THE UNIVERSITY OF
SYDNEY

Vision-based Navigation for Autonomous Underwater Vehicles

Ian Mahon

A thesis submitted in fulfillment
of the requirements for the degree of
Doctor of Philosophy

Australian Centre for Field Robotics
School of Aerospace, Mechanical and Mechatronic Engineering
The University of Sydney

Submitted September 2007
Emended July 2008

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.

Abstract

Ian Mahon
The University of Sydney

Doctor of Philosophy
July 2008

Vision-based Navigation for Autonomous Underwater Vehicles

This thesis investigates the use of vision sensors in Autonomous Underwater Vehicle (AUV) navigation, which is typically performed using a combination of dead-reckoning and external acoustic positioning systems. Traditional dead-reckoning sensors such as Doppler Velocity Logs (DVLs) or inertial systems are expensive and result in drifting trajectory estimates. Acoustic positioning systems can be used to correct dead-reckoning drift, however they are time consuming to deploy and have a limited range of operation. Occlusion and multipath problems may also occur when a vehicle operates near the seafloor, particularly in environments such as reefs, ridges and canyons, which are the focus of many AUV applications.

Vision-based navigation approaches have the potential to improve the availability and performance of AUVs in a wide range of applications. Visual odometry may replace expensive dead-reckoning sensors in small and low-cost vehicles. Using onboard cameras to correct dead-reckoning drift will allow AUVs to navigate accurately over long distances, without the limitations of acoustic positioning systems.

This thesis contains three principal contributions. The first is an algorithm to estimate the trajectory of a vehicle by fusing observations from sonar and monocular vision sensors. The second is a stereo-vision motion estimation approach that can be used on its own to provide odometry estimation, or fused with additional sensors in a Simultaneous Localisation And Mapping (SLAM) framework. The third is an efficient SLAM algorithm that uses visual observations to correct drifting trajectory estimates.

Results of this work are presented in simulation and using data collected during several deployments of underwater vehicles in coral reef environments. Trajectory estimation is demonstrated for short transects using the sonar and vision fusion and stereo-vision approaches. Navigation over several kilometres is demonstrated using the SLAM algorithm, where stereo-vision is shown to improve the estimated trajectory produced by a DVL.

Acknowledgements

Firstly I need to thank my supervisor Stefan Williams and the ACFR underwater robotics team: Matthew Johnson-Roberson, Oscar Pizarro and Paul Rigby, who have made submarines work and field trips memorable. I must also thank Nigel Price, who generously supplied his time, and made his house and boat on the Hawkesbury River available as the world's most picturesque submarine testing facility.

Additionally, I owe a thank-you to the SeaBED group at WHOI: Hanumant Singh, Richard Camilli, Ryan Eustice, Chris Roman and Oscar Pizarro (again), who supplied our AUV, and provided the opportunity to join them in Woods Hole and on research cruises to learn how it all works.

Sadly, not all my time as a student has been spent sailing between coral lagoons. Fortunately a great crew of ACFR landlubbers have made the past few years enjoyable. In particular, to Alex Brooks, Alex Green, David Johnson, Tobias Kaupp, Craig Lobsey, Alex Makarenko, George Mathews, Sharon Ong, Fabio Ramos, Ben Upcroft and Rosalind Wang, thanks for the good times.

And finally, I must thank my parents, who have supported me on this unexpectedly long journey through university.

Contents

List of Figures

List of Tables

Notation

Convention

Scalars are represented by regular lowercase symbols
Vectors are represented by bold lowercase symbols
Matrices are represented by bold uppercase symbols

General Geometry

${}^{i}\mathbf{p}_{j}$	Pose of frame j relative to frame i
${}^{i}\mathbf{t}_{j}$	Position of j in frame i
${}^{i}x_{j}$	X-axis coordinate for the position of j in frame i
${}^{i}y_{j}$	Y-axis coordinate for the position of j in frame i
${}^{i}z_{j}$	Z-axis coordinate for the position of j in frame i
${}^{i}\boldsymbol{\psi}_{j}$	Euler angles describing the orientation of j relative to frame i
${}^{i}\phi_{j}$	Roll Euler angle of j relative to frame i
${}^{i}\theta_{j}$	Pitch Euler angle j relative to frame i
${}^{i}\psi_{j}$	Yaw Euler angle j relative to frame i
${}^{i}\mathbf{v}_{j}$	Velocity of j relative to frame i
${}^{i}\dot{x}_{j}$	X-axis velocity of j relative to frame i
${}^{i}\dot{y}_{j}$	Y-axis velocity of j relative to frame i
${}^{i}\dot{z}_{j}$	Z-axis velocity of j relative to frame i
${}^{i}\dot{\boldsymbol{\psi}}_{j}$	Rate of change of Euler angles of j relative to frame i
${}^{i}\dot{\phi}_{j}$	Rate of change of roll Euler angle of j relative to frame i
${}^{i}\dot{\theta}_{j}$	Rate of change of pitch Euler angle of j relative to frame i
${}^{i}\dot{\psi}_{j}$	Rate of change of yaw Euler angle of j relative to frame i

ω_j Local body rotation rates of j
p_j Local roll rate of j
q_j Local pitch rate of j
r_j Local yaw rate of j

${}^j_i\mathbf{R}$ Matrix rotating frame i to be oriented with frame j

${}^j_i\mathbf{E}$ Matrix transforming local body rates of i to Euler rates in frame j

Pose Composition Operations

$\ominus {}^i\mathbf{p}_j$	Pose inverse operation
$\mathbf{J}_\ominus$	Jacobian of the pose inverse operation
${}^i\mathbf{p}_j \oplus {}^j\mathbf{p}_k$	Pose head to tail operation
$\mathbf{J}_\oplus$	Jacobian of the head to tail operation
$\mathbf{J}_{\oplus 1}$	Jacobian of the head to tail operation with respect to the first pose parameter only
$\mathbf{J}_{\oplus 2}$	Jacobian of the head to tail operation with respect to the second pose parameter only
$\ominus {}^i\mathbf{p}_j \oplus {}^i\mathbf{p}_k$	Pose tail to tail operation
${}_\ominus\mathbf{J}_\oplus$	Jacobian of the tail to tail operation

Computer Vision

$\mathbf{c}$ Image coordinates
$\mathbf{l}$ Parameters for a line in an image

$\mathbf{K}$ Camera intrinsic parameter matrix

$\mathbf{F}$ Fundamental matrix relating two views

General Estimation

$\mathbf{x}$	State vector
$\mathbf{P}$	State covariance
$\mathbf{y}$	Information vector
$\mathbf{Y}$	Information matrix
$\mathbf{z}$	Observation
$\hat{\mathbf{z}}$	Predicted observation
$\boldsymbol{\nu}$	Innovation
$\mathbf{S}$	Innovation covariance

Kalman Filter

$\mathbf{W}$	Kalman weights
$\hat{\mathbf{x}}^+(t_{k-1})$	Prior state estimate
$\hat{\mathbf{x}}^-(t_k)$	Prediction state estimate
$\hat{\mathbf{x}}^+(t_k)$	Posterior state estimate
$\mathbf{x}^*(t_k)$	Augmented state vector
$\mathbf{P}^+(t_{k-1})$	Prior state covariance
$\mathbf{P}^-(t_k)$	Prediction state covariance
$\mathbf{P}^+(t_k)$	Posterior state covariance
$\mathbf{P}^*(t_k)$	Augmented state covariance

Information Filter

$\hat{\mathbf{y}}^+(t_{k-1})$	Prior information vector
$\hat{\mathbf{y}}^-(t_k)$	Prediction information vector
$\hat{\mathbf{y}}^+(t_k)$	Posterior information vector
$\mathbf{Y}^+(t_{k-1})$	Prior information matrix
$\mathbf{Y}^-(t_k)$	Prediction information matrix
$\mathbf{Y}^+(t_k)$	Posterior information matrix

Process Models

$\mathbf{f}[\cdot]$	Process model
$\nabla_x \mathbf{f}$	Jacobian of the process model with respect to the estimated states
$\mathbf{f}_v[\cdot]$	Vehicle model
$\nabla_x \mathbf{f}_v$	Jacobian of the vehicle model with respect to the vehicle states
$\mathbf{w}$	Process model noise
$\mathbf{Q}$	Process model noise covariance

Observation Models

$\mathbf{h}[\cdot]$	Observation model
$\nabla_x \mathbf{h}$	Jacobian of the observation model with respect to all estimated states
$\nabla_{x_v} \mathbf{h}$	Jacobian of the observation model with respect to the vehicle states
$\nabla_{f_i} \mathbf{h}$	Jacobian of the observation model with respect to the states of feature i
$\mathbf{v}$	Observation model noise
$\mathbf{R}$	Observation model noise covariance

Initialisation Models

$\mathbf{g}[\cdot]$	Initialisation model
$\mathbf{g}_f[\cdot]$	Feature initialisation model
$\mathbf{g}_p[\cdot]$	Pose initialisation model

Chapter 1

Introduction

This thesis deals with the problem of estimating the trajectory of an Autonomous Underwater Vehicle (AUV). In particular, vision sensors are considered to augment or replace traditional navigation solutions such as Doppler Velocity Logs (DVLs), Inertial Measurement Units (IMUs) and acoustic positioning systems.

1.1 Motivation

In recent years, an increasing number of AUVs have been deployed in a wide range of applications including geological surveys [51, 70, 116], biological studies [53, 98], marine archeology [3, 4, 77] and defence [42–47]. The growing interest in AUVs has been motivated by their advantages over manned submersibles and towed sensor platforms, such as the possibility to be deployed from smaller ships with fewer supporting crew members, and the ability to follow trajectories close to the seafloor, providing high resolution sensing even in complex environments with uneven topography such as canyons, ridges and reefs.

Aiming to provide new capabilities for marine science in Australia, the Australian Centre for Field Robotics acquired a SeaBED [99] AUV in 2005. In the time since, a number of applications have been proposed:

- Biodiversity assessment. The Australian Institute of Marine Science and the New South Wales Department of Environment and Climate Change are interested in performing biodiversity surveys to monitor existing marine protected areas, and to aid in the identification of additional environments that require protection. The selection of protected areas has often proceeded with little knowledge of the chosen environments, resulting in criticism from industries such as fishing and tourism that have

been affected by the changes. The acquisition of biodiversity surveys will enable more informed decisions on the selection of protected areas, providing better conservation outcomes while minimising the effects on industry.

- Geological surveying. The Ocean Drilling Program is interested in surveying 'drowned reef' environments that have been caused by the rise in sea levels over the last twenty thousand years. Visual models will aid the selection of locations where core drilling is to be performed. Such sites are of great interest given current climate change concerns, and may provide insights into past variations in ocean conditions and their effect on coral reefs.

- Resource exploration. Oil and gas companies are interested in obtaining high resolution models of seafloor features observed in coarse bathymetric maps. The models will be used to decide locations for exploratory drilling in areas likely to contain valuable resources.

- Marine archaeology. In cooperation with conservation and tourism groups, archaeologists would like to perform visual surveys of shipwrecks. In addition to possible archaeological outcomes, the surveys could be used to monitor the impact of increasing tourism-related activities occurring at such sites.

1.2 Underwater Navigation

Precise navigation is critical for an AUV to perform accurate trajectory following and mapping, and aids the recovery of the vehicle once a mission is complete. Unfortunately many sources of localisation for air and surface vehicles such as the Global Positioning System (GPS) and radio navigation beacons are unavailable underwater due to the rapid attenuation of electromagnetic signals. Current AUVs typically navigate using dead-reckoning and acoustic positioning systems, however there has recently been interest in terrain-based methods that are capable of inferring the position of a vehicle from observations of properties such as the profile or appearance of the seafloor [27, 59, 62].

1.2.1 Dead-reckoning

The most simple approach to navigation is dead-reckoning, in which the position of a vehicle is calculated by integrating motion observations acquired by onboard sensors. Typical dead-reckoning sensors for underwater vehicles include IMUs, which measure accelerations and rotation rates of the vehicle relative to an inertial frame, and DVLs, which use acoustic

signals to measure the velocity of the vehicle relative to the seafloor. Fusing the observations of multiple dead-reckoning instruments such as a DVL and IMU has been demonstrated to provide superior positioning to either sensor individually [8, 9].

Unfortunately, relying solely on dead-reckoning results in drifting navigation estimates, where the positioning error increases without bound the longer the vehicle operates. To enable accurate long-term navigation, dead-reckoning drift must be corrected by an external observation of the vehicle's position.

1.2.2 Acoustic Positioning

Due to the attenuation of electromagnetic radiation in water, acoustic signals are typically used for communication in the ocean. A range of acoustic positioning solutions exist, the most common of which are Long Base-Line (LBL) and Ultra-Short Base-Line (USBL) systems.

LBL systems involve deploying and surveying the position of a network of acoustic transponder beacons and acquiring a local sound speed profile, which can be expensive and time consuming. The range of the vehicle from a beacon can then be inferred by measuring the elapsed time between transmitting a signal and receiving a reply from a transponder. If multiple beacons are interrogated, the position of the vehicle can be estimated through triangulation. The selection of an operating frequency controls a trade-off between the resolution and range of an acoustic system. Most LBL systems use a frequency around 12kHz, proving accuracy within a few metres and a range of a few kilometres, while high-frequency systems operating around 300kHz can provide centimetre level positioning accuracy within a range of approximately 100 metres [62, 109].

In a USBL system, a transceiver containing an array of transducers is installed on a ship supporting the AUV deployment. The range and bearing of a transponder fixed to an AUV can then be estimated using phase differencing. While USBL systems may be more convenient than deploying a network of beacons, due to their lower accuracy they are more appropriate for homing, docking and recovery operations than precision navigation [62, 97, 102].

Precise navigation using only an acoustic positioning system is difficult due to low update rates. For example, periods of up to 20 seconds between updates has been reported for an LBL system [109]. High precision positioning is commonly achieved by fusing acoustic systems with dead-reckoning sensors [91, 108–110], however the resulting navigation solution retains the restricted operating range of the acoustic system. Additional problems with acoustic systems are occlusion and erroneous observations caused by multipath returns,

which are likely to occur when the vehicle is operating near the seafloor, particularly in the complex formations such as reefs, ridges and canyons that are the focus of many AUV applications.

1.2.3 Terrain-based Methods

Due to the limitations and cost of traditional underwater navigation solutions, there has recently been interest in the use of terrain-based methods to either augment or replace existing systems.

Odometry estimation approaches based on stereo-vision have been developed for a new class of small, low-cost AUVs such as AQUA [20, 36, 37, 53, 54] and Starbug [21–23], for which the size and cost of a DVL or IMU are inappropriate. Visual odometry is performed by extracting a set of feature observations from stereo images, allowing the 3D position of each feature to be triangulated. The motion of the vehicle can then be estimated by tracking the features through a sequence of images. Like dead-reckoning, visual odometry provides a drifting trajectory estimate with errors increasing over time, however this approach has proved sufficient for the target application of navigating short transects.

If an accurate map is available, the position of a vehicle can be inferred by correlating observations of the environment with the map. Map-based localisation has been demonstrated using bathymetric maps and sonar observations of the seafloor profile, where the position of the vehicle is estimated by a particle filter [57, 58, 90, 111]. Navigation in small areas has also been demonstrated on an AUV equipped with a camera, by registering a visual observation of the seafloor within a previously generated image-mosaic [38–40].

Unfortunately, detailed maps suitable for localisation are not available in many applications. This has motivated research in Simultaneous Localisation And Mapping (SLAM) approaches, in which the position of a vehicle and a map of the environment are jointly estimated. The vehicle starts with no knowledge of the environment, then builds up a map while it operates. 'Loop-closure' events, in which previously visited areas of the environment are observed allow drifting position estimates to be corrected.

In the original SLAM formulation [100, 101], the map consists of a sparse set of features that are used as navigation landmarks, much like the beacons in an acoustic positioning system. An estimate of the vehicle pose and feature positions and the uncertainty in the estimates are maintained using an Extended Kalman Filter (EKF). The computational complexity of a straightforward implementation of this approach is quadratic in the number of features. While a significant amount of research has been performed on managing this complexity [7, 41, 112, 115], scaling these techniques to long missions covering large distances remains an

open problem. The EKF-SLAM approach has been demonstrated for an underwater vehicle using observations of artificial sonar reflectors [112, 114].

SLAM has also been implemented using a particle filter for an AUV performing visual surveys of a shallow hydrothermal vent area [69, 70]. In this application, the map consists of the locations of natural bubble plumes and artificial acoustic reflectors observed by a sonar.

Another map representation used in underwater SLAM applications is an image-mosaic [32, 33]. Translational vehicle motions are estimated by correlating consecutive images, and overlapping images at loop-closure points allow dead-reckoning errors to be corrected. Complex vehicle motions including rotations are not supported however, and the mosaic map representation is only suitable for environments where a planar approximation of the seafloor is appropriate.

A promising recent development is the Visually Augmented Navigation (VAN) framework [26, 27, 29], in which the current vehicle state and a set of past poses can be estimated using an EKF or Extended Information Filter (EIF). The relative pose of the vehicle between images in a loop-closure situation is used to correct dead-reckoning drift, however unlike mosaic-based methods which perform dense image registration, sparse sets of features are associated to calculate arbitrary vehicle motions without constraints on the seafloor topology. Performing the estimation process using an EIF results in a significant computational efficiency advantage over EKF-based approaches, however recovering state estimates and covariances is difficult due to the information-form representation of the probability distribution maintained by the filter.

1.3 Objectives

In this thesis, two use-cases for vision sensors in underwater navigation are investigated.

- Firstly, the use of cameras for odometry estimation is considered. Improvements in visual odometry will aid the development of small, low-cost AUVs without a need for large and expensive sensors such as an IMU or DVL. Such developments promise to make AUVs more accessible for low-budget applications such as scientific surveys which lack the funding of applications such as the defence or oil and gas industries.

- Secondly, the use of cameras in conjunction with traditional dead-reckoning sensors is considered, where images are used to create loop-closure observations in a SLAM framework. This research has the potential the allow the development of robust AUVs capable of operating over larger distances and longer mission times, without the need

to deploy a set of beacons. This is of particular importance for AUVs operating near the seafloor and in complex environments, where the performance of acoustic position systems is limited.

1.4 Contributions

Towards the objective of improving vision-based navigation for autonomous underwater vehicles, the following contributions are presented in this thesis:

- An approach to estimate the odometry of an underwater vehicle through the fusion of sonar and vision observations. Novel aspects of this work include:
  - ▷ The use of a SLAM-style algorithm with a highly dynamic state vector to estimate the motion of a vehicle by tracking a collection of temporarily maintained features.
  - ▷ The use of sonar range observations to initialise the estimated positions of features tracked by a vision system.
- A stereo-vision relative pose estimation algorithm, suitable for visual odometry or SLAM applications. Contributions within this work include:
  - ▷ A survey and evaluation of existing motion estimation approaches, leading to an understanding of the assumptions and approximations in each method, their effects on the accuracy and speed of the algorithm, and the applications in which their use is appropriate.
  - ▷ An approach to efficiently calculate the maximum likelihood motion parameters for a stereo-rig, based on the bundle adjustment algorithm typically used in monocular vision problems.
  - ▷ A survey and evaluation of outlier rejection frameworks, leading to an understanding of the robustness of each approach to the errors expected in vision data.
  - ▷ An outlier rejection approach based on robust estimation, allowing the removal of erroneous observations and associations even when only a small set of features is available.
- A SLAM algorithm based on the VAN framework, suitable for large-scale mapping applications. Contributions include:

- ▷ The application of Cholesky modifications to a factorisation of the information matrix, resulting in improvements in the efficiency of state estimate and covariance recovery operations.
- ▷ A method to recover optimal estimates of the current vehicle pose states in constant time, allowing prediction and observation operations to be performed efficiently without corrupting the filter with approximate estimates.
- ▷ A covariance recovery method based on the 'sparse inverse' matrix, which is used to efficiently produce the vehicle pose covariances required to generate loop-closure hypotheses.

1.5 Thesis Structure

Chapter 2 provides background information on the computer vision geometry and algorithms used within the visual navigation approaches presented in this thesis.

Chapter 3 presents a novel approach for estimating the motion of an underwater vehicle by fusing observations from sonar and vision sensors. The algorithm is demonstrated on data acquired by a vehicle lacking traditional dead-reckoning sensors.

Chapter 4 presents the development of a stereo-vision relative pose estimation algorithm, which is demonstrated in a visual odometry application.

Chapter 5 presents a SLAM algorithm based on the VAN framework. The algorithm is demonstrated using the novel stereo-vision relative pose estimation algorithm to provide loop-closure observations that correct the drift in the estimated trajectory of an AUV equipped with a DVL.

Finally, **Chapter 6** provides conclusions and directions for future research.

Chapter 2

Computer Vision

2.1 Introduction

This chapter provides background information on the computer vision operations required by the visual navigation methods presented in this thesis. The properties of algorithms used to extract and associate visual features are investigated, and the camera model used to relate visual observation of an object to its location in the world is presented.

2.2 Feature Extraction and Association

A large number of algorithms to extract and associate visual features have been developed for a wide range of applications including robot localisation, object recognition and augmented reality [74, 75, 94]. A distinction is commonly made between algorithms appropriate for small-baseline images that are acquired from similar viewpoints, and wide-baseline images obtained from camera poses separated by large distances. Wide-baseline feature association is considered the more difficult problem due to larger variations in the position and appearance of observed objects. In vision-based navigation applications, small-baseline images are acquired when performing odometry estimation, while wide-baseline images are likely to occur in loop-closure situations when performing SLAM.

2.2.1 Small-baseline

Examples of small-baseline feature extraction algorithms include the Harris corner detector [49], Moravec interest operator [78], and FAST corner detector [92, 93]. These algorithms achieve low computational complexity by assuming there will be little or no change in the

scale of observed objects, allowing features to be extracted by evaluating the properties of fixed-sized windows of pixels. The limitation of this approach is the significant deterioration in extraction repeatability that occurs with variations in scale [94].

Simple and fast methods are also used for feature association in small-baseline applications. A feature is typically described by a window of pixels surrounding its location in an image, and tracking or matching is performed by optimising a simple cost function such as the Sum of Square Differences (SSD), Sum of Absolute Differences (SAD) or Normalised Cross-Correlation (NCC) [88], which assume the camera motion consists only of a translation.

While small-baseline algorithms provide little robustness to viewpoint and scale variations, excellent performance can be achieved at high frame rates when applied to their target application of images with small camera motion [82, 83].

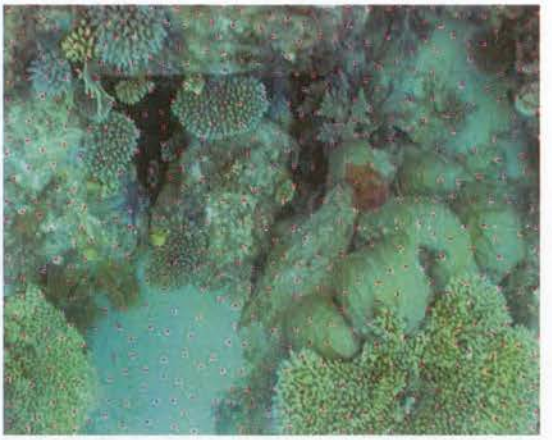
For experiments presented in this thesis involving small-baseline images, implementations of the Harris corner detector and Lucas-Kanade [67] tracking algorithm within the OpenCV[1] library have been used to extract and associate features. An example of feature extraction and tracking in small-baseline images of an underwater scene is presented in Figure 2.1. In this example, the Harris corner detector has been configured to search for features using a square 5 by 5 pixel window, and the number of features is limited by requiring a minimum of 20 pixels between extracted corners. The Lucas-Kanade tracker has been configured to use a 11 by 11 window of pixels to describe features, and tracking is initialised at the feature coordinates in the previous image. Both corner extraction and tracking is performed on greyscale versions of the original colour images.

Table 2.1 lists the feature extraction and tracking statistics for the images in Figure 2.1. The dimensions of each image is 720 by 576 pixels, and processing times were collected on a 2.0 GHz Pentium M processor.

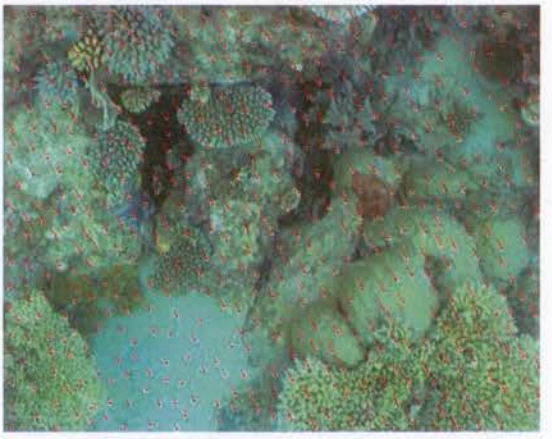
Image	Extracted features	Tracked features	Extraction time (s)	Tracking time (s)
One	514	493	0.112	0.089
Two	520	501	0.086	0.121
Three	532	514	0.086	0.094

Table 2.1: Small-baseline feature extraction and association statistics. Harris corner features were extracted in each image and tracked to the next frame using the Lucas-Kanade algorithm. The apparent motion between images of the features nearest to the camera is approximately 15 pixels. The features extracted from image three were tracked to a forth frame to calculate the statistics in the last row

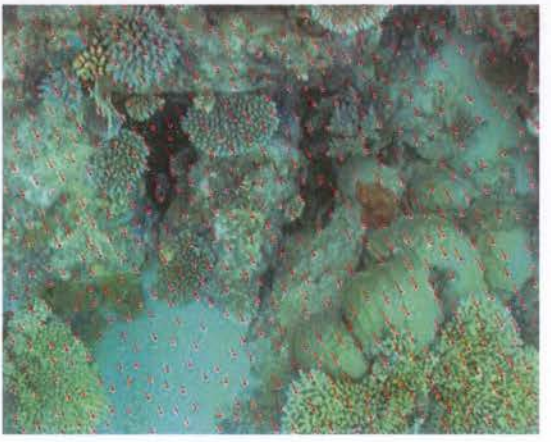
[1]OpenCV is available from Intel at `http://www.intel.com/technology/computing/opencv/` (last accessed March 22nd 2007)



(a) Image one



(b) Image two



(c) Image three

Figure 2.1: Small-baseline feature extraction and tracking. Harris corner features have been extracted and tracked using the Lucas-Kanade algorithm. The locations of features are marked by red and white circles, and the red lines in images two and three show the tracks joining the image coordinates of a feature in the previous and current image.

2.2.2 Wide-baseline

Rapid progress has recently occurred in the development of feature extraction and association algorithms for wide-baseline images. Examples include Maximally Stable Extremal Regions (MSER) [72], the Scale Invariant Feature Transform (SIFT) [66], and Speeded-Up Robust Features (SURF) [5]. Wide-baseline feature extraction and association approaches are designed to be robust against scale, illumination and viewpoint changes, resulting in algorithms that are more computationally complex than small-baseline approaches.

In an evaluation of algorithms suitable for wide-baseline applications, SIFT was recommended due to good association precision (the ratio of correct matches to total matches) [74, 75]. However, for all the tested algorithms, high precision was only achieved at low recall rates (only a small number of features are associated). Since this survey was performed before the development of the SURF algorithm, a brief investigation into the properties of SIFT and SURF will be performed.

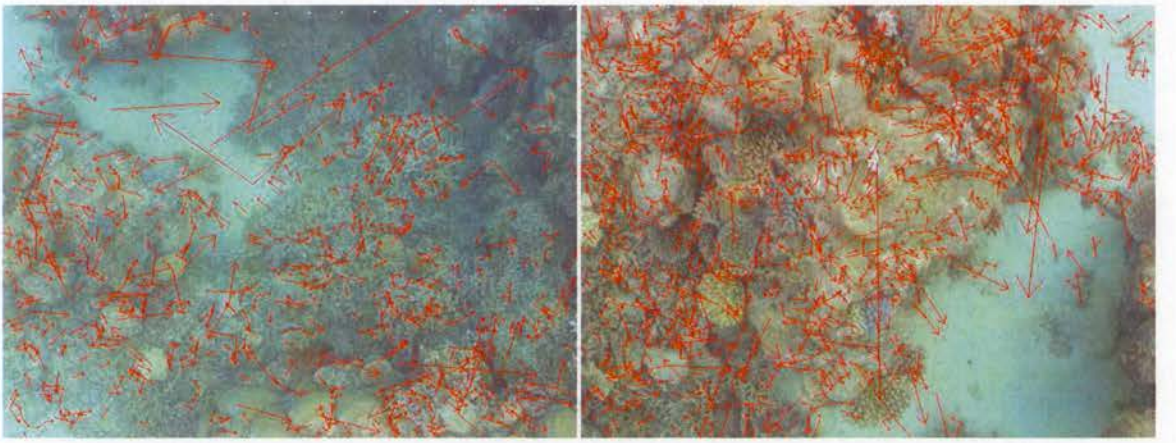
Figures 2.2, 2.3, 2.4 and 2.5 display sets of features extracted and associated by the SIFT[2] and SURF[3] algorithms in two pairs of wide-baseline images of underwater scenes. The feature extraction and association statistics are listed in Table 2.2. The dimensions of each image is 680 by 512 pixels, and the processing times were collected on a 2.0 GHz Pentium M processor. The processing times for SURF are significantly lower than SIFT, however the small numbers of associated features is consistent with the low recall rates reported for other wide-baseline approaches.

Image Pair	Feature Type	Left Features	Right Features	Associated Features	Left Time (s)	Right Time (s)	Association Time (s)
One	SIFT	727	1294	14	0.903	1.257	0.363
	SURF	801	1283	19	0.286	0.423	0.153
Two	SIFT	732	797	286	0.932	0.949	0.224
	SURF	718	737	174	0.268	0.277	0.068

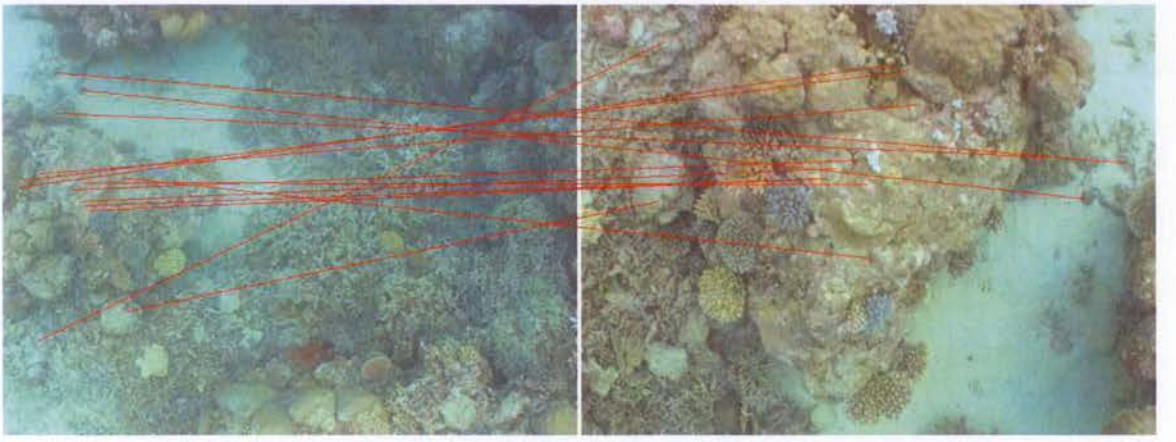
Table 2.2: Wide-baseline Feature extraction and association statistics. Image pair one is shown in Figures 2.2 and 2.3. Image pair two is shown in Figures 2.4 and 2.5.

[2] An implementation of the SIFT algorithm has been supplied by David Lowe

[3] SURF version 1.0.9 from `http://www.vision.ee.ethz.ch/~surf/` (last accessed March 22nd 2007)

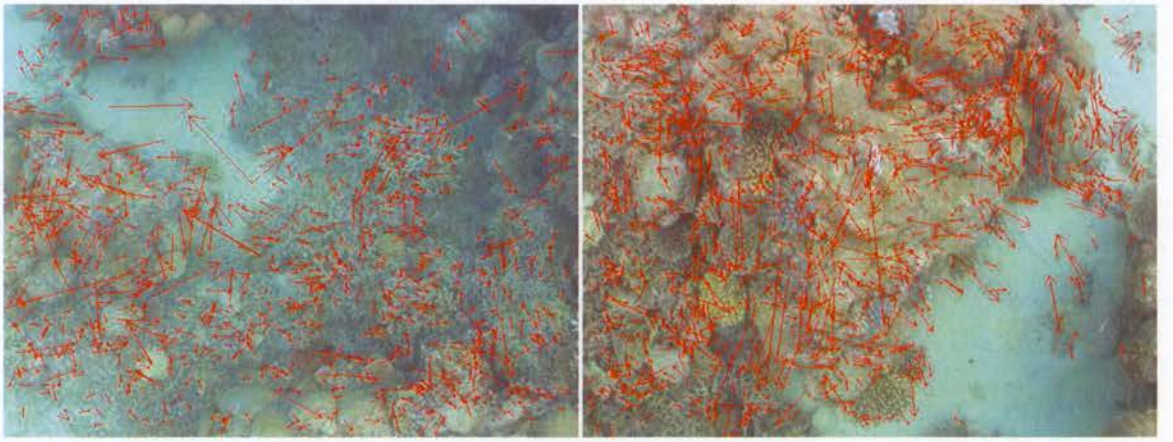


(a) Extracted SIFT features.

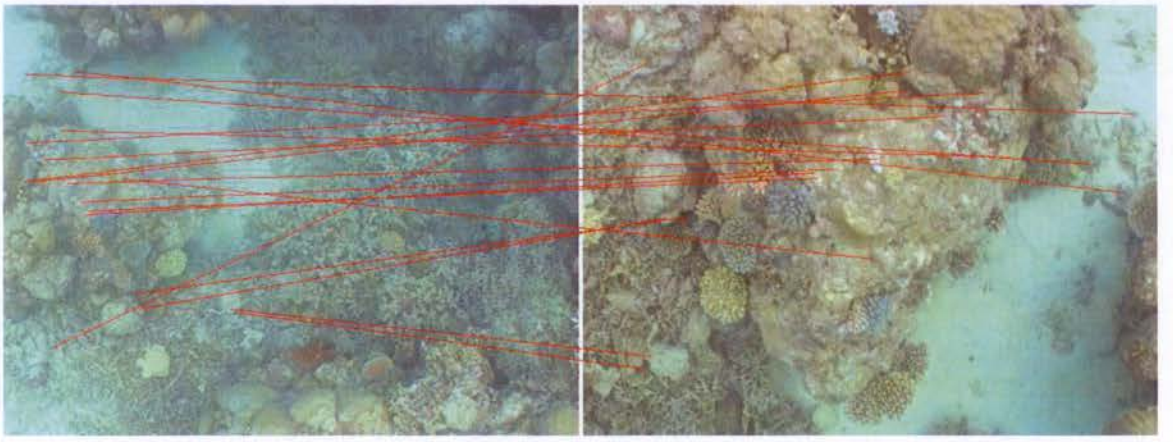


(b) Associated SIFT features

Figure 2.2: Wide-baseline feature extraction and association using the SIFT algorithm on example image pair one. The location of each extracted feature is marked by the tail of an arrow in (a). The size and direction of an arrow represents the scale and orientation of the corresponding feature. Lines are shown connecting the image coordinates of associated features in (b). Out of 727 features extracted from the left image and 1294 extracted from the right image, 14 features were associated between the two views.

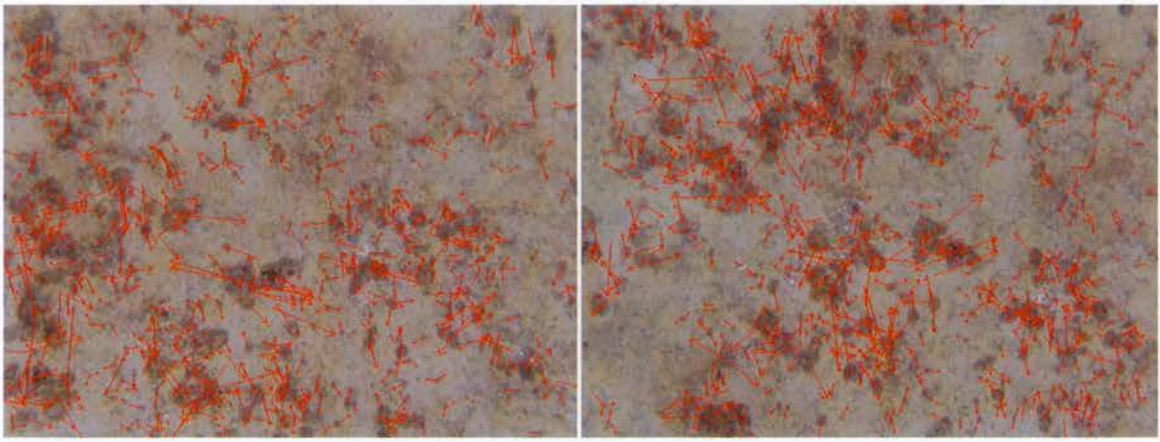


(a) Extracted SURF features.

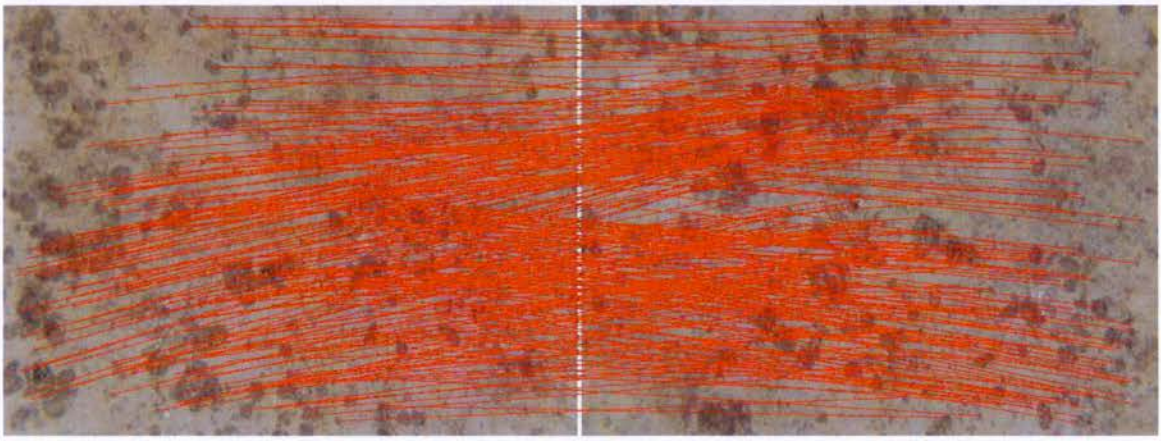


(b) Associated SURF features

Figure 2.3: Wide-baseline feature extraction and association using the SURF algorithm on example image pair one. The location of each extracted feature is marked by the tail of an arrow in (a). The size and direction of an arrow represents the scale and orientation of the corresponding feature. Lines are shown connecting the image coordinates of associated features in (b). Out of 801 features extracted from the left image and 1283 extracted from the right image, 19 features were associated between the two views.

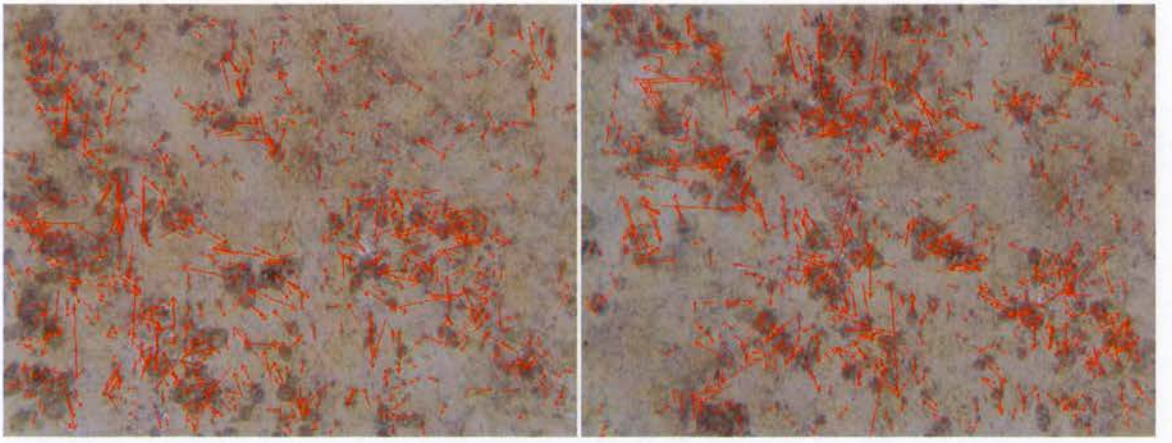


(a) Extracted SIFT features

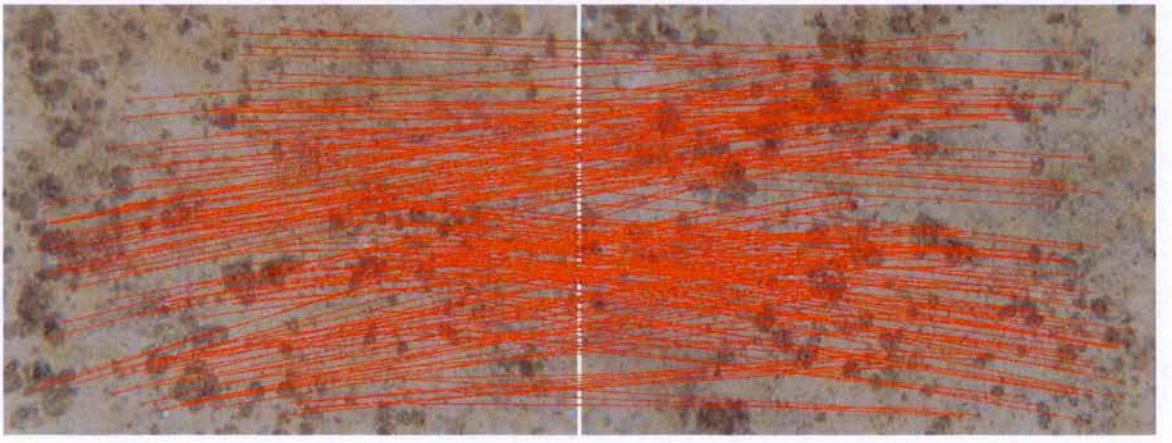


(b) Associated SIFT features

Figure 2.4: Wide-baseline feature extraction and association using the SIFT algorithm on example image pair two. The location of each extracted feature is marked by the tail of an arrow in (a). The size and direction of an arrow represents the scale and orientation of the corresponding feature. Lines are shown connecting the image coordinates of associated features in (b). Out of 732 features extracted from the left image and 797 extracted from the right image, 286 features were associated between the two views.



(a) Extracted SURF features



(b) Associated SURF features

Figure 2.5: Wide-baseline feature extraction and association using the SURF algorithm on example image pair two. The location of each extracted feature is marked by the tail of an arrow in (a). The size and direction of an arrow represents the scale and orientation of the corresponding feature. Lines are shown connecting the image coordinates of associated features in (b). Out of 718 features extracted from the left image and 737 extracted from the right image, 174 features were associated between the two views.

2.3 Camera Geometry

To calculate the motion of a camera from feature observations, the geometry relating image coordinates to a feature's position in the world must be known. A standard pin-hole camera model[4] is illustrated in Figure 2.6, in which the image plane and center of projection are separated by the focal length f, and the principal point $\mathbf{p}$ is shown at the intersection of the principal axis and the image plane. A feature at position ${}^c\mathbf{t}_i$ relative to the camera frame is projected to image coordinates $\mathbf{c}_i$.

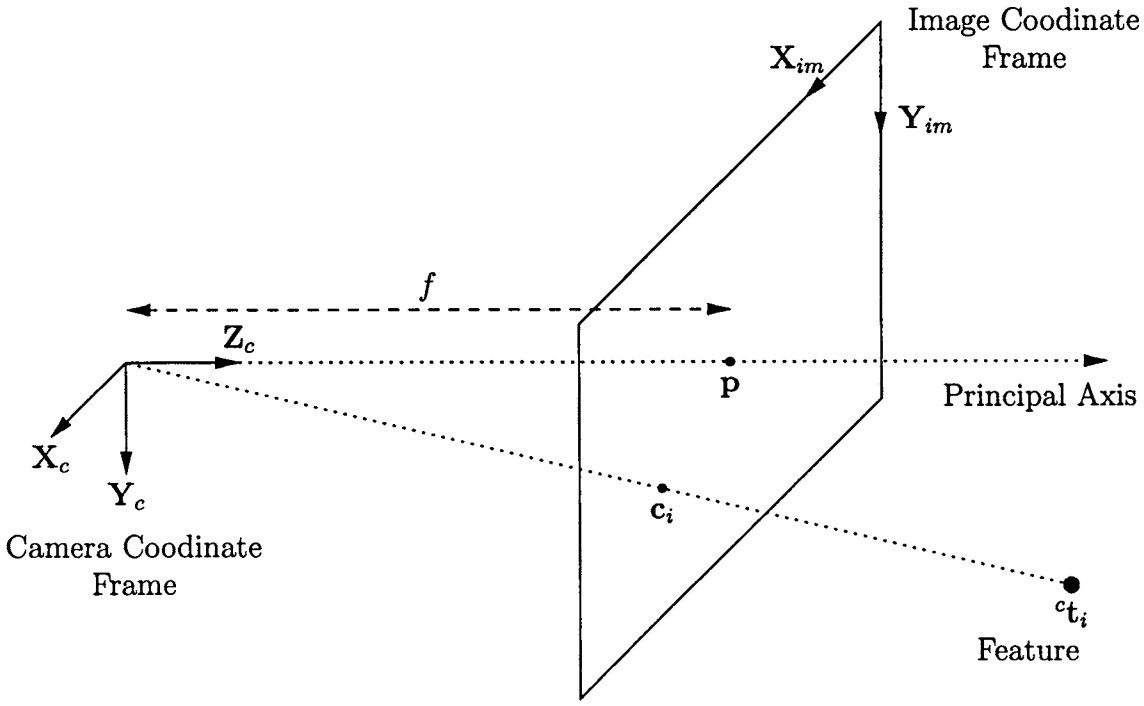


Figure 2.6: Camera observation model. A feature at a position ${}^c\mathbf{t}_i$ in the camera reference frame is projected to the image coordinates $\mathbf{c}_i$.

[4] The camera model has been adapted from the Camera Calibration Toolbox for Matlab, obtained at `http://www.vision.caltech.edu/bouguetj/calib_doc/` (last accessed March 22nd 2007), which has been used to calibrate the cameras for the experiments presented in this thesis.

2.3.1 Intrinsic Parameters

The internal geometry of a camera is defined by its focal length and the image coordinates of the principal point. If the focal length in millimetres is f and the number of pixels per millimetre in the image u-axis and v-axis are m_u and m_v respectively, the focal length measured in pixel units for each axis are

$$f_u = f m_u \tag{2.1}$$

$$f_v = f m_v \tag{2.2}$$

If the 3D coordinates of feature i relative to the camera frame are ${}^c\mathbf{t}_i = [{}^cx_i \ {}^cy_i \ {}^cz_i]^\mathsf{T}$, the normalised coordinates of the feature are

$$\mathbf{n}_i = \begin{bmatrix} n_{u_i} \\ n_{v_i} \end{bmatrix} = \begin{bmatrix} {}^cx_i / {}^cz_i \\ {}^cy_i / {}^cz_i \end{bmatrix} \tag{2.3}$$

If the image coordinates of the principal point are $\mathbf{p} = [p_u \ p_v]^\mathsf{T}$, the image coordinates of the feature observation are

$$\mathbf{c}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} p_u + f_u n_{u_i} \\ p_v + f_v n_{v_i} \end{bmatrix} \tag{2.4}$$

The four values p_u, p_v, f_u and f_v are known as the intrinsic parameters of a camera. When arranged into the camera intrinsic parameter matrix

$$\mathbf{K} = \begin{bmatrix} f_u & 0 & p_u \\ 0 & f_v & p_v \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

Equation 2.4 can be simplified using homogeneous coordinates

$$\mathbf{c}_i = \mathbf{K} \left[\mathbf{I}_{(3\times3)} \mid \mathbf{0}_{(3\times1)} \right] {}^c\mathbf{t}_i \tag{2.6}$$

2.3.2 Extrinsic Parameters

The extrinsic parameters of a camera consist of its pose relative to an external reference frame. A camera pose is represented by the vector ${}^e\mathbf{p}_c = [{}^e\mathbf{t}_c^\mathsf{T}, {}^e\boldsymbol{\psi}_c^\mathsf{T}]^\mathsf{T}$, where ${}^e\mathbf{t}_c$ and ${}^e\boldsymbol{\psi}_c$ contain the position and orientation parameters of the camera relative to frame e. Details on pose and orientation representations are provided in Appendix A.

The position of feature i in the external reference frame ${}^{e}\mathbf{t}_i$ is transformed to the camera frame with the coordinate transformation

$$ {}^{c}\mathbf{t}_i = {}^{c}_{e}\mathbf{R}({}^{e}\mathbf{t}_i - {}^{e}\mathbf{t}_c) \tag{2.7} $$

The function projecting the position of a feature into the image coordinates is then

$$ \begin{aligned} \mathbf{c}_i &= \mathbf{proj}[{}^{e}\mathbf{p}_c, {}^{e}\mathbf{t}_i] \\ &= \mathbf{K}\left[{}^{c}_{e}\mathbf{R} \mid {}^{c}\mathbf{t}_e\right] {}^{e}\mathbf{t}_i \end{aligned} \tag{2.8} $$

The Jacobians of the projected image coordinates with respect to the camera pose and feature position states are

$$ \frac{\partial \mathbf{proj}[{}^{e}\mathbf{p}_c, {}^{e}\mathbf{t}_i]}{\partial {}^{e}\mathbf{p}_c} = \frac{\partial \mathbf{c}_i}{\partial {}^{c}\mathbf{t}_i} \frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{p}_c} \tag{2.9} $$

$$ \frac{\partial \mathbf{proj}[{}^{e}\mathbf{p}_c, {}^{e}\mathbf{t}_i]}{\partial {}^{e}\mathbf{t}_i} = \frac{\partial \mathbf{c}_i}{\partial {}^{c}\mathbf{t}_i} \frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{t}_i} \tag{2.10} $$

in which $\frac{\partial \mathbf{c}_i}{\partial {}^{c}\mathbf{t}_i}$ is the Jacobian of the projection function of Equation 2.4

$$ \frac{\partial \mathbf{c}_i}{\partial {}^{c}\mathbf{t}_i} = \begin{bmatrix} f_u/{}^{c}z_i & 0 & -(f_u {}^{c}x_i)/{}^{c}z_i^2 \\ 0 & f_v/{}^{c}z_i & -(f_v {}^{c}y_i)/{}^{c}z_i^2 \end{bmatrix} \tag{2.11} $$

and $\frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{p}_c}$ and $\frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{t}_i}$ are the Jacobians of the coordinate transformation function of Equation 2.7 relative to the camera pose and feature position states

$$ \frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{p}_c} = \begin{bmatrix} -{}^{c}_{e}\mathbf{R} & \frac{\delta {}^{c}_{e}\mathbf{R}}{\delta {}^{c}\phi_e}({}^{e}\mathbf{t}_i - {}^{e}\mathbf{t}_c) & \frac{\delta {}^{c}_{e}\mathbf{R}}{\delta {}^{c}\theta_e}({}^{e}\mathbf{t}_i - {}^{e}\mathbf{t}_c) & \frac{\delta {}^{c}_{e}\mathbf{R}}{\delta {}^{c}\psi_e}({}^{e}\mathbf{t}_i - {}^{e}\mathbf{t}_c) \end{bmatrix} \tag{2.12} $$

$$ \frac{\partial {}^{c}\mathbf{t}_i}{\partial {}^{e}\mathbf{t}_i} = {}^{c}_{e}\mathbf{R} \tag{2.13} $$

2.3.3 Image Distortion

Significant deviations from the ideal projective camera model of Figure 2.6 occur due to lens distortion, which is typically modelled using the plumb-line [10] method consisting of radial and tangential distortion components.

The distorted normalised coordinates for feature i are given by the equation

$$ \boldsymbol{d}_i = \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \mathbf{n}_i + \begin{bmatrix} 2k_4 n_{u_i} n_{v_i} + k_5(r^2 + 2n_{u_i}^2) \\ k_4(r^2 + 2n_{v_i}^2) + 2k_5 n_{u_i} n_{v_i} \end{bmatrix} \tag{2.14} $$

where k_1, k_2 and k_3 are radial distortion coefficients, k_4 and k_5 are tangential distortion coefficients, and the radial distance from the principal axis is

$$r = \sqrt{n_{u_i}^2 + n_{v_i}^2} \tag{2.15}$$

For cameras operating underwater, additional image distortion is caused by refraction as light passes from water to air. However, if a flat viewport is used, and a camera is oriented such that its principal axis is perpendicular to the viewport, the resulting image distortion is radially distributed [96].

In this thesis, it is assumed that all image distortions, including those due to refraction, can be modelled using the standard plumb-line parameters. During the camera calibration procedures performed for experiments presented in this thesis, the reprojection errors for corners of a planar checkerboard target in water were equivalent to those for calibrations performed in air. This suggests the standard plumb-line parameters are sufficient to model the distortions produced in the camera systems tested.

While the assumptions of an ideally oriented camera and purely radial distortions may not be perfectly met, other model parameters such as the tangential distortion parameters designed for poorly centered lenses may partially compensate for any remaining errors. To minimise any range-dependant modelling errors, the calibration process was performed with the cameras at a distance from the calibration target similar to the operating altitude of the AUVs that gathered the data used in this thesis.

2.3.4 Two-view Geometry

The intrinsic and extrinsic parameters of two cameras can be used to constrain plausible feature observations in two images [50]. The geometry of two views is encapsulated in the 3×3 fundamental matrix, which is defined by the equation

$$\mathbf{c}_2^{\mathsf{T}} \, {}_1^2\mathbf{F} \, \mathbf{c}_1 = 0 \tag{2.16}$$

in which ${}_1^2\mathbf{F}$ is the fundamental matrix mapping feature observations in image 1 to image 2, and $\mathbf{c}_1$ and $\mathbf{c}_2$ are the homogeneous coordinates of a feature in two images.

If the two-view geometry is known, the fundamental matrix can be calculated with

$${}_1^2\mathbf{F} = \mathbf{K}_2^{-T} \left[{}^2\mathbf{t}_1 \right]_\times {}_1^2\mathbf{R} \, \mathbf{K}_1^{-1} \tag{2.17}$$

where $[\cdot]_\times$ is the skew-symmetric matrix implementing the vector cross product.

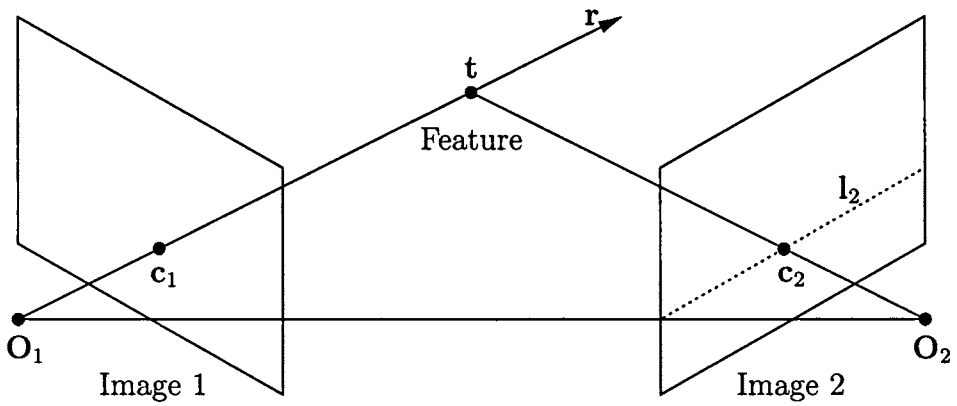
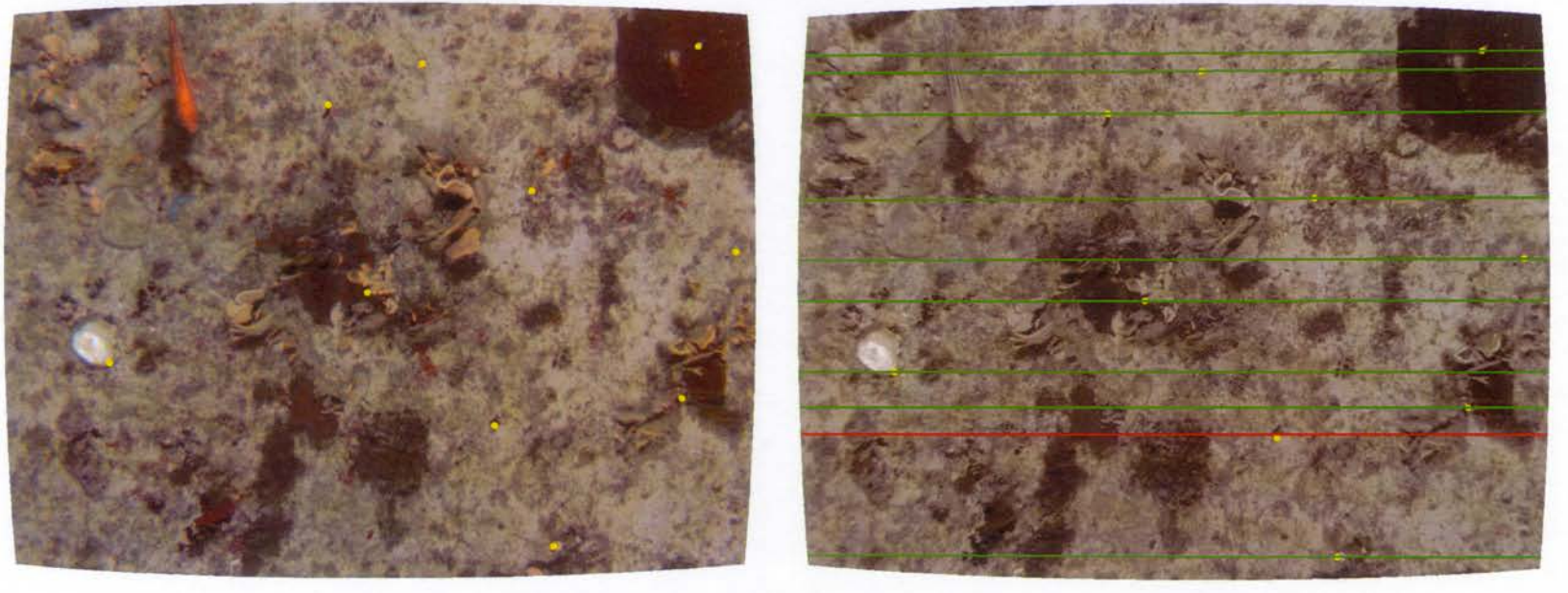


Figure 2.7: Epipolar geometry of two views. A feature at location $\mathbf{t}$ is observed at coordinates $\mathbf{c}_1$ in the image produced by the camera centered at $\mathbf{O}_1$. From the first camera observation alone, the depth of the feature is unknown, however it must lie on the ray $\mathbf{r}$. In the image produced by the camera centered at $\mathbf{O}_2$, the ray is imaged as the epipolar line $\mathbf{l}_2$, on which the feature observation $\mathbf{c}_2$ must lie.

The epipolar geometry constraint is illustrated in Figure 2.7, where the feature coordinates $\mathbf{c}_1$ in image one restricts the possible location of the feature in a second image to an epipolar line $\mathbf{l}_2$. The epipolar line can be calculated from the fundamental matrix using the equation

$$\mathbf{l}_2 = {}^2_1\mathbf{F}\,\mathbf{c}_1 \tag{2.18}$$

The epipolar constraint can be used to restrict the search for a feature in a second image, or test the consistency of observations of a feature in two views. Figure 2.8 demonstrates the use of epipolar lines to reject a poorly tracked feature in a pair of images acquired from a stereo-rig. Since epipolar geometry only provides a one-dimensional constraint for the location of a feature within an image, feature observation outliers containing errors parallel to an epipolar line cannot be rejected using this method.



(a) Left image

(b) Right image

Figure 2.8: Outlier rejection using epipolar lines. The effects of lens distortion have been removed from each image to produce straight epipolar lines. Feature coordinates are marked by yellow circles in both images, and the left image observations have been used to generate epipolar lines for the right image. In this example, one poorly tracked feature was rejected (shown by the red epipolar line) due to inconsistency with the epipolar geometry.

2.4 Summary

This chapter has presented a summary of computer vision algorithms and equations that will be used by the vision-based navigation approaches presented in the remainder of this thesis.

Feature extraction and association algorithms suitable for real-time applications are available for small-baseline applications. An implementation of the Harris corner detector and Lucas-Kanade tracker will be used for small-baseline problems in Chapters 3 and 4.

Wide-baseline feature extraction and association algorithms are computationally expensive and produce low recall rates. These properties will be considered in Chapter 5 when selecting a framework to perform SLAM using visual features.

The epipolar geometry of two views has been demonstrated to provide a one-dimensional constraint on feature observations in two images. Since feature observations containing errors parallel to the direction of an epipolar line cannot be detected with this method. Additional outlier rejection methods are considered in Chapter 4.

Chapter 3

Odometry from the Fusion of Sonar and Monocular Vision

3.1 Introduction

This chapter presents a novel approach to estimate the trajectory of the Oberon ROV, which is sparsely instrumented relative to most scientific vehicles. The sensor suite onboard Oberon consists of a mechanically scanned pencil-beam sonar, a monocular vision system, an integrated compass/tilt attitude sensor and a fibre-optic gyroscope measuring the yaw rate of the vehicle. Due to the lack of a traditional dead-reckoning sensor such as an IMU or DVL, the motion of the vehicle must be estimated by tracking a set of features extracted from observations of the environment.

