2016

Visual Odometry and Mapping in Natural Environments for Arbitrary Camera Motion Models

Terzakis, George

http://hdl.handle.net/10026.1/6686

http://dx.doi.org/10.24382/3764

Plymouth University

VISUAL ODOMETRY AND MAPPING IN NATURAL ENVIRONMENTS FOR ARBITRARY CAMERA MOTION MODELS

by

GEORGE TERZAKIS

A thesis submitted to Plymouth University

in partial fulfillment for the degree of

DOCTOR OF PHILOSOPHY

School of Computing and Mathematics

In collaboration with
UTC Aerospace Systems

April 2016

Acknowledgments

I would like to express my gratitude to Phil Culverhouse, Guido Bugmann, Sanjay Sharma and Bob Sutton not only for supervising my work, but most of all, for giving me the opportunity to open the door to a realm of wonderful ideas; fair chances are hard to come-by these days and I am deeply grateful for the one given to me.

A great deal of appreciation goes to fellow PhD students as well as to the university staff directly or indirectly involved in the project for their help and friendship.

Special thanks to Chris Gregory and Mike Durrant from UTC Aerospace Systems for their interest in the research as well as for their help by providing us with quality inertial measurement units.

Finally, I am grateful to Tatiana for having read these lines a few years from now and to Valentina for her support during this forthcoming period. Clearly looking back at my retirement, I am happy to be still able to change the future and have a shot at making it happen, hoping that I will not lose hope in the - not so distant - past.

Author's Declaration

At no time during the registration for the degree of *Doctor of Philosophy* has the author been registered for any other University award without prior agreement of the Graduate Sub-Committee.

Work submitted for this research degree at the Plymouth University has not formed part of any other degree either at Plymouth University or at another establishment.

This study was financed with the aid of a studentship form the *Engineering and Physical Sciences Research Council* (*EPSRC*).

Relevant scientific seminars and conferences were attended at which work was presented and several papers were published.

The author has published the following papers in international peer-reviewed journals:

- Terzakis, G., et al. (2015). Fitting multiple projective models using clustering-based Markov chain Monte Carlo inference. Image and Vision Computing **33**: 15-25.
- Terzakis, G., et al. (2014). On quaternion based parametrization of orientation in computer vision and robotics. Journal of Engineering Science and Technology Review **7(1)**: 15-24.

Finally, the author has presented the following papers in conferences:

- Terzakis, G., et al. (2011). Unconstrained Visual Planar Odometry in Natural or Texturally Unstructured Scenes. Postgraduate Conference for Computing, Applications and Theory (PCCAT). Plymouth University, UK**:** 17-23.

Word count of main body of thesis: 53246

GEORGE TERZAKIS

Signed: ………………………………………..

Date: ..………………………………………….

PHD EXAMINERS

External:	Dr. Manolis Lourakis	(Foundation for Research and Technology - Hellas)
Internal:	Prof. Roman Borishyuk	(Plymouth University)

PHD SUPERVISORY TEAM

Director of Studies:	Dr. Phil Culverhouse	(Plymouth University)
2nd Supervisor:	Dr. Guido Bugmann	(Plymouth University)
3d Supervisor:	Dr. Sanjay Sharma	(Plymouth University)

Abstract

This is a thesis on outdoor monocular visual SLAM in natural environments. The techniques proposed herein aim at estimating camera pose and 3D geometrical structure of the surrounding environment. This problem statement was motivated by the GPS-denied scenario for a sea-surface vehicle developed at Plymouth University named *Springer*. The algorithms proposed in this thesis are mainly adapted for the Springer's environmental conditions, so that the vehicle can navigate on a vision based localization system when GPS is not available; such environments include estuarine areas, forests and the occasional semi-urban territories.

The research objectives are constrained versions of the ever-abiding problems in the fields of *multiple view geometry* and *mobile robotics.* The research is proposing new techniques or improving existing ones for problems such as scene reconstruction, relative camera pose recovery and filtering, always in the context of the aforementioned landscapes (i.e., rivers, forests, etc.). Although visual tracking is paramount for the generation of data point correspondences, this thesis focuses primarily on the geometric aspect of the problem as well as with the probabilistic framework in which the optimization of pose and structure estimates takes place. Besides algorithms, the deliverables of this research should include the respective implementations and test data for these algorithms in the form of a software library and a dataset containing footage of estuarine regions taken from a boat, along with synchronized sensor logs.

This thesis is not the final analysis on vision based navigation. It merely proposes various solutions for the localization problem of a vehicle navigating in natural environments either on land or on the surface of the water. Although these solutions can be used to provide position and orientation estimates when GPS is not available, they have limitations and there is still a vast new world of ideas to be explored.

Contents

List of tables and figures

List of terms

AR	Augmented reality
AUV	Autonomous Unmanned Vehicle
DOF	Degrees of Freedom
EIF	Extended Information Filter
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
GPS	Global Positioning System
HMM	Hidden Markov Model
ID	Identifier
IEKF	Iterated Extended Kalman Filter
IF	Information Filter
IKF	Iterated Kalman Filter
IMU	Inertial Measurement Unit
KF	Kalman Filter
KKT	Karush-Kuhn-Tucker (conditions)
LK	Lucas – Kanade
LM	Levenberg – Marquardt
LS	Least Squares
MCMC	Markov Chain Monte Carlo
MER	Mars Exploration Rover
MLE	Maximum Likelihood Estimate
MRF	Markov Random Field
MSE	Mean Squared Error

OpenCV	Open source Computer Vision library
PD	Positive Definite (matrix)
PF	Particle Filter
PnP	Perspective-n-Point
PSD	Positive Semi-definite (matrix)
PTAM	Parallel Tracking and Mapping
RANSAC	RANdom SAmpling Consensus
SFM	Structure From Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SO	Special Orientation (group)
SURF	Speeded-Up Robust Features
SVD	Singular Value Decomposition

Chapter 1

Introduction

This is a thesis on vision based localization and mapping using a single camera in outdoor natural environments. The methods discussed and proposed herein aim at estimating camera pose and 3D geometrical structure of the surrounding environment in order to be used in localization systems for vehicles navigating in natural environments.

The main objectives, assumptions and constraints in this study were shaped by a scenario in which a sea vessel named “Springer” would have to navigate exclusively on visual aids for a short period of time. Springer (Figure 1.1) is a GPS-guided small catamaran developed at Plymouth University for the exploration of coastal waters such as river estuaries. The current study investigates and proposes solutions that could successfully aid the vehicle in keeping track of its waypoints in situations wherein positioning feedback is intermittent or entirely absent. This scenario directly incurs numerous significant challenges for the vision based localization system such as unconstrained camera motion and unstructured image content (i.e., no repeatability in recognizable patterns). Furthermore, the Springer does not employ active sensors, a fact that precludes the use of ranging sensors, which would have been a significant advantage in terms of robustness and accuracy of the vision based localization system.

This thesis proposes frameworks of existing and new algorithms that could potentially be used for vehicle/camera localization problems wherein the environmental limitations resemble the ones that apply to the Springer’s case. It should be stressed that the majority of existing solutions in literature concern conditions that, in many ways, do not apply in the case of Springer: In particular, these solutions usually concern indoor environments with recognizable unique features, involve additional ranging or global positioning sensors, incorporate knowledge of accurate motion priors and landmark models; moreover, data collection becomes an easy task when performed indoors (e.g. a desk or an office workspace is much more convenient than a boat cruising the river). It therefore becomes evident that the limitations imposed on the research problems that this thesis is dealing with are very challenging and the methods proposed, as well as the results and conclusions drawn, are of significance to the community, especially for researchers dealing with problems that dwell in the realm of field robotics.

1.1 The “Springer”

The “Springer” is an unmanned sea-surface vehicle (Figure 1.1) custom-built by Plymouth university technicians. It uses two battery powered DC motors in each hull providing differential thrust drive. It is also equipped with 3 standard GPS receivers and 3 magnetic compasses for global positioning and single-axis orientation feedback respectively.

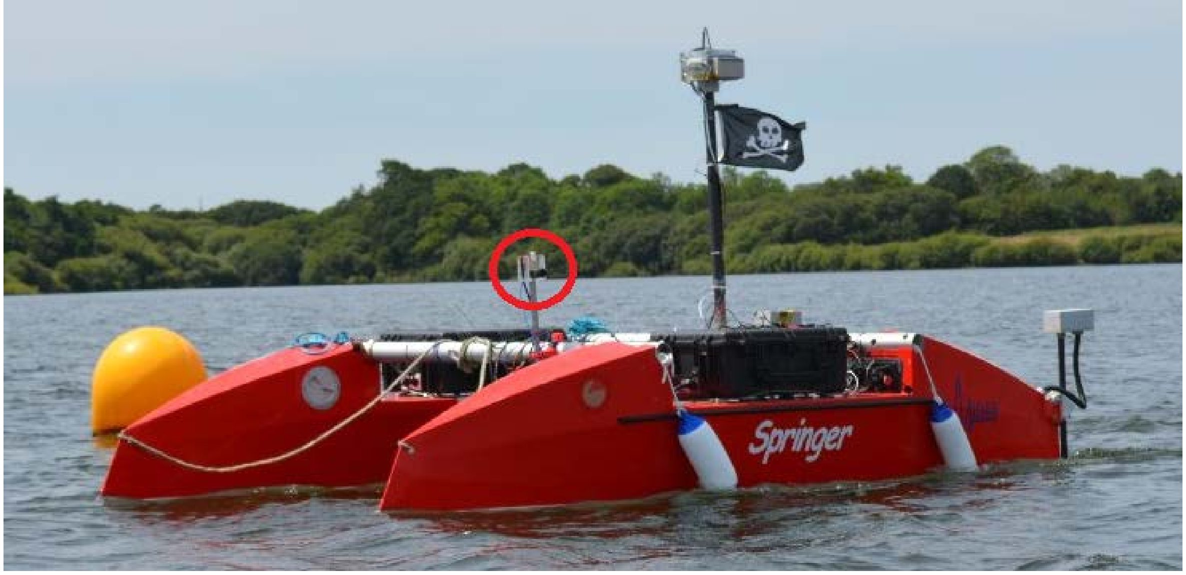

Figure 1.1. The unmanned surface vehicle known as “Springer”. The camera (circled in red) is mounted on the left hull, pointing sideways.

The original version of Springer (Naeem, Xu et al. 2010) was able to execute simple navigation missions throughout a programed list of locations (waypoints) using global positioning satellite (GPS) feedback. Typical scenarios would involve search-and-rescue missions or collection of water samples. To achieve its objectives, the vehicle utilizes a simple planar motion model with constant speed. Thus, navigation becomes a matter of controlling the angle (yaw) about the vertical axes, provided high-accuracy azimuth readings from the magnetic sensors. Satisfactory proximity to a waypoint is decided by means of a distance threshold from the vehicle’s current position. Once the vehicle has successfully “hit” a waypoint, it then adjusts its orientation objective as the offset from its current heading to the direction from its centroid (as a 2-dimensional object) to the next waypoint location. Sensor sampling and controller actions are executed in 1s intervals to synchronize with the GPS reception events.

In the updated version of the Springer, it was decided that strictly monocular vision (as opposed to stereo vision) would be employed during the GPS-denied runs. This decision was based on the fact that scene depth varies significantly and could therefore easily produce degenerate disparity in standard stereo rigs. Although the Springer has plenty of room for a second camera that could potentially form a large baseline vector with the first and such an idea would be advantageous in terms of efficient scene structure recovery, it was however abandoned primarily due to the limited (at the time, USB3 cameras were not available and fast high definition image capturing in real time would require very specialized and expensive equipment – the Springer is cost effective by definition) computational resources. Camera calibration is typically performed offline, either in the lab or onboard the Springer. Calibration “on the fly” was not a primary goal of this research and therefore is not addressed in this thesis.

1.2 The necessary assumptions imposed by the Springer’s design objectives on the vision based localization problem

Transferring the existing principles and techniques for vision - based localization to natural environments and particularly to ones that involve the seabed in the camera view requires a rigorous problem statement which will take into consideration the fundamental limitations incurred by the nature of the environment. The current section enlists these limitations in the form of conditions and constraints which will be assumed valid throughout this thesis, unless stated otherwise.

1.2.1 Non-degenerate visibility of coastal structure

The Springer’s “habitat” involves river estuarine areas and lakes. Evidently, if the vehicle is going to plan its motion relying partially or entirely on visual feedback, it follows that the existence of some visible structure on the coast must be present in the captured images. This requirement emerges from the fact that the seabed cannot be used to reliably extract motion information. Thus, the useful portion of the image concerns only the region depicting the estuary and therefore capturing degenerate views where all coastal structures are depicted as a blob is not an unlikely configuration, provided that the distance to the shore varies significantly. Subsequently, throughout this thesis it is assumed that the video sequences do not contain images in which the coast vanishes over the horizon. Figures 1.2 and 1.3 illustrate typical examples of degenerate and "useful" depictions of the estuary in captured images. It should be noted that degenerate coastal depictions pose a significant problem in data collection because they occur very often in the captured sequences.

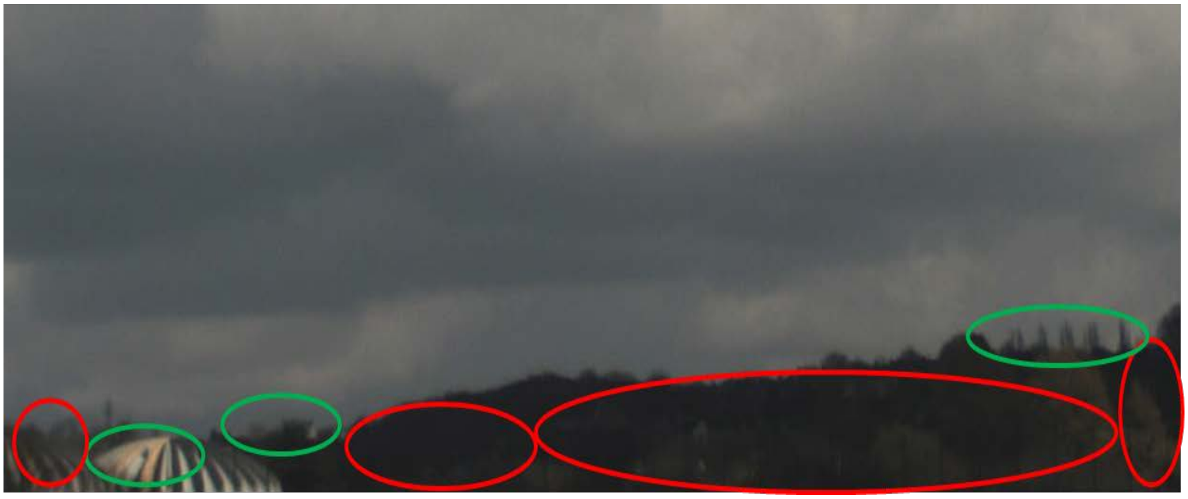

Figure 1.2. Example of a degenerate image of the coast. The structures circled in red are blurred and insufficient for reliable motion tracking. A few valid useful regions are indicated in green, mostly because they involve sharp edge corners.

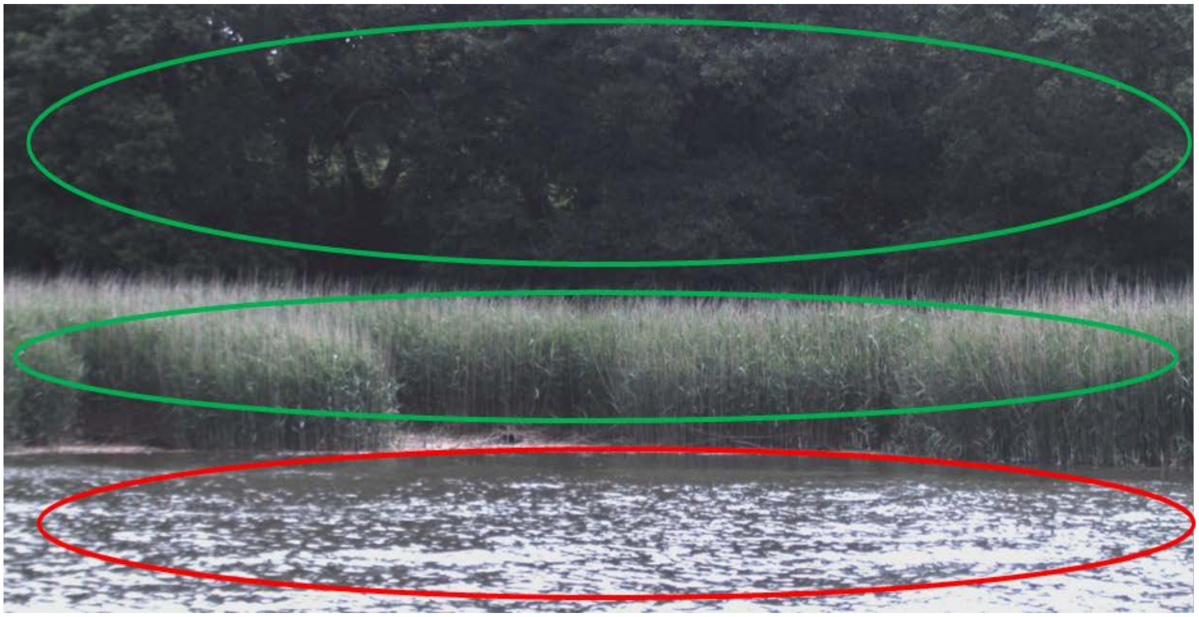

Figure1.3. Example of useful estuarine depiction. Coastal vegetation (circled in green) covers a large portion of the image and the structure is relatively clear.

1.2.2 Global positioning feedback unavailable or intermittent

The primary assumption in this problem concerns the lack of global positioning feedback for sufficiently long periods of time to render navigation impossible without any other means of localization.

1.2.3 Unknown motion dynamics

The Springer navigates by controlling its yaw angle. It does not make use of a differential thrust model. This leaves very little margin for accurate motion prediction because of two reasons: a) Motion control events are extremely slow (1s) compared to the camera frame rate (24-30 Hz) and, b) The planar motion assumption is not representative of the changes in orientation of a vehicle cruising on the surface of the water; although this coarse approximation is acceptable for GPS based navigation in calm waters, it presents however large prediction errors in image space regardless of weather conditions and therefore cannot be employed as a motion model in the context of filtration.

A model of motion dynamics is an essential part of standard algorithms for vehicle localization and in the case of Springer would have allowed for relatively accurate predictions of the vehicle's position thereby improving the tracking accuracy significantly. The fact that no model of vehicle dynamics is available shifts the focus of the problem of vision-aided navigation from the Springer and generalizes it into one of camera pose estimation in the very same environmental conditions defined by the Springer's primal objectives.

1.2.4 Passive sensing only

The Springer does not use active sensors. This means that laser ranging sensors, ultrasonic transducers, etc. cannot be used for depth recovery. This makes the problem significantly more ill-posed, since nearly all motion parameters have to be extracted from the camera alone. Although the Springer is equipped with high accuracy magnetic sensors, their utility as aids to vision based motion estimation is outperformed by inertial sensors which can provide accurate readings in 3D as opposed to the relatively slower 2D readings of compasses. Typically, gyros provide high accuracy readings and can be used as reliable predictors of relative orientation.

Other passive sensors include accelerometers and the Springer's thruster encoders. However, these sensors were not employed due to the tradeoff between the actual gains and the effort required to cope with their shortcomings in practice. In particular, motor encoders and accelerometers would be more useful in conjunction with a model of the vehicle's motion dynamics. However, without a motion model, acceleration readings not only have to be "cleared" of gravity and rotational velocity "contaminants", but they also must be integrated twice in order to produce a displacement estimate, which in practice entails high uncertainty. On the other hand, motor encoder readings do not necessarily reflect actual motion in the water as they do on land in the case of differential drive mobile robots; moreover, the readings reflect thrust and therefore, in order to be converted into a velocity estimate, they must be considered in conjunction with water drag, which actually requires the knowledge of vehicle-specific motion dynamics. To recapitulate, accelerometers and motor encoders could be useful, but they require special modeling effort, which was not amongst the priorities of the current research.

1.3 Intuition and research objectives

Generally stated, this is a problem involving vision based localization from image sequences of natural scenes. An intuitive depiction of the vision-aided localization principle suited for a sea surface vehicle is illustrated in Figure 1.4; the camera is turned to the coast so that it may capture the visible structure in order to track landmarks, and subsequently apply algorithms for pose estimation.

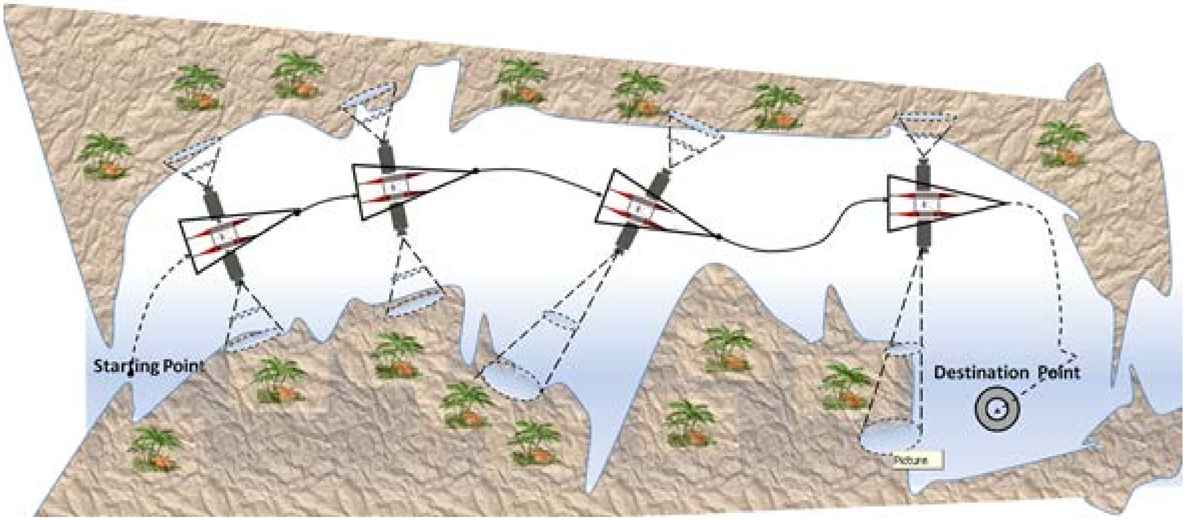

Figure 1.4. The concept of vision aided navigation using visible coastal structure for landmark tracking.

1.3.1 An abstract view of the problem

Regardless of limitations, the problem of vision based navigation is still open and more innovative ideas should be anticipated in the near future. The goal of the research presented in this thesis was to examine the full spectrum of techniques in computer vision, geometry, optimization and machine learning and combine methodologies in order to establish one or more frameworks for vision based localization systems that provide reliable position and orientation feedback for a reasonable period of time and under reasonable assumptions.

1.3.2 Objectives

It is very important to stress that vision based/aided navigation of vehicles, although a simple notion to grasp, it however entails a very broad range of disciplines and scientific areas. This thesis will explain the background, before applying them to the research problem. It is therefore prudent at this stage to state the objectives and the types of research questions that will be addressed in this thesis in a more intuitive manner and in regards to the general objective which is the achievement of vehicle localization in natural coastal environments using a single camera. These objectives can be summarized in the following:

- ***Choosing appropriate method for tracking.*** At a primitive level, methodologies for recovery of motion through vision rely on the detection, tracking and recognition of landmarks at acceptable levels of accuracy and repeatability. This thesis does not aim at investigating new algorithms for feature tracking and detection but rather at choosing suitable techniques from the existing state-of-the-art pool that perform best in the context of the problem specifics (i.e., natural environments). A brief technical introduction to the fundamentals of tracking and landmark detection and how such information is used by state of the art algorithms for localization and mapping is given in Chapter 2.
- ***Understanding and making use of the existing state-of-the-art techniques in 3D computer vision, geometric modelling and robotics.*** This thesis focuses predominantly on the aspects of the problem that are pertinent to geometry, machine learning, optimization and robotics (filtering) and the adaptation and/or improvement of existing algorithms for triangulation, 3D reconstruction, camera localization and mapping. Chapter 3 reviews popular techniques in the *geometry of multiple views* in order to pre-empt their use in integrated solutions presented in Chapters 6, 7 and 8. Details of these techniques are given in the appendices.
- ***Design of a framework in which algorithms will be synchronized and interconnected.*** Part of the problem concerns implementation details and how algorithms are linked together in a framework orchestrated by a finite-state machine. These frameworks correspond to the conceptual pipeline of the integrated solution; in other words, it is the operational network by which the various algorithms are interconnected in order to produce the pose and map estimates. This thesis examines two different framework models and elaborates on the trade-off between efficiency and ease of implementation in the context of different conditions (e.g., slow/fast camera motion, weather conditions, use of gyroscopic sensors, etc.).
- ***Propose new algorithms for relative pose estimation, scene reconstruction and optimization.*** A significant part of the research questions addressed in this thesis have to do with the proposition of novel algorithms and/or improvements of existing ones under a limited scope in comparison to the general problem statement (i.e., solutions to specific sub-problems pertinent to vision aided localization and mapping) where applicable in the aforementioned proposed frameworks. These

problems involve the following techniques: Stochastic Filtering (Chapters 2, 6, 7, 8), 2-view reconstruction (Chapters 3, 6, 8), the perspective-n-point problem (Chapters 6, 8), non-linear optimization and sensor fusion (Chapters 7, 8) and the parametrization of camera pose (Chapters 4, 6, 7, 8).

- ***Deliver the implementation of the algorithms described in this thesis in the form of executable code.*** Although not explicitly stated in the general problem statement, it can be however inferred from the aforementioned goals that an important deliverable should be a software code archive. The fulfilment of the listed objectives, in part or in whole, involves the implementation of the respective algorithms in some programming language. These algorithms should become publically available to the community for use and/or improvement either as a library, code snippets or pseudo-code.
- ***Deliver video sequences of estuarine and natural landscapes synchronized with sensor logs***. Provided the distinctiveness of the environmental conditions associated with the Springer, another very important deliverable concerns the image sequences on which the results of the proposed methods will be obtained. To the best of the author's knowledge, there are very few such datasets in regards to natural environments and most likely, none in terms of sequences of natural coastal areas captured from the vantage point of a surface vehicle cruising in the water.

1.4 Examples of modern integrated solutions for real-time or offline camera pose and mapping

Although it will become clear in the following chapters, it is worth noting here that motion recovery through vision is equivalent to the estimation of the locations of the tracked landmarks in the real world. Thus, nearly all algorithms involving camera pose estimation typically output a map estimate containing the 3-dimensional locations of the tracked landmarks, a process widely known in the vision community as *structure from motion* (SFM). In pure vision applications, the reconstruction of the tracked features is more important, while in robotic applications, obtaining the camera pose estimate is a first priority, albeit the map can be also useful for obstacle avoidance or even geometric scene recognition and loop closure.

1.4.1 Offline structure recovery from multiple views

A very popular project aspiring to recover 3D models of outdoor touristic sites around the world using multiple pictures of the same scene taken from different vantage points is *photo tourism* (Snavely, Seitz et al. 2006). Although this is an offline application which processes images solely for the sake of 3D reconstruction, the underlying principles involving the optimization of camera pose and scene structure over all camera poses and scene features apply identically. Figure 1.5 illustrates the sparse reconstruction of the Colosseum and the respective estimated camera poses.

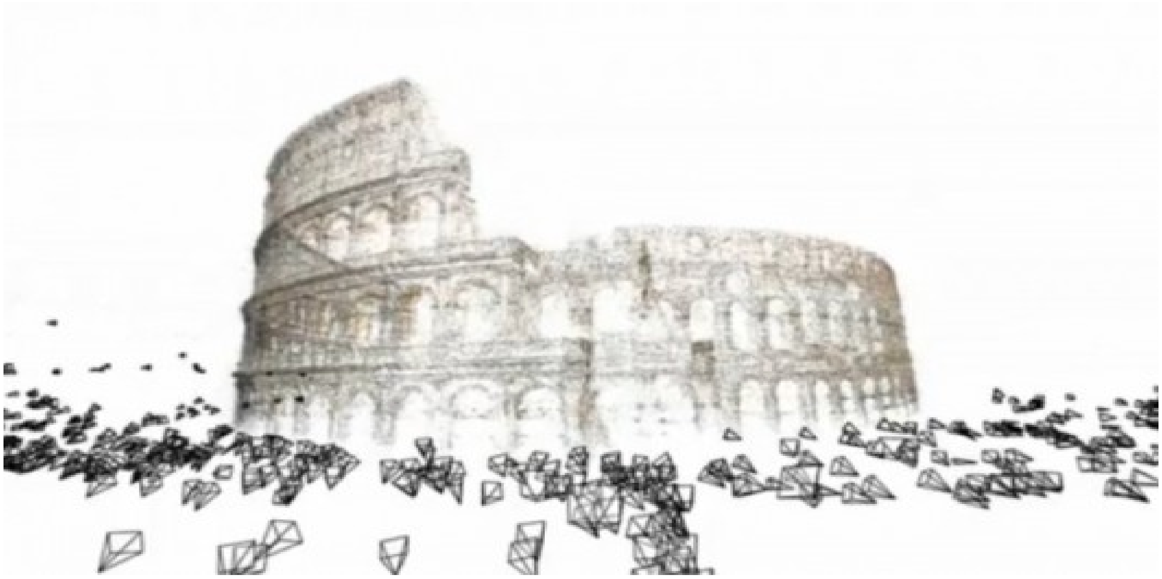

Figure 1.5. An illustration of a 3D reconstruction of the Colosseum with Photo tourism (Snavely, Seitz et al. 2006)[1].

1.4.2 Real-time pose estimation and mapping

A very significant implementation for real-time structure from motion is *parallel tracking and mapping* (PTAM) (Klein and Murray 2007, Klein and Murray 2009). In principle, PTAM integrates sparse point tracking and camera motion estimates in order to acquire scene geometry in real time, while a separate thread uses these estimates as a starting point in a global refinement of the map along with all the previous camera poses. Although PTAM was mostly tested with indoor sequences, it is relatively efficient in outdoor scenes with moderate depth.

[1] Permission to reproduce screenshots of photo tourism granted by Noah Snavely.

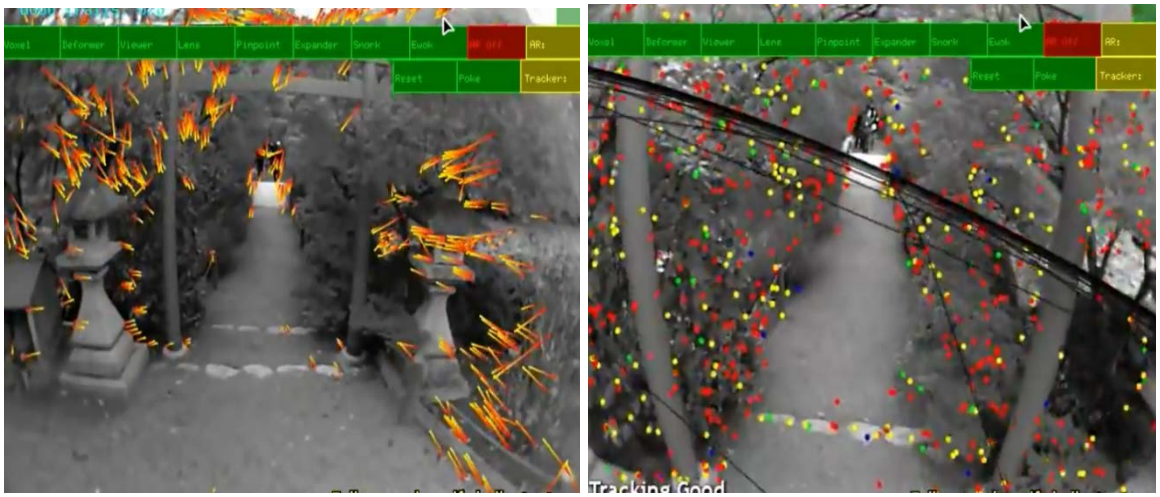

Figure 1.6. Outdoor mapping with PTAM (Klein and Murray 2007, Klein and Murray 2009)[2].

Figure 1.6 illustrates PTAM running on an outdoor sequence; preliminary detection and tracking of features is shown on the left. PTAM was originally intended for augmented reality applications, but it soon became a popular general tool for camera pose estimation and mapping. The image on the right in Figure 1.6 illustrates a plane detected in the scene (shown as a grid) on which animation enhancements can be added.

1.4.3 Vision based localization in field robotics

Numerous early methods have been proposed (primarily for motion recovery) in the context of robotic applications, most of which are typically environment and vehicle oriented (Tomasi and Kanade 1992, Irani, Rousso et al. 1997, Baumela, Agapito et al. 2000, Qian and Chellappa 2001, Qian, Chellappa et al. 2001). Although some of the solutions and techniques introduced in the late 1990s and early 2000s appeared to be promising, significantly fewer eventually became part of the standard implementations of today (Nistér, Naroditsky et al. 2004, Scaramuzza, Fraundorfer et al. 2009, Scaramuzza and Fraundorfer 2011). It is worth noting here that the majority of outdoor visual SLAM applications involve the use of a single camera and henceforth any reference to a vision based localization method will be assumed as such, unless explicitly stated otherwise.

A significant amount of work has been invested on flying autonomous unmanned vehicles (AUVs) utilizing ground facing cameras in outdoor environments. Amongst the

[2] Permission to reproduce PTAM captured images granted by David Murray.

prominent pieces of work on flying AUVs is that of Dunkley, Sturm, Engel and Cremers with ground facing cameras (Dunkley, Engel et al. , Engel, Sturm et al. 2014). The group approaches the problem either by employing a fairly accurate model of the vehicle's dynamics, or by using a depth sensor to obtain scale; at the same time, a gyro is used to provide reliable attitude estimates and camera tracking results are used as measurements for position estimation. Another accurate and efficient visual odometry algorithm for Mars rovers was proposed by Lentaris, Stamoulias and Lourakis (Lentaris, Stamoulias et al. 2015). Although most of the results of Sturm's group were obtained from indoor sequences, very similar approaches (Achtelik, Brunet et al. 2012, Faessler, Fontana et al. 2015) have been successfully tested in large-scale outdoor environments. The ground facing camera arrangement offers several advantages in comparison to totally unconstrained attitude.

There have been relatively few scenarios involving vision based localization of vehicles in completely natural environments without the aid of high-resolution depth sensors. Such is the case of two Mars exploration rovers (MER), equipped with a stereo rig pointing to the ground from a skewed angle (Maimone, Cheng et al. 2007). Due to its high computational cost, the visual odometry system was used only for short-term corrections. Another recent representative method for visual odometry and mapping in natural terrains was introduced by Konolige and Agrawal (Konolige, Agrawal et al. 2011). In this case, a ground vehicle uses a stereo rig in conjunction with a gyro for accurate odometry estimation in the desert.

1.5 Summary and structure of the remaining thesis

This introductory chapter aimed at transforming a very general problem statement into a list of coherent and sensible research objectives while paying absolute respect to the vastness and magnitude of the underlying principles.

It is important to stress the fact that the "Springer" research problem can only be examined and improved in terms of its sub-problems and cannot be solved in a one-off fashion, mainly because it actually requires solutions and innovations which lie along independent research directions, ranging from problems in standard control engineering all the way to elaborate solutions tangent to stochastic processes and geometry.

The remaining chapters are structured as follows:

Chapter 2. A brief overview of the fundamental concepts in robotic localization and mapping.

Chapter 3. Introduction to the geometry of multiple views with the primary focus on the recovery of camera relative pose and scene structure from two views.

Chapter 4. Two prominent representations of orientation are discussed: Euler's axis-angle formalism and stereographic projection.

Chapter 5. Frameworks for the implementation of SLAM: The disjoint and overlapping scene paradigms.

Chapter 6. Algorithms for relative pose odometry as quick solutions to the localization problem. Also, subsidiary techniques are discussed, such as the "orthogonal Procrustes" method in the context of a planar odometry example and the use of gyroscopic data in 3D relative-pose odometry.

Chapter 7. Introduction to the GraphSLAM algorithm for least squares and an algorithm for fusion of 3D gyroscope data with 2D global position measurements.

Chapter 8. This chapter is about the full map-based localization problem in the context of natural environments. The perspective-n-point problem is discussed with details in this chapter because it is the most fundamental algorithm in map-based visual odometry (and this why the topic was not discussed in Chapter 3). Also, details on the implementation of bundle adjustment are provided in the form of algorithms. In overall, two methods for visual odometry and mapping are presented: a) An overlapping scene localization framework using only camera input and, b) A disjoint scene localization framework for a single camera aided by gyroscopic inputs. These two frameworks combine all the methods and algorithms proposed in this thesis. Results are provided in sequences from natural environments with emphasis on coastal areas from the vantage point of a moving van in the woods or boat in the water.

Chapter 9. Conclusion of this thesis. A brief synopsis of contributions in regards to the original objectives and suggestions for further research.

The appendices contain detailed algorithms, proofs, properties, derivations and descriptions of the concepts presented in Chapters 2, 3, 4, 6, 7 and 8.

Chapter 2

Fundamental concepts in vision based localization and mapping

This chapter provides a brief overview of the fundamental concepts that govern the algorithms and techniques covered throughout this thesis. The process of robotic localization follows the common paradigm of pose prediction and refinement through measurements. In other words, a model of motion is used to predict the vehicle’s position and thereafter this impression is refined by receiving feedback from the environment through sensors. In the case of vision aided navigation, sensory feedback corresponds to camera input in the form of special features such as points or entire patches in images; these features have special characteristics that make them distinguishable across a set of images of the same scene.

In most robotic applications, the locations of landmarks are known a priori, so that when the robot identifies them using its sensors, it can infer its location based on the measurement model associated with the sensor. However, where visual landmarks are concerned, prior knowledge may not be the case, especially in outdoor environments with scenes involving significant clutter; thus, landmark features are actually detected online while cruising and are thereafter tracked in subsequent images of the scene, thereby providing measurements of relative pose instead of absolute. It follows that the map is populated incrementally as the vehicle is cruising. Points are most commonly used as features and will be the only type of visual landmark employed in the algorithms described in this thesis; although it is possible to detect patches as well as points, these patches however, are not expected to have unique appearance in natural landscapes and therefore their repeatability in identification will be significantly diminished. Similarly, offline landmark acquisition is not generally plausible in such landscapes, since not only scene backgrounds have striking similarities in natural backgrounds, but most importantly, it would be a very difficult task to manually “cherry-pick” visual landmarks on the coastline in sequences taken from a boat.

2.1 Simultaneous localization and mapping

Localization and mapping is one of the most prominent problems in field robotics and has attracted the attention of a great number of researchers in the recent years (Smith and Cheeseman 1986, Dissanayake, Newman et al. 2001, Fidaleo and Medioni 2007, Thrun and Leonard 2008). Modern techniques for *simultaneous localization and mapping* (SLAM) regard the position and orientation of a robot as a stochastic quantity and their aim is to estimate the underlying distribution. Thus, the pose (i.e., position and orientation) of the robot $x_t \in \mathbb{R}^6$ (3 parameters for orientation and 3 for position) at time $t \in \mathbb{Z}$ (time is discrete) is a statistical estimate in terms of the maximum likelihood optimality criterion. In quite the same way, the map of the environment comprises a list of landmark locations M (also referred to as the *map*), that are also treated as random variables. The pose x_t and map[3] M at time *t* comprise the *state* of the SLAM algorithm.

The goal of SLAM is to obtain the maximum likelihood (MLE) estimate of the pose of the robot and the state of the environment (map) at time *t*, given a number of measurements $m_1, .., m_t$ and process control inputs[4] $u_0, .., u_{t-1}$. Execution of SLAM can either be characterized as *offline* or *online*. Offline SLAM algorithms estimate all poses $x_0, .., x_t$ and the map *M* given all past and present measurement and control vectors:

$$\begin{bmatrix} \hat{x}_{0:t} \\ \widehat{M} \end{bmatrix} = \underset{x_{0:t}, M}{\operatorname{argmax}} \, p(x_{0:t}, M | m_{1:t}, u_{0:t-1}) \tag{2.1}$$

where the subscript notation $t_1 \colon t_2$ in a sequence denotes all time instances from t_1 to t_2. In online (real-time) SLAM, only the most recent pose x_t and the map *M* are estimated given all past and present measurement and control input vectors:

[3] Time indices may apply to the map, but they are usually omitted. In effect, the map obeys a hard equality transition constraint from the previous time instance to the next.

[4] The reader should bear in mind that sometimes, for the sake of simplicity, control inputs may be omitted from the argument list of the transition function as they are not stochastic quantities and can be regarded as part of the function itself.

$$\begin{bmatrix}\hat{x}_t \\ \widehat{M}\end{bmatrix} = \underset{x_t,M}{\operatorname{argmax}}\, p(x_t, M|m_{1:t}, u_{0:t-1}) \tag{2.2}$$

The SLAM posterior $p(x_t, M|m_{1:t}, u_{0:t-1})$ is also known as *state belief* at time *t* (Thrun, Burgard et al. 2005). Evidently, the state belief is the marginal of x_t over $x_{0:t-1}$ in the joint conditional distribution of equation (2.1).

The assumptions behind the modern SLAM paradigm are depicted in the Bayes network of Figure 2.1. The network clearly implies that SLAM is a discrete stochastic process, in which the pose evolves by means of a transition law $p(x_t|x_{t-1}, u_{t-1})$ while the process gains information from the environment according to a measurement likelihood, $p(m_t|x_t, M)$. Moreover, the process has the Markovian property, which states that x_t is conditionally dependent only on x_{t-1}; also, each measurement is conditionally dependent only on the current state and therefore independent of all previous (and future) measurements. The transition distribution typically represents the uncertainty in the motion of the robot, while the measurement likelihood corresponds to a “noisy” model of landmark perception.

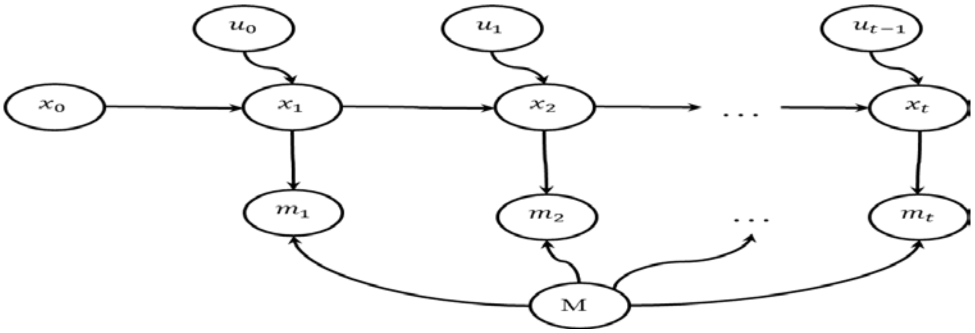

Figure 2.1. The SLAM paradigm depicted as a Bayes network.

2.1.1 The Gaussianity assumption

Assuming that both pose transition and measurement likelihood are normal distributions, then the joint distribution represented by the Bayes network of Figure 2.1 will also be normally distributed. Thus, mean vector and covariance matrix of the state belief can be copied directly from the mean and covariance of the joint posterior. Provided that the pose prior, motion transition law and measurement likelihood are normal distributions, the joint estimate is obtained by minimizing a generally non-linear quadratic function,

$$
\begin{aligned}
q(x_{0:t}, M) = {} & (x_0 - \mu_0)^T \Sigma_0^{-1} (x_0 - \mu_0) \\
& + \sum_{k=1}^{t} \Big[\big(x_k - g_{k-1}(x_{k-1}, u_{k-1})\big)^T R_k^{-1} \big(x_k - g_{k-1}(x_{k-1}, u_{k-1})\big) \\
& + \big(m_k - f_k(x_k, M)\big)^T Q_k^{-1} \big(m_k - f_k(x_k, M)\big) \Big]
\end{aligned}
\tag{2.3}
$$

where $x_0 \sim N(\mu_0, \Sigma_0)$ is the pose of the vehicle at $t = 0$, R_t and Q_t are the covariance matrices of the pose transition and measurement likelihood respectively at time t; g_t is a function that captures the motion dynamics of the robot at time $t - 1$ and f_t is a function that models the relationship between measurement, pose and map at time t. Figure 2.2 presents a conceptual illustration of SLAM indicating the correspondence between the quadratic constraints of equation (2.3) and the perception of landmarks from the various positions as well as the transition from one pose to another. Landmarks are appearing as stars and the poses of the robot as triangles. The dashed ellipses contain landmark groups that are visible to the vehicle's sensor(s) at a specific time instance. Dotted lines associate groups of landmarks with pose vectors and correspond to quadratic constraints associated with landmark observations. The thick arrows indicate the transition from one pose to another and correspond to quadratic constraints related to motion transitions.

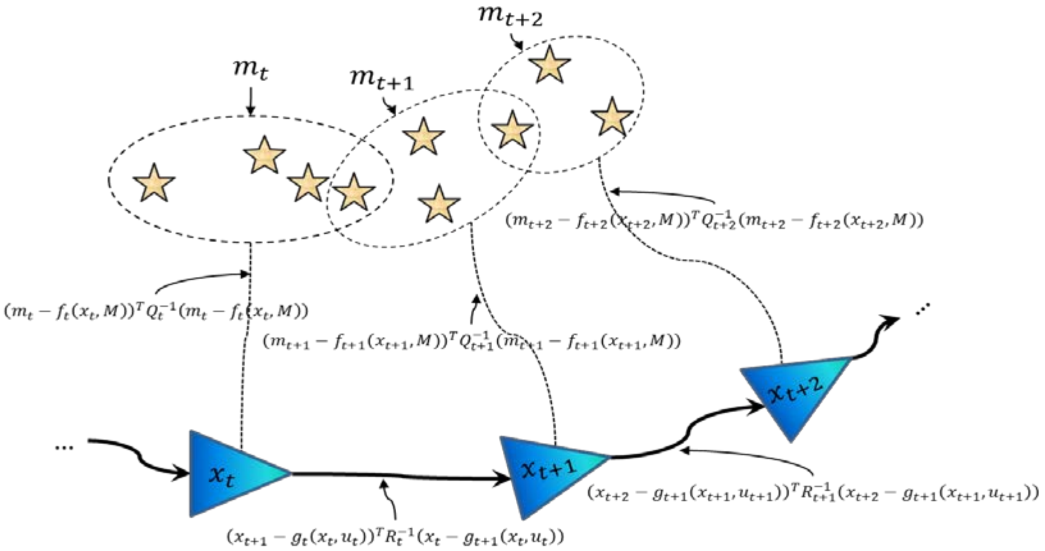
m_{t+2}
m_t
m_{t+1}
$(m_{t+2} - f_{t+2}(x_{t+2}, M))^T Q_{t+2}^{-1}(m_{t+2} - f_{t+2}(x_{t+2}, M))$
$(m_t - f_t(x_t, M))^T Q_t^{-1}(m_t - f_t(x_t, M))$
$(m_{t+1} - f_{t+1}(x_{t+1}, M))^T Q_{t+1}^{-1}(m_{t+1} - f_{t+1}(x_{t+1}, M))$
...
x_{t+2}
...
x_t
x_{t+1}
$(x_{t+2} - g_{t+1}(x_{t+1}, u_{t+1}))^T R_{t+1}^{-1}(x_{t+2} - g_{t+1}(x_{t+1}, u_{t+1}))$
$(x_{t+1} - g_t(x_t, u_t))^T R_t^{-1}(x_t - g_{t+1}(x_t, u_t))$

Figure 2.2. Graphical illustration of SLAM.

The Kalman filter. Most robotic applications are concerned only with the most recent location of the robot and therefore have no need for state estimates that belong to the past.

An obvious solution to obtaining the current pose and map estimates is to optimize $q(x_{0:t}, M)$ over the present and all past states, a process which entails heavy computational burden due to the number of variables involved. A very efficient alternative to the computation of the state belief is the *Kalman filter (Kalman 1960, Smith and Cheeseman 1986)*. The Kalman filter (KF) is a real-time approach to recursively obtaining the state belief from the current measurement likelihood and the previous state belief. The advantage here is that the information of all past measurements and state transitions is encompassed in one single marginal distribution and therefore computations involve only three distributions in total. In other words, this is a quadratic function optimization problem with only three terms, as opposed to the number of terms in the cost function of equation (2.3).

The KF computes the new state posterior in two steps: The first step involves *prediction*, which computes the marginal distribution of the state vector over all previous control vectors and measurements (also known as *predicted state belief*); the next step is the *measurement update* (sometimes called *innovation*), involving the computation of the state belief from the current measurement and the predicted state belief.

In its standard formulation, the KF concerns linear transition and measurement models. To cope with this limitation, the extended Kalman filter[5] (EKF) was proposed; the EKF employs the standard KF formulas on linearized approximations of the transition and measurement models respectively. A significant drawback of the EKF is that approximations in the measurement step will almost certainly produce sub-optimal estimates attributed to the fact that the filter is primarily designed to do the prediction in one single step; to work around this issue, the iterated extended Kalman filter (IEKF) was introduced; in the IEKF, the measurement update is implemented as a Gauss-Newton iteration (Bell and Cathey 1993) over the function,

$$q_t(x_t, M) = \left(m_t - f_t(x_t, M)\right)^T Q_t^{-1}\left(m_t - f_t(x_t, M)\right) + (x_t - \bar{\mu}_t)^T \bar{\Sigma}_t^{-1}(x_t - \bar{\mu}_t) \qquad (2.4)$$

where $\bar{\mu}_t$ and $\bar{\Sigma}_t$ are the mean and covariance matrix of the pose in the predicted state belief.

The information filter. The information filter (IF), is an alternative implementation of the KF from the aspect of state belief representation. In particular, the posterior is

[5] The reader should bear in mind that henceforth, any reference to the Kalman filter in this thesis will imply the extended Kalman filter.

represented using the so-called *Fisher parameters* (information matrix and vector) of a normal distribution as opposed to the traditional moment parametrization (covariance matrix and mean vector). The IF, like the KF, computes the new state belief in two steps, the prediction and measurement update. In contrast to the KF, the bulk of computations involving the inversion of the information matrix now migrate to the prediction step, while the measurement update involves simply a few matrix multiplications. For further reading on information filters the reader is deferred to *probabilistic robotics* (Thrun, Burgard et al. 2005). In direct analogy to the EKF, for cases in which the motion and/or measurement model are non-linear, there exists the extended information filter (EIF) which uses linearized approximations of the transition/measurement functions in order to obtain solutions for the SLAM posterior.

One of the advantages of IFs over KFs, is the fact that the canonical parametrization of a Gaussian can encapsulate a quadratic constraint by means of direct entries to the information matrix and vector. Thus, it is possible to populate the information matrix and vector with an arbitrary number of constraints and obtain marginals at will. In other words, IFs offer a great deal of flexibility in terms of when and how a number of poses and/or landmarks can be marginalized-out. It is therefore possible to perform semi-offline SLAM filtering using an approach which became known as GraphSLAM (Thrun and Montemerlo 2006). Another important feature of IFs in SLAM is the operation known as *sparsification* of the posterior involving the disengagement of “weak” links in the information matrix. Thus, sparsification essentially concerns the elimination of correlations that develop through time between distant landmarks (through pose variables), so that the information matrix becomes block-diagonal and easier to invert (Thrun, Liu et al. 2004).

The particle filter. A very popular alternative to the KF and IF for arbitrary distributions is the particle filter (PF). Under this approach, the posterior is represented using a set of samples, called *particles*. The predicted state belief is obtained by sampling the transition conditional probability, conditioned on randomly picked particles of the posterior. In the measurement step, the samples are tagged with *importance weights* reflecting their measurement likelihood. Thus, the new posterior is obtained by resampling according to the importance weights.

Of particular interest to the visual SLAM community is a hybrid PF-KF approach by Montemerlo and Thrun called *FastSLAM* (Montemerlo, Thrun et al. 2002). In FastSLAM, the particles in the posterior are moments of Gaussian distributions corresponding to updated (KF-fashion) predicted state particles for each landmark. The rationale behind FastSLAM relies on the observation that landmarks are conditionally independent of each other given the sequence of poses until the present time. Thus, it is possible to apply the KF measurement step independently for each landmark and predicted pose particle. Although the algorithm seemingly has an update complexity $O(NK)$ where N is the number of particles and K is the number of landmarks, the authors observe that the possible posteriors (which are Gaussians) per pose particle are arranged in the leaves of a balanced binary tree and therefore accessing each of these posteriors requires time logarithmic in K. It turns-out that the overall complexity of an update can be reduced to $O(NlogK)$ time.

2.2 Visual features as landmarks

In this thesis, visual landmarks correspond to image points classified as distinguishable *features* by means of criteria associated to their invariance to rotation, translation, scaling and brightness. The selection process involves the maximization of local image criteria that determine the invariance of the surrounding image patch. Occasionally, for each feature, a *descriptor vector* of statistical measures that characterize the surrounding patch is generated. Descriptors are used for feature matching across a pair of images, called *reference image* and *query image*. In visual SLAM, the positions of matched features in the reference and query image are used to triangulate the position of the landmark in the real world. Further matching/tracking of the reference features in subsequent camera frames will result in additional quadratic constraints in the SLAM filter, thereby reducing uncertainty of the pose of the camera/vehicle and the map. Figure 2.3 not only illustrates the concept of matched features from in multiple images taken from a moving camera, but also demonstrates the emerging geometric relationships between the image positions, the camera centers and their actual locations in the real world.

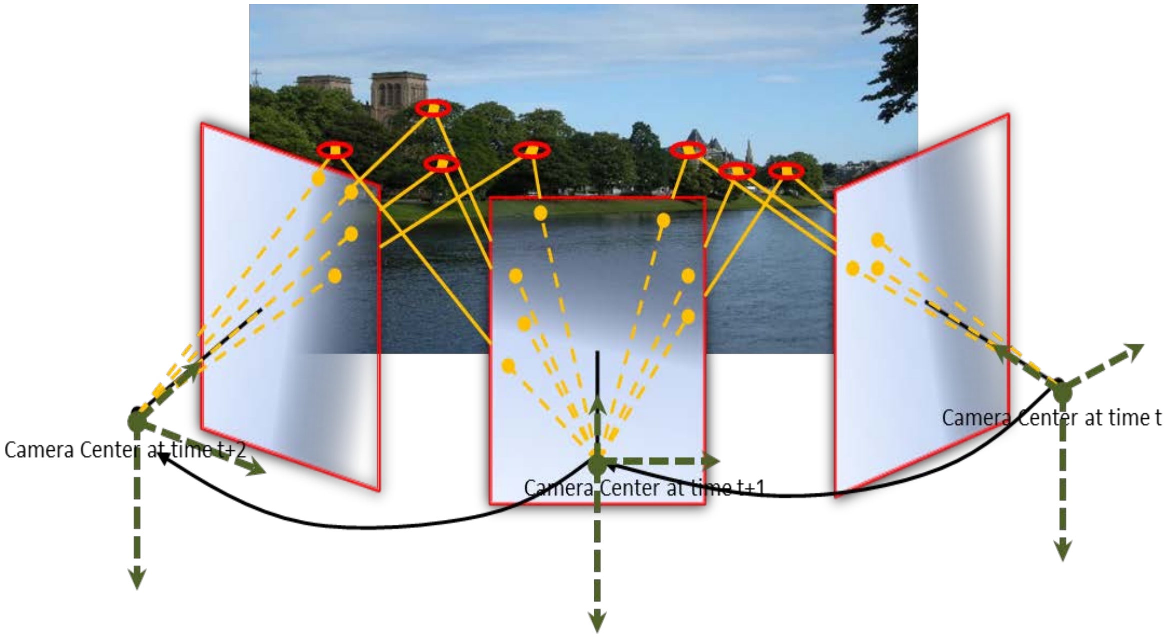

Figure 2.3. Point-features tracked across multiple images taken from a moving camera.

2.2.1 Feature detection

A wide variety of visual feature types have been proposed in literature during recent years with each case presenting certain advantages over the others with respect to the existing matching methods and the application context (Babbar, Bajaj et al. 2010). A few representative examples of popular feature types are Harris corners (Harris and Stephens 1988, Shi and Tomasi 1994), the scale invariant feature transform (SIFT) (Lowe 1999), speeded-up robust features (SURF) (Bay, Tuytelaars et al. 2006) and features from accelerated segment test (FAST) (Rosten and Drummond 2006). In particular, Harris corners, as well as the SURF and FAST features became extremely popular in the visual SLAM research community, mainly because of the advantageous trade-off between detection time and quality of the features.

As the camera moves away from a scene, the tracked features will gradually move out of the frame; therefore, as time progresses, it becomes necessary to detect new sets of features. Since SLAM applications typically operate in real time, feature detection time is an important factor to consider when designing the algorithm. In fact, FAST features provide one of the best known detection times thus far and have been successfully used in PTAM (Klein and Murray 2007) an augmented reality application (AR) which quickly became popular amongst visual SLAM researchers. FAST is the standard feature detection method in this thesis.

2.2.2 Feature matching

Feature matching refers to the process of finding pairs of features in the reference and the query image such that, for some given metric, the distance between the respective descriptors is the smallest possible. To enhance the efficiency of the distance-based matching, Lowe has proposed a rejection criterion based on the ratio of the distances of the first and second nearest neighbor of the matching point. Illustrations of such matchings are shown in Figures 2.4 and 2.5 using FAST and SURF features respectively. The query image is shown on the left; the red quadrilateral captures the perspective distortion estimated from the matched feature locations on the facade of the church.

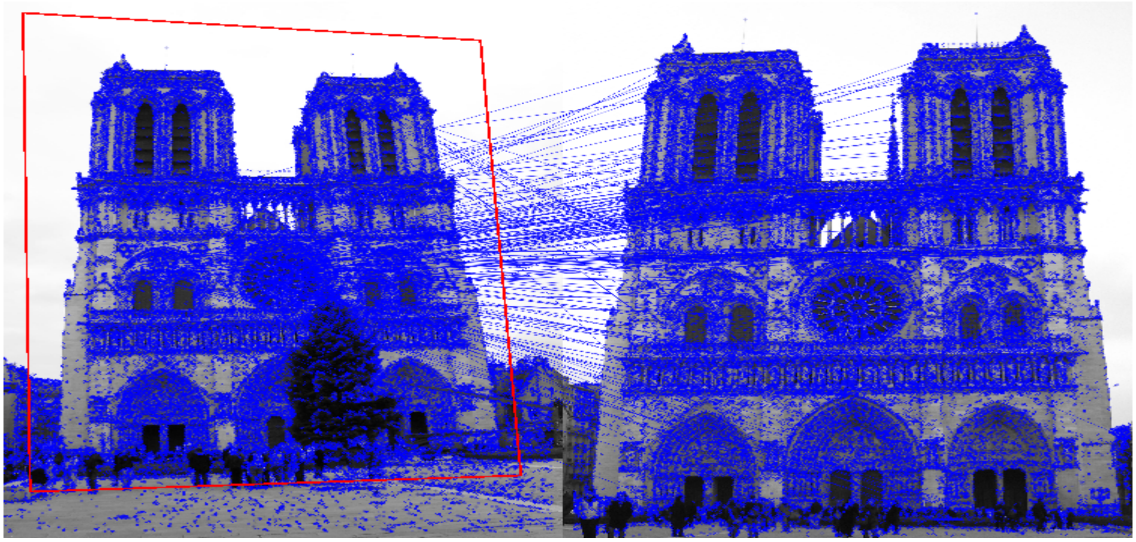

Figure 2.4. Feature matching between two pictures of a church using the FAST descriptors.

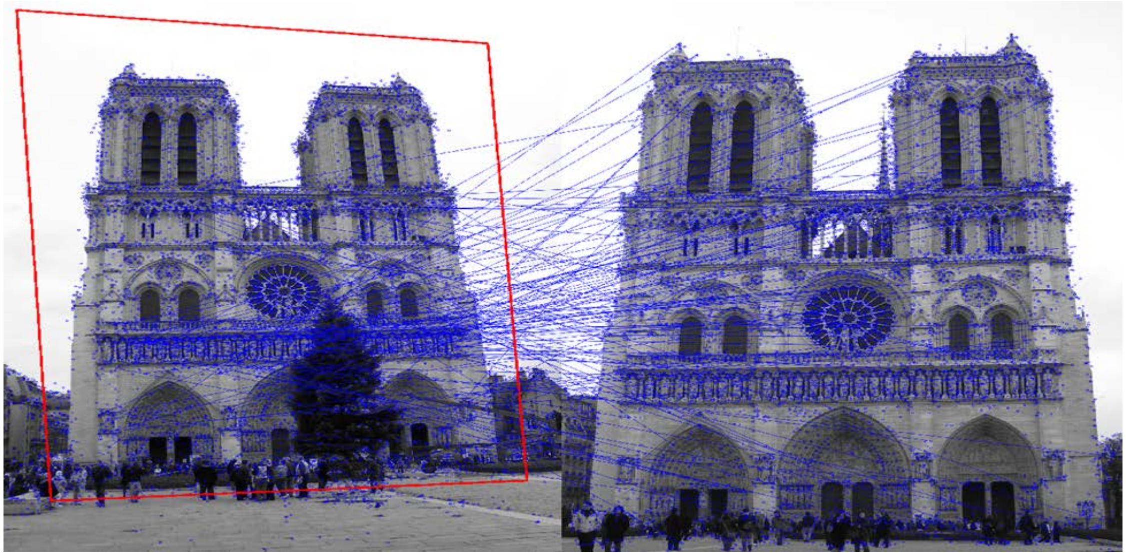

Figure 2.5. Feature matching between two pictures of a church using SURF descriptors.

The detected FAST features in Figure 2.4 clearly outnumber the detected SURF features in Figure 2.5. This difference is attributed not only to the user-defined parameters of the detectors, but also to the difference in the quality of descriptors. Moreover, it would make sense that some detectors may outperform others with respect to a certain types of invariance. Figures 2.4 and 2.5 suggest that both FAST and SURF perform similarly when the query image differs from the reference by minor projective distortion. However, as shown in Figures 2.6 and 2.7, SURF clearly outperforms FAST when the query image differs by a significant amount of scaling and rotation from the reference. Images were lifted from the Emgu open source library (Emgu 2013) examples; Emgu is a C# “wrapper” for the OpenCV libary (Bradski 2000). The cereal box is contained in the query image at a much smaller scale. In Figure 2.6, although the FAST detector has generated many features, the matching is very poor (notice that the estimated perspective distortion quadrilateral is practically a line); in Figure 2.7 however, the features on the cereal box have matched correctly with the ones on the scaled box in the query image.

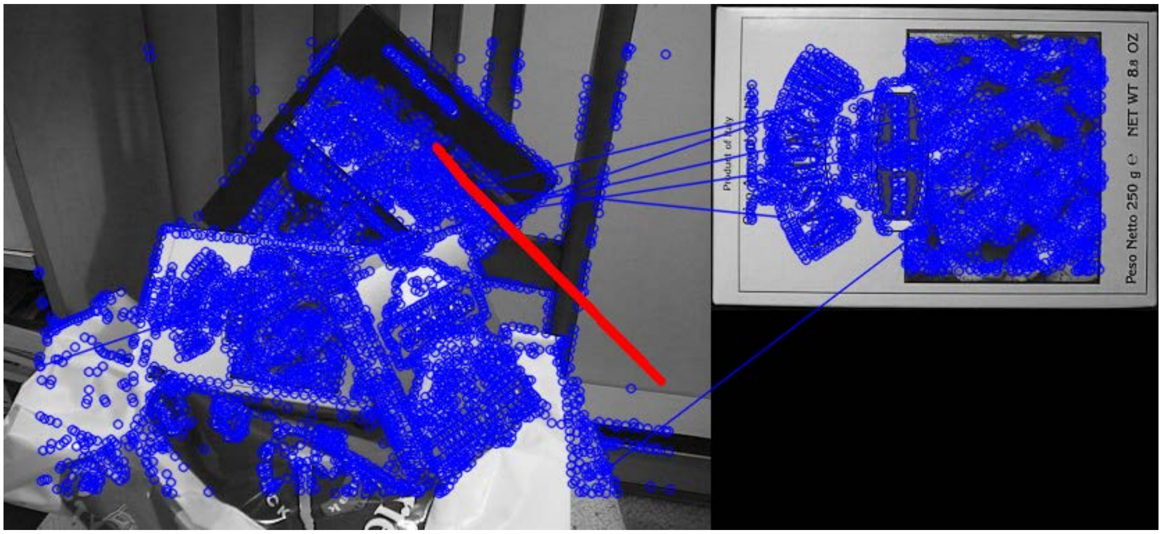

Figure 2.6. Feature matching between images of a cereal box using FAST.

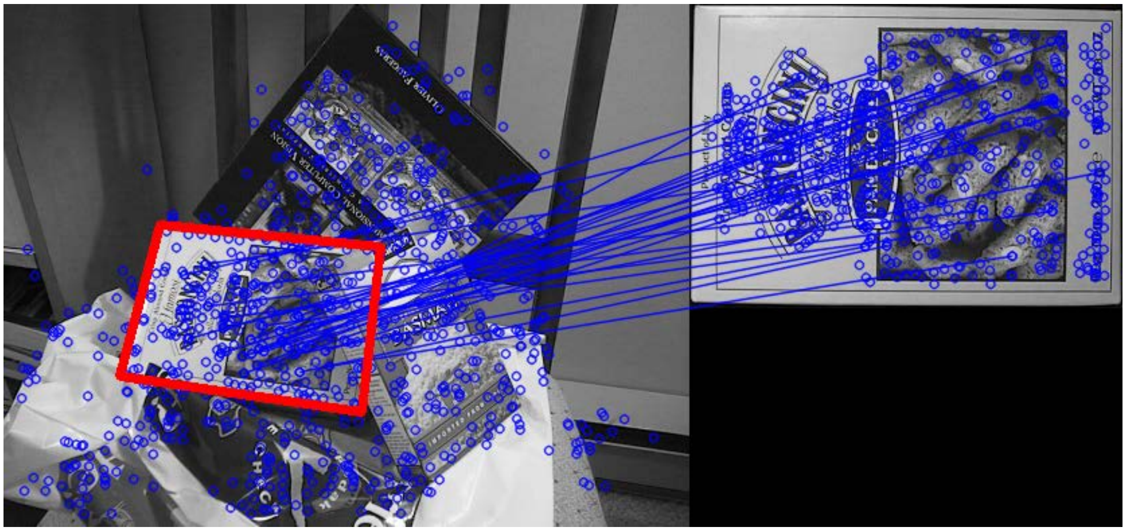

Figure 2.7. Feature matching between images of a cereal box using SURF.

Landmark matching is the main pillar of SLAM algorithms. It is imperative that features are matched as accurately as possible in order to achieve significant reduction of the uncertainty in the state estimate throughout the measurement step. Feature matches in various images of the same scene such as the ones in Figures 2.4-7, are independent of the measurements in intermediate images in the sequence and therefore can decrease the state uncertainty. A significant drawback of independent feature matching is that, even a single mismatch can easily mislead the entire measurement update when the erroneous match lies at a long distance from the true match, since the SLAM quadratic cost function tends to exaggerate large errors. It should also be noted that another practical disadvantage of independent descriptor matching is execution time.

A standard strategy to avoid the egregious mismatches produced by matching descriptors in entire images is local patch searching in the query image. One way of doing this is to use the predicted pose to estimate a region in which the matching feature is expected to appear and thereafter search for a match in this region. Such an approach is employed in PTAM. Others, such as Davison (Davison, Reid et al. 2007) prefer to use entire patches as features, but this is a solution that typically works in man-made environments (mostly indoors) where these patches are distinctively detectable with high repeatability.

2.2.3 Sparse optical flow and the KL tracker

A traditional alternative to descriptor matching in local patches is *optical flow* field estimation (Horn and Schunck 1981). The goal of optical flow methods is to estimate the motion of image pixels through time. The underlying assumption in this approach is that the reference and query images are considered to be temporal versions of the very same visual content that changes its spatial configuration through time. In other words, it is assumed that pixels simply change their locations from one image to another without changing their intensity. This is the fundamental assumption in optical flow estimation known as the *brightness constancy assumption*.

