject to control inputs, the *system* or *process model*, and a model relating the noisy measurements to the state, the *measurement* or *observation model*. The state vector contains all relevant information required to describe the system under investigation. The most general model is the nonlinear state-space model. However, many applications allow for special cases.

An outline of the state-space forms and filtering methods used in this thesis is given in Section 2.4.1 and 2.4.2, respectively. Section 2.4.3 shows, how recursive filters are applied to the fundamental problems of camera *localisation* and *mapping* providing a brief review of the literature in this area as well.

2.4.1 State-Space Models

This section introduces the most general form of the nonlinear state-space model and two important special cases: the linear Gaussian model and the nonlinear model with linear Gaussian substructure. The models are given in discrete-time. Though not stated explicitly in the sequel, it should be noted that the measurement model can comprise a number of equations, one for each specific sensor, thus providing a natural interface for sensor fusion.

2.4.1.1 Nonlinear Model

In its most general form, the system model is described by the relation:

$$\mathbf{x}_{t+T} = f(\mathbf{x}_t, \mathbf{u}_{t+T}, \mathbf{v}_t), \tag{2.25a}$$

where $\mathbf{u}_{t+T}$ is the known input to the system and $\mathbf{v}_t$ is an unknown unpredictable input modelled as a stochastic process and referred to as *process noise*. The measurement model is described by the implicit relation:

$$\mathbf{0} = h(\mathbf{x}_t, \mathbf{y}_t, \mathbf{e}_t), \tag{2.25b}$$

where $\mathbf{y}_t$ denotes the observation made on the system at time t and $\mathbf{e}_t$ denotes the *measurement noise*, an unpredictable disturbance modelled as a stochastic process. Model (2.25) is very general. Many practical problems can be modelled with additive noise and an explicit measurement relation resulting in:

$$\mathbf{x}_{t+T} = f(\mathbf{x}_t, \mathbf{u}_{t+T}) + \mathbf{v}_t \tag{2.26a}$$

$$\mathbf{y}_t = h(\mathbf{x}_t) + \mathbf{e}_t. \tag{2.26b}$$

Model (2.26) can be stated in terms of the probabilities required in (2.24) using:

$$p(\mathbf{x}_{t+T}|\mathbf{x}_t, \mathbf{u}_{t+T}) = p_{\mathbf{v}_t}(\mathbf{x}_{t+T} - f(\mathbf{x}_t, \mathbf{u}_{t+T})|\mathbf{x}_t, \mathbf{u}_{t+T}) \tag{2.27a}$$
$$p(\mathbf{y}_t|\mathbf{x}_t) = p_{\mathbf{e}_t}(\mathbf{y}_t - h(\mathbf{x}_t)|\mathbf{x}_t), \tag{2.27b}$$

where $p_{\mathbf{v}_t}$ and $p_{\mathbf{e}_t}$ denote the pdf of $\mathbf{e}_t$ and $\mathbf{v}_t$, respectively.

2.4.1.2 Linear Gaussian Model

The linear model, subject to Gaussian noise, is the most restrictive, but, at the same time, the most important special state-space form, as it allows for an optimal analytic solution of the recursive filtering problem given by the Kalman filter. Under the linear Gaussian assumption, the system and the measurement model are both linear functions of the state:

$$\mathbf{x}_{t+T} = F_t\mathbf{x}_t + G_t^u\mathbf{u}_{t+T} + G_t^v\mathbf{v}_t \tag{2.28a}$$
$$\mathbf{y}_t = H_t\mathbf{x}_t + H_t^e\mathbf{e}_t, \tag{2.28b}$$

where F_t, G_t^u, G_t^v, H_t and H_t^e are matrices and where $\mathbf{v}_t$ and $\mathbf{e}_t$ are stochastical vectors distributed according to $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, Q_t)$ and $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, R_t)$. Alternatively, model (2.28) can be described by the distributions:

$$p_{\mathbf{v}_t}(\mathbf{x}_{t+T} - F_t\mathbf{x}_t + G_t^u\mathbf{u}_{t+T}|x_t, u_t) = \mathcal{N}(\mathbf{x}_{t+T}; F_t\mathbf{x}_t - G_t^u\mathbf{u}_{t+T}, G_t^vQ_tG_t^{vT}) \tag{2.29a}$$
$$p_{\mathbf{e}_t}(\mathbf{y}_t - H_t\mathbf{x}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; H_t\mathbf{x}_t, H_t^eR_tH_t^{eT}). \tag{2.29b}$$

Here $\mathcal{N}(\hat{\mathbf{x}}, P)$ denotes a multivariate normal distribution with mean $\hat{\mathbf{x}}$, covariance P and corresponding Gaussian pdf $\mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}, P)$ in $\mathbf{x}$.

2.4.1.3 Nonlinear Model with Linear Gaussian Substructure

Under the nonlinear model with linear Gaussian substructure, conditioned on one part of the state space denoted $\mathbf{x}^n$, the remaining part $\mathbf{x}^l$ is linear and Gaussian. Compared to the nonlinear model, this special model class allows for more efficient filtering algorithms such as the *marginalised particle filter (MPF)* or *Rao-Blackwellised particle filter*, see Section 2.4.2.3. The general state-space form of this model is:

$$\mathbf{x}_{t+T}^n = f_t^n(\mathbf{x}_t^n, \mathbf{u}_{t+T}^n) + F_t^n(\mathbf{x}_t^n)\mathbf{x}_t^l + G_t^{vn}(\mathbf{x}_t^n)\mathbf{v}_t^n \tag{2.30a}$$
$$\mathbf{x}_{t+T}^l = f_t^l(\mathbf{x}_t^n) + F_t^l(\mathbf{x}_t^n)\mathbf{x}_t^l + G_t^{ul}(\mathbf{x}_t^n)\mathbf{u}_{t+T}^l + G_t^{vl}(\mathbf{x}_t^n)\mathbf{v}_t^l \tag{2.30b}$$
$$\mathbf{y}_t = h_t(\mathbf{x}_t^n) + H_t(\mathbf{x}_t^n)\mathbf{x}_t^l + H_t^e(\mathbf{x}_t^n)\mathbf{e}_t, \tag{2.30c}$$

with $\mathbf{x}_t^l \sim \mathcal{N}(\hat{\mathbf{x}}_t^l, P_t)$, $\begin{bmatrix} \mathbf{v}_t^n \\ \mathbf{v}_t^l \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} Q_t^n & 0 \\ 0 & Q_t^l \end{bmatrix}\right)$ and $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, R_t)$. Note that, for simplicity, $\mathbf{v}_t^n$ and $\mathbf{v}_t^l$ are assumed mutually independent. F_t^n, G_t^{vn}, F_t^l, G_t^{ul}, G_t^{vl}, H_t and H_t^e are matrices depending on $\mathbf{x}_t^n$ and f_t^n, f_t^l and h_t denote nonlinear functions.

The model defined by (2.30) is very general. In many practical applications it is sufficient to consider a special case of it. An important special case occurs, if the system model is linear Gaussian and the measurement model is nonlinear. This leads to a substantial reduction in the computational complexity of the filtering problem, since one covariance matrix can be used for all particles. If the system model is almost linear, it can be linearised in order to reduce the computational complexity. This is of special interest in real-time implementations, such as the one considered in this thesis. Special cases of model (2.30) and their implications are treated in [91].

2.4.2 Filtering Methods

This section presents three approximate solutions to the recursive filtering problem: the particle filter (PF), which is also referred to as *sequential Monte Carlo method*, the extended Kalman filter (EKF), which is an extension of the Kalman filter (KF) to nonlinear systems and the marginalised particle filter (MPF), which combines the standard particle filter and the Kalman filter in order to exploit a linear Gaussian substructure in a nonlinear model.

2.4.2.1 Particle Filter

The particle filter can be used to obtain a stochastic approximation of the filtering densities (2.24), even if a nonlinear state-space model is given in its most general form (2.25). Since the seminal paper [89], much has been written about the particle filter as well as applications using it. The theory is described in [92, 93]. Interesting variations of this theory that try to improve the proposal density are, for instance, the *auxiliary particle filter* [94], the *Gaussian particle filter* [95] and the *unscented particle filter* [96]. Applications of the particle filter to the problem of camera tracking can be found, for instance, in [45, 97–99].

The key idea of the particle filter is to represent the posterior density (2.23) by a set of $n \gg 1$ samples or *particles* $\{\mathbf{x}_t^{[i]}\}_{i=1}^n$ and associated weights $\{w_t^{[i]}\}_{i=1}^n$, such that:

$$p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t) \approx \sum_{i=1}^{n} w_t^{[i]} \delta(\mathbf{x}_t - \mathbf{x}_t^{[i]}), \tag{2.31}$$

where $\delta(\cdot)$ denotes the Dirac delta function. It can be shown that, as $n \to \infty$, the approximation (2.31) approaches the true posterior. The particles are identically and independently distributed (i.i.d.) samples from a *proposal distribution* $q(\cdot)$ and the weights $w_t^{[i]}$ are normalised *important weights* according to:

$$w_t^{[i]} \propto \frac{p(\mathbf{X}_t^{[i]}|\mathbf{Y}_t, \mathbf{U}_t)}{q(\mathbf{X}_t^{[i]}|\mathbf{Y}_t, \mathbf{U}_t)}, \tag{2.32a}$$

or in a recursive fashion:

$$w_t^{[i]} \propto w_{t-T}^{[i]} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{[i]}) p(\mathbf{x}_t^{[i]}|\mathbf{x}_{t-T}^{[i]}, \mathbf{u}_t)}{q(\mathbf{x}_t^{[i]}|\mathbf{x}_{t-T}^{[i]}, \mathbf{Y}_t, \mathbf{U}_t)}. \tag{2.32b}$$

The only restriction imposed on $q(\cdot)$ is that its support includes the support of $p(\cdot)$. Note if the proposal distribution is chosen to be the prediction, the update of the importance weights simplifies to:

$$w_t^{[i]} \propto w_{t-T}^{[i]}\, p(\mathbf{y}_t|\mathbf{x}_t^{[i]}). \tag{2.33}$$

This is the most common choice of importance density due to its simplicity, although it is not optimal. Details regarding the principle of *importance sampling* can be found, for instance, in [95].

It can be shown that, using the recursive propagation of the samples and weights according to (2.32b), the approximate distribution will degenerate. After a few iterations all but one particle will have negligible weight. A commonly used measure for the degeneracy of the particle set is the *effective sample size* N_{eff} [90]. An approximation of this measure can be calculated from the normalised importance weights (2.32b):

$$\hat{N}_{\text{eff},t} = \frac{1}{\sum_{i=1}^{n} \left(w_t^{[i]}\right)^2}. \tag{2.34}$$

The effective sample size measures the number of particles that contribute to the support of the estimated probability density function. A solution to the degeneracy problem is to introduce a *resampling* step as suggested in [89].

The basic idea of resampling is to eliminate particles with small weight and to concentrate on particles with large weight. This is done by drawing a new particle set $\{\mathbf{x}_t^{[i]*}\}_{i=1}^n$ from the approximate distribution (2.31) with replacement, such that $\Pr(\mathbf{x}_t^{[i]*} = \mathbf{x}_t^{[i]}) = w_t^{[i]}$, and by setting $\{w_t^{[i]*}\}_{i=1}^n = \frac{1}{n}$. Different resampling methods are compared in [100]. In this thesis, *systematic resampling* [90] is used. This is the most efficient resampling algorithm, as it requires only one random number and takes time linear in the number of particles. In *sampling importance resampling (SIR)*, the particle set is resampled in each iteration, while in *sequential importance sampling (SIS)*, resampling is performed only when it is necessary according to some measure such as $\hat{N}_{\text{eff},t}$. Algorithm 2.2 presents the SIR filter as suggested in [90]. Note that in the SIR filter, the proposal distribution is chosen to be the prediction. A minimum variance estimate $\hat{\mathbf{x}}_t$ of the state can be obtained after step 2 of this algorithm as a weighted sample mean:

$$\hat{\mathbf{x}}_{t|t} = \sum_{i=1}^{n} w_t^{[i]} \mathbf{x}_{t|t}^{[i]}, \tag{2.36a}$$

and the covariance of the estimate is given by the sample covariance:

$$P_{t|t} = \sum_{i=1}^{n} w_t^{[i]} \left((\mathbf{x}_{t|t}^{[i]} - \hat{\mathbf{x}}_{t|t})(\mathbf{x}_{t|t}^{[i]} - \hat{\mathbf{x}}_{t|t})^T\right). \tag{2.36b}$$

Algorithm 2.2 Sampling Importance Resampling (SIR) Filter

1. Initialise the filter: $\{\mathbf{x}_{0|-1}^{[i]}\}_{i=1}^{n} \sim p(\mathbf{x}_0)$ and $\{w_0^{[i]}\}_{i=1}^{n} = \frac{1}{n}$ for $i = 1, \ldots, n$, where $p(\mathbf{x}_0^n)$ can be arbitrary, but it is assumed known. Let $t \coloneqq 0$.

2. Measurement update: compute the importance weights according to (2.33) and normalise:
$$w_t^{[i]} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{[i]})\, w_{t-T}^{[i]}}{\sum_{j=1}^{n} p(\mathbf{y}_t|\mathbf{x}_t^{[j]})\, w_{t-T}^{[j]}}, \quad i = 1, \ldots, n.$$

3. Resample.

4. Time update: generate n i.i.d. samples $\{\mathbf{x}_{t+T|t}^{[i]}\}_{i=1}^{n} \sim p(\mathbf{x}_{t+T}|\mathbf{x}_{t|t}^{[i]}, \mathbf{u}_{t+T})$.

5. Let $t \coloneqq t + T$ and repeat from 2.

2.4.2.2 Extended Kalman Filter

The extended Kalman filter (EKF) is an approximate extension of the Kalman filter (KF) to nonlinear systems. The Kalman filter, as developed in [88], provides an analytic solution to the linear Gaussian problem. Given that the properties of the linear Gaussian model defined in Section 2.4.1.2 hold and that the a priori pdf is Gaussian, the posterior density is at every time step Gaussian and, hence, can be parametrised by a mean and a covariance. The main idea is then to update the moments using a linear filter so that the covariance of the estimation error is minimised. Derivations of the Kalman filter can be found, for instance, in [28, 101].

As many practical applications require nonlinear models, different approximative extensions of the Kalman filter have been developed. The extended Kalman filter, which linearises the nonlinear model by applying a first order Taylor expansion around the latest state estimate, is the most commonly used. It is treated in different books on filtering, *e.g.* [28, 101, 102]. Further approximative extensions are the second order extended Kalman filter [103], the iterated extended Kalman filter [101, 102] and the unscented Kalman filter [104]. All of these are concerned with improving the possibly poor performance of the extended Kalman filter in the presence of high model nonlinearities, however, at the expense of additional complexity. Applying the extended Kalman filter to tracking problems is widespread in computer vision as well as in robotics, see for instance [28, 39, 105]. Further references are given in Section 2.4.3. The extended Kalman filter — in this thesis applied to the problem of visual-inertial camera tracking and recursive feature localisation — is introduced in the sequel.

Assuming that the considered system is modelled according to (2.25), with the additional property that $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, Q_t)$ and $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, R_t)$, the extended Kalman filter applies a first order Taylor expansion around the latest estimate in order to obtain an approximate linear

Gaussian model, which can then be estimated optimally. The linearised system model is:

$$\begin{aligned}\mathbf{x}_{t+1} &\approx f(\hat{\mathbf{x}}_{t|t}, \mathbf{u}_{t+T}, \mathbf{0}) + F_t(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}) + G_t^v \mathbf{v}_t \\ &= F_t \mathbf{x}_t + f(\hat{\mathbf{x}}_{t|t}, \mathbf{u}_{t+T}, \mathbf{0}) - F_t \hat{\mathbf{x}}_{t|t} + G_t^v \mathbf{v}_t, \quad (2.37a)\end{aligned}$$

with Jacobians:

$$F_t \coloneqq \frac{\partial f(\mathbf{x}, \mathbf{u}_{t+T}, \mathbf{0})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_{t|t}} \qquad G_t^v \coloneqq \frac{\partial f(\hat{\mathbf{x}}_{t|t}, \mathbf{u}_{t+T}, \mathbf{v})}{\partial \mathbf{v}}\Big|_{\mathbf{v}=\mathbf{0}}.$$

The linearised measurement model is:

$$\begin{aligned}\mathbf{0} &\approx h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{y}_t, \mathbf{0}) + H_t(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t-T}) + H_t^e \mathbf{e}_t \\ &= H_t \mathbf{x}_t + h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{y}_t, \mathbf{0}) - H_t \hat{\mathbf{x}}_{t|t-T} + H_t^e \mathbf{e}_t. \quad (2.37b)\end{aligned}$$

with Jacobians:

$$H_t \coloneqq \frac{\partial h(\mathbf{x}, \mathbf{y}_t, \mathbf{0})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_{t|t-T}} \qquad H_t^e \coloneqq \frac{\partial h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{y}_t, \mathbf{e})}{\partial \mathbf{e}}\Big|_{\mathbf{e}=\mathbf{0}}.$$

The extended Kalman filter is given in Algorithm 2.3. A thorough derivation can be found in [28]. The standard Kalman filter can be obtained from Algorithm 2.3 by substituting:

$$-h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{y}_t, \mathbf{0}) = \mathbf{y}_t - H_t \hat{\mathbf{x}}_{t|t-T} \quad \text{and} \quad f(\hat{\mathbf{x}}_{t|t}, \mathbf{u}_{t+T}, \mathbf{0}) = F_t \hat{\mathbf{x}}_{t|t} - G_t^u \mathbf{u}_{t+T}.$$

2.4.2.3 Marginalised Particle Filter

Recursively estimating the posterior density $p(\mathbf{x}_t|\mathbf{Y}_t, \mathbf{U}_t)$ in case of a nonlinear state-space model (2.25) can be accomplished by the particle filter as outlined in Section 2.4.2.1. However, if the model is in the form of (2.30), that is it contains a conditionally linear substructure, subject to Gaussian noise, this can be exploited to obtain better estimates, *i.e.* estimates with reduced variance [106] and possibly reduced computational costs as well [107]. The method is referred to as marginalised particle filter (MPF) or Rao-Blackwellised particle filter. The theory is described, for instance, in [91–93]. Practical applications and experiences are gathered in [108]. The MPF has been used in the context of visual SLAM as well, see for instance [56, 109, 110]. Further applications are given in Section 2.4.3. The MPF is introduced briefly in the sequel. A thorough derivation can be found in [91, 111].

Consider a state vector $\mathbf{x}_t$ that can be partitioned according to:

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{x}_t^n \\ \mathbf{x}_t^l \end{bmatrix}, \quad (2.39)$$

Algorithm 2.3 Extended Kalman Filter (EKF)

1. Initialise the filter:
$$\hat{\mathbf{x}}_{0|-1} = \mathbf{x}_0 \quad \text{and} \quad P_{0|-1} = \mathrm{cov}(\mathbf{x}_0),$$
where the a priori pdf $p(\mathbf{x}_0)$ is Gaussian. Let $t \coloneqq 0$.

2. Measurement update:
$$\begin{aligned}
\mathbf{z}_t &= \mathbf{0} - h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{y}_t, \mathbf{0}) \\
S_t &= H_t P_{t|t-T} H_t^T + H_t^e R_t H_t^{eT} \\
K_t &= P_{t|t-T} H_t^T S_t^{-1} \\
\hat{\mathbf{x}}_{t|t} &= \hat{\mathbf{x}}_{t|t-T} + K_t \mathbf{z}_t \\
P_{t|t} &= (\mathrm{I} - K_t H_t) P_{t|t-T},
\end{aligned}$$
where $\mathbf{z}_t$ is referred to as *innovation* or *residual*, S_t as *innovation covariance*, and K_t as *Kalman gain*.

3. Time update:
$$\begin{aligned}
\hat{\mathbf{x}}_{t+T|t} &= f(\hat{\mathbf{x}}_{t|t}, \mathbf{u}_{t+T}, \mathbf{0}) \\
P_{t+T|t} &= F_t P_{t|t} F_t^T + G_t^v Q_t G_t^{vT}.
\end{aligned}$$

4. Let $t \coloneqq t + T$ and repeat from 2.

where $\mathbf{x}_t^n$ denotes the nonlinear part of the state space and $\mathbf{x}_t^l$ denotes the part that is linear Gaussian, if conditioned on $\mathbf{x}_t^n$. By marginalising out $\mathbf{x}_t^l$ from the full posterior density (2.23):

$$p(\mathbf{X}_t|\mathbf{Y}_t,\mathbf{U}_t) = p(\mathbf{x}_t^l,\mathbf{X}_t^n|\mathbf{Y}_t,\mathbf{U}_t) = \underbrace{p(\mathbf{x}_t^l|\mathbf{X}_t^n,Y_t,\mathbf{U}_t)}_{\text{KF}}\underbrace{p(\mathbf{X}_t^n|Y_t,\mathbf{U}_t)}_{\text{PF}}, \tag{2.40}$$

the particle filter can be used to obtain an approximate estimate of $p(\mathbf{X}_t^n|Y_t,\mathbf{U}_t)$ and the Kalman filter, since $\mathbf{X}_t^n$ is given, provides an optimal estimate of $p(\mathbf{x}_t^l|\mathbf{X}_t^n,Y_t,\mathbf{U}_t)$. The MPF is a clever combination of both algorithms. An alternative interpretation as a Kalman filter bank is developed in [111, 112].

Under the first interpretation, which is used in this thesis, the posterior is represented by a set of $n \gg 1$ particles $\{\mathbf{x}_t^{n,[i]}, \hat{\mathbf{x}}_t^{l,[i]}, P_t^{[i]}\}_{i=1}^n$, where each particle contains a Gaussian distribution $\mathcal{N}(\hat{\mathbf{x}}_t^l, P_t)$ and an associated importance weight $w_t^{[i]}$. Assuming that the system under consideration is modelled according to (2.30), the marginalised particle filter — in this representation inherently using a SIR filter (*cf.* Algorithm 2.2) — is given in Algorithm 2.4. Note that for the sake of notional brevity, the dependence of $\mathbf{x}_t^{n,[i]}$ in h_t, H_t, f_t, F_t^n and G_t^{vn} and the dependence of $\mathbf{u}_{t+T}^n$ in f_t given in model (2.30) are suppressed. The expressions for the case, where $\mathbf{v}_t^n$ and $\mathbf{v}_t^l$ are not mutually independent, can be found in [91]. The minimum variance estimate $\hat{\mathbf{x}}_{t|t}^n$ and the covariance $P_{t|t}^n$ of the nonlinear partition can be obtained from (2.36) by substituting $\mathbf{x}_{t|t}^{[i]} = \mathbf{x}_{t|t}^{n,[i]}$. The estimate and covariance of the linear partition are:

$$\hat{\mathbf{x}}_{t|t}^l = \sum_{i=1}^n w_t^{[i]} \hat{\mathbf{x}}_{t|t}^{l,[i]} \tag{2.42a}$$

$$P_{t|t}^l = \sum_{i=1}^n w_t^{[i]} \big(P_{t|t}^{[i]} + (\hat{\mathbf{x}}_t^{l,[i]} - \hat{\mathbf{x}}_{t|t}^l)(\hat{\mathbf{x}}_t^{l,[i]} - \hat{\mathbf{x}}_{t|t}^l)^T\big). \tag{2.42b}$$

As mentioned in Section 2.4.1.3, many practical applications allow for special cases of model (2.30). In the case of linear system dynamics and nonlinear measurements, one covariance matrix can be used for all particles. This leads to a substantial reduction in the computational complexity of Algorithm 2.4. Important special cases and their implications are treated thoroughly in [91, 107].

2.4.3 SLAM and Localisation

Simultaneous localisation and mapping (SLAM) is historically related to robot navigation: a robot acquires a map of its environment and simultaneously localises itself relative to this map, while all it is given is a sequence of measurements $\mathbf{Y}_t$ and controls $\mathbf{U}_t$. Measurements are typically fused from a number of sensors such as the global positioning system (GPS), laser rangefinder, radar or inertial sensors (sensor fusion). Localisation is the subproblem of determining the pose relative to a given map. It is therefore included in the SLAM problem. One of the first formalisations of the SLAM problem can be found in [113]. Since then extensive

Algorithm 2.4 Marginalised particle filter (MPF), inherently using a SIR filter

1. Initialise the particle set and the associated Gaussians according to step 1 of the SIR filter and the extended Kalman filter, introduced in Algorithm 2.2 and 2.3, respectively.

2. Measurement update:

   (a) Particle filter measurement update: use step 2 of the SIR filter with:

   $$\mathbf{x} \coloneqq \mathbf{x}^n$$
   $$p(\mathbf{y}_t|\mathbf{x}_t^{n,[i]}) = \mathcal{N}\big(\mathbf{y}_t; h_t + H_t\hat{\mathbf{x}}_{t|t-T}^{l,[i]},\, H_t P_{t|t-T} H_t^T + H_t^e R_t H_t^{eT}\big). \tag{2.41a}$$

   (b) Kalman filter measurement update: use step 2 of the EKF with:

   $$\hat{\mathbf{x}} \coloneqq \hat{\mathbf{x}}^{l,[i]}, \quad P \coloneqq P^{[i]}, \quad \mathbf{z}_t = \mathbf{y}_t - h_t - H_t\hat{\mathbf{x}}_{t|t-T}^{l,[i]}, \quad i = 1, \ldots, n.$$

   (c) Resample.

3. Time update:

   (a) Particle filter time update: use step 4 of the SIR filter with:

   $$\mathbf{x} \coloneqq \mathbf{x}^n, \quad \mathbf{u} \coloneqq \mathbf{u}^n,$$
   $$p(\mathbf{x}_{t+T}^n|\mathbf{x}_{t|t}^{n,[i]}, \mathbf{u}_{t+T}^n) = \mathcal{N}\big(\mathbf{x}_{t+T}^n; f_t^n + F_t^n\hat{\mathbf{x}}_{t|t}^{l,[i]}, F_t^n P_{t|t} F_t^{nT} + G_t^{vn} Q_t^n G_t^{vnT}\big).$$

   (b) Kalman filter *corrective measurement update*: the prediction of the nonlinear states in step 3a provides information about the linear states that can be incorporated by interpreting the nonlinear time update (2.30a) as an artificial measurement:

   $$\underbrace{\mathbf{x}_{t+T}^n - f_t^n}_{\mathbf{y}_t} = F_t^n\mathbf{x}_t^l + G_t^{vn}\mathbf{v}_t^n$$

   and, using this equation, performing a second corrective measurement update analogously to step 2b. Assume that the corrected linear states are denoted $\{\hat{\mathbf{x}}_{t|t}^{l,[i],*}, P_{t|t}^{[i],*}\}$.

   (c) Kalman filter time update: (*cf.* step 3 of the EKF)

   $$\hat{\mathbf{x}}_{t+T|t}^{l,[i]} = f_t^l + F_t^l\hat{\mathbf{x}}_{t|t}^{l,[i],*} + G_t^{ul}\mathbf{u}_{t+T}^l, \quad P_{t+T|t}^{[i]} = F_t^l P_{t|t}^{[i],*} F_t^{lT} + G_t^{vl} Q_t^l G_t^{vlT}.$$

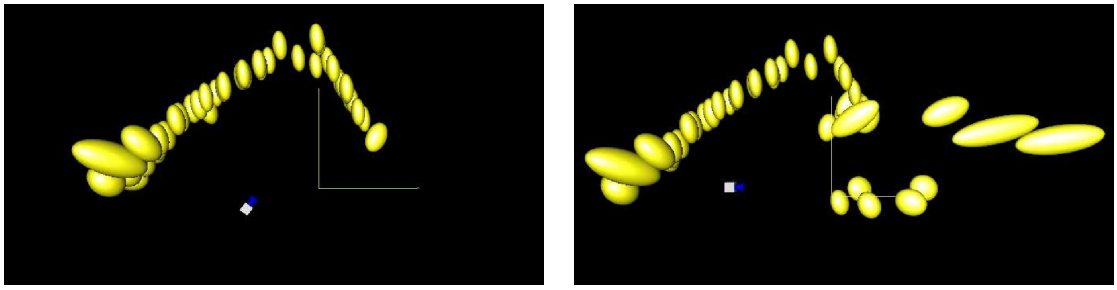
4. Let $t \coloneqq t + T$ and repeat from 2a.

research has been undertaken in the robotics community. A survey of the essential solution methods and significant implementations is given in [114]. Recent advances are gathered in [115]. The textbook [28] provides mathematical derivations and implementations for the most important algorithms in probabilistic robotics. Localisation and mapping are first treated separately and then in combination yielding SLAM. The *online* — in contrast to the *full* — localisation problem involves the estimation of the momentary pose — in case of SLAM along with the map — and a representation of the confidence in the estimate without storing the entire history of measurements, controls or states. The tools for achieving this are given in Section 2.4.1 and 2.4.2, respectively. Considering the structure from motion problem (*cf.* Section 2.3.4), where the camera trajectory and the scene structure are estimated from an image sequence, and model-based camera tracking (*cf.* Section 2.3.5), where the camera trajectory is estimated relative to a given 3D model of the target scene, both by using computer vision techniques, the parallels to the SLAM and the localisation problem in robotics are obvious. However, it should be noted that, among others, one major difference complicates vision-based tracking compared to robot localisation: unless additional sensors are used, there are no control inputs $\mathbf{U}_t$ available to predict the pose of a handheld camera. Nowadays, the computer vision and the robotics community are closing in on each other: Recursive filters as well as sensor fusion are frequently exploited concepts in computer vision [44, 98, 99, 105, 116–121] and the use of cameras is widespread in robotics [109, 122–125]. The today established notions *visual* and *monocular SLAM* are evidence of this.

The remainder of this section gives examples of how the problem of camera pose and structure estimation has been solved using concepts developed in robotics. A brief introduction and critical discussion of the following algorithms — playing an important role in this thesis — is given: the localisation based on the extended Kalman filter denoted *EKF-Localisation*, the most classical SLAM approach, the *EKF-SLAM*, and two advanced SLAM techniques, *FastSLAM* and *MPF-SLAM*.

2.4.3.1 EKF-Localisation

It is straightforward to put the problem of real-time model-based camera tracking in the nonlinear state-space form (2.25). Consider the moving camera as a dynamic system, where the state vector $\mathbf{x}_t$ contains information related to the momentary pose and kinematics of the camera and where the observations $\mathbf{Y}_t$ are obtained by processing images (*cf.* Section 2.1.4). A recursive filter such as the extended Kalman filter can then be used to estimate the camera trajectory. The major benefit from using the recursive filtering framework instead of the computer vision techniques introduced in Section 2.3.1.1 lies in the possibility to perform sensor fusion. This is easily achieved by formulating one specific measurement equation for each sensor and opens up for compensating for the lack of control inputs $\mathbf{U}_t$ in vision-only tracking. Visual-inertial sensor fusion is most frequently proposed in literature and the advantages of this special sensor combination are well-proven, see for instance [2, 44, 121] and the references therein. In outdoor environments GPS has been used, *e.g.* in [84, 120, 126, 127]. An efficient and robust markerless visual-inertial camera localisation system is developed in Chapter 3.



(a) *Consistent map.* ***(b)*** *Corrupted map.*

Figure 2.12: *EKF-SLAM: (a) illustrates a consistent map reconstructed by moving a camera in front of a corner of a room. The mapped point landmarks are shown with their location covariances. (b) presents the corruption of the map after an erroneous measurement update.*

2.4.3.2 EKF-SLAM

EKF-SLAM denotes the classical recursive solution to the structure from motion problem. The key idea is to augment the state $\mathbf{x}_t$, as introduced in Section 2.4.3.1, with a map $\mathbf{m}_t$ of the environment and to estimate the joint posterior $p(\mathbf{m}_t, \mathbf{x}_t | \mathbf{Y}_t, \mathbf{U}_t)$ using the extended Kalman filter. The map is typically represented by a vector composed of the 3D locations $\mathbf{m}_t^{(i)}$ of point landmarks, but also other parametrisations such as lines [39, 40] or surfaces [128, 129] have been proposed. Clearly, for any reasonable number of features, the vector $\mathbf{m}_t$ is significantly larger than $\mathbf{x}_t$ that is being estimated in Section 2.4.3.1, resulting in an enormous computational complexity of the filter algorithm.

The EKF solution to visual SLAM is widespread in literature [39, 40, 105, 130] and several topics have been addressed such as computational complexity [118, 131, 132], loop closing and map consistency [118, 132, 133] and robustness and relocation [33, 134–136]. See [114] for a state of the art of the essential solution methods. Visual-inertial sensor fusion is also investigated in this context [137–139].

Besides the topics mentioned above, one major problem inherent to the EKF-SLAM approach is its sensitivity to incorrect measurements due to the joint estimation of the camera states and the map in one state vector. Since the camera states and the map have to be treated simultaneously in the measurement update, one incorrect measurement cannot only corrupt the camera states momentarily, it can also introduce an irreparable inconsistency into the map. This will inevitably lead to a filter divergence and require a complete initialisation. An example is given in Figure 2.12. This effect is also discussed, for instance, in [116, 140].

2.4.3.3 FastSLAM

FastSLAM, as first introduced in [141] and extended in [142], is an algorithm that utilises the idea of the marginalised particle filter to obtain a solution to SLAM that scales with the number

of features. Let N be the number of features $\mathbf{m}_t^{(i)}$ in the map $\mathbf{m}_t$ and let $\mathbf{x}_t$ denote the camera pose. The key idea pertains then to the fact that the full posterior $p(\mathbf{m}_t, \mathbf{X}_t|\mathbf{Y}_t, \mathbf{U}_t)$ can be factorised into $N + 1$ densities:

$$p(\mathbf{m}_t, \mathbf{X}_t|\mathbf{Y}_t, \mathbf{U}_t) = p(\mathbf{X}_t|Y_t, \mathbf{U}_t) \prod_{i=1}^{N} p(\mathbf{m}_t^{(i)}|\mathbf{X}_t, \mathbf{Y}_t, \mathbf{U}_t). \tag{2.43}$$

Stated differently, the features $\mathbf{m}_t^{(i)}$ are mutually independent, given the camera trajectory $\mathbf{X}_t$. Note that the observation (2.43) has the same structure as (2.40). Utilising this structure and assuming that the camera trajectory is represented by a particle set, N separate Gaussians, one for each individual feature, can be associated to each particle representing the map. The theory and implementations of the FastSLAM algorithm are thoroughly treated in the recent textbook [28] and a comparison to the EKF-SLAM approach can be found in [143].

Utilising the FastSLAM algorithm — originally devised to solve the SLAM problem for mobile robots, where the dimension of the state space is small — in monocular SLAM is a recent trend in computer vision as well as in robotics [109, 117]. However, one major problem of FastSLAM should be noted. The correlations between the features and the camera pose are coded in the diversity of the particles and their associated maps. During the resampling step, as particles are removed, covariance information is lost. Over time this effect — referred to as *particle depletion* — can lead to an inconsistent map estimate and to a limited ability to close loops [144]. Hence, in the context of FastSLAM, it is of special interest to prolong the survival times of the particles. This topic is treated in Chapter 5.

2.4.3.4 MPF-SLAM

In this thesis, *MPF-SLAM* refers to a SLAM system, where the marginalised particle filter is used to estimate the posterior distribution $p(\mathbf{m}_t, \mathbf{X}_t|\mathbf{Y}_t, \mathbf{U}_t)$. Such a system is developed in [56, 110] for being able to exploit inertial data together with FastSLAM. The MPF-SLAM algorithm is a generalisation of FastSLAM in the respect that, besides the landmark positions, also a conditionally linear Gaussian substructure $\mathbf{x}_t^l$, present in the dynamic system of the camera can be marginalised. For this purpose, the dynamic system has to be put in the form of model (2.30). The concept of the MPF-SLAM algorithm is explained by the following factorisation:

$$p(\mathbf{m}_t, \mathbf{x}_t^l, \mathbf{X}_t^n|\mathbf{Y}_t, \mathbf{U}_t) = p(\mathbf{X}_t^n|Y_t, \mathbf{U}_t)p(\mathbf{x}_t^l|\mathbf{X}_t^n, Y_t, \mathbf{U}_t) \prod_{i=1}^{N} p(\mathbf{m}_t^{(i)}|\mathbf{x}_t^l, \mathbf{X}_t^n, \mathbf{Y}_t, \mathbf{U}_t) \tag{2.44}$$

and the idea behind it is to extend the FastSLAM algorithm to be able to cope with a state vector of a higher dimension. This is achieved by exploiting the benefits of the marginalised particle filter compared to the standard particle filter as mentioned in Section 2.4.2.3. Being able to cope with a higher state dimension is of special interest in the context of visual-inertial SLAM, where, besides the momentary pose, kinematics and biases related to the camera and

the inertial sensors have to be estimated as well. The robustness and performance of such a visual-inertial system is investigated in Chapter 5. The problem of loosing covariance information with the particle trajectories applies for MPF-SLAM as well.

3

Model-based Visual-Inertial Tracking

This chapter is devoted to developing a hybrid model-based camera tracking system, which fuses measurements from a camera and low cost inertial sensors, *i.e.* from a camera IMU system as presented in Section 2.2. The fusion is based on the extended Kalman filter (EKF) (*cf.* Section 2.4.2.2). This work was performed within the EU project MATRIS [66], where the objective was to develop a visual-inertial tracking system that required no markers or other special infrastructure in the environment (*cf.* Section 2.1.4). To achieve this, a markerless top down image processing approach relying on a 3D model of the target environment is used.

The aim of visual-inertial sensor fusion is to overcome some fundamental limitations of vision-only tracking:

High computing power: The image processing used to generate camera measurements (*cf.* Section 2.1.4) is computational intensive, especially without a continuous image sequence and a reasonable prediction of the camera pose. By exploiting kinematic information provided by inertial sensors, a more accurate prediction of the camera pose can be obtained. The improved prediction allows for reducing the computational cost of the image processing and the demand for camera measurements in general.

Slow camera motions: Pure vision-based tracking systems usually can handle only slow camera motions. This is to some extent due to the computational demand, but also due to inherent camera characteristics such as motion blur, which disturbs the feature registration, and the comparatively low sample rate. Inertial sensors can be sampled at a much higher rate than cameras and the algorithms for processing their measurements are less computationally intense (*cf.* Section 2.2.1). They are suitable for capturing fast movements, while the camera provides an absolute reference that can compensate for the drift in the integrated inertial data (*cf.* Section 2.2.1.3).

Occlusion: If too many features are occluded, the camera pose cannot be estimated using

computer vision techniques (*cf.* Section 2.3.1). Inertial sensors — in general — always deliver measurements. The pose can therefore be updated at any time, even though the accuracy of the estimate deteriorates quickly without an external reference, which can be provided by the camera measurements.

Jitter: Computer vision techniques tend to introduce jitter, *i.e.* unwanted variations, into the motion estimates. The reason is that the current pose estimate usually relies only on the currently visible feature set, while past poses have no significant influence. The amount of jitter increases when the feature set changes significantly between frames. This is typically the case, when the outlier rate is high and a robust estimation technique such as RANSAC (*cf.* Section 2.3.3) is used. In recursive filtering (*cf.* Section 2.4.2), the influence of past poses can be increased via the process noise and the influence of measurements can be lessened via the measurement noise so that a smoother trajectory can be obtained. Moreover, an accurate pose prediction simplifies the outlier rejection.

Given the considerations above, it is obvious that cameras and inertial sensors have complementary properties and thus provide a promising sensor combination. The advantages of visual-inertial sensor fusion are extensively treated in literature, especially in the field of augmented reality. Various methods have been proposed, many of them using the extended Kalman filter [26, 44, 121, 137, 145–149]. An open question is, however, whether and how accelerometer readings should be incorporated into the pose estimation. Resolving this issue is one major topic of this chapter. Different state-space models are therefore designed and evaluated within the visual-inertial tracking system developed in this chapter.

Section 3.1 gives an overview of the visual-inertial tracking framework. The individual building blocks, initialisation, markerless image processing and sensor fusion are treated in Sections 3.2 through 3.4. Section 3.5 describes how a 3D model of the target environment can be acquired and discusses further issues that arise when setting up the system. In Section 3.6, the results are evaluated both under controlled conditions and in real world environments. Conclusions are drawn in Section 3.7.

3.1 Approach

The architecture of the visual-inertial tracking framework is outlined in Figure 3.1. The CamIMU (*cf.* Section 2.2) provides synchronised inertial data and camera images. The image processing component generates point correspondences between the 3D model of the target scene and the camera images. The extended Kalman filter is then used to fuse these with the measurements from the IMU to obtain a pose estimate. The pose estimates are exploited for the markerless image processing, *i.e.* a top down approach is used, thus making the vision and the sensor fusion system tightly coupled.

The workflow of the software system is specified in Figure 3.2. At time t, a data packet consisting of the camera frame t and the sequence of IMU readings for the interval $[t-(r-1)T, t]$ is captured. Here, T denotes the sample time of the IMU and rT denotes the sample time of

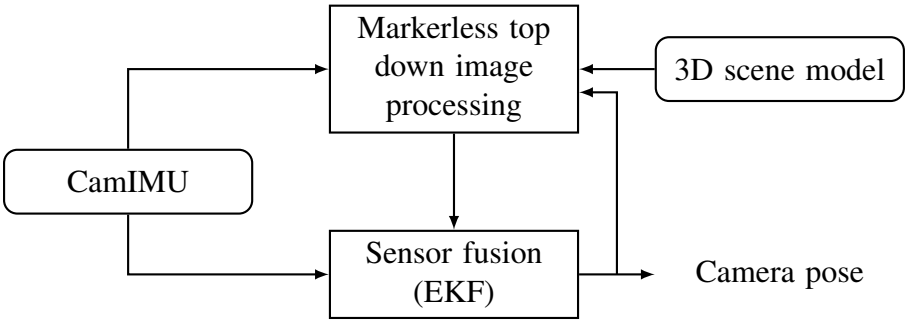


Figure 3.1: *Architecture of the visual-inertial tracking framework.*

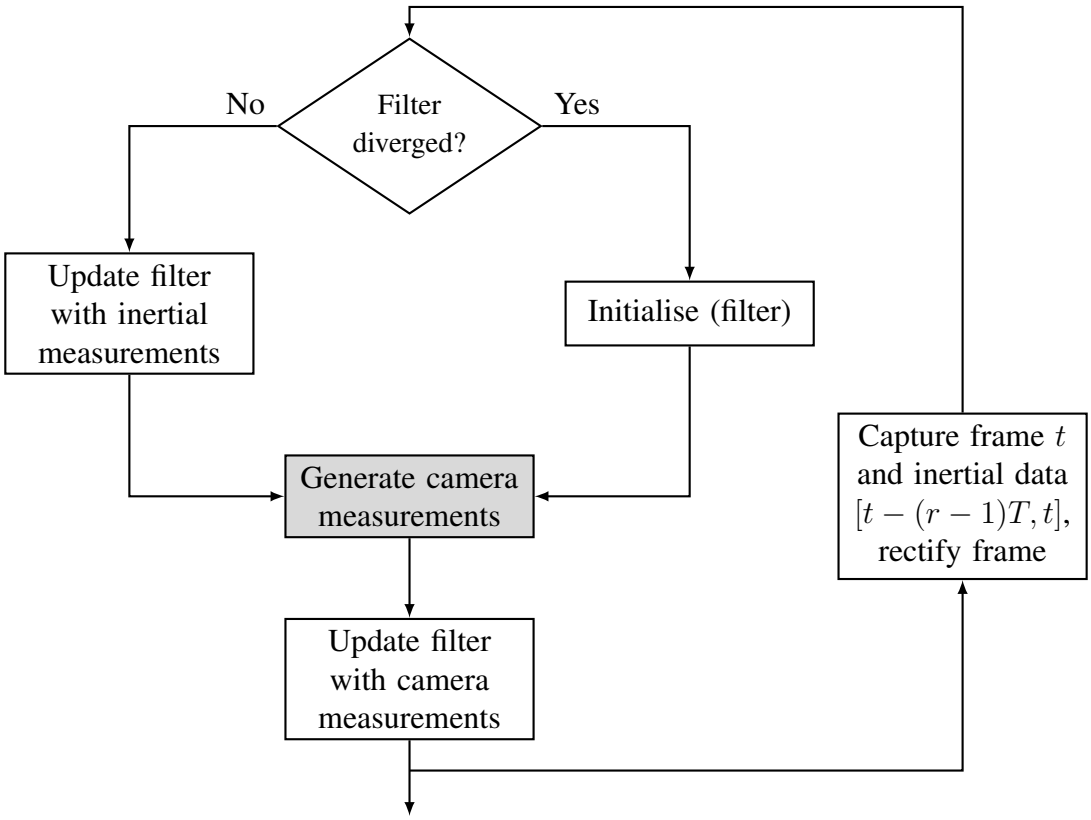


Figure 3.2: *Workflow of the visual-inertial tracking framework. The grey box is refined in Figure 3.3.*

the camera. The image is resampled to compensate for distortion (*cf.* Section 2.1.1.2). A divergence test is performed on the state estimate, based on which the system either initialises or processes the inertial measurements up to time t. Either way a pose estimate is obtained and used for generating point correspondences between the 3D model of the environment and the camera image. The extended Kalman filter is then updated using the correspondences. The initialisation of the tracking system is discussed in Section 3.2. The top down image processing approach used to generate the camera measurements is described in Section 3.3. Section 3.4 covers the fusion of camera and IMU measurements as well as related topics such as outlier rejection and divergence monitoring.