3.2 Method

The sonar and vision sensors onboard Oberon possess complementary properties, that when combined, provide a solution to the motion estimation problem. The sonar is able to observe the range to features, however its sparse observations of the environment make re-observing features unlikely, and it cannot provide discriminating feature descriptions to enable features to be recognised. In contrast, the camera lacks the ability to observe the range to features, however the large field of view enables features to be observed multiple times through a sequence of images, and vision data can provide rich feature descriptions to allow data association.

The motion of the vehicle will be estimated by visually tracking a set of features that have been initialised using sonar range observations. This approach is similar to bearing-only

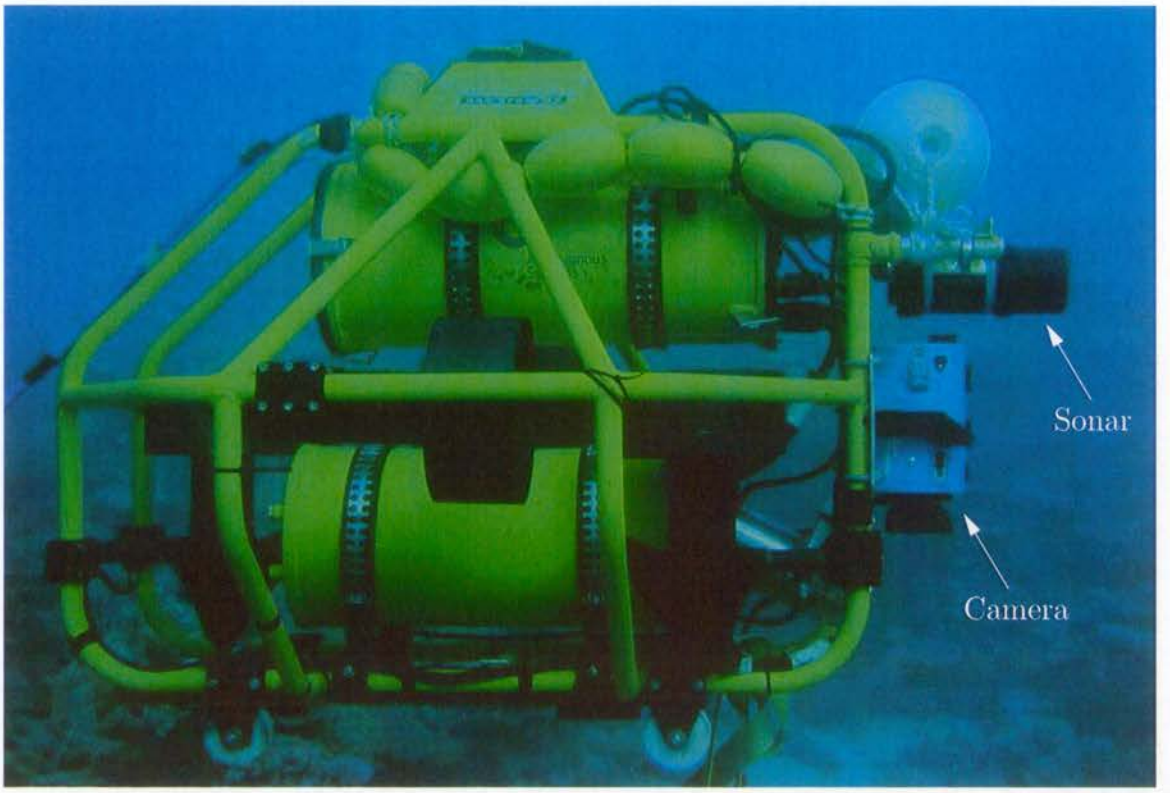


Figure 3.1: Configuration of the sonar and vision sensors onboard the Oberon ROV. The camera is oriented to view the seafloor, while the sonar scans through a 180° arc in a vertical plane perpendicular to the vehicle's direction of travel.

SLAM [2, 11, 31] in which odometry sensors are used to provide an estimate of the scale of motion, enabling the range of features to be calculated. In this case however, the sonar range-finder is used to measure the distance of features, which makes the motion of the vehicle observable.

The approach also has similarities to the Structure From Motion (SFM) problem, in which the trajectory of a camera and the location of a set of features are recovered up to a scale ambiguity [50, 106]. In the method presented here, the additional sonar range measurements allow the scale of the vehicle motion to be estimated.

The configuration of the sonar and vision sensors onboard the Oberon ROV are displayed in Figure 3.1. The adjacent location of the two sensors allows many sonar returns to be registered in images acquired by the camera. The process of initialising and observing features is illustrated in Figure 3.2.

The feature initialisation process is complicated by two problems leading to inexact registration of the seafloor location causing a sonar return within an image. Firstly, the sonar

and camera observations are not synchronised, and while the motion of the vehicle between the times the observations are acquired can be estimated, some error will inevitably remain to corrupt the visual feature position estimate. Secondly, the exact location on the seafloor feature causing a sonar return is unknown due to the diverging sonar beam. The best option is to find a feature on the seafloor near the center of the sonar-beam, however the vision system will track a feature near, but not necessarily exactly at the location causing the sonar return. As a result, the range of the feature observed by the sonar will not exactly match the range of the feature tracked by the vision system.

The current state of the vehicle, the pose of the vehicle at the time the last image was acquired, and the position of a set of features are estimated in an EKF using the procedure illustrated in Figure 3.3. The estimation process is similar to feature-based SLAM algorithms, however in this application the features are not used to generate loop-closure observations, but to compensate for the lack of dead-reckoning sensors by providing observations of the current vehicle motion.

To overcome the feature observation registration issues, the uncertainty of sonar range observations are inflated to compensate for the potential difference in the distances to the feature causing the sonar return and the feature tracked by the vision system. Additionally, maintaining the pose of the vehicle when the last image was acquired in the filter minimises the error in the estimated vehicle motion between the times the vision and sonar observations are acquired.

The state vector for this estimation problem is highly dynamic, with feature and pose states regularly augmented and removed. Since the features are not designed to be used for loop-closure situations as the would be in a SLAM application, they are removed once they pass out of the camera's field of view. Similarly, since past vehicle poses are of no further interest once a new image has been acquired, previously augmented poses are removed from the filter.

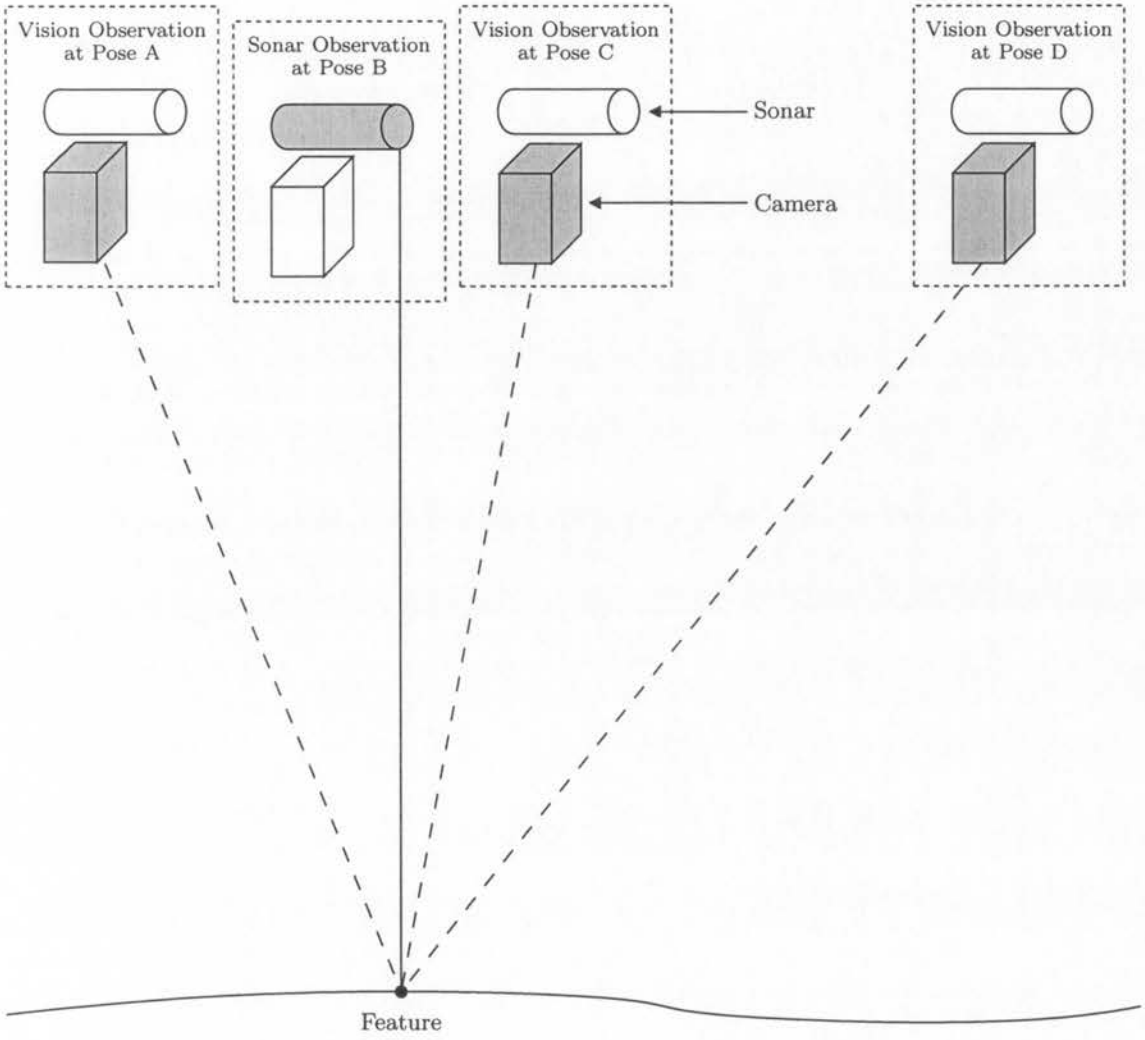


Figure 3.2: The feature initialisation and observation process. The vision and sonar sensors are shown at four poses corresponding to four times at which observations have been acquired. The visual observation at pose A and sonar range measurement at pose B are used to initialise the position of a feature. Additional visual observations at poses C and D allow the motion of the vehicle to be estimated.

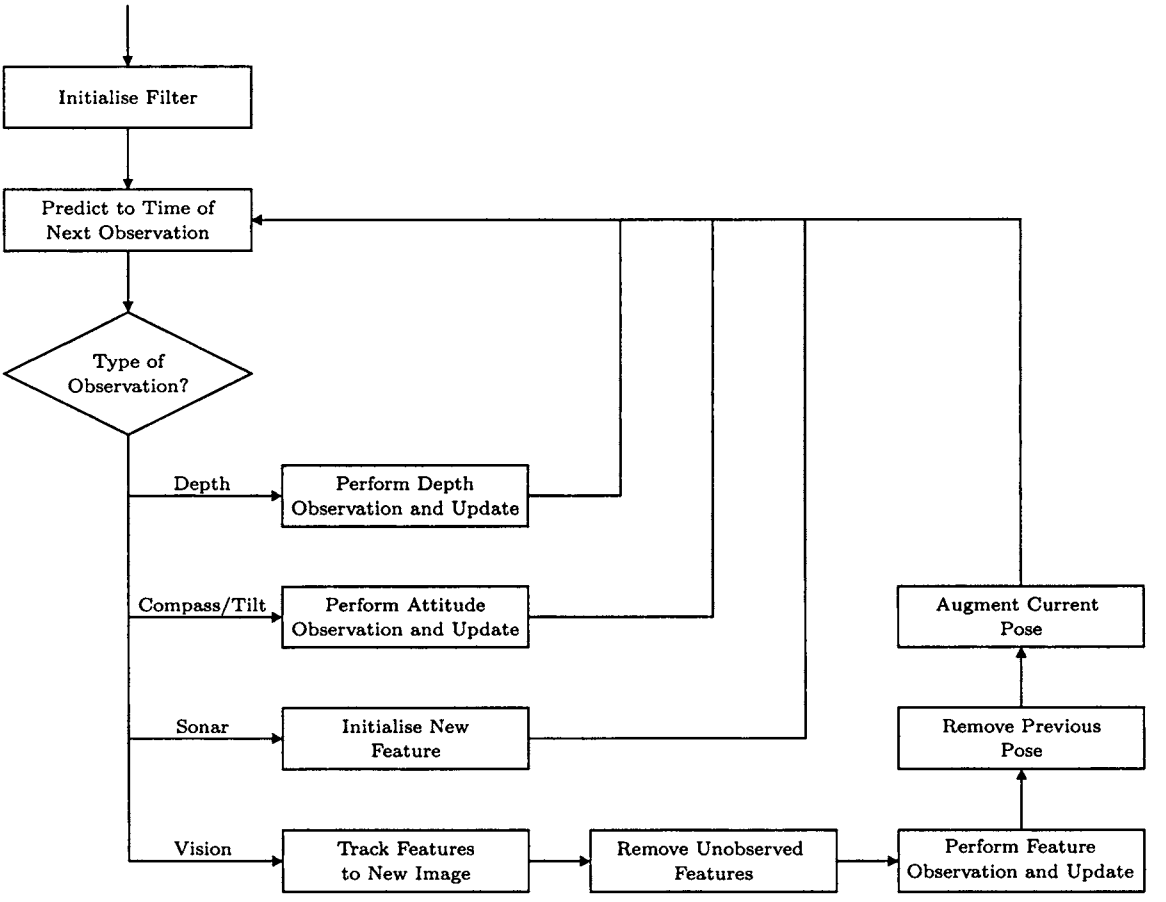


Figure 3.3: The motion estimation process for the Oberon ROV. The current vehicle state, the pose of the vehicle at the time the last image was acquired, and the position of a set of features are estimated using the typical EKF predict, observe and update cycle.

3.2.1 The Estimated State Vector

The estimated current vehicle state, previous vehicle poses and feature positions are represented by the state vector

$$\hat{\mathbf{x}}^{+}(t_k) = \begin{bmatrix} \hat{\mathbf{x}}_v^{+}(t_k) \\ \hat{\mathbf{x}}_t^{+}(t_k) \\ \hat{\mathbf{x}}_m^{+}(t_k) \end{bmatrix} \tag{3.1}$$

in which $\hat{\mathbf{x}}_v^{+}(t_k)$ contains parameters defining the current state of the vehicle, $\hat{\mathbf{x}}_t^{+}(t_k)$ represents the trajectory states of a set of previous vehicle poses, and $\hat{\mathbf{x}}_m^{+}(t_k)$ represents the map states of a set of features.

The current state of the vehicle is represented by the vector

$$\hat{\mathbf{x}}_v^{+}(t_k) = \begin{bmatrix} {}^{n}\hat{\mathbf{t}}_v^{+}(t_k) \\ {}^{n}\hat{\boldsymbol{\psi}}_v^{+}(t_k) \\ {}^{v}\hat{\mathbf{v}}_v^{+}(t_k) \\ \hat{\boldsymbol{\omega}}_v^{+}(t_k) \end{bmatrix} \tag{3.2}$$

in which ${}^{n}\hat{\mathbf{t}}_v^{+}(t_k)$ and ${}^{n}\hat{\boldsymbol{\psi}}_v^{+}(t_k)$ represent the position and orientation of the vehicle in the navigation frame, ${}^{v}\hat{\mathbf{v}}_v^{+}(t_k)$ represents the velocity of the vehicle measured relative to the vehicle frame and $\hat{\boldsymbol{\omega}}_v^{+}(t_k)$ contains the local body rotation rates of the vehicle at time t_k.

The trajectory vector consists of m past vehicle poses

$$\hat{\mathbf{x}}_t^{+}(t_k) = \begin{bmatrix} \hat{\mathbf{x}}_{p_1}^{+}(t_k) \\ \hat{\mathbf{x}}_{p_2}^{+}(t_k) \\ \vdots \\ \hat{\mathbf{x}}_{p_m}^{+}(t_k) \end{bmatrix} \tag{3.3}$$

in which each vehicle pose is represented by its position and orientation relative to the navigation frame

$$\hat{\mathbf{x}}_{p_i}^{+}(t_k) = \begin{bmatrix} {}^{n}\hat{\mathbf{t}}_{p_i}^{+}(t_k) \\ {}^{n}\hat{\boldsymbol{\psi}}_{p_i}^{+}(t_k) \end{bmatrix} \tag{3.4}$$

The map vector consist of n features

$$\hat{\mathbf{x}}_m^{+}(t_k) = \begin{bmatrix} \hat{\mathbf{x}}_{f_1}^{+}(t_k) \\ \hat{\mathbf{x}}_{f_2}^{+}(t_k) \\ \vdots \\ \hat{\mathbf{x}}_{f_n}^{+}(t_k) \end{bmatrix} \tag{3.5}$$

where each feature represented by its position in the navigation frame

$$\hat{\mathbf{x}}_{f_i}^{+}(t_k) = \left[{}^{n}\hat{\mathbf{t}}_{f_i}^{+}(t_k)\right] \tag{3.6}$$

Uncertainty in the estimated states is represented by the covariance matrix

$$\mathbf{P}^{+}(t_k) = \begin{bmatrix} \mathbf{P}_{vv}^{+}(t_k) & \mathbf{P}_{vt}^{+}(t_k) & \mathbf{P}_{vm}^{+}(t_k) \\ \mathbf{P}_{vt}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tt}^{+}(t_k) & \mathbf{P}_{tm}^{+}(t_k) \\ \mathbf{P}_{vm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{mm}^{+}(t_k) \end{bmatrix} \tag{3.7}$$

where $\mathbf{P}_{vv}^{+}(t_k)$, $\mathbf{P}_{tt}^{+}(t_k)$ and $\mathbf{P}_{mm}^{+}(t_k)$ are the covariances of the vehicle, trajectory and map states respectively, and $\mathbf{P}_{vt}^{+}(t_k)$ represents the cross-covariance between the vehicle and trajectory, $\mathbf{P}_{vm}^{+}(t_k)$ is the cross-covariance between the vehicle and map, and $\mathbf{P}_{tm}^{+}(t_k)$ is the cross-covariance between the trajectory and map states.

3.2.2 The Estimation Process

The state estimate and covariance are updated using the three-stage EKF prediction, observation and update process.

Prediction

The evolution of the estimated states is described by a process model of the form

$$\mathbf{x}(t_k) = \mathbf{f}\left[\mathbf{x}(t_{k-1}), t_k\right] + \mathbf{w}(t_k) \tag{3.8}$$

in which $\mathbf{w}(t_k)$ is a vector of process model errors at time t_k. The process error sequence is assumed to be zero-mean, temporally uncorrelated with covariance $\mathbf{Q}(t_k)$.

In this application, the past poses and features are assumed to be stationary, resulting in a process model of the form

$$\begin{bmatrix} \mathbf{x}_v(t_k) \\ \mathbf{x}_t(t_k) \\ \mathbf{x}_m(t_k) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_v\left[\mathbf{x}_v(t_{k-1}), t_k\right] \\ \mathbf{x}_t(t_{k-1}) \\ \mathbf{x}_m(t_{k-1}) \end{bmatrix} + \begin{bmatrix} \mathbf{w}_v(t_k) \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{3.9}$$

in which $\mathbf{f}_v[\cdot,\cdot]$ is the vehicle process model and $\mathbf{w}_v(t_k)$ is the vehicle process noise vector. In this application, the constant velocity vehicle model presented in Appendix B.2 is used to predict the motion of the vehicle.

The process model covariance has the form

$$\mathbf{Q}(t_k) = \begin{bmatrix} \mathbf{Q}_v(t_k) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.10}$$

where $\mathbf{Q}_v(t_k)$ is the vehicle process model covariance.

The standard EKF equations to calculate the predicted estimate and covariance are

$$\hat{\mathbf{x}}^-(t_k) = \mathbf{f}\left[\hat{\mathbf{x}}^+(t_{k-1}), t_k\right] \tag{3.11}$$

$$\mathbf{P}^-(t_k) = \nabla_x \mathbf{f}(t_k)\, \mathbf{P}^+(t_{k-1})\, \nabla_x^\mathsf{T} \mathbf{f}(t_k) + \mathbf{Q}(t_k) \tag{3.12}$$

in which $\nabla_x \mathbf{f}(t_k)$ is the Jacobian of the process model with respect to the state vector evaluated at the prior estimates. In this application, the process model Jacobian has the form

$$\nabla_x \mathbf{f}(t_k) = \begin{bmatrix} \nabla_x \mathbf{f}_v(t_k) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.13}$$

where the vehicle model Jacobian is defined by

$$\nabla_x \mathbf{f}_v(t_k) = \left.\frac{\partial \mathbf{f}_v}{\partial \mathbf{x}_v}\right|_{\hat{\mathbf{x}}_v^+(t_{k-1})} \tag{3.14}$$

Taking advantage of the sparse process model Jacobian, the covariance propagation formula of Equation 3.12 can be simplified to

$$\begin{bmatrix} \mathbf{P}_{vv}^-(t_k) & \mathbf{P}_{vt}^-(t_k) & \mathbf{P}_{vm}^-(t_k) \\ \mathbf{P}_{vt}^{-\mathsf{T}}(t_k) & \mathbf{P}_{tt}^-(t_k) & \mathbf{P}_{tm}^-(t_k) \\ \mathbf{P}_{vm}^{-\mathsf{T}}(t_k) & \mathbf{P}_{tm}^{-\mathsf{T}}(t_k) & \mathbf{P}_{mm}^-(t_k) \end{bmatrix} =$$

$$\begin{bmatrix} \nabla_x \mathbf{f}_v(t_k)\, \mathbf{P}_{vv}^+(t_{k-1})\, \nabla_x^\mathsf{T} \mathbf{f}_v(t_k) + \mathbf{Q}_v(t_k) & \nabla_x \mathbf{f}_v(t_k)\, \mathbf{P}_{vt}^+(t_{k-1}) & \nabla_x \mathbf{f}_v(t_k)\, \mathbf{P}_{vm}^+(t_{k-1}) \\ \mathbf{P}_{vt}^{+\mathsf{T}}(t_{k-1})\, \nabla_x^\mathsf{T} \mathbf{f}_v(t_k) & \mathbf{P}_{tt}^+(t_{k-1}) & \mathbf{P}_{tm}^+(t_{k-1}) \\ \mathbf{P}_{vm}^{+\mathsf{T}}(t_{k-1})\, \nabla_x^\mathsf{T} \mathbf{f}_v(t_k) & \mathbf{P}_{tm}^{+\mathsf{T}}(t_{k-1}) & \mathbf{P}_{mm}^+(t_{k-1}) \end{bmatrix} \tag{3.15}$$

Observation

Observations are assumed to be acquired according to a model of the form

$$\mathbf{z}(t_k) = \mathbf{h}\left[\mathbf{x}(t_k), t_k\right] + \mathbf{v}(t_k) \tag{3.16}$$

in which $\mathbf{v}(t_k)$ is an observation error, which is assumed to be from a zero-mean and temporally uncorrelated sequence with covariance $\mathbf{R}(t_k)$.

The observation model is used to generate an innovation $\boldsymbol{\nu}(t_k)$ and innovation covariance $\mathbf{S}(t_k)$ defined by

$$\boldsymbol{\nu}(t_k) = \mathbf{z}(t_k) - \mathbf{h}\left[\hat{\mathbf{x}}^-(t_k), t_k\right] \tag{3.17}$$

$$\mathbf{S}(t_k) = \nabla_x \mathbf{h}(t_k)\, \mathbf{P}^-(t_k)\, \nabla_x^{\mathsf{T}} \mathbf{h}(t_k) + \mathbf{R}(t_k) \tag{3.18}$$

in which $\nabla_x \mathbf{h}(t_k)$ is the Jacobian of the observation function evaluated at the predicted state estimate

$$\nabla_x \mathbf{h}(t_k) = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}^-(t_k)} \tag{3.19}$$

In this application, observations are acquired from a depth sensor using the model in Section 3.2.5, the attitude sensor using the model in Section B.3.2 and visual features using a model presented in Section 3.2.5.

Update

Once an observation has been performed, the state estimate and covariance are updated with the EKF update equations

$$\hat{\mathbf{x}}^+(t_k) = \hat{\mathbf{x}}^-(t_k) + \mathbf{W}(t_k)\, \boldsymbol{\nu}(t_k) \tag{3.20}$$

$$\mathbf{P}^+(t_k) = \mathbf{P}^-(t_k) - \mathbf{W}(t_k)\, \mathbf{S}(t_k)\, \mathbf{W}^{\mathsf{T}}(t_k) \tag{3.21}$$

where the Kalman weights are given by

$$\mathbf{W} = \mathbf{P}^-(t_k)\, \nabla_x^{\mathsf{T}} \mathbf{h}(t_k)\, \mathbf{S}^{-1}(t_k) \tag{3.22}$$

3.2.3 Feature Initialisation

Initialisation of a new feature is performed in two steps. Firstly, registration must be performed by locating the image coordinates of a feature that can be initialised by a sonar range observation. Secondly, the state vector is augmented with the position of the new feature relative to the navigation frame.

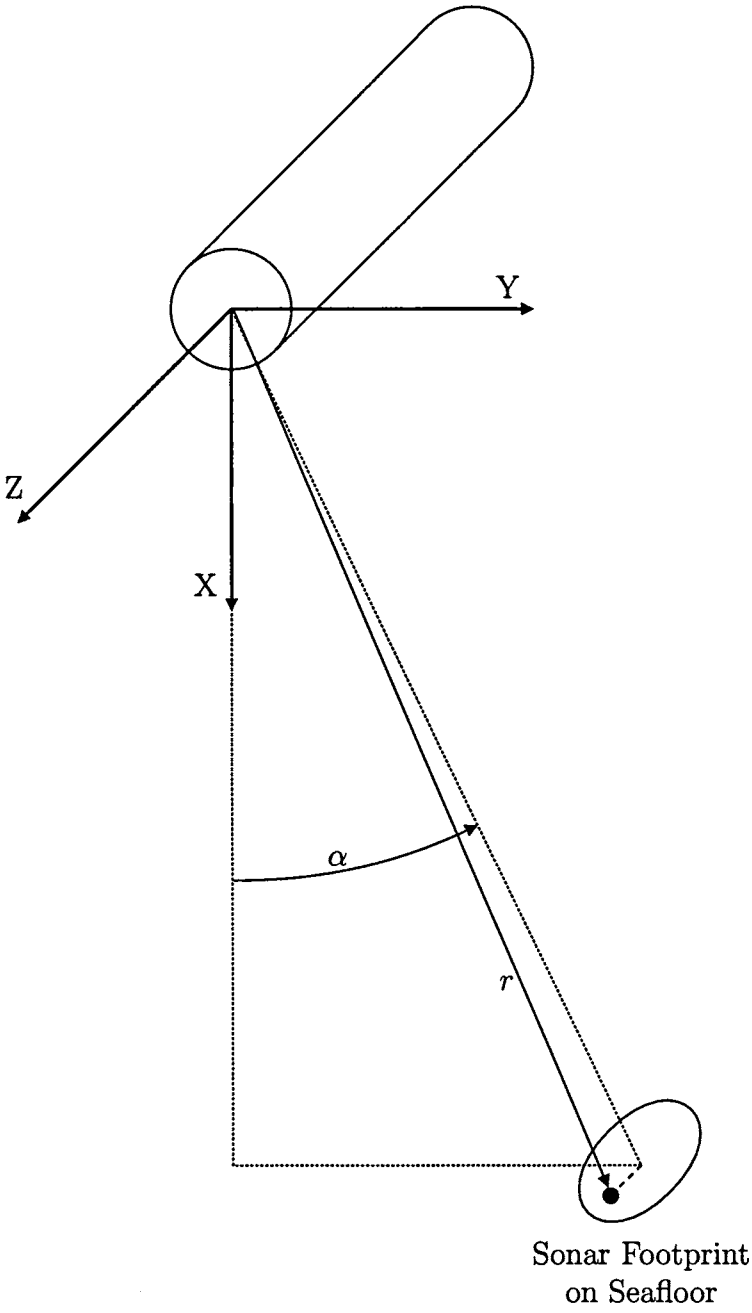


Figure 3.4: The sonar reference frame and observation states. An area of the seafloor defined by the sonar beamwidth is illuminated. The observed azimuth α corresponds to the center of the sonar beam, however the observed range r could correspond to a feature anywhere in the sonar beam.

Registration

The sonar reference frame and observation states are displayed in Figure 3.4. The location of the feature causing the sonar return within the diverging beam is unknown. Therefore when attempting to find a visual feature that can be initialised with a sonar range measurement, a feature near the middle of the sonar footprint will be found. The estimated coordinates for the middle of the sonar footprint on the seafloor (represented by the symbol o) relative to the sonar frame are

$$^{s}\mathbf{t}_{o} = \begin{bmatrix} r\cos(\alpha) \\ r\sin(\alpha) \\ 0 \end{bmatrix} \tag{3.23}$$

where r is the range measurement and α is the sonar azimuth angle as shown in Figure 3.4.

The estimated current pose of the sonar relative to the navigation frame is given by

$$^{n}\mathbf{p}_{s} = {}^{n}\hat{\mathbf{p}}_{v}^{+}(t_k) \oplus {}^{v}\mathbf{p}_{s} \tag{3.24}$$

where $^{n}\hat{\mathbf{p}}_{v}^{+}(t_k)$ is the estimated current pose of the vehicle, $^{v}\mathbf{p}_{s}$ is the pose of the sonar relative to the vehicle frame that is known from calibration, and $\oplus$ is the head to tail pose composition operation defined in Appendix A.

The coordinates of the middle of the sonar footprint in the navigation frame are then

$$^{n}\mathbf{t}_{o} = {}^{n}\mathbf{t}_{s} + {}^{n}_{s}\mathbf{R}\,{}^{s}\mathbf{t}_{o} \tag{3.25}$$

If the pose of the vehicle at the time the last image was acquired is at index j in the trajectory vector, the estimated pose of the camera in the navigation frame when the last image was acquired is given by

$$^{n}\mathbf{p}_{c} = {}^{n}\hat{\mathbf{p}}_{p_j}^{+}(t_k) \oplus {}^{v}\mathbf{p}_{c} \tag{3.26}$$

where $^{n}\hat{\mathbf{p}}_{p_j}^{+}(t_k)$ is the estimate of vehicle pose j and $^{v}\mathbf{p}_{c}$ is the pose of the camera relative to the vehicle frame known from calibration.

The image coordinates for the middle of the sonar footprint can then be calculated using the image projection function of Equation 2.8

$$\mathbf{c}_{o}(t_k) = \mathbf{proj}\left[{}^{n}\mathbf{p}_{c}, {}^{n}\mathbf{t}_{o}\right] \tag{3.27}$$

The image coordinates calculated by Equation 3.27 are unlikely to correspond to a good feature for future tracking that can be well localised. Therefore, these coordinates are used

to initialise a search for a nearby corner feature within the sonar footprint on the seafloor.

Feature Augmentation

An augmented estimate vector including the new feature states is produced using an initialisation function of the form

$$\hat{\mathbf{x}}^{*+}(t_k) = \mathbf{g}\left[\hat{\mathbf{x}}^{+}(t_k), \mathbf{z}(t_k)\right] \tag{3.28}$$

$$\begin{bmatrix} \hat{\mathbf{x}}_v^{*+}(t_k) \\ \hat{\mathbf{x}}_t^{*+}(t_k) \\ \hat{\mathbf{x}}_m^{*+}(t_k) \\ \hat{\mathbf{x}}_{f_i}^{*+}(t_k) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_v^{+}(t_k) \\ \hat{\mathbf{x}}_t^{+}(t_k) \\ \hat{\mathbf{x}}_m^{+}(t_k) \\ \mathbf{g}_f\left[\hat{\mathbf{x}}^{+}(t_k), \mathbf{z}_i(t_k)\right] \end{bmatrix} \tag{3.29}$$

in which $\mathbf{g}_f[\cdot,\cdot]$ is the feature initialisation model that relates the feature states to the observed and estimated states.

The observation vector used to initialise a feature i is

$$\mathbf{z}_i(t_k) = \begin{bmatrix} u_i(t_k) \\ v_i(t_k) \\ r_i(t_k) \end{bmatrix} \tag{3.30}$$

in which $u_i(t_k)$ and $v_i(t_k)$ are the coordinates of the feature in the latest image, and $r_i(t_k)$ is the range observed by the sonar at the current vehicle pose.

The position of the feature in the navigation frame ${}^n\mathbf{t}_i$ that is consistent with the observation vector is given by the intersection of a sphere with radius $r_i(t_k)$ that is centred at the estimated current pose of the sonar ${}^n\mathbf{t}_s$, and a ray passing through the of the previous camera position ${}^n\mathbf{t}_c$ with a direction defined by the image coordinates $u_i(t_k)$ and $v_i(t_k)$. The initialised feature position is not the middle of the sonar footprint as calculated in Equation 3.25, since the observed image coordinates $u_i(t_k)$ and $v_i(t_k)$ correspond to a feature near (but not necessarily exactly at) the image coordinates of the center of the sonar footprint as calculated in 3.27.

The sphere and ray constraints are defined by the equations

$$\left({}^n\mathbf{t}_i - {}^n\mathbf{t}_s\right) \cdot \left({}^n\mathbf{t}_i - {}^n\mathbf{t}_s\right) = r_i^2(t_k) \tag{3.31}$$

$${}^n\mathbf{t}_i = {}^n\mathbf{t}_c + \lambda \mathbf{d}_i \tag{3.32}$$

where the direction of the ray from the camera to feature i in the navigation frame is

$$\mathbf{d}_i = {}^n_c\mathbf{R} \begin{bmatrix} (u_i(t_k) - p_u)/f_u \\ (u_i(t_k) - p_v)/f_v \\ 1 \end{bmatrix} \tag{3.33}$$

in which p_u, p_v, f_u and f_u are the intrinsic camera parameters defined in Chapter 2.

Substituting Equation 3.32 into Equation 3.31 results in the quadratic

$$A\lambda^2 + B\lambda + C = 0 \tag{3.34}$$

where the coefficients are

$$A = \mathbf{d}_i \cdot \mathbf{d}_i \tag{3.35}$$

$$B = 2({}^n\mathbf{t}_c - {}^n\mathbf{t}_s) \cdot \mathbf{d}_i \tag{3.36}$$

$$C = ({}^n\mathbf{t}_c - {}^n\mathbf{t}_s) \cdot ({}^n\mathbf{t}_c - {}^n\mathbf{t}_s) - r_i^2(t_k) \tag{3.37}$$

The initialised feature position in the navigation frame is therefore

$$\mathbf{g}_f\left[\hat{\mathbf{x}}^+(t_k), \mathbf{z}_i(t_k)\right] = {}^n\mathbf{t}_c + \lambda\mathbf{d}_i \tag{3.38}$$

where λ is the solution to the quadratic in Equation 3.34

$$\lambda = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \tag{3.39}$$

The augmented state covariance matrix is generated by augmenting the observation covariance and propagating the result through the Jacobian of the initialisation function

$$\mathbf{P}^{*+}(t_k) = \nabla\mathbf{g}(t_k) \begin{bmatrix} \mathbf{P}^+_{vv}(t_k) & \mathbf{P}^+_{vt}(t_k) & \mathbf{P}^+_{vm}(t_k) & \mathbf{0} \\ \mathbf{P}^{+\mathsf{T}}_{vt}(t_k) & \mathbf{P}^+_{tt}(t_k) & \mathbf{P}^+_{tm}(t_k) & \mathbf{0} \\ \mathbf{P}^{+\mathsf{T}}_{vm}(t_k) & \mathbf{P}^{+\mathsf{T}}_{tm}(t_k) & \mathbf{P}^+_{mm}(t_k) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_i(t_k) \end{bmatrix} \nabla^{\mathsf{T}}\mathbf{g}(t_k) \tag{3.40}$$

The initialisation function Jacobian has the form

$$\nabla\mathbf{g}(t_k) = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \nabla_v\mathbf{g}_f(t_k) & \nabla_t\mathbf{g}_f(t_k) & \mathbf{0} & \nabla_z\mathbf{g}_f(t_k) \end{bmatrix} \tag{3.41}$$

where $\nabla_v \mathbf{g}_f(t_k)$, $\nabla_t \mathbf{g}_f(t_k)$ and $\nabla_z \mathbf{g}_f(t_k)$ and are the Jacobians of the feature initialisation function in Equation 3.38. with respect to the vehicle, trajectory and observation states.

$$\nabla_v \mathbf{g}_f(t_k) = \left.\frac{\partial \mathbf{g}_f}{\partial \mathbf{x}_v}\right|_{(\hat{\mathbf{x}}^+(t_k),\mathbf{z}(t_k))} \tag{3.42}$$

$$\nabla_t \mathbf{g}_f(t_k) = \left.\frac{\partial \mathbf{g}_f}{\partial \mathbf{x}_t}\right|_{(\hat{\mathbf{x}}^+(t_k),\mathbf{z}(t_k))} \tag{3.43}$$

$$\nabla_z \mathbf{g}_f(t_k) = \left.\frac{\partial \mathbf{g}_f}{\partial \mathbf{z}}\right|_{(\hat{\mathbf{x}}^+(t_k),\mathbf{z}(t_k))} \tag{3.44}$$

3.2.4 Pose Augmentation

Augmenting the state vector with the current vehicle pose is performed using the initialisation function

$$\hat{\mathbf{x}}^{*+}(t_k) = \mathbf{G}\hat{\mathbf{x}}^+(t_k) \tag{3.45}$$

$$= \begin{bmatrix} \hat{\mathbf{x}}_v^+(t_k) \\ \hat{\mathbf{x}}_t^+(t_k) \\ \mathbf{G}_v\hat{\mathbf{x}}_v^+(t_k) \\ \hat{\mathbf{x}}_m^+(t_k) \end{bmatrix} \tag{3.46}$$

in which the matrix $\mathbf{G}_v$ extracts the pose states from the vehicle state vector.

$$\mathbf{G}_v = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.47}$$

The augmented covariance matrix is

$$\mathbf{P}^{*+}(t_k) = \mathbf{G} \begin{bmatrix} \mathbf{P}_{vv}^+(t_k) & \mathbf{P}_{vt}^+(t_k) & \mathbf{P}_{vm}^+(t_k) \\ \mathbf{P}_{vt}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tt}^+(t_k) & \mathbf{P}_{tm}^+(t_k) \\ \mathbf{P}_{vm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{mm}^+(t_k) \end{bmatrix} \mathbf{G}^\mathsf{T} \tag{3.48}$$

$$= \begin{bmatrix} \mathbf{P}_{vv}^+(t_k) & \mathbf{P}_{vt}^+(t_k) & \mathbf{P}_{vv}^+(t_k)\,\mathbf{G}_v^\mathsf{T} & \mathbf{P}_{vm}^+(t_k) \\ \mathbf{P}_{vt}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tt}^+(t_k) & \mathbf{P}_{vt}^{+\mathsf{T}}(t_k)\,\mathbf{G}_v^\mathsf{T} & \mathbf{P}_{tm}^+(t_k) \\ \mathbf{G}_v\mathbf{P}_{vv}^+(t_k) & \mathbf{G}_v\mathbf{P}_{vt}^+(t_k) & \mathbf{G}_v\mathbf{P}_{vv}^+(t_k)\,\mathbf{G}_v^\mathsf{T} & \mathbf{G}_v\mathbf{P}_{vm}^+(t_k) \\ \mathbf{P}_{vm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{tm}^{+\mathsf{T}}(t_k) & \mathbf{P}_{vm}^{+\mathsf{T}}(t_k)\,\mathbf{G}_v^\mathsf{T} & \mathbf{P}_{mm}^+(t_k) \end{bmatrix} \tag{3.49}$$

3.2.5 Visual Feature Observations

A visual observation of feature i is made according to a model of the form

$$\mathbf{z}_i(t_k) = \mathbf{h}_i[\mathbf{x}(t_k)] + \mathbf{v}_i(t_k) \tag{3.50}$$

The observation vector consists of the image coordinates for feature i

$$\mathbf{z}_i(t_k) = \begin{bmatrix} u_i(t_k) \\ v_i(t_k) \end{bmatrix} \tag{3.51}$$

and the observation covariance has the form

$$\mathbf{R}_i = \begin{bmatrix} \sigma_u^2(t_k) & 0 \\ 0 & \sigma_v^2(t_k) \end{bmatrix} \tag{3.52}$$

The predicted observation is given by the image projection function of Equation 2.8

$$\mathbf{h}[\mathbf{x}(t_k)] = \mathbf{proj}[{}^n\hat{\mathbf{p}}_c(t_k), {}^n\hat{\mathbf{t}}_i^-(t_k)] \tag{3.53}$$

in which ${}^n\hat{\mathbf{p}}_c(t_k)$ is the estimated pose of the camera in the navigation frame, given by

$${}^n\hat{\mathbf{p}}_c(t_k) = {}^n\hat{\mathbf{p}}_v^-(t_k) \oplus {}^v\mathbf{p}_c \tag{3.54}$$

The Jacobian of the feature observation function with respect to the current vehicle pose is

$$\begin{aligned} \nabla_{x_v}\mathbf{h}(t_k) &= \frac{\partial \mathbf{h}}{\partial {}^n\mathbf{p}_v}\Big|_{(\hat{\mathbf{x}}^-(t_k))} \\ &= \frac{\partial \mathbf{proj}[{}^n\mathbf{p}_c, {}^n\mathbf{t}_i]}{\partial {}^n\mathbf{p}_c}\Big|_{({}^n\hat{\mathbf{p}}_c(t_k), {}^n\hat{\mathbf{t}}_i^-(t_k))} \mathbf{J}_{\oplus 1}\big|_{({}^n\hat{\mathbf{p}}_v^-(t_k), {}^v\mathbf{p}_c)} \end{aligned} \tag{3.55}$$

The Jacobian of the observation function with respect to the feature states is

$$\begin{aligned} \nabla_{f_i}\mathbf{h}(t_k) &= \frac{\partial \mathbf{h}}{\partial {}^n\mathbf{t}_i}\Big|_{(\hat{\mathbf{x}}^-(t_k))} \\ &= \frac{\partial \mathbf{proj}[{}^n\mathbf{p}_c, {}^n\mathbf{t}_i]}{\partial {}^n\mathbf{t}_i}\Big|_{({}^n\hat{\mathbf{p}}_c(t_k), {}^n\hat{\mathbf{t}}_i^-(t_k))} \end{aligned} \tag{3.56}$$

3.3 Simulation

Due to a lack of ground truth and issues with the quality of the data acquired during field deployments of the Oberon ROV, simulations will form the main verification of the motion estimation method presented in this chapter.

The simulated environment and vehicle trajectory are illustrated in Figure 3.5. A randomly generated height-map is used to model the seafloor, and the vehicle traverses a roughly rectangular path that is approximately 65 metres in length.

The feature initialisation procedure is simulated to recreate the registration errors expected in real data. A visualisation of a simulation displaying the estimated states is shown in Figure 3.6. The errors in the registration of sonar and vision observations are represented by the black lines joining seafloor locations causing a sonar return and the positions of features tracked by the vision system.

The parameters used to generate the simulated sensor data noise are listed in Table 3.1, and the process and observation model parameters used by the filter are stated in Table 3.2. The only difference is the inflated sonar range observation uncertainty used in the filter to compensate for errors when registering the sonar and vision data. These simulation parameters are based on the properties of the sensors used on the Oberon platform.

Errors in the estimated vehicle states are shown with 95% confidence bounds in Figure 3.7. The errors in the vehicle's z coordinate position and orientation states are constrained by the direct observations supplied by the depth and attitude sensors, however the errors in the vehicle's x and y coordinate position estimates drift in a manner typical of dead-reckoning. The errors in all states are well bounded by their confidence intervals, suggesting the filter is appropriately tuned.

The innovations from depth, yaw-rate and attitude observations are shown in Figures 3.8, 3.9 and 3.10. The peaks in the yaw-rate innovations after approximately 100, 185 and 270 seconds are caused by the sharp turns at the corners of the rectangular vehicle trajectory. The rapid angular accelerations occurring at these times significantly deviate from the assumptions of the constant velocity vehicle model, however the filter is able to contend with these maneuvers due to the global heading observations provided by the compass.

The innovations for visual feature observation are displayed in Figure 3.11. The consistency of the 95% confidence bounds suggest the inflated sonar range uncertainty used in the feature initialisation model is appropriately tuned for the simulated data.

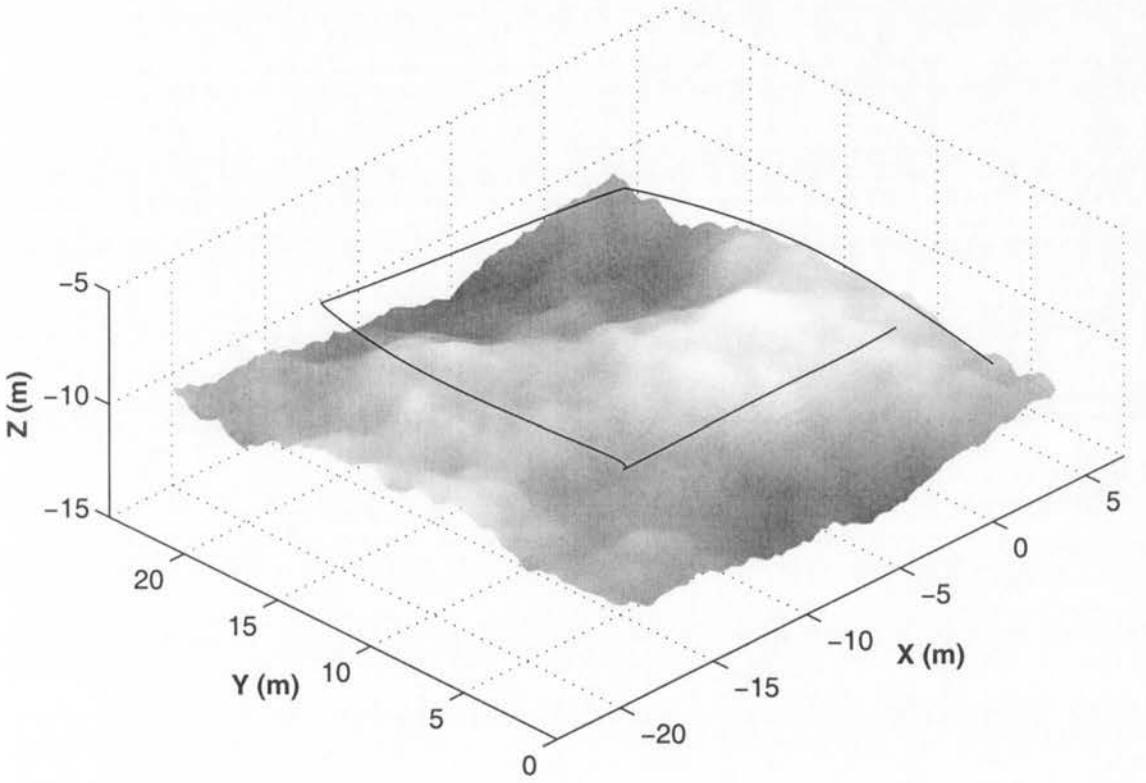


Figure 3.5: Simulation terrain and vehicle trajectory. The seafloor is modeled by a randomly generated height-map, and the roughly rectangular vehicle path has a total length of approximately 65 metres.

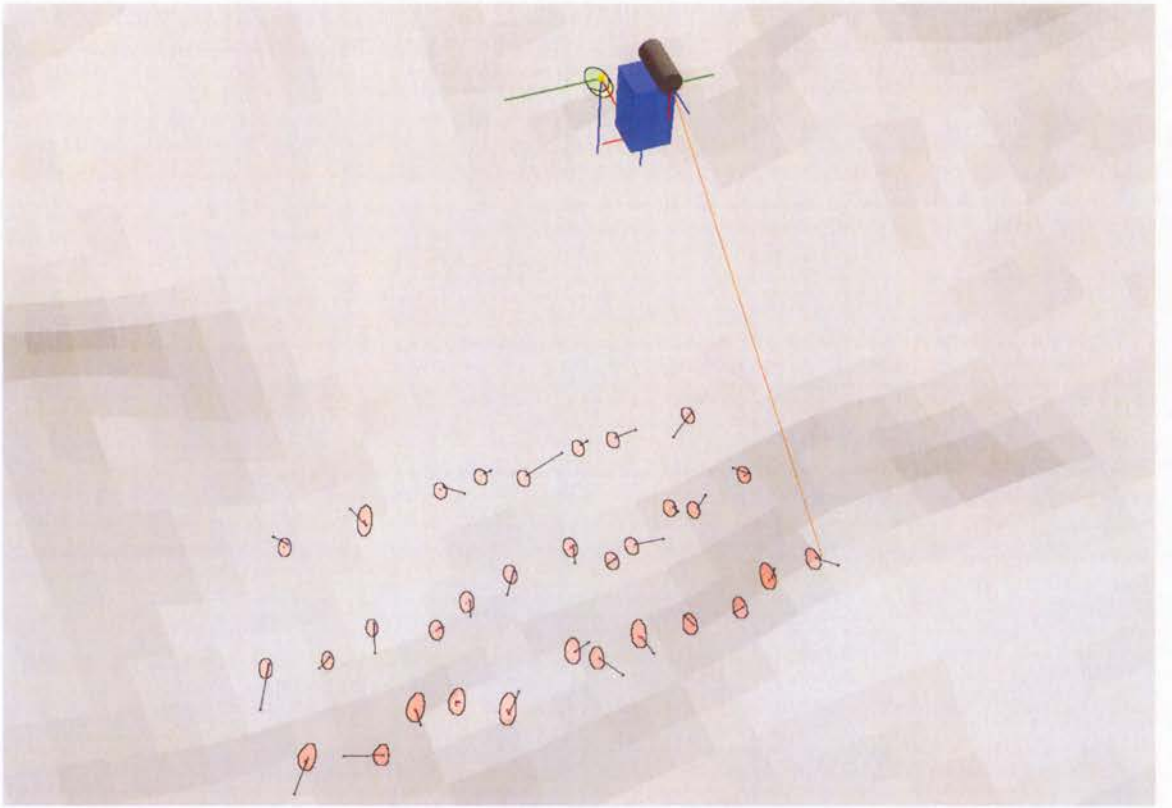


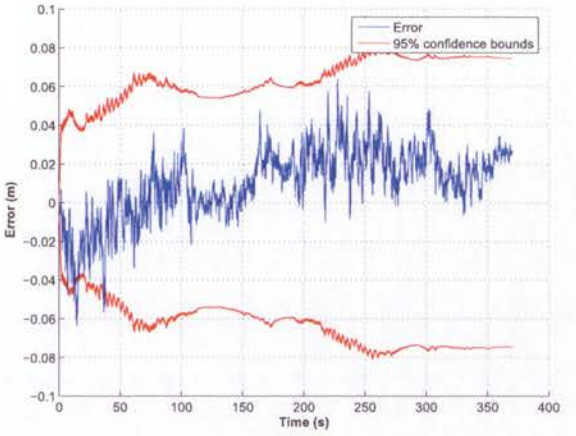
Figure 3.6: Simulation visualisation displaying estimated vehicle and feature states. The sonar is represented by a black cylinder, the camera is displayed as a blue cuboid and the vehicle frame origin is marked by a yellow sphere located just behind the two sensors. The uncertainties in the current and previous vehicle pose are represented by transparent yellow and green 95% confidence ellipsoids respectively, and the feature estimate uncertainties are shown in red. A recent sonar observation (represented by an orange ray) has been used to initialise a new feature. The seafloor locations that have caused sonar returns are marked by black spheres, while the location tracked by the vision system is shown in blue. The black lines joining the sonar and visual feature locations represent the data registration errors.

Depth observation noise standard deviation	σ_p	0.02	m
Yaw-rate observation noise standard deviation	$\sigma_{\dot{\psi}}$	0.25	deg/s
Roll tilt observation noise standard deviation	σ_ϕ	0.75	deg
Pitch tilt observation noise standard deviation	σ_θ	0.75	deg
Heading observation noise standard deviation	σ_ψ	0.75	deg
Sonar range observation noise standard deviation	σ_r	**0.05**	m
Sonar azimuth observation noise standard deviation	σ_α	0.9	deg
Sonar elevation observation noise standard deviation	σ_β	0.9	deg
Camera u-axis observation noise standard deviation	σ_u	0.5	$pixels$
Camera v-axis observation noise standard deviation	σ_v	0.5	$pixels$

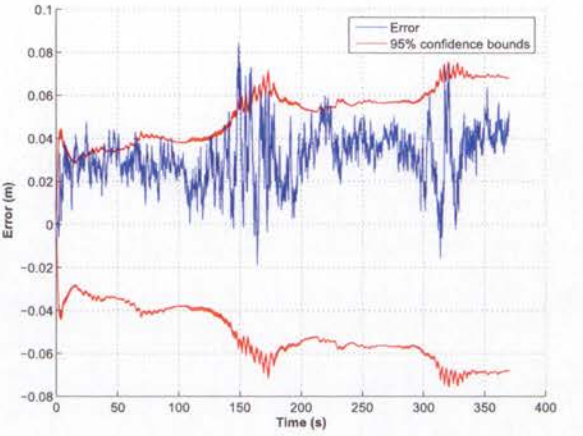
Table 3.1: Simulation sensor observation parameters. The values listed in this table have been used to generate errors in the simulated senor observations.

Vehicle model x-axis acceleration standard deviation	$\sigma_{\dot{v}_x}$	0.03	m/s^2
Vehicle model y-axis acceleration standard deviation	$\sigma_{\dot{v}_y}$	0.01	m/s^2
Vehicle model z-axis acceleration standard deviation	$\sigma_{\dot{v}_z}$	0.01	m/s^2
Vehicle model x-axis angular acceleration standard deviation	$\sigma_{\dot{p}}$	0.25	deg/s^2
Vehicle model y-axis angular acceleration standard deviation	$\sigma_{\dot{q}}$	0.25	deg/s^2
Vehicle model z-axis angular acceleration standard deviation	$\sigma_{\dot{r}}$	0.25	deg/s^2
Depth observation noise standard deviation	σ_p	0.02	m
Yaw-rate observation noise standard deviation	$\sigma_{\dot{\psi}}$	0.25	deg/s
Roll tilt observation noise standard deviation	σ_ϕ	0.75	deg
Pitch tilt observation noise standard deviation	σ_θ	0.75	deg
Heading observation noise standard deviation	σ_ψ	0.75	deg
Sonar range observation noise standard deviation	σ_r	**0.08**	m
Sonar azimuth observation noise standard deviation	σ_α	0.9	deg
Sonar elevation observation noise standard deviation	σ_β	0.9	deg
Camera u-axis observation noise standard deviation	σ_u	0.5	$pixels$
Camera v-axis observation noise standard deviation	σ_v	0.5	$pixels$

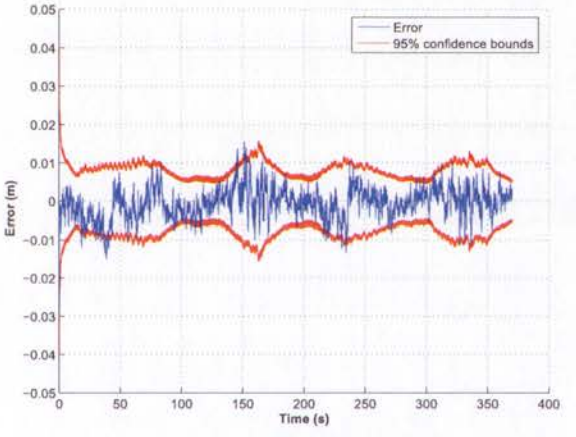
Table 3.2: Simulation process and observation model parameters. The values listed in this table have been used in the EKF process and observation models. The inflated sonar range observation uncertainty has been used to compensate for errors that occur when registering sonar and vision observations of a feature.



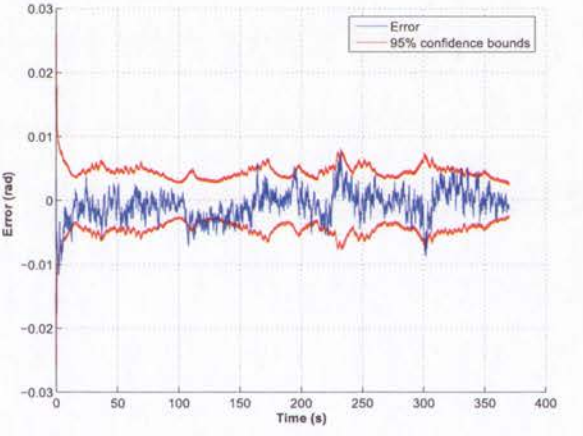
(a) X-position error



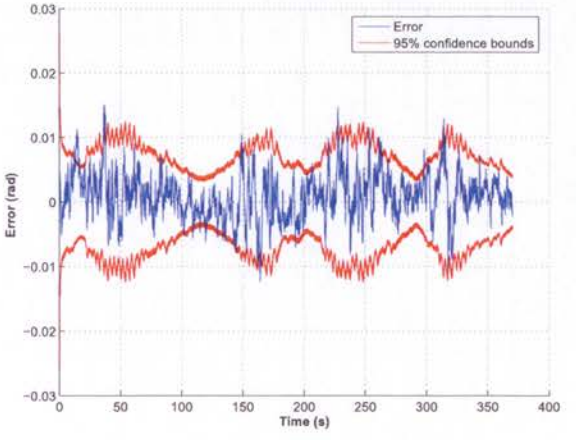
(b) Y-position error



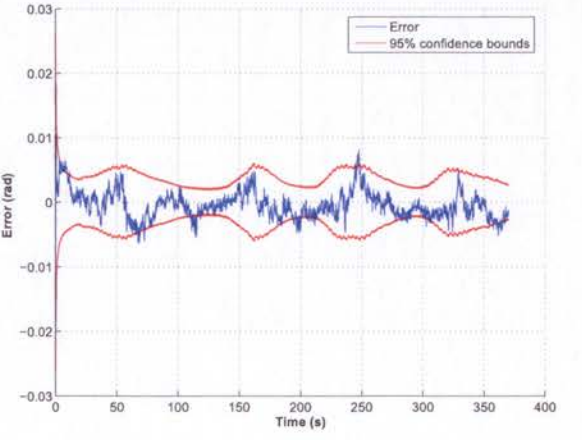
(c) Z-position error



(d) Roll Euler angle error



(e) Pitch Euler angle error



(f) Yaw Euler angle error

Figure 3.7: Simulation vehicle pose estimate errors

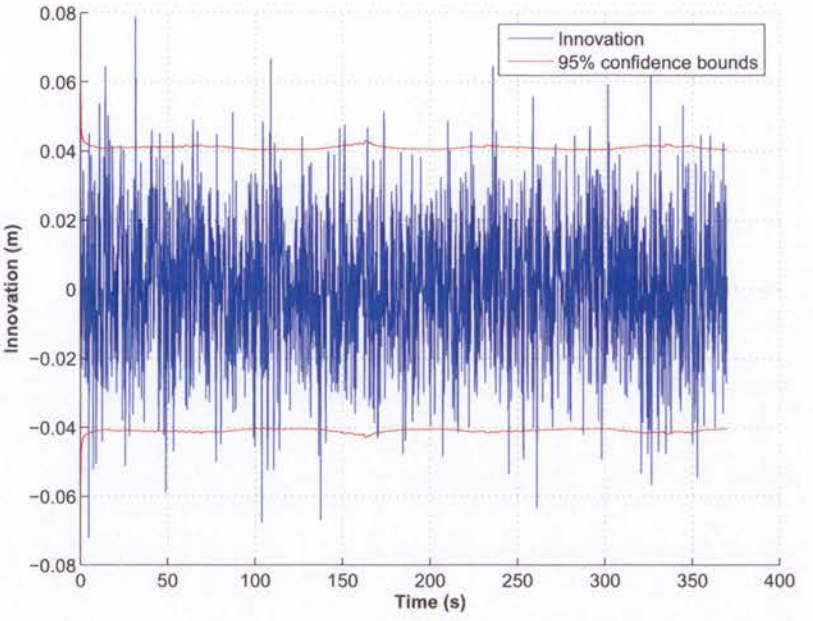


Figure 3.8: Simulation depth observation innovations

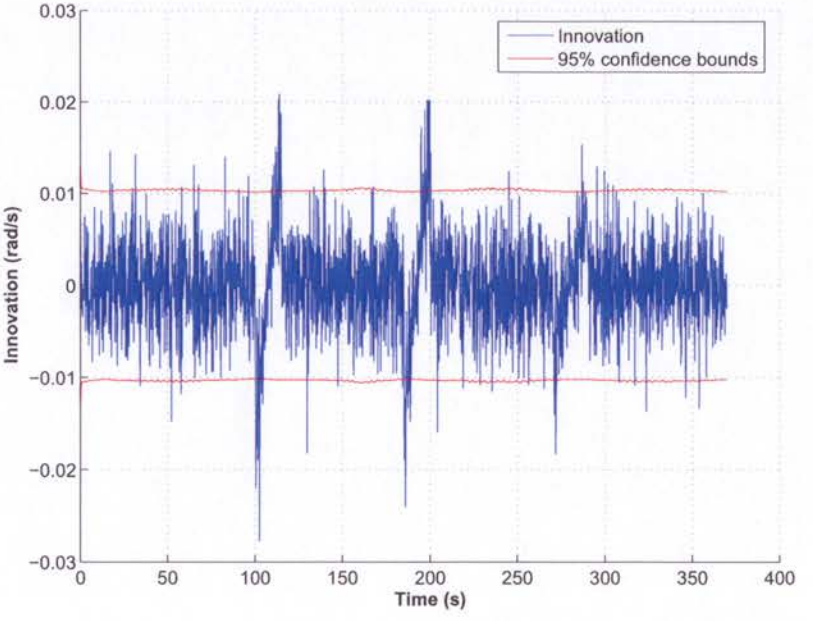
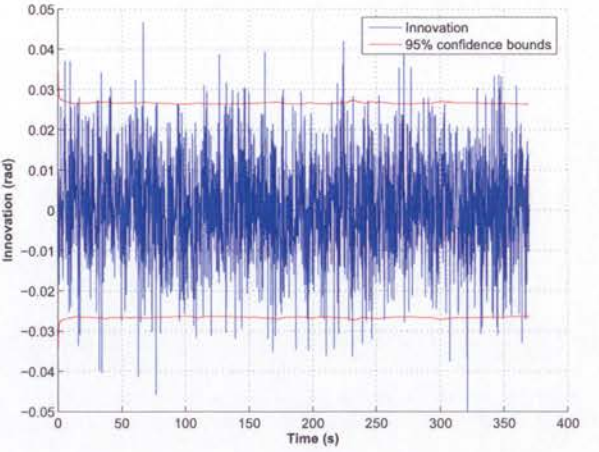
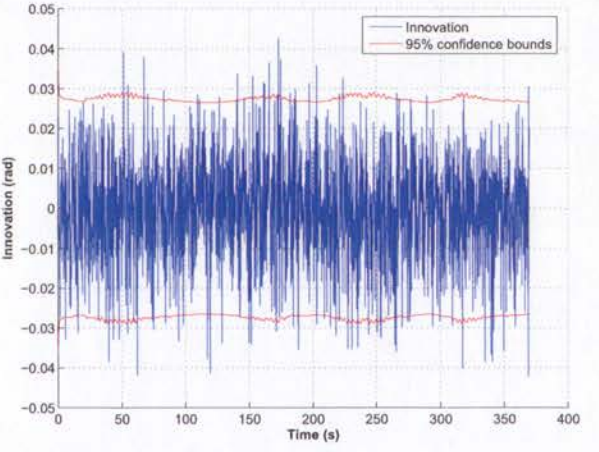


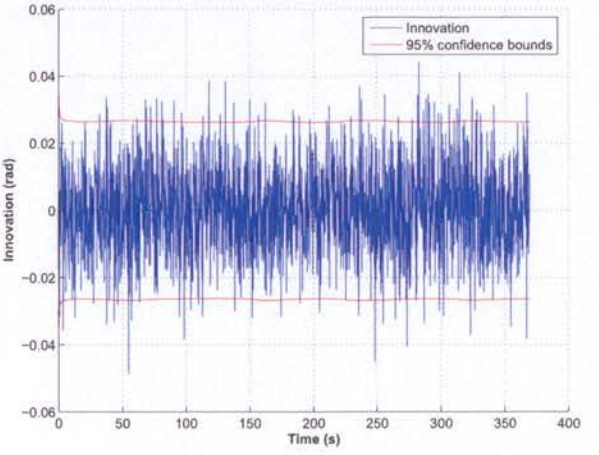
Figure 3.9: Simulation yaw-rate observation innovations



(a) Roll

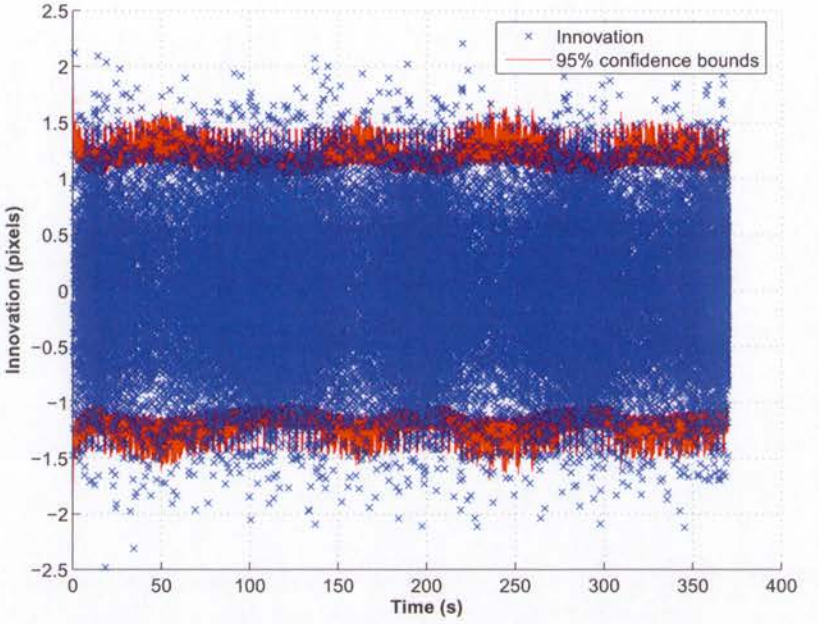


(b) Pitch

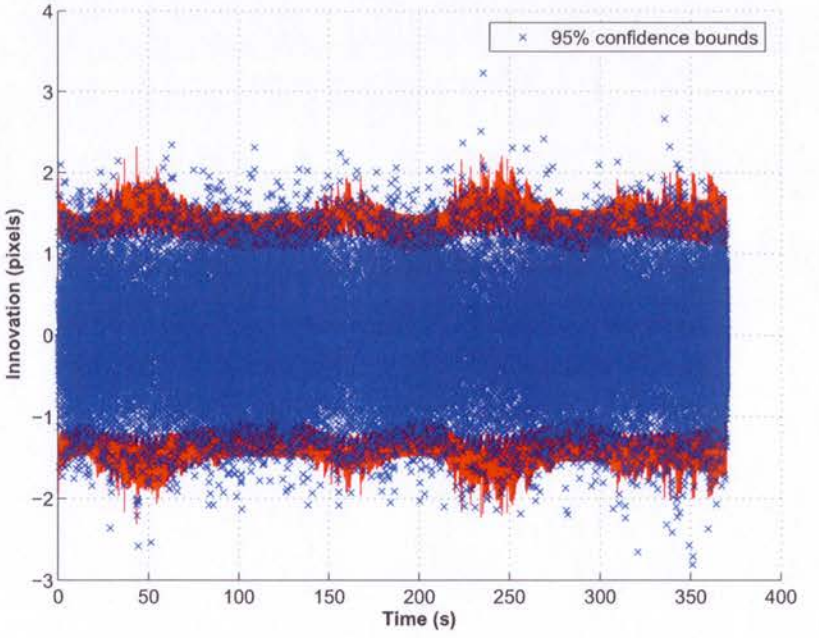


(c) Heading

Figure 3.10: Simulation vehicle attitude observation innovations



(a) u-coordinate



(b) v-coordinate

Figure 3.11: Simulation visual feature observation innovations

3.4 Results

The localisation approach presented in this chapter has been applied to data acquired during a deployment of the Oberon ROV at Ribbon Reef 3 off the coast of Queensland, Australia. The data consists of several transects over a set of coral bommies at a site called Tracy's Wonderland.