Consider the smooth pixel intensity function $I(x(t), y(t), t)$ at location $(x(t), y(t))$ at time t. Provided that the intensity of this point is preserved through time, it follows that I is constant and therefore, $\frac{dI}{dt} = 0$. Expanding the derivative of $I(x(t), y(t), t)$ using the chain rule yields the so-called *gradient constraint equation*:

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t}\frac{dt}{dt} = 0 \Leftrightarrow \frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y = -\frac{\partial I}{\partial t} \tag{2.5}$$

The temporal derivatives of the position, $\left(\frac{dx}{dt}, \frac{dy}{dt}\right)$ are just the velocity (optical flow) vector, (v_x, v_y) at $(x(t), y(t))$. Clearly, optical flow cannot be recovered by a single gradient constraint equation, since there are two unknowns involved. The usual assumption that further constrains the unknown optical flow is that neighboring pixels share the same velocity. This way, the unknown flow can be recovered by optimizing a functional that involves multiple gradient constraint equations. In principle, the functional used to solve optical flow is a quadratic of the form,

$$E(v_x, v_y) = \sum_x \sum_y w(x,y) \left(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} \right)^2 \tag{2.6}$$

where $w(x, y)$ is a weighting function (usually Gaussian) assigning importance to each quadratic term inversely proportional to the distance from the center of the patch.

Under favorable conditions, the brightness constancy assumption is approximately true; however in general, it is a weak assumption and can easily lead to divergent results. To cope with reasonable violations of the brightness constancy assumption, one may consider the gradient constraint equation as the first order Taylor approximation of the difference between the brightness values $I(t)$ and $I(t + 1)$. Thus, the quadratic term of (2.6) is a linear approximation of a smooth function and therefore, the optimization problem can be solved using the Gauss-Newton method for improved accuracy. In this context, the slightly modified cost function at step *k* would be,

$$E_k(\delta v_x, \delta v_y) = \sum_x \sum_y w(x,y) \left(\frac{\partial I}{\partial x} \left(v_x^{(k-1)} + \delta v_x \right) + \frac{\partial I}{\partial y} \left(v_y^{(k-1)} + \delta v_y \right) + \frac{\partial I}{\partial t} \right)^2 \tag{2.7}$$

where $\left(v_x^{(k)}, v_y^{(k)}\right)$ is the optical flow estimate at step k and $(\delta v_x, \delta v_y)$ is the difference between the current estimate and the previous one. Evidently, $\left(v_x^{(0)}, v_y^{(0)}\right)$ is the solution of equation (2.6). Note here that the cost function in equations (2.6-7) can be modified to register local affine deformations (Lucas and Kanade 1981).

There have been many variations in the formulation of optical flow estimation proposed in literature. However, the cornerstones in optical flow theory are the seminal papers by Horn - Schunck (Horn and Schunck 1981), and Lucas - Kanade (Lucas and Kanade 1981). In fact, the iterative solution of equation (2.7) introduced by Lucas and Kanade is known as the LK (or KL) tracker and is employed in almost all modern optical flow implementations. A very reliable and popular technique that implements the LK tracker by estimating optical flow throughout image pyramid levels is the so-called *pyramidal LK tracker* (Bouguet 2001). A more recently proposed formulation of the optical flow cost function involves fitting polynomials in local image patches (Farnebäck and Westin 2006).

Optical flow estimation offers a fast alternative to local patch descriptor matching. In natural scenes such as trees, bushes or forests, the flow is generally robust between successive frames (Figure 2.8) due to the uniqueness of the neighborhood of the feature inside a patch and the background clutter such as grass, foliage, etc. The remaining outliers can be handled by enforcing geometric constraints between the matched features. The same outlier rejection principles apply to local descriptor matching.

Mismatches in optical flow tracking are usually associated with fast camera motion (motion blur), abrupt variations in scene brightness and the so-called *aperture problem*; bad tracking due to the aperture problem occurs when the spatial gradient in the neighborhood of a point has a constant direction along an edge, thereby yielding a degenerate system of gradient constraints.

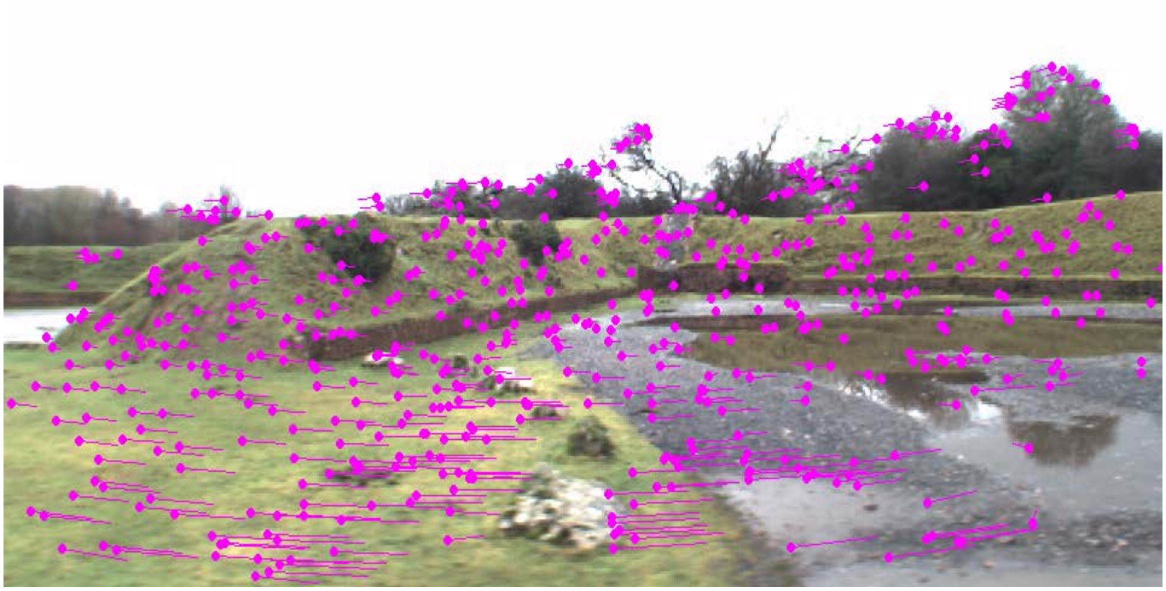

Figure 2.8. Sparse optical flow vectors between the 1st and 6th frame of the sequence estimated using the pyramidal LK tracker.

2.3 Summary

2.3.1 Optical flow based tracking vs Feature matching in natural scenes

The imagery produced by natural landscapes such as parks, forests, river banks, etc., has rich texture which practically minimizes the frequency of occurrence of the aperture problem in optical flow tracking. Moreover, scene illumination is generally uniform and constant, thereby favoring local patch methods in general. Thus, a relatively slow-moving camera should produce fairly robust tracking results for use in a visual SLAM algorithm.

On the other hand, outdoor environments do not favor long-term matching of individual point descriptors; although *good features to track* (Shi and Tomasi 1994) are abundant in natural scenes, these features cannot be reliably matched in long-term, due to the similarity of textures such as foliage, tree-trunks, distant shorelines or hills, etc. and extreme scale variations. In overall, one should expect reliable short-term tracking using optical flow or other means of local search, without however being able to match interest points in the long run. Most of the algorithms presented in this thesis use the pyramidal LK tracker to obtain image measurements; although the SLAM algorithm assumes that these measurements are independent of the previous ones, in practice however, trailing tracking is applied with short-lived features (i.e., discarded after 3-5 frames) in order to ensure minor drifts.

Although independent feature detection and matching in query images is a viable option, it however poses certain practical problems, the most important of all being the possibility of erroneous matches that lie very far apart in image space. As described earlier in this chapter, such matches have a drastically detrimental impact on the state estimate. Furthermore, detection and matching time are sometimes restrictive even for offline executions of visual SLAM. The local feature matching alternative, although appealing, it however demonstrated in practice that, it is all too often possible, features that were distinct in the reference image, would get the exact same match in the query image; the frequent occurrence of these mismatches also affects the subsequent pose estimate significantly.

2.3.2 Using the OpenCV KL tracker: Pros and cons

As mentioned in Chapter 1, the work described in this thesis focuses primarily on the methodologies tangent to geometric vision and SLAM while it makes use of existing solutions for feature detection and tracking as implemented in the OpenCV library. These implementations impose limitations in the results of the aforementioned methodologies in numerous ways. Although the SLAM framework tries to compensate for these limitations, there are always issues that require a custom-made approach. Custom-made implementations for feature detection and tracking are beyond the scope of this thesis, but they are a significant part of the solutions proposed and therefore should be a priority in the context of future research.

The results of the algorithms in the following chapters were obtained using the OpenCV implementation of the pyramidal LK tracker. The algorithm is fast and highly reliable for moderate camera motion and can track features accurately across 2-3 frames, depending on the velocity magnitude. Although SLAM assumes independent measurements, in practice however feature measurements are obtained by means of trailing tracking across no more than 1-2 consecutive frames. In other words, the LK tracker instead of tracking the original patch to the current image, it uses the previous frame (or the one before the previous frame) as reference. Under this approach, drift is likely to appear in the measurements and, although it may not become apparent immediately, it will eventually incur a long-term error primarily on the pose estimate as well as on the scale and position of the most recent map-points. This somewhat unorthodox workaround is a result of an important limitation imposed by the implementation of the tracker in OpenCV. In particular, it is not possible to track individual patches, but rather a set of locations from one image to another. As will be noted in the conclusions of this research, it is imperative to modify the functionality of the LK tracker in order to achieve better integration with the SLAM framework and push the accuracy of the proposed algorithms to their full potential.

Chapter 3

The geometry of two views

Depictions of the real world on camera are typically modelled as perspective projections. In particular, tracked features in a video sequence can be regarded as 2D perspective projections of 3D points on the image plane through the camera center. Under this projection model, a set of geometrical properties emerges from pairs of images of the same scene captured from different locations, also known as the geometry of two views. The geometry of two views provides not only the basis for measurement models in SLAM, but also for the formulation of algorithms and rules for carrying-out tasks such as outlier rejection, estimation of relative camera pose, detection of degenerate projective configurations and most importantly, scene reconstruction. This chapter gives an appropriate introduction to the theory and the methods that will be used and examined throughout the rest of this thesis.

3.1 The pinhole camera model

The pinhole camera model is a reliable theoretical description of how the real world is depicted onto an image. Consider a plane π and a point O in 3D Euclidean space. Let $(x_{\pi}, y_{\pi},)$ be a basis of the plane π; also, let $(x_O, y_O, z_O\ )$ be the three unit vectors of a frame attached to O such that $x_O = x_{\pi}$ and $y_O = y_{\pi}$ and z_O is parallel to the normal of the plane. The distance of O from π along z_O is f. For a 3D point M, the intersection point m^E of the ray that passes through M and O and the plabe π is the projection of M on π with respect to the center of projection, O. Figure 3.1 illustrates the principles of perspective projection.

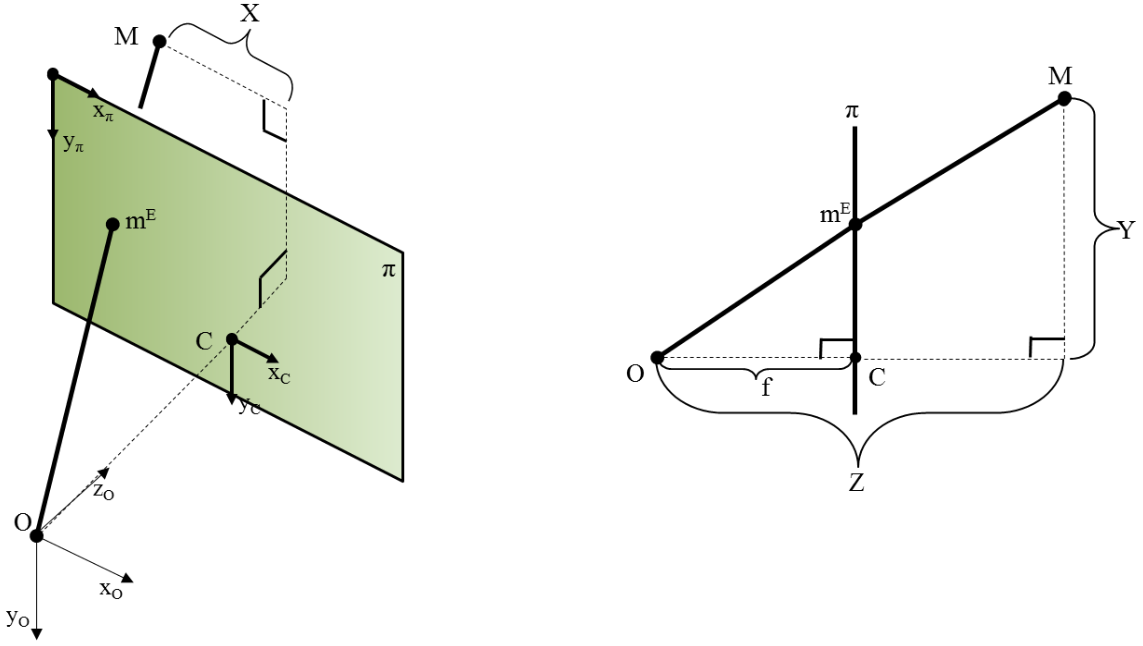

Figure 3.1. Euclidean projection of a 3D point on a plane with respect to the projection center.

The coordinate vector $m^E$of the Euclidean projection of M on the plane π can be computed in terms of the frame (x_π, y_π) and the distance *f* as follows:

$$\tilde{m}^E \propto \underbrace{\begin{bmatrix} f & 0 & \kappa_x \\ 0 & f & \kappa_y \\ 0 & 0 & 1 \end{bmatrix}}_{L} M = LM \tag{3.1}$$

where ~ denotes the homogeneous representation of a vector, ∝ denotes equality up to scale, κ_x, κ_y are the coordinates of C with respect to (x_π, y_π), L is dubbed the *matrix of Euclidean projection parameters* and $M = [X \quad Y \quad Z]^T$ is the world point in terms of the coordinate frame (x_O, y_O, z_O) attached to O. In terms of camera projection, π is known as the *image plane*, O is the *camera center* (also, *focus of projection*), the triad (x_O, y_O, z_O) is the respective *camera frame* and *f* is the *focal length*. Finally, the ray defined by the camera center and the unit vector z_O is known as the *optical* axis.

To relate the Euclidean projection on a plane with the respective location of the point on the image, the horizontal and vertical ratios s_x and s_y of pixels per length-unit associated with the camera are required. By plugging these ratios into equation (3.1), we may directly associate the image point p with its 3D location M as follows:

$$\tilde{p} \propto \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{S} \underbrace{\begin{bmatrix} f & 0 & \kappa_x \\ 0 & f & \kappa_y \\ 0 & 0 & 1 \end{bmatrix}}_{L} M = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{K \\ Matrix\ of \\ intrinsic \\ parameters}} M = KM \tag{3.2}$$

where $f_x = s_x f$, $f_y = s_y f$, $c_x = s_x \kappa_x$, $c_y = s_y \kappa_y$ are the camera *intrinsic parameters* and $K = SL$ the respective matrix as the product of the scale factors (matrix S) with the Euclidean projection parameters (matrix L). Please note here that the intrinsic parameters may additionally include three *distortion coefficients* used to undo radial distortion present in the captured image. Throughout this thesis, it will be assumed that radial distortion is either rectified or negligible and therefore the term *intrinsic parameters* will refer only to the matrix of equation (3.2).

3.1.1 Calibration

Camera calibration refers to the process of estimating the camera intrinsic parameters. The most popular method to calibrate a camera is the checkerboard method (Zhang 1999). The usual implementation of this idea relies on the estimation of the planar homography that removes perspective distortion from the checkerboard plane and scales it up to match the dimensions of the image (Duane 1971).

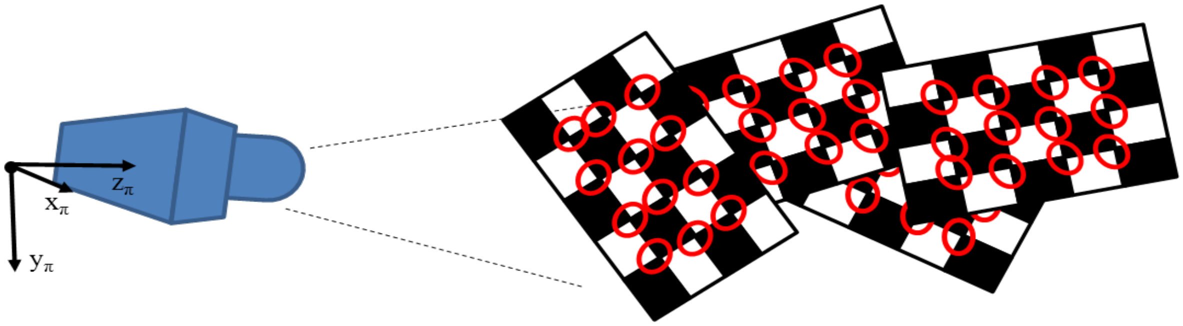

Figure 3.2. Camera calibration using multiple views of a checkerboard.

The corners formed by the points of contact between neighboring black squares (see red circles in Figure 3.2) have distinguishable characteristics in the image and can be detected very efficiently by a corner detector. A sequence of images of the board in various poses with respect to the camera local frame (x_π, y_π, z_π) is captured. For a single image of a 5×4 chessboard, there exist 4×3 such corners yielding 24 equations for the 8 unknown elements of the homography matrix (the 9th is set to 1). Following a series of manipulations, the camera intrinsic parameters can be extracted from the elements of the homography (Bradski and Kaehler 2008). Other techniques for calibration using checkerboards seek to recover the 2D projection of an “elusive” structure of 3D projective space known as the *absolute quadric* (Faugeras, Luong et al. 2004); the projection of the absolute quadric characterizes the affine distortion of space in 2D, which in effect, is expressed by the matrix of intrinsic parameters of the camera.

3.2 The geometry of two views

The set of corresponding positions in two images is characterized by a set of geometric properties which can be useful for the estimation of the respective 3D points and the rejection of outliers in the context of visual SLAM. The projections of a 3D point in two views with camera centers that differ by a translation and rotation give rise to the so-called *epipolar geometry* stemming from a coplanarity constraint. Epipolar geometry does not apply in pure rotational camera motion and in cases of scenes with fully coplanar arrangements of points; such configurations of camera positions or observed points are known as *degeneracies*.

3.2.1 Triangulation

Consider two camera views of a scene with respective coordinate frames that differ by some rigid transformation. The process of estimating the 3D locations of features in the world from their respective tracked positions in two images is called *triangulation*. There have been essentially two prominent triangulation techniques proposed in literature that are optimal by means of a chosen criterion. In this thesis, for simplicity of illustration, the suboptimal *midpoint* triangulation technique (Trucco and Verri 1998) is briefly outlined. For a thorough treatment of the topic of triangulation, the reader is referred to the papers by Hartley (Hartley and Sturm 1997) and Kanatani (Kanatani, Sugaya et al. 2008).

Let p_1 and p_2 be the corresponding feature locations in two images of a scene and let *M* be the respective real world point. Also, let *R* be the rotation matrix that aligns the first camera frame with the second and *b* the vector that connects the first and second camera center (in the coordinate frame of the first camera), also known as the *baseline*. Now, consider the ray l_1 that passes through the first camera center O_1 and the normalized

Euclidean coordinates of p_1. The parametric equation of l_1 (in the coordinate frame of O_1) is,

$$l_1(\kappa) = \kappa K^{-1}\bar{p}_1 = \kappa m_1 \tag{3.3}$$

where κ is a scalar, $\bar{p}_1 = [p_1^T \quad 1]^T$ is the normalized homogenous representation of p_1, $m_1 = K^{-1}\bar{p}_1$ is called the *normalized Euclidean projection* of p_1 and K is the matrix of camera intrinsic parameters. In a similar manner, the parametric equation of the ray l_2 that passes through the second camera center O_2 and the normalized Euclidean projection m_2 (also in the coordinate frame of O_1) is,

$$l_2(\lambda) = \lambda R K^{-1}\bar{p}_2 + b = \lambda R m_2 + b \tag{3.4}$$

where λ is a scalar.

For the sake of simplicity, from this point onwards, image coordinates will be dropped from expressions and the normalized Euclidean coordinates will be used instead. Using the cross product, a vector w, mutually perpendicular to the rays in (3.3) and (3.4) is obtained:

$$w = m_1 \times (R m_2) \tag{3.5}$$

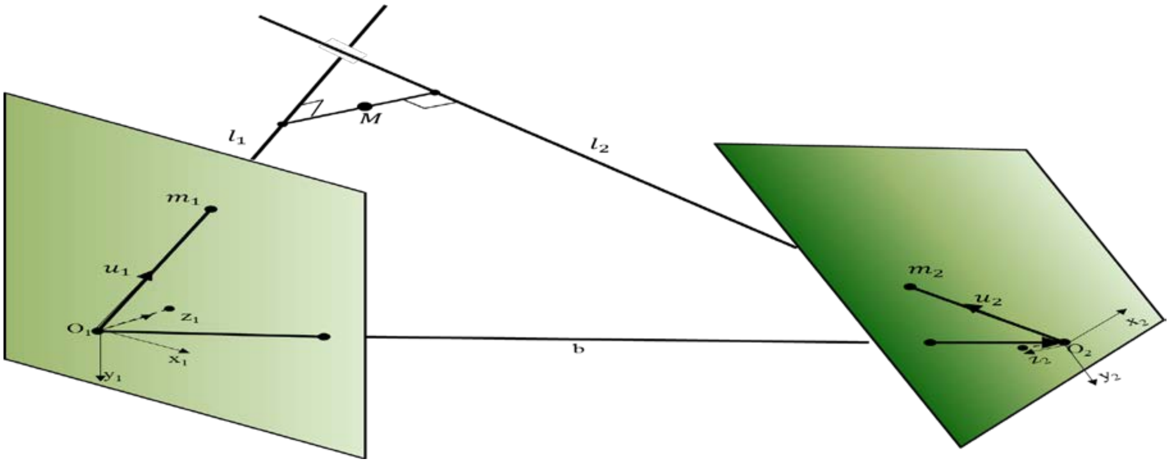

Figure 3.3. The midpoint triangulation method.

Define vectors $u_1 = m_1$ and $u_2 = R m_2$ to lighten expressions in the following derivations. From equations (3.3-5) and introducing a third parameter ρ associated with the line section that connects l_1 and l_2 through M, the triangulation of M now becomes a matter of finding appropriate values κ^*, λ^*, ρ^* for parameters $\kappa, \lambda, \rho \in \mathbb{R}$ such that,

$$\kappa^* u_1 = \lambda^* u_2 + b + \rho^* w \tag{3.6}$$

Equation (3.6) defines a 3x3 system of linear equations in terms of κ, λ and ρ with the following solution:

$$\begin{bmatrix} \kappa^* \\ \lambda^* \\ \rho^* \end{bmatrix} = [u_1 \quad -u_2 \quad -w]^{-1} b \tag{3.7}$$

The 3D point estimate is therefore computed (in the first/left camera frame) as follows:

$$M = \kappa^* u_1 - \frac{\rho^*}{2} w = b + \lambda^* u_2 + \frac{\rho^*}{2} w \tag{3.8}$$

The reader is deferred to Appendix C for more details on the use of triangulation in the context of visual SLAM and the derivatives of the recovered parameters κ^*, λ^* and ρ^* with respect to camera pose.

3.2.2 The epipolar constraint

Epipolar geometry in two views concerns the geometry of the plane induced by the baseline and the projection rays that connect the real-world location of a feature with the two projection (camera) centers. Thus, for each pair of correspondences there exists a plane that contains the respective world point and the baseline vector. Figure 3.4 illustrates the epipolar plane of a pair of correspondences. The projections of the two camera centers in the first and second view are the epipoles e_1, e_2. The lines λ_1 and λ_2 defined by the epipoles and the normalized Euclidean projections m_1, m_2 are known as the *epipolar lines*; epipolar lines are in fact, the projections of the two rays that pass through the camera centers O_1, O_2 and the point *M* onto the opposite image planes.

Suppose again that the baseline is b and the rotation matrix that aligns the first camera frame with the second is *R*. Again, let the normalized Euclidean projections of *M* be m_1 and m_2. Also, let M_1 and M_2 be the coordinates of *M* in the first and second camera frame respectively. Then, the relationship between M_2 and M_1 is,

$$M_2 = R^T (M_1 - b) \tag{3.9}$$

Let now u_1 and u_2 be the direction vectors (in the first camera coordinate frame) of the projection rays that connect the first and second camera center with the point *M*. Evidently, u_1, u_2 and *b* span the *epipolar plane* that corresponds to *M*.

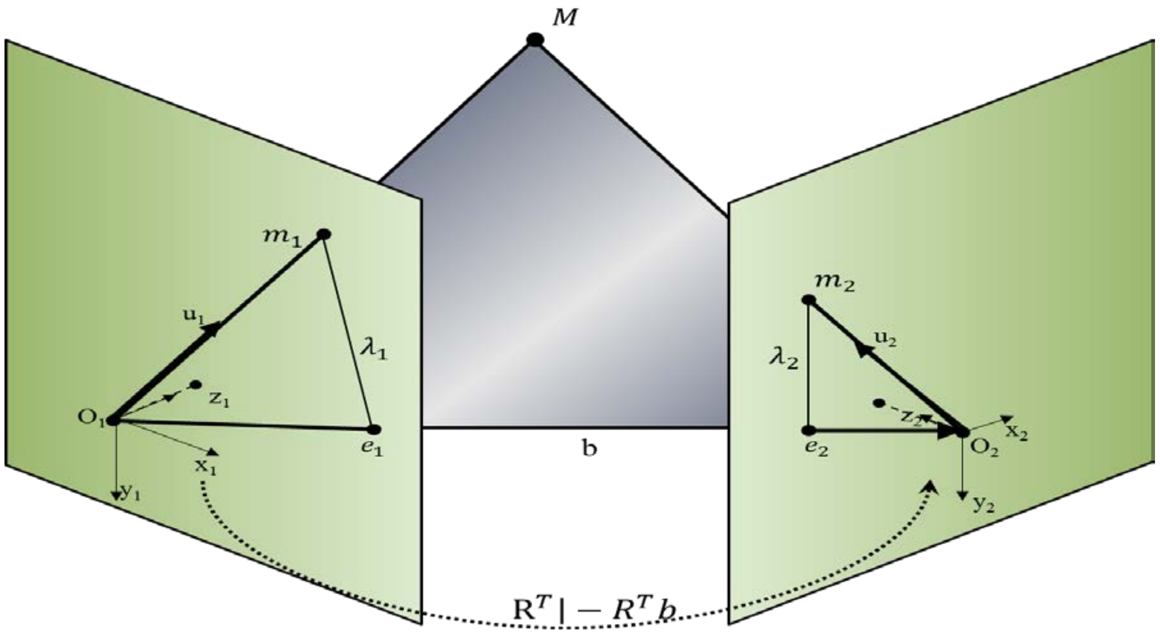

Figure 3.4. Epipolar plane induced by corresponding projections.

An equivalent way of expressing this coplanarity is by considering the orthogonality relationship between u_2 and the cross product of b and u_1:

$$u_2 \cdot (u_1 \times b) = 0 \tag{3.10}$$

where $\cdot$ is the inner product operator. Since $u_2 = u_1 - b = M_1 - b$ and $u_1 = Ru_2 + b = RM_2 + b$, substituting in equation (3.10) yields,

$$(M_1 - b)^T\big([b]_\times(RM_2 + b)\big) = 0 \tag{3.11}$$

where $[b]_\times$ is the cross product skew symmetric matrix of b. By simply applying the distributive law and taking into consideration the fact that $b^T[b]_\times = [b]_\times b = 0$, the following constraint is obtained:

$$M_1{}^T[b]_\times RM_2 = 0 \quad \Leftrightarrow \quad M_2{}^T R^T [b]_\times M_1 = 0 \tag{3.12}$$

By definition, the normalized Euclidean projections are projectively equal to M_1 and M_2 and therefore the constraint can be re-expressed in terms of m_1 and m_2:

$$m_1{}^T([b]_\times R)m_2 = 0 \quad \Leftrightarrow \quad m_2{}^T(R^T[b]_\times)m_1 = 0 \tag{3.13}$$

The essential matrix. The 3×3 matrix $R^T[b]_\times$ is known in literature as the *essential matrix* and is usually denoted with the letter *E*. Thus, the epipolar constraint is in its widely recognized form (Longuet-Higgins 1987)[6] is:

$$m_2{}^T E m_1 = 0 \tag{3.14}$$

The fundamental matrix. Considering that the image projections can be expressed in terms of their normalized Euclidean projections through $\tilde{p}_1 \propto K m_1$ and $\tilde{p}_2 \propto K m_2$, substituting into equation (3.14) a similar bilinear relationship is obtained (this time, in image coordinates):

$$\tilde{p}_2{}^T \underbrace{(K^{-T} E K^{-1})}_{F} \tilde{p}_1 = 0 \tag{3.15}$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the 2D homogenous image coordinates of the projections in the two views, K^{-T} is a shortcut notation for $(K^{-1})^T$ and the matrix $F = K^{-T} E K^{-1}$ is called the *fundamental matrix* (Luong and Faugeras 1996). The fundamental matrix defines a bilinear relationship between the two views directly in image coordinates and therefore, it may be of more use than the essential matrix when the camera intrinsics are not known.

The concept of epipolar geometry can be straightforwardly ported from Euclidean space to the space of image coordinates, also known as *uncalibrated space*. Thus, each pair of correspondences p_1 and p_2 and the two camera centers define an epipolar plane that contains the two epipoles in uncalibrated space. It follows that the epipolar lines in image coordinates are defined as the lines that connect the epipoles with the projections p_1 and p_2. As in Euclidean space, epipolar lines in uncalibrated space are back-projections of the rays that pass through the opposite camera center and the point *M* in image coordinates. Thus, for image point p_1, the corresponding projection in the opposite view should lie on the uncalibrated epipolar line γ_2:

$$\gamma_2 \propto \tilde{p}_1^T F^T \tag{3.16}$$

In quite a similar manner the respective epipolar line γ_1 of p_2 in the first view is:

[6] Evidently, there can be many essential matrices corresponding to the same set of 2-view projections depending on the interpretation given to the rigid transformation that links the two camera poses. In this thesis, the rotation matrix R is perceived as the matrix that contains the second camera frame directions (expressed in the first camera frame) as its columns and b is the baseline vector (also in the first camera coordinate frame): Thus, the essential matrix will always be given by $E = R^T[b]_\times$.

$$\gamma_1 \propto \tilde{p}_2^T F \tag{3.17}$$

where γ_2 and γ_1 have arbitrary scale and the ~ symbol is omitted because they represent lines. It is quite evident in Figure 3.4 that the epipoles belong to all epipolar lines in their respective view. Thus, for every pair of correspondences m_1 and m_2, it follows that $m_1^T E^T \tilde{e}_2 = 0$ and $m_2^T E \tilde{e}_1 = 0$ (obviously, the exact same relationships hold in uncalibrated space). The latter clearly implies that e_1 and e_2 are the right and left null spaces of E respectively. Similarly, in uncalibrated space, the epipoles are the right and left null spaces of the fundamental matrix. A detailed list of properties of the essential/fundamental matrix along with the respective derivations and proofs is given in Appendix D.

3.2.3 Methods for the computation of the fundamental/essential matrix

Estimation of the epipolar constraint although not a trivial task, has been the subject of research for decades and countless algorithms have been proposed for its computation. For this reason, the text will assume that the essential matrix has already been estimated from image correspondences by means of an efficient algorithm of choice. A few recommended methods are, the 8-point algorithm (Longuet-Higgins 1987), Hartley's modification to the 8-point algorithm (Hartley 1995) and various robust approaches using RANSAC (Fischler and Bolles 1981) such as Philip Torr's PLUNDER and MLESAC (Torr, Zisserman et al. 1998, Torr and Zisserman 2000) or O. Chum's DEGENSAC (Chum, Werner et al. 2005). In this thesis, the estimation technique of preference is a RANSAC based version of Hartley's modified 8-point algorithm, as implemented in OpenCV.

3.2.4 Extracting relative camera pose from the essential matrix

In short, there are two prominent methods for extracting relative orientation and baseline from the essential matrix: The first one relies on the SVD[7] and is described in detail by Ma, Soatto and Kosecka (Ma, Soatto et al.) as well as by Nister (in the appendix of the 5-point paper), whereas the second is a much earlier work, somewhat neglected in literature, by Horn (Horn 1990). In this thesis, Horn's approach is the preferred method not only because it relies on a mathematically elegant observation, but also because it requires nothing more

[7] Refer to section 2 of Appendix D for detailed proofs of the underlying theorems that justify the use of the SVD.

than a matrix multiplication and addition (the SVD is not necessary). Detailed derivations of the formulas in this section are given in Appendix D.

It should be noted here that the straightforward way to ensure that E is indeed an essential matrix, is to impose rank-2 and two equal non-vanishing singular values on its singular values decomposition (SVD). Assuming that E is an essential matrix, then there exist four possible up-to-scale (baseline is typically normalized to unit length) relative camera poses consistent with E which are combination pairs of two rotation matrices and two unit baseline vectors.

A necessary pre-processing step is to eliminate arbitrary scale from the essential matrix, which is equivalent to normalizing the baseline. To set the baseline to unit length, E is divided with the square root of half of the trace of its Gramm matrix, $E^T E$ (or, EE^T):

$$E_n = \frac{1}{\sqrt{\frac{Tr(E^T E)}{2}}} E = \frac{1}{\sqrt{\frac{Tr(EE^T)}{2}}} E \tag{3.18}$$

Let $b = [b_1 \quad b_2 \quad b_3]^T$ be the normalized baseline vector. The absolute values of the baseline components can then be extracted from the diagonal elements of the Gramm matrix as follows:

$$b_1^2 = 1 - [E_n^T E_n]_{11} \tag{3.19}$$

$$b_2^2 = 1 - [E_n^T E_n]_{22} \tag{3.20}$$

$$b_3^2 = 1 - [E_n^T E_n]_{33} \tag{3.21}$$

where $[E_n^T E_n]_{ij}$ denotes the element of $E_n^T E_n$ in the i^{th} row and j^{th} column. To resolve the sign ambiguity, the largest squared component is assumed to be a positive square root and the remaining signs are inferred from the off-diagonal elements of $E_n^T E_n$:

$$E_n^T E_n = \begin{bmatrix} b_2^2 + b_3^2 & -b_1 b_2 & -b_1 b_3 \\ -b_1 b_2 & b_1^2 + b_3^2 & -b_2 b_3 \\ -b_1 b_3 & -b_2 b_3 & b_1^2 + b_2^2 \end{bmatrix} \tag{3.22}$$

It suffices to recover one baseline vector from $E_n^T E_n$ as described above and the second baseline will simply be the same vector pointed at the opposite direction.

Having recovered a baseline vector, the two possible rotation matrices are computed with the following simple formula (section 3 of Appendix D):

$$R = C_n^T \pm [b]_\times E_n^T \tag{3.23}$$

where $[b]_\times$ is the cross-product skew symmetric matrix associated with b and C_n is the matrix of cofactors of E_n. To avoid confusion caused by minor variations in the definition of the adjoint/adjugate in literature, the correct formula for C_n is given below:

$$C_n = \begin{bmatrix} \begin{vmatrix} e_{22} & e_{23} \\ e_{32} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{21} & e_{23} \\ e_{31} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{21} & e_{22} \\ e_{31} & e_{32} \end{vmatrix} \\ -\begin{vmatrix} e_{12} & e_{13} \\ e_{32} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{13} \\ e_{31} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{12} \\ e_{31} & e_{32} \end{vmatrix} \\ \begin{vmatrix} e_{12} & e_{13} \\ e_{22} & e_{23} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{13} \\ e_{21} & e_{23} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{vmatrix} \end{bmatrix}$$

$$\Leftrightarrow C_n = \begin{bmatrix} e_{22}e_{33} - e_{32}e_{23} & -(e_{21}e_{33} - e_{31}e_{23}) & e_{21}e_{32} - e_{31}e_{22} \\ -(e_{12}e_{33} - e_{32}e_{13}) & e_{11}e_{33} - e_{31}e_{13} & -(e_{11}e_{32} - e_{31}e_{12}) \\ e_{12}e_{23} - e_{22}e_{13} & -(e_{11}e_{23} - e_{21}e_{13}) & e_{11}e_{22} - e_{21}e_{12} \end{bmatrix} \tag{3.24}$$

The four possible relative pose configurations recovered from the essential matrix are illustrated in Figures 3.6 and 3.7 in term of the location of the observed points.

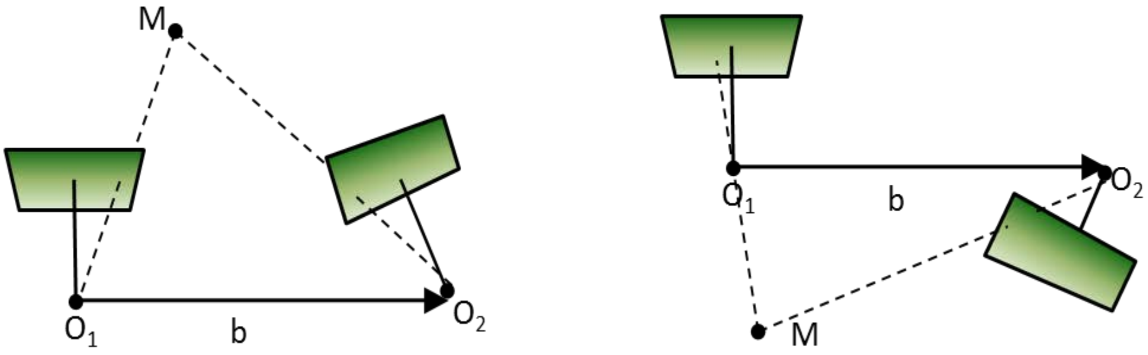

Figure 3.6. Camera orientation with respect to the observed points for "positive" baseline direction.

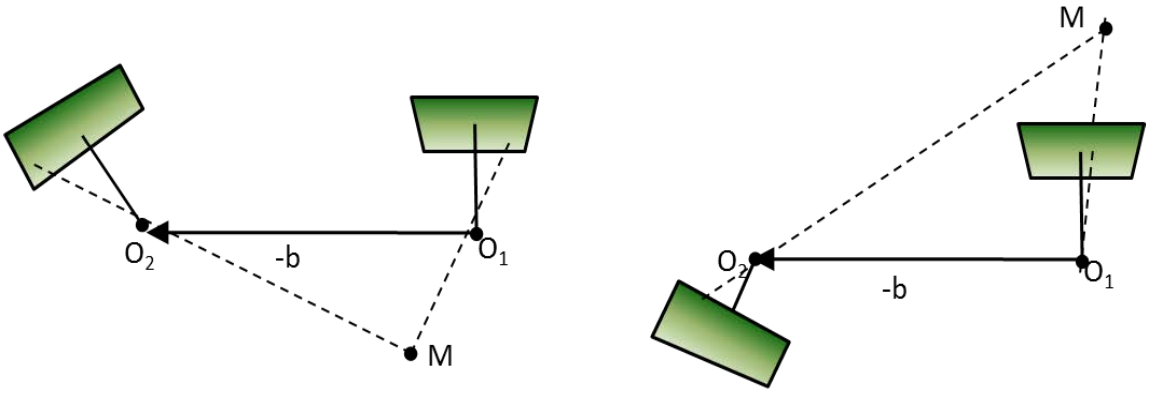

Figure 3.7. Camera orientation with respect to the observed points for "negative" baseline direction.

It is clear that only one combination of the two baselines and two rotation matrices aligns both cameras behind the observed point. This suggests that, in order to resolve the ambiguity between the four solutions, a reconstruction of the scene must be obtained and the transformation that yields positive (or negative) signs in both depths of an observed point as observed from the two camera views should be the correct one. In other words, the point should lie in front of both cameras. With noisy data, this may not be the case for all points even for the correct transformation and therefore in practice this is a matter of voting.

3.2.5 Scene reconstruction in two views from known correspondences and relative pose

In this section, a general method for 3D scene reconstruction in two views from known relative pose and correspondences is presented. This method is also used to disambiguate the four relative pose solutions extracted from the essential matrix.

Let $M = [X \quad Y \quad Z]^T$ be the real-world location of an observed point expressed in the first camera coordinate frame and let m_1 and m_2 be the respective normalized Euclidean projections in the two camera views. Also, let b denote the baseline vector in the coordinate frame of the first camera and R be the orthonormal matrix containing the directions of the second camera frame (expressed in the first camera coordinate frame) arranged column-wise. It follows that M can be expressed in terms of m_1 as follows:

$$M = Zm_1 \tag{3.25}$$

The location of M in the second camera frame is then $R^T(M-b)$. The normalized Euclidean coordinates of M are given by the following:

$$m_2 = \frac{1}{1_z^T R^T(M-b)} K R^T(M-b) \tag{3.26}$$

where $1_z = [0 \quad 0 \quad 1]^T$. Substituting (3.25) into (3.26) yields a relationship in which the only unknown is the depth *Z*:

$$\begin{aligned} &\left(1_z^T R^T(Zm_1 - b)\right)m_2 = R^T(Zm_1 - b) \\ \Leftrightarrow\ &\left(1_z^T R^T(Zm_1 - b)\right)m_2 - R^T(Zm_1 - b) = 0 \end{aligned} \tag{3.27}$$

For any vectors a, b, c of arbitrary size, it is easy to prove that $(a^T b)c = (ca^T)b$. With this identity at hand, the relationship in (3.27) becomes:

$$\begin{aligned} &(m_2 1_z^T)R^T(Zm_1 - b) - R^T(Zm_1 - b) = 0 \\ \Leftrightarrow\ &(m_2 1_z^T - I_3)R^T(Zm_1 - b) = 0 \end{aligned} \tag{3.28}$$

where I_3 is the 3×3 identity matrix. Equation (3.28) is an over-determined system in *Z* and yields two solutions, one for each projection component, provided that the measurements are completely noise-free. However, in most cases the two solutions do not agree and, furthermore, we observed that in the majority of these cases, disparity tends to concentrate either on the *x* or on the *y* axis, thereby making one solution more “reliable” than the other. The proposed workaround is to regard *Z* as a minimizer of the following optimization problem:

$$\underset{Z}{minimize}\{(Zm_1 - b)^T R C R^T (Zm_1 - b)\} \tag{3.29}$$

where *C* is the following non-invertible positive semi-definite (PSD) matrix:

$$C = (m_2 1_z^T - I_3)^T(m_2 1_z^T - I_3) = \begin{bmatrix} 1 & 0 & -x_2 \\ 0 & 1 & -y_2 \\ -x_2 & -y_2 & x_2^2 + y_2^2 \end{bmatrix} \tag{3.30}$$

and x_2 and y_2 are the coordinates of m_2 in the directions of the local *x* and *y* axis respectively. Taking the derivative of the quadratic expression in (3.29) in terms of *Z* and setting it to zero, yields the following minimizer:

$$Z = \frac{m_1^T R C R^T b}{m_1^T R C R^T m_1} \tag{3.31}$$

The expression in (3.31) is a robust depth estimate which takes the direction of disparity into consideration thereby avoiding the "pitfall" of having to choose between two solutions for depth without any criteria at hand on how to make that choice. It should be however noted that estimation can yield very erroneous (e.g., negative depth) results if disparity is very noisy in both axes; in such a case, it is preferable to discard the point.

Examples of 2-view reconstructions using the methods described in the previous sections are illustrated in Figures 3.8-11. Features were detected with SIFT and the LK tracker was used to establish correspondences. Flow fields are shown on the images on the right (in green are new features detected in the second image for subsequent tracking). RANSAC outliers were omitted from the reconstruction and do not appear in the flow field illustration. Finally, since camera intrinsics were unknown in all cases, the reconstructions present a discrepancy up to an affine transformation with the ground truth (made-up intrinsics were used). Algorithm 3.1 describes the steps for relative pose and structure recovery from an essential matrix.

Algorith 3.1. ***3D Reconstruction and recovery of relative pose from two views***

Input*: a) Set of normalized Euclidean correspondences $m_1{}^{(i)}$ and $m_2{}^{(i)}$ b) Essential matrix, E.*

Output*: a) Camera relative pose (R, b), b) 3D coordinates of all points $M^{(i)}$.*

```
comment    Obtain the SVD of E:
$[\mathrm{U}, \mathrm{S}, \mathrm{V}] \leftarrow \mathrm{svd}(\mathrm{E})$
comment    Obtain a "normalized essential matrix" by removing scale and at the same time impose the necessary (and capable) condition of exactly two and equal singular values:
$\mathrm{E_n} \leftarrow \mathrm{U} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathrm{V^T}$
AcceptReconstruction ← False.
comment    Compute the absolute values of the baseline components $b_1$, $b_2$, $b_3$ from $E_n^T E_n$
$\mathrm{b_1} \leftarrow \sqrt{1 - [\mathrm{E_n^T E_n}]_{11}}$
$\mathrm{b_2} \leftarrow \sqrt{1 - [\mathrm{E_n^T E_n}]_{22}}$
```

```
$b_3 \leftarrow \sqrt{1 - [E_n^T E_n]_{33}}$
comment    Choosing the greatest component (in absolute value) as positive and working-out the remaining signs from the off-diagonal elements of the essential matrix
If ( $\max\{b_1, b_2, b_3\} = b_1$):
    If ($[E_n^T E_n]_{12} > 0$)
        $b_2 \leftarrow -b_2$
    If ($[E_n^T E_n]_{13} > 0$):
        $b_3 \leftarrow -b_3$
Else If ( $\max\{b_1, b_2, b_3\} = b_2$):
    If ($[E_n^T E_n]_{12} > 0$):
        $b_1 \leftarrow -b_1$
    If ($[E_n^T E_n]_{23} > 0$):
        $b_3 \leftarrow -b_3$
Else:
    If ($[E_n^T E_n]_{13} > 0$):
        $b_1 \leftarrow -b_1$
    If ($[E_n^T E_n]_{23} > 0$):
        $b_2 \leftarrow -b_2$
comment    Storing the two possible baselines
$\text{Baselines} \leftarrow \{(b_1, b_2, b_3) \quad , \quad (-b_1, -b_2, -b_3)\}$
comment    Find the matrix of cofactors of $E_n$
$C_n \leftarrow \begin{bmatrix} e_{22}e_{33} - e_{32}e_{23} & -(e_{21}e_{33} - e_{31}e_{23}) & e_{21}e_{32} - e_{31}e_{22} \\ -(e_{12}e_{33} - e_{32}e_{13}) & e_{11}e_{33} - e_{31}e_{13} & -(e_{11}e_{32} - e_{31}e_{12}) \\ e_{12}e_{23} - e_{22}e_{13} & -(e_{11}e_{23} - e_{21}e_{13}) & e_{11}e_{22} - e_{21}e_{12} \end{bmatrix}$
comment    Store the two possible rotation matrices
$\text{Rotations} \leftarrow \{C_n^T + [\text{Baselines}(1)]_\times E_n^T \quad , \quad C_n^T + [\text{Baselines}(2)]_\times E_n^T\}$
comment    Find the best scene reconstruction for the 4 possible relative poses
$\text{BestReconstruction} \leftarrow \{\text{Rotations}(1), \text{Baselines}(1)\}$
$\text{MinCount} \leftarrow \infty$
For each baseline $b$ in  Baselines:
    For each rotation matrix $R$ in  Rotations:
        $\text{errorCount} \leftarrow 0$
```

```
For each pair of correspondences $(\mathrm{m}_1, \mathrm{m}_2)$:
    $\mathrm{C} = (\mathrm{m}_2 1_3^{\mathrm{T}} - \mathrm{I}_3)^{\mathrm{T}}(\mathrm{m}_2 1_3^{\mathrm{T}} - \mathrm{I}_3)$
    $\mathrm{Z}_1 \leftarrow \frac{\mathrm{m}_1^{\mathrm{T}}\mathrm{RCR}^{\mathrm{T}}\mathrm{b}}{\mathrm{m}_1^{\mathrm{T}}\mathrm{RCR}^{\mathrm{T}}\mathrm{m}_1}$
    $\mathrm{Z}_2 \leftarrow 1_{\mathrm{z}}^{T}\mathrm{R}^{\mathrm{T}}(\mathrm{Z}_1\mathrm{m}_1 - \mathrm{b})$
    If $(\mathrm{Z}_1 \leq 0)$ Or $(\mathrm{Z}_2 \leq 0)$:
        $\mathrm{errorCount} \leftarrow \mathrm{errorCount} + 1$
    Else:
        $\mathrm{M} = \mathrm{Z}_1\mathrm{m}_1$
If $(\mathrm{errorCount} < \mathrm{minCount})$:
    $\mathrm{BestReconstruction} \leftarrow \{\mathrm{R}, \mathrm{b}\}$
    $\mathrm{minCount} \leftarrow \mathrm{errorCount}$
```

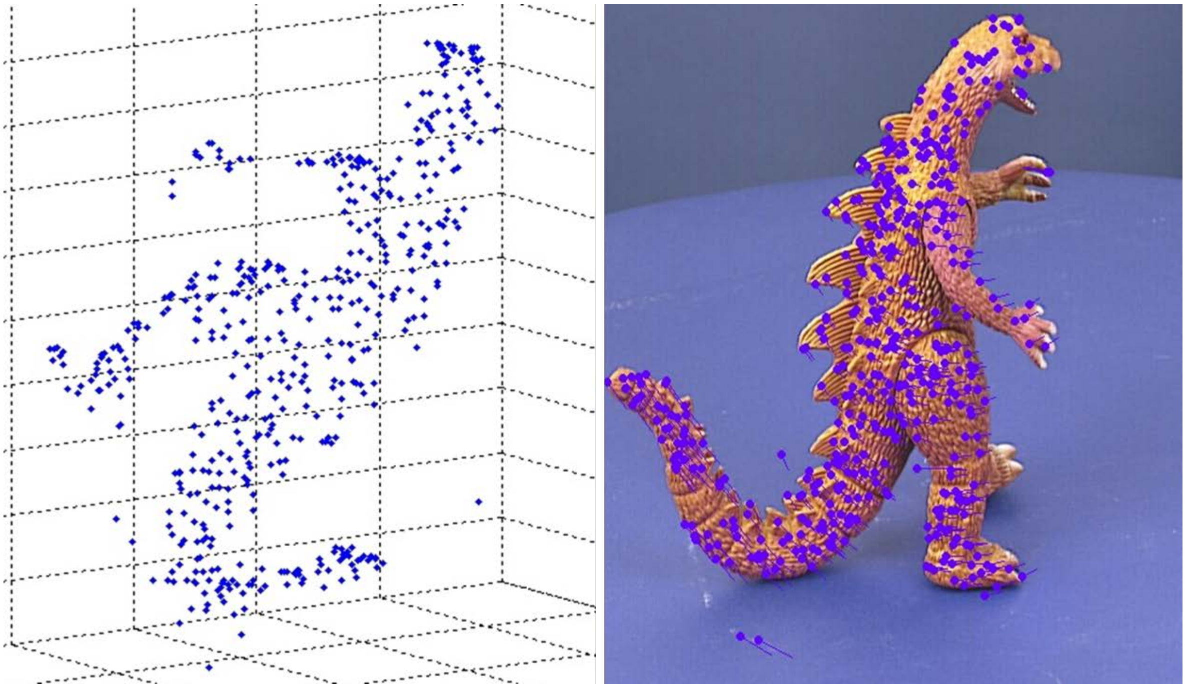

Figure 3.8. A reconstruction of the famous Hannover dinosaur (Niem and Buschmann 1995) from the first two views of the sequence.

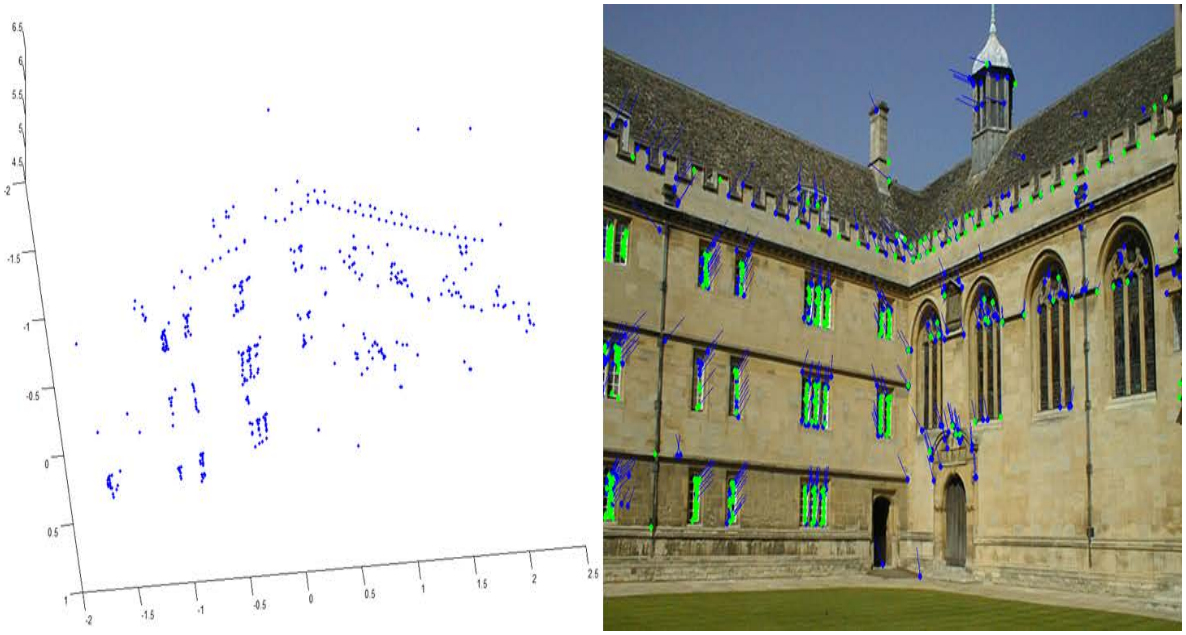

Figure 3.9. A reconstruction Oxford Wadham college from two views (Werner and Zisserman 2002).

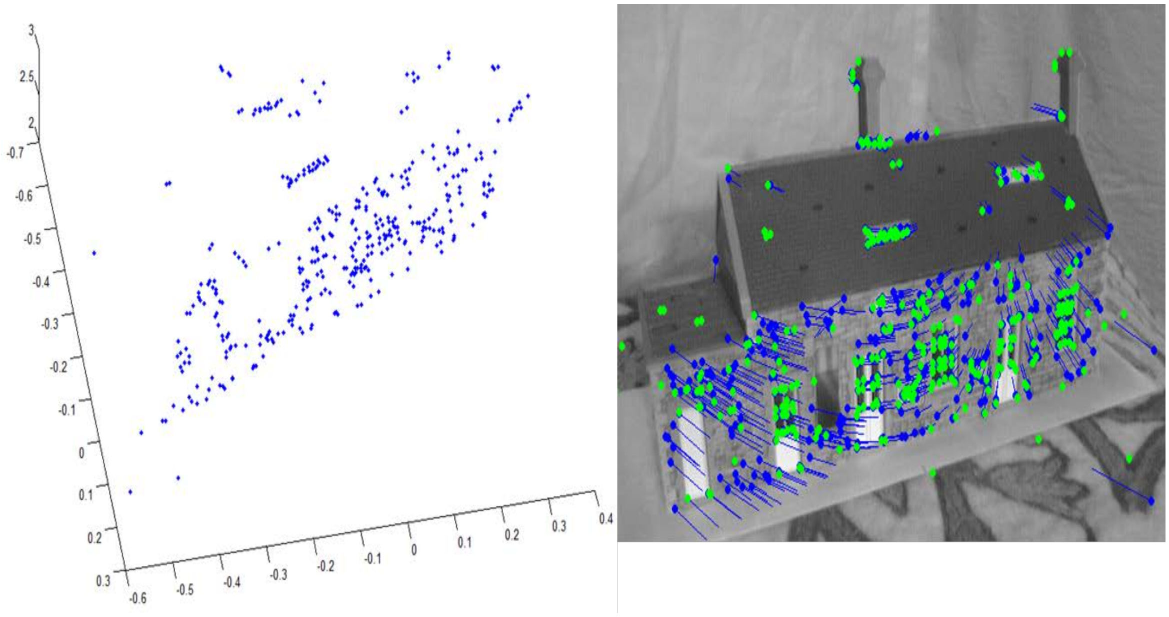

Figure 3.10. A reconstruction of a model house[8] from two views.

[8] Images retrieved from http://www.robots.ox.ac.uk/~vgg/data/data-mview.html

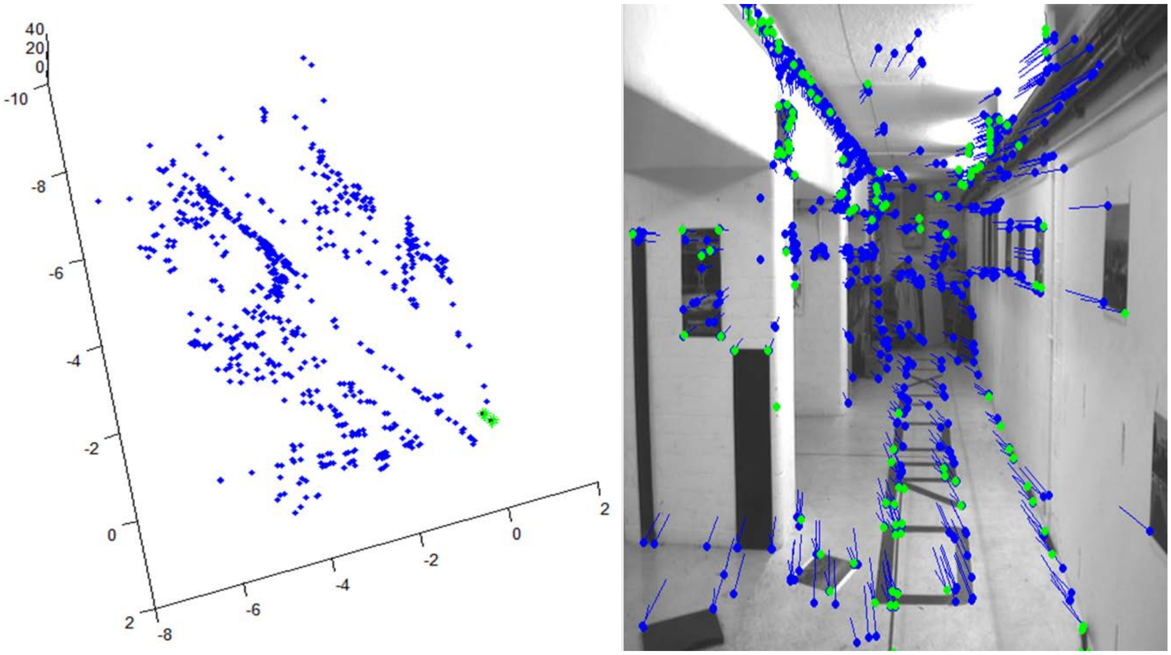

Figure 3.11. Reconstruction from the first two frames of the "corridor" sequence[8]. The two camera locations are shown as green spots on the left.

3.2.6 Scene reconstruction from the essential matrix: Where's the hack?

There have been many authors who presented good to high quality results in terms of recovering camera pose and scene structure from the essential matrix. To name a few renowned researchers, Pollefeys (Pollefeys, Van Gool et al. 2004), Nister (Nistér 2004), Zisserman and Hartley (Hartley and Zisserman 2003) have presented remarkable scene reconstructions with their specialized algorithms for the computation of the fundamental matrix (and subsequently, of the essential matrix through known camera intrinsics). To the best of my knowledge, with the exception of Nister's algorithm[9], what these methods have in common is that they do not directly address the two primary constraints associated with the essential matrix: a) It has exactly two singular values and, b) These singular values are equal. Typically, the aforementioned algorithms enforce these constraints after the optimization. In my opinion, the ramifications of this strategy can be unpredictable, depending on the level of noise in the data. Enforcing exactly two non-zero, equal singular values is a brutal way of imposing constraints and can potentially alter the relative pose estimate to an extent at which no iterative refinement can recover from (for instance, in my experience, a nearly 180^0 "flipped" baseline is not an unlikely occurrence even in "mildly"

[9] Of course, many variations of the 5-point algorithm have been proposed since Nister's paper, but they do not essentially add something new to the concept.

contaminated data). As shown in Appendix D, a matrix E is an essential matrix if and only if it can be written as the sum of two tensor products,

$$E = v_1 u_1^T + v_2 u_2^T \tag{3.32}$$

where $v_1, v_2, u_1, u_2 \in \mathbb{R}^3$ are unit vectors such that, $v_1^T v_2 = u_1^T u_2 = 0$. Equation (3.32) is equivalent to the definition of two equal singular values. It is worth nothing here that the respective expression for the fundamental matrix simply involves a scaling factor applied to the first or the second tensor product of (3.32), hence the 6 DOF as opposed to the 5 in the case of the essential matrix. It is clear that any optimization that pays respect to the true distribution of the elements of the fundamental/essential matrix should enforce the associated orthogonality/orthonormality constraints. An alternative approach would be to use the standard formula for the essential matrix given in equation (3.13) and impose orthonormality constraints on the columns of the rotation matrix and a unit-norm constraint on the baseline (see Chapter 4, section 4.2 for a parametrization using stereographic coordinates). Either way, a properly constrained optimization involves a Lagrangian (or some parametrized expression) with the standard 8-point algorithm cost function and 5 Lagrange multipliers (2 for orthogonality and 3 for unit norms). The Karush-Kuhn-Tucker (KKT) conditions will eventually lead to a 4th degree polynomial system in the components of v_1, v_2, u_1, u_2. This system is relatively hard to obtain analytically and it requires a special category of algorithms known as Groebner basis solvers (Lazard 1983) to solve it.

Nister's solution (Nistér 2004) deserves special reference in this section for being the only method (to the best of my knowledge) that actually solves for the essential matrix while strictly abiding by the orthogonality constraints. The idea is to recover the essential matrix from the 4-dimensional null space of the data matrix. The constraints yield a polynomial system containing reasonably-sized expressions (only 4 unknowns up to arbitrary scale) and can be solved in a relatively uncomplicated manner. Of course, the problem with this method is that it does not generalize to the most usual formulation of the problem which is an overdetermined system[10], in which case the null space of the data

[10] Nister mentions in his paper that the method generalizes to the overdetermined case if the 4 eigenvectors of the data matrix corresponding to the 4 smallest singular values are taken instead of the 4 null-space basis vectors used in the 5-point case. I believe that this is an arbitrary assertion. Clearly, one is free to employ the singular vectors to diagonalize the data matrix (see my PnP formulation in Chapter 8, section 1.3), but there is no justification about why the constrained minimum should be in the space of the smallest 4 singular values. It is however a sane conjecture from a greedy point of view.

matrix is rarely non-empty. For completeness, I would like to mention a parametrization by Vincent Lui and Tom Drummond for an iterative solution of the 5-point problem estimation (Lui and Drummond 2007). Other solutions for the 5-point problem involve Hongdong's (Li and Hartley 2006) and Kukelova's (Kukelova, Bujnak et al. 2008) methods.

An alternative approach to solving the overdetermined constrained essential matrix optimization problem would be to use iterative methods over a 6 or 5 DOF parametrization (such as the one discussed in Chapter 4, section 4.2). Unfortunately, the search space is not convex and although convergence is reached relatively fast, there are no guarantees as to whether the solution is a global minimum or not. A category of methods that convert nonconvex problems into convex and thereafter use iterative optimization to reach the global minimum are known as *convexification* methods (Bertsekas 1979) and can be potentially employed to reach the true minimum. It is my view that the problem can be further researched from the angle of primal-dual methods, without necessarily excluding a possible analytical solution which will however inevitably involve a 4th degree polynomial system.

3.3 Degenerate configurations

Since the early days of photogrammetry, it was a well-known fact that certain scene depictions or configurations of camera locations gave rise to ambiguities in the reconstruction. In visual SLAM the occurrences of *planar scenes* and *purely rotational camera motion* may give rise to ambiguities in the recovery of relative pose. In theory, other degenerate configurations may include situations in which the 3D locations of the tracked features or the camera centers are situated on a twisted cubic, but they are unlikely to occur in practice. For a detailed introduction to degenerate configurations, the reader is deferred to the chapter on degeneracies in *multiple view geometry in computer vision* by Hartley and Zisserman (Hartley and Zisserman 2003).

There exist three distinct degenerate situations that, in practice, manifest themselves in exactly the same way (Figure 3.12). In particular, in any of these three aforementioned degeneracies, the set of correspondences between two views are linked via a projectivity (homography). The first two types of degeneracies involve features with 3D locations that lie on the same plane. In addition, the notion of coplanarity can be extended to include point-clouds that are located very far away from the camera center, practically at *infinity* (i.e., they are lying in a plane infinitely far away from the camera center, called the *plane at infinity*). The third case concerns purely rotational camera motion and is likely to occur when the frame attached to the camera center is not translating. Regardless, all three degenerate types are being dealt with in the same way since the correspondences are linked via a projective transformation.

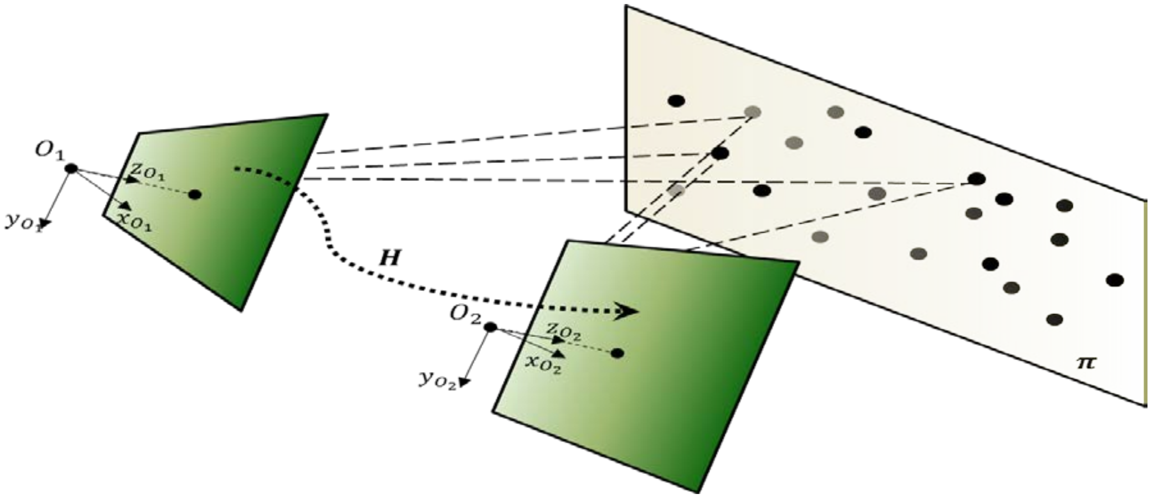

Figure 3.12. The most common degenerate configuration in which the 3D locations of the tracked features lie in a plane.

Coplanarity in SLAM almost certainly occurs because the 3D locations of the features are very far away, practically at infinity. A very simple method to diagnose this degeneracy is to fit a homography to the tracked features and obtain the mean squared error (MSE) of the homography based predicted feature locations (Pollefeys, Van Gool et al. 2004). If the MSE is below some threshold, then it follows that the features are linked by means of a projectivity and the preferred course of action would be to skip to the next frame in the sequence.

Similarly, in cases where the camera appears to be purely rotating, the solution is to skip frames until the baseline contributes significantly in the disparity of the feature locations. Testing for this degeneracy can be reliably made through prior knowledge of orientation which is usually provided by an inertial measurement unit (IMU) or gyro. The idea is to use the IMU based rotation matrix in order to *rectify* the feature locations in the second view so that the rigid transformation between the rectified camera frame and the first camera frame becomes a pure translation, equal to the baseline that links the two camera centers. Once the features are rectified, solving for the baseline becomes a simple homogeneous linear least squares (LS) problem. If the baseline norm is below some threshold (low disparity), then the frame is not processed and the next frame in the sequence is sampled.

It should be noted that a robust criterion that detects both coplanarity and pure rotational motion was proposed by Kanatani (Kanatani 1998). Kanatani's *geometric information criterion* fits 3 distinct models to the data: a) A general motion model $\hat{\mathcal{S}}$ for non-vanishing baseline length, b) A homography-based model $\hat{\mathcal{S}}_\pi$ for the case of coplanar world points and, c) A pure camera rotation model $\hat{\mathcal{S}}_R$. The dominant type of motion can thereafter by selected by the following two tests.

***Planarity test*:** Coplanarity of the observed world points is decided based on the validity (coplanar if true) of the following inequality:

$$\frac{r^2\left[\,\hat{\mathcal{S}}_\pi\right]}{r^2\left[\,\hat{\mathcal{S}}\right]} < 3 + \frac{4}{N-5} \tag{3.33}$$

where $r^2[\,.\,]$ denotes the (average) squared residual of data fitness to the model in brackets and *N* is the number of data points.