3.2 System Initialisation

The system is said to be initialised when the estimate of the camera pose is based purely on prior information instead of a pose prediction as during the ordinary *predictive tracking* loop. In robotics, the initialisation phase is referred to as *global localisation*. The result of this phase is an initial estimate that must meet the following requirements: it is appropriate to initialise the extended Kalman filter, *i.e.* the error of the estimate is known and can be represented by a Gaussian, and it is reasonably accurate to initialise the image processing method used. The tracking system developed in this thesis provides three different initialisation modes that vary in their degree of automation.

3.2.1 Manual

The easiest way to initialise the system is to enter a fixed initial orientation and position and to align the pose of the CamIMU device by hand so that the feature registration snaps on. This method is sufficient for small-scale desktop environments and has the advantage of requiring no special preparation.

3.2.2 Semi-Automatic

In the absence of severe magnetic disturbances, for instance caused by large amounts of ferrous material in the target scene, the 3D orientation provided by the IMU (*cf.* Section 2.2.1) can be used to initialise the rotational states. In this case, only the position of the CamIMU has to be aligned by hand. The semi-automatic method is more convenient, however, it requires calibration of the relative rotation between the global frame and the frame defined by the 3D model of the target scene, the world frame (*cf.* Section 2.2.2.4).

3.2.3 Fully Automatic

A fully automatic initialisation method based on a database of *reference images* and bottom up feature matching (*cf.* Section 2.1.4.1) has been developed [150, 151]. It is intended for cases

where the manual method is in-sufficient — *e.g.* in large-scale environments — and the IMU cannot be used due to magnetic disturbances. Similar techniques are used in [31, 49, 152].

A reference image is here defined as a camera frame with a known pose and an associated set of distinctive features — this thesis uses SIFT (scale invariant feature transform) [48] features — with known 3D coordinates. In order to handle multiple reference images efficiently, the full-automatic initialisation process is divided into two steps: First, the database is queried with the current camera frame to find the closest reference image using a similarity measure based on colour histograms. An advanced similarity measure using local histograms has been developed in the MATRIS project [153]. In the second step, the pose obtained from the reference image is verified and refined using the associated features. These are matched against the features extracted from the live image and an improved pose estimate is computed from the resulting point correspondences (*cf.* Section 2.3.1) using the RANSAC algorithm (*cf.* Section 2.3.3). The details of this method are given in [5, 154]. The fully automatic approach requires a sufficiently dense database of reference images.

3.3 Top Down Image Processing

In order to obtain an efficient tracking method, it is essential to exploit the pose prediction provided by the sensor fusion system in the markerless image processing. This is achieved by the *analysis-by-synthesis* technique. With this technique, a 3D model of the target scene, typically a CAD model, is used to predict the appearances of features in the camera images, usually by projecting the model from the predicted camera poses. Assuming that the 3D model is sufficiently accurate, the performance of this image processing approach depends on the accuracy of the pose prediction: A more accurate prediction reduces the computational demand and increases the accuracy of the feature registration. The analysis-by-synthesis technique has been proposed earlier, mostly using a geometric CAD model, a *wireframe* model, of the target scene in combination with edge features [87, 147, 155]. Textured CAD models have been used as well [44, 156].

In this thesis, a textured CAD model of the target scene is used to predict the local appearances of corner features. The model is projected from the predicted camera pose in order to obtain a synthetic image that resembles the camera frame. For establishing correspondences, small *texture windows (patches)* of the synthetic image, centred around the projected locations of the corner features, are registered in the camera frame using a variation of the Kanade-Lucas tracker (*cf.* Section 2.1.4.2) that includes an affine photometric model. In order to increase the convergence radius, the registration is performed in a coarse-to-fine fashion using image pyramids (*cf.* Section 2.1.4.2). This relaxes the requirements with respect to the accuracy of the predicted pose and, hence, increases the robustness of the overall system as well as its ability to initialise from a rough initial pose.

The workflow of the analysis-by-synthesis approach is outlined in Figure 3.3. A simple two-sided test object is here used as target. The middle part of the figure denotes the sequence of operations that are performed on the image data illustrated on both sides. The left side refers

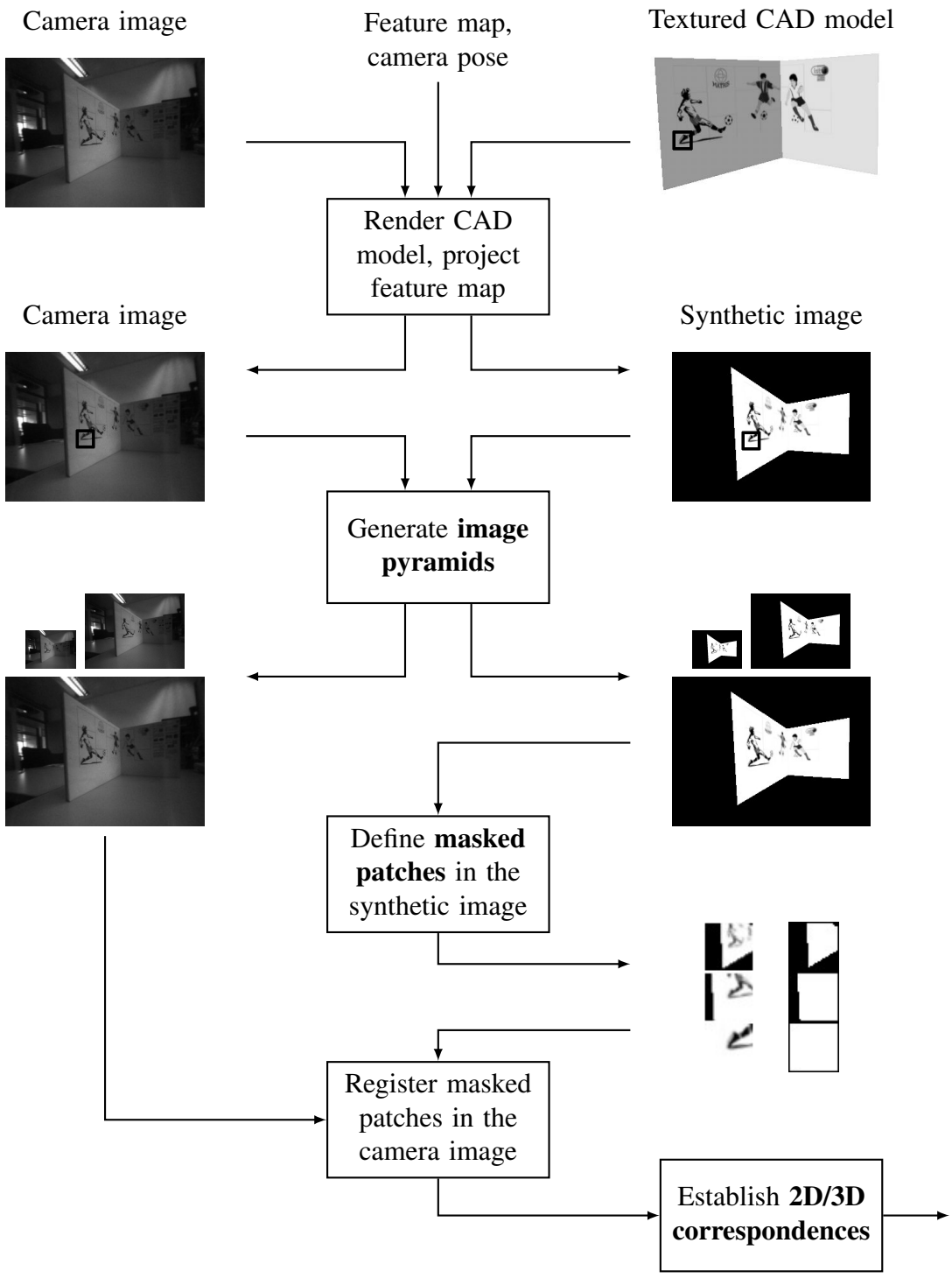


Figure 3.3: *Workflow of the top down image processing approach.*

to the camera image and the right side to the synthetic image. The registration is performed in greyscale.

The process starts with projecting the CAD model from the predicted camera pose. This is done by *rendering* the model on the graphics card using *OpenGL (open graphics library)* [157]. The image intensities are then obtained from the *framebuffer*. Note that the background colour — black in Figure 3.3 — appears in the synthetic image in regions that are not covered by the model. These regions will be excluded from the registration as described in the sequel.

Assume that a feature map, *i.e.* a set of suitable 3D points on the model, is given. This map is then projected to find a subset of features that are visible in the camera frame. One such feature is indicated before and after the projection in Figure 3.3.

In the next step the image pyramids are generated for both the camera image and the synthetic image. Two additional resolution levels are generated in Figure 3.3. Depending on the expected feature displacements, the number of levels can be varied, *e.g.* it can be increased during the initialisation phase. The rendering process introduces aliasing effects. If, in addition, the camera is out of focus, the registration can be stabilised by lowpass filtering the synthetic image before constructing the image pyramid so that both images are equally blurred.

Rectangular patches, centred around the predicted positions of the visible features, are then defined on each level of the synthetic image pyramid. Figure 3.3 shows the three patch resolutions defined for the marked feature. Note that parts of the background appear in the higher resolutions. In order to exclude these from the registration, a binary mask is generated for each patch using the *depth buffer* [1].

The masked patches are then registered in the camera image applying the coarse-to-fine strategy introduced in Section 2.1.4.2. Figure 3.4 shows the result of a successful registration.

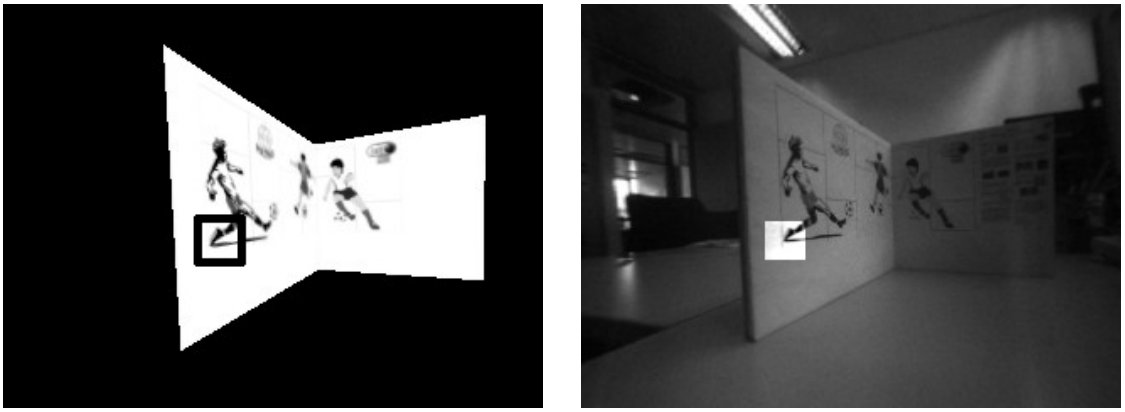
The last step is to establish 2D/3D correspondences by associating the 3D locations of the registered features to their 2D locations in the camera image, whereat the latter are normalised using (2.3b).

The mathematical details of the iterative patch registration are given in Section 3.3.1. The management of the feature map is treated in Section 3.3.2 and Section 3.3.3 presents results.

3.3.1 Iterative Patch Registration

This section provides the mathematical details for registering a masked patch of a synthetic image in a camera image given a predicted position. Assuming that the CAD model and the camera pose used to generate the synthetic image are both reasonably accurate, the synthetic image resembles the camera image with respect to content and perspective distortion. This is a suitable condition for an iterative registration technique such as the Kanade-Lucas Tracker (KLT) (*cf.* Section 2.1.4.2). However, due to changing light conditions and the photometric response function of the camera used, the effective intensities of the synthetic image can differ significantly from those of the camera image (*cf.* Figure 3.4). It turns out that this effect can be

[1] The depth buffer is written during the rendering process and indicates background pixels by the value 1



(a) *Synthetic image and considered patch.* ***(b)*** *Registration in the camera image.*

Figure 3.4: *Patch registration: (a) shows the synthetic image, with the considered patch marked by a rectangle, and (b) illustrates the spatial and photometric registration of this patch in the camera image. Note how the contrast and brightness of the camera image has been adjusted locally during the photometric registration.*

compensated for locally by extending the basic KLT with a relatively simple affine photometric model that has been used earlier in [32].

Let $I(\mathbf{m}_p)$ be the intensity value of pixel $\mathbf{m}_p$ in the synthetic image and $J(\mathbf{m}_p)$ analogously in the camera image. The relation between the pixel intensities is modelled as:

$$I(\mathbf{m}_p) = \lambda J(\mathbf{m}_p + \mathbf{d}) + \delta \qquad \forall \mathbf{m}_p \in W, \tag{3.1}$$

where $\mathbf{d} = [d_x, d_x]^T$ denotes the displacement, λ can be regarded as a parameter adjusting the contrast and δ as a parameter adjusting the brightness of the texture window W. Here, W denotes the masked patch area of the synthetic image. The registration problem is then to find the parameter vector $\alpha = [d_1, d_2, \lambda, \delta]^T$ that minimises:

$$\hat{\alpha} = \arg\min_{\alpha} = \sum_{W} \big(I(\mathbf{m}_p) - \lambda J(\mathbf{m}_p + \mathbf{d}) + \delta\big)^2 \qquad \forall \mathbf{m}_p \in W. \tag{3.2}$$

Substituting $\lambda J(\mathbf{m}_p + \mathbf{d}) + \delta$ for its first order Taylor expansion and setting the derivatives of (3.2) with respect to α to zero, results in a linear system that can be solved for α:

$$G\alpha = \mathbf{b}, \tag{3.3a}$$

where

$$G = \sum_{W} \mathbf{s}(\mathbf{m}_p)\mathbf{s}(\mathbf{m}_p)^T \qquad \forall \mathbf{m}_p \in W \tag{3.3b}$$

$$\mathbf{b} = \sum_{W} \big(I(\mathbf{m}_p) - J(\mathbf{m}_p)\big)\mathbf{s}(\mathbf{m}_p) \qquad \forall \mathbf{m}_p \in W, \tag{3.3c}$$

with

$$\mathbf{s}(\mathbf{m}_p) = \begin{bmatrix} I_x(\mathbf{m}_p) & I_y(\mathbf{m}_p) & I(\mathbf{m}_p) & 1 \end{bmatrix}^T . \tag{3.3d}$$

Here, $I_x(\mathbf{m}_p)$ and $I_y(\mathbf{m}_p)$ denote the horizontal and vertical image gradient in $I(\mathbf{m}_p)$. The details of the linearisation can be found, for instance, in [58] and in the recent book [13]. In order to obtain an accurate estimate $\hat{\alpha}$, Newton-Raphson-style iterations of (3.3) are performed until either a maximum number of iterations is exceeded or the change in $\hat{\alpha}$ is negligible. Note that subpixel accuracy can be achieved with this method.

Initial values for the photometric parameters λ and δ can be obtained from the mean and the standard deviation of a local window centred around the predicted position in the synthetic and the camera image, respectively. The registration is assumed valid, if the final residual is below a threshold. This threshold depends on the properties of the target scenario and can be tuned experimentally.

3.3.2 Feature Management

This section treats the feature management in the visual-inertial tracking framework presented in this chapter. Several aspects of this topic are discussed: structure of the feature map, detection, deletion, distribution and number of features in the map.

The structure of the feature map is simple. It is a set of 3D points — each one denoting a feature — on the surface of the CAD model, preferably at distinctive corners in the texture. The features need no special descriptors, as the patches used for the registration are obtained from the synthetic images. The management of the feature map is implemented as a set of elementary rules.

In order to ensure a minimum number of visible features in the map, new features are added, if too few are predicted to be visible in the current camera image. Corner points are detected in free and valid areas of the synthetic image using the simple but efficient corner detector FAST (Features from Accelerated Segment Test) [158]. An area is said to be free and valid if it contains no background pixels and is not already covered by another feature in the map. In order to obtain an even distribution of features in the images, a minimum pixel distance is enforced between the features during the detection. The depth buffer can be used to obtain the 3D coordinates corresponding to the detected corners. The 3D coordinates are then added to the map.

Features are deleted from the map, if their registration fails in more than 50% of the frames, where they should be visible.

Another facet of feature management is to select an appropriate subset of the visible features to try to register in the current image. This plays an important role in real-time systems, since registering all visible features is computationally costly. It is particularly important, when the viewing distance to the target object varies significantly. This is exemplified in Figure 3.5(e,f), where many features are added to the map when the camera is close to the target object, while, at a large distance, most of them become redundant in the sense that their

registration will not improve the pose estimate. In order to avoid the generation of redundant camera measurements, a minimum pixel distance is also enforced during the registration. More advanced, but also significantly more expensive methods are proposed, for instance, in [159, 160].

In applications, where the viewing distance does not change significantly, it is advantageous to select an appropriate static feature map offline in order to save computational power for feature management and detection.

3.3.3 Results

Figure 3.5 demonstrates the potential of the model-based markerless image processing method developed in this section. The features on the target object are registered under a wide range of possible light conditions and views. In Figure 3.5, successful registration is shown in under- and overexposed camera images (a,b), from nearly orthogonal views onto the sides of the target object (c,d) and from significantly different viewing distances of $10\,\mathrm{cm}$ and $350\,\mathrm{cm}$, respectively, (e,f).

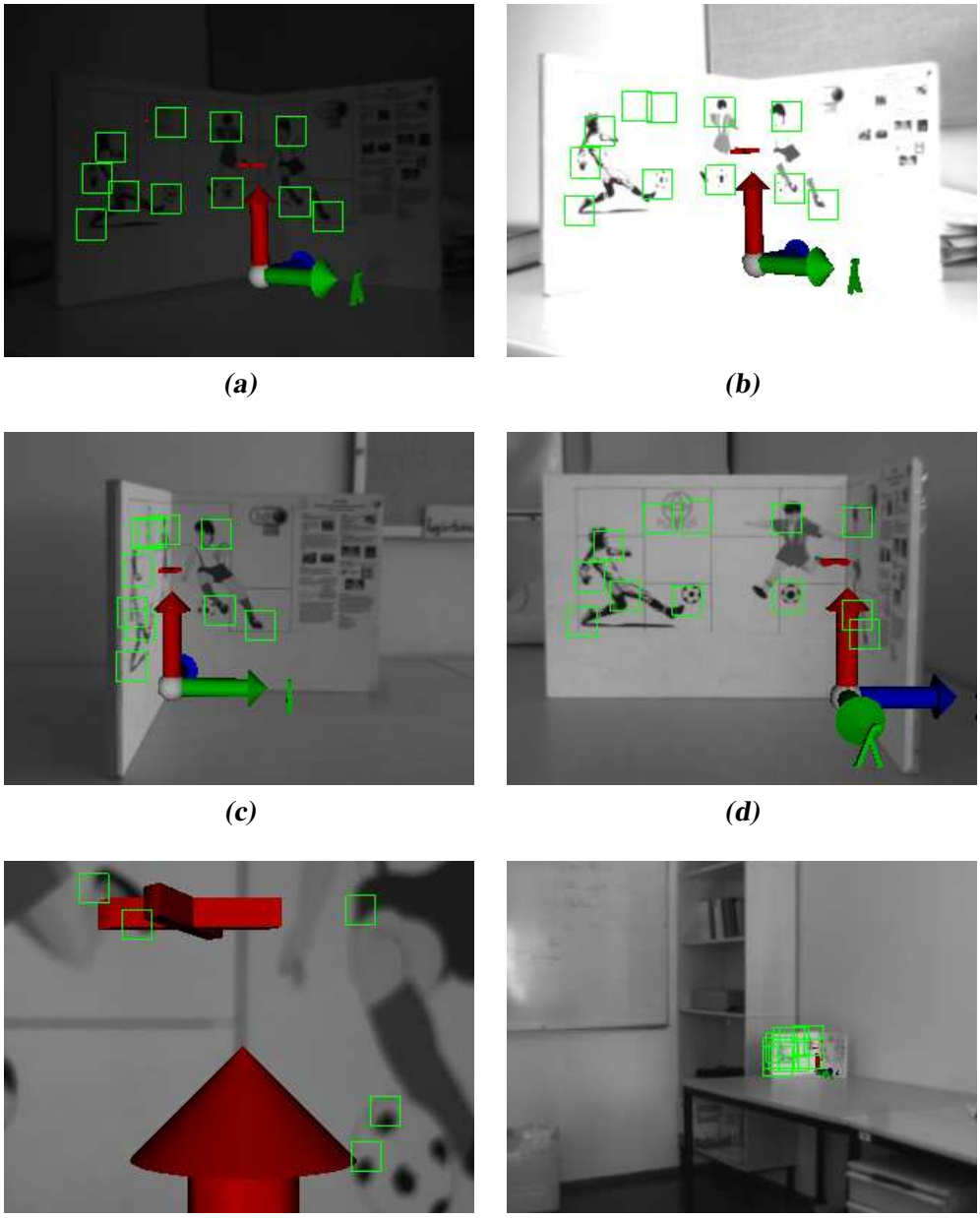
This performance is obtained using two crucial techniques: the affine photometric model used for registering the synthetic patches in the camera images and the analysis-by-synthesis technique used to construct those patches appropriately. The affine photometric model allows for tracking features under varying light conditions. The analysis-by-synthesis technique makes sure that, given a reasonably accurate pose prediction, the correct level of detail and the correct perspective distortion of the patches are obtained. This makes successful feature registration from a variety of different view points possible. Moreover, this technique provides further advantages: the graphics hardware can be utilised to project the CAD model efficiently, the feature tracking is drift-free and the pose prediction provided by the sensor fusion system is exploited to an extremely high degree to improve the efficiency of the registration.

However, it is important to note that these advantages are conditioned on a sufficiently accurate prediction of the camera pose and model of the target scene. These aspects are treated in Section 3.4 and 3.5, respectively.

3.4 Sensor Fusion

This section is concerned with combining camera and inertial measurements using the extended Kalman filter (*cf.* Section 2.4.2.2) to obtain the camera pose. The objective of this section is to develop a fusion strategy that allows for a tight integration of the camera and the inertial sensors and that provides an accurate pose prediction that can be fed to the vision system so that a stable and efficient final system is obtained.

Systems have been reported with a loose coupling between the vision and inertial subsystem: in [147] inertial sensors are used as prediction devices, in [149] as backup, in [44, 126, 161] the 2D/3D correspondences are used to obtain a camera pose or motion measurement that is then fed to an extended Kalman filter. In the system developed here, a tight integration of the



(a) *(b)* *(c)* *(d)* *(e)* *(f)*

Figure 3.5: *Augmented camera images showing the successful feature registration (indicated by green rectangles) and pose estimation (indicated by the pose of the augmented coordinate system) under different light conditions and varying viewing angles and distances with respect to the target object.*

camera and the inertial sensors is obtained by fusing the respective measurements, 2D/3D correspondences, 3D angular velocities and 3D linear accelerations uniformly with an extended Kalman filter. Outliers among the camera measurements are also handled in the filter.

Furthermore, the proposed systems differ in whether and how accelerometers are used for the pose estimation: in [147, 148, 162] only gyroscopes are used, in [44] accelerometers are modelled as inclinometers in the fashion of an attitude and heading reference system (*cf.* Section 2.2.1) and in [26, 121] the measurements are modelled according to (2.15) as in a strapdown inertial navigation system. Henceforth, the above fusion strategies are denoted *gyro*, *gravity* and *acceleration model*, respectively. As accelerometers can introduce a severe drift in the position estimate (*cf.* Section 2.2.1.3), it is not clear, whether they improve the tracking performance or introduce instabilities. In particular, this may depend on the accuracy and performance of the image processing approach used. Another open question is, whether the inertial data should be treated as measurements or as control inputs. Both approaches are encountered in literature [121, 139].

In order to solve these issues for the visual-inertial tracking framework developed here, four state-space models, representing the different fusion strategies mentioned above, are developed in Section 3.4.1. An experimental performance evaluation under controlled conditions and in real world environments is provided in Section 3.6. Outlier rejection and divergence monitoring are treated in Section 3.4.2 and 3.4.3, respectively.

3.4.1 Tracking Models

Four state-space models, representing different fusion strategies, are developed incrementally in the sequel using the notation of Section 2.4.1. The models are denoted gyro, gravity, acceleration and acceleration input model.

3.4.1.1 Gyro Model

In the gyro model, camera measurements, *i.e.* 2D/3D point correspondences, and gyroscope measurements, *i.e.* 3D angular velocities are fused. Both types of observations are represented by a separate measurement equation. The state vector $\mathbf{x}$ comprises:

$$\mathbf{x}^T = \begin{bmatrix} \mathbf{s}_w^T & \dot{\mathbf{s}}_w^T & \mathbf{q}_{sw}^T & \boldsymbol{\omega}_s^T & \mathbf{b}_s^{\omega T} \end{bmatrix}, \tag{3.4}$$

where $\mathbf{s}_w$ denotes the position, $\dot{\mathbf{s}}_w$ the velocity, $\mathbf{q}_{sw}$ the orientation quaternion, $\boldsymbol{\omega}_s$ the angular velocities and $\mathbf{b}_s^{\omega}$ the gyroscope biases of the IMU. Note that, in order to simplify the expressions in the time update, the position $\mathbf{s}_w$ and orientation $\mathbf{q}_{sw}$ of the IMU with respect to the world frame are estimated in the state vector. However, the camera pose is easily obtained from the state vector using (2.12) with the calibrated hand-eye transformation. Rotations are here represented by unit quaternions. This parametrisation is not minimal, implying that the constraint of unit magnitude has to be enforced (*cf.* Section 3.4.3). However, in contrast to the axis angle form, this representation is singularity-free. Moreover, it provides linear expressions for the concatenation of rotations (*cf.* Appendix B.2) and a simple expression for the

rotation and its derivative (*cf.* Appendix B.2). The process model assumes constant velocity and constant angular velocity. It defines no control inputs. The model is nonlinear but has a linear Gaussian substructure:

$$\begin{bmatrix} \mathbf{s}_{w,t+T} \\ \dot{\mathbf{s}}_{w,t+T} \\ \mathbf{q}_{sw,t+T} \\ \boldsymbol{\omega}_{s,t+T} \\ \mathbf{b}^{\omega}_{s,t+T} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} + \frac{T^2}{2}\mathbf{v}^{\ddot{s}}_{w,t} \\ \dot{\mathbf{s}}_{w,t} + T\mathbf{v}^{\ddot{s}}_{w,t} \\ \exp\left(-\frac{T}{2}(\boldsymbol{\omega}_{s,t} + \mathbf{v}^{\omega}_{s,t})\right) \odot \mathbf{q}_{sw,t} \\ \boldsymbol{\omega}_{s,t} + \mathbf{v}^{\omega}_{s,t} \\ \mathbf{b}^{\omega}_{s,t} + \mathbf{v}^{b^{\omega}}_{s,t} \end{bmatrix}, \tag{3.5}$$

where $\mathbf{v}^{\ddot{s}}_{w,t}$, $\mathbf{v}^{\omega}_{s,t}$ and $\mathbf{v}^{b^{\omega}}_{s,t}$ denote time independent Gaussian process noise that is uncorrelated in all components. In order to use this model in an extended Kalman filter, the model must be linearised as in (2.37a). The required Jacobians can be calculated automatically using a computer algebra system such as Mathematica [163]. The integration of the angular velocities used to obtain $\mathbf{q}_{sw,t+T}$ is derived in Appendix B.3. The respective Jacobian is provided there, too. The linear Gaussian gyroscope measurement model is given as (2.16):

$$\mathbf{y}^{\omega}_{s,t} = \boldsymbol{\omega}_{s,t} + \mathbf{b}^{\omega}_{s,t} + \mathbf{e}^{\omega}_{s,t}. \tag{3.6}$$

Here, $\mathbf{y}^{\omega}_{s,t}$ denotes the angular velocities, and $\mathbf{e}^{\omega}_{s,t}$ denotes time independent Gaussian measurement noise. Let $(\mathbf{m}_{n,t}, \mathbf{m}_{w,t})$ be a 2D/3D correspondence with mutually independent Gaussian measurement noise $\mathbf{e}_{n,t} \sim \mathcal{N}(\mathbf{0}_2, R_{nn,t})$ and $\mathbf{e}_{w,t} \sim \mathcal{N}(\mathbf{0}_3, R_{ww,t})$. By defining the stacked vectors $\mathbf{y}^c_t \coloneqq [\mathbf{m}^T_{n,t}, \mathbf{m}^T_{w,t}]^T$ and $\mathbf{e}^c_t \coloneqq [\mathbf{e}^T_{n,t}, \mathbf{e}^T_{w,t}]^T \sim \mathcal{N}(\mathbf{0}_5, R_t)$ with:

$$R_t = \begin{bmatrix} R_{nn,t} & 0_{2\times 3} \\ 0_{3\times 2} & R_{ww,t} \end{bmatrix}, \tag{3.7}$$

the camera measurement can be modelled in the form of (2.25b):

$$\begin{aligned} \mathbf{0}_2 &= h(\mathbf{x}_t, \mathbf{y}^c_t, \mathbf{e}^c_t) \\ &= \begin{bmatrix} \mathrm{I}_2 & -(\mathbf{m}_{n,t} + \mathbf{e}_{n,t}) \end{bmatrix} R_{cs}(R_{sw,t}(\mathbf{m}_{w,t} + \mathbf{e}_{w,t} - \mathbf{s}_{w,t}) - \mathbf{c}_s). \end{aligned} \tag{3.8}$$

Note that this expression is obtained from the homogeneous collinearity constraint (2.7b) by inserting the hand-eye transformation (2.12). The linearised model required in the EKF measurement update can be obtained from (2.37b) by calculating the respective Jacobians. The camera measurements are processed sequentially, each in an individual EKF measurement update.

3.4.1.2 Gravity Model

In the gravity model, the accelerometers are modelled as inclinometers, *i.e.* they are used to estimate the sensor attitude by assuming that the body acceleration is negligible. The state vector $\mathbf{x}$ comprises additional parameters for the accelerometer biases $\mathbf{b}^a_s$ yielding:

$$\mathbf{x}^T = \begin{bmatrix} \mathbf{s}^T_w & \dot{\mathbf{s}}^T_w & \mathbf{q}^T_{sw} & \boldsymbol{\omega}^T_s & \mathbf{b}^{aT}_s & \mathbf{b}^{\omega T}_s \end{bmatrix}. \tag{3.9}$$

The motion model assumes constant velocity and constant angular velocity, extending (3.5) with:

$$\mathbf{b}^a_{s,t+T} = \mathbf{b}^a_{s,t} + \mathbf{v}^{b^a}_{s,t}, \tag{3.10}$$

where $\mathbf{v}^{b^a}_{s,t}$ denotes time independent Gaussian process noise driving the accelerometer biases. The accelerometer measurement model is obtained from (2.15) by assuming that the body acceleration $\ddot{\mathbf{s}}_{w,t}$ is white zero-mean noise included in the Gaussian process noise $\mathbf{e}^a_{s,t}$. This assumption introduces a correlation between the process noise $\mathbf{v}^{\ddot{s}}_{w,t}$ and $\mathbf{e}^a_{s,t}$, which would actually require a modification of the extended Kalman filter. This correlation is, however, not modelled here. The measurement equation becomes:

$$\mathbf{y}^a_{s,t} = -R_{sw,t}\mathbf{g}_w + \mathbf{b}^a_{s,t} + \mathbf{e}^a_{s,t}. \tag{3.11}$$

As $R_{sw,t} = \mathrm{rot}(\mathbf{q}_{sw,t})$, this model is in the form of (2.26b). The linearised model can be obtained analogously to (2.37b), where the required Jacobian is provided in (B.12) in Appendix B.2. The gyroscope (3.6) and the camera measurement model (3.8) are the same.

3.4.1.3 Acceleration Model

The acceleration model estimates the body acceleration $\ddot{\mathbf{s}}_w$ in the state vector $\mathbf{x}$ yielding:

$$\mathbf{x}^T = \begin{bmatrix}\mathbf{s}_w^T & \dot{\mathbf{s}}_w^T & \ddot{\mathbf{s}}_w^T & \mathbf{q}_{sw}^T & \boldsymbol{\omega}_s^T & \mathbf{b}_s^{aT} & \mathbf{b}_s^{\omega T}\end{bmatrix}. \tag{3.12}$$

The motion model assumes constant acceleration and constant angular velocity changing the upper three equations of the process model (3.5) to:

$$\begin{bmatrix}\mathbf{s}_{w,t+T}\\ \dot{\mathbf{s}}_{w,t+T}\\ \ddot{\mathbf{s}}_{w,t+T}\end{bmatrix} = \begin{bmatrix}\mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} + \frac{T^2}{2}(\ddot{\mathbf{s}}_{w,t} + \mathbf{v}^{\ddot{s}}_{w,t})\\ \dot{\mathbf{s}}_{w,t} + T(\ddot{\mathbf{s}}_{w,t} + \mathbf{v}^{\ddot{s}}_{w,t})\\ \ddot{\mathbf{s}}_{w,t} + \mathbf{v}^{\ddot{s}}_{w,t}\end{bmatrix}. \tag{3.13}$$

Here, the accelerometer measurement model involves also the body acceleration $\ddot{\mathbf{s}}_{w,t}$ resulting in (2.15):

$$\mathbf{y}^a_{s,t} = R_{sw,t}(\ddot{\mathbf{s}}_{w,t} - \mathbf{g}_w) + \mathbf{b}^a_{s,t} + \mathbf{e}^a_{s,t}. \tag{3.14}$$

As discussed in Section 2.2.1.3, this measurement function requires an accurate prediction of the orientation $\mathbf{q}_{sw,t}$, as errors in $\mathbf{q}_{sw,t}$ result in an imaginary body acceleration, that propagates quadratically into the position. The gyroscope (3.6) and the camera measurement model (3.8) are the same.

3.4.1.4 Acceleration Input Model

In order to reduce the dimensionality of the state space, the inertial measurements are considered to be control inputs $\mathbf{u}_t^T = [\mathbf{y}^{\omega T}_{s,t}, \mathbf{y}^{aT}_{s,t}]$ instead of measurements in the acceleration input model. The reduced state vector $\mathbf{x}$ comprises:

$$\mathbf{x}^T = \begin{bmatrix}\mathbf{s}_w^T & \dot{\mathbf{s}}_w^T & \mathbf{q}_{sw}^T & \mathbf{b}_s^{aT} & \mathbf{b}_s^{\omega T}\end{bmatrix}. \tag{3.15}$$

The process model is obtained by inserting the measurement equations (3.6) and (3.14) into the constant acceleration, constant angular velocity model resulting in:

$$\begin{bmatrix} \mathbf{s}_{w,t+T} \\ \dot{\mathbf{s}}_{w,t+T} \\ \mathbf{q}_{sw,t+T} \\ \mathbf{b}^{\omega}_{s,t+T} \\ \mathbf{b}^{a}_{s,t+T} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} + \frac{T^2}{2} R_{ws,t}(\mathbf{y}^{a}_{s,t+T} - \mathbf{b}^{a}_{s,t} - \mathbf{v}^{a}_{s,t}) + \frac{T^2}{2}\mathbf{g}_w \\ \dot{\mathbf{s}}_{w,t} + TR_{ws,t}(\mathbf{y}^{a}_{s,t+T} - \mathbf{b}^{a}_{s,t} - \mathbf{v}^{a}_{s,t}) + T\mathbf{g}_w \\ \exp\big(-\frac{T}{2}(\mathbf{y}^{\omega}_{s,t+T} - \mathbf{b}^{\omega}_{s,t} - \mathbf{v}^{\omega}_{s,t})\big) \odot \mathbf{q}_{sw,t} \\ \mathbf{b}^{\omega}_{s,t} + \mathbf{v}^{b^{\omega}}_{s,t} \\ \mathbf{b}^{a}_{s,t} + \mathbf{v}^{b^{a}}_{s,t} \end{bmatrix}. \tag{3.16}$$

Note that the measurement noises $\mathbf{e}^{a}_{s,t}$ and $\mathbf{e}^{\omega}_{s,t}$ given in (3.6) and (3.14), respectively, are here included in the process noise $\mathbf{v}^{a}_{s,t}$ and $\mathbf{v}^{\omega}_{s,t}$, respectively. The camera measurement model (3.8) is the same. By treating the angular velocities and accelerations as input signals, six states and two measurement update steps can be eliminated. The tuning becomes easier, as fewer noises have to be adjusted. Moreover, these noises can be determined experimentally from the IMU device used by looking at the standard deviations of several seconds of sensor data captured while the device is stationary. There are two further ramifications to mention. As the state space comprises no angular velocities, the orientation can only be predicted, when inertial readings are available. Furthermore, the state space contains no body acceleration, implying that correlations between the orientation and the body acceleration are lost.

3.4.2 Outlier Rejection

In markerless image processing, outliers are likely, especially if a coarse-to-fine approach is used. Since the extended Kalman filter is sensitive to incorrect measurements, it is crucial to detect outliers and to prohibit them from being used in the measurement update. In vision-only tracking the most common way to handle outliers is to check spatial constraints using the RANSAC algorithm (*cf.* Section 2.3.3), or variations of this. This tends to be too restrictive. The visual-inertial sensor fusion algorithm, however, has the advantage of providing temporal information about where the features should occur in the images in order to match the inertial data and the motion model. It turns out that this information is sufficient for detecting outliers, for instance by analysing the innovations. Let $\mathbf{z}^{(i)}_t$ be the innovation computed for the i^{th} 2D/3D correspondence at time t and let $S^{(i)}_t$ be its covariance with $\mathbf{z}^{(i)}_t \sim \mathcal{N}(\mathbf{0}, S^{(i)}_t)$ (*cf.* Algorithm 2.3). The *normalised residual*, $\bar{\mathbf{z}}^{(i)}_t$, is then obtained from the innovation as:

$$\bar{\mathbf{z}}^{(i)}_t = S^{(i)-1/2}_t \mathbf{z}^{(i)}_t \sim \mathcal{N}(\mathbf{0}, \mathrm{I}_2) \tag{3.17}$$

and a standard χ^2 test, with the null hypothesis that the i^{th} correspondence is correct, can be constructed. The measurement update should be dropped if:

$$\mathbf{z}^{(i)T}_t S^{(i)-1}_t \mathbf{z}^{(i)}_t > \chi^2_{\alpha,2}, \tag{3.18}$$

where α is referred to as *significance level* of the test. A common value for α is 0.05 yielding the threshold 5.991. During the experiments, this threshold was found to be too restrictive. Real outliers produce considerably larger residuals. A much less restrictive threshold 15, corresponding to the significant level $\alpha = 10^{-4}$, is therefore used.

3.4.3 Divergence Monitoring

The purpose of divergence monitoring is to detect a failure in the tracking system that requires external reinitialisation (*cf.* Figure 3.2). In vision-only tracking a failure is usually assumed when too few features are registered. The visual-inertial sensor fusion algorithm, however, allows the system to continue tracking for a short period of time without visible features. External reinitialisation is only desired, if integrating the inertial data has introduced a severe drift, which inevitably leads to filter divergence. The process of examining the state of the extended Kalman filter for signs of divergence is denoted by *divergence monitoring*. Here, this process is implemented as a combination of simple tests, where the filter is said to diverge if at least one of the tests fails:

Innovations: Divergence monitoring is related to outlier rejection. Under normal operation, the innovations $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, S_t)$ should resemble zero mean Gaussian white noise. If the distribution changes, this indicates that the filter is diverging. The change is usually detected by lowpass filtering a *distance measure* s_t obtained from the residuals and comparing this to a threshold. Here, the *geometric moving average (GMA)*:

$$g_t = \lambda g_{t-T} + (1-\lambda)s_t \quad \text{with} \quad 0 \ll \lambda < 1 \tag{3.19}$$

is used to detect a change in the mean, *i.e.*:

$$s_t = \frac{1}{\sqrt{n_z}} \mathbf{1}_{n_z}^T \bar{\mathbf{z}}_t, \tag{3.20}$$

where the $n \times 1$ vector $\bar{\mathbf{z}}_t$ denotes the normalised residual (*cf.* Section 3.4.2) and $\mathbf{1}_n$ is a vector with n unit elements. Note that, in combination with the outlier rejection, this term is bounded by the outlier threshold. Alternative lowpass filters and distance measures can be found, for instance, in [103], where the topic of change detection is thoroughly treated.

State uncertainty: The above test is unlikely to detect a divergence when no features are visible. Since the inertial measurements provide no absolute reference, the innovations are less meaningful. In particular, the acceleration input model treats the inertial data as control inputs. However, the lack of camera measurements results in an increasing state uncertainty. Hence, the norm of the state covariance is computed and compared to a threshold. Here, the Frobenius norm [164] is used and a threshold between 3 and 5 is chosen during the experiments.

Quaternion norm: The EKF measurement update step does not preserve the norm of the orientation quaternion. A slight change from unit length is corrected for by normalising the quaternion. A significant change, however, indicates that the filter has diverged. The threshold 0.05 is chosen during the experiments. By normalising the quaternion, the filter ceases to propagate the conditional mean. In order to adapt the state covariance

appropriately, a method proposed in [165] is used. Let $\mathbf{q}^*_{sw,t}$ and $P^*_{qq,t}$ be the minimum squared error estimate of the quaternion at time t. The normalisation step:

$$\mathbf{q}_{sw,t} = \frac{\mathbf{q}^*_{sw,t}}{\|\mathbf{q}^*_{sw,t}\|} \tag{3.21a}$$

$$P_{qq,t} = P^*_{qq,t} + (\mathbf{q}_{sw,t} - \mathbf{q}^*_{sw,t})(\mathbf{q}_{sw,t} - \mathbf{q}^*_{sw,t})^T, \tag{3.21b}$$

where $\mathbf{q}_{sw,t}$ and $P_{qq,t}$ constitute the new constrained estimate used in the subsequent time update, is performed after each measurement update.

3.5 Offline Preparation

This section describes the offline preparation required to use the visual-inertial tracking system in a special target environment. The calibration of the CamIMU hardware and related issues are treated in Section 3.5.1. The generation of the input data required by the software system, in particular, by the vision subsystem, is described in Section 3.5.2.