Unfortunately, the camera onboard the Oberon ROV during these deployments was configured to use an auto-focus mode resulting in varying intrinsic camera parameters. When processing this data, the visual feature observation covariance has been inflated to partially compensate for errors in the intrinsic camera calibration parameters, however the vehicle motion estimates are still corrupted by the poorly modelled observations.

The Harris corner detection algorithm has been used to extract features from the images, and the visual feature tracking process is performed using the Lucas-Kanade tracker. Figure 3.12 demonstrates the extraction and tracking of features initialised by sonar observations.

The filter parameters used to process the data acquired at Tracy's Wonderland are listed in Table 3.3. Apart from the inflated visual feature observation uncertainty, the observation model parameters are the same as those used in simulation. The process model uncertainties have been increased however, to match the large accelerations caused by surging water movement.

Vehicle model x-axis acceleration noise standard deviation	$\sigma_{\dot{v}_x}$	0.1	m/s^2
Vehicle model y-axis acceleration noise standard deviation	$\sigma_{\dot{v}_y}$	0.1	m/s^2
Vehicle model z-axis acceleration noise standard deviation	$\sigma_{\dot{v}_z}$	0.1	m/s^2
Vehicle model x-axis angular acceleration standard deviation	$\sigma_{\dot{p}}$	0.75	deg/s^2
Vehicle model y-axis angular acceleration standard deviation	$\sigma_{\dot{q}}$	0.75	deg/s^2
Vehicle model z-axis angular acceleration standard deviation	$\sigma_{\dot{r}}$	0.75	deg/s^2
Depth observation noise standard deviation	σ_p	0.02	m
Yaw-rate observation noise standard deviation	$\sigma_{\dot{\psi}}$	0.25	deg/s
Roll tilt observation noise standard deviation	σ_ϕ	0.75	deg
Pitch tilt observation noise standard deviation	σ_θ	0.75	deg
Heading observation noise standard deviation	σ_ψ	0.75	deg
Sonar range observation noise standard deviation	σ_r	0.08	m
Sonar azimuth observation noise standard deviation	σ_α	0.9	deg
Sonar elevation observation noise standard deviation	σ_β	0.9	deg
Camera u-axis observation noise standard deviation	σ_u	1.5	$pixels$
Camera v-axis observation noise standard deviation	σ_v	1.5	$pixels$

Table 3.3: Vehicle and observation model parameters for experiments at Tracy's Wonderland.

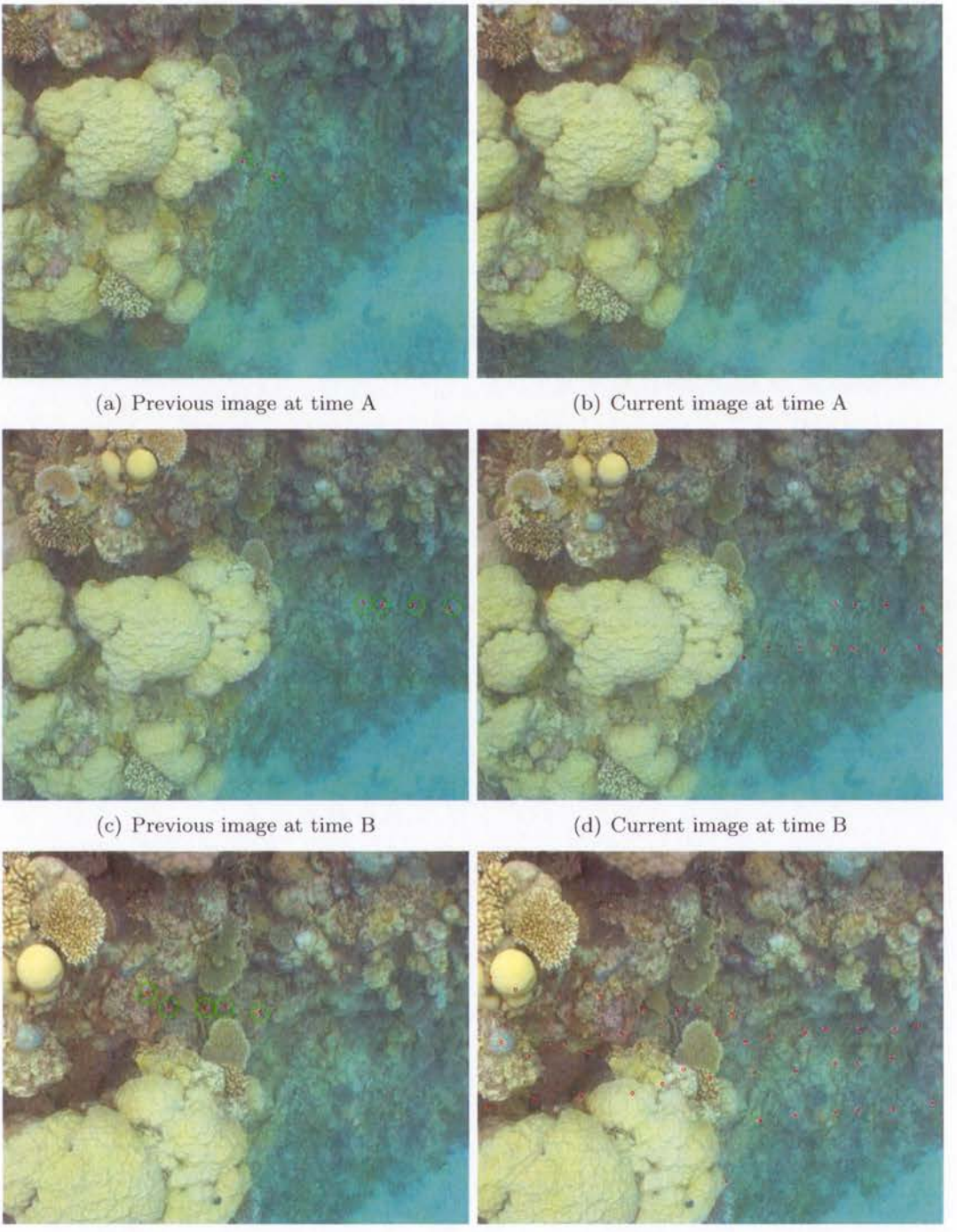
The depth observation innovations for a first transect at Tracy's Wonderland are shown in Figure 3.13. The obvious inconsistency of the innovation sequence is a result of the camera modelling errors (if the visual feature observations are ignored, the depth innovations are well bounded by the 95% confidence intervals). The large innovations are caused by biased visual feature observations resulting in the vehicle estimate converging to an incorrect solution that is incompatible with the (valid) depth observations. While not observable, the incorrect camera model parameters are likely to be causing similar errors in the x and y coordinate position estimates.

The yaw-rate and attitude observation innovations for the first transect are shown in Figures 3.14 and 3.15. The spikes in the yaw and yaw-rate observation innovations after 130 seconds are caused by a rapid change of orientation.

The innovations from visual observations of features are shown in Figure 3.16. The inconsistency of feature observations caused by the imperfect camera calibration is shown by the large magnitude of the innovations. The inflated visual observation uncertainty can be seen to result in correspondingly large confidence bounds which allow most observations to pass a normalised innovation gate test.

The estimated vehicle trajectory for the first transect is shown in Figure 3.17. The irregularities of the estimated trajectory correlate well with the surge-induced vehicle motion apparent in the image sequence. An environment model generated for the first transect generated from the trajectory estimate is shown in Figure 3.18. The environment model was created by registering all sonar returns against the estimated vehicle trajectory, creating a surface mesh by interpolating between the sonar samples, and overlaying the vision data by projecting the images onto the terrain structure. While no ground truth is available for this deployment, the consistency of the structure and visual data suggest reasonable accuracy in the estimated trajectory.

The estimated trajectory for a second transect performed at Tracy's Wonderland is presented in Figure 3.19. The relatively smooth appearance of the trajectory relative to that of transect one is due to a larger commanded velocity preventing the vehicle from being pushed backwards by the surging water movement. A reconstructed environment model for this transect is shown in Figure 3.20.



(a) Previous image at time A

(b) Current image at time A

(c) Previous image at time B

(d) Current image at time B

(e) Previous image at time C

(f) Current image at time C

Figure 3.12: Feature extraction and tracking on the Oberon ROV. The images on the left-hand side show sonar observations that have been used to initialise the range to features. The estimated sonar footprint on the seafloor is marked by a green ellipse, and the image coordinates of new features are marked with red and white circles. The images on the right-hand side demonstrate tracking of the newly initialised features.

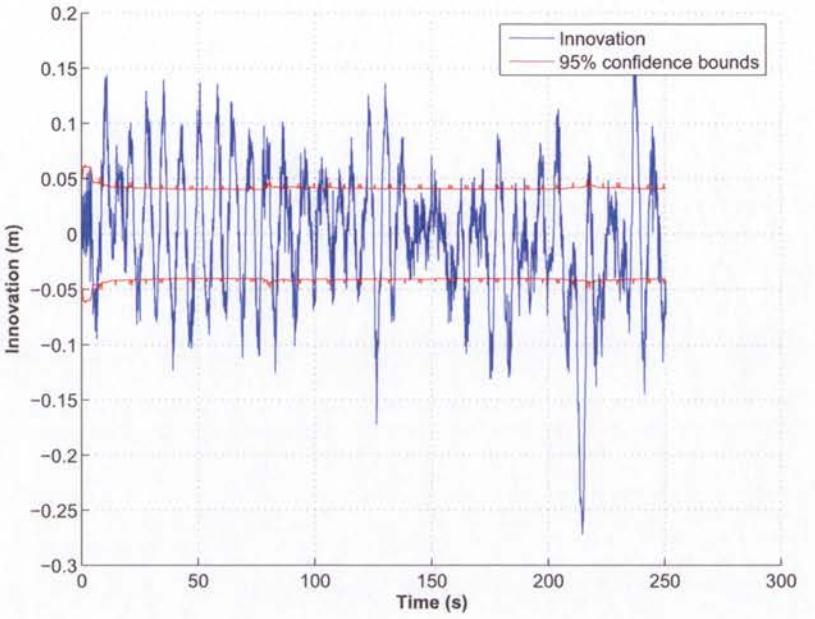


Figure 3.13: Depth observation innovations for Tracy's Wonderland transect one. The large and correlated innovations are caused by biased visual feature observations resulting in the vehicle estimate converging to an incorrect solution that is incompatible with valid depth observations.

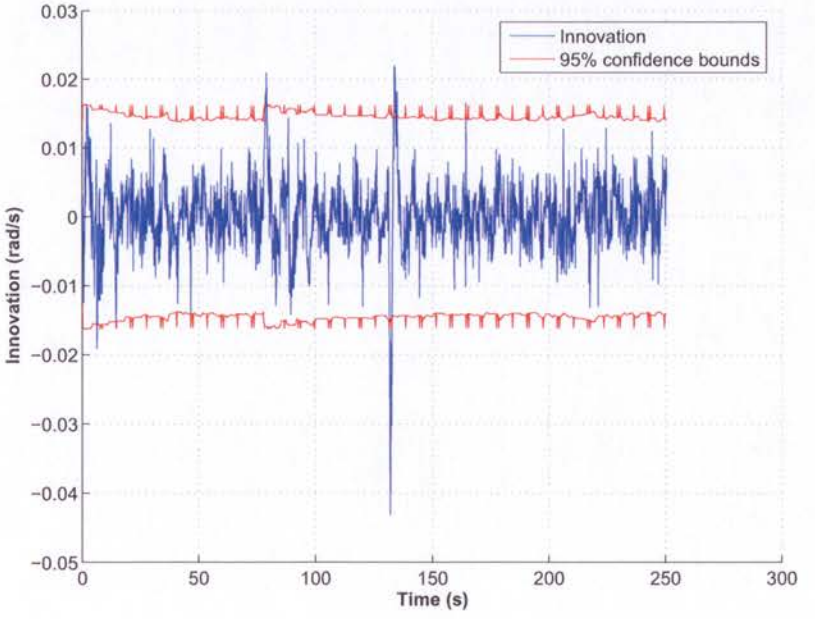
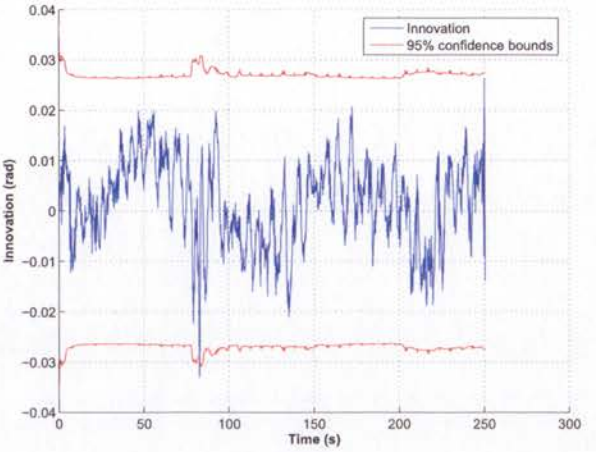
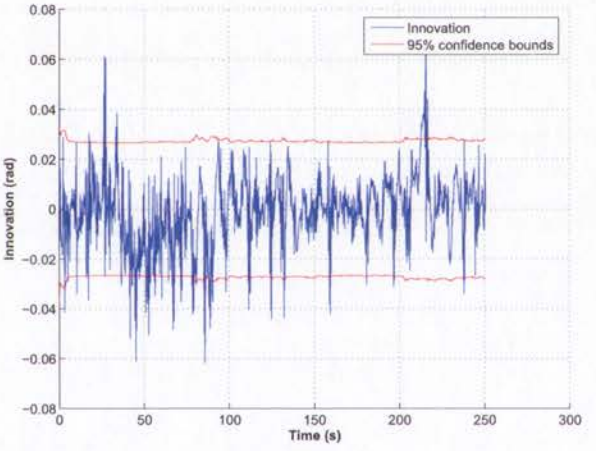


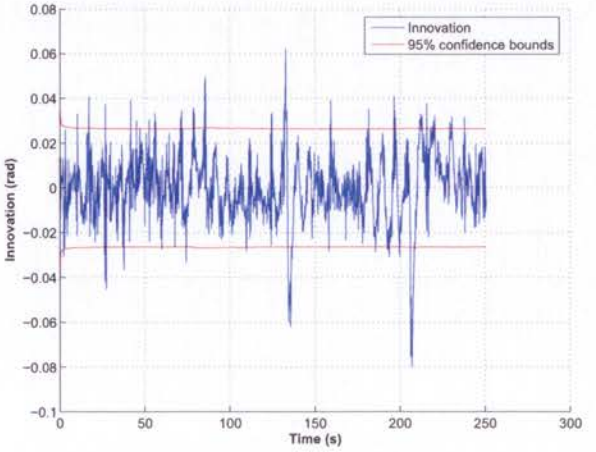
Figure 3.14: Yaw-rate observation innovations for Tracy's Wonderland transect one. The spike after approximately 130 seconds is caused by a rapid change of vehicle orientation.



(a) Roll

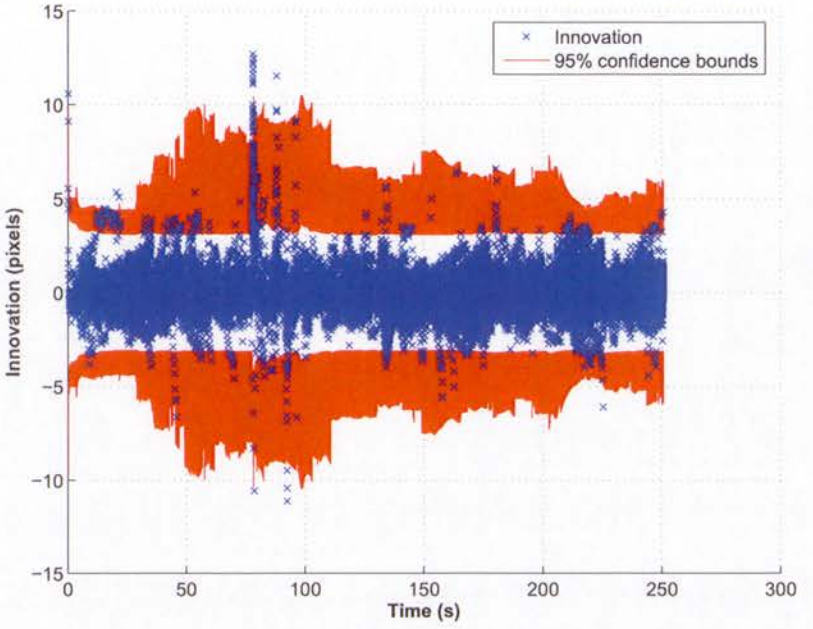


(b) Pitch

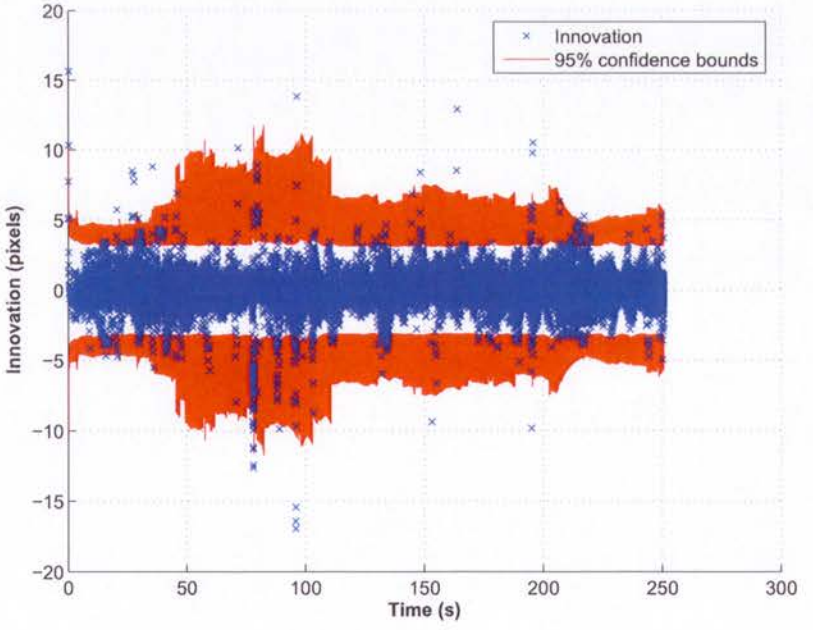


(c) Heading

Figure 3.15: Vehicle attitude observation innovations for Tracy's Wonderland transect one.



(a) u-coordinate



(b) v-coordinate

Figure 3.16: Visual feature observation innovations for Tracy's Wonderland transect one. The large innovations present are due by the use of incorrect camera calibration parameters.

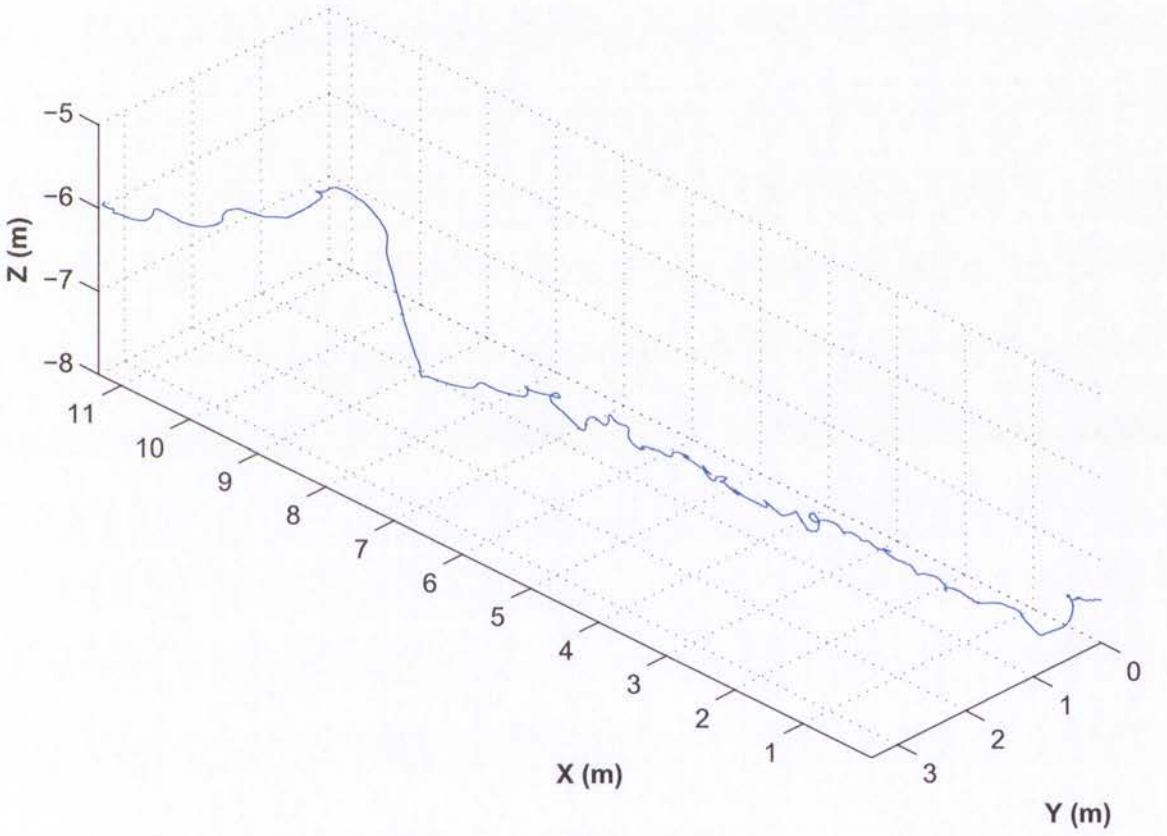


Figure 3.17: Estimated vehicle trajectory for Tracy's Wonderland transect one. The estimated vehicle path has reconstructed the irregularities in vehicle motion produced by the surging water movement.

(a) Overhead view of complete transect

(b) Close-up view of showing structure

Figure 3.18: Reconstructed environment model for Tracy's Wonderland transect one.

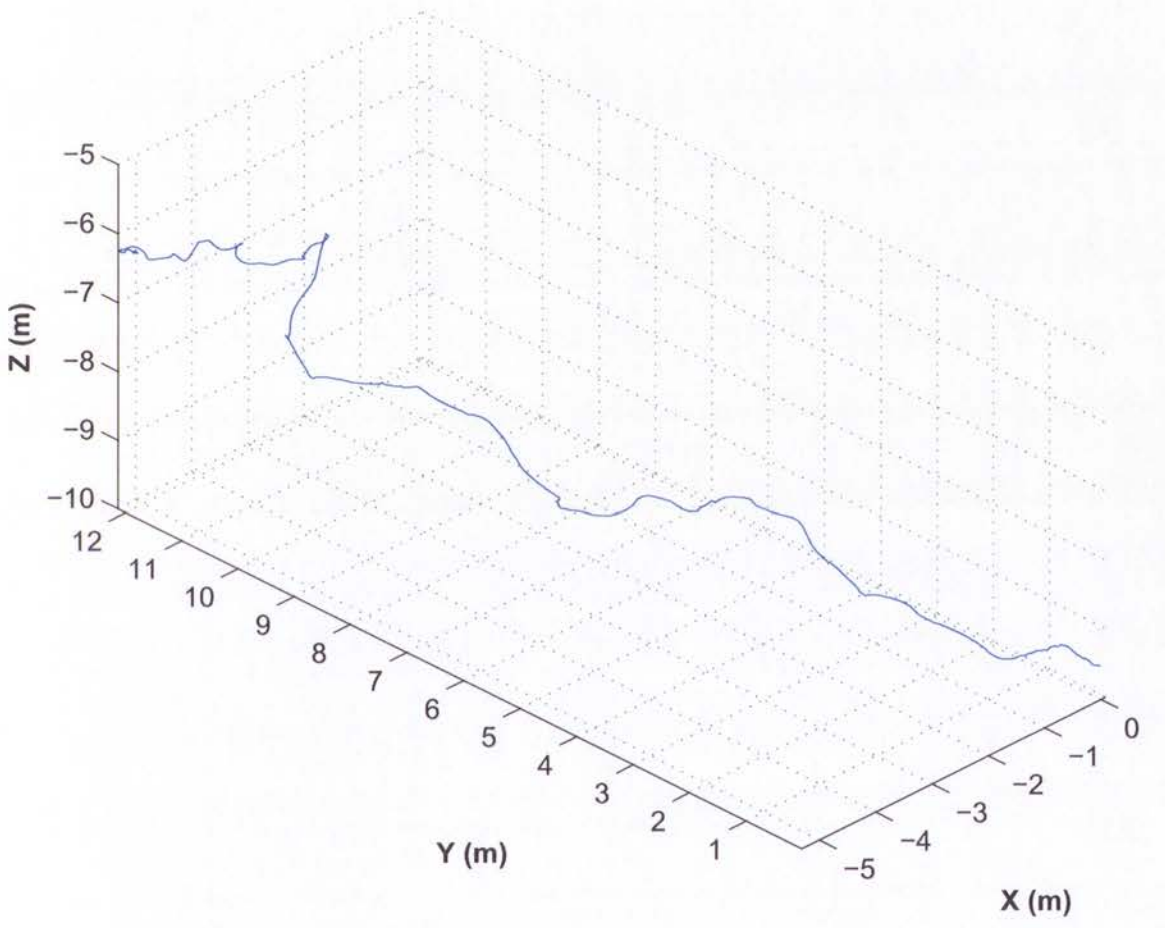


Figure 3.19: Estimated vehicle trajectory for Tracy's Wonderland transect two. The relatively smooth appearance of the trajectory relative to that of transect one shown in Figure 3.17 is due to a larger commanded velocity preventing the vehicle from being pushed backwards by wave action.

(a) Overhead view of complete transect

(b) Close-up view of showing structure

Figure 3.20: Reconstructed environment model for Tracy's Wonderland transect two.

3.5 Limitations

The motion estimation method presented in this chapter is complicated by two problems relating to the selection of process and observations model parameters:

1. The uncertainty of sonar range observations needs to be inflated to compensate for the inexact registration of sonar and vision data. The registration errors are a function of the seafloor topography however, so the range uncertainty will need to be tuned differently for operation in flat sandy areas or reef environments containing large depth variations.

2. The process model parameters need to be tuned to match the accelerations of the vehicle. In addition to intentional motion produced by the vehicle's thrusters, this also includes external influences from environmental conditions such as currents and swell. Unfortunately, the magnitude of these external influences is unlikely to be known in advance and will change over time.

An additional concern with this approach is the influence of bad sonar observations. While visual observations of poorly initialised features can be rejected using an innovation gate if the motion of the vehicle is well constrained, in this application without any dead-reckoning sensors, a large amount of uncertainty is injected into the vehicle pose states in each prediction step. It is therefore likely that observations of poorly initialised features could be accepted into the filter, particularly when the number of estimated features is low.

The robustness of this motion estimation method could be improved by adding a cheap inertial system. Using an inertial system would allow the accelerations of the vehicle to be observed instead of being treated as disturbances to the constant velocity vehicle model, making environmental conditions less influential in the selection of process model parameters. Additionally, the motion estimates produced by the inertial system would aid the rejection of poorly initialised features. In a combined system, the sonar and visual feature observations could constrain the dead-reckoning drift produced by integrating the inertial observations, and allow the biases in the inertial sensors to be estimated.

3.6 Summary

This chapter has presented a novel approach to estimate the motion of an underwater vehicle. A SLAM-style algorithm is used to provide odometry from a set of natural features tracked using a vision system. The feature positions are initialised by fusing a visual observation with a range measurement acquired with a sonar. The vehicle pose at the time of

the last image is maintained in the state vector to improve registration of sonar and visual observations.

Results have been shown for short transects in simulation and on data acquired during deployment of the Oberon ROV at Tracy's Wonderland in Queensland, Australia. While the real-world results are corrupted by the use of camera with varying intrinsic parameters, simulations show the filter can be made consistent if the process and observation models are properly tuned.

The problem of selecting model parameters that are a dependent on environmental conditions is a limitation of the proposed approach. Additional sensors such as an IMU could improve reliability. In such a system, the information from the sonar and vision sensors could be used to restrict the drift of the inertial navigation solution.

An alternative method for odometry estimation based on stereo vision will be presented in Chapter 4.

Chapter 4

Relative Pose Estimation from Stereo Vision

4.1 Introduction

This chapter investigates methods to estimate the motion of a vehicle using stereo-vision. Potential applications include odometry estimation for small or low-cost AUVs, or providing loop-closure observations within a SLAM framework to correct dead-reckoning drift.

4.1.1 Requirements

The properties of an AUV are significantly different from the indoor and ground vehicles that have been the focus of most robotics research. Since the motion of an AUV is not constrained to a ground plane, a complete six degree of freedom motion solution must be estimated.

Additional requirements for the motion estimation algorithm result from the need for an AUV to supply artificial lighting when operating at depths where natural light is not available. Since lighting is a significant drain on the limited power available onboard an AUV, low frame-rates are typically used to prolong mission times. A requirement for the relative pose estimation system developed in this chapter is therefore the ability to operate using only two stereo image pairs containing a small amount of overlap.

To enable the stereo-vision relative pose estimates to be fused with observations from additional sensors in a SLAM algorithm, the motion estimator must be unbiased, and a consistent covariance for the estimate errors must be generated. Additionally, individual visual

motion estimates must be independent from one another, and independent from other motion observations applied to the filter.

4.1.2 Related Research

Estimating the motion of a robot using a stereo-rig has been an active research subject for several decades. The most common framework for solving this problem involves finding the motion parameters that align sparse sets of triangulated features that are associated using visual feature descriptions. This approach was first demonstrated on indoor research platforms in the 1970s with the aim of future applications in planetary exploration and submersible vehicles [78]. Improvement in feature association algorithms and motion estimation error models have resulted in recent implementations of this framework being applied to ground vehicles [83, 87] in addition to the originally foreseen applications of Mars rovers [13, 86] and underwater vehicles [23, 53].

An alternative stereo odometry approach [76] involves the use of the Iterative Closest Point (ICP) algorithm [6] to optimise both features associations and the motion parameters. From an initial motion hypothesis, ICP performs an iterative process of guessing feature associations (by matching closest points) and optimising the motion estimate for the current association hypotheses until convergence occurs. ICP has often been used in robotics to calculate the motion of a platform using 3D range scanners such as lasers where no distinguishing description of each feature is available. However, since rich visual feature descriptions are available, repeatedly guessing associations using geometry is redundant for visual motion applications.

A related problem, known as 'structure from motion', has been extensively studied by the computer vision community [50]. Structure from motion typically involves estimating the motion of a monocular vision system and the position of a set of observed features. An efficient method to calculate the maximum likelihood structure from motion parameters, known as 'bundle adjustment' [50, 106], has been adapted for stereo odometry applications [24, 103].

4.1.3 The Relative Pose Estimation Process

A typical relative pose estimation algorithm can be divided into four stages: feature extraction, feature association, outlier rejection and motion estimate optimisation. A process diagram illustrating the function of each stage is shown in Figure 4.1.

The first two steps in this process can be performed using the algorithms described in Chapter 2. A prior motion estimate can be used to make tracking or association more efficient by

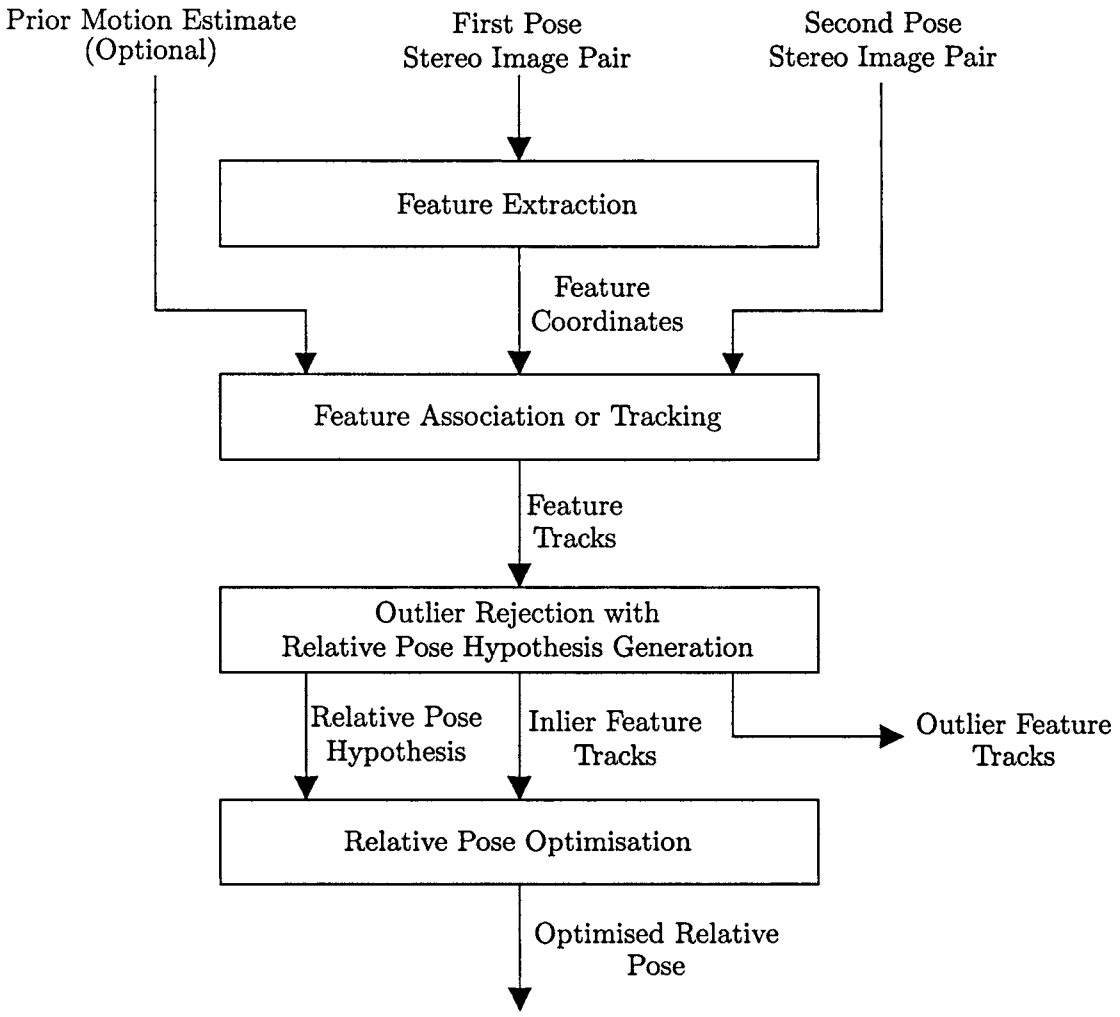


Figure 4.1: The stereo-vision relative pose estimation process

initialising a tracker with better initial location hypotheses or reducing the set of plausible matches evaluated in an association algorithm. The compatibility of using a motion prior with the independence requirements for relative pose estimates will be discussed in Section 4.4

In the outlier rejection stage, poorly tracked or incorrectly associated feature are removed to prevent corruption of the final motion estimate. This is typically performed using a hypothesise and verify framework, in which multiple motion hypotheses are generated and evaluated by testing the consistency of the feature observations.

In the last stage of the relative pose estimation process, the remaining inlier features are used to generate a final motion estimate. Most motion estimation methods require optimisation

of a non-linear cost function, which is performed using iterative algorithms such as Gauss-Newton or Levenberg-Marquardt that require an initial parameter hypothesis.

Due to the existence of satisfactory solutions, the feature extraction and association problems will not receive further attention in this chapter. Motion estimation algorithms suitable for the production of motion hypotheses within an outlier rejection framework and the calculation of a final motion estimate will be considered in Section 4.2. Outlier rejection frameworks are presented and evaluated in Section 4.3.

4.2 Relative Pose Estimation

In this section, a series of motion estimation algorithms are analysed and tested in simulations approximating odometry and loop-closure conditions. The simulations have been performed to gain insights into the consequences of approximations made in the motion estimation approaches, and an understanding of the conditions in which the approximations are valid. A new approach providing a good trade-off between computational efficiency and accuracy will be proposed, and the suitability of each method for the motion hypothesis generation and optimisation roles discussed in Section 4.1.3 will be discussed.

4.2.1 Methods

Minimisation of 3D Registration Error Euclidean Distances

Estimating the motion of a stereo-rig using by minimising Euclidean registration errors was first demonstrated in pioneering vision research applied to indoor robots [78], and has more recently been used to calculate stereo odometry on ground vehicles [61, 71] and an underwater vision system [53].

Figure 4.2 shows a stereo rig at two poses (a and b) and the positions of a feature relative to each pose (${}^a\mathbf{t}_i$ and ${}^b\mathbf{t}_i$ respectively). The feature positions are related by the coordinate transformation function

$$ {}^b\mathbf{t}_i = {}^b_a\mathbf{R}({}^a\mathbf{t}_i - {}^a\mathbf{t}_b) \tag{4.1} $$

If a set of at least three points are observed by the stereo-rig relative at both locations, the pose of frame b relative to pose a can be calculated. In this estimation problem, the parameter vector $\mathbf{x}$ consists only of the relative pose states

$$ \mathbf{x} = \begin{bmatrix} {}^a\mathbf{p}_b \end{bmatrix} \tag{4.2} $$

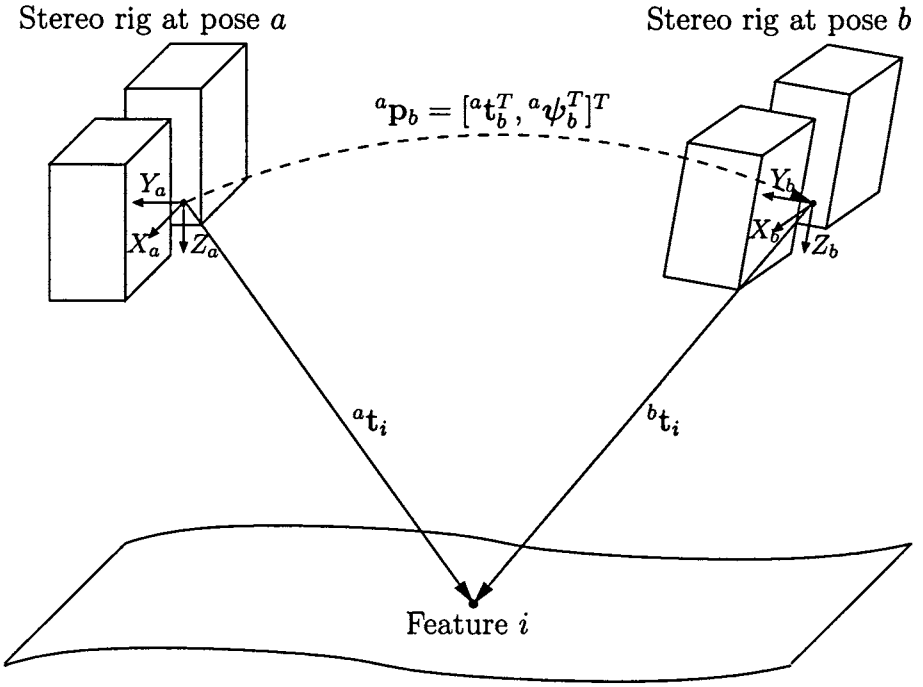


Figure 4.2: The position of feature i is shown relative to two poses of a stereo-rig: a and b. The stereo-rig's reference frame is shown coincident with the left camera.

Let $\mathbf{z}_{ai}$ and $\mathbf{z}_{bi}$ be the observed positions of feature i relative to poses a and b respectively. A prediction for the position of feature i relative to pose b can be generated by propagating $\mathbf{z}_{ai}$ through the coordinate transformation function in Equation 4.1 using the current relative pose estimate ${}^a\hat{\mathbf{p}}_b$. The difference between the observed and predicted feature position relative to frame b is the 3D registration error

$$\epsilon_i = \mathbf{z}_{bi} - {}^b_a\mathbf{R}(\mathbf{z}_{ai} - {}^a\hat{\mathbf{t}}_b) \tag{4.3}$$

The registration error measures the fit of the motion model parameters to the feature observations. For a complete vector of residuals of n features $\epsilon = \left[\epsilon_1^\mathsf{T}, \epsilon_2^\mathsf{T}, \ldots, \epsilon_n^\mathsf{T}\right]^\mathsf{T}$, the relative pose parameters that minimise the sum of square registration error Euclidean distances is defined by

$$\begin{aligned} \hat{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\epsilon^\mathsf{T}\epsilon\right) \\ &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \epsilon_i^\mathsf{T}\epsilon_i\right) \end{aligned} \tag{4.4}$$

The most attractive property of this approach is the existence of a closed form solution,

Input:

- The observed positions of a set of n points relative to frames a and b: $\{\mathbf{z}_{a1}, \mathbf{z}_{a2}, ..., \mathbf{z}_{an}\}$ and $\{\mathbf{z}_{b1}, \mathbf{z}_{b2}, ..., \mathbf{z}_{bn}\}$.

Output:

- A relative pose estimate ${}^a\hat{\mathbf{p}}_b = [{}^a\hat{\mathbf{t}}_b^\mathsf{T}, {}^a\hat{\boldsymbol{\psi}}_b^\mathsf{T}]^\mathsf{T}$ that minimises the sum of square registration errors $\sum_{i=1}^{n} \left(\boldsymbol{\epsilon}_i^\mathsf{T} \boldsymbol{\epsilon}_i\right)$, where $\boldsymbol{\epsilon}_i$ is defined in Equation 4.3.

Procedure:

1: $\mathbf{a} \leftarrow \sum_{i=1}^{n} (\mathbf{z}_{ai})/n$

2: $\mathbf{b} \leftarrow \sum_{i=1}^{n} (\mathbf{z}_{bi})/n$

3: $\mathbf{C} \leftarrow \sum_{i=1}^{n} (\mathbf{z}_{bi} - \mathbf{b})(\mathbf{z}_{ai} - \mathbf{a})^\mathsf{T}$

4: $[\mathbf{U}, \mathbf{D}, \mathbf{V}] \leftarrow \mathbf{svd}(\mathbf{C})$

5: $\mathbf{S} \leftarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{det}(\mathbf{UV}^\mathsf{T}) \end{bmatrix}$

6: ${}^b_a\mathbf{R} \leftarrow \mathbf{USV}^\mathsf{T}$

7: ${}^a\hat{\mathbf{t}}_b \leftarrow \mathbf{a} - {}^b_a\mathbf{R}^\mathsf{T}\mathbf{b}$

8: ${}^a\hat{\boldsymbol{\psi}}_b \leftarrow \mathbf{matrixToEuler}({}^b_a\mathbf{R})$

Algorithm 4.1: Calculation of the relative pose minimising the sum of square registration error Euclidean distances.

allowing for efficient calculation of the optimal parameters. Equation 4.4 is typically solved using an algorithm based on the singular value decomposition (SVD) matrix factorisation [48, 107]. The psuedocode of the implementation used to test this motion estimation method in simulation is listed in Algorithm 4.1.

Some variants of this approach weight individual registration errors with the inverse of the distances to the features, since distant features are triangulated less accurately than those closer to the cameras. In any case, this method does not correctly consider the uncertainties in triangulated 3D feature positions that are known to have larger errors in depth than other directions.

Minimisation of 3D Registration Error Mahalanobis Distances

Estimating the motion of a stereo-rig by minimising registration error Mahalanobis distances was proposed in [73], where it is suggested that the the least-squares Euclidean distance

method produces noisy motion estimates due to the fact that it ignores the uncertainties in the triangulated feature positions. Fitting a Gaussian distribution to each triangulated feature and weighting the registration errors appropriately was shown to result in an improved motion estimator. This method was originally applied to indoor robots and has since been applied to ground vehicles on Earth [85, 87] and Mars [13].

If the distribution of errors in the triangulated 3D feature estimates relative to the stereo-rig frames a and b are Gaussian with covariance Σ_{ai} and Σ_{bi} respectively, the covariance of the registration error ϵ_i defined in Equation 4.3 is

$$\Sigma_{\epsilon_i}[\mathbf{x}] = \Sigma_{bi} + {}^{b}_{a}\mathbf{R}\,(\Sigma_{ai})\,{}^{b}_{a}\mathbf{R}^{\mathsf{T}} \tag{4.5}$$

Note that the registration error covariance is a function of the estimated motion parameters since the uncertainty of the feature positions must be propagated to a common reference using the current relative pose estimate.

Assuming independent feature observations, the complete covariance of the residuals for n features is the block diagonal matrix

$$\Sigma_{\epsilon}[\mathbf{x}] = \begin{bmatrix} \Sigma_{\epsilon_1}[\mathbf{x}] & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Sigma_{\epsilon_2}[\mathbf{x}] & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Sigma_{\epsilon_n}[\mathbf{x}] \end{bmatrix} \tag{4.6}$$

The relative pose estimate that minimise the sum of square registration error Mahalanobis distances is given by

$$\begin{aligned} \hat{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\epsilon^{\mathsf{T}} (\Sigma_{\epsilon}[\mathbf{x}])^{-1} \epsilon \right) \\ &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \epsilon_i^{\mathsf{T}} (\Sigma_{\epsilon_i}[\mathbf{x}])^{-1} \epsilon_i \right) \end{aligned} \tag{4.7}$$

Since no closed-form solution is available, Equations 4.7 must be solved using an iterative optimisation algorithm starting from an initial parameter estimate (such as the Euclidean solution).

Unfortunately, the objective function in Equation 4.7 is not a standard least squares problem like those presented in Appendix C, since the weighting matrix is a function of the estimated parameters. In [85, 87], this problem was alleviated by approximating the registration error covariance $\Sigma_{\epsilon}[\mathbf{x}]$ as a constant in each iteration of an algorithm equivalent to the Gauss-Newton method presented in Appendix C.3.1. While this simplification results in an

objective function that can be solved by standard optimisation algorithms, it also results in an approximate calculation of the objective function gradient that is used to update the estimated parameters.

An alternative approach is to convert the weighted least squares problem of Equation 4.7 into an ordinary least-squares problem using the technique described in Section C.1.3. The conversion is performed by defining the standardised residual as

$$\epsilon^{\diamond} = \sqrt{(\Sigma_{\epsilon}^{-1})}\epsilon \tag{4.8}$$

Equation 4.7 then becomes an ordinary least-squares problem

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left((\epsilon^{\diamond})^{\mathsf{T}} \epsilon^{\diamond} \right) \tag{4.9}$$

This method is computationally inefficient because the weighting matrix $\sqrt{(\Sigma_{\epsilon}^{-1})}$ needs to be recalculated at each iteration since it is a function of the parameters being optimised. In simulations presented in Section 4.2.2, this method is implemented to understand the consequences of the constant weight matrix approximation.

An additional issue with the minimum Mahalanobis distance approach arises when attempting to calculate the covariance of the motion estimates produced by Equation 4.7. The Jacobian of the coordinate transformation function in Equation 4.1 with respect to the motion parameters is required, however this must be evaluated at some estimate of the feature positions. While the maximum likelihood feature parameters could be calculated for the estimated motion, this will not be performed since it is inconsistent with the goal of this approach which aims to eliminate the structure parameters from the estimation process. Since no better feature position estimates are available, the Jacobian is evaluated at the observed triangulated feature positions $\mathbf{z}_{ai}$

$$\boldsymbol{\nabla}_i = \left[-{}^b_a\mathbf{R} \quad \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\phi_b}\Big|_{{}^a\hat{\boldsymbol{\psi}}_b} (\mathbf{z}_{ai} - {}^a\hat{\mathbf{t}}_b) \quad \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\theta_b}\Big|_{{}^a\hat{\boldsymbol{\psi}}_b} (\mathbf{z}_{ai} - {}^a\hat{\mathbf{t}}_b) \quad \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\psi_b}\Big|_{{}^a\hat{\boldsymbol{\psi}}_b} (\mathbf{z}_{ai} - {}^a\hat{\mathbf{t}}_b) \right] \tag{4.10}$$

The complete Jacobian for the estimates of all n features relative to pose b is then $\boldsymbol{\nabla} = \left[\boldsymbol{\nabla}_1^{\mathsf{T}}, \boldsymbol{\nabla}_2^{\mathsf{T}}, \ldots \boldsymbol{\nabla}_n^{\mathsf{T}} \right]^{\mathsf{T}}$, and the covariance of the estimated parameters is

$$\begin{aligned} \Sigma_{\mathbf{x}} &= (\boldsymbol{\nabla}^{\mathsf{T}} \Sigma_{\epsilon}^{-1} \boldsymbol{\nabla})^{-1} \\ &= (\sum_{i=1}^{n} \boldsymbol{\nabla}_i^{\mathsf{T}} \Sigma_{\epsilon_i}^{-1} \boldsymbol{\nabla}_i^{\mathsf{T}})^{-1} \end{aligned} \tag{4.11}$$

In the simulations presented in Section 4.2.2, the covariance produced using this method will be compared to solutions where the maximum likelihood feature positions are estimated

from the observations acquired at both poses.

Minimisation of 2D Image Reprojection Errors in One Pose

In contrast to the two previous methods that used 3D registration error residuals, image based quantities will now be considered to measure the fit of a motion hypothesis to the observed data.

If the position of a feature is triangulated from observations in images in frame a, given a hypothesis of the motion parameters it is then possible to reproject the 3D feature location into the images at frame b to obtain the expected feature coordinates. The reprojected image coordinates can be compared with the actual observed image coordinates to measure the quality of the motion estimate. Investigation of this error model has been inspired by the used of robustified reprojection errors in the images at one pose to score motion hypotheses in a ground vehicle stereo odometry application [82, 83].

As in the 3D registration methods, the parameter vector for this estimation problem consists only of the relative pose states

$$\mathbf{x} = \begin{bmatrix} {}^{a}\mathbf{p}_{b} \end{bmatrix} \tag{4.12}$$

The observation vector for feature i has the form

$$\mathbf{z}_i = \begin{bmatrix} \mathbf{c}_{bl_i} \\ \mathbf{c}_{br_i} \end{bmatrix} \tag{4.13}$$

where $\mathbf{c}_{bl_i}$ and $\mathbf{c}_{br_i}$ are the observed image coordinates (in pixels) of feature i in the left and right images acquired at pose b.

The triangulated 3D position of a feature relative to frame a ($\mathbf{z}_{ai}$) is reprojected into the left and right image of the stereo rig at pose b, forming the predicted observation

$$\begin{aligned} \hat{\mathbf{z}}_i &= \mathbf{f}_i[\mathbf{x}] \\ &= \begin{bmatrix} \mathbf{proj}[{}^{a}\hat{\mathbf{p}}_{bl}, \mathbf{z}_{ai}] \\ \mathbf{proj}[{}^{a}\hat{\mathbf{p}}_{br}, \mathbf{z}_{ai}] \end{bmatrix} \end{aligned} \tag{4.14}$$

where $\mathbf{proj}[\cdot,\cdot]$ is the image projection function of Equation 2.8, and ${}^{a}\hat{\mathbf{p}}_{bl}$ and ${}^{a}\hat{\mathbf{p}}_{br}$ represent the estimated pose of the left and right cameras at pose b relative to pose a. These pose estimates can be calculated with

$${}^{a}\hat{\mathbf{p}}_{bl} = {}^{a}\hat{\mathbf{p}}_{b} \oplus {}^{s}\mathbf{p}_{l} \tag{4.15}$$

$${}^{a}\hat{\mathbf{p}}_{br} = {}^{a}\hat{\mathbf{p}}_{b} \oplus {}^{s}\mathbf{p}_{r} \tag{4.16}$$

in which ${}^{s}\mathbf{p}_l$ and ${}^{s}\mathbf{p}_r$ are the pose of the left and right cameras relative to the stereo-rig's reference frame (known from calibration), and $\oplus$ is the head to tail pose composition operation defined in Appendix A.

The Jacobian of the observation model in Equation 4.14 with respect to the estimated states is

$$\begin{aligned}\boldsymbol{\nabla}_i &= \frac{\partial \mathbf{f}_i[\mathbf{x}]}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}} \\ &= \begin{bmatrix} \frac{\partial \mathbf{proj}[{}^{a}\mathbf{p}_{bl}, {}^{a}\mathbf{t}_i]}{\partial {}^{a}\mathbf{p}_b}\Big|_{({}^{a}\mathbf{p}_{bl}={}^{a}\hat{\mathbf{p}}_{bl}, {}^{a}\mathbf{t}_i=\mathbf{z}_{ai})} \\ \frac{\partial \mathbf{proj}[{}^{a}\mathbf{p}_{bl}, {}^{a}\mathbf{t}_i]}{\partial {}^{a}\mathbf{p}_b}\Big|_{({}^{a}\mathbf{p}_{bl}={}^{a}\hat{\mathbf{p}}_{bl}, {}^{a}\mathbf{t}_i=\mathbf{z}_{ai})} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial \mathbf{proj}[{}^{a}\mathbf{p}_{bl}, {}^{a}\mathbf{t}_i]}{\partial {}^{a}\mathbf{p}_{bl}}\Big|_{({}^{a}\mathbf{p}_{bl}={}^{a}\hat{\mathbf{p}}_{bl}, {}^{a}\mathbf{t}_i=\mathbf{z}_{ai})} \mathbf{J}_{\oplus 1({}^{a}\hat{\mathbf{p}}_b, {}^{s}\mathbf{p}_l)} \\ \frac{\partial \mathbf{proj}[{}^{a}\mathbf{p}_{bl}, {}^{a}\mathbf{t}_i]}{\partial {}^{a}\mathbf{p}_{bl}}\Big|_{({}^{a}\mathbf{p}_{bl}={}^{a}\hat{\mathbf{p}}_{bl}, {}^{a}\mathbf{t}_i=\mathbf{z}_{ai})} \mathbf{J}_{\oplus 1({}^{a}\hat{\mathbf{p}}_b, {}^{s}\mathbf{p}_r)} \end{bmatrix}\end{aligned} \tag{4.17}$$

The residual for feature i is the reprojection error, defined as the difference between the observed and reprojected image coordinates

$$\boldsymbol{\epsilon}_i = \mathbf{z}_i - \hat{\mathbf{z}}_i \tag{4.18}$$

The optimal relative stereo pose parameters for this error model are given by

$$\begin{aligned}\hat{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\boldsymbol{\epsilon}^{\mathsf{T}} \boldsymbol{\epsilon} \right) \\ &= \underset{\mathbf{x}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \boldsymbol{\epsilon}_i^{\mathsf{T}} \boldsymbol{\epsilon}_i \right)\end{aligned} \tag{4.19}$$

There is no closed form solution for Equation 4.19, therefore it must be solved using an iterative optimisation algorithm.

The uncertainty of the feature positions triangulated in frame a are not considered by this method. It can be seen from Equation 4.19 that image errors in all directions are treated equally (there is no non-uniform weight matrix), however it is known that the uncertainty of triangulated features is not equal in all directions (uncertainty in depth is typically greater). Errors in the triangulated feature positions are propagated to image coordinate errors, so it could be expected that this method will produce sub-optimal results. The effects of ignoring triangulated feature uncertainties will be investigated when this method is applied to simulated data in Section 4.2.2.

Bundle Adjustment

Bundle adjustment [50, 65, 106] is commonly used to estimate camera motion, scene structure and camera calibration model parameters from a sequence of images. Bundle adjustment has typically been used in monocular vision structure from motion problems, where features are observed over potentially long sequences of images. There has recently been interest in applying bundle adjustment to stereo odometry in [103], and [24] in which a robustified version of the bundle adjustment cost-function is used.

A possible reason for the lack of interest in bundle adjustment in robotics may be concern for the speed of large optimisation problems in applications that must run in real-time. This should not be a issue for the target AUV application where low frame rates mean features are likely to only be visible in consecutive image pairs.