***Rotation test*:** Pure rotational motion is decided if the following inequality is true:

$$\frac{r^2\left[\,\hat{\mathcal{S}}_R\right]}{r^2\left[\,\hat{\mathcal{S}}\right]} < 3 + \frac{14}{N-5} \tag{3.34}$$

Although detection of degeneracies is important for robust camera motion estimation, the only preemptive measures against degeneracies in the algorithms presented in this thesis concern the very likely case of coplanarity due to distance (i.e., world points practically lying in the plane at infinity). One of the few research-friendly conditions of Springer's missions is the assumption of constant motion when the vehicle is localizing itself through vision, which practically eliminates the need to deal with the problem of fitting multiple motions, except for the case of very distant scene backgrounds.

3.3.1 Image rectification with known relative orientation

Provided reliable prior information on relative orientation between two camera frames, it is possible to remove the effects of rotational motion from the image projections in the second (right) image by creating a new virtual view (indicated with dashed quadrilateral) in which all projections are presumably the result of pure translational motion. This way, motion equations become linear in the translation components and standard least squares optimization can be applied.

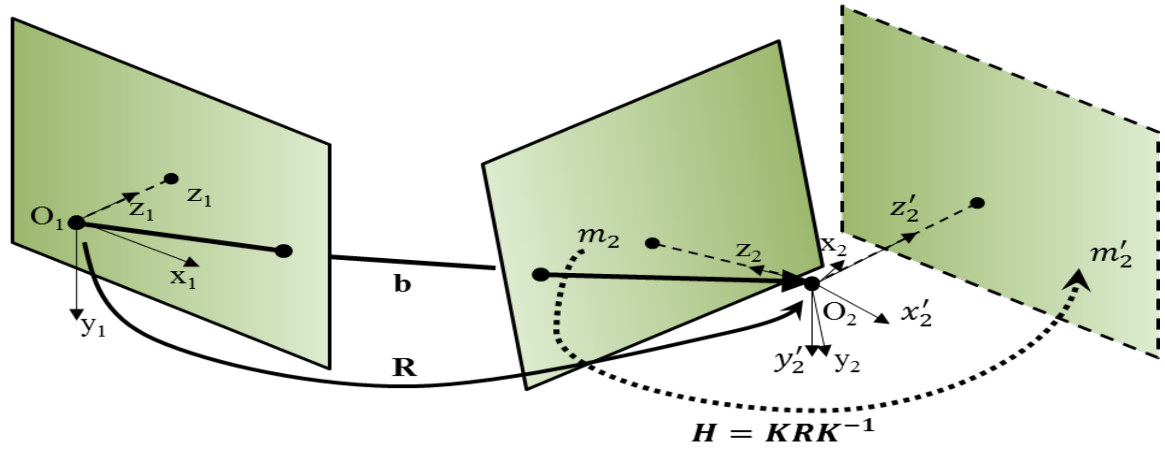

Figure 3.13. Rectification of the second view using a conjugate rotation.

Suppose that R is the rotation matrix that aligns the first camera frame with the second. Figure 3.13 illustrates rectification using prior knowledge of relative orientation between the two views. To rectify the second view with respect to the first, one simply needs to apply the inverse rotation R^T to the second camera frame. It follows from equation (3.9) that the 3D locations M_2' of the features in the virtual camera view will transform in terms of their locations in the original camera frame as follows:

$$M_2' = RM_2 \tag{3.35}$$

Thus, one is able to compute the virtual image locations p_2' of the features in the unrotated view in terms of their locations p_2 in the original image as follows:

$$\tilde{p}_2' \propto \underbrace{KRK^{-1}}_{H} \tilde{p}_2 \tag{3.36}$$

Equation (3.36) clearly points-out that the rectified features can be obtained by applying a homography $H = KRK^{-1}$ to the second/right view; this homography is also known as *conjugate rotation (Hartley and Zisserman 2003).*

3.4 Stereo vs Single camera in scenes of varying depth

Consider a stereo rig with two identical cameras placed at a specific distance B apart along the x-axis of their local frames, both having identical orientations (Figure 3.14).

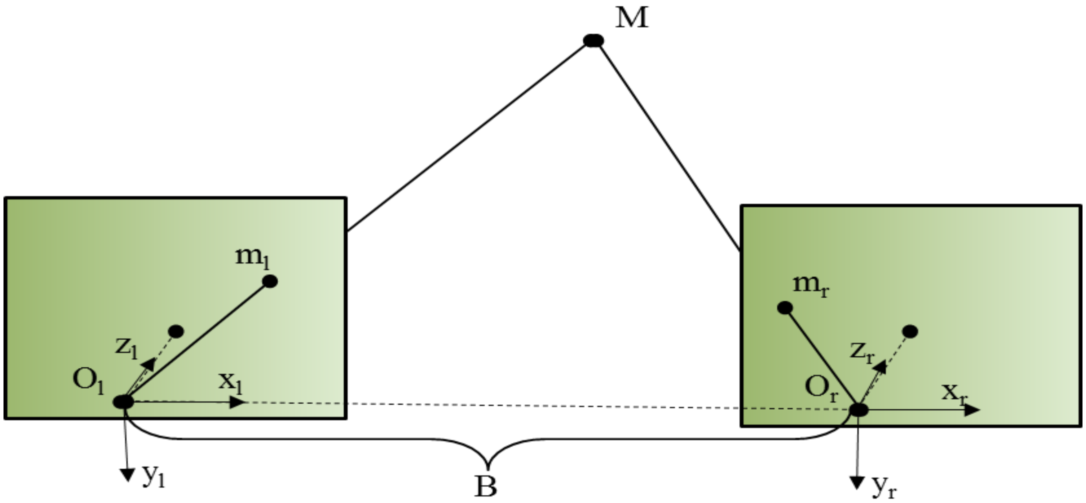

Figure 3.14. Euclidean projections of a point on a standard stereo rig.

This configuration can eliminate the scale ambiguity in the recovery of scene structure when processing video sequences, provided that correspondences between tracked points have been established in both stereo views.

Let $M = [X \quad Y \quad Z]^T$ be a world point with Euclidean projections[11] $m_l = [x_l \quad y_l]^T$ and $m_r = [x_r \quad y_r]^T$ on the left and right image of the rig respectively. Since the baseline lies along the x-axes of the two cameras, it follows that $y_l = y_r = y$. It now becomes more convenient to study the geometry of the rig by adopting the top view shown in Figure 3.15. It is very easy to observe the triangle similarities, $MLm_l \sim O_l C_l m_l$ and $MLm_r \sim O_r C_r m_r$. Also, taking into consideration that $B = x_l - x_r + k_r + k_l$ (the negation on x_r is used because m_r is on the negative side of the x-axis of the local frame; it turns out that the negation is necessary even when m_r is on the positive side) and following a few trivial substitutions, it turns out that,

$$\lambda = f \frac{B - (x_l - x_r)}{x_l - x_r} \tag{3.37}$$

Thus, the depth Z can be obtained as follows:

[11] Please note that in section 3.3, m_l and m_r are general Euclidean coordinates (not necessarily normalized).

$$Z = \lambda + f = f \frac{B}{\underbrace{x_l - x_r}_{d}} \tag{3.38}$$

where $d = x_l - x_r$ is the disparity between m_l and m_r.

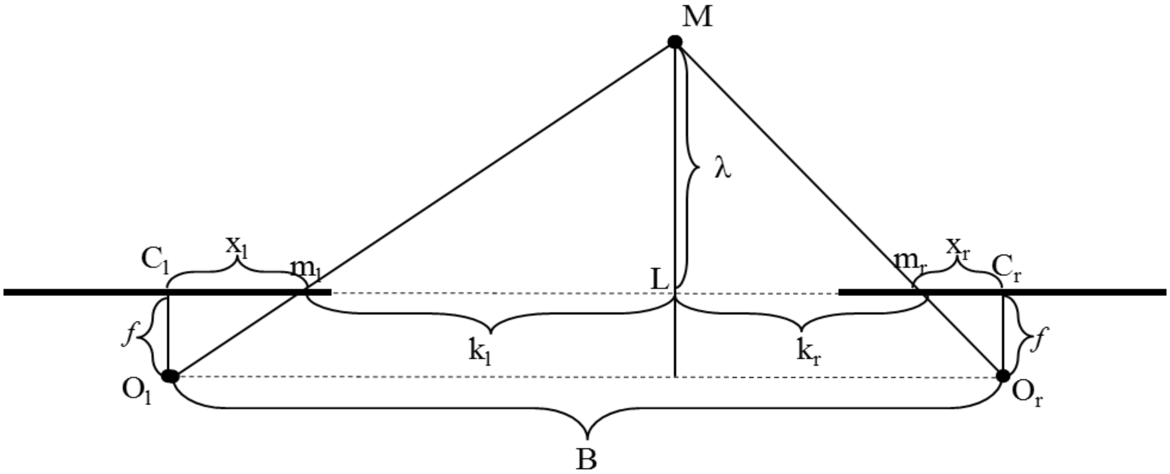

Figure 3.15. Top view of the stereo rig.

The stereo approach allows for scale recovery in terms of scene geometry and therefore it can be used as a SLAM observation model that accounts for the actual translation with respect to the previous vehicle/rig location in time. Although the recovery of scene depth is essential in visual SLAM, stereo based depth estimates present significant limitations imposed by sensor resolution. In particular, any pair of projections with a disparity value below a threshold c will always have the same coordinates in both stereo views. Equation (3.37) implies that for any point with depth $Z > fB/c$ the disparity will be zero; hence, the point will be effectively mapped to infinity. Considering that $d = B - (k_r + k_l)$, a very obvious solution would be to increase the baseline length; however, this is an impractical solution for scenes with background depth that ranges over a few hundred meters such as river banks, forests, fields, etc. A more intuitive depiction of the manifestation of degenerate disparity is illustrated in Figure 3.16. As the point moves further away from the center of projection, the projection rays become approximately parallel and the reconstruction has increasingly poor quality as a result of the fact that minor perturbations of the directions of the rays have a high impact in the position of the reconstructed point. The dashed rays designate the uncertainty cone for each projection while the solid lines indicate the ground truth; the blue shaded rhombus indicates the uncertainty region for the reconstructed point.

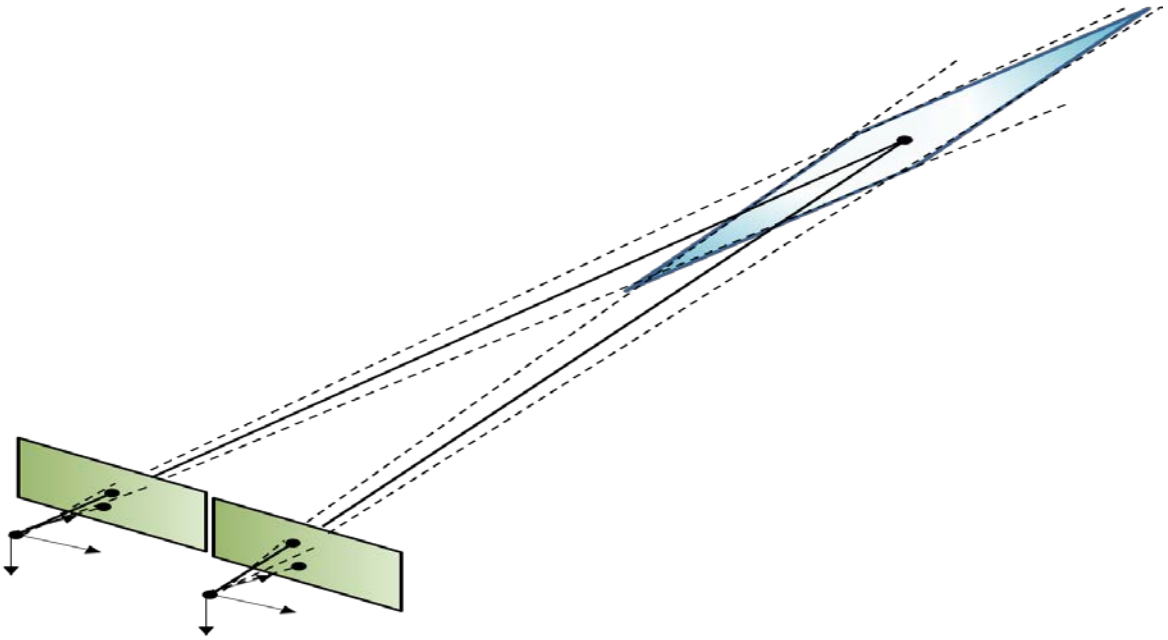

Figure 3.16. Uncertainty region (rhombus in shaded blue) in the estimated position of a point with a stereo rig.

3.5 Summary

The geometry of multiple views delivers the primary tools for the design and implementation of the measurement models used to infuse information to the visual SLAM posterior. Moreover, as shown in this chapter, techniques such as the rectification of views, as well as algorithms for scene reconstruction and relative camera pose estimation from the essential matrix will be useful in terms initializing visual SLAM, obtaining initial guesses for camera motion and rejecting outliers.

It is worth noting here that using equations (3.35) and (3.36) and provided certain assumptions (e.g., planar motion and planar scenes) or information regarding relative orientation between two views, it is possible to obtain a single linear equation that associates the positions of the features in the two respective frames without having to invoke their 3D locations in the expression. In other words, this equation can be used in the context of ordinary least squares to obtain either a solution, or a very reliable starting point for non-linear optimization. These approaches are typically employed in algorithms estimating odometry incrementally by computing the rigid transformation between consecutive pairs of frames (relative pose estimation).

A significant conclusion drawn in this chapter is the very small applicability of stereo rigs in the type of environments that the current study is concerned with. The variations in scene depth in natural environments from a few meters and up to several hundred meters demands prohibiting baseline lengths. Therefore, it is clear that a single camera is the only viable option for relatively unconstrained visual SLAM applications in natural environments.

Chapter 4

Orientation parametrization

Representation of attitude is a major problem in fields such as computer vision, robotics and aerospace engineering. Unavoidably, equations of motion contain rotation matrices in the respective expressions. Unfortunately, rotation matrices are highly redundant representations and incur a number of orthonormality constraints which are very hard to impose in systems of equations and/or optimization algorithms. The most popular alternative is the use of unit quaternions (Horn 1987, Markley and Mortari 1999, Schmidt and Niemann 2001). However, although much more compact than rotation matrices, quaternions are also redundant representations in terms of orientation, since a unit-norm constraint must be imposed in the measurement step of a filter. It is therefore necessary to resort to a method of enforcing this constraint during optimization, such as Lagrange multipliers (Wah and Wu 1999); alternatively, it is preferable to parametrize the quaternion in a way such that the number of degrees of freedom (DOF) of the representation drops down to 3.

4.1 Axis-angle parametrization and quaternions

Any rotation in 3D is equivalent to a rotation about one axis by some angle θ. In this axis-angle representation, if u is some vector on the direction of the rotation axis, then this vector can be regarded as a complete representation of the rotation, if $\|u\| = \theta$. In other words, in order to fully specify a rotation, only an angle θ and a direction vector $u = [u_1 \quad u_2 \quad u_3]^T$ about which the rotation takes place are required; and since one is free to choose the length of this vector, it would be reasonable to make it so that this length encodes the angle of the rotation, θ. Simply put, the axis-angle representation is nothing but a very compact encoding of a rotation with 3 DOF using the 3 components u_1, u_2, u_3 of the vector that defines the rotation in a way such that, $\|u\| = \theta$ where $\theta \in [-\pi, \pi)$.

To obtain the rotation matrix from some given axis-angle representation, one may simply employ the formula of Rodrigues (Faugeras 1993):

$$R = \begin{cases} I_3 + \frac{sin\theta}{\theta}[u]_\times + \frac{1 - cos\theta}{\theta^2}(uu^T - I_3), & \theta \neq 0 \\ I_3\,, \quad \theta = 0 \end{cases} \tag{4.1}$$

where the direction of the rotation is defined by the direction of u using the right-hand-thumb rule, I_3 is the 3×3 identity matrix and $[u]_\times$ is the cross-product skew symmetric matrix:

$$[u]_\times = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}$$

For the special case where $\theta = 0$, the rotation axis becomes ambiguous and the formula of Rodrigues is valid in the limit.

At this point, a very useful alternative expression for the rotation matrix in terms of a quaternion $q = (q_0, v)$ is given (Markley and Mortari 1999):

$$R = (2q_0^2 - 1)I_3 + 2vv^T + 2q_0[v]_\times \tag{4.2}$$

where $q_0 = \cos\frac{\theta}{2}$ and $v = \sin\frac{\theta}{2}\frac{u}{\theta}$.

The rotation in equation (4.1) is already axis-angle parameterized and one could argue against whether it is necessary to use quaternions in order to obtain the derivatives of R, since one can simply work directly on Rodrigues' formula. As shown in Appendix A, the derivatives of the rotation are better computed using the chain rule on quaternion derivatives. This generally improves an, otherwise, very long and painful series of computations.

4.2 A rational parametrization using stereographic projection

The axis-angle parametrization uses the minimum number of 3 DOF required to specify a rotation and it can be computed in a straightforward manner. However, the derivatives of the rotation matrix (section 4.1 of Appendix A) contain expressions in which irrational functions (sinusoids) are present. The latter implies that, in the course of computations, these functions will certainly inflict approximations to the final result thereby deteriorating its numerical accuracy. Hence, a rational parametrization, if possible, is always superior to one that makes use of irrational functions. It is possible to achieve such a parametrization by considering a homeomorphism from $\mathbb{R}^3$ to the 4D unit sphere.

4.2.1 Projecting unit quaternions on the 3D hyperplane

Unit quaternions can be regarded as points on the hypersphere in 4D Euclidean space centered at the origin:

$$q_0{}^2 + q_1{}^2 + q_2{}^2 + q_3{}^2 = 1 \tag{4.3}$$

where q_0, q_1, q_2, q_3 are the quaternion components. Let now $S \equiv (0, 0, 0, -1)$ be the "South Pole" of this 4D sphere and also let ε be the 3D equatorial hyperplane containing the origin of $\mathbb{R}^4$ as shown in Figure 4.1. Let now $r(t)$ be the ray from S that passes through a point $\psi = (x, y, z)$ of the equatorial plane, parameterized by t (note that parentheses are used to denote points, while square brackets are used for vectors):

$$r(t) = (0, 0, 0, -1) + t[x \quad y \quad z \quad 1]^T \tag{4.4}$$

The ray intersects the surface of the sphere at q. Point q is therefore back-projected on ψ through the ray.

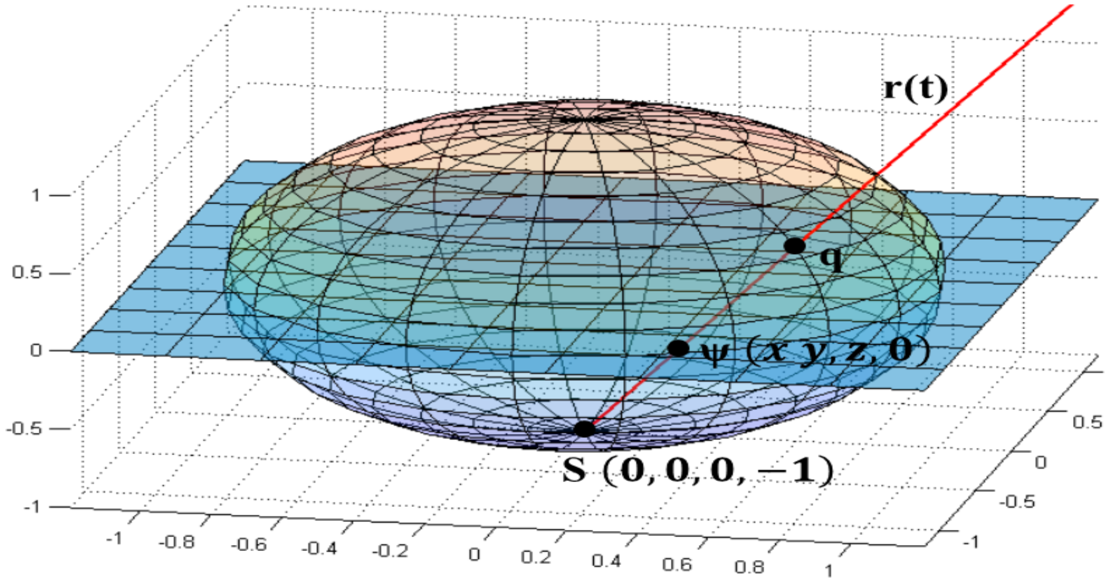

Figure 4.1. The 4D spherical hypersurface of unit quaternions.

Since q lies on the sphere, its coordinates should verify equation (4.3). Moreover, since it also lies on the ray, substituting equation (4.4) in (4.5) yields,

$$(tx)^2 + (ty)^2 + (tz)^2 + (t-1)^2 = 1$$

which provides the following non-trivial solution for *t* in terms of ψ:

$$t = \frac{2}{x^2 + y^2 + z^2 + 1} = \frac{2}{\|\psi\|^2 + 1} \tag{4.5}$$

Finally, substituting the value of *t* from equation (4.5) into (4.3), yields the coordinates of the corresponding unit quaternion in *x*-*y*-*z* parameters:

$$q = \left(\frac{2x}{\|\psi\|^2 + 1}, \frac{2y}{\|\psi\|^2 + 1}, \frac{2z}{\|\psi\|^2 + 1}, \frac{1 - \|\psi\|^2}{\|\psi\|^2 + 1} \right) \tag{4.6}$$

It is worth noting here that, in order to express the center of projection in terms of the stereographic parameters, one needs to include “infinite” values for the parameters x, y, z. This means that the quaternion $(0,0,0,-1)$ cannot be expressed with real values of the stereographic projection parameters. In practice however, one may use very large values for x, y and z and get a very close approximation of the quaternion. Furthermore, the same rotation can be represented by $-\psi/\|\psi\|^2$ and therefore $(0,0,0,-1)$ corresponds to a stereographic triplet comprised of real numbers given by $\psi = (0,0,0)$. It turns-out that the stereographic coordinates behave well in the neighborhood of the South Pole and converge fast towards $(0,0,0,-1)$ during the Gauss-Newton method, even for tolerance levels below 10^{-9}. The price that one pays here is the very high values for the parameter vector components; however, it has been shown (Terzakis, Culverhouse et al. 2014) that these values are well within representation range.

4.2.2 Finding the back-projection of a quaternion on the equatorial plane

Given a point $\psi = (x, y, z)$, the coordinates q_0, q_1, q_2, q_3 of the quaternion can be calculated directly from equation (4.6).

The opposite conversion is equally straightforward, without many computations involved. Let (q_0, q_1, q_2, q_3) a given unit quaternion. As a first step, the squared norm $\|\psi\|^2$ is calculated as follows:

$$\|\psi\|^2 = \frac{1 - q_3}{q_3 + 1} \tag{4.7}$$

From equations (4.7) and (4.6), the *x*, *y*, *z* coordinates of the quaternion's back-projection on the equatorial plane can be easily calculated as follows:

$$\psi = \left(\frac{q_0}{1 - q_3}, \frac{q_1}{1 - q_3}, \frac{q_2}{1 - q_3}\right) \tag{4.8}$$

For a detailed tutorial and extensive list of properties on quaternions and the parametrizations mentioned in this chapter, the reader is referred to Appendix A.

4.3 Comparing parametrization approaches

The most typical use of orientation parametrization in computer vision is in the context of non-linear optimization using the Levenberg-Marquardt (LM) method (Levenberg 1944, Marquardt 1963, Dennis Jr and Schnabel 1996); the LM heuristic is the preferred method for orientation and position refinement in computer vision, robotics and relative fields such as 3D graphics (Neugebauer and Klein 1999, Fitzgibbon 2003, Lourakis and Argyros 2005, Mirzaei and Roumeliotis 2008). It therefore seems reasonable to examine the behavior of the axis-angle vector against stereographic coordinates in the setting of such a non-linear optimization problem.

4.3.1 Using Wahba's problem as a benchmark to evaluate performance of parametrizations in iterative optimization

To evaluate the performance of the axis-angle parametrization against stereographic coordinates, random data for Grace Wahba's problem (Wahba 1965) were generated and iterative optimization was executed to convergence. Wahba's problem is a quadratic minimization problem over the elements of a rotation matrix that aligns N directions ($N \geq 3$) in 3D space:

$$\underset{p}{minimize} \|R(p)U - V\|_F^2 \tag{4.9}$$

where is a 3 DOF parameter vector, $R(p)$ the rotation matrix corresponding to p, $V \in \mathbb{R}^{3\times N}$ is the matrix that contains the direction vectors in the first coordinate frame arranged column-wise and $U \in \mathbb{R}^{3\times N}$ is the matrix of rotated directions in column-wise arrangement; $\|.\|_F$ denotes the Frobenius norm of a matrix (i.e., the trace of its Gramm matrix).

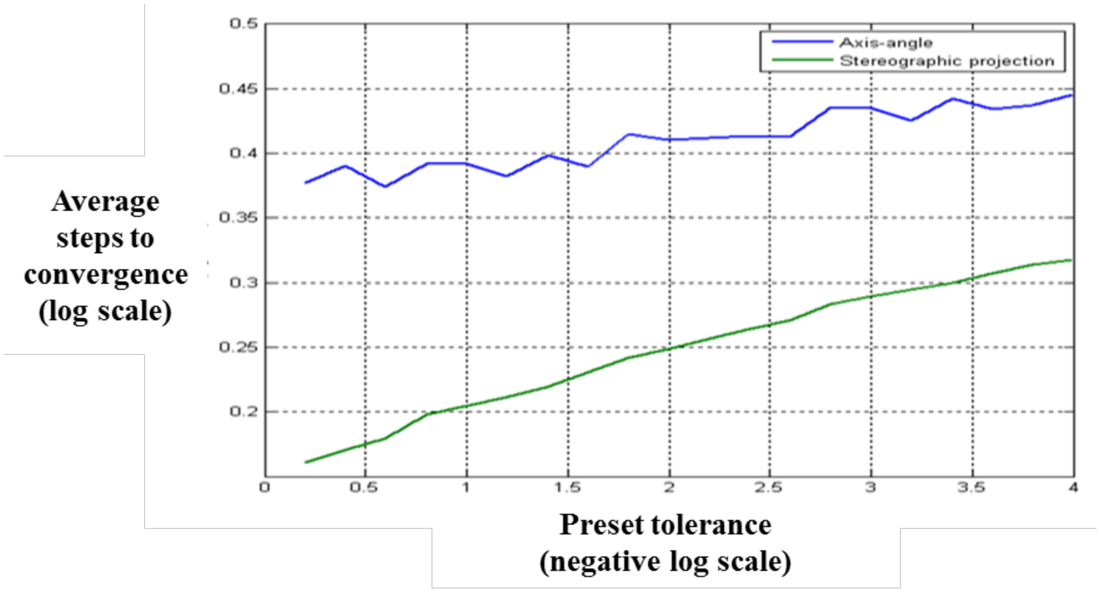

Figure 4.2. Performance comparison in the context of Gauss-Newton iteration (fixed starting point).

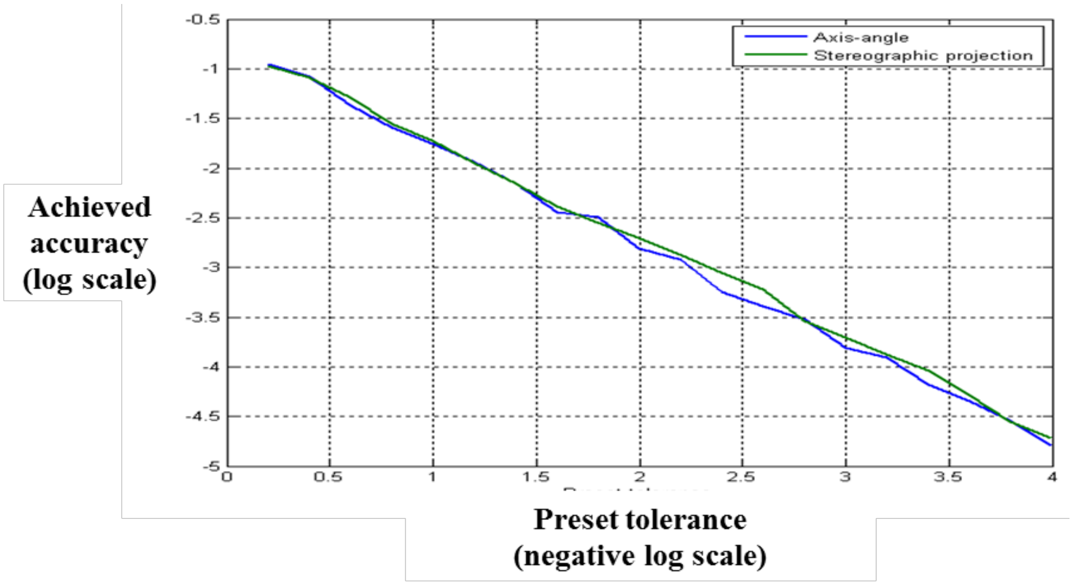

Figure 4.3. Plot of average error norm following convergence (fixed starting point).

Using the two parametrizations, a series of 5 LM executions were performed for 20 different error tolerance values using randomly generated rotation matrices and common starting points (a rotation matrix corresponding to $\frac{\pi}{4}$ about all three axes)[12]. For each set of executions, the average number of iterations to convergence and final error norm was

[12] As will be described in section 4.5, Whaba’s problem is not convex and therefore convergence to the optimal solution can be affected by the choice of initial solution. It was decided that comparing performance from random starting points would be interesting as it may provide clues about the error surface morphology under the two rival parameter vectors.

recorded. Figures 4.2-3 illustrate the results obtained from multiple executions using the same starting point.

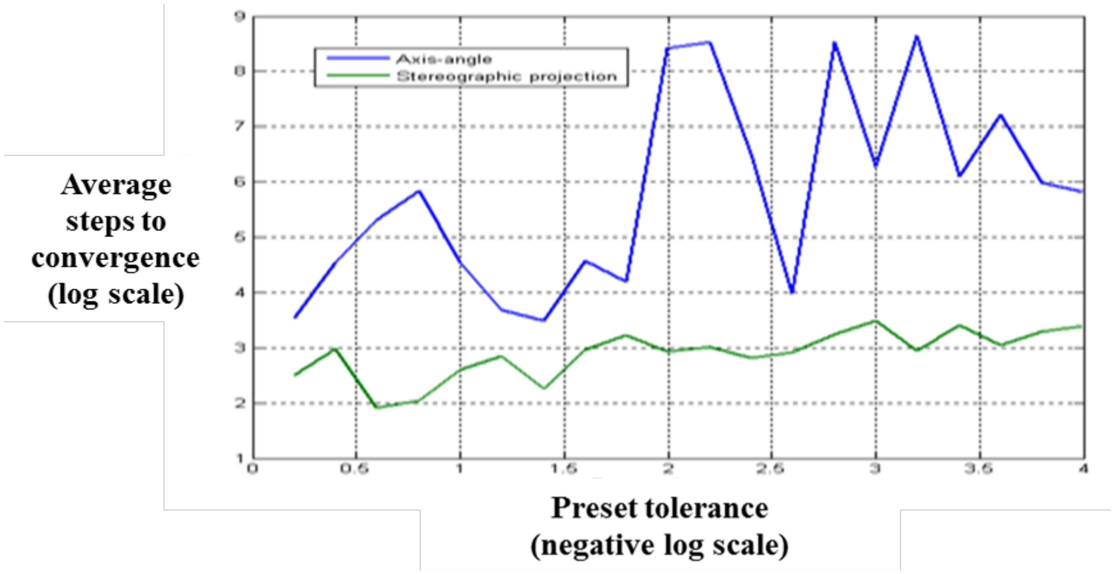

Figure 4.4. Performance comparison in the context of Gauss-Newton iteration (random starting point).

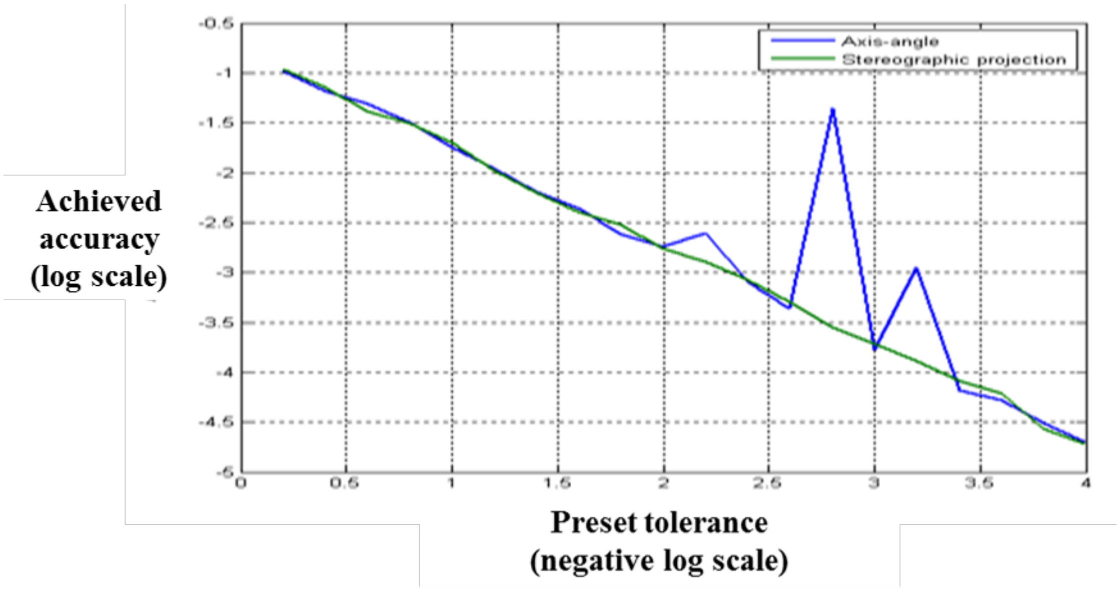

Figure 4.5. Plot of average error norm following convergence (random starting point).

The results suggest marginal superiority of the stereographic projection over the axis-angle parametrization. In fact, the average number of iterations to convergence lies in the range of 6-39 steps, while the respective numbers for the axis-angle parameters range from 11 to even 1887.5 in log scale. In fact, with the stereographic parameters, lowering the error tolerance by 10 units in the negative log scale has practically no effect in the execution time.

The same experimentation was repeated with random starting points (Figures 4.4-5). The results indicate that, once again, the stereographic projection parameters demonstrate a very fast and stable converge in the range of 12-50 steps.

4.3.2 Convergence to the center of projection

The only point on the unit sphere that cannot be represented with real values of the stereographic projection parameters is the chosen center of projection, which is the quaternion $(0, 0, 0, -1)$. One would expect the LM iteration to demonstrate instability in and cause the parameter vector to "explode" while in the neighborhood of the South Pole. Surprisingly, neither of the two happens (Figures 4.6-8).

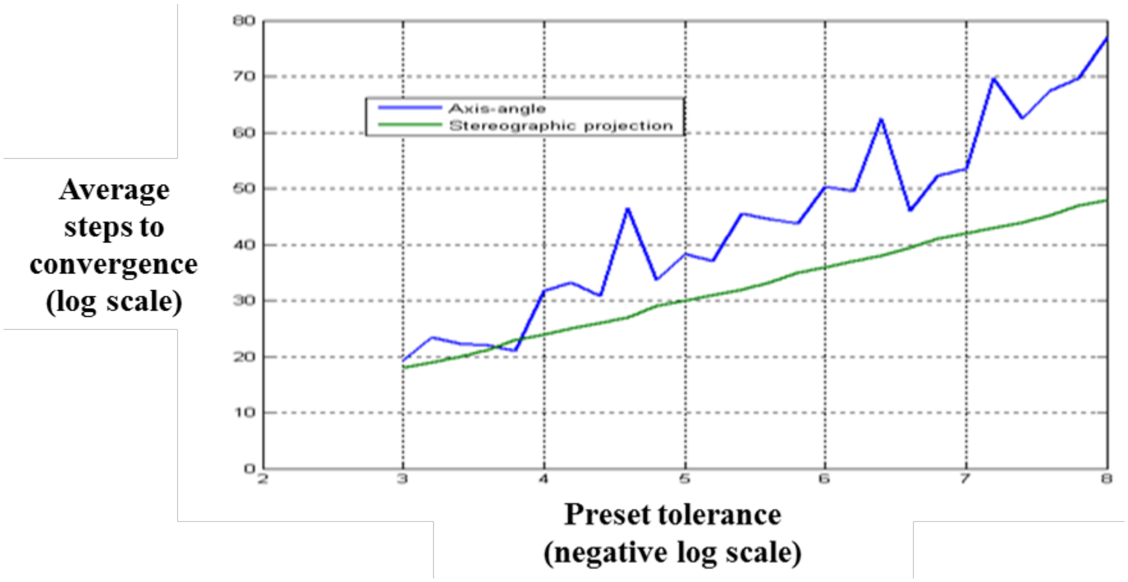

Figure 4.6. Performance comparison in the context of Gauss-Newton iteration for randomly generated quaternions in the neighborhood of the South Pole.

The experiments described in the previous section were repeated using quaternions in a very close vicinity of the projection center. In particular, random quaternions of the following form were generated:

$$q(r) \propto (\delta r_1, \delta r_2, \delta r_3, -1 + \delta r_4)$$

where $\propto$ denotes projective equality, δ can be loosely regarded as a *rate of divergence* from the pole (typically, $\delta = 0.001$) and r is a random number in $[0,1]$. This time, in order to push things further to the extremes, the desired tolerance levels were shifted to very low numbers in the range of 2 to 8 in the negative log scale.

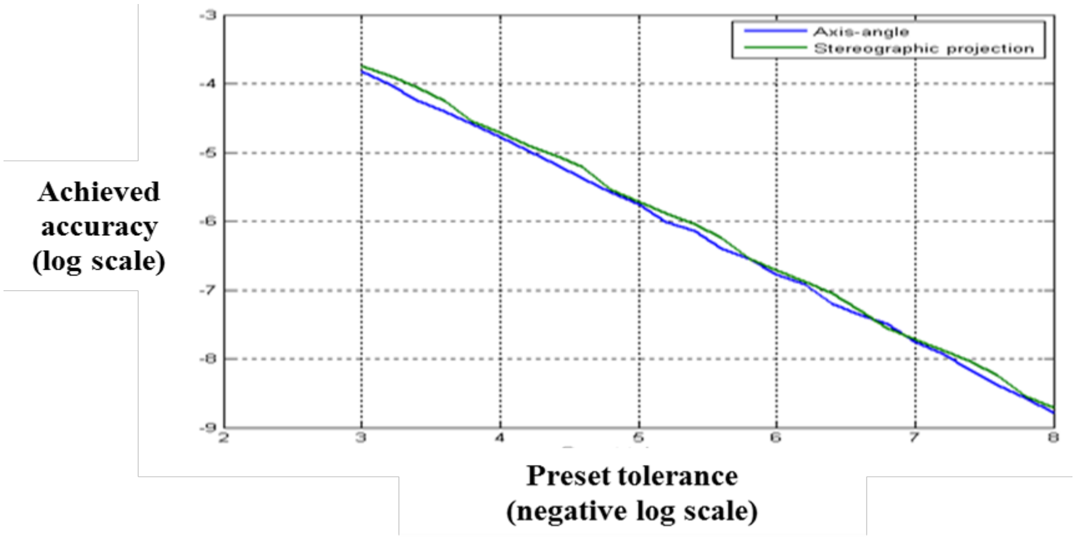

Figure 4.7. Plot of average error norm following convergence.

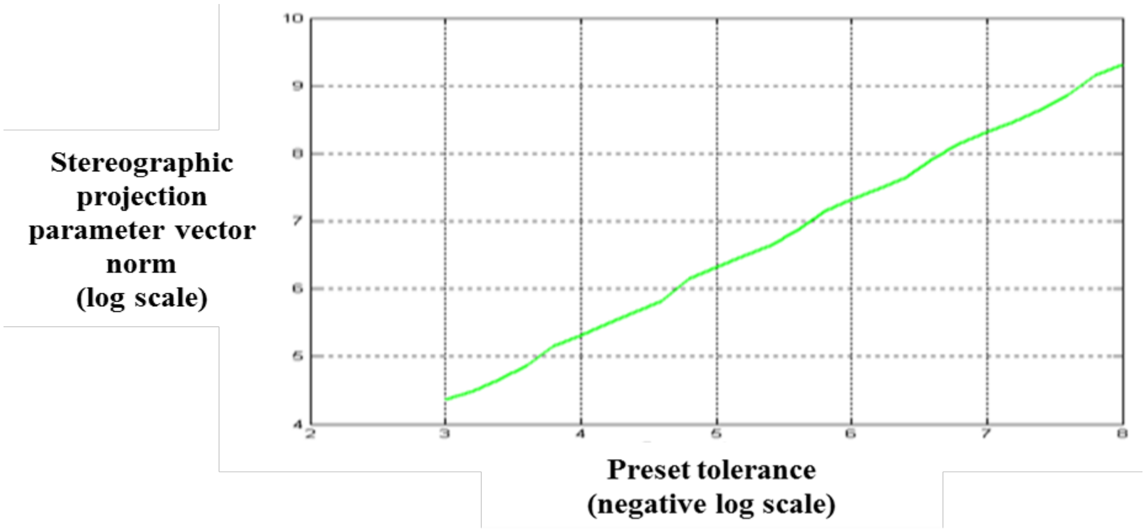

Figure 4.8. Average norm of the stereographic projection parameter vector in terms of preset tolerance.

Convergence remains extremely fast as shown in Figure 4.6, while both parametrization reached convergence at all times (Figure 4.7); the most important news however, is that the growth of the stereographic parameters follows the tolerance in a linear fashion as shown in Figure 4.8. This means that the South Pole can be well represented in practice with accuracy equal to the one of any other point on the sphere with the very "cheap" tradeoff of using relatively large numbers, yet clearly, well within floating point representation range.

4.4 Rational representation of directions in spaces of arbitrary dimensionality

Fundamentally, stereographic projection is a rational encoding scheme for directions in any number of dimensions. Thus, the mapping can be employed to parametrize vectors of fixed or bounded length. Of particular interest are the cases of norm/ball/box constraints in non-linear optimization and the parametrization of the essential matrix and will be described here.

4.4.1 Ball constraints

Ball constraints are very important in bundle adjustment, especially when optimizing the bundle of rays in only two camera views. In such a case, due to scale ambiguity, the baseline vector should be constrained by means of hard bounds as follows:

$$\alpha \leq \|b\|^2 \leq \beta \tag{4.10}$$

where $b \in \mathbb{R}^3$ is the baseline vector and $\beta > \alpha > 0$ are the two bounds.

There have been quite a few methods proposed in literature for enforcing this constraint (Kanzow, Yamashita et al. 2004, Nocedal and Wright 2006, Jia and Zhu 2011) and in the majority of cases, these methods modify the LM step or the method itself, in order to force the new estimate within the feasible set.

What is proposed here is an uncomplicated, simple parametrization scheme based on stereographic projection for the encoding of ball constraints so that the aforementioned optimization problems can be cast and subsequently solved without constraints. The encoding uses 3 parameters to control the length and direction of the baseline vector as follows:

$$b(\kappa, \lambda, t) = \left(\alpha + \frac{(\beta - \alpha)}{1 + \exp(-t)}\right)\left(\frac{1}{1 + \kappa^2 + \lambda^2}\begin{bmatrix} 2\kappa \\ 2\lambda \\ 1 - \kappa^2 - \lambda^2 \end{bmatrix}\right) \tag{4.11}$$

where $\kappa, \lambda \in \mathbb{R}$ are the stereographic coordinates encoding the direction of b and $t \in \mathbb{R}$ is the length parameter. It is clear that the length of the vector is expressed using the logistic function. As a result, the Jacobian of the baseline will be the following:

$$\nabla b = \left[\underbrace{-\frac{1}{\|b\|}\begin{bmatrix} b_0^2 - b_2\|b\| - \|b\|^2 & b_1 b_0 \\ b_1 b_0 & b_1^2 - \|b\|b_2 - \|b\|^2 \\ b_0(\|b\| + b_2) & b_1(\|b\| + b_2) \end{bmatrix}}_{\frac{\partial b}{\partial(\kappa,\lambda)}} \quad \underbrace{\frac{(\|b\| - \beta)(\|b\| - \alpha)}{\|b\|(\beta - \alpha)} b}_{\frac{\partial b}{\partial t}} \right] \tag{4.12}$$

where b_0, b_1, b_2 are the components of the baseline vector. It is worth noting that the Jacobian contains rational expressions of the baseline components exclusively (i.e., the parameters do not appear anywhere).

4.4.2 Parametrization of rotation and baseline in the essential matrix

Consider the primary definition of the essential matrix:

$$E = R^T [b]_\times \tag{4.13}$$

where R contains the relative orientation frame arranged column-wise and $[b]_\times$ is the cross product matrix of the baseline vector. The derivatives of the essential matrix in terms of the stereographic parameter vector $\psi = (x, y, z)$ are the following:

$$\frac{\partial E}{\partial \psi_i} = \frac{\partial R}{\partial \psi_i}^T [b]_\times \tag{4.14}$$

where $\psi_i, i = 1,2,3$ are the components of ψ. The derivatives of the rotation matrix can be obtained analytically as simple polynomial expressions of the quaternion components according to equations (A.31-25) and (A.44-46) in section 4.3 of Appendix A.

The unit-norm baseline is parametrized by $(\kappa, \lambda) \in \mathbb{R}^2$ as described in the previous section. The derivative of E in terms of κ, λ can be obtained according to equations (A.44-46) as follows:

$$\nabla_{\kappa,\lambda} E = R^T \left[\sum_{i=1}^{3} G_i \frac{\partial b_i}{\partial \kappa} \quad \sum_{i=1}^{3} G_i \frac{\partial b_i}{\partial \lambda} \right] \tag{4.15}$$

where b_i are the components of b and G_i are the so-called Lie infinitesimal generators of the 3D special orthogonal group $SO(3)$:

$$\{G_1, G_2, G_3\} = \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right\} \tag{4.16}$$

And the Jacobian of b in terms of the stereographic coordinates (κ, λ) can be obtained according to equations (A.44-46) in the 2D case:

$$\nabla b = \begin{bmatrix} b_0^2 - b_2 - 1 & b_1 b_0 \\ b_1 b_0 & b_1^2 - b_2 - 1 \\ b_2(1 + b_0) & b_2(1 + b_1) \end{bmatrix} \tag{4.17}$$

What is of particular interest in the above is that the derivatives are all simple polynomial expressions which can be easily hard-coded in a computer program. Furthermore, according to Lui and Drummond[13] (Lui and Drummond 2007), in the 5-point case, the iteration should generally converge to a minimum in the context of RANSAC.

4.5 The solution to Wahba's problem: Absolute orientation in a nutshell

Obtaining the rotation matrix that maximizes the alignment of multiple directions in space is a major requirement in many computer vision applications. Although this problem was first introduced to the aerospace engineering research community by Grace Wahba in 1965 (Wahba 1965), it however became known to computer vision researchers as the problem of *absolute orientation* by Berthold Horn 22 years later (Horn 1987). Suppose that for a set of known direction (unit) vectors $f_1, f_2, \dots, f_n \in \mathbb{R}^3, n \geq 3$, the corresponding unit vectors $d_1, d_2, \dots, d_n \in \mathbb{R}^3$ in an unknown coordinate frame are given. Provided that at least one triplet of linear independent vectors exists in the data, then the problem of absolute orientation is defined as follows:

$$\begin{gathered} \underset{R}{minimize} \left\{ J = \sum_{i=1}^{n} \|Rf_i - d_i\|^2 \right\} \\ subject\ to{:}\, R^T R = I \ \ and \ \ det(R) = 1 \end{gathered} \tag{4.18}$$

Equivalently, the problem can be alternatively cast as a maximization:

[13] In their paper, Lui and Drummond propose a parametrization of the essential matrix and run the Gauss-Newton method in the 5-point case without however clarifying why the solution is constrained to 5 points only. The most plausible explanation is that their parametrization produces a cost function expressions that requires linear time to evaluate at each step of the non-linear optimization.

$$\underset{R}{maximize}\left\{J=\sum_{i=1}^{n} d_i^T R f_i\right\}$$
$$subject\ to\colon R^T R = I\ \ and\ \ det(R)=1 \tag{4.19}$$

The cost function of (4.19) is the most suitable starting point for obtaining an analytical solution for R. Let $q=(q_0,v), q_0\in\mathbb{R}, v\in\mathbb{R}^3$ be a unit quaternion that corresponds to R. Then, substituting R from equation (4.2) yields the following expression for the cost function:

$$J=\sum_{i=1}^{n} d_i^T\big((2q_0^2-1)I_3+2vv^T+2q_0[v]_\times\big)f_i$$

$$\Leftrightarrow J=\sum_{i=1}^{n}\left(\underbrace{(2q_0^2-1)d_i^T f_i}_{q^T A_1^{(i)} q}+\underbrace{v^T(2d_i f_i^T)v}_{q^T A_2^{(i)} q}+\underbrace{2q_0 d_i^T[v]_\times f_i}_{q^T A_3^{(i)} q}\right) \tag{4.20}$$

It is relatively easy to discern that each term of the sum is in turn, a sum of three quadratic terms $q^T A_1^{(i)} q$, $q^T A_2^{(i)} q$ and $q^T A_3^{(i)} q$. The three matrices $A_1^{(i)}$, $A_2^{(i)}$, $A_3^{(i)}$ are computed as follows:

$$A_1^{(i)}=d_i^T f_i\begin{bmatrix}1&0&0&0\\0&-1&0&0\\0&0&-1&0\\0&0&0&-1\end{bmatrix} \tag{4.21}$$

$$A_2^{(i)}=\begin{bmatrix}0&0_{1\times3}\\2f_i\times d_i&0_{3\times3}\end{bmatrix} \tag{4.22}$$

$$A_3^{(i)}=\begin{bmatrix}0&0_{1\times3}\\0_{3\times1}&2d_i f_i^T\end{bmatrix} \tag{4.23}$$

The cost function in (4.20) now becomes quadratic in q:

$$J=q^T\underbrace{\left[\sum_{i=1}^{n}\Big(A_1^{(i)}+A_2^{(i)}+A_3^{(i)}\Big)\right]}_{C}q=q^T C q \tag{4.24}$$

Since J is a quadratic, C can be replaced by the respective symmetric matrix $D=\frac{(C+C^T)}{2}$ in the cost function. Thus, the maximization problem of (4.19) now becomes:

$$\underset{q}{maximize}\{J = q^T D q\} \\ subject\ to: q^T q = 1 \tag{4.25}$$

The solution that follows, to the best of my knowledge, was first publicly introduced to the aerospace engineering community by Davenport in 1968 (Davenport 1968, Davenport 1971)[14]. The exact same solution was introduced to the compute vision community by Berthold Horn 19 years later in his famous paper on the *solution of absolute orientation using quaternions* (Horn 1987). With this necessary brief historical review in place, consider now the Lagrangian of the problem in (4.25):

$$\mathcal{L} = q^T D q + \lambda(1 - q^T q) \tag{4.26}$$

where $\lambda \in \mathbb{R}$ is a Lagrange multiplier associated with the unit-norm constraint. Taking the derivative of $\mathcal{L}$ in terms of q and setting it to zero yields:

$$Dq = \lambda q \tag{4.27}$$

Evidently, the solution is an eigenvector of D. Since D is symmetric, it follows that its eigenvectors will form an orthonormal basis of $\mathbb{R}^4$. Thus, with a simple substitution in (4.26), it becomes clear that the solution should be the eigenvector that corresponds to the largest eigenvalue.

4.6 Summary

Two minimal orientation parametrization schemes were introduced and compared in this chapter: The traditional and somewhat popular axis-angle approach and the parametrization using stereographic projection of a plane onto the quaternion sphere. Both schemes present degeneracies in one point of the respective parameter domains. The axis-angle parameters are ambiguous for near-zero angles and therefore the derivatives in this region are either very large or non-existent. On the other hand, the stereographic projection parameters cannot represent the center of projection. Strictly in terms of rotations, this is not an issue, since all rotations are well represented in one hemisphere and therefore the rotation that corresponds to the center of projection is also the point $(0, 0, 0, 1)$ on the surface of the sphere. Both degeneracies appear in practice not very often and when they do, the efficient

[14] Although Davenport in his 1968 book does not explicitly mentions quaternions, his solution to the problem of absolute orientation, in all practical respects makes use of the axis-angle parametrization which eventually yields the solution as a 4D vector with a unit-norm constraint.

work-around is to either perturb the respective quaternion to a very close neighboring point or simply choose to represent the so-called ”shadow” which is the negated quaternion.

A significant advantage of the stereographic projection approach is that it yields a rational parametrization of the quaternion sphere, while the axis-angle parameters involve trigonometric functions in the respective expressions. Furthermore, the quaternion Jacobian contains polynomial expressions of its components. This is a much more compact and succinct expression which not only promotes numerical accuracy, but it also reduces the complexity of the expressions in the derivatives. Moreover, Gauss-Newton iteration results indicate that under stereographic projection parametrization, convergence is relatively faster for the majority of random optimization runs.

Stereographic projection is also an elegant way of imposing norm constraints in the context of optimization problems. Typically, norm constraints pose a problem in iterative optimization because they make it hard for the process to “wander” in feasible space. Stereographic projection eliminates norm constraints and yields an unconstrained optimization problem which is guaranteed to converge to a feasible solution.

Chapter 5

Overview of implementation strategies for SLAM

Although rarely directly addressed in literature, implementation requires the greatest of effort in the development pipeline of visual SLAM applications, not only because of the complexity of the underlying mathematical models, but also because of the multiplicity of ways that these algorithms can be placed together in a functioning framework. It is only during implementation that issues that are generally regarded as trivial and are almost never addressed in literature, become very important for the overall performance of the framework and the choice of the pertinent data structures as well as the respective "book-keeping" algorithms makes a huge difference in the outcome. It is worth noting here that, interfacing with a public software library for matrix algebra and vision primitives is absolutely necessary in large scale SLAM applications and therefore the interface of the library can impose constraints on the development process. An example of such a constraint is the case of the LK tracker implementation in OpenCV: Although the tracker uses local image patches as reference for searching, it nevertheless requires an entire reference image as input instead of a list of patches. Under the *overlapping scene* implementation model for SLAM, this limitation imposed the need to address feature tracking with multiple calls to the LK tracker in order to accommodate features that were detected in different frames. The *disjoint scene* SLAM model implements *trailing tracking* (i.e., tracking on a frame-to-frame basis) and therefore does not require multiple calls to the LK tracker. The trade-off however is that certain camera views have weak correlations with others due to the lack of sufficient numbers of mutually visible features.

One of the issues that mathematics do not address, yet it is of outmost importance in practice, is feature management during execution either as lists of measurements, or as 3D points in the map. Visual features have a relatively short life-span in regards to the overall video sequence. As the camera moves throughout the environment, features sooner or later will travel out of the field of view or get rejected as outliers. It is therefore imperative to retain a sufficient number of visible inliers at all times in order for SLAM to be able to perform posterior updates robustly. Considering that patch matching in the context of natural environments is likely to produce spurious results, it is assumed that the new features obtained by means of some detector do not match any previous ones. In other words, points are not treated as landmarks in the global sense but rather as short-lived distinctive features over a small sequence of consecutive frames.

One of the most important goals in terms of functionality of the SLAM framework is to create, manage and update the data structures associated with the map and the respective feature measurements in the sequence; furthermore, to synchronize the invocation of functions associated with motion prediction, measurement handling and map updates. There are two dominant approaches to the implementation of the SLAM framework in this thesis: a) The *disjoint scene* SLAM pipeline in which feature detection occurs only in specific frames and only after the number of visible features has dropped below a threshold and, b) the *overlapping scene* SLAM pipeline in which new features are added to the map on a frame-to-frame basis.

5.1 Scene objects

For the needs of the analysis in the following sections, the notion of *scene* is briefly introduced. A scene is a subset of frames of the sequence in all of which, a (user-defined) minimum number of inliers are visible. This definition suggests that features can be visible across different scenes. Figure 5.1 illustrates the concept of a general scene. As new features can be arbitrarily added to the map, the dashed ellipses indicate the one that are visible in each of the 3 camera views. The features shown inside the circle with the bold outline define the scene, since they are visible in all 3 views.

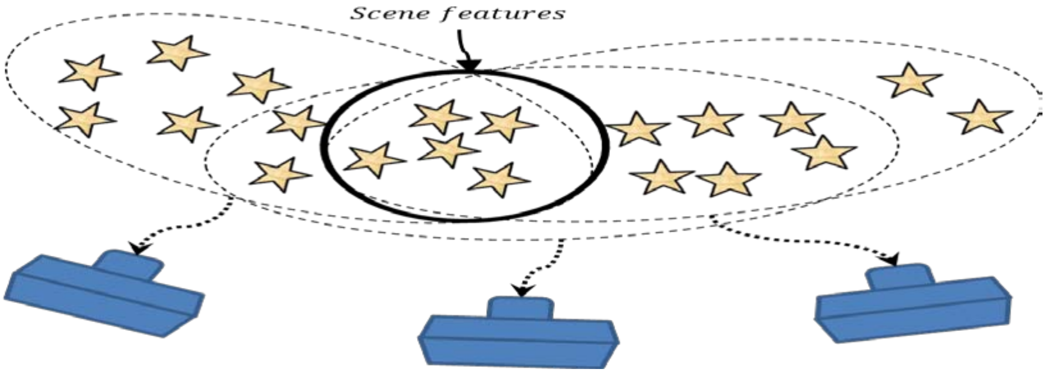

Figure 5.1. Graphical illustration of a scene composed of 3 views.

5.2 Disjoint scene SLAM

The disjoint SLAM pipeline is a relatively simplistic approach to feature management with certain advantages, especially from an implementation point of view: A number of points is obtained from a frame and thereafter these points are tracked in the subsequent frames until the number of inliers falls below a certain threshold. Once this happens, a completely new set of features is detected in the current frame and a new scene object is created. The greatest advantage of the disjoint scene approach is the ease of implementation, since the map is updated upon the first frame of the scene and only these new features are tracked until the end of the scene without any intermediate additions. The apparent downside here is that drift is likely to appear in scene joints, since there is no measurement overlap between the last frame of the previous scene and the first frame of the new scene. On the other hand, the disjoint scene model is ideal for cases where relative pose odometry is employed since the algorithm itself examines the features only in the context of two consecutive views. Figure 5.2 uses an undirected graph illustration in which edges denote the disjoint visibility of features in groups of consecutive frames comprising adjacent scenes. Notice how all features of the first scene are visible in the first node, but they “dissipate” in subsequent views (2 and 3); eventually, the last node in the scene (3) observes only one landmark and therefore a new scene is created with “freshly” detected features (red stars).

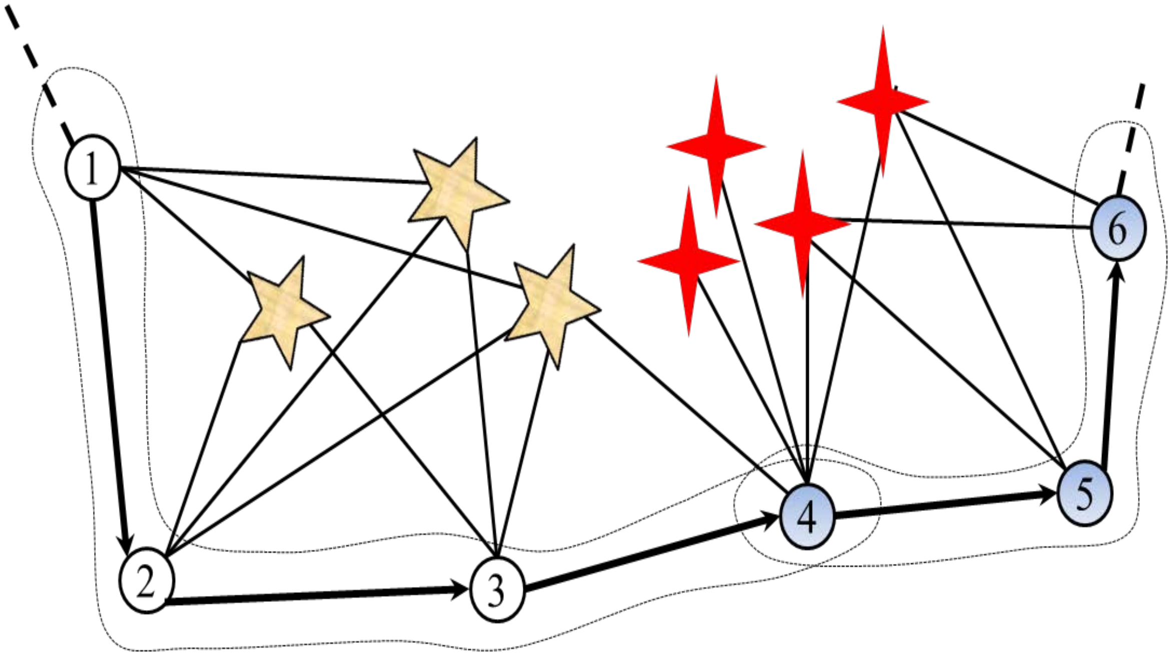

Figure 5.2. The disjoint scene SLAM conceptual model loosely depicted as an undirected graph; edges between camera pose nodes (circles) and landmarks (stars) denote observability.

Amongst other tasks involving map updates and measurement book-keeping, the scene object marks the features that the tracker has most recently failed to track. Only the “surviving” valid features are thereafter tracked in the next captured frame. Tracking continues until the number of features drops below a minimum threshold and thereafter, a new scene is created (which implies a brand new set of features). The process is illustrated in the flowchart of Figure 5.3.

The initial frame of a scene remains stored until the scene is destroyed. Feature tracking can be done using feature locations either in the initial or the previous frame. This approach accommodates the requirement of the OpenCV LK tracker for a reference image as input instead of a set of patches per feature. Alternatively, the scene object may use feature location estimates provided externally (e.g., the back-projected estimates of the map by the SLAM algorithm).

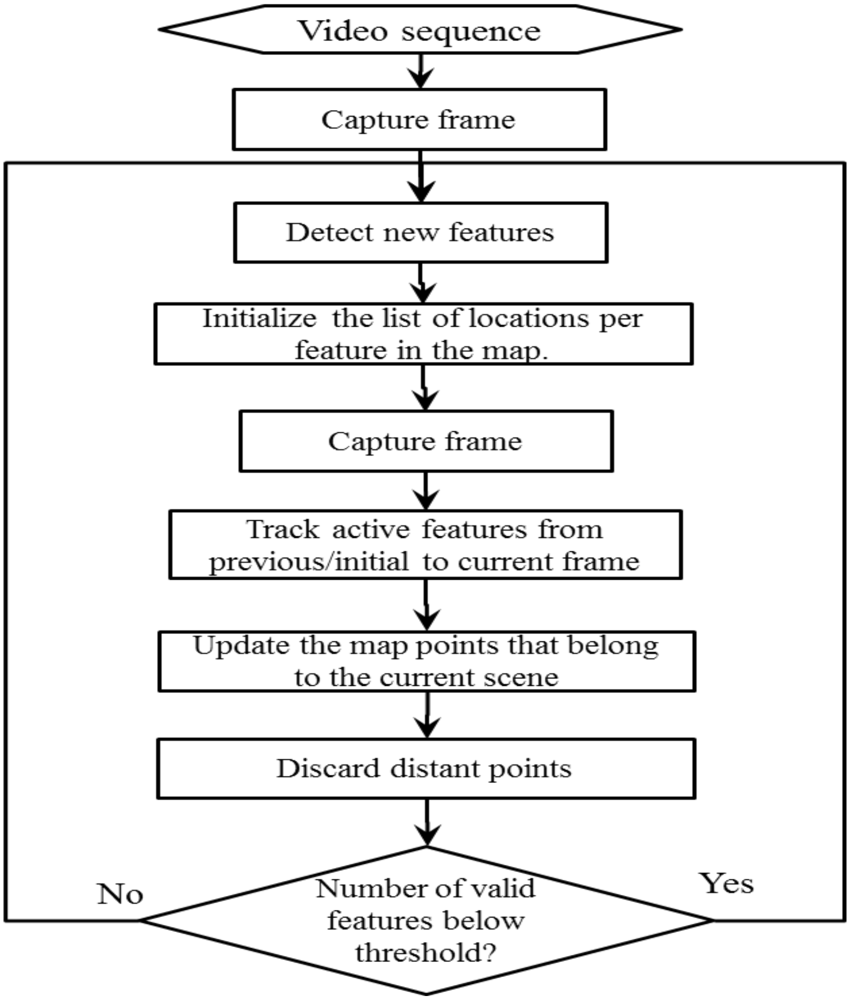
Video sequence
Capture frame
Detect new features
Initialize the list of locations per feature in the map.
Capture frame
Track active features from previous/initial to current frame
Update the map points that belong to the current scene
Discard distant points
Number of valid features below threshold?
No
Yes

Figure 5.3. Flowchart illustrating the general operation concept of disjoint scene SLAM.

5.3 Overlapping scene SLAM

The overlapping scene approach corresponds to the standard visual SLAM execution paradigm. Initialization is similar to the case of disjoint scenes: A set of features are detected in the first frame and tracked to the second in order to obtain a reconstruction that will serve as the initial map. The features are thereafter tracked to subsequent frames until they are discarded as outliers. As new measurements are obtained from incoming frames, new features are also detected. Using the existing map and the respective tracked features locations, the camera pose is estimated and based on this new pose, the features detected in the previously captured frame are triangulated and added to the map. Figure 5.4 illustrates how new features are detected and tracked through the sequence, thereby yielding a "tighter" network of measurements.

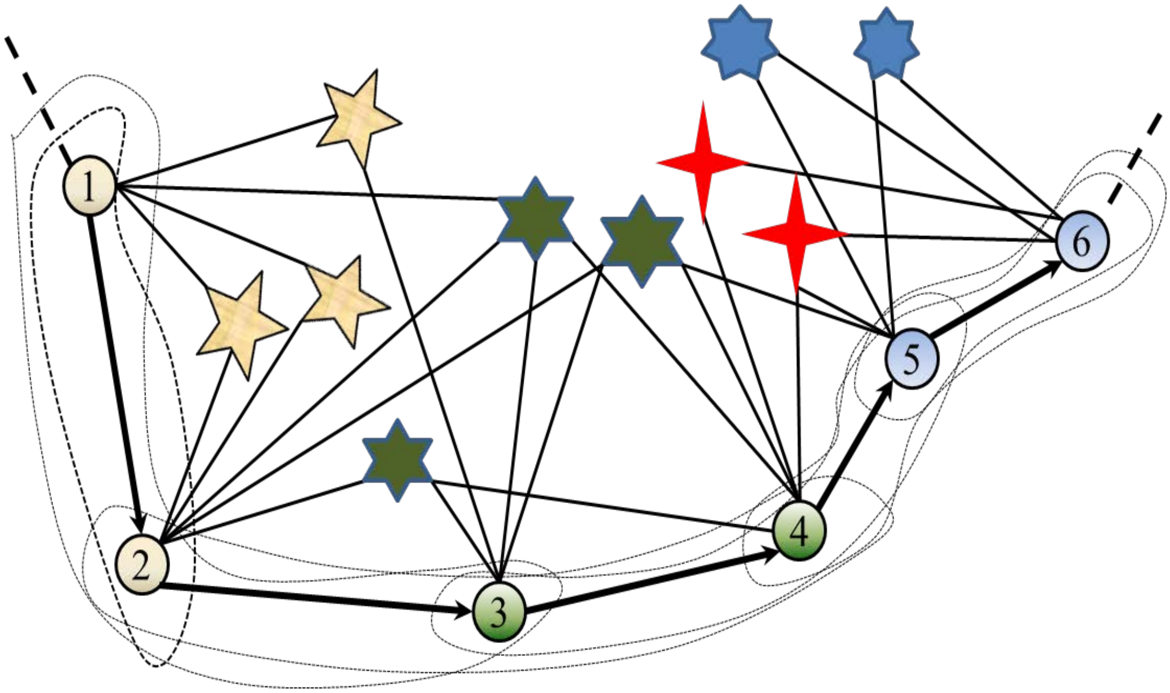

Figure 5.4. The overlapping scene SLAM conceptual model loosely depicted as an undirected graph.

5.3.1 Interfacing with the LK tracker in OpenCV

As mentioned earlier, the LK tracker in OpenCV requires the entire reference image as input, while the respective features are specified by their coordinates in the image. Although this is not a problem in the case of disjoint scenes, it however poses a serious functionality problem when one needs to track features detected in multiple frames as it happens in the case of overlapping scenes. The natural workaround is to generate separate lists with features detected in the same frame and invoke the LK tracker multiple times. Although this is a reasonable strategy at first glance, it however incurs a significant memory burden to the application, since the "home-frames" of all currently active (inliers) features will have to be stored in memory; depending on the currently active features, the number of stored images can be prohibitively high. The final solution is essentially a compromise in which only the features detected in two previous frames are tracked directly from their frames of origin; all other active features are tracked from their previously measured locations in the two most recent frames. This solution is practical and deals with the problem of having to store large numbers of images, but on the other hand, it opens the back-door for drift contamination in the measurements. To minimize the presence of drift, a maximum feature “life-span” is specified (typically, 3-5 frames). The process is described in the flowchart of Figure 5.5.