3.5.1 CamIMU

The CamIMU and its properties are presented in Section 2.2.2. In order to use this device with the tracking framework developed here, the intrinsic parameters of the camera as well as the hand-eye transformation between the camera and the IMU must be calibrated and the calibration must be repeated every time the objective is changed or adjusted. In this thesis, an extended version of the freely available camera calibration toolbox for Matlab [18] is used to perform the calibration. The toolbox implements the calibration procedure outlined in Section 2.1.2 and has been extended to estimate also the hand-eye transformation with the algorithm described in Section 2.2.2.1. The method requires inertial data to be recorded for each calibration image while holding the CamIMU statically. More precise results are obtained by averaging several seconds of data. Furthermore, the measured standard deviations provide suitable settings for the measurement noises in the extended Kalman filter.

3.5.2 Input Data

The markerless image processing algorithm and the initialisation process require input data. The former requires a textured CAD model of the target object or environment [2]. A manually created CAD model of a simple test object is shown in Figure 3.3. Textured CAD models of more complex environments can be created semi-automatically with a commercial 3D modelling tool such as REALVIZ ImageModeler [167]. They can also be generated automatically

[2]The model is assumed to be given in the standard *Virtual Reality Modeling Language (VRML)* file format [166]

from an image sequence using computer vision techniques (*cf.* Section 2.3). Textured models have been obtained from perspective camera images [168] and from fisheye images [169]. Note that, in order to be used with the visual-inertial tracking framework, the 3D model needs special properties: it has to be aligned with gravity (*cf.* Section 2.2.2.2) and given in absolute units (*cf.* Section 2.2.2.3). A method to automatically generate textured CAD models that meet the special requirements of the tracking method presented here has been developed and implemented in the MATRIS project.

The input data required by the initialisation process depends on the initialisation mode used (*cf.* Section 3.2). The manual initialisation requires the definition of a suitable initial orientation and position. The semi-automatic method requires calibration of the relative rotation between the global coordinate system and the coordinate system associated with the CAD model. This transformation can be computed at run-time using the equations given in Section 2.2.2.4, once the system has been initialised manually. The fully automatic method requires a database of reference images. These can be calibrated manually given the CAD model [5], or they can be exported during the automatic model generation step [156, 169].

3.6 Experimental Setup and Results

This section provides an experimental performance evaluation of the visual-inertial tracking system developed in this chapter. The focus is on the comparison of the four different sensor fusion models developed in Section 3.4.1, *i.e.* the effects of using accelerometers on the tracking performance. The system initialisation is not the subject of evaluation in this thesis. Three different target environments are considered in the experiments: a small-scale scenario, *desktop*, a mid-scale scenario, *room*, and a large-scale scenario, *foyer*. In each of the three scenarios, the developed system is able to estimate the trajectory of the CamIMU online and stably at 25 Hz using images with resolution 320×240 pixels. In particular, *virtual augmentations*, *i.e.* virtual graphics or characters that are drawn on top of the camera images in real-time, show no visible jitter or drift, even during fast movements and when the CamIMU is very close. An example is shown in Figure 3.14(b) in the sequel. Moreover, the analysis-by-synthesis based image processing algorithm revealed no problems with textures from cameras other than the one used for the tracking.

For an in-depth evaluation representative data sequences (synchronised images and IMU data) have been captured from each scenario. The CamIMU was controlled by an industrial robot in the desktop environment and by hand in the room and foyer test case. The results are presented in the sequel. The tracking performance and computational efficiency of the four different state-space models denoted gyro, gravity, acceleration and acceleration input are first investigated in the desktop scenario, where the reference trajectory of the robot is available, see Section 3.6.1. The evaluation shows that the acceleration input model performs best with respect to both criteria. Sections 3.6.2 and 3.6.3 present a qualitative evaluation in the room and foyer scenario, respectively, demonstrating the high stability and accuracy achieved, also under realistic conditions, with the acceleration models.

Table 3.1: *Desktop: standard deviation noises assuming equal noise in all dimensions. Note that the process and measurement noise settings related to the accelerations differ significantly between the models and data sequences. In order to assure meaningful evaluation results, appropriate settings have been derived for each model from the model assumptions and the reference trajectory of the robot.*

	gyro	**gravity**	**acceleration**	**acceleration input**
fast sequence				
$\mathbf{v}_{w,t}^{\ddot{s}}$	4.	4.	0.1	–
$\mathbf{e}_{s,t}^{a}$	–	0.84	0.14	–
slow sequence				
$\mathbf{v}_{w,t}^{\ddot{s}}$	0.19	0.19	0.1	–
$\mathbf{e}_{s,t}^{a}$	–	0.28	0.14	–
both				
$\mathbf{v}_{s,t}^{a}$	–	–	–	0.14
$\mathbf{v}_{s,t}^{\omega}$	0.1	0.1	0.1	0.01
$\mathbf{v}_{s,t}^{b^{\omega}}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$
$\mathbf{v}_{s,t}^{b^{a}}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-5}$
$\mathbf{e}_{s,t}^{\omega}$	0.01	0.01	0.01	–
$\mathbf{e}_{n,t}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$

Table 3.2: *Room, foyer: standard deviation noises assuming equal noise in all dimensions. The acceleration model is used in these test cases.*

	room	**foyer**
$\mathbf{v}_{w,t}^{\ddot{s}}$	0.1	1.
$\mathbf{v}_{s,t}^{\omega}$	0.1	0.1
$\mathbf{v}_{s,t}^{b^{\omega}}$	$5.5 \cdot 10^{-4}$	$5.5 \cdot 10^{-4}$
$\mathbf{e}_{s,t}^{a}$	0.04	0.18
$\mathbf{e}_{s,t}^{\omega}$	0.023	0.06
$\mathbf{e}_{n,t}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$

The noise settings used during the experiments are outlined in Table 3.1 for the desktop test case and in Table 3.2 for the test cases room and foyer. The noises are given as standard deviations assuming equal noise in all dimensions. The noise affecting the feature registration has been determined experimentally by looking at the change in the 2D pixel locations while the camera was stationary. The 3D model coordinates, which are obtained from the target scene model, are assumed certain.

(a) *(b)*

Figure 3.6: *Desktop: two frames of the image sequences, where the CamIMU is nearest to (a) and farthest from (b) the target object.*

Table 3.3: *Desktop: root mean square prediction errors in pixels for the different state-space models.*

	gyro		**gravity**		**acceleration**		**acceleration input**	
	mean	std	mean	std	mean	std	mean	std
fast sequence	3.82	4.04	3.83	4.05	0.91	0.59	0.77	0.45
slow sequence	0.51	0.47	0.50	0.47	0.59	0.37	0.69	0.40

3.6.1 Test Case: Desktop

This small-scale scenario denotes the simple two-sided test object shown previously in Figures 3.3 and 3.4. The CAD model has been created by hand. In order to obtain reference data, the CamIMU was mounted on a robotic arm while observing the target object. The robotic arm performed a continuous movement in the shape of an eight at various speeds. Figure 3.6 shows camera images from the outer points of this movement. Two data sets are considered for the tests, a slow sequence and a fast sequence. The estimated accelerations and angular velocities are shown in Figure 3.7.

As the fast sequence contains significant body accelerations, which violate the assumptions of the gravity model (*cf.* Section 3.4.1.2) severely, a simple detection criterion is applied. Basically, the accelerometer measurement is rejected at time t, if

$$\left| \|\mathbf{y}^a_{s,t} - \mathbf{b}^a_{s,t}\| - \|\mathbf{g}_w\| \right| > D^a, \tag{3.22}$$

where $\mathbf{y}^a_{s,t}$ denotes the measured acceleration, $\mathbf{b}^a_{s,t}$ is the bias estimate, $\mathbf{g}_w$ is the gravity vector in the world frame and D^a is a user-defined threshold, chosen as $D^a = 0.4$ during the experiments. Further body acceleration detection criteria can be found, for instance, in [170, 171].

The prediction errors of the features in the images, *i.e.* the Euclidean distances between the predicted feature positions and the registered positions, are used as indicators of the tracking

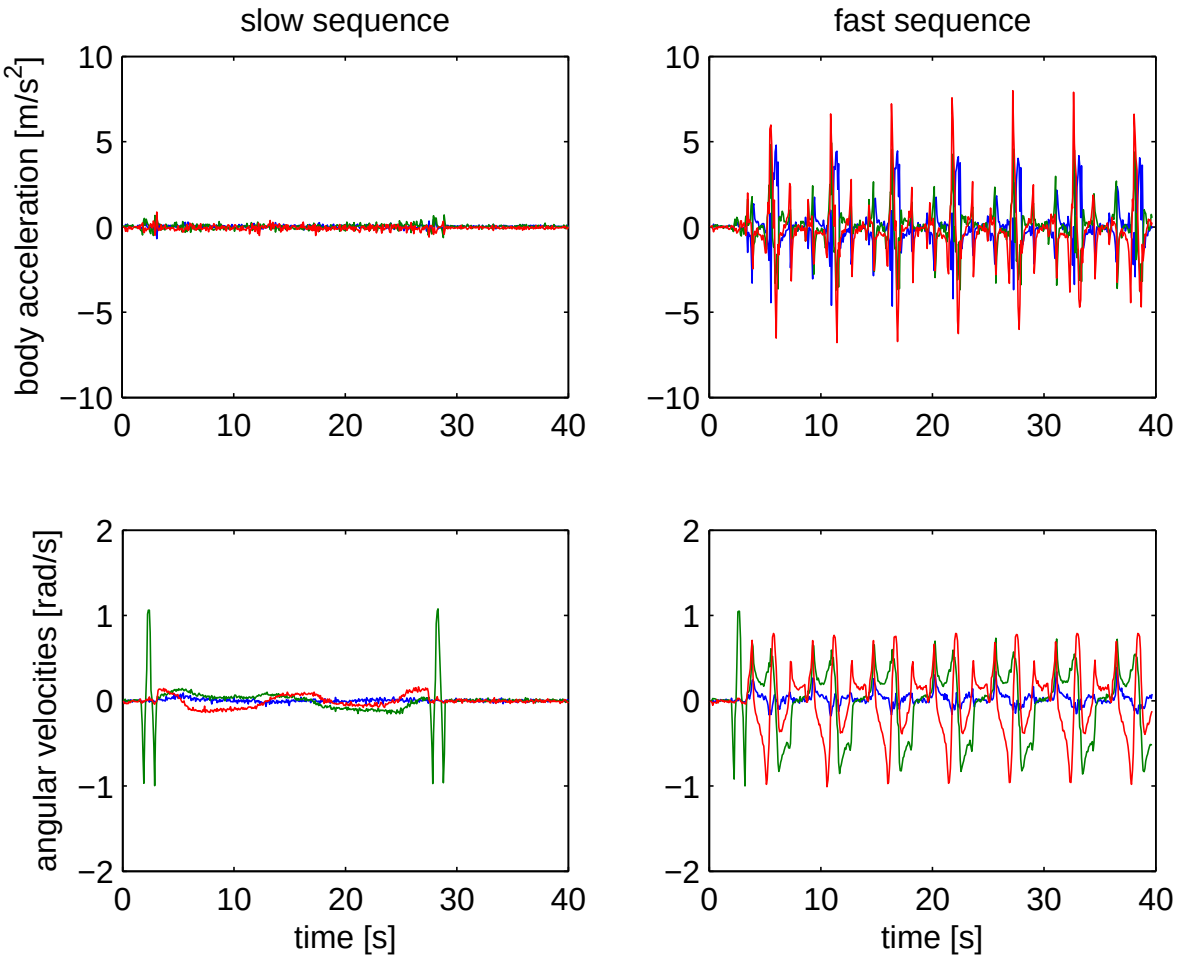


Figure 3.7: *Desktop: body accelerations and angular velocities of the slow and the fast sequence as estimated by the system running in the acceleration mode.*

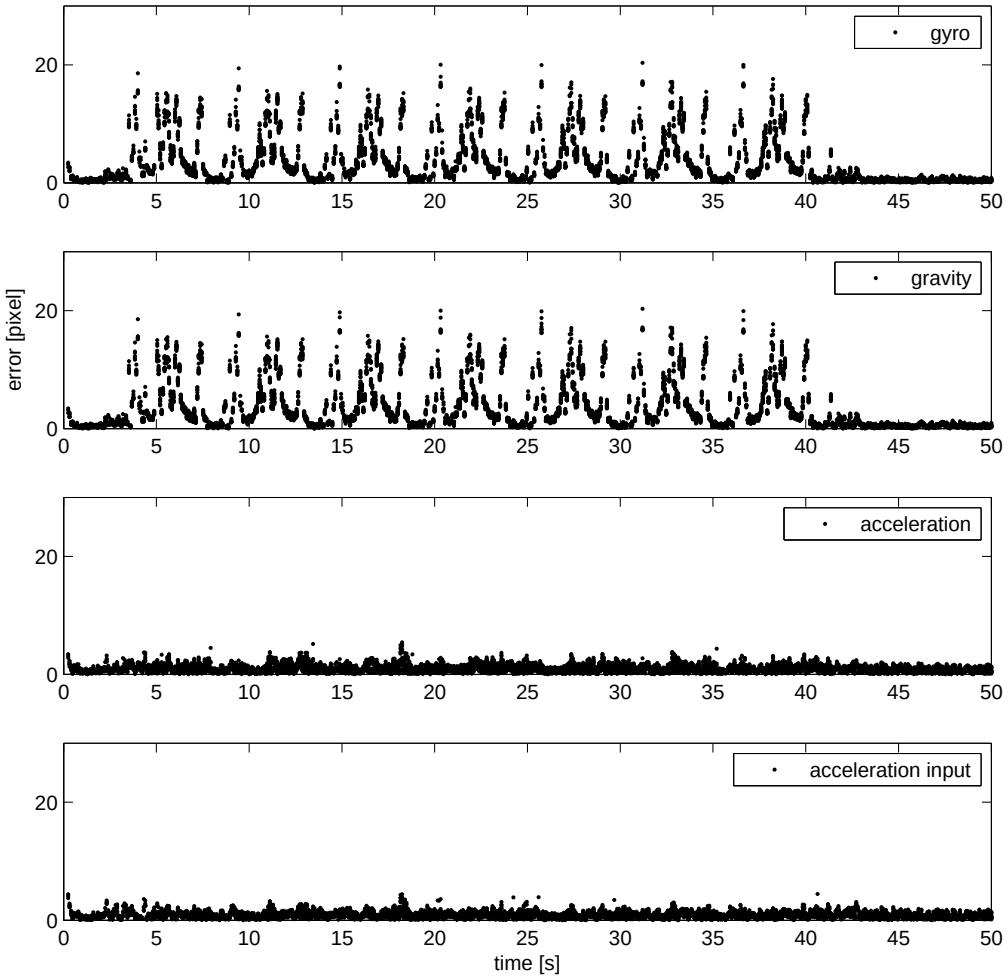
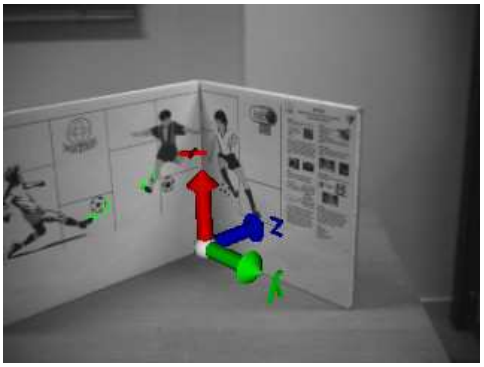


Figure 3.8: *Desktop, fast sequence: the prediction errors of the registered features show clearly the superiority of the acceleration and the acceleration input model. The outliers are already removed.*

(a) *gyro model* ***(b)*** *gravity model*



(c) *acceleration model*

Figure 3.9: *Desktop, slow sequence: after* $22.32\,s$ *of tracking with only two features, marked by green crosses, the gyro model (a) has drifted significantly, while the gravity and acceleration model (b,c) show comparably stable results. This can be seen from the pose of the augmented coordinate system.*

performance. Table 3.3 shows clearly how the acceleration and the acceleration input model outperform the other models on the fast sequence producing a significantly smaller root mean square prediction error. This also becomes apparent in Figure 3.8, where the prediction errors produced by these models remain constant, while the gyro and the gravity model produced errors that are highly correlated with the acceleration peaks in Figure 3.7. On the slow sequence, all models perform comparably well, as expected.

Table 3.3 also shows that the gravity model, which uses the accelerometers as inclinometers to estimate the attitude of the CamIMU, gives no improvement compared to the gyro model on any of the two sequences. This is explained by the fact that a relatively high feature density is used for the experiments, providing sufficient information about the orientation. Figure 3.9 presents results on the slow sequence, where only two camera measurements per image are delivered by the vision system, thus revealing the benefits of using the accelerometers in the gravity as well as in the acceleration model.

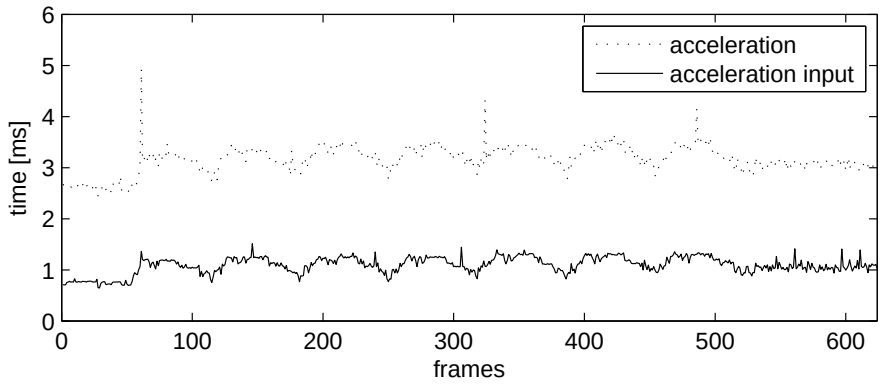


Figure 3.10: *Desktop, fast sequence: per frame processing time needed for the EKF time and measurement updates by the acceleration and the acceleration input model. This has been measured on a* 2.2 *GHz laptop. Note that the processing time varies due to a varying number of vision measurements per frame. The three peaks are explained by frame misses, where the IMU buffer contained twice as many IMU readings (*cf. *Figure 3.2).*

Finally, Table 3.3 shows that the acceleration and the acceleration input model yield comparable results, indicating that the missing correlations between the orientation and the body acceleration in the acceleration input model do not impact the tracking performance. The acceleration input model yields as stable and precise estimation results as the acceleration model, but with significantly reduced computational costs. Figure 3.10 shows the processing time needed by both models for the EKF time and measurement updates. In this specific experimental setup, using the acceleration input model reduces the computational costs by a factor of 2.7.

The experimental results presented so far allow for the following conclusions. The gravity model is superior to the gyro model by reducing the number of features needed to correct the drift in the gyroscopes, but both models show — as expected — a poor tracking quality in the presence of body accelerations. The acceleration and the acceleration input model also reduce the number of features needed and provide significantly higher prediction quality. Under slow movements, where the constant acceleration model is actually overparameterised, they still perform just as well without introducing instabilities. Moreover, treating the inertial data as control inputs significantly reduces the computational burden, while keeping the tracking quality.

To give an idea of the absolute accuracy of the overall system under quick motions, the input trajectory of the robot has been transformed into the spatial and time reference of the fast sequence and compared to the camera trajectory produced by the acceleration and the acceleration input model. The average Euclidean position error obtained in this comparison is 1 cm and the average error in the Euler angles $0.76°$. Note that the transformations used to synchronise the trajectories contribute to the estimation errors, so that the actual accuracy can be assumed higher.

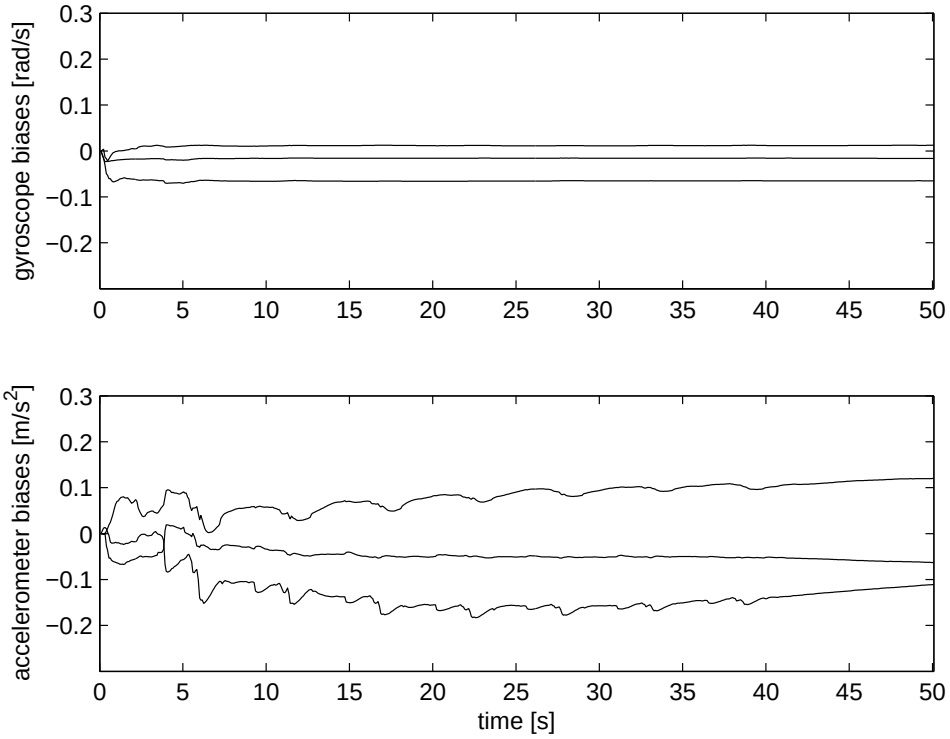


Figure 3.11: *Desktop, fast sequence: gyroscope and accelerometer biases as estimated by the acceleration model.*

Table 3.4: *Desktop: root mean square prediction errors in pixels for the acceleration model with and without accelerometer bias estimation.*

	w/ bias estimation		**w/o bias estimation**	
	mean	std	mean	std
fast sequence	0.908	0.586	0.913	0.589
slow sequence	0.586	0.373	0.725	0.412

Finally, the online bias estimation has been investigated on the fast sequence using the acceleration model. The result is shown in Figure 3.11. The gyroscope bias estimates — initialised to zero — converge quickly and stay constant over the short duration of the experiment, while the accelerometer bias estimates change with the orientation. This indicates that the gyroscope biases are estimated reliably, while other errors such as calibration or model errors tend to show up incorrectly in the accelerometer bias parameters. As a matter of fact, the online accelerometer bias estimation turns out to make the tracking instable without significantly improving the overall tracking performance when corrective camera measurements are available. Table 3.4 presents the root mean square prediction errors for the features with and without online accelerometer bias estimation. It shows that the performance gain is minor. Moreover, the system turns out to be very sensitive to the accelerometer bias process noise. Furthermore, the gyroscope biases are the predominant sources of error, not only for the rotational, but also for the translational states for the acceleration models. Therefore only the gyroscope biases are estimated online during the subsequent experiments. This reduces the state vector by three states and hence saves further computational power.

3.6.2 Test Case: Room

This mid-scale scenario denotes a room of base area $5 \times 4\,\mathrm{m}$, where the CAD model, created with REALVIZ ImageModeler, consists of three walls. The fourth wall contains mostly windows and is therefore not included in the model. Equally, the floor and the ceiling are ignored, as they provide no discriminative features. The textured CAD model and several pictures of the room are shown in Figure 3.12. A high-quality digital camera has been used to obtain the textures for the model.

In this scenario, the CamIMU is handheld and allowed to move without constraints in the room. The captured data sequences feature a significant range of rotations and translations as demonstrated in Figure 3.13. This is all handled robustly by the system running in the acceleration and the acceleration input mode. Two 360° rotations are performed whereas one wall is not included in the CAD model. Tracking is continued by integrating inertial data during periods without visible features, as indicated in Figure 3.14(c). The vision system is able to recover the features, once, they occur in the field of view again. When the user moves the camera very close to the target scene seeing only few features, the augmentation still remains in its place (Figure 3.14(b)). Finally, some fast movements with free accelerations of up to

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

Figure 3.12: *Room: the textured CAD model and real pictures.*

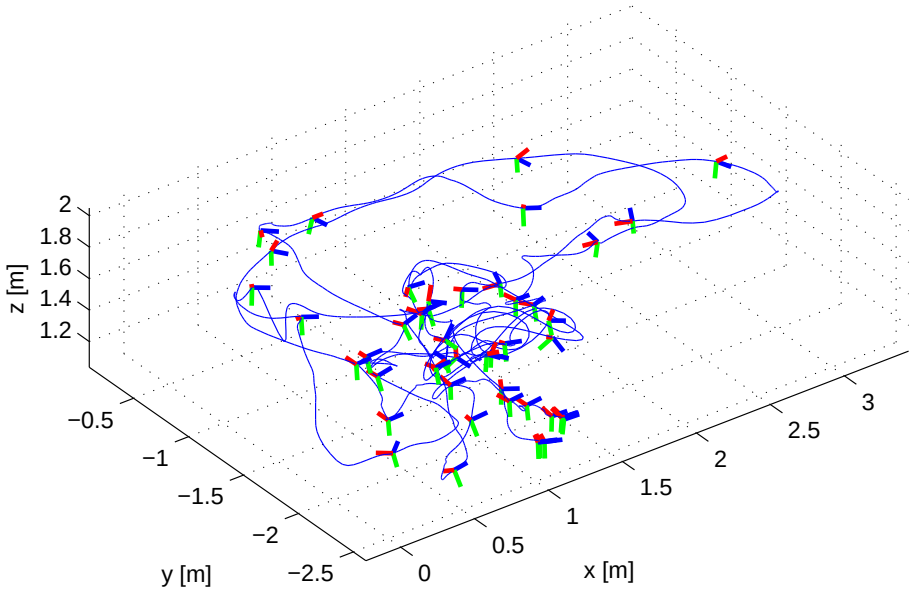


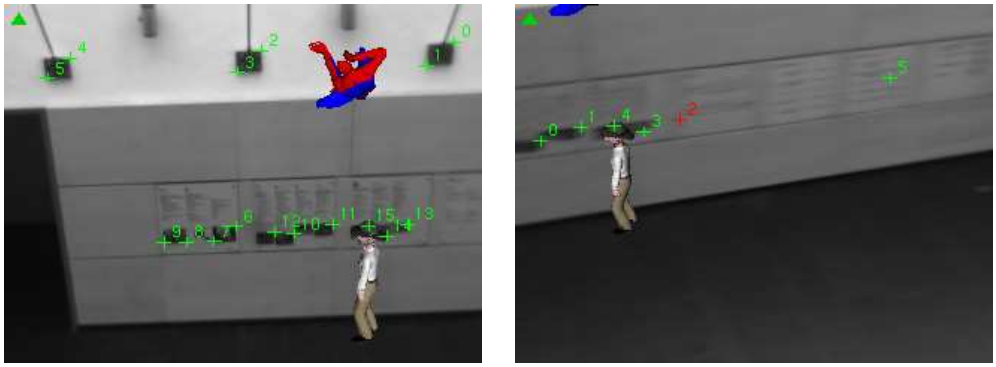
Figure 3.13: *Room: camera trajectory in world coordinates as estimated by the proposed system. The coordinate systems denote the camera orientation for every* 50^{th} *frame.*

(a) *(b)* *(c)* *(d)* *(e)* *(f)*

Figure 3.14: *Room: some augmented frames demonstrating the range of possible movements (a-c) and the stable operation of the system under occlusion (d) and varying light conditions (d,f). Note that the wall observed in (c) is not part of the model. However, the augmentation in the lower left corner stays in its place. In (d) the green crosses indicate registered features and the red ones mark features where the registration failed due to the occlusion.*

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

Figure 3.15: *Foyer: the textured CAD model and real pictures from different viewpoints.*



(a) *(b)*

Figure 3.16: *Foyer: camera frames overlaid with feature positions and virtual objects. Note the significant motion blur in (b), where features were nevertheless registered.*

$12.9\,\mathrm{m/s^2}$ and rotations up to $9.6\,\mathrm{rad/s}$ are tracked successfully. Figure 3.14(d-f) illustrated the stable operation of the system under occlusion and varying light conditions.

3.6.3 Test Case: Foyer

The CAD model of this large-scale scenario contains one wall (4.5×8.5 m) — roughly modelled as one plane — of a large open foyer inside a building. The textured model and two pictures from different viewpoints, that have been used to create the model with REALVIZ ImageModeler, are illustrated in Figure 3.15. During the experiments, the scene is viewed from the first floor (*cf.* Figure 3.15(b)), where the wall included in the model is at a distance of approximately $7\,\mathrm{m}$. The tracking system is initialised by hand, as a ferrous handrail at this floor causes magnetic disturbances. Figure 3.16 shows two augmented live camera images. Note that the features on the posters look quite similar due to the big viewing distance. An accurate pose prediction is therefore crucial to maintain the tracking.

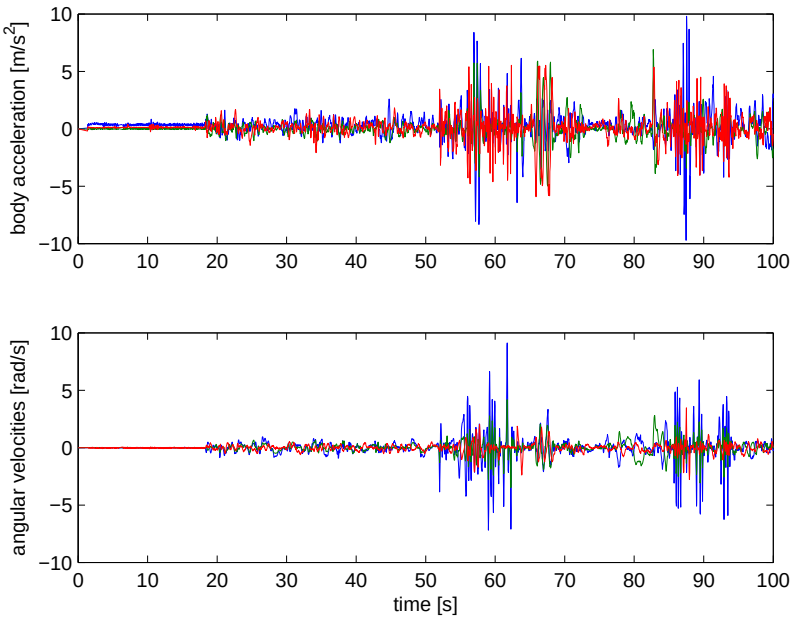


Figure 3.17: *Foyer: body accelerations and angular velocities as estimated by the system running in the acceleration mode.*

This sequence can not be processed using the gravity or the gyro model, although the camera movements are indeed mainly rotational. The reason for this is that the rotations are off-centre and hence produce high linear accelerations. Figure 3.17 shows the significant body accelerations and angular velocities, estimated with the acceleration model, which tracks robustly throughout the whole sequence.

Figure 3.18 presents the prediction errors for the features with the detected outliers marked by stars. The root mean square prediction error over the whole image sequence is 1.42 pixels with 1.47 pixels standard deviation. Some increased displacements between time 60 and 70 are explained by camera measurements repeatedly missing for up to half a second. Without visible features the pose is based solely on integrated IMU data.

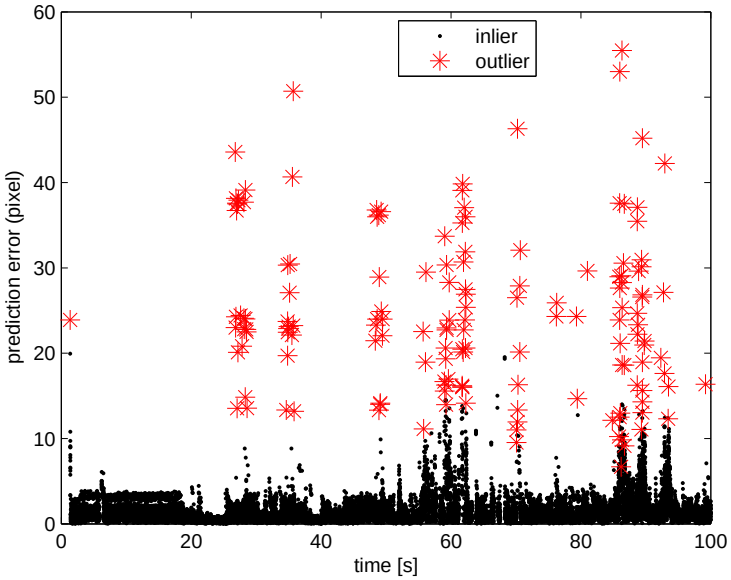


Figure 3.18: *Foyer: prediction errors for the features produced by the acceleration model.*

3.7 Conclusion

In this chapter, a model-based visual-inertial tracking framework is described that requires no markers or other special infrastructure in the environment. It uses an affine illumination invariant image processing method, which exploits the pose prediction obtained from the sensor fusion algorithm and a textured CAD model of the environment to predict the appearances of corner features in the camera images. Moreover, different sensor fusion models, all allowing for a tight integration of the camera and the inertial sensors, have been evaluated experimentally. The open questions posed in the beginning of Section 3.4 are solved for the described tracking framework. The experimental evaluation clearly shows the benefits of the acceleration models, *i.e.* a better tracking quality and reduced demand for camera measurements. By combining the top down markerless image processing algorithm with the fusion strategy of the acceleration input model, a very efficient and robust final system is obtained, which is shown to work robustly also in realistic mid- and large-scale environments, under varying light conditions, fast camera movements and even short periods without camera measurements. To summarise, the following improvements are achieved compared to vision-only tracking:

- increased robustness against rapid camera motions.
- reduced computational costs, both due to an accurate prediction of the feature appearances and locations in the images and a reduced demand on camera measurements in general.
- stable pose estimation with only few visible features.
- compensation for short-term loss of camera measurements, for instance, due to occlusion.
- reduced jitter and drift.

It is important to note that, due to the image processing approach used, the tracking system presented here is suitable for environments containing richly textured planar regions and walls. However, the analysis-by-synthesis technique has also been applied successfully to poorly textured environments by exploiting prominent contours, see for instance [41, 42, 155]. The developed system allows for a smooth exchange of the image processing algorithm, thus making it easy to extend its applicability to such scenarios.

Model-based approaches provide stable tracking and are suitable for a wide range of applications. However, the model generation and maintenance can be time consuming and the assumption of a static environment is often violated. Subsequently, Chapters 4 and 5 are devoted to simultaneous localisation and mapping using computer vision techniques (*cf.* Section 2.3) and recursive filtering techniques (*cf.* Section 2.4).

4

Visual SLAM

Many of the recent real-time markerless camera tracking systems rely on the existence of a complete 3D model of the target environment. Section 2.3.5 gives a brief overview. Also the visual-inertial tracking system developed in Chapter 3 assumes that a scene model is available as a textured CAD model. Model-based approaches are suitable for a wide range of applications and state of the art techniques allow for stable, accurate and drift-free camera registration, given the absolute reference of the model. However, sometimes it is not feasible to model the entire target scene in advance, in particular, if parts are not static, or, if it is desirable to use an existing CAD model, which is not complete. In order to allow camera movements beyond the parts of the environment modelled in advance, it is required to derive additional 3D structure online, *i.e.* to perform localisation and mapping simultaneously. The development of such a system in a vision-only setup is the topic of this chapter. The additional exploitation of inertial data is investigated in the subsequent Chapter 5.

Two prominent research topics are related to the problem of estimating the camera trajectory and scene structure from an image sequence: structure from motion (SFM), originally developed in computer vision, and simultaneous localisation and mapping (SLAM), originally developed in robotics. How they pose the estimation problem and the strategy to solve it is described in Section 2.3 and 2.4, respectively. Structure from motion is often associated with words such as robustness, accuracy and bundle adjustment and with the aim to solve the full localisation problem. SLAM is instead associated with recursive filtering, uncertainty measures and with the aim to solve the online localisation problem. Moreover, in the area of SLAM, effort has been used to develop algorithms that scale with the size of the map. The problems of FastSLAM and MPF-SLAM, as two representatives, are expounded in Section 2.4.3.3 and 2.4.3.4, respectively. The aim of this chapter is to unify the ideas of both lines of work — robust estimation and recursive filtering — in an algorithm, that is capable of solving the considered estimation problem with minimal drift in real-time.

Section 4.1 outlines the approach and the workflow of the method developed in this chapter.

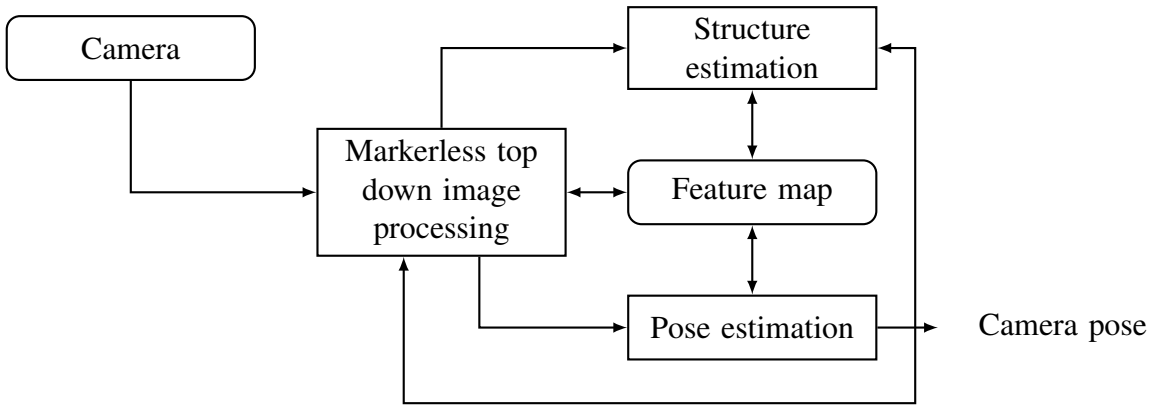


Figure 4.1: *Architecture of the visual SLAM system.*

Section 4.2 proceeds with the reconstruction of 3D geometry — 3D points in space and camera poses — from camera measurements. Section 4.3 develops a measure of feature quality. Results demonstrating the performance of the system on real and synthetic image sequences are presented in Section 4.4 and final conclusions are drawn in Section 4.5.

4.1 Approach

The basic structure of the tracking system developed here fits into the sequential structure from motion framework. Figure 4.1 illustrates the major building blocks and how they are related. The markerless image processing component generates point correspondences between the image obtained from the camera and the features in the map. These are used to obtain the camera pose, which is then used to estimate the feature locations. In both steps standard computer vision techniques are applied (*cf.* Section 2.3). When implemented in a naive way, this loop, the regular sequential structure from motion loop, will inevitably lead to significant drift in the camera registration. In order to avoid this and to obtain a robust real-time system with minimal drift, the standard algorithm is enhanced in several respects:

Robust estimation: The 3D geometry estimation processes, *i.e.* the reconstruction of the camera pose and the 3D locations of the features, are based on robust algorithms in each step.

Uncertainty propagation: As in the extended Kalman filter, geometry estimation is performed in a stochastic framework, where all parameters and measurements are modelled as Gaussian random variables.

Recursive feature refinement: As in FastSLAM, an extended Kalman filter is associated to

each feature for refining its 3D location recursively, once it has been initialised by triangulation.

Feature monitoring: The quality and consistency of each feature are monitored in order to select appropriate *3D features*, *i.e.* features that are already initialised in 3D, for pose estimation.

Epipolar constraint: Both the collinearity constraint (*cf.* Section 2.1.3.1) and the epipolar constraint (*cf.* Section 2.1.3.2) are used for estimating the camera pose. This allows for incorporating features from the time of their initial detection, irrespectively of their 3D state.

The resulting algorithm is effectively a combination of sequential structure from motion and FastSLAM with only one particle. The stochastic model is simplified, as the camera and the features are assumed mutually independent. However, experiments show a significant reduction of the drift in the camera registration.

The workflow of the software system is specified in Figure 4.2. It is assumed that sufficient 3D information about the target environment is available to initialise the predictive tracking loop. This knowledge can be represented as, for instance, a CAD model of an object in the scene, a set of 3D features or a marker placed in the environment. The reference frame of the prior knowledge defines the world coordinate system.

The first step in the tracking loop is to register visible map features in the camera image, using a markerless top down image processing algorithm. The method is similar to the one developed in Section 3.3. However, the corner features are tracked across subsequent camera images rather than from a synthetic image to the camera image. The iterative patch registration therefore includes — besides the affine photometric model — an affine motion model and further enhancements to reduce the drift in the feature localisations and to allow for the reliable detection of outliers. Hence, it provides accurate camera measurements with few outliers. The details can be found in [7]. The previous pose is used to predict the visibility of the features in the current frame. Note that computer vision techniques are in general not recursive. Camera poses are typically estimated for each frame independently, using the available camera measurements. As camera motion is not explicitly modelled, the previous pose must serve as prediction.

After the registration step, suitable sets of 2D/3D and 2D/2D correspondences are established. The pose estimation is divided into two stages. An initial pose is computed from the 2D/3D correspondences using the RANSAC algorithm (*cf.* Section 2.3.3) in order to remove outliers. The resulting inliers are then used in a refinement stage, where epipolar constraints given by the 2D/2D correspondences and the previous camera pose are incorporated. The estimated camera pose is the used to either triangulate or refine the 3D locations of the registered features. RANSAC is used to perform robust triangulation, the result of which serves as prior estimate for the extended Kalman filter. On demand, new features are extracted in free areas of the image and added to the feature map.

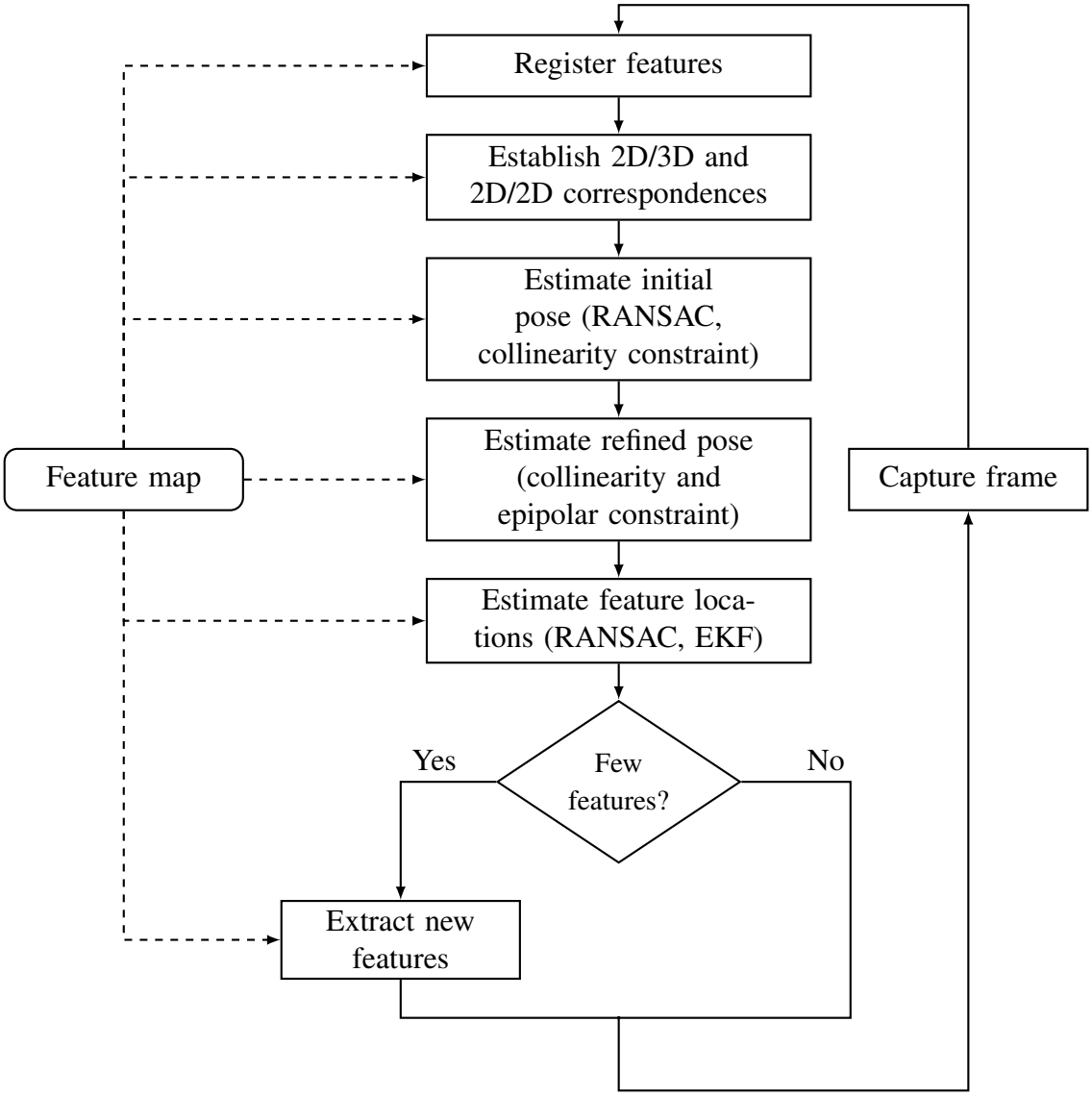


Figure 4.2: *Workflow of the visual SLAM system: predictive tracking loop.*

The management of the feature map is here solved similarly to the visual-inertial tracking system presented in Chapter 3 (*cf.* Section 3.3.2). However, as the 3D feature locations are estimated online and features are tracked across subsequent camera images, the structure of the feature map is more complex. Both the image registration and the reconstruction, require information associated with each feature. The details can be found in [6, 7]. Moreover, the selection of appropriate features for pose estimation depends not only on spatial considerations but also on the quality of the feature with respect to its localisation in 3D. A feature quality measure is therefore developed in Section 4.3.