In this application, the use of a calibrated stereo rig is assumed. Bundle adjustment will be used to calculate the maximum likelihood motion and feature position parameters. If n features are observed, the parameter vector is

$$\mathbf{x} = \begin{bmatrix} {}^{a}\mathbf{p}_{b} \\ {}^{a}\mathbf{t}_{1} \\ \vdots \\ {}^{a}\mathbf{t}_{n} \end{bmatrix} \tag{4.20}$$

Image reprojection errors will be used to evaluate the fit of a set of model parameters to the observed feature image coordinates. Since the feature positions are estimated, the reprojection errors in the images acquired at both poses a and b are a function of the estimated parameters. This is in contrast to the previously presented approach that minimised image reprojection errors in one stereo image pair only.

The observation vector for n features is $\mathbf{z} = \left[\mathbf{z}_1^\mathsf{T}, \mathbf{z}_2^\mathsf{T}, \ldots, \mathbf{z}_n^\mathsf{T}\right]^\mathsf{T}$, where each $\mathbf{z}_i$ contains the image coordinates of feature i in the left and right images acquired at pose a ($\mathbf{c}_{al_i}$ and $\mathbf{c}_{ar_i}$) and pose b ($\mathbf{c}_{bl_i}$ and $\mathbf{c}_{br_i}$)

$$\mathbf{z}_i = \begin{bmatrix} \mathbf{c}_{al_i} \\ \mathbf{c}_{ar_i} \\ \mathbf{c}_{bl_i} \\ \mathbf{c}_{br_i} \end{bmatrix} \tag{4.21}$$

The observation model has the form

$$\begin{aligned}\hat{\mathbf{z}} &= \mathbf{f}[\mathbf{x}] \\ &= \begin{bmatrix}\hat{\mathbf{z}}_1 \\ \hat{\mathbf{z}}_2 \\ \vdots \\ \hat{\mathbf{z}}_n\end{bmatrix}\end{aligned} \tag{4.22}$$

in which the predicted observation for each feature is

$$\begin{aligned}\hat{\mathbf{z}}_i &= \mathbf{f}_i[\mathbf{x}] \\ &= \begin{bmatrix}\mathbf{proj}[{}^s\mathbf{p}_l, {}^a\hat{\mathbf{t}}_i] \\ \mathbf{proj}[{}^s\mathbf{p}_r, {}^a\hat{\mathbf{t}}_i] \\ \mathbf{proj}[{}^a\hat{\mathbf{p}}_{bl}, {}^a\hat{\mathbf{t}}_i] \\ \mathbf{proj}[{}^a\hat{\mathbf{p}}_{br}, {}^a\hat{\mathbf{t}}_i]\end{bmatrix}\end{aligned} \tag{4.23}$$

The residual vector is defined as

$$\boldsymbol{\epsilon} = \mathbf{z} - \hat{\mathbf{z}} \tag{4.24}$$

and the maximum likelihood parameter estimate is defined by

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left(\boldsymbol{\epsilon}^T \boldsymbol{\Sigma}_{\mathbf{z}}^{-1} \boldsymbol{\epsilon}\right) \tag{4.25}$$

Under the assumption of independent observations, the observation covariance matrix has the form

$$\boldsymbol{\Sigma}_{\mathbf{z}} = \begin{bmatrix}\boldsymbol{\Sigma}_{\mathbf{z}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\mathbf{z}_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_{\mathbf{z}_n}\end{bmatrix} \tag{4.26}$$

in which $\boldsymbol{\Sigma}_{\mathbf{z}_1}$ is the covariance of the image coordinates for feature i.

The estimate covariance is given by

$$\boldsymbol{\Sigma}_{\mathbf{x}} = (\boldsymbol{\nabla}^T \boldsymbol{\Sigma}_{\mathbf{z}}^{-1} \boldsymbol{\nabla})^{-1} \tag{4.27}$$

where $\boldsymbol{\nabla}$ is the Jacobian of the observation model with respect to the parameter states,

which has the sparse form

$$\nabla = \frac{\partial \mathbf{f}[\mathbf{x}]}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_2 & \mathbf{0} & \mathbf{B}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{A}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_n \end{bmatrix} \tag{4.28}$$

where each $\mathbf{A}_i$ represents the Jacobian of the predicted observation of feature i with respect to the relative pose states

$$\mathbf{A}_i = \frac{\partial \mathbf{f}_i[\mathbf{x}]}{\partial {}^a\mathbf{p}_b}\Big|_{\hat{\mathbf{x}}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \frac{\partial \mathbf{proj}[{}^a\mathbf{p}_{bl}, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{p}_{bl}}\Big|_{({}^a\hat{\mathbf{p}}_{bl}, {}^a\hat{\mathbf{t}}_i)} \mathbf{J}_{\oplus 1({}^a\hat{\mathbf{p}}_b, {}^s\mathbf{p}_l)} \\ \frac{\partial \mathbf{proj}[{}^a\mathbf{p}_{br}, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{p}_{br}}\Big|_{({}^a\hat{\mathbf{p}}_{br}, {}^a\hat{\mathbf{t}}_i)} \mathbf{J}_{\oplus 1({}^a\hat{\mathbf{p}}_b, {}^s\mathbf{p}_r)} \end{bmatrix} \tag{4.29}$$

and each $\mathbf{B}_i$ represents the Jacobian of the predicted observations of feature i with respect to the states of feature i

$$\mathbf{B}_i = \frac{\partial \mathbf{f}_i[\mathbf{x}]}{\partial {}^a\mathbf{t}_i}\Big|_{\hat{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \mathbf{proj}[{}^s\mathbf{p}_l, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{t}_i}\Big|_{({}^a\hat{\mathbf{t}}_i)} \\ \frac{\partial \mathbf{proj}[{}^s\mathbf{p}_r, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{t}_i}\Big|_{({}^a\hat{\mathbf{t}}_i)} \\ \frac{\partial \mathbf{proj}[{}^a\mathbf{p}_{bl}, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{t}_i}\Big|_{({}^a\hat{\mathbf{p}}_{bl}, {}^a\hat{\mathbf{t}}_i)} \\ \frac{\partial \mathbf{proj}[{}^a\mathbf{p}_{br}, {}^a\mathbf{t}_i]}{\partial {}^a\mathbf{t}_i}\Big|_{({}^a\hat{\mathbf{p}}_{br}, {}^a\hat{\mathbf{t}}_i)} \end{bmatrix} \tag{4.30}$$

The bundle adjustment optimal parameter estimate and covariance are typically performed using a variant of the Levenberg-Marquardt algorithm which takes advantage of the sparse observation Jacobian [50, 65]. A brief description of the sparse Levenberg-Marquardt algorithm is provided in Appendix C.

A New Proposal: Efficient Maximum Likelihood 3D Registration

Two issues with the minimum Mahalanobis 3D registration error were previously identified. First, it required optimisation of an objective function that was not a standard least squares problem. Second, the maximum likelihood parameters of the feature states were not calculated, forcing the system to be linearised at a suboptimal estimate when calculating the covariance of the solution. In this section, a motion estimation method that combines the best aspects of the 3D registration and bundle adjustment approaches will be presented. The proposed approach calculates the maximum likelihood motion and structure parameters using the 3D registration error model.

The triangulated 3D feature positions provide an accurate summary of the raw image coordinate observations, therefore this method is expected to generate similar results to bundle adjustment. However, since the triangulated 3D feature positions are a more compact and convenient representation of the observations than the image coordinate data, this approach can be expected to have a computational efficiency advantage over bundle adjustment.

As in the bundle adjustment approach, the parameter vector will contain the motion parameters and the position states for n features

$$\mathbf{x} = \begin{bmatrix} {}^{a}\mathbf{p}_b \\ {}^{a}\mathbf{t}_1 \\ \vdots \\ {}^{a}\mathbf{t}_n \end{bmatrix} \tag{4.31}$$

The observation vector consists of the feature positions relative to frame a and b calculated through triangulation $\mathbf{z} = \left[\mathbf{z}_1^\mathsf{T}, \mathbf{z}_2^\mathsf{T}, \ldots, \mathbf{z}_n^\mathsf{T},\right]^\mathsf{T}$ where

$$\mathbf{z}_i = \begin{bmatrix} \mathbf{z}_{ai} \\ \mathbf{z}_{bi} \end{bmatrix} \tag{4.32}$$

The observation model has the form

$$\begin{aligned} \hat{\mathbf{z}} &= \mathbf{f}[\mathbf{x}] \\ &= \begin{bmatrix} \hat{\mathbf{z}}_1 \\ \hat{\mathbf{z}}_2 \\ \vdots \\ \hat{\mathbf{z}}_n \end{bmatrix} \end{aligned} \tag{4.33}$$

where the predicted observation for each feature consists of its position relative to frame a and b

$$\begin{aligned}\hat{\mathbf{z}}_i &= \mathbf{f}_i[\mathbf{x}] \\ &= \begin{bmatrix} {}^a\hat{\mathbf{t}}_i \\ {}^b_a\mathbf{R}({}^a\hat{\mathbf{t}}_i - {}^a\hat{\mathbf{t}}_b) \end{bmatrix}\end{aligned} \tag{4.34}$$

The observation model is relatively simple when compared to that of bundle adjustment in Equation 4.23. Summarising the observations of a feature using the triangulated 3D position results in a smaller observation vector of six states for two 3D feature positions, instead of the eight states for four sets of image coordinates used in bundle adjustment. Additionally, the observation model consists of a single transformation to the frame of the second stereo-rig pose, compared to bundle adjustment in which the 3D feature position estimate must be transformed to each of the four camera frames, and then projected to image coordinates.

The maximum likelihood estimate and covariance are given by

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\, \boldsymbol{\epsilon}^\mathsf{T} \boldsymbol{\Sigma}_\mathbf{z}^{-1} \boldsymbol{\epsilon} \tag{4.35}$$

$$\boldsymbol{\Sigma}_\mathbf{x} = (\boldsymbol{\nabla}^\mathsf{T} \boldsymbol{\Sigma}_\mathbf{z}^{-1} \boldsymbol{\nabla})^{-1} \tag{4.36}$$

The Jacobian of the observation model with respect to the parameters has the same sparse form as the bundle adjustment approach, allowing Equations 4.35 and 4.36 to be solved using the sparse Levenberg-Marquardt algorithm described in Appendix C.

The Jacobian is

$$\begin{aligned}\boldsymbol{\nabla} &= \left.\frac{\partial \mathbf{f}[\mathbf{x}]}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}} \\ &= \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_2 & \mathbf{0} & \mathbf{B}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{A}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_n \end{bmatrix}\end{aligned} \tag{4.37}$$

where the Jacobian of the predicted observations of feature i with respect to the relative

pose states is

$$\mathbf{A}_i = \frac{\partial \mathbf{f}_i[\mathbf{x}]}{\partial {}^a\mathbf{p}_b}\Big|_{\hat{\mathbf{x}}} \tag{4.38}$$
$$= \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -{}^b_a\mathbf{R} & \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\phi_b}\Big|_{{}_b\hat{\psi}_a}({}^a\hat{\mathbf{t}}_i - {}^a\hat{\mathbf{t}}_b) & \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\theta_b}\Big|_{{}_b\hat{\psi}_a}({}^a\hat{\mathbf{t}}_i - {}^a\hat{\mathbf{t}}_b) & \frac{\partial {}^b_a\mathbf{R}}{\partial {}^a\psi_b}\Big|_{{}_b\hat{\psi}_a}({}^a\hat{\mathbf{t}}_i - {}^a\hat{\mathbf{t}}_b) \end{bmatrix}$$

and the Jacobian with respect to the feature states is

$$\mathbf{B}_i = \frac{\partial \mathbf{f}_i[\mathbf{x}]}{\partial {}^a\mathbf{t}_i}\Big|_{\hat{\mathbf{x}}} \tag{4.39}$$
$$= \begin{bmatrix} \mathbf{I} \\ {}^b_a\mathbf{R} \end{bmatrix}$$

Note the difference in Equation 4.38 to the Jacobian of the minimum Mahalanobis distance algorithm in Equation 4.10. In Equation 4.10, the system was linearised around the observed feature position $\mathbf{z}_{ai}$, while in Equation 4.38 the system is linearised at the maximum likelihood feature position ${}^a\hat{\mathbf{t}}_i$, which is produced by fusing observations from both poses.

In Section 4.2.2 the accuracy and efficiency of the maximum likelihood 3D registration approach will be compared to bundle adjustment to understand the effect of using the triangulated feature position observation representation instead of the raw image coordinates. This method will also be compared to the minimum Mahalanobis distance algorithm to understand the consequences of eliminating the structure parameters from the estimated state vector.

4.2.2 Simulation

To compare the proposed relative pose estimation methods, each has been applied to simulated data approximating odometry and loop-closure conditions.

Modelling

The simulations have been modelled on the stereo-rig onboard the Seabed AUV (see Appendix B), and the typical operating conditions of the vehicle. The stereo cameras are oriented to look downwards to the seafloor, have a horizontal field of view of 40°, a vertical field of view of 30°, and are separated by a $70mm$ baseline. Simulated image coordinate of feature observations are corrupted by Gaussian noise with a standard deviation of 0.4 pixels.

In all simulations, the stereo-rig is initially located 2m above the seafloor, which is a typical operating altitude for the SeaBED AUV. The seafloor is simulated as a plane, which is the least informative surface and a reasonable model for the flat sand or rock terrains often encountered underwater. To simulate odometry conditions, 50 features are observed as the stereo-rig moves forward 0.5m (a translation of -0.5m in the stereo-frame y-axis as shown in Figure 4.2). To simulate loop-closure conditions, 6 features are observed and the stereo-rig is translated 0.3 metres in each of the x, y and z axes, undergoes roll and pitch 10° and a yaw of 90°.

Implementation Details

The Euclidean distance method has been implemented using the procedure described in Algorithm 4.1.

Three approaches to minimising the minimum Mahalanobis registration error cost function have been implemented to gain an understanding of the effects of the approximate linearisation: The Gauss-Newton algorithm using the constant approximation to the weight matrix during each iteration as described in Section 4.2.1 (equivalent to the iterative linearisation algorithm used in [85, 87]), the Levenberg-Marquardt algorithm using the same approximation, and the Levenberg-Marquardt algorithm using the weighting procedure described in Section 4.2.1 that leads to a correct linearisation.

Minimising the image reprojection error at one pose has been implemented using the Levenberg-Marquardt algorithm described in Section C.3.2. The bundle adjustment and maximum likelihood 3D registration methods have been performed using an implementation of the sparse Levenberg-Marquardt algorithm [65].

All iterative optimisations have been performed using the minimum Euclidean distance solution as an initial motion parameter hypothesis. The triangulated observed feature positions relative to the first pose of the stereo-rig have been used as initial structure parameter estimates for the bundle adjustment and maximum likelihood 3D registration algorithms.

The minimum gradient magnitude and minimum step size relative magnitude termination parameter values used in all optimisations were $\epsilon_1 = 1e^{-5}$ and $\epsilon_2 = 1e^{-5}$ respectively, and the initial damping factor $\mu_0 = 1e^{-3}$ was used in all applications of the Levenberg-Marquardt algorithm (see Appendix C for an explanation of these parameters).

Evaluation

The odometry and loop-closure simulations have each been performed 50 times. Plots showing the errors for each motion estimator are presented in Appendix E. The results of the odometry simulations are summarised in Tables 4.1 and 4.2, which list the mean and root mean square errors for each motion estimation method. The results of the loop-closure simulations are summarised in Tables 4.3 and 4.4.

Since bundle adjustment calculates the maximum likelihood motion parameters from the raw observations, it will be considered the benchmark to which the alternative approaches are compared. As expected, no significant biases are present in the bundle adjustment errors in Tables 4.1 and 4.3, and the root mean square errors are equal or better than the other methods in Tables 4.2 and 4.4. Additionally, in the plots presented in Appendix E, the bundle adjustment estimator errors are well bounded by the 95% confidence intervals.

The Euclidean relative pose estimator displays significant biases in Tables 4.1 and 4.3, and large errors are listed in Tables 4.2 and 4.4. This is consistent with the poor accuracy expected from this approach, since it ignores the uncertainties in the triangulated feature positions. However, due to the existence of a closed form solution, the Euclidean estimator remains useful as an initialisation point for iterative optimisation algorithms.

To produce the summaries for the minimum Mahalanobis 3D registration error method in Tables 4.1, 4.2, 4.3 and 4.4, the weighting procedure resulting in correct linearisation has been used. Figure 4.3 compares the convergence of all three algorithms considered to optimise the Mahalanobis registration error objective function. The Gauss-Newton algorithm using the approximate linearisation fails to converge from the initial estimate in Figure 4.3(a), while partial convergence is also demonstrated by both the Gauss-Newton and Levenberg-Marquardt algorithms in all simulations. The approximate linearisation also results in slower convergence towards the optimal solution. The correctly linearised method displays complete convergence, however the maximum likelihood 3D registration algorithm is a more efficient method that obtains the same solution.

Calculating the motion covariance is also a problem with the Mahalanobis distance method. In the odometry simulation plots of Appendix E.1, the covariances obtained by evaluating the observation model Jacobians at the observed feature positions do not diverge from the maximum likelihood solution, since the translation-only camera motion causes the observation model to be linear. However, when the camera motion includes a rotation, as in the loop-closure simulations plots shown in Appendix E.2, the covariances diverge from the maximum likelihood solutions due to the linearisation of the observation model at a poor estimate.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean Error	-0.5844	-0.0543	0.0372	-0.0181	0.1597	-0.0109
Mahalanobis Error	0.0223	-0.0397	0.0090	-0.0162	-0.0058	0.0023
One Pose Reprojection Error	-0.0083	9.5059	-1.0904	2.6983	0.0016	0.0023
Bundle Adjustment	0.0224	-0.0555	0.0090	-0.0161	-0.0058	0.0022
ML 3D Registration	0.0223	-0.0396	0.0090	-0.0162	-0.0058	0.0023

Table 4.1: Mean errors in relative pose estimates from 50 odometry simulations.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean Error	1.7813	4.6611	0.6905	1.3282	0.4909	0.0596
Mahalanobis Error	0.1899	0.3808	0.0459	0.1059	0.0525	0.0195
One Pose Reprojection Error	0.8476	9.9944	1.1429	2.8374	0.2376	0.0687
Bundle Adjustment	0.1897	0.3836	0.0459	0.1060	0.0525	0.0196
ML 3D Registration	0.1899	0.3808	0.0459	0.1059	0.0525	0.0195

Table 4.2: Root mean square errors in relative pose estimates from 50 odometry simulations.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean Error	1.8161	1.4390	0.6998	-0.6056	-0.5143	-0.0004
Mahalanobis Error	0.5179	-0.5800	-0.0017	-0.1797	0.1851	-0.0302
One Pose Reprojection Error	-6.0672	-14.2023	2.0970	2.1482	4.7556	0.6300
Bundle Adjustment	0.4997	-0.5721	0.0105	-0.1711	0.1854	-0.0286
ML 3D Registration	0.5180	-0.5798	-0.0017	-0.1797	0.1851	-0.0302

Table 4.3: Mean errors in relative pose estimates from 50 loop-closure simulations.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean Error	10.4160	15.2057	2.0339	3.5267	5.1324	0.5655
Mahalanobis Error	3.2691	4.3823	0.5137	1.1189	1.4558	0.2492
One Pose Reprojection Error	9.4160	16.2907	2.6552	3.3206	5.4583	0.8584
Bundle Adjustment	3.2646	4.3673	0.5153	1.1173	1.4508	0.2496
ML 3D Registration	3.2690	4.3823	0.5136	1.1189	1.4558	0.2492

Table 4.4: Root mean square errors in relative pose estimates from 50 loop-closure simulations.

The relative pose estimates produced by minimising image reprojection errors at only one pose show significant biases in Tables 4.1 and 4.3. This is a result of the uncertainty in feature observations relative to frame a not being considered in the cost function. As demonstrated in Figure 4.4, this estimator favours motion hypotheses that orient the cameras to look in the direction of largest uncertainty in the feature positions. Since the uncertainty in the depth of a triangulated feature position is much larger than in other directions, when applied to the odometry simulations, the estimator incorrectly explains the observations using a pitching motion.

Figure 4.5 shows the average errors in the x-axis position and pitch angle estimates produced by minimising image reprojection errors in only one pose, as the pitch of the stereo-rig is varied between 0° for downward looking cameras, and 90° for forward looking cameras. The simulations have been repeated for varying numbers of features and different variances in the feature z-coordinates in the world frame (deviation from a planar ground surface). As expected, the estimator biases are smaller with forward looking cameras whose principal axis is aligned with the direction of motion. While a robustified version of the one-pose reprojection error cost function has been used successfully to score motion hypothesis in a ground vehicle application with forward looking cameras [82, 83], the bias produced when simulating downward looking cameras operating above a planar seafloor suggests this approach may be unsuitable for the target AUV application.

In Tables 4.1, 4.2, 4.3 and 4.4, the maximum likelihood 3D registration method produced near identical results to bundle adjustment. Additionally, in the simulation plots presented in Appendix E, the errors are well bounded by the 95% confidence intervals. Figure 4.6 displays the processing time required to calculate the bundle adjustment and maximum likelihood 3D registration estimates for both odometry and loop-closure simulations. Processing times were collected on a 2.0 GHz Pentium M processor. As expected, the more compact and convenient observation representation of the maximum likelihood 3D registration method results in a significant gain in computational efficiency.

To test the assumption that the Euclidean solution is a valid initialisation point for the iterative optimisation algorithms, the simulations have been repeated using the true motion parameters as the initial estimate. In Figure 4.7, the iterative optimisation algorithms converge to the same minimiser when initialised with the Euclidean solution or true parameters, suggesting the Euclidean solution is indeed a suitable initial parameter estimate.

Due to its ability to produce estimates similar to bundle adjustment with greater computational efficiency, the maximum likelihood 3D registration method initialised at the Euclidean solution will be used in relative pose estimation problems in this thesis.

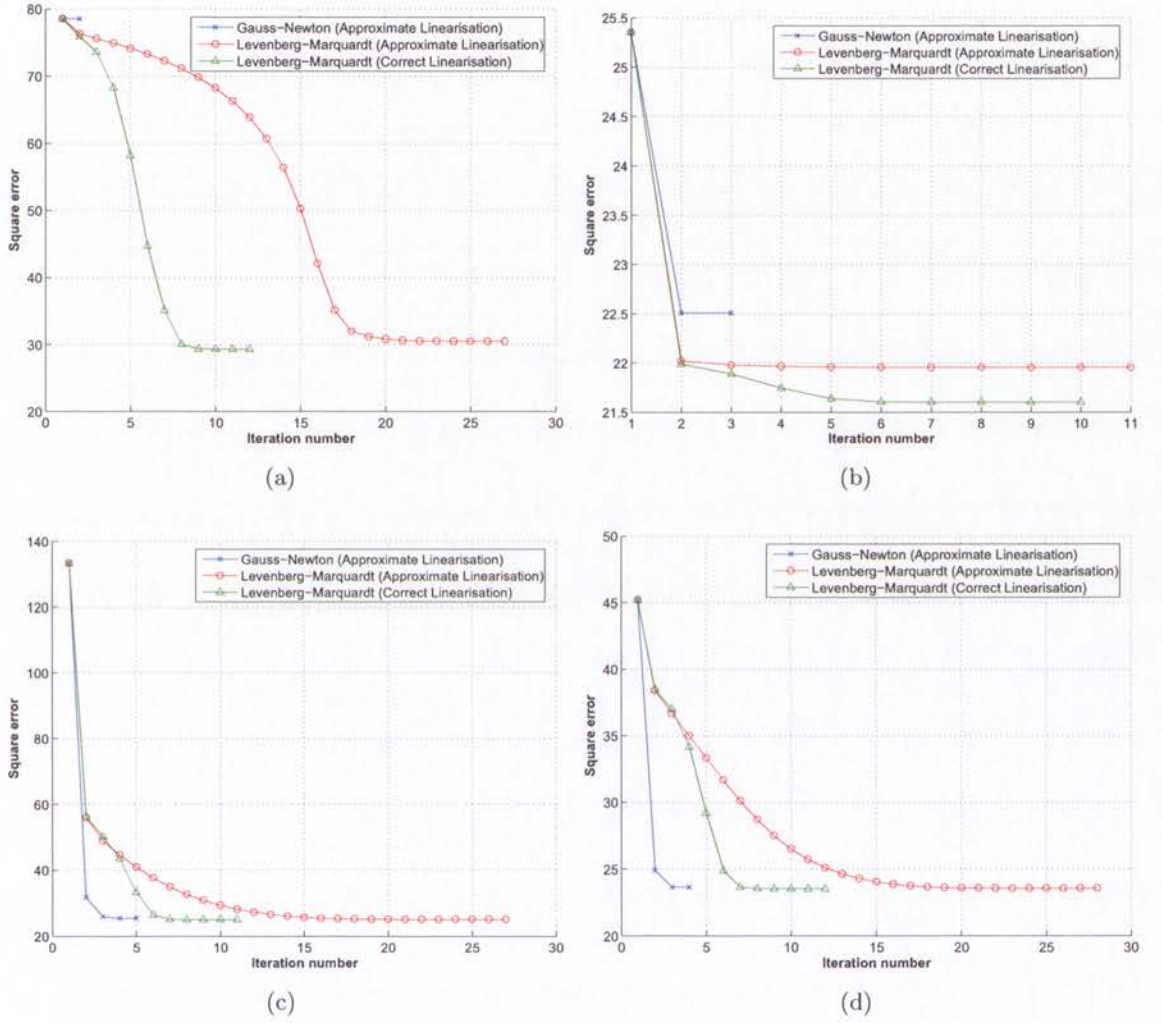


Figure 4.3: Convergence of Mahalanobis distance minimisation algorithms in four simulations. The Gauss-Newton algorithm using the approximate linearisation fails to converge from the initial estimate in (a), while partial convergence is also demonstrated by both the Gauss-Newton and Levenberg-Marquardt algorithms in all simulations. The approximate linearisation also results in slower convergence towards the optimal solution. The Levenberg-Marquardt algorithm using the weighting technique to produce a correct linearisation converges to the correct solution, however it is computationally inefficient.

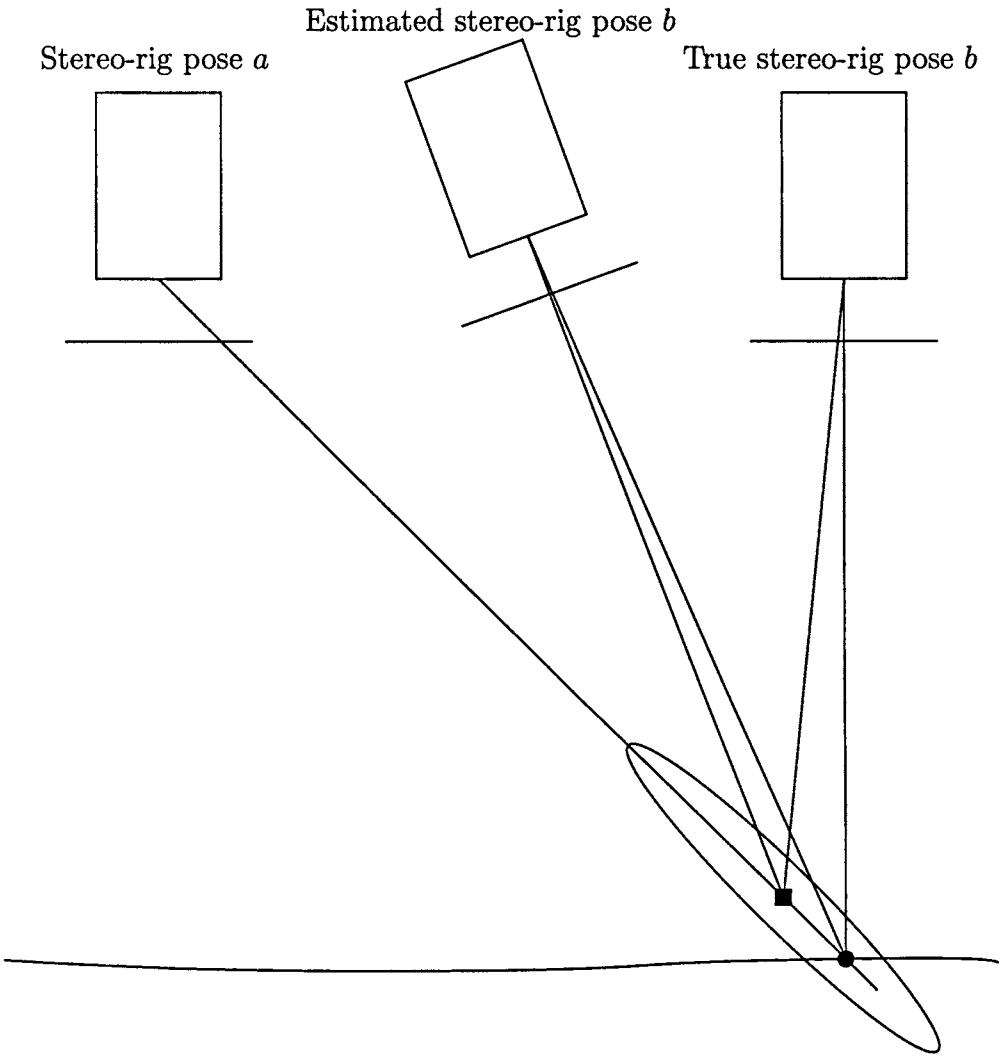
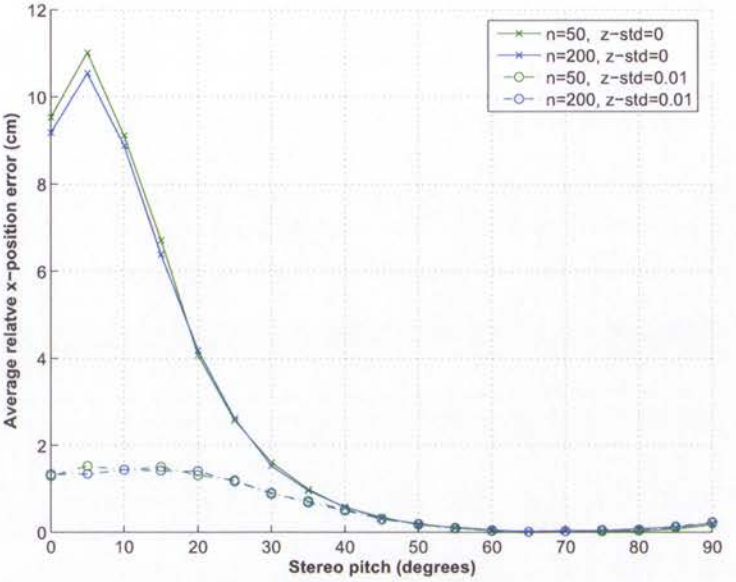
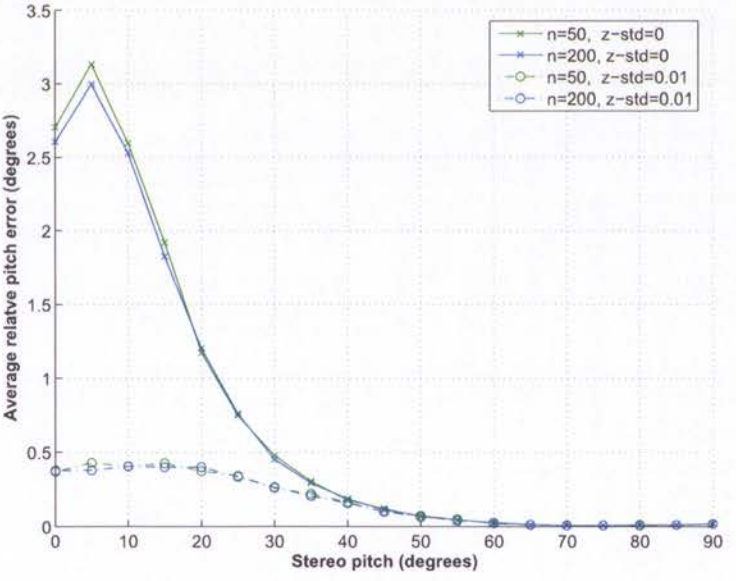


Figure 4.4: Bias caused by minimising image reprojection errors at only one pose. The location of a feature on the terrain is marked by a circle. The estimated feature position calculated by triangulating the observations at pose a is marked with a square, and the uncertainty is shown with an ellipse. Minimising image reprojection errors only in the images acquired at frame b results in an estimator favouring a rotation that causes the cameras to look down the direction of largest feature uncertainty. The bias is caused by the estimator ignoring the uncertainty in the triangulated feature position, which is not equal in all directions.

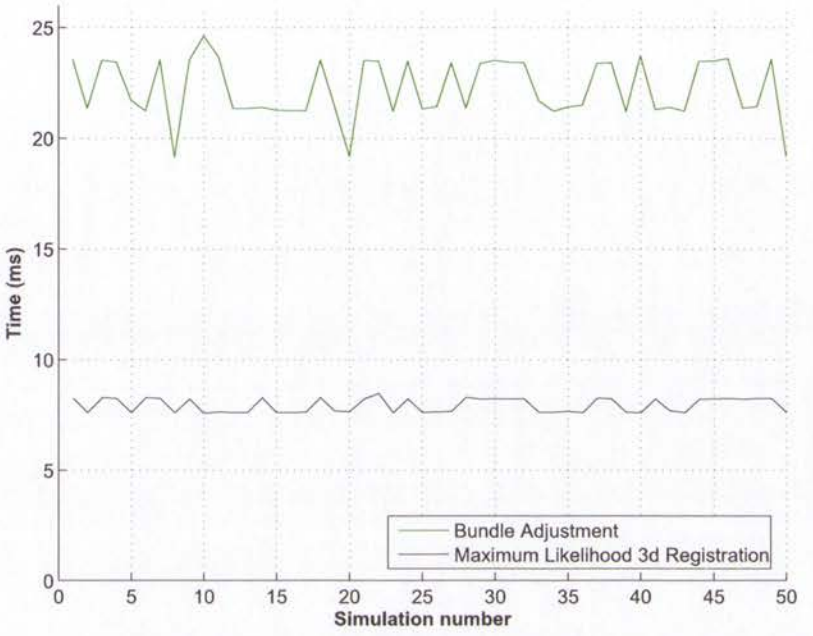


(a) Average x-position estimate error

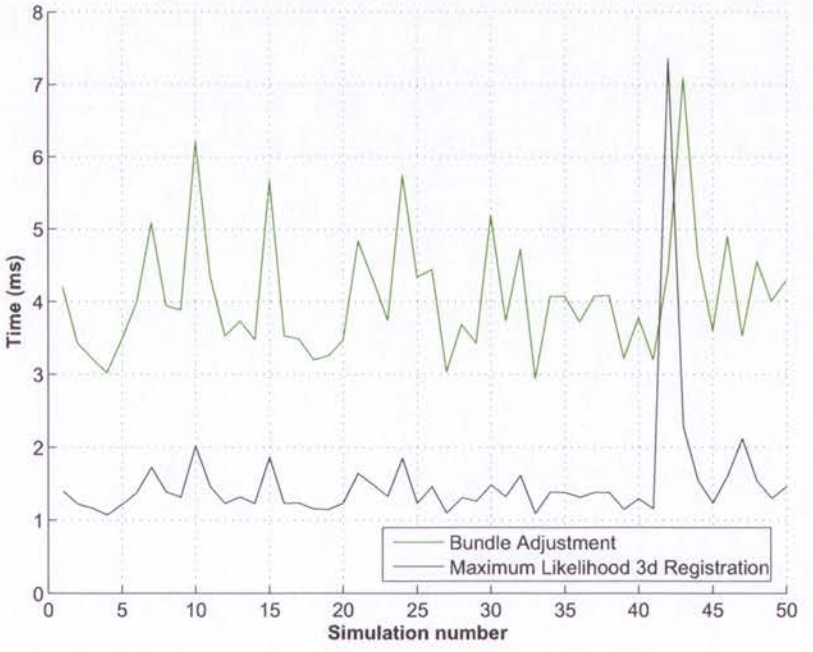


(b) Average Pitch Euler angle estimate error

Figure 4.5: Bias in the estimated relative x-position and pitch estimates produced by minimising image reprojection errors at only one pose. A pitch of zero degrees corresponds to the stereo-rig looking downwards towards the seafloor, while a pitch of 90 degrees results in the cameras looking forwards in the direction of motion. 50 simulations were performed, and the average error in the relative pose estimate states are plotted as a function of the pitch of the stereo-rig relative to the vehicle. The simulations have been performed with varied values for the number of features (n) and the standard-deviation of the z-axis positions of the features in the world frame (z-std).

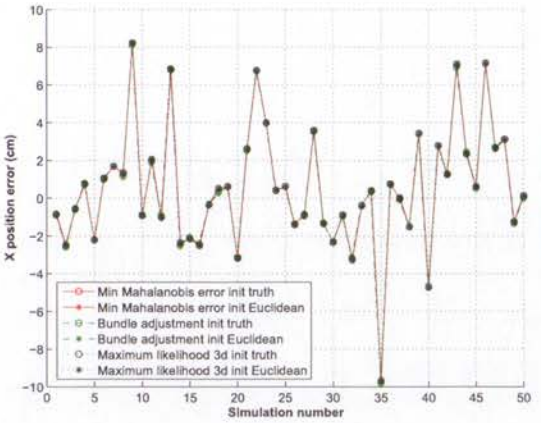


(a) Odometry simulation processing times

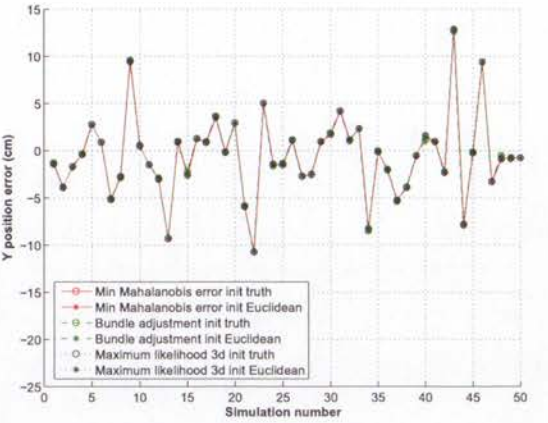


(b) Loop-closure simulation processing times

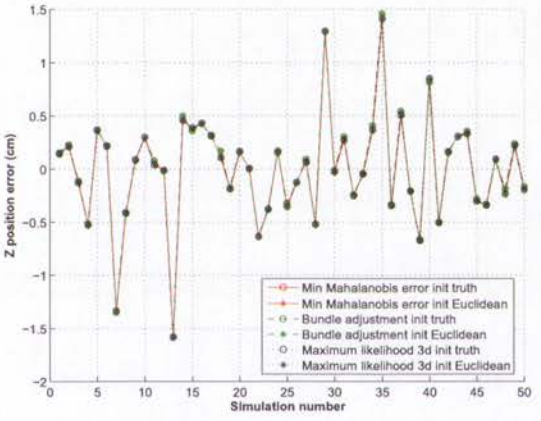
Figure 4.6: Comparison of the computational efficiency of the bundle adjustment and maximum likelihood 3D registration estimators for odometry and loop-closure simulations. The processing times were collected on a 2.0 GHz Pentium M processor. 50 features were used in the odometry simulations, while 6 features were used in the loop-closure simulations.



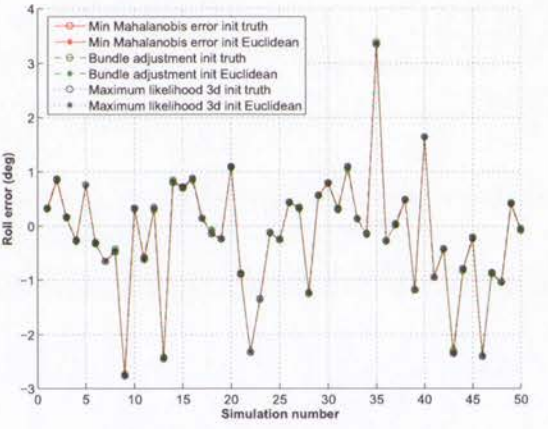
(a) X-position error



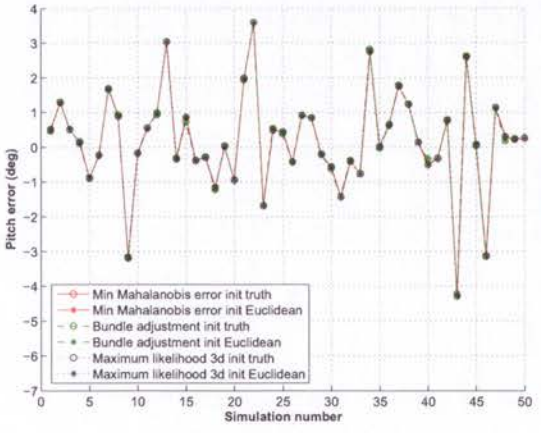
(b) Y-position error



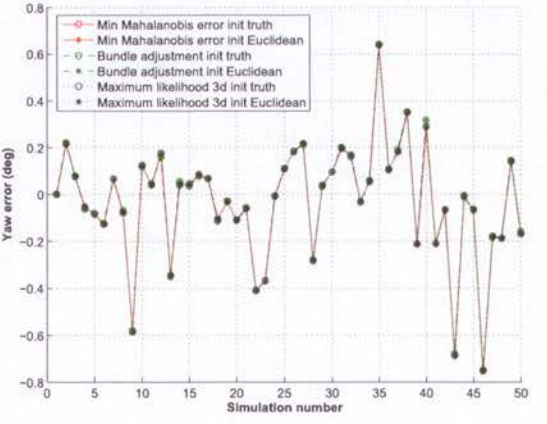
(c) Z-position error



(d) Roll Euler angle error



(e) Pitch Euler angle error



(f) Yaw Euler angle error

Figure 4.7: Errors in relative pose estimates from iterative optimisation algorithms. The optimisation algorithms have been initialised at the true motion parameters and the minimum Euclidean error solution. The algorithms converge to the same solution, suggesting the Euclidean solution is a valid initial parameter hypothesis.

4.3 Outlier Rejection

In Section 4.2, several motion estimation algorithms were evaluated when applied to simulated data in which the feature observation errors were distributed according to a zero-mean Gaussian as expected by the observation model. Unfortunately, a significant portion of feature observations are often outliers that are inconsistent with the assumption of normally distributed errors, and the least-squares estimators previously investigated are known to be highly sensitive to such data. In this section, outlier rejection approaches will be investigated to enable relative pose estimates to be produced from data corrupted with the types of errors typically found in real image data. The maximum likelihood 3D registration relative pose estimator previously selected for its performance on outlier-free data will provide the preferred method for generating motion hypotheses within the outlier rejection algorithms.

Typical causes of feature observation outliers include location errors, association errors and moving objects. Location errors result from failures in the extraction or tracking algorithms that calculate the image coordinates of a feature. Causes of location errors include variation in the visual appearance of a feature at different viewpoints and inconsistent lighting. Detection of location outliers can be difficult since the magnitude of their errors is typically small. Association outliers are caused by incorrect matching of features observed in different image frames. The image coordinate errors produced by association failures are typically large, and can therefore cause wildly incorrect motion hypotheses if they are not removed from the estimation process. The outliers produced by moving objects are similar to association errors, since the observed feature coordinates in each stereo image pair are consistent, however their apparent movement between image pairs is not compatible with the motion of the cameras. An additional concern with moving objects is their ability to produce multiple features that are inconsistent with the true camera motion, but consistent with each other. This is likely to result in a scenario in which one outlier masks the presence of others, increasing the difficulty of their detection.

The simplest form of outlier rejection when using a calibrated stereo rig is the application of epipolar geometry constraints on the observations of a feature within a stereo image pair. While epipolar geometry constraints can reject many location outliers, feature observations containing errors parallel to epipolar lines cannot be detected. Additionally, outliers caused by association errors and moving objects may contain feature coordinates that are consistent with epipolar geometry in each image pair. While epipolar constraints should be applied to features observations extracted from each stereo image pair, there is a need for an additional outlier rejection method that considers observations from multiple image pairs.

The outlier rejection approaches investigated in this section require two components: a method to generate motion hypotheses, and a test to determine if the observations of a

feature are consistent with a motion hypothesis.

In Section 4.3.1, options for outlier classification tests will be presented. In section 4.3.2, two frequently used outlier rejection frameworks will be briefly summarised, and the development of an approach to create motion hypotheses using robust estimation is presented. Combinations of motion hypothesis generators and outlier classification tests will be evaluated on simulated data in Section 4.3.3.

4.3.1 Outlier Classification Tests

Outlier tests based on image reprojection errors [13, 60] and 3D point registration residuals [85, 87] have been proposed in stereo-vision motion estimation applications. Since the maximum likelihood 3D registration motion estimator was selected in Section 4.2, outlier classification methods based on 3D registration residuals will be considered here.

3D Registration Error Euclidean Distance Test

Recalling Equation 4.3, the 3D registration error for feature i is

$$\epsilon_i = \mathbf{z}_{bi} - {}^{b}_{a}\mathbf{R}(\mathbf{z}_{ai} - {}^{a}\hat{\mathbf{t}}_{b})$$

The Euclidean inlier acceptance test for feature i is

$$\text{Feature } i \text{ is an } \begin{cases} \text{inlier}, & \text{if } \epsilon_i^{\mathsf{T}}\epsilon_i \leq k_e \\ \text{outlier}, & \text{if } \epsilon_i^{\mathsf{T}}\epsilon_i > k_e \end{cases} \tag{4.40}$$

where the scalar k_e is the square Euclidean error threshold.

The selection of a threshold for the Euclidean test could be expected to be difficult, since registration errors are not distributed equally in all directions, and the triangulated positions of different features will have varying levels of uncertainty. The performance of a given threshold will vary with different applications and environments. A conservative threshold may be required to prevent the rejection of a large number of valid features, which is likely to result in the acceptance of misclassified outliers that will corrupt the final relative pose estimate.

3D Registration Error Mahalanobis Distance Test

In Section 4.2 it was shown that consideration of feature observation uncertainties resulted in an improved motion estimator. It could be expected that the consideration of the regis-

tration error uncertainties will similarly yield an outlier classifier with superior performance to the Euclidean error test.

The registration error covariance for feature i was defined in Equation 4.5 to be

$$\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}_i} = \boldsymbol{\Sigma}_{bi} + {}^{b}_{a}\mathbf{R}\,\boldsymbol{\Sigma}_{ai}\,{}^{b}_{a}\mathbf{R}^{\mathsf{T}}$$

The Mahalanobis inlier acceptance test is then

$$\text{Feature } i \text{ is an } \begin{cases} \text{inlier}, & \text{if } \boldsymbol{\epsilon}_i^{\mathsf{T}}\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}_i}^{-1}\boldsymbol{\epsilon}_i \leq k_m \\ \text{outlier}, & \text{if } \boldsymbol{\epsilon}_i^{\mathsf{T}}\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}_i}^{-1}\boldsymbol{\epsilon}_i > k_m \end{cases} \tag{4.41}$$

where k_m is a scalar threshold .

While the main motivation for the Mahalanobis distance test is an expected improvement in classification performance, an additional benefit is theoretical justification for the selection of a threshold. Since the square registration error Mahalanobis distances are distributed according to a chi-square distribution with three degrees of freedom, the expected inlier acceptance rate for a given threshold can be obtained from statistical tables. For example, 95% of inliers should be accepted with a threshold of 7.815 (accurate to three decimal places). In contrast to a threshold for the Euclidean error test, the expected performance for a Mahalanobis test threshold is constant for all environments and applications.

4.3.2 Relative Pose Hypothesis Generation Frameworks

To classify features as inliers or outliers using the Euclidean or Mahalanobis test, a hypothesis for the relative pose parameters must first be generated. The two most commonly used hypothesis generation frameworks in stereo odometry applications are iterative rejection and RANSAC. Recently an alternative method using a modified RANSAC algorithm that minimises a robust cost function based on image reprojection errors has been used [82, 83]. This approach could be considered an M-estimator evaluated using multiple initial parameter hypotheses.

After briefly describing the iterative rejection and RANSAC approaches, the main focus of this section will be the development of a robust relative pose estimator by gaining an understanding of the effects of the M-estimator parameters on the final solution. The performance of the robust estimation hypothesis generation method will then be compared to the iterative rejection and RANSAC frameworks under simulation in Section 4.3.3.

Iterative Estimation and Rejection

A simple outlier rejection approach is produced by iteratively estimating the motion parameters using all features considered inliers (initially all features), and rejecting those features that are evaluated to be outliers due to inconsistency with the motion estimate. This two step process is repeated, generating a sequence of parameter hypotheses converging to an estimate produced by a consistent set of features classified as inliers.

Iterative outlier rejection approaches have been applied to ground vehicle stereo odometry applications in [85, 87] in which motion estimates are generated using the minimum Mahalanobis registration error method, and [104] in which motion estimates are produced using bundle adjustment. Different strategies have been used in the selection and number of outliers rejected in each iteration. In [85, 87] the worst matching features with a residual greater than a threshold are eliminated, and in [104] the mean and standard deviation of the residuals are calculated, and all features with residuals more than 1.5 standard deviations from the mean are classified as outliers.

The iterative rejection framework will be tested by rejecting the feature with the largest squared error (Euclidean or Mahalanobis depending on the rejection test being used) if it is larger than the outlier rejection threshold. If rejecting the feature evaluated as the worst outlier is assumed to improve the motion estimate, the decision to discard one feature at a time results in the most robust possible implementation of the iterative rejection approach at the cost of the computational efficiency that could be achieved by rejecting multiple features in each iteration. The algorithm terminates if all remaining features are classified as inliers, or if less than three feature remain, in which case a motion hypothesis cannot be calculated and all remaining features are classified as outliers.

The robustness of the iterative rejection algorithm could be expected to be limited, since least-squares estimators are known to be highly sensitive to outliers. Feature observations with large errors are likely to corrupt the motion hypotheses such that the classification of inliers and outliers is unreliable.

RANSAC

The RANdom SAmple Consensus (RANSAC) algorithm [30, 50] aims to find a parameter estimate uncorrupted by outliers through the evaluation of multiple hypotheses generated from minimal subsets of features. Each minimal set contains the smallest number of features required to generate a motion hypothesis, and the support for each hypothesis is measured by counting the number of features with observations compatible with the motion parameters. The hypothesis evaluated to have the highest level of support is selected, and data

points inconsistent with the chosen parameter estimate are rejected as outliers. Examples of the RANSAC approach applied to stereo odometry include [1, 13, 53, 60] in which motion hypotheses are generated using the Euclidean method.

The number of hypotheses generated and evaluated in a RANSAC algorithm is calculated to provide a specified level of certainty that at least one of the minimal sets consists only of inliers. If the fraction of outliers in a set of data is α, and the number of data samples required to generate a hypothesis is n, the probability that at least one of m sample sets contains inliers only is given by

$$p = 1 - \left(1 - (1-\alpha)^n\right)^m \tag{4.42}$$

The number of iterations required to achieve a desired confidence level p that at least one sample set contains inliers only is then

$$m = \frac{\ln(1-p)}{\ln\left(1-(1-\alpha)^n\right)} \tag{4.43}$$

In most applications, the fraction of outliers cannot be known in advance. The adaptive RANSAC algorithm [50] maintains an estimate of the outlier ratio by initially assuming a conservatively large value, which is then updated after each hypothesis is evaluated.

While RANSAC is designed to produce a minimal set containing no outliers, even inlier observations contain noise that will be propagated to a motion estimate. Only the sparse discreet points in the parameter space generated from minimal sets can be investigated by RANSAC, and no minimal set is likely to produce the true motion parameters, no matter how many iterations are performed. For this reason, good outlier classification with the RANSAC approach may only be expected when a clear separation between inliers and outliers exists, allowing the use of a conservative test threshold that compensates for errors in the motion hypotheses.

The Mahalanobis test of Equation 4.41 will produce the expected inlier acceptance rate only when the tested motion hypothesis approaches the true motion parameters, therefore the RANSAC method can be expected to have a lower inlier acceptance rate than predicted from the chi-square distribution statistics. Reliable generation of a motion hypothesis approaching the true parameters requires some filtering or optimisation over a larger portion of the data than a minimal set.

Robust Estimation

The sensitivity of least squares estimators to erroneous data has motivated the development of robust statistical methods that provide superior resistance to outliers. An M-estimator [55] (maximum likelihood type estimator) will be considered for generating relative pose estimates from feature observations containing outliers.

The maximum-likelihood 3D registration motion estimator described in Section 4.2.1 minimised the sum of square normalised residuals

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{n} \boldsymbol{\epsilon}_i^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{z}_i}^{-1} \boldsymbol{\epsilon}_i \tag{4.44}$$

Defining the normalised residual for feature i as $r_i = \sqrt{\boldsymbol{\epsilon}_i^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{z}_i}^{-1} \boldsymbol{\epsilon}_i}$, an M-estimator minimises the sum of a function of the residuals

$$\hat{\mathbf{x}}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho(r_i) \tag{4.45}$$

where ρ is an increasing function with a slower growth-rate than least-squares, preventing large outliers from dominating the objective function. Candidate ρ-functions can be divided into two categories: monotone and redescending.

A monotone ρ-function is convex in r. Examples include Huber's proposal [55]

$$\rho(r) = \begin{cases} \frac{r^2}{2} & \text{if} \quad |r| \leq c \\ c\left(|r| - \frac{c}{2}\right) & \text{if} \quad |r| > c \end{cases} \tag{4.46}$$

the 'Fair' function [89]

$$\rho(r) = c^2 \left(\frac{|r|}{c} - \log\left(1 + \frac{|r|}{c}\right) \right) \tag{4.47}$$

and the least powers function [89]

$$\rho(r) = \frac{|r|^\nu}{\nu} \tag{4.48}$$

A redescending ρ-function has a derivative that approaches zero as the size of the residual approaches infinity. Redescending ρ-functions provide better outlier resistance than monotone ρ-functions, however they can introduce many local minima into the objective function. Locating the global minima of a redescending M-estimator cost function is therefore more difficult.

Examples of redescending ρ-functions are the Cauchy function [89] (named for its optimality

when applied to the Cauchy distribution)

$$\rho(r) = \frac{c^2}{2} \log\left(1 + \left(\frac{r}{c}\right)^2\right) \tag{4.49}$$

Tukey's biweight function [89]

$$\rho(r) = \begin{cases} \frac{c^2}{6}\left(1 - \left(1 - \left(\frac{r}{c}\right)^2\right)^3\right) & \text{if } |r| \leq c \\ \frac{c^2}{6} & \text{if } |r| > c \end{cases} \tag{4.50}$$

and Welsch's function [89]

$$\rho(r) = \frac{c^2}{2}\left(1 - \exp\left(-\left(\frac{r}{c}\right)^2\right)\right) \tag{4.51}$$

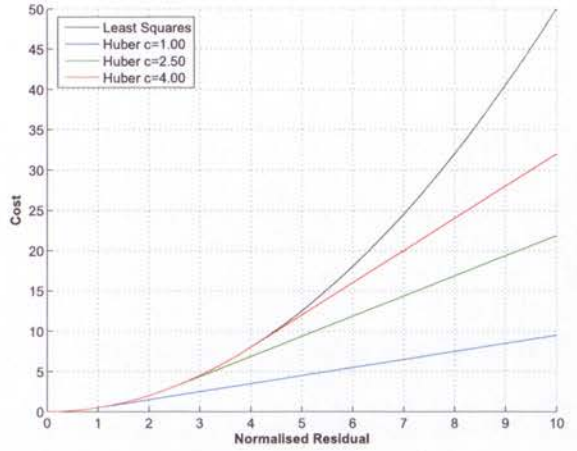
The tuning parameter c (or ν in the case of least powers) controls a trade-off between the statistical efficiency (variance) and robustness of the M-estimator. As $c \to \infty$, or $\nu \to 2$, the M-estimator approaches the maximum likelihood estimator (efficient but not robust), while smaller values of c increase resistance to outliers at the cost of the lower efficiency (higher variance) resulting from down-weighting or even ignoring data.

The influence of a datapoint on an estimate is proportional to the objective function gradient. Figures 4.8 and 4.9 show the monotone and redescending ρ-functions and their gradients (ψ-functions) for a selection of values of the tuning parameters c or ν. In the ψ-functions of Figures 4.8 and 4.9 all functions except least powers can be seen to bound the influence of an outlier (as a result least powers is considered 'quasi-robust' [89]). The ψ-functions in Figure 4.9 demonstrate that the influence of large outliers approaches zero for a redescending M-estimator.

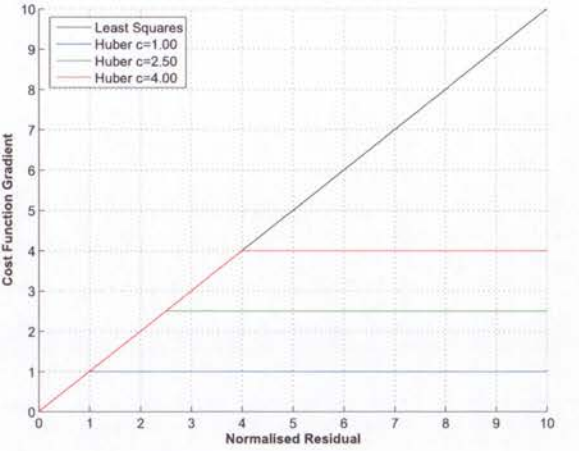
Robust estimation has been used in a stereo odometry application in [82, 83], where the Cauchy function of Equation 4.49 with the tuning parameter ignored (equivalent to $c = 1$) was used as a robustification kernel for image reprojection errors. This cost function was optimised using multiple initial hypotheses generated using a RANSAC-style algorithm [81].

In typical robust estimation applications (e.g. [82, 83]), outliers are not truly rejected, instead their influence is reduced through weighting. Unfortunately this results in an estimator with unknown variance, which is incompatible with the desire to produce a sequence of relative pose estimates suitable for data-fusion using an EKF.

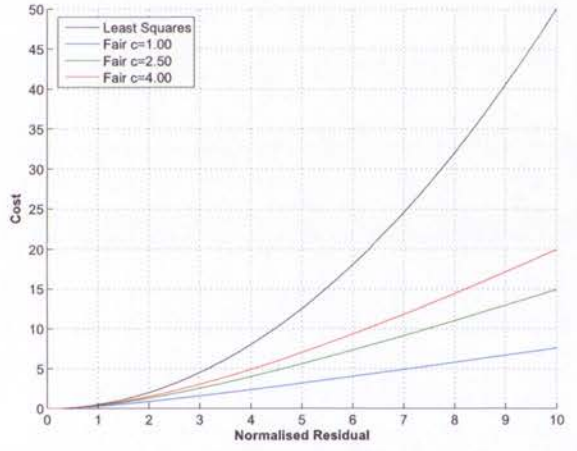
Here, instead of using an M-estimator to produce a final motion estimate, it will be considered for generating motion hypotheses from which outliers are classified. The maximum likelihood estimate and covariance will then be produced from the remaining inlier features.



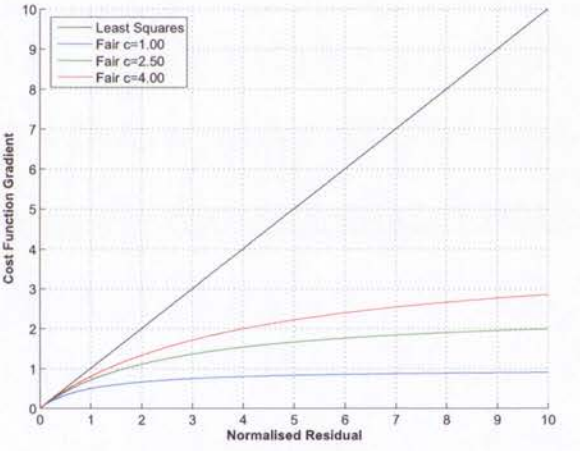
(a) Huber's ρ-function



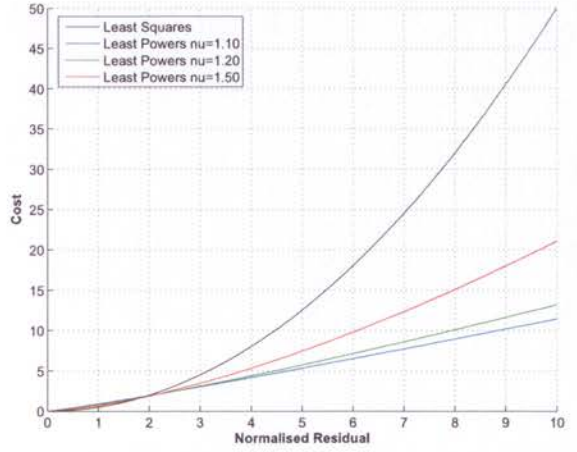
(b) Huber's ψ-function



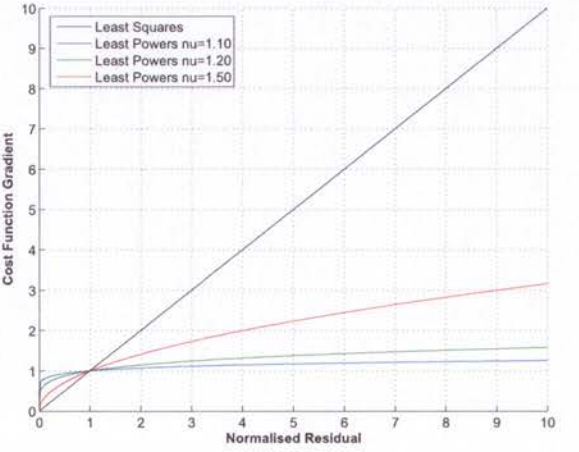
(c) The 'Fair' ρ-function



(d) The 'Fair' ψ-function

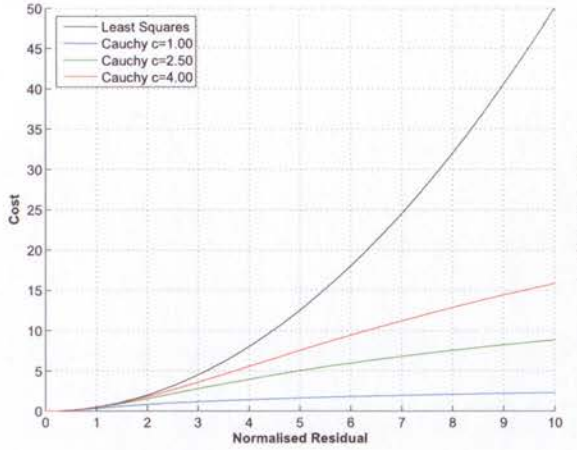


(e) Least power ρ-function

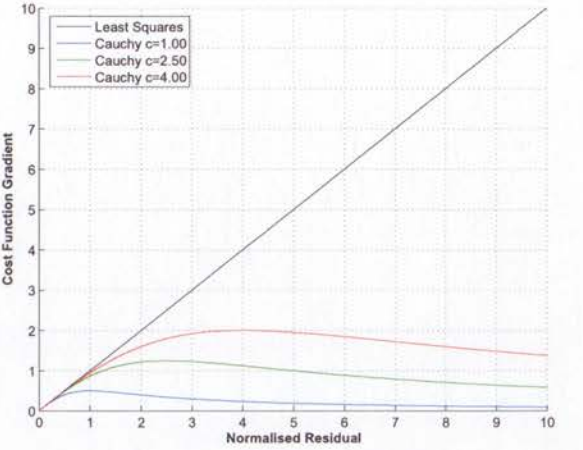


(f) Least power ψ-function

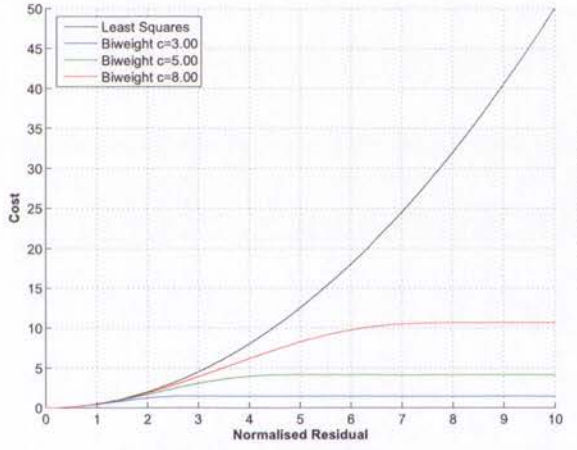
Figure 4.8: Monotone ρ-functions and their first derivatives (ψ-functions).