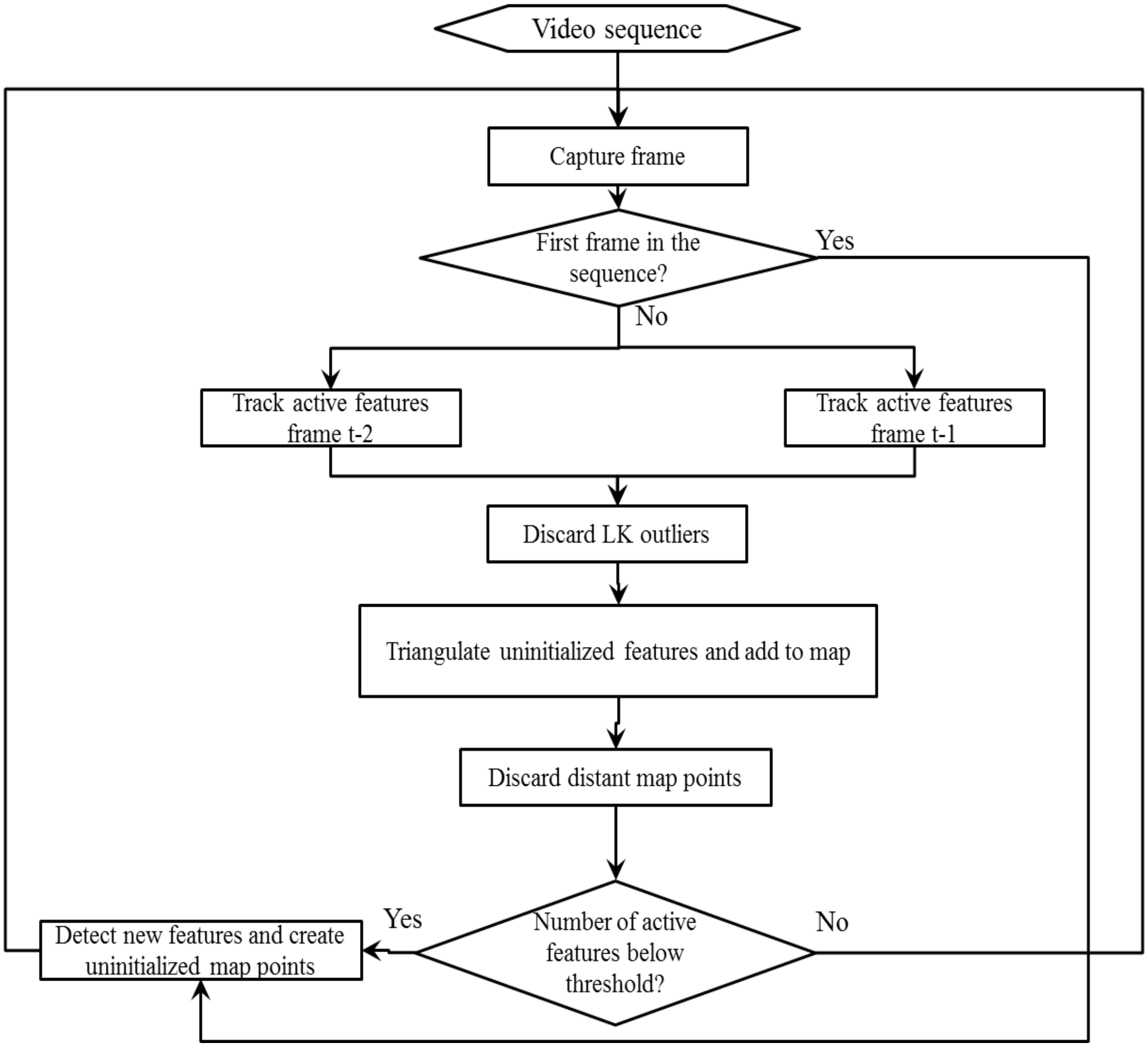
Video sequence
Capture frame
First frame in the sequence?
Yes
No
Track active features frame t-2
Track active features frame t-1
Discard LK outliers
Triangulate uninitialized features and add to map
Discard distant map points
Yes
Number of active features below threshold?
No
Detect new features and create uninitialized map points

Figure 5.5. Flowchart illustrating the general operation concept of overlapping scene SLAM.

5.4 Sparse optical flow vs Local descriptor matching

The level of accuracy in feature matching has a tremendous impact on motion and map estimation. The OpenCV LK tracker is admittedly a very good choice for such an algorithm; however, the downside of this approach in regards to visual SLAM is that the OpenCV implementation does not address each feature as a separate patch, but rather as a location in an input image, thereby making it difficult to track multiple features based on their original appearance. An alternative approach which addresses this problem directly at its root is *descriptor matching*.

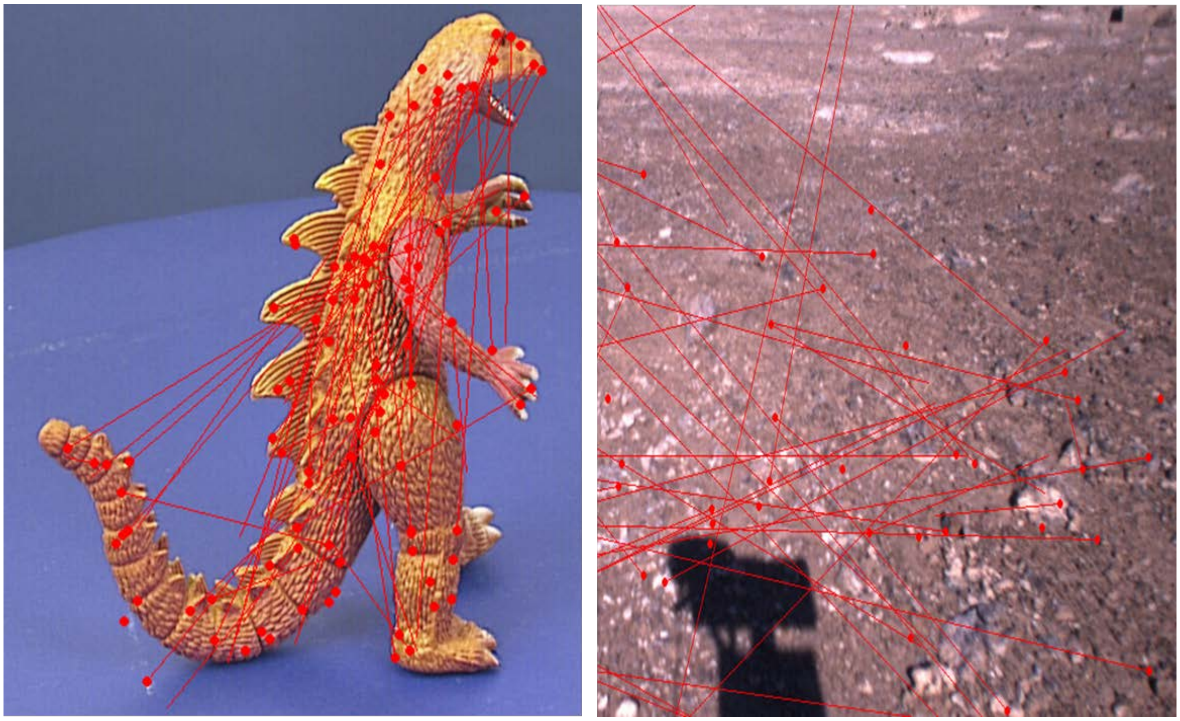

Figure5.6. Descriptor matching with the Hannover dinosaur (left) and the Devon island Mars rover dataset (Furgale, Carle et al. 2012) on the right.

Descriptor matching refers to the process of independently detecting features in a query image and thereafter attempting to match an existing list of descriptors with the descriptors of the new features. This approach has the two significant advantages: a) It simply requires the storage of a descriptor instead of the entire reference image and, b) Successful matching is usually accurate with respect to the quality of the descriptor. The great weakness of such an approach is that the matching process does not take any image topological criteria into consideration, which means that erroneous matches can produce arbitrarily large errors. Figure 5.6 illustrates numerous mismatches presenting significant error with descriptor matching over features located in regions with similar appearance such as soil or the dinosaur’s skin. It should be noted however that this behavior was observed in OpenCV implementations of several types of detectors.

It becomes evident that, although most of the largely mismatched pairs in Figure 5.6 can be generally spotted and subsequently discarded for being inconsistent with epipolar geometry, it will only take a few such misclassified outliers to significantly divert the new camera pose estimate and there is no guarantee that the outlier rejection scheme will eliminate all of them. The obvious solution to this problem is *local descriptor matching*. As the term implies, local descriptor matching refers to the same process of feature detection and matching in the much narrower scope of an image region. Typically, these regions are chosen in the neighborhood of the feature, or in the neighborhood of its predicted location (by means of a motion model). This way, the possibility of large errors is minimized. We observed that the process very frequently tends to match neighboring features with the very same newly detected feature, especially when the texture is cluttered (e.g., leaves, grass, etc.). In practice, this is highly detrimental for SLAM, even more than the casual shift observed with the LK tracker. In particular, several "fused" matches can lead to a large deviation in the pose estimate which manifests with large scale discrepancies in the new map points as opposed to the existing ones; one the other hand, slow shift is relatively acceptable because its effects can be partially mitigated by local reprojection error adjustment.

5.5 Handling measurements and map "maintenance"

Acceptance of new measurements is done on the basis of their consistency with epipolar geometry and their classification as valid optical flow by the LK tracker. Since no motion model is provided, the only information as to the validity of the correspondences originates in the tracking error and their individual fitness with respect to the estimated fundamental matrix.

The LK tracker returns a success flag for each tracked feature which typically becomes false when the point "slides" off the image or certain other computation-related criteria are violated. Based on the success flag, a flow vector maximum length constraint is imposed to enforce the relatively slow and constant velocity assumption. For the remaining features, the fundamental matrix is computed using random sampling consensus (RANSAC) (Fischler and Bolles 1981) or an inference based Markov chain Monte Carlo (MCMC) method (Terzakis, Culverhouse et al. 2015). The set of outliers obtained by the estimation algorithm are removed from the "surviving" features. Figures 5.7 and 5.8 illustrate tracking results before and after outlier rejection.

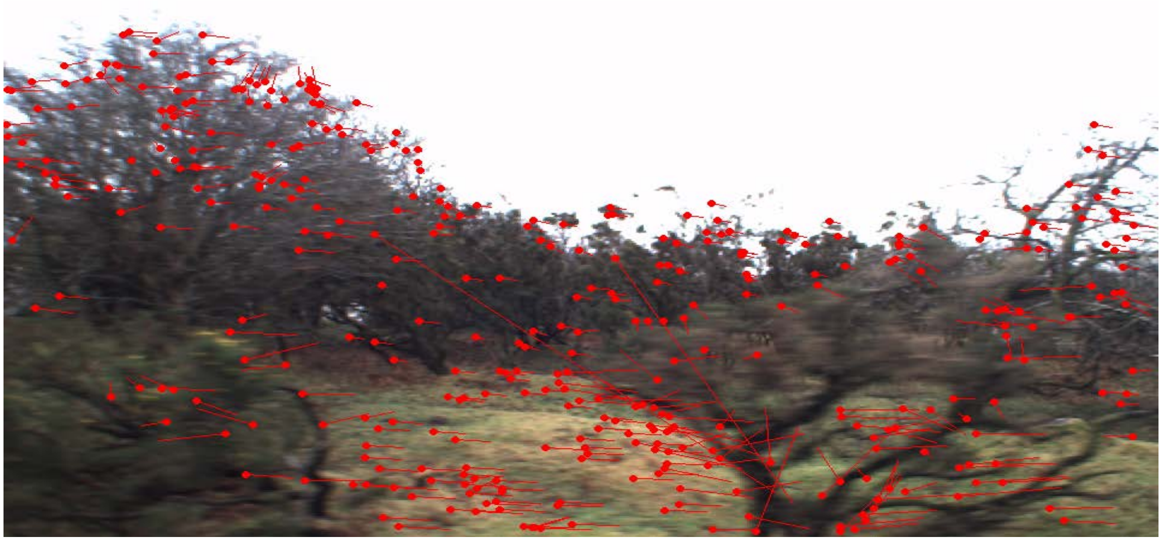

Figure 5.7. The optical flow vectors after the removal of unsuccessfully tracked features according to the LK tracker.

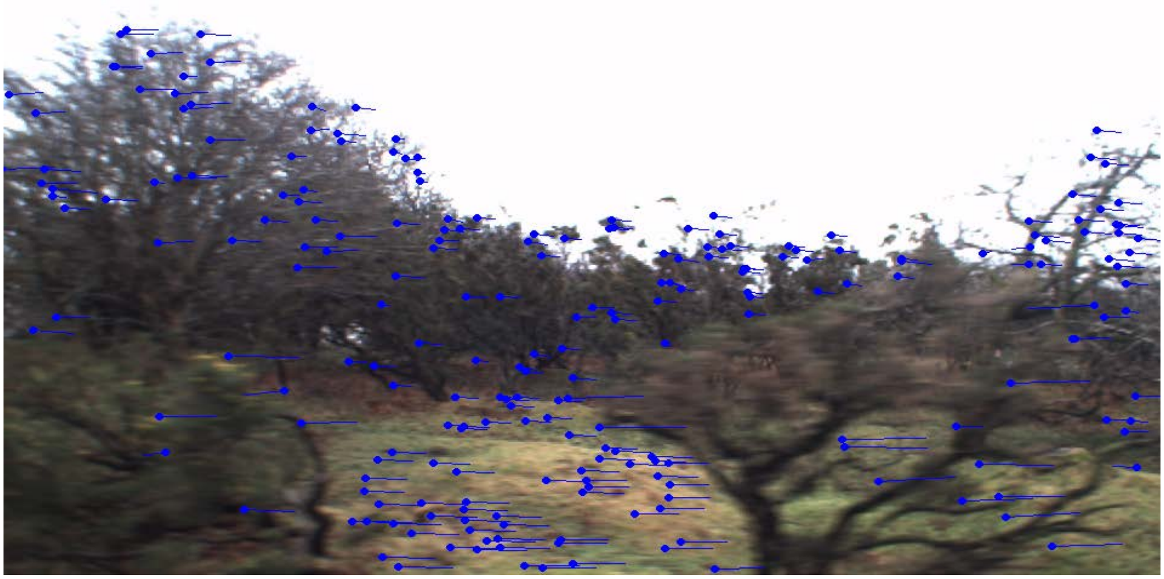

Figure 5.8. The remaining optical flow following removal of outliers inconsistent with the RANSAC/MCMC based computation of the fundamental matrix.

5.6 Summary

Management of tracking throughout frame sequences is a matter of significant practical importance in the implementation of visual SLAM. Organizing and book-keeping of the feature locations in the scene directly affects not only the quality of the results, but also the execution time of SLAM posterior computation. There can be many variations as to how feature management is performed, but there are certain invariant aspects to it, including the periodic detection of new features as well as the need to reject outliers as more frames are added to the scene. In the overlapping scene paradigm, scenes are overlapping in the sense that new features are being detected before the active pool of interest points has been depleted. In this thesis, in SLAM applications where gyroscopic sensors are used, the disjoint scene paradigm is adopted, since orientation is known with relatively high accuracy and it is not necessary to detect new features very often; the overlapping scene paradigm is employed in applications where the SLAM is relying only on a camera and therefore feature tracking is effectively the only input to the process.

Feature management also involves outlier rejection. It is worth re-iterating the outlier rejection pipeline here: The first round of measurement handling involves the removal of outlier features that were flagged by the LK tracker. Subsequently, the fundamental matrix is calculated with a RANSAC approach on the remaining inliers and the flagged outliers are removed from the list. With the measurements at hand, the new pose and the depth of the previously detected features is estimated and a weighted iterative refinement of the reprojection error is performed using a robust estimator. Features with depth that deviates significantly from the median depth in a specific view are discarded on the grounds that, as illustrated in Figure 3.16, they entail higher uncertainty which could affect subsequent pose estimation.

Chapter 6

Relative pose odometry

A quick way of obtaining camera trajectory estimates along some executed route, from a design and implementation point of view, is to simply obtain relative pose estimates between consecutive frames and thereby compute the position of the camera in the world as the integral of those estimates. This process is usually referred to as *relative pose odometry* and provides a quick solution to the localization problem.

Relative pose odometry estimates can be easily afflicted by drift, since the global pose is obtained as an integral of noisy pairwise rigid transformations. It follows that in order to apply this method, certain standards in terms of tracking accuracy must be met, and/or additional quality measurements in regards to camera motion should be provided. In the context of the general problem statement of this thesis, relative pose odometry can be a reasonable solution for short-term vehicle localization in the presence of a reliable gyroscopic sensor that would eliminate orientation from the vector of unknowns. Although angular velocity readings are stochastic and therefore orientation will eventually present drift, the amount of noise with which they are contaminated is significantly small and therefore the orientation integral can be very reliable for the greatest part of the sequence.

6.1 Relative pose odometry: A probabilistic approach

Consider an application in which 3D mapping is not required and only camera pose is of significance; furthermore, suppose that efficient relative pose tracking estimates are available for each pair of consecutive frames. Then, absolute pose can be incrementally estimated from the previous pose, using a pose transition and a measurement model that depends on the feature locations in the previous and current frame. Figure 6.1 illustrates the relative pose tracking approach with a Markov random field (MRF). It is clear that the 4-cliques formed between the normalized Euclidean projections m_t^+, m_{t+1}^-, x_t and x_{t+1}

correspond to the measurement constraint, while the 3-cliques between u_t, x_t and x_{t+1} correspond to the state transition constraint.

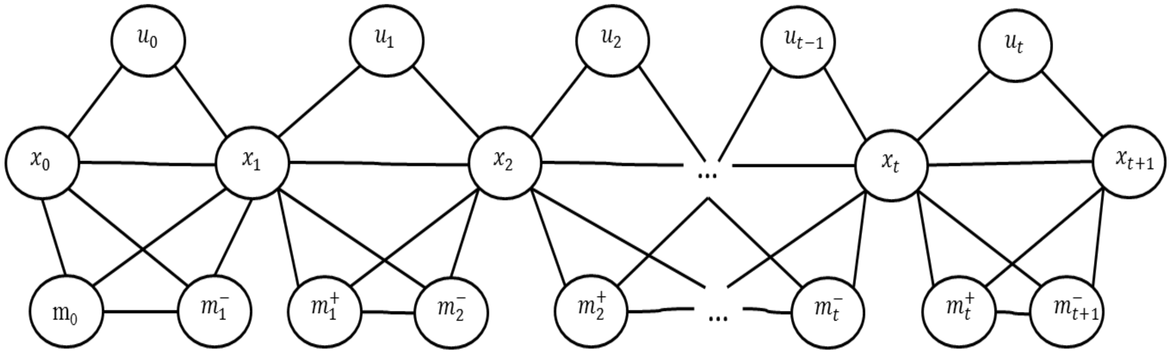

Figure 6.1. A Markov network illustrating the concept of visual odometry as SLAM without mapping.

The main assumption in relative pose based odometry is that for every pair of consecutive poses there exist a set of point correspondences in the respective frames which is independent of correspondences in the previous pair of frames. In particular, for a frame captured at time t, there exist two sets of features with normalized Euclidean projections m_t^- and m_t^+; the former set gets matched with interest points in the previous frame, while the latter set will be tracked to the next frame. In other words, one may think of the tracking scheme in terms of independent, new feature detections at each time step. In practice, it is simply assumed that the tracked feature locations in a frame can also be treated as a new set of interest points in the next time instance and therefore can be tracked to the next frame independently of what happened in the past.

The obvious consequence of this pairwise independence assumption is that the measurement likelihood does not, in practice, account for the accumulated uncertainty in feature locations. This is an implementation issue that can be dealt with by incorporating the variance of m_t^- into the measurement constraint corresponding to the next pair of frames at times t and $t+1$. Alternatively, one may choose to ignore the accumulated variance in the feature locations and that could be a reasonable strategy in natural sceneries, since the drifted location of a feature can be regarded as a new interest point altogether, if the respective patch can be reliably tracked in the next frame.

Let $x_t = [\eta_t^T \quad s_t^T]^T$ where $\eta_t,\ s_t \in \mathbb{R}^3$ are the orientation and position (in world coordinates) vectors respectively. Using the projections of a world point onto m_{t+1}^- and m_t^+, the following relationship is obtained:

$$m_{t+1}^- = \frac{1}{1_z^T R_t^T (d_t m_t^+ - b_t)} R_t^T (d_t m_t^+ - b_t) \tag{6.1}$$

where R_t is the rotation matrix that represents the change in camera orientation from time t to time $t+1$, d_t is the depth of the feature in the t^{th} camera frame, $b_t = R_t(s_{t+1} - s_t)$ is the baseline vector between the camera poses at times t and $t+1$, expressed in the t^{th} camera frame and $1_z = [0 \quad 0 \quad 1]^T$. Considering that m_{t+1}^- and m_t^+ are measured quantities, it follows that equation (6.1) contributes a non-linear quadratic constraint to the posterior in which the only stochastic unknowns are the baseline b_t, the depth d_t and the relative orientation rotation matrix R_t.

6.1.1 Integrating the pose posterior

Suppose that the marginal distribution $p(x_t | m_{1:t}^-, m_{o:t-1}^+)$ is known. Then, it becomes evident from Figure 6.1 that the marginal distribution of x_{t+1} given all past and present measurements is given by,

$$\begin{aligned} p(x_{t+1} | m_{1:t+1}^-, m_{o:t}^+) \\ \propto \int_{-\infty}^{+\infty} \varphi(m_{t+1}^-, m_t^+, x_{t+1}, x_t) p(x_{t+1} | x_t, u_t)\, p(x_t | m_{1:t}^-, m_{o:t-1}^+) dx_t \end{aligned} \tag{6.2}$$

where $\varphi(m_{t+1}^-, m_t^+, x_{t+1}, x_t)$ is a factor associated with the observation model of equation (6.1), $p(x_{t+1} | x_t, u_t)$ is the pose transition Gaussian conditional distribution and $p(x_t | m_{1:t}^-, m_{o:t-1}^+, u_{1:t})$ is the pose belief (also normal) at time t. To compute the marginal one simply needs to obtain an estimate for the joint of x_t, x_{t+1}. Then, it follows that the estimate of x_{t+1} will be described by the respective mean and covariance in the joint. The computation of the joint entails the optimization of a quadratic function:

$$\begin{aligned} q_t(x_{t+1}, x_t) = log\varphi(m_{t+1}^-, m_t^+ x_{t+1}, x_t) \\ + \left(x_{t+1} - g_t(x_t, u_{t-1})\right)^T R_t^{-1} \left(x_{t+1} - g_t(x_t, u_{t-1})\right) \\ + (x_t - \mu_t)^T \Sigma_t^{-1} (x_t - \mu_t) \end{aligned} \tag{6.3}$$

Thus, provided that $log\varphi$ is a non-linear function, the solution for x_{t+1} can be obtained only by means of some method for non-linear optimization of $q_t(x_{t+1}, x_t)$.

6.2 Planar odometry for ground facing camera: The orthogonal Procrustes method

Equation (6.1) associates the relative pose transformation with the projections of observed points in two views. The standard way of resolving relative camera pose is through essential matrix as described in Chapter 3, section 2.4. The projections are typically regarded as normally distributed; however the same does not hold for the camera pose. The cause of this lies with the orthonormality constraints that accompany the rotation matrix which must be imposed either by means of a parametrization scheme, or with the use of Lagrange multipliers in the context of least squares. Under extremely favorable conditions which involve minor or no noise in the data, these constraints can be relaxed and the problem can be resolved linearly. In such cases, orthogonality is imposed a posteriori by obtaining the relative orientation matrix as the closest (in the Frobenius norm) orthonormal matrix to the recovered solution with ordinary LS. This technique is called the *orthogonal Procrustes* method (Wahba 1965, Schönemann 1966) and has been well understood amongst aerospace engineers since the mid-1960s. In the particular case of visual odometry, if the problem of relative pose can be conditioned in such a way as to limit the number of unknowns and at the same time guarantee reasonable percentages of inliers in the projections, then it can be cast as ordinary least squares and thereafter solve for orientation using the orthogonal Procrustes.

Although planar odometry is not directly applicable to the broad category of vision based localization problems related to the objectives of this thesis, it however poses a fine example of a relative pose problem which can be linearized due to only 2 unknown non-zero elements in the rotation matrix and the assumption that all observed points have the same depth (Figure 6.1). Suppose that the camera is approximately moving in a plane parallel to the ground. Also, consider that the height of the motion plane is reasonably close to the ground (up to 2 m), such that all feature points are considered coplanar. It is therefore possible to construct an overdetermined linear system in terms of the baseline and the components of a 2D rotation matrix which can ultimately lead to the recovery of a reasonable estimate of relative camera pose. Figure 6.1 illustrates the concept of the ground facing camera motion.

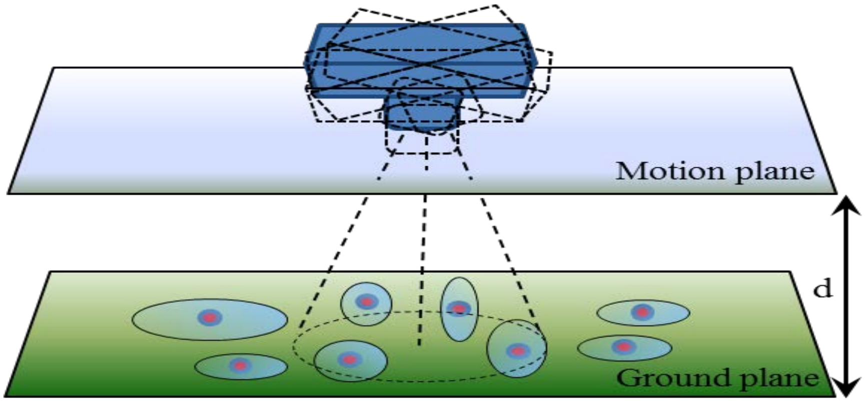

Figure 6.1. Approximate planar motion by a ground facing camera.

In Figure 6.1, the multiple dashed camera outlines and optical axes imply that the camera pose may slightly deviate in orientation (i.e., about the two axes orthogonal to the optical axis) and position (i.e., along the optical axis). The height of the motion plane over the ground is *d*. The canonical ellipses represent the uncertainty in the feature locations due to tracking and projective distortion caused by camera tilts or shifts from the parallel plane.

6.2.1 Estimating unconstrained relative pose with ordinary least squares

Assuming that the actual 3D locations of the features in the real word lie in a plane and that the camera motion is roughly parallel to it, then, accounting for uncertainty due to tracking error and projective distortion, one may consider the space of possible image locations as the interiors of ellipses (Figure 6.1). Moreover, if the image coordinates of the features are Gaussian and independent of each other, then these are canonical ellipses and the respective covariance matrices are diagonal. This is a fairly reasonable assumption which is also convenient for the formulation of a weighted linear LS optimization problem.

Since the camera is moving parallel to the ground and the actual world locations of the visible features are coplanar, they all have the same depth *d*. Moreover, the sought rotation matrix will represent a rotation about the optical axis and therefore will have four unknown components as follows:

$$A = \begin{bmatrix} \alpha_1 & \alpha_3 & 0 \\ \alpha_2 & \alpha_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

Given the above assumptions, equation (6.1) reduces to the following:

$$m_{t+1}^{-} = A^T(m_t^{+} - b_t) \tag{6.5}$$

where d is the common depth for all features. Dropping the + and – superscripts such that $m_0 = m_t^+$ and $m_1 = m_{t+1}^-$ equation (6.5) becomes,

$$m_1 = A^T m_0 + c_t \tag{6.6}$$

where $c_t = [c_x \quad c_y] - A^T b_t$. Now, multiplying by $1_x^T = [1 \quad 0 \quad 0]$ and $1_y^T = [0 \quad 1 \quad 0]$ two equations with distinct unknowns are obtained:

$$1_x^T m_1^N = [\alpha_1 \quad \alpha_2 \quad 0] m_0^N + c_x \tag{6.7}$$

and,

$$1_y^T m_1^N = [\alpha_3 \quad \alpha_4 \quad 0] m_0^N + c_y \tag{6.8}$$

Equations (6.7) and (6.8) correspond to two independent LS problems for α_1, α_2, c_x and α_3, α_4, c_y respectively. Please note here that this separation into two problems is possible only because of the independence assumption on the coordinates of the tracked feature points. In any other case in which the image coordinates of the features are correlated, the two equations should belong to the same LS problem.

6.2.2 Orthogonal Procrustes

Clearly, the optimization is not constrained in terms of the sought rotation matrix and therefore the estimated values for α_1, α_2, α_3 and α_4 will not generally abide orthonormality. It is therefore necessary to obtain the closest rotation matrix R to the recovered estimate A by means of some metric. This is known as the orthogonal Procrustes method, very well known to aerospace engineers since the 1960s (Schönemann 1966, Kristof and Wingersky 1971, Ten Berge 1977). It was not until nearly 30 years later that Horn (Horn, Hilden et al. 1988) (re)introduced the technique to the vision community in his solution for the so-called *problem of absolute orientation* which is essentially a slightly more sophisticated version of Wahba's problem. According to the Procrustean approach, the closest rotation matrix to the solution A of the LS problem defined in equations (6.7) and (6.8) R is given by,

$$R = \Lambda(\Lambda^T\Lambda)^{-\frac{1}{2}} \tag{6.9}$$

And the matrix $(\Lambda^T\Lambda)^{-\frac{1}{2}}$ is defined as follows:

$$(\Lambda^T\Lambda)^{-\frac{1}{2}} = \frac{1}{s_1}v_1v_1^T + \frac{1}{s_2}v_2v_2^T + \frac{1}{s_3}v_3v_3^T \tag{6.10}$$

where s_1, s_2, s_3 are the singular values of Λ and v_1, v_2, v_3 are the eigenvectors of $\Lambda^T\Lambda$. Assuming that Λ will have 3 non-zero singular values, it follows from equations (6.9) and (6.10) that R can be obtained as follows:

$$R = UV^T \tag{6.11}$$

where $\Lambda = USV^T$ is the singular value decomposition (SVD) of Λ.

By assumption, the camera rotates only about the optical axis (z_O); therefore, the rotation angle θ can be unambiguously retrieved from the matrix R. Thus, θ and b can be used as the initial estimates in further optimization of the cost function in equation (6.3).

6.2.3 Assigning variance to the measurements

The variance of optical flow estimates is usually obtained from the LS optimization problem of equations (1.6) and (1.7) (Paragios, Chen et al. 2006). Suppose that optical flow is provided by the LS solution of the form, $v = (W^TW)^{-1}W^Ty$ where $y \sim N(\hat{y}, \Sigma)$ is normally distributed with mean $\hat{y}$ and covariance matrix Σ. Then, it follows that v is also normally distributed and by applying variance propagation properties for linear relationships, the covariance matrix $C(v)$ of v will be given by,

$$C(v) = (W^TW)^{-1}W^T\Sigma W(W^TW)^{-1} \tag{6.12}$$

The covariance matrix of the tracked location is therefore, $Q = C(v)$.

There are cases in which the implementation of the optical flow tracker does not reveal the matrix W, but rather provides optical flow error statistics such as the average absolute difference in pixel intensity between the patches surrounding the interest point. In such cases, a model that describes the relationship between the distance ε from ground truth and absolute intensity error e is fitted by means of regression:

$$\varepsilon = h(e) \tag{6.13}$$

Typically h is linear or quadratic. Training data are obtained by tracking various feature points in a scene and thereafter mapping the observed distances from the ground truth to specific absolute intensity errors. Thus, each absolute intensity error value is mapped to an average squared error. The parameters of the model are then obtained by applying LS regression to the aforementioned mapped pairs. Finally, the variance σ_ε^2 of ε is estimated from the fitted values $h(e_i)$ as follows:

$$\hat{\sigma}_\varepsilon^2 = \frac{1}{N-1}\sum_{i=1}^{N} h^2(e_i) \tag{6.14}$$

where $\hat{\sigma}_\varepsilon^2$ is the unbiased variance estimator and N is the number of data pairs. Assuming that the covariance matrix of the optical flow error vector is diagonal and isotropic, then using a linear approximation of ε in terms of its components, it can be easily proven that the variance in both axes is approximately equal to σ_ε^2.

$$Q \approx \sigma_\varepsilon^2 I_2 \tag{6.15}$$

where I_2 is the 2 × 2 identity matrix.

6.2.4 Optimal solution and outlier screening

The coplanarity of the observed points on the ground plane is a degenerate configuration in epipolar geometry (Torr, Zisserman et al. 1995). Thus, the RANSAC based computation of the fundamental matrix cannot be employed. However, an alternative solution exists, based on the same principle. In particular, the image locations of the features in two views are now related through the ground plane by a projective transformation (homography). The estimation of the homography can also be posed as a quadratic problem and various methods have been proposed for the formulation and optimization of the respective cost function (Ma, Soatto et al. , Hartley and Zisserman 2003); most of these methods can be applied in the context of RANSAC.

6.2.5 Odometry estimates

Various video sequences were taken with a ground facing camera being held at constant height with the optical axis being perpendicular to the ground plane. Since the footage was taken by a person walking while holding the camera, it follows that the trajectory was not precisely planar, nor was the optical axis orthogonal to the ground plane at all times. The method was evaluated for camera only sensory input and unknown motion dynamics. Figures 6.2-5 illustrate odometry estimates for known trajectories and comparisons with ground truth; on the left, the recovered odometry is shown in blue; on the right, the red line indicates GPS based ground truth, scaled to match the estimated trajectory (superimposed) proportions.

The presence of drift in the camera pose estimate becomes obvious in all of Figures 6.2-5, especially in terms of orientation. It was actually observed that the aforementioned error typically peaks when the camera is rotating rapidly around corners in particular. Furthermore, applying RANSAC did not improve the odometry significantly, which leads to the conclusion that orientation is the most "sensitive" estimate in visual planar odometry and, most likely, in all SLAM applications in which the camera is the only available sensor. It is clear that without any other means of knowing orientation, the camera-only estimate drifts quickly in relative pose odometry. Conversely, position estimates tend to be more agile when the camera orientation is approximately fixed and motion involves translation only. This is a general rule which is indirectly ratified by the results of section 6.3 in which the effects of orientation are removed from feature correspondences by means of fairly accurate relative orientation estimates.

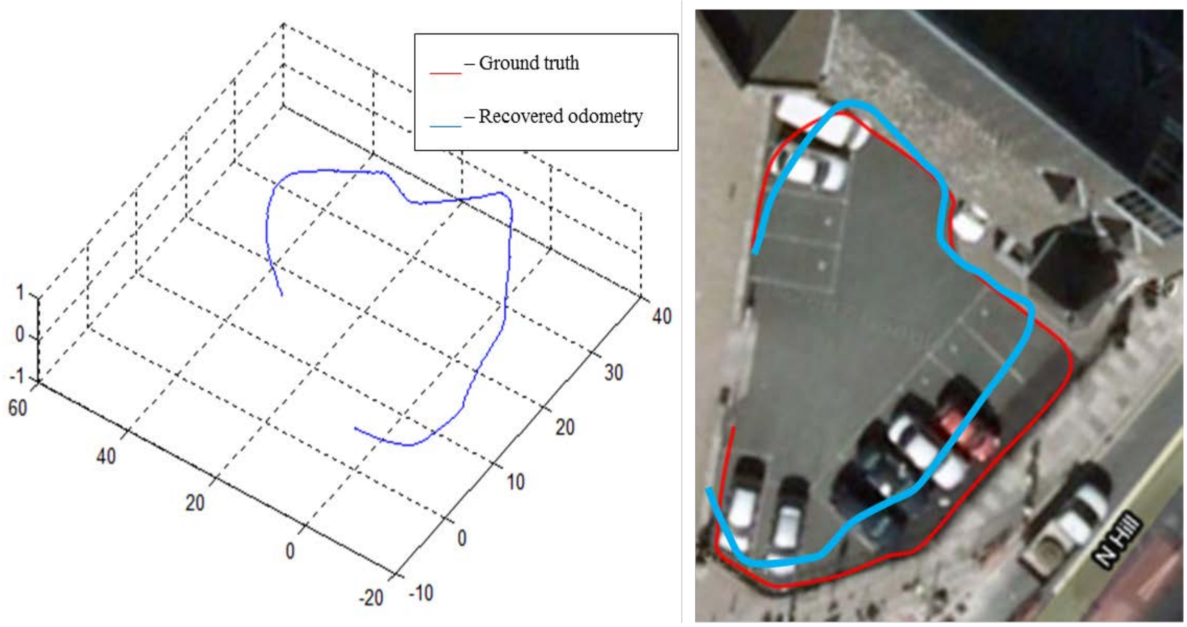

Figure 6.2. Estimated odometry from a walk along the contour of the university parking.

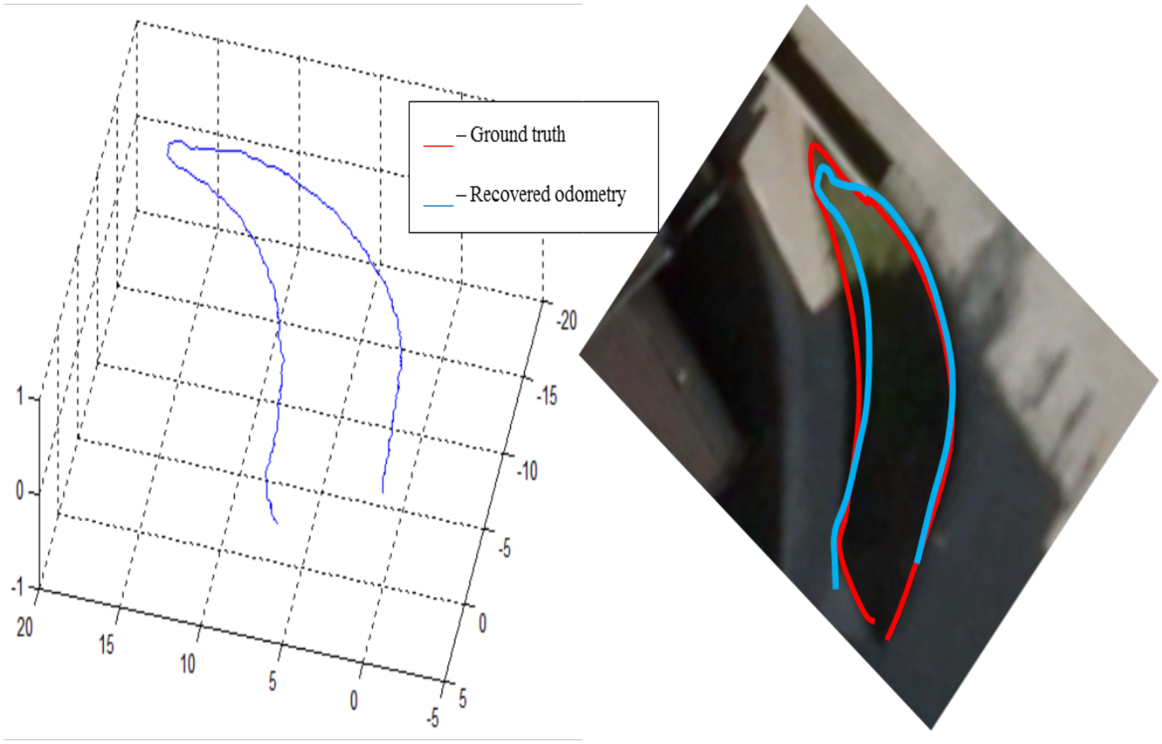

Figure 6.3. Estimated odometry from a walk along the contour of a grass strip.

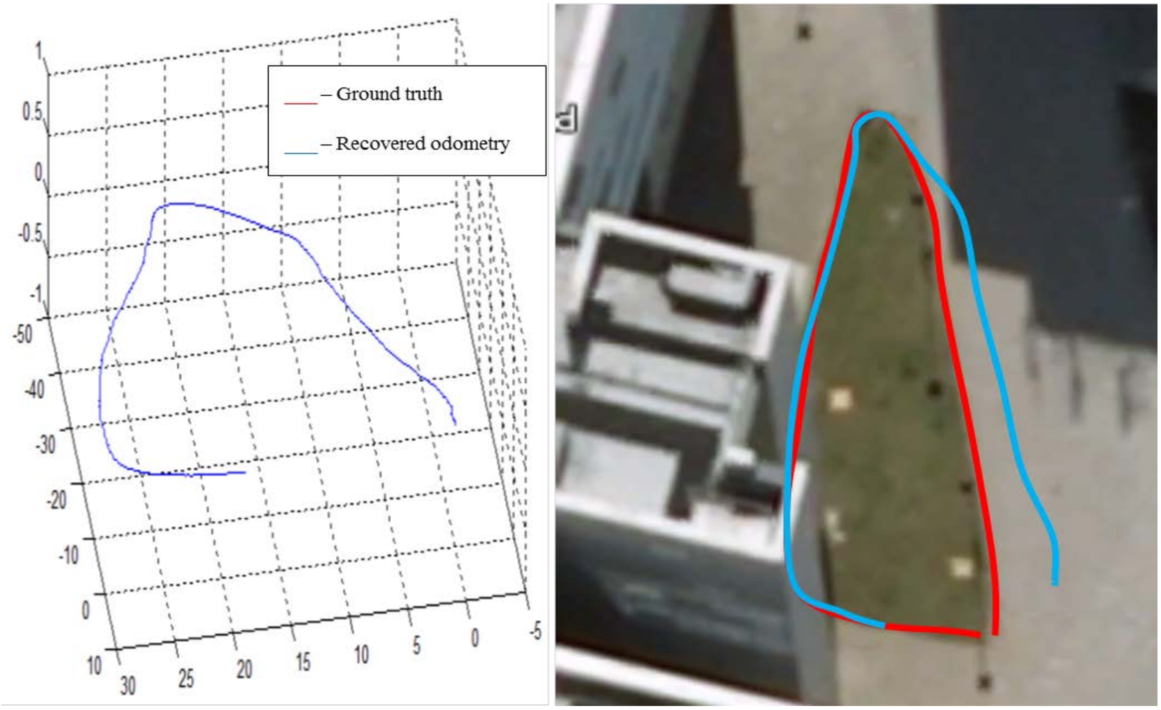

Figure 6.4. Estimated odometry from along the perimeter of a courtyard.

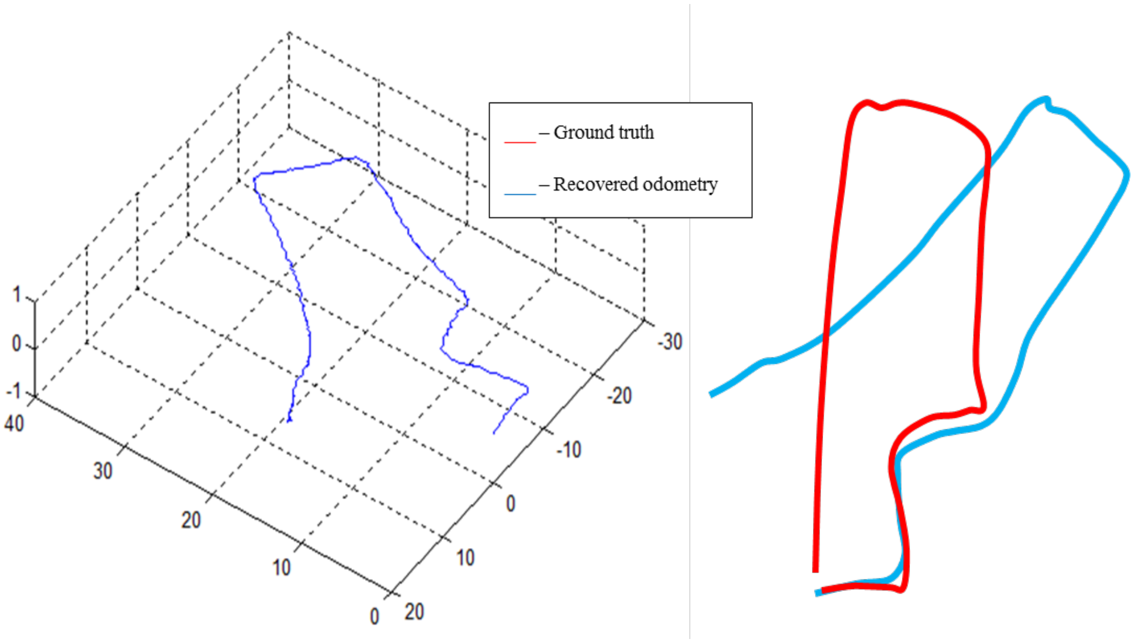

Figure 6.5. Estimated odometry from a walk along a path in the university park (satellite photo not available).

6.3 Inertial measurement units

Inertial measurement units (IMUs) are sensors designed to provide acceleration and angular velocity readings along and about the axes of the associated coordinate frame. Typical commercial IMUs provide 6 DOF input, corresponding to 3 angular velocities and 3 accelerations. Other types of IMU sensors may include 3 additional readings corresponding to the direction of gravity (9 DOF in total). For the experiments reported in this thesis, the Goodrich SiIMU 02[15] was used (Figure 6.7). Nominal sampling rates for SiIMU 02 range from 100 to 300 Hz. The IMU coordinate frame does not represent a right handed coordinate system. Regardless, a transformation M must be applied to the samples in order to obtain the respective angular velocity and acceleration vectors in the camera right-handed frame shown on the right. The transformation depends on the orientation of the IMU with respect to the camera.

[15] Courtesy of UTC Aerospace Systems.

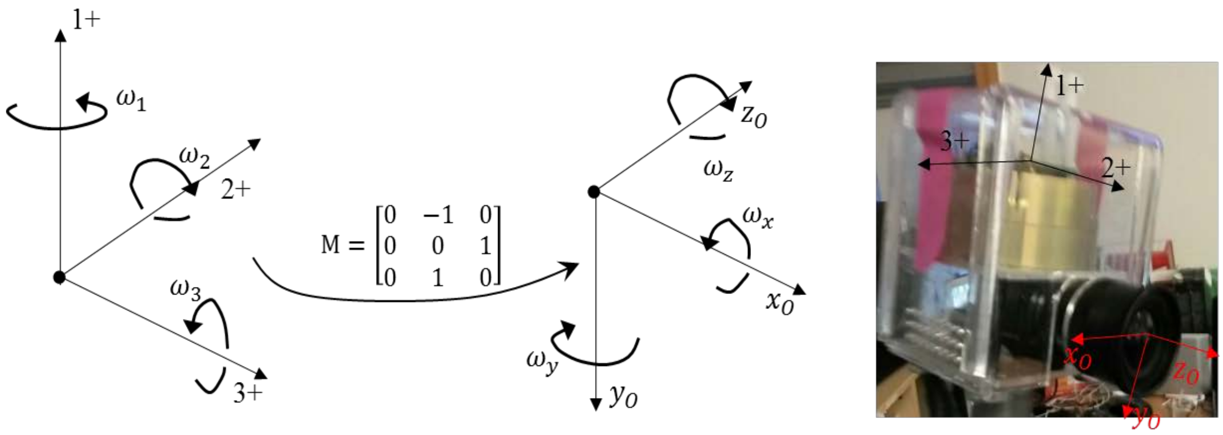

Figure 6.7. The SiIMU02 coordinate frame is shown with the 3 axes labeled $1+$, $2+$ and $3+$ and a picture of the IMU mounted on the camera.

Suppose that ω_k and a_k are the angular velocity and acceleration vectors[16] reported by the IMU in the camera frame at time k and T is the sampling period. Then, the quaternion Δq_k representing the change in IMU orientation at time k is,

$$\Delta q_k = \left(cos\left(\frac{\|\omega_k\|T}{2}\right), sin\left(\frac{\|\omega_k\|T}{2}\right)\frac{\omega_k}{\|\omega_k\|}\right) \tag{6.16}$$

The new orientation quaternion q_{k+1} can be computed from q_k using Δq_k as follows:

$$q_{k+1} = \Delta q_k \otimes q_k \tag{6.17}$$

where $\otimes$ denotes quaternion multiplication.

In theory, it is possible to recover the translation of the camera using the acceleration readings. Suppose that the gravity vector g and the initial linear velocity v_0 of the camera are known in terms of some world coordinate frame. Then, the linear velocity vector v_{k+1} at time $k+1$ can be estimated from v_k and q_k using a_k and ω_k as follows:

$$v_{k+1} = v_k + \left[(\Delta R_k R_k)^T \left(a_k - \underbrace{\omega_k \times (R_k v_k)}_{\substack{Centrifugal \\ acceleration}} \right) - g \right] T \tag{6.18}$$

where ΔR_k and R_k are the rotation matrices corresponding to Δq_k and q_k. The cross product $\omega_k \times (R_k v_k)$ is the centrifugal acceleration caused by the rotational motion of the

[16] Indexing symbol k is used to distinguish IMU sampling time instances from frame capture time instances.

IMU and therefore must be subtracted from the readings in order to obtain pure linear acceleration.

Assuming that also the initial position s_0 of the camera is known, then the position vector s_{k+1} can be computed recursively from s_k and v_k:

$$s_{k+1} = s_k + v_k T \tag{6.19}$$

6.3.1 IMU-camera calibration

Even if the IMU is manually aligned to the camera frame, there is always the need for recovering the misalignment in the form of a rigid transformation. It can be inferred from equations (6.16-18) that the error in pose estimates from inertial input is additive and therefore even the slightest alignment error may affect the estimates in the long run. The most widely used method for IMU-camera calibration was proposed by Mizraei and Roumeliotis (Mirzaei and Roumeliotis 2008) and it essentially involves a non-linear optimization process over the IMU estimates against known camera pose using the famous chessboard pattern. Although Mizraei and Roumeliotis proposed an iterative KF, the optimization problem can also be cast as an offline non-linear least squares problem.

In this thesis, only angular rates are utilized in the applications involving inertial input and therefore only relative orientation between camera and IMU is of interest. Suppose M is the unknown rotation that contains the IMU frame column-wise in terms of the camera frame and ω_I is the angular velocity vector in the IMU frame. It follows that the angular velocity vector in the camera frame will be:

$$\omega_C = M\omega_I \tag{6.21}$$

Let now $\theta_C = \omega_C \Delta t$ and $\theta_I = \omega_I \Delta t$ be the axis-angle vectors corresponding to ω_C and ω_I. The rotation matrix R_C corresponding to the change in camera orientation due to ω_C is given by Rodrigues' formula in equation (4.1):

$$R_C = I_3 + \frac{sin\|M\theta_I\|}{\|M\theta_I\|}[M\theta_I]_\times + \frac{1 - cos\|M\theta_I\|}{\|M\theta_I\|^2}\big((M\theta_I)(M\theta_I)^T - I_3\big) \tag{6.22}$$

Making use of the identity $[M\theta_I]_\times = M[\theta_I]_\times M^T$ (proved in Lemma D.3 of Appendix D) and by observing that $\|M\theta_I\| = \|\theta_I\|$ owed to the fact that M is orthonormal, equation (6.22) becomes:

$$R_C = I_3 + \frac{sin\|M\theta_I\|}{\|M\theta_I\|} M[\theta_I]_\times M^T + \frac{1 - cos\|M\theta_I\|}{\|M\theta_I\|^2}\left(M\theta_I {\theta_I}^T M^T - I_3\right)$$

$$\Leftrightarrow R_C = MM^T + \frac{sin\|\theta_I\|}{\|\theta_I\|} M[\theta_I]_\times M^T + \frac{1 - cos\|\theta_I\|}{\|\theta_I\|^2}\left(M\theta_I {\theta_I}^T M^T - MM^T\right)$$

$$\Leftrightarrow R_C = M \underbrace{\left(I_3 + \frac{sin\|\theta_I\|}{\|\theta_I\|}[\theta_I]_\times + \frac{1 - cos\|\theta_I\|}{\|\theta_I\|^2}\left(\theta_I {\theta_I}^T - I_3\right)\right)}_{R_I} M^T = MR_I M^T \qquad (6.23)$$

where R_I is the rotation matrix corresponding to the orientation change of the IMU frame. It follows, that relationship in (6.23) holds for a sequence of successive IMU rotations $R_I = R_{I,1}, R_{I,2}, \dots, R_{I,n}$:

$$R_C = \left(MR_{I,1}M^T\right)\left(MR_{I,2}M^T\right) \dots = MR_{I,1}(M^T M)R_{I,2} \dots = M \underbrace{\left(R_{I,1}R_{I,2} \dots R_{I,n}\right)}_{R_I} M^T \qquad (6.24)$$

The relationship in (6.23) and (6.24) is a quadratic equation in the matrix M and can be solved, given measurements for R_C; what is very interesting, is that these measurements cannot only be obtained using a chessboard pattern but also with an algorithm for relative pose. Thus, in theory, it is possible to calibrate relative orientation without any aids to provide ground-truth. It should be noted that for minor misalignments, the uncalibrated product $MR_I M^T$ can be an acceptable estimate of the camera orientation.

6.4 Relative pose odometry in 3D using inertial measurements

Resolving two-view geometry in 3D is strictly a non-linear problem and no relaxations such as the one explained in section 2.1 are possible without at least first involving prior knowledge on the pose vector or making assumptions about camera motion (for example, purely translational motion). Unfortunately, the measurement model involves not only projection, which is a non-linear operation by definition, but also a rotation matrix in both the numerator and denominator. If the effects of the camera rotation can somehow be undone as explained in Chapter 3, section 3.6, then it can be shown (section 4.1) that the baseline can be estimated up to arbitrary scale from an overdetermined linear homogenous system. An alternative approach to recovering relative pose between two camera views involves the extraction of the rigid transformation from the essential matrix as explained in

Chapter 3. This approach however, except for the 5-point case (Nister's algorithm), entails a certain risk associated with the relaxation of the required orthogonality constraints (see Chapter 3, section 2.6 for details) in the 8-point algorithm.

6.4.1 Recovering baseline with known relative orientation

Angular velocity readings from IMUs are typically robust and generally accumulate very little drift for periods of several minutes or even more. Thus, inertial measurements can provide very reliable short-term orientation estimates which can be used to rectify the feature locations in two views (as discussed in Chapter 3, section 3.6). This approach has been employed in several cases of visual SLAM applications involving vehicles equipped with IMUs (Kneip, Chli et al. 2011, Achtelik, Lynen et al. 2012, Weiss, Achtelik et al. 2012).

Consider the case in which relative orientation is known a priori by means of an inertial measurement unit. For brevity of notation, the rectified projection of the *i*th 3D point in the latest camera view is denoted with $m_{t+1}^{(i)}$ and the projection on the reference view with $m_t^{(i)}$. Then according to equation (6.1), m_{t+1} and m_t are related through the following equation:

$$m_{t+1}^{(i)} = \frac{1}{1_z^T\left(d_t^{(i)} m_t^{(i)} - b_t\right)}\left(d_t^{(i)} m_t^{(i)} - b_t\right) \tag{6.25}$$

where $1_z^T = [0 \quad 0 \quad 1]$, $d_t^{(i)}$ is the feature depth in terms of the camera frame at time *t* and b_t is the baseline vector linking camera position at times *t* and *t+1* expressed also in the camera frame at time *t*. The relationship of equation (6.25) can be rearranged in to the following:

$$1_z^T\left(d_t^{(i)} m_t^{(i)} - b_t\right) m_{t+1}^{(i)} = d_t^{(i)} m_t^{(i)} - b_t \tag{6.26}$$

And, provided that the inner-to-outer-product property $(a^T b)c = (aa^T)b$ holds for any three vectors $a, b, c \in \mathbb{R}^3$, equation (6.26) can be re-written as follows:

$$m_{t+1}^{(i)} 1_z^T\left(d_t^{(i)} m_t^{(i)} - b_t\right) = d_t^{(i)} m_{t+1}^{(i)} - b_t$$

$$\Leftrightarrow (m_{t+1} 1_z^T - I_3)\left(d_t^{(i)} m_t - b_t\right) = 0_3 \tag{6.27}$$

where I_3 is the 3×3 identity matrix and 0_3 is the 3×1 zero vector.

Using the relationship in (6.27) we can construct a quadratic cost function over the depth and the baseline as follows:

$$J = \sum_i \left(d_t^{(i)} m_t^{(i)} - b_t\right)^T C_i \left(d_t^{(i)} m_t^{(i)} - b_t\right) \tag{6.28}$$

where C_i is an i^{th} datum-dependent matrix:

$$C_i = \left(m_{t+1}^{(i)} 1_z^T - I_3\right)^T \left(m_{t+1}^{(i)} 1_z^T - I_3\right) \tag{6.29}$$

Taking the derivative of J in terms of $d_t^{(i)}$ yields the following expression for $d_t^{(i)}$:

$$d_t^{(i)} = \frac{\left(m_t^{(i)}\right)^T C_i b}{\left(m_t^{(i)}\right)^T C_i m_t^{(i)}} \tag{6.30}$$

This is exactly the solution given in equation (3.31) for $R = I_3$. Plugging back into (6.28), eliminates feature depth from the cost function leaving us with an ordinary least squares problem on the components of b_t:

$$J = b_t^T \underbrace{\left[\sum_i \left(\frac{m_t^{(i)}\left(m_t^{(i)}\right)^T C_i}{\left(m_t^{(i)}\right)^T C_i m_t^{(i)}} - I_3\right)^T C_i \left(\frac{m_t^{(i)}\left(m_t^{(i)}\right)^T C_i}{\left(m_t^{(i)}\right)^T C_i m_t^{(i)}} - I_3\right)\right]}_{Q} b_t = b_t^T Q b_t \tag{6.31}$$

Ideally (i.e., no noise in the data), the solution for b_t should be the one-dimensional null space of Q. However, in the majority of cases, Q will be a full rank positive semidefinite matrix and the solution is obtained by optimizing equation (6.26) with a hard unit-norm constraint on b_t in order to avoid the trivial solution. In this case, the minimizer is the eigenvector corresponding to the smallest eigenvalue of Q. In the unlikely case in which the null space of Q has dimension greater than 1, then for all intents and purposes, the null vector which yields the smallest value of J is chosen. Please note here that the form of data matrix Q in equation (6.31) is not numerically stable (the term in the denominator vanishes for small disparity values) and should be reformulated to avoid divisions by small numbers.

6.4.2 3D relative pose odometry estimates with inertial input

The majority of test sequences were taken from a moving car or a boat in natural environments such as parks and rivers. Vehicle velocity did not exceed 7 knots per hour (3.601 m/s) either on land or sea. Scene background depth ranges from 2 to 500 meters. Although vehicle velocity was generally constant, yet slight variations in speed where unavoidable; subsequently, scale awareness drifts in the global pose estimate, mainly towards the end of the trajectories. For comparison, the odometry estimate was squashed on the *x*-*z* plane and thereafter rotated and the ground truth was scaled by dividing all baseline vectors by the GPS speed at the origin (Figures 6.8-10). The relative pose model has no memory and therefore velocity must be assumed constant. Thus, the baseline error in relative pose estimates essentially corresponds to the actual variations in the speed of the vehicle and was estimated on average at 2.43%. This error was calculated from the standardized ground truth pairwise baseline lengths in terms of the GPS based speed of the vehicle at the origin.

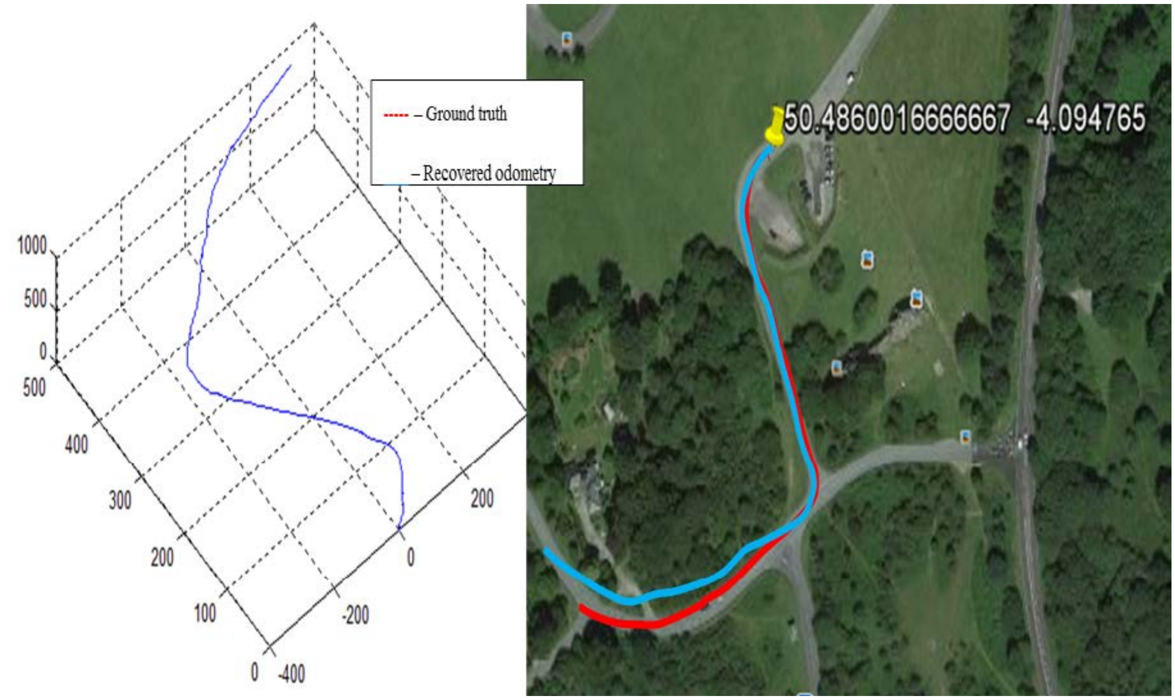

Figure 6.8. Relative pose odometry from a park ride (approximately 0.5 km) on a car. Plot scale on the left is arbitrary and therefore axes do not represent actual length units).

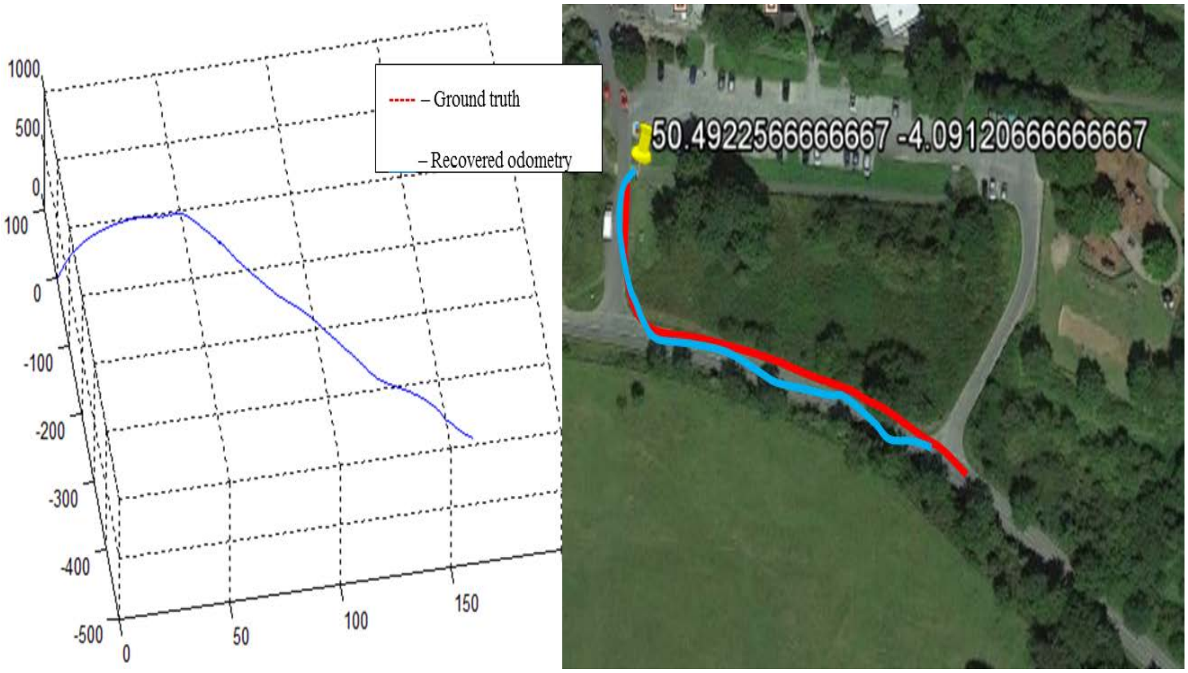

Figure 6.9. Relative pose odometry from a ride in countryside residential areas (0.6 km). Plot scale on the left is arbitrary and therefore axes do not represent actual length units.

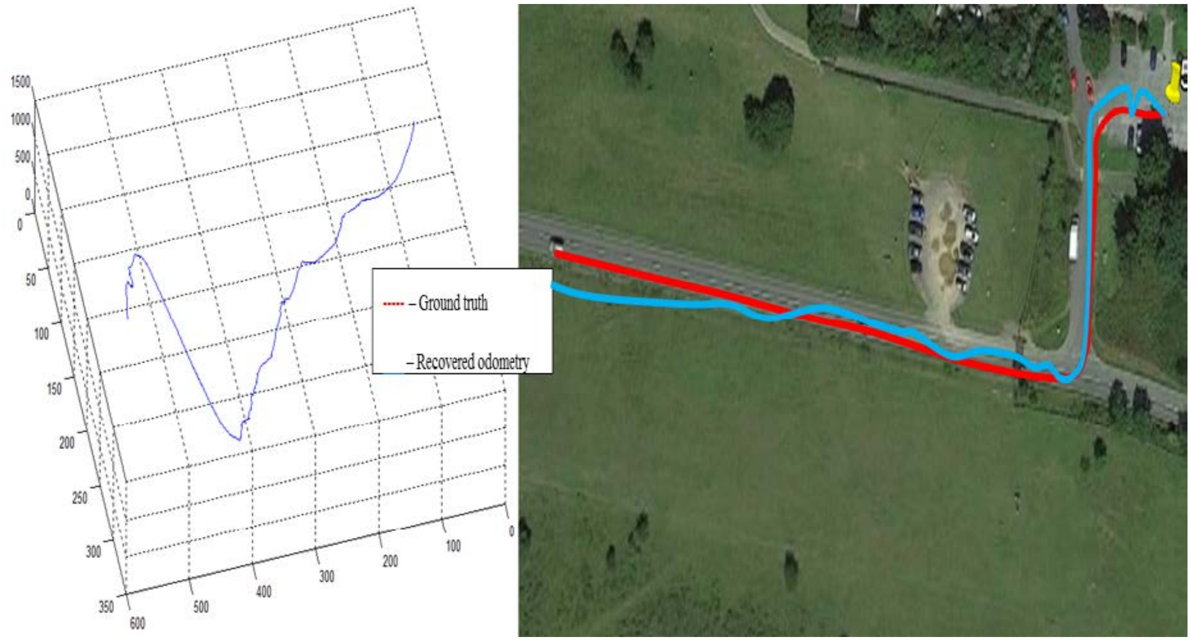

Figure 6.10. Relative pose odometry from a ride in countryside residential areas (1.1 km). Plot scale on the left is arbitrary and therefore axes do not represent actual length units.

6.5 Summary

This chapter introduced some preliminary work and techniques that could be useful either as parts of more elaborate algorithms, or as quick solutions for short-term odometry estimates. The orthogonal Procrustes method was discussed and employed in the context of planar odometry. Although not an obvious fact, the Procrustean method is ubiquitous in computer vision algorithms, as many of the problems concerning projections of the 3D world onto the camera are inherently not linear because of the division by depth and the direct/indirect orthogonality constraints; hence, linear relaxations often require the enforcement of the constraints after the optimization. Furthermore, methodologies for 3D odometry with gyroscope input were examined with results that appear encouraging in relatively short distances.

The concept of relative pose odometry relies on the independent tracking of features in pairs of consecutive frames, thereby recovering the relative camera pose. Thus, current pose is obtained from the previous one by applying the estimated transformation. The features are treated as if they were actually detected in the first frame of the current pair and therefore tracking uncertainty does not account for the past. This is a very reasonable assumption in natural scenes with grass, dense foliage, etc., because the rich texture of the background is likely to yield corner-like neighborhoods, even if they are not high-quality (in terms of the Harris criterion). One important aspect of relative pose odometry is that the map does not appear in the filtration and therefore, theoretically, an arbitrary number of features can be used in the measurement step without having a cubic-scaling impact on execution time and state vector.