The rest of this chapter focuses on the enhanced 3D estimation processes (*cf.* Section 4.2) and their experimental evaluation (*cf.* Section 4.4). In particular, the value of the basic stochastic model is investigated.

4.2 3D Geometry Estimation

Geometry estimation in 3D includes the initial triangulation of the 3D feature locations treated in Section 4.2.1, the recursive refinement in the extended Kalman filter described in Section 4.2.2 and the two pose estimation stages discussed in Section 4.2.3.

All building blocks required to perform these estimations are given in Chapter 2. The homogeneous collinearity constraint (2.7b) is used for the linear triangulation (*cf.* Section 2.3.2) of the feature locations and as measurement equation in the extended Kalman filter. The observations are here camera views. The image collinearity constraint (2.7a) serves two purposes. It is used in the cost function for the nonlinear least squares (LS) and weighted least squares (WLS) optimisation of the camera pose (*cf.* Section 2.3.1). Here, the observations are point correspondences. Moreover, it is used for determining the consistent observations, *i.e.* the inliers, during the RANSAC (*cf.* Section 2.3.3) iterations of the first pose computation stage and the initial triangulation. The reasons for this choice are given in Section 2.1.3.1. The symmetric epipolar distance (2.20) is used in the cost function of the second pose optimisation step, for incorporating the observed 2D/2D correspondences. The camera pose is represented by the six-dimensional vector $\mathbf{s}^r = [\mathbf{r}_{cw}^T, \mathbf{w}_c^T]^T$ (*cf.* Section 2.1.1.4). The minimal axis-angle parametrisation $\mathbf{r}_{cw}^T$ is chosen since it is suitable for nonlinear optimisation.

The three entities involved in 3D geometry estimation are 2D feature locations, given as normalised image points $\mathbf{m}_n$, 3D feature locations, given as world points $\mathbf{m}_w$, and camera poses $\mathbf{s}^r$. Depending on the type of estimation performed, the role of measurement and unknown changes. Camera poses are obtained from point correspondences and 3D points are obtained from camera views. All of the three entities are modelled as mutually independent Gaussian random vectors:

$$\mathbf{m}_n \sim \mathcal{N}(\hat{\mathbf{m}}_n, R_{nn}) \quad \mathbf{m}_w \sim \mathcal{N}(\hat{\mathbf{m}}_w, P_{ww}) \quad \mathbf{s}^r \sim \mathcal{N}(\hat{\mathbf{s}}^r, P_{\mathbf{s}^r\mathbf{s}^r}). \tag{4.1}$$

The noise in the 2D feature locations is a tuning parameter. In the experiments, equally distributed noise with 1 pixel standard deviation in both dimensions is assumed over the whole

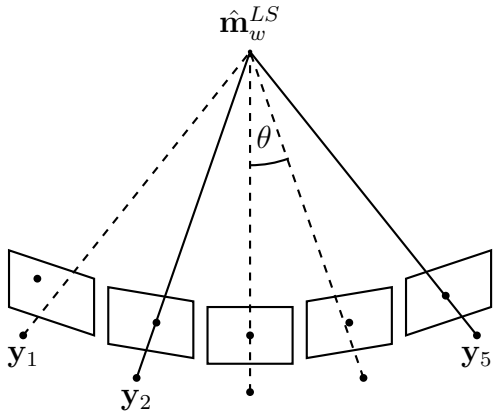


Figure 4.3: *Robust triangulation: a buffer of camera views, where subsequent views are separated by a sufficiently large angle θ, is used for robust triangulation. Here, the least squares estimate $\hat{\mathbf{m}}_w^{LS}$ is obtained from the camera views $\mathbf{y}_2$ and $\mathbf{y}_5$, respectively, and is consistent with the views $\mathbf{y}_3$ through $\mathbf{y}_4$. Measurement $\mathbf{y}_1$ is an outlier.*

camera image, *i.e.* $R_{pp} = \mathrm{I}_2$, where R_{pp} denotes the covariance in the pixel coordinate system. The transformation to the normalised image frame is $R_{nn} = K^{-1}R_{pp}K^{-T}$, where K denotes the matrix of the intrinsic camera parameters, K^{-1} is given in (2.3b) and K^{-T} denotes the transposed inverse of K (*cf.* (4.2) in the sequel). If the camera images exhibit severe distortion, *e.g.* if a wide-angle cameras is used, it is sensible to assume increasing noise towards the image borders. A heuristic method is proposed in [172]. As in the extended Kalman filter, a first order Gauss approximation is used to approximate nonlinear functions of Gaussians. Let $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, P_{xx})$ be a Gaussian random vector. The nonlinear function $\mathbf{y} = f(\mathbf{x})$ is then approximately distributed according to $\mathbf{y} \sim \mathcal{N}(\hat{\mathbf{y}}, P_{yy})$ with [173]:

$$\hat{\mathbf{y}} \approx f(\hat{\mathbf{x}}), \quad P_{yy} \approx F P_{xx} F^T \quad \text{and} \quad F \coloneqq \left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}. \tag{4.2}$$

The relations (4.2) yield the exact moments of $\mathbf{y}$, if $f(\mathbf{x})$ is linear, and a fair approximation, if $f(\mathbf{x})$ is fairly linear or P_{xx} is fairly small. The first order Gauss approximation is very efficient compared to other transformations such as the second order Gauss approximation [103], the unscented transform [174] and the Monte Carlo transform [175]. Moreover, it turns out to work well in the practical application presented here.

4.2.1 Robust Feature Triangulation

At least two camera views are needed to triangulate a 3D point (*cf.* Section 2.3.2). However, in order to obtain a more reliable and accurate initial estimate, a larger window of camera views is used. The idea is illustrated in Figure 4.3. Let the eight-dimensional vector

Algorithm 4.1 Robust triangulation

Assume that a windowed buffer $\{\mathbf{y}_j\}_{j=1}^m$, with $m \geq 4$ in the experiments, of camera views is associated to feature (i). The deterministic sample consensus algorithm given in the sequel can then be used to obtain a least squares estimate $\hat{\mathbf{m}}_{w,t}^{LS}$ of the 3D feature location and the subset of camera views $\mathcal{I} \subseteq \{\mathbf{y}_j\}_{j=1}^m$ consistent with this estimate. An example configuration is shown in Figure 4.3.

1. Let $k \coloneqq 1$.
2. If the angle between the normalised view rays defined by $\mathbf{y}_k$ and $\mathbf{y}_m$, respectively, is smaller than a threshold β, with $\beta = 5^\circ$ in the experiments, go to 5.
3. Calculate a least squares estimate $\hat{\mathbf{m}}_{w,t}^{LS}$ from $\mathbf{y}_k$ and $\mathbf{y}_m$ as outlined in Section 2.3.2.
4. Determine the set of inliers $\mathcal{I}$ by applying a user-defined outlier threshold (2 pixels in the experiments) to the residuals of the image collinearity constraint (2.7a). If the inlier ratio is bigger than a threshold (60% in the experiments), stop with success.
5. Let $k \coloneqq k + 1$. If $k < m$, repeat from 2, else stop with failure.

$$\mathbf{y}_t^{(i)} \coloneqq [\hat{\mathbf{m}}_{n,t}^{(i)T}, \hat{\mathbf{s}}_t^{rT}]^T \tag{4.3}$$

denote the camera view of the i^{th} feature in the map at time t and let $\mathbf{d}_t^{(i)}$ be the corresponding view ray (*cf.* Section 2.1.3.1) with:

$$\mathbf{d}_t^{(i)} = \mathrm{rot}^T(\hat{\mathbf{r}}_{cw}) K^{-1} [\hat{\mathbf{m}}_{n,t}^{(i)T}, 1]^T, \tag{4.4}$$

where K denotes the matrix of the intrinsic camera parameters, K^{-1} is given in (2.3b) and $\hat{\mathbf{r}}_{cw}$ is obtained from $\hat{\mathbf{s}}_t^{rT}$. A windowed buffer $\{\mathbf{y}_j^{(i)}\}_{j=1}^m$ of past camera views is then associated to each feature (i), which is not yet initialised in 3D. Note that, in order to simplify the expressions, the subscript j is here used as buffer index. However, it can be replaced by the timestamp of the respective camera view to fit definition (4.3). After the pose estimation, the current view is added to the buffer as $\mathbf{y}_{m+1}^{(i)}$, if the buffer is empty or if:

$$\frac{\mathbf{d}_m^{(i)} \cdot \mathbf{d}_{m+1}^{(i)}}{\|\mathbf{d}_m^{(i)}\| \|\mathbf{d}_{m+1}^{(i)}\|} \geq \cos(\alpha). \tag{4.5}$$

This assures the angle between the normalised view rays of subsequent elements to be larger than a threshold α. In other words, subsequent elements differ sufficiently. The threshold $\alpha = 1^\circ$ and the buffer capacity 20 are used in the experiments. For the sake of notional brevity, the superscript (i) is suppressed in the sequel. However, all calculations refer to a single feature (i).

Once a buffer $\{\mathbf{y}_j\}_{j=1}^m$ exceeds a minimum size (4 in the experiments), Algorithm 4.1 is used to obtain an initial least squares estimate $\hat{\mathbf{m}}_{w,t}^{LS}$ of the 3D point and the subset of camera views $\mathcal{I} \subseteq \{\mathbf{y}_j\}_{j=1}^m$ that are consistent with this estimate. Hence, $\mathcal{I}$ is the set of inliers. The time subscript t is suppressed in the sequel. A valid estimate $\hat{\mathbf{m}}_w^{LS}$ is then used to compute a weighted least squares estimate $\hat{\mathbf{m}}_w^{WLS}$ from $\mathcal{I}$. Let

$$\mathbf{0}_2 = h(\mathbf{m}_w, \mathbf{y}) \coloneqq d_h(\hat{\mathbf{m}}_n, \mathbf{m}_w, \hat{\mathbf{s}}^r) \tag{4.6}$$

denote the measurement relation given by the homogeneous collinearity constraint $d_h(\cdot)$ (2.7b). The formulation of the weighted least squares problem is then analogous to (2.18):

$$\hat{\mathbf{m}}_w^{WLS} = \arg\min_{\mathbf{m}_w} \sum_{\mathbf{y}_j \in \mathcal{I}} h(\mathbf{m}_w, \mathbf{y}_j)^T P^{-1}(\mathbf{y}_j) h(\mathbf{m}_w, \mathbf{y}_j), \tag{4.7a}$$

where $P(\mathbf{y}_j)$ denotes the covariance of camera view $\mathbf{y}_j$. A first order Taylor expansion around the least squares estimate $\hat{\mathbf{m}}_w^{LS}$ provides a closed expression for (4.7a) [103]:

$$\hat{\mathbf{m}}_w^{WLS} \approx \left(\sum_{y_j \in I} H_j^T P^{-1}(\mathbf{y}_j) H_j \right)^{-1} \sum_{\mathbf{y}_j \in \mathcal{I}} H_j^T P^{-1}(\mathbf{y}_j) \left(H_j \hat{\mathbf{m}}_w^{LS} - h(\hat{\mathbf{m}}_w^{LS}, \mathbf{y}_j) \right) \tag{4.7b}$$

$$\operatorname{cov}(\hat{\mathbf{m}}_w^{WLS}) \approx \left(\sum_{y_j \in I} H_j^T P^{-1}(\mathbf{y}_j) H_j \right)^{-1}, \tag{4.7c}$$

with Jacobian:

$$H_j \coloneqq \left. \frac{\partial h(\mathbf{m}_w, \mathbf{y}_j)}{\partial \mathbf{m}_w} \right|_{\mathbf{m}_w = \hat{\mathbf{m}}_w^{LS}}$$

and $P_{ww} \coloneqq \operatorname{cov}(\hat{\mathbf{m}}_w^{WLS})$. The covariance $P(\mathbf{y}_j)$ results from (4.2) using the stochastic model given in (4.1):

$$P(\mathbf{y}_j) \approx \begin{bmatrix} H_j^n & H_j^{\mathbf{s}} \end{bmatrix} \begin{bmatrix} R_{nn,j} & 0 \\ 0 & P_{\mathbf{s}^r \mathbf{s}^r, j} \end{bmatrix} \begin{bmatrix} H_j^{nT} \\ H_j^{\mathbf{s}T} \end{bmatrix}, \tag{4.7d}$$

with Jacobians:

$$H_j^n \coloneqq \frac{\partial h(\hat{\mathbf{m}}_w^{LS}, \mathbf{y}_j)}{\partial \mathbf{m}_n} \qquad\qquad H_j^{\mathbf{s}} \coloneqq \frac{\partial h(\hat{\mathbf{m}}_w^{LS}, \mathbf{y}_j)}{\partial \mathbf{s}}.$$

The weighted least squares estimate $\hat{\mathbf{m}}_w^{WLS}$ is accepted, if the following criteria are fulfilled:

- The ratio of camera views that are consistent with $\hat{\mathbf{m}}_w^{WLS}$ is larger than a threshold (60% in the experiments).
- The 3D location $\hat{\mathbf{m}}_w^{WLS}$ is in front of all consistent cameras.

- The ratio $\lambda_{\max}/z_c$ between the square root of the largest eigenvalue, $\lambda_{\max}$, of $\mathrm{cov}(\hat{\mathbf{m}}_w^{WLS})$ and the distance of the feature to the camera, z_c, is smaller than a threshold (10% in the experiments). In other words, the uncertainty in depth is significantly smaller than the parameter estimate itself (*cf.* Figure 4.7). Otherwise, the extended Kalman filter is likely to diverge due to large linearisation errors.

A valid estimate $\hat{\mathbf{m}}_w^{WLS}$ with covariance $\mathrm{cov}(\hat{\mathbf{m}}_w^{WLS})$ is used as a priori distribution to initialise the extended Kalman filter associated to the considered feature. If the estimate is rejected, the triangulation is tried again, when the next camera view $\mathbf{y}_{m+1}$ is added to the buffer.

4.2.2 Robust Feature Refinement

Once, the extended Kalman filter associated to a feature is initialised, standard measurement updates are performed to estimate the 3D location recursively. The prediction step is not needed, as the features are assumed stationary. Each registration of the feature in a camera image provides a new measurement $\mathbf{y}_t$ as defined in (4.3). The measurement model is basically given in (4.6) and the linearised covariance of the measurement noise is given in (4.7d). These expressions are here restated in the standard notation of the extended Kalman filter framework. Let $\mathbf{e}_t \coloneqq [\mathbf{e}_{n,t}^T, \mathbf{e}_{\mathbf{s}^r,t}^T]^T \sim \mathcal{N}(\mathbf{0}_8, R_t)$ with:

$$R_t = \begin{bmatrix} R_{nn,t} & 0_{2\times 6} \\ 0_{6\times 2} & P_{\mathbf{s}^r\mathbf{s}^r,t} \end{bmatrix} \tag{4.8}$$

denote the measurement noise and let $\mathbf{x}_t \coloneqq \mathbf{m}_{w,t}$ denote the state vector. Relation (4.6) can then be put in the form of the nonlinear measurement model (2.25b):

$$\begin{aligned} \mathbf{0}_2 &= h(\mathbf{x}_t, \mathbf{y}_t, \mathbf{e}_t) \\ &= \begin{bmatrix} \mathrm{I}_2 & -(\hat{\mathbf{m}}_{n,t} + \mathbf{e}_{n,t}) \end{bmatrix} \mathcal{T}(\mathbf{x}_t, \hat{\mathbf{s}}^{rT} + \mathbf{e}_{\mathbf{s}^r,t}^T). \end{aligned} \tag{4.9}$$

Here, the homogeneous collinearity constraint is completed with noise. The linearised model required in the EKF measurement update (*cf.* Algorithm 2.3) can be obtained from (2.37b) by calculating the respective Jacobians. The complicated derivative $\frac{\partial \,\mathrm{rot}(\mathbf{r}_{cw})\mathbf{m}_w}{\partial \mathbf{r}_{cw}}$ is provided in (B.11) in Appendix B.2.

In order to obtain a robust algorithm, the outlier rejection procedure described in Section 3.4.2 is applied also here. A standard χ^2 test with significance level $\alpha = 0.05$ is used. The outlier test is given in (3.18).

4.2.3 Robust Pose Estimation

The pose is estimated in two stages. Assume that appropriate sets of 2D/3D and 2D/2D correspondences are provided by the procedure described in Section 4.3. Then, in the first stage a rough pose is computed from the 2D/3D correspondences using the Levenberg-Marquardt algorithm with the previous pose as initial guess. The RANSAC framework is used to estimate

the pose in order to remove possible outliers. The resulting inliers are used in the second stage, where epipolar constraints given by the 2D/2D correspondences and the previous camera pose are incorporated. Both stages are detailed in the sequel.

4.2.3.1 Estimation from 2D/3D Correspondences

The aim of this section is to compute a weighted least squares estimate $\hat{\mathbf{s}}_t^{rWLS}$ of the camera pose with covariance $\mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS})$. The procedure is derived analogously to Section 4.2.1 by changing the role of measurement and unknown. Expressions similar to the camera measurement model developed in (3.8) are obtained. Let the five-dimensional vector

$$\mathbf{y}_t^{nw,(i)} \coloneqq [\hat{\mathbf{m}}_{n,t}^{(i)T}, \hat{\mathbf{m}}_{w,t}^{(i)T}]^T \tag{4.10}$$

denote the i^{th} 2D/3D correspondence with $\mathbf{m}_{n,t}^{(i)} \sim (\hat{\mathbf{m}}_{n,t}^{(i)}, R_{nn,t}^{(i)})$ and $\mathbf{m}_{w,t}^{(i)} \sim (\hat{\mathbf{m}}_{w,t}^{(i)}, P_{ww,t}^{(i)})$, where the estimate of $\mathbf{m}_{w,t}^{(i)}$ is obtained from the extended Kalman filter associated to the respective feature in the map. Let $\{\mathbf{y}_t^{nw,(i)}\}_{i=1}^n$ denote a set of such correspondences established for time t. Note that (i) is here used for indexing the 2D/3D correspondences rather than the feature map. Furthermore, let

$$\mathbf{0}_2 = h(\mathbf{s}_t^r, \mathbf{y}_t^{nw}) \coloneqq d_n(\hat{\mathbf{m}}_{n,t}, \hat{\mathbf{m}}_{w,t}, \mathbf{s}_t^r) \tag{4.11}$$

denote the measurement relation, here based on the image collinearity constraint $d_n(\cdot)$ (2.7a). The formulation of the weighted least squares problem is then analogous to (2.18):

$$\hat{\mathbf{s}}_t^{rWLS} = \arg\min_{\mathbf{s}_t^r} \sum_{i=1}^{n} h(\mathbf{s}_t^r, \mathbf{y}_t^{nw,(i)})^T P^{-1}(\mathbf{y}_t^{nw,(i)}) h(\mathbf{s}_t^r, \mathbf{y}_t^{nw,(i)}), \tag{4.12a}$$

where the quantity inside the sum is referred to as the Mahalanobis distance (*cf.* (2.18)) and $P(\mathbf{y}_t^{nw,(i)})$ denotes the covariance of correspondence $\mathbf{y}_t^{nw,(i)}$. Analogously to (4.7d), the first order Taylor approximation of this covariance is:

$$P(\mathbf{y}_t^{nw,(i)}) \approx \begin{bmatrix} H_t^{n,(i)} & H_t^{w,(i)} \end{bmatrix} \begin{bmatrix} R_{nn,t}^{(i)} & 0 \\ 0 & P_{ww,t}^{(i)} \end{bmatrix} \begin{bmatrix} H_t^{n,(i)T} \\ H_t^{w,(i)T} \end{bmatrix}, \tag{4.12b}$$

with Jacobians:

$$H_t^{n,(i)} \coloneqq \frac{\partial h(\hat{\mathbf{s}}_t^{rWLS}, \mathbf{y}_t^{nw,(i)})}{\partial \mathbf{m}_n} \qquad H_t^{w,(i)} \coloneqq \frac{\partial h(\hat{\mathbf{s}}_t^{rWLS}, \mathbf{y}_t^{nw,(i)})}{\partial \mathbf{m}_w}.$$

The problem (4.11) is solved by using the Levenberg-Marquardt algorithm with the previous pose $\hat{\mathbf{s}}_{t-T}^{rWLS}$ as initial guess. The covariances $P(\mathbf{y}_t^{nw,(i)})$ are estimated once before the minimisation using the initial guess as linearisation point. In order to remove possible outliers, (4.12a) is solved iteratively. The RANSAC procedure is outlined in Algorithm 4.2. However,

Algorithm 4.2 Robust pose estimation from 2D/3D correspondences

Assume that a set of 2D/3D correspondences $\{\mathbf{y}_t^{nw,(i)}\}_{i=1}^n$, with $n \geq 4$ in the experiments, is given. The following algorithm can then be used to obtain a weighted least squares estimate $\hat{\mathbf{s}}_t^{rWLS}$ of the camera pose and the subset of correspondences $\mathcal{I} \subseteq \{\mathbf{y}_t^{nw,(i)}\}_{i=1}^n$ that are consistent with this estimate. Let m denote the maximum number of iterations and p the size of a sample set, with $m = 50$ and $p = 4$ during the experiments:

1. Let $k := 1$ and $\hat{\mathbf{s}}_t^{rWLS} := \hat{\mathbf{s}}_{t-T}^{rWLS}$.
2. Calculate a weighted least squares estimate $\hat{\mathbf{s}}_t^{rWLS}$ from all correspondences using (4.12).
3. Determine the set of inliers $\mathcal{I}$ by applying a user-defined outlier threshold (3 pixels in the experiments) to the Mahalanobis distances. If the inlier ratio is bigger than a threshold (60% in the experiments), go to 5.
4. Iterate (RANSAC).
   (a) Randomly choose a sample set of p correspondences.
   (b) Calculate a weighted least squares estimate $\hat{\mathbf{s}}_t^{rWLS}$ from the sample set using (4.12).
   (c) Determine $\mathcal{I}$. If the inlier ratio is bigger than a threshold go to 5
   (d) Let $k := k + 1$. If $k < m$, repeat from 4a, else stop with failure.
5. Compute a final estimate $\hat{\mathbf{s}}_t^{rWLS}$ from the set $\mathcal{I}$ and calculate the covariance $\mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS})$ using (4.12c).

experiments show that on average only few iterations are needed, since the image processing approach and the feature selection procedure (*cf.* Section 4.3) used ensure few outliers. Assuming that a pose estimate $\hat{\mathbf{s}}_t^{rWLS}$ and a set of inliers $\mathcal{I}$ is obtained from Algorithm 4.2, the covariance $\mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS})$ can be approximated analogously to (4.7c):

$$\mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS}) \approx \left(\sum_{\mathbf{y}_t^{nw,(i)} \in I} H_t^{(i)T} P^{-1}(\mathbf{y}_t^{nw,(i)}) H_t^{(i)} \right)^{-1}, \tag{4.12c}$$

with Jacobian:

$$H_t^{(i)} := \left. \frac{\partial h(\mathbf{s}_t^r, \mathbf{y}_t^{nw,(i)})}{\partial \mathbf{s}_t^r} \right|_{\mathbf{s}_t^r = \hat{\mathbf{s}}_t^{rWLS}}$$

and $P_{\mathbf{s}^r\mathbf{s}^r,t} := \mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS})$.

4.2.3.2 Refinement Adding Epipolar Constraints

The weighted least squares estimate $\hat{\mathbf{s}}_t^{rWLS}$ obtained from the first stage is refined by incorporating epipolar constraints implied by the 2D/2D correspondences and the previous camera pose $\hat{\mathbf{s}}_{t-T}^{rWLS}$. Let the four-dimensional vector

$$\mathbf{y}_t^{nn,(j)} \coloneqq [\hat{\mathbf{m}}_{n,t}^{(j)T}, \hat{\mathbf{m}}_{n,t-T}^{(j)T}]^T \tag{4.13}$$

denote the j^{th} 2D/2D correspondence and let $\{\mathbf{y}_t^{nn,(j)}\}_{j=1}^m$ denote a set of such correspondences established for time t. Furthermore, let

$$0 = h(\mathbf{s}_t^r, \hat{\mathbf{s}}_{t-T}^{rWLS}, \mathbf{y}_t^{nn}) \coloneqq d_s(\hat{\mathbf{m}}_{n,t}, \hat{\mathbf{m}}_{n,t-T}, E) \tag{4.14}$$

denote the measurement relation, where the symmetric epipolar distance $d_s(\cdot)$ is given in (2.20) and the essential matrix E is obtained from $\mathbf{s}_t^r$ and $\hat{\mathbf{s}}_{t-T}^{rWLS}$ as in (2.8). In the second pose estimation stage, the image collinearity constraints implied by the consistent 2D/3D correspondences $\mathcal{I} \subseteq \{\mathbf{y}_t^{nw,(i)}\}_{i=1}^n$ and the epipolar constraints given by the 2D/2D correspondences $\{\mathbf{y}_t^{nn,(j)}\}_{j=1}^m$ and the previous pose $\hat{\mathbf{s}}_{t-T}^{rWLS}$ are minimised simultaneously. The estimate of the first stage is used as initial guess in:

$$\hat{\mathbf{s}}_t^{rWLS} = \underset{\mathbf{s}_t^r}{\arg\min} \sum_{\mathbf{y}_t^{nw,(i)} \in I} h(\mathbf{s}_t^r, \mathbf{y}_t^{nw,(i)})^T P^{-1}(\mathbf{y}_t^{nw,(i)}) h(\mathbf{s}_t^r, \mathbf{y}_t^{nw,(i)}) + \sum_{j=1}^m \rho\big(h^2(\mathbf{s}_t^r, \hat{\mathbf{s}}_{t-T}^{rWLS}, \mathbf{y}_t^{nn,(j)})\big). \tag{4.15}$$

The TUKEY influence function $\rho(\cdot)$ (2.19) is used to handle outliers among the 2D/2D correspondences. The first term in 4.15 is identical to (4.12a). The covariances $P(\mathbf{y}_t^{nw,(i)})$ are again estimated once before the minimisation using the initial guess as linearisation point. In the second term, since weights are provided by $\rho(\cdot)$, a least-squares formulation is used without taking the covariances into account. The final covariance $\mathrm{cov}(\hat{\mathbf{s}}_t^{rWLS})$ is obtained from (4.12c) using the refined estimate as linearisation point. The effective covariance can be assumed smaller, as more information — provided by the second term — is actually included in the estimate.

4.3 Feature Quality and Selection

In order to reduce the error drift in the 3D estimation processes, it is crucial to select only the best features in the map for providing 3D constraints to the pose estimation. The aim of this section is therefore to develop a measure of feature quality that can be used as selection criterion.

The problem of monitoring the quality of a feature is related to divergence monitoring as treated in Section 3.4.3. Hence, the same change detection technique is applied here: a geometric moving average (3.19) $g_t^{(i)}$ (with $\lambda = 0.95$ in the experiments) of the distance measure $s_t^{(i)}$ serves as indicator for the quality of feature (i). Here, $s_t^{(i)}$ is chosen to be the normalised residual of the image collinearity constraint, computed after pose estimation. Using the notation and definitions of Section 4.2.3.1, $s_t^{(i)}$ is:

$$s_t^{(i)} = h(\hat{\mathbf{s}}_t^{rWLS}, \mathbf{y}_t^{nw,(i)})^T \left(P(\mathbf{y}_t^{nw,(i)}) + H_t^{(i)} P_{\mathbf{s}^r\mathbf{s}^r,t} H_t^{(i)T}\right)^{-1} h(\hat{\mathbf{s}}_t^{rWLS}, \mathbf{y}_t^{nw,(i)}), \tag{4.16}$$

where $\hat{\mathbf{s}}_t^{rWLS}$ denotes the pose of the current image obtained from the procedure in Section 4.2.3.2.

The indicator $g_t^{(i)}$ is initialised to a high value (50 in the experiments) after triangulation and is then updated according to (3.19), each time the feature is successfully registered. Before the pose computation at time t, a threshold is then applied to $g_t^{(i)}$, in order to decide, whether the registered feature (i) should provide a collinearity or an epipolar constraint. Hence, a feature, which repeatedly shows up as a mismatch or as roughly localised in 3D, does not affect the pose estimate.

4.4 Experimental Setup and Results

This section provides an experimental evaluation of the visual SLAM system developed in this chapter. The accuracy and computational efficiency of the system are evaluated in three different experiments. Especially the value of the improvements made to the 3D estimation processes and the basic stochastic model are investigated. The individual setups and results are presented subsequently. The system parameters used during the experiments are summarised in Table 4.1.

4.4.1 Test Case: Room

This test case denotes a corner of a room, *i.e.* a mid-scale environment with simple geometry that allows for qualitative evaluation of the reconstruction accuracy without ground truth. A cubic object with known 3D geometry is placed in the scene and serves as prior knowledge to initialise the predictive tracking loop. The computational efficiency and the accuracy of the feature reconstruction subsystem are investigated in this section. In particular, it is shown, how the developed algorithms improve the reconstruction accuracy compared to a basic implementation of the structure from motion loop.

In this scenario, the developed system is able to estimate the camera trajectory in real-time while reconstructing 3D points in a volume of approximately $2 \times 2.8 \times 2.5$ m around the origin defined by the known object. Figure 4.4 shows images of the target setup. For an in-depth evaluation, a representative image sequence has been captured. The sequence starts with continuous camera movements, where the whole scene as shown in Figure 4.4(a) is scanned

Table 4.1: *Parameter settings used during the experiments. The * indicates quantities that are weighted with the respective covariance.*

parameter	value	unit
camera		
field of view	35	[°]
image resolution	320×240	[pixel]
covariance R_{pp} (*cf.* Section 4.2)	I_2	[pixel]
robust feature triangulation (*cf.* Section 4.2.1)		
buffer capacity	20	[frames]
minimum angle α	1	[°]
minimum angle β	5	[°]
outlier threshold	2	[pixel]
minimum inlier ratio	60	[%]
maximum depth uncertainty	10	[%]
robust feature refinement (*cf.* Section 4.2.2)		
significance level α	0.05	
robust pose estimation (*cf.* Section 4.2.3)		
maximum number of iterations m	50	
outlier threshold	3	[pixel*]
minimum inlier ratio	60	[%]
feature quality and selection (*cf.* Section 4.3)		
initial value for g_t	50	[pixel]
forgetting factor λ	0.95	
selection threshold for g_t	2	[pixel*]

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

Figure 4.4: *Room: (a) illustrates a wide-angle view on the entire target scene, (b) shows an image from the camera used for the experiments. It has a much narrower field of view of 35°. In (c), the CAD model of the known object is shown. The object coordinate system defines the reference frame for tracking and reconstruction.*

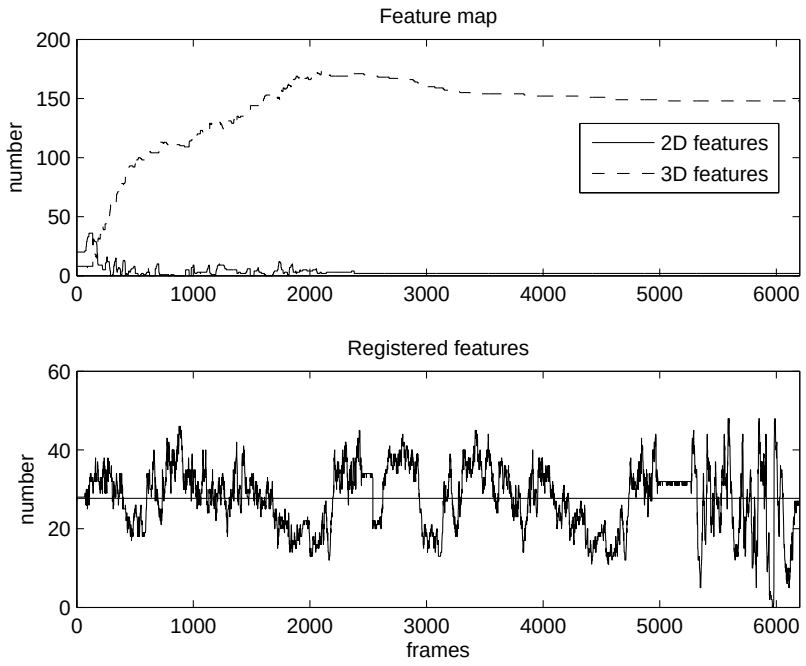


Figure 4.5: *Room: the upper plot illustrates the development of the feature map in terms of the number of 2D and 3D features at each frame. 2D features denote features that are not yet initialised in 3D. The lower plot shows the number of features registered in each frame with the mean indicated by the straight line.*

once. Afterwards, the movements are faster and the camera revisits parts of the environment. In order to show the ability of the image processing method to recover feature tracks and to evaluate the computational efficiency of the reconstruction component, the addition of new features is manually disabled, once the model is acquired during the first part of the sequence. This makes sure that the tracking thenceforward is entirely based upon the model acquired already. Figure 4.5 presents the development of the feature map. The upper plot shows a constant addition and reconstruction of new point features until around frame 2000, where the scene structure has been acquired with sufficient feature density. Note that new features are initialised in 3D within a few frames. The structure estimation is disabled manually at frame 2540. After that there is a slight decrease in the number of features until frame 3380, after which the map is almost stable until the end of the sequence. This demonstrates the ability of the developed system to recover feature tracks. On average, 27 features are registered per camera frame. The progression is given in the lower plot of Figure 4.5 and the jitter especially at the end of the sequence indicates that the camera moves fast.

Figure 4.6 demonstrates the computational performance of tracking and structure estimation on this image sequence. The upper part contains plots of the processing time needed for tracking — including feature prediction, registration, selection and pose computation — and

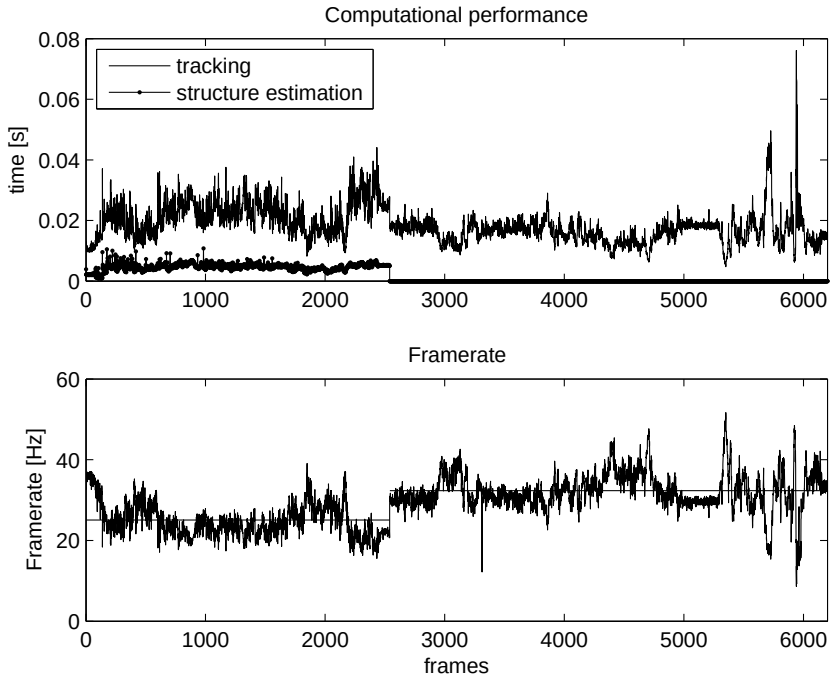


Figure 4.6: *Room: the upper graph plots the processing time needed for tracking and structure estimation in comparison. The lower graph plots the frame rate with the averages before and after disabling the structure estimation indicated by the straight lines.*

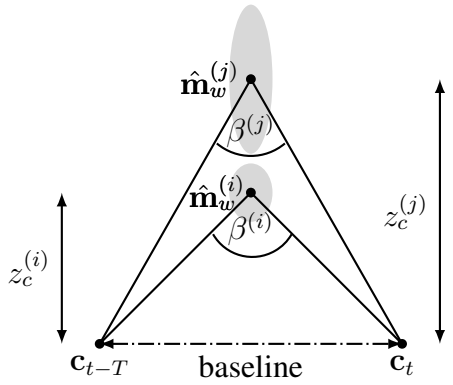
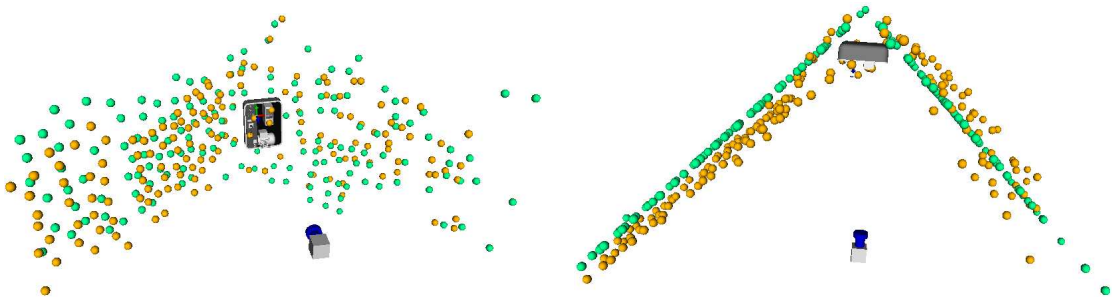


Figure 4.7: *Triangulation: The ellipses indicate the 3D location covariances $P_{ww}^{(i)}$ and $P_{ww}^{(j)}$. Obviously, $\beta^{(i)} > \beta^{(j)}$ and $\hat{\mathbf{m}}_w^{(j)}$ is reconstructed with larger uncertainty in depth $z_c^{(j)}$. Hence, distant features require a larger baseline to meet the angle and the depth uncertainty constraint (*cf. *Section 4.2.1).*

for reconstruction — including feature detection, triangulation and refinement. The dashed curve drops to zero after disabling the reconstruction component. The figure shows clearly, how the reconstruction is performed efficiently, needing on average one fifth of the time of the tracking. Note that the robust triangulation is the most expensive part of the reconstruction. The recursive estimation is much more efficient. As mentioned before, in the mid-scale environment considered here, features are initialised in 3D within a few frames. In contrast, the triangulation of distant features in a large-scale environment requires a comparatively large baseline in order to obtain a sensible prior estimate to initialise the extended Kalman filter. This is enforced by the angle and the depth uncertainty constraint (*cf.* Section 4.2.1) and is illustrated in Figure 4.7. If a sufficient baseline is not provided, the features are nevertheless used in the second pose estimation stage. However, computational resources are spent on the triangulation attempts, thus limiting the applicability of the approach to large-scale environments.

The lower part of Figure 4.5 plots the frame rate incorporating the time needed for the image resampling to compensate for distortion (*cf.* Section 2.1.1.2). On average, a frame rate of 31.49 Hz is achieved while reconstructing features and a frame rate of 43.59 Hz without structure estimation. This has been measured on a desktop PC with 3 GHz Intel Pentium IV CPU. The frame rate depends of course on many different factors: the number of features in the current field of view, the overall number of features in the map, the continuity of the camera movements, occlusions in the scene and the depth of the scene. Additional time is needed for image acquisition and virtual augmentations. Altogether the developed system achieves frame rates around 20 Hz, when a reasonable number of features is used for tracking and a mid-scale environment is considered, as during this experiment. The average number of iterations (*cf.* Algorithm 4.2) needed during the pose computation throughout this image sequence is 1.03 with a maximum of 5. In 97.49% of the frames, no iterations are needed at all.



(a) Frontal view *(b) Top view*

Figure 4.8: *Room: two views on the reconstructed point clouds obtained from the system running in advanced (green) and basic mode (orange). The known object defining the world reference frame and the initial camera view are presented as well. Note how the orange point cloud is biased compared to the green one.*

Figure 4.8 illustrates two reconstructed point clouds in comparison, together with the CAD model of the known object, the world coordinate system and the initial camera view. The green point cloud has been generated with the system running the complete algorithm as developed in this chapter. Subsequently, this mode is referred to as *advanced system mode*. The green point set clearly exhibits the two dominant planes of the target scene. In contrast, the orange point cloud reveals obvious errors. A bias shows in the estimated depths of the points on the left wall, reconstructed first, and the errors increase significantly over the right wall. This reconstruction has been generated with the system running in a *basic mode*, where important enhancements made to the standard structure from motion loop are disabled. In the basic mode, the system still uses the drift-free image processing method and the robust triangulation (*cf.* Section 4.2.1) and pose computation (*cf.* Section 4.2.3). However, the feature selection (*cf.* Section 4.3) based on the lowpass filtered normalised residuals, the incorporation of the epipolar constraints (*cf.* Section 4.2.3.2) and the stochastic model including the feature refinement (*cf.* Section 4.2.2) are disabled.

From groups of features, which are known to be nearly coplanar, as the features on the two walls of the target scene, compensating planes can be estimated. By computing the distances of the features to these planes, the approximate reconstruction error can be quantified. Using this method, the reconstruction error of the developed system running in the advanced mode, is in this specific experimental setup measured with 0.26 cm mean and 0.20 cm standard deviation. Note that this experiment is not suitable to identify a drift in the reconstruction, as the planes are not known by ground truth.