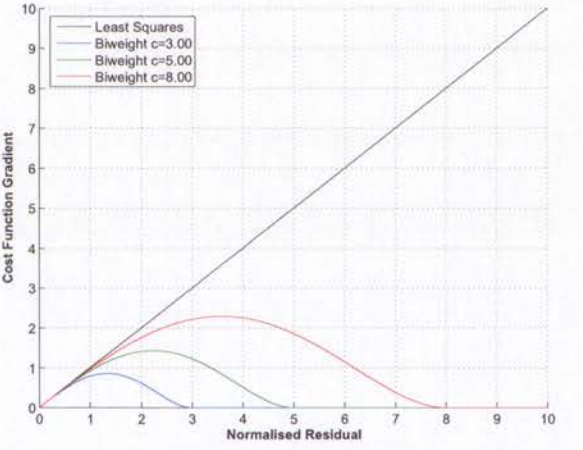
(a) The Cauchy ρ-function



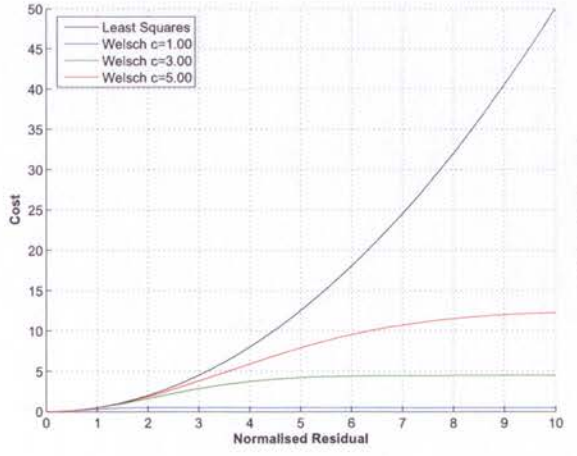
(b) The Cauchy ψ-function



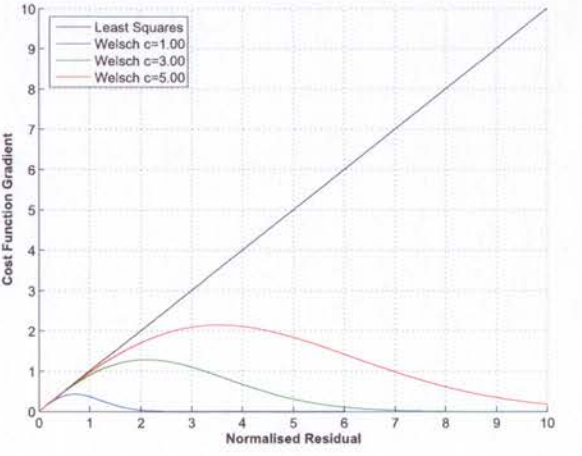
(c) Tukey's biweight ρ-function



(d) Tukey's biweight ψ-function



(e) Welsch's ρ-function



(f) Welsch's ψ-function.

Figure 4.9: Redescending ρ-functions and their first derivatives (ψ-functions).

Selection of an M-estimator requires the specification of a ρ-function, tuning parameter and initialisation method. The selection process will be performed by gaining an understanding of the effect of each decision on the performance of the robust M-estimator on simulated data approximating vision data complete with outliers. Since visual observation errors are naturally bounded by the size of the images, after appropriate simulation one can have confidence in the robustness of an estimator since the worst-case conditions have been tested.

Choosing a ρ-function is challenging since their performance depends on the distribution of the outliers. Rey [89] in general recommends the least powers function with values of $\nu \approx 1.1$, while Hoaglin [52] suggests monotone functions are suitable when operating in the neighbourhood of the normal distribution (recommending Huber's proposal), while redescending functions such as Tukey's Biweight are preferable when extreme observations are present. In Section 4.3.3, the six ρ-functions in Equations 4.46 to 4.51 will be tested on simulated data approximating the location and association outliers expected in real image data.

To select an initialisation method, the minimum Euclidean distance and maximum likelihood estimates will be considered as initial hypotheses for monotone M-estimators, while monotone M-estimates and the RANSAC random sampling procedure will be evaluated for redescending M-estimators.

4.3.3 Simulation

The candidate hypothesis generation frameworks and outlier rejection tests have been applied to odometry and loop-closure simulations similar to those presented in Section 4.2, with the addition of location and association feature observation outliers.

Outlier Modelling

Location outliers are simulated with errors in the observation coordinates of one image (the left image acquired at the second stereo-rig pose has been chosen arbitrarily). An x-axis image coordinate error is drawn uniformly from the range between 2 and 25 standard deviations of the image observation model noise (0.8 to 10 pixels). Location outliers with large y-axis coordinate errors are not generated since they could be removed using epipolar geometry constraints (the baseline of the stereo-rig is assumed to approximately parallel to the image x-axes). Association outliers are modelled by swapping the second pose observations with those of another feature visible in all four frames.

In odometry simulations, 5 location outliers and 5 association outliers are produced in addition to 50 inlier features. The feature observation errors for a single odometry simulation are shown in Figure 4.10. Similarly, loop-closure simulations consist of 2 location outliers and 2 association outliers in addition to 6 inlier features. Considering the feature association precision demonstrated in Chapter 2, these are conservatively large error ratios.

Selection of a Robust Estimator

To aid the selection of a ρ-function, tuning parameter and initialisation method for the robust M-estimator, 50 odometry and loop-closure simulations containing outliers have been performed.

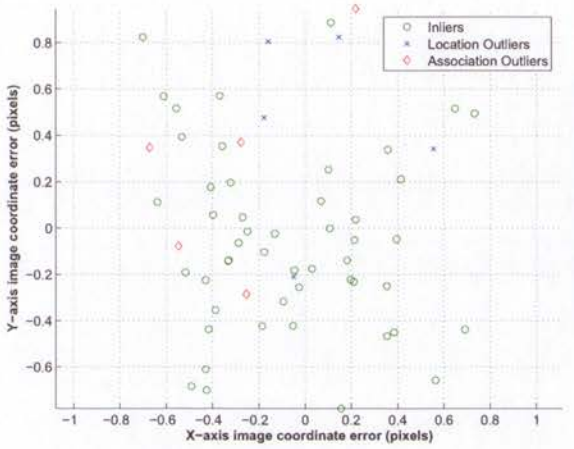
Table 4.5 presents the root mean square errors in the robust relative pose estimates produced by monotone M-estimators using Huber's, the 'Fair' and the least powers ρ-functions in odometry simulations. Similarly, Table 4.6 displays the errors produced by the monotone M-estimators when applied to loop-closure simulations. A selection of tuning parameter values have been tested for each ρ-function, and each M-estimator was initialised at the Euclidean and maximum likelihood motion estimates.

As expected, the M-estimators are seen to converge to the maximum likelihood estimator for larger tuning parameter values. The Euclidean solution proved to be the better initial estimate in the presence of outliers, with consistently lower errors than the maximum likelihood solution.

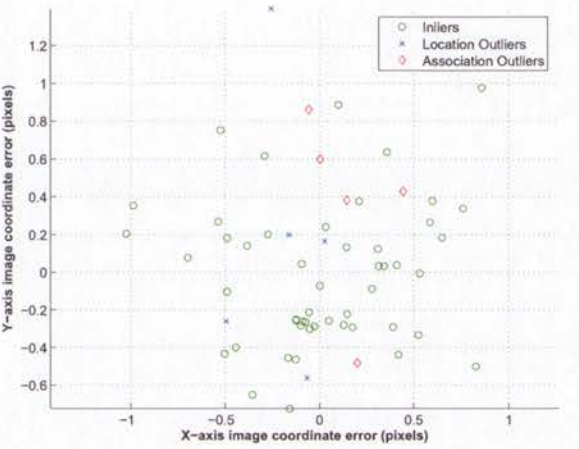
The results for all tested monotone ρ-functions and tuning parameter values are similar, with all estimates containing significant errors. The unsuitability of monotone ρ-functions for this application is not surprising, since the presence of association outliers causes the error distribution to deviate far from normality, and the monotone ρ-functions lack the resistance to large errors provided by redescending ρ-functions.

While monotone M-estimators will not be considered further as a means to produce a final motion hypothesis, one remaining possible use is the initialisation point for redescending M-estimators. Huber's proposal with the tuning parameter $c = 2.5$ will be considered for this role due to its consistent convergence, however since each of the tested ρ-functions produced similar results, little difference could be expected from any of the three tested functions.

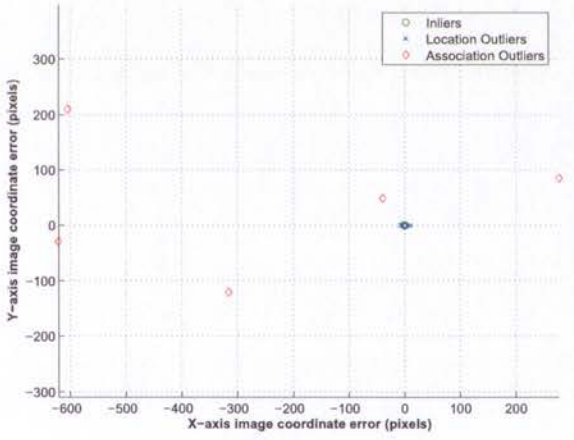
Table 4.7 presents the root mean square errors in the relative pose estimates produced by redescending M-estimators using the Cauchy, biweight and Welsch's redescending ρ-functions applied to odometry simulations. Table 4.8 shows the redescending M-estimator errors when applied to loop-closure simulations. The redescending M-estimators have been



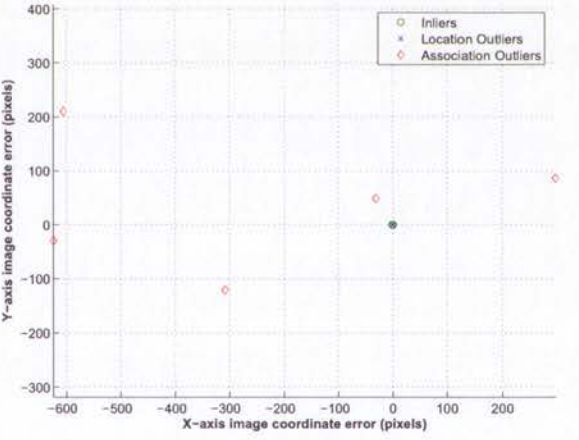
(a) First pose left image errors



(b) First pose right image errors



(c) Second pose left image errors



(d) Second pose right image errors

Figure 4.10: Feature observation errors in an odometry simulation containing outliers. 5 location outliers and 5 association outliers are simulated in addition to 50 inlier features. The observation errors in the images acquired at the first pose shown in (a) and (b). The observation errors in the images acquired at the second pose are shown in (c) and (d), where a different scale is used due to the presence of large errors in association outliers. The errors in (c) and (d) appear similar, since observations of a association outlier are consistent within the stereo image pair.

initialised at the monotone estimate and using the multiple random sampling approach of RANSAC.

None of the tested combinations of redescending ρ-function and tuning parameter displayed reliable convergence from the monotone estimate to the true motion parameters. This is an unfortunate result since performing only one optimisation from a single initialisation point has a significant computation efficiency advantage over the random sampling approach involving multiple initial hypotheses.

All redescending ρ-functions performed well when initialised with the multiple random sampling approach, however the Cauchy function stands out as the best choice due its superior ability to converge from a noisy initial estimate. This can best be observed by comparing the performance of the Cauchy ρ-function when initialised at the monotone estimates to the biweight and Welsch's function in Table 4.7. While the M-estimator using the Cauchy function converged from the monotone estimate to the similar solutions to those found using random sample initialisation, the M-estimators using Tukey's biweight and Welsch's functions were regularly unable to improve upon the initial monotone estimate. The reason for this difference can be seen in Figure 4.9: Tukey's biweight and Welsch's ρ-functions have a rejection point beyond which a datapoint has no influence on the resulting estimate. While this is good from an outlier resistance viewpoint, it means the objective function contains many constant regions containing no gradient for an optimisation algorithm to follow towards a minimum.

The Cauchy ρ-function with a tuning parameter value of $c = 2.5$ initialised using random-sampling will be used when comparing the robust estimation outlier rejection framework to iterative rejection and RANSAC.

Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean	7.7338	6.0666	2.2839	1.6657	2.0601	1.3385
ML 3D Registration	1.0386	41.8301	1.8736	10.9175	0.5238	1.7907

ρ-function	Tuning Parameter	Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Huber	c=1.00	Euclidean	0.4935	24.4913	1.5245	6.4751	0.1418	0.2917
Huber	c=1.00	ML 3D Registration	0.4936	24.4914	1.5245	6.4751	0.1419	0.2916
Huber	c=2.50	Euclidean	0.4974	24.8946	1.5330	6.5777	0.1426	0.2863
Huber	c=2.50	ML 3D Registration	0.6982	25.2607	1.9120	6.7192	0.2136	0.2891
Huber	c=4.00	Euclidean	0.5124	25.2847	1.5539	6.6798	0.1460	0.2890
Huber	c=4.00	ML 3D Registration	0.7192	25.4179	1.9284	6.7588	0.2189	0.2858
Fair	c=1.00	Euclidean	0.5263	26.1089	1.5604	6.8905	0.1516	0.2931
Fair	c=1.00	ML 3D Registration	0.5263	26.1089	1.5604	6.8905	0.1516	0.2931
Fair	c=2.50	Euclidean	0.5525	27.6914	1.5866	7.2959	0.1573	0.3112
Fair	c=2.50	ML 3D Registration	0.5525	27.6914	1.5866	7.2959	0.1573	0.3112
Fair	c=4.00	Euclidean	0.5760	29.1338	1.6103	7.6680	0.1629	0.3348
Fair	c=4.00	ML 3D Registration	0.5760	29.1339	1.6103	7.6680	0.1629	0.3348
Least Powers	ν=1.10	Euclidean	0.6351	29.8886	1.5709	7.8592	0.1841	0.3572
Least Powers	ν=1.10	ML 3D Registration	0.9519	30.1516	2.0391	7.9846	0.2892	0.3574
Least Powers	ν=1.20	Euclidean	0.6652	32.9994	1.6394	8.6310	0.1890	0.4326
Least Powers	ν=1.20	ML 3D Registration	1.1152	33.5282	2.1554	8.8378	0.3424	0.4297
Least Powers	ν=1.50	Euclidean	1.2937	38.3470	2.1031	10.0582	0.4432	0.5965
Least Powers	ν=1.50	ML 3D Registration	1.2936	38.3470	2.1031	10.0582	0.4432	0.5965

Table 4.5: Root mean square errors of monotone M-estimators and possible initial hypotheses for 50 odometry outlier simulations. While each of the M-estimators perform better than the maximum likelihood estimator in the presence of outliers, significant errors remain, particularly in the y-axis motion estimates.

Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Euclidean	32.0181	44.4383	21.2501	11.3976	15.6368	20.0739
ML 3D Registration	57.9700	53.6224	63.4011	45.6931	25.6721	29.9416

ρ-function	Tuning Parameter	Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Huber	c=1.00	Euclidean	48.3554	41.8853	29.0102	21.0378	15.9735	4.6219
Huber	c=1.00	ML 3D Registration	67.4345	58.5525	65.2652	41.2618	23.9046	20.3116
Huber	c=2.50	Euclidean	48.4353	41.9854	29.0514	21.0638	15.9751	4.5959
Huber	c=2.50	ML 3D Registration	66.7592	59.9587	64.7116	41.0015	24.5510	20.1533
Huber	c=4.00	Euclidean	48.7082	42.0450	29.2746	21.1993	15.9695	4.6062
Huber	c=4.00	ML 3D Registration	67.1243	60.0829	64.7819	41.0250	24.5996	20.0915
Fair	c=1.00	Euclidean	48.5804	43.5962	29.4425	21.1364	17.0490	4.6215
Fair	c=1.00	ML 3D Registration	67.6512	58.6285	65.5204	41.3965	23.9886	20.3530
Fair	c=2.50	Euclidean	54.1092	45.7986	46.1433	30.1437	17.7683	12.2312
Fair	c=2.50	ML 3D Registration	67.0281	58.8044	65.7692	41.1845	24.2695	20.0273
Fair	c=4.00	Euclidean	53.9096	45.8786	46.3120	30.2761	17.9089	12.1968
Fair	c=4.00	ML 3D Registration	66.9159	58.9064	66.0092	41.2635	24.5332	19.9304
Least Powers	ν=1.10	Euclidean	52.5082	46.0297	36.2921	24.6668	18.0794	7.2893
Least Powers	ν=1.10	ML 3D Registration	65.6074	58.7068	66.7862	41.4433	25.0080	20.8591
Least Powers	ν=1.20	Euclidean	57.8992	51.1945	57.2258	34.0759	22.5062	13.3890
Least Powers	ν=1.20	ML 3D Registration	64.3077	57.9668	66.9236	41.4145	25.7875	20.0519
Least Powers	ν=1.50	Euclidean	59.5553	51.8708	64.6739	42.2230	24.6413	23.5241
Least Powers	ν=1.50	ML 3D Registration	60.1892	54.8424	65.4390	42.8207	25.4058	24.0831

Table 4.6: Root mean square errors of monotone M-estimators and possible initial hypotheses for 50 loop-closure outlier simulations. Each of the monotone M-estimators fail in the presence of large outliers caused by feature association errors in the loop-closure simulations.

Initial Hypothesis			x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Monotone			0.4974	24.8946	1.5330	6.5777	0.1426	0.2863
ρ-function	Tuning Parameter	Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Cauchy	c=1.00	Monotone	0.0675	0.3308	0.0641	0.0927	0.0183	0.0240
Cauchy	c=1.00	Random Sampling	0.0678	0.3331	0.0647	0.0934	0.0184	0.0240
Cauchy	c=2.50	Monotone	0.0637	0.2907	0.0524	0.0802	0.0173	0.0218
Cauchy	c=2.50	Random Sampling	0.0637	0.2915	0.0525	0.0804	0.0173	0.0217
Cauchy	c=4.00	Monotone	0.0660	0.2928	0.0505	0.0806	0.0178	0.0216
Cauchy	c=4.00	Random Sampling	0.0659	0.2926	0.0506	0.0806	0.0178	0.0216
Biweight	c=3.00	Monotone	1.3822	24.9711	1.6269	6.5972	0.3517	0.2871
Biweight	c=3.00	Random Sampling	0.2584	1.3387	0.2223	0.3684	0.0714	0.1035
Biweight	c=5.00	Monotone	2.0268	23.8405	1.4502	6.2920	0.5102	0.2982
Biweight	c=5.00	Random Sampling	0.1310	0.4945	0.0979	0.1407	0.0348	0.0387
Biweight	c=8.00	Monotone	1.0055	18.7548	1.1559	4.9182	0.2616	0.2623
Biweight	c=8.00	Random Sampling	0.0791	0.3598	0.0733	0.0974	0.0218	0.0298
Welsch	c=1.00	Monotone	0.6587	24.9893	1.5693	6.6041	0.1760	0.2866
Welsch	c=1.00	Random Sampling	2.9312	4.9635	0.7584	1.4338	0.8515	0.3550
Welsch	c=3.00	Monotone	1.2256	21.1474	1.2729	5.5624	0.3160	0.2776
Welsch	c=3.00	Random Sampling	0.0820	0.4565	0.0732	0.1201	0.0215	0.0324
Welsch	c=5.00	Monotone	0.4738	11.2315	0.6012	2.9341	0.1320	0.1087
Welsch	c=5.00	Random Sampling	0.0725	0.3768	0.0611	0.1015	0.0197	0.0248

Table 4.7: Root mean square errors of redescending M-estimators and the initial motion hypothesis for 50 odometry outlier simulations. The initial Monotone M-estimator motion hypothesis was generated using Huber's ρ-function with the tuning parameter c=2.5. When initialised using multiple hypotheses generated from random samples of features, each of the redescending M-estimators achieve superior outlier rejection to the monotone M-estimators in Table 4.5. The M-estimators using the Biweight and Welsch ρ-functions fail to converge from the initial hypothesis provided by the monotone M-estimator, however the Cauchy ρ-function demonstrates superior convergence by finding similar solutions when using the monotone estimate and the random sampling initialisation method.

Initial Hypothesis			x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Monotone			48.4353	41.9854	29.0514	21.0638	15.9751	4.5959
ρ-function	Tuning Parameter	Initial Hypothesis	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
Cauchy	c=1.00	Monotone	30.1700	16.6024	14.7181	12.0480	6.9380	3.2734
Cauchy	c=1.00	Random Sampling	0.6443	0.7965	0.2504	0.2098	0.2557	0.1173
Cauchy	c=2.50	Monotone	32.0560	16.7639	22.5404	13.6518	6.6654	3.5344
Cauchy	c=2.50	Random Sampling	0.6199	0.9599	0.2437	0.2068	0.3148	0.1171
Cauchy	c=4.00	Monotone	36.1775	18.1197	27.7638	15.9393	7.8340	3.0751
Cauchy	c=4.00	Random Sampling	0.7683	1.2151	0.2866	0.2567	0.3984	0.1454
Biweight	c=3.00	Monotone	48.4390	41.9792	29.0588	21.0645	15.9747	4.5948
Biweight	c=3.00	Random Sampling	35.1858	18.4604	38.9401	16.8121	14.2102	26.5833
Biweight	c=5.00	Monotone	48.3924	41.9371	29.0436	21.0515	15.9554	4.5979
Biweight	c=5.00	Random Sampling	4.9267	5.2921	1.7450	1.6658	1.7898	1.7674
Biweight	c=8.00	Monotone	48.3269	41.7959	29.0122	21.0347	15.9315	4.5567
Biweight	c=8.00	Random Sampling	13.9642	14.6918	14.9168	8.7503	2.7332	5.5917
Welsch	c=1.00	Monotone	48.4370	41.9661	29.0531	21.0638	15.9713	4.5946
Welsch	c=1.00	Random Sampling	4.1923	4.2965	1.9942	1.3910	1.3786	2.0284
Welsch	c=3.00	Monotone	48.3307	41.9185	29.0804	21.0419	15.9641	4.5703
Welsch	c=3.00	Random Sampling	2.8841	2.6710	1.1821	0.9973	0.9230	0.5084
Welsch	c=5.00	Monotone	48.0058	41.6284	28.9632	20.9560	15.9025	4.5074
Welsch	c=5.00	Random Sampling	1.6473	2.0652	0.5093	0.5619	0.6982	0.2120

Table 4.8: Root mean square errors of redescending M-estimators and the initial motion hypothesis for 50 loop-closure outlier simulations. The initial Monotone M-estimator motion hypothesis was generated using Huber's ρ-function with the tuning parameter c=2.5. Each of the redescending M-estimators fail to converge from the initial hypothesis provided by the monotone M-estimator, however the random sampling initialisation method results in good outlier rejection, particularly when using the Cauchy ρ-function.

Comparison of Outlier Rejection Approaches

To compare the novel robust estimation outlier rejection approach with iterative rejection and RANSAC, each has been applied to odometry and loop-closure simulations containing outliers. The iterative rejection and RANSAC frameworks have been tested using the Euclidean and maximum likelihood 3D registration estimation methods to produce motion hypotheses, and the Euclidean and Mahalanobis outlier tests to perform classification.

Figure 4.11 shows the Receiver Operating Characteristic (ROC) curves for the outlier classifiers applied to 50 odometry and loop-closure simulations. The curves on the ROC plots are produced by repeating the simulations using a range of outlier rejection thresholds. As expected, the maximum likelihood motion hypothesis generation method and the Mahalanobis classification test that consider the uncertainties of feature observations generally outperform the Euclidean alternatives.

While the iterative rejection algorithm matched the outlier classification performance of the robust estimation framework in odometry simulations in Figure 4.11(a), its performance in loop-closure simulations in Figure 4.11(b) was poor. Large association outliers catastrophically corrupted the motion hypotheses, resulting in the rejection of many inlier features. In 17 out of the 50 loop-closure simulations, the iterative rejection algorithm was unable to converge to a minimal set of three inlier features, and was therefore unable to produce a final motion estimate. The classification performance of RANSAC in Figure 4.11 was steady in both odometry and loop-closure simulations. The novel robust estimation approach clearly provides the best outlier classifier, dominating the alternatives in the loop-closure simulations in Figure 4.11(b).

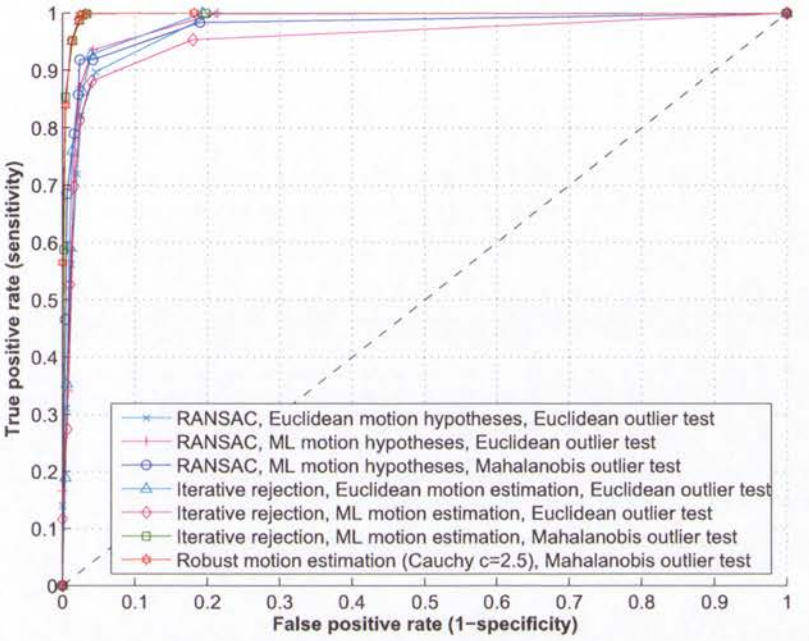
The inlier acceptance rate for each framework using the Mahalanobis test are compared to the expected rate in Figure 4.12. The inlier acceptances rates for iterative rejection during the loop-closure simulations of Figure 4.12(b) are low, since the algorithm was unable to reliably converge to a minimal set of inlier features. As predicted, the number of inliers accepted by RANSAC was lower than could be expected from better motion hypotheses. The superior motion hypotheses produced by the robust estimation method are seen to result in inlier acceptance rates close to the expected values. The inlier acceptance rate converges to the expected rate from above, which is understandable since the estimator is optimising a function related to the acceptance test.

Tables 4.9 and 4.10 present the mean and root mean square errors produced by calculating the maximum likelihood motion parameters on the features classified as inliers by each outlier rejection framework in the 50 odometry simulations. Tables 4.11 and 4.12 present the mean and root mean square errors in the 50 loop-closure simulations. Plots showing all estimator errors are presented in Appendix F. To produces these results, the iterative

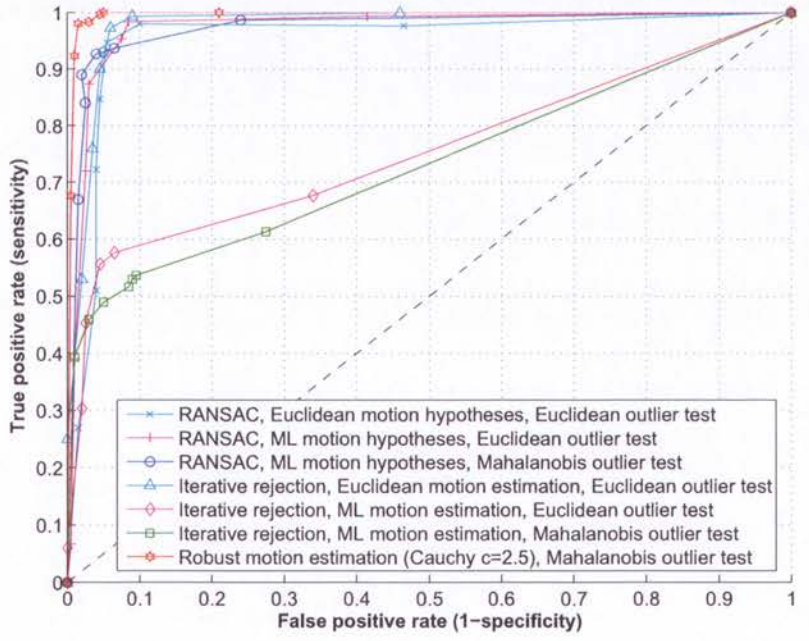
rejection and RANSAC frameworks used the maximum likelihood 3D registration method to produce motion hypotheses, and the Mahalanobis outlier rejection test with a threshold of 7.815, resulting in an expected inlier acceptance rate of 95%.

The reduced inlier acceptance rate of RANSAC results in the higher estimate variances observed in Tables 4.10 and 4.12 when compared to the robust method. Additionally since more outliers are accepted in comparison to the robust approach, a relatively large number of estimates from the RANSAC framework exceed the 95% confidence bounds in Appendix F.

The novel outlier rejection approach based on robust estimation developed in this section has been demonstrated to be superior to the commonly used iterative rejection and RANSAC approaches. The robust method will therefore be used for all relative pose estimation problems in the remainder of this thesis.

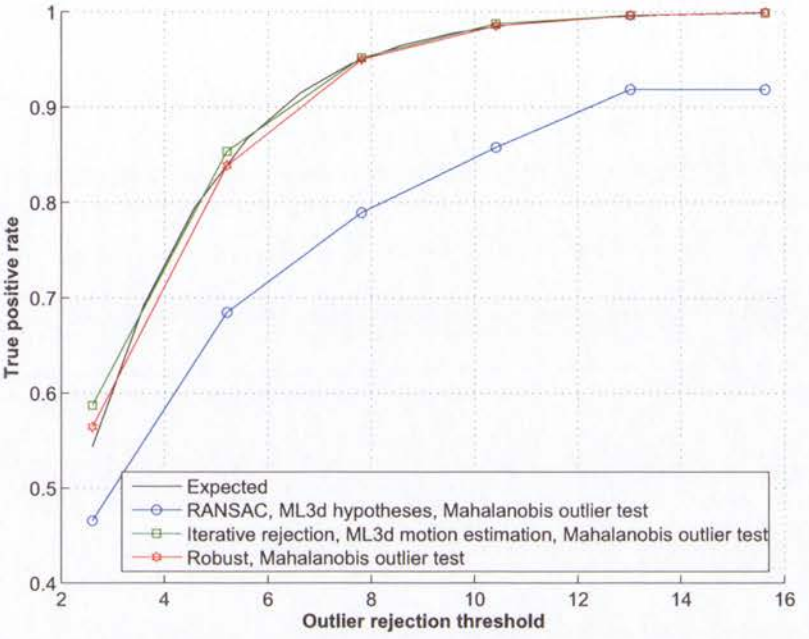


(a) Odometry simulations

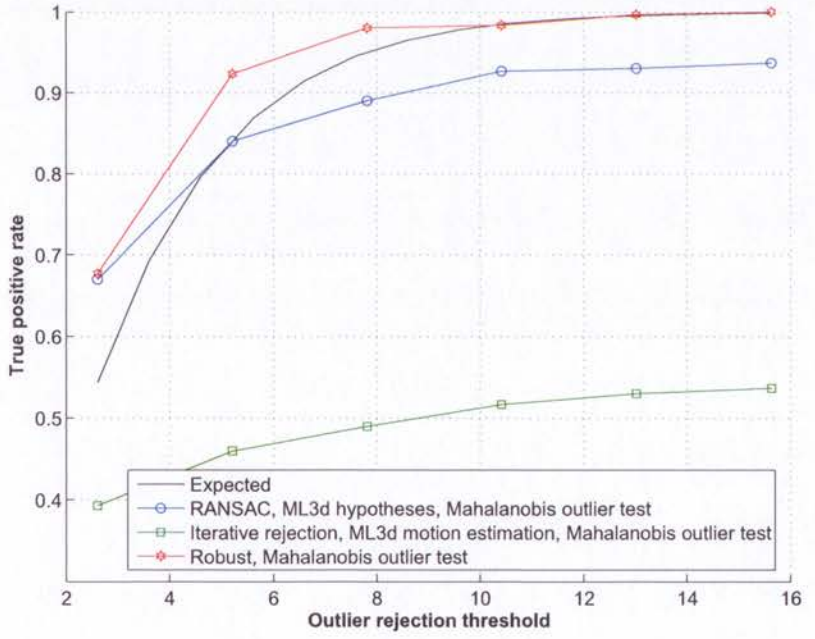


(b) Loop-closure simulations

Figure 4.11: Outlier classification ROC curves for 50 odometry simulations (a) and 50 loop-closure simulations (b). The dotted line represents the line of no discrimination expected from randomly guessed classifications.



(a) Odometry simulations



(b) Loop-closure simulations

Figure 4.12: Inlier acceptance rate of the Mahalanobis outlier test, applied to 50 odometry simulations (a) and 50 loop-closure simulations (b). The black line shows the expected inlier acceptance rate for perfect hypotheses equal to the true motion states.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
RANSAC	-0.0044	-0.0359	0.0064	-0.0146	0.0018	0.0063
Iterative Rejection	-0.0173	-0.0969	0.0062	-0.0333	0.0041	-0.0039
Robust	-0.0169	-0.0905	0.0054	-0.0315	0.0040	-0.0041

Table 4.9: Mean errors in relative pose estimates from 50 odometry simulations with outliers. The final motion estimates were produced using the maximum likelihood 3D registration method applied to all remaining features classified as inliers.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
RANSAC	0.1043	0.5932	0.0813	0.1661	0.0296	0.0430
Iterative Rejection	0.0776	0.3481	0.0537	0.0974	0.0206	0.0218
Robust	0.0742	0.3551	0.0542	0.0985	0.0198	0.0217

Table 4.10: Root mean square errors in relative pose estimates from 50 odometry simulations with outliers. The final motion estimates were produced using the maximum likelihood 3D registration method applied to all remaining features classified as inliers.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
RANSAC	-0.1022	-0.0757	0.1381	0.0394	0.0214	0.0089
Iterative Rejection	-11.8132	-16.2584	-6.7849	-7.0041	-4.9820	-47.4698
Robust	-0.0412	-0.0445	0.1241	0.0154	0.0099	0.0005

Table 4.11: Mean errors in relative pose estimates from 50 loop-closure simulations with outliers. The final motion estimates were produced using the maximum likelihood 3D registration method applied to all remaining features classified as inliers. The iterative rejection method failed to find a minimum of three consistent features to produce a motion estimate in 17 of the 50 simulations. The statistics listed for the iterative rejection method are produced from the remaining 33 simulations.

Method	x (cm)	y (cm)	z (cm)	ϕ (deg)	θ (deg)	ψ (deg)
RANSAC	0.8547	1.0632	0.5130	0.2887	0.3485	0.2116
Iterative Rejection	45.9831	32.2236	35.0337	18.8904	10.6793	64.7483
Robust	0.6437	0.9148	0.4076	0.2201	0.2988	0.1521

Table 4.12: Root mean square errors in relative pose estimates from 50 loop-closure simulations with outliers.The final motion estimates were produced using the maximum likelihood 3D registration method applied to all remaining features classified as inliers. The iterative rejection method failed to find a minimum of three consistent features to produce a motion estimate in 17 of the 50 simulations. The statistics listed for the iterative rejection method are produced from the remaining 33 simulations.

4.4 Considerations for Data Fusion

Four requirements for the relative pose estimator were established in Section 4.1 to allow the estimates to be used in an EKF data fusion algorithm:

1. The motion estimator must be unbiased
2. Consistent estimate covariances must be produced
3. The stereo relative pose estimates must be independent from other observations
4. Individual relative pose estimates must be independent from one another

The first two requirements have been considered in Sections 4.2 and 4.3, however the independence requirements have not yet been addressed.

In Section 4.1 it was mentioned that a prior motion estimate is commonly used to speed up feature tracking or limit possible feature associations. A concern is therefore whether or not the use of a prior motion estimate is likely to result in a correlated stereo relative pose estimate. This question may best be answered by considering the events required for the prior estimate to bias the stereo estimate. Firstly a set of outlier features consistent with the prior estimate must be found in the images, the feature observations must satisfy epipolar geometry constraints, and finally a sufficient number of features must be consistent with a single motion estimate. If a reasonable minimal number of consistent features (e.g. six) are required to produce a trusted motion estimate, such an occurrence is highly unlikely. The prior motion estimate and stereo relative pose estimate can therefore be safely considered to be independent.

To ensure a sequence of stereo relative pose estimates are independent from each other, reuse of feature observations must be avoided. For example, if the pose of b relative to a and the pose of c relative to b are calculated, the images acquired at pose b will be used in both estimates. If a feature was used in the calculation of both relative pose estimates, they would be correlated due to the errors in the feature observations at pose b that are propagated to both motion estimates. To avoid correlated estimates, a new set of features must be used in the generation of each stereo relative pose estimate.

4.5 Overview of the Algorithm

Sections 4.2 and 4.3 of this chapter have dealt with the selection of motion estimation and outlier rejection approaches for a stereo vision relative pose estimation algorithm. The

final algorithm will now be summarised, including the specification of the parameters and thresholds that will be used in future applications of the relative pose estimation approach.

The stereo relative pose estimation algorithms consists of four steps corresponding to the blocks in the process diagram presented in Figure 4.1:

1. Features are extracted in the first stereo image pair. The selection of a feature extraction algorithm will be application specific, varying with the environment and use-case (i.e. odometry or loop-closure).

2. The extracted features are tracked or associated with feature locations in the second image pair. The algorithm selected to perform this operation will also vary depending on the application.

3. The majority of outliers are rejected by applying epipolar constraints in each stereo image pair. Remaining outliers are rejected by calculating a robust relative pose hypothesis using the Cauchy ρ-function with a tuning parameter value of $c = 2.5$, then using the Mahalanobis outlier rejection test with the threshold $k_m = 7.815$ designed to accept 95% of inliers. The robust estimate is calculated using the random sample initialisation method, where each initial hypothesis is calculated by the maximum likelihood 3D registration method on a minimal set of 3 randomly selected features.

4. A final relative pose estimate and covariance is produced from the remaining inlier features by the maximum likelihood 3D registration algorithm initialised at the robust relative pose estimate.

4.6 Results

The stereo-vision relative pose estimation algorithm developed in this chapter has been applied to images acquired by the SeaBED AUV during a biodiversity assessment survey at Ningaloo Reef near Exmouth in Western Australia. The dataset contains images of sponge beds at a depth of approximately forty metres. The AUV acquired stereo image pairs at a rate of one hertz while travelling at a velocity of approximately half a metre per second. The DVL was used to maintain an altitude of two metres above the seafloor. These operating conditions result in features remaining within view of the cameras for two or three image pairs.

Feature extraction and tracking has been performed using the Harris corner detector and Lukas-Kanade tracker. Figure 4.13 shows the features extracted and tracked from two consecutive stereo image pairs. The features rejected as outliers due to epipolar constraints and inconsistency with the robust relative pose estimate are also shown.

While no ground truth is available, the DVL onboard the SeaBED AUV provides an alternate estimated trajectory for comparison. An initial stereo calibration was acquired in a swimming pool, and a rough measurement of the pose of the stereo-rig relative to the vehicle reference frame origin (located at the DVL) was obtained. Unfortunately, the pool dataset later proved insufficient to provide a good calibration, and the temporary viewport used in this first deployment of the stereo-rig appears to have deformed when under pressure, resulting in additional uncalibrated lens distortion. The initial stereo-rig calibration parameters acquired from the pool dataset were therefore optimised along with the pose of the stereo-rig relative to the vehicle frame, by minimising the difference between the relative pose estimates produced by the stereo-rig and DVL. This calibration was performed using a 30 second section of data that has not been reused in any further tests.

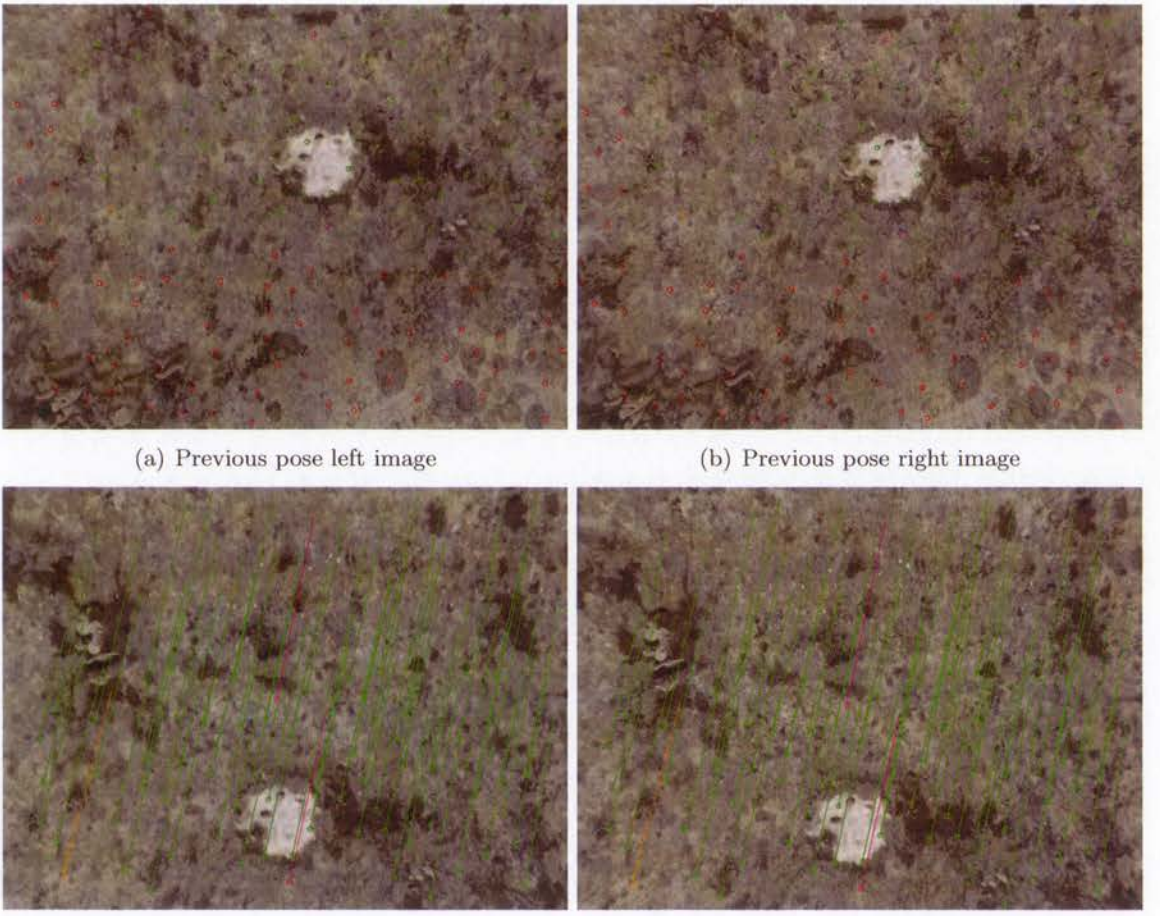
Figure 4.14 shows estimated vehicle trajectories produced by the stereo relative pose estimation algorithm and an EKF combining the DVL velocity, tilt sensor and compass observations. The desired vehicle path is a 100 metre linear transect heading north. The individual relative pose estimates between the vehicle locations of consecutive image pairs are shown in Figure 4.15, and the estimates of the aggregated vehicle pose relative to the initial vehicle pose are shown displayed in Figure 4.16. The stereo odometry estimates from a longer and more complex vehicle trajectory are shown in Figure 4.17, and the individual relative pose and aggregated pose estimates are shown in Figures 4.18 and 4.19 respectively.

The individual relative pose estimates produced by the stereo-rig in Figures 4.15 and 4.18 show excellent consistency with the DVL-based estimates. The stereo estimates capture not only the general vehicle motion of half a metre forwards (as may be expected due to the optimisation of the stereo calibration parameters to match the DVL), but also the small variations in the position and orientation states.

The cumulative relative pose estimate averages in Figures 4.15 and 4.18 show small biases that result in slow drift in the aggregated vehicle pose estimates of Figures 4.16 and 4.19.

While the stereo calibration parameters are likely to imperfect, some of the differences between the two estimated trajectories could be attributed to the DVL and compass. The deviations in the estimated roll angles in Figure 4.18(d) after the turning maneuver suggest a heading-dependant bias, probably due to compass errors when optimising the stereo calibration parameters.

When performing visual odometry, the aggregated orientation estimates will always accumulate errors that propagate into large position state errors. For long trajectories, absolute orientation sensors (such as the tilt sensors and compass used in the DVL) will be necessary to provide correction for the drifting estimates.



(a) Previous pose left image

(b) Previous pose right image

(c) Current pose left image

(d) Current pose right image

Figure 4.13: Stereo odometry images and features. Feature positions are marked by circles, and tracks are drawn by lines linking the image coordinates of each feature in the previous and current poses. Features that could not be tracked are drawn in red, features with observations within a stereo pair that did not satisfy epipolar constraints are drawn in orange, and features that were inconsistent with the final motion estimate are drawn in pink. Accepted features that were used to produce the final motion estimate are drawn in green.

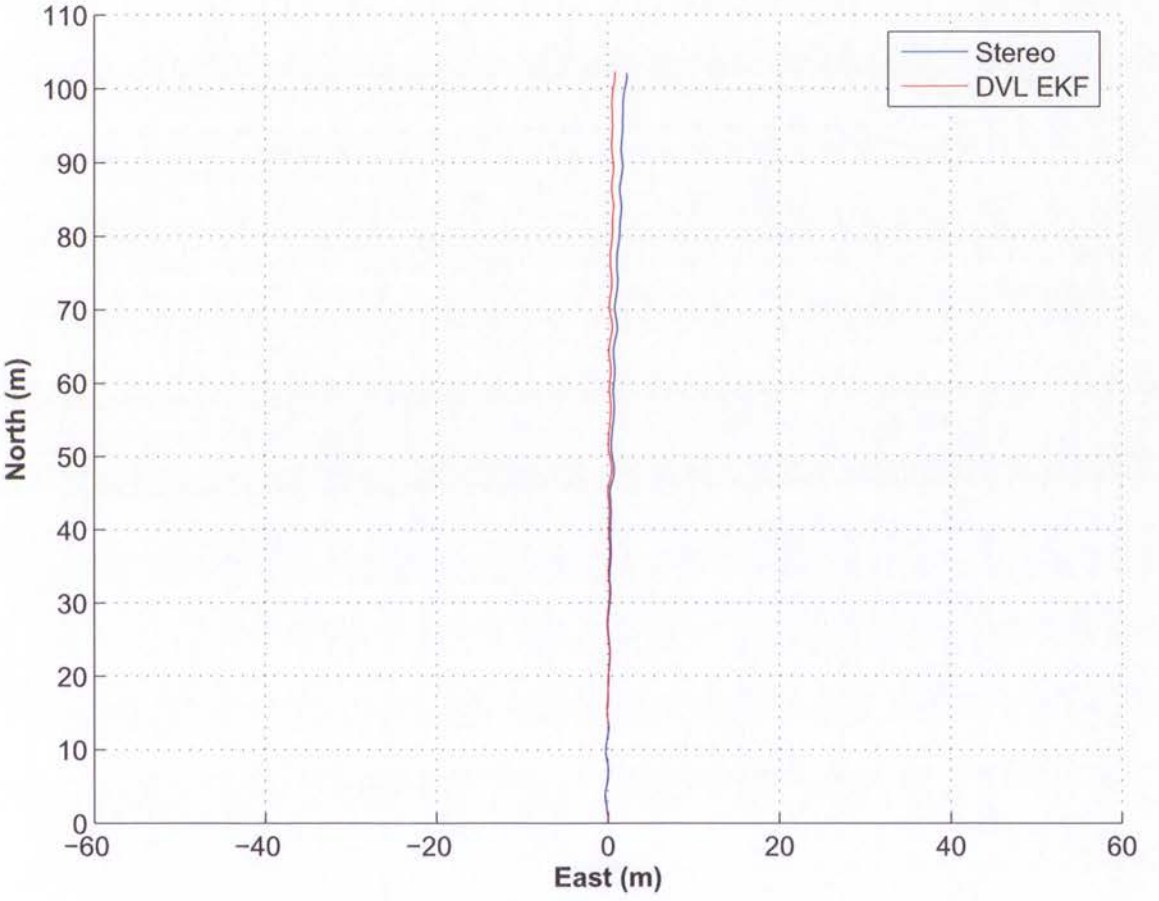
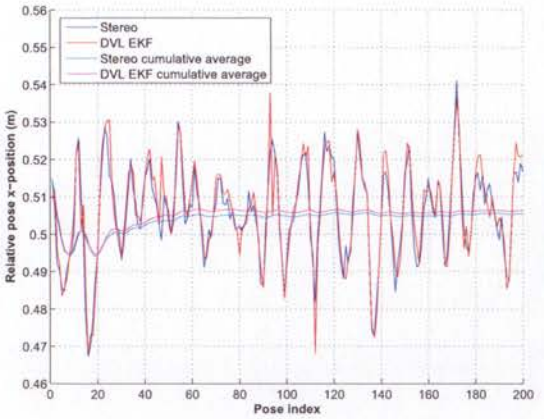
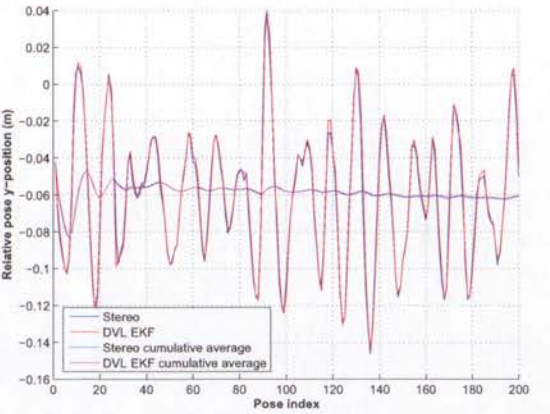


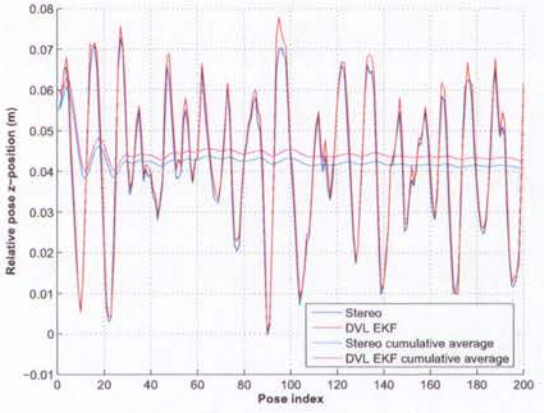
Figure 4.14: Comparison of stereo odometry and DVL-based vehicle trajectory estimates for a linear transect of approximately 100 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.



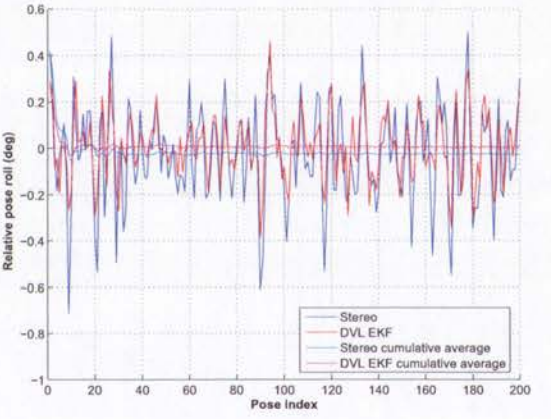
(a) X-position



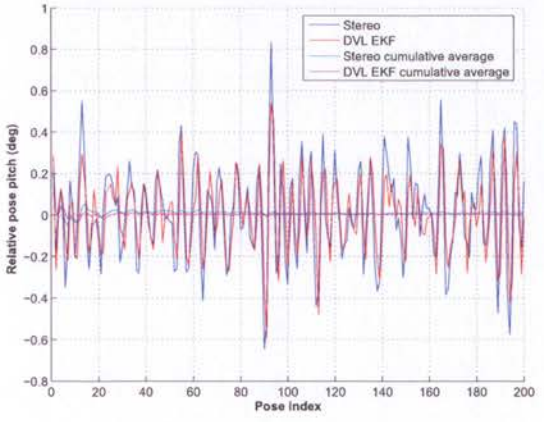
(b) Y-position



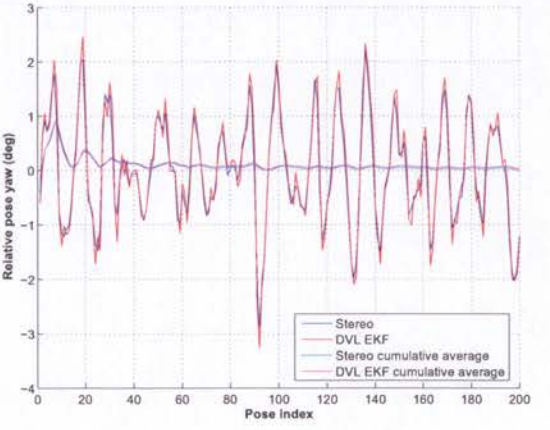
(c) Z-position



(d) Roll Euler angle

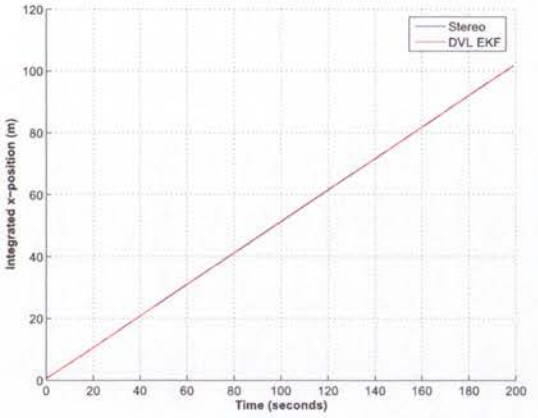


(e) Pitch Euler angle

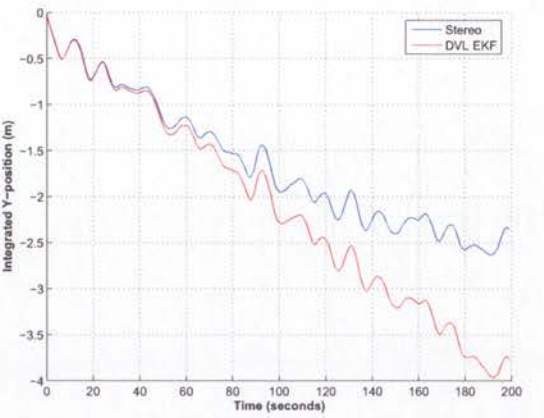


(f) Yaw Euler angle

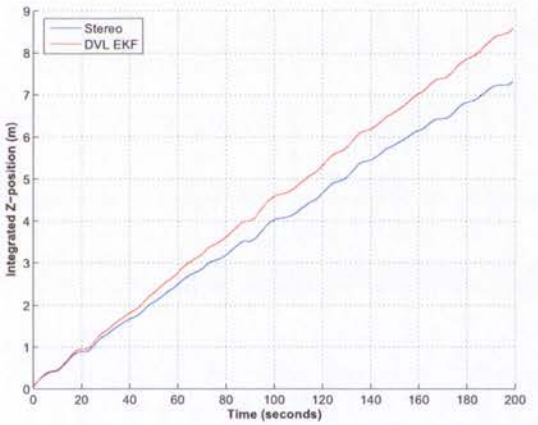
Figure 4.15: Comparison of stereo odometry and DVL-based estimates of the relative vehicle poses between successive image pairs for a linear transect of approximately 100 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.



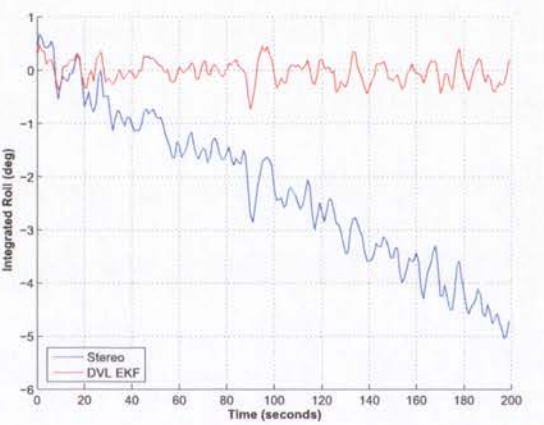
(a) X-position



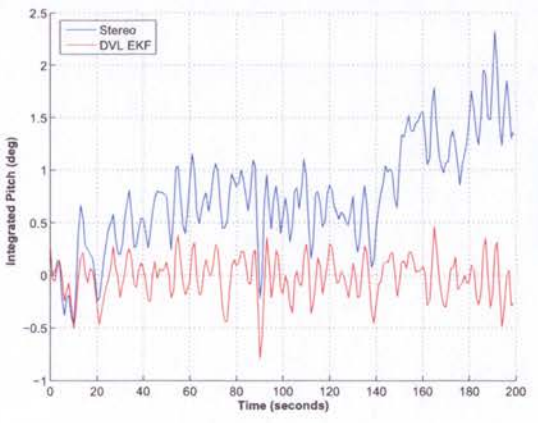
(b) Y-position



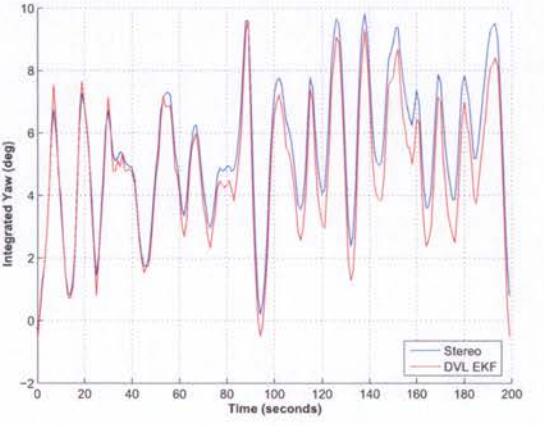
(c) Z-position



(d) Roll Euler angle



(e) Pitch Euler angle



(f) Yaw Euler angle

Figure 4.16: Comparison of stereo odometry and DVL-based estimates of the current vehicle pose relative to the initial vehicle pose for a linear transect of approximately 100 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.