For a ground facing camera performing planar motion it is possible to obtain a linear equation in the elements of the rotation matrix and the baseline; the solutions of this ordinary LS optimization can be refined further throughout Gauss-Newton iteration. In the case of general camera motion, prior relative orientation estimates are necessary in order to obtain a linear homogenous equation on the baseline for each correspondence by means of rectification. The scaled baseline estimate obtained can thereafter be used in subsequent iterative optimization. In either case, planar or general, the relative pose solution does not reflect the true scale of the scene and if there exists no input regarding velocity or a motion model, then the recovered odometry is likely to present local scale discrepancies in comparison to ground truth. Results indicate that purely vision based odometry has significant issues with orientation as drift becomes apparent after 10-15 meters in the case of the ground facing camera. On the other hand, provided gyroscope angular rates, the baseline is fairly accurate and the recovered odometry presents an error that almost exclusively depends on the inaccuracies of the motion model and almost not at all on inherent drift even for distances up to 1.5 km. For instance, in the simple case of constant speed, such motion model discrepancies become apparent, in Figures 6.8-10 when the vehicle has clearly reduced speed in order to go around a sharp turn.

Chapter 7

The GraphSLAM approach to least squares and sensor fusion

At a fundamental level, SLAM is always cast as a (potentially nonlinear) least squares optimization problem. What varies depending on the specifics of the application, is the algorithm that carries-out this optimization. The Kalman filter is a prominent representative of these algorithms; it improves the posterior in a step-by-step fashion by gradually integrating new measurement information. This approach usually reflects a discrete-time sequence of events in which new measurements concern only the current state and therefore there is no need to access past variables. In practice however, there are cases in which the measurement model may depend on past states/variables. Such measurement models include global positioning sensors, or generally any type of sensor model that involves sampling at much slower rates than the ones that the process itself progresses through time. An example of such am application is visual SLAM with gyroscopic input. Typically, angular rates are sampled in the range of 100-300 Hz, whereas a standard camera frame rate is 24-30 Hz; this implies that more than one angular samples are obtained between two frame captures. Although a KF can still deal with the different sampling rates, it however requires the design of a special transition model and an unnecessarily large state vector. A much more elegant solution would be to use an alternative representation of the posterior which would allow for a variable number of poses in the joint. The representation of Gaussian distributions employed by information filters can provide this flexibility.

Information filters utilize an alternative representation of the SLAM posterior using the information matrix and information vector, as opposed to the traditional moment parametrization of normal distributions. The information matrix and information vector are known as the *canonical parameters*, or *Fisher parameters*; there exists a *duality* between the canonical and moment parameters and so is between information filters and Kalman filters. Modern SLAM researchers find significant advantages in IFs, associated with the ability to block-diagonalize the information matrix (Wang 2011). Furthermore, the canonical parameters correspond directly to the *normal equations* obtained from the sum of quadratic constraints associated with motion and observations. In other words, the canonical parametrization not only is the formulation of the Gauss-Newton method for the variables in the posterior, but also allows for the addition or removal of such variables by expanding the information matrix and vector or marginalizing-out existing ones (thereby reducing the size of the canonical parameters). Figure 7.1 illustrates this concept of information matrix expansion and reduction in the context of SLAM.

7.1 Filtering with a state vector of variable size

By definition, the normal equations of a least squares system can be regarded as the Fisher parametrization of the solution variable. Based on this fact, Thrun and Montemerlo proposed GraphSLAM (Thrun and Montemerlo 2006), a technique that incorporates new information or marginalizes existing directly into the Fisher parameters. In other words, GraphSLAM is a general technique for expanding or downsizing and solving a least squares system in parts or in whole. From a filtering perspective, one may regard GraphSLAM as an information filter with a state vector of variable size. Thus, the filter keeps track of a matrix Ω and a vector ξ, the dimensions of which may grow with the addition of new variables or be reduced by the marginalization of existing ones. In other words, the state vector does not have a fixed size. Moreover, the quadratic constraints corresponding to factors related to state transitions (motion model) and landmark observations (measurements) are being incorporated directly into Ω and ξ as shown in Figure 7.1.

7.1.1 GraphSLAM rules

Consider a joint Gaussian variable $S \in \mathbb{R}^n$, such that $S = \{X, Y\}$ where $Y \in \mathbb{R}^d$ and $X \in \mathbb{R}^{n-d}$ with a distribution represented by the information matrix Ω and information vector ξ. A quadratic constraint of the form, $\left(m - f(Y)\right)^T \Psi^{-1}\left(m - f(Y)\right)$, where f is a differentiable function at a given point $\hat{y}$, Ψ is a positive semi-definite matrix and m is a constant, can be incorporated into the joint distribution with the following operations:

$$\Omega \leftarrow \Omega + \left(\frac{\partial f}{\partial Y}\bigg|_{\hat{y}} A_Y \right)^T \Psi^{-1} \left(\frac{\partial f}{\partial Y}\bigg|_{\hat{y}} A_Y \right) \tag{7.1}$$

where $\frac{\partial f}{\partial Y}\big|_{\hat{y}}$ is the Jacobian of f evaluated at $\hat{y}$ and A_Y is a matrix such that, $Y = A_Y S$. The respective update to the information vector will be,

$$\xi \leftarrow \xi + \left(\frac{\partial f}{\partial Y}\bigg|_{\hat{y}} A_Y \right)^T \Psi^{-1} \left(m - \left(f(\hat{y}) - \frac{\partial f}{\partial Y}\bigg|_{\hat{y}} \hat{y} \right) \right) \tag{7.2}$$

7.1.2 Making motion predictions and incorporating measurements

Figure 7.1 illustrates the updates to the information matrix and vector imposed by the transition and measurement constraints. In particular, pose x_{t-1} advances to x_t and observation $m_t^{(k)}$ of the k^{th} landmark is obtained. The dashed arrows point to the regions in Ω and ξ updated by the quadratic constraints that correspond to the state transition and landmark observation.

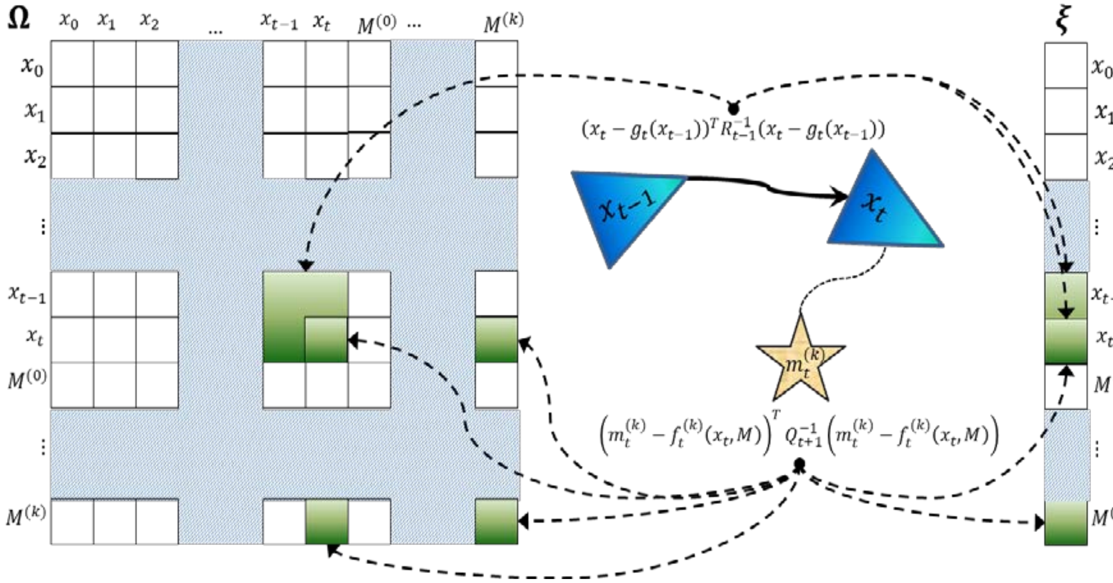
Ω
x_0
x_1
x_2
…
x_{t-1}
x_t
$M^{(0)}$
…
$M^{(k)}$
ξ
$(x_t - g_t(x_{t-1}))^T R_{t-1}^{-1}(x_t - g_t(x_{t-1}))$
x_{t-1}
x_t
$m_t^{(k)}$
$\left(m_t^{(k)} - f_t^{(k)}(x_t, M)\right)^T Q_{t+1}^{-1}\left(m_t^{(k)} - f_t^{(k)}(x_t, M)\right)$

Figure 7.1. Populating the information matrix and information vector during SLAM.

One may consider GraphSLAM as a semi-offline filter, because it allows for the marginalization of arbitrary subsets of the joint. Thus, it is possible to keep a relatively limited number of variables in the joint just enough to accommodate real-time execution, while allowing for “local" iterative optimization. Frequent marginalization of past pose variables and landmarks that are not visible anymore should keep the dimensionality of the information matrix below a certain boundary, thereby accommodating the use of iterative optimization in real time. For convenience, the marginals of the multivariate Gaussian using the Fisher parameters are given in Table 7.1; the information matrix and vector of the remaining variables (X) appear with a bar ($\bar{\Omega}_{XX}$ and $\bar{\xi}_X$).

	Information matrix	Information vector
Joint distribution	$\Omega = \begin{bmatrix} \Omega_{XX} & \Omega_{XY} \\ \Omega_{YX} & \Omega_{YY} \end{bmatrix}$	$\xi = \begin{bmatrix} \xi_X \\ \xi_Y \end{bmatrix}$
Marginal distribution of X	$\bar{\Omega}_{XX} = \Omega_{XX} - \Omega_{XY}\Omega_{YY}^{-1}\Omega_{YX}$	$\bar{\xi}_X = \xi_X - \Omega_{XY}\Omega_{YY}^{-1}\xi_Y$

Table 7.1. Marginals of the multivariate Gaussian using the canonical parametrization.

7.2 Fusing 3D gyroscopic data with 2D global positioning measurements

The trajectory of vehicles on the ground or the surface of the sea, although theoretically 3D, essentially takes place on an approximately planar surface. When localizing a boat or a car using a “slow” (i.e., sampling rates of 5 Hz or lower) position sensor, the feedback is sufficient to accurately position the vehicle on the map when the trajectory has grown significantly in scale. However, localized information about the vehicle’s pose entails a great deal of uncertainty, not only because of the very slow sampling rate of the position sensor, but also because the respective measurements are 2D, while the vehicle’s trajectory occasionally deviates from a strictly planar motion pattern (for instance, due to rough sea or wave patterns generated by other vessels). It is therefore difficult to draw conclusions about the vehicle’s short-term behavior based only on position feedback; such behavior may be indicative of critical situations such as a boat being overturned. In this regard, IMUs offer accurate high frequency angular velocity samples about all three axes of motion and therefore can provide efficient attitude estimates with little drift over time.

7.2.1 Scenario of a surface vehicle with disrupted 2D position feedback

Consider the scenario of a surface vehicle equipped with an IMU and a 2D global position sensor with intermittent signal reception. The IMU *z*-axis is pointed outwards from the side and the *x*-axis is pointing to the front of the vessel as shown in Figure 7.2 (subsequently, acceleration in the *x*-axis is always positive). Suppose that position measurements arrive intermittently in time intervals which may be several seconds long. During this time, the vehicle is predicting its pose based on IMU angular rates and possibly, on some motion model. If the distance covered by the vehicle during this "dead reckoning" interval is relatively large, then the next position measurement can bring a significant improvement in the odometry estimate throughout this period (i.e., the sequence of poses since the last position measurement). Figure 7.2 illustrates a sequence of $n+1$ poses of a vehicle that travels approximately in a plane between two successive position feedback reception events (measurements denoted as θ_t and θ_{t+n}).

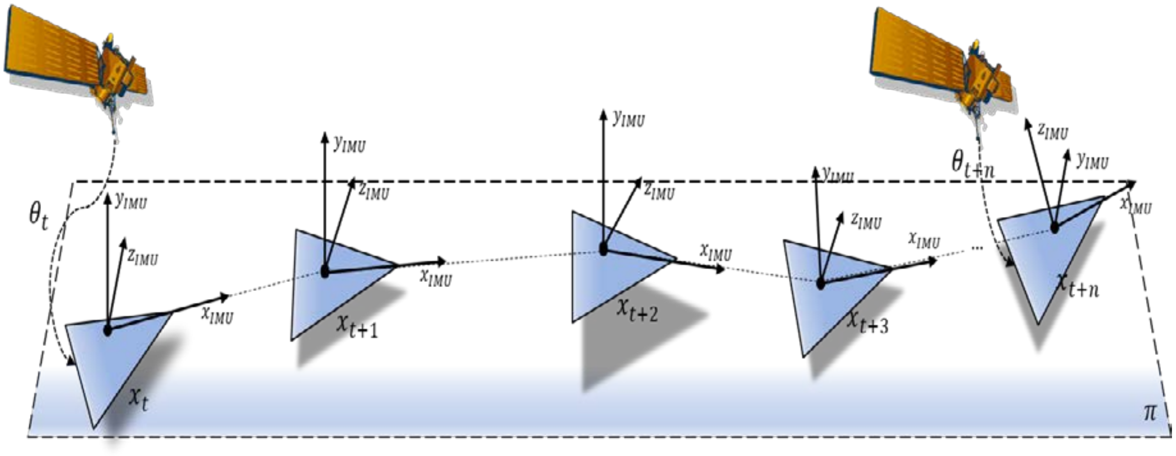

Figure 7.2. Illustration of a vehicle navigating approximately on a planar surface. Position feedback is presumably obtained from a satellite.

7.2.2 Using GraphSLAM for position updates over multiple vehicle poses

Since positioning feedback occurs sparsely while angular readings arrive at very high rates, marginalization of past states would only make sense when a new measurement arrives. In the meanwhile however, the vehicle has traveled only on gyroscopic input and therefore uncertainty in the estimated odometry increases through time. The GraphSLAM approach to filtering can ideally accommodate this progression by expanding the Fisher parameters and adding more pose variables, until the next position measurement is obtained.

Once the position information becomes available, then using the measurement models described in the next sections, the entries of the information matrix (and vector) associated with the vessel's odometry since the last measurement interception are update and a Gauss-Newton iterative method is employed to further refine the odometry estimate.

7.2.3 Fusion of high frequency 3D attitude input with low frequency 2D position measurements: Motivation

It is very reasonable for a vehicle such as a boat to deviate from a strictly planar trajectory for a variety of reasons which may be associated with tidal waves, water drag, wind, etc. Although these deviations have varying magnitude, nevertheless they are transient in nature and therefore, on average, one can assume that the vehicle is moving on a planar surface. In most cases, global positioning feedback is provided in terms of 2D coordinates on this plane.

There have been many algorithms proposed in literature for the fusion of IMU and 2D position feedback, almost exclusively based on KF formulations (Zhang, Gu et al. 2005, Caron, Duflos et al. 2006, Wendel, Meister et al. 2006). To the best of my knowledge, none of these applications particularly addresses the problem of different sampling rates and dimensionality of the measurements. The gyroscope on one hand provides a high frequency 3D input signal which carries significant information regarding short-term vehicle motion; on the other hand, global position sensors are typically 2D, slow and cannot capture transient motion.

The idea in the approach introduced here is to use the gyroscope and the vehicle's motion model (if available) to integrate odometry estimates until a position measurement is obtained. The estimates are used for short-term vehicle navigation, but with GraphSLAM, they are not marginalized out of the filter's posterior as in the case of the KF. Eventually, when a new position measurement arrives, new information does not only impact the most recent pose, but all poses that are currently active (i.e., not marginalized out) in the information matrix-vector. In other words, the new position measurement updates the entire odometry, without however losing the high frequency information contained in it. This is a "retrospective" rectification approach which can be very useful for planners that require a detailed view of the vehicle's past states in order to produce a short-term plan.

7.2.4 A linear measurement model

Let $x_t = [\eta_t^T \quad s_t^T]^T$ be the pose of the vehicle, where $\eta_t, s_t \in \mathbb{R}^3$ are the orientation parameter and position vectors. One way of incorporating the information from the position measurement into the posterior is to simply update the Fisher parameters with a linear constraint between the last pose and the position reported by the measurement:

$$(\theta_t - I_{2\times3}s_t)^T Q^{-1}(\theta_t - I_{2\times3}s_t) \tag{7.3}$$

where $\theta_t \in \mathbb{R}^2$ is the position measurement, $Q \in \mathbb{R}^{2\times2}$ is the measurement covariance and $I_{2\times3}$ is the 2×3 identity matrix.

The measurement model of equation (7.3) is simple and, although seemingly concerns only the x-y coordinates of the last position vector, the existing correlations between poses in the information matrix will distribute measurement information to multiple past pose variables. What is important about the measurement model of (7.3) is that it is linear and therefore solving for the mean of the posterior as $\mu = \Omega^{-1}\xi$ should yield updated estimates for all active poses without the need to run iterative optimization.

7.2.5 A measurement model based on a fitted motion plane

Consider the sequence of consecutive poses $x_{t:t+n}$ between two successive position measurements at times t and $t+n$. Define vectors b_i such that,

$$b_i = s_{t+i} - \hat{s}, \qquad i \in \{0, \dots, n\} \tag{7.4}$$

where $s_t \in \mathbb{R}^3$ is the position of the vehicle at time t and $\hat{s} = \frac{1}{n+1}\sum_{i=0}^{n} s_{t+i}$ is the mean of s_{t+i}. Snce, excluding transients, the vehicle is approximately moving on a planar surface, then, the vectors b_i define a 3D ellipsoid which can be approximated by a 2D plane. This plane is dubbed *motion plane* (Figure 7.4). Since the motion plane estimate is a best-fit plane (in the least squares sense) to a sequence of vehicle positions, it follows that any basis pair of vectors v_1 and v_2 (shown in red in Figure 7.4) can be obtained as the first two principle directions of the ellipsoid defined by b_i; the third principle direction should account for transient movements off the plane (for instance, briefly heading up while riding a wave). To obtain the closest plane to the aforementioned ellipsoid, one simply needs to consider the SVD of the following matrix:

$$B = \begin{bmatrix} b_0^T \\ \vdots \\ b_n^T \end{bmatrix} \tag{7.5}$$

A basis of the motion plane can be obtained from the first two columns v_1 and v_2 of matrix V in the SVD, $B = UDV^T$ (corresponding to the two largest singular values of B).

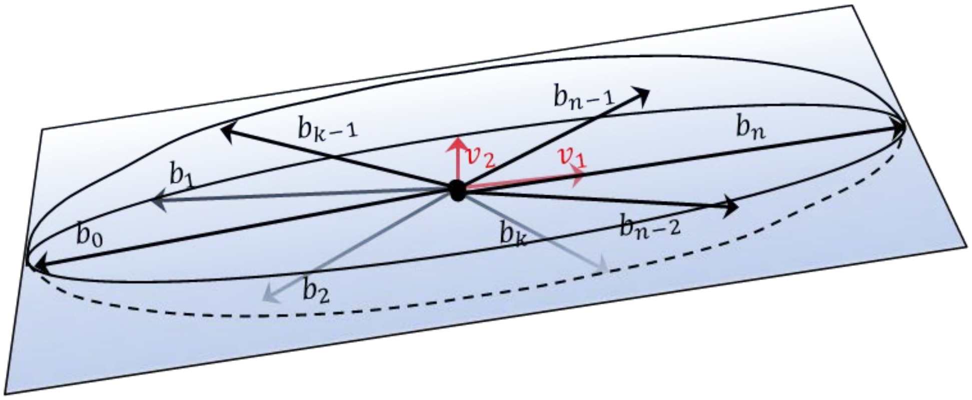

Figure 7.4. Illustration of the motion plane as an approximation of the ellipsoidal 3D subspace defined by $b_0, b_1, b_2, \dots, b_{k-1}, b_k, \dots, b_{n-1}, b_n$.

Consider the matrix A such that,

$$A = [v_1 \quad v_2] = VI_{3\times2} \tag{7.6}$$

where $I_{3\times2}$ is the 3×2 identity matrix. As a direct consequence of orthonormality, $A^TA = I_2$ and therefore the projection operator P onto the motion plane is,

$$P = (A^TA)^{-1}A^T = I_{2\times3}V^T \tag{7.7}$$

Using P, it is now possible to derive a relationship between θ_{t+n} and the vehicle position estimate s_{t+n} on the motion plane:

$$P\theta_{t+n} = Ps_{t+n} \Leftrightarrow \underbrace{P(\theta_{t+n} - s_{t+n})}_{f_{t,n}(s_{t:t+n})} = 0 \tag{7.8}$$

Since P is a projector obtained from the SVD of B, it follows that it is a function of all the position vectors from time t to time $t + n$. Thus, function $f_{t,n}(s_{t:t+n}) = P(\theta_{t+n} - s_{t+n})$ is a nonlinear regularization quadratic constraint.

7.2.6 Derivatives of the motion plane projector

Let $\hat{s}_t, \dots, \hat{s}_{t+n}$ be the current estimates of the means of position vectors $s_t, \dots, s_{t+n}$. In order to apply the Gauss-Newton method and update the information matrix and vector according to equations (7.1) and (7.2) in each step, $f_{t,n}$ should be linearized in the neighborhood of $\hat{s}_{t:t+n}$. Since $P = I_{2\times3}V^T$ and V is generated by the SVD, the derivatives of U and V in terms of the elements of B must be obtained. A very helpful method to compute the Jacobian of the SVD has been introduced by Papadopoulos and Lourakis (Papadopoulo and Lourakis 2000) and will be summarized briefly here (see Appendix B for more details on the results that follow).

The derivatives of $U = [u_{ij}]$ and $V = [v_{ij}]$ with respect to B=$[b_{ij}]$ are,

$$\frac{\partial U}{\partial b_{ij}} = U\Omega_U^{ij} \ , \ \frac{\partial V}{\partial b_{ij}} = -V\Omega_V^{ij} \tag{7.9}$$

The matrices $\Omega_U^{ij} = \left[\left(\omega_U^{ij}\right)_{kl}\right]$ and $\Omega_V^{ij} = \left[\left(\omega_V^{ij}\right)_{kl}\right]$ are antisymmetric and therefore have zeros in the diagonal, while their non-diagonal elements verify the following linear system:

$$\begin{cases} d_l\left(\omega_U^{ij}\right)_{kl} + d_k\left(\omega_V^{ij}\right)_{kl} = u_{ik}v_{jl} \\ d_k\left(\omega_U^{ij}\right)_{kl} + d_l\left(\omega_V^{ij}\right)_{kl} = -u_{il}v_{jk} \end{cases} \tag{7.10}$$

where d_k are the singular values of B. Thus, it is possible to compute the derivatives of U and V per element of B by simply solving the system of equation (7.10) and thereafter substituting in equation (7.9).

Unfortunately, the complexity of the method is $O((MN)^4)$ where M, N are the numbers of rows and columns of B. Although in this case the dimension of the row space of B is 3 (i.e., $N = 3$), the complexity of derivation is still $O(M^4)$ constituting the computation of the derivatives of V practically impossible at runtime for $M > 3$. However, one may follow a significantly less cumbersome path by considering the SVD of the 3×3 Gram matrix $C = B^TB$, such that $C = VD^2V^T$. The derivatives of V with respect to the elements of C can be calculated in 9×9 steps in which only the elements of an antisymmetric matrix $W^{ij} = \left[\left(w^{ij}\right)_{kl}\right]$ are required[17] and can be obtained by collapsing the system of equation (7.10) into a single equation for $\left(w^{ij}\right)_{kl}$ that yields the following solution:

$$\left(w^{ij}\right)_{kl} = \begin{cases} \dfrac{v_{ik}v_{jl}}{d_l^2 - d_k^2} & , \quad k \neq l \\ 0 & , \quad k = l \end{cases} \tag{7.11}$$

[17] In the case of the SVD of the Gram matrix, it is easy to see that $U = V$ and therefore, if $W^{ij} = \Omega_U^{ij}$, then the fact that $(W^{ij})^T = \Omega_V^{ij}$ follows from the definition of Ω_U^{ij} and $\Omega_V^{ij}$by Papadopoulos and Lourakis. Thus, the antisymmetric matrix W^{ij} suffices for the computation of the Jacobian of V.

And the derivatives $\frac{\partial V}{\partial c_{ij}}$ of *V* with respect to the elements of *C* can now be computed from equation (7.9) as follows:

$$\frac{\partial V}{\partial c_{ij}} = VW^{ij} \tag{7.12}$$

Having $\frac{\partial V}{\partial c_{ij}}$, the sought derivatives $\frac{\partial V}{\partial b_{ij}}$ of *V* with respect to the elements of *B* can be obtained using the chain rule:

$$\frac{\partial V}{\partial b_{ij}} = \sum_{k=1}^{3}\sum_{l=1}^{3}\frac{\partial V}{\partial c_{kl}}\frac{\partial c_{kl}}{\partial b_{ij}} \tag{7.13}$$

Where the derivative $\frac{\partial c_{kl}}{\partial b_{ij}}$ of the elements of *C* with respect to the elements of *B* is obtained as follows:

$$\frac{\partial c_{kl}}{\partial b_{ij}} = \begin{cases} 0 \quad , & k \neq j \; and \; l \neq j \\ b_{il} \quad , & l \neq i \; and \; k = j \\ b_{ik} \quad , & k \neq i \; and \; l = j \\ 2b_{ij} \quad , & (k,l) = (i,j) \end{cases} \tag{7.14}$$

From equations (7.11-14), it follows that the Jacobian of *V* with respect to *B* can be computed in time $T(9 \times 9 + 9 \times M) = O(M)$, which is linear in *M*. The computation of the Gramm matrix does not affect the linearity of the overall complexity, since it is also linear in *M*, given that *B* has 3 columns.

7.2.7 Learning GPS priors on relative position

Position estimates from GPS sensors encapsulate uncertainty originating in numerous sources such as the number of visible satellites and their positions, signal strength, interference, etc. Although modelling each and every stochastic parameter is impractical, it is possible however to use maximum likelihood in order to learn the covariance matrix and the bias of a distribution on relative position measurements.

Suppose that for a sufficiently localized application[18], a GPS reading is mapped to a coordinate vector $\theta \in \mathbb{R}^2$ using the Haversine formula (Goodwin 1910). Let $s|\theta \sim N(\theta + c, \Sigma)$ where c is a bias constant associated with the sensor. Also, let $p(\Delta s|s_1, s_2) =$

[18] The scale of the entire trajectory is small enough to regard GPS based relative positions as coplanar vectors.

$\delta\big(\Delta s - (s_2 - s_1)\big)$, where δ is the Dirac delta function; this is an elegant way of casting a hard constraint between Δs and s_1, s_2 as a probability distribution. Figure 7.5 illustrates the stochastic model of the relative position estimate using conditional distributions $s_1|\theta_1$ and $s_2|\theta_2$. The GPS based position prior $p(\theta)$ is assumed to be a zero-mean Gaussian with a diagonal covariance matrix with arbitrarily large non-zero elements; this is also an elegant way of stating that the prior is practically uninformative.

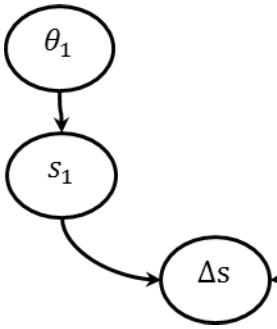

Figure 7.5. A Bayes network illustrating conditional dependencies between relative position Δs and the two conditional variables $s_1|\theta_1$ and $s_2|\theta_2$.

Learning the parameters c and Σ on absolute locations is very impractical since ground truth cannot generally be recovered from maps in resolutions of a few meters. On the other hand however, it is very easy to accurately measure relative distances. In this context, the posterior probability of the relative position estimate Δs given the relative position measurement $\Delta\theta$, is given by the following marginal:

$$p(\Delta s|\Delta\theta = \theta_2 - \theta_1) \propto \int\int p(s_1|\theta_1)p(s_1 + \Delta s|\theta_1 + \Delta\theta)p(\theta_1)d\theta_1 ds_1 \qquad (7.15)$$

where $p(\theta_1)$ can be omitted in practice. Considering that $p(s_1|\theta_1)$, $p(s_2|\theta_2)$ are Gaussians with the same parameters, it is easy to verify that the marginal of equation (7.15) is a normal distribution with mean $\Delta\theta$ and covariance matrix 2Σ. Moreover, it turns out that the data likelihood function does not depend on individual values s_1, s_2 and θ_1, θ_2 but rather on their differences, $\Delta\theta$ and Δs. Thus, for a sequence of data observations $(\Delta s_i, \Delta\theta_i), i = 1, \dots, N$ maximization of the data likelihood can be stated as follows:

$$\underset{\Sigma}{\operatorname{argmax}}\{\mathcal{L}(\Sigma; \Delta s_{1:N}, \Delta\theta_{1:N})\} \sim \underset{\Sigma}{\operatorname{argmin}}\left\{N\ln(|\Sigma|) + \sum_{i=1}^{N}\left(\frac{\Delta s_i - \Delta\theta_i}{\sqrt{2}}\right)^T \Sigma^{-1}\left(\frac{\Delta s_i - \Delta\theta_i}{\sqrt{2}}\right)\right\} \tag{7.16}$$

where ~ denotes equivalence of optimization problems. Evidently, the bias cancels-out in the posterior and for the same reason, it also does not appear in the data likelihood function. On the other hand, the covariance matrix Σ can now be easily recovered using the standard MLE formula for Gaussian distributions (Koller and Friedman 2009):

$$\hat{\Sigma} = \frac{1}{2N}\sum_{i=1}^{N}(\Delta s_i - \Delta\theta_i)\,(\Delta s_i - \Delta\theta_i)^T \tag{7.17}$$

7.2.8 Results

Depending on the number of IMU samples between valid GPS readings, the filter is able to update the trajectory up to a number of poses in the past. Each measurement imposes the following update to the information matrix:

$$\Omega \leftarrow \Omega + \left(\left.\frac{\partial f_{t,n}}{\partial s_{t:t+n}}\right|_{\hat{s}_{t:t+n}}\right)^T A^T Q_t^{-1} A\left(\left.\frac{\partial f_{t,n}}{\partial s_{t:t+n}}\right|_{\hat{s}_{t:t+n}}\right) \tag{7.18}$$

where $\hat{s}_{t:t+n}$ is the current mean of $s_{t:t+n}$, $\left.\frac{\partial f_{t,n}}{\partial s_{t:t+n}}\right|_{\hat{s}_{t:t+n}}$ is the Jacobian of $f_{t,n}$ evaluated at $\hat{s}_{t:t+n}$, Q_t is a covariance matrix obtained according to equation (7.17) and A is a variable “cropping” matrix such that, $s_{t:t+n} = Ax_{t:t+n}$. The respective update to the information vector is,

$$\xi \leftarrow \xi + \left(\frac{\partial f_{t,n}}{\partial s_{t:t+n}}\right)^T A^T Q_t^{-1}\left(\frac{\partial f_{t,n}}{\partial s_{t:t+n}}\hat{s}_{t:t+n} - f_{t,n}(\hat{s}_{t:t+n})\right) \tag{7.19}$$

Figures 7.6-8 illustrate the recovered trajectories for 3 routes of length 0.8, 1.2 and 0.6 km. Orientation is estimated purely on the gyro readings and intermittent position measurements (blue diamond markers) are used to correct the entire set of past vehicle poses all the way to the previous measurement. The red crosses indicate a dense GPS based trajectory used as ground truth. Axis unit length is 1 m in all dimensions.

The filter uses a simple smoothing constraint in lieu of a motion model between 3 consecutive positions (a Gaussian regularization factor on the second derivative of the position); for the first transition, the vehicle is assumed to be travelling along the *x*-axis at a speed equal to the speed over ground provided by the GPS. Motion updates occur roughly every 6-7 IMU samples (i.e., approximately 30 Hz). Clearly, the measurement model of equation (7.8) casts a non-linear quadratic constraint into the cost function corresponding to the joint posterior and for this reason, a Levenberg-Marquardt algorithm is employed in order to execute the Gauss-Newton method over the batch of active position variables. Note here that IMU based orientation estimates are highly accurate with very little drift and the GPS can hardly compete in that aspect.

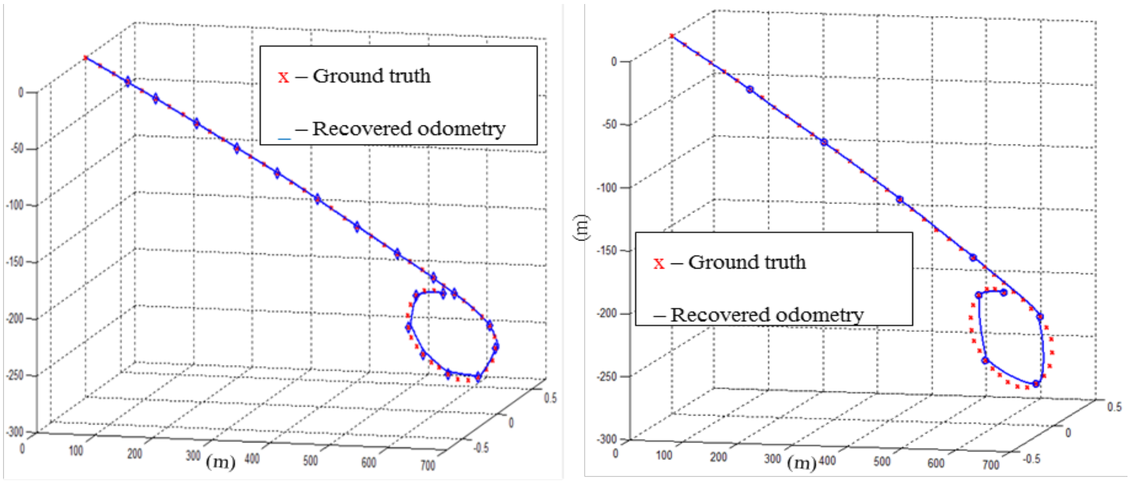

Figure 7.6. Recovered odometry for an approximately 0.8 km long route for a GPS reception period of 3 and 6 s respectively.

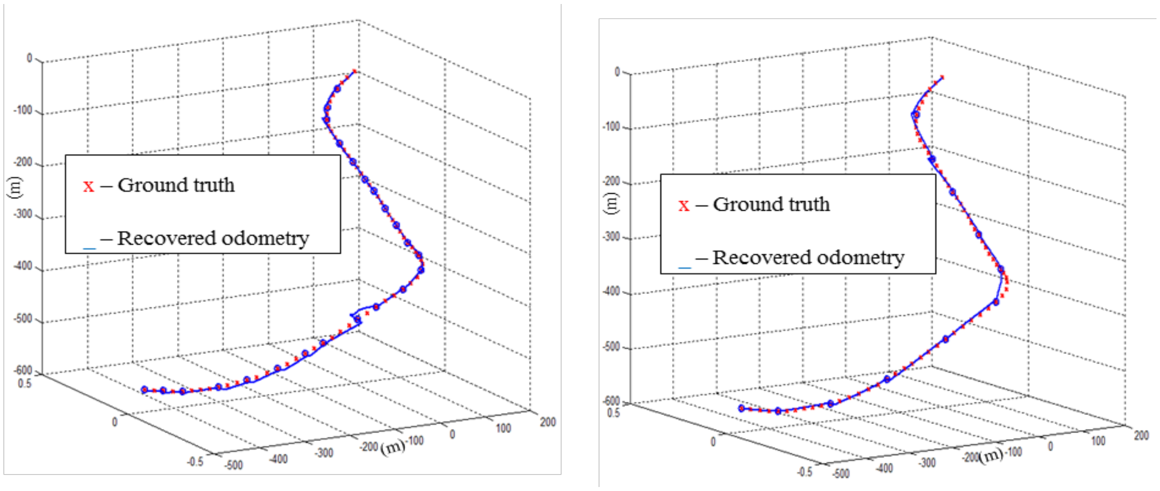

Figure 7.7. Recovered odometry for an approximately 1.2 km long route for a GPS reception period of 3 and 6 s respectively.

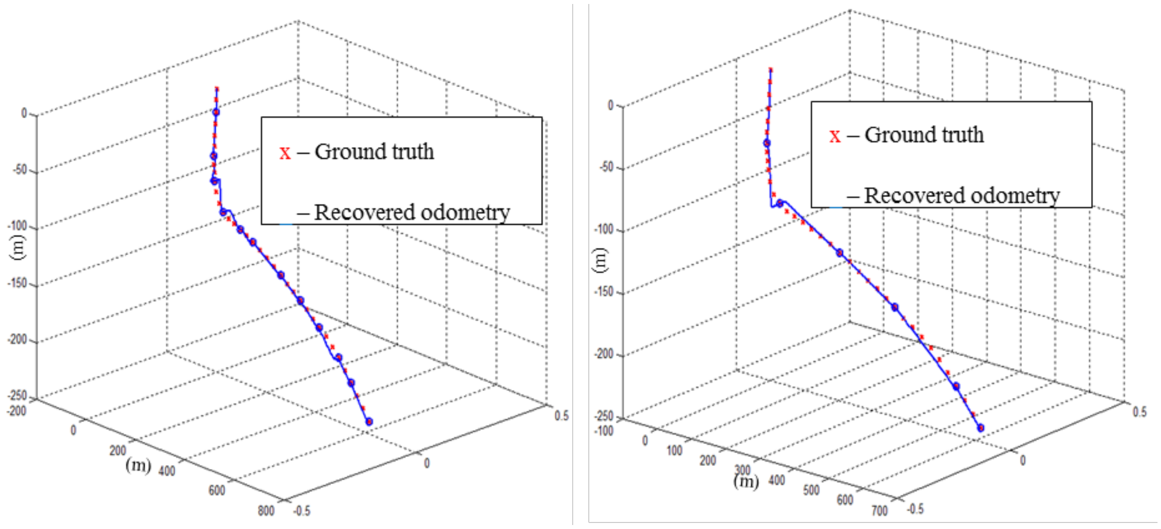

Figure 7.8. Recovered odometry for an approximately 0.6 km long route for a GPS reception period of 3 and 6 s respectively.

A measurement update has a drastic impact on the pose variables that are currently active in the information matrix due to the fact that the derivatives of the projector cast strong correlations between the position vectors and the GPS measurement. Thus, upon exiting the optimization loop, discontinuities may become apparent in the transition from the last vector in the previous batch of positions to the first vector of the most recent batch. To mitigate these discontinuities when "crossing the border" from one batch to another, a regularizing constraint is applied to the difference of the first position vector in the new batch and the last position vector in the previous batch. Figures 7.6-8 indicate that there exists a small amount of variance accompanied by minor "wobbling" (lack of smoothness) in the *y*-axis which can be attributed to the fact that GPS is essentially uninformative in this direction, unless the recovered motion plane is significantly skewed from the GPS plane. Table 7.2 illustrates the average distance from the original dense GPS samples in each one of the 3 routes (0.6, 0.8 and 1.2 km) for batch sizes of 30, 70, 120 and 150, roughly corresponding to intermittent GPS operation in respective intervals of 2 s, 3 s, 4 s and 6 s. The distance between the dense GPS point sequence and a position vector in the recovered odometry is simply the closest GPS point to that particular position.

	Average distance from dense GPS points per route		
GPS sampling time	1 (0.6 km)	2 (0.8 km)	3 (1.2 km)
2 s	0.0294 m	0.2154 m	0.1432 m

3 s	0.5355 m	0.7692 m	0.6771 m
4 s	1.1230 m	1.7064 m	1.3806 m
6 s	1.2998 m	2.6534 m	2.3207 m

Table 7.2. Average distance of the recovered odometry from the original dense GPS point sequence (used as ground truth) for 3 different routes and 4 GPS sampling rates.

7.3 Summary

An alternative filtering approach was introduced in this chapter. Using the Fisher parametrization of Gaussian distributions, it becomes possible to manage an arbitrary number of variables in the posterior at run-time by incorporating the respective quadratic constraints directly into the distribution parameters without necessarily having to resolve the estimates instantaneously, in contrast to purely online algorithms such as the EKF. Thus, the information of a measurement can be disseminated into an arbitrary number of pose vectors comprising the state joint variable at some given step of the stochastic process.

The flexibility of information filters in terms of using arbitrary number of active variables in the state vector provides an ideal solution for cases in which measurement information is provided at a much slower pace compared to the rate of progression of the process in time. The GPS measurement is a classic example of such a “slow” measurement model. Using GPS information in the context of the EKF, although reduces uncertainty in the state belief, it is however a cause of an abrupt discontinuity with respect to the previous belief state.

An alternative method for introducing 2D position information in a discrete stochastic process over the pose of a vehicle through time was also presented in this chapter. In particular, a two measurement models are proposed: a) A linear model which guarantees linear-time solution for pose estimates since the last measurement update and, b) a nonlinear model relying on the concept of approximately planar motion and the comparison of the position measurement with the predicted pose on this plane. This approach not only models the correlations between past states (poses) and the position reading provided by the GPS, but also provides a “conduit” (the motion plane itself) for information to flow from the 2D observation into the 6 dimensions of all active pose vectors. It should be however stressed that the aforementioned measurement approaches can be easily adopted in the context of GraphSLAM, whereas in the case of a KF, it would require a very complicated transition model.

Chapter 8

Monocular visual SLAM in natural environments: Algorithms and frameworks

In this chapter, the full visual SLAM problem in the context of natural environments is addressed. Two different approaches are examined: a) The standard monocular visual SLAM approach in which an initial reconstruction is obtained and thereafter, camera pose is estimated based on the existing map (which is refined by means of bundle adjustment and populated by new triangulated features) and, b) A more “relaxed” approach to visual SLAM, in which the map is not actively used to obtain camera pose, but merely for scale propagation and outlier rejection; in particular, camera pose is estimated directly from image correspondences with the linear solution for baseline described in Chapter 6 (section 4.1) using gyroscopic input to eliminate the effects of rotation from the correspondences.

The perspective-n-point (PnP) problem is discussed in detail in this chapter. The PnP algorithm is the “main engine” of the SLAM framework and it is therefore examined in relation to epipolar geometry and in regards to the various ways in which it can be used to recover camera pose in scenes involving unlimited depth variation and noisy tracking. It follows that potential improvements to the existing solutions are discussed/proposed, especially to accommodate the needs of problems associated with the aforementioned limitations present in natural scenes.

Bundle adjustment is another important aspect of the full visual SLAM problem and therefore it deserves a section here, more than anywhere else in this thesis. It is a process extremely useful in mitigating accumulated errors caused either by 3D point back-projections through the map when using the PnP, or by mismatches in tracking. A feature oriented version of bundle adjustment is discussed, wherein the reconstructed points are parametrized only in terms of their depth in the frame in which they were originally detected; moreover, the use of stereographic projection to constrain the baseline norm is proposed when a 2-view scene reconstruction is optimized.

Finally, results from the two approaches for SLAM are reported, mostly pictorially, in terms of the recovered odometries against outlines of ground truth in satellite photos. Most importantly, this chapter pinpoints the actual weaknesses of traditional visual SLAM techniques in natural landscapes and it elaborates on why certain less complicated and not-so-traditional techniques demonstrate remarkable robustness in odometry estimates for reasonably long sequences.

8.1 The perspective-n-point problem

The perspective-n-point (PnP) problem deals with the recovery of camera pose from a known set of world points and their on-camera projections. The PnP problem is overdetermined for any number of points above 6, while the minimum-size configuration involves 3 correspondences. Similarly to the problem of Euclidean epipolar geometry (i.e., the recovery of structure and relative pose from the essential matrix), the special cases of the PnP have also been solved analytically and these solutions are typically employed in the context of RANSAC with good results. The overdetermined case was fully solved by Hesch and Roumeliotis (Hesch and Roumeliotis 2011) using Groebner basis solvers for the resulting quartic polynomials. Other less complicated and easier to reproduce (from an implementation standpoint) least squares based solutions exist, but they actually employ the Procrustean logic (Chapter 6, section 2.2) either directly or indirectly, thereby enforcing orthogonality in a retrospective manner.

8.1.1 The P3P problem

The P3P problem can be solved fast and relatively easy, despite the fact that it involves quartic polynomials. In the full visual SLAM approach using only a camera in this thesis, the P3P algorithm is used in the context of RANSAC in order to estimate new camera pose as new frames from the sequence are processed. Figure 8.1 illustrates the setup of the P3P problem.

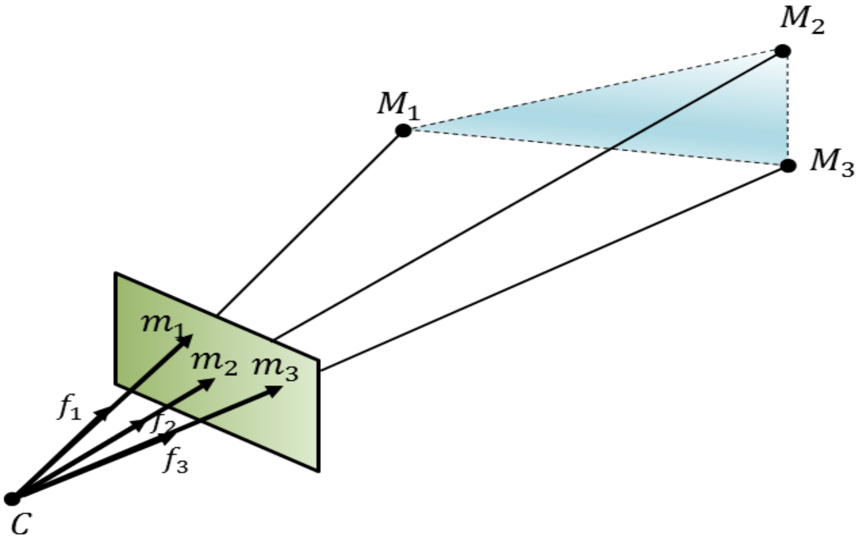

Figure 8.1. The P3P problem setup. World points M_1, M_2 M_3 correspond to the normalized Euclidean projections m_1, , m_2, m_3 on a camera centered at C.

For 3 known map points M_1, M_2 , M_3 and their normalized Euclidean projections m_1, m_2, m_3 onto a camera plane of unknown orientation and location, the 3 cosine laws apply for triangles M_1CM_2, M_1CM_3 and M_2CM_3:

$$l_1^2 + l_2^2 - 2l_1l_2\frac{m_1^T m_2}{\|m_1\|\|m_2\|} = \left\|\overrightarrow{M_1M_2}\right\|^2 \tag{8.1}$$

$$l_1^2 + l_3^2 - 2l_1l_3\frac{m_1^T m_3}{\|m_1\|\|m_3\|} = \left\|\overrightarrow{M_1M_3}\right\|^2 \tag{8.2}$$

$$l_2^2 + l_3^2 - 2l_2l_3\frac{m_2^T m_3}{\|m_2\|\|m_3\|} = \left\|\overrightarrow{M_2M_3}\right\|^2 \tag{8.3}$$

where $l_1 = \left\|\overrightarrow{CM_1}\right\|$, $l_2 = \left\|\overrightarrow{CM_2}\right\|$, $l_3 = \left\|\overrightarrow{CM_3}\right\|$ are the lengths of the segments CM_1, CM_2 and CM_3. The P3P problem can be thought of as the *ultimate high-school exercise on the law of cosines*. Interestingly, the system of equations (8.1-3) is an exact description of the P3P problem from every possible aspect and always leads to a 4th degree polynomial system. There have been many solutions proposed in literature, the most prominent of which are described in a great paper by Haralick (Haralick, Lee et al. 1994). Recently, Kneip proposed a different parametrization (Kneip, Scaramuzza et al. 2011) which, although brilliant as a conception, it nevertheless does not improve the final solution, as it still requires the computation of quartic polynomial roots. For the RANSAC based P3P solver used in this research, the parametrization of Grunert (Grunert 1841) is used in conjunction with a simple 4th order polynomial solver (Lourakis and Zabulis 2013)[19]. Grunert's algorithm yields a solution only for the lengths l_1, l_2, l_3. Solving for the camera pose requires just a few more simple steps.

The orientation matrix R containing the camera frame in a column-wise fashion can be obtained by matching a known orthogonal triad (i.e., three orthogonal directions) in the world frame with the same triad in the camera frame (Black 1964). With l_1, l_2, l_3 in estimated, this triad can be chosen to be the directions of vectors $\overrightarrow{M_1M_2}$, $\overrightarrow{M_1M_2} \times \overrightarrow{M_1M_3}$ and $\overrightarrow{M_1M_2} \times \left(\overrightarrow{M_1M_2} \times \overrightarrow{M_1M_3}\right)$. Thus, in the world frame, the triad (w_1, w_2, w_3) is:

$$w_1 = \frac{\overrightarrow{M_1M_2}}{\left\|\overrightarrow{M_1M_2}\right\|} \tag{8.4}$$

$$w_2 = \frac{\overrightarrow{M_1M_2} \times \overrightarrow{M_1M_3}}{\left\|\overrightarrow{M_1M_2} \times \overrightarrow{M_1M_3}\right\|} \tag{8.5}$$

$$w_3 = w_1 \times w_2 \tag{8.6}$$

In the camera frame, the corresponding directions (u_1, u_2, u_3) are:

$$u_1 = \frac{l_2 f_2 - l_1 f_1}{\|\mathrm{l}_2 f_2 - l_1 f_1\|} \tag{8.7}$$

$$u_2 = \frac{(l_2 f_2 - l_1 f_1) \times (l_3 f_3 - l_1 f_1)}{\|(l_2 f_2 - l_1 f_1) \times (l_3 f_3 - l_1 f_1)\|} \tag{8.8}$$

$$u_3 = u_1 \times u_2 \tag{8.9}$$

where $f_1 = m_1/\|\mathrm{m}_1\|$, $f_2 = m_2/\|\mathrm{m}_2\|$, $f_3 = m_3/\|\mathrm{m}_3\|$ are the unit vectors along the projection rays in the directions of m_1, m_2, m_3 in the camera frame. Obtaining the rotation matrix R is a matter of a simple multiplication:

$$R = [w_1 \quad w_2 \quad w_3][u_1 \quad u_2 \quad u]^T \tag{8.10}$$

Finally, the baseline vector b from the world origin to the camera center in the world frame (in other words, the position of the camera) is given by the following difference of barycenters:

$$b = \frac{1}{3}\left(M_1 + M_2 + M_3 - R(l_1 f_1 + l_2 f_2 + l_3 f_3)\right) \tag{8.11}$$

[19] Code can be downloaded from: http://users.ics.forth.gr/~lourakis/posest/

It should be noted here that the quartic polynomial can yield up to four solutions for l_1, l_2, l_3 and each one will correspond to a camera pose. It follows that the only way to choose the correct solution is by testing it on a fourth correspondence.

8.1.2 The overdetermined PnP problem and the cases of concealed Procrustes

The special cases for $n = 3,4,5,6$ are of particular interest because the respective solution spaces are more confined than the general case. Furthermore, these algorithms can be used in the context of RANSAC to produce robust pose estimates. However, the overdetermined case is the most natural configuration in practice and it is often desirable (instead of resorting to Monte Carlo methods) to solve directly from the entire available data.

The most common formulation of the PnP problem is based on the reprojection error. Consider n world points $M_1, M_2, \ldots, M_n$, $n > 2$ and their respective normalized Euclidean projections $m_1, m_2, \ldots, m_n$ in the camera frame. Let R be the rotation matrix containing the directions of the camera frame (in world coordinates) arranged column-wise and b be the baseline vector from the world origin to the camera center in world coordinates. Then, the PnP problem can be stated as follows:

$$\underset{R,b}{minimize} \sum_{i=1}^{n} \left\| m_i - \frac{1}{r_3^T(M_i - b)} R^T(M_i - b) \right\|^2 \tag{8.12}$$

$$subject\ to{:}\, R^T R = 1\,,\ \det(R) = 1$$

where r_3 is the 3$^{\text{d}}$ column of R. For $n = 3$, substituting $m_i = \|m_i\| f_i$ and $r_3^T(M_i - b) = l_i / \|m_i\|$ in equation (8.12) and with a few manipulations, one is able to derive the camera pose solutions given in equations (8.10-11).

There have been several algorithms proposed for the solution of the overdetermined PnP problem. They can be roughly divided into two categories: a) Solutions which pay respect to the orthogonality constraint and always lead to a system of quartic polynomials (or multiple quadratics) and, b) Solutions that solve a linear system (least squares) and enforce the orthogonality constraint directly or indirectly in the process. In the first category one finds the method by Hesch and Roumeliotis (Hesch and Roumeliotis 2011) or the most recent by Zheng and Wu (Zheng and Wu 2015); amongst linear solvers, it is worth mentioning *EPnP* by Lepetit (Lepetit, Moreno-Noguer et al. 2009) for having employed a very clever data transformation.

I would like to stress here that any method that employs a linear solver (and that includes SVD) at any stage is a Procrustean method either in a straightforward or a concealed manner. This claim can be justified with the consideration that data are assumed to be normally, independently and (usually) identically distributed and therefore a least squares solution is nothing but a marginal over a very large joint variable that includes both data and camera pose. By solving for the pose without taking the constraints into consideration at some particular stage of the solution, one is treating the camera pose as a normally distributed variable, something that is egregiously false. Orthonormality constraints imply that the camera pose follows a *multimodal* distribution (hence the more-than-one solutions which may or may not have multiplicity above 1) and therefore using least squares is a brutal way of ignoring the true distribution of the pose variables. This does not mean that *EPnP* or other linear solutions are not effective. It merely implies that this category of methods can often fail in situations that polynomial solvers may be much more robust, especially when the data are contaminated with high levels of noise. It should be noted here that natural environments presenting significant variations in scene depth are actually such cases. Typically, tracking noise is augmented through triangulation, especially in the cases of distant world points, and unfortunately, this noise is propagated into the new pose estimate through the PnP.

8.1.3 A formulation for the overdetermined PnP problem

To the best of the author’s knowledge, the following PnP formulation and parametrization is novel. Consider the following slightly-altered version of the optimization problem in equation (8.12):

$$\underset{R,t}{minimize} \left\{ J = \sum_{i=1}^{n} \|(r^3 M_i + t)m_i - (RM_i + t)\|^2 \right\} \qquad (8.13)$$

$$subject\ to: R^T R = 1\ ,\ \det(R) = 1$$

where R contains the camera frame directions arranged row-wise (as opposed to the column-wise convention adopted in this thesis), $t = -Rb$ is the world origin in the camera frame and r^3 is the 3^{d} row of R as a 1×3 vector. Using vector $1_z = [0 \quad 0 \quad 1]^T$ and the identity $(a^T b)c = (ca^T)b$ for any 3 vectors, the expression of the cost function J becomes:

$$J = \sum_{i=1}^{n} \|1_z^T (RM_i + t)m_i - (RM_i + t)\|^2$$

where $C_i^{1/2} = 1_z^T m_i - I_3$. Define $C_i = \left(C_i^{1/2}\right)^T C_i^{1/2}$. With further manipulations in (8.14) the following expression for J is obtained:

$$J = \sum_{i=1}^{n} (RM_i + t)^T C_i (RM_i + t) \tag{8.15}$$

Now, define a matrix D_i such that, $D_i r = RM_i$ where $r = [r^1 \quad r^2 \quad r^3]^T$ is the 9×1 vector of the stacked rows of R:

$$D_i = \begin{bmatrix} M_i^T & 0_{1\times 3} & 0_{1\times 3} \\ 0_{1\times 3} & M_i^T & 0_{1\times 3} \\ 0_{1\times 3} & 0_{1\times 3} & M_i^T \end{bmatrix} \tag{8.16}$$

where $0_{1\times 3}$ is the 1×3 zero vector. The cost function of (8.15) can now be written in terms of r and t as follows:

$$J = \sum_{i=1}^{n} (D_i r + t)^T C_i (D_i r \ + t) \tag{8.17}$$

Taking the derivative of J in terms of t and setting it zero yields the following relationship between t and r:

$$\left(\sum_{i}^{n} C_i\right) t = -\left(\sum_{i}^{n} C_i D_i\right) r \tag{8.18}$$

What is of particular interest in equation (8.18) is the apparent role of matrix C_i loosely as a spatial “weight”; this view also allows us to see t as a “weighted” average of the three directions r^1, r^2 and r^3. Provided that $\sum_i^n C_i$ is invertible (which most likely should be the case in the overdetermined problem), substituting back in (8.17) yields an expression for J only in terms of r:

$$J = r^T \underbrace{\sum_{i=1}^{n} \left(D_i - \left(\sum_{i}^{n} C_i\right)^{-1} \left(\sum_{i}^{n} C_i D_i\right)\right)^T C_i \left(D_i - \left(\sum_{i}^{n} C_i\right)^{-1} \left(\sum_{i}^{n} C_i D_i\right)\right)}_{\Omega} r \tag{8.19}$$

where Ω is a data-dependent positive semidefinite matrix. The problem now becomes a *quadratically constrained quadratic program* in r. In the very unlikely case that Ω has 1-dimensional null space, then the unique element of the basis would be the solution; in any other case in which the dimension of the null space is greater than 1, the solution should be the linear combination of the basis elements that fulfil the orthonormality constraints.

In practice, the matrix Ω will almost always have empty null space. In this case, Ω will have 9 distinct eigenvectors $\omega_1, \omega_2, \ldots, \omega_9 \in \mathbb{R}^9$ corresponding to 9 strictly positive eigenvalues $\sigma_1, \sigma_2, \ldots, \sigma_9$. Evidently, $\omega_1, \omega_2, \ldots, \omega_9$ constitute a basis for $\mathbb{R}^9$ and therefore r can be written as a linear combination of the eigenvectors of Ω:

$$r = \sum_{i=1}^{9} \alpha_i \omega_i = \sum_{i=1}^{9} \alpha_i \underbrace{\begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix}}_{\omega_i} \tag{8.20}$$

where the parametrization $\omega_i = [u_i^T \quad v_i^T \quad w_i^T]^T$ in terms of vectors $u_i, v_i, w_i \in \mathbb{R}^3$ is introduced for convenience of notation in subsequent definitions and derivations. Substituting from (8.20) into the cost function in (8.19) yields:

$$J = (\alpha_1\omega_1 + \alpha_2\omega_2 + \cdots + \alpha_9\omega_9)^T \Omega (\alpha_1\omega_1 + \alpha_2\omega_2 + \cdots + \alpha_9\omega_9)$$

$$\Leftrightarrow J = (\alpha_1\omega_1 + \cdots + \alpha_9\omega_9)^T \left(\alpha_1 \underbrace{(\Omega\omega_1)}_{\sigma_1\omega_1} + \cdots + \alpha_9 \underbrace{(\Omega\omega_9)}_{\sigma_9\omega_9} \right)$$

$$\Leftrightarrow J = (\alpha_1\omega_1 + \alpha_2\omega_2 + \cdots + \alpha_9\omega_9)^T (\sigma_1\alpha_1\omega_1 + \cdots + \sigma_9\alpha_9\omega_9)$$

$$\Leftrightarrow J = \sum_{i=1}^{9} \sigma_i \alpha_i^2 = \alpha^T \Sigma a \tag{8.21}$$

where $\alpha^T = [\alpha_1 \;\; \ldots \;\; \alpha_9]$ and $\Sigma = diag\{\sigma_i\}$. Thus, the optimization problem can now be re-written as follows:

$$\begin{gathered} \underset{a}{minimize} \{J = \alpha^T \Sigma a\} \\ subject\ to: \\ \alpha^T(U^T V)\alpha = \alpha^T(U^T W)\alpha = \alpha^T(V^T W)\alpha = 0 \\ \alpha^T(U^T U)\alpha = \alpha^T(V^T V)\alpha = \alpha^T(W^T W)\alpha = 1 \end{gathered} \tag{8.22}$$

where $U, V, W \in \mathbb{R}^{3\times 9}$ are the matrices formed column-wise by vectors u_i, v_i, w_i respectively. The cost function of (8.21) is a simpler expression compared to the one in (8.19); yet on the other hand, the constraints have now become more complicated. What is important however about the formulation of (8.22) is that it is relatively easy to discern from the unit-norm constraints that the cost function is a (scaled by 3) convex combination of σ_i. This means, that one is able to choose $\alpha_1 = \alpha_2 = \alpha_3 = 0$ in order to eliminate the 3 largest singular values (typically given in descending order). With this choice, we are left with the problem of satisfying the constraints. Therefore, the optimization problem yields a system of 6 quadratic equations over exactly 6 unknowns (i.e., $\alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9$):

$$\begin{aligned} \alpha'^T(U^T V)\alpha' = \alpha'^T(U^T W)\alpha' = \alpha'^T(V^T W)\alpha' = 0 \\ \alpha'^T(U^T U)\alpha' = \alpha'^T(V^T V)\alpha' = \alpha'^T(W^T W)\alpha' = 1 \end{aligned} \tag{8.23}$$

where $\alpha' = [\alpha_4 \quad \alpha_5 \quad \alpha_6 \quad \alpha_7 \quad \alpha_8 \quad \alpha_9]^T$. To the best of the author's knowledge, this system is novel and it appears tractable in terms of finding a solution that can be hard-coded in a program without the use of a Groebner bases solver. Please note that eliminating the 3 largest singular values does not necessarily guarantee (although likely) that the solution of the quadratic system in (8.23) will give the global minimum. However, if the system is solvable in near-constant time, then it would be just a matter of finding a correct combination of zeros[20]. Further work and experimentation is deferred for a postdoctoral stage of the research.

8.2 Bundle adjustment

Iterative adjustment of the reprojection error (commonly known as *bundle adjustment)* between 2 or more views is very important for the stability of online SLAM. Bundle adjustment generally improves the current estimate of the map and camera pose through iterative optimization of the reprojection error. Klein and Murray (Klein and Murray 2007) use "local bundle adjustment" (Mouragnon, Lhuillier et al. 2009) to refine their SLAM posterior to the extent that the estimate is reliable enough to keep the process stable. In the SLAM frameworks described in this chapter, bundle adjustment is typically executed over 2 or 3 views, in order to primarily minimize the error in the new pose estimate prior to the triangulation of recently detected features.

8.2.1 Depth based parametrization of world points in bundle adjustment

In the context of the visual SLAM applications described in this thesis, landmarks are not known a priori. In fact they are points detected in image space and thereafter back-projected onto the normalized Euclidean plane. Thus, it is fair to regard their normalized

[20] This needs to be shown. It would make sense (from a greedy point of view) that when the global minimum is attained, three (at least) of the components of α are zero.

Euclidean projections as non-stochastic quantities, for being the only known "ground truth" about them. Thus, the map location of the feature in the world can be parametrized by means of its depth and the camera pose in its *home frame* (i.e., frame of original detection):

$$M = \exp(d)\, R_h h + b_h \tag{8.25}$$

where d is the logarithm of depth in the home frame, h is the normalized Euclidean projection of the feature in the home frame and (R_h, b_h) is the respective camera frame (using the default convention in this thesis which states that R_h contains the camera frame in a column-wise arrangement and b_h is the position of the camera in world coordinates). The exponential parametrization is used to impose depth positivity. The bundle adjustment cost function is therefore formulated as follows:

$$J = \sum_{v \in \mathcal{V}} \sum_{i \in \mathcal{F}(v)} \left\| m_i - \frac{1}{1_z^T R_v^T (M_i - b_v)} R_v^T (M_i - b_v) \right\|^2 \tag{8.26}$$

where $1_z = [0 \quad 0 \quad 1]^T$, $\mathcal{V}$ is an indexing set for camera views and $\mathcal{F}(v)$ is an indexing set for the features in the view $v \in \mathcal{V}$. It is worth mentioning that since M is parametrized by the feature's depth parameter d and home camera pose (R_h, b_h), it follows that the dimensionality of the respective variables in bundle adjustment has an $O(n)$ scaling as opposed to the $O(3n)$ of standard bundle adjustment. Although it still is linear scaling, it however makes a significant difference for relatively large number of features, especially when inverting, for instance, 300-dimensional instead of 900-dimensional dense matrices.

8.2.2 Constraining scale with stereographic projection

Bundle adjustment is generally restricted to a region of the search space that roughly corresponds to the scale of the map. Thus, there is generally no need to constrain scale at this stage. However, the initial scene reconstruction obtained with Algorithm 3.1 enforces a unit-norm baseline in order to cope with the scale ambiguity for the first time. Thus, to refine the reprojection error in the initialization stage between the first two views, the optimization problem must be cast with a unit-norm baseline constraint:

$$\underset{b,R,d_i}{minimize} \left\{ J = \sum_{i \in \mathcal{F}_0} \left\| m_i - \frac{1}{1_z^T R^T (exp(d_i) h_i - b)} R^T (exp(d_i) h_i - b) \right\|^2 \right\} \tag{8.27}$$
$$subject\ to: \|b\| = 1\ and\ \ R^T R = I_3$$

where d_i and h_i are the depth logarithm and normalized Euclidean projection of the i^{th} feature in the first view and $\mathcal{F}_0$ is an indexing set for the initial set of correspondences. For simplicity, the first camera frame is taken as the world frame. The optimization problem of (8.26) is a non-linear, quadratically constrained, quadratic program. The existing iterative solutions are based on the Levenberg-Marquardt heuristic (Levenberg 1944, Marquardt 1963, Dennis Jr and Schnabel 1996) which can be regarded as an adaptive version of the Gauss-Newton method. The constraint in (8.27) can be a significant problem during the LM execution because a step can very easily lead out of the feasible set of solutions (i.e., yield a baseline that does not have unit length). The most usual way of coping with the constraints is to manipulate the step of the process so that it remains in feasible space (Kanzow, Fukushima et al. 2002, Gong, Meng et al. 2015). However, a very elegant and uncomplicated way of achieving this is with the use of stereographic projection (see Chapter 4, section 4.1 and Appendix A for details on the parametrization). In this case, b can be parametrized by two parameters $\kappa, \lambda \in \mathbb{R}$ and the optimization problem of (8.27) becomes unconstrained:

$$\underset{\kappa,\lambda,R,d_i}{minimize}\left\{J = \sum_{i\in\mathcal{F}_0}\left\|m_i - \frac{R^T\big(exp(d_i)h_i - b(\kappa,\lambda)\big)}{1_z^T R^T\big(exp(d_i)h_i - b(\kappa,\lambda)\big)}\right\|^2\right\} \tag{8.28}$$

where the parametrized baseline $b(\kappa, \lambda)$ is now given by:

$$b(\kappa,\lambda) = \frac{1}{1+\kappa^2+\lambda^2}\begin{bmatrix} 2\kappa \\ 2\lambda \\ 1-\kappa^2-\lambda^2 \end{bmatrix} \tag{8.29}$$

The Jacobian of the baseline is the 3D version of the 4D Jacobian in equations (A.44-46) of Appendix A:

$$\underset{\kappa,\lambda}{\nabla} b = -\begin{bmatrix} b_1^2 - b_3 - 1 & b_1 b_2 \\ b_1 b_2 & b_2^2 - b_1 - 1 \\ b_1(1+b_3) & b_2(1+b_3) \end{bmatrix} \tag{8.30}$$

where b_1, b_2, b_3 are the 3 components of b.

Please note here that bundle adjustment is implemented using the GraphSLAM method to build the least squares normal equations and the Fisher parameters to represent them as the information matrix and vector of a multivariate Gaussian joint distribution. In other words, the optimization problem is formulated by updating the information matrix and vector of the SLAM posterior in the fashion described in Chapter 7, section 1. The very same approach is employed by the least squares solver of Edward Rosten's TooN library (Rosten 2013).

8.2.3 Robust estimation

Robust statistics (Maronna, Martin et al. 2006, Huber 2011) offer significant solutions in estimation problems that can cope with percentages of outliers that reach up to 30% of the data. A prominent category of techniques in robust statistics are *M-estimators* (M-stands for *maximum likelihood*) which interface gracefully with least squares methods, including algorithms for nonlinear optimization such as the Levenberg – Marquardt algorithm.

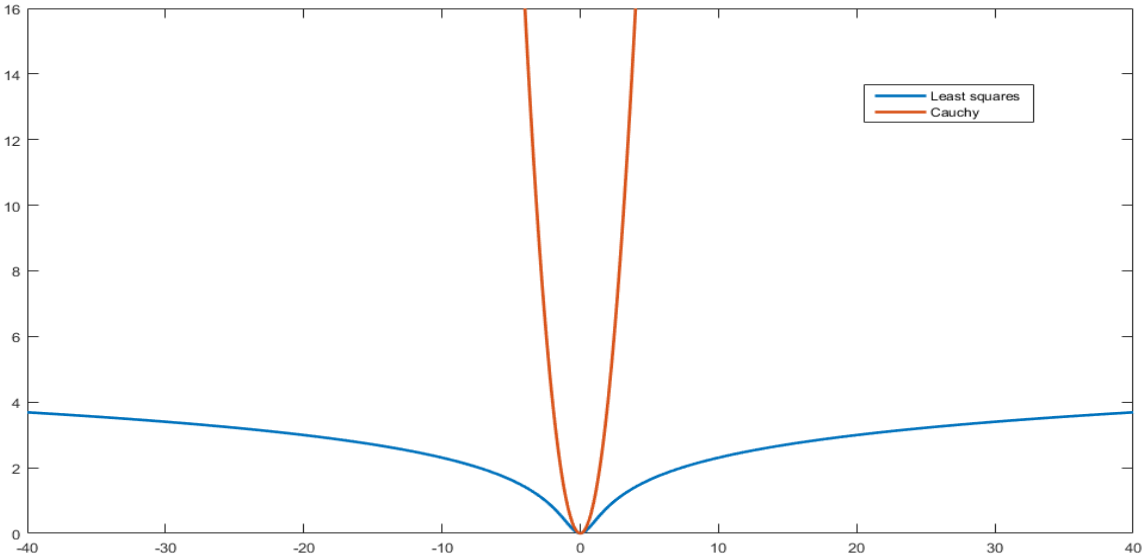

Figure 8.2. The Cauchy cost function, $\frac{c^2}{2} log\left(1+\left(\frac{x}{c}\right)^2\right)$ and the least squares cost function $\left(\frac{x}{c}\right)^2$ for $c = 1$.