4.4.2 Test Case: Simulation

In this test case, the improvement in the camera localisation and the reconstruction accuracy gained by the advanced system mode is quantified with help of ground truth. Therefore the

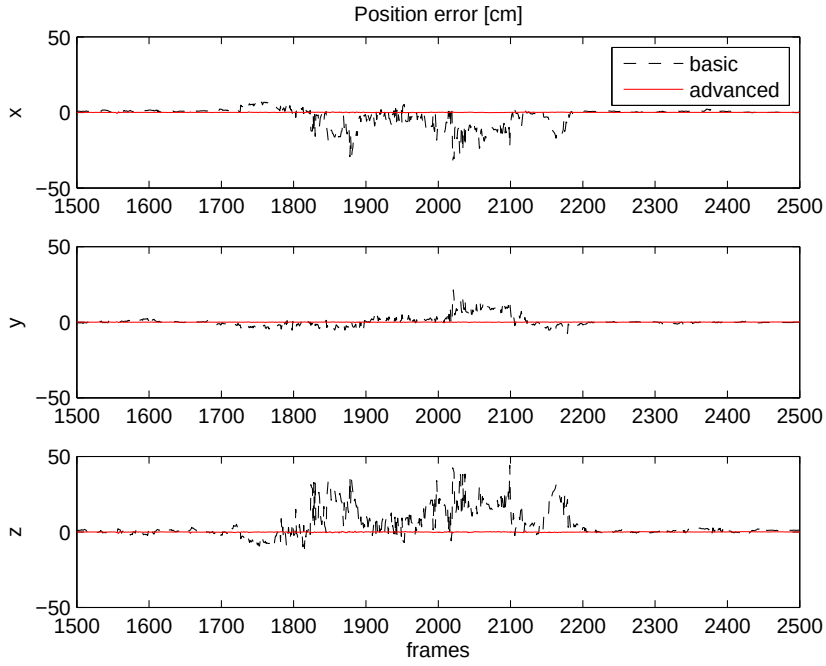


Figure 4.9: *Simulation: position errors obtained from the basic and the advanced system mode on a part of the synthetic image sequence.*

image sequence of the room test case is reproduced synthetically. This is done by building an idealised CAD model of the target scene consisting of two orthogonal planes with a real texture and rendering this with the internal camera parameters and the estimated trajectory. The errors in the estimated camera position and orientation obtained from the advanced and the basic system mode are compared in Figures 4.9 and 4.10. The average errors are given in Table 4.2. The results demonstrate clearly the superiority of the advanced mode. A further conclusion to draw from Figures 4.9 and 4.10 concerns the feature registration in the developed system. The error plots for the basic mode show high peaks but no systematic drift. Hence, the errors accumulated during the exploration of the scene is reset, when previously reconstructed features return into the field of view. This again shows the ability of the image processing method to recover feature tracks.

In this test case, the 3D geometry of the target scene is known by ground truth. Hence, the reconstruction errors can be quantified exactly by computing the distances of the points to the accordant planes in the model. The mean of the absolute error is $0.17\,\text{cm}$ and the standard deviation is $0.23\,\text{cm}$. Figure 4.11 plots the errors against the distances of the features from the origin, where the reconstruction started. The upper plot shows the orthogonal distances from the respective plane. The mean is $0.05\,\text{cm}$ indicating a small bias, whereas the errors tend to increase with the distance from the origin. However, the increasing errors are consistently reflected by the 3D covariances associated with the features. This can be seen in the lower

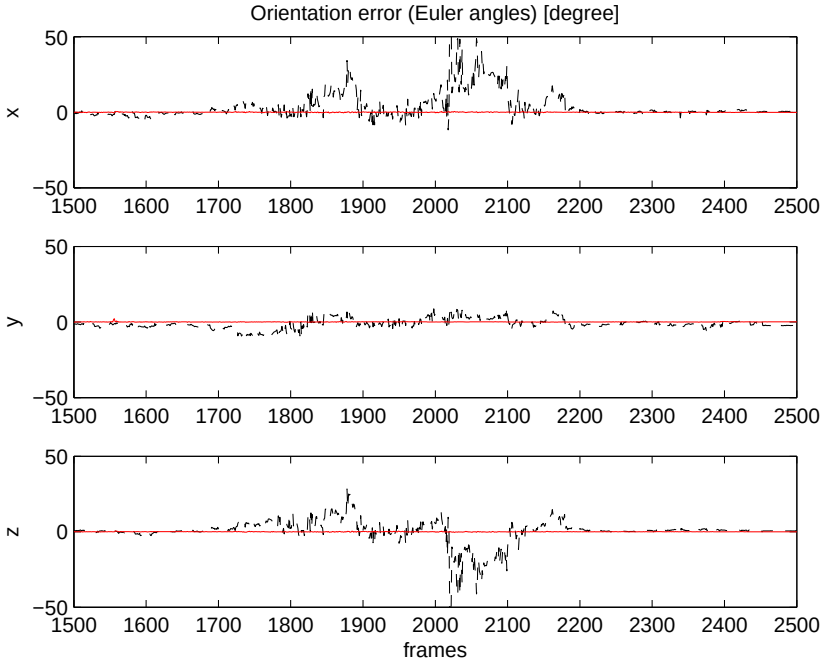


Figure 4.10: *Simulation: orientation errors obtained from the basic and the advanced system mode on a part of the synthetic image sequence.*

Table 4.2: *Simulation: average camera localisation errors obtained from the basic and the advanced system mode on the synthetic image sequence. The orientation error is given in Euler angles.*

	basic mode		**advanced mode**	
	mean	std	mean	std
position error [cm]				
x	1.49	3.18	0.09	0.17
y	0.86	1.80	0.08	0.19
z	2.60	6.16	0.18	0.27
orientation error [°]				
x	2.38	4.72	0.17	0.61
y	1.71	2.03	0.15	0.34
z	1.94	3.93	0.12	0.45

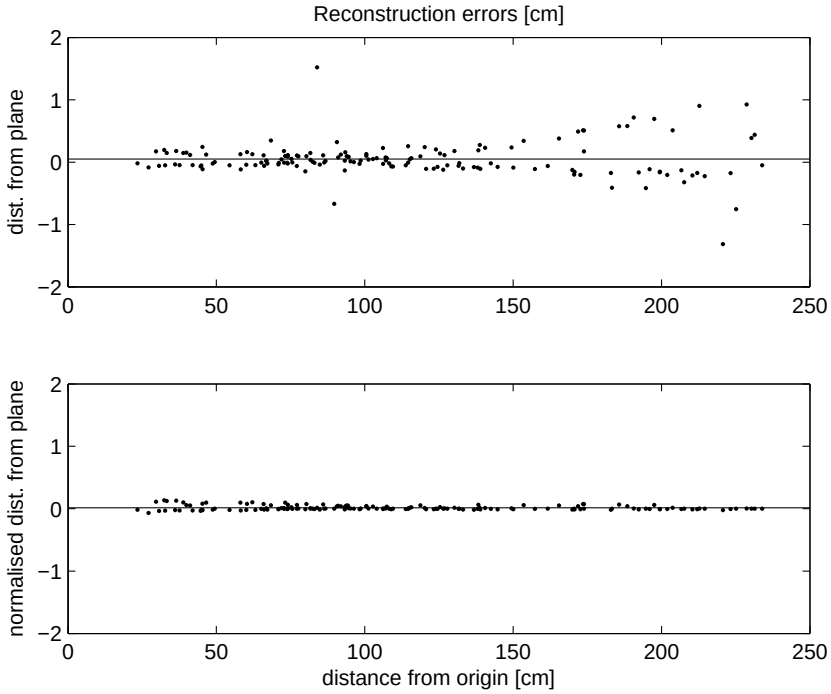


Figure 4.11: *Simulation: reconstruction errors obtained from the advanced system mode. The upper graph plots the orthogonal distances of the 3D features from the model planes against the distance from the origin. The lower graph plots the distances weighted with the covariances (Mahalanobis distances) associated with the features. The mean errors are indicated by the straight lines.*

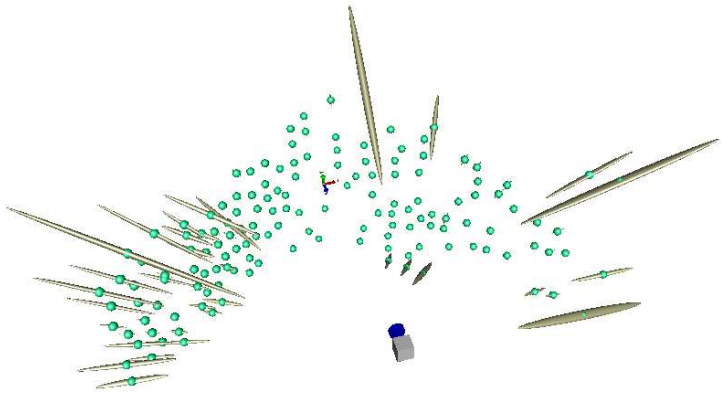


Figure 4.12: *Simulation: reconstructed point cloud. The mapped point landmarks are shown with their location covariances. The points in the outer regions of the target scene tend to be more uncertain. Some uncertain features, which are only just triangulated, are also visible close to the origin.*

graph of Figure 4.11, where the Mahalanobis distances are shown. Let $\hat{\mathbf{m}}_w^{(i)} = [x_w, y_w, z_w]^T$ be the estimated 3D location of feature (i) and let $P_{ww}^{(i)}$ be the associated covariance. Let $\mathbf{p} = [a, b, c, d]$ denote a known homogeneous plane. The orthogonal distance $\hat{d}^{(i)} := d(\hat{\mathbf{m}}_w^{(i)}, \mathbf{p})$ is then:

$$d(\hat{\mathbf{m}}_w^{(i)}, \mathbf{p}) = \frac{ax_w + by_w + cz_w + d}{\sqrt{a^2 + b^2 + c^2}} \tag{4.17}$$

and the normalised distance $\bar{d}^{(i)}$ is:

$$\bar{d}^{(i)} = (JP_{ww}^{(i)}J^T)^{-1/2}\hat{d}^{(i)} \tag{4.18}$$

with $J = [a/l, b/l, c/l]$ and $l = \sqrt{a^2 + b^2 + c^2}$. The transformation of the covariance follows from (4.2). In order to make the graphs in Figure 4.11 comparable, the Mahalanobis distances are normalised with the minimum weight. The resulting plot shows an even error distribution. This demonstrates the consistency of the 3D covariances and the value of the basic stochastic model. An illustration of the covariances in 3D space is given in Figure 4.12.

4.4.3 Test Case: Loop

In this test case, the ability of the developed system to close loops is investigated. A $2.1 \times 1.5 \times 2.5$ m part of a corridor serves as test environment (*cf.* Figure 4.13). Starting from the known object as in the previous experiments, the developed system is able to close the loop over the two walls, the floor and the ceiling. Figure 4.14 demonstrates the point clouds obtained from processing a captured image sequence in the advanced (green) and in the basic system mode (orange). The orange point cloud shows a significant drift, while the green cloud demonstrates a closed loop and allows for identifying details in the scene structure. These results show that

Figure 4.13: *Loop: a fisheye view on the target scene, where posters are placed to obtain discriminative features. Note, that the poster leaning on the right wall and the poster fixed to the ceiling are slightly curved and that the bottom is not completely coplanar.*

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

Figure 4.14: *Loop: reconstructed point clouds obtained from the system running in the advanced (green) and in the basic mode (orange). The orange point cloud shows a significant drift. In contrast, the green point cloud allows for identifying scene structures visible in the real environment (*cf. *Figure 4.13): (a) demonstrates the consistent reconstruction of points on the slightly curved posters in the ceiling and the right wall and (b) shows details of the left wall and the floor. Points are reconstructed in different depths on the left wall, on the extinguisher, the frame of the balcony door and the balcony floor itself.*

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

Figure 4.15: *Loop: (a) shows the first image of the sequences, where the wireframe model of the known object is perfectly registered. In (b), the amount of drift after the complete loop is demonstrated. In (c) the camera localisation error is reset by recovering features reconstructed during the first part of the sequence.*

a significant drift reduction is accomplished by the developed enhancements. Nevertheless, the drift is not eliminated completely as demonstrated in Figure 4.15. After the loop, when the known object returns into the field of view, an error of several pixels is visible in the augmentation. Since the drift is small, previously reconstructed features are recovered and the error in the camera localisation is reset. Moreover, the 3D feature positions are corrected locally via the recursive refinement. However, the correction has no global effects, as the features are estimated each in a separated Kalman Filter, *i.e.* correlations among the features are not modelled.

Figure 4.16 plots the frame rates achieved when processing the image sequence in the advanced system mode on a laptop with 1.66 GHz Intel core duo CPU. The average frame rate is 33 Hz and a mean of 29 features are registered in each frame. The frame rate decreases slowly with the increasing number of features in the map. At the end of the sequence, 468 active features are in the map. Hence, the developed algorithm scales well with the number of features.

4.5 Conclusion

In this chapter, a vision-only camera tracking system is described, which needs only minimal prior 3D knowledge about the target environment and is capable of performing robust camera localisation and mapping simultaneously in real-time. Hence, the system is suitable for enabling real-time tracking in partially known environments also beyond the areas originally modelled.

The main contribution of this chapter is to unify the robustness and efficiency of standard sequential structure from motion techniques with the statistical model of the extended Kalman filter to obtain an efficient and robust algorithm with minimal drift. This is achieved by using robust computer vision methods throughout the estimation processes and by performing the

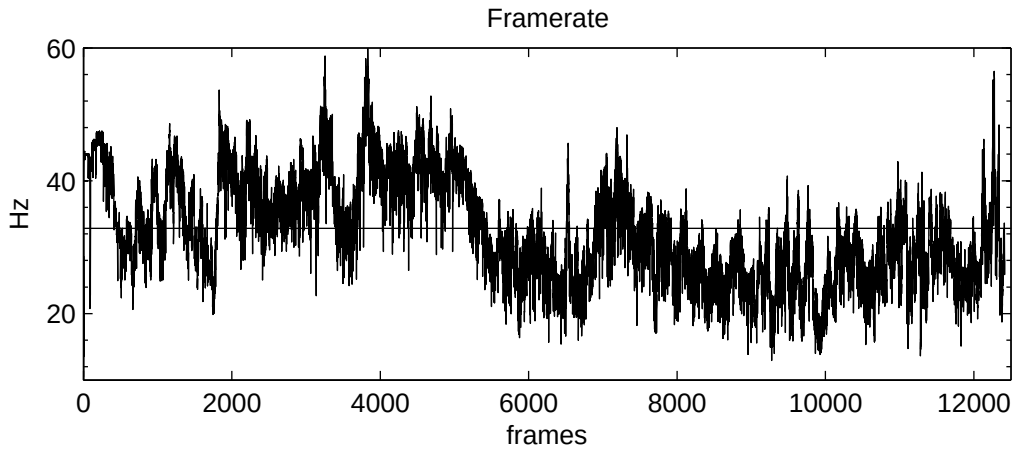


Figure 4.16: *Loop: the frame rates include the time needed for image resampling, tracking and structure estimation. Image acquisition and visualisation of virtual augmentations are not taken into account. The average frame rate is indicated by the straight line.*

estimations in the stochastic framework of the extended Kalman filter. The robust computer vision methods are:

- The image processing method allows for a reliable detection of outliers already during the feature registration.
- The initial triangulation of features is computed from a large window of camera views in a robust manner.
- During feature refinement, a χ^2 test prohibits the use of incorrect measurements in the EKF measurement update.
- The first pose estimation stage is performed robustly in the RANSAC framework.
- In the second pose estimation state, the TUKEY influence function is used to handle outliers among the 2D/2D correspondences.

Using the stochastic framework permits recursive estimation of the 3D feature locations and provides a measure of their location uncertainties. This measure is given by the 3D covariances and the lowpass filtered normalised residuals. The latter are used to select appropriate features for providing 3D constraints to the pose estimation. As a further enhancement to the ordinary structure from motion loop, features are used for the pose estimation from the time of their initial detection, irrespectively of their 3D state, by giving epipolar constraints.

The experimental evaluation, using both simulated and realistic image sequences, clearly shows the benefits of the developed method compared to an ordinary sequential structure from

motion system. These benefits are a higher accuracy in the camera localisation and the reconstructed feature locations and a significant drift reduction. Moreover, in the considered mid-scale indoor environments, the basic stochastic model turns out to provide a consistent uncertainty measure of the 3D feature locations.

The visual SLAM system presented in this chapter achieves a significant drift reduction while scaling well with the number of features in the map. However, the stochastic model is not globally consistent due to the unmodelled correlations between the camera pose and the feature locations. This can lead to an inability to close large loops. Moreover, the triangulation method used to obtain prior estimates of the 3D feature locations is not suitable for distant features that appear in large-scale environments. An alternative method to initialise the extended Kalman filters associated with the features from a single camera view is proposed in [176]. The idea is to replace the three-dimensional Euclidean state by a six-dimensional *inverse depth parametrisation* that involves the camera pose and the depth of the feature. As shown in [176], a large uncertainty in depth can still be represented by a Gaussian with the inverse depth parametrisation. Furthermore, the system developed in this chapter makes no use of additional sensors to obtain a better pose prediction. All of these issues are addressed in the subsequent Chapter 5, which aims at combining and extending the results of the current chapter and Chapter 3.

5

Towards Visual-Inertial SLAM

It is possible to create robust algorithms for simultaneous localisation and mapping that scale with the number of features by decoupling pose and structure estimation. The visual SLAM system described in Chapter 4 uses this and works well in mid-scale environments. As indicated in Section 4.5, the system could be made more suitable for large-scale environments with some changes. The areas to be addressed are the following:

Extended stochastic model: A globally consistent map estimate can be obtained by modelling the correlations between the camera pose and the feature locations properly.

Sensor fusion: A visual-inertial SLAM algorithm is obtained by combining the ideas developed in Chapters 3 and 4.

Undelayed initialisation of features in 3D: Distant features can be used by replacing the triangulation method developed in Section 4.2.1 with a method that allows to initialise features from a single camera view.

In FastSLAM (*cf.* Section 2.4.3.3), the camera pose is represented by a particle set, and the correlations between the camera pose and the feature locations are coded in the diversity of the particles and their associated maps. Moreover, the marginalised particle filter (*cf.* Section 2.4.2.3) provides a suitable framework to handle the fusion of inertial and camera measurements. The combination of both techniques in the MPF-SLAM algorithm (*cf.* Section 2.4.3.4) addresses the first two areas. Furthermore, by using an inverse depth parametrisation of the 3D feature locations (*cf.* Section 4.5), the last point is addressed.

The visual-inertial SLAM system that emerges from the above ideas is conceptually illustrated in Figure 5.1. Both the architecture and the workflow of the system are essentially a combination of the developments in Chapters 3 and 4. Many of the described concepts can be reused. In the system the CamIMU (*cf.* Section 2.2) provides synchronised inertial data and

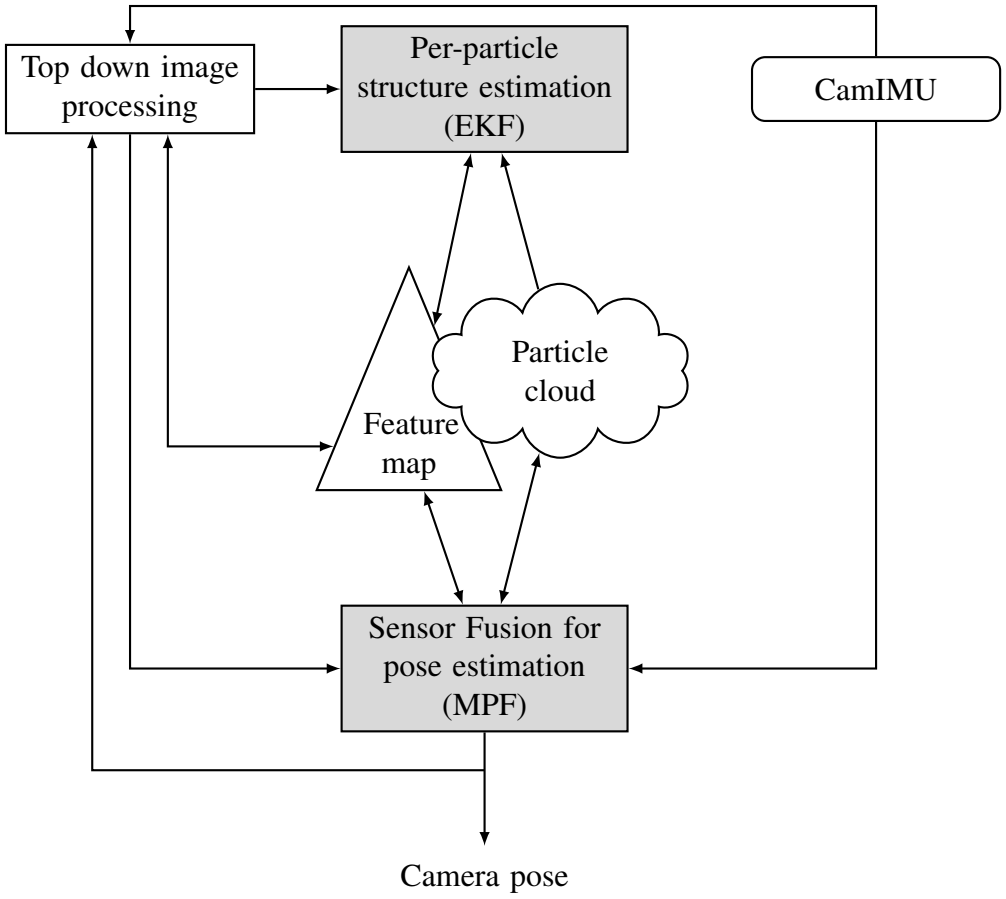


Figure 5.1: *Architecture of the visual-inertial SLAM system. The essential algorithms used in the building blocks marked in grey are treated in Section 5.1 and 5.2, respectively. In order to obtain an efficient algorithm with respect to both storage and computational time, the feature map is organised as a tree, where paths are shared between particles [28, 141].*

Table 5.1: *Particles are composed of a partitioned state vector* $\mathbf{x}^T = [\mathbf{x}^{nT}, \hat{\mathbf{x}}^{lT}]$ *that contains information related to the IMU,* i.e. *its pose, kinematics and biases, and a set of Gaussian estimates* $\mathcal{N}(\hat{\mathbf{m}}_w, P_{ww})$ *of the individual 3D feature locations* $\mathbf{m}_w$*. The posterior probability distribution at time* t *is shown.*

	IMU	**feature 1**	$\cdots$	**feature j**	$\cdots$	**feature m**
particle 1	$(\mathbf{x}^n, \hat{\mathbf{x}}^l, P)_{t\|t}^{[1]}$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[1],(1)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[1],(j)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[1],(m)}$
			$\vdots$			
particle i	$(\mathbf{x}^n, \hat{\mathbf{x}}^l, P)_{t\|t}^{[i]}$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[i],(1)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[i],(j)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[i],(m)}$
			$\vdots$			
particle n	$(\mathbf{x}^n, \hat{\mathbf{x}}^l, P)_{t\|t}^{[n]}$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[n],(1)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[n],(j)}$	$\cdots$	$(\hat{\mathbf{m}}_w, P_{ww})_{t\|t}^{[n],(m)}$

camera images. The inertial measurements are collected between the images and then processed by the marginalised particle filter. A pose prediction can be obtained from the particle set. It is exploited to generate correspondences between the image and features in the map. The registered feature locations are then used to refine the pose in the marginalised particle filter, *i.e.* to reweight the particles, and, in return, to improve the respective 3D locations in each particle.

The concepts of top down image processing, feature management and feature quality monitoring developed in Chapter 3 and 4, respectively, can be reused in the framework described here. However, as the 3D feature locations are estimated separately in each particle, the structure of the feature map is more complex. Each particle associates its own 3D feature map, represented by an extended Kalman filter for each individual feature. The resulting particle structure is illustrated in Table 5.1, where the notation used subsequently is introduced as well.

This chapter treats the major building blocks marked in grey in Figure 5.1. Section 5.1 investigates robust fusion of inertial and camera measurements in the marginalised particle filter in real-time. The developed algorithm is evaluated in a simple experimental setup, where the camera measurements are provided by a marker based tracking system (*cf.* Section 2.1.4) and the 3D locations of the marker corners are assumed to be known. Section 5.2 deals with the estimation of the 3D feature locations on a per-particle basis using the inverse depth parametrisation suggested in Section 4.5. A simple markerless test setup is then used to compare the resulting algorithm to the visual SLAM system developed in Chapter 4 and preliminary qualitative results of this comparison are presented. Section 5.3 draws final conclusions. The markerless image processing is not addressed here.

The aim of this chapter is to provide the foundation for a visual-inertial SLAM system that is suitable for large-scale environments. The essential algorithms are developed and a proof-of-concept implementation is evaluated in a simple test environment.

5.1 Real-Time Visual-Inertial Pose Estimation using MPF

The fusion of vision-based and inertial tracking technology is extensively treated in literature, especially in the field of augmented reality. The extended Kalman filter is the prominent tool for the pose estimation, similar to the design in Chapter 3. The particle filter is hardly used in this context. In [146], a sampling importance resampling (SIR) filter (*cf.* Algorithm 2.2) is used to fuse point correspondences and gyroscope measurements. The measured angular velocities are used as control inputs in the time update, while a random walk is applied to the position. A SIR filter is also used in [177] to estimate the camera pose and kinematics from point correspondences and 3D orientation provided by an IMU (*cf.* Section 2.2.1). In the recent SLAM system [56, 110], the state space is further augmented with time-varying IMU biases. As the velocities, accelerations and biases are conditionally linear and subject to Gaussian noise, the high-dimensional state vector is handled by a marginalised particle filter. The state-space model basically corresponds to the acceleration model developed in Section 3.4.1.3. However, the computational complexity of the proposed algorithm is inevitably high and can therefore not be used in a real-time application.

In contrast, the aim of this section is to develop a strategy for fusing angular velocities, linear accelerations and 2D/3D point correspondences in a marginalised particle filter, which is able to operate robustly in real-time. This is achieved in two major steps: developing an efficient state-space model based on the results of Chapter 3 and extending the marginalised particle filter as presented in Algorithm 2.4. The marginalised particle filter is extended with automatic model-switching, improved orientation proposals, adaptive process noise and mixture proposals for the translational states.

Section 5.1.1 develops a nonlinear state-space model with linear Gaussian substructure (*cf.* Section 2.4.1.3) that can be used in the marginalised particle filter, and Section 5.1.2 provides a problem-specific reformulation of the general algorithm that incorporates the extensions mentioned above. The performance of the proposed strategy is evaluated in combination with a marker based tracking system and results from a comparison to previous visual-inertial fusion strategies are presented in Section 5.1.3. Conclusions are found in Section 5.1.4.

5.1.1 State-Space Model

The choice of the state-space model is based on two results of Chapter 3:

- From the four different tracking models investigated in Chapter 3, the acceleration input model performs best with respect to both tracking performance and computational efficiency.

- The estimation of the accelerometer biases makes the tracking instable without significantly improving the overall tracking performance when corrective camera measurements are available. Good tracking results are obtained also by estimating only the gyroscope biases online.

Considering these results, the nonlinear form of the dynamic model is obtained from the acceleration input model developed in Section 3.4.1.4 by eliminating the accelerometer bias parameters $\mathbf{b}_s^a$. The state vector $\mathbf{x}$ comprises:

$$\mathbf{x}^T = \begin{bmatrix} \mathbf{s}_w^T & \dot{\mathbf{s}}_w^T & \mathbf{q}_{sw}^T & \mathbf{b}_s^{\omega T} \end{bmatrix}, \tag{5.1a}$$

where $\mathbf{s}_w$ denotes the position, $\dot{\mathbf{s}}_w$ the velocity, $\mathbf{q}_{sw}$ the orientation quaternion and $\mathbf{b}_s^\omega$ the gyroscope biases of the IMU. The inertial measurements are considered to be control inputs $\mathbf{u}_t^T = [\mathbf{y}_{s,t}^{\omega T}, \mathbf{y}_{s,t}^{aT}]$. The nonlinear process model is:

$$\begin{bmatrix} \mathbf{s}_{w,t+T} \\ \dot{\mathbf{s}}_{w,t+T} \\ \mathbf{q}_{sw,t+T} \\ \mathbf{b}_{s,t+T}^\omega \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} + \frac{T^2}{2} R_{ws,t}(\mathbf{y}_{s,t+T}^a - \mathbf{v}_{s,t}^a) + \frac{T^2}{2}\mathbf{g}_w \\ \dot{\mathbf{s}}_{w,t} + TR_{ws,t}(\mathbf{y}_{s,t+T}^a - \mathbf{v}_{s,t}^a) + T\mathbf{g}_w \\ \exp\big(-\frac{T}{2}(\mathbf{y}_{s,t}^\omega - \mathbf{b}_{s,t}^\omega - \mathbf{v}_{s,t}^\omega)\big) \odot \mathbf{q}_{sw,t} \\ \mathbf{b}_{s,t}^\omega + \mathbf{v}_{s,t}^{b^\omega} \end{bmatrix}, \tag{5.1b}$$

where $\mathbf{v}_{s,t}^a$, $\mathbf{v}_{s,t}^\omega$ and $\mathbf{v}_{s,t}^{b^\omega}$ denote time independent Gaussian process noise that is uncorrelated in all components. Using (5.1) in the marginalised particle filter, requires the partition of $\mathbf{x}$ into a nonlinear part $\mathbf{x}^n$ and a conditionally linear Gaussian part $\mathbf{x}^l$ and the separation of (5.1b) into two equations according to (2.30a) and (2.30b), respectively. The gyroscope biases $\mathbf{b}_s^\omega$ and the velocity $\dot{\mathbf{s}}_w$ are eligible for $\mathbf{x}^l$. However, the process model (5.1b) is nonlinear, implying that one separate covariance matrix is required for each particle (*cf.* Section 2.4.2.3). As pointed out in [107], in this case, the marginalised particle filter becomes less computationally efficient, if many states are marginalised. Hence, in order to obtain a real-time capable algorithm, the partition:

$$\mathbf{x}^{nT} = \begin{bmatrix} \mathbf{s}_w^T & \dot{\mathbf{s}}_w^T & \mathbf{q}_{sw}^T \end{bmatrix} \qquad \mathbf{x}^l = \mathbf{b}_s^\omega \tag{5.2a}$$

is chosen in this thesis. The reasons for this choice are given in the sequel: The gyroscope biases can be regarded as quasi-static parameters. The Kalman filter is therefore the more appropriate estimation tool, see *e.g.* [28]. Moreover, adding $\mathbf{b}_s^\omega$ to $\mathbf{x}^n$, would increase the demand for particles and random numbers per particle. In contrast, adding $\dot{\mathbf{s}}_w$ to $\mathbf{x}^n$, is not expected to have these effects due to its correlation with $\mathbf{s}_w$. With partition (5.2a) and, in order to linearise the orientation update, approximation (B.17b) in Appendix B.3, the time update of $\mathbf{x}^n$ is:

$$\underbrace{\begin{bmatrix} \mathbf{s}_{w,t+T} \\ \dot{\mathbf{s}}_{w,t+T} \\ \mathbf{q}_{sw,t+T} \end{bmatrix}}_{\mathbf{x}_{t+T}^n} = \underbrace{\begin{bmatrix} \mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} + \frac{T^2}{2}(R_{ws,t}\mathbf{y}_{s,t+T}^a + \mathbf{g}_w) \\ \dot{\mathbf{s}}_{w,t} + T(R_{ws,t}\mathbf{y}_{s,t}^a + \mathbf{g}_w) \\ \mathbf{q}_{sw,t} - \frac{T}{2}\,\mathrm{S}^{\mathrm{q}}(\mathbf{q}_{sw,t})\mathbf{y}_{s,t+T}^\omega \end{bmatrix}}_{f_t^n(\mathbf{x}_t^n, \mathbf{u}_{t+T}^n)} + \underbrace{\begin{bmatrix} 0_{3\times3} \\ 0_{3\times3} \\ \frac{T}{2}\,\mathrm{S}^{\mathrm{q}}(\mathbf{q}_{sw,t}) \end{bmatrix}}_{F_t^n(\mathbf{x}_t^n)} \underbrace{\mathbf{b}_{s,t}^\omega}_{\mathbf{x}_t^l} + \underbrace{\begin{bmatrix} 0_{3\times3} & -\frac{T^2}{2}R_{ws,t} \\ 0_{3\times3} & -TR_{ws,t} \\ \frac{T}{2}\,\mathrm{S}^{\mathrm{q}}(\mathbf{q}_{sw,t}) & 0_{4\times3} \end{bmatrix}}_{G_t^{vn}(\mathbf{x}_t^n)} \underbrace{\begin{bmatrix} \mathbf{v}_{s,t}^\omega \\ \mathbf{v}_{s,t}^a \end{bmatrix}}_{\mathbf{v}_t^n}, \tag{5.2b}$$

and the time update of $\mathbf{x}^l$ is:

$$\underbrace{\mathbf{b}^{\omega}_{s,t}}_{\mathbf{x}^l_{t+T}} = \underbrace{\mathrm{I}_3}_{F^l_t(\mathbf{x}^n_t)} \underbrace{\mathbf{b}^{\omega}_{s,t}}_{\mathbf{x}^l_t} + \underbrace{\mathrm{I}_3}_{G^{vl}_t(\mathbf{x}^n_t)} \underbrace{\mathbf{v}^{b^{\omega}}_{s,t}}_{\mathbf{v}^l_t}. \tag{5.2c}$$

During the particle filter time update (*cf.* Algorithm 2.4), the exact orientation update (B.15) in Appendix B.3 is used. This ensures that the sampled quaternions have unit length. The camera measurement model is derived from (3.8):

$$\begin{aligned} \mathbf{0}_2 &= \begin{bmatrix} \mathrm{I}_2 & -(\mathbf{m}_{n,t} + \mathbf{e}_{n,t}) \end{bmatrix} R_{cs}(R_{sw,t}(\mathbf{m}_{w,t} + \mathbf{e}_{w,t} - \mathbf{s}_{w,t}) - \mathbf{c}_s) \\ &\coloneqq h_t(\mathbf{x}^n_t, \mathbf{m}_{n,t}, \mathbf{m}_{w,t}, \mathbf{e}^c_t), \end{aligned} \tag{5.3a}$$

where $\mathbf{e}^c_t \coloneqq [\mathbf{e}^T_{n,t}, \mathbf{e}^T_{w,t}]^T \sim \mathcal{N}(\mathbf{0}_5, R_t)$ with:

$$R_t = \begin{bmatrix} R_{nn,t} & 0_{2\times 3} \\ 0_{3\times 2} & P_{ww,t} \end{bmatrix}, \tag{5.3b}$$

according to the notation used in Chapter 4 and Table 5.1. By interpreting the latest estimate $\hat{\mathbf{m}}_{w,t|t-T}$ of the 3D point as part of the implicit measurement, applying a first order Taylor expansion according to (2.37b) and substituting $\mathbf{y}_t \coloneqq \mathbf{0}_2$, the measurement equation (5.3a) can be put in the required form (2.30c):

$$\mathbf{y}_t = h_t(\mathbf{x}^n_t, \mathbf{m}_{n,t}, \hat{\mathbf{m}}_{w,t|t-T}) + H^e_t(\mathbf{x}^n_t)\mathbf{e}^c_t, \tag{5.4a}$$

where:

$$H^e_t(\mathbf{x}^n_t) \coloneqq \left.\frac{\partial h_t(\mathbf{x}^n_t, \mathbf{m}_{n,t}, \hat{\mathbf{m}}_{w,t|t-T}, \mathbf{e}^c_t)}{\partial \mathbf{e}^c}\right|_{\mathbf{e}^c=\mathbf{0}_5}. \tag{5.4b}$$

Treating the Gaussian estimate $\mathcal{N}(\hat{\mathbf{m}}_{w,t|t-T}, P_{ww,t|t-T})$ as a noisy measurement differs from the proper MPF-SLAM assumption, where the map is considered to be a part of the linear state partition $\mathbf{x}^l$, see *e.g.* [56, 110]. However, both interpretations result in exactly the same measurement equation (5.4a).

In order to compute the importance weights $w_t \propto w_{t-T}\, p(\mathbf{y}_t|\mathbf{x}^n_t)$ in the measurement update (step 2a) of the marginalised particle filter (*cf.* Algorithm 2.4), the likelihood $p(\mathbf{y}_t|\mathbf{x}^n_t)$ is required. This is obtained by inserting the measurement equation (5.4a) into (2.41a) and using the fact that the camera measurements are conditionally independent. Let $(\mathbf{m}_{n,t}, \hat{\mathbf{m}}_{w,t|t-T})^{(j)}$, with $j = 1, \ldots, m$, denote a set of 2D/3D correspondences at time t. The likelihood is then:

$$p(\mathbf{y}_t = \mathbf{0}_2|\mathbf{x}^n_t) = \prod_{j=1}^{m} \mathcal{N}\big(\mathbf{0}_2; h_t(\mathbf{x}^n_t, \mathbf{m}^{(j)}_{n,t}, \hat{\mathbf{m}}^{(j)}_{w,t|t-T}), H^{e,(j)}_t(\mathbf{x}^n_t) R^{(j)}_t H^{e,(j)T}_t(\mathbf{x}^n_t)\big), \tag{5.5}$$

with $R^{(j)}_t$ from (5.3b) and $H^{e,(j)}_t(\mathbf{x}^n_t)$ from (5.4b). Note that the importance weights are evaluated for each particle $[i], i = 1, \ldots, n$. The complete expression for the likelihood (5.5) is

therefore:

$$p(\mathbf{y}_t = \mathbf{0}_2 | \mathbf{x}_t^{n,[i]}) = \prod_{j=1}^{m} \mathcal{N}\big(\mathbf{0}_2; h_t(\mathbf{x}_t^{n,[i]}, \mathbf{m}_{n,t}^{(j)}, \hat{\mathbf{m}}_{w,t|t-T}^{[i],(j)}), H_t^{e,[i],(j)}(\mathbf{x}_t^{n,[i]}) R_t^{[i],(j)} H_t^{e,[i],(j)T}(\mathbf{x}_t^{n,[i]})\big), \quad (5.6)$$

where the registered feature positions $\mathbf{m}_{n,t}^{(j)}$ are shared between particles.

5.1.2 Enhanced MPF

The aim of this section is to optimise the general marginalised particle filter, which is introduced in Algorithm 2.4, for the specific estimation problem described in Section 5.1.1, so that a robust algorithm is obtained. Algorithm 5.1 is a reformulation of the general algorithm, which accounts for the system architecture and workflow outlined in Figure 5.1 and exploits the multi-rate characteristics of the sensors. Note that the order of measurement and time update is changed.

In preliminary experiments Algorithm 5.1 works fairly well on simulated data. However, when processing realistic inertial and camera measurements with few enough particles to allow for real-time performance ($n \leq 100$), there is a significant drift in the translational states causing frequent filter divergences. The observed drift indicates comparatively large errors in the orientation samples that propagate into velocity and position when the acceleration input model is used. This effect — that is not observed with the extended Kalman filter (*cf.* Section 3.6) — can be ascribed to the somewhat excessive process noise needed by the particle filter to spread out the samples sufficiently during the time update. Sections 5.1.2.1 through 5.1.2.3 present several extensions to Algorithm 5.1. These extensions aim to improve the accuracy of the orientation samples and the robustness of the filter. The motivation is given by the preliminary observations described above and the result of Chapter 3 that, with small orientation errors, the acceleration input model provides superior tracking performance.

5.1.2.1 Automatic Model-Switching and Improved Orientation Proposals

The experiments in Section 3.6.1 (*cf.* Figure 3.9) show that in the absence of significant body acceleration $\ddot{\mathbf{s}}_{w,t}$, the gravity model (*cf.* Section 3.4.1.2) can be used to improve the orientation estimate while performing as well as the acceleration input model. Since the inclinometer measurement equation (3.11) used in the gravity model contains no information about the translational states, no drift can be introduced into these parameters. Hence, the drift observed when using Algorithm 5.1 can be reduced during periods of low body acceleration by switching between the acceleration input and the gravity model based on whether body acceleration is detected or not. The detection criterion (3.22) is slightly improved here. The acceleration input model developed in Section 5.1.1 is used, if:

$$\big| \|\mathbf{y}_{s,t}^a\| - \|\mathbf{g}_w\| \big| > D^a, \quad (5.7)$$

Algorithm 5.1 Visual-inertial marginalised particle filter

Let $\left\{(\mathbf{x}^n, \hat{\mathbf{x}}^l, P)_{t|t}^{[i]}\right\}_{i=1}^n$ denote a set of weighted particles representing the posterior filter state at time t after the camera measurements have just been used. Assume that T is the sample time of the IMU and kT is the sample time of the camera. The algorithm for processing the inertial control inputs $\{\mathbf{u}_{t+rT}^{nT}\}_{r=1}^k$ up to and including the next set of camera measurements $\left\{(\mathbf{m}_{n,t+kT}, \hat{\mathbf{m}}_{w,t+kT|t})^{(j)}\right\}_{j=1}^m$ is then given by the following steps:

1. Let $r \coloneqq 1$.
2. Time update:
   (a) Particle filter time update (step 3a of the MPF): Generate n i.i.d. samples $\{\mathbf{x}_{t+rT|t}^{n,[i]}\}_{i=1}^n \sim p(\mathbf{x}_{t+rT}^n | \mathbf{x}_{t|t}^{n,[i]}, \mathbf{u}_{t+rT})$ based on (5.2b).
   (b) Kalman filter corrective measurement update (step 3b of the MPF): Compute the corrected linear states $\{\hat{\mathbf{x}}_{t+(r-1)T|t}^{l,[i],*}, P_{t+(r-1)T|t}^{[i],*}\}_{i=1}^n$ based on (5.2b).
   (c) Kalman filter time update (step 3c of the MPF): Predict the linear states $\{\hat{\mathbf{x}}_{t+rT|t}^{l,[i]}, P_{t+rT|t}^{[i]}\}_{i=1}^n$ based on (5.2c).
   (d) Let $r \coloneqq r + 1$. If $r \leq k$, repeat step 2.
3. Compute the weighted sample mean $\hat{\mathbf{x}}_{t+rT|t}^n$ according to (2.36). The estimate can be exploited for top down image processing.
4. Particle filter measurement update (step 2a of the MPF): Compute the importance weights $w_{t+rT}^{[i]} \propto w_t^{[i]}\, p(\mathbf{y}_t | \mathbf{x}_t^{n,[i]}), i = 1, \ldots, n$ and normalise. If $m \geq 1$, use (5.6), else use $p(\mathbf{y}_t | \mathbf{x}_t^{n,[i]}) = 1$.
   Note that the Kalman filter measurement update (step 2b of the MPF) is dropped, as the measurement equation (5.4a) does not depend on the linear state partition. The linear states are corrected only in step 2b.
5. Resample.

during a longer time period D^T. The thresholds $D^a = 0.4$ and $D^T = 0.4\,\mathrm{s}$ are chosen in the experiments. The aim of this section is to develop a gravity model that suits the marginalised particle filter and improves the accuracy of the orientation samples by using the inclination measurements during the time update. This technique is also utilised in *FastSLAM 2.0* [142], where the camera measurements are used to improve the proposal distribution for both orientation and position.