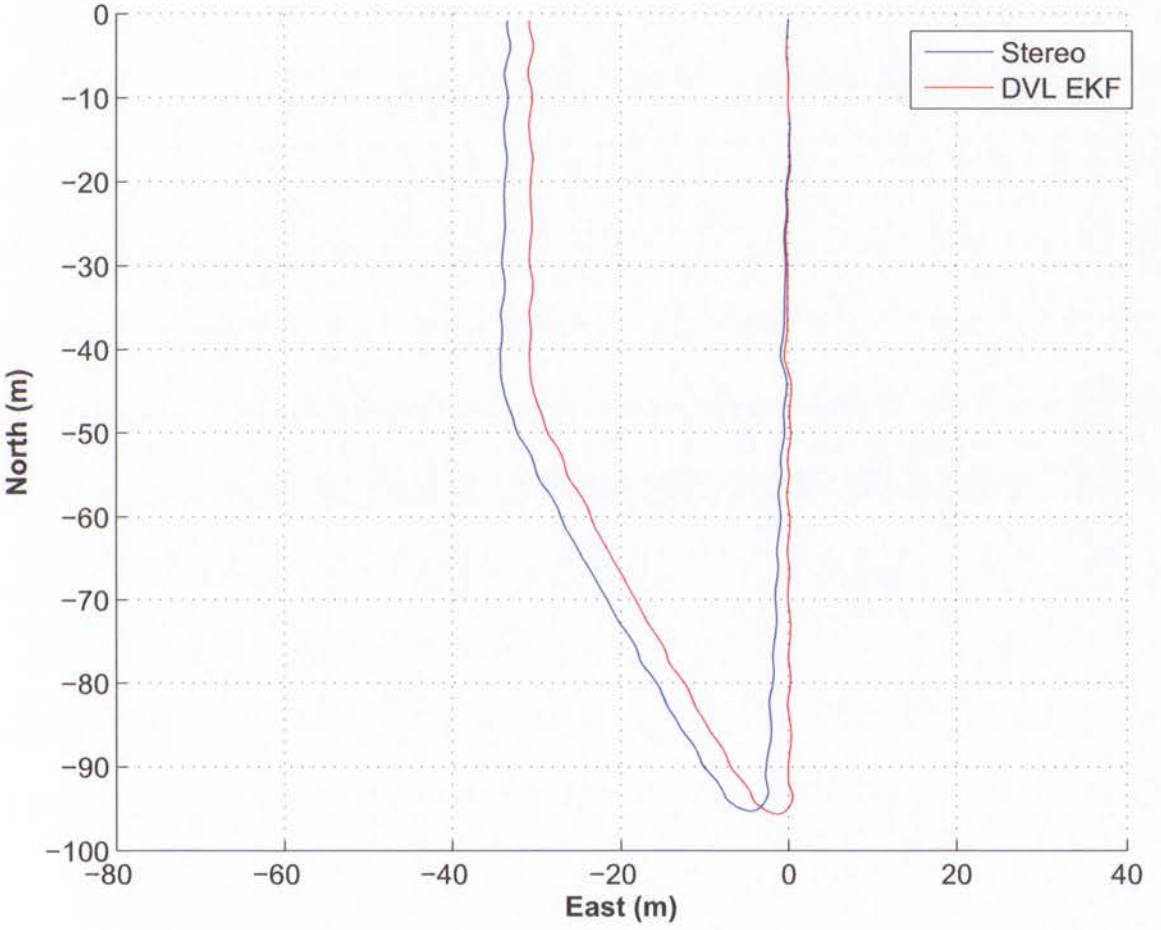
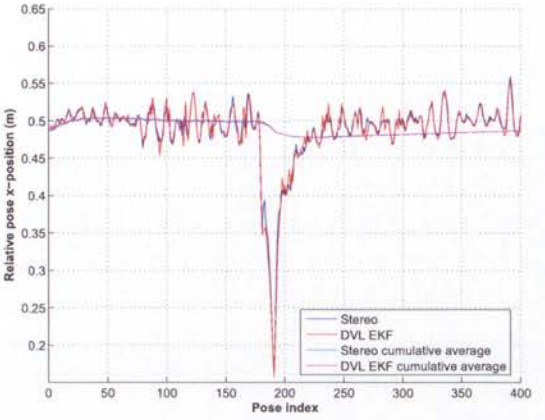
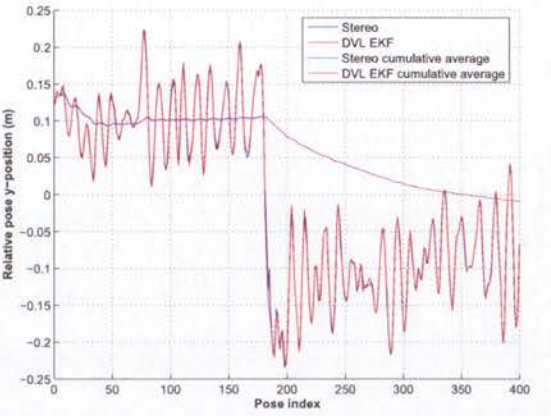


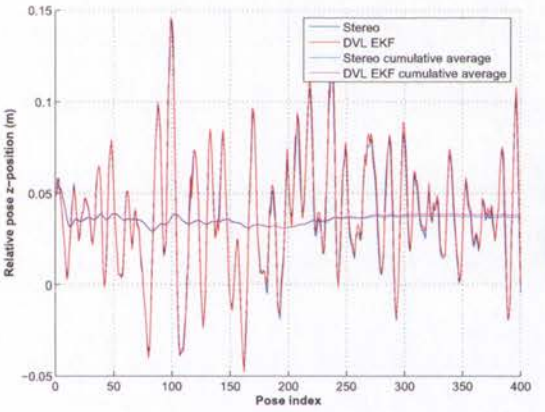
Figure 4.17: Comparison of stereo odometry and DVL-based vehicle trajectory estimates for a turning maneuver with a total path length of approximately 200 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.



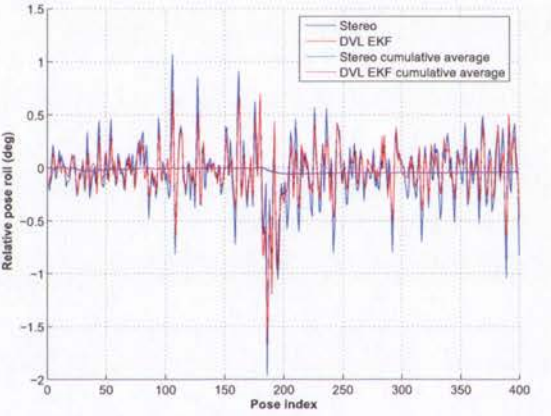
(a) X-position



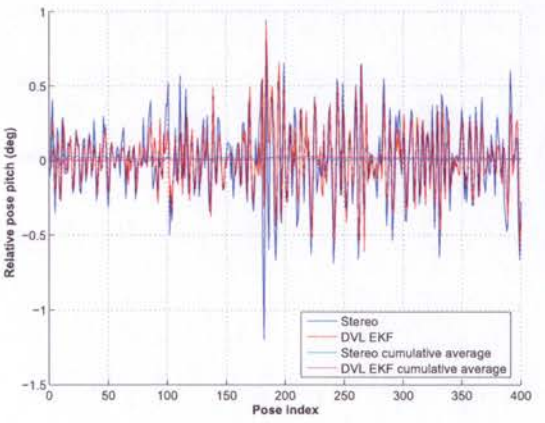
(b) Y-position



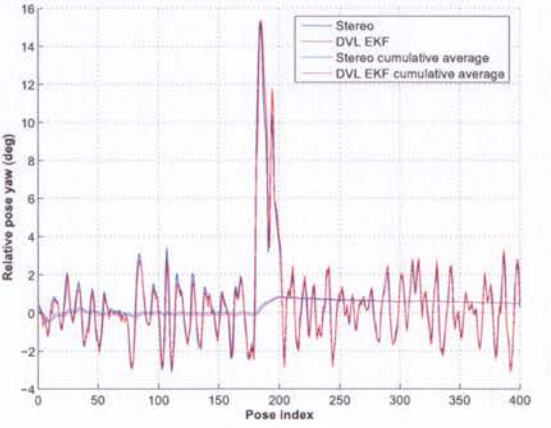
(c) Z-position



(d) Roll Euler angle

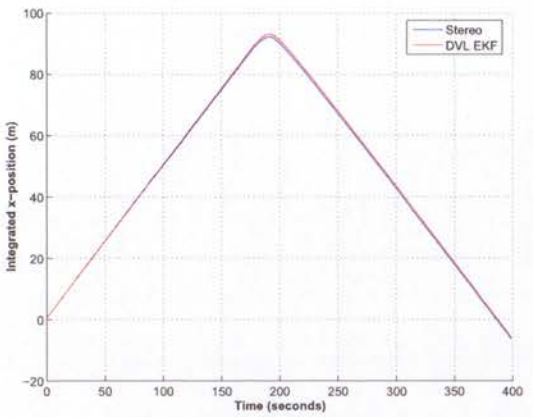


(e) Pitch Euler angle

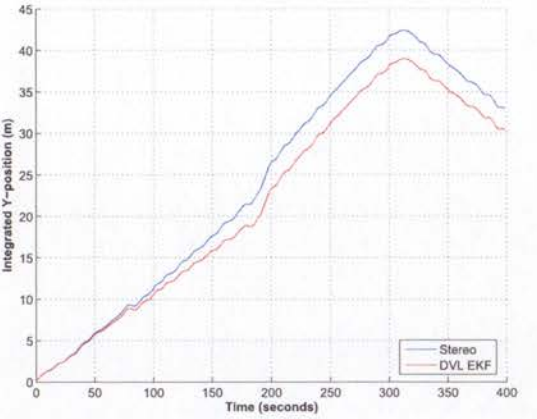


(f) Yaw Euler angle

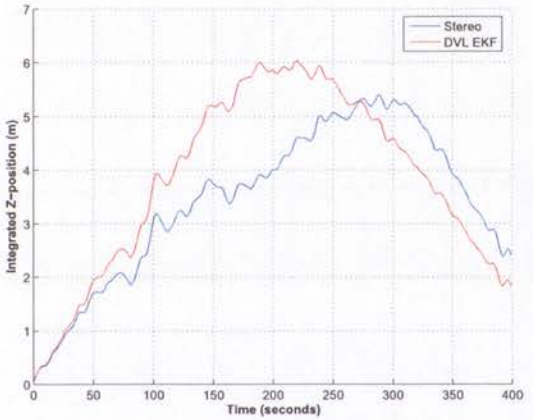
Figure 4.18: Comparison of stereo odometry and DVL-based estimates of the relative vehicle poses between successive image pairs for a turning maneuver with a total path length of approximately 200 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.



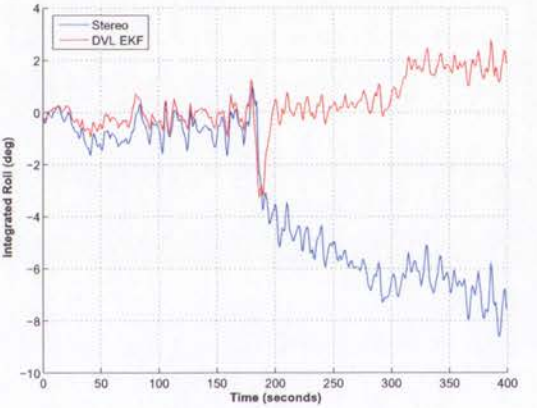
(a) X-position



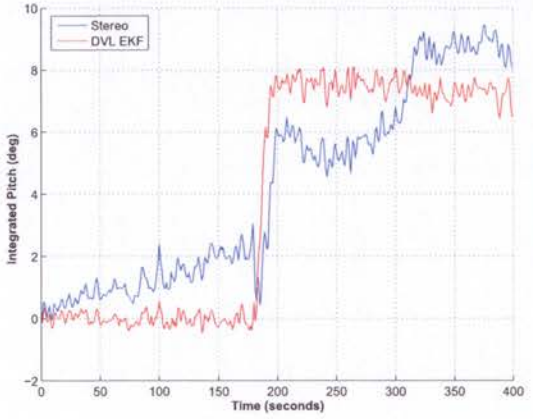
(b) Y-position



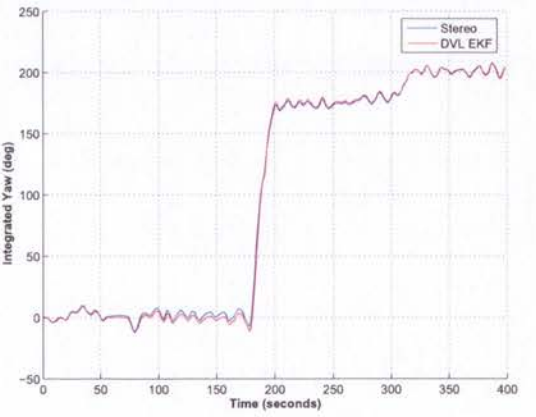
(c) Z-position



(d) Roll Euler angle



(e) Pitch Euler angle



(f) Yaw Euler angle

Figure 4.19: Comparison of stereo odometry and DVL-based estimates of the current vehicle pose relative to the initial vehicle pose for a turning maneuver with a total path length of approximately 200 metres. The DVL-based estimate has been produced using an EKF fusing velocity, tilt and compass observations.

4.7 Summary

This chapter has detailed the development of a stereo-vision relative pose estimation algorithm, focusing on the motion parameter estimation and outlier rejection problems once a set of features have been extracted and tracked between images.

A range of relative pose estimation methods that minimise various cost functions were reviewed and tested under simulation. Approaches that ignore some or all observation uncertainties were shown to result in estimators with large errors and biases.

A novel motion parameter estimation approach based on minimising feature 3D registration errors was presented, and shown to result in a computationally efficient, unbiased estimator with known covariance. Summarising stereo observations of a feature as a triangulated position estimate and covariance was shown to produce similar results to bundle adjustment, while resulting in significant efficiency improvements due to a more compact observation vector and a less complex observation model.

A selection of outlier rejection methods were reviewed and tested on simulated data corrupted by the types of outliers expected in vision data. It was shown that while an iterative outlier rejection approach produces acceptable results in the presence of outliers with only small errors, the algorithm fails in the presence of large outliers such as those caused by incorrect feature associations. RANSAC provided better results, but was shown to accept fewer inlier features than expected due to a lack of filtering in its motion hypotheses.

A novel outlier rejection approach based on a robust redescending M-estimator was presented. Instead of being used in a typical role to provide a final parameter estimate without identifying outliers, the robust estimator was used to provide motion hypotheses for an outlier classification test. A maximum likelihood parameter estimate and covariance was then be obtained from the remaining inliers. The robust estimator method was shown to produce better motion hypotheses than RANSAC, resulting in a superior outlier classifier and motion estimator.

The stereo-vision relative pose estimation algorithm was demonstrated on real data by estimating of the odometry of an AUV. Similar results to a trajectory produced using a DVL were obtained. The use of the relative pose estimator in a loop-closure application will be demonstrated in the Chapter 5.

Chapter 5

Simultaneous Localisation and Mapping using Stereo Vision

5.1 Introduction

This chapter presents a view-based SLAM algorithm suitable for large-scale exploration applications. Experimental results using the stereo-camera relative pose estimation algorithm developed in Chapter 4 to generate loop-closure observations are presented, demonstrating the ability of vision sensors to correct positioning errors accumulated from dead-reckoning.

5.2 SLAM Frameworks

Firstly, the suitability of SLAM frameworks to large-scale visual navigation will be evaluated, considering the properties of the feature extraction and association operations investigated in Chapter 2. The two candidate frameworks are feature-based and view-based SLAM, which are distinguished by the selection of variables estimated by a filter. In feature-based SLAM [19, 41, 63, 101, 112], the positions of features extracted from sensor observations are estimated, while in view-based SLAM [26–29], the state vector is augmented with a set of vehicle poses at locations where sensor data is acquired.

When adapting the feature-based SLAM framework to a new application, two operations must be implemented: feature selection and data association. Feature selection is the process of extracting new features from sensor data, and deciding which will be added to the state vector. Data association addresses the problem of matching new observations to existing features when the vehicle returns to a previously visited location in a situation

known as a 'loop-closure'. Loop-closure observations are essential to the SLAM process, since they provide correction of dead-reckoning drift accumulated over time. In a typical implementation of feature-based SLAM using an EKF, the computational complexity of prediction and observation operations is quadratic in the number of augmented features.

The most common platform for SLAM research has been indoor robots equipped with laser range-finders. Feature-based SLAM is well suited to this application, since a small set of reflective features or corners can be easily extracted with high repeatability from the sensor data. The sparse distribution of such features solves the feature selection problem, allows data association to be performed using geometry, and makes the quadratic complexity of feature-based EKF-SLAM manageable.

In contrast, the properties of the algorithms used to extract and associate features in wide-baseline images make feature-based SLAM poorly suited to visual navigation. While rich visual feature descriptions provide a solution for data association, feature selection becomes a difficult problem due to the large number of features that can be extracted from a single image, and the low extraction repeatability and association rates. Augmenting the state vector with huge numbers of features is infeasible due to the resulting computational complexity, however if only a small set of features are selected, the probability of successfully associating a feature to produce a loop-closure observation is low.

The critical advantage of the view-based framework for vision application is the ability to use all the sensor data, rather than a sparse set of selected features. In the view-based framework, a loop-closure observation is applied by constraining the relative location of two augmented poses. Instead of guessing in advance which features are likely to be matched in the future, the relative pose constraint can be created using whatever portion of the sensor data can be matched when the association process is performed.

An additional benefit of the view-based approach is its ability to handle delayed observations. Visual feature extraction and association are time consuming processes, so a delay is likely to occur between the time an image is acquired and a loop-closure observation is produced. However, in the view-based framework, a relative pose constraint can be applied between two augmented poses whenever the image analysis operations are complete.

The view-based SLAM framework also has a computational efficiency advantage when the estimation process is performed in information form. The information matrix for a view-based SLAM problem is exactly sparse, resulting in a significant benefit over approaches with dense covariance or information matrices caused by marginalising past vehicle poses [27].

Due to avoidance of the feature selection problem, the inherent ability to handle delayed observations and the efficiency when using the information form, the view-based SLAM framework will be utilised in this thesis for large-scale visual navigation.

5.3 Viewpoint Augmented Navigation (VAN)

5.3.1 The Estimated State Vector

In the VAN framework, the current vehicle state is estimated along with a selection of past vehicle poses, leading to a state estimate vector of the form

$$\hat{\mathbf{x}}^{+}(t_k) = \begin{bmatrix} \hat{\mathbf{x}}^{+}_{p_1}(t_k) \\ \vdots \\ \hat{\mathbf{x}}^{+}_{p_n}(t_k) \\ \hat{\mathbf{x}}^{+}_{v}(t_k) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}^{+}_{t}(t_k) \\ \hat{\mathbf{x}}^{+}_{v}(t_k) \end{bmatrix} \tag{5.1}$$

and covariance matrix of the form

$$\mathbf{P}^{+}(t_k) = \begin{bmatrix} \mathbf{P}^{+}_{tt}(t_k) & \mathbf{P}^{+}_{tv}(t_k) \\ \mathbf{P}^{+\mathsf{T}}_{tv}(t_k) & \mathbf{P}^{+}_{vv}(t_k) \end{bmatrix} \tag{5.2}$$

where $\hat{\mathbf{x}}^{+}_{t}(t_k) = \left[\hat{\mathbf{x}}^{+\mathsf{T}}_{p_1}(t_k), \ldots, \hat{\mathbf{x}}^{+\mathsf{T}}_{p_n}(t_k)\right]^{\mathsf{T}}$ is a vector of trajectory states consisting of n past vehicle vectors.

In the information form, the filter maintains the information matrix $\mathbf{Y}^{+}(t_k)$, which is the inverse of the covariance matrix

$$\mathbf{Y}^{+}(t_k) = \left[\mathbf{P}^{+}(t_k)\right]^{-1} \tag{5.3}$$

and the information vector $\hat{\mathbf{y}}^{+}(t_k)$, which is related to the state estimate by

$$\hat{\mathbf{y}}^{+}(t_k) = \mathbf{Y}^{+}(t_k)\,\hat{\mathbf{x}}^{+}(t_k) \tag{5.4}$$

For the VAN framework, the information vector and matrix have the form

$$\hat{\mathbf{y}}^{+}(t_k) = \begin{bmatrix} \hat{\mathbf{y}}^{+}_{t}(t_k) \\ \hat{\mathbf{y}}^{+}_{v}(t_k) \end{bmatrix} \tag{5.5}$$

$$\mathbf{Y}^{+}(t_k) = \begin{bmatrix} \mathbf{Y}^{+}_{tt}(t_k) & \mathbf{Y}^{+}_{tv}(t_k) \\ \mathbf{Y}^{+\mathsf{T}}_{tv}(t_k) & \mathbf{Y}^{+}_{vv}(t_k) \end{bmatrix} \tag{5.6}$$

5.3.2 The Estimation Process

The estimation process used in an Extended Information Filter (EIF) consists of the same predict, observe and update steps as the EKF presented in Chapter 3.

Prediction with Augmentation

When performing a prediction operation to propagate the vehicle state forwards to a new timestep, the current vehicle pose should be kept in the state vector if it marks the location where an observation such as an image which may be used in future loop-closure observations was acquired

The augmented predicted estimate and covariance are

$$\hat{\mathbf{x}}^{-}\left(t_k\right) = \begin{bmatrix} \hat{\mathbf{x}}_t^{+}\left(t_{k-1}\right) \\ \hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right) \\ \mathbf{f}_v\left[\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)\right] \end{bmatrix} \tag{5.7}$$

$$\mathbf{P}^{-}\left(t_k\right) = \begin{bmatrix} \mathbf{P}_{tt}^{+}\left(t_{k-1}\right) & \mathbf{P}_{tv}^{+}\left(t_{k-1}\right) & \mathbf{P}_{tv}^{+}\left(t_{k-1}\right)\nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right) \\ \mathbf{P}_{tv}^{+\mathsf{T}}\left(t_{k-1}\right) & \mathbf{P}_{vv}^{+}\left(t_{k-1}\right) & \mathbf{P}_{vv}^{+}\left(t_{k-1}\right)\nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right) \\ \nabla_x\mathbf{f}_v\left(t_k\right)\mathbf{P}_{tv}^{+\mathsf{T}}\left(t_{k-1}\right) & \nabla_x\mathbf{f}_v\left(t_k\right)\mathbf{P}_{vv}^{+}\left(t_{k-1}\right) & \nabla_x\mathbf{f}_v\left(t_k\right)\mathbf{P}_{vv}^{+}\left(t_{k-1}\right)\nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right) + \mathbf{Q}\left(t_k\right) \end{bmatrix} \tag{5.8}$$

where $\mathbf{f}_v\left[\cdot\right]$ is the vehicle process function, $\nabla_x\mathbf{f}_v\left(t_k\right)$ is the Jacobian of the vehicle process function relative to the vehicle states, and $\mathbf{Q}\left(t_k\right)$ is the vehicle process model covariance as defined in Section 3.2.2.

Using the relationship between the covariance and information forms stated in Equations 5.3 and 5.4, the prediction step can be represented with the augmented prediction information vector and matrix

$$\hat{\mathbf{y}}^{-}\left(t_k\right) = \begin{bmatrix} \hat{\mathbf{y}}_t^{+}\left(t_{k-1}\right) \\ \hat{\mathbf{y}}_v^{+}\left(t_{k-1}\right) - \nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right)\mathbf{Q}^{-1}\left(t_k\right)\left(\mathbf{f}_v\left[\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)\right] - \nabla_x\mathbf{f}_v\left(t_k\right)\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)\right) \\ \mathbf{Q}^{-1}\left(t_k\right)\left(\mathbf{f}_v\left[\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)\right] - \nabla_x\mathbf{f}_v\left(t_k\right)\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)\right) \end{bmatrix} \tag{5.9}$$

$$\mathbf{Y}^{-}\left(t_k\right) = \begin{bmatrix} \mathbf{Y}_{tt}^{+}\left(t_{k-1}\right) & \mathbf{Y}_{tv}^{+}\left(t_{k-1}\right) & \mathbf{0} \\ \mathbf{Y}_{tv}^{+\mathsf{T}}\left(t_{k-1}\right) & \mathbf{Y}_{vv}^{+}\left(t_{k-1}\right) + \nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right)\mathbf{Q}^{-1}\left(t_k\right)\nabla_x\mathbf{f}_v\left(t_k\right) & -\nabla_x^{\mathsf{T}}\mathbf{f}_v\left(t_k\right)\mathbf{Q}^{-1}\left(t_k\right) \\ \mathbf{0} & -\mathbf{Q}^{-1}\left(t_k\right)\nabla_x\mathbf{f}_v\left(t_k\right) & \mathbf{Q}^{-1}\left(t_k\right) \end{bmatrix} \tag{5.10}$$

Equation 5.9 requires the prior vehicle pose state estimate $\hat{\mathbf{x}}_v^{+}\left(t_{k-1}\right)$, and both Equations 5.9 and 5.10 require the vehicle process function Jacobian evaluated at the prior vehicle

	Joint $p(\boldsymbol{\alpha}, \boldsymbol{\beta})$	Marginal $p(\boldsymbol{\alpha}) = \int p(\boldsymbol{\alpha}, \boldsymbol{\beta}) d\boldsymbol{\beta}$
Covariance Form	$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_\alpha \\ \hat{\mathbf{x}}_\beta \end{bmatrix}$	$\hat{\mathbf{x}} = \hat{\mathbf{x}}_\alpha$
	$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{\alpha\alpha} & \mathbf{P}_{\alpha\beta} \\ \mathbf{P}_{\beta\alpha} & \mathbf{P}_{\beta\beta} \end{bmatrix}$	$\mathbf{P} = \mathbf{P}_{\alpha\alpha}$
Information Form	$\hat{\mathbf{y}} = \begin{bmatrix} \hat{\mathbf{y}}_\alpha \\ \hat{\mathbf{y}}_\beta \end{bmatrix}$	$\hat{\mathbf{y}} = \hat{\mathbf{y}}_\alpha - \mathbf{Y}_{\alpha\beta}\mathbf{Y}_{\beta\beta}^{-1}\hat{\mathbf{y}}_\beta$
	$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_{\alpha\alpha} & \mathbf{Y}_{\alpha\beta} \\ \mathbf{Y}_{\beta\alpha} & \mathbf{Y}_{\beta\beta} \end{bmatrix}$	$\mathbf{Y} = \mathbf{Y}_{\alpha\alpha} - \mathbf{Y}_{\alpha\beta}\mathbf{Y}_{\beta\beta}^{-1}\mathbf{Y}_{\beta\alpha}$

Table 5.1: Marginalisation in covariance and information form. Adapted from [27].

pose estimate $\nabla_x \mathbf{f}_v(t_k)$. In an information filter the state estimates are not immediately available, and efficient algorithms for their recovery will be a major focus of this chapter.

Prediction without Augmentation

Only vehicle poses corresponding to the locations where data such as images were acquired need to be maintained in the filter. In most cases, the prediction operation should be performed without augmenting the state vector with an additional pose.

The equations for prediction without augmentation can be obtained through marginalisation of the previous vehicle pose from Equations 5.9 and 5.10. Equations for marginalisation in covariance and information form are presented in Table 5.1. From [27], the resulting prediction information vector and matrix are

$$\hat{\mathbf{y}}^-(t_k) = \begin{bmatrix} \hat{\mathbf{y}}_t^+(t_{k-1}) - \mathbf{Y}_{tv}^+(t_{k-1})\,\boldsymbol{\Omega}^{-1}(t_k)\left(\hat{\mathbf{y}}_v^+(t_{k-1}) - \nabla_x^\mathsf{T}\mathbf{f}_v(t_k)\,\mathbf{Q}^{-1}(t_k)\,\boldsymbol{\delta}(t_k)\right) \\ \mathbf{Q}^{-1}(t_k)\,\nabla_x\mathbf{f}_v(t_k)\,\boldsymbol{\Omega}^{-1}(t_k)\,\hat{\mathbf{y}}_v^+(t_{k-1}) + \boldsymbol{\Psi}(t_k)\,\boldsymbol{\delta}(t_k) \end{bmatrix} \tag{5.11}$$

$$\mathbf{Y}^-(t_k) = \begin{bmatrix} \mathbf{Y}_{tt}^+(t_{k-1}) - \mathbf{Y}_{tv}^+(t_{k-1})\,\boldsymbol{\Omega}^{-1}(t_k)\,\mathbf{Y}_{tv}^{+\mathsf{T}}(t_{k-1}) & \mathbf{Y}_{tv}^+(t_{k-1})\,\boldsymbol{\Omega}^{-1}(t_k)\,\nabla_x^\mathsf{T}\mathbf{f}_v(t_k)\,\mathbf{Q}^{-1}(t_k) \\ \mathbf{Q}^{-1}(t_k)\,\nabla_x\mathbf{f}_v(t_k)\,\boldsymbol{\Omega}^{-1}(t_k)\,\mathbf{Y}_{tv}^{+\mathsf{T}}(t_{k-1}) & \boldsymbol{\Psi}(t_k) \end{bmatrix} \tag{5.12}$$

where

$$\boldsymbol{\delta}(t_k) = \mathbf{f}_v\left[\hat{\mathbf{x}}_v^+(t_{k-1})\right] - \nabla_x\mathbf{f}_v(t_k)\,\hat{\mathbf{x}}_v^+(t_{k-1}) \tag{5.13}$$

$$\boldsymbol{\Omega}(t_k) = \mathbf{Y}_{vv}^+(t_{k-1}) + \nabla_x^\mathsf{T}\mathbf{f}_v(t_k)\,\mathbf{Q}^{-1}(t_k)\,\nabla_x\mathbf{f}_v(t_k) \tag{5.14}$$

$$\boldsymbol{\Psi}(t_k) = \left(\mathbf{Q}(t_k) + \nabla_x\mathbf{f}_v(t_k)\left[\mathbf{Y}_{vv}^+(t_{k-1})\right]^{-1}\nabla_x^\mathsf{T}\mathbf{f}_v(t_k)\right)^{-1} \tag{5.15}$$

Observation

An observation yields an innovation vector measuring the difference between the actual and predicted observation

$$\boldsymbol{\nu}(t_k) = \mathbf{z}(t_k) - \mathbf{h}\left[\hat{\mathbf{x}}^-(t_k)\right] \tag{5.16}$$

Update

The information vector and matrix are updated to incorporate an observation using the equations

$$\hat{\mathbf{y}}^+(t_k) = \hat{\mathbf{y}}^-(t_k) + \mathbf{i}(t_k) \tag{5.17}$$

$$\mathbf{Y}^+(t_k) = \mathbf{Y}^-(t_k) + \mathbf{I}(t_k) \tag{5.18}$$

where the update vector and matrix are

$$\mathbf{i}(t_k) = \nabla_x \mathbf{h}(t_k)\, \mathbf{R}^{-1}(t_k) \left(\boldsymbol{\nu}(t_k) + \nabla_x \mathbf{h}(t_k)\, \hat{\mathbf{x}}^-(t_k)\right) \tag{5.19}$$

$$\mathbf{I}(t_k) = \nabla_x \mathbf{h}(t_k)\, \mathbf{R}^{-1}(t_k)\, \nabla_x^{\mathsf{T}} \mathbf{h}(t_k) \tag{5.20}$$

As was the case for prediction, state estimates are required for the observation and update operations in information form. A typical SLAM observation is sparse however, with only a few of the estimated states being observed, and only estimates of the observed states are required to calculate the innovation vector and update the information vector and information matrix.

5.3.3 Loop-closure Hypotheses

Each time data such as an image is acquired, the compatibility of the current and augmented vehicle poses for a loop closure observation should be evaluated to produce a set of loop-closure hypotheses. Identifying a small subset of all the possible pairs of poses is important to reduce the number of hypotheses that need to undergo computationally expensive image analysis.

Evaluating whether a pair of poses should form a loop-closure hypothesis requires their joint distribution. For example, the predicted joint distribution for the previous pose at index i

and the current vehicle pose is defined by the estimate vector and covariance matrix

$$\hat{\mathbf{x}}^{-}_{(i,v)}(t_k) = \begin{bmatrix} \hat{\mathbf{x}}^{-}_{i}(t_k) \\ \hat{\mathbf{x}}^{-}_{v}(t_k) \end{bmatrix} \tag{5.21}$$

$$\mathbf{P}^{-}_{(i,v)}(t_k) = \begin{bmatrix} \mathbf{P}^{-}_{ii}(t_k) & \mathbf{P}^{-}_{iv}(t_k) \\ \mathbf{P}^{-\mathsf{T}}_{iv}(t_k) & \mathbf{P}^{-}_{vv}(t_k) \end{bmatrix} \tag{5.22}$$

A suitable pair of poses for a loop-closure hypothesis have a joint distribution that suggest it is likely that the poses are close enough for images acquired at each pose to overlap. The exact test used to decide if a joint distribution is compatible for a loop-closure observation is dependent on the properties of the sensor. The test used for a stereo-camera rig is presented along with experimental results in Section 5.9.

The joint distributions for a pair of poses is formed through the marginalisation of all other estimated states. As shown in Table 5.1 this can be done efficiently in the covariance form by extracting the relevant subvectors and submatrices from the state estimate vector and covariance matrix. Marginalisation in the information form is computationally expensive however, requiring the inversion of the sections of the information matrix corresponding to the marginalised states.

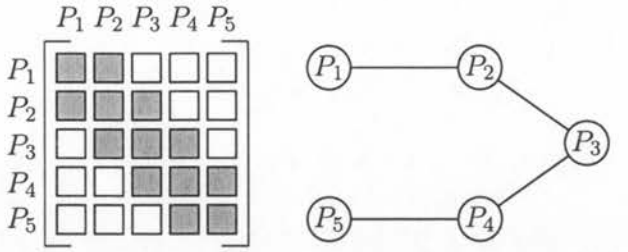
5.3.4 Properties of the VAN Framework in the Information Form

Sparsity of the information matrix

In information-form VAN, elements of the information matrix off the block-diagonal are non-zero only if an observation relating the two corresponding poses has been applied to the filter. Figure 5.1 shows an example of an information matrix sparsity pattern and Markov graph that results from dead-reckoning. Since each pose is related to the previous and next pose through odometry, dead-reckoning results in a block tri-diagonal matrix. The Markov graph provides a visual representation of the relationship between the estimated variables, with an edge in the graph corresponding to a non-zero block in the information matrix.

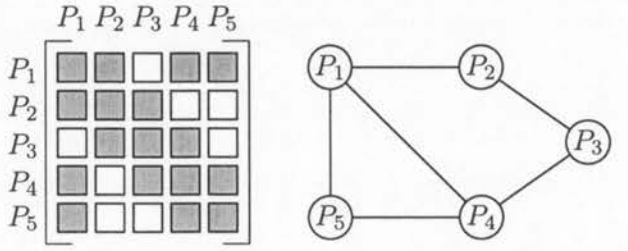
When loop-closure observations are applied to the filter, additional non-zero elements in the information matrix are created at the locations corresponding to the two observed poses. Figure 5.2 displays the information matrix resulting from taking the system presented in Figure 5.1, and adding loop closure observations between the first pose and last two poses.

The sparsity of the information matrix is important for the computational efficiency and storage requirements of the filter. In large-scale applications with many augmented poses,



(a) Information matrix (b) Markov graph

Figure 5.1: Information matrix sparsity pattern and Markov graph for a set of five poses related only by odometry constraints.



(a) Information matrix (b) Markov graph

Figure 5.2: Information matrix sparsity pattern and Markov graph for a SLAM problem consisting of five poses. In this example, loop-closure observations have been applied to constrain the final two poses relative to the initial pose.

EKF-based SLAM approaches are infeasible due to dense covariance matrices requiring more memory than is available on current computers.

Efficient Observation Operations

The information-form update of Equations 5.17 and 5.18 consists of simple addition operations where only the entries in the information vector and matrix corresponding to the observed states are changed. Depending on the information matrix storage format used, an observation can be a constant time operation in information form, which is a significant advantage over the quadratic complexity of an EKF update.

The Need for State Estimate and Covariance Recovery

In Section 5.3.2 it was shown that state estimates are required in the information form prediction, observation and update operations, and in Section 5.3.3 it was shown that state covariances are required to generate loop-closure hypotheses. Efficient algorithms for state estimate and covariance recovery are critical for the performance of an information filter.

In a previous VAN implementation [27–29], state estimates and covariances were recovered using a Cholesky factorisation calculated each time an image was acquired, however the use of iterative methods was suggested. In this chapter, direct methods for efficient state estimate and covariance recovery will be further investigated through the use of Cholesky factorisation modifications. Using Cholesky factorisation modifications has been proposed in a similar information filtering problem, however was not implemented due to the complexity of the algorithms and the lack of an available implementation [17, 18].

5.4 The Cholesky Factorisation

The Cholesky factorisation is commonly used to solve linear systems of the form

$$\mathbf{AX} = \mathbf{B} \tag{5.23}$$

where $\mathbf{A}$ is a positive definite symmetric matrix and $\mathbf{X}$ is a matrix of unknowns.

In this SLAM application, the Cholesky factorisation of the information matrix will be used to recover state estimates and covariances by solving the relationships

$$\mathbf{Y}^{+}(t_k)\,\hat{\mathbf{x}}^{+}(t_k) = \hat{\mathbf{y}}^{+}(t_k) \tag{5.24}$$

$$\mathbf{Y}^{+}(t_k)\,\mathbf{P}^{+}(t_k) = \mathbf{I} \tag{5.25}$$

where $\mathbf{I}$ is an identity matrix of the same dimensions as the information matrix.

The LDL^{T} form of the Cholesky decomposition of the matrix $\mathbf{A}$ is defined by

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^{\mathsf{T}} \tag{5.26}$$

where the $\mathbf{L}$ is a lower triangular matrix with all elements on the diagonal equal to one, and $\mathbf{D}$ is a diagonal matrix.

The solution to a system of equations in the form of Equation 5.23 is calculated from the Cholesky factorisation using a two-step forwards and backward solve process. First, a forward solve step is performed on the lower triangular system

$$\mathbf{LZ} = \mathbf{B} \tag{5.27}$$

to recover the forward-solve result $\mathbf{Z}$. In the forward solve operation, the rows of $\mathbf{Z}$ are recovered in order from first to last. The solution $\mathbf{X}$ can then be recovered using a backward

Input:

- A matrix M containing the lower triangle of a symmetric positive definite matrix **A**.

Output:

- The elements of M are modified to contain the elements of **L** below the diagonal, and the elements of **D** on the diagonal, where **L** and **D** are as defined by the factorisation $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^{\mathsf{T}}$.

Procedure:

1: **for** $j = 1$ to n **do**
2: **for** $i = j + 1$ to n **do**
3: $\mathbf{M}_{ij} \leftarrow \mathbf{M}_{ij}\mathbf{M}_{jj}^{-1}$
4: **end for**
5: **for** $k = j + 1$ to n **do**
6: **for** $i = k$ to n **do**
7: $\mathbf{M}_{ik} \leftarrow \mathbf{M}_{ik} - \mathbf{M}_{ij}\mathbf{M}_{jj}\mathbf{M}_{kj}^{\mathsf{T}}$
8: **end for**
9: **end for**
10: **end for**

Algorithm 5.1: A dense, block-based right-looking Cholesky factorisation algorithm.

solve operation on the upper triangular system

$$\mathbf{D}\mathbf{L}^{\mathsf{T}}\mathbf{X} = \mathbf{Z} \tag{5.28}$$

in which the rows of **X** are recovered in order from last to first.

For the Cholesky decomposition of a sparse matrix, the structure of the factor **L** is related to the sparsity pattern of the original matrix **A**. Non-zero elements in the Cholesky factor are present at the locations of all non-zeros elements in the original matrix, however additional non-zeros known as 'fill-in' are introduced. Fill-in is undesirable, since additional non-zero elements increase the computational complexity of the factorisation and equation-solving processes.

Many algorithms to calculate the Cholesky decomposition exist [14]. The experiments presented in Section 5.9 use an efficient sparse up-looking algorithm [12], however the simpler right-looking approach of Algorithm 5.1 will be used here to demonstrate the process of fill-in. During each iteration of the algorithm, lines 2 and 3 produces the j'th column of the factorisation, and lines 5 to 9 update the remaining active subset (rows and columns $j + 1$ to n) of the matrix by eliminating the j'th variable.

Figure 5.3 illustrates each iteration of the right-looking algorithm during the factorisation the matrix of Figure 5.2. The fill-in produced in the Cholesky factor is equivalent to the

additional edges produced by marginalising a variable from the Markov graph (also known as the elimination graph in linear algebra).

5.4.1 Reducing Fill-in with Variable Reordering

Fill-in can be reduced by reordering the variables to change the sequence in which they are eliminated during the factorisation process. Since finding the optimal permutation that produces minimal fill-in is NP-hard, heuristic-based approaches are typically used [14, 56].

A popular ordering method is the Approximate Minimum Degree (AMD) algorithm [14, 56]. In each iteration of a right-looking Cholesky factorisation, the AMD algorithm employs the greedy strategy of selecting for elimination the variable corresponding to the graph node with the smallest degree (the number of neighbours) or equivalently the sparsest row of the remaining active submatrix to be factorised.

If the variable ordering is defined by a row-permutation matrix $\mathbf{O}$, the permuted system of linear equations has the form

$$\mathbf{OAO}^{\mathsf{T}}\,\mathbf{OX} = \mathbf{OB} \tag{5.29}$$

Defining $\mathbf{A}_O = \mathbf{OAO}^{\mathsf{T}}$, $\mathbf{X}_O = \mathbf{OX}$ and $\mathbf{B}_O = \mathbf{OB}$, the permuted system of equations becomes $\mathbf{A}_O\mathbf{X}_O = \mathbf{B}_O$, which is in the standard form of Equation 5.23 and can be solved using the standard forward and backward solve process.

Figure 5.4 illustrated the factorisation process for the matrix previously used in Figure 5.3 using the variable ordering produced by AMD. The selected order in which the poses are eliminated is $5, 4, 1, 2, 3$, corresponding to the row permutation matrix

$$\mathbf{O} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.30}$$

The benefit of variable ordering can be observed by comparing the three blocks of fill-in produced using the natural ordering in Figure 5.3 with the one block of fill-in when the AMD ordering is used in Figure 5.4.

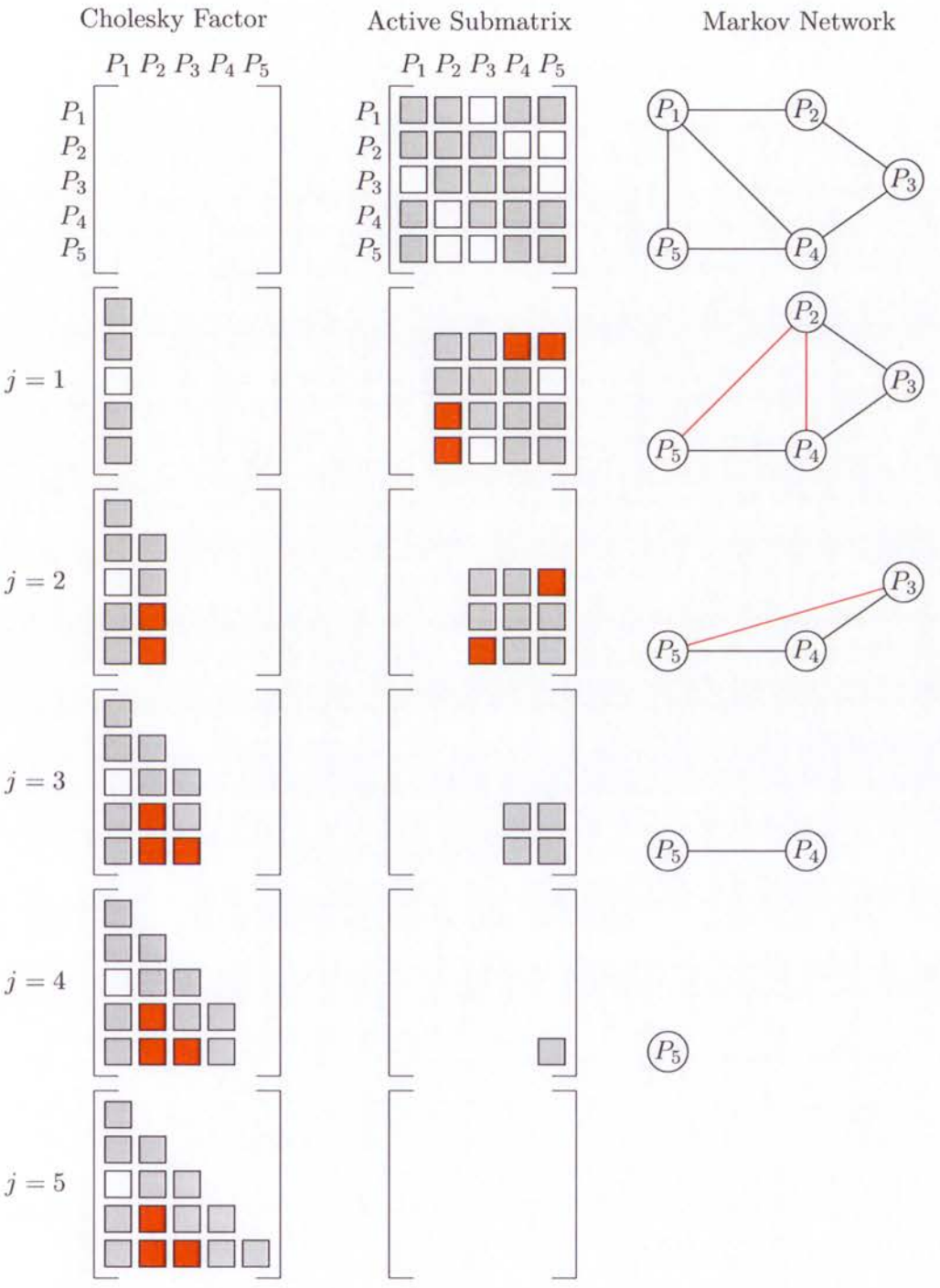


Figure 5.3: Cholesky factorisation using the right-looking algorithm and the natural variable ordering. During each iteration of the algorithm, the j'th column of the factor is produced by dividing the j'th column of the active submatrix by its element on the diagonal. The j'th variable is then eliminated (marginalised) from the active submatrix. Fill-in elements (non-zeros at the locations of zeros in the original matrix) are shown in red.

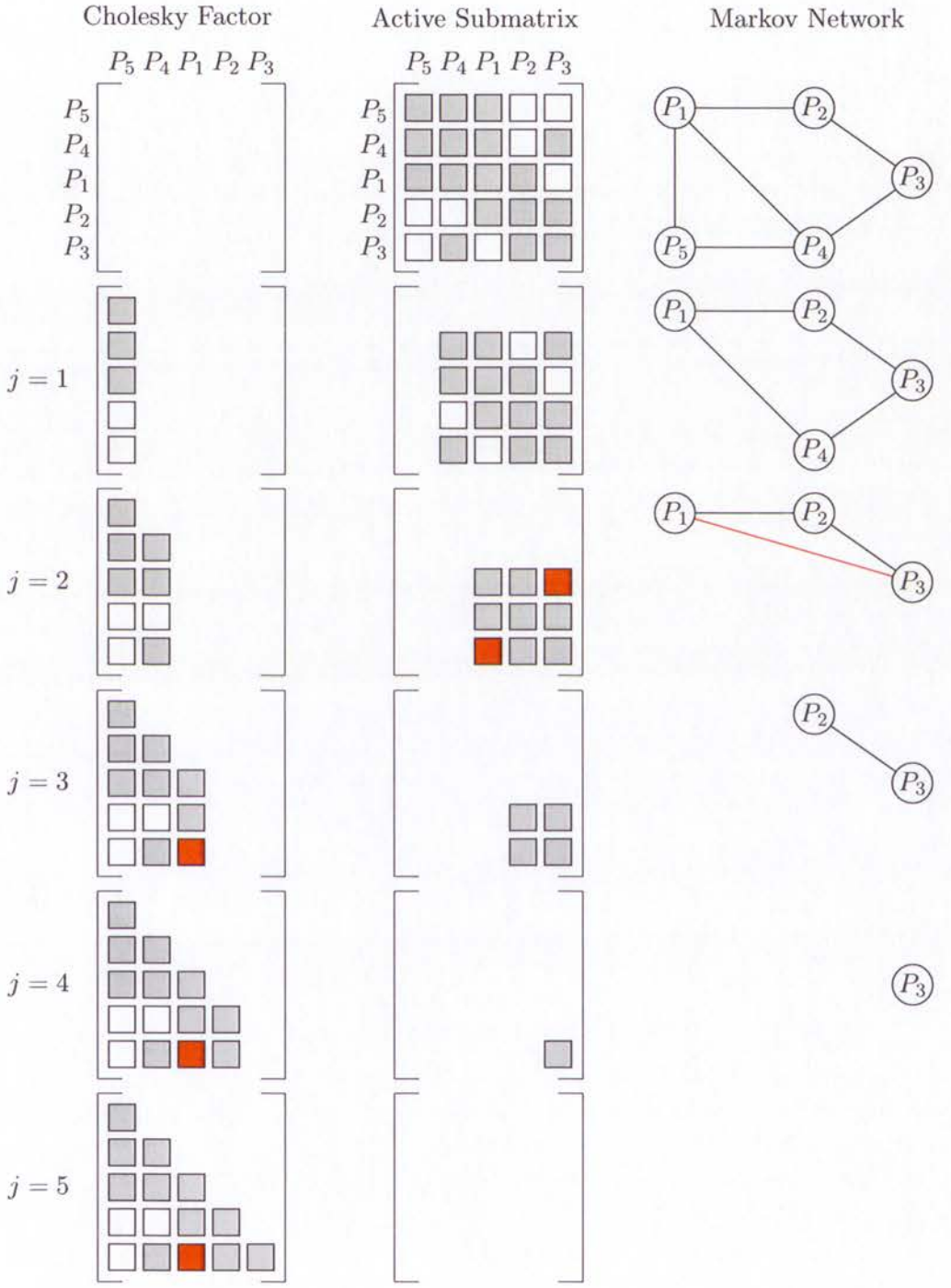


Figure 5.4: The Cholesky factorisation of the matrix of Figure 5.3 using a fill-in reducing ordering produced by the AMD algorithm. Only one block of fill-in is created compared to the 3 blocks produced with the natural ordering.

5.4.2 Scalability

The computational complexity of the Cholesky factorisation of a dense n by n matrix is $O(n^3)$. For sparse matrices however, the computational complexity is dependent on the number of non-zeros in the Cholesky factor, which is influenced by the structure of the matrix being factorised and the variable ordering

To investigate the scalability of the Cholesky factorisation when applied to VAN information matrices, the number of augmented poses and the number of loop-closure observations applied to the filter are assumed to grow linearly over time. Since the structure of the factorised matrix affects the complexity of the factorisation process, experiments have been performed on simulated pose networks created using three observation generation strategies:

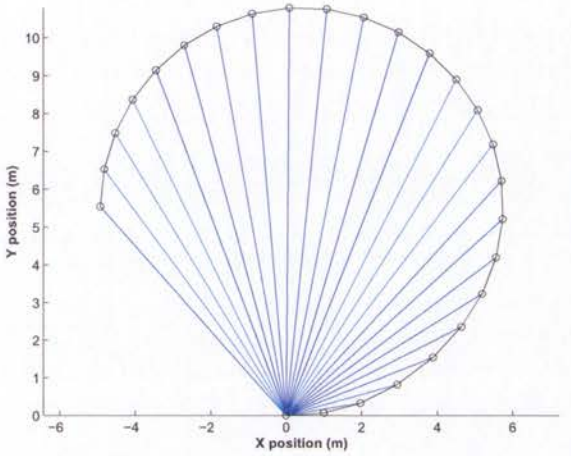
1. The current pose is observed relative to the *first* k poses
2. The current pose is observed relative to the *previous* k poses
3. The current pose is observed relative to k *random* poses

where k is an integer variable. Example pose networks for each of the observation generation strategies are shown in Figure 5.5.

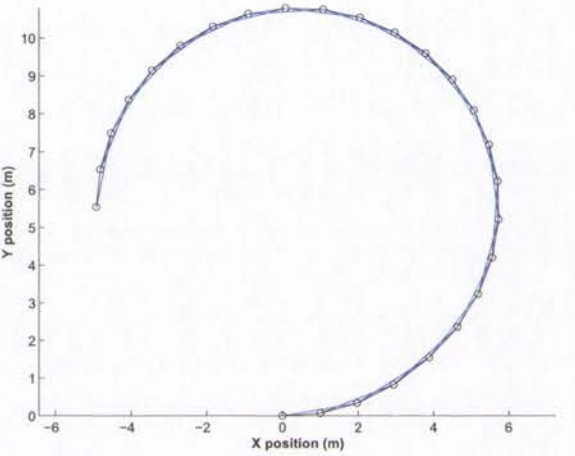
The benefit of variable ordering is demonstrated in Figure 5.6, which displays the growth in the number of non-zero elements in the Cholesky factor and complexity of the factorisation process in experiments where each pose was linked to the first k poses. When the natural ordering is used, elimination of the first pose in the first iteration of the factorisation process results in complete fill-in. The presence of $O(n^2)$ non-zero elements in the Cholesky factor causes the complexity of the factorisation process to be $O(n^3)$. Fill-in is easy to avoid in this case however, by ordering the first k poses last. When the AMD algorithm is used, the number of non-zeros in the Cholesky factor grows linearly, causing the complexity of the factorisation process to be $O(n)$.

The strategy of linking each pose to the previous k poses simulates the pose network that could be expected in applications such as visual odometry or range-finder scan-matching where new data is registered with recent observations. Such systems result in a band-diagonal information matrix. As shown in Figure 5.7, in this case the number of non-zeros in the Cholesky factor grows linearly with the number of augmented poses, and the complexity of the factorisation process is $O(n)$.

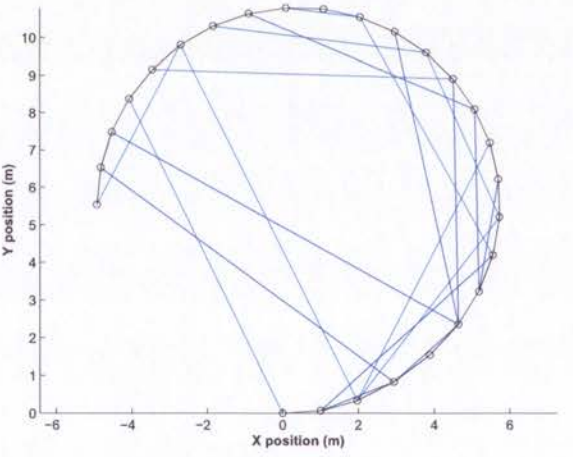
The random relative pose generating strategy is likely to be the best model for a typical SLAM implementation. In Figure 5.7, the AMD variable ordering significantly reduces the amount of fill-in, however $O(n^2)$ non-zero elements are still produced in the factorisation resulting in a complexity of $O(n^3)$.



(a) Each pose linked to the first k poses

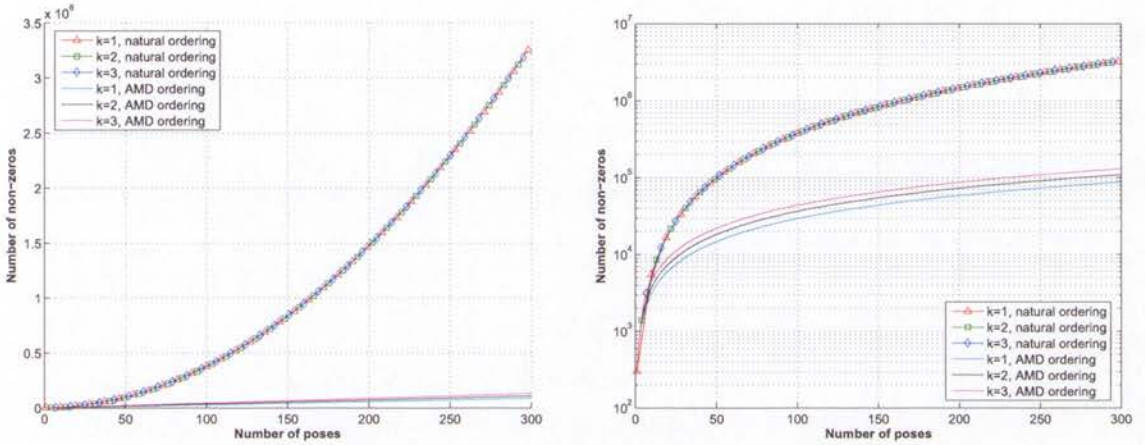


(b) Each pose linked to the previous k poses

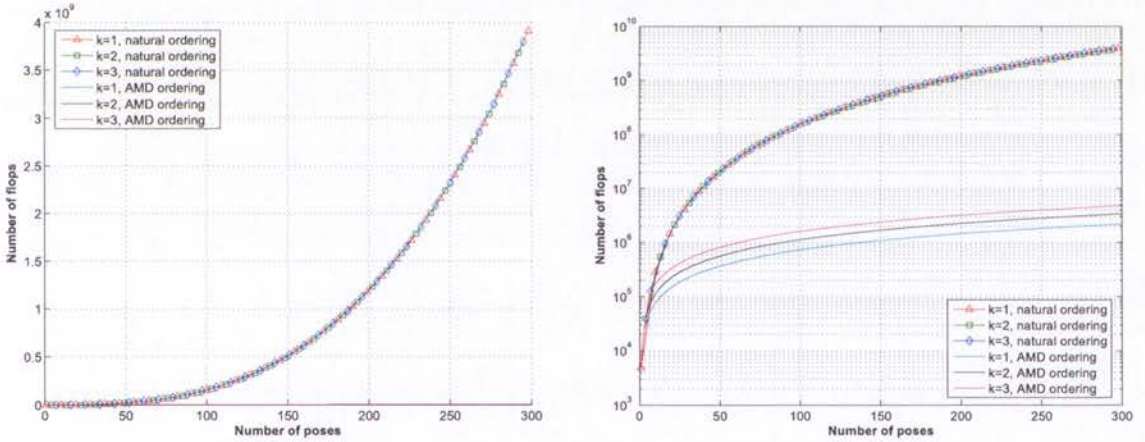


(c) Each pose linked to k random poses

Figure 5.5: Example pose networks for the Cholesky factorisation scalability experiments. Black circles mark the location of poses, and black and blue lines represent odometry and loop-closure relationships respectively. 25 poses are shown in each graph, with loop-closure observations produced by the (a) first (b) previous and (c) random linking strategies. In the examples shown, the location of each pose is constrained relative to one previous pose (corresponding to $k = 1$).

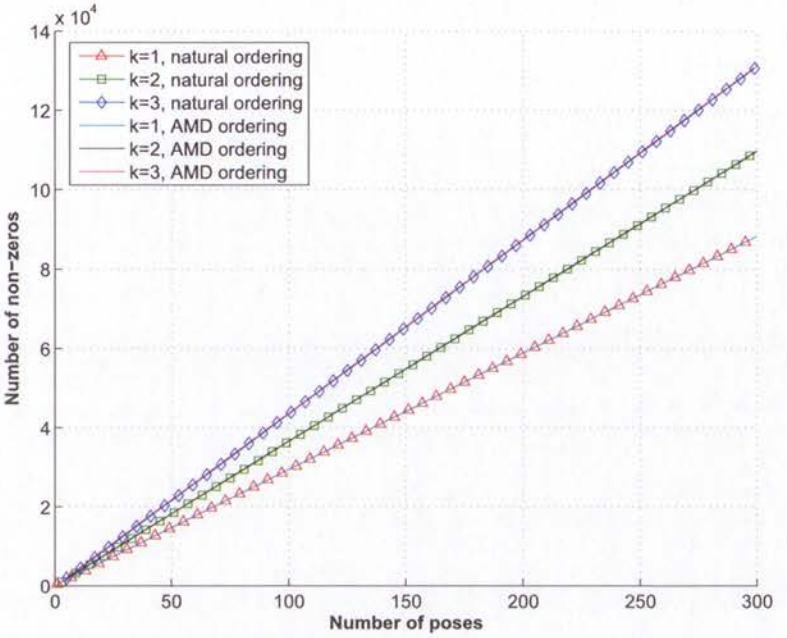


(a) Number of non-zeros elements in the Cholesky factor in linear (left) and log (right) scale

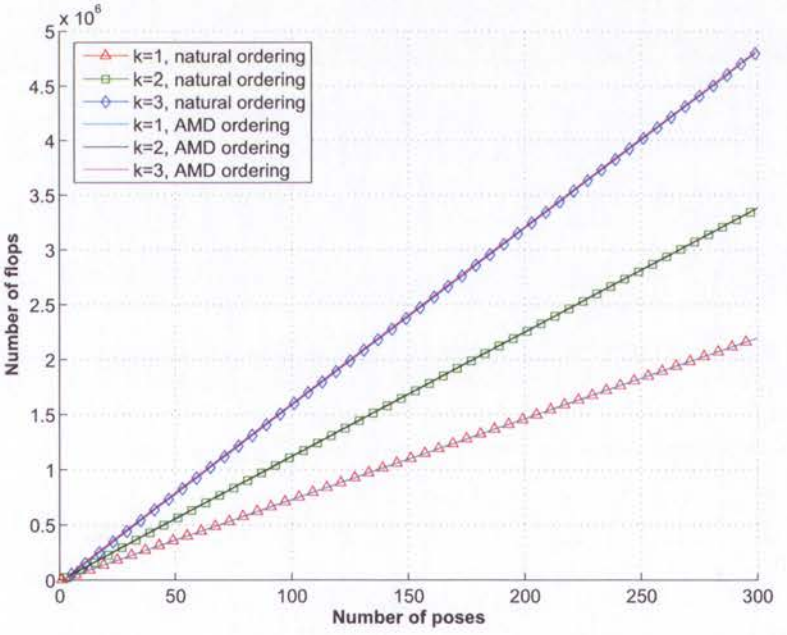


(b) Number of floating point operations required in the Cholesky factorisation process in linear (left) and log (right) scale

Figure 5.6: Cholesky factorisation scalability results when each new pose is linked to the first k poses. When the natural ordering is used, removing the first pose (to which all other poses are linked) causes complete fill-in, resulting in $O(n^2)$ non-zeros and $O(n^3)$ floating point operations. Fill-in can easily be avoided by eliminating the first k poses last, causing the number of non-zeros in the Cholesky factor to grow linearly, and the complexity of the factorisation process to be $O(n)$.