The core idea behind M-estimators is τo choose a monotonically increasing, bounded function $\rho(x)$ which will a give a convex, strictly positive, even objective function $\rho(\|e_i\|^2)$ that upper-bounds penalization of errors in order to mitigate the effect of extreme outliers on the estimate and formulate the following minimization problem:

$$\underset{p}{minimize}\left\{J = \sum_{i \in I} \rho(\|e_i\|^2)\right\} \tag{8.31}$$

where p is the sought parameter vector and e_i is the error/residual for the i^{th} datum. There have been quite a few objective functions proposed for M-estimation. They exhibit a "V"-

shaped pattern in a neighborhood of 0, while they approach a finite upper-bound asymptotically as the error grows larger. The Cauchy function shown in Figure 8.2 is a typical representative of these objective functions.

Taking the derivative of *J* and setting it to zero should give the following:

$$\sum_{i\in I}\left(\left.\frac{\partial\rho}{\partial x}\right|_{x=\|e_i\|^2}\right)e_i^T\frac{\partial e_i}{\partial p}=0 \tag{8.32}$$

Provided that the error is a linear (or linearized in the case of Gauss-Newton method) term of the form $e_i = y_i - A_i p$ then equation (8.32) leads to the following relationship for p:

$$\sum_{i\in I}\left(\left.\frac{\partial\rho}{\partial x}\right|_{x=\|e_i\|^2}\right)A_i^T(y_i-A_ip)=0 \tag{8.33}$$

Setting $w_i = \left.\frac{\partial\rho}{\partial x}\right|_{x=\|e_i\|^2}$>0 in (8.49) gives:

$$\sum_{i\in I}w_iA_i^T(y_i-A_ip)=0 \tag{8.34}$$

which, for all intents and purposes yields an ordinary weighted least squares solution:

$$p=(A^TWA)^{-1}A^TWy \tag{8.35}$$

where W is the diagonal matrix of weights (repeated a number of times equal to the dimensionality of y_i per error term), A contains the matrices A_i stacked row-wise and y is the observation vector as a large column.

The only “loose end” in the solution of equation (8.34) is the scale of the error. The errors scale differently with the parameters. For instance, reprojection errors should scale according to scene depth and camera speed; hence, different scenes are likely to exhibit different error ranges. It is therefore necessary to first standardize the errors before using them to obtain the weights. The most common method to scale the errors is to use the so-called *Median Absolute Deviation* (MAD), which is the median of the absolute deviation from the median (in the case of residuals, it is simply the median):

$$MAD=\underset{i}{median}\{|e_i|\} \tag{8.36}$$

The MAD is regarded to be more robust to outliers than standard deviation estimators. Provided that errors are normally distributed, it turns-out that the error scale estimate (standard deviation) is given by:

$$s = \underbrace{1.4826}_{K} \times MAD \tag{8.37}$$

where $K = 1.4826$ is a constant which is the result of the following requirement:

$$P\left(-\frac{MAD}{s} \leq \frac{e_i}{s} \leq \frac{MAD}{s}\right) = \frac{1}{2} \tag{8.38}$$

And since we typically assume that $\frac{e_i}{s}$ is a standard Gaussian variable, it follows that s can be computed from its cumulative function. More on the subject can be found in the books by Huber, Maronna, Hoaglin (Hoaglin, Mosteller et al. 1983, Maronna, Martin et al. 2006, Huber 2011).

Algorithm 8.1 describes the Levenberg-Marquardt method combined with robust estimation for bundle adjustment. To keep it simple, the algorithm describes adjustment of reprojection error in the case of the initializing two views in SLAM (hence, pose is 5D because it is parametrized with stereographic coordinates); however, roughly the same procedure applies (with minor differences primarily in the measurement formulas) to an arbitrary set of consecutive frames in the sequence.

Algorithm 8.1. ***Robust bundle adjustment (for the initial 2-view reconstruction)***

Input*: a) Indexing sets of visible features in the two views, $\mathcal{F}_1$ and $\mathcal{F}_2$.*

b) Measurements $h_i, i \in \mathcal{F}_1$ and $g_j, j \in \mathcal{F}_2$.

c) Feature depth logarithms $d_i, i \in \mathcal{F}_1$.

d) Relative camera pose (ψ, κ, λ) where ψ are orientation parameters and $\kappa, \lambda \in \mathbb{R}$ are the stereographic coordinates of the unit-norm baseline.

e) An M-estimator objective function ρ.

Output: *a) Updated feature depth estimates $\hat{d}_i, i \in \mathcal{F}_1$.*

b) Updated relative camera pose estimate $(\hat{\psi}, \hat{\kappa}, \hat{\lambda})$.

```
comment     Number of measurements.
n ← $|\mathcal{F}_1 \cap \mathcal{F}_2|$
comment     Create the information matrix as the $(n + 5) \times (n + 5)$ identity matrix.
```

```
$\Omega \leftarrow I_{(n+5)}$
comment    Assign current depth and pose estimates to a $(n + 5) \times 1$ vector.
$\mu \leftarrow \left[\hat{d}_{f_1} \ \dots \ \hat{d}_{f_n} \ \ \hat{\psi} \ \hat{\kappa} \ \hat{\lambda}\right]$ s.t. $f_1, \dots, f_n \in \mathcal{F}_1 \cap \mathcal{F}_2$
comment    Initial information vector is equal to the mean.
$\xi \leftarrow \mu$
comment    The LM constants associated with termination conditions.
$\varepsilon_1 \leftarrow 10^{-5}$; $\varepsilon_2 \leftarrow 10^{-12}$; Timeout $\leftarrow 40$
comment    Working-out the squared errors and the M-estimator weights.
SqErrors $\leftarrow \emptyset$
SqError $\leftarrow 0$
$\hat{\psi} = [\mu(n+1) \quad \mu(n+2) \quad \mu(n+3)]$; $\hat{\kappa} = \mu(n+4)$; $\hat{\lambda} = \mu(n+5)$
$\hat{R} = \text{rotation matrix}(\hat{\psi})$; $\hat{b} = \text{baseline vector}(\hat{\kappa}, \hat{\lambda})$
For each $i \in \mathcal{F}_1 \cap \mathcal{F}_2$:
    $\hat{d}_i = \mu(i)$
    $\hat{M}_i \leftarrow \exp(\hat{d}_i) h_i$
    $e_i^2 \leftarrow \left\| g_i - \frac{\hat{R}^T(\hat{M}_i - \hat{b})}{1_z^T \hat{R}^T(\hat{M}_i - \hat{b})} \right\|^2$
    SqErrors $\leftarrow$ SqErrors $\cup \{e_i^2\}$
    SqError $\leftarrow$ SqError $+ w_i e_i^2$
$\text{MAD} = \sqrt{\text{median}\{\text{SqErrors}\}}$
$s = 1.4826 \times \text{MAD}$
NormSqErrors $\leftarrow \frac{\text{SqErrors}}{s^2}$
$W \leftarrow \{w_i = \rho'(e_i^2) \mid i \in \mathcal{F}_1 \cap \mathcal{F}_2\}$
comment    Entering the LM main loop.
minError $\leftarrow$ SqError; Found $\leftarrow$ (SqError $\leq \varepsilon_1$)
$\tau \leftarrow 10^{-3}$; $k \leftarrow 0$
While (Not Found And $k \leq$ Timeout):
    PreviousSqError $\leftarrow$ SqError
    $k \leftarrow k + 1$
    $\Omega_{\text{temp}} \leftarrow \Omega$; $\xi_{\text{temp}} \leftarrow \xi$
```

```
For each $\mathrm{i} \in \mathcal{F}_1 \cap \mathcal{F}_2$:
    comment    Create a “cropping” matrix for $d_i$ and $\psi, \kappa, \lambda$.
    $\mathrm{A} \leftarrow \{\mathrm{A}\text{ is a } 5 \times (\mathrm{n}+5) \text{ matrix such that: } [\hat{\psi}\ \hat{\kappa}\ \hat{\lambda}\ \hat{\mathrm{d}}_\mathrm{i}] = \mathrm{A}\mu\}$.
    $[\hat{\psi}\ \hat{\kappa}\ \hat{\lambda}\ \hat{\mathrm{d}}_\mathrm{i}] \leftarrow \mathrm{A}\mu$; $\hat{\mathrm{M}} \leftarrow \exp(\hat{\mathrm{d}}_\mathrm{i})\mathrm{h}_\mathrm{i}$;
    $\hat{\mathrm{R}} = \text{rotation matrix}(\hat{\psi})$; $\hat{\mathrm{b}} = \text{baseline vector}(\hat{\kappa}, \hat{\lambda})$
    comment    Update the Fisher parameters. G is the measurement Jacobian.
    $\mathrm{G}_\mathrm{i} = \left[ \left.\frac{\partial\left(\frac{\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}{1_\mathrm{z}^\mathrm{T}\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}\right)}{\partial\psi}\right|_{\hat{\psi}} \quad \left.\frac{\partial\left(\frac{\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}{1_\mathrm{z}^\mathrm{T}\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}\right)}{\partial\kappa}\right|_{\hat{\kappa}} \quad \left.\frac{\partial\left(\frac{\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}{1_\mathrm{z}^\mathrm{T}\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}\right)}{\partial\lambda}\right|_{\hat{\lambda}} \quad \left.\frac{\partial\left(\frac{\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}{1_\mathrm{z}^\mathrm{T}\mathrm{R}^\mathrm{T}(\mathrm{M}_\mathrm{i}-\mathrm{b})}\right)}{\partial\mathrm{d}_\mathrm{i}}\right|_{\hat{\mathrm{d}}_\mathrm{i}} \right]$
    $\mathrm{w}_\mathrm{i} = \mathrm{W}(\mathrm{i})$
    $\Omega_{\mathrm{temp}} \leftarrow \Omega_{\mathrm{temp}} + \mathrm{w}_\mathrm{i}\mathrm{A}^\mathrm{T}\mathrm{G}_\mathrm{i}^\mathrm{T}\mathrm{G}_\mathrm{i}\mathrm{A}$
    $\xi_{\mathrm{temp}} \leftarrow \xi_{\mathrm{temp}} + \mathrm{w}_\mathrm{i}\mathrm{A}^\mathrm{T}\mathrm{G}_\mathrm{i}^\mathrm{T}\left(\mathrm{g}_\mathrm{i} + \mathrm{G}_\mathrm{i}\begin{bmatrix}\hat{\psi}\\ \hat{\kappa}\\ \hat{\lambda}\\ \hat{\mathrm{d}}_\mathrm{i}\end{bmatrix} - \frac{\hat{\mathrm{R}}^\mathrm{T}(\hat{\mathrm{M}}_\mathrm{i}-\hat{\mathrm{b}})}{1_\mathrm{z}^\mathrm{T}\hat{\mathrm{R}}^\mathrm{T}(\hat{\mathrm{M}}_\mathrm{i}-\hat{\mathrm{b}})}\right)$
    comment    Obtaining a new estimate.
    $\Omega_{\mathrm{temp}} \leftarrow \Omega_{\mathrm{temp}} + \tau\mathrm{I}$
    $\xi_{\mathrm{temp}} \leftarrow \xi_{\mathrm{temp}} + \tau\mu$
    $\mu_{\mathrm{temp}} \leftarrow \Omega_{\mathrm{temp}}^{-1}\xi_{\mathrm{temp}}$
comment    Working-out the squared errors and the M-estimator weights.
$\mathrm{SqErrors} \leftarrow \emptyset$
$\mathrm{SqError} \leftarrow 0$
For each $\mathrm{i} \in \mathcal{F}_1 \cap \mathcal{F}_2$:
    comment    Create a “cropping” matrix for $d_i$ and $(\psi, b)$.
    $\mathrm{A} \leftarrow \{\mathrm{A}\text{ is a } 5 \times (\mathrm{n}+5) \text{ matrix s.t.: } [\hat{\psi}\ \hat{\kappa}\ \hat{\lambda}\ \hat{\mathrm{d}}_\mathrm{i}] = \mathrm{A}\mu_{\mathrm{temp}}\}$
    $[\hat{\psi}\ \hat{\kappa}\ \hat{\lambda}\ \hat{\mathrm{d}}_\mathrm{i}] \leftarrow \mathrm{A}\mu_{\mathrm{temp}}$
    $\hat{\mathrm{R}} = \text{rotation matrix}(\hat{\psi})$; $\hat{\mathrm{b}} = \text{baseline vector}(\hat{\kappa}, \hat{\lambda})$
    comment    Compute the 3D point in the first camera frame.
    $\hat{\mathrm{M}}_\mathrm{i} \leftarrow \exp(\hat{\mathrm{d}}_\mathrm{i})\mathrm{h}_\mathrm{i}$
    $\mathrm{w}_\mathrm{i} = \mathrm{W}(i)$
    $\mathrm{e}_\mathrm{i}^2 \leftarrow \left\| \mathrm{g}_\mathrm{i} - \frac{\hat{\mathrm{R}}^\mathrm{T}(\hat{\mathrm{M}}_\mathrm{i}-\hat{\mathrm{b}})}{1_\mathrm{z}^\mathrm{T}\hat{\mathrm{R}}^\mathrm{T}(\hat{\mathrm{M}}_\mathrm{i}-\hat{\mathrm{b}})} \right\|^2$
```

```
        $\text{SqErrors} \leftarrow \text{SqErrors} \cup \{e_i^2\}$
        $SqError \leftarrow \text{SqError} + w_i e_i^2$
    $\text{MAD} = \sqrt{\text{median}\{SqErrors\}}$
    $\text{s} = 1.4826 \times \text{MAD}$
    $\text{NormSqErrors} \leftarrow \frac{\text{SqErrors}}{\text{s}^2}$
    $\text{W}_{\text{temp}} \leftarrow \{\text{w}_\text{i} = \rho'(\text{SqErrors(i)}) | \text{i} \in \mathcal{F}_1 \cap \mathcal{F}_2\}$
    If $(|\text{PreviousSqError} - \text{SqSerror}| < \varepsilon_2)$:
        $\text{Found} \leftarrow \text{True}$
    Else:
        If $(\text{minError} > \text{SqError})$:
            $\text{minError} \leftarrow \text{SqError}$
            $\mu \leftarrow \mu_{\text{temp}}$
            $\xi \leftarrow \Omega_{\text{temp}} \mu_{\text{temp}}$
            $\tau \leftarrow \tau/10$
            $\text{W} \leftarrow \text{W}_{\text{temp}}$
            $\text{Found} \leftarrow (\text{minError} < \varepsilon_1)$
        Else:
            $\tau \leftarrow 10\tau$
```

8.3 Visual SLAM using only a camera

As reported in Chapter 1, a largely successful solution for monocular SLAM is Georg Klein and David Murray’s PTAM (Klein and Murray 2007, Klein and Murray 2009). PTAM has proved that it is possible to obtain reliable real-time scene reconstruction and camera pose at the same time. In general terms, PTAM follows the standard monocular SLAM pipeline (see Chapter 5, section 3), which is synopsized in the following steps:

i) *2-View scene reconstruction and reprojection error refinement.*
ii) Tracking and recovery of camera pose using a PnP algorithm.
iii) Bundle adjustment.
iv) New feature detection and triangulation.
v) Repeat from step (ii).

Evidently, the estimates obtained in steps *(i)*, *(ii)* and *(iii)* in the above pipeline will be decisive for the progress of the SLAM algorithm. It is therefore important to obtain a reliable scene reconstruction from the first two views of the sequence, in order to build a map that will subsequently yield reliable pose estimates in the following frames through the PnP algorithm. Two-view scene reconstruction was discussed in Chapter 3, sections 2.5-6. Nister's algorithm (used in PTAM) is so far the only method that delivers a "true" essential matrix, but it only works on 5 points; however, it could be used in the context of RANSAC in the overdetermined case. Note here that the RANSAC version of the 8-point algorithm (OpenCV) has provided satisfactory results in the current research.

8.3.1 Map management and pose estimation

Maintaining a map is vital for camera-only SLAM, because it acts as the "conduit" that couples point projections in previous views with their correspondences in the current view in order to estimate the new camera pose through the PnP algorithm. The first reconstruction is obtained from the essential matrix (Algorithm 3.1) and it is refined with two-view robust bundle adjustment (Algorithm 8.1). New points are detected in every frame after initialization, depending on a threshold on the number of visible map points. For each detected point, a new map point is created but marked as *uninitialized* to prevent the PnP algorithm from implicating it in subsequent computations for pose estimation. Once the new pose estimate is obtained, it is thereafter refined through bundle adjustment over the past 2 or 3 views. Finally, following pose refinement, the uninitialized features are triangulated and marked as *initialized* in the map (so that they can be used for pose estimation in the next execution of the PnP algorithm).

Further to the rejection of measurements that are not consistent with epipolar geometry (RANSAC outliers), there are two more outlier screening rules: The first rule simply disables map points that were assigned a weight less than 0.3 by the M-estimator during the last iteration of bundle adjustment. In the author's experience, if this rule is triggered more than 10 times, then the map has most likely reached critical levels of corruption. The second rule disables map points that have a depth above 60% of the depth's MAD. As will be discussed in the following sections, this rule can help in scenes that have bounded depth and minimal variation; however, in natural sequences with very large depth variance, this rule is doing more harm than good, because it is not adaptive to the distribution of depth.

8.3.2 Sequences from estuarine and forestall areas: Results

If the map contains distant points, then this will result in very noisy pose estimates and the SLAM process will have to reset or abort at a very early stage. Usually, distant points tend to act as "inductors of uncertainty" because their large depth will augment small tracking errors and the uncertainty region in 3D space becomes arbitrarily large (see Figure 3.16 in Chapter 3, section 4 for an intuitive illustration). In natural environments, this can be a huge problem as scene depth varies significantly from a few meters to several hundred meters. Such a case is demonstrated in the scene reconstructions of Figures 8.3 and 8.5 from a park sequence in Yelverton, Devon, in which, although certain points lie in relatively short distances away from the camera, the majority of the features is found on trees situated at the far end of the park. These distant features will back-project to very unstable world points which, in turn, will affect the subsequent pose estimates.

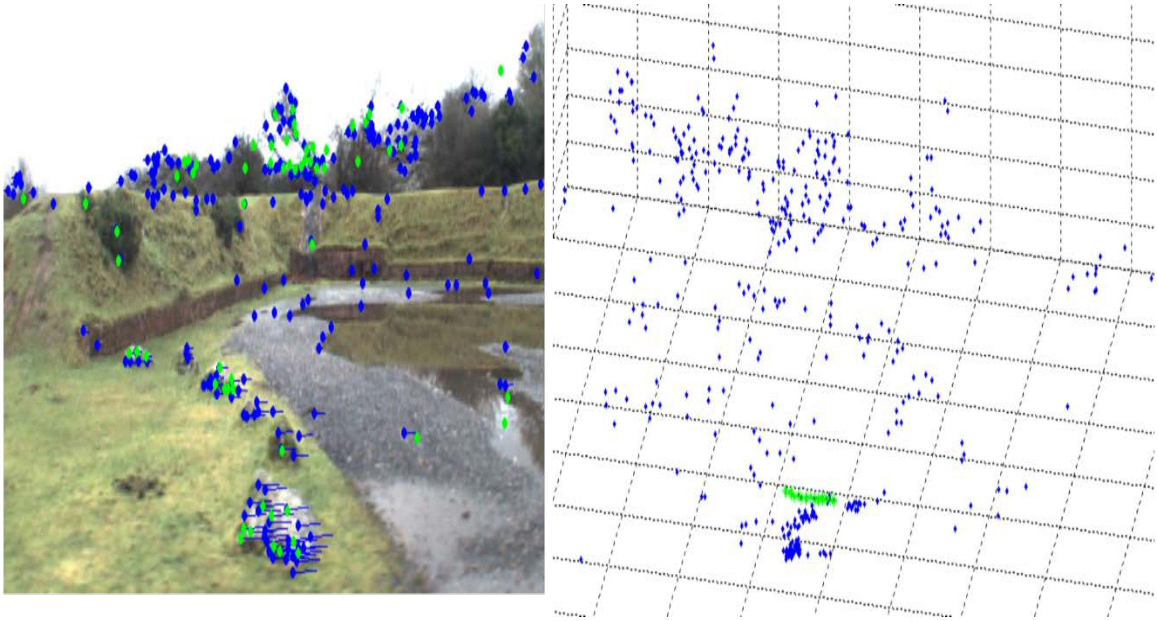

Figure 8.3. Scene reconstruction (right) from the initial flow field (left) of a sequence in a park at Yelverton, Devon, UK.

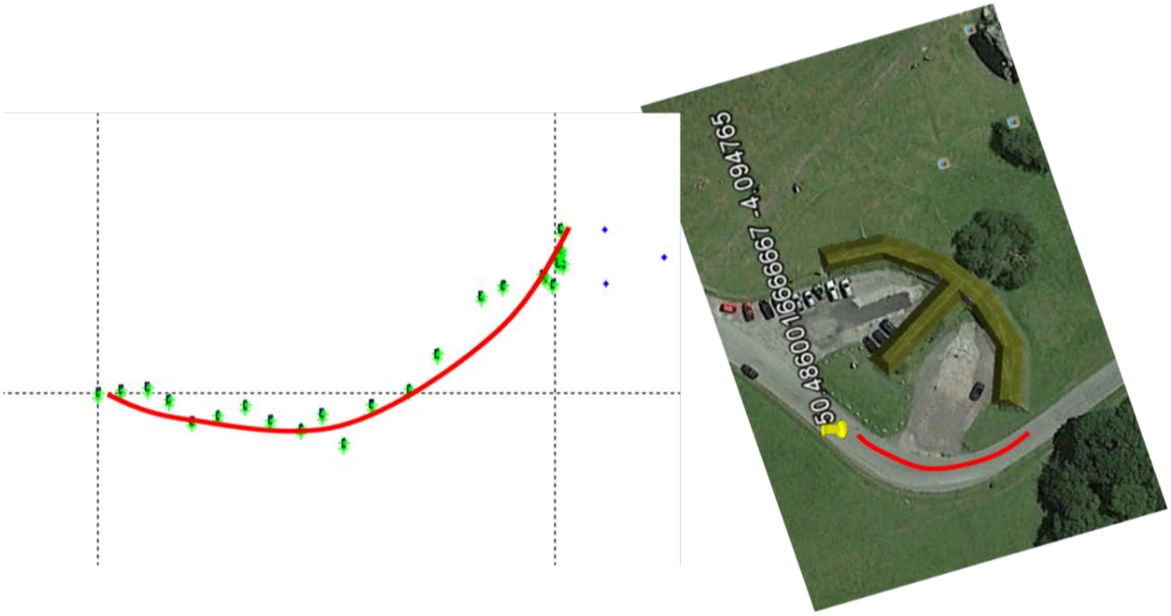

Figure 8.4. A close-up into camera poses in the first 27 frames of the park sequence. The red line indicates a GPS based ground truth trend (spline interpolation of GPS locations). The SLAM algorithm begins to become unstable roughly on the 21st frame in the sequence.

Figure 8.4 shows a close-up of the camera odometry for the first 27 frames against a GPS based spline segment. Evidently, the pose estimates after the 21st frame begin to approximate a single point instead of roughly following the trend indicated by the red spline, suggesting that the input to the PnP algorithm is very noisy. This input is comprised of the map and the measurements (tracked features); and although tracking "picks-up" a certain amount of drift (recall from section 3.1 in Chapter 5 in that OpenCV's LK tracker is used to track from the previous frame or the frame before last due to a limitation in its implementation), it becomes evident that the map is the main source of the problem with apparent detrimental effects in the pose estimate.

Figure 8.5 illustrates single camera SLAM during the first frames of a sequence from the vantage point of a moving boat in the river Tamar near Morwellham, Devon. This is a most difficult case as the majority of valid features lie in the distant background and the surface of the water cannot be used for tracking. Nevertheless, the reconstruction appears to be reasonably realistic. However, the subsequent camera pose estimates become very unstable after just 5 frames from the beginning of the sequence. This clearly indicates that not only the tracking is noisy, but also that the map is not very reliable. This becomes evident in the magnified view of Figure 8.6 in the reconstruction plot. Odometry becomes unstable very soon after SLAM initialization and, although bundle adjustment reduces the error, the pose estimate does not improve, which suggests that the optimization was initialized from a very bad estimate.

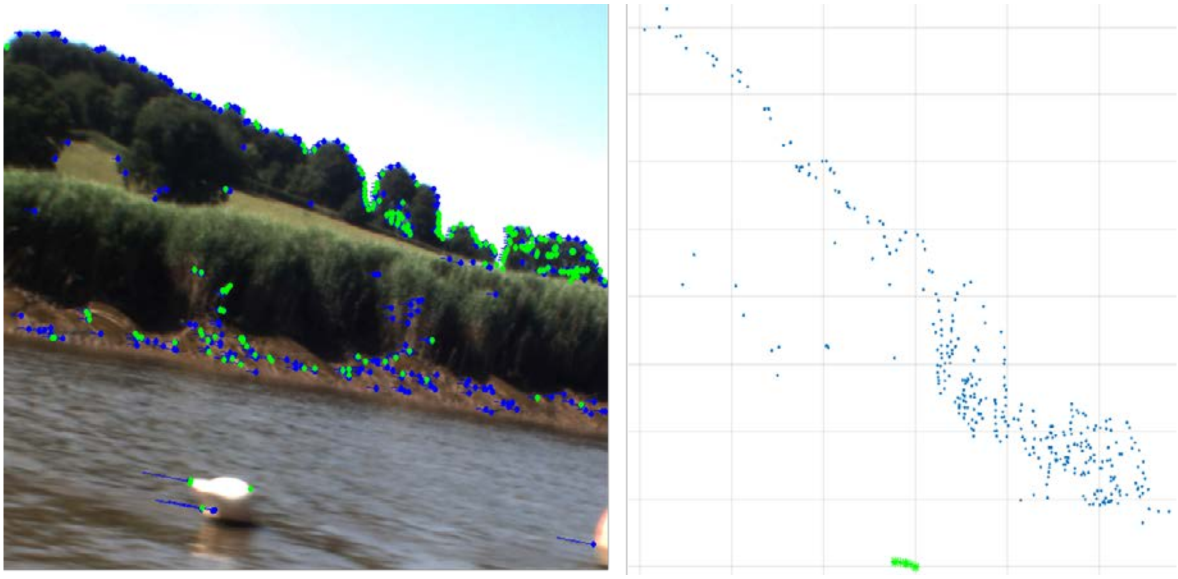

Figure 8.5. Scene reconstruction (right) from the initial flow field (left) of a sequence in the Tamar river near MorwellHam, Devon.

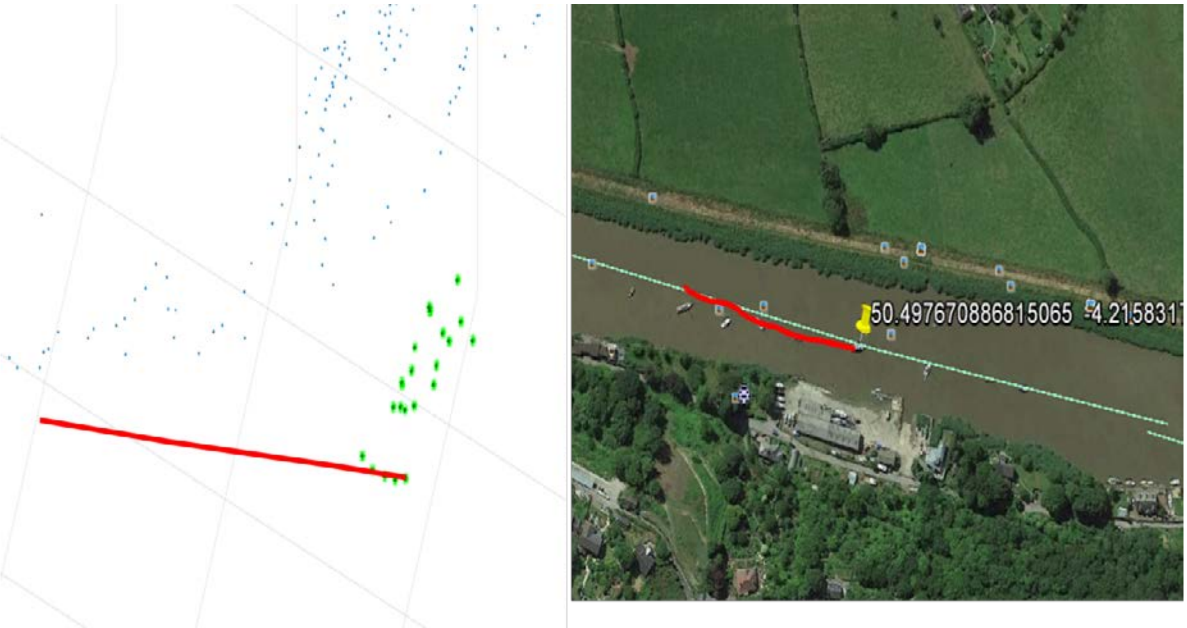

Figure 8.6. A close-up into camera poses in the first 21 frames of the Morwellham sequence in the Tamar river. The red line indicates a GPS based ground truth trend (spline interpolation of GPS locations). The estimated pose becomes unstable very early (just after the 6th frame).

Interestingly, a slightly more "agile" behavior of the classic, map-based monocular SLAM approach was observed in an estuarine sequence where scene depth presents very little variance compared to the Morwellham sequence (Figures 8.7-8). In this case (a sequence from the Tamar river near Calstock), the features have approximately same depth, lying at about 1-3 meters away from the camera; on the other hand however, from this distance, the attitude (rolling and pitching) and speed of the boat (roughly 3-4 knots) cause significant impact in the stability of flow estimation. Thus, in this sequence, noise originates primarily in tracking, but it accumulates in both map and pose in exactly the same way it does in the cases of the Yelverton park and Morwellham sequences. The results are slightly better than the ones observed in scenes with greater depth variation; however, the tradeoff here is that the tracker cannot fully compensate for the abrupt changes in attitude and therefore certain measurements are noisy, thereby affecting the results of the PnP pose estimates. Figure 8.7 illustrates initial reconstruction and flow, while Figure 8.8 shows a close-up on camera odometry with respect to the GPS based ground truth trend (a spline segment fitted on GPS readings).

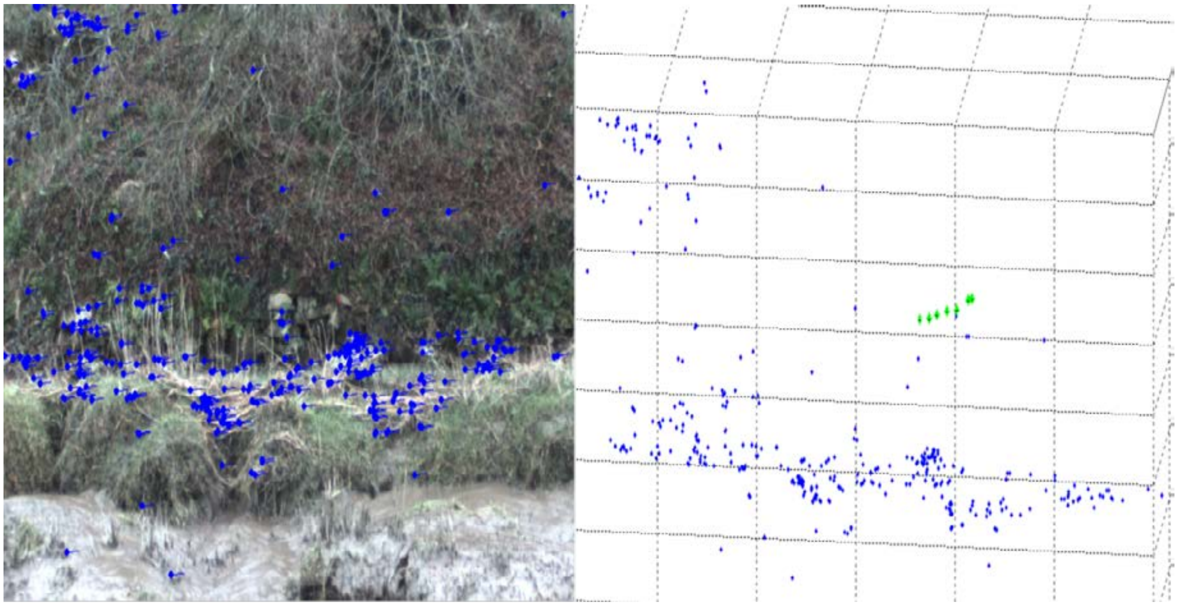

Figure 8.7. Scene reconstruction (right) from the initial flow field (left) of a sequence in the Tamar river near Calstock, Devon.

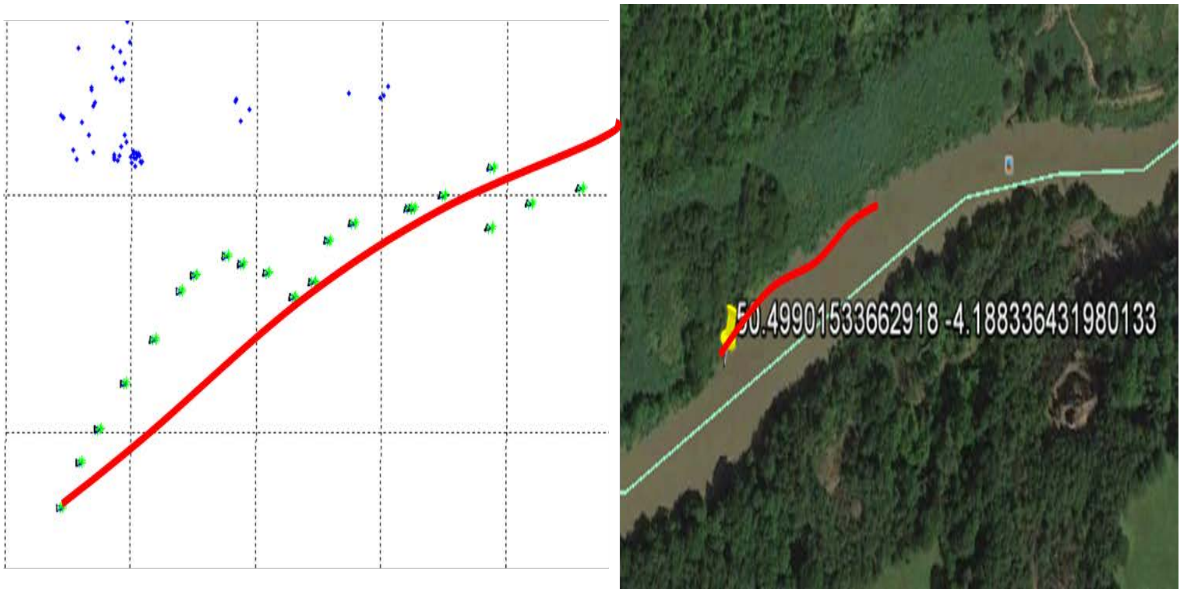

Figure 8.8. A close-up into camera poses in the first 21 frames of the Calstock sequence in the Tamar river. The red line indicates a GPS based ground truth trend (spline interpolation of GPS locations). The estimated pose begins to become unstable after 30 frames.

8.3.3 Noise induction through the map in visual SLAM: A “chicken and egg” problem

The problem of monocular SLAM in the context of the restrictions in this research is very ill-posed. In loose terms, one can say that the degrees of freedom marginally outnumber the inputs. Specifically, the environment is very unconstrained in terms of depth, brightness and texture, there are no prior models for camera motion and no special sensors can be used, except gyroscopes. Particularly in the no-gyroscope case examined in the current section, the only process input was the tracked feature locations.

The map is the result of an initial reconstruction between the first two camera views and it is thereafter populated with new triangulated features as the sequence progresses. Since epipolar geometry is the result of matched features, it follows that the map is estimated exclusively on the tracked features. Even if these measurements bear very little noise, it is clear that, in scenes with large variation in depth, these small amounts of noise can occasionally produce largely erroneous 3D points, some of which will eventually be used in subsequent pose estimation, which, in turn, will encapsulate the exaggerated error of previous measurements. This is a process very reminiscent of electrical induction wherein tiny amounts of noise are circulated in a loop through the map and the latest camera pose and eventually result in very uncertain estimates. Of course, this loop can be broken with more sensory data and a motion prior, none of which are available (with the exception of gyroscope) in the case of this research.

Partial remedies to the noise induction problem described above would involve the selection of less uncertain points that produce small reprojection errors; this classification can be based, for instance, on the M-estimator weights. Of course, when measurements are not reliable, the PnP based camera pose estimate will be inaccurate, thereby leading again to large reprojection errors, even for map points that performed well so far; this is a dilemma of the type, “whom to believe”, the camera pose, or the map? Again, solutions exist, which would of course involve resetting the map and initiating a new set of features, but the overall approach begins to grow unnecessarily complicated.

It is the author’s belief that a more drastic remedy to the problem of noise induction through the map would be to eliminate the map from the pose estimation process without necessarily eliminating mapping itself. Circumventing the map from relative pose estimation clearly suggests that the solution must be obtained directly from the correspondences. In the case of general motion, this can only be done through the essential matrix. The problem with this approach is that, as discussed in Chapter 3, section 2.6, to this day, there is no reproducible (in the sense of ease of implementation and fast execution times) algorithm to solve for the essential matrix in an overdetermined setup, except of course for the usual unreliable, Procrustean approaches. Nister’s 5-point algorithm can be extremely useful in a RANSAC framework and further research in that direction should be a priority.

8.4 Visual SLAM without the map

Suppose that, in addition to the conditions of the monocular SLAM scenario of the previous section, gyroscope readings from an IMU are available throughout the entire duration of motion. Angular rate inputs are generally robust and suffer from minor drift and therefore the respective relative orientation estimates are very accurate and can be used to rectify the projections as discussed in Chapter 3, section 3.1. Following rectification, the sequence can be treated as the result of purely translational camera motion. In the purely translational motion case, relative pose can be recovered directly from the correspondences as described in Chapter 6, section 4.1 using the following cost function:

$$J = b^T \underbrace{\left[\sum_i \left(\frac{m_0^{(i)} \left(m_0^{(i)} \right)^T C_i}{\left(m_0^{(i)} \right)^T C_i m_0^{(i)}} - I_3 \right)^T C_i \left(\frac{m_0^{(i)} \left(m_0^{(i)} \right)^T C_i}{\left(m_0^{(i)} \right)^T C_i m_0^{(i)}} - I_3 \right) \right]}_{Q} b^T = b^T Q b \tag{8.39}$$

where $\left(m_0^{(i)}, m_1^{(i)} \right)$ is the i^{th} pair of normalized Euclidean projections (subscripts *0* and *1* denote previous and current pose), b is the baseline and C_i is a positive semidefinite matrix depending exclusively on the projections in the second (most recent) view:

$$C_i = \left(m_1^{(i)} 1_z^T - I_3 \right)^T \left(m_1^{(i)} 1_z^T - I_3 \right) \tag{8.40}$$

where $1_z = [0 \quad 0 \quad 1]^T$. Please note that equation (8.39) is just a formal way of introducing the problem, but it is not numerically stable in its current form; when implementing the least squares solver, it is advised to (at least) avoid having the term $\left(m_0^{(i)} \right)^T C_i m_0^{(i)}$ in the denominator, as it is likely to vanish in cases of low disparity. In the numerically stable formulation, features with very low disparity will simply be ignored by the relative pose solver because they contribute a trivial equation ($0b = 0$) in the overdetermined system.

What is worth noting about the formulation of equation (8.39), besides the obvious absence of rotation matrix, is that now there is no need to solve for relative pose through the map, as it can be done directly from correspondences. This way, tracking noise does not become augmented in distant map points and therefore the impact on the new pose is minimal to none at all. Also, the features that correspond to very uncertain 3D points can be isolated during bundle adjustment and can either be removed or simply be assigned a small weight by the M-estimator.

8.4.1 Map-less SLAM: Probabilistic approach and intuition

If relative orientation is known, then the computation of the essential matrix reduces to a simple least squares problem and a very reliable scene reconstruction can be obtained, provided some good quality tracking in the second view. Recall that in section 2.3 of this chapter, depth based parametrization of 3D points was introduced as follows:

$$M = Z_h R_h h + b_h \tag{8.41}$$

where $\mathrm{Z_h} > 0$ is the depth of the point in its "home" view (i.e., the frame in which it was originally detected), R_h is the rotation matrix containing the camera frame (expressed in the in world frame) arranged column-wise and b_h is the position of the home camera center in the world. Suppose that g is the rectified projection (using the relative orientation matrix estimated from gyroscope input) of the feature in the frame that followed the home frame. This camera view is dubbed as "base". From equation (3.31) the depth Z_h in the home view is given by:

$$Z_h = \frac{h^T C(b_s - b_h)}{h^T Ch} = \frac{h^T C b_{hs}}{h^T Ch} \qquad (8.42)$$

where $b_{hs} = b_s - b_h$ is the baseline (in world coordinates) between the camera centers in the base and the home view and C is the following non-invertible positive semidefinite matrix:

$$C = (g1_z^T - I_3)^T (g1_z^T - I_3) \qquad (8.43)$$

From (8.42) and (8.41) the world point M can now be expressed in terms of the 12 parameters corresponding to the pose of the camera in the home and base view:

$$M = \frac{h^T C b_{hs}}{h^T Ch} R_h h + b_h \qquad (8.44)$$

It is important to stress that equation (8.44) implies that the only stochastic quantities in M are the home and base view pose variables. This means that, regardless of number of points, the map formally becomes dependent only on pose variables and therefore the respective degrees of freedom become practically constant, as opposed to the usual formulations in visual SLAM (for instance, the one introduced in section 2 of this chapter) in which the state vector has polynomial scaling in the number of map points. An alternative way of parametrizing M in terms of the camera pose in the home and base views is to use a triangulation method (see section 2.1 in Chapter 3), which would most likely be slightly more accurate, but would also incur additional computational burden in terms of the derivatives. In contrast, the Jacobian of the expression in equation (8.44) can be computed in constant time.

In probabilistic terms, one may regard the parametrization of equation (8.44) as a marginal of the SLAM posterior over M. Consider the Bayes network of Figure 8.9

illustrating the parametrization of M as a conditional distribution in the circled region. Since h and g are always instantiated variables, it follows that the distribution of M is a conditional distribution on the home and base poses x_h and x_g. Note that for the sake of simplicity, the network contains only one measurement variable; the general case is a straightforward generalization.

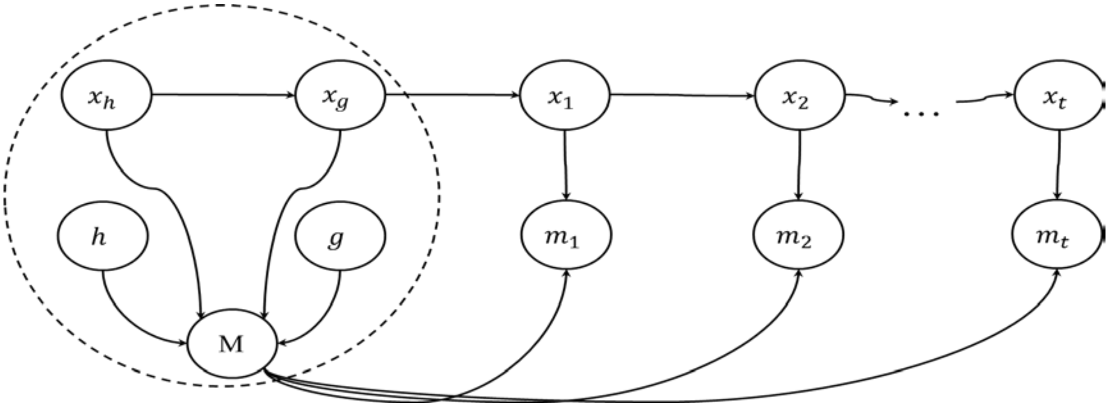

Figure 8.9. Map parametrization in visual SLAM. In the circled region, the map point M is conditionally dependent on camera pose (x_h, x_g) and measurements (h, g) in the home and base views.

Marginalizing M out of the SLAM posterior practically entails the (soft) substitution of each map point variable in the measurement likelihood with an expression that contains x_h and x_g. Thus, in practice, one obtains a new least squares formulation in which the only unknowns are the camera poses in the various views throughout the sequence. In SLAM terms, this means that a new state belief $Bel^*(x_{1:t}, x_h, x_g) = p(x_{1:t}, x_h, x_g | h, g, m_{1:t})$ is now obtained from the original belief, $Bel(x_{1:t}, x_h, x_g, M) = p(x_{1:t}, x_h, x_g, M | h, g, m_{1:t})$ as follows:

$$Bel^*(x_{1:t}, x_h, x_g) = \int Bel(x_{1:t}, x_h, x_g, M) dM \qquad (8.45)$$

where $x_{1:t} = \{x_1, \dots, x_t\}$.

Let $x_t = [\psi_t^T \quad b_t^T]^T$ where ψ_t in the orientation parameter vector and b_t is the camera position in the world. Also, let $p(m_t|M, x_t) = N(f(M, x_t), Q_t)$ be the measurement likelihood associated with the i^{th} feature at time $t \geq 1$ where function f is the projective mapping between the tracked feature position m_t and the respective map point:

$$m_t = \underbrace{\frac{1}{1_z^T R^T (M - b_t)} R^T (M - b_t)}_{f(M, x_t)} \tag{8.46}$$

where $1_z = [0 \quad 0 \quad 1]^T$ and $R_t = R(\psi_t)$ is the rotation matrix corresponding to the camera orientation parameters at time *t*.

As discussed in Appendix C, when M is marginalized out of the posterior, a new measurement likelihood $q(m_t | x_t, x_g, x_h) = N(f^*(h, g, x_t, x_g, x_h), Q^*{}_t)$ can be constructed from equation (8.46), in which the projection mapping is obtained by substituting M in (8.46) with the expression of equation (8.44):

$$m_t = \underbrace{\frac{1}{1_z^T R_t^T \left(\frac{h^T C b_{hs}}{h^T C h} R_h h + b_h - b_t \right)} R_t^T \left(\frac{h^T C b_{hs}}{h^T C h} R_h h + b_h - b_t \right)}_{f^*(h, g, x_t, x_g, x_h)} \tag{8.47}$$

For all intents and purposes, equation (8.47) describes a simple substitution in the least squares formulation of the SLAM problem. However, it should be stressed that, although *h* is non-stochastic (since it is the location of the feature's original detection), its corresponding location g in the base view is stochastic due to tracking uncertainty. This uncertainty should now be taken into consideration in this "map-less" measurement term. The covariance matrix of the new measurement likelihood can be obtained (approximately) by applying linear propagation of covariance in a linearized version of f^* in the neighborhood of the measured value of *g*:

$$Q_t^* = Q_t + \frac{\partial f^*}{\partial g} Q_g \left(\frac{\partial f^*}{\partial g} \right)^T \tag{8.8}$$

where $\frac{\partial f^*}{\partial g}$ is the Jacobian of f^* with respect to the measurement in the base view and Q_h is the covariance matrix of the prior distribution $p(g)$.

Please note here that it was observed during experimentation that the strict use of covariance matrices during filtering can quickly lead to over-confident state estimates which have almost "no regard" for the gradient (the product of the inverse of very large information matrices with the Jacobian yields very small changes in the Gauss-Newton method). This is most likely a consequence of the fact that the chosen variance of the measurement likelihood is typically not a realistic estimate of the true uncertainty entailed by the tracker. In practice, for the needs of bundle adjustment, the measurement constraints are weighted by some measure of patch discrepancy; however, the posterior is always assumed to have an identity covariance matrix upon initialization of the Levenberg-Marquardt iteration. In other words, variance is not carried over to the latest SLAM posterior in order to avoid over-confident estimates.

8.4.2 Adjusting the reprojection error without the map

Having marginalized-out the map, the measurement model of equation (8.47) will introduce constraints only over the current camera pose and the feature's home and base-view poses. This means that the information matrix will only contain pose variables and therefore is expected to be very compact in size. This means that information matrix inversions will take place in practically constant time. Furthermore, provided that the parametrization of equation (8.41) is employed, information matrix updates in a single step of the Gauss-Newton method will complete in linear time. This allows for the use of large numbers of features without a significant impact in optimization times.

The major downside of the map-less approach has to do with the fact that, for the sake of marginalizing the map out of the posterior, a great deal of "faith" is invested in the base-view measurements. In other words, it is expected that the base-view measurements are very accurate. Of course, this cannot be generally true and some of these measurements may introduce large errors which can easily skew the results of bundle adjustment. The natural workaround, which is employed in this research, is to discard these correspondences on the basis of M-estimator weights by means of simple thresholding (provided the previous rejection of outliers in the RANSAC based computation of the essential matrix). This is an effective approach and very easy to apply, since the removal of a map point is equivalent to removing a quadratic constraint without changing the joint distribution. Thus, one is actually able to remove a measurement during the execution of bundle adjustment (which is the case in Algorithm 8.2). Another strategy would be to rectify some of these correspondences by adopting the classic measurement model and optimizing the respective 3D locations. However, such a solution would require careful selection of erroneous correspondences by means of some robust statistical criterion that would add more design complexity to the overall solution.

Algorithm 8.2 describes iterative adjustment of the base pose after the initial scene reconstruction without using a map (first camera frame is world reference). The algorithm rejects measurements that correspond to an M-estimator weight below a user-defined threshold. Notice that now, the Fisher parameters are only 5-dimensional, regardless of number of correspondences.

Algorithm 8.2. ***Robust "map-less" bundle adjustment (for the initial 2-view reconstruction)***

***Input**: a) Indexing sets of visible features in the two views, $\mathcal{F}_1$ and $\mathcal{F}_2$.*

b) Measurements $h_i, i \in \mathcal{F}_1$ and $g_j, j \in \mathcal{F}_2$.

c) Measurement covariance matrices $Q_j, j \in \mathcal{F}_2$.

d) Relative camera pose (ψ, κ, λ) where ψ are orientation parameters and $\kappa, \lambda \in \mathbb{R}$ are the stereographic coordinates of the unit-norm baseline.

e) An M-estimator objective function ρ.

f) A weight threshold $t < 1$ for constraint rejection (map-points).

***Output:** Updated relative camera pose estimate $(\hat{\psi}, \hat{\kappa}, \hat{\lambda})$.*

```
comment        Number of measurements.
$n \leftarrow |\mathcal{F}_1 \cap \mathcal{F}_2|$
comment        Create the information matrix as the $5\times 5$ identity matrix.
$\Omega \leftarrow I_5$
comment        Assign current depth and pose estimates to a $(n + 6m) \times 1$ vector.
$\mu \leftarrow [\hat{\psi}\ \ \hat{\kappa}\ \ \hat{\lambda}]$
comment        Initial information vector is equal to the mean.
$\xi \leftarrow \mu$
comment        The LM constants associated with termination conditions.
$\varepsilon_1 \leftarrow 10^{-5}$;  $\varepsilon_2 \leftarrow 10^{-12}$; Timeout $\leftarrow 40$
comment        Working-out the squared errors and the M-estimator weights.
SqErrors $\leftarrow \emptyset$
SqError $\leftarrow 0$
$\hat{\psi} = [\mu(1)\quad \mu(2)\quad \mu(3)]$; $\hat{\kappa} = \mu(4)$; $\hat{\lambda} = \mu(5)$
$\hat{R}$ = rotation matrix$(\hat{\psi})$; $\hat{b}$ = baseline vector$(\hat{\kappa}, \hat{\lambda})$
For each i $\in \mathcal{F}_1 \cap \mathcal{F}_2$:
    $C_i = (g_i 1_z^T - I_3)^T (g_i 1_z^T - I_3)$
    $\widehat{M}_i \leftarrow \frac{h_i^T C_i \widehat{b}}{h_i^T C_i h_i} \widehat{R} h_i$
    $e_i^2 \leftarrow \left(g_i - \frac{\widehat{R}^T(\widehat{M}_i - \widehat{b})}{1_z^T \widehat{R}^T(\widehat{M}_i - \widehat{b})}\right)^T Q_{g_i}^{-1} \left(g_i - \frac{\widehat{R}^T(\widehat{M}_i - \widehat{b})}{1_z^T \widehat{R}^T(\widehat{M}_i - \widehat{b})}\right)$
    $\mathrm{SqErrors} \leftarrow \mathrm{SqErrors} \cup \{e_i^2\}$
    $\mathrm{SqError} \leftarrow \mathrm{SqError} + w_i e_i^2$
$\mathrm{MAD} = \sqrt{\mathrm{median}\{\mathrm{SqErrors}\}}$
$s = 1.4826 \times \mathrm{MAD}.$
$\mathrm{NormSqErrors} \leftarrow \frac{\mathrm{SqErrors}}{s^2}$
$W \leftarrow \{w_i = \rho'(e_i^2) | i \in \mathcal{F}_1 \cap \mathcal{F}_2\}$
***comment***      *Entering the LM main loop*
$\mathrm{minError} \leftarrow \mathrm{SqError}$; $\mathrm{Found} \leftarrow (\mathrm{SqError} \leq \varepsilon_1)$
$\tau \leftarrow 10^{-3}$; $k \leftarrow 0$
$BadConstraints \leftarrow \emptyset$
**While** (**Not** Found **And** $k \leq$ Timeout):
    $\mathrm{PreviousSqError} \leftarrow \mathrm{SqError}$
    $k \leftarrow k + 1$
    $\Omega_{temp} \leftarrow \Omega$; $\xi_{temp} \leftarrow \xi$ ; $[\widehat{\psi}\ \ \widehat{\kappa}\ \ \widehat{\lambda}] \leftarrow \mu$
    $\widehat{R} = \text{rotation matrix}(\widehat{\psi})$; $\widehat{b} = \text{baseline vector}(\widehat{\kappa}, \widehat{\lambda})$
    **For each** $i \in (\mathcal{F}_1 \cap \mathcal{F}_2) - \mathrm{BadConstraints}$:
        ***comment***      *Use the constraint only if the previous weight is above threshold.*
        **If** $(W(i) \leq t)$:
            ***comment***      *Update the Fisher parameters with linearized constraints*
            $G_i = \left[ \left.\frac{\partial\left(\frac{R^T(M_i - b)}{1_z^T R^T(M_i - b)}\right)}{\partial\psi}\right|_{\widehat{\psi}} \quad \left.\frac{\partial\left(\frac{R^T(M_i - b)}{1_z^T R^T(M_i - b)}\right)}{\partial\kappa}\right|_{\widehat{\kappa}} \quad \left.\frac{\partial\left(\frac{R^T(M_i - b)}{1_z^T R^T(M_i - b)}\right)}{\partial\lambda}\right|_{\widehat{\lambda}} \right]$
            $w_i = W(i)$
            $\Omega_{temp} \leftarrow \Omega_{temp} + w_i G_i^T Q_{g_i}^{-1} G_i$
        $\xi_{\text{temp}} \leftarrow \xi_{\text{temp}} + w_i G_i^T Q_{g_i}^{-1} \left( g_i + G_i \begin{bmatrix} \hat{\psi} \\ \hat{\kappa} \\ \hat{\lambda} \end{bmatrix} - \frac{\hat{R}^T(\hat{M}_i - \hat{b})}{1_z^T \hat{R}^T(\hat{M}_i - \hat{b})} \right)$

    **comment**    *Mark the constraint as "bad". It will not be used next time.*

    **Else:**

        $\text{BadConstraints} \leftarrow \text{BadConstraints} \cup \{i\}$

**comment**    Obtaining a new joint estimate

$\Omega_{\text{temp}} \leftarrow \Omega_{\text{temp}} + \tau I$

$\xi_{\text{temp}} \leftarrow \xi_{\text{temp}} + \tau\mu$

$\mu_{\text{temp}} \leftarrow \Omega_{\text{temp}}^{-1} \xi_{\text{temp}}$

**comment**    *Working-out the squared errors and the M-estimator weights.*

$\text{SqErrors} \leftarrow \emptyset$
$\text{SqError} \leftarrow 0$

**For each** $i \in \mathcal{F}_1 \cap \mathcal{F}_2$:

    $[\hat{\psi} \;\; \hat{\kappa} \;\; \hat{\lambda}] \leftarrow \mu_{\text{temp}}$

    $\hat{R} = \text{rotation matrix}(\hat{\psi}); \hat{b} = \text{baseline vector}(\hat{\kappa}, \hat{\lambda})$

    **comment**    *Compute the 3D point in the first camera frame.*

    $C_i = (g_i 1_z^T - I_3)^T (g_i 1_z^T - I_3)$

    $\hat{M}_i \leftarrow \frac{h_i^T C_i \hat{b}}{h_i^T C_i h_i} \hat{R} h_i$

    $w_i = W(i)$

    $e_i^2 \leftarrow \left( g_i - \frac{\hat{R}^T(\hat{M}_i - \hat{b})}{1_z^T \hat{R}^T(\hat{M}_i - \hat{b})} \right)^T Q_{g_i}^{-1} \left( g_i - \frac{\hat{R}^T(\hat{M}_i - \hat{b})}{1_z^T \hat{R}^T(\hat{M}_i - \hat{b})} \right)$

    $\text{SqErrors} \leftarrow \text{SqErrors} \cup \{e_i^2\}$

    $\text{SqError} \leftarrow \text{SqError} + w_i e_i^2$

    $\text{MAD} = \sqrt{\text{median}\{\text{SqErrors}\}}$

    $s = 1.4826 \times \text{MAD}$

    $\text{NormSqErrors} \leftarrow \frac{\text{SqErrors}}{s^2}$

$W_{\text{temp}} \leftarrow \{ w_i = \rho'(\text{SqErrors}(i)) | i \in \mathcal{F}_1 \cap \mathcal{F}_2 \}$

**If** $(|\text{PreviousSqError} - \text{SqSerror}| < \varepsilon_2)$:

    $\text{Found} \leftarrow \text{True}$

**Else:**
```

```
If (minError > SqError):
        $\text{minError} \leftarrow \text{SqError}$
        $\mu \leftarrow \mu_{\text{temp}}$
        $\xi \leftarrow \Omega_{\text{temp}}\mu_{\text{temp}}$
        $\tau \leftarrow \tau/10$
        $W \leftarrow W_{\text{temp}}$
        $\text{Found} \leftarrow (\text{minError} < \varepsilon_1)$
Else:
        $\tau \leftarrow 10\tau$
```

8.4.3 The disjoint scene approach for visual SLAM with input from a gyroscope

In the camera-only SLAM framework (see introduction of section 3), it is absolutely necessary to detect new features as densely as possible in order to enhance the stability of the pose estimate. Although the same approach could be used in the case of map-less visual SLAM, the implementations of map-less algorithms in this thesis adopt the disjoint SLAM paradigm (Chapter 5, section 2). In disjoint scene SLAM, the detection of new features is triggered only when the existing active pool of features has dropped below a threshold; furthermore, once the new set is detected, the remaining active features in the previous batch become inactive. In other words, a new scene is initiated which has no features in common with the previous. One can argue that this approach is not so robust compared to the classic, overlapping scene SLAM. On the other hand however, the average scene typically spans 4-5 frames on average, suggesting tight sampling of new features. The pros of this method have to do with execution time (feature detections typically require several milliseconds in the best case) and ease of implementation because the map data structure becomes less complex. The disjoint scene SLAM approach can be synopsized in the following steps:

i) *Feature detection in the home (previous) view and tracking in the base (current) view.*

ii) *Recovery of relative pose in the home and base view using Algorithm 3.1.*

iii) *Tracking and recovery of camera pose directly from the correspondences (as described in the introduction of section 4 in this chapter).*

iv) *Bundle adjustment without a map (Algorithm 8.2).*

v) *If number of visible inliers remains above a threshold, go to step (iii).*

vi) *Repeat from step (i).*

Although a simplification, as will be shown in results, the disjoint SLAM approach worked well in practice for the gyroscope-aided visual SLAM implementations discussed in this section. On the other hand however, adoption of the classic overlapping scene paradigm is very likely to improve performance not only in terms of minimizing drift in the pose estimate, but also in terms of the quality of reconstruction and therefore should be included in future revisions.

8.4.4 Map-less visual SLAM in natural landscapes: Results

The gyroscope-aided, map-less visual SLAM framework was tested not only on the sequences in section 3.2 of this chapter, but on several other estuarine sequences obtained from the Tamar river in Devon. The distances covered range from 40m to 1.5 km. Figures 8.10-18 illustrate the recovered odometries and 3D respective reconstructions; also, comparisons against GPS based approximate ground truth are overlaid on satellite imagery.

Comparisons against approximate ground truth were created by scaling the GPS by the speed over ground at the origin of motion and aligning it with the recovered odometry. The units of the 3D plots in Figures 8.10-18 are meters; however, a scale discrepancy of approximately 0.36 (the actual trajectory generally has larger scale) should be taken into account. It should be noted that the recovered odometries in satellite images appear as orthographic projections on the *x*-*z* (ground) plane. Also, distant points (beyond 300 m) from the camera upon original detection are discarded for display purposes. Usually, the IMU *x*-*z* plane is not aligned with the ground and this misalignment. This suggests that a displacement of several hundreds of meters in the *x*-*z* plane entails a similar shift in the *y* axis for an angle of just 10-20^o between the original IMU *x*-*z* plane and the ground.

Figures 8.10-8.13 illustrate the recovered odometry from four sequences recorded in forestall areas of Devon from a moving van. It is worth noting that the depth varies significantly from a few meters all the way to 1000 meters; however, due to the fact that these sequences were shot on land, the SLAM algorithm can reliably track ground features which have non-degenerate disparity that is representative of camera motion, while very distant features are simply ignored by the pose estimation algorithm. It follows that many of these features actually make it to the map (as very distant points), but do not really contribute in the pose estimate.

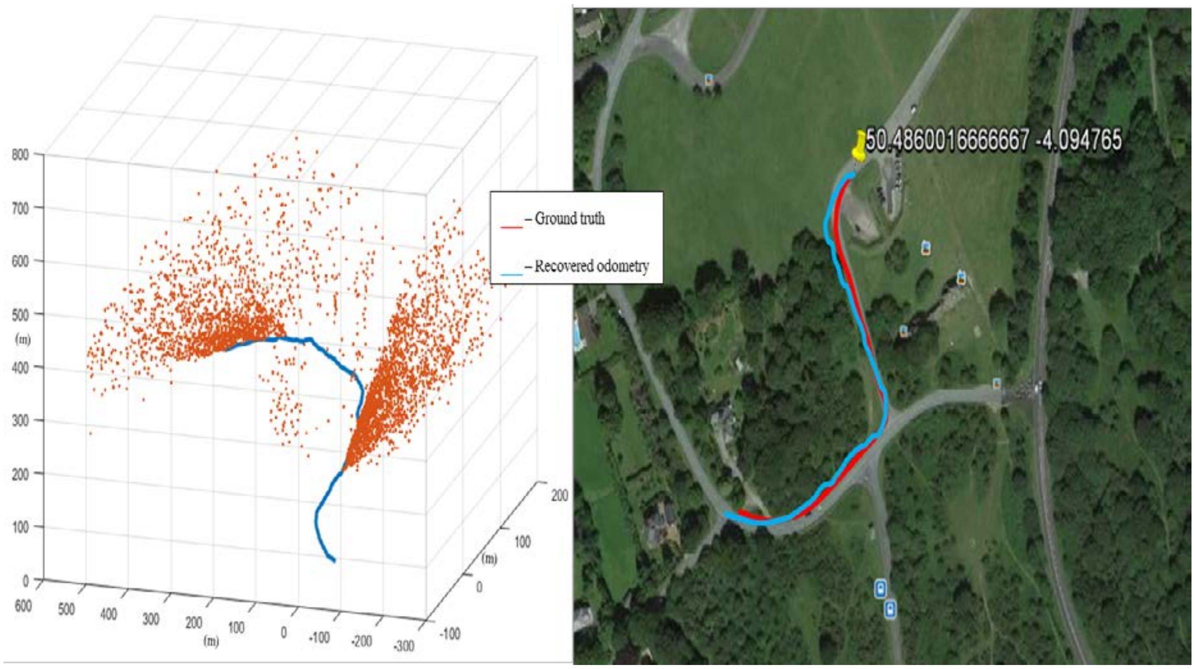

Figure 8.10. Recovered odometry and map (left) for the 1.2 km park sequence in Axtown, Devon[21]. Approximate ground truth (GPS based spline) shown in red on the right.

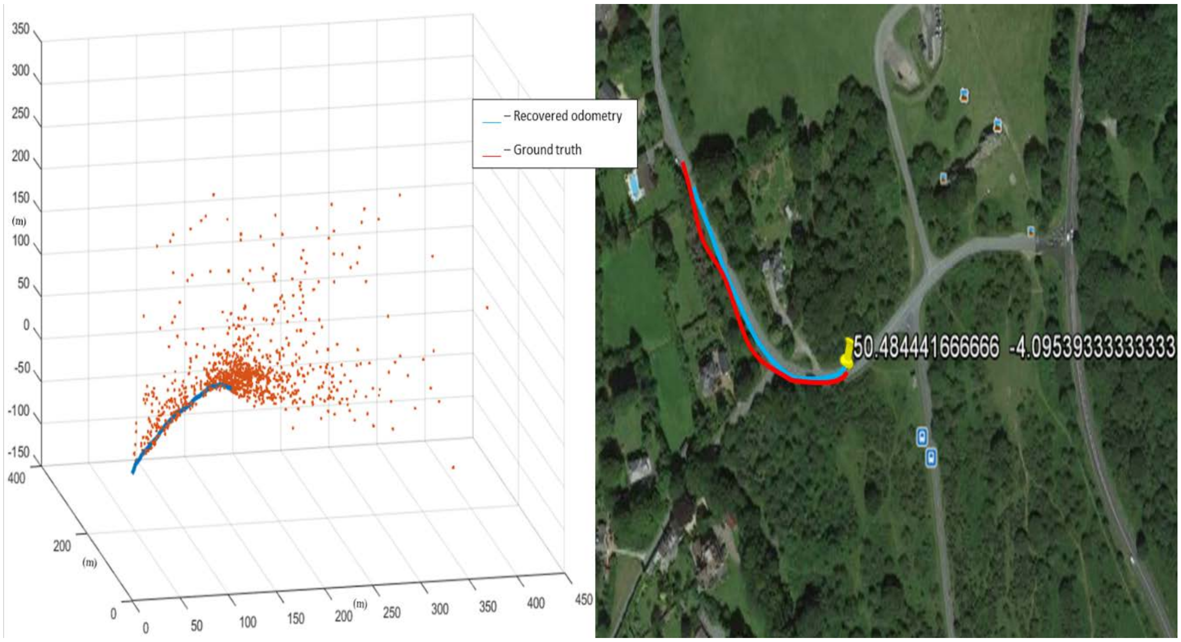

Figure 8.11. Recovered odometry and map (left) for a 0.9 km long forest route near Axtown, Devon[22]. Approximate ground truth (GPS based spline) shown in red on the right.