The following procedure is performed separately for each particle $[i], i = 1, \ldots, n$: At time t, the quaternion sample $\mathbf{q}_{sw,t|t}^{[i]}$ is transformed to the linear state partition $\hat{\mathbf{x}}_{t|t}^{l,[i]}$ using:

$$\mathbf{x}_{t|t}^{l,[i],+} \coloneqq \begin{bmatrix} \mathbf{q}_{sw,t|t}^{[i]} \\ \hat{\mathbf{b}}_{s,t|t}^{\omega,[i]} \end{bmatrix} \quad \text{and} \quad P_{t|t}^{l,[i],+} \coloneqq \begin{bmatrix} 0_{4\times4} & 0_{4\times3} \\ 0_{3\times4} & P_{\mathbf{b}^\omega\mathbf{b}^\omega,t|t}^{[i]} \end{bmatrix}, \tag{5.8}$$

with $P_{\mathbf{b}^\omega\mathbf{b}^\omega,t|t}^{[i]} = \mathrm{cov}(\hat{\mathbf{b}}_{s,t|t}^{\omega,[i]})$. The gravity model implies that a constant velocity model can be used for the nonlinear time update of the remaining translational states:

$$\begin{bmatrix} \mathbf{s}_{w,t+T} \\ \dot{\mathbf{s}}_{w,t+T} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t} + T\dot{\mathbf{s}}_{w,t} \\ \dot{\mathbf{s}}_{w,t} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} \mathbf{v}_{w,t}^{\ddot{s}}. \tag{5.9a}$$

The gyroscope reading $\mathbf{y}_{s,t+T}^{\omega}$ is then used as control input $\mathbf{u}_{t+T}^{l} = \mathbf{y}_{s,t+T}^{\omega}$ in the time update of the linear states:

$$\begin{bmatrix} \mathbf{q}_{sw,t+T} \\ \mathbf{b}_{s,t+T}^{\omega} \end{bmatrix} = \begin{bmatrix} \exp\big(-\frac{T}{2}(\mathbf{y}_{s,t+T}^{\omega} - \mathbf{b}_{s,t}^{\omega} - \mathbf{v}_{s,t}^{\omega})\big) \odot \mathbf{q}_{sw,t} \\ \mathbf{b}_{s,t}^{\omega} + \mathbf{v}_{s,t}^{b^\omega} \end{bmatrix}. \tag{5.9b}$$

The nonlinear orientation update implies that an extended Kalman filter is needed in each particle. The accelerometer reading $\mathbf{y}_{s,t+T}^{a}$ is used in the measurement update of the linear states. The measurement equation is obtained from (3.11) by eliminating the accelerometer bias parameters $\mathbf{b}_{s,t}^{a}$:

$$\mathbf{y}_{s,t}^{a} = -R_{sw,t}\mathbf{g}_w + \mathbf{e}_{s,t}^{a}. \tag{5.10}$$

At time $t + T$, a quaternion is sampled from the posterior Gaussian:

$$\mathbf{q}_{sw,t+T|t+T}^{[i]} \sim \mathcal{N}(\hat{\mathbf{x}}_{t+T|t+T}^{l,[i],+}, P_{t+T|t+T}^{l,[i],+}) \tag{5.11a}$$

and moved back to the nonlinear partition $\mathbf{x}_{t+T|t+T}^{n,[i]}$. This affects the remaining linear states $\hat{\mathbf{x}}_{t+T|t+T}^{l,[i]}$. Let:

$$\hat{\mathbf{q}} \coloneqq \hat{\mathbf{q}}_{sw,t+T|t+T}^{[i]}, \quad \mathbf{q} \coloneqq \mathbf{q}_{sw,t+T|t+T}^{[i]}, \quad \hat{\mathbf{b}} \coloneqq \hat{\mathbf{b}}_{s,t+T|t+T}^{\omega,[i]},$$

with

$$\mathrm{cov}\left(\begin{bmatrix} \hat{\mathbf{q}} \\ \hat{\mathbf{b}} \end{bmatrix}\right) = \begin{bmatrix} P_{qq} & P_{qb} \\ P_{bq} & P_{bb} \end{bmatrix}.$$

From the Gaussian conditioning operation [28] it follows that:

$$\hat{\mathbf{b}}|\mathbf{q} \sim \mathcal{N}\big(\hat{\mathbf{b}} + P_{bq}P_{qq}^{-1}(\hat{\mathbf{q}} - \mathbf{q}), P_{bb} - P_{bq}P_{qq}^{-1}P_{qb}\big). \tag{5.11b}$$

The procedure above requires modifications in the structure of Algorithm 5.1: With the additional measurement (5.10), the corrective Kalman filter measurement update is replaced by an extended Kalman filter measurement update and the importance weights $w_t^{[i]}$ of the particles have to be updated at IMU sample rate T. Let $R_{sw,t|t-T}^{[i]} = \mathrm{rot}(\hat{\mathbf{q}}_{sw,t|t-T}^{[i]})$ and $P_{qq,t|t-T}^{[i]} \coloneqq \mathrm{cov}(\hat{\mathbf{q}}_{sw,t|t-T}^{[i]})$. By applying a first order Taylor expansion to (5.10), the likelihood becomes:

$$p(\mathbf{y}_{s,t}^a|\mathbf{x}_t^{n,[i]}) = \mathcal{N}(\mathbf{y}_{s,t}^a; -R_{sw,t|t-T}^{[i]}\mathbf{g}_w, H_t^{[i]}P_{qq,t|t-T}^{[i]}H_t^{[i]T} + R_t), \tag{5.12}$$

with

$$H_t^{[i]} = \frac{\partial(-\mathrm{rot}(\mathbf{q}_{sw})\mathbf{g}_w)}{\partial \mathbf{q}_{sw}}\Big|_{\mathbf{q}_{sw}=\hat{\mathbf{q}}_{sw,t|t-T}^{[i]}} \tag{5.13}$$

and $\mathbf{e}_{s,t}^a \sim \mathcal{N}(\mathbf{0}_3, R_t)$.

5.1.2.2 Adaptive Process Noise

When using the acceleration input model, the drift observed in the translational states correlates with the body acceleration. A natural idea is therefore to use the magnitude of the predicted body acceleration $\ddot{\mathbf{s}}_{w,t|t-T}$ to adapt the acceleration process noise $\mathbf{v}_{s,t}^a$ used in the time update of the nonlinear states (5.2b). The adaption is done separately for each particle $[i]$ using:

$$\mathbf{v}_{s,t}^{a,[i]} = \mathbf{v}_{s,t}^a + \alpha\big\|\ \ddot{\mathbf{s}}_{w,t|t-T}^{[i]}\ \big\|, \tag{5.14}$$

with $\ddot{\mathbf{s}}_{w,t|t-T}^{[i]} = R_{ws,t|t-T}^{[i]}\mathbf{y}_{s,t}^a + \mathbf{g}_w$. The value $\alpha = 3$ is chosen in the experiments. If the considered application does not require a constant sample time, as is the case in this thesis, the same mechanism can be used to adapt the number of particles.

5.1.2.3 Mixture Proposal

Using the adaptive acceleration process noise improves the robustness of the filter and moderates the effects of the drift over a short period of time, but does not solve the problem. The reason is that all particles might be affected depending on the quality of their respective orientation sample. In order to avoid this in the first place, a mixture proposal is used for the translational states: When the acceleration input model is applied, half of the particles are sampled according to the constant acceleration model given in (5.2b) and the rest are sampled using the constant velocity model given in (5.9a). This ensures the existence of both particles that are not affected by a potential drift and others that are able to track high body accelerations, for instance caused by shifts in the direction of movement. The adaptive process noise mechanism is also applied to the body acceleration process noise $\mathbf{v}_{w,t}^{\ddot{s}}$, which differs from the accelerometer process noise $\mathbf{v}_{s,t}^a$. The mixture proposal affects the nonlinear time update of the translational states, while all other equations given in Section 5.1 remain the same.

5.1.2.4 Final Algorithm

The modifications of Algorithm 5.1 proposed in Sections 5.1.2.1 through 5.1.2.3 affect the processing of the inertial data. Algorithm 5.2 provides the procedure to incorporate the extensions into Algorithm 5.1. It replaces steps 2a through 2c. Note that the time subscripts in Algorithm 5.1 have to be changed accordingly, as the gravity model involves a measurement update at IMU sample rate of both the linear and the nonlinear states.

Algorithm 5.2 Enhanced marginalised particle filter

If body acceleration is detected, use the acceleration input model:

1. Particle filter time update:
   (a) Sample n new orientations based on (5.2b).
   (b) Adapt the process noise for the translational states using (5.14) in each particle $[i], i = 1, \ldots, n$.
   (c) Sample a mixture proposal for the translational states (*cf.* Section 5.1.2.3).
2. Kalman filter corrective measurement update and Kalman filter time update as in Algorithm 5.1.

Else, use the gravity model. For each particle $[i], i = 1, \ldots, n$:

1. Transform the quaternion to the linear partition using (5.8).
2. Particle filter and extended Kalman filter time update based on (5.9a) and (5.9b), respectively.
3. Particle filter and extended Kalman filter measurement update based on (5.12) and (5.10), respectively.
4. Transform the quaternion to the nonlinear partition using (5.11).

5.1.3 Experimental Setup and Results

In this section, the developed algorithm for visual-inertial pose estimation in the marginalised particle filter is evaluated in a simple experimental setup. The CamIMU is used to obtain a synchronised stream of camera images and inertial measurements. The camera measurements and a reference trajectory are provided by a marker based tracking system. One rectangular marker — as illustrated in Figure 2.6(b) — is detected in the camera images. As the 3D locations of the corners are assumed to be known, four 2D/3D point correspondences are obtained. The camera poses provided by the marker tracking system, the *marker poses*, are used

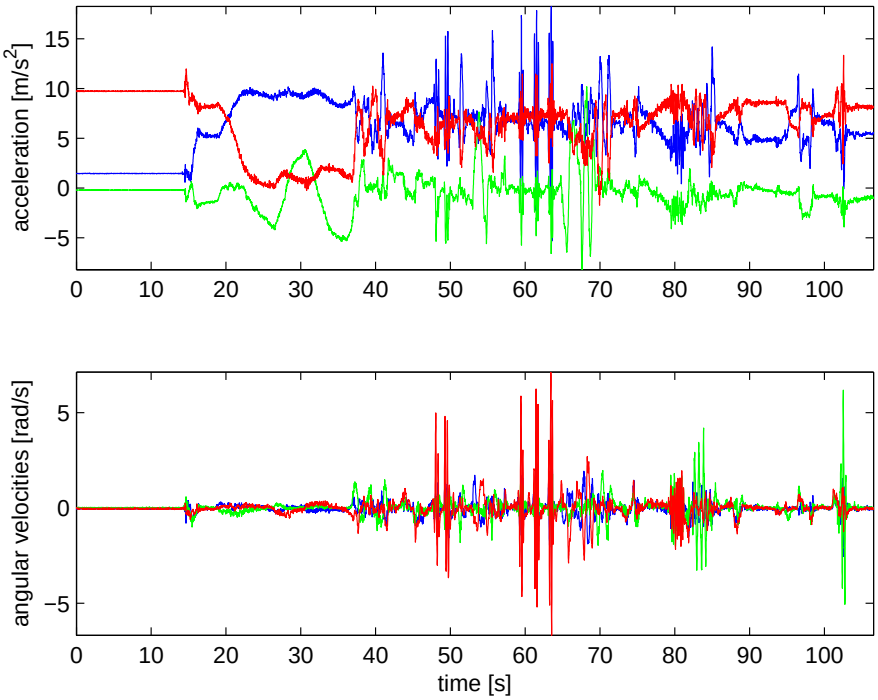


Figure 5.2: *Acceleration and angular velocity signals of the test sequence in order to give an idea of the range of the performed movements.*

as reference for the results presented subsequently. The marker poses are obtained from the point correspondences by using standard computer vision techniques (*cf.* Section 2.3.1.1). The marker poses are also used for initialising the filter and for reinitialising it after a divergence. The filter is said to diverge, if all particles have a negligible importance weight:

$$w_t^{[i]} < D^w \qquad \forall i \in [1 \dots N], \tag{5.15}$$

with $w_t^{[i]}$ from step 4 of Algorithm 5.1 before the normalisation and $D^w = 1 \cdot 10^{-16}$ in the experiments. Further divergence detection methods can be found, for instance, in [28].

Including image processing and visualisation, the developed algorithm operates smoothly at $25\,\text{Hz}$ using $n \leq 100$ particles and images with resolution 320×240 pixels. This has been measured on a $2.2\,\text{GHz}$ laptop. For an in-depth evaluation a challenging data sequence (synchronised images and IMU data) has been captured with some quick shifts in the direction of movement and erratic motions, which are characteristic for a handheld camera (*cf.* Figure 5.2). Based on this test sequence the precision, robustness and efficiency of the proposed pose estimation method, subsequently referred to as *MPF*, is compared to the following previous visual-inertial fusion strategies introduced in Section 5.1:

EKF: This denotes the acceleration input strategy developed in Chapter 3

Table 5.2: *System parameters and standard deviation noises — assuming equal noise in all dimensions — used during the experiments. As the marker corners are known by ground truth, the 3D feature locations are assumed certain. Note that the settings are tuned experimentally based on the test sequences and that the quantitative results presented in this section depend on the settings chosen.*

	MPF	**EKF**	**PF**	**Complex MPF**
process noise				
$\mathbf{v}_{s,t}^{\omega}$	0.02	0.02	0.02	0.1
$\mathbf{v}_{s,t}^{a}$	0.5	0.5	–	–
$\mathbf{v}_{w,t}^{\dot{s}}$	0.5	–	2.4	5.0
$\mathbf{v}_{s,t}^{b^{\omega}}$	$5 \cdot 10^{-4}$	$5 \cdot 10^{-4}$	–	$5 \cdot 10^{-4}$
$\mathbf{v}_{s,t}^{b^{a}}$	–	–	–	$1 \cdot 10^{-3}$
measurement noise				
$\mathbf{e}_{s,t}^{\omega}$	–	–	–	0.02
$\mathbf{e}_{s,t}^{a}$	0.1	–	–	0.4
$\mathbf{e}_{n,t}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$	$7 \cdot 10^{-3}$
system parameters				
N	100	–	300	100
D^T (*cf.* Section 5.1.2.1)	0.4	–	–	–
D^a (*cf.* (5.7))	0.4	–	–	–
D^w (*cf.* (5.15))	$1 \cdot 10^{-15}$	–	$1 \cdot 10^{-15}$	$1 \cdot 10^{-15}$
α (*cf.* (5.14))	3	–	–	–

PF: This denotes the fusion of gyroscope data and point correspondences in the SIR filter as proposed in [146]. The angular velocities are used as control inputs in the time update, where a random walk is applied to the position.

Complex MPF: This method denotes the pose estimation part of the SLAM system proposed in [56, 110]. Here, the acceleration model is used and the accelerometer biases are estimated online. Moreover, the general marginalised particle filter as presented in Algorithm 2.4 is used without exploiting the multi-rate characteristics of the camera and the IMU as in Algorithm 5.1.

The system parameters and the noise settings used during the experiments are summarised in Table 5.2. The results are presented in Sections 5.1.3.1 through 5.1.3.3.

5.1.3.1 Comparison to EKF

In this section, the computational performance and precision of the proposed method (MPF) and the extended Kalman filter (EKF) are compared. Moreover, the ability of the proposed method to estimate the gyroscope biases is investigated.

Table 5.3: *Average camera localisation errors obtained from the MPF, the EKF and the complex MPF. The orientation error is given in Euler angles.*

	EKF		complex MPF		MPF
position error [cm]					
x	0.26	$<$	0.46	$>$	0.38
y	0.26	$<$	0.47	$<$	0.80
z	0.27	$<$	0.52	$<$	0.69
orientation error [°]					
x	0.57	$<$	0.72	$<$	1.43
y	0.45	$<$	0.59	$<$	0.91
z	0.33	$<$	0.51	$<$	0.95

To give an idea of the computational complexity of both filters, the average computing time needed for processing the data of one frame — *i.e.* four inertial data at $25\,\text{Hz}$ camera frame rate and four 2D/3D correspondences — is $0.51\,\text{ms}$ using the EKF and $15.18\,\text{ms}$ with the MPF. Note that the current MPF implementation is not optimised. This can be done, for instance, by parallelisation [178, 179]. Table 5.3 compares the average camera localisation errors obtained from the EKF and the MPF (columns one and three) when considering the marker poses as ground truth. Better estimates are provided by the EKF. Note, however, that the reference trajectory obtained from the marker tracking system is also subject to an unknown error. Moreover, the EKF provides a smoother trajectory while the MPF tends to introduce jitter.

The ability of the proposed method to estimate the gyroscope biases is investigated on a data sequence with repeated movements and some stationary parts. The results are presented in Figure 5.3. The bias estimates obtained from the MPF method converge properly to approximately the same values as those obtained from the EKF even though the convergence is slower. Without the extensions described in Section 5.1.2.1, the convergence is significantly worse. The two major conclusions to draw from this figure are: First, the MPF method is able to estimate the gyroscope biases correctly, and second, the automatic model-switching and improved orientation proposals have a positive effect on the estimation of the gyroscope bias parameters. This again improves the quality of the orientation samples and as a result the quality of the position samples, when the acceleration input model is used. The reason for the improved convergence is the accelerometer measurement update in step 3 of Algorithm 5.2. The accelerometer measurements provide information about the orientation, which due to the correlations in the covariance matrix contributes to the estimation of the gyroscope biases. If the acceleration model is always used, the biases are estimated only during the corrective Kalman filter measurement update in step 2b of Algorithm 5.1.

The results presented in this section show the superiority of the EKF over the MPF, if real-time pose estimation relative to a known map of the environment is considered. However, the motivation for using the MPF is to obtain a robust solution for simultaneous localisation

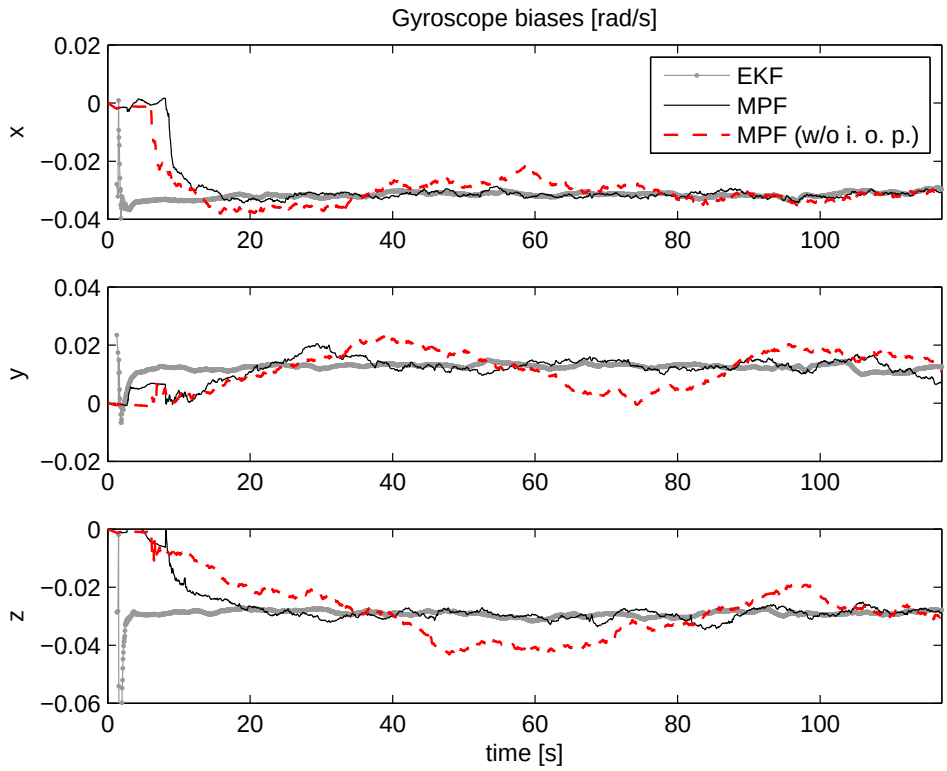


Figure 5.3: *Gyroscope bias estimates obtained from the EKF, the MPF and the MPF without automatic model-switching and improved orientation proposals (MPF w/o i.o.p.).*

and mapping that scales with the number of features, which is not provided by the extended Kalman filter (*cf.* Section 2.4.3.2).

5.1.3.2 Comparison to PF

In this section, the efficiency, robustness and precision of the proposed method is compared to the method based on the standard particle filter (PF), where only gyroscopes are used [146]. Especially, the benefits obtained by using accelerometer measurements are investigated.

In [146], the time update for the orientation states is effectively the same as in (5.1b), whereas nothing is said about the gyroscope biases. Hence, these are fixed to reasonable values obtained from the experiment in Section 5.1.3.1. In order to take the lower computational complexity of the particle filter method into account, $n = 300$ particles are allowed for this model. Thus, approximately the same amount of computational resources is used by both methods. With the constant position model as proposed in [146], no reasonable results can be obtained with the considered test data. The constant velocity model given in (5.9a) is therefore used in this experiment.

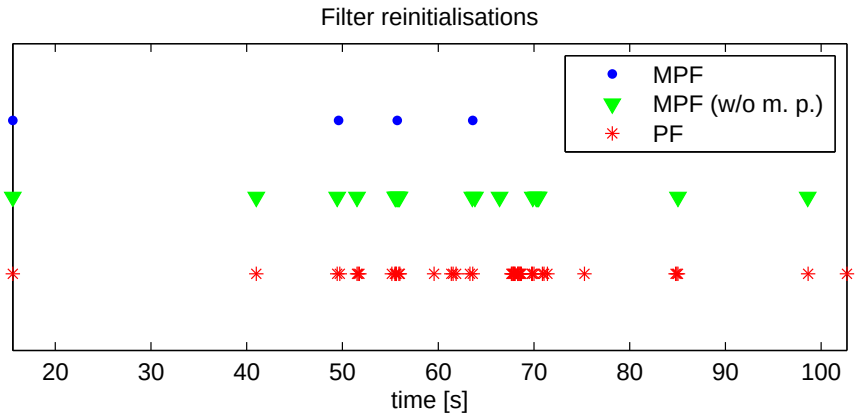
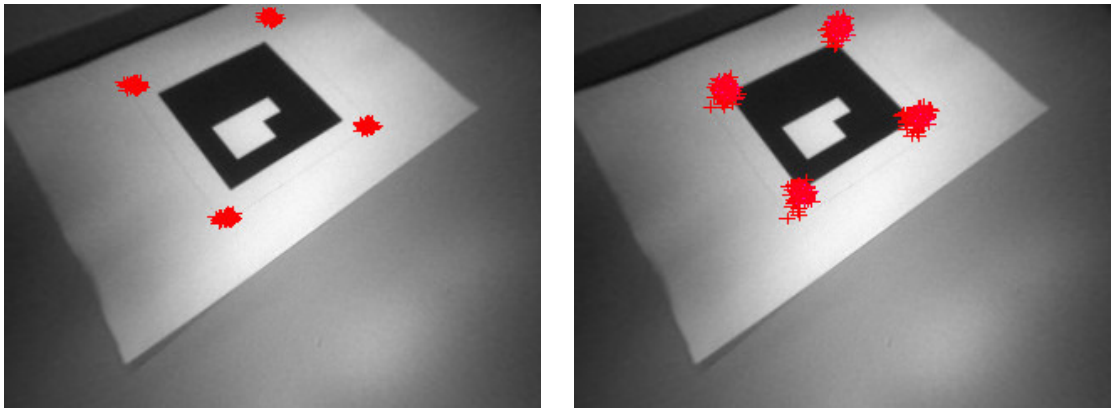


Figure 5.4: *Filter reinitialisations required by the PF, the MPF and the MPF without the mixture proposal extension (MPF w/o m. p.): the latter means in this context, that all particles are sampled on base of the constant velocity model given in* (5.9a), i.e. *that the accelerometers are only used for adapting the process noise but not for sampling the translational states.*

With the proposed algorithm using only one third of the particles in the regular particle filter, the estimation errors obtained from both filters turn out to be comparable during successful filter operation. Concerning the orientation, this results from the fact, that both filters are initialised with the same gyroscope bias values and use the same orientation update rule, as the MPF operates mainly in the acceleration input mode on the considered test sequence. However, as presented in Figure 5.4, the particle filter method shows repeated filter divergences. The difference between the methods is most apparent after shifts in the direction of movement, where the position errors peak for the regular particle filter. In contrast, the method developed here keeps the track in most of the cases by exploiting the information given by the accelerometer measurements. In order to achieve comparable robustness with the particle filter method, the process noise and the number of particles must be increased significantly. As this system aims at real-time performance, the effects of increasing the number of particles is not investigated in detail here. In [146], a study with thousands of particles is given. The contribution of the mixture proposal extension to the robustness of the MPF also shows up in Figure 5.4 in terms of a reduced number of filter divergences. Figure 5.5 presents an exemplary frame from the test sequence, where the particle filter diverges, while the marginalised particle filter is able to keep the track.

Figure 5.6 provides information about the efficiency of the particle filter and the method developed here in terms of the effective fraction of particles $P_{\text{eff},t} := \hat{N}_{\text{eff,t}}/n$, where the effective sample size $\hat{N}_{\text{eff,t}}$ is introduced in (2.34). The quantity $P_{\text{eff},t} \in [0, 1]$ measures the fraction of particles contributing to the support of the estimated probability density function and determines the fraction of particles surviving the resampling process. It can be regarded as an indicator for the quality of the proposal distribution and, as computational power is spend



(a) *PF* ***(b)*** *MPF*

Figure 5.5: *The overlaid crosses denote the features projected from the pose particles. (a) shows how the PF diverges after a change in the direction of motion, as no features project into the neighbourhood of the marker corners, implying that all particles have negligible weights. (b) demonstrates how the MPF continues the tracking, as the features project into the close neighbourhood of the marker corners. Note how the feature projections are spread out in (b) due to the automatic process noise increase described in Section 5.1.2.2.*

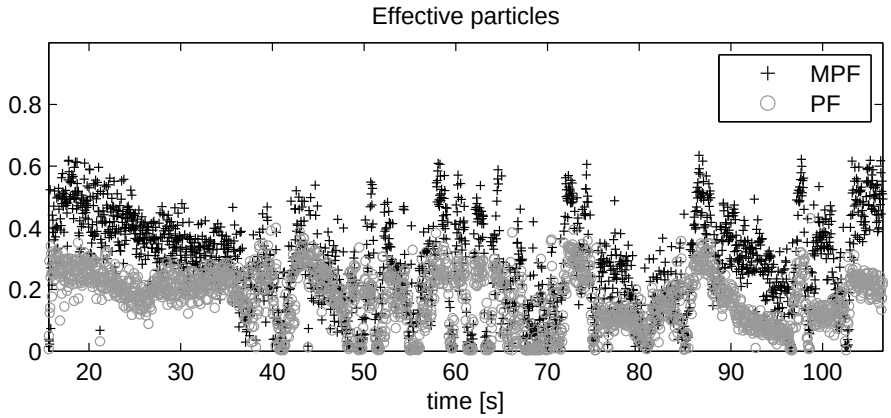


Figure 5.6: *Fraction of effective particles from the MPF and the PF.*

when propagating particles in regions of low likelihood, for the efficiency of the filter. Figure 5.6 shows higher values for the proposed method, implying that the survival time of the particles and the quality of the proposal distribution are improved compared to the particle filter method. This is of special interest in the context of FastSLAM and MPF-SLAM, as covariance information is lost with the particle trajectories (*cf.* Section 2.4.3.3).

To summarise the results presented in this section, the proposed method shows improved results with respect to robustness and filter efficiency by exploiting the accelerometer measurements in the way developed in Section 5.1.2. Moreover, the marginalised particle filter method requires less particles than the particle filter to achieve the same precision, though the computational demand per particle is higher.

The ability to detect high body accelerations in advance is of further interest in the context of simultaneous localisation and mapping, when pose and structure estimation are decoupled. In this case, the information can be used to prevent possible corruptions of the map states by prohibiting feature initialisations or map updates in case of high acceleration, where the pose samples are not likely to be well distributed. In contrast, when using the extended Kalman filter, the camera states and the map have to be treated simultaneously in the measurement update (*cf.* Section 2.4.3.2).

5.1.3.3 Comparison to Complex MPF

In this section, the computational performance, efficiency, robustness and precision of the proposed MPF and the complex MPF method are compared. The latter denotes the pose estimation part of the simultaneous localisation and mapping system described in [56, 110]. The differences between the two methods are described in the beginning of Section 5.1.3. Note that for the complex MPF the noise settings are more difficult to tune, as treating the inertial data as measurements requires two additional noise processes.

To give an impression of the computational demands of the complex MPF, the average computing time needed to process the data of one frame is observed to be 83.64 ms. In contrast, 15.18 ms are used by the method developed here, yielding a reduction of the computational costs by a factor of 5.5.

Figure 5.7 shows the estimation errors obtained from the complex MPF and the MPF on the considered test sequence. The average errors are included in Table 5.3. Figure 5.8 presents the number of filter reinitialisations required by both methods. The complex MPF performs better with respect to estimation accuracy, while the MPF method yields better stability. It requires only four filter reinitialisations, while the complex MPF shows continuous divergences appearing mainly during combined rotational and translational camera motions. This observation is similar to what is described for Algorithm 5.1 in the beginning of Section 5.1.2. Note that the results with respect to the estimation precision are affected by the reinitialisations, where the estimation errors are reset.

Figure 5.9 presents the fraction of effective particles for both methods. The complex MPF shows higher values, as the motion model used is more predictive than in the proposed method. This is due to the mixture proposal for the translational states and the adaptive process noise.

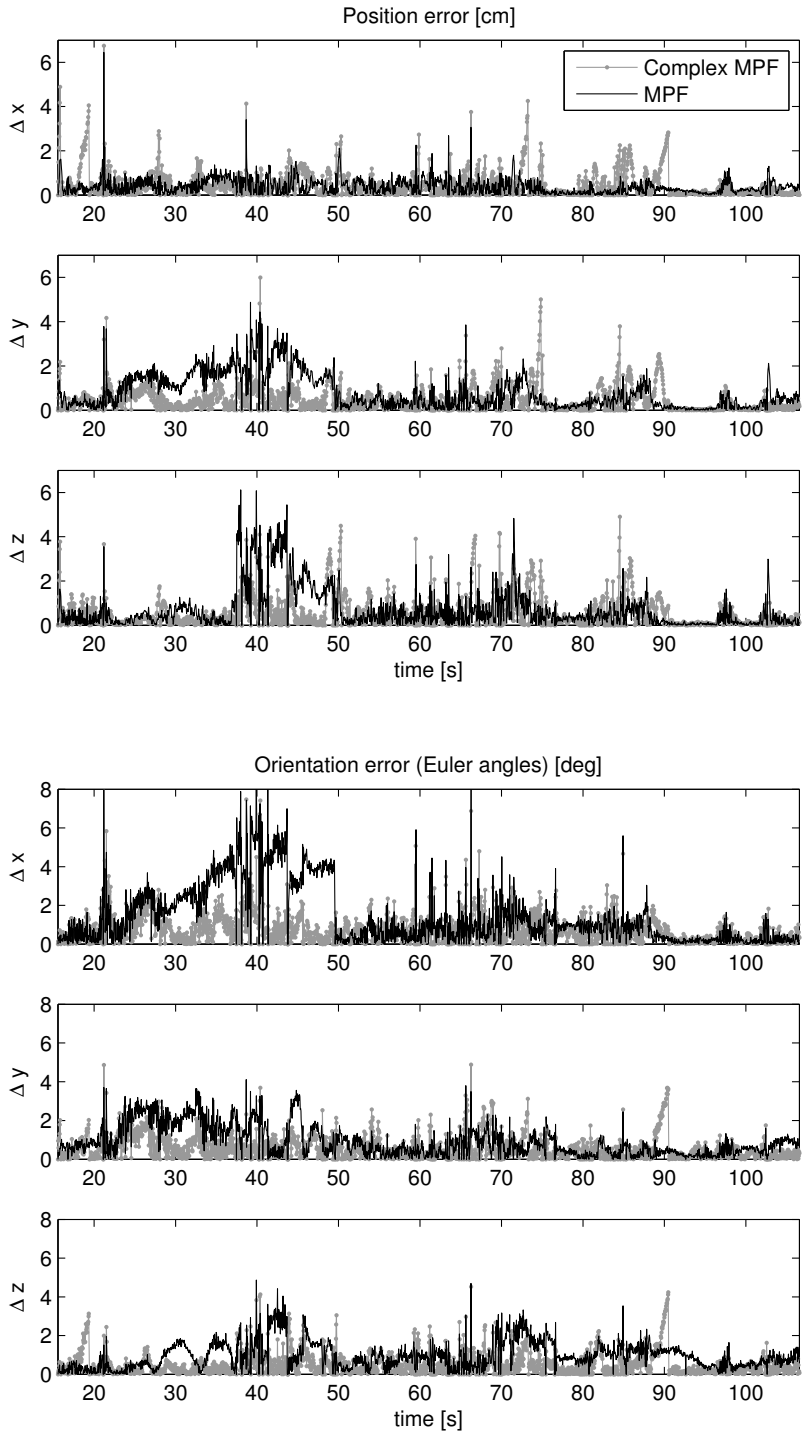


Figure 5.7: *Camera localisation errors obtained from the complex MPF and the MPF.*

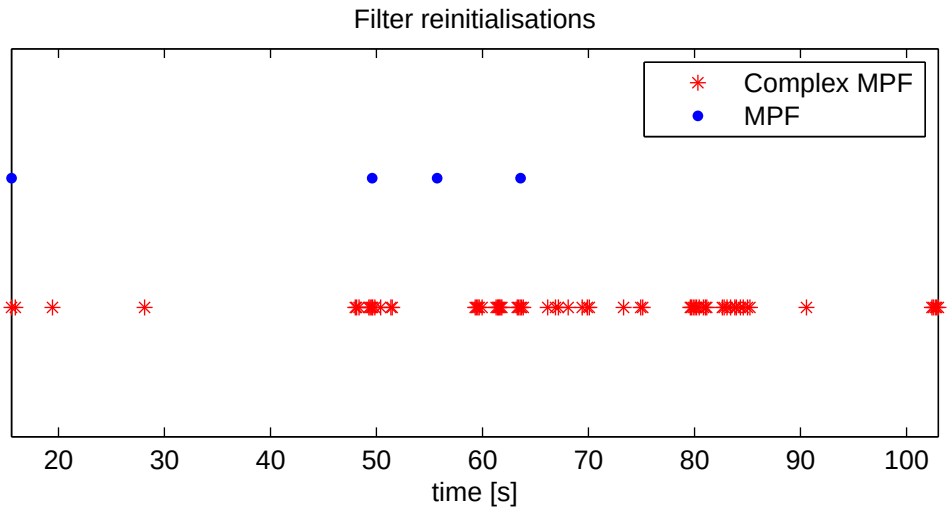


Figure 5.8: *Filter reinitialisations required by the complex MPF and the MPF.*

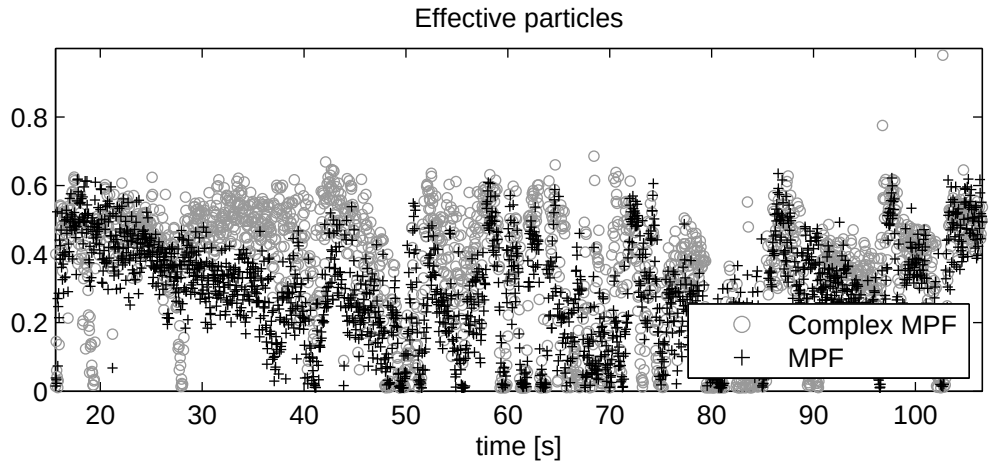


Figure 5.9: *Fraction of effective particles from the complex MPF and the MPF.*

However, there is a trade-off between the efficiency and the robustness of the filter. The parameter settings of the method developed here can be modified in order to match the requirements of the considered application, *i.e.* in order to emphasise either robustness or efficiency.

To summarise the results of this section, the MPF provides reduced computational demands and — with the parameter settings chosen in the experiments (*cf.* Table 5.2) — an improved tracking stability, while the complex MPF yields a higher precision and filter efficiency.

5.1.4 Summary

Section 5.1 presents a strategy capable of fusing vision-based and inertial tracking technologies in real-time using the marginalised particle filter. The proposed method is developed in several steps. A nonlinear state-space model with linear Gaussian substructure is developed in Section 5.1.1 for the problem of real-time visual-inertial camera pose estimation. In Section 5.1.2, the general marginalised particle filter is adapted to the specific characteristics of the considered application and improved with several extensions: automatic model-switching, improved orientation proposals, adaptive process noise and mixture proposals for the translational states. The evaluation of the final method in combination with a marker-based tracking system shows that the developed extensions increase the precision, the efficiency and most notably the robustness of the filter. More stable results can be obtained compared to previous fusion models based upon the standard particle filter and the general marginalised particle filter, especially in the presence of erratic camera motions. When considering the problem of pose estimation relative to a known map, the extended Kalman filter turns out to be superior to the fusion strategy based on the marginalised particle filter. However, in the context of simultaneous localisation and mapping, the extended Kalman filter is not the preferred solution, since it does not scale with the number of features and does not permit to decouple pose and structure estimation.

5.2 Per-Particle Feature Estimation

This section treats the estimation of the 3D feature locations on a per-particle basis. It aims at completing the localisation algorithm developed in Section 5.1 with map estimation so that a simultaneous localisation and mapping system according to the architecture illustrated in Figure 5.1 is obtained. A comparable SLAM system is proposed in [56, 110], however, using an Euclidean representation for the features.

In Algorithm 5.1, map estimation and management are incorporated after the measurement update in step 4. The workflow is identical to that of the visual SLAM system developed in Chapter 4 (*cf.* Figure 4.2): Structure is estimated based on the estimated camera pose, followed by the optional extraction of new features in free areas of the image. Indeed, the robust triangulation and refinement method developed in Section 4.2.1 and 4.2.2, respectively, can be reused here. This can be done by performing the estimation step separately in each particle using the camera pose it represents. However, on a per-particle basis, the robust triangulation

algorithm is too time consuming. Moreover, as pointed out in Section 4.5, this method is not suitable for distant features, as the 3D initialisation must be delayed until a sufficiently large baseline and, hence, an acceptable initial depth estimate is available. The reason is that the extended Kalman filter requires a Gaussian representation with small covariance for the involved random variables in order to assure a small linearisation error. Methods for undelayed feature initialisation in bearing-only tracking proposed in literature are: *Federated Information Sharing (FIS)* [180] and inverse depth parametrisation [117, 176].

Federated information sharing is a method where for each new feature a set of location hypotheses is included in the map, and less likely members are pruned over time. It is an approximation to the Gaussian sum filter [181] with additive growth in the problem size.

The inverse depth method denotes a change from the minimal Euclidean feature representation to a six-dimensional representation that involves the camera pose and the depth of the feature. In this thesis, the inverse depth approach is used. It is easy to incorporate into the structure of the developed algorithms. It allows features to be used from the time of their initial observation. Moreover, even infinitely distant features can be used for the pose estimation, *i.e.* a separate 2D/2D measurement equation as in Chapter 4 is not needed. Since the camera pose is given in each particle, the six-dimensional estimation problem reduces to a three-dimensional problem when considered on a per-particle basis.

Section 5.2.1 develops the inverse depth measurement model and the modifications required to incorporate this model into Algorithm 5.1. The resulting method is then compared to the visual SLAM system developed in Chapter 4 in the simple test setup of Section 5.1.3. Preliminary qualitative results of this comparison are presented in Section 5.2.2 and conclusions are found in Section 5.2.3.

5.2.1 Inverse Depth Measurement Model

In the inverse depth model, 3D points are parametrised according to the definition of the collinearity constraint (*cf.* Section 2.1.3.1). When a feature is first observed, a ray can be drawn from the optical centre $\mathbf{c}_{w,0}$ of the camera through the position $\mathbf{m}_n$ of the feature in the image plane. The 3D location $\mathbf{m}_w$ of the feature then lies on this ray at an unknown depth d. Figure 5.10 illustrates this relation. The directional vector $\mathbf{p}_w = m(\theta_w, \phi_w)$ is parametrised by two angles, azimuth θ_w and elevation ϕ_w, expressed in the world frame w. The transformation of these angles to Cartesian coordinates is:

$$\mathbf{p} = m\left(\begin{bmatrix}\theta\\ \phi\end{bmatrix}\right) = \begin{bmatrix}\cos(\phi)\sin(\theta)\\ -\sin(\phi)\\ \cos(\phi)\cos(\theta)\end{bmatrix}, \tag{5.16a}$$

where $\mathbf{p}$ has unit length. The inverse transformation is:

$$\begin{bmatrix}\theta\\ \phi\end{bmatrix} = m^{-1}\left(\begin{bmatrix}\mathbf{p}_x\\ \mathbf{p}_y\\ \mathbf{p}_z\end{bmatrix}\right) = \begin{bmatrix}\arctan(\mathbf{p}_x, \mathbf{p}_z)\\ \arctan(-\mathbf{p}_y, \sqrt{\mathbf{p}_x^2 + \mathbf{p}_z^2})\end{bmatrix}. \tag{5.16b}$$

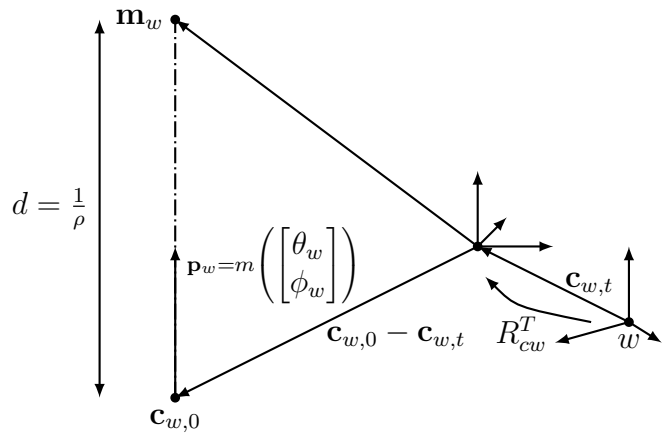


Figure 5.10: *Illustration of the inverse depth parametrisation: The directional vector $\mathbf{p}_w$, expressed in the world frame w is given in spherical coordinates θ_w and ϕ_w.*

The advantage of the non-minimal representation is that the depth d of the feature along the ray is coded in a separate parameter. As an infinite interval in depth corresponds to a finite interval in inverse depth $\rho = \frac{1}{d}$, features can be initialised in 3D from one camera view by estimating ρ instead of d. Consider the simple arithmetic example: If the prior inverse depth estimate is chosen to be $\hat{\rho}_{0|-1} = 0.5$ with variance $\sigma_{\rho,0|-1} = \mathrm{var}(\hat{\rho}_{0|-1}) = 0.25$, the 95% acceptance region covers the inverse depth range $\rho \in [0, 1]$. This corresponds to the depth range $d \in [1, \infty]$ including even features infinitely far away.

With the inverse depth parametrisation, the 3D location of a feature is represented by a six-dimensional state vector:

$$\mathbf{x}^f = \begin{bmatrix}\mathbf{c}_{w,0} & \theta_w & \phi_w & \rho\end{bmatrix}, \tag{5.17}$$

where $\mathbf{c}_{w,0}$ denotes the optical centre of the first camera view, θ_w and ϕ_w denote the azimuth and respectively the elevation angle parametrising the directional vector defined by the first camera view and ρ denotes the inverse depth along the ray defined by the optical centre and the directional vector. The transformation of $\mathbf{x}^f$ to the Euclidean representation $\mathbf{m}_w$ is:

$$\mathbf{m}_w = \mathbf{c}_{w,0} + \frac{1}{\rho} m\left(\begin{bmatrix}\theta_w \\ \phi_w\end{bmatrix}\right), \tag{5.18}$$

where $m([\theta_w, \phi_w]^T)$ is given in (5.16a).

In order to incorporate the inverse depth parametrisation into Algorithm 5.1, three models must be derived: The 3D feature locations are estimated, each in a separate extended Kalman filter. The initialisation of the extended Kalman filter from one camera view is addressed in Section 5.2.1.1. The measurement model required to perform the filter updates is provided in Section 5.2.1.2. Finally, the measurement likelihood required to use the features for pose estimation in the marginalised particle filter is developed in Section 5.2.1.3.