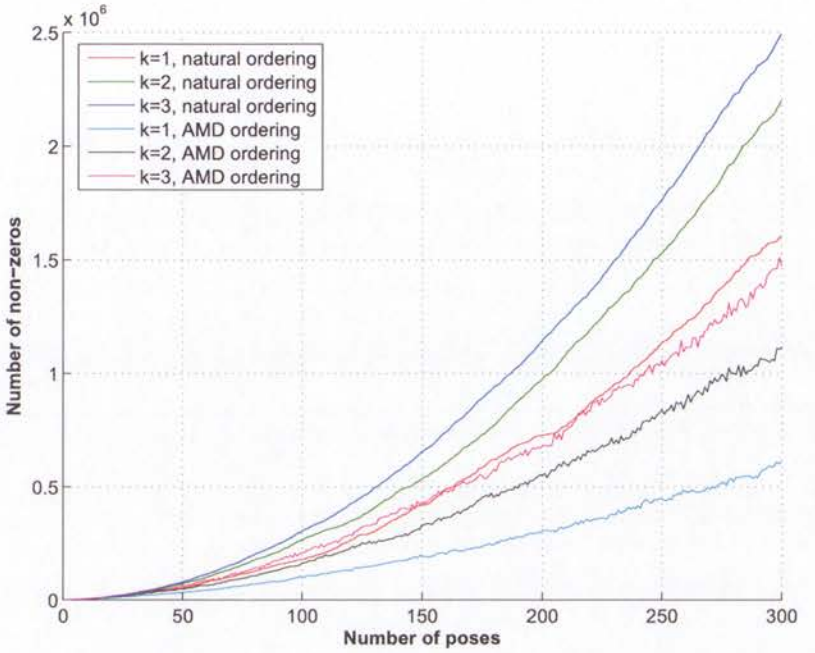


(a) Number of non-zeros elements in the Cholesky factor

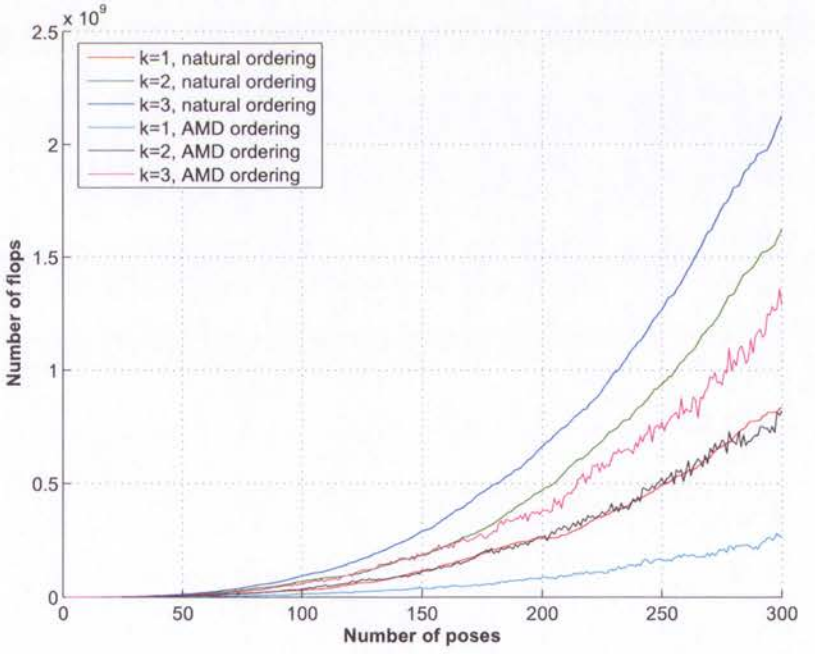


(b) Number of floating point operations required in the Cholesky factorisation process

Figure 5.7: Cholesky factorisation scalability results when each new pose is linked to the k most recent previous poses (except the previous pose which for which an odometry relationship already exists). The AMD algorithm provides no benefit in this case since the natural ordering is optimal. The number of non-zeros in the Cholesky factor grows linearly, causing the complexity of the factorisation process to be $O(n)$.



(a) Number of non-zeros elements in the Cholesky factor



(b) Number of floating point operations required in the Cholesky factorisation process

Figure 5.8: Cholesky factorisation scalability results when each new pose is linked to k randomly selected previous poses. Fill-in is reduced by the AMD algorithm, however $O(n^2)$ non-zeros are still produced in the Cholesky factor. The computational complexity of the factorisation process is therefore $O(n^3)$.

5.4.3 Factor Modification

If a previously factorised system of equations is changed, it is often possible to efficiently modify the existing factor instead of repeating the computationally expensive factorisation process. In this section, the complex equations and algorithms used to compute modified components of a sparse Cholesky factorisation will not be presented. Instead, the focus will be on illustrating which components of the factor change, and the resulting computational complexity of the operation. Further details on the Cholesky modification algorithms can be found in [15, 16], and the implementation used in the experiments presented in the chapter is described in [12].

Row Addition

In a VAN filter, new rows are added to the information vector and information matrix in the prediction with augmentation operation of Equations 5.9 and 5.10.

Suppose a new variable is to be added to the state vector at index k. The original factorised matrix partitioned at row and column k has the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{13} \\ \mathbf{A}_{13}^{\mathsf{T}} & \mathbf{A}_{33} \end{bmatrix} \tag{5.31}$$

with factorisation

$$\mathbf{L}\mathbf{D}\mathbf{L}^{\mathsf{T}} = \begin{bmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{31} & \mathbf{L}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11} & \\ & \mathbf{D}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \mathbf{L}_{31}^{\mathsf{T}} \\ & \mathbf{L}_{33}^{\mathsf{T}} \end{bmatrix} \tag{5.32}$$

where $\mathbf{L}_{11}$ and $\mathbf{L}_{33}$ are lower triangular matrices, and $\mathbf{D}_{11}$ and $\mathbf{D}_{33}$ are diagonal matrices.

If the new column of $\mathbf{A}$ has the form

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}_1 \\ r_2 \\ \mathbf{r}_3 \end{bmatrix} \tag{5.33}$$

the modified matrix is

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{r}_1 & \mathbf{A}_{13} \\ \mathbf{r}_1^{\mathsf{T}} & r_2 & \mathbf{r}_3^{\mathsf{T}} \\ \mathbf{A}_{13}^{\mathsf{T}} & \mathbf{r}_3 & \mathbf{A}_{33} \end{bmatrix} \tag{5.34}$$

with factorisation

$$\bar{\mathbf{L}}\bar{\mathbf{D}}\bar{\mathbf{L}}^{\mathsf{T}} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \bar{\mathbf{l}}_{21} & 1 & \\ \mathbf{L}_{31} & \bar{\mathbf{l}}_{32} & \bar{\mathbf{L}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11} & & \\ & \bar{d}_{22} & \\ & & \bar{\mathbf{D}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \bar{\mathbf{l}}_{21}^{\mathsf{T}} & \bar{\mathbf{L}}_{31}^{\mathsf{T}} \\ & 1 & \bar{\mathbf{l}}_{32}^{\mathsf{T}} \\ & & \bar{\mathbf{L}}_{33}^{\mathsf{T}} \end{bmatrix} \tag{5.35}$$

where $\bar{\mathbf{l}}_{21}$ is a row vector and $\bar{\mathbf{l}}_{32}$ a column vector, and the overhead bar has been used to identify modified matrix and factor components.

Equations and algorithms for the modified factor components can be found in [15]. The key result of Equation 5.35 is that $\mathbf{L}_{11}$ and $\mathbf{L}_{31}$ are unchanged, and only the elements of the factor in columns $k+1$ to n are modified.

Row Deletion

The row deletion operation is the opposite of row addition. A two step process of row deletion and addition can be used to perform an arbitrary change to a row and column of the factorised matrix.

Starting with a matrix with the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{a}_{12} & \mathbf{A}_{13} \\ \mathbf{a}_{12}^{\mathsf{T}} & a_{22} & \mathbf{a}_{23} \\ \mathbf{A}_{13}^{\mathsf{T}} & \mathbf{a}_{23}^{\mathsf{T}} & \mathbf{A}_{33} \end{bmatrix} \tag{5.36}$$

with factorisation

$$\mathbf{L}\mathbf{D}\mathbf{L}^{\mathsf{T}} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{l}_{21} & 1 & \\ \mathbf{L}_{31} & \mathbf{l}_{32} & \mathbf{L}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11} & & \\ & d_{22} & \\ & & \mathbf{D}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \mathbf{l}_{21}^{\mathsf{T}} & \mathbf{L}_{31}^{\mathsf{T}} \\ & 1 & \mathbf{l}_{32}^{\mathsf{T}} \\ & & \mathbf{L}_{33}^{\mathsf{T}} \end{bmatrix} \tag{5.37}$$

removing the row and column corresponding to block 2, results in the modified matrix

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{13} \\ \mathbf{A}_{13}^{\mathsf{T}} & \mathbf{A}_{33} \end{bmatrix} \tag{5.38}$$

with factorisation

$$\bar{\mathbf{L}}\bar{\mathbf{D}}\bar{\mathbf{L}}^{\mathsf{T}} = \begin{bmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{31} & \bar{\mathbf{L}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11} & \\ & \bar{\mathbf{D}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \mathbf{L}_{31}^{\mathsf{T}} \\ & \bar{\mathbf{L}}_{33}^{\mathsf{T}} \end{bmatrix} \tag{5.39}$$

As was the case in the row addition operation, only the columns of the factorisation to the right of the column corresponding to the removed variable need to be modified.

Update and Downdate

A modification of the factorised matrix of the form

$$\bar{\mathbf{A}} = \mathbf{A} + \mathbf{W}\mathbf{W}^{\mathsf{T}} \tag{5.40}$$

where $\mathbf{W}$ is an n by k matrix, is known as a rank-k update. Similarly, a modification of the form

$$\bar{\mathbf{A}} = \mathbf{A} - \mathbf{W}\mathbf{W}^{\mathsf{T}} \tag{5.41}$$

is known as a rank-k downdate.

In a VAN filter, update modifications of the information matrix occur during the prediction operation of Equation 5.12, and downdate modifications occur in the observation update of Equation 5.18.

If the update or downdate modification matrix $\mathbf{W}$ has the form

$$\mathbf{W} = \begin{bmatrix} \mathbf{0} \\ \mathbf{W}_2 \\ \mathbf{0} \end{bmatrix} \tag{5.42}$$

the modified matrix is

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ \mathbf{A}_{12}^{\mathsf{T}} & \mathbf{A}_{22} \pm \mathbf{W}_2\mathbf{W}_2^{\mathsf{T}} & \mathbf{A}_{23} \\ \mathbf{A}_{13}^{\mathsf{T}} & \mathbf{A}_{23}^{\mathsf{T}} & \mathbf{A}_{33} \end{bmatrix} \tag{5.43}$$

with factorisation

$$\bar{\mathbf{L}}\bar{\mathbf{D}}\bar{\mathbf{L}}^{\mathsf{T}} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{L}_{12} & \bar{\mathbf{L}}_{22} & \\ \mathbf{L}_{13} & \bar{\mathbf{L}}_{23} & \bar{\mathbf{L}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{D}_{11} & & \\ & \bar{\mathbf{D}}_{22} & \\ & & \bar{\mathbf{D}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \mathbf{L}_{12}^{\mathsf{T}} & \mathbf{L}_{13}^{\mathsf{T}} \\ & \bar{\mathbf{L}}_{22}^{\mathsf{T}} & \bar{\mathbf{L}}_{23}^{\mathsf{T}} \\ & & \bar{\mathbf{L}}_{33}^{\mathsf{T}} \end{bmatrix} \tag{5.44}$$

As was the case for the row addition and row deletion operations, all components of the factor to the left of the modified columns in Equation 5.44 are unchanged, while those to the right need to be recalculated. This result is logical if the Cholesky factorisation method presented in Section 5.4 is considered. Each variable corresponding to the columns in order from left to right is marginalised from the remaining system to produce the lower triangular factor. Therefore, if any column of the original matrix $\mathbf{A}$ is modified, the columns to the right need to be updated to reflect the changes to the marginalised variable.

5.4.4 Maintaining a Cholesky Factor of the VAN Information Matrix

The information-form VAN filter operations of Section 5.3.2 can all be described using the row addition, row deletion, update and downdate modifications.

The prediction with augmentation operation in Equations (5.9) and (5.10) can be implemented with row additions for the new pose variables, and an update on the previous pose states. Similarly, the prediction without augmentation operation in Equations (5.11) and (5.12) can be implemented with row removal and row addition modifications to perform the changes to the current vehicle pose states, and a downdate on the previous pose states. The observation update in Equations (5.17) and (5.18) can be implemented with a single update modification.

In the VAN algorithm presented in this chapter, modifications are used to maintain an up-to-date factor after prediction operations and observations of the current vehicle states. However, when a loop-closure observations is applied between past poses, the structure of the information matrix is significantly changed, causing the previous variable ordering to be ineffective in minimising fill-in. Therefore when a loop-closure observation is applied to the filter, a new variable ordering is found and a new factorisation of the information matrix is acquired.

5.4.5 Constraining the Variable Ordering for Efficient VAN Operations

Prediction and vehicle state observations are the most common operations in a SLAM filter (the number of loop-closure observations is relatively small). After considering the modified factorisations of Equations 5.35, 5.39 and 5.44, it is obvious that ordering the vehicle states last will minimise the amount of work required to perform the factorisation modification operations. If the current vehicle states are ordered last, the number of elements in the factor that need to be recalculated is constant (independent of the number of augmented poses), allowing the Cholesky factor modifications for the prediction and vehicle state observation operations to be performed in constant time.

While ordering the vehicle states last may not result in the the minimal amount of fill-in, the benefit of constant-time prediction and observation operations outweigh the additional computational complexity due to the additional fill-in caused by this constraint.

5.5 State Estimate Recovery

In this section, two methods previously used to recover state estimates from an information-form VAN filter will be briefly summarised. An alternative approach to recover part of the state vector using an updated Cholesky factor of the information matrix will be presented.

5.5.1 Complete State Recovery

The complete state vector estimate can be recovered by solving the relationship

$$\mathbf{Y}^{+}(t_k)\,\hat{\mathbf{x}}^{+}(t_k) = \hat{\mathbf{y}}^{+}(t_k) \tag{5.45}$$

using the Cholesky factorisation of the information matrix and the process described in Section 5.4.

The efficiency of the forwards and backward solve process used to solve Equation 5.45 is dependent on the sparsity of the Cholesky factor $\mathbf{L}$. If the Cholesky factor contains $O(n)$ non-zero elements, as is the case for VAN systems with only odometry constraints or a constant number of loop-closure observations, the complete state estimate vector can be recovered in $O(n)$ time. However in general where the Cholesky factor contains $O(n^2)$ non-zeros, as can be expected in SLAM applications where the number of loop-closure observations grows linearly with the number of poses, the complexity of calculating the complete estimate vector is $O(n^2)$.

5.5.2 Approximate Partial State Recovery

In a previous VAN implementation [27, 28], approximate estimates of the current vehicle states were produced by partitioning the state vector into a 'local' portion consisting of the states to be recovered, and the remaining 'benign' states for which an approximate estimate is available. Using the subscript l for the local subvector and b for the benign states, the partitioned version of Equation 5.45 is

$$\begin{bmatrix} \mathbf{Y}_{bb}^{+}(t_k) & \mathbf{Y}_{bl}^{+}(t_k) \\ \mathbf{Y}_{bl}^{+\mathsf{T}}(t_k) & \mathbf{Y}_{ll}^{+}(t_k) \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_b^{+}(t_k) \\ \hat{\mathbf{x}}_l^{+}(t_k) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{y}}_b^{+}(t_k) \\ \hat{\mathbf{y}}_l^{+}(t_k) \end{bmatrix} \tag{5.46}$$

If the benign states have not changed significantly since they were last recovered, providing a good approximation $\tilde{\mathbf{x}}_b(t_k)$ (a tilde will be used to denote all approximate estimates), an

approximate estimate of the local states can be calculated with

$$\tilde{\mathbf{x}}_l(t_k) = \left[\mathbf{Y}_{ll}^{+}(t_k)\right]^{-1}\left(\hat{\mathbf{y}}_l^{+}(t_k) - \mathbf{Y}_{bl}^{+\mathsf{T}}(t_k)\,\tilde{\mathbf{x}}_b(t_k)\right) \tag{5.47}$$

Assuming the current vehicle pose is linked only to the previous pose (which can be enforced by performing vehicle state observations before any loop-closure observations), only one block of $\mathbf{Y}_{bl}^{+\mathsf{T}}(t_k)$ corresponding to the previous-to-current pose cross-information submatrix contains non-zero elements. The approximate vehicle state estimate can therefore be recovered in constant time.

The assumption underlying this approximation is that the past vehicle poses have not been significantly updated by observations applied to the filter since the estimates of the benign states were last recovered. For high-frequency observations from accurate devices such as depth sensors, attitude sensors or a DVL, the estimates of the observed states can not accumulate a significant error. Therefore the updates provided by such observations are small and the approximation provides accurate estimates. However, loop-closures or observations from external positioning systems (GPS, LBL etc.) that correct large errors in drifting estimates cause large updates to the estimated trajectory states. After such an observation has been applied to the filter, the accuracy of the approximation will be poor, and the complete state vector including new estimates of the benign states should be recovered using the method of Section 5.5.1.

5.5.3 Exact Partial State Recovery

In Section 5.4.3 it was shown that a Cholesky factorisation can be efficiently modified to reflect changes in the factorised matrix. If the Cholesky factorisation is used to find the solution to a system of linear equations, the result of the forward solve step (the solution $\mathbf{Z}$ of Equation 5.27) can also be efficiently updated to reflect changes in the factorised matrix $\mathbf{A}$ and right-hand-side matrix $\mathbf{B}$.

Following the convention of Section 5.4, the system of linear equations will be presented partitioned into three blocks. If the modification is performed to the variables ordered last, the modified lower-triangular system $\bar{\mathbf{L}}\bar{\mathbf{Z}} = \bar{\mathbf{B}}$ in the forward solve step has the form

$$\begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{L}_{21} & \mathbf{L}_{22} & \\ \mathbf{L}_{31} & \mathbf{L}_{32} & \bar{\mathbf{L}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \\ \bar{\mathbf{Z}}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \bar{\mathbf{B}}_3 \end{bmatrix} \tag{5.48}$$

Since only the last block of the forward solve result is changed, the updated $\bar{\mathbf{Z}}$ can be calculated in constant time. In the backward solve step, the upper-triangular system $\bar{\mathbf{D}}\bar{\mathbf{L}}^{\mathsf{T}}\bar{\mathbf{X}} = \bar{\mathbf{Z}}$

has the form

$$\begin{bmatrix} \mathbf{D}_{11} & & \\ & \mathbf{D}_{22} & \\ & & \bar{\mathbf{D}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathsf{T}} & \mathbf{L}_{21}^{\mathsf{T}} & \mathbf{L}_{31}^{\mathsf{T}} \\ & \mathbf{L}_{22}^{\mathsf{T}} & \mathbf{L}_{32}^{\mathsf{T}} \\ & & \bar{\mathbf{L}}_{33}^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} \bar{\mathbf{X}}_1 \\ \bar{\mathbf{X}}_2 \\ \bar{\mathbf{X}}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \\ \bar{\mathbf{Z}}_3 \end{bmatrix} \tag{5.49}$$

While all elements of the solution $\bar{\mathbf{X}}$ are changed, the backward solve operation recovers the variables in reverse order from last to first row. The last block of $\bar{\mathbf{X}}$ can therefore be calculated in constant time by solving

$$\bar{\mathbf{D}}_{33}\bar{\mathbf{L}}_{33}^{\mathsf{T}}\bar{\mathbf{X}}_3 = \bar{\mathbf{Z}}_3 \tag{5.50}$$

If the current vehicle pose variables are ordered last and the forwards substitution result is updated along with the Cholesky factorisation each time it is modified during a prediction or observation step, this approach allows the current vehicle state estimates to be recovered in constant time. This is an important improvement over the method of Section 5.5.2, since it allows prediction and observation operations to be performed without corrupting the filter with approximate estimates. As a result the EIF will have the same optimality properties as an EKF solution.

5.6 Covariance Recovery

In this section, two previously used methods to recover covariances from an information-form VAN filter will be briefly summarised. An alternative approach to recover the previous pose submatrices on the block diagonal of the covariance matrix will be presented.

5.6.1 Complete Inverse Recovery

The complete covariance matrix can be recovered by solving the equation

$$\mathbf{Y}^{+}(t_k)\,\mathbf{P}^{+}(t_k) = \mathbf{I} \tag{5.51}$$

using the Cholesky decomposition of the information matrix. While an information matrix may be sparse, the corresponding covariance matrix will be dense, resulting in long computation times and large storage requirements.

For VAN systems that result in a Cholesky factor with $O(n)$ non-zero elements, the complete covariance matrix can be recovered in $O(n^2)$ time. However in general where the Cholesky factor contains $O(n^2)$ non-zeros, the complexity of recovering the complete covariance matrix is $O(n^3)$.

Recovering the complete covariance matrix is only feasible for problems with small state vectors. However, in a VAN application, the complete covariance matrix is never required. Sections 5.6.2 and 5.6.3 present methods to recover components of the covariance matrix.

5.6.2 Recovery of Columns of the Inverse

The j'th column of the covariance matrix can be recovered by solving the equation

$$\mathbf{Y}^{+}(t_k)\,\mathbf{P}^{+}_{*j}(t_k) = \mathbf{I}_{*j} \tag{5.52}$$

where $\mathbf{P}^{+}_{*j}(t_k)$ is the j'th column of the covariance matrix, and $\mathbf{I}_{*j}$ is the j'th column of an identity matrix with the same dimensions as the information matrix.

For VAN systems that result in a Cholesky factor with $O(n)$ non-zero elements, a column of the covariance matrix can be recovered in $O(n)$ time. However in general where the Cholesky factor contains $O(n^2)$ non-zeros, the complexity of recovering a column is $O(n^2)$.

In a previous VAN implementation [27, 28], this approach has been used to recover the current vehicle pose covariance and past pose to current pose cross-covariances that are required to assemble the joint-covariance matrices used to find loop-closure hypothesis pairs.

5.6.3 Recovery of the Sparse Inverse

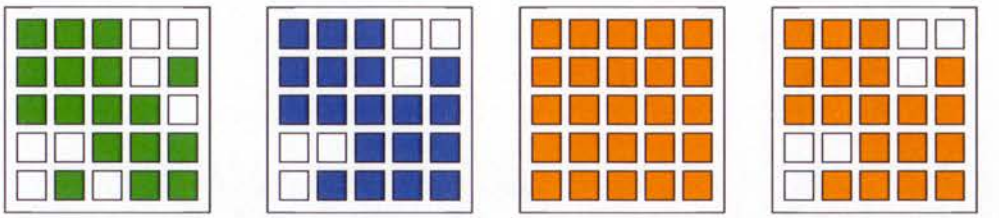
In addition to the current vehicle pose covariance and previous-to-current pose cross-covariances, creating the joint pose distributions used for loop-closure hypothesis generation requires the covariance of the augmented poses, which are located on the block-diagonal of the covariance matrix. The covariance recovery methods of Sections 5.6.1 and 5.6.2 are inefficient for this task, since they result in the calculation of many irrelevant elements of the inverse.

An alternative recovery method [25, 80, 106] can be derived from the Cholesky decomposition $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^{\mathsf{T}}$ using the Takahashi relationship (described in more detail in Appendix D)

$$\mathbf{A}^{-1} = (\mathbf{L}^{\mathsf{T}})^{-1}\mathbf{D}^{-1} - \mathbf{A}^{-1}(\mathbf{L} - \mathbf{I}) \tag{5.53}$$

If Equation 5.53 is used to calculate the lower triangle of the inverse, the upper triangular component $(\mathbf{L}^{\mathsf{T}})^{-1}$ which contains ones on its diagonal can be ignored. Individual elements of the inverse can therefore be calculated using the recursive relationship

$$[\mathbf{A}^{-1}]_{ij} = [\mathbf{D}^{-1}]_{ij} - \sum_{k=j+1}^{n} [\mathbf{A}^{-1}]_{ik}\,\mathbf{L}_{kj} \quad \text{for } i \leq j \tag{5.54}$$



(a) Original Matrix (b) Factor ($\mathbf{L}+\mathbf{L}^\mathsf{T}$) (c) Inverse (d) Sparse Inverse

Figure 5.9: The sparse inverse matrix. In general, the inverse of a sparse matrix is dense. The sparse inverse matrix contains the elements of the inverse at the locations of non-zeros in the Cholesky factor. The sparse inverse may contain more non-zero elements than the original matrix due to fill-in.

Equation 5.54 describes an element of the inverse in column j in terms of other elements of the inverse in columns j to n, along with the Cholesky factorisation components $\mathbf{L}$ and $\mathbf{D}$. If the matrices $\mathbf{A}$ and $\mathbf{L}$ are sparse, not all elements of the inverse need to be recovered. The set of elements of the inverse at the locations of non-zeros in the Cholesky factor is known as the 'sparse inverse', which is illustrated in Figure 5.9. All elements of the sparse inverse can be calculated using only other members of the sparse inverse and the factorisation components [25]. A procedure to calculate the lower triangle of the sparse inverse matrix is presented in Algorithm 5.2.

When applied to the factorisation of a VAN information matrix, the sparse inverse includes the block-diagonal, providing a method to recover the augmented pose covariances. For systems resulting in a Cholesky factor with $O(n)$ non-zero elements, the sparse inverse can be recovered in $O(n)$ time. However in general where the factor contains $O(n^2)$ non-zeros, the complexity of recovering the sparse inverse is $O(n^3)$.

Input:

- A lower triangular matrix $\mathbf{L}$ and diagonal matrix $\mathbf{D}$ corresponding to the Cholesky factorisation of a matrix $\mathbf{A}$.

Output:

- A sparse lower triangular matrix $\mathbf{Z}$ containing elements of the inverse of $\mathbf{A}$ at the locations of all non-zeros in $\mathbf{L}$.

Procedure:

```
for j = n to 1 do
  for i = n to j do
    if L_ij ≠ 0 then
      Z_ij ← [D^-1]_ij
      for k = j + 1 to i do
        if L_kj ≠ 0 then
          Z_ij ← Z_ij − Z_ik L_kj
        end if
      end for
      for k = i + 1 to n do
        if L_kj ≠ 0 then
          Z_ij ← Z_ij − Z_ki L_kj
        end if
      end for
    end if
  end for
end for
```

Algorithm 5.2: Calculation of the lower triangle of the sparse inverse matrix, which consists of all elements of the inverse at the locations of non-zeros in the Cholesky factor $\mathbf{L}$. During each iteration of the algorithm, the elements of the sparse inverse in column j are calculated using the previously calculated elements in columns $j+1$ to n.

5.7 Conservative Covariances for Loop-closure Hypotheses

Since recovering covariances from an information filter is computationally expensive, a previous VAN implementation [27–29] has used covariances recovered at previous timesteps to generate loop-closure hypotheses. Since the uncertainty of augmented past poses can only decrease, the use of old covariances is a conservative strategy. The filter is not corrupted in any way, since no approximate values are used in any prediction or observation operation. The use of conservative covariances will however increase the number of loop-closure hypotheses generated.

The conservative pose covariances can be used to create an approximation of the predicted joint distribution covariance of the form

$$\tilde{\mathbf{P}}_{(i,v)}(t_k) = \begin{bmatrix} \tilde{\mathbf{P}}_{ii}(t_k) & \mathbf{P}^{-}_{iv}(t_k) \\ \mathbf{P}^{-\mathsf{T}}_{iv}(t_k) & \mathbf{P}^{-}_{vv}(t_k) \end{bmatrix} \tag{5.55}$$

where $\tilde{\mathbf{P}}_{ii}(t_k)$ is the conservative covariance of pose i, and $\mathbf{P}^{-}_{iv}(t_k)$ and $\mathbf{P}^{-}_{vv}(t_k)$ are the optimal past-to-current cross-covariance and current pose covariances, which can be recovered from the vehicle columns of the covariance matrix using the method of Section 5.6.2.

To maintain the set of approximate covariances, the (presently optimal) covariance of the current vehicle pose is added each time a new pose is augmented to the state vector. When an observation such as a loop-closure that will significantly change the past pose distributions is applied to the filter, the approximate covariances should be updated.

5.7.1 Updating the Conservative Covariance of a Single Pose

In previous VAN applications [27, 29], each time a loop-closure observation is applied to the filter, an EKF update (as described in Section 3.2.2) is performed on the approximate joint distribution covariance to yield an updated covariance for the past pose.

Since all of the estimated poses are correlated, a loop-closure observation will reduce the uncertainty of all trajectory states. This approach however only reduces the uncertainty in one of the maintained pose covariances, leaving the others highly conservative.

5.7.2 Updating the Conservative Covariance of all Poses

The sparse inverse recovery method of Section 5.6.3 provides a method to efficiently update all the augmented pose covariances. While this operation is more computationally complex than the single pose EKF update, the reduction in the conservative pose uncertainties will

cause fewer loop-closure hypotheses to be analysed. In the experiment performed in Section 5.9, recovering the covariances of all poses using the sparse inverse method is shown to result in an overall reduction in processing time.

5.8 Overview of the Algorithm

The SLAM algorithm presented in this chapter maintains the following variables:

- The information vector $\hat{\mathbf{y}}^{+}(t_k)$ and information matrix $\mathbf{Y}^{+}(t_k)$.
- The $\mathbf{LDL}^{\mathsf{T}}$ Cholesky factorisation of the information matrix and the forward solve result.
- A set of conservative covariances $[\tilde{\mathbf{P}}_1(t_k), \tilde{\mathbf{P}}_2(t_k), \ldots, \tilde{\mathbf{P}}_n(t_k)]$ for the augmented vehicle poses.
- A flag specifying if the state vector should be augmented with the current pose on the next prediction operation.

The algorithm consists of the standard EIF three-step predict, observe and update process.

The procedure performed in a prediction operation is illustrated in Figure 5.10. If the augmentation flag is set, the current vehicle covariance is added to the set of approximate pose covariances. The information vector and information matrix are updated and the Cholesky factorisation and forward solve results are modified, all of which can be performed in constant time.

The procedure to perform a vehicle state observation and update is illustrated in Figure 5.11. In the experiments performed in Section 5.9, this operation applies to observations of the current depth, attitude and velocity of the vehicle. All components of the operation can be performed in constant time.

The most computationally expensive task in the VAN framework is processing data that has the potential to create a loop-closure observation. This procedure is illustrated in Figure 5.12, where computational complexity figures have been listed assuming the worst-case scenario where the growth of the number of non-zero elements in the Cholesky factor is $O(n^2)$ in the number of augmented poses. Under this assumption, which is likely to occur when the number of loop closures grows linearly with the number of poses, the complexity of identifying loop closure hypotheses is $O(n^2)$, and applying a loop-closure observation is $O(n^3)$.

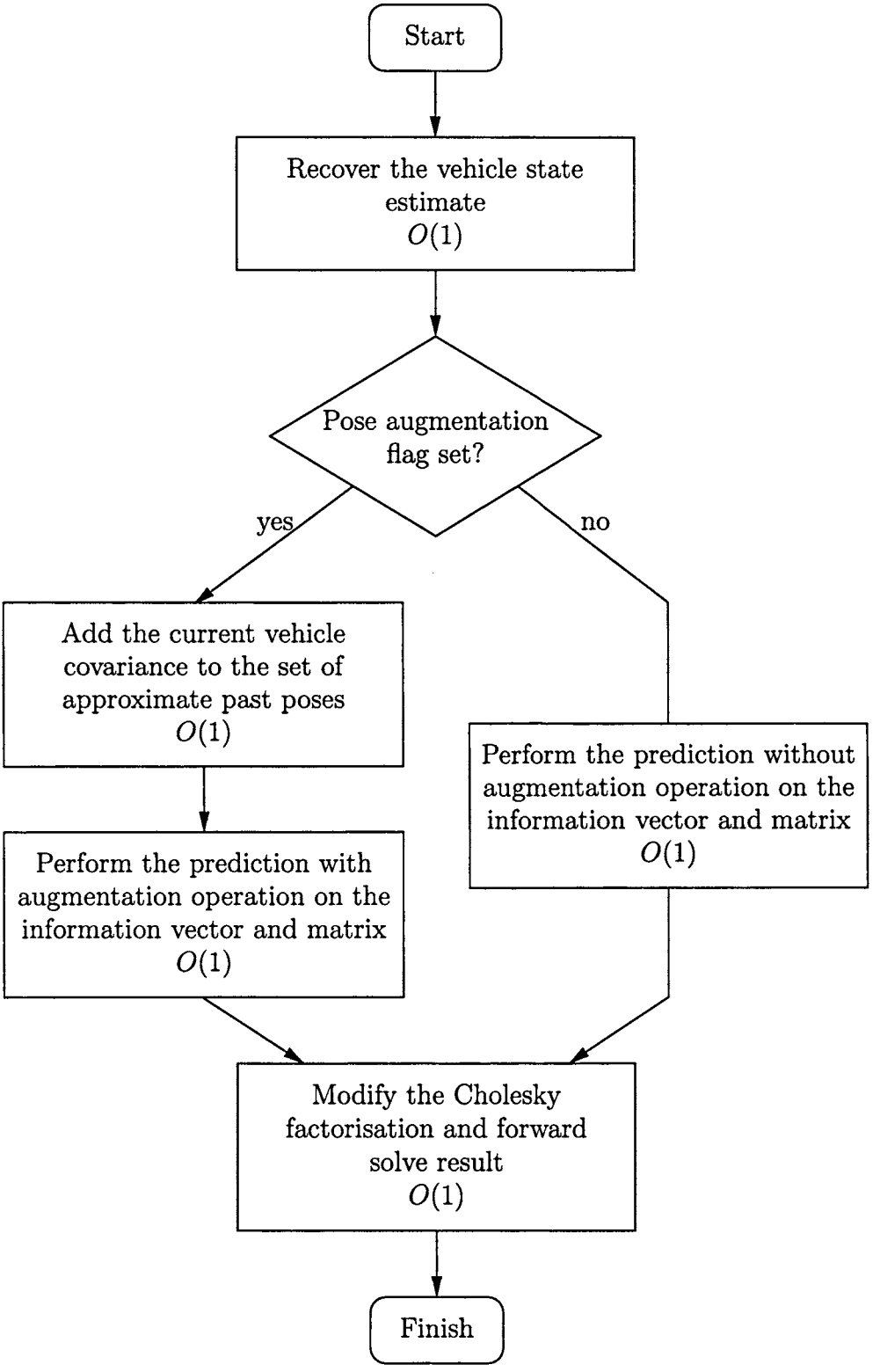


Figure 5.10: The VAN prediction process. The operation is performed in constant time.

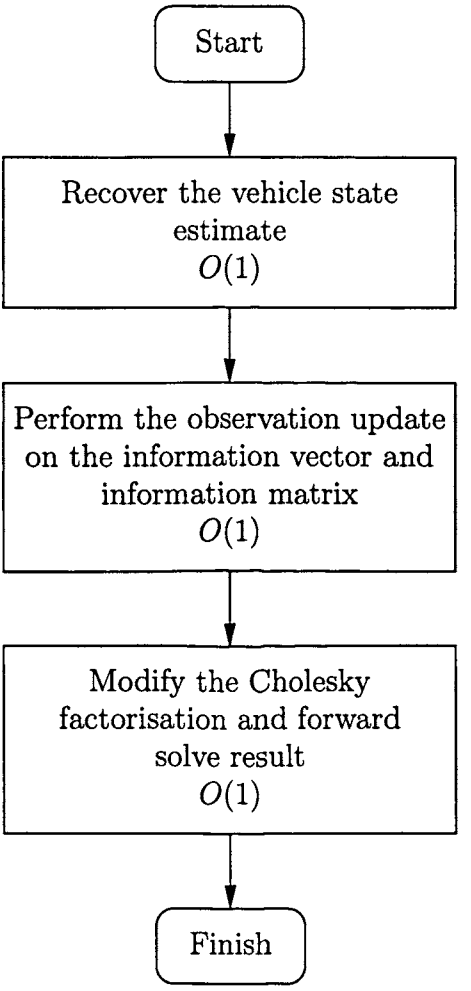


Figure 5.11: The vehicle-state observation and update process. This process applies to observations of the vehicle depth, attitude and velocity and can be performed in constant time.

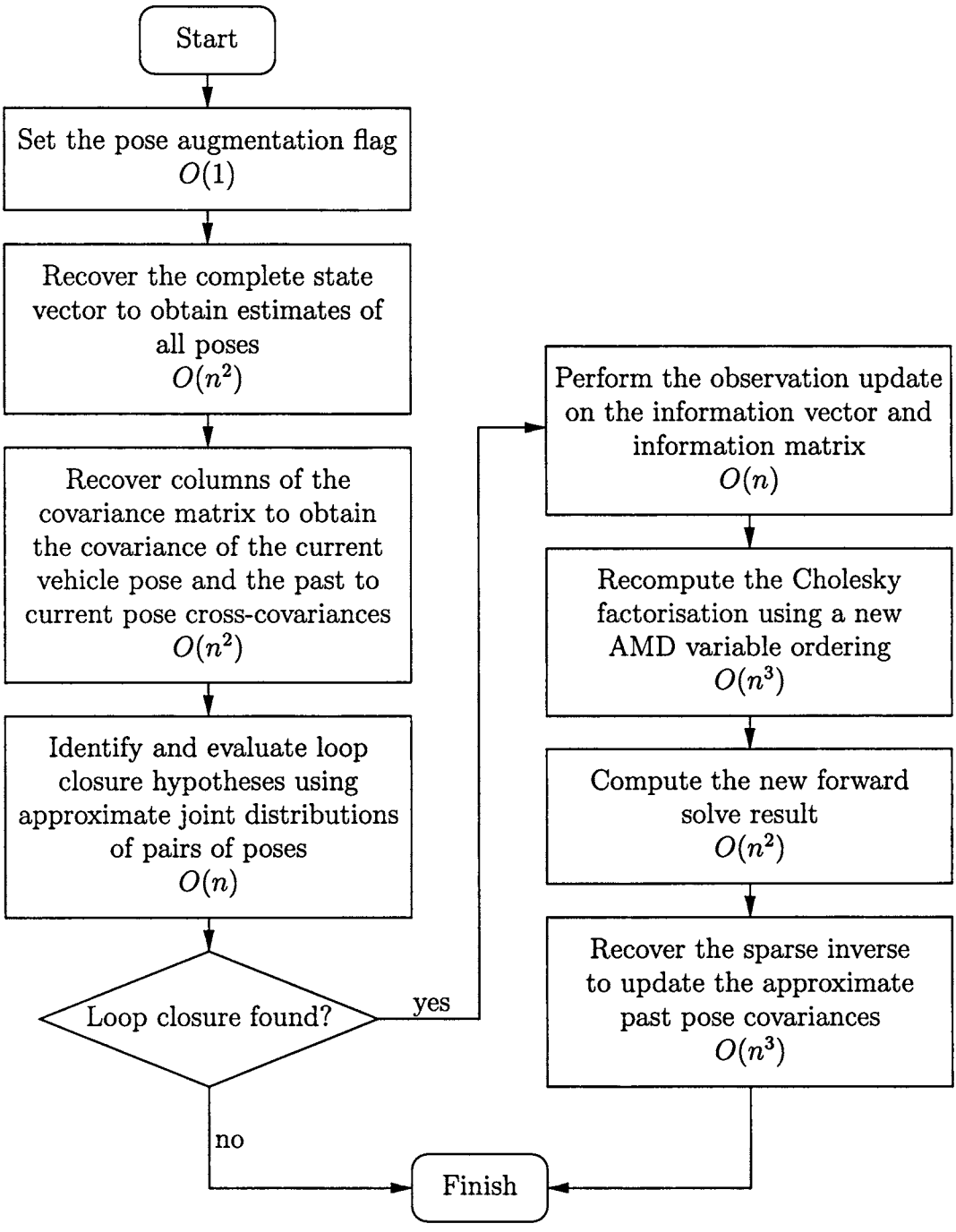


Figure 5.12: The VAN loop-closure observation process. Each time an image is acquired, a flag is set to ensure the vehicle pose is augmented during the next prediction operation. The computational complexity stated in each component assumes the worst-case scenario where the growth of the number of non-zeros in the Cholesky factor is $O(n^2)$ in the number of poses.

5.8.1 A Final Note on Scalability

Throughout this chapter, the computational complexity of operations in the VAN algorithm have been considered by assuming that the filter runs forever and the state vector is continuously augmented with new poses. However, since AUV surveys consist of finite-length missions, a more appropriate measure of scalability is how the required computing power grows as a survey pattern is scaled to larger areas.

Two strategies to scale a reference survey pattern to larger areas as illustrated in Figure 5.13 are considered. The scalability of the VAN algorithm will be evaluated by analysing the time required to process of a loop-closure observation at the end of the mission.

If the vehicle trajectory is simply scaled as demonstrated on the left-hand side of Figure 5.13, the number of loop closures remains constant. Therefore the growth of non-zero elements in the final Cholesky factorisation is linear in the surveyed area, and the complexity of all operations is at worst $O(n)$.

Alternatively, if the survey is scaled by repeating the original pattern as demonstrated on the right-hand side of Figure 5.13, the number of loop closure constraints grows linearly with the area covered. In the worst case, the growth in the number of non-zeros in the Cholesky factor will be $O(n^2)$ in the number of poses and the area covered, resulting in a complexity of $O(n^3)$ for the most expensive operations (the Cholesky factorisation and recovery of the sparse inverse).

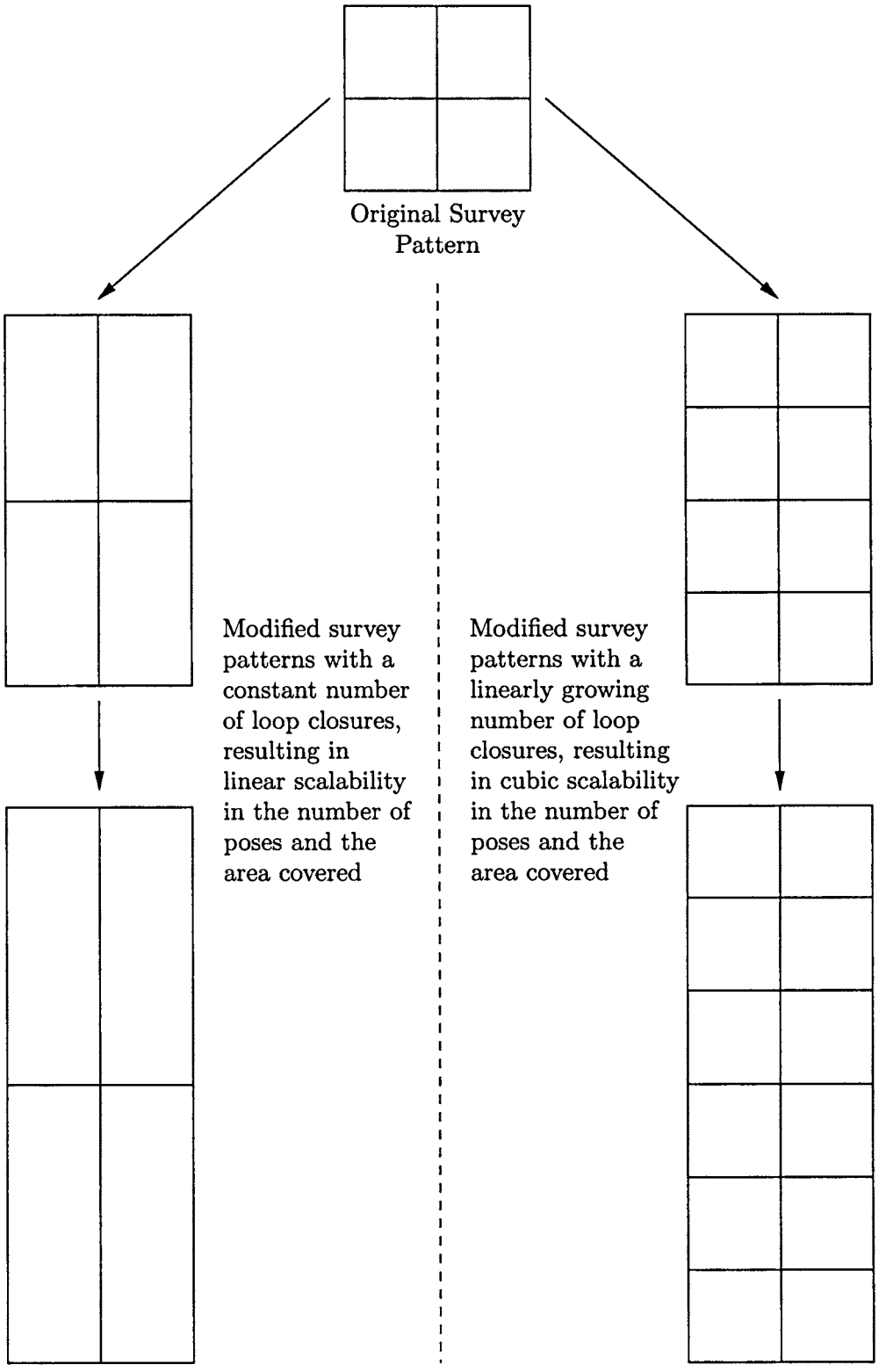


Figure 5.13: Scalability of the VAN algorithm to larger survey areas.

5.9 Results

The VAN algorithm described in this chapter has been applied to data acquired by the SeaBED AUV, using the stereo-vision relative pose estimation algorithm developed in Chapter 4 to provide loop-closure observations.

5.9.1 Stereo Vision Relative Pose Observation

Incorporating the stereo vision data into the VAN filter requires a sensor model that relates the relative stereo pose observation to the estimated vehicle poses.

Figure 5.14, illustrates two vehicle poses ${}^{n}\mathbf{p}_{v_i}$ and ${}^{n}\mathbf{p}_{v_j}$ at index i and j in the state vector. If the pose of the stereo-rig relative to the vehicle frame is ${}^{v}\mathbf{p}_{s}$, the stereo-rig poses relative to the navigation frame are

$$ {}^{n}\mathbf{p}_{s_i}(t_k) = {}^{n}\mathbf{p}_{v_i}(t_k) \oplus {}^{v}\mathbf{p}_{s} \tag{5.56} $$

$$ {}^{n}\mathbf{p}_{s_j}(t_k) = {}^{n}\mathbf{p}_{v_j}(t_k) \oplus {}^{v}\mathbf{p}_{s} \tag{5.57} $$

The observed relative stereo pose ${}^{s_i}\mathbf{p}_{s_j}$ as a function of the estimated vehicle poses is then

$$ \begin{aligned} \mathbf{h}_s[\mathbf{x}(t_k)] &= {}^{s_i}\mathbf{p}_{s_j} \\ &= \ominus {}^{n}\mathbf{p}_{s_j} \oplus {}^{n}\mathbf{p}_{s_i} \\ &= \ominus({}^{n}\mathbf{p}_{v_j} \oplus {}^{v}\mathbf{p}_{s}) \oplus ({}^{n}\mathbf{p}_{v_i} \oplus {}^{v}\mathbf{p}_{s}) \end{aligned} \tag{5.58} $$

The Jacobian of the observation function relative to the states of vehicle pose i is

$$ \frac{\partial \mathbf{h}_s[\mathbf{x}(t_k)]}{\partial {}^{n}\mathbf{p}_{v_i}} = \mathbf{J}_{\oplus 2}\Big|_{{}^{s_j}\mathbf{p}_n, {}^{n}\mathbf{p}_{s_i}} \mathbf{J}_{\oplus 1}\Big|_{{}^{n}\mathbf{p}_{v_i}, {}^{v}\mathbf{p}_{s}} \tag{5.59} $$

The Jacobian of the observation function relative to the states of vehicle pose j is

$$ \frac{\partial \mathbf{h}_s[\mathbf{x}(t_k)]}{\partial {}^{n}\mathbf{p}_{v_j}} = \mathbf{J}_{\oplus 1}\Big|_{{}^{s_j}\mathbf{p}_n, {}^{n}\mathbf{p}_{s_i}} \mathbf{J}_{\ominus}\Big|_{{}^{n}\mathbf{p}_{s_j})} \mathbf{J}_{\oplus 1}\Big|_{{}^{n}\mathbf{p}_{v_j}, {}^{v}\mathbf{p}_{s}} \tag{5.60} $$

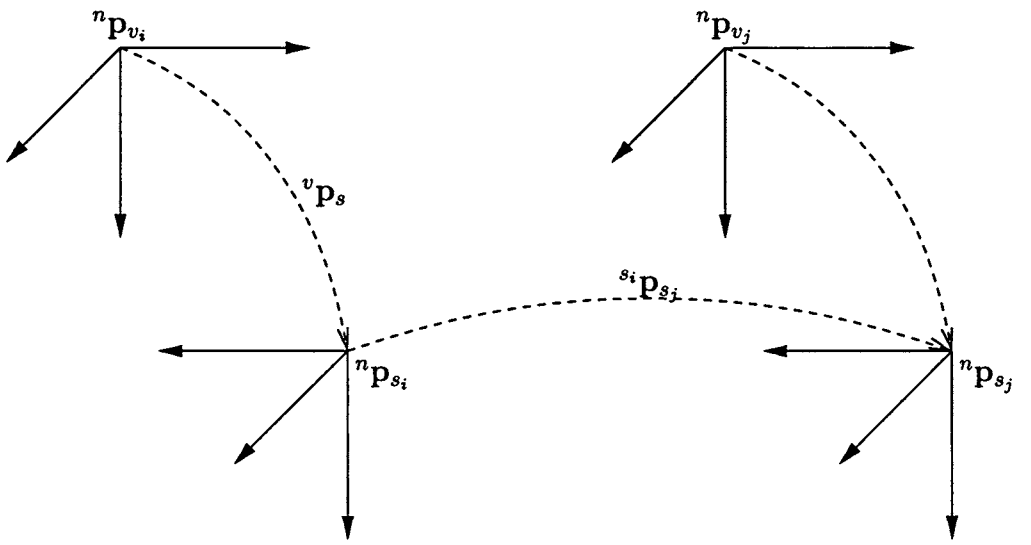


Figure 5.14: Stereo-vision relative pose observation reference frames. ${}^{n}\mathbf{p}_{v_i}$ and ${}^{n}\mathbf{p}_{v_j}$ are a pair of vehicle poses measured relative to the navigation frame, which are estimated in the state vector. ${}^{n}\mathbf{p}_{s_i}$ and ${}^{n}\mathbf{p}_{s_j}$ are the poses of the stereo-rig corresponding to each vehicle pose. The pose of the stereo-rig relative to the vehicle is defined by ${}^{n}\mathbf{p}_{s}$. An observation of the relative stereo pose ${}^{s_i}\mathbf{p}_{s_j}$ is used to improve the estimates of the vehicle poses.

5.9.2 Loop-Closure Hypothesis Generation for Stereo Vision

Since visual feature extraction and association is computationally expensive, generating a small set of loop-closure hypotheses on which image analysis will be performed is critical for the efficiency of the VAN algorithm. Deciding if a pair of poses are accepted as a loop-closure hypothesis is performed by evaluating their joint distributions to estimate the likelihood that images acquired at each pose overlap.

Since a large number of pose pairs will be evaluated, the efficiency of the hypothesis acceptance test is important. The likelihood of overlapping images is evaluated using the highly simplified environment and sensor model illustrated in Figure 5.15. The terrain is modelled as a planar surface, the roll and pitch of the vehicle is assumed to be zero, and the field of view of the stereo-rig is modelled by a bounding cone such that the heading of the vehicle does not affect the region of image overlap.

Under the assumptions of the simplified model, the area of the seafloor imaged by a stereo-rig with a field of view α at an altitude of a_i is a circle with radius

$$r_i = a_i \tan\left(\frac{\alpha}{2}\right) \tag{5.61}$$

If the two-dimensional separation of the stereo-poses i and j in the X-Y plane is

$$\boldsymbol{\delta}_{ij}\left(t_k\right) = \begin{bmatrix} \delta x_{ij}\left(t_k\right) \\ \delta y_{ij}\left(t_k\right) \end{bmatrix} \tag{5.62}$$

the images acquired at each pose overlap if

$$\sqrt{\delta x_{ij}^2 + \delta y_{ij}^2} < \left(r_i + r_j\right) \tag{5.63}$$

The 2D stereo pose separation $\boldsymbol{\delta}_{ij}\left(t_k\right)$ is not known exactly. Its probability distribution can be calculated from the predicted joint distribution of vehicle poses i and j produced by the VAN filter, which has the form

$$\hat{\mathbf{x}}^{-}_{(v_i,v_j)}\left(t_k\right) = \begin{bmatrix} \hat{\mathbf{x}}^{-}_{v_i}\left(t_k\right) \\ \hat{\mathbf{x}}^{-}_{v_j}\left(t_k\right) \end{bmatrix} \tag{5.64}$$

$$\mathbf{P}^{-}_{(v_i,v_j)}\left(t_k\right) = \begin{bmatrix} \mathbf{P}^{-}_{v_iv_i}\left(t_k\right) & \mathbf{P}^{-}_{v_iv_j}\left(t_k\right) \\ \mathbf{P}^{-\mathsf{T}}_{v_iv_j}\left(t_k\right) & \mathbf{P}^{-}_{v_jv_j}\left(t_k\right) \end{bmatrix} \tag{5.65}$$

Through linearisation of the pose composition operation, the joint distribution of stereo poses i and j is defined by the mean and covariance

$$\boldsymbol{\mu}_{(s_i,s_j)}\left(t_k\right) = \begin{bmatrix} \boldsymbol{\mu}_{s_i}\left(t_k\right) \\ \boldsymbol{\mu}_{s_j}\left(t_k\right) \end{bmatrix} \tag{5.66}$$

$$\boldsymbol{\Sigma}_{(s_i,s_j)}\left(t_k\right) = \begin{bmatrix} \boldsymbol{\Sigma}_{s_is_i}\left(t_k\right) & \boldsymbol{\Sigma}_{s_is_j}\left(t_k\right) \\ \boldsymbol{\Sigma}^{\mathsf{T}}_{s_is_j}\left(t_k\right) & \boldsymbol{\Sigma}_{s_js_j}\left(t_k\right) \end{bmatrix} \tag{5.67}$$

where the estimated stereo poses are

$$\boldsymbol{\mu}_{s_i}\left(t_k\right) = \hat{\mathbf{x}}^{-}_{v_i}\left(t_k\right) \oplus {}^{v}\mathbf{p}_s \tag{5.68}$$

$$\boldsymbol{\mu}_{s_j}\left(t_k\right) = \hat{\mathbf{x}}^{-}_{v_j}\left(t_k\right) \oplus {}^{v}\mathbf{p}_s \tag{5.69}$$

and the covariance submatrices are

$$\boldsymbol{\Sigma}_{s_is_i}\left(t_k\right) = \left[\mathbf{J}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_i}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \mathbf{P}^{-}_{v_iv_i}\left(t_k\right) \left[\mathbf{J}^{\mathsf{T}}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_i}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \tag{5.70}$$

$$\boldsymbol{\Sigma}_{s_is_j}\left(t_k\right) = \left[\mathbf{J}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_i}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \mathbf{P}^{-}_{v_iv_j}\left(t_k\right) \left[\mathbf{J}^{\mathsf{T}}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_j}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \tag{5.71}$$

$$\boldsymbol{\Sigma}_{s_js_j}\left(t_k\right) = \left[\mathbf{J}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_j}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \mathbf{P}^{-\mathsf{T}}_{v_iv_j}\left(t_k\right) \left[\mathbf{J}^{\mathsf{T}}_{\oplus 1}\Big|_{\left(\hat{\mathbf{x}}^{-}_{v_j}(t_k),{}^{v}\mathbf{p}_s\right)}\right] \tag{5.72}$$

The 2D stereo pose separation distribution mean and covariance are therefore

$$\boldsymbol{\mu}_{\delta_{ij}}(t_k) = \mathbf{A}\big(\boldsymbol{\mu}_{s_j}(t_k) - \boldsymbol{\mu}_{s_i}(t_k)\big) \tag{5.73}$$

$$\boldsymbol{\Sigma}_{\delta_{ij}}(t_k) = \mathbf{A}\big(\boldsymbol{\Sigma}_{s_j s_j}(t_k) + \boldsymbol{\Sigma}_{s_i s_i}(t_k) - \boldsymbol{\Sigma}_{s_i s_j}(t_k) - \boldsymbol{\Sigma}^{\mathsf{T}}_{s_i s_j}(t_k)\big)\mathbf{A}^{\mathsf{T}} \tag{5.74}$$

where the matrix $\mathbf{A}$ extracts the x-axis and y-axis positions from a six degree of freedom pose vector.

The likelihood of image overlap is calculated by integrating the 2D pose separation probability distribution over the circular region defined in Equation 5.63. In the experiments presented in this chapter, the integration is performed approximately by sampling the pose separation distribution on a 20 by 20 cell grid as demonstrated in Figure 5.16, and pose pairs with an overlap likelihood greater than 0.5% are accepted as loop-closure hypotheses.

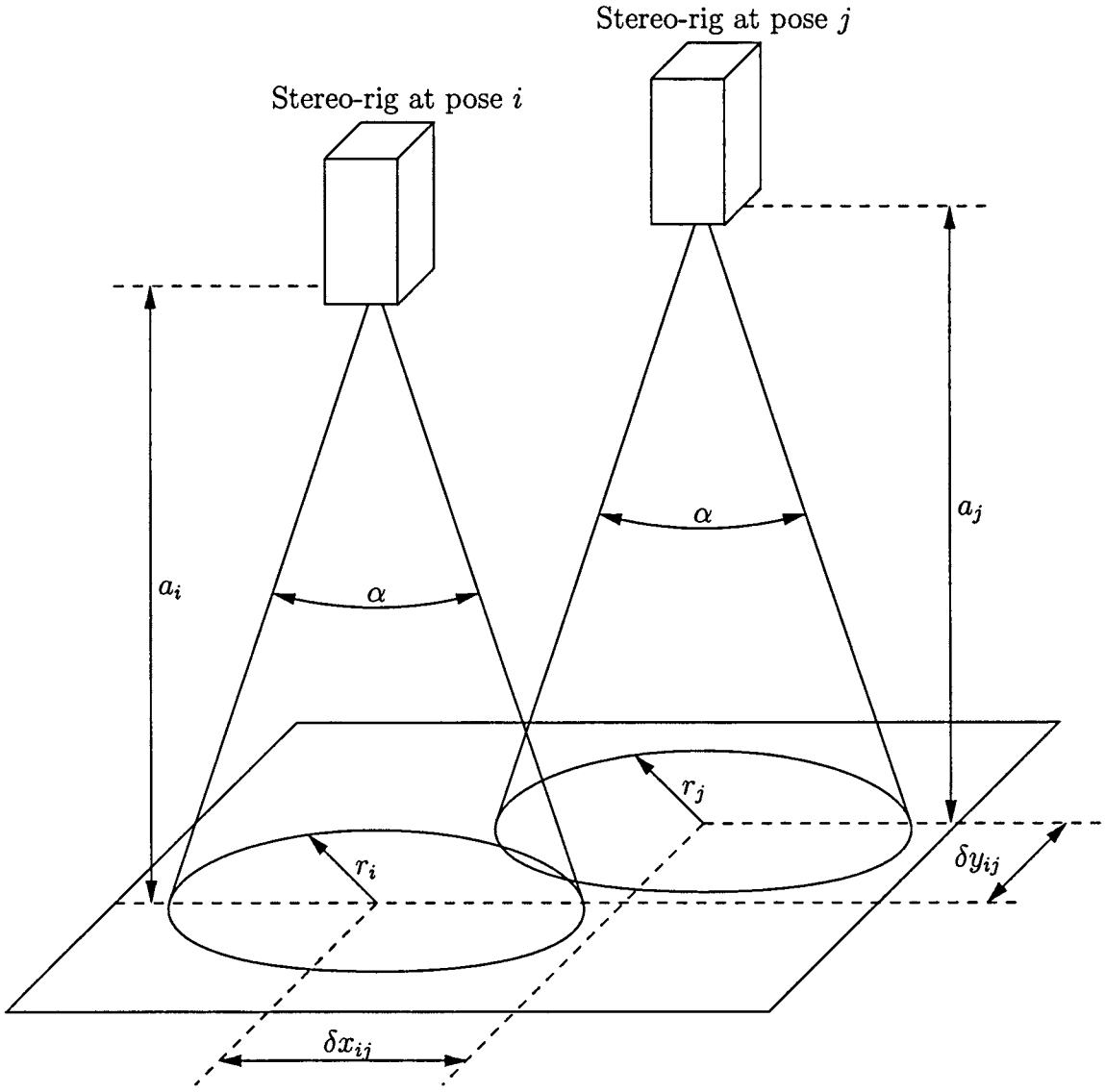


Figure 5.15: Simplified image overlap model for loop-closure hypotheses. A planar terrain structure is assumed, along with zero vehicle roll and pitch, and a constant radial field of view of α. The altitudes of the stereo rig are a_i and a_j, resulting in circular images of radius r_i and r_j. Under these assumptions, the overlapping images occur if $\sqrt{\delta x_{ij}^2 + \delta y_{ij}^2} < (r_i + r_j)$.

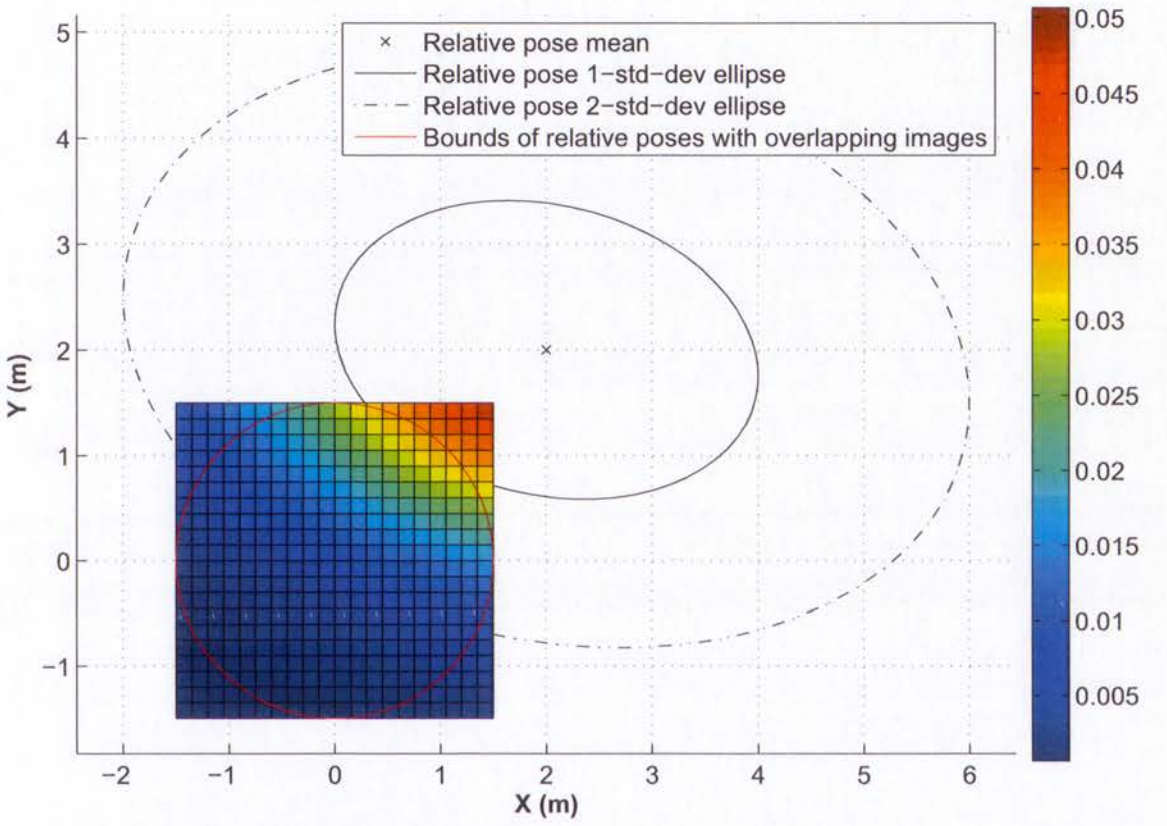


Figure 5.16: Calculating the likelihood of overlapping images. In this example, the mean and covariance of the 2D stereo pose separation distribution are $\boldsymbol{\mu} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} 4 & -0.5 \\ -0.5 & 2 \end{bmatrix}$. The mean of the relative pose distribution is marked by a cross, while the one and two standard deviation ellipses are drawn in black. The red circle shows the bounds of pose separation vectors that support overlapping images. The pose separation probability is sampled on a grid, and cells within the overlap bounds are integrated to estimate the likelihood of overlapping images. A 20 by 20 grid has been used, requiring the calculation of 400 samples. The colour of each grid cell displays the evaluated relative pose likelihood. In this example, the likelihood of overlapping images was estimated to be approximately 8.25%.