[21] Demo video available at: https://youtu.be/A1UCqTG_RqQ

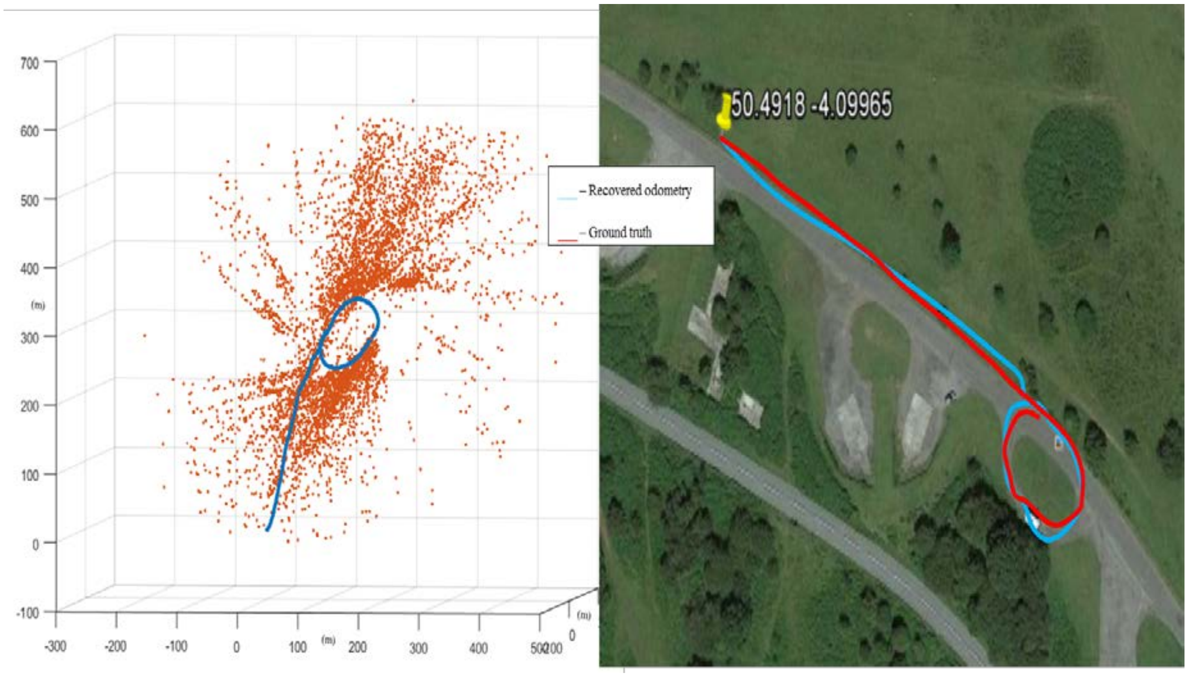

Figure 8.12. Recovered odometry and map (left) for a 0.8 km long route in a forestall area between Crapstone and Axtown, Devon[23]. Approximate ground truth (GPS based spline) shown in red on the right.

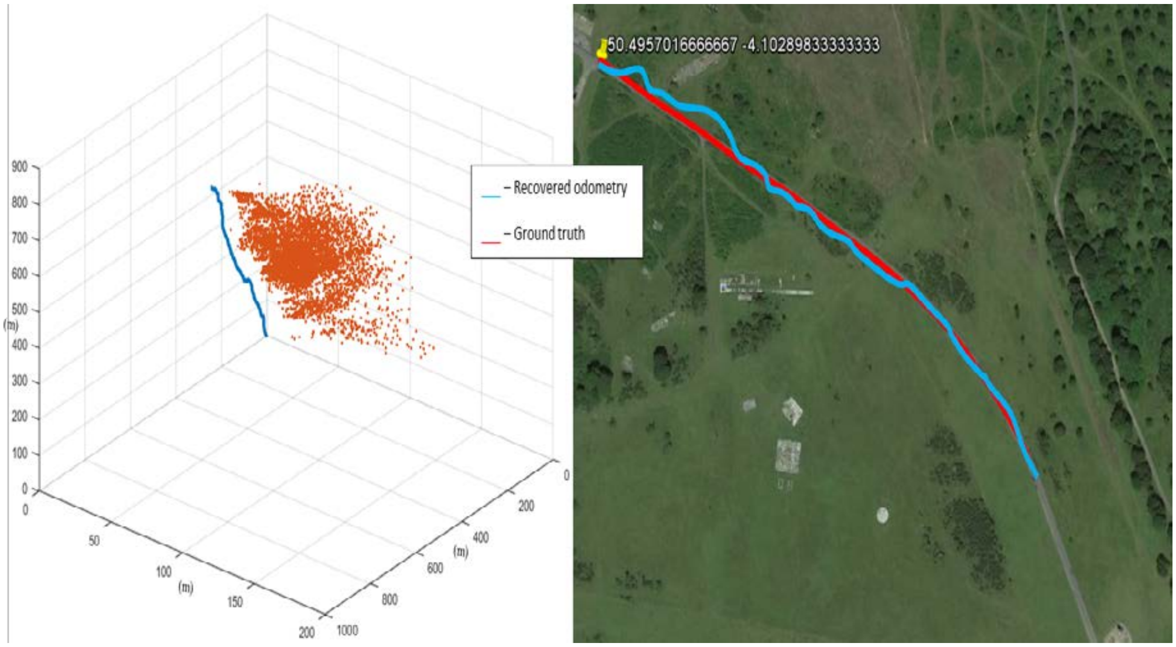

Figure 8.13. Recovered odometry and map (left) for a 0.6 km long route in fields between Crapstone and Yelverton in Devon[24]. Approximate ground truth (GPS based spline) shown in red on the right.

[22] Demo video available at: https://youtu.be/uXdVmgukWTc
[23] Demo video available at: https://youtu.be/mFeIY4vULbg
[24] Demo video available at: https://www.youtube.com/watch?v=_5pg1Bp1Z8E

Sequences in parks and forests shot from a moving van differ significantly from sequences taken from a cruising boat. Main reason being, the sea bed typically occupies a large portion of the image thereby rendering tracking in this region useless; although a number of features are detected on the surface of the water, they generally get rejected soon afterwards by the RANSAC based algorithm for essential matrix computation. Thus, in practice, the features with non-degenerate disparity are significantly fewer than the ones in in the cases of Figures 8.10-13. This problem can be dealt with by increasing the numbers of features, as they do not significantly affect execution time owed to the circumvention of the map from the pose estimation algorithm. Another significant difference between land and water surface sequences is that the motion of the boat very often includes significant pitching and rolling, as opposed to the approximately single-axis translational motion of the van. Figures 8.14-18 illustrate recovered odometries and scene reconstructions in maps from sequences of estuarine natural sceneries taken from boat cruising the Tamar river in Devon.

It should again be stressed that, although the distance units in the plots are meters, an approximate scale discrepancy of 0.36 (actual trajectory is larger) should be taken into consideration when viewing the odometry-map plots, since the initializing baseline length is always taken to be 1 by the SLAM algorithm, as opposed to the true length which usually ranges from 2m to 2.5m (according to values for speed over ground reported by the GPS). Also, it is worth commenting on the fact that every odometry plot exhibits motion that in the long-term appears planar, yet the motion plane itself has an inclination in terms of the world *x*-*z* plane; this inclination is caused by the misalignment of the IMU frame with the motion plane at the origin and as a result, the plots give the impression that the vehicle is gradually “gaining altitude”. For the same reason, distant map points appear to be very distant along the *y*-axis.

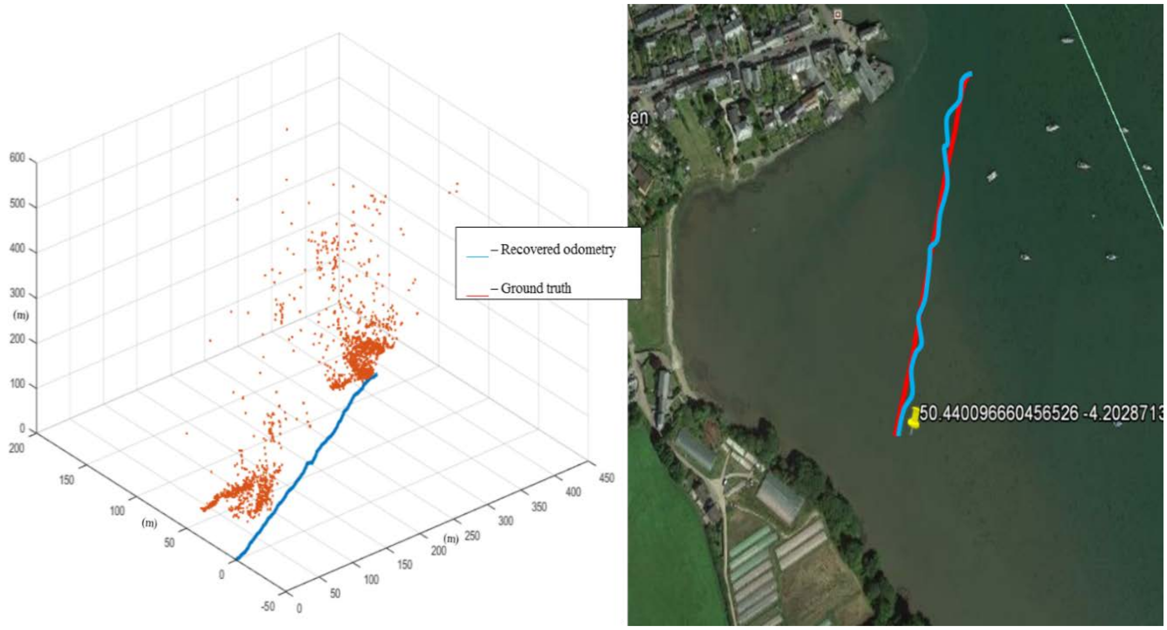

Figure 8.14. Recovered odometry and map (left) for a boat cruise near Cargreen, Devon[25]. Approximate ground truth (GPS based spline) shown in red on the right.

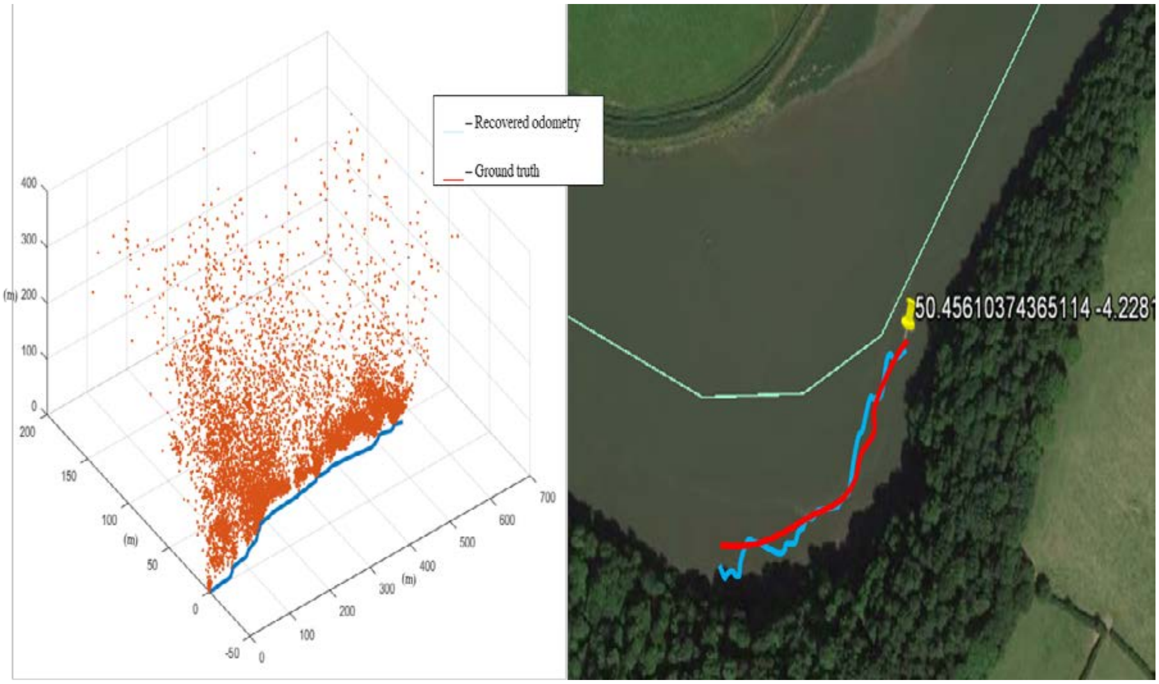

Figure 8.15. Recovered odometry and map (left) for a boat cruise approximately 50 m long near Halton Quay[26]. Approximate ground truth (GPS based spline) shown in red on the right.

[25] Demo video available at: https://youtu.be/LlRobDi-LME

[26] Demo video available at: https://youtu.be/7bmKTLHa-TE

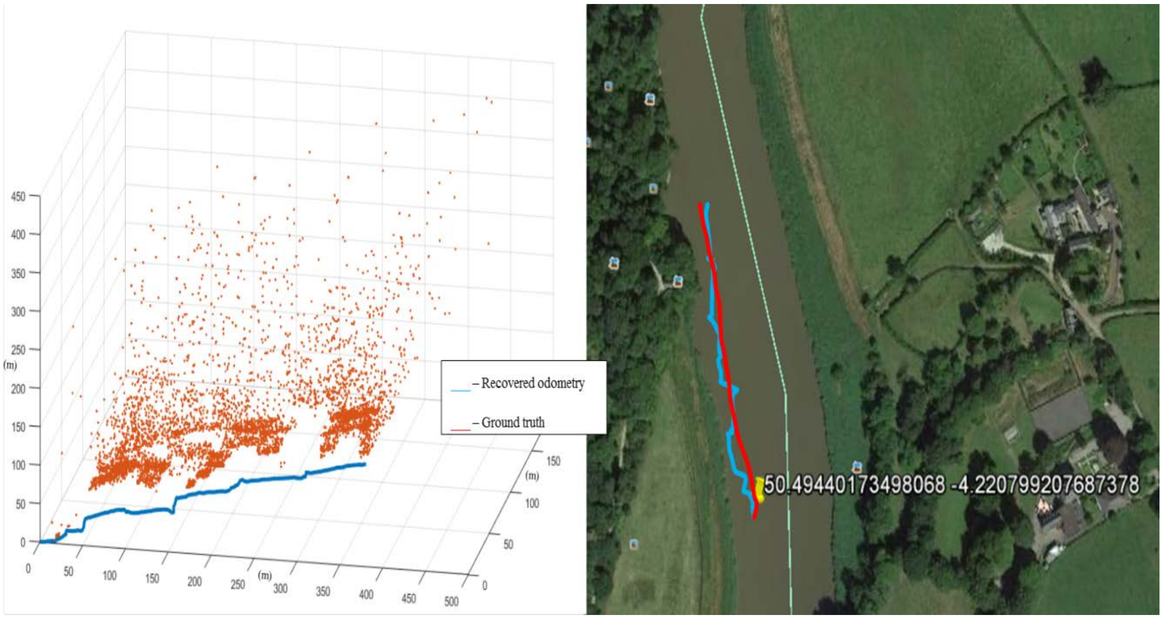

Figure 8.16. Recovered odometry and map (left) for a boat cruise near Bohetherick[27]. Approximate ground truth (GPS based spline) shown in red on the right.

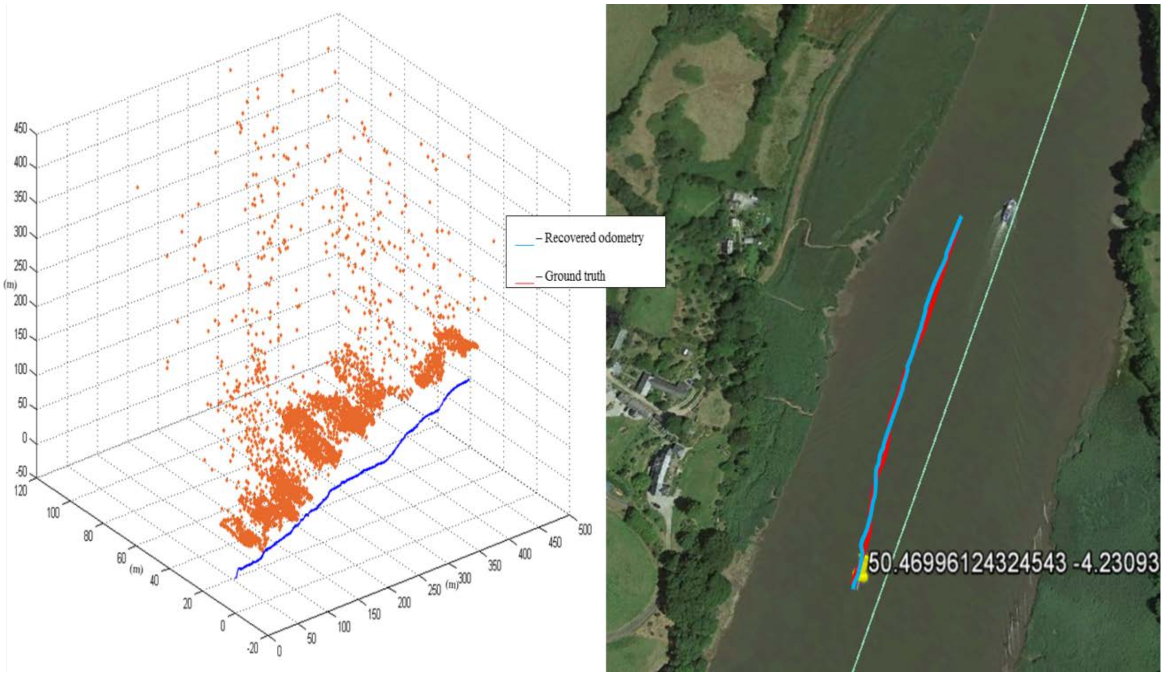

Figure 8.17. Recovered odometry and map (left) for a boat cruise in Halton Quay[28]. Approximate ground truth (GPS based spline) shown in red on the right.

[27] Demo video available at: https://youtu.be/QnhJ9HsKNKU

[28] Demo video available at: https://youtu.be/uMthIsPmrS0

8.4.5 Looking ahead: Camera-only visual SLAM without a map

It is the author's conviction that the successful odometry results illustrated in this section are primarily the result of the fact that camera pose is estimated directly from the correspondences and that the gyroscope inputs, albeit a significant aid, simply add stability to the SLAM posterior when tracking quality becomes low due to abrupt changes in vehicle attitude. Whether true or not, this conjecture must be put to the test with algorithms that recover camera pose directly from correspondences without the need of a map.

The only known "substitute" of the PnP algorithm for the recovery of camera pose directly from correspondences is essential matrix estimation. As discussed in Chapter 3, section 2.6, Nister's 5-point algorithm is the only known method for the computation of the essential matrix that abides by the two equal singular values constraint and it has shown remarkable results in practice. It is therefore a priority to use the 5-point algorithm (or the alternative iterative implementation proposed in Chapter 4, section 4.2) in order to establish whether the map-less approach is viable in the context of natural environments without the aid of a gyroscope.

It should be stressed however that, even if results with the 5-point algorithm turn-out in favor of the map-less approach, these results cannot be regarded as the outcome of a full test, but rather as a strong indication that the concept (i.e., map circumvention) is a valid strategy. The main argument here is that Nister's algorithm does not solve the overdetermined system, which is the case in the type of problems examined in this thesis. To formally dismiss or prove the conjecture of the success of map circumvention in visual SLAM in natural environments, one would have to employ a non-Procrustean solver for the essential matrix in the overdetermined setup. Chapter 3, section 2.6 elaborates on the formulation of the optimization problem for such a solver and it should also be a priority working towards this direction.

8.5 Summary

The opening sections of this chapter give details on standard algorithms for pose estimation (PnP) and filtering (bundle adjustment) in order to preempt two distinct approaches/frameworks for monocular SLAM in natural environments described in later sections. The first framework involves the use of a map structure containing the 3D locations of the observed features in order to estimate the new camera pose from the latest measurements, while the second, uses gyroscope measurements to formulate the pose estimation problem without using the map (i.e., directly from correspondences).

Results suggest that when employing the classic framework for visual SLAM wherein a map is gradually populated with new triangulated points and thereafter it is being used for new camera pose estimation, the SLAM posterior becomes very unstable in a matter of 20-30 frames in the sequence. One reason for this could be the fact that the problem is very ill-posed in the sense that the only information available comes from the camera in the form of feature image locations; however, it has been shown that this framework works well in practice in indoor environments and limited workspaces (PTAM). The most striking difference between the PTAM workspaces and the natural environments in this thesis is the fact that natural scenes present great depth variation and therefore reconstructing distant points can occasionally "blow" errors out of proportion (see Figure 3.16 in Chapter 3, section 3.4); in turn, this error is carried over into the new pose estimate and this loop corrupts the SLAM posterior rapidly. Removing "bad" points from the map is a solution, but the selection algorithm would have to be sophisticated enough in order to retain the points that have low tracking error and at the same time, reasonable disparity for motion estimation. The more the depth varies, the more difficult this task becomes.

The second and most successful visual SLAM framework introduced in this chapter is a "map-less" approach with the use of gyroscope input (angular rates) from an IMU. With information on relative orientation in place, it is possible to formulate the pose estimation problem purely in terms of image correspondences. Thus, the map is completely circumvented in terms of pose recovery. Furthermore, the SLAM measurement likelihood corresponding to a feature can also be parametrized in terms of the "home" and "base" views corresponding to the original detection and first-time tracking camera views for this particular feature. Although this parameterization heavily depends on cherry-picking the correspondences, it however is a more straightforward problem than the one of rejecting reconstructed 3D points and it generally works in practice as shown in Figures 8.10-17. The results suggest that the recovered odometry is robust within reasonable tolerance from GPS based approximations of ground truth. The distortion to the overall shape of the estimated camera positions is practically undetected to the naked eye and the differences from ground truth are most likely to originate in local scale discrepancies. Relatively accurate odometry does not necessarily suggest that the quality of all the recovered 3D locations is high, again primarily due to the wide range of depth. Distant points with small disparity will most likely produce very uncertain 3D locations (recall that these points are practically uninformative to the pose estimation algorithm). To ensure a better reconstruction, robust bundle adjustment should be executed over the map and the map points with low M-estimator weights should be omitted from the map.

Chapter 9

Conclusion

This thesis presented research on a two-fold problem in the context of natural landscapes: The estimation of camera odometry and the recovery of the sparse geometrical structure of the surrounding environment without prior knowledge of motion dynamics. Two fundamental approaches where investigated: Standard monocular visual SLAM with a map-centric approach to pose estimation and the so-called “map-less” Visual SLAM approach in which camera pose is estimated directly from the correspondences, with results clearly favoring the second approach (which may marginally include the algorithms for pairwise odometry discussed in Chapter 6). In either framework, classic algorithms for camera relative pose and scene reconstruction were considered and novel formulations were proposed, while paying respect to the prime objective of the research (i.e., visual odometry and mapping in natural environments without motion models and active sensors). These algorithms include the computation of the essential matrix, scene reconstruction from epipolar geometry, the Perspective-n-point (PnP) problem and bundle adjustment.

One of the most important conclusions to be highlighted is that the map-less approach appears to be working well in the context of environments with high depth variation, yet there are many facets of this solution yet to be investigated. In particular, it has not yet been established what would be the effectiveness of the method without accurate orientation priors (gyroscope). Furthermore, the methods for tracking and feature matching where more-less fixed in this research (OpenCV implementation of the LK tracker and FAST/SIFT features) and the SLAM algorithm would have to compensate for their limitations, especially for the inability of the tracker to track distinct image patches. This thesis has not yet quantified meticulously the effectiveness of the map-less approach for a number of reasons associated with existing state of the art in algorithms, limitation of available datasets and time restrictions.

9.1 Contributions to knowledge with respect to initial objectives

To the best of the author’s knowledge, the problem setting of the current research is unique and is not generally met in its entirety in literature. On one hand, visual SLAM from the vantage point of a cruising boat has not been particularly explored, while the ill-conditioned configuration of the problem (no motion model and gyroscopic aids only) on the other is hardly met in literature, even in the cases of visual SLAM applications in land. Thus, working methods under the current problem setting would potentially be of interest to the research community in field robotics and computer vision.

9.1.1 The “map-less” approach to visual SLAM

Although the standard formulation of visual SLAM is map-centric and there have been many successful implementations to support this approach, it is the author’s conclusion that when these approaches are taken “outside” in natural landscapes, then the map becomes the weakest link in the posterior updates.

In this research, an alternative approach to the visual SLAM framework is proposed, wherein the map is marginalized out of the posterior and odometry is estimated directly from the correspondences, without however regressing to a simple scheme for relative pose odometry (Chapter 8, section 4). The map-less approach is still a filter in which map points are parametrized in terms of their home (detection image) and base (first tracking measurement) view and the measurement likelihood is conditioned on the home and base view pose instead of the map point. This is an approach that works well in practice and circumvents the need to clean the map from noisy points which can be a very complicated issue when depth varies significantly. Another important aspect of the map-less approach is that the SLAM posterior does not scale with the number of features; this means that there are no limitations in practice as to how many points will be used for camera pose estimation and bundle adjustment. Finally, although the map does not directly participate in computations, it can still be recovered from camera pose and feature measurements in the respective home and base views.

9.1.2 Parametrization of 3D orientation and directions in spaces of arbitrary dimensionality with stereographic coordinates

Orientation constitutes a major topic in robotics, vision, space engineering and other disciplines dealing with problems that involve moving rigid objects. In a more general statement, the notion of orthonormality is ubiquitous in problems associated with the aforementioned disciplines. In this research, parametrizing orientation has been decisive for the proposition of novel algorithms, as well as for the modification and implementation of existing ones.

In Chapter 4, stereographic projection is discussed from the vantage point of 3D vision applications as an elegant parametrization, not only for 3D orientation, but more generally, for directions in spaces of arbitrary dimensionality. It turns-out that under this parametrization, rotation matrices not only become rational, but also have polynomial derivatives in the quaternion components (Appendix A, section 4.1), which, to the best of the author's knowledge is a novel observation. Furthermore, a 3 DOF parametric encoding of ball constraints based on 3D stereographic coordinates was proposed for use in iterative optimization (Chapter 4, section 4.1); this encoding eliminates the ball constraints from the optimization problem, in contrast to far more complicated standard techniques which typically have to manipulate the step of the iterative process in order to keep the new estimate in the feasible domain.

9.1.3 Pose estimation directly from correspondences: Formulations and potential solutions for the constrained essential matrix problem

Although not entirely a new technique, this thesis re-introduces the use of gyroscope input in camera pose estimation from correspondences in order to motivate the concept of a map-less SLAM algorithm. The method is originally described in Chapter 6, section 4.

In the camera-only case, the only way to estimate camera pose is through the essential matrix. In Chapter 3, section 2.6, a new formulation for the constrained essential matrix problem is given. In particular, the essential matrix is expressed as the sum of the tensor products of two pairs of orthogonal 3D vectors. The same section provides arguments that, although Nister's method is the best solution so far and gives very good results in the context of a RANSAC based solution, a more efficient solution to the relative pose problem would come from the essential matrix in the overdetermined setup.

A novel parametrization method for iterative optimization is proposed in Chapter 4, section 4.2 for the solution of the constrained essential matrix problem, based on stereographic projection. The method uses 3 stereographic coordinates to parametrize the rotation matrix and 2 more stereographic coordinates to parametrize the unit-norm baseline. The method should work in the 5-point setting as, for example, the Lui and Drummond iterative method (Lui and Drummond 2007).

9.1.4 Bundle adjustment without a map

Algorithm 8.2 describes a novel method for iterative optimization of the reprojection error without using the map. In this approach, the baseline is constrained to unit-length using stereographic coordinates and the map points are parametrized by the feature's home and base view measurements and the respective camera poses. This approach excludes the map from the optimization and circumvents the scaling problem of SLAM in the number of points.

9.1.5 A new formulation for the overdetermined PnP problem

In Chapter 8, section 1.3 a new formulation of the overdetermined PnP optimization problem is given. The cost function is a scaled convex combination of the eigenvalues of a known positive semidefinite matrix derived from the data. Based on this observation, one is able to adopt a greedy approach in which the 3 unknowns (out of a total of 9) corresponding to the largest singular values are chosen to be equal to zero and therefore the remaining 6 DOF are determined directly from the constraints, which is a system of quadratic 6 quadratic polynomials and is likely to admit a relatively simple solution.

9.1.6 Implementations of the algorithms described in the thesis

With the exception of Chapter 7, all algorithms described in this thesis where implemented in C# using the OpenCV "wrapper" library known as Emgu (Emgu 2013). The algorithms of Chapter 7 were implemented in Matlab (Guide 1998).

9.1.7 Dataset of estuarine and forest sequences from a moving vehicle

Many sequences have been recorded from a cruising boat in the Tamar river, Devon. These recordings include IMU and GPS logs synchronized with video in terms of frame indexes. Similar sequences were recorded from a moving van in forestall areas of Devon.

9.2 Looking ahead

This research has established the notion of map-less visual SLAM towards achieving robust localization in natural landscapes where scene depth varies significantly. Although encouraging results have been achieved with the aid of gyroscopic input, further research into the map-less approach is required in order to establish beyond any doubt that the successful recovery of odometry is primarily attributed to the circumvention of the map and not to the prior knowledge of relative orientation.

9.2.1 Camera-only map-less visual SLAM using constrained essential matrix computation

The only way to circumvent the map in visual SLAM without gyroscope input would be to extract camera pose from the essential matrix. To do this however, the essential matrix must not be obtained by a Procrustean least squares approach, but rather with the fully constrained optimization problem that ultimately leads to a 3rd or 4th degree polynomial system. Thus, it is a first priority to use Nister's algorithm in the context of RANSAC to examine the effectiveness of the map-less approach in comparison to the results presented in Chapter 8.

Although using Nister's algorithm should give a strong indication as to whether the map-less approach is effective without an IMU, it leaves however margin for doubt, due to the fact that the essential matrix estimate relies only on 5 correspondences. The overdetermined case is the actual setting of the problems in this thesis and therefore the constrained essential matrix computation in this context should be investigated. The iterative formulation based on stereographic projection converges extremely fast, but unfortunately, a heuristic rule that could safely "guide" the algorithm to the global minimum has yet to be found. This is a very interesting problem tangent to modern optimization approaches known as *convexification methods* (Bertsekas 1979).

Alternatively, a clever parametrization of the essential matrix may ultimately lead to a system of polynomials that can be solved without the need of Groebner solvers.

9.2.2 Incorporating motion priors

The research problem in its current form is very ill-posed. However, a motion model can make a huge difference in terms of efficiency and robustness. Thus, it is crucial to repeat the experimentation described in Chapters 6, 7, 8 with a vehicle for which motion dynamics have been well studied and modelled. Particularly the case of differential thrust dynamics (incidentally, Springer's design) is a very challenging model albeit ubiquitous in marine robotics. It would be therefore extremely interesting and useful for the community to adopt the methods proposed in this thesis to the types of surface vehicles that have this characteristic (i.e., differential thrust actuation). I would like to defer the reader here to an excellent piece of work on differential thrust dynamics by Wei Wang (Wang and Clark 2007).

9.2.3 Implementation of a new tracker for features detected in different images

The OpenCV LK tracker was employed in all implementations of the algorithms described in this thesis. It was mentioned in Chapter 5 that the most important limitation of the function is that it does not work on image patches, but rather on entire images by matching a given list of reference locations. This has been a problem in the case of overlapping SLAM, since the visible map-points where originating in various different home views. To rectify this problem, the SLAM framework invokes the LK tracker twice in order to track from the previous frame and the one before last.

A new implementation of the tracker (possibly the LK method again) is an important task that could improve performance significantly. Furthermore, it will reduce complexity of the SLAM finite state machine implementation.

9.2.4 Loop closure

Loop closure has not been amongst the objectives of this research. However, obtaining a map of 3D points would imply that this information can be potentially used in the near/distant future in order to enhance the SLAM posterior with a measurement. Since the map is essentially a point-cloud, it would be a very interesting problem to investigate the plausibility of generating and matching geometrical features of these point-clouds. There has been a great deal of work in the fields of geometric modeling and graphics (Bookstein 1989, Watt and Watt 1992, Golovanov 2014, Zollhöfer, Nießner et al. 2014) and could be potentially ported to provide solutions for loop closure in camera-only SLAM.

Appendix A

Quaternions and parametrization of attitude

A.1 Quaternions: A quick walkthrough

The numeral system of quaternions is an extension of complex numbers and was first introduced by Rowan Hamilton in 1843. The algebra of quaternions is equipped with addition and multiplication which is generally non-commutative. The set of quaternions is equal to $\mathbb{R}^4$ and usually is denoted with $\mathbb{H}$. In fact, quaternions can be represented as 4D vectors. The set of quaternions includes the imaginary elements *i*, *j* and *k*, which, together with the real number 1, form the set of *basis elements*, such that:

$$i^2 = j^2 = k^2 = ijk = -1$$

From the above, it can be seen that multiplication of the basis elements is not generally commutative. Specifically:

$$ij = k \;\; and\, ji = \; -k \;\; , \qquad jk = i \;\; and \;\; kj = -i \; , \quad ki = j \;\; and\, ik = \; -j$$

Using the basis elements and for any $q_0, q_1, q_2, q_3 \in \mathbb{R}$, one may obtain the general form of a quaternion *q* as:

$$q = q_0 + iq_1 + jq_2 + kq_3$$

It follows that *q* can also be denoted as a 4D vector, or a 4-tuple:

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} q_0 \\ v \end{bmatrix} \qquad or, \qquad q = \big(q_0, (q_1, q_2, q_3)\big) = (q_0, v)$$

where $v = [q_1 \quad q_2 \quad q_3]^T$ is the vector containing the *imaginary parts* and q_0 the *scalar part* of the quaternion.

A.1.1 Addition and multiplication

Quaternion addition and multiplication is a straightforward generalization of the multiplication of complex numbers using all four basis elements $(1, i, j, k)$. Hence, for any two quaternions $q = (q_0 \quad v)$ and $r = (r_0 \quad u)$, the corresponding sum and product are given as follows:

$$q + r = (q_0 + r_0, v + u) \tag{A.1}$$

$$qr = (q_0 r_0 - \ v \cdot u, v \times u + q_0 u + r_0 v) \tag{A.2}$$

The quaternion product qr can also be conveniently written in matrix form:

$$qr = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} r = Qr \tag{A.3}$$

The permuted rq product can also be written in matrix form using an expansion of q:

$$rq = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} r = Q^* r \tag{A.4}$$

The 4x4 matrices Q and Q^* differ only in that the lower-right-hand 3×3 (skew-symmetric) sub-matrix is transposed.

A.1.2 Norm and conjugate

Again, the concept of a conjugate quaternion comes as a natural extension of the conjugacy in complex numbers. Hence, the conjugate of q is,

$$\bar{q} = (q_0, -q_1, -q_2, -q_3) = (q_0, -v)$$

Accordingly, the norm $|q|$ of q, is given by,

$$|q| = \sqrt{q\bar{q}} = \sqrt{{q_0}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2}$$

From the above, the definition of the *inverse* quaternion q^{-1} follows naturally:

$$q^{-1} = \frac{\bar{q}}{|q|^2}$$

A very useful relationship between the product matrix Q of q and the product matrix $\bar{Q}$ of the conjugate quaternion $\bar{q}$ can be easily observed:

$$\bar{Q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \\ -q_1 & q_0 & -q_3 & q_2 \\ -q_2 & q_3 & q_0 & -q_1 \\ -q_3 & -q_2 & q_1 & q_0 \end{bmatrix} = Q^T \tag{A.5}$$

In quite the same way, $\bar{Q}^* = Q^{*T}$. Moreover, it turns out with a little algebra that the 4×4 product matrices Q and Q^* are orthogonal if q is a *unit quaternion* (i.e., has a unit norm). This observation will come handy in the following sections.

A.1.3 The composite product between arbitrary quaternions and 3D vectors

A quaternion can also be thought of as a scalar and a vector organized as a tuple. It follows that quaternions with a zero scalar part are representations of 3D vectors. We now define the *composite product* operator $L_q(r): \mathbb{H} \times \mathbb{R}^3 \rightarrow \mathbb{R}^3$ (acting on quaternions and 3D vectors) between the quaternion q and the vector r as follows:

$$L_q(r) = qr\bar{q} = (Qr)\bar{q} \tag{A.6}$$

where $q = (q_0\ , q_1, q_2, q_3)$ is a quaternion and $r = (0\ , r_1, r_2, r_3)$ a 3D vector represented as a purely imaginary quaternion.

It can be shown that the composite product maps purely imaginary quaternions onto the same set and that their norm remains unchanged. Moreover, using the matrices Q and $\bar{Q}$ in equations (A.3-4) regarding quaternion products, the composite product of equation (A.6) can be written as the following matrix product:

$$L_q(r) = qrq^{-1} = (Qr)\bar{q} = \bar{Q}^*(Qr) = (Q^{*T}Q)r \tag{A.7}$$

The matrix product $Q^{*T}Q$ of equation (A.8) yields the following matrix:

$$Q^{*T}Q = \begin{bmatrix} |q|^2 & 0 & 0 & 0 \\ 0 & {q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 0 & 2(q_1q_2 + q_0q_3) & {q_0}^2 - {q_1}^2 + {q_2}^2 - {q_3}^2 & 2(q_2q_3 - q_0q_1) \\ 0 & 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & {q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2 \end{bmatrix} \tag{A.8}$$

A.1.4 Unit quaternions as representations of rotations

The attention is now focused exclusively on unit quaternions. As observed earlier, the 4×4 product matrices Q and Q^* are orthogonal if q is a unit quaternion. It follows from equation (A.7) that $Q^{*T}Q$ should also be orthogonal. Most importantly, the 3×3 lower-right sub-matrix $R(q)$ of $Q^{*T}Q$,

$$R(q) = \begin{bmatrix} {q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & {q_0}^2 - {q_1}^2 + {q_2}^2 - {q_3}^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & {q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2 \end{bmatrix} \quad \text{(A.9)}$$

is also an orthogonal matrix. In fact, to motivate the following parametrization, $R(q)$ is a rotation matrix.

If a quaternion q has $|q| = 1$ (i.e., unity norm), then this intuitively implies (as in the case of complex numbers) that there exists an angle θ such as:

$$|q| = {q_0}^2 + \|v\|^2 \;\; s.t.: \; cos^2\theta = {q_0}^2 \; and \; sin^2\theta = \|v\|^2$$

From the above, the rotation matrix $R(q)$ can be parameterized in terms of θ and v. This practically means that vector v effectively defines an axis about which the 3D vector r is rotated (the direction of the rotation is determined by the direction of the axis vector using the right-hand thumb rule). The latter can be formalized with the following theorem (Kuipers 1999):

Theorem 2.1: *For any unit quaternion $q = \left(cos\frac{\theta}{2}, sin\frac{\theta}{2}v\right)$, $v \in \mathbb{R}^3$ and for any vector $r \in \mathbb{R}^3$ the action of the operator, $L_q(r) = qr\bar{q}$ is equivalent to a rotation about the axis and direction of q (following the right-hand-thumb rule) by an angle θ.*

A.2 Obtaining a unit quaternion from a rotation matrix

The matrix in equation (A.9) provides a straightforward formula for the conversion between the quaternion form and the corresponding rotation matrix. The reverse process is somewhat more complicated due to the inherent ambiguity in the squared terms contained in the diagonal of $R(q)$. This ambiguity however is eliminated in the course of the computations described in the following paragraphs.

Let $R(q) = [r_{ij}]$ be the given rotation matrix and $q = (q_0, q_1, q_2, q_3)$ the sought equivalent unit quaternion. By appropriately multiplying by +1 or -1 the diagonal elements of $R(q)$ and then adding them together, the following relationships for ${q_0}^2$, ${q_1}^2$, ${q_2}^2$, ${q_3}^2$ are obtained in terms of r_{11}, $r_{22}$and r_{33} (Horn 1987):

$$4{q_0}^2 = 1 + r_{11} + r_{22} + r_{33} \tag{A.10}$$

$$4{q_1}^2 = 1 + r_{11} - r_{22} - r_{33} \tag{A.11}$$

$$4{q_2}^2 = 1 - r_{11} + r_{22} - r_{33} \tag{A.12}$$

$$4{q_3}^2 = 1 - r_{11} - r_{22} + r_{33} \tag{A.13}$$

Also, from the off-diagonal elements of $R(q)$ the following relationships are obtained with similar processing of the element pairs (r_{21}, r_{12}) , (r_{31}, r_{13}) , (r_{32}, r_{23}) :

$$4q_0q_1 = r_{32} - r_{23} \tag{A.14}$$

$$4q_0q_2 = r_{13} - r_{31} \tag{A.15}$$

$$4q_0q_3 = r_{21} - r_{12} \tag{A.16}$$

$$4q_1q_2 = r_{21} + r_{12} \tag{A.17}$$

$$4q_2q_3 = r_{32} + r_{23} \tag{A.18}$$

$$4q_3q_1 = r_{31} + r_{13} \tag{A.19}$$

Equations (A.10-13) are the starting point of the conversion, since one of them will provide the solution for any one of the components of the quaternion. Each of these equations will have a real solution, which, zero excluded, corresponds to a negative and a positive number that share the same absolute value. However, the negative solution can be discarded due to the fact that the angle of rotation in quaternions cannot exceed π (since any angle greater than that will be subsumed into the rotation axis vector as a change of direction and not in the angle itself). In other words, if all the components of a unit quaternion are negated, then the resulting quaternion will represent the original rotation matrix. This means that we are free to choose the sign (typically positive) of the quaternion component obtained by one of equations (A.10-13).

From the above, one may solve for the rest of the components using equations (A.14-19). Selecting one of equations (A.10-13) to solve for one of the quaternion components and thereafter solving for the other components using the remaining equations yields the following four possible solutions:

$$q_R{}^{(0)}(R) = \frac{1}{2}\begin{bmatrix} \sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{32}-r_{23})\Big/\sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{13}-r_{31})\Big/\sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{21}-r_{12})\Big/\sqrt{1+r_{11}+r_{22}+r_{33}} \end{bmatrix} \tag{A.20}$$

$$q_R{}^{(1)}(R) = \frac{1}{2}\begin{bmatrix} (r_{32}-r_{23})\Big/\sqrt{1+r_{11}-r_{22}-r_{33}} \\ \sqrt{1+r_{11}-r_{22}-r_{33}} \\ (r_{21}+r_{12})\Big/\sqrt{1+r_{11}-r_{22}-r_{33}} \\ (r_{31}+r_{13})\Big/\sqrt{1+r_{11}-r_{22}-r_{33}} \end{bmatrix} \tag{A.21}$$

$$q_R{}^{(2)}(R) = \frac{1}{2}\begin{bmatrix} (r_{13}-r_{31})\Big/\sqrt{1-r_{11}+r_{22}-r_{33}} \\ (r_{21}+r_{12})\Big/\sqrt{1-r_{11}+r_{22}-r_{33}} \\ \sqrt{1-r_{11}+r_{22}-r_{33}} \\ (r_{32}+r_{23})\Big/\sqrt{1-r_{11}+r_{22}-r_{33}} \end{bmatrix} \tag{A.22}$$

$$q_R{}^{(3)}(R) = \frac{1}{2}\begin{bmatrix} (r_{21} - r_{12}) \Big/ \sqrt{1 - r_{11} - r_{22} + r_{33}} \\ (r_{31} + r_{13}) \Big/ \sqrt{1 - r_{11} - r_{22} + r_{33}} \\ (r_{32} + r_{23}) \Big/ \sqrt{1 - r_{11} - r_{22} + r_{33}} \\ \sqrt{1 - r_{11} - r_{22} + r_{33}} \end{bmatrix} \tag{A.23}$$

To choose which solution is the most suitable (with respect to using the greatest component solution as starting point) we consider q_R to be a function $q_R(R)\colon SO(3) \rightarrow \mathbb{H}$ that maps a rotation matrix to one of the quaternions given by equations (A.20-23) depending on certain conditions involving the elements of the diagonal of R. We may use q_R to implement a simple routine that converts a rotation matrix to a quaternion (Diebel 2006):

$$q_R(R) = \begin{cases} q_R{}^{(0)}(R)\ , & if\ \ r_{22} \geq -r_{33}\ ,\ r_{11} \geq -r_{22}\ \ ,\ r_{11} \geq -r_{33} \\ q_R{}^{(1)}(R)\ , & if\ \ r_{22} \leq -r_{33}\ ,\ r_{11} \geq r_{22}\ \ ,\ r_{11} \geq r_{33} \\ q_R{}^{(2)}(R)\ , & if\ \ r_{22} \geq r_{33}\ \ ,\ r_{11} \leq r_{22}\ \ ,r_{11} \leq -r_{33} \\ q_R{}^{(3)}(R)\ , & if\ \ r_{22} \leq r_{33}\ \ ,\ r_{11} \leq -r_{22}\ \ ,r_{11} \leq r_{33} \end{cases} \tag{A.24}$$

For the sake of completeness, the derivation for the set of inequalities that must hold in order for $q_R{}^{(1)}(R)$ to be the preferred solution is provided below:

Assume that for some rotation matrix, we solve equations (A.10-13) and find that $q_1 = \frac{1}{2}\sqrt{1 + r_{11} - r_{22} - r_{33}}$ is the largest component. The following inequalities will therefore be true:

$$q_1 \geq q_0 \Rightarrow \sqrt{1 + r_{11} - r_{22} - r_{33}} \geq \sqrt{1 + r_{11} + r_{22} + r_{33}}$$

$$\Leftrightarrow\ r_{22} \leq -r_{33} \tag{A.25}$$

$$q_1 \geq q_2 \Rightarrow \sqrt{1 + r_{11} - r_{22} - r_{33}} \geq \sqrt{1 - r_{11} + r_{22} - r_{33}}$$

$$\Leftrightarrow r_{11} \geq r_{22} \tag{A.26}$$

$$q_1 \geq q_2 \Rightarrow \sqrt{1 + r_{11} - r_{22} - r_{33}} \geq \sqrt{1 - r_{11} - r_{22} + r_{33}}$$

$$\Leftrightarrow r_{11} \geq r_{33} \tag{A.27}$$

The inequalities found in equations (A.25-27) are indeed the ones that should hold if ${q_R}^{(1)}(R)$ is chosen. The derivation of the other 3 conditions is similar.

A.3 Unit quaternions using the axis-angle parametrization

To achieve the minimum number of DOF (i.e., 3), it is necessary to revert back to the axis-angle encoding, this time using the following quaternion parametrization:

$$q = \left(cos\frac{v}{2}, sin\frac{v}{2}\left(\frac{u_1}{v}, \frac{u_2}{v}, \frac{u_3}{v}\right)\right) \tag{A.28}$$

where $u = [u_1 \quad u_2 \quad u_3]^T$ is the rotation axis vector and $v = \sqrt{{u_1}^2 + {u_2}^2 + {u_3}^2}$ is the norm of *u*. The DOF of *q* have now dropped to 3.

A.3.1 Obtaining the derivatives of the rotation matrix with respect to the axis-angle vector

Perhaps the most important entity in iterative non-linear optimization is the matrix of partial derivatives of the objective function, otherwise known as the *Jacobian*. It is therefore necessary to obtain analytical expressions of the derivatives of the rotation matrix with respect to the axis vector *u*. The general idea behind the derivation is to obtain the partial derivatives of the rotation matrix with respect to q_0, q_1, q_2, q_3 and their partial derivatives with respect to *u* in order apply the chain rule. Specifically, the derivatives of $R(q)$ with respect to *u* are given by,

$$\frac{\partial R(q(u))}{\partial u_i} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(u)}{\partial u_i} \tag{A.29}$$

The series of steps that leads to the calculation of the partial derivatives of a rotation matrix with respect to the axis-angle vector *u* will now be described. In the first step, the corresponding quaternion is calculated as follows:

$$q = \left(cos\frac{v}{2} \quad sin\left(\frac{v}{2}\right)\frac{u_1}{v} \quad sin\left(\frac{v}{2}\right)\frac{u_2}{v} \quad sin\left(\frac{v}{2}\right)\frac{u_2}{v}\right) \tag{A.30}$$

where v is the norm of the vector $u = [u_1 \quad u_2 \quad u_3]^T$ that encodes angle and axis, such that, $v = \sqrt{{u_1}^2 + {u_2}^2 + {u_3}^2}$.

Obtaining u from q is fairly straightforward, given that $v \in [0, \pi]$; we obtain $v = 2acosq_0$ and thereafter, $u = \frac{2acosq_0}{sin(acosq_0)} [q_1 \quad q_2 \quad q_3]^T$.

Let now $F_j = \frac{\partial R(q)}{\partial q_j}$ be the matrix of partial derivatives of the rotation with respect to the quaternion components $q_j, j = 0, 1, 2, 3$. The derivatives can be easily calculated as follows:

$$F_0 = \frac{\partial R(q)}{\partial q_0} = 2 \begin{bmatrix} q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \tag{A.31}$$

$$F_1 = \frac{\partial R(q)}{\partial q_1} = 2 \begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & -q_1 & -q_0 \\ q_3 & q_0 & -q_1 \end{bmatrix} \tag{A.32}$$

$$F_2 = \frac{\partial R(q)}{\partial q_2} = 2 \begin{bmatrix} -q_2 & q_1 & q_0 \\ q_1 & q_2 & q_3 \\ -q_0 & q_3 & -q_2 \end{bmatrix} \tag{A.33}$$

$$F_3 = \frac{\partial R(q)}{\partial q_3} = 2 \begin{bmatrix} -q_3 & -q_0 & q_1 \\ q_0 & -q_3 & q_2 \\ q_1 & q_2 & q_3 \end{bmatrix} \tag{A.34}$$

In the next step, the partial derivatives of the components of the quaternion with respect to the axis-angle vector u are computed:

$$G(u) = \frac{\partial q(u)}{\partial u} = \begin{bmatrix} \frac{\partial q(u)}{\partial u_1} & \frac{\partial q(u)}{\partial u_2} & \frac{\partial q(u)}{\partial u_3} \end{bmatrix} \tag{A.35}$$

The columns of $G(u)$ are given below in terms of u_1, u_2, u_3 and v:

$$\frac{\partial q(u)}{\partial u_1} = \begin{bmatrix} -\frac{u_1 sin\frac{v}{2}}{2v} \\ \frac{sin\frac{v}{2}}{v} + \frac{{u_1}^2 cos\frac{v}{2}}{2v^2} - \frac{{u_1}^2 sin\frac{v}{2}}{v^3} \\ u_1 u_2 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \\ u_1 u_3 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \end{bmatrix} = \frac{1}{2v^3} \begin{bmatrix} -v^2 u_1 sin\frac{v}{2} \\ 2({u_2}^2 + {u_3}^2) sin\frac{v}{2} + {u_1}^2 v cos\frac{v}{2} \\ u_1 u_2 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \\ u_1 u_3 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \end{bmatrix} \tag{A.36}$$

$$\frac{\partial q(u)}{\partial u_2} = \begin{bmatrix} -\frac{u_2 sin\frac{v}{2}}{2v} \\ u_1 u_2 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \\ \frac{sin\frac{v}{2}}{v} + \frac{{u_2}^2 cos\frac{v}{2}}{2v^2} - \frac{{u_2}^2 sin\frac{v}{2}}{v^3} \\ u_2 u_3 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \end{bmatrix} = \frac{1}{2v^3} \begin{bmatrix} -v^2 u_2 sin\frac{v}{2} \\ u_1 u_2 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \\ 2({u_1}^2 + {u_3}^2) sin\frac{v}{2} + {u_2}^2 v cos\frac{v}{2} \\ u_2 u_3 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \end{bmatrix} \tag{A.37}$$

$$\frac{\partial q(u)}{\partial u_3} = \begin{bmatrix} -\frac{u_3 sin\frac{v}{2}}{2v} \\ u_1 u_3 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \\ u_2 u_3 \left(\frac{cos\frac{v}{2}}{2v^2} - \frac{sin\frac{v}{2}}{v^3} \right) \\ \frac{sin\frac{v}{2}}{v} + \frac{{u_3}^2 cos\frac{v}{2}}{2v^2} - \frac{{u_3}^2 sin\frac{v}{2}}{v^3} \end{bmatrix} = \frac{1}{2v^3} \begin{bmatrix} -v^2 u_3 sin\frac{v}{2} \\ u_1 u_3 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \\ u_2 u_3 \left(v cos\frac{v}{2} - 2 sin\frac{v}{2} \right) \\ 2({u_1}^2 + {u_2}^2) sin\frac{v}{2} + {u_3}^2 v cos\frac{v}{2} \end{bmatrix} \tag{A.38}$$

The last step trivially involves the substitution of the derivatives of equations (A.31-38) in equation (A.29). Specifically, by adopting the convention $G(u) = [g_{ij}]$, the partial derivatives of the rotation matrix with respect to u can be computed as follows:

$$\frac{\partial R(q(u))}{\partial u_i} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(u)}{\partial u_i} = \sum_{j=0}^{3} F_j g_{ji} \tag{A.39}$$

A.4 Unit quaternions using stereographic projection parameters

For a parameter vector $\psi = (x, y, z) \in \mathbb{R}^3$ the corresponding unit quaternion is,

$$q = \left(\frac{2x}{\|\psi\|^2 + 1}, \frac{2y}{\|\psi\|^2 + 1}, \frac{2z}{\|\psi\|^2 + 1}, \frac{1 - \alpha^2}{\|\psi\|^2 + 1} \right) \qquad \text{(A.40)}$$

where $\|\psi\|^2 = x^2 + y^2 + z^2$. Conversely, given the coordinates q_0, q_1, q_2, q_3 of a unit quaternion the conversion is equally straightforward. Initially, α^2 is obtained:

$$\|\psi\|^2 = \frac{1 - q_3}{q_3 + 1} \qquad \text{(A.41)}$$

Then, the parameter vector is,

$$\psi = (x, y, z) = \left(\frac{q_0(\|\psi\|^2 + 1)}{2}, \frac{q_1(\|\psi\|^2 + 1)}{2}, \frac{q_2(\|\psi\|^2 + 1)}{2} \right)$$

$$\Leftrightarrow \quad \psi = \left(\frac{q_0}{1 + q_3}, \frac{q_1}{1 + q_3}, \frac{q_2}{1 + q_3} \right) \qquad \text{(A.42)}$$

A.4.1 Quaternion derivatives with respect to stereographic coordinates

The process of finding the derivatives of the rotation matrix $R(q)$ is directly analogous to the one described in section A.3.1. The only difference has to do with the matrix of partial derivatives of the quaternion with respect to ψ:

$$H(\psi) = \frac{\partial q(\psi)}{\partial \psi} = \begin{bmatrix} \frac{\partial q(\psi)}{\partial x} & \frac{\partial q(\psi)}{\partial y} & \frac{\partial q(\psi)}{\partial z} \end{bmatrix} \qquad \text{(A.43)}$$

where,

$$\frac{\partial q(\psi)}{\partial x} = \begin{bmatrix} \frac{2}{\|\psi\|^2 + 1} - \frac{4x^2}{(\|\psi\|^2 + 1)^2} \\ -\frac{4xy}{(\|\psi\|^2 + 1)^2} \\ -\frac{4xz}{(\|\psi\|^2 + 1)^2} \\ \frac{-4x}{(\|\psi\|^2 + 1)^2} \end{bmatrix} = -\begin{bmatrix} q_0^2 - q_3 - 1 \\ q_0 q_1 \\ q_0 q_2 \\ q_0(1 + q_3) \end{bmatrix} \qquad \text{(A.44)}$$

$$\frac{\partial q(\psi)}{\partial y} = \begin{bmatrix} -\frac{4xy}{(\|\psi\|^2+1)^2} \\ \frac{2}{\|\psi\|^2+1} - \frac{4y^2}{(\|\psi\|^2+1)^2} \\ -\frac{4yz}{(\|\psi\|^2+1)^2} \\ \frac{-4y}{(\|\psi\|^2+1)^2} \end{bmatrix} = -\begin{bmatrix} q_1 q_0 \\ q_1^2 - q_3 - 1 \\ q_1 q_2 \\ q_1(1+q_3) \end{bmatrix} \tag{A.45}$$

$$\frac{\partial q(\psi)}{\partial z} = \begin{bmatrix} -\frac{4xz}{(\|\psi\|^2+1)^2} \\ -\frac{4yz}{(\|\psi\|^2+1)^2} \\ \frac{2}{\|\psi\|^2+1} - \frac{4z^2}{(\|\psi\|^2+1)^2} \\ \frac{-4z}{(\|\psi\|^2+1)^2} \end{bmatrix} = -\begin{bmatrix} q_2 q_0 \\ q_2 q_1 \\ q_2^2 - q_3 - 1 \\ q_2(1+q_3) \end{bmatrix} \tag{A.46}$$

A.4.3 Rotation matrix derivatives with respect to stereographic coordinates

The derivatives of the rotation matrix can be calculated using the matrices F_0, F_1, F_2, F_3 in equations (A31-34):

$$\frac{\partial R(q(\psi))}{\partial x} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(\psi)}{\partial x} = \sum_{j=0}^{3} F_j \frac{\partial q_j(\psi)}{\partial x} \tag{A.47}$$

$$\frac{\partial R(q(\psi))}{\partial y} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(\psi)}{\partial y} = \sum_{j=0}^{3} F_j \frac{\partial q_j(\psi)}{\partial y} \tag{A.48}$$

$$\frac{\partial R(q(\psi))}{\partial z} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(\psi)}{\partial z} = \sum_{j=0}^{3} F_j \frac{\partial q_j(\psi)}{\partial z} \tag{A.49}$$

Thus, the derivatives of $R(\psi)$ with respect to the elements of ψ are obtained as follows:

$$\frac{\partial R(q(\psi))}{\partial \psi_i} = \sum_{j=0}^{3} \frac{\partial R(q)}{\partial q_j} \frac{\partial q_j(\psi)}{\partial \psi_i} = \sum_{j=0}^{3} F_j \frac{\partial q_j(\psi)}{\partial \psi_i} \tag{A.50}$$

where $(\psi_1, \psi_2, \psi_3) = (x, y, z)$ and $\frac{\partial q_j(\psi)}{\partial \psi_i}$ is the derivative of the *j*th component of the quaternion with respect to the *i*th component of the parameter vector.

Further to formulas (A.47-50), the derivatives of the 3 rotation rows r^1, r^2, r^3 as 3×1 vectors are given by the following expressions that contain the elements of the rotation matrix and the components of the quaternion:

$$\frac{\partial r^1}{\partial \psi} = -2 \begin{bmatrix} q_0(r_{11} - 2q_3 - 1\,) & q_1(r_{11} - 2q_3 - 1\,) & q_2(r_{11} + 1) \\ q_0 r_{12} + q_3^2 - q_0^2 + q_3 & q_1 r_{12} - \frac{1}{2} r_{32} - q_2 & q_2 r_{12} - \frac{1}{2} r_{13} - q_1 \\ q_0 r_{13} - \frac{1}{2} r_{23} - q_2 & q_1 r_{13} - q_3^2 + q_1^2 - q_3 & q_2 r_{13} + \frac{1}{2} r_{12} - q_0 \end{bmatrix} \tag{A.51}$$

Similarly, the Jacobian of r^2 with respect to ψ is:

$$\frac{\partial r^2}{\partial \psi} = -2 \begin{bmatrix} q_0 r_{21} - q_3^2 + q_0^2 - q_3 & q_1 r_{21} - \frac{1}{2} r_{23} - q_2 & q_2 r_{21} - \frac{1}{2} r_{31} - q_1 \\ q_0(r_{22} - 2q_3 - 1) & q_1(r_{22} + 1) & q_2(r_{22} - 2q_3 - 1) \\ q_0 r_{23} + \frac{1}{2} r_{13} + q_1 & q_1 r_{23} + \frac{1}{2} r_{21} + q_0 & q_2 r_{23} - q_3^2 + q_2^2 - q_3 \end{bmatrix} \tag{A.52}$$

Finally, the derivative of $r^3$is:

$$\frac{\partial r^3}{\partial \psi} = -2 \begin{bmatrix} q_0 r_{31} + \frac{1}{2} r_{32} + q_2 & q_1 r_{31} - q_3^2 + q_1^2 - q_3 & q_2 r_{31} + \frac{1}{2} r_{21} + q_0 \\ q_0 r_{32} - \frac{1}{2} r_{31} - q_1 & q_1 r_{32} + \frac{1}{2} r_{12} - q_0 & q_2 r_{32} - q_3^2 + q_2^2 - q_3 \\ q_0(r_{33} - 1) & q_1(r_{33} + 2q_3 + 1) & q_2(r_{33} + 2q_3 + 1) \end{bmatrix} \tag{A.53}$$

where $r_{ij} = [R]_{ij}$ is the element of the rotation matrix in the *i*th row and *j*th column.

A.5 Parameter differentiation for variance propagation

It is very often necessary to propagate variance from one set of parameters to another in the context of Gauss-Newton iteration. The conversion between axis-angle and stereographic projection parameters will always have to go through the quaternion. Thus, the Jacobian of both 3D parameters with respect to the quaternion is required.

The derivatives of the equatorial plane coordinates $\psi = (x, y, z)$ with respect to the quaternion q follow from equation (A.42):

$$\frac{\partial\psi}{\partial q} = \frac{1}{(q_3+1)^2}\begin{bmatrix} q_3+1 & 0 & 0 & -q_0 \\ 0 & q_3+1 & 0 & -q_1 \\ 0 & 0 & q_3+1 & -q_2 \end{bmatrix} \tag{A.54}$$

The derivatives of the stereographic projection parameters $\psi = (x, y, z)$ with respect to the axis-angle parameters $u = (u_1, u_2, u_3)$ can be computed as follows:

$$\frac{\partial\psi}{\partial u} = \frac{\partial\psi}{\partial q}\begin{bmatrix} \frac{\partial q}{\partial u_1} & \frac{\partial q}{\partial u_2} & \frac{\partial q}{\partial u_3} \end{bmatrix} = \frac{\partial\psi}{\partial q}\frac{\partial q}{\partial u} \tag{A.55}$$

In order to obtain the derivative of axis-angle parameters with respect to stereographic projection parameters, it is first necessary to compute $\frac{\partial u}{\partial q}$:

$$\frac{\partial u}{\partial q} = \begin{bmatrix} \frac{2q_1}{1-{q_0}^2} + \frac{2q_0q_1\cos^{-1}q_0}{(1-{q_0}^2)^{3/2}} & \frac{2\cos^{-1}q_0}{(1-{q_0}^2)^{1/2}} & 0 & 0 \\ \frac{2q_2}{1-{q_0}^2} + \frac{2q_0q_2\cos^{-1}q_0}{(1-{q_0}^2)^{3/2}} & 0 & \frac{2\cos^{-1}q_0}{(1-{q_0}^2)^{1/2}} & 0 \\ \frac{2q_3}{1-{q_0}^2} + \frac{2q_0q_3\cos^{-1}q_0}{(1-{q_0}^2)^{3/2}} & 0 & 0 & \frac{2\cos^{-1}q_0}{(1-{q_0}^2)^{1/2}} \end{bmatrix} \tag{A.56}$$

Applying the chain rule yields the derivative of *u* with respect to *ψ*:

$$\frac{\partial u}{\partial\psi} = \frac{\partial u}{\partial q}\begin{bmatrix} \frac{\partial q}{\partial x} & \frac{\partial q}{\partial y} & \frac{\partial q}{\partial z} \end{bmatrix} = \frac{\partial u}{\partial q}\frac{\partial q}{\partial\psi} \tag{A.57}$$

A.6 Derivatives of the sum of weighted squared rotation matrix elements

Suppose that the sum of squared rotation matrix components is weighted by $s_1, s_2, \dots, s_9 > 0$.