5.2.1.1 Feature Initialisation

Features are estimated on a per-particle basis. The initialisation procedure is repeated in each particle $[i], i = 1, \ldots, n$ using the registered feature position $\mathbf{m}_{n,t}$ and the respective pose sample. Note that the position $\mathbf{s}_{w,t}$ and orientation $\mathbf{q}_{sw,t}$ of the IMU are estimated in the marginalised particle filter. The camera position $\mathbf{c}_{w,t}$ and orientation $\mathbf{q}_{cw,t}$ are obtained from (2.12a) using the hand-eye transformation. The prior state parameters $\mathbf{c}_{w,0}$ are immediately given by the camera position, *i.e.*:

$$\hat{\mathbf{c}}_{w,0,0|-1} = \mathbf{c}_{w,t}. \tag{5.19a}$$

The angles θ_w and ϕ_w are obtained from the camera orientation and the registered feature position $[x, y]^T \coloneqq \mathbf{m}_{n,t}$ as:

$$\begin{bmatrix} \hat{\theta}_w \\ \hat{\phi}_w \end{bmatrix}_{0|-1} = f(\mathbf{m}_{n,t}, \mathbf{q}_{cw,t}) = m^{-1} \left(\operatorname{rot}(\mathbf{q}_{cw,t})^T \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \right). \tag{5.19b}$$

The initial value for ρ and its variance are design parameters. In the experiments, they are chosen as in the arithmetic example above. The values can be derived heuristically to cover a certain working space in the acceptance region. Let $R_{nn,t}$ denote the covariance of the measurement noise in the registered feature position and $\sigma_{\rho,0|-1}$ the initial variance of ρ, the prior covariance $P_{\mathbf{x}^f,0|-1} \coloneqq \operatorname{cov}(\mathbf{x}^f_{0|-1})$ is then:

$$P_{\mathbf{x}^f,0|-1} = \begin{bmatrix} 0_{3\times3} & 0_{3\times2} & 0 \\ 0_{2\times3} & JR_{nn,t}J^T & 0 \\ 0_{1\times3} & 0_{1\times2} & \sigma^2_{\rho,0|-1} \end{bmatrix}, \tag{5.19c}$$

with Jacobian:

$$J \coloneqq \left. \frac{\partial f(\mathbf{m}_n, \mathbf{q}_{cw,t})}{\partial \mathbf{m}_n} \right|_{\mathbf{m}_n = \mathbf{m}_{n,t}}.$$

Note that all rows and columns related to $\mathbf{c}_{w,0}$ are zero, as the camera pose is given by the particle. Hence, these parameters do not change during the filter update implying that they can be removed from the state vector $\mathbf{x}^f$ and treated as deterministic input to the observation model instead. This reduces the estimation problem to three dimensions and results in substantial savings in the computational cost.

5.2.1.2 Feature Estimation

Analogously to the method developed in Section 4.2.2, standard extended Kalman filter measurement updates are performed to estimate the 3D feature locations in each particle. In contrast to Section 4.2.2, the measurement equation is here in the form of (2.26b), *i.e.* it is formulated explicitly with additive measurement noise. This is a result of the fact that the pose is

given by the particle and can therefore be considered a deterministic input to the observation model rather than a noisy measurement. Let $\mathbf{y}_{n,t} = \mathbf{m}_{n,t}$ denote the measured image position of a feature with measurement noise $\mathbf{e}_{n,t} \sim \mathcal{N}(\mathbf{0}_2, R_{nn,t})$. The measurement equation then is:

$$\begin{aligned}\mathbf{y}_{n,t} &= h(\mathbf{x}_t^f, \mathbf{s}_{w,t}, \mathbf{q}_{sw,t}) + \mathbf{e}_{n,t} \\ &= \mathcal{P}_n\Big(R_{cs}\Big(R_{sw}\big(\rho(\mathbf{c}_{w,0} - \mathbf{s}_w) + m([\theta_w^T, \phi_w^T]^T)\big) - \rho\,\mathbf{c}_s\Big)\Big) + \mathbf{e}_{n,t},\end{aligned} \tag{5.20}$$

where R_{cs} and $\mathbf{c}_s$ denote the hand-eye transformation (*cf.* Section 2.2.2) and $\mathcal{P}_n$ applies the normalised pinhole projection (2.5a). The linearised model is obtained according to (2.37b) with $H_t^e := \mathrm{I}_2$ and $\mathbf{y}_{n,t}$ solved for explicitly. Moreover, the EKF measurement update is performed according to step 2 of Algorithm 2.3 with $\mathbf{z}_t = \mathbf{y}_{n,t} - h(\hat{\mathbf{x}}_{t|t-T}^f, \mathbf{s}_{w,t}, \mathbf{q}_{sw,t})$.

5.2.1.3 Likelihood

In order to use the 3D feature locations $\mathbf{x}^f$ for the pose estimation in the marginalised particle filter, the likelihood (5.5) needs to be adapted to the measurement model (5.20). By applying a first order Taylor expansion around $\hat{\mathbf{x}}_{t|t-T}^f$, equation (5.20) can be put in the required form (2.30c):

$$\mathbf{y}_{n,t} = h_t(\mathbf{x}_t^f, \mathbf{s}_{w,t}, \mathbf{q}_{sw,t}) + \mathbf{e}_{n,t} \tag{5.21a}$$

$$\approx \underbrace{h_t(\hat{\mathbf{x}}_{t|t-T}^f, \mathbf{x}_t^n) - H_t(\mathbf{x}_t^n)\hat{\mathbf{x}}_{t|t-T}^f}_{h_t(\mathbf{x}_t^n)} + H_t(\mathbf{x}_t^n)\mathbf{x}_t^f + \mathbf{e}_{n,t}, \tag{5.21b}$$

with:

$$H_t(\mathbf{x}_t^n) \coloneqq \frac{\partial h_t(\mathbf{x}^f, \mathbf{x}_t^n)}{\partial \mathbf{x}^f}\Big|_{\mathbf{x}^f = \hat{\mathbf{x}}_{t|t-T}^f}. \tag{5.21c}$$

Note that $\mathbf{x}_t^f := \mathbf{x}_t^l$ is here considered to be the linear state partition. The linearised model (5.21b) can now be inserted into (2.41a). Assume that a set $\{\mathbf{y}_{n,t}^{(j)}\}_{j=1}^m$ of registered feature positions, each with measurement noise $\mathbf{e}_{n,t} \sim \mathcal{N}(\mathbf{0}_2, R_{nn,t})$ is available at time t. The likelihood is then:

$$p(\{\mathbf{y}_{n,t}^{(j)}\}_{j=1}^m | \mathbf{x}_t^n) = \prod_{j=1}^m \mathcal{N}\big(\mathbf{y}_{n,t}^{(j)}; h_t(\hat{\mathbf{x}}_{t|t-T}^f, \mathbf{x}_t^n), H_t^{(j)}(\mathbf{x}_t^n) P_{\mathbf{x}^f,t|t-T} H_t^{(j)T}(\mathbf{x}_t^n) + R_{nn,t}\big). \tag{5.22}$$

Note that (5.22) is evaluated for each particle $[i], i = 1, \ldots, n$. The complete notation can be formulated analogously to (5.6).

5.2.2 Experimental Setup and Results

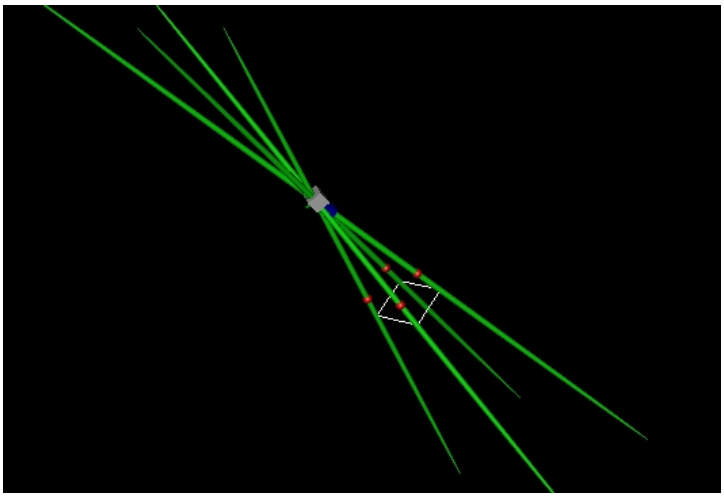
The aim of this section is to provide a proof-of-concept of the ideas presented in this chapter and to present preliminary qualitative results. The MPF-SLAM system that emerges from

incorporating the per-particle feature estimation method developed in Section 5.2 into the algorithm for visual-inertial pose estimation in the marginalised particle filter described in Section 5.1 is here compared to the visual SLAM system from Chapter 4. The focus is on the comparison of the structure estimation methods and their representations of the uncertainties in the feature locations treated in Section 5.2.2.1 and 5.2.2.2, respectively. Section 5.2.2.3 presents qualitative results and observations with respect to the camera pose estimation. A part of the test data sequence used in Section 5.1.3 — more precisely, second 20 to 50 (*cf.* Figure 5.2) — where the CamIMU orbits the marker and then performs some quick motions, is used for the evaluation. The marker corners are tracked in the images using the markerless image processing method presented in Chapter 3.

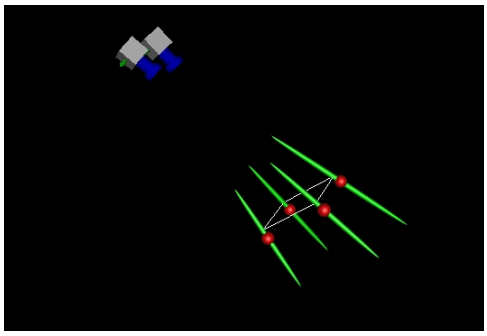
By using the reference poses obtained from the marker based tracking system to initialise the SLAM process, both the MPF-SLAM method developed in this chapter and the visual SLAM system — here denoted V-SLAM — developed in Chapter 4 are able to estimate the trajectory of the CamIMU in real-time while mapping the locations of the four marker corners. The system parameters and noise settings of the previous experiments given in Table 5.2 and 4.1, respectively, are used here. Moreover, as proposed in Section 5.1.3, the map update is prohibited in the MPF-SLAM system, when body acceleration is detected (*cf.* Section 5.1.2.1).

5.2.2.1 Feature Initialisation and Estimation

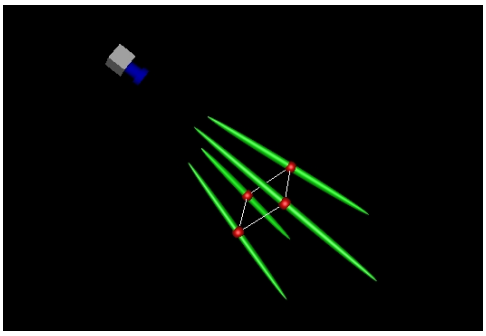
In this section, the different feature initialisation and estimation methods are compared. The inverse depth parametrisation used in the MPF-SLAM method allows for an undelayed initialisation from one camera view. In contrast, the Euclidean representation used in the V-SLAM system requires delayed triangulation from a sufficiently large baseline. Initial and refined location estimates obtained from both methods are illustrated in Figure 5.11. The according images overlaid with the projected covariances are shown in Figure 5.12. Note that the Euclidean location estimates, illustrated for the MPF-SLAM method in Figure 5.11, are obtained from the feature states (5.17) by using a first order Gauss approximation of the transformation function (5.18) (*cf.* Equation 4.2). The approximation is poor, if the uncertainty in ρ is high. However, the results are only used for visualisation purposes. The same approximation is applied to (2.5a) in order to obtain the projected covariances in terms of pixels shown in Figure 5.12. The initial location uncertainties presented in Figure 5.11(a) show the principle of the inverse depth parametrisation. The elongated ellipsoids meet in the optical centre of the camera and surround the view rays towards the marker corners. The undelayed feature initialisation comes at the expense of inaccurate initial estimates: The initial feature locations indicated by the spheres are far off the corners of the rectangle. This is due to the heuristic choice of $\hat{\rho}_{0|-1}$, since no prior knowledge about the depth parameter is available (*cf.* Section 5.2.1). However, the ground truth locations are included in the 99% confidence regions indicated by the ellipsoids and the refined estimates illustrated in Figure 5.11(b) and (d) show a correct convergence. The initial location estimates obtained from the V-SLAM system (*cf.* Figure 5.11(c)) are comparatively accurate. However, the triangulation is delayed, until a sufficiently large baseline is available. In this special case, 7 to 9 camera views are used for the estimation of the initial



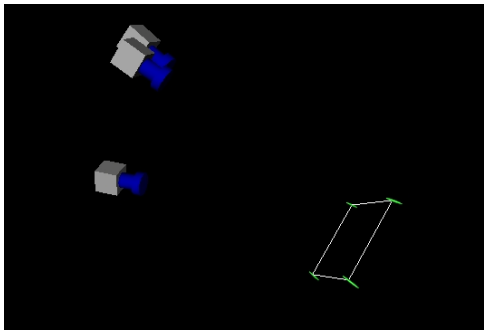
(a) *MPF-SLAM, image* 1*: initialisation.*



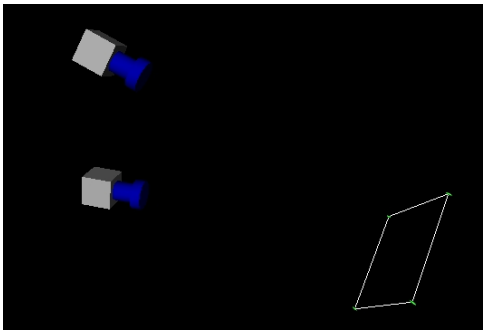
(b) *MPF-SLAM, image* 19*.*



(c) *V-SLAM, image* 19*: initialisation.*

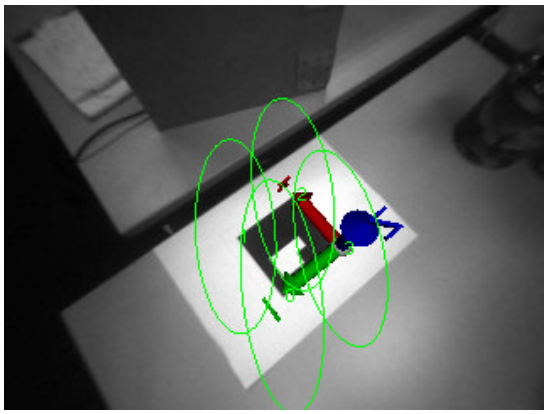


(d) *MPF-SLAM, image* 100*.*

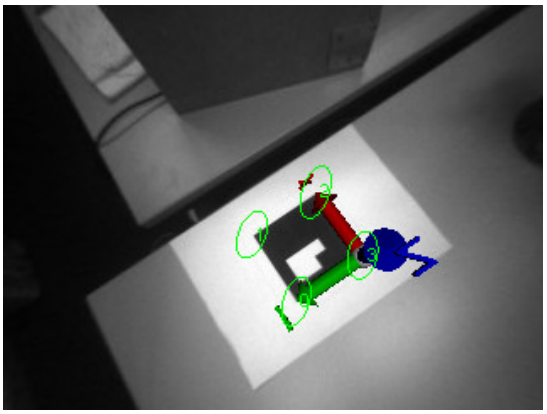


(e) *V-SLAM, image* 100*.*

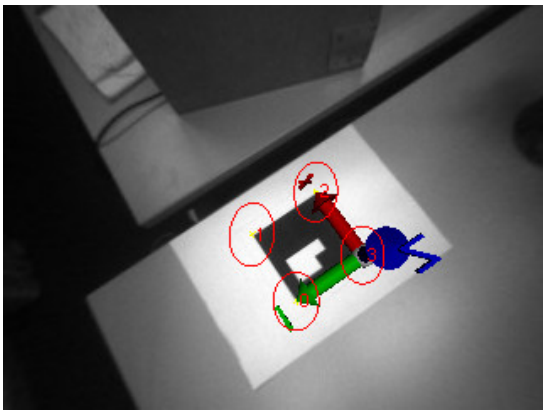
Figure 5.11: *The camera views and the mapped 3D locations of the marker corners with their location covariances obtained from both the MPF-SLAM and the V-SLAM method are shown at different images of the test data sequence. The rectangle denotes the location of the marker. See Figure 5.12 for the according camera frames.*



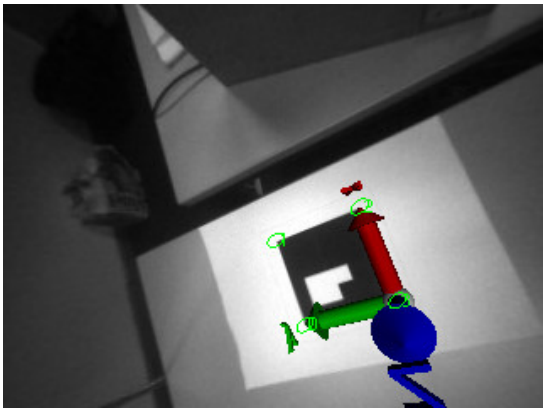
(a) MPF-SLAM: image 1.



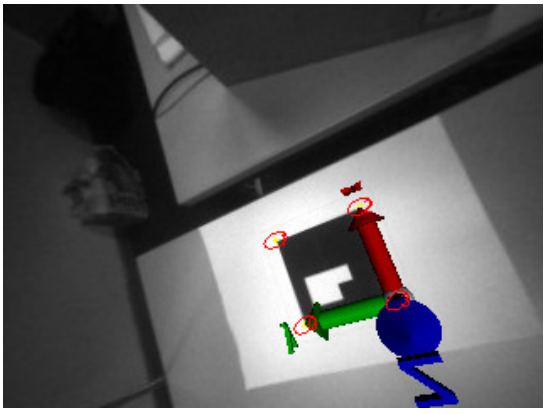
(b) MPF-SLAM: image 19.



(c) V-SLAM: image 19.

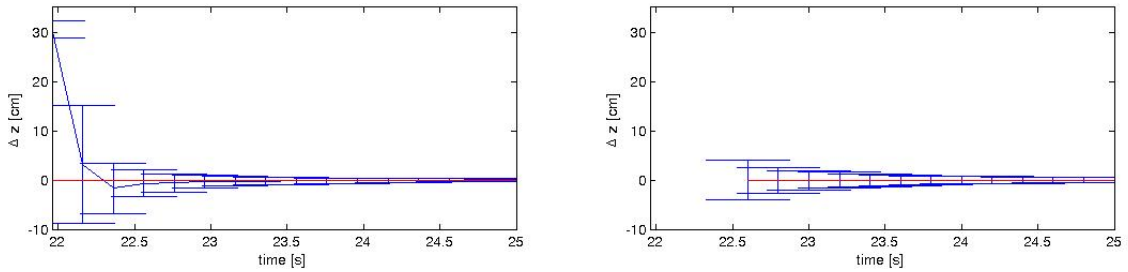


(d) MPF-SLAM: image 100.



(e) V-SLAM: image 100.

Figure 5.12: *Camera images augmented with a virtual coordinate system — the world frame — and overlaid with ellipses demonstrating the projected 3D location covariances of the mapped marker corners.*



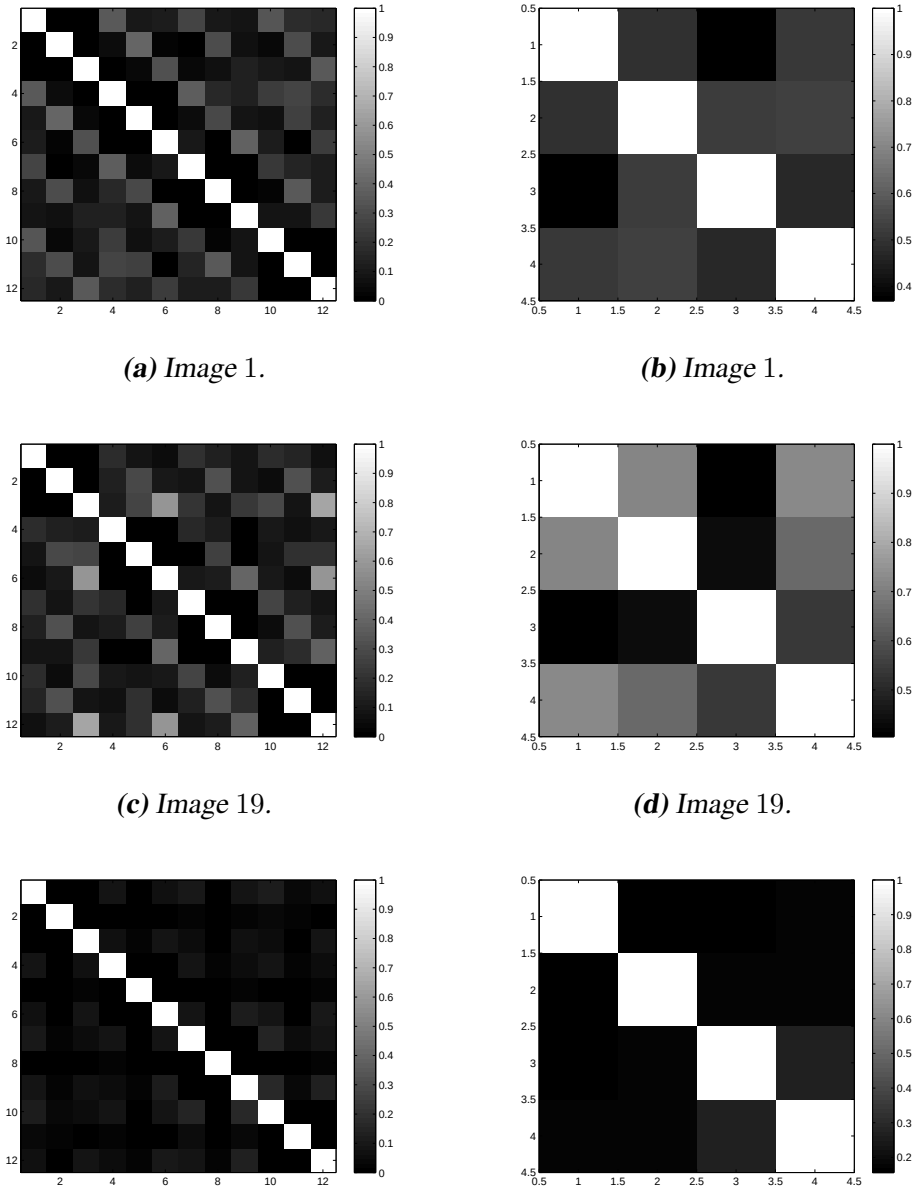
(a) *MPF-SLAM.* ***(b)*** *V-SLAM.*

Figure 5.13: *Estimation errors of one marker corner in the z-dimension and 99% confidence regions obtained from the MPF-SLAM and the V-SLAM method. Note the undelayed feature initialisation when using the MPF-SLAM method.*

weighted least squares estimate (*cf.* Section 4.2.1). Figure 5.13 shows the estimation errors of one of the marker corners in the z-dimension with their 99% confidence regions. The initial error obtained from the undelayed initialisation is comparatively large. However, the error is consistently represented by the variance and the estimates of both methods converges to the correct solution. The final root mean square mapping errors obtained from both methods are small in this simple experimental environment. The V-SLAM method provides slightly less error with $0.01\,\text{cm}$, while $0.14\,\text{cm}$ are measured for the MPF-SLAM system. By increasing the number of particles from $n = 100$ to $n = 200$, the error reduces to $0.08\,\text{cm}$. However, the average computing time needed for processing the data of one frame — including pose estimation and map update — increases from $21.37\,\text{ms}$ to $43.61\,\text{ms}$, where on average 9.14% of the processing time is allotted to the map update.

5.2.2.2 Representation of Uncertainty

As mentioned in the beginning of this chapter, the reason for using FastSLAM mapping, *i.e.* for estimating the map separately in each particle, is to obtain a consistent representation of the correlations between the camera and the features in the diversity of the particles and their associated maps. Figure 5.14 illustrates the correlations between the mapped locations of the four marker corners at the time instances given in Figure 5.11. The correlations between the location parameters $\theta_w^{(j)}$, $\phi_w^{(j)}$ and $\rho^{(j)}$, with $j = 1, \ldots, 4$, estimated in the per-particle extended Kalman filters (*cf.* Section 5.2.1.1), and the block correlations between the four features are shown. The covariance matrix is obtained from the particles and their associated maps by defining the stacked vectors $\mathbf{x}_{t|t}^{l,[i]} = [\theta_w^{(1)}, \phi_w^{(1)}, \rho^{(1)}, \ldots, \theta_w^{(4)}, \phi_w^{(4)}, \rho^{(4)}]^{[i]T}$ and using these in (2.42). The same procedure can be used to obtain the correlations between the states related to the IMU and the features. As shown in Figure 5.14(c) and (d), at an early reconstruction state, the largest cross correlations develop between the depth parameters of the features, most notably between feature 1 and 2 and 1 and 4. These are also the most uncertain parameters. At

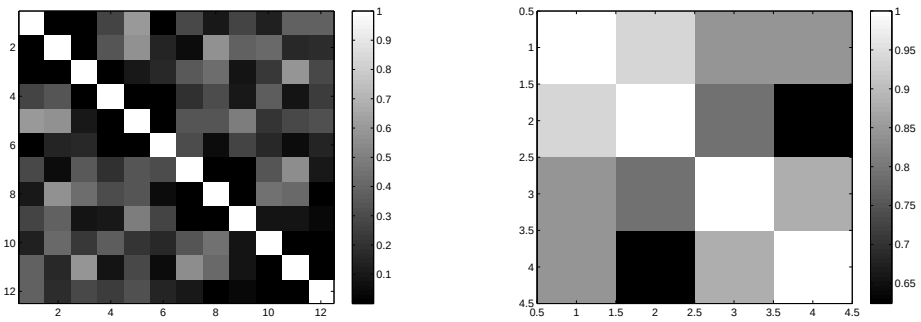


(a) *Image* 1. ***(b)*** *Image* 1.

(c) *Image* 19. ***(d)*** *Image* 19.

(e) *Image* 100. ***(f)*** *Image* 100.

Figure 5.14: *MPF-SLAM: illustration of the correlations between the four estimated feature locations with* $n = 100$ *particles: The correlations between the single location parameters* $\theta_w^{(j)}$, $\phi_w^{(j)}$ *and* $\rho^{(j)}$, *with* $j = 1, \ldots, 4$, *are shown on the left side and the block correlations between the four features — generated from the blocks with the* 2*-norm — are shown on the right side. Note that the covariance matrix is normalised with the block diagonal elements in order to show the correlations more explicitly.*



(a) *Image* 100. ***(b)*** *Image* 100.

Figure 5.15: *MPF-SLAM: illustration of the correlations between the four estimated feature locations with* $n = 1000$ *particles (*cf. *Figure 5.15).*

a later state, when the feature estimates are well defined in all directions (*cf.* Figure 5.11(d)), the off-diagonal entries become very small as illustrated in Figure 5.14(e) and (f). Although the survival time of the particles is shown to be extended compared to a regular particle filter (*cf.* Section 5.1.3.2), this indicates a particle depletion problem, which can lead to underestimated uncertainty and a limited ability to close loops (*cf.* Section 2.4.3.3). Again, this effect can be reduced by increasing the number of particles at the expense of higher computational costs. Figure 5.15 illustrates the correlations with $n = 1000$ particles.

5.2.2.3 Camera Pose Estimation

A notable observation concerning the pose estimation is that the V-SLAM system introduces jitter, while the MPF-SLAM method provides a smooth trajectory. This is discussed in the beginning of Chapter 3. Figure 5.16 compares the position estimates obtained for parts of the test data sequence, where this effect is most visible. Jitter shows in both the first part of the sequence, where the CamIMU performs continuous motions, and the final part, where the camera moves quickly. Indeed, the V-SLAM system fails several times during the final part of the sequence, while the MPF-SLAM method is able to keep the track by using the information from the inertial measurements. When considering the trajectory obtained from the marker based tracking system as ground truth, the V-SLAM method provides higher accuracy. The root mean square estimation error is $0.24\,\text{cm}$ for the V-SLAM system and $1.11\,\text{cm}$ for the MPF-SLAM method. Note that the high correlation between the trajectory obtained from the marker based tracking system and the V-SLAM method in parts results from the fact that similar computer vision techniques (*cf.* Section 2.3.1.1) are used. Figure 5.17 illustrates the estimation errors obtained from the MPF-SLAM method for two dimensions of the camera position. Note how the variances increase during periods of quick camera motions. This results from the adaptive process noise technique developed in Section 5.1.2.2. The estimates

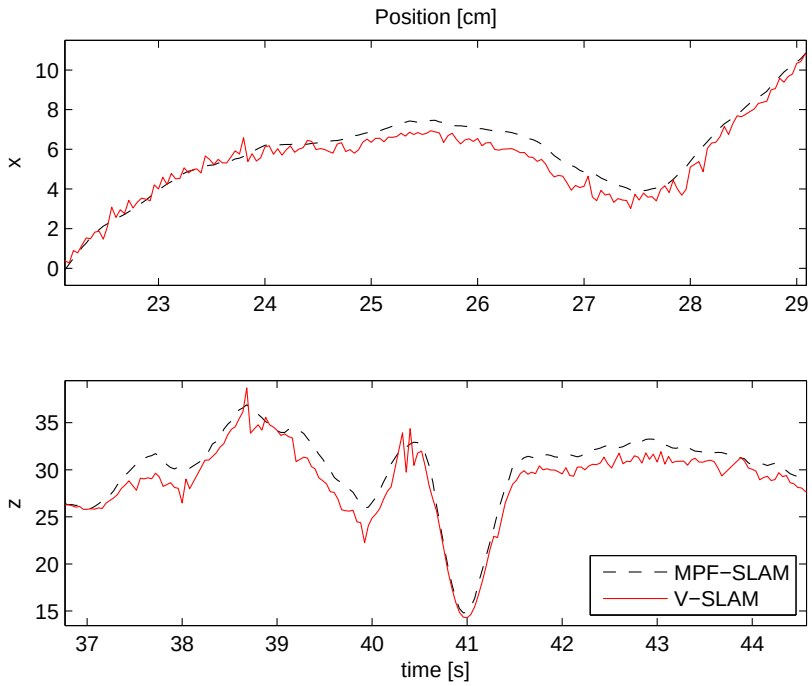


Figure 5.16: *Position estimates obtained from the MPF-SLAM and the V-SLAM system. The plots focus on different parts of the test data sequence.*

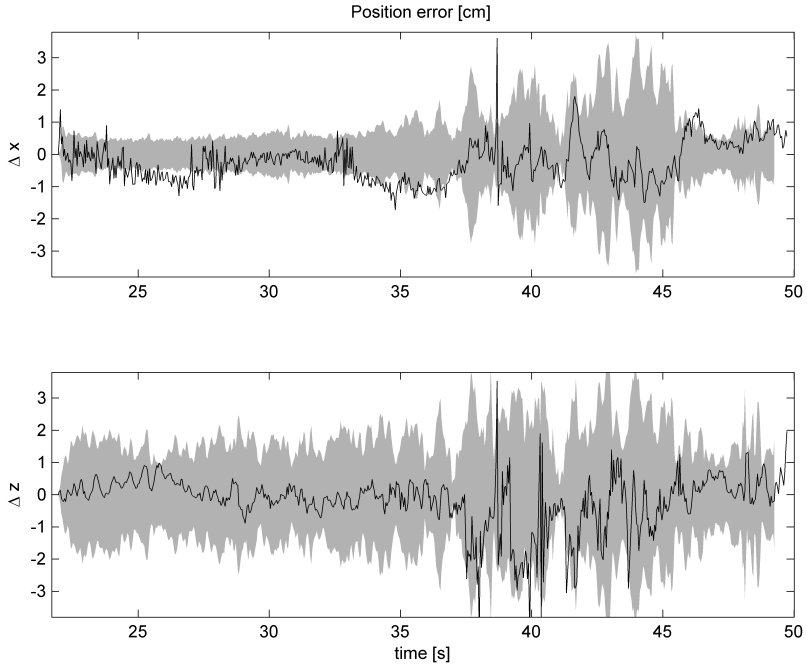


Figure 5.17: *MPF-SLAM: estimation errors and* 99% *confidence regions for two dimensions of the camera position with* $n = 100$ *particles. The confidence regions are obtained from the respective diagonal elements of the covariance, which is estimated from the particle cloud using* (2.42b).

stay mostly within the 99% confidence bounds indicated in the figure, except for the first part of the upper plot, where the covariances are underestimated. Note, however, that the reference trajectory obtained from the marker tracking system is also subject to an unknown error.

5.2.3 Summary

This section completes the visual-inertial pose estimation method developed in Section 5.1 with structure estimation on a per-particle basis so that a MPF-SLAM system is obtained. The Euclidean representation of the 3D feature locations is replaced with an inverse depth parametrisation that allows for initialising features from one camera view and is more suitable for handling distant features. The resulting algorithm is evaluated in the simple experimental environment used in Section 5.1.3 and is compared to the method described previously in Chapter 4. The inverse depth approach is shown to provide undelayed, computationally efficient and consistent initialisation of the extended Kalman filters associated to the map features in each particle. Compared to the previous method, the stochastic model is extended, as correlations between the states related to the IMU and the features are coded in the particle cloud. However, further evaluation of the development of the correlations in a more complex environment is required in order to investigate the ability of the system to close large loops (*cf.* Section 5.2.2.2). The visual-inertial SLAM system developed here yields a smoother trajectory and is more robust against quick camera motions, while the previous visual SLAM method provides more accurate estimates of the camera pose — when considering the marker poses as ground truth — and the 3D feature locations. High accuracy is a typical property of computer vision techniques (*cf.* Section 2.3.4). Both the estimation accuracy and the amount of correlations represented in the particle cloud, can be improved by increasing the number of particles, however, at the expense of higher computational costs.

5.3 Conclusion

This chapter provides the foundation for simultaneous localisation and mapping in large-scale environments. The visual SLAM system developed in Chapter 4 is enhanced to a visual-inertial system that combines the marginalised particle filter for sensor fusion with map estimation on a per-particle basis. The essential algorithms for pose and structure estimation are developed and evaluated in Section 5.1 and 5.2, respectively, and a proof-of-concept is demonstrated in a simple test setup.

6

Concluding Remarks

This thesis investigates robustness and accuracy of real-time markerless camera tracking for mobile augmented reality in both known and unknown environments of different complexity. The major concepts used to achieve the specified aims are: visual-inertial sensor fusion, decoupling of pose and structure estimation and unification of robust computer vision and recursive filtering techniques. The conclusions from this work are given in Section 6.1 and future work is suggested in Section 6.2.

6.1 Conclusion

First, a model-based camera tracking system that fuses visual and inertial measurements in an extended Kalman filter is developed. The system is equipped with an affine illumination invariant feature registration method that can exploit an accurate pose prediction to improve computational efficiency. It uses a textured CAD model of the environment to predict the appearances of corner features in the camera images. The system is shown to work robustly in realistic environments of different complexity, under varying light conditions, fast and erratic camera motions and even short periods without camera measurements, *e.g.* due to occlusion. Compared to vision-only tracking, it shows less jitter and the computational cost is reduced both due to an accurate prediction of the feature appearances and locations in the images and a reduced demand on camera measurements in general. In the process of developing this method, different state-space models are derived and compared under controlled experimental conditions. The models differ mostly in whether and how accelerometer data are used. The acceleration model, where gyroscope and accelerometer data are incorporated as in a strapdown inertial navigation system, demonstrates the best tracking properties. Moreover, treating the inertial data as control inputs is shown to significantly reduce the computational burden, while maintaining the tracking quality.

In order to address tracking in partially known environments, a vision-only system is developed, which is capable of performing robust simultaneous camera localisation and mapping in real-time. The system unifies the robustness and efficiency of standard computer vision techniques with a simplified stochastic model and recursive estimation of the 3D feature locations, each in a separate extended Kalman filter. Compared to an ordinary sequential structure from motion system, the developed method is demonstrated to provide a higher accuracy in both camera and feature localisation and significantly less drift. The experimental evaluation is performed on simulated data and in realistic mid-scale environments. Moreover, the method is robust against outliers and scales with the size of the map. This is mainly achieved by decoupling pose and structure estimation. In the considered test environments, the simplified stochastic model provides a sufficient uncertainty measure for the 3D feature locations. However, due to the unmodelled correlations between the features and the camera pose, this system cannot produce a globally consistent map. Moreover, it makes no use of additional sensors and is — due to the delayed feature initialisation — not suitable for distant features.

In order to address these issues, a conceptual solution for a visual-inertial SLAM system with increased potentiality for large-scale environments is provided, and a proof-of-concept is demonstrated to function in a simple test setup. The idea is to combine the marginalised particle filter for sensor fusion with FastSLAM mapping and undelayed feature initialisation. In order to achieve this, a novel marginalised particle filter algorithm for pose estimation and an inverse depth measurement model for recursive estimation of the 3D feature locations on a per-particle basis are developed. Using these improvements, the previous vision-only system is shown to be enhanced in several ways: The stochastic model is extended, as correlations between the features and the camera pose are represented in the particle cloud. The system is provided with the beneficial effects of visual-inertial sensor fusion. In particular, reduced jitter and a higher robustness in the presence of erratic camera motions are demonstrated. Finally, the inverse depth parametrisation allows for undelayed initialisation of the features from one camera view and is more suitable for handling distant features. At the same time, the decoupling of pose and structure estimation is maintained, thus allowing for an algorithm that scales with the size of the map. However, the enhancements pointed out above go together with a significantly higher computational complexity and a reduced accuracy in the camera and feature localisation, at least with a particle number that allows for real-time operation.

6.2 Future Work

The contributions of this thesis are successive and lead towards a complex visual-inertial system for real-time simultaneous localisation and mapping. The foundation for such a system is provided in Chapter 5.

As pointed out in Section 5.2.3, the performance of the developed method with respect to both estimation accuracy and the amount of correlations represented in the particle cloud, can be improved by increasing the number of particles. In order to preserve real-time performance with a larger particle set, computational costs can be saved, for instance, by using the graphics

hardware and applying parallelisation [178, 179].

Moreover, the results of this thesis indicate that a special loop closing mechanism is needed — when using a particle number that allows for real-time operation — in order to obtain a consistent map estimate. In the area of real-time camera tracking, where the camera moves freely in 3D space, loop closing for FastSLAM is an unsolved problem and, hence, an interesting and new topic for future research. Indeed, loop detection can be solved purely based on camera images, see *e.g.* [49, 182]. However, the problem of propagating this information globally into the map, especially in case of underestimated uncertainty, is unsolved. Recent ideas to express the SLAM problem as a problem of graph optimisation can be used to start with, see *e.g.* [183–185].

Another suggestion for future research is to investigate further sensors such as global positioning system (GPS). Small GPS enhanced inertial measurement units are nowadays available and provide a portable tracking setup. Moreover, the absolute position data can be used to extend the visual-inertial SLAM algorithm developed in this thesis to large-scale outdoor environments. Camera tracking systems that use GPS data are described, for instance, in [84, 120, 126, 127, 186].

Appendix

A

Notation

A.1 Abbreviations and Acronyms

Abbreviation	Meaning
AHRS	attitude and heading reference system
CAD	computer aided design
CamIMU	integrated camera IMU device
CCD	charge-coupled device
CMOS	complementary metal oxide semiconductor
DLT	Direct Linear Transform
DOF	degrees of freedom
DSP	digital signal processor
EKF	extended Kalman filter
FAST	Features from Accelerated Segment Test
FIS	Federated Information Sharing
GPS	global positioning system
IMU	inertial measurement unit
KF	Kalman filter
KLT	Kanade-Lucas tracker
LM	Levenberg-Marquardt
LS	least squares
MEMS	microelectromechanical systems
MPF	marginalised particle filter
OpenGL	open graphics library
pdf	probability density function
PF	particle filter
POSIT	pose from orthography and scaling with iterations
RANSAC	random sampling consensus

Abbreviation	Meaning
SFM	structure from motion
SIFT	scale invariant feature transform
SIR	sampling importance resampling
SIS	sequential importance sampling
SLAM	simultaneous localisation and mapping
SSD	sum of squared differences
SVD	singular value decomposition
VRML	virtual reality modeling language
WLS	weighted least squares

A.2 Symbols and Mathematical Notation

Notation	Meaning
Camera Parameters	
$\kappa_1, \ldots, \kappa_5$	radial and tangential distortion coefficients
d_x, d_y	width and height of a sensor pixel in metric units
f	focal length in metric units
f_x, f_y	focal length in x- and y-direction in pixels per metric unit
h_x, h_y	coordinates of the principal point in pixels
K	intrinsic parameter matrix
P	perspective projection matrix
s_α	skew coefficient in pixels per metric unit
Coordinate Systems and Transformations	
c	camera coordinate system
d	distorted image coordinate system
g	global (earth-fixed) coordinate system
i	image coordinate system
n	normalised image coordinate system ($f = 1$)
p	pixel coordinate system
s	sensor (IMU) coordinate system
w	world coordinate system
$\mathcal{D}$	mapping from n to d
$\mathcal{K}$	mapping from d to p
$\mathcal{P}_n$	mapping from c to n
$\mathcal{P}$	mapping from w to p
$\mathcal{T}$	mapping from w to c
R_{ab}	rotation matrix (rotates points or vectors from coordinate system b to a)

Notation	Meaning
$\mathbf{r}_{ab}$	three-dimensional axis angle representation of R_{ab}
$\mathbf{q}_{ab}$	four-dimensional quaternion representation of R_{ab}
$\mathbf{q}^L$	left-hand side quaternion multiplication matrix
$\mathbf{q}^R$	right-hand side quaternion multiplication matrix
$\bar{\mathbf{q}}$	quaternion conjugate
$\odot$	quaternion multiplication
$\mathrm{rot}(\cdot)$	conversion to a rotation matrix
$\mathbf{s}$	camera pose parameter vector
$\mathbf{s}^q$	seven-dimensional camera pose parameter vector $[\mathbf{q}_{cw}; \mathbf{c}_w]$
$\mathbf{s}^r$	six-dimensional camera pose parameter vector $[\mathbf{r}_{cw}; \mathbf{w}_c]$
Filtering	
$\delta(\cdot)$	Dirac delta function
$\mathbf{x}$	state or parameter vector
$\mathbf{x}^l$	linear partition of $\mathbf{x}$
$\mathbf{x}^n$	nonlinear partition of $\mathbf{x}$
$\mathbf{x}^{[i]}$	i^{th} sample of $\mathbf{x}$
$\hat{\mathbf{x}}_{t\|t-T}$	one step prediction estimate of $\mathbf{x}_t$ given $\mathbf{Y}_{t-T}$ and $\mathbf{U}_t$
$\hat{\mathbf{x}}_{t\|t}$	posterior estimate of $\mathbf{x}_t$ given $\mathbf{Y}_t$ and $\mathbf{U}_t$
$\mathbf{X}_t$	history of states up to time t
$\mathbf{y}_{a,t}$	measurement at time t expressed in coordinate system a
$\mathbf{Y}_t$	history of measurements up to time t
$\mathbf{u}_t$	known control input at time t
$\mathbf{U}_t$	history of known control inputs up to time t
$\mathbf{e}_{a,t}$	measurement noise expressed in coordinate system a
$\mathbf{v}_{a,t}$	process noise expressed in coordinate system a
$\mathbf{z}_t$	innovation or residual at time t
$\bar{\mathbf{z}}_t$	normalised innovation or residual at time t
S_t	innovation covariance at time t
$\mathrm{cov}(\mathbf{x})$	covariance of $\mathbf{x}$
$\mathrm{var}(x)$	variance of scalar x
$\mathcal{N}(\hat{\mathbf{x}}, P)$	multivariate normal distribution with mean $\hat{\mathbf{x}}$ and covariance P
$\mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}, P)$	multivariate Gaussian pdf evaluated in $\mathbf{x}$
N_{eff}	effective particle size
P_{eff}	effective percentage of particles
$\Pr(\mathcal{A})$	probability of statement $\mathcal{A}$
Miscellaneous	
a	scalar or coordinate system

Notation	Meaning
$\mathbf{a}$	vector or origin of coordinate system a
$\mathbf{a}_b$	$\mathbf{a}$ expressed in coordinate system b
$\dot{\mathbf{a}}_{b,t}$	linear velocity of $\mathbf{a}$ at time t expressed in coordinate system b
$\ddot{\mathbf{a}}_{b,t}$	linear body acceleration of $\mathbf{a}$ at time t expressed in coordinate system b
$(\mathbf{v})_i$	i^{th} element of vector $\mathbf{v}$
$\mathbf{0}_n$	n-dimensional zero vector
A	matrix
A^n	multiplication of n matrices A
I_n	$n \times n$ identity matrix
$(\cdot)^T$	transpose
$(\cdot)^{-1}$	inverse
$(\cdot)^{-T}$	transposed inverse
$(f \circ g)(\cdot) = f\big(g(\cdot)\big)$	composition of functions f and g
$\mathbf{l}_a$	epipolar line expressed in coordinates system a
$\boldsymbol{\omega}_{a,t}^{ab}$	angular velocity at time t from coordinate system b to a expressed in a (^{ab} is suppressed, if the notation is unambiguous)
$\mathbf{m}_a$	point $\mathbf{m}$ expressed in coordinate system a
$\mathbf{m}_w^{(j)}$	j^{th} world point
$\mathbf{m}_p^{(ij)}$	observation of the j^{th} world point in the i^{th} image in pixels
$(\mathbf{m}_n^{(i)}, \mathbf{m}_w^{(i)})$	i^{th} 2D/3D correspondence
$\mathbf{g}$	gravity vector
$\bar{\mathbf{g}} = \mathbf{g}/\|\mathbf{g}\|$	direction of gravity

B

Transformations and Kinematics

B.1 Representation of Rotation

A 3D rotation has three degrees of freedom. However, there are many ways to represent such a rotation using a varying number of parameters, each with its individual advantages and disadvantages [39, 187, 188]. The most commonly used representations are orthogonal matrices, axis angle, unit quaternions and Euler angles. The three representations used in this thesis are introduced in the sequel.