A.6.1 Sum of weighted squared diagonal elements

It is more convenient to separate the sum of the weighted squared diagonal elements *D* and take the derivative:

$$\begin{aligned} \frac{\partial D}{\partial q_0} = 4q_0[&(s_1+s_5+s_9)q_0^2 + (s_1-s_5-s_9)q_1^2 + (-s_1+s_5-s_9)q_2^2 \\ &+ (-s_1-s_5+s_9)q_3^2] \end{aligned} \tag{A.58}$$

$$\frac{\partial D}{\partial q_1} = 4q_1[(s_1 - s_5 - s_9)q_0^2 + (s_1 + s_5 + s_9)q_1^2 + (-s_1 - s_5 + s_9)q_2^2 + (-s_1 + s_5 - s_9)q_3^2] \tag{A.59}$$

$$\frac{\partial D}{\partial q_2} = 4q_2[(-s_1 + s_5 - s_9)q_0^2 + (-s_1 - s_5 + s_9)q_1^2 + (s_1 + s_5 + s_9)q_2^2 + (s_1 - s_5 - s_9)q_3^2] \tag{A.60}$$

$$\frac{\partial D}{\partial q_3} = 4q_3[(-s_1 - s_5 + s_9)q_0^2 + (-s_1 + s_5 - s_9)q_1^2 + (s_1 - s_5 - s_9)q_2^2 + (s_1 + s_5 + s_9)q_3^2] \tag{A.61}$$

A.6.2 Sum of weighted squared off-diagonal elements

Taking now the weighted sum of squared off-diagonal elements *F* of the rotation matrix:

$$\frac{\partial F}{\partial q_0} = 8\big[(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)q_1q_2q_3 + q_0\big((s_6 + s_8)q_1^2 + (s_3 + s_7)q_2^2 + (s_2 + s_4)q_3^2\big)\big] \tag{A.62}$$

$$\frac{\partial F}{\partial q_1} = 8\big[(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)q_0q_2q_3 + q_1\big((s_6 + s_8)q_0^2 + (s_2 + s_4)q_2^2 + (s_3 + s_7)q_3^2\big)\big] \tag{A.63}$$

$$\frac{\partial F}{\partial q_2} = 8\big[(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)q_0q_1q_3 + q_2\big((s_3 + s_7)q_0^2 + (s_2 + s_4)q_1^2 + (s_6 + s_8)q_3^2\big)\big] \tag{A.64}$$

$$\frac{\partial F}{\partial q_3} = 8\big[(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)q_0q_1q_2 + q_3\big((s_2 + s_4)q_0^2 + (s_3 + s_7)q_1^2 + (s_6 + s_8)q_2^2\big)\big] \tag{A.65}$$

A.6.2 Full derivatives

Taking the sum of (A58-61) and (A.62-65) yields the derivative with respect to the components of the quaternion:

$$
\begin{aligned}
\frac{\partial(D+F)}{\partial q_0} = 4q_0 \Bigg(& \underbrace{(s_1 + s_5 + s_9)}_{\alpha} q_0^2 + \underbrace{(s_1 - s_5 - s_9 + 2s_6 + 2s_8)}_{\beta} q_1^2 \\
& + \underbrace{(-s_1 + s_5 - s_9 + 2s_3 + 2s_7)}_{\gamma} q_2^2 \\
& + \underbrace{(-s_1 - s_5 + s_9 + 2s_2 + 2s_4)}_{\delta} q_3^2 \Bigg) \\
& + 8\underbrace{(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)}_{\varepsilon} q_1 q_2 q_3
\end{aligned}
\tag{A.66}
$$

$$
\begin{aligned}
\frac{\partial(D+F)}{\partial q_1} = 4q_1 \Bigg(& \underbrace{(s_1 - s_5 - s_9 + 2s_6 + 2s_8)}_{\beta} q_0^2 + \underbrace{(s_1 + s_5 + s_9)}_{\alpha} q_1^2 \\
& + \underbrace{(-s_1 - s_5 + s_9 + 2s_2 + 2s_4)}_{\delta} q_2^2 \\
& + \underbrace{(-s_1 + s_5 - s_9 + 2s_3 + 2s_7)}_{\gamma} q_3^2 \Bigg) \\
& + 8\underbrace{(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)}_{\varepsilon} q_0 q_2 q_3
\end{aligned}
\tag{A.67}
$$

$$
\begin{aligned}
\frac{\partial(D+F)}{\partial q_2} = 4q_2 \Bigg(& \underbrace{(-s_1 + s_5 - s_9 + 2s_3 + 2s_7)}_{\gamma} q_0^2 \\
& + \underbrace{(-s_1 - s_5 + s_9 + 2s_2 + 2s_4)}_{\delta} q_1^2 + \underbrace{(s_1 + s_5 + s_9)}_{\alpha} q_2^2 \\
& + \underbrace{(s_1 - s_5 - s_9 + 2s_6 + 2s_8)}_{\beta} q_3^2 \Bigg) \\
& + 8\underbrace{(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)}_{\varepsilon} q_0 q_1 q_3
\end{aligned}
\tag{A.68}
$$

$$
\begin{aligned}
\frac{\partial(D+F)}{\partial q_3} = 4q_3 \Bigg(& \underbrace{(-s_1 - s_5 + s_9 + 2s_2 + 2s_4)}_{\delta} q_0^2 \\
& + \underbrace{(-s_1 + s_5 - s_9 + 2s_3 + 2s_7)}_{\gamma} q_1^2 \\
& + \underbrace{(s_1 - s_5 - s_9 + 2s_6 + 2s_8)}_{\beta} q_2^2 + \underbrace{(s_1 + s_5 + s_9)}_{\alpha} q_3^2 \Bigg) \\
& + 8\underbrace{(-s_2 + s_3 + s_4 - s_6 - s_7 + s_8)}_{\varepsilon} q_0 q_1 q_2
\end{aligned}
\tag{A.69}
$$

What is of great interest in the derivative expressions of (A.66-69) is that there are only 5 distinct coefficients, $\alpha, \beta, \gamma, \delta, \varepsilon$. Using these “shortcut”, the derivatives can be written as follows:

$$\frac{\partial(D+F)}{\partial q_0} = 4q_0(\alpha q_0^2 + \beta q_1^2 + \gamma q_2^2 + \delta q_3^2) + 8\varepsilon q_1 q_2 q_3 \qquad \text{(A.70)}$$

$$\frac{\partial(D+F)}{\partial q_1} = 4q_1(\beta q_0^2 + \alpha q_1^2 + \delta q_2^2 + \gamma q_3^2) + 8\varepsilon q_0 q_2 q_3 \qquad \text{(A.71)}$$

$$\frac{\partial(D+F)}{\partial q_2} = 4q_2(\gamma q_0^2 + \delta q_1^2 + \alpha q_2^2 + \beta q_3^2) + 8\varepsilon q_0 q_1 q_3 \qquad \text{(A.72)}$$

$$\frac{\partial(D+F)}{\partial q_3} = 4q_3(\delta q_0^2 + \gamma q_1^2 + \beta q_2^2 + \alpha q_3^2) + 8\varepsilon q_0 q_1 q_2 \qquad \text{(A.73)}$$

where,

$$\alpha = s_1 + s_5 + s_9 \qquad \text{(A.74)}$$

$$\beta = s_1 - s_5 - s_9 + 2s_6 + 2s_8 \qquad \text{(A.75)}$$

$$\gamma = -s_1 + s_5 - s_9 + 2s_3 + 2s_7 \qquad \text{(A.76)}$$

$$\delta = -s_1 - s_5 + s_9 + 2s_2 + 2s_4 \qquad \text{(A.77)}$$

$$\varepsilon = -s_2 + s_3 + s_4 - s_6 - s_7 + s_8 \qquad \text{(A.78)}$$

Appendix B

Derivatives of the SVD

A somewhat more detailed derivation of the brilliant method for the estimation of the derivatives of the SVD by Papadopoulo and Lourakis is provided here. Consider the SVD $B = UDV^T$ where $B = \left[b_{ij}\right]$. Taking the derivatives of the matrices in both sides of the SVD, the following relationship is obtained (right side is expanded using the product derivation rule):

$$\frac{\partial B}{\partial b_{ij}} = \frac{\partial U}{\partial b_{ij}} DV^T + U\frac{\partial D}{\partial b_{ij}} V^T + UD\frac{\partial V^T}{\partial b_{ij}} \tag{B.1}$$

Evidently, $\frac{\partial B}{\partial b_{ij}}$ is a matrix with elements $\left(\frac{\partial B}{\partial b_{ij}}\right)_{kl}$ such that,

$$\left(\frac{\partial B}{\partial b_{ij}}\right)_{kl} = \delta[k - i\,, l - j] \tag{B.2}$$

where $\delta[m\,, n]$ is the 2D Kronecker delta function. Moreover, orthogonality for U and V yields the following:

$$U^T U = I \Rightarrow \frac{\partial U^T}{\partial b_{ij}} U + \underbrace{U^T \frac{\partial U}{\partial b_{ij}}}_{\Omega_U^{ij}} = 0 \tag{B.3}$$

and,

$$V^T V = I \Rightarrow \underbrace{\frac{\partial V^T}{\partial b_{ij}}}_{\Omega_V^{ij}} V + V^T \frac{\partial V}{\partial b_{ij}} = 0 \tag{B.4}$$

where Ω_U^{ij} and Ω_V^{ij} are antisymmetric matrices and used as shortcuts for the following expressions:

$$\Omega_U^{ij} = U^T \frac{\partial U}{\partial b_{ij}} \tag{B.5}$$

and,

$$\Omega_V^{ij} = \frac{\partial V^T}{\partial b_{ij}} V \tag{B.6}$$

Now, multiplying equation (B.1) with U^T from the left and V from the right yields the following:

$$U^T \frac{\partial B}{\partial b_{ij}} V = U^T \frac{\partial U}{\partial b_{ij}} D + \frac{\partial D}{\partial b_{ij}} + D \frac{\partial V^T}{\partial b_{ij}} V^T \tag{B.7}$$

Substituting the expressions for Ω_U^{ij} and Ω_V^{ij} from equations (B.5-6) into (B.7) the following is obtained:

$$U^T \frac{\partial B}{\partial b_{ij}} V = \Omega_U^{ij} D + \frac{\partial D}{\partial b_{ij}} + D\Omega_V^{ij} \tag{B.8}$$

Since Ω_U^{ij} and Ω_V^{ij} are antisymmetric, it follows that they have zero elements in the diagonal and therefore, the diagonal of the sum on the right side of equation (B.8) is equal to the diagonal of $\frac{\partial D}{\partial b_{ij}}$. Moreover, considering that the derivative of B with respect to b_{ij} is a matrix with a 1-entry in the *i*th row, *j*th column and all other entries are zero as shown in equation (B.2), it follows that the *k*th diagonal element of $U^T \frac{\partial B}{\partial b_{ij}} V$ is the product, $u_{ik} v_{jk}$ where $U = [u_{ij}]$ and $V = [v_{ij}]$. Thus, the derivatives of the singular values are,

$$\frac{\partial d_k}{\partial b_{ij}} = u_{ik} v_{jk} \tag{B.9}$$

Considering now equation (B.8) in terms of the off-diagonal elements on both sides, the following 2×2 system is obtained for the elements of Ω_U^{ij} and Ω_V^{ij} in the *k*th row and *l*th column:

$$\begin{cases} d_l\left(\omega_U^{ij}\right)_{kl} + d_k\left(\omega_V^{ij}\right)_{kl} = u_{ik}v_{jl} \\ d_k\left(\omega_U^{ij}\right)_{kl} + d_l\left(\omega_V^{ij}\right)_{kl} = -u_{il}v_{jk} \end{cases} \qquad \text{(B.10)}$$

where $\Omega_U^{ij} = \left[\left(\omega_U^{ij}\right)_{kl}\right]$ and $\Omega_V^{ij} = \left[\left(\omega_V^{ij}\right)_{kl}\right]$. The derivatives of U and V can then be easily computed from equations (B.5-6) as follows:

$$\frac{\partial U}{\partial b_{ij}} = U\Omega_U^{ij} \ , \ \ \frac{\partial V}{\partial b_{ij}} = -V\Omega_V^{ij} \qquad \text{(B.11)}$$

Appendix C

Map marginalization in a single scene

Consider the isolated case of a single scene comprised of t-views in which the sequence of commonly observed feature vectors is, $m_k \in \mathbb{R}^{2N}$ where N is the number of features and $k \in \{1, \dots, t\}$. Moreover, let $x_k = [\eta_k^T \quad s_k^T]^T$ where $\eta_k, s_k \in \mathbb{R}^3$ are the camera orientation parameters and position (expressed in global coordinates) respectively and $M \in \mathbb{R}^{3N}$ the locations of the features in the world in terms of the first camera frame.

Suppose now that there exists some function h such that, $p(M|x_{0:1}, m_0, m_1) = \delta\big(M - h(x_0, x_1, m_0, m_1)\big)$ where δ denotes the Dirac delta function. Then, incorporating this constraint into the standard SLAM Bayes network yields the slightly modified filtration process depicted in Figure C.1 (control inputs are omitted for simplicity).

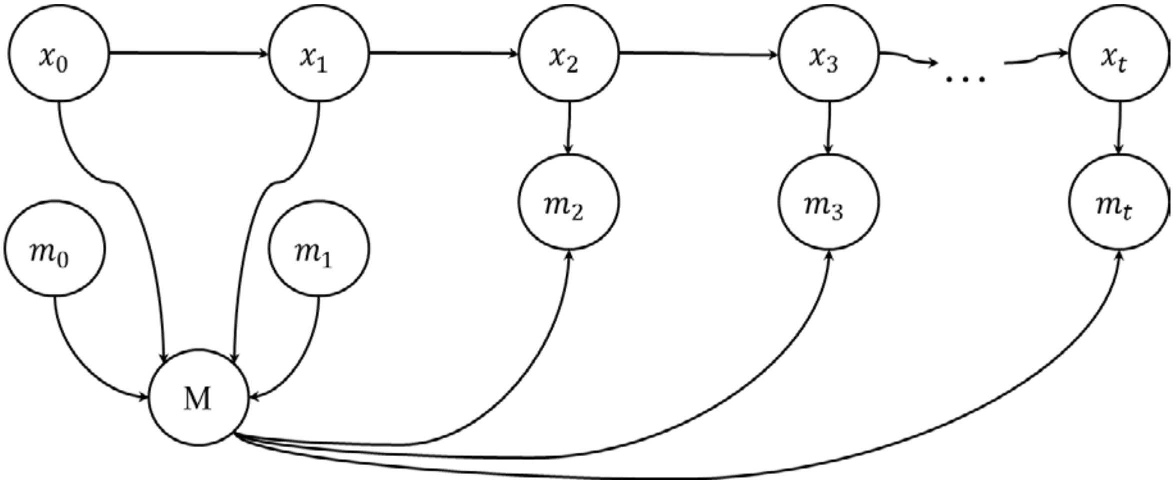

Figure C.1. The slightly modified SLAM paradigm that incorporates the initialization of the map from the first two views in the scene.

C.1 Marginalizing-out the map

Clearly, the first two views are used for initialization and the tracked feature locations m_0 and m_1 do not appear as likelihoods in the posterior, but rather as prior distributions $p(m_0) = \delta(\hat{m}_0)$ and $p(m_1) = N(\hat{m}_1, Q_1)$ where $\hat{m}_0$ and $\hat{m}_1$ are the respective means. The original feature locations bear no uncertainty, but the tracked locations in the second view are stochastic due to the uncertainty entailed by the "noisy" optical flow estimation algorithm. Thus, the computation of SLAM posteriors with measurement information commences after the third frame has been sampled. To marginalize the map out of the posterior $Bel(x_{0:2}, M) = p(x_{0:2}, M|m_2)$ in order to obtain the joint pose marginal $Bel^*(x_{0:2}) = p(x_{0:2}|m_2)$, one simply needs to integrate:

$$\begin{aligned} &Bel^*(x_2, x_{0:1}) \\ &\propto \int\int p(m_2|x_2, M)p(x_2|x_1)p(M|x_{0:1}, m_{0:1})p(x_1|x_0)p(x_0)p(m_1)dM\, dm_1 \end{aligned} \tag{C.1}$$

But since $p(M|x_{0:1}, m_0, m_1) = \delta\big(M - h(x_0, x_1, m_0, m_1)\big)$, the integral of equation (C.1) yields,

$$Bel^*(x_2, x_{0:1}) \propto \underbrace{\int p\big(m_2|x_2, h(x_{0:1}, m_{0:1})\big)p(m_1)dm_1}_{q(m_2|x_2,x_{0:1})} \underbrace{p(x_2|x_1)p(x_1|x_0)p(x_0)}_{q(x_2,x_{0:1})} \tag{C.2}$$

where the product $p(x_2|x_1)p(x_1|x_0)p(x_0)$ can be regarded as a prior $q(x_2, x_{0:1})$ for $x_{0:2}$ and the marginal distribution $\int p\big(m_2|x_2, h(x_{0:1}, m_{0:1})\big)p(m_1)dm_1$ as the respective measurement likelihood, $q(m_2|x_2, x_{0:1})$. Note here that since the prior of m_0 is a Dirac delta function, the variable is instantiated in the expression that yields the posterior.

It follows that for any posterior, the prior of m_1 is subsumed into all likelihood terms via marginalization, thereby yielding $q(m_k|x_2, x_{0:1})$ for $k = 2, \ldots, t$. Thus, similarly to equation (C.2), the marginal of $x_3, x_{0:1}$ over x_2, M is computed as follows (after the substitution of *M* with $h(x_{0:1}, m_{0:1})$):

$$\begin{aligned} &Bel^*(x_3, x_{0:1}) \\ &\propto \int \underbrace{\left(\int p\big(m_3|x_3, h(x_{0:1}, m_{0:1})\big)p\big(m_2|x_2, h(x_{0:1}, m_{0:1})\big)p(m_1)dm_1\right)}_{q(m_3|x_3,x_{0:1})q(m_2|x_2,x_{0:1})} \underbrace{q(x_2, x_{0:1})p(x_3|x_2)}_{Bel^*(x_2,x_{0:1})} dx_2 \end{aligned} \tag{C.3}$$

It is therefore clear that the joint $x_k, x_{0:1}, k = 2, \dots, t$ has the dynamics of a typical hidden Markov model (HMM) in which the measurement likelihood and the state transition are Gaussian distributions.

C.1.1 The new filter

The parametrization in terms of $x_{0:1}$ and consequently, the marginalization of M out of the posterior, effectively gives rise to a new filter for time instances $k = 2, \dots, t$. The new state vector y_k is comprised of the pose at time k and the initial poses x_0 and x_1:

$$y_k = \begin{bmatrix} x_k \\ x_1 \\ x_0 \end{bmatrix} \tag{C.4}$$

The state transition $q(y_k|y_{k-1})$ is essentially the motion model $p(x_k|x_{k-1})$ for $k > 2$ and the state prior at time $k = 2$ is given by equation (C.2).

$$q(y_k|y_{k-1}) = p(x_k|x_{k-1}) \tag{C.5}$$

What is of particular interest, is the new measurement likelihood, provided that now the map is replaced by $h(x_{0:1}, m_{0:1})$ in the observation model. Let $p(m_k|x_k, M) = N(f(x_k, M), Q_k)$ be the original measurement likelihood. It follows, that the new measurement likelihood $q(m_k|y_k)$ will be the following distribution:

$$q(m_k|y_k) = N(f^*(y_k), Q_k^*) \tag{C.6}$$

where $f^*(y_k) = f\big(x_k, h(x_{0:1}, m_{0:1})\big)$ and Q_k^* is a covariance matrix that not only incorporates the uncertainty of the original likelihood, but also the additional uncertainty entailed by the optical flow estimates. Considering that m_0 is a non-stochastic quantity, using the Jacobian of f with respect to m_1, Q_k^* is estimated as follows:

$$Q_k^* = Q_k + \frac{\partial f^*}{\partial m_1} Q_1 \left(\frac{\partial f^*}{\partial m_1}\right)^T \tag{C.7}$$

where Q_1 is the covariance of optical flow estimates in the second view in the scene.

C.1.2 Obtaining the map

The marginalization of m_1 works very well in terms of obtaining a new filter over y_k, yet for the same reason, it is impossible to recover the map from this filter. A look at the marginal that yields the map over the poses suggests that marginalizing m_1 out this time does not eliminate the map from the problem; instead, it leads to a full-fledged SLAM problem with all the pathologies associated with the size of *M*.

The practical alternative is to obtain an approximation of *M* from the belief of y_k by simply substituting $x_{0:1}$ and m_1 in h. Although this is not the true SLAM estimate, however, provided that tracking uncertainty in the second frame is relatively low, then the recovered map can be fairly reliable. In quite the same way, an approximate covariance matrix of *M* is obtained by propagating variance through h. Thus,

$$\Sigma_M = \frac{\partial h}{\partial x_{0:1}} \Sigma_{x_{0:1}} \left(\frac{\partial h}{\partial x_{0:1}} \right)^T \tag{C.8}$$

where $\Sigma_{x_{0:1}}$ is the covariance matrix of $x_{0:1}$ in the posterior of y_k.

C.2 Triangulation of 3D points

Clearly, one way of parametrizing the map in terms of $m_{0:1}$ and $x_{0:1}$ is triangulation. For the needs of this thesis, the suboptimal midpoint method suffices to demonstrate the merits of map marginalization. The steps in the derivation of the triangulated coordinates of a 3D feature location using the midpoint method are re-introduced briefly here, in order to compute the derivative of the recovered parameters.

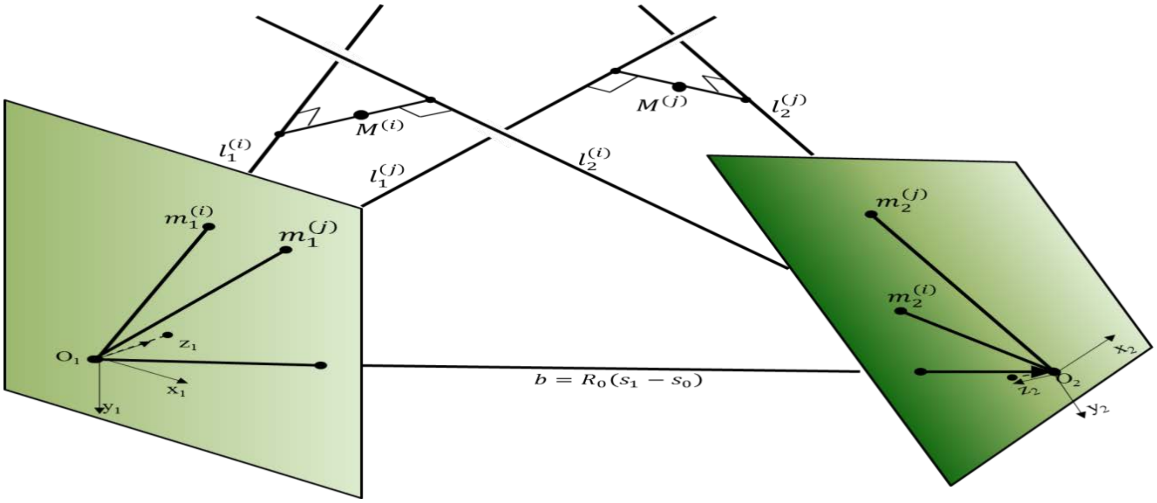

Figure C.1. An illustration of the midpoint triangulation method.

Let $m_1^{(i)}$ and $m_2^{(i)}$ be the corresponding locations of the i^{th} feature in views of a scene. Also, let *R* be the relative orientation rotation matrix and *b* the baseline. Finally, let

$l_1^{(i)}$ be the ray that passes through the first camera center O_1 and the projection $m_1^{(i)}$. The parametric equation of $l_1^{(i)}$ in the coordinate frame of O_1 is,

$$l_1^{(i)}(\kappa) = \kappa K^{-1}\overline{m}_1^{(i)} \tag{C.9}$$

where $\overline{m}_1^{(i)} = \left[\left(m_1^{(i)}\right)^T \quad 1\right]^T$ is the normalized homogenous representation of $m_1^{(i)}$ and K is the matrix of camera intrinsic parameters. Similarly, the ray $l_2^{(i)}$ that passes through the second camera center O_2 and the projection $m_2^{(i)}$ is,

$$l_2^{(i)}(\lambda) = \lambda RK^{-1}\overline{m}_2^{(i)} + R_0(s_1 - s_0) \tag{C.10}$$

where κ, λ are free real parameters, R_0 is the rotation matrix corresponding to the initial camera orientation and $R_0(s_1 - s_0)$ is the baseline vector in the coordinate frame of the first view. Finally, a vector $w^{(i)}$, perpendicular to both rays is obtained using the cross product operator:

$$w^{(i)} = \left(K^{-1}\overline{m}_1^{(i)}\right) \times \left(RK^{-1}\overline{m}_2^{(i)}\right) \tag{C.11}$$

Define vectors $u^{(i)} = K^{-1}\overline{m}_1^{(i)}$ and $v^{(i)} = R_0^T R_1 K^{-1}\overline{m}_2^{(i)}$ where R_1 is the rotation matrix that corresponds to the orientation of the second camera frame. Then, there exist values $\kappa^*, \lambda^*, \rho^* \in \mathbb{R}$ that verify the following:

$$\kappa^* u = \lambda^* v + R_0(s_1 - s_0) + \rho^* w \tag{C.12}$$

Equation (C.12) defines a 3x3 system of linear equations in terms of k, λ and ρ with the following solution:

$$\tau^{(i)} = \begin{bmatrix}\kappa^* \\ \lambda^* \\ \rho^*\end{bmatrix} = \begin{bmatrix}u^{(i)} & -v^{(i)} & -w^{(i)}\end{bmatrix}^{-1} R_0(s_1 - s_0) \tag{C.13}$$

Thus, the triangulated point is,

$$M^{(i)} = \left[u^{(i)} \quad 0_3 \quad -\frac{1}{2}w^{(i)}\right]\tau^{(i)} = R_0(s_1 - s_0) + \left[0_3 \quad v^{(i)} \quad \frac{1}{2}w^{(i)}\right]\tau^{(i)} \tag{C.14}$$

where 0_3 is the 3D zero-vector.

C.3 Jacobian of the triangulated point

C.3.1 Shortcut notation for derivatives of products of matrices with vectors

Although the derivatives of rotation matrices in terms of the respective orientation parameters are tensors, the derivatives of the products rotation matrices with vectors are matrices and a shortcut expression is derived in this subsection.

Consider the product $A(\gamma)\beta$ where A is a matrix and $\beta \in \mathbb{R}^3$ and $\gamma = (\gamma_1, \gamma_2, \dots, \gamma_N) \in \mathbb{R}^N$ is a vector of parameters. Define the following shortcut notation for the product derivative,

$$\Delta_\gamma^R\langle\beta\rangle = \frac{\partial(A(\gamma)\beta)}{\partial\gamma} = \left[\frac{\partial A(\gamma)}{\partial\gamma_1}\beta \quad \frac{\partial A(\gamma)}{\partial\gamma_2}\beta \quad \dots \quad \frac{\partial A(\gamma)}{\partial\gamma_N}\beta\right] \tag{C.15}$$

The notation $\Delta_\gamma^R\langle\beta\rangle$ will henceforth denote the derivative of the product $A(\eta)\beta$ in terms of γ and will be called *the delta matrix of A,β in terms of γ*.

C.3.2 Derivatives with respect to orientation

Let R_0 and R_1 are the rotation matrices that correspond to the orientation parameter vectors η_0 and η_1. It follows that the relative pose rotation matrix will be, $R = R_1 R_0^T$. The Jacobian of $u^{(i)}$, $v^{(i)}$ and $w^{(i)}$ with respect to $\eta_0$and η_1 are:

$$\frac{\partial u^{(i)}}{\partial\eta_0} = \frac{\partial u^{(i)}}{\partial\eta_1} = 0_{3\times3} \tag{C.16}$$

where $0_{3\times3}$ is the 3 × 3 zero-matrix.

$$\frac{\partial v^{(i)}}{\partial\eta_0} = R_1 \Delta_{\eta_0}^{R_0^T}\langle K^{-1}\bar{m}_1^{(i)}\rangle \tag{C.17}$$

$$\frac{\partial v^{(i)}}{\partial\eta_1} = \Delta_{\eta_1}^{R_1}\langle R_0^T K^{-1}\bar{m}_1^{(i)}\rangle \tag{C.18}$$

$$\frac{\partial w^{(i)}}{\partial\eta_0} = [u]_\times \frac{\partial v^{(i)}}{\partial\eta_0} \tag{C.19}$$

$$\frac{\partial w^{(i)}}{\partial \eta_1} = [u]_\times \frac{\partial v^{(i)}}{\partial \eta_1} \tag{C.20}$$

Let $A = [u^{(i)} \quad -v^{(i)} \quad -w^{(i)}]$ be the matrix of coefficients in the 3×3 linear system of equation (C.13) that yields the triangulation parameters. The derivatives of A with respect to η_0 and η_1 are tensors and therefore obtained from the columns of the Jacobians of $u^{(i)}$, $v^{(i)}$ and $w^{(i)}$:

$$\frac{\partial A}{\partial (\eta_0)_k} = \left[0_{3\times 3} \quad -\left(\frac{\partial v^{(i)}}{\partial \eta_0}\right)_k \quad -\left(\frac{\partial w^{(i)}}{\partial \eta_0}\right)_k\right] \tag{C.21}$$

and,

$$\frac{\partial A}{\partial (\eta_1)_k} = \left[0_{3\times 3} \quad -\left(\frac{\partial v^{(i)}}{\partial \eta_1}\right)_k \quad -\left(\frac{\partial w^{(i)}}{\partial \eta_1}\right)_k\right] \tag{C.22}$$

It is now straightforward to obtain the Jacobian of A^{-1} as follows:

$$\frac{\partial A^{-1}}{\partial (\eta_0)_k} = A^{-1} \frac{\partial A}{\partial (\eta_0)_k} A^{-1} \tag{C.23}$$

and,

$$\frac{\partial A^{-1}}{\partial (\eta_1)_k} = A^{-1} \frac{\partial A}{\partial (\eta_1)_k} A^{-1} \tag{C.24}$$

The solution of equation (C.13) is now expressed in terms of the pose vectors:

$$\tau^{(i)} = [u^{(i)} \quad -v^{(i)} \quad -w^{(i)}]^{-1} R_0 (s_1 - s_0) \tag{C.25}$$

Thus, the derivatives of $\tau^{(i)}$ with respect to η_1 and η_0 are the following:

$$\frac{\partial \tau^{(i)}}{\partial \eta_0} = \Delta_{\eta_0}^{A^{-1}} \langle R_0 (s_1 - s_0) \rangle + A^{-1} \Delta_{\eta_0}^{R_0} \langle s_1 - s_0 \rangle \tag{C.26}$$

and,

$$\frac{\partial \tau^{(i)}}{\partial \eta_1} = \Delta_{\eta_1}^{A^{-1}} \langle R_0 (s_1 - s_0) \rangle \tag{C.27}$$

C.3.3 Derivatives with respect to the position vectors

Since $u^{(i)}$, $v^{(i)}$ and $w^{(i)}$ do not depend on the baseline, it follows that their derivatives with respect to the position vectors will be zero. It is therefore straightforward to obtain the Jacobian of $\tau^{(i)}$ with respect to s_0 and s_1:

$$\frac{\partial \tau^{(i)}}{\partial s_0} = -A^{-1}R_0 \tag{C.28}$$

and,

$$\frac{\partial \tau^{(i)}}{\partial s_1} = A^{-1}R_0 \tag{C.29}$$

C.3.4 Derivatives of the triangulated point

The Jacobian of $M^{(i)}$ can now be recovered from equation (C.14) using the derivatives of $\tau^{(i)}$. The vector $u^{(i)}$ is a constant and therefore does not depend on η_0, η_1, s_0 and s_1, while $w^{(i)}$ depends only on relative orientation.

The derivative of $M^{(i)}$ with respect to camera orientation in the first frame is,

$$\frac{\partial M^{(i)}}{\partial \eta_0} = \left(u^{(i)}[1 \quad 0 \quad 0] - \frac{1}{2}w^{(i)}[0 \quad 0 \quad 1]\right)\frac{\partial \tau^{(i)}}{\partial \eta_0} - \frac{1}{2}[0 \quad 0 \quad 1]\tau^{(i)}\frac{\partial w^{(i)}}{\partial \eta_0} \tag{C.30}$$

Similarly, the derivative with respect to η_1 is,

$$\frac{\partial M^{(i)}}{\partial \eta_1} = \left(u^{(i)}[1 \quad 0 \quad 0] - \frac{1}{2}w^{(i)}[0 \quad 0 \quad 1]\right)\frac{\partial \tau^{(i)}}{\partial \eta_1} - \frac{1}{2}[0 \quad 0 \quad 1]\tau^{(i)}\frac{\partial w^{(i)}}{\partial \eta_1} \tag{C.31}$$

In terms of the position vectors s_0 and s_1, the derivatives will be slightly different, provided that $u^{(i)}$ and $w^{(i)}$ do not depend on the baseline:

$$\frac{\partial M^{(i)}}{\partial s_0} = \left(u^{(i)}[1 \quad 0 \quad 0] - \frac{1}{2}w^{(i)}[0 \quad 0 \quad 1]\right)\frac{\partial \tau^{(i)}}{\partial s_0} \tag{C.32}$$

and,

$$\frac{\partial M^{(i)}}{\partial s_1} = \left(u^{(i)}[1 \quad 0 \quad 0] - \frac{1}{2}w^{(i)}[0 \quad 0 \quad 1]\right)\frac{\partial \tau^{(i)}}{\partial s_1} \tag{C.33}$$

Appendix D

Properties of the Euclidean epipolar constraint and scene reconstruction from two views

D.1 Disambiguation

The essential matrix can be written as the product of an orthonormal matrix and a cross-product skew symmetric matrix. This product has several different manifestations in literature, depending on how one interprets geometrically the orthonormal matrix and the cross product vector. In this analysis, to avoid ambiguities and in order to be consistent with the contents of this thesis as well as the given formulas on the properties of the essential matrix and scene reconstruction, the following formalism is used:

$$E = R^T[b]_\times \tag{D.1}$$

where R contains the three unit vectors comprising the second camera coordinate frame arranged column-wise and b is the baseline vector expressed in the first camera frame. To motivate the above formalism, a quick overview on how local coordinates transform from one camera frame to another in regards to the orthonormal matrix R.

D.1.2 Moving between coordinate frames

The goal of this section is to derive a formula for the coordinates of a 3D point *M* in a given camera frame located at a location *b*, given the orientation of the frame as three unit vectors (u_1, u_2, u_3), where b and u_1, u_2, u_3 are expressed in terms of the first camera frame (w_1, w_2, w_3) which is chosen for global reference. Figure D.1 illustrates the two frames, the baseline and the vectors that connect the two camera centers with the point *M*.

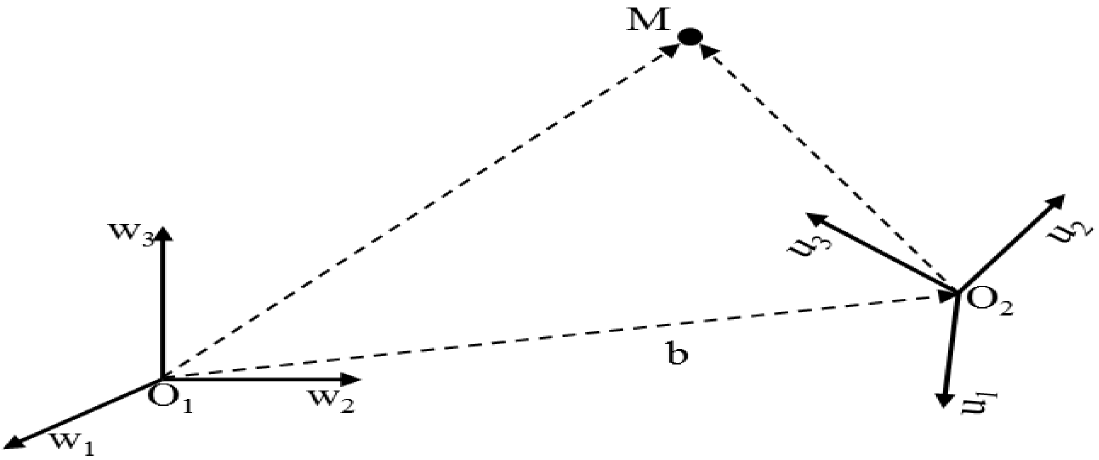

Figure D.1. The two camera frames; the baseline and the vectors that connect the two camera centers O_1 and O_2 with *M* are indicated with dashed lines.

Let now $M_1 = M$ be the coordinates of M in the first camera frame and M_2 the respective coordinates in the second camera frame. Also, let $(O_2M)_1$ be the vector O_2M expressed in the coordinate frame of the first camera. Then, the coordinates of M_2 will be the projections of $(O_2M)_1$ on the direction vectors u_1, u_2, u_3:

$$M_2 = \begin{bmatrix} u_1^T(O_2M)_1 \\ u_2^T(O_2M)_1 \\ u_3^T(O_2M)_1 \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ u_3^T \end{bmatrix} (O_2M)_1 = [u_1 \quad u_2 \quad u_3]^T (O_2M)_1 = R^T(O_2M)_1 \qquad \text{(D.2)}$$

But $(O_2M)_1$ can be written as the difference of vectors $(O_1M)_1 = M_1$ and *b*:

$$(O_2M)_1 = M - b \qquad \text{(D.3)}$$

Substituting from (D.3) into (D.2) yields:

$$M_2 = R^T(M_1 - b) \qquad \text{(D.4)}$$

D.2 Properties of the essential matrix

The most important properties of the essential matrix are listed in this section. These properties are very useful either as constraints or as supplementary formulas in the course of scene structure and relative camera pose estimation.

Lemma D.1. *Suppose E is an essential matrix. Then, the epipoles e_1 and e_2 are the left and right null spaces of E respectively.*

Proof. Since the epipoles are the scaled images of the camera centers, then there exist λ_1 and λ_2 such that $e_2 = -\lambda_2 R^T b$ and $e_1 = \lambda_1 b$. By the expression of the essential matrix in equation (D.1), $E = R^T[b]_\times$. Therefore:

$$e_2^T E = -\lambda_2 (R^T b)^T R^T [b]_\times = -\lambda_2 b^T \underbrace{R R^T}_{I_3} [b]_\times = -\lambda_2 \underbrace{b^T [b]_\times}_{-b \times b = 0} = 0 \tag{D.5}$$

Also,

$$E e_1 = \lambda_2 R^T \underbrace{[b]_\times b}_{b \times b = 0} = 0 \tag{D.6}$$

Lemma D.2. *Any point m_1, m_2 has an associated epipolar line l_1, l_2 in the opposite view given by:*

$$l_2 = E m_1 = e_2 \times m_2 \qquad l_1 = E m_2 = e_1 \times m_1$$

Proof. The proof is a direct consequence of the epipolar constraint for m_1 and m_2.

Lemma *D.3* For any orthonormal matrix $R \in \mathbb{R}^{3\times3}$ and for any vector $a \in \mathbb{R}^3$ the following holds:

$$[Ra]_\times = R[a]_\times R^T \tag{D.7}$$

Proof. Let *R* be an orthonormal matrix. Then by the definition of cross-product,

$$(Ra) \times b = [Ra]_\times b \tag{D.8}$$

Multiplying (D.8) with R^T from the left yields,

$$R^T\big((Ra) \times b\big) = R^T [Ra]_\times b \tag{D.9}$$

We now resort to the following property of the cross product that holds for any linear transformation *M* and any pair of vectors *a*, *b*:

$$(Ma) \times (Mb) = M(a \times b) \tag{D.10}$$

Making use of the cross product property in (D.10), equation (D.9) becomes:

$$(R^T R) a \times (R^T b) = R^T [Ra]_\times b$$

$$\Leftrightarrow a \times (R^T b) = R^T [Ra]_\times b$$

$$\Leftrightarrow [a]_\times R^T b = R^T [Ra]_\times b \;\Leftrightarrow ([a]_\times R^T - R^T [Ra]_\times) b = 0$$

$$\Leftrightarrow [a]_\times = R^T[Ra]_\times R \tag{D.11}$$

A very significant theorem that provides a simple but hard criterion for the existence of an essential matrix is the following (Faugeras, Luong et al. 2004).

Theorem D.1. *A $3x3$ matrix E is an essential matrix if and only if it has a singular value decomposition $E = USV^T$ such that:*

$$S = diag\{\sigma, \sigma, 0\}, \qquad \sigma > 0$$

where U, V are orthonormal matrices.

Proof. Let *E* be an essential matrix. Then, *E* is given by, $E = R^T[b]_\times$. Taking E^TE yields,

$$E^TE = (R^T[b]_\times)^T R^T[b]_\times = [b]_\times{}^T RR^TR^T[b]_\times = [b]_\times{}^T[b]_\times \tag{D.12}$$

Let now *R₀* be the rotation that aligns the baseline vector with the *z*-axis, that is, $R_0b = [0 \quad 0 \quad ||b||]^T$. Then, using lemma D.3, it follows that $[a]_\times = R_0^T[R_0b]_\times R_0$. Substituting in (D.7) yields,

$$E^TE = \left(R_0{}^T[R_0a]_\times R_0\right)^T R_0{}^T[R_0a]_\times R_0 = R_0{}^T[R_0a]_\times{}^T[R_0a]_\times[R_0a]_\times R_0$$

$$\Leftrightarrow E^TE = R_0{}^T\begin{bmatrix} 0 & ||b|| & 0 \\ -||b|| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & -||b|| & 0 \\ ||b|| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0$$

$$\Leftrightarrow E^TE = R_0{}^T\begin{bmatrix} ||b||^2 & 0 & 0 \\ 0 & ||b||^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0 \tag{D.13}$$

The decomposition of equation (D.13) is a SVD (one out many) of E^TE. Hence, *E* will also decompose as follows:

$$E = U\begin{bmatrix} ||b|| & 0 & 0 \\ 0 & ||b|| & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0 \tag{D.14}$$

for some orthonormal matrix *U*.

Let now *E* be a $3x3$ matrix with two exactly non-zero singular values which are equal. In this case, it is easier to proceed by gradually constructing the sought result. The decomposition of the matrix *E* is,

$$E = U\begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix}V^T = USV^T \tag{D.15}$$

where *U* and *V* are orthonormal matrices. The construction that follows is relying on the observation that *S* can be written in the following way:

$$S = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_z\left(-\frac{\pi}{2}\right)SR_z\left(-\frac{\pi}{2}\right)^T \tag{D.16}$$

or,

$$S = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_z\left(\frac{\pi}{2}\right)SR_z\left(\frac{\pi}{2}\right)^T \tag{D.17}$$

where $R_z\left(-\frac{\pi}{2}\right)$ and $R_z\left(\frac{\pi}{2}\right)$ are rotations by $\pm\pi/2$ about the *z* axis. We observed that the product of *S* with the preceding or following rotation yields a skew symmetric matrix:

$$S = \underbrace{\left(R_z\left(\frac{\pi}{2}\right)S\right)}_{skew\ symmetric} R_z\left(\frac{\pi}{2}\right)^T = \begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}R_z\left(\frac{\pi}{2}\right)^T \tag{D.18}$$

or,

$$S = R_z\left(-\frac{\pi}{2}\right)\underbrace{\left(SR_z\left(-\frac{\pi}{2}\right)^T\right)}_{skew\ symmetric} = \begin{bmatrix} 0 & \sigma & 0 \\ -\sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}R_z\left(-\frac{\pi}{2}\right)^T \tag{D.19}$$

Now, the property of skew symmetric matrices in lemma D.3 can be of great use in a heuristic sense. In particular, it guarantees that for any rotation matrix *U* and for any skew symmetric matrix $S_\times$, the matrix $US_\times U^T$ is also a skew symmetric matrix. In the light of this consequence and choosing (D.18), the SVD of *E* can be expressed as follows:

$$\begin{aligned} E &= UR_z\left(-\frac{\pi}{2}\right)\begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}V^T \\ &= U\underbrace{R_z\left(-\frac{\pi}{2}\right)^T V^T V R_z\left(-\frac{\pi}{2}\right)}_{equal\ to\ identity}\begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}R_z\left(-\frac{\pi}{2}\right)^T V^T \end{aligned}$$

$$\Leftrightarrow E = \underbrace{\left(UR_z\left(-\frac{\pi}{2}\right)^T V^T\right)}_{a\ rotation\ matrix}\underbrace{\left(VR_z\left(-\frac{\pi}{2}\right)\right)\begin{bmatrix}0 & -\sigma & 0\\ \sigma & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\left(VR_z\left(-\frac{\pi}{2}\right)\right)^T}_{a\ skew\ symmetric\ matrix\ by\ virtue\ of\ lemma\ D.3} \quad \text{(D.20)}$$

In the very same way, one arrives at a similar result starting from (D.19):

$$E = \underbrace{\left(UR_z\left(\frac{\pi}{2}\right)^T V^T\right)}_{a\ rotation\ matrix}\ \underbrace{\left(VR_z\left(\frac{\pi}{2}\right)\right)\begin{bmatrix}0 & \sigma & 0\\ -\sigma & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\left(VR_z\left(\frac{\pi}{2}\right)\right)^T}_{a\ skew\ symmetric\ matrix\ by\ virtue\ of\ lemma\ D.3} \quad \text{(D.21)}$$

The only remaining “loose end” now is to show that matrices $UR_z\left(-\frac{\pi}{2}\right)^T V^T$ and $UR_z\left(\frac{\pi}{2}\right)^T V^T$ are indeed rotation matrices. Since the product involves both U and V, then the sign of the product of their determinants should be positive due to the fact that these matrices participate in the SVD of $E^T E$ which always has a positive determinant. And since the determinant of R_z is also positive, it follows that the two aforementioned products are orthonormal matrices with positive determinants, hence rotations. And that concludes the proof.

Lemma D.4. *If E is an essential matrix such that $E = R^T[b]_\times$, then:*

$$Tr(EE^T) = Tr(E^T E) = 2\|b\|^2$$

Proof. Taking $E^T E$ yields:

$$E^T E = (R^T[b]_\times)^T R^T[b]_\times = -[b]_\times{}^2 \quad \text{(D.22)}$$

If now $b = [b_1 \quad b_2 \quad b_3]^T$, then,

$$E^T E = -\begin{bmatrix}0 & -b_3 & b_2\\ b_3 & 0 & -b_1\\ -b_2 & b_1 & 0\end{bmatrix}^2 = \begin{bmatrix}b_2{}^2 + b_3{}^2 & -b_1 b_2 & -b_1 b_3\\ -b_1 b_2 & b_1{}^2 + b_3{}^2 & -b_2 b_3\\ -b_1 b_3 & -b_2 b_3 & b_1{}^2 + b_2{}^2\end{bmatrix} \quad \text{(D.23)}$$

From (D.23) it is clear that $Tr(E^T E) = 2(b_1^2 + b_2^2 + b_3^2) = 2\|b\|^2$.

In the case of EE^T, the following is obtained:

$$EE^T = R^T[b]_\times (R^T[b]_\times)^T = -R^T[b]_\times{}^2 R$$

$$\Leftrightarrow EE^T = (R^T[b]_\times R)(R^T[b]_\times R) \quad \text{(D.24)}$$

Once again, lemma D.3 states that $[b]_\times = R^T[Rb]_\times R$ for any orthogonal matrix R. Hence, (D.24) becomes:

$$EE^T = \left(R^T \underbrace{(R[b]_\times R^T)}_{[b]_\times} R\right)\left(R^T \underbrace{(R[b]_\times R^T)}_{[b]_\times} R\right) = -[b]_\times{}^2 \tag{D.25}$$

And since $\|R^T b\| = \|b\|$, it follows from (D.18) that $Tr(EE^T) = 2\|b\|^2$.

Theorem D.5. A 3×3 non-zero matrix E is an essential matrix if and only if the following relationship holds:

$$EE^T E = \frac{Tr(E^T E)}{2} E$$

Proof. Proving that the validity of the relationship implies that E is an essential matrix could be done through its SVD. Let $E = USV^T$ where U, V^T are orthonormal matrices and S is a diagonal matrix with positive entries. Substituting in the given relationship, yields:

$$EE^T E - \frac{Tr(EE^T)}{2} E = US^2U^TUSV^T - \frac{Tr(EE^T)}{2} USV^T = 0$$

$$\Leftrightarrow S^2 = \frac{Tr(EE^T)}{2} S \tag{D.26}$$

Let s_1, s_2, s_2 be the singular values of S. Then, the trace of EE^T should be equal to the sum of the squared singular values:

$$Tr(EE^T) = s_1^2 + s_2^2 + s_3^2 \tag{D.27}$$

It follows from (D.26) and (D.27) that,

$$\begin{aligned} 2s_1^3 &= (s_1^2 + s_2^2 + s_3^2)s_1 \\ 2s_2^3 &= (s_1^2 + s_2^2 + s_3^2)s_2 \\ 2s_3^3 &= (s_1^2 + s_2^2 + s_3^2)s_1 \end{aligned} \tag{D.28}$$

Since E is non-zero, one singular value must be strictly positive. Without constraining generality, let s_1 be strictly positive. It follows that,

$$s_1^2 = \frac{(s_1^2 + s_2^2 + s_3^2)}{2} \Leftrightarrow s_1^2 = s_1^2 + s_2^2 \tag{D.29}$$

Substituting in the expression for the second singular value in (D.23) yields:

$$s_2^3 = (s_2^2 + s_3^2)s_2 \Leftrightarrow s_2 s_3^2 = 0 \tag{D.30}$$

It follows that either s_2 or s_3 is zero. If they are both zero, then so is s_1 which is a contradiction. Exactly 2 of the 3 singular values are non-zero and have the same value, $s_1 = s_2 = \frac{Tr(EE^T)}{2}$.

Consider now the opposite direction case in which we know that E is an essential matrix. Taking the given relationship and substituting from (D.22) and using the fact that $[b]_\times^2 = bb^T - \|b\|^2 I$, we have:

$$\begin{aligned} E \underbrace{E^T E}_{-[b]_\times^2} &= -E \underbrace{[b]_\times^2}_{bb^T - \|b\|^2 I} = \|b\|^2 E - \underbrace{E}_{R^T[b]_\times} bb^T = \|b\|^2 E - R^T \underbrace{[b]_\times b}_{0_{3\times1}} b^T \\ &= \underbrace{\|b\|^2}_{\frac{Tr(E^TE)}{2}} E = \frac{Tr(E^T E)}{2} E \end{aligned} \tag{D.31}$$

where $0_{3\times1}$ is the zero 3×1 vector. And that concludes the proof in the opposite direction.

D.3 Extracting relative pose from the essential matrix

The method for relative pose extraction detailed in this section is loosely based on the brilliant observation by Berthold Horn (Horn 1990) that the matrix of cofactors of an essential matrix can be expressed in terms of the rotation matrix, the skew symmetric matrix of the baseline and the essential matrix itself.

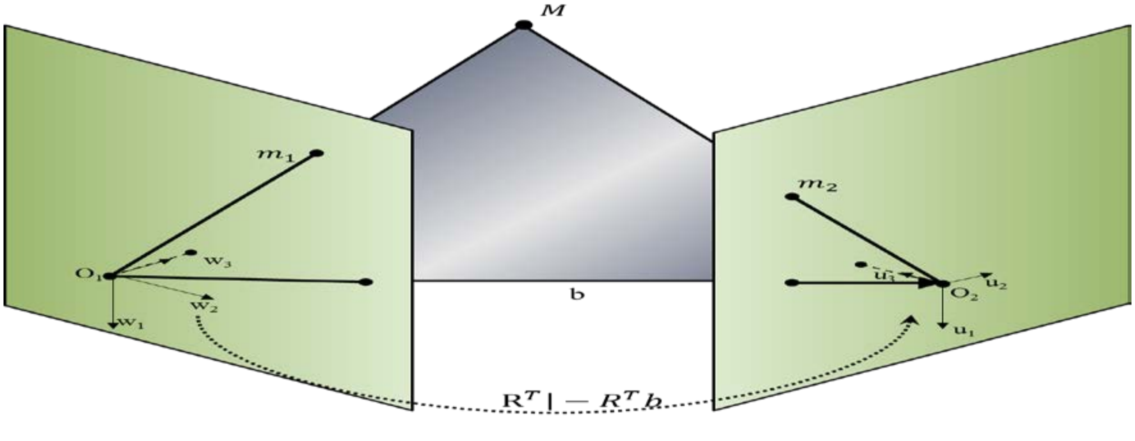

Figure D.1. The geometry induced by the projections of a 3D point *M* in two camera views.

Consider a 3D point *M* and its normalized Euclidean projections m_1 and m_2 in two views as show in Figure D.1. With the essential matrix in place, the next step is to obtain the rotation matrix R and the unit-length baseline vector b. As a first step, from theorem D.1, scale can be removed from the essential matrix by dividing it with $\|b\|$. Also, lemma D.4 states that $Tr(E^T E) = 2\|b\|^2$; thus, a “normalized” essential matrix E_n is obtained as follows:

$$E_n = \frac{E}{\sqrt{\frac{Tr(E^T E)}{2}}} \tag{D.32}$$

D.3.1 Baseline

From lemma D.4 it is easy to extract the absolute values of the baseline components as follows:

$$|b_1| = \sqrt{1 - [E_n^T E_n]_{11}} \tag{D.33}$$

$$|b_2| = \sqrt{1 - [E_n^T E_n]_{22}} \tag{D.34}$$

$$|b_3| = \sqrt{1 - [E_n^T E_n]_{33}} \tag{D.35}$$

where $[E_n^T E_n]_{ij}$ denotes the element of $E_n^T E_n$ in the *i*^th row and *j*^th column. To resolve the sign ambiguity, the largest squared component is assumed to be a positive square root and the remaining signs are inferred from the off-diagonal elements of $E_n^T E_n$:

$$E_n^T E_n = \begin{bmatrix} b_2^2 + b_3^2 & -b_1 b_2 & -b_1 b_3 \\ -b_1 b_2 & b_1^2 + b_3^2 & -b_2 b_3 \\ -b_1 b_3 & -b_2 b_3 & b_1^2 + b_2^2 \end{bmatrix} \tag{D.36}$$

It suffices to recover one baseline vector from $E_n^T E_n$ as described above, as the second baseline will simply be a vector of opposite direction.

D.3.2 Orientation

Recovering the rotation matrix requires slightly more elaborate pre-processing. Consider the matrix of cofactors of E_n:

$$C_n = \begin{bmatrix} \begin{vmatrix} e_{22} & e_{23} \\ e_{32} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{21} & e_{23} \\ e_{31} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{21} & e_{22} \\ e_{31} & e_{32} \end{vmatrix} \\ -\begin{vmatrix} e_{12} & e_{13} \\ e_{32} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{13} \\ e_{31} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{12} \\ e_{31} & e_{32} \end{vmatrix} \\ \begin{vmatrix} e_{12} & e_{13} \\ e_{22} & e_{23} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{13} \\ e_{21} & e_{23} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{vmatrix} \end{bmatrix} \tag{D.37}$$

Standard tensor notation is adopted to denote matrix rows and columns as well as elements for the following derivations. Thus, for instance, e_j^i is the element of E_n in the *i*th row and the *j*th column. Also, e^i is the *i*th row of E_n as a 1×3 vector, while e_j is the *j*th column as a 3×1 vector. With notation in place, we observe that C_n can be written as follows:

$$C_n = \begin{bmatrix} ((e^2)^T \times (e^3)^T)^T \\ ((e^3)^T \times (e^1)^T)^T \\ ((e^1)^T \times (e^2)^T)^T \end{bmatrix} \tag{D.38}$$

Also, E_n can be expressed in terms of cross-products as follows:

$$E_n = R^T [b]_\times = \begin{bmatrix} {r_1}^T \\ {r_2}^T \\ {r_3}^T \end{bmatrix} [b]_\times = \begin{bmatrix} {r_1}^T [b]_\times \\ {r_2}^T [b]_\times \\ {r_3}^T [b]_\times \end{bmatrix} = \begin{bmatrix} -([b]_\times r_1)^T \\ -([b]_\times r_2)^T \\ -([b]_\times r_3)^T \end{bmatrix} = \begin{bmatrix} -(b \times r_1)^T \\ -(b \times r_2)^T \\ -(b \times r_3)^T \end{bmatrix} \tag{D.39}$$

Substituting from (D.39) in (D.38) yields triple products in the rows of C_n; applying the well-known *triple product expansion* formula leaves cross product expression only between the columns of *R* (intermediate result) which can also be eliminated from the expression due to orthonormality (final expression on the right):

$$C_n = \begin{bmatrix} \big((b \times r_2) \times (b \times r_3)\big)^T \\ \big((b \times r_3) \times (b \times r_1)\big)^T \\ \big((b \times r_1) \times (b \times r_2)\big)^T \end{bmatrix} = \begin{bmatrix} \Big(b \cdot \underbrace{(r_2 \times r_3)}_{r_1}\Big) b^T \\ \Big(b \cdot \underbrace{(r_3 \times r_1)}_{r_2}\Big) b^T \\ \Big(b \cdot \underbrace{(r_1 \times r_2)}_{r_3}\Big) b^T \end{bmatrix} = \begin{bmatrix} (b \cdot r_1) b^T \\ (b \cdot r_2) b^T \\ (b \cdot r_3) b^T \end{bmatrix} \tag{D.40}$$

It is now easy to re-arrange the inner products in (D.35) in order to factor-out the columns of *R*:

$$C_n = \begin{bmatrix} ({r_1}^T b) b^T \\ ({r_2}^T b) b^T \\ ({r_3}^T b) b^T \end{bmatrix} = \begin{bmatrix} {r_1}^T (b b^T) \\ {r_2}^T (b b^T) \\ {r_3}^T (b b^T) \end{bmatrix} = R^T (b b^T) \tag{D.41}$$

And a well-known skew symmetric matrix property is,

$$bb^T = I_3 + [b]_\times^2 \tag{D.42}$$

Substituting (D.42) into (D.41) yields:

$$C_n = R^T\big(I_3 + [b]_\times^2\big) = R^T + \underbrace{(R^T[b]_\times)}_{E_n}[b]_\times = R^T + E_n[b]_\times \Leftrightarrow R = C_n^T + [b]_\times E_n^T \tag{D.43}$$

And since b is sign-ambiguous, it follows that there exist two possible rotation matrices and can be obtaining by flipping the sign of $[b]_\times$ in (D.43):

$$R = C_n^T \pm [b]_\times E_n^T \tag{D.44}$$

Bibliography

Achtelik, M., et al. (2012). SFly: Swarm of micro flying robots. Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE.

Achtelik, M. W., et al. (2012). Visual-inertial SLAM for a small helicopter in large outdoor environments. Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE.

Babbar, G., et al. (2010). "Comparative study of image matching algorithms." International Journal of Information Technology and Knowledge Management **2**(2): 337-339.

Baumela, L., et al. (2000). Motion estimation using the differential epipolar equation, IEEE.

Bay, H., et al. (2006). "Surf: Speeded up robust features." Computer Vision–ECCV 2006: 404-417.

Bell, B. M. and F. W. Cathey (1993). "The iterated Kalman filter update as a Gauss-Newton method." Automatic Control, IEEE Transactions on **38**(2): 294-297.

Bertsekas, D. (1979). "Convexification procedures and decomposition methods for nonconvex optimization problems." Journal of Optimization Theory and Applications **29**(2): 169-197.

Black, H. D. (1964). "A passive system for determining the attitude of a satellite." AIAA journal **2**(7): 1350-1351.

Bookstein, F. L. (1989). "Principal warps: Thin-plate splines and the decomposition of deformations." IEEE Transactions on Pattern Analysis & Machine Intelligence(6): 567-585.

Bouguet, J. Y. (2001). Pyramidal implementation of the affine lucas kanade feature tracker—description of the algorithm, Technical report). Intel Corporation.

Bradski, G. (2000). "The opencv library." Doctor Dobbs Journal **25**(11): 120-126.

Bradski, G. and A. Kaehler (2008). Learning OpenCV: Computer vision with the OpenCV library, O'Reilly Media.

Caron, F., et al. (2006). "GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects." Information Fusion **7**(2): 221-230.

Chum, O., et al. (2005). Two-view geometry estimation unaffected by a dominant plane. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, IEEE.

Davenport, P. B. (1968). A vector approach to the algebra of rotations with applications, National Aeronautics and Space Administration.

Davenport, P. B. (1971). "Attitude determination and sensor alignment via weighted least squares affine transformations."

Davison, A. J., et al. (2007). "MonoSLAM: Real-time single camera SLAM." Pattern Analysis and Machine Intelligence, IEEE Transactions on **29**(6): 1052-1067.

Dennis Jr, J. E. and R. B. Schnabel (1996). Numerical methods for unconstrained optimization and nonlinear equations, Siam.

Diebel, J. (2006). "Representing attitude: Euler angles, unit quaternions, and rotation vectors." Matrix.

Dissanayake, M. G., et al. (2001). "A solution to the simultaneous localization and map building (SLAM) problem." Robotics and Automation, IEEE Transactions on **17**(3): 229-241.

Duane, C. B. (1971). "Close-range camera calibration." Photogramm. Eng **37**: 855-866.

Dunkley, O., et al. "Visual-Inertial Navigation for a Camera-Equipped 25g Nano-Quadrotor." Technical University, Munich.

Emgu, C. (2013). Emgu CV: OpenCV in .NET (C#, VB, C++ and more), Recuperado.

Engel, J., et al. (2014). "Scale-aware navigation of a low-cost quadrocopter with a monocular camera." Robotics and Autonomous Systems.

Faessler, M., et al. (2015). "Autonomous, Vision-based Flight and Live Dense 3D Mapping with a Quadrotor Micro Aerial Vehicle." Journal of Field Robotics.

Farnebäck, G. and C. F. Westin (2006). "Affine and deformable registration based on polynomial expansion." Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006: 857-864.

Faugeras, O. (1993). Three-dimensional computer vision: a geometric viewpoint, the MIT Press.

Faugeras, O., et al. (2004). The geometry of multiple images: the laws that govern the formation of multiple images of a scene and some of their applications, MIT press.

Fidaleo, D. and G. Medioni (2007). Model-assisted 3d face reconstruction from video. Analysis and modeling of faces and gestures, Springer**:** 124-138.

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

Fitzgibbon, A. W. (2003). "Robust registration of 2D and 3D point sets." Image and Vision Computing **21**(13): 1145-1153.

Furgale, P., et al. (2012). "The Devon Island rover navigation dataset." The International Journal of Robotics Research: 0278364911433135.

Golovanov, N. (2014). Geometric modeling.

Gong, Y., et al. (2015). "Bound constrained bundle adjustment for reliable 3D reconstruction." Optics express **23**(8): 10771-10785.

Goodwin, H. (1910). The Haversine in Nautical Astronomy. US Naval Institute Proceedings.

Grunert, J. A. (1841). "Das pothenotische problem in erweiterter gestalt nebst über seine anwendungen in der geodäsie." Grunerts archiv für mathematik und physik **1**: 238-248.

Guide, M. U. s. (1998). "The mathworks." Inc., Natick, MA **5**: 333.

Haralick, B. M., et al. (1994). "Review and analysis of solutions of the three point perspective pose estimation problem." International Journal of Computer Vision **13**(3): 331-356.

Harris, C. and M. Stephens (1988). A combined corner and edge detector. Alvey vision conference, Manchester, UK.

Hartley, R. and A. Zisserman (2003). Multiple view geometry in computer vision, Cambridge University Press.

Hartley, R. I. (1995). In defence of the 8-point algorithm, IEEE.

Hartley, R. I. and P. Sturm (1997). "Triangulation." Computer vision and image understanding **68**(2): 146-157.

Hesch, J. A. and S. I. Roumeliotis (2011). A direct least-squares (DLS) method for PnP. Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE.

Hoaglin, D. C., et al. (1983). Understanding robust and exploratory data analysis, Wiley New York.

Horn, B. K., et al. (1988). "Closed-form solution of absolute orientation using orthonormal matrices." JOSA A **5**(7): 1127-1135.

Horn, B. K. P. (1987). "Closed-form solution of absolute orientation using unit quaternions." JOSA A **4**(4): 629-642.

Horn, B. K. P. (1990). "Recovering baseline and orientation from essential matrix." Journal of the Optical Society of America.

Horn, B. K. P. and B. G. Schunck (1981). "Determining optical flow." Artificial intelligence **17**(1-3): 185-203.

Huber, P. J. (2011). Robust statistics, Springer.

Irani, M., et al. (1997). "Recovery of ego-motion using region alignment." Pattern Analysis and Machine Intelligence, IEEE Transactions on **19**(3): 268-272.

Jia, C.-x. and D.-t. Zhu (2011). "Projected gradient trust-region method for solving nonlinear systems with convex constraints." Applied Mathematics-A Journal of Chinese Universities **26**(1): 57-69.

Kalman, R. E. (1960). "A new approach to linear filtering and prediction problems." Journal of basic Engineering **82**(1): 35-45.

Kanatani, K. (1998). "Geometric information criterion for model selection." International Journal of Computer Vision **26**(3): 171-189.

Kanatani, K., et al. (2008). Triangulation from two views revisited: Hartley-Sturm vs. optimal correction, Citeseer.

Kanzow, C., et al. (2002). Levenberg-Marquardt methods for constrained nonlinear equations with strong local convergence properties, Citeseer.

Kanzow, C., et al. (2004). "Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints." Journal of Computational and Applied Mathematics **172**(2): 375-397.

Klein, G. and D. Murray (2007). Parallel tracking and mapping for small AR workspaces. Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on, IEEE.

Klein, G. and D. Murray (2009). PTAM: Parallel tracking and mapping on a camera phone. Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on, IEEE.

Kneip, L., et al. (2011). Robust real-time visual odometry with a single camera and an imu. BMVC.

Kneip, L., et al. (2011). A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE.

Koller, D. and N. Friedman (2009). Probabilistic graphical models: principles and techniques, MIT press.

Konolige, K., et al. (2011). "Large-scale visual odometry for rough terrain." Robotics Research: 201-212.

Kristof, W. and B. Wingersky (1971). A generalization of the orthogonal Procrustes rotation procedure to more than two matrices. Proceedings of the Annual Convention of the American Psychological Association, American Psychological Association.

Kuipers, J. B. (1999). Quaternions and rotation sequences, Princeton university press Princeton, NJ, USA:.

Kukelova, Z., et al. (2008). Polynomial Eigenvalue Solutions to the 5-pt and 6-pt Relative Pose Problems. BMVC.

Lazard, D. (1983). Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. Computer algebra, Springer**:** 146-156.

Lentaris, G., et al. (2015). "HW/SW co-design and FPGA acceleration of visual odometry algorithms for rover navigation on Mars."

Lepetit, V., et al. (2009). "Epnp: An accurate o (n) solution to the pnp problem." International Journal of Computer Vision **81**(2): 155-166.

Levenberg, K. (1944). "A method for the solution of certain problems in least squares." Quarterly of applied mathematics **2**: 164-168.

Li, H. and R. Hartley (2006). Five-point motion estimation made easy. Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, IEEE.

Longuet-Higgins, H. (1987). "A computer algorithm for reconstructing a scene from two projections." Readings in computer vision: issues, problems, principles, and paradigms: 61.

Lourakis, M. and A. A. Argyros (2005). Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment?, IEEE.

Lourakis, M. and X. Zabulis (2013). Model-based pose estimation for rigid objects. Computer Vision Systems, Springer**:** 83-92.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. Computer vision, 1999. The proceedings of the seventh IEEE international conference on, Ieee.

Lucas, B. D. and T. Kanade (1981). An iterative image registration technique with an application to stereo vision. IJCAI.

Lui, V. and T. Drummond (2007). "An iterative 5-pt algorithm for fast and robust essential matrix estimation." IJCV **74**(2): 117-136.

Luong, Q. T. and O. D. Faugeras (1996). "The fundamental matrix: Theory, algorithms, and stability analysis." International Journal of Computer Vision **17**(1): 43-75.

Ma, Y., et al. An Invitation to 3D Vision. 2004, Springer.

Maimone, M., et al. (2007). "Two years of visual odometry on the mars exploration rovers." Journal of Field Robotics **24**(3): 169-186.

Markley, F. L. and D. Mortari (1999). "How to estimate attitude from vector observations."

Maronna, R. A., et al. (2006). "Robust Statistics: Theory and Methods. 2006." J. Wiley.

Marquardt, D. W. (1963). "An algorithm for least-squares estimation of nonlinear parameters." Journal of the Society for Industrial & Applied Mathematics **11**(2): 431-441.

Mirzaei, F. M. and S. I. Roumeliotis (2008). "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation." Robotics, IEEE Transactions on **24**(5): 1143-1156.

Montemerlo, M., et al. (2002). FastSLAM: A factored solution to the simultaneous localization and mapping problem. AAAI/IAAI.

Mouragnon, E., et al. (2009). "Generic and real-time structure from motion using local bundle adjustment." Image and Vision Computing **27**(8): 1178-1193.

Naeem, W., et al. (2010). "Design of an unmanned surface vehicle for environmental monitoring."

Neugebauer, P. J. and K. Klein (1999). Texturing 3d models of real world objects from multiple unregistered photographic views. Computer Graphics Forum, Wiley Online Library.

Niem, W. and R. Buschmann (1995). Automatic modelling of 3D natural objects from multiple views. Image Processing for Broadcast and Video Production, Springer**:** 181-193.

Nistér, D. (2004). "An efficient solution to the five-point relative pose problem." Pattern Analysis and Machine Intelligence, IEEE Transactions on **26**(6): 756-770.

Nistér, D., et al. (2004). Visual odometry. Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, IEEE.

Nocedal, J. and S. Wright (2006). Numerical optimization, Springer Science & Business Media.

Papadopoulo, T. and M. I. Lourakis (2000). Estimating the jacobian of the singular value decomposition: Theory and applications. Computer Vision-ECCV 2000, Springer**:** 554-570.

Paragios, N., et al. (2006). Handbook of mathematical models in computer vision, Springer Science & Business Media.

Pollefeys, M., et al. (2004). "Visual modeling with a hand-held camera." International Journal of Computer Vision **59**(3): 207-232.

Qian, G. and R. Chellappa (2001). "Structure from motion using sequential monte carlo methods."

Qian, G., et al. (2001). "Robust structure from motion estimation using inertial data." JOSA A **18**(12): 2982-2997.

Rosten, E. (2013). TooN

Rosten, E. and T. Drummond (2006). Machine learning for high-speed corner detection. Computer Vision–ECCV 2006, Springer**:** 430-443.

Scaramuzza, D. and F. Fraundorfer (2011). "Visual odometry [tutorial]." Robotics & Automation Magazine, IEEE **18**(4): 80-92.

Scaramuzza, D., et al. (2009). Real-time monocular visual odometry for on-road vehicles with 1-point ransac, Ieee.

Schmidt, J. and H. Niemann (2001). Using quaternions for parametrizing 3–D rotations in unconstrained nonlinear optimization.

Schönemann, P. H. (1966). "A generalized solution of the orthogonal Procrustes problem." Psychometrika **31**(1): 1-10.

Shi, J. and C. Tomasi (1994). Good features to track. Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on, IEEE.

Smith, R. C. and P. Cheeseman (1986). "On the representation and estimation of spatial uncertainty." The International Journal of Robotics Research **5**(4): 56-68.

Snavely, N., et al. (2006). Photo tourism: exploring photo collections in 3D. ACM Transactions on Graphics (TOG), ACM.

Ten Berge, J. M. (1977). "Orthogonal Procrustes rotation for two or more matrices." Psychometrika **42**(2): 267-276.

Terzakis, G., et al. (2014). "On quaternion based parameterization of orientation in computer vision and robotics." Journal of Engineering Science and Technology Review **7**(1): 82-93.

Terzakis, G., et al. (2015). "Fitting multiple projective models using clustering-based Markov chain Monte Carlo inference." Image and Vision Computing **33**: 15-25.

Thrun, S., et al. (2005). Probabilistic robotics, MIT press Cambridge.

Thrun, S. and J. J. Leonard (2008). "Simultaneous localization and mapping." Springer handbook of robotics: 871-889.

Thrun, S., et al. (2004). "Simultaneous localization and mapping with sparse extended information filters." The International Journal of Robotics Research **23**(7-8): 693-716.

Thrun, S. and M. Montemerlo (2006). "The graph SLAM algorithm with applications to large-scale mapping of urban structures." The International Journal of Robotics Research **25**(5-6): 403-429.

Tomasi, C. and T. Kanade (1992). "Shape and motion from image streams under orthography: a factorization method." International Journal of Computer Vision **9**(2): 137-154.

Torr, P., et al. (1995). Robust detection of degenerate configurations for the fundamental matrix, IEEE.

Torr, P. H., et al. (1998). "Robust detection of degenerate configurations while estimating the fundamental matrix." Computer vision and image understanding **71**(3): 312-333.

Torr, P. H. S. and A. Zisserman (2000). "MLESAC: A new robust estimator with application to estimating image geometry." Computer vision and image understanding **78**(1): 138-156.

Trucco, E. and A. Verri (1998). Introductory techniques for 3-D computer vision, Prentice Hall New Jersey.

Wah, B. and Z. Wu (1999). The theory of discrete Lagrange multipliers for nonlinear discrete optimization, Springer.

Wahba, G. (1965). "A least squares estimate of satellite attitude." SIAM review **7**(3): 409-409.

Wang, W. and C. M. Clark (2007). "Autonomous control for a differential thrust rov."

Wang, Z. (2011). Simultaneous Localization and Mapping: Exactly Sparse Information Filters, World Scientific.

Watt, A. and M. Watt (1992). "Advanced animation and rendering techniques." Addison-Wesley.

Weiss, S., et al. (2012). Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE.

Wendel, J., et al. (2006). "An integrated GPS/MEMS-IMU navigation system for an autonomous helicopter." Aerospace Science and Technology **10**(6): 527-533.

Werner, T. and A. Zisserman (2002). New techniques for automated architectural reconstruction from photographs. Computer Vision—ECCV 2002, Springer**:** 541-555.

Zhang, P., et al. (2005). Navigation with IMU/GPS/digital compass with unscented Kalman filter. Mechatronics and Automation, 2005 IEEE International Conference, IEEE.

Zhang, Z. (1999). Flexible camera calibration by viewing a plane from unknown orientations, Published by the IEEE Computer Society.

Zheng, E. and C. Wu (2015). Structure from Motion Using Structure-less Resection. Proceedings of the IEEE International Conference on Computer Vision.

Zollhöfer, M., et al. (2014). "Real-time Non-rigid Reconstruction using an RGB-D Camera." ACM Transactions on Graphics (TOG) **33**(4): 156.