Rotation Matrix

The 3×3 rotation matrix R gives a simple way of representing a 3D rotation. However, it has nine parameters leading to the high dimensional space of the orthogonal constraint. It has the following properties:

$$RR^T = R^T R = I_3 \Rightarrow R^{-1} = R^T \tag{B.1}$$

$$\det(R) = 1 \tag{B.2}$$

The rotation matrix R can be directly used to rotate a vector or a point from one coordinate system to another. Let $\mathbf{m}_b$ be a point expressed in coordinate system b. Then $\mathbf{m}_a$ is:

$$\mathbf{m}_a = R_{ab}\mathbf{m}_b. \tag{B.3a}$$

Analogously:

$$\mathbf{m}_b = R_{ba}\mathbf{m}_a, \tag{B.3b}$$

with $R_{ba} = R_{ab}^T$.

Axis Angle

Under the axis angle representation, a 3D rotation is defined as a three-dimensional vector $\mathbf{r} = [a, b, c]^T$, where the direction is that of the rotation axis and the norm equals the rotation angle $\theta = \|\mathbf{r}\|$. The conversion to a rotation matrix R is given by the Rodrigues' formula:

$$R = \mathrm{rot}(\mathbf{r}) = \mathrm{I}_3 + k(\theta)\,\mathrm{S}(\mathbf{r}) + g(\theta)\,\mathrm{S}(\mathbf{r})^2, \tag{B.4}$$

where I_3 denotes the 3×3 identity matrix, $k(\theta) = \frac{\sin(\theta)}{\theta}$ and $g(\theta) = \frac{1-\cos(\theta)}{\theta}$. $\mathrm{S}(\mathbf{v})$ denotes the skew-symmetric matrix defined by the three-dimensional vector $\mathbf{v}$:

$$\mathrm{S}(\mathbf{v}) = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}. \tag{B.5}$$

The axis angle form constitutes a minimal rotation parametrisation and is hence suited for nonlinear optimisation. This representation has also been used in the extended Kalman filter framework, see for instance [6] and [39], respectively. However, a singularity is given by $\theta = 0$ requiring special treatment.

Unit Quaternion

A quaternion $\mathbf{q} = [q_w, q_x, q_y, q_z]^T$ is defined as a four-dimensional vector. A three-dimensional vector $\mathbf{v}$ is identified with the quaternion $[0, \mathbf{v}^T]$. In order to simplify the notation, let $\mathbf{q}_v = [q_x, q_y, q_z]^T$. The multiplication $\odot$ of two quaternions $\mathbf{q}$ and $\mathbf{p}$ is:

$$\mathbf{q} \odot \mathbf{p} = \begin{bmatrix} q_w p_w - \mathbf{q}_v \cdot \mathbf{p}_v \\ q_w \mathbf{p}_v + p_w \mathbf{q}_v + \mathbf{q}_v \times \mathbf{p}_v \end{bmatrix}, \tag{B.6a}$$

where $\mathbf{q}_v \cdot \mathbf{p}_v$ denotes the scalar (inner) product and $\mathbf{q}_v \times \mathbf{p}_v$ denotes the cross (outer) product of two three-dimensional vectors. The multiplication can also be written as a matrix multiplication:

$$\mathbf{q} \odot \mathbf{p} = \underbrace{\begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & -q_z & q_y \\ q_y & q_z & q_w & -q_x \\ q_z & -q_y & q_x & q_w \end{bmatrix}}_{q^L} \begin{bmatrix} p_w \\ p_x \\ p_y \\ p_z \end{bmatrix} \tag{B.6b}$$

$$= \underbrace{\begin{bmatrix} p_w & -p_x & -p_y & -p_z \\ p_x & p_w & -p_z & p_y \\ p_y & p_z & p_w & -p_x \\ p_z & -p_y & p_x & p_w \end{bmatrix}}_{p^R} \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} \tag{B.6c}$$

The multiplication is associative but not commutative. The quaternion conjugate $\bar{\mathbf{q}}$ is defined as:

$$\bar{\mathbf{q}} = \begin{bmatrix} q_w \\ -\mathbf{q}_v \end{bmatrix}. \tag{B.7}$$

The inverse quaternion $\mathbf{q}^{-1}$ is defined as:

$$\mathbf{q}^{-1} = \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|^2}, \tag{B.8}$$

implying that $\mathbf{q}^{-1} = \bar{\mathbf{q}}$, if $\mathbf{q}$ is a unit quaternion, that is $\|\mathbf{q}\| = 1$.

A 3D rotation is represented by a unit quaternion $\mathbf{q}$. The conversion to a rotation matrix R is given by:

$$R = \mathrm{rot}(\mathbf{q}) = \begin{bmatrix} q_w^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_z q_w) & 2(q_z q_x + q_w q_y) \\ 2(q_x q_y + q_w q_z) & q_w^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_w q_x) \\ 2(q_z q_x - q_w q_y) & 2(q_y q_z + q_w q_x) & q_w^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix} \tag{B.9}$$

The relation to the axis angle representation $\mathbf{r}$ is:

$$q_w = \cos\left(\frac{\theta}{2}\right) \quad \text{and} \quad \mathbf{q}_v = \sin\left(\frac{\theta}{2}\right)\mathbf{v}, \tag{B.10}$$

with $\theta = \|\mathbf{r}\|$ and $\mathbf{v} = \mathbf{r}/\|\mathbf{r}\|$.

The unit quaternion constitutes a singularity-free representation of a 3D rotation. It is most commonly used in robotics and computer vision [105, 121]. However, this representation is not minimal, resulting in the constraint of unit magnitude.

B.2 Rotation Derivative

Using one of the vector parametrisations given in Section B.1 in an extended Kalman filter framework requires the computation of the first derivative (Jacobian) of the rotation matrix with respect to this parametrisation. In this thesis, the Jacobians of $R\mathbf{m}$ — the rotation of a point — with respect to $\mathbf{r}$ and $\mathbf{q}$ are of special interest and are therefore given below.

The Jacobian of $R\mathbf{m}$ with respect to $\mathbf{r}$ is given by the 3×3 matrix:

$$\begin{aligned} \frac{\partial R\mathbf{m}}{\partial \mathbf{r}} = \frac{\cos(\theta) - k(\theta)}{\theta^2}(\mathbf{r}\times\mathbf{m})\mathbf{r}^T + \frac{\sin(\theta) - 2\theta g(\theta)}{\theta^3}(\mathbf{r}\times(\mathbf{r}\times\mathbf{m}))\mathbf{r}^T \\ - k(\theta)\,\mathrm{S}(\mathbf{m}) + g(\theta)\big(-\mathrm{S}(\mathbf{r}\times\mathbf{m}) + (\mathbf{r}\cdot\mathbf{m})\,\mathrm{I}_3 - \mathbf{m}\mathbf{r}^T\big), \end{aligned} \tag{B.11}$$

where θ, $k(\theta)$, $g(\theta)$, $\mathrm{S}(\cdot)$ and I_3 are defined as in B.1.

The Jacobian of $R\mathbf{m}$ with respect to $\mathbf{q}$ is given by the 3×4 matrix:

$$\frac{\partial R\mathbf{m}}{\partial \mathbf{q}} = \begin{bmatrix} d_0 & d_1 & d_2 & d_3 \\ -d_3 & -d_2 & d_1 & d_0 \\ d_2 & -d_3 & -d_0 & d_1 \end{bmatrix} \tag{B.12}$$

with

$$\begin{aligned} d_0 &= 2(q_w m_x - q_z m_y + q_y m_z) \\ d_1 &= 2(q_x m_x + q_y m_y + q_z m_z) \\ d_2 &= 2(-q_y m_x + q_x m_y + q_w m_z) \\ d_3 &= 2(-q_z m_x - q_w m_y + q_x m_z). \end{aligned}$$

B.3 Angular Velocities and Unit Quaternions

The instantaneous angular velocity from coordinate frame b to coordinate frame a expressed in coordinate frame a is defined as a three-dimensional vector $\boldsymbol{\omega}_a^{ab}$. It can be transformed according to:

$$\boldsymbol{\omega}_a^{ab} = -\boldsymbol{\omega}_a^{ba} \tag{B.13a}$$

$$\boldsymbol{\omega}_a^{ab} = \mathbf{q}_{ab} \odot \boldsymbol{\omega}_b^{ab} \odot \mathbf{q}_{ba}. \tag{B.13b}$$

Note that $\boldsymbol{\omega}$ is identified with the quaternion $[0, \boldsymbol{\omega}^T]^T$ as introduced in Section B.1.

The continuous-time differential equation describing the dynamic model of a unit quaternion is:

$$\dot{\mathbf{q}}_{ab,t} = \frac{1}{2}\, \boldsymbol{\omega}_{a,t}^{ab} \odot \mathbf{q}_{ab,t}. \tag{B.14}$$

Integrating (B.14) with respect to time — assuming that $\boldsymbol{\omega}_{a,t}^{ab}$ is piece-wise constant between the sampling instants kT — results in the discrete-time dynamic model:

$$\mathbf{q}_{ab,t+T} = \exp\left(\frac{T}{2}\, \boldsymbol{\omega}_{a,t}^{ab}\right) \odot \mathbf{q}_{ab,t}, \tag{B.15}$$

where

$$\exp(\mathbf{v})^T = \begin{bmatrix} \cos(\|\mathbf{v}\|) & \frac{\mathbf{v}^T}{\|\mathbf{v}\|} \sin(\|\mathbf{v}\|) \end{bmatrix}$$

is the quaternion exponential. A detailed derivation of these relations is given for instance in [13, 188]. Equation (B.15) can be used to integrate angular velocity samples, *e.g.* those obtained from a 3D gyroscope, to an absolute orientation. Defining $\mathbf{v} = \frac{T}{2}\, \boldsymbol{\omega}_a^{ab}$, the Jacobian of (B.15) with respect to $\boldsymbol{\omega}_a^{ab}$ is given by the 4×3 matrix:

$$\frac{\partial \exp(\mathbf{v}) \odot \mathbf{q}_{ab}}{\partial \boldsymbol{\omega}_a^{ab}} = \frac{T}{2} \mathbf{q}_{ab}^R \begin{bmatrix} -\frac{\mathbf{v}^T}{\|\mathbf{v}\|} \sin(\|\mathbf{v}\|) \\ \frac{1}{\|\mathbf{v}\|} \left[\mathrm{I}_3 - \frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2}\right] \sin(\|\mathbf{v}\|) + \frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2} \cos(\|\mathbf{v}\|) \end{bmatrix}, \tag{B.16}$$

and the Jacobian of (B.15) with respect to $\mathbf{q}_{ab}$ is given by the 4×4 matrix $\exp(\mathbf{v})^L$. Note that a small angle approximation of (B.15) results in the much simpler linear equation:

$$\mathbf{q}_{ab,t+T} \approx \left(\mathrm{I}_4 - \frac{T}{2}\, \mathrm{S}^{\omega}(\boldsymbol{\omega}_a^{ab})\right) \mathbf{q}_{ab,t} \tag{B.17a}$$

$$= \mathbf{q}_{ab,t} - \frac{T}{2}\, \mathrm{S}^{\mathrm{q}}(\mathbf{q}_{ab,t}) \boldsymbol{\omega}_a^{ab}, \tag{B.17b}$$

where

$$\mathrm{S}^{\omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{bmatrix}$$

and

$$\mathrm{S}^{\mathrm{q}}(\mathbf{q}) = \begin{bmatrix} -q_x & -q_y & -q_z \\ q_w & q_z & -q_y \\ -q_z & q_w & q_x \\ q_y & -q_x & q_w \end{bmatrix}.$$

The approximation (B.17), however, does not keep $\mathbf{q}_{ab,t+T}$ at unit length.

From (B.13b) and (B.14) it can be shown that the angular velocity is identical in each point of a rigid body. Let $\mathbf{q}_{ac,t} = \mathbf{q}_{ab} \odot \mathbf{q}_{bc,t}$, where coordinate frame a and b are rigidly connected. It is shown that $\boldsymbol{\omega}_{a,t}^{ac} = \boldsymbol{\omega}_{a,t}^{bc}$ holds:

$$\begin{aligned} \boldsymbol{\omega}_{a,t}^{ac} &= 2(\dot{\mathbf{q}}_{ac,t} \odot \mathbf{q}_{ca,t}) = 2\left(\frac{\partial \mathbf{q}_{ab} \odot \mathbf{q}_{bc,t}}{\partial t} \odot \mathbf{q}_{cb,t} \odot \mathbf{q}_{ba}\right) = 2\left(\frac{\partial \mathbf{q}_{ab}^{L} \mathbf{q}_{bc,t}}{\partial t} \odot \mathbf{q}_{cb,t} \odot \mathbf{q}_{ba}\right) \\ &= 2\big((\dot{\mathbf{q}}_{ab}^{L} \mathbf{q}_{bc,t} + \mathbf{q}_{ab}^{L} \dot{\mathbf{q}}_{bc,t}) \odot \mathbf{q}_{cb,t} \odot \mathbf{q}_{ba}\big) = 2\big((\mathbf{q}_{ab} \odot \dot{\mathbf{q}}_{bc,t}) \odot \mathbf{q}_{cb,t} \odot \mathbf{q}_{ba}\big) \\ &= \mathbf{q}_{ab} \odot \big(2(\dot{\mathbf{q}}_{bc,t} \odot \mathbf{q}_{cb,t}\big) \odot \mathbf{q}_{ba} = \mathbf{q}_{ab} \odot \boldsymbol{\omega}_{b,t}^{bc} \odot \mathbf{q}_{ba} = \boldsymbol{\omega}_{a,t}^{bc}. \end{aligned} \tag{B.18}$$

Note that $\dot{\mathbf{q}}_t = \frac{\partial \mathbf{q}_t}{\partial t}$ and $\dot{\mathbf{q}} = 0$, if $\mathbf{q}$ is static.

List of Figures

List of Tables

Bibliography

[1] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. *Computer & Graphics*, 2008. accepted.

[2] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. In *IEEE Virtual Reality Conference (VR)*, Reno, Nevada, March 2008.

[3] G. Bleser, C. Wohleber, and D. Stricker. Fast and Stable Tracking for AR fusing Video and Inertial Sensor Data. In *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 109–115, Plzen, CZ, February 2006.

[4] J. Chandaria, G. Thomas, B. Bartczak, K. Koeser, R. Koch, M. Becker, G. Bleser, D. Stricker, C. Wohlleber, M. Felsberg, J. Hol, T. Schön, J. Skoglund, P. Slycke, and S. Smeitz. Real-time Camera Tracking in the Matris Project. In *International Broadcasting Convention (IBC)*, pages 321–328, Amsterdam, Netherlands, September 2006.

[5] G. Bleser, Y. Pastarmov, and D. Stricker. Real-time 3D Camera Tracking for Industrial Augmented Reality Applications. In *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 47–54, Plzen, CZ, February 2005.

[6] G. Bleser, M. Becker, and D. Stricker. Real-time vision-based tracking and reconstruction. *Journal of Real-Time Image Processing*, 2(2–3):161–175, November 2007.

[7] G. Bleser, H. Wuest, and D. Stricker. Online Camera Pose Estimation in Partially Known and Dynamic Scenes. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 56–65, Santa Barabara, CA, October 2006.

[8] M. Becker, G. Bleser, A. Pagani, Y. Pastarmov, D. Stricker, F. Vial, J. Weidenhausen, C. Wohlleber, and H. Wuest. Visual Tracking for Augmented Reality: No Universal Solution but Many Powerful Building Blocks. In *Virtuelle und Erweiterte Realität: 2. Workshop der GI-Fachgruppe VR/AR*, pages 107–118, 2005.

[9] M. Zoellner, D. Stricker, G. Bleser, and Y. Pastarmov. iTACITUS — Novel Interaction and Tracking Paradigms for Mobile AR. In *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, pages 110–117, Brighton, UK, October 2007.

[10] M. Becker, G. Bleser, A. Pagani, D. Stricker, and H. Wuest. An Architecture for Prototyping and Application Development of Visual Tracking Systems. In *IEEE 3DTV-Conference: Capture, Transmission and Display of 3D Video*, 2007.

[11] G. Bleser and D. Stricker. Using the Marginalised Particle Filter for Real-Time Visual-Inertial Sensor Fusion. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Cambridge, UK, September 2008.

[12] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.

[13] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3-D Vision*, volume 26. Springer Verlag, 2003.

[14] J. Heikkilae and O. Silven. A Four-step Camera Calibration Procedure with Implicit Image Correction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, San Juan, Puerto Rico, June 1997.

[15] F. Devernay and O. Faugeras. Straight lines have to be straight. *Machine Vision and Applications*, 13:14–24, 2001.

[16] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision (ICCV)*, Kerkyra (Corfu), Greece, September 1999.

[17] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3 edition, 2007.

[18] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab. URL `http://www.vision.caltech.edu/bouguetj/calib_doc`. Accessed July 4th, 2008.

[19] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004.

[20] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 3 edition, January 1968.

[21] H. Kato and M. Billinghurst. Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In *International Workshop on Augmented Reality*, page 85–94, San Francisco, USA, October 1999.

[22] M. Billinghurst and H. Kato. Collaborative Mixed Reality. In *International Symposium on Mixed Reality (ISMR)*, pages 261–284, Yokohama, Japan, March 1999.

[23] L. Naimark and E. Foxlin. Circular Data Matrix Fiducial System and Robust Image Processing for a Wearable Vision-Inertial Self-Tracker. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Darmstadt, Germany, September 2002.

[24] J. Mooser, S. You, and U. Neumann. Tricodes: A Barcode-Like Fiducial Design for Augmented Reality Media. In *IEEE International Conference on Multimedia and Expo*, pages 1301–1304, Toronto, Ontario, Canada, July 2006.

[25] ARToolKit. URL `http://www.hitl.washington.edu/artoolkit/`. Accessed August 8th, 2008.

[26] E. Foxlin and L. Naimark. VIS-Tracker: A Wearable Vision-Inertial Self-Tracker. In *IEEE Virtual Reality (VR)*, Los Angeles, CA, March 2003.

[27] G. Thomas, J. Jin, and C. Urquhart. A Versatile Camera Position Measurement System for Virtual Reality TV Production. In *International Broadcasting Convention (IBC)*, Amsterdam, Netherlands, September 1997.

[28] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[29] H. Yang, M. Pollefeys, G. Welch, J.-M. Frahm, and A. Ilie. Differential camera tracking through linearizing the local appearance manifold. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, June 2007.

[30] A. Comport, E. Malis, and P.Rives. Accurate Quadrifocal Tracking for Robust 3D Visual Odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[31] L. Vacchetti, V. Lepetit, and P. Fua. Fusing Online and Offline Information for Stable Real-Time 3D Tracking in Real-time. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 26, pages 1391–1391, Madison, Wisconsin, June 2003.

[32] T. Zinßer, C. Gräßl, and H. Niemann. Efficient Feature Tracking for Long Video Sequences. In *Deutsche Arbeitsgemeinschaft Mustererkennung (DAGM)*, pages 326–333, Tübingen, Germany, August 2004.

[33] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway. Robust Real-Time SLAM Using Scale Prediction and Exemplar Based Feature Description. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, June 2007.

[34] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, June 2007.

[35] G. Hager and P. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[36] F. Jurie and M. Dhome. A Simple and Efficient Template Matching Algorithm. In *International Conference on Computer Vision (ICCV)*, pages 544–549, Vancouver, British Columbia, Canada, July 2001.

[37] D. Stricker. Tracking with Reference Images: A Real-Time and Markerless Tracking Solution for Out-Door Augmented Reality Applications. In *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, pages 91–96, Glyfada, Nr Athens, Greece, November 2001.

[38] I. Matthews, T. Ishikawa, and S. Baker. The Template Update Problem. In *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 26, pages 810–815, June 2004.

[39] Z. Zhang and O. Faugeras. *3D Dynamic Scene Analysis*. Springer, 1992.

[40] P. Smith, I. Reid, and A. J. Davison. Real-Time Monocular SLAM with Straight Lines. In *British Machine Vision Conference (BMVC)*, Edinburgh, September 2006.

[41] G. Thomas. Real-Time Camera Pose Estimation for Augmenting Sports Scenes. In *Conference on Visual Media Production (CVMP)*, London, UK, November 2006.

[42] O. Koch and S. Teller. Wide-Area Egomotion Estimation from Known 3D Structure. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, June 2007.

[43] T. Drummond and R. Cipolla. Real-Time Visual Tracking of Complex Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:932–946, 2002.

[44] G. Reitmayr and T. Drummond. Going out: Robust Model-based Tracking for Outdoor Augmented Reality. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 109–118, Santa Barabara, CA, October 2006.

[45] M. Pupilli and A. Calway. Real-time Camera Tracking Using Known 3D Models and a Particle Filter. In *International Conference on Pattern Recognition (ICPR)*, Hong Kong, August 2006.

[46] G. Simon, V. Lepetit, and M.-O. Berger. Computer Vision Methods for Registration: Mixing 3D Knowledge and 2D Correspondences for Accurate Image Composition. In *International Workshop on Augmented Reality*, San Francisco, California, November 1998.

[47] A. I. Comport, E. Marchand, and F. Chaumette. A real-time tracker for markerless augmented reality. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Tokyo, Japan, October 2003.

[48] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91–110, November 2004.

[49] P. Newman and K. Ho. SLAM- Loop Closing with Visually Salient Features. In *IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005.

[50] K. Mikolajczyk and C. Schmid. Comparison of affine-invariant local detectors and descriptors. In *European Signal Processing Conference (EUSIPCO)*, Vienna, Autstria, September 2004.

[51] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1615–1630, 2005.

[52] V. Lepetit and P. Fua. Keypoint Recognition using Randomized Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006.

[53] A. Boffy and Y. Genc. Real-Time Feature Matching using Adaptive and Spatially Distributed Classification. In *British Machine Vision Conference (BMVC)*, Edinburgh, September 2006.

[54] M. Grabner, H. Grabner, and B. Horst. Tracking via Discriminative Online Learning of Local Features. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, MN, June 2007.

[55] A. J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *International Conference on Computer Vision (ICCV)*, volume 2, Nice, France, October 2003.

[56] T. Schön, R. Karlsson, D. Törnqvist, and F. Gustafsson. A Framework for Simultaneous Localization and Mapping Utilizing Model Structure. In *International Conference on Information Fusion*, Quebec, Canada, July 2007.

[57] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan, November 2007.

[58] B. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, April 1981.

[59] H. Jin, P. Favaro, and S. Soatto. Real-Time Feature Tracking and Outlier Rejection with Changes in Illumination. In *IEEE International Conference on Computer Vision (ICCV)*, pages 684–689, Vancouver, British Columbia, Canada, July 2001.

[60] S. Benhimane and E. Malis. Real-time image-based tracking of planes using Efficient Second-order Minimization. In *International Conference on Intelligent Robots and Systems*, pages 943–948, Sendai, Japan, October 2004.

[61] S. Benhimane, A. Ladikos, V. Lepetit, and N. Navab. Linear and Quadratic Subsets for Template-Based Tracking. In *Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, June 2007.

[62] J. Y. Bouguet. Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm. Technical report, Intel Corporation, Microprocessor Research Labs, 2002. Accessed August 6th, 2008.

[63] XSens. MTx 3DOF orientation tracker. URL `http://www.xsens.com`. Accessed July 7th, 2008.

[64] D. Titterton and J. Weston. *Strapdown Inertial Navigation Technology*. American Institute of Aeronautics and Astronautics, 2 edition, 2004.

[65] A. Lawrence. *Modern Inertial Technology: Navigation, Guidance, and Control*. Mechanical Engineering Series. Springer-Verlag, 1998.

[66] European project MATRIS (Markerless real-time Tracking for Augmented Reality Image Synthesis). URL `http://www.ist-matris.org/`. Accessed July 9th, 2008.

[67] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. In *Journal of the Optical Society of America*, volume 4, pages 629–642, April 1987.

[68] J. D. Hol, T. B. Schön, and F. Gustafsson. Relative Pose Calibration of a Spherical Camera and an IMU. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Cambridge, UK, September 2008.

[69] 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.

[70] T. Baillot, S. Julier, D. Brown, and M. Livingston. A Tracker Alignment Framework for Augmented Reality. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 142–150, Tokyo, October 2003.

[71] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle. Review and analysis of solutions of the 3-point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, December 1994.

[72] R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya, and M. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6): 1426–1446, November 1989.

[73] D. DeMenthon and L. Davis. Model-Based Object Pose in 25 Lines of Code. In *European Conference on Computer Vision (ECCV)*, pages 335–343, Santa Margherita Ligure, Italy, 1992.

[74] D. Oberkampf, D. DeMenthon, and L. Davis. Iterative Pose Estimation using Coplanar Feature Points. *Computer Vision and Image Understanding*, 63(3):495 – 511, May 1996.

[75] P. Huber. *Robust Statistics*. Wiley, 1981.

[76] P. Meer, D. Mintz, D. Kim, and A. Rosenfeld. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, April 1991.

[77] O. Faugeras and Q.-T. Luong. *The Geometry of Multiple Images*. MIT Press, 2001.

[78] R. Hartley and P. Sturm. Triangulation. In *International Conference on Computer Analysis of Images and Patterns*, pages 190–197, Prague, Czech Republic, September 1995.

[79] R. Bolles and M. Fischler. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *International Joint Conference on Artificial Intelligence (IJCAI)*, Vancouver, Canada, 1981.

[80] D. Nister. Preemptive RANSAC for Live Structure and Motion Estimation. In *International Conference on Computer Vision (ICCV)*, Nice, France, October 2003.

[81] R. Subbarao and P. Meer. Beyond RANSAC: User Independent Robust Regression. In *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, New York, 2006.

[82] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis. In *International Workshop on Vision Algorithms: Theory and Practice*, volume 1883, pages 298–372, London, UK, 1999. Springer-Verlag.

[83] K. Cornelis. *From Uncalibrated Video to Augmented Reality*. PhD thesis, Catholic University Leuven, Naamsestraat 22, 3000 Leuven, Belgien, November 2004.

[84] E. Mouragnon, F. Dekeyser, P. Sayd, M. Lhuillier, and M. Dhome. Real time Localization and 3D Reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 363–370, 2006.

[85] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. In *European Conference on Computer Vision (ECCV)*, Graz, Austria, May 2006.

[86] K. Koeser, V. Haertel, and R. Koch. Robust Feature Representation for Efficient Camera Registration. In *Deutsche Arbeitsgemeinschaft Mustererkennung (DAGM)*, Berlin, Germany, September 2006.

[87] H. Wuest and D. Stricker. Tracking of Industrial Objects by Using CAD Models. *Journal of Virtual Reality and Broadcasting*, 4(1):155–164, 2006.

[88] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME-Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[89] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Radar and Signal Processing*, pages 107–112, Bombay, India, January 1993.

[90] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[91] T. Schön, F. Gustafsson, and P.-J. Nordlund. Marginalized Particle Filters for Mixed Linear Nonlinear State-Space Models. *IEEE Transactions on Signal Processing*, 3(7): 2279–2289, July 2005.

[92] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer-Verlag, New York, 2001.

[93] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and Computing*, 10:197–208, 2000.

[94] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 95(446):590–599, June 1999.

[95] A. Doucet. On sequential Monte Carlo methods for Bayesian filtering. Technical report, Cambridge University, 1998.

[96] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan. The Unscented Particle Filter. Technical report, Engineering Department, Cambridge University, August 2000.

[97] M. Isard and A. Blake. CONDENSATION - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[98] G. Qian and R. Chellappa. Structure From Motion Using Sequential Monte Carlo Methods. *Computer Vision*, 2:614–621, 2001.

[99] M. Pupilli and A. Calway. Real-time Camera Tracking Using a Particle Filter. In *British Machine Vision Conference (BMVC)*, pages 519–528, Oxford, England, September 2005. Department of Computer Science, University of Bristol, BMVA Press.

[100] J. Hol, T. Schön, and F. Gustaffson. On Resampling Algorithms for Particle Filters. In *Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, September 2006.

[101] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, 2000.

[102] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, Inc, 1970.

[103] F. Gustafsson. *Adaptive Filtering and Change Detection*. John Wiley & Sons, September 2000.

[104] E. A. Wan and R. van der Merwe. The Unscented Kalman Filter for Nonlinear Estimation. In *IEEE Adaptive Systems for Signal Processing, Communications and Control Symposium*, pages 153–158, Lake Louise, AB, Canada, October 2000.

[105] A. J. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29: 1052–1067, 2007.

[106] A. Doucet, N. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, pages 613 – 624, 2001.

[107] R. Karlsson, T. Schön, and F. Gustafsson. Complexity Analysis of the Marginalized Particle Filter. *IEEE Transactions on Signal Processing*, 53(11):4408–4411, November 2005.

[108] T. Schön, R. Karlsson, and F. Gustafsson. The Marginalized Particle Filter in Practice. In *IEEE Aerospace Conference*, Big Sky, USA, March 2006.

[109] R. Sim, P. Elinas, M. Griffin, A. Shyr, and J. J. Little. Design and analysis of a framework for real-time vision-based SLAM using Rao-Blackwellised particle filters. In *Canadian Conference on Computer and Robotic Vision (CRV)*, Quebec City, QC, 2006.

[110] R. Karlsson, T. B. Schön, D. Törnqvist, G. Conte, and F. Gustafsson. Utilizing Model Structure for Efficient Simultaneous Localization and Mapping for a UAV Application. In *IEEE Aerospace Conference*, Big Sky, MT, USA, March 2008.

[111] G. Hendeby. *Performance and Implementation Aspects of Nonlinear Filtering*. PhD thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden, February 2008.

[112] G. Hendeby, R. Karlsson, and F. Gustafsson. A new formulation of the rao-blackwellized particle filter. Technical report, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, August 2007.

[113] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *The fourth International Symposium on Robotics Research*, pages 467–474, Cambridge, MA, USA, 1988. MIT Press.

[114] H. Durrant-Whyte and T. Bailey. Simultaneous Localisation and Mapping (SLAM):. Part I — The Essential Algorithms. *Robotics and Automation Magazine*, 13:99–110, June 2006.

[115] T. Bailey and H. Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part II — State of the art. *Robotics and Automation Magazine*, 13:129–147, 2006.

[116] M. Pupilli and A. Calway. Real-time Visual SLAM with Resilience to Erratic Motion. In *IEEE Computer Vision and Pattern Recognition*, New York, NY, June 2006.

[117] E. Eade and T. Drummond. Scalable Monocular SLAM. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 469–476, New York, NY, June 2006.

[118] L. A. Clemente, A. J. Davison, I. Reid, J. Neira, and J. D. Tardos. Mapping Large Loops with a Single Hand-Held Camera. In *Robotics: Science and Systems (RSS)*, 2007.

[119] B. Leibe, N. Cornelis, K. Cornelis, and L. V. Gool. Dynamic 3D Scene Analysis from a Moving Vehicle. In *Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, USA, June 2007.

[120] P. Mordohai, J.-M. Frahm, A. Akbarzadeh, B. Clipp, C. Engels, D. Gallup, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q.-X. Y. Yang, H. Stewenius, H. Towles, G. Welch, R. Yang, M. Pollefeys, and D. Nister. Real-time video-based reconstruction of urban environments. In *ISPRS Working Group V/4 Workshop 3D-ARCH 2007: 3D Virtual Reconstruction and Visualization of Complex Architectures*, ETH Zurich, Switzerland, July 2007.

[121] J. Hol, T. Schön, H. Luinge, P. Slycke, and F. Gustafsson. Robust Real-Time Tracking by Fusing Measurements from Inertial and Vision Sensors. *Journal of Real-Time Image Processing*, 2(2-3):149–160, November 2007.

[122] S. Se, D. G. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21: 735–758, 2002.

[123] R. Sim, M. Griffin, A. Shyr, and J. Little. Scalable real-time vision-based SLAM for planetary rovers. In *IEEE IROS Workshop on Robot Vision for Space Applications*, Edmonton, AB, 2005.

[124] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard. Visually Navigating the RMS Titanic with SLAM information filters. In *Robotics: Science and Systems (RSS)*, Cambridge, MA, USA, June 2005.

[125] R. Sim and J. J. Little. Autonomous vision-based exploration and mapping using hybrid maps and rao-blackwellised particle filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

[126] T. Oskiper, Z. Zhu, S. Samarasekera, and R. Kumar. Visual Odometry System Using Multiple Stereo Cameras and Inertial Measurement Unit. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Minneapolis, Minnesota, June 2007.

[127] G. Reitmayr and T. Drummond. Initialisation for Visual Tracking in Urban Environments. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan, November 2007.

[128] A. Gee, D. Chekhlov, W. Mayol, and A. Calway. Discovering planes and collapsing the state space in visual SLAM. In *British Machine Vision (BMVC)*, Warwick, September 2007.

[129] D. Chekhlov, A. Gee, A. Calway, and W. Mayol-Cuevas. Ninja on a Plane: Automatic Discovery of Physical Planes for Augmented Reality Using Visual SLAM. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan, September 2007.

[130] H. Jin, P. Favaro, and S. Soatto. A Semi-direct Approach to Structure From Motion. *The Visual Computer*, 19(6):377–394, October 2003.

[131] J. Knight, A. Davison, and I. Reid. Towards Constant Time SLAM using Postponement. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 406–412, Maui, October 2001.

[132] C. Estrada, J. Neira, and J. D. Tardos. Hierarchical SLAM: realtime accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, August 2005.

[133] J. Castellanos, J. Neira, and J. Tardos. Limits to the consistency of EKF-based SLAM. In *5th Symposium on Intelligent Autonomous Vehicles (IAV)*, Lisbon, Portugal, 2004.

[134] J. Neira and J. D. Tardos. Data Association in Stochastic Mapping Using the Joint Compatibility Test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, December 2001.

[135] A. Vedaldi, H. Jin, P. Favaro, and S. Soatto. KALMANSAC: Robust Filtering by Consensus. In *IEEE International Conference on Computer Vision (ICCV)*, Beijing, China, October 2005.

[136] B. Williams, P. Smith, G. Klein, and I. Reid. Automatic Relocalisation for a Single-Camera Simultaneous Localisation and Mapping System. In *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, April 2007.

[137] G. Qian, R. Chellappa, and Q. Zheng. Robust Structure from Motion Estimation Using Inertial Data. *Optical Society of America*, 18:2982–2997, December 2001.

[138] P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardos. Inertial Aiding of Inverse Depth SLAM using a Monocular Camera. In *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[139] J. Kim and S. Sukkarieh. Real-time implementation of airborne inertial-SLAM. *Robotics and Autonomous Systems*, 55:62–71, 2007.

[140] M. Pupilli. *Particle Filtering for Real-time Camera Localisation*. PhD thesis, University of Bristol, October 2006.

[141] M. Montemerlo and S. Thrun. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002.

[142] M. Montemerlo and S. Thrun. FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003.

[143] M. Calonder. EKF SLAM vs. FastSLAM: A Comparison. Technical report, Computer Vision Lab, ETH Lausanne, 2006.

[144] T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, USA, May 2006.

[145] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa. Accurate image overlay on video see-through HMDs using vision and accelerometers. In *IEEE Virtual Reality Conference (VR)*, pages 247–254, New Brunswick, NJ, USA, March 2000.

[146] G. Qian, R. Chellappa, and Q. Zheng. Bayesian structure from motion using inertial information. In *International Conference on Image Processing (ICIP)*, Rochester, NY, USA, September 2002.

[147] G. Klein and T. Drummond. Tightly Integrated Sensor Fusion for Robust Visual Tracking. In *British Machine Vision Conference (BMVC)*, volume 2, pages 787–796, Cardiff, September 2002.

[148] B. Jiang, U. Neumann, and S. You. A Robust Hybrid Tracking System for Outdoor Augmented Reality. In *IEEE Virtual Reality Conference (VR)*, Chicago, Illinois, USA, March 2004.

[149] M. Aron, G. Simon, and M.-O. Berger. Use of inertial sensors to support video tracking. *Computer Animation and Virtual Worlds*, 18:57 – 68, 2007.

[150] G. Thomas, J. Chandaria, B. Bartczak, K. Koeser, R. Koch, M. Becker, G. Bleser, D. Stricker, C. Wohlleber, M. Felsberg, F. Gustafsson, J. D. Hol, T. B. Schön, J. Skoglund, P. J. Slycke, and S. Smeitz. Realtime Camera Tracking in the MATRIS Project. *SMPTE Motion Imaging Journal*, 116(7/8):266–271, July/August 2007.

[151] 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(2-3):69–79, November 2007.

[152] J. Kim and I.-S. Kweon. Robust feature matching for loop closing and localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3905–3910, San Diego, CA, October 2007.

[153] M. Felsberg and J. Hedborg. Real-time view-based pose recognition and interpolation for tracking initialization. *Journal of Real-Time Image Processing*, 2(2-3):103–116, 2007.

[154] G. Bleser. Entwicklung eines 3D-markerlosen Kamera-Trackingverfahrens zur Echtzeit-Augmented-Reality-Bildsynthese. Master's thesis, Institute for Computational Visualistics, Koblenz University, Koblenz, Germany, October 2004.

[155] H. Wuest, F. Wientapper, and D. Stricker. Adaptable Model-based Tracking Using Analysis-by-Synthesis Techniques. In *International Conference on Computer Analysis of Images and Patterns (CAIP)*, Vienna, Austria, August 2007.

[156] K. Koeser, B. Bartczak, and R. Koch. Robust GPU-Assisted Camera Tracking using Free-form Surface Models. *Journal of Real-Time Image Processing*, 2(2-3):133–147, November 2007.

[157] D. Shreiner, M. Woo, J. Neider, and T. Davis. *OpenGL Programming Language. The official guide to learning OpenGL, Version 2.* Addison-Wesley, 5 edition, 2005.

[158] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1508–1511, Beijing, China, October 2005.

[159] A. J. Davison. Active Search for Real-Time Vision. In *IEEE International Conference on Computer Vision (ICCV)*, Beijing, China, October 2005.

[160] H. Wuest, A. Pagani, and D. Stricker. Feature Management for efficient Camera Tracking. In *Asian Conference on Computer Vision (ACCV)*, Tokyo, Japan, November 2007.

[161] L. Armesto, J. Tornero, and M. Vincze. Fast Ego-motion Estimation with Multi-rate Fusion of Inertial and Vision. *International Journal of Robotics and Research*, 26:577–589, June 2007.

[162] S. You and U. Neumann. Fusion of Vision and Gyro Tracking for Robust Augmented Reality Registration. In *Virtual Reality (VR)*, Yokohama, Japan, March 2001.

[163] S. Wolfram. Mathematica. URL `http://www.wolfram.com/products/mathematica/index.html`. Accessed August 8th, 2008.

[164] G. H. Golub and C. F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, London, 3 edition, 1996.

[165] S. Julier and J. LaViola. On Kalman Filtering With Nonlinear Equality Constraints. *IEEE Transactions on Signal Processing*, 55:2774–2784, 2007.

[166] VRML. Virtual Reality Modeling Language. URL `http://www.w3.org/MarkUp/VRML/`. Accessed August 16th, 2008.

[167] REALVIZ. ImageModeler. URL `http://imagemodeler.realviz.com`. Accessed August 10th, 2008.

[168] M. Pollefeys, L. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. In *International Journal of Computer Vision*, pages 207–232, 2004.

[169] 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 (2-3):81–101, November 2007.

[170] H. Rehbinder and X. Hu. Drift-free attitude estimation for accelerated rigid bodies. In *IEEE International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, May 2001.

[171] T. Harada, T. Mori, and T. Sato. Development of a Tiny Orientation Estimation Device to Operate under Motion and Magnetic Disturbance. *The International Journal of Robotics Research*, 26(6):547–559, 2007. doi: 10.1177/0278364907079272.

[172] J. Montiel and A. Davison. A Visual Compass based on SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1917–1922, Orlando, Florida, May 2006.

[173] K.-R. Koch. *Parameter estimation and hypothesis testing in linear models*. Springer, Berlin Heidelberg New York, 1988.

[174] S. J. Julier and J. K. Uhlmann. Unscented Filtering and Nonlinear Estimation. *Proceedings of the IEEE*, 92:401–422, March 2004.

[175] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, New York, 1999.

[176] J. Montiel, J. Civera, and A. J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. In *Internaltional Conference of the Royal Statistical Society (RSS)*, Belfast, Ireland, September 2006.

[177] F. Ababsa and M. Mallem. Inertial and vision head tracker sensor fusion using a particle filter for augmented reality systems. In *International Symposium on Circuits and Systems (ISCAS)*, Vancouver BC, Canada, May 2004.

[178] G. Hendeby, J. Hol, R. Karlsson, and F. Gustafsson. A Graphics Processing Unit Implementation of the Particle Filter. Technical Report LiTH-ISY-R-2812, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, August 2007.

[179] G. Hendeby, J. Hol, R. Karlsson, and F. Gustafsson. A Graphics Processing Unit Implementation of the Particle Filter. In *European Statistical Signal Processing Conference*, pages 1639–1643, Poznan, Poland, September 2007.

[180] J. Sola, A. Monin, M. Devy, and T. Lemaire. Undelayed initialization in bearing only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005.

[181] D. L. Alspach and H. Sorenson. Nonlinear Bayesian estimation using Gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17(4):439–448, August 1972.

[182] S. Frintrop and A. B. Cremers. Top-down Attention Supports Visual Loop Closing. In *European Conference on Mobile Robots (ECMR)*, Freiburg, September 2007.

[183] E. Eade and T. Drummond. Monocular SLAM as a graph of coalesced observations. In *IEEE International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil, October 2007.

[184] H. Andreasson, T. Duckett, and A. Lilienthal. Mini-SLAM: Minimalistic Visual SLAM in Large-Scale Environments Based on a New Interpretation of Image Similarity. In *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[185] B. Steder, G. Grisetti, S. Grzonka, C. Stachniss, A. Rottmann, and W. Burgard. Learning Maps in 3D using Attitude and Noisy Vision Sensors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, October 2007.

[186] W. Fong, S. Ong, and A. Nee. A Differential GPS Carrier Phase Technique for Precision Outdoor AR Tracking. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Cambridge, UK, September 2008.

[187] J. Stuelpnagel. On the Parameterization of the Three-Dimensional Rotation Group. *SIAM Review*, 6(4):422–430, 1964.

[188] M. D. Shuster. A Survey of Attitude Representations. *The Journal of the Astronautical Sciences*, 41:439–517, 1993.