5.9.3 Experiment at Ningaloo Marine Park

The SLAM algorithm developed in this chapter has been applied to data collected at the Ningaloo Marine Park near Exmouth in Western Australia. In a deployment to survey sea-sponges, the SeaBED AUV traversed a grid pattern within a square region of 150 by 150 metres. The desired trajectory of the vehicle is illustrated in Figure 5.17. The trajectory is approximately 2.2 kilometres in length, and required approximately 75 minutes to complete. The ocean depth at the survey site is approximately 40 metres, and the AUV maintained an altitude of 2 metres above the seafloor using range measurements from the DVL. Plots of depth and altitude observations acquired during the mission are provided in Figures 5.18 and 5.19.

Figure 5.20 compares dead-reckoning trajectory estimates produced using the approximate vehicle state recovery method of Section 5.5.2, and the exact recovery method presented in Section 5.5.3. The dead-reckoning estimates were produced using the VAN algorithm to fuse DVL velocity, attitude and depth observations. The filter was augmented with the vehicle pose each time an image was acquired, however no loop-closure hypotheses or observations were generated. The trajectory of the vehicle differs from the mission plan in Figure 5.17, since a bug in the planning software caused no goal points to be produced for the short legs at the edges of the grid pattern. The vehicle is therefore observed to cut the corners between the longer grid legs. Due to the high-frequency observations and accurate sensors used in this application, corrections to the past pose estimates are small, and the accuracy of the approximation is surprisingly good. The maximum difference in the vehicle position estimates produced by the two dead-reckoning filters is nine centimetres.

A comparison of dead-reckoning processing times using each vehicle state recovery method is shown in Figure 5.21. When testing the approximate partial state recovery method, Cholesky modifications were used to keep the factorisation updated, however the forward solve result was not calculated or maintained. The exact partial state recovery process is slightly more efficient due to the complexity of the matrix inverse operation required by the approximate method.

The loop-closure hypothesis generation and observation process for this application is illustrated in Figures 5.22 and 5.23, which show the state of the VAN filter before and after the vehicle reaches the first cross-over point in the survey pattern. A large correction in the estimated trajectory and a decrease in the past pose covariances due to loop-closure observations are shown in Figure 5.22, while a detailed view of the pose network at the cross-over point is presented in Figure 5.23.

In this experiment, the SURF algorithm was used to extract and associate visual features for loop-closures observations. Examples of the stereo-image pairs and visual features used

Method	North (m)	East (m)
GPS	-39.25	-23.96
SLAM	-31.43	-34.64
Dead-reckoning	-17.96	-8.84

Table 5.2: Final vehicle position estimates for the Ningaloo Marine Park experiment. The final position estimates are shown at a time just after the AUV surfaced and a GPS position observation was available.

to produce loop-closure observations are presented in Figures 5.24 and 5.25. In both cases SURF is able to extract and associate a sufficient number of features to estimate the relative stereo pose.

A comparison of the estimated trajectories produced by dead-reckoning and SLAM is shown in Figure 5.26. A total of 111 loop-closure observations were applied to the SLAM filter, shown by the red lines joining observed poses. Applying the loop-closure observations results in a trajectory estimate that suggests the vehicle drifted approximately 30 metres south-west of the desired survey area. While no ground truth for the survey is available, arguments for the superiority of the SLAM solution can be created by considering the self-consistency of each estimated trajectory, and the consistency of the final vehicle position estimates with GPS observations acquired after the vehicle surfaced at the end of the mission.

The superior self-consistency of the SLAM solution can be observed in mosaics of images acquired at trajectory cross-over points. Figures 5.27 and 5.28 present mosaics for the cross-over points marked 'A' and 'B' within the dead-reckoning and SLAM trajectory estimates in Figure 5.26. The mosaic of the dead-reckoning cross-over point in Figure 5.27 is inconsistent, since images hypothesised to overlap contain no common features. In contrast, the mosaic of Figure 5.28 produced using vehicle pose estimates from SLAM displays accurately registered overlapping images, demonstrating the correction of dead-reckoning drift.

Estimates of the final vehicle position at the end of the mission produced by dead-reckoning, SLAM and GPS are listed in Table 5.2. The difference between the SLAM estimate and GPS is approximately half that of the dead-reckoned solution. It is likely that a large portion of the error in the SLAM solution was accumulated in the descent to the seafloor and ascent to the surface, since during these times no visual observations are available to correct drifting estimates.

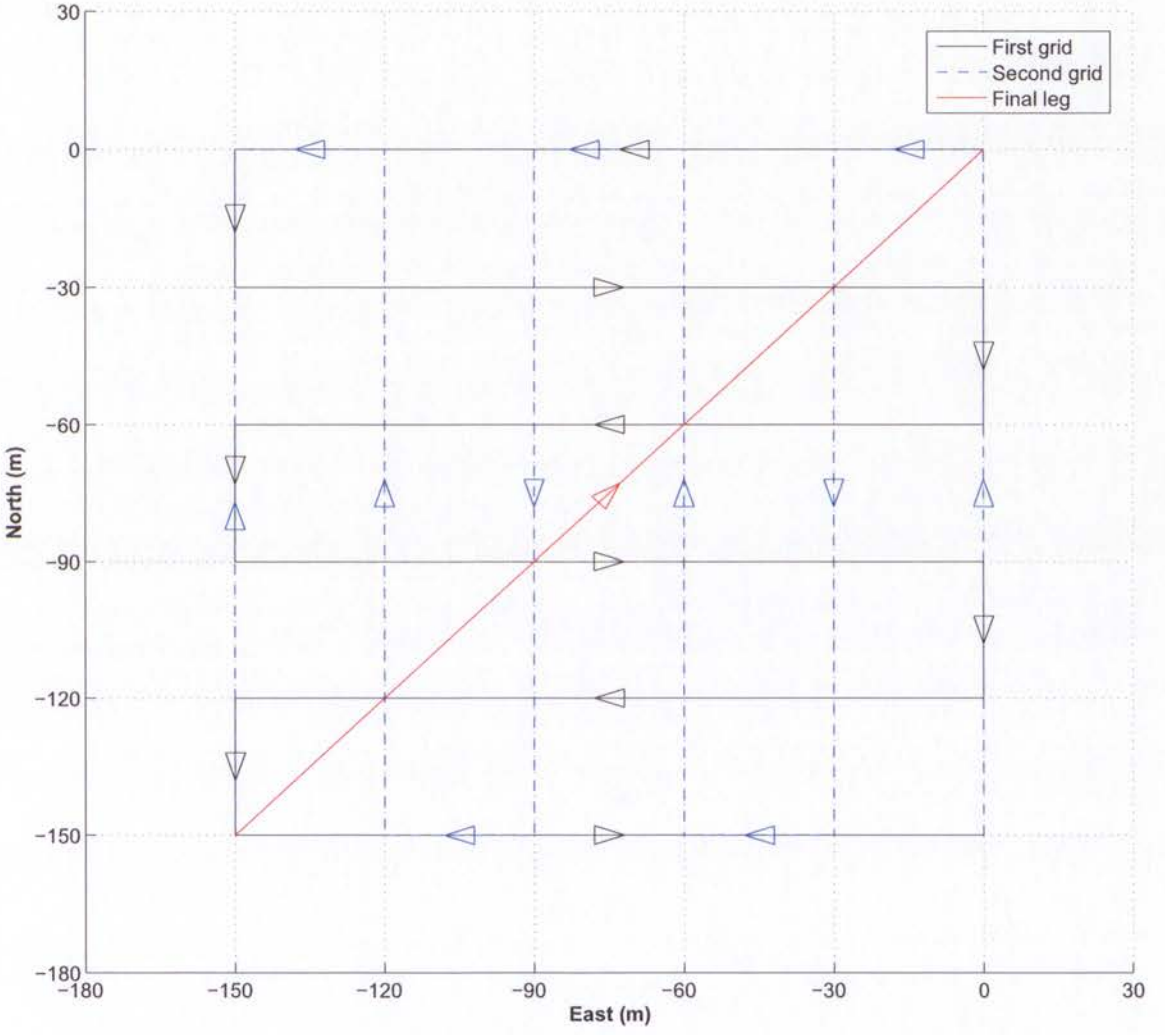


Figure 5.17: Mission plan for the Ningaloo Marine Park experiment. The plan consists of two overlapping grids and a final leg back to the origin in a 150 metre by 150 metre region. The vehicle starts at the origin located at the north-east corner of the grid. The direction of the vehicle is shown for each leg with an arrow.

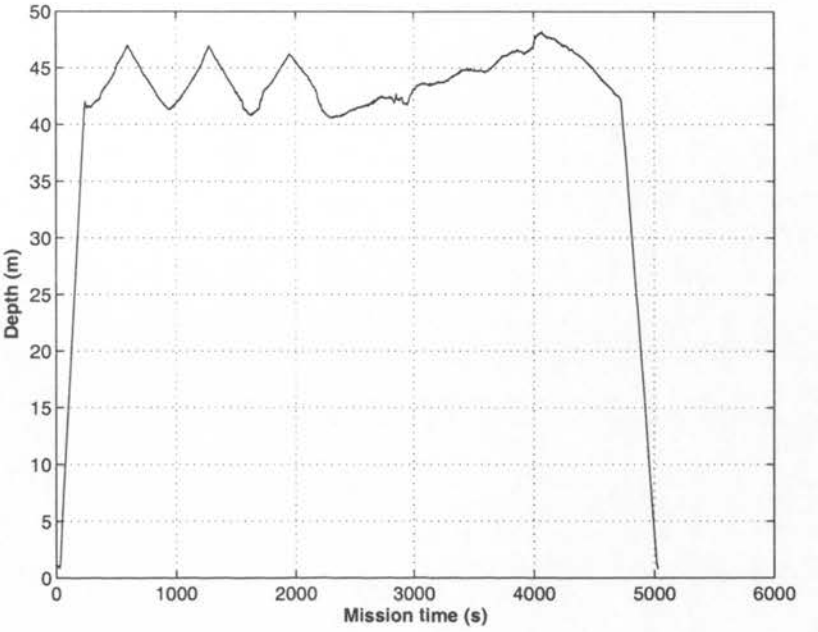


Figure 5.18: Vehicle depth observations for the Ningaloo Marine Park experiment. The depth at the survey site varied between approximately 41 and 48 metres.

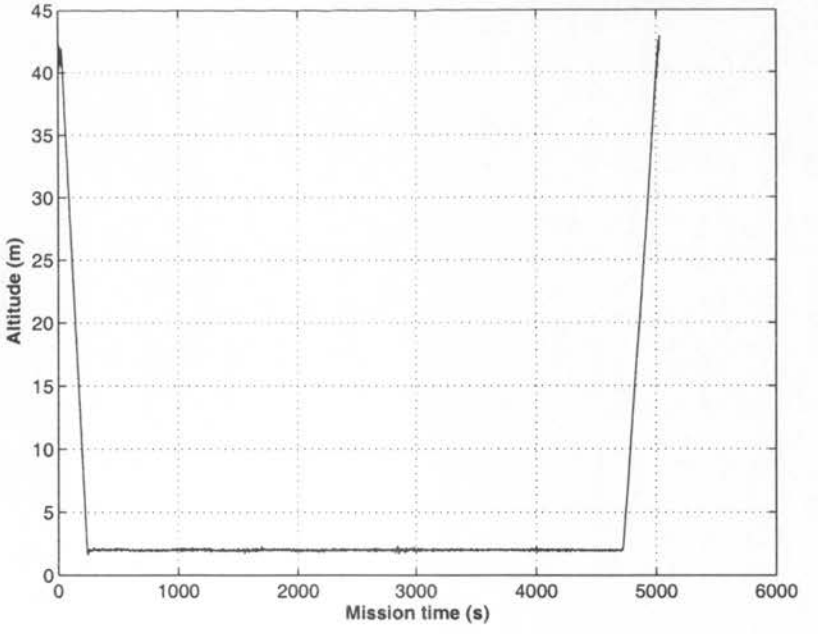


Figure 5.19: Vehicle altitude observations for the Ningaloo Marine Park experiment. Range measurements from the DVL were used to maintain an altitude of 2 metres above the seafloor.

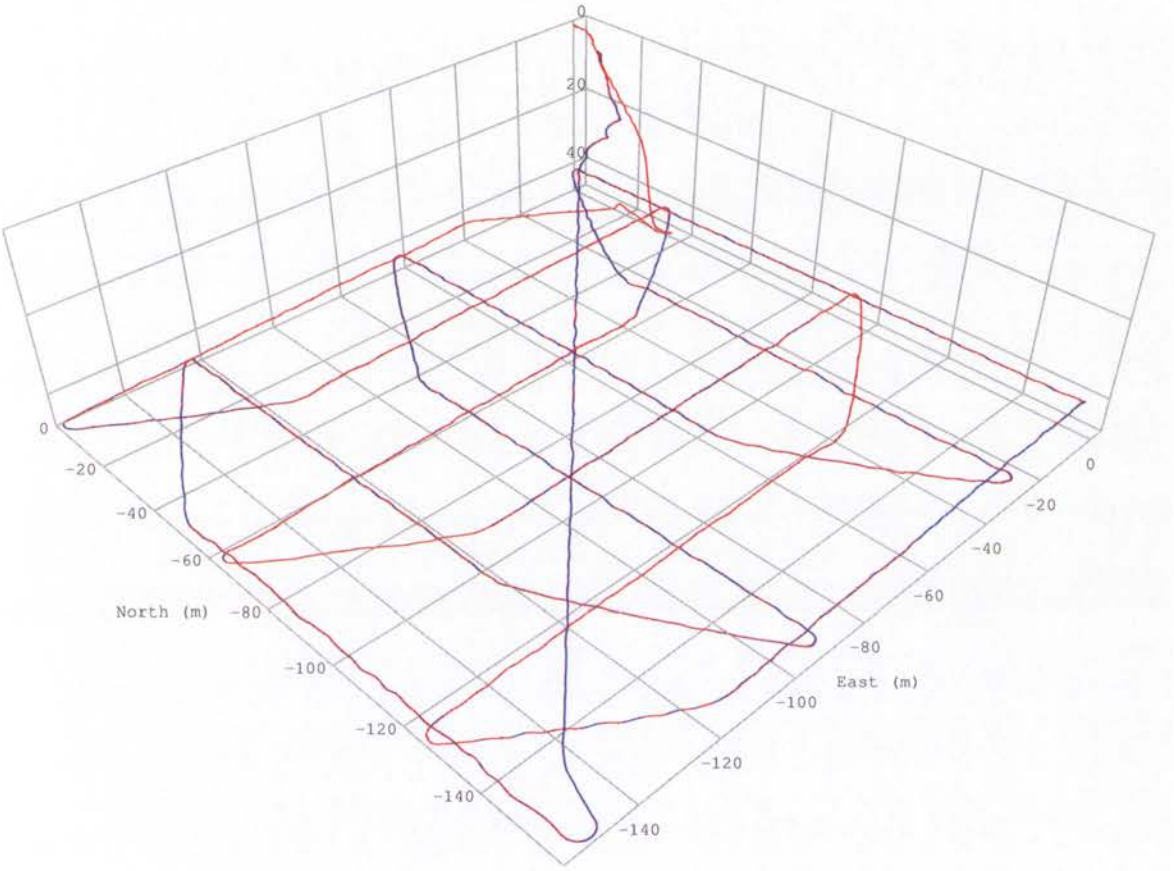


Figure 5.20: Dead-reckoning trajectory estimates for the Ningaloo Marine Park experiment. The SeaBED AUV performed a survey pattern at a depth of approximately 40 metres and an altitude of 2 metres above the seafloor. Vehicle pose estimates produced using the optimal vehicle state recovery method of Section 5.5.3 are drawn in blue, and the estimates produced using the approximate partial state recovery method of Section 5.5.2 are shown in red. In this application, the accuracy of the approximate state recovery method is surprisingly good, with a maximum difference of 9cm from the optimal solution. The trajectory of the vehicle differs from the mission plan in Figure 5.17, since a bug in the planning software caused no goal points to be produced for the short legs at the edges of the grid pattern. The vehicle is therefore observed to cut the corners between the longer grid legs.

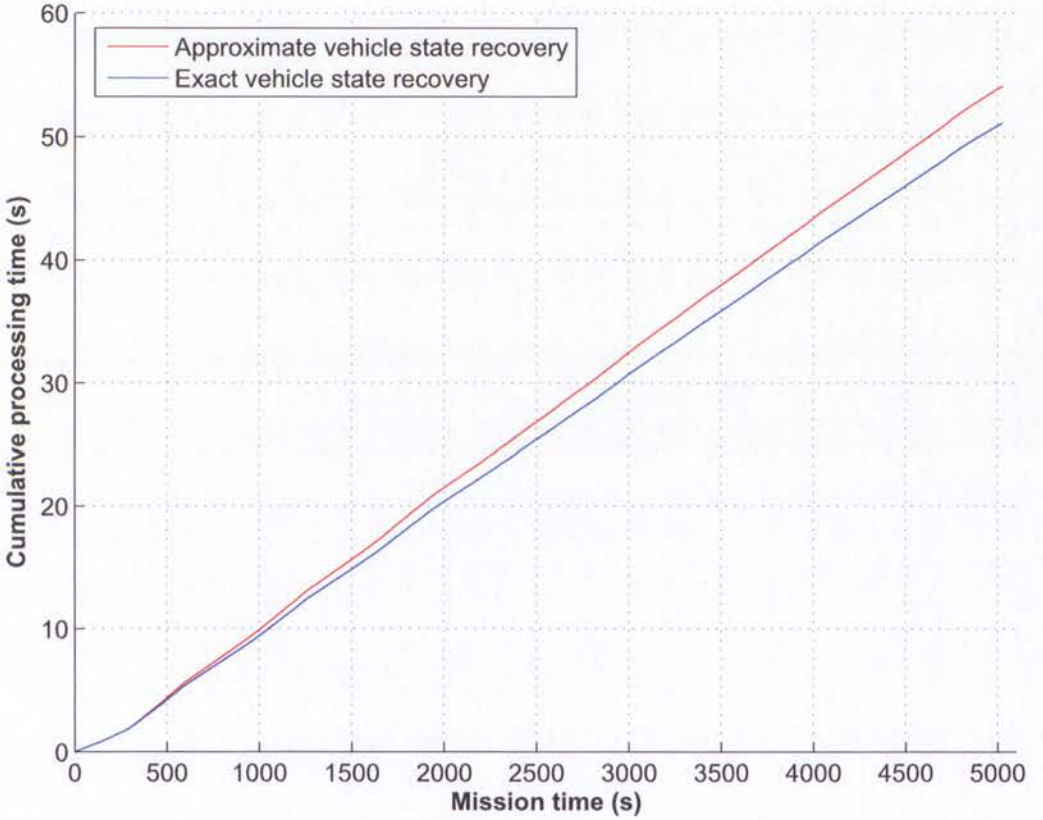
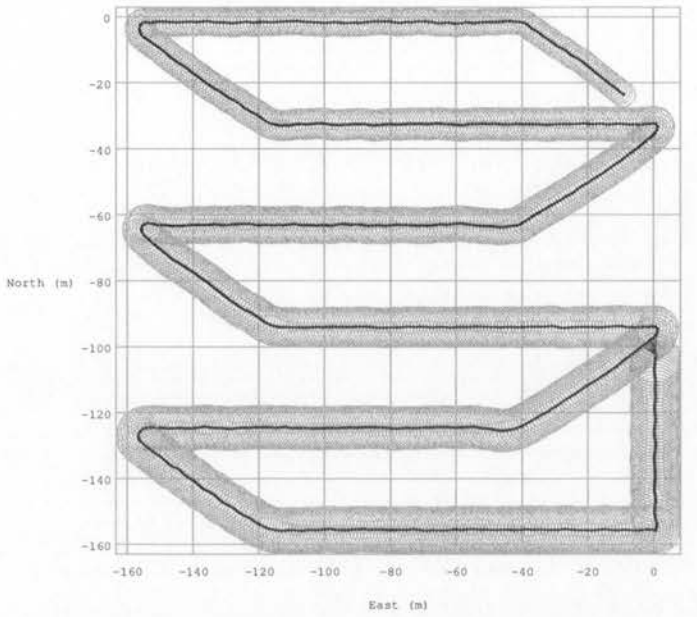
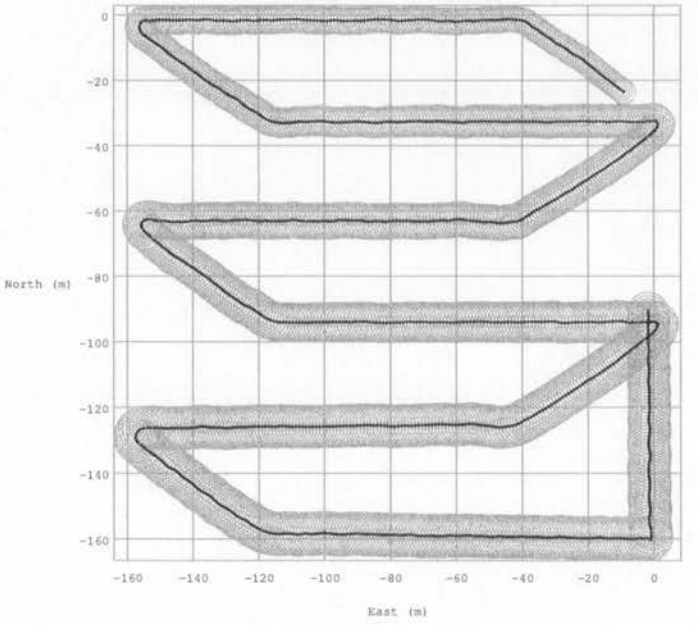


Figure 5.21: Dead-reckoning processing times. Cumulative processing times are shown for dead-reckoning using the approximate and exact partial state recovery methods. The exact state recovery approach proves to be slightly more efficient than the approximate method which requires a matrix inverse operation.

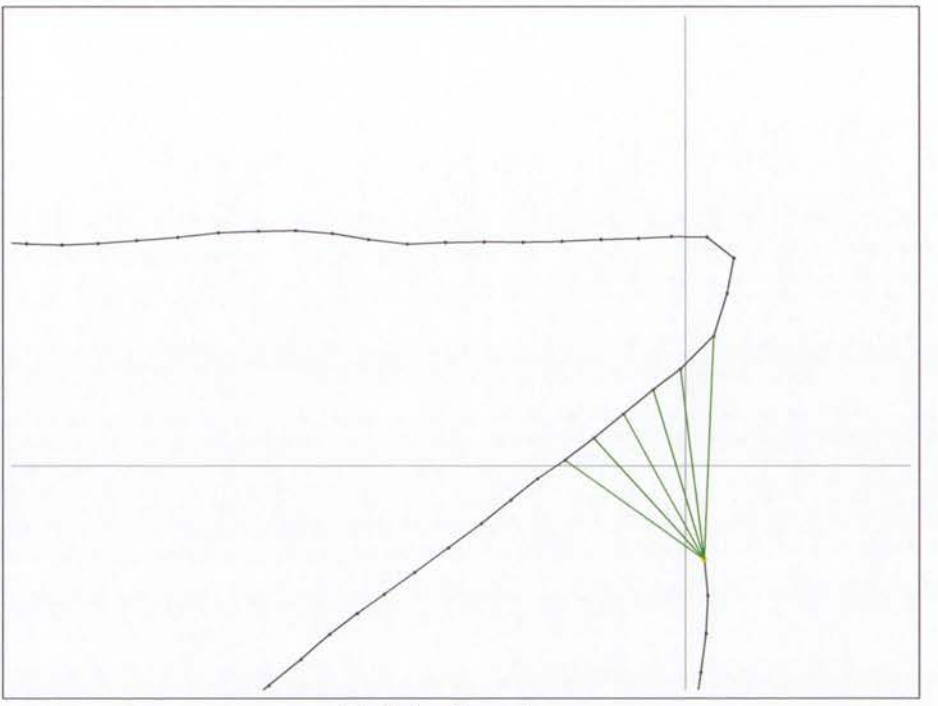


(a) Before loop-closure

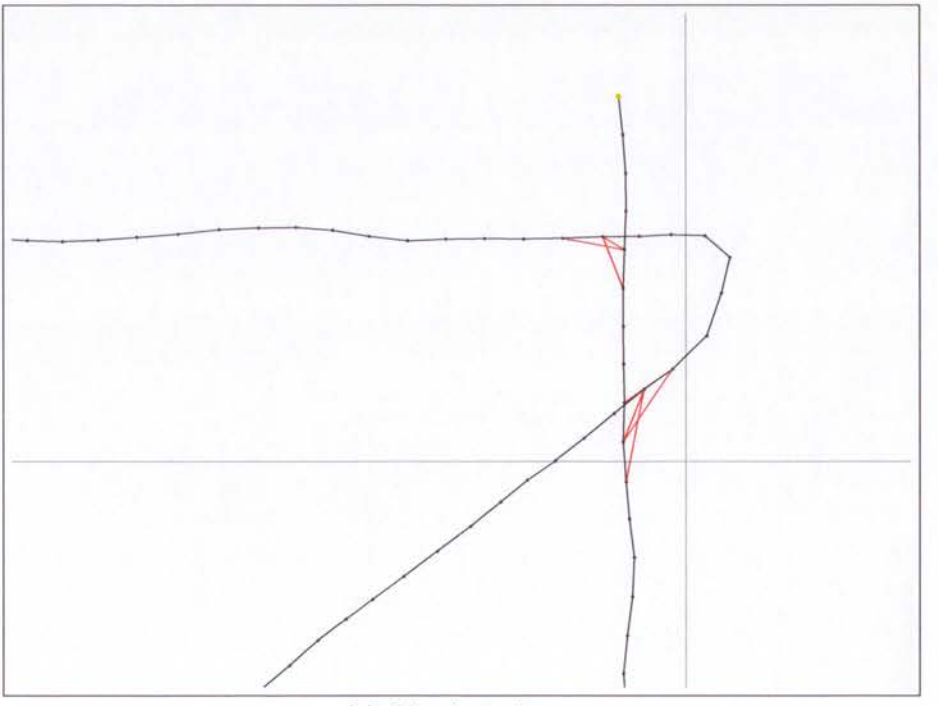


(b) After loop-closure

Figure 5.22: The stereo-vision loop-closure process. Vehicle pose estimates are represented by black dots, and their uncertainties are shown by 95% confidence ellipsoids drawn in grey. Poses linked by odometry constraints are connected by black lines. In (a), green lines connect pairs of poses forming loop-closure hypotheses, and in (b) red lines are drawn between pairs where a loop-closure observation has been applied. The loop-closure observations in (b) result in a correction of the estimated vehicle trajectory, and a decrease in the uncertainty of many poses.



(a) Before loop-closure



(b) After loop-closure

Figure 5.23: The stereo-vision loop-closure process (detail view). Loop-closure hypotheses for pairs of poses likely to contain image overlap are shown by green lines in (a), and poses linked by loop-closure observations are connected by red lines in (b).

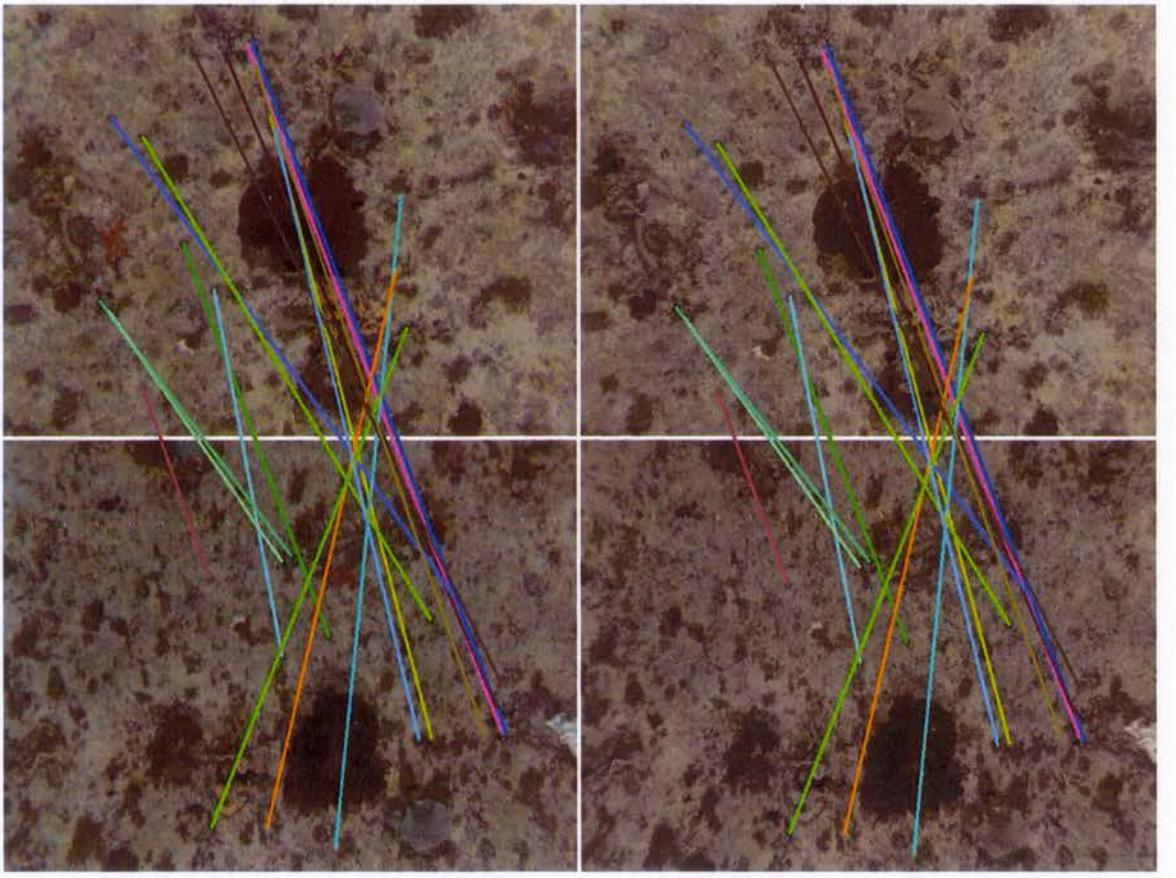


Figure 5.24: Stereo-vision loop-closure observation example one. The left and right stereo images acquired at a first pose are shown on top, while the stereo images acquired at a second pose are shown below. Features associated between all images are marked by lines joining their locations in both left and right frames.

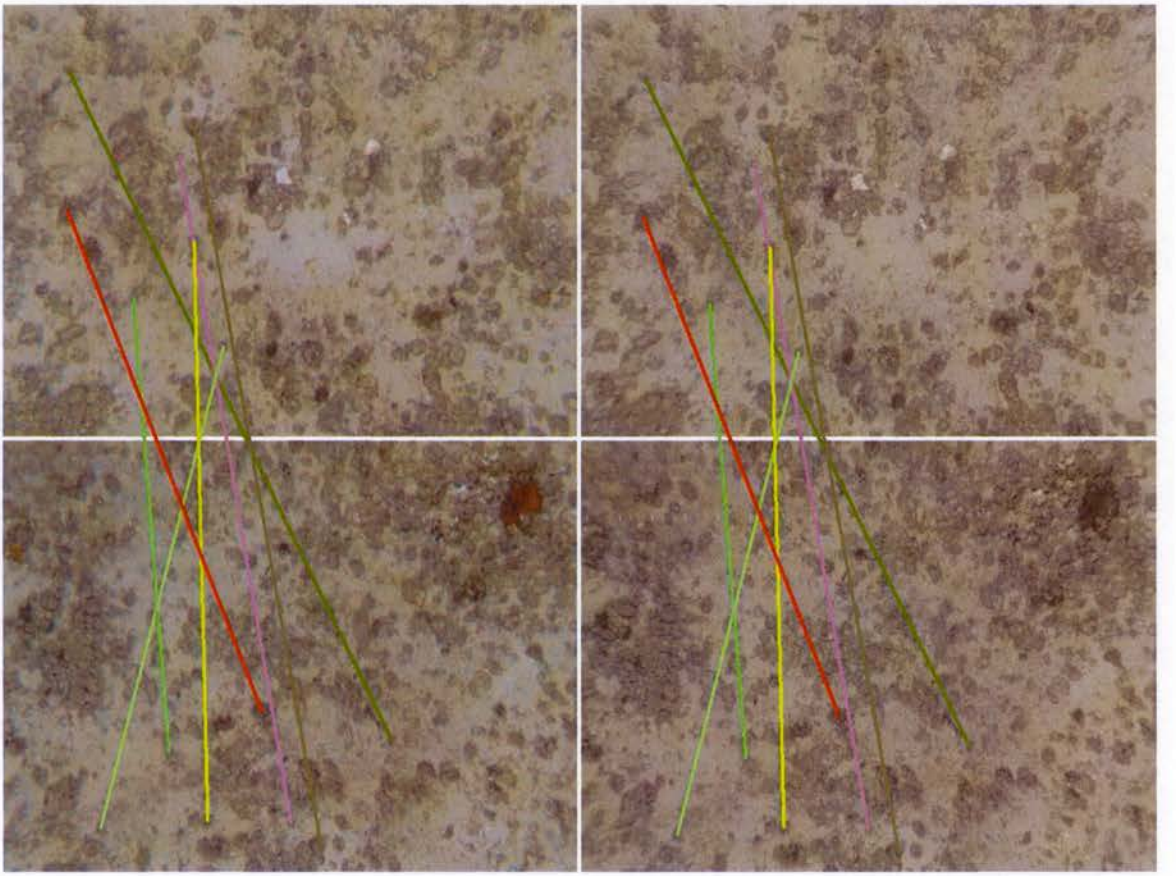


Figure 5.25: Stereo-vision loop-closure observation example two. The left and right stereo images acquired at a first pose are shown on top, while the stereo images acquired at a second pose are shown below. Features associated between all images are marked by lines joining their locations in both left and right frames.

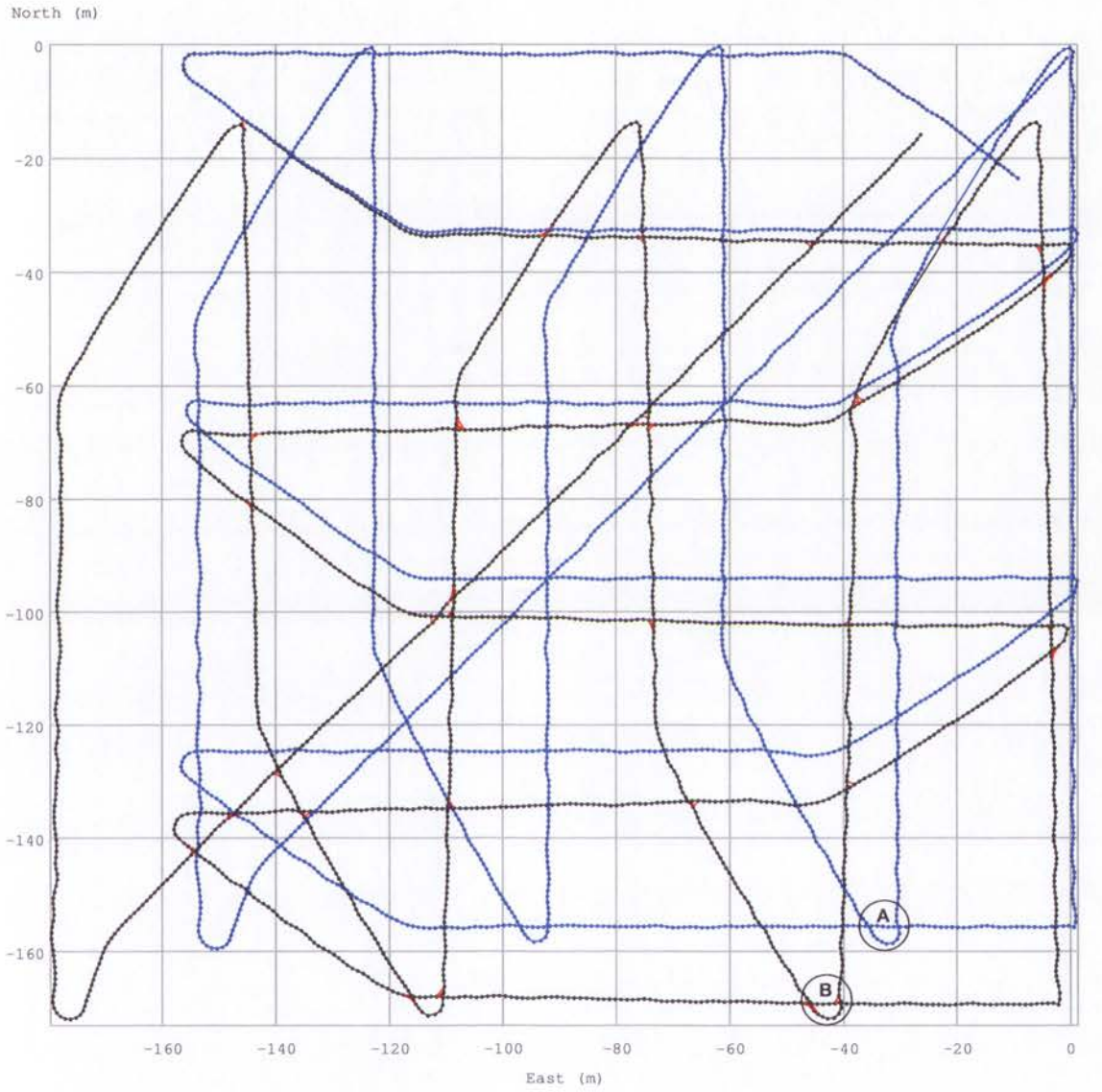


Figure 5.26: Comparison of dead-reckoning and SLAM vehicle trajectory estimates. The SLAM trajectory is shown in black, while the dead-reckoning estimates are shown in blue. The SLAM estimates suggest the vehicle has drifted approximately 30 metres south-west of the desired survey area. Mosaics of images acquired at the the trajectory cross-over points marked 'A' and 'B' are shown in Figures 5.27 and 5.28.

(a) Images overlaid in chronological order

(b) Images overlaid in reverse order

Figure 5.27: Inconsistency of the dead-reckoning vehicle trajectory. Mosaics are shown for image acquired at the estimated trajectory cross-over point marked 'A' in Figure 5.26. Images predicted to contain overlap do not match due to significant localisation errors.

(a) Images overlaid in chronological order

(b) Images overlaid in reverse order

Figure 5.28: Consistency of the SLAM vehicle trajectory. Mosaics are shown for image acquired at the estimated trajectory cross-over point marked 'B' in Figure 5.26.

Update Method	Loop-closure hypotheses	Loop-closure observations
None	2754	112
Single pose EKF	2596	112
Sparse inverse	1186	111
Optimal	1186	111

Table 5.3: Loop-closure statistics for conservative pose updating strategies.

The final state vector for the Ningaloo Marine Park experiment contains 25884 variables from 2157 poses. The sparsity pattern of the final information matrix is shown in Figure 5.29. The information matrix contains 939528 non-zero elements and is 99.86% sparse. Most of the non-zero elements result from odometry constraints, however each of the 111 loop-closure observations result in a block of non-zeros below the block tri-diagonal.

The sparsity pattern of the final Cholesky factor is shown in Figure 5.30. The advantage of using a fill-in reducing permutation can be observed by comparing the factors produced using the natural and AMD variable orderings. The variable ordering produced by the AMD algorithm results in a factor with approximately one-sixth of the number of non-zeros produced by the natural ordering.

An evaluation of strategies to update the conservative past pose covariances is presented in Figure 5.31, where the trace of the covariances is used as a measurement of their uncertainty. For comparison, optimal (non-conservative) values were produced by recovering the true pose covariances using the sparse inverse method. For a survey pattern with few cross-over points, applying a single pose EKF update after each loop-closure provides little benefit. The strategy of updating the conservative poses using the sparse inverse method after each loop-closure produces near optimal results in this application, since the dead-reckoning observations (depth, attitude and velocity) between loop-closure observations provide only small updates to past poses. The number of loop-closure hypotheses and observations produced when using each conservative pose update strategy are listed in Table 5.3. As expected, using the sparse inverse method results in a significant reduction in the number of generated loop-closure hypotheses.

The growth in the number of non-zero elements in the Cholesky factor for the Ningaloo experiment is shown in Figure 5.32. In general the number of non-zeros for SLAM is $O(n^2)$ in the number of poses. Due to the sparse set of cross-over points in the Ningaloo experiment however, the number of non-zeros caused by new poses and odometry constraints (which grow linearly) outnumber those from loop-closure observations. As a result, in this case the growth in the number of non-zeros is not much worse than linear.

The processing times for prediction operations is shown in Figure 5.33. Prediction without

Operation	Count	Mean (s)	Minimum (s)	Maximum (s)
Feature extraction from a single image	952	0.1631	0.0952	0.2680
Feature association between a stereo image pair	476	0.0214	0.0024	0.0771
Feature association between two stereo image pairs	1186	0.0104	0.0018	0.0339

Table 5.4: SURF image analysis statistics. Processing times were acquired on a 2.0 GHz Pentium M processor.

augmentation is a constant time operation, however when augmenting poses, on occasion the capacity of the information vector, information matrix and Cholesky factor need to be resized. In this implementation the capacity of the vectors and matrix are always doubled, leading to a decreasing frequency of resize events and amortised constant time complexity.

The processing times for vehicle state observations including depth, attitude and velocity observations are shown in Figure 5.34. As expected, all observations of the current vehicle states are constant time operations.

The processing times for the operations required to generate loop-closure hypotheses are shown in Figure 5.35. A total of 2303531 pose pairs were tested, producing 1186 loop-closure hypotheses resulting in 111 loop-closure observations. While recovering the state estimate vector and columns of the covariance matrix are in general $O(n^2)$ operations, the growth of their processing times in this experiment is not much worse than linear due to the near linear growth in the number of non-zeros in the Cholesky factor.

The processing times for the operations required to apply loop-closures observations to the filter are shown in Figure 5.36. Updating the information matrix is a $O(n)$ operation in this implementation due to the use of a compressed row storage format requiring $O(n)$ values to be shifted when a new non-zero element is inserted. As expected, recalculating the Cholesky factorisation and recovering the sparse inverse are the most expensive operations. While the computational complexity of these operations in general is $O(n^3)$, the growth of their processing times in this experiment is not much worse than linear due to the near linear growth in the number of non-zeros in the Cholesky factor.

Statistics for the visual feature extraction and association operations performed by the SURF algorithm to create loop-closure observations are listed in Table 5.4. In this experiment, the images were scaled down from their original size 1360x1024 to 680x512 pixels.

If the localisation filter and image analysis is performed on the same CPU, processing the entire mission required 8 minutes and 9 seconds. If cached visual feature locations

and descriptions are used, simulating the process of using dedicated hardware for image analysis, processing the mission required 4 minutes and 10 seconds. The processing times presented in this chapter have been measured on a 2.0 GHz Pentium M processor. Currently all processing is performed off-line on logged data, however these timing results suggest an on-line implementation is feasible.

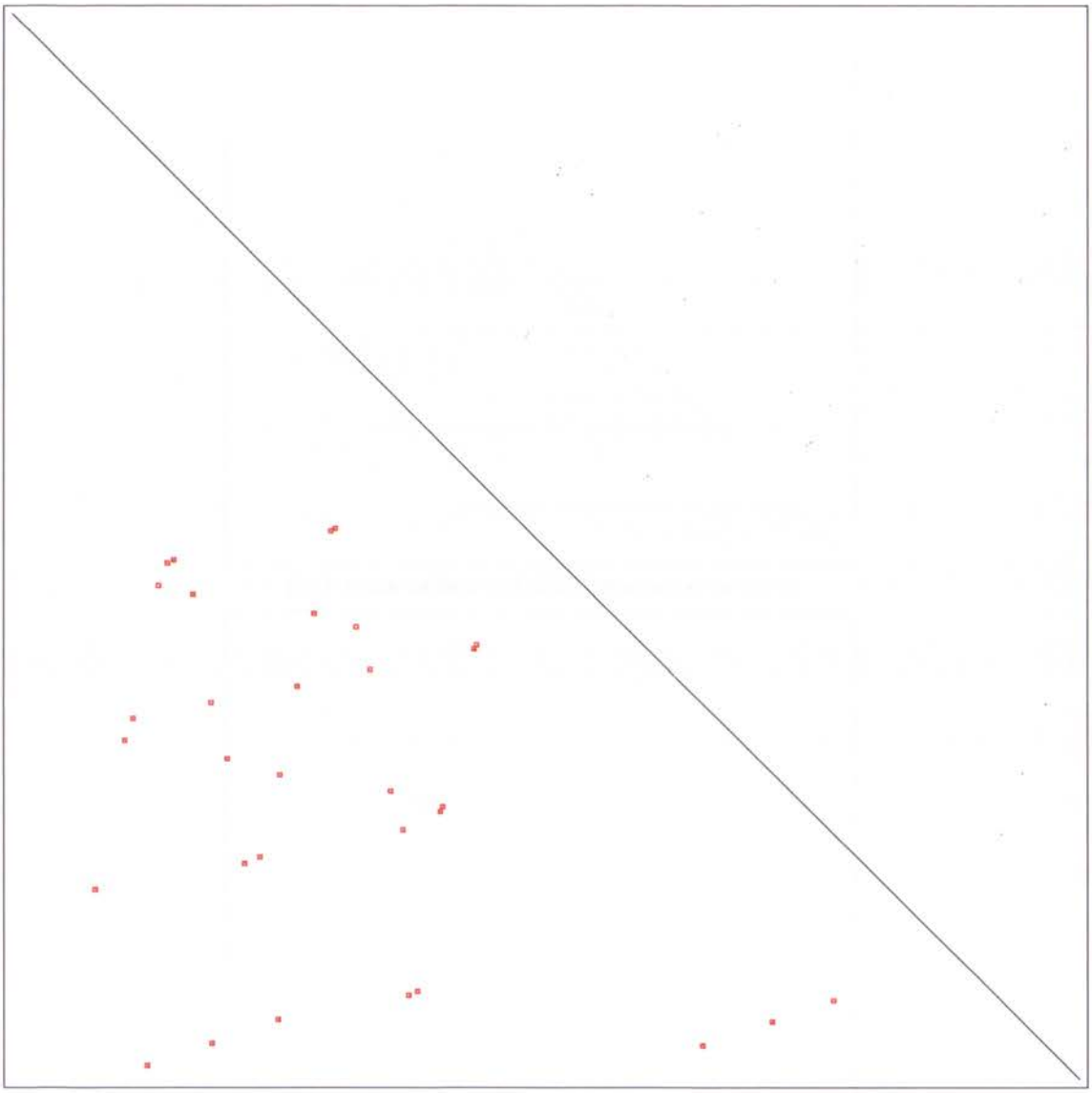
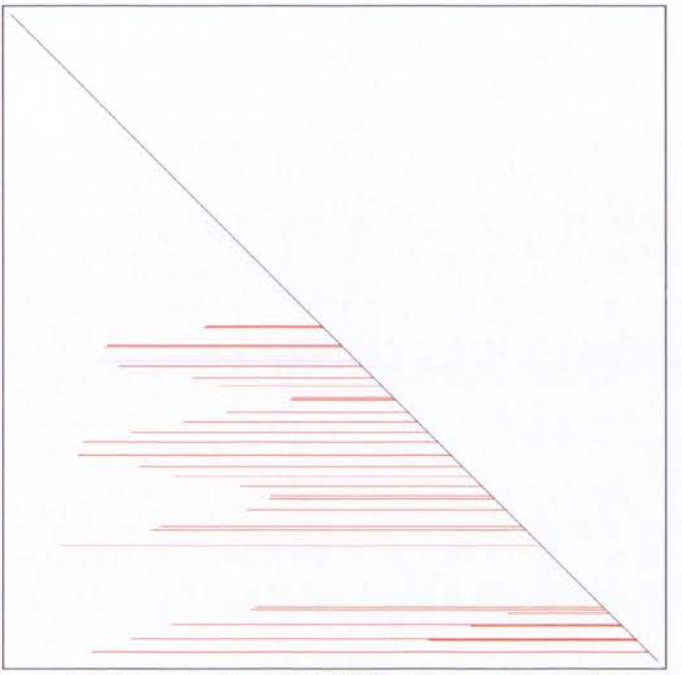
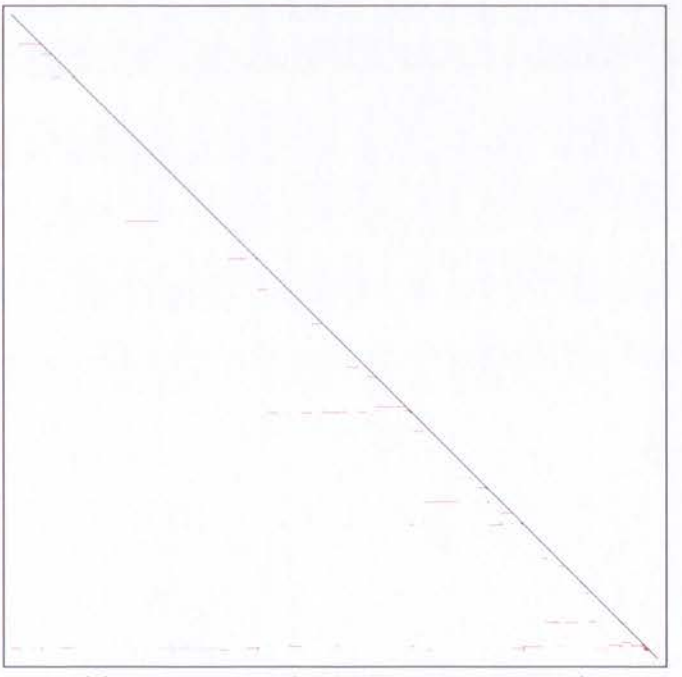


Figure 5.29: Structure of the final information matrix. The final state vector contains 25884 variables from 2157 poses. The information matrix is 99.86% sparse, containing 939528 non-zero elements. Most of the non-zero elements result from poses and odometry constraints, and are located in the block tri-diagonal. Each of the 111 loop-closure observations applied to the filter result in a block of non-zero elements above and below the block tri-diagonal. In this image, each pixel represents a 12x12 submatrix, and black pixels show non-zero elements. In the lower triangle, red squares have been drawn around groups of non-zero blocks at each of the 32 trajectory cross-over points where loop-closures were found.



(a) Natural ordering (5165838 non-zero elements)



(b) AMD ordering (804222 non-zero elements)

Figure 5.30: Structure of final Cholesky factors produced with natural and AMD variable orderings. In each image, a pixel corresponds to a 12x12 element submatrix. Non-zero elements at the location of non-zeros in the information matrix are drawn in black, while those produced by fill-in are shown in red. The AMD variable reordering has resulted in approximately one-sixth of the number of non-zeros produced by the natural ordering.

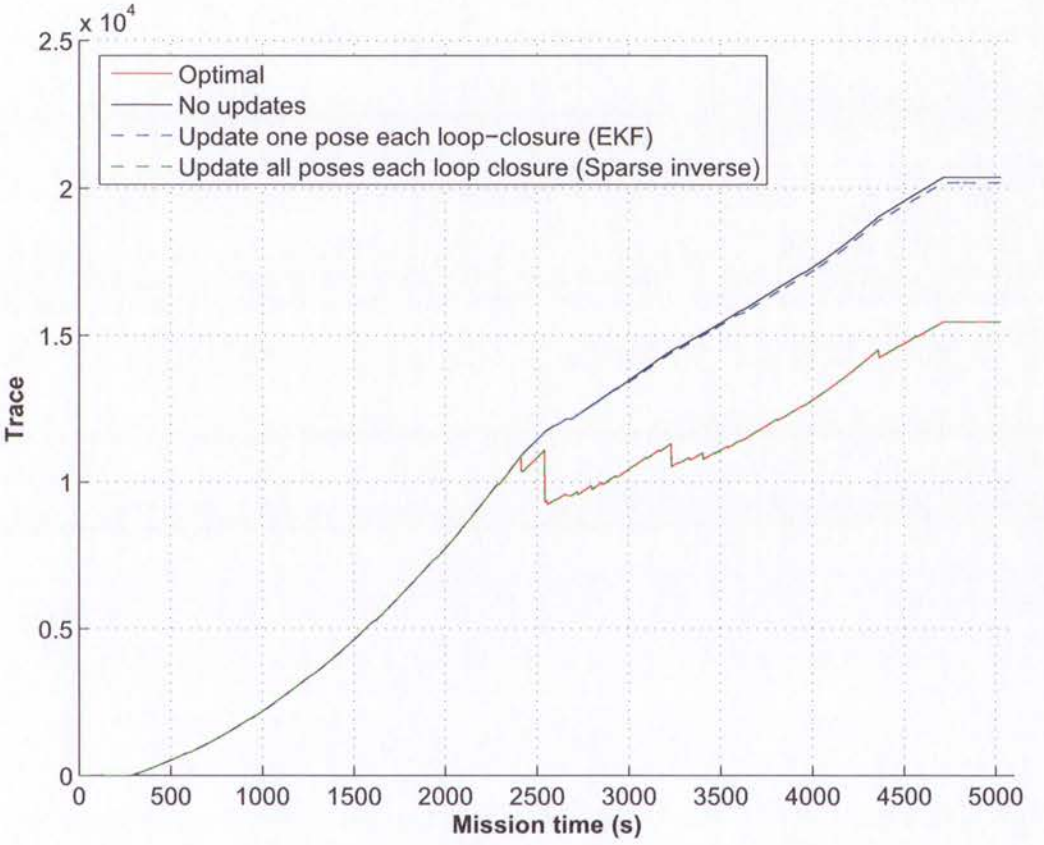


Figure 5.31: Evaluation of conservative covariance updating strategies. The trace of the conservative pose covariance submatrices has been used as a measure of their uncertainty. In an exploration style mission with few loop-closures, updating a single pose covariance after each loop-closure with the EKF update method produces little benefit. Updating all pose covariances using the sparse inverse recovery method each time a loop-closure observation is applied to the filter maintains conservative covariances that are close to optimal.

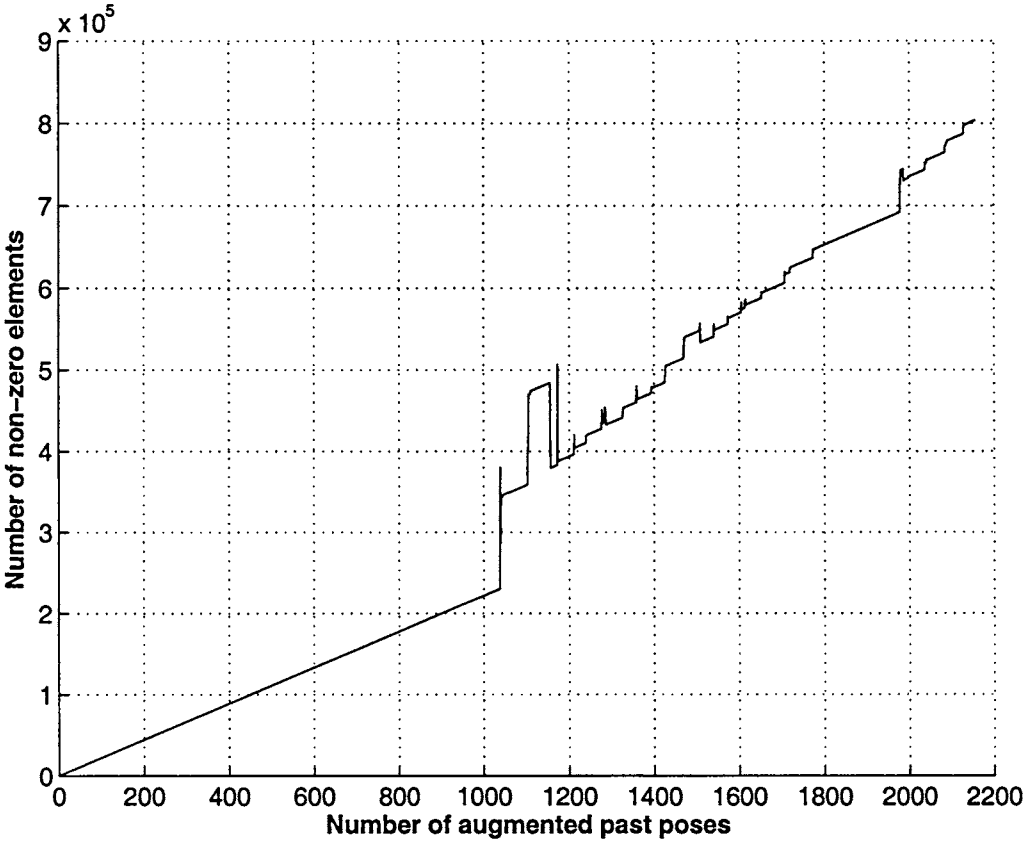
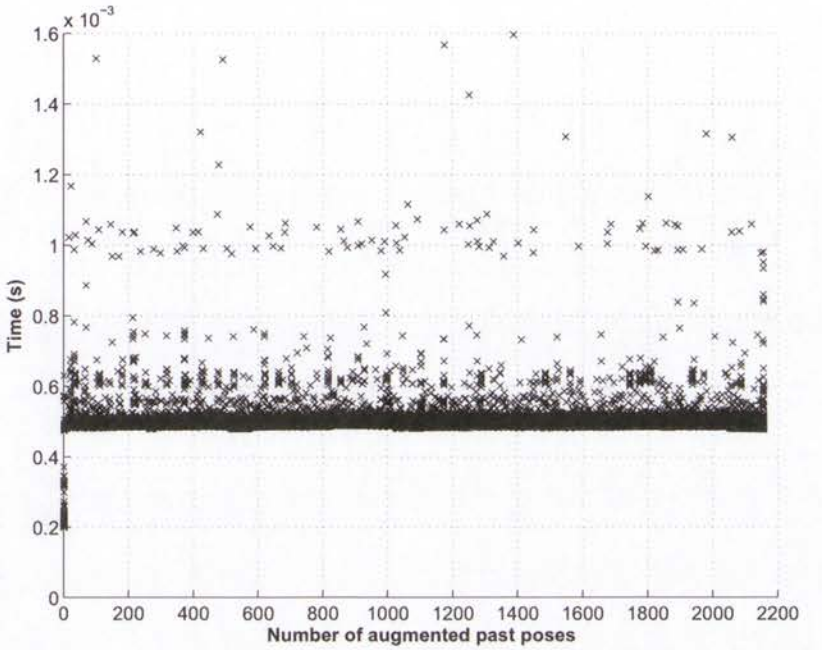
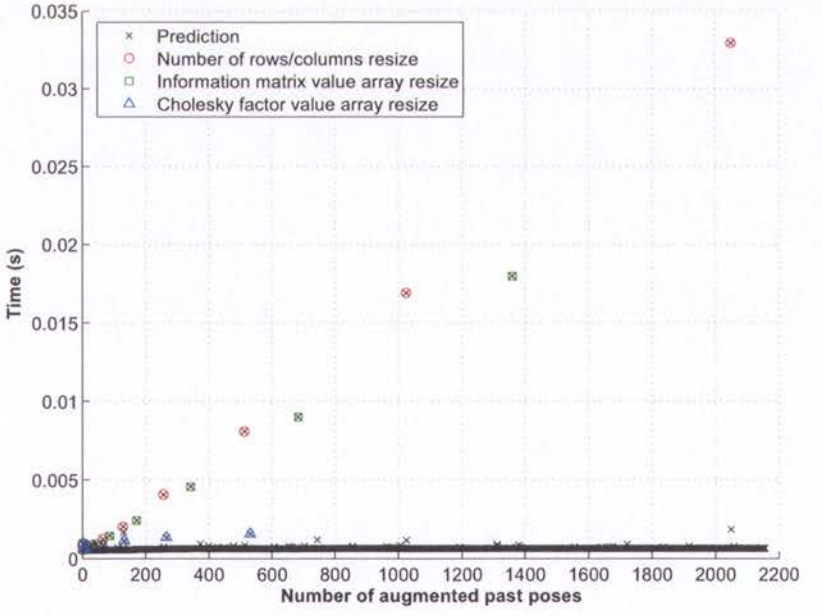


Figure 5.32: Growth of the number of non-zero elements in the Cholesky factor. The number of non-zeros grows linearly with the number of augmented poses between loop-closure observations, the first of which occurs when there are 1039 augmented past poses. The greedy nature of the AMD algorithm produces an irregular growth pattern, with some loop-closures resulting in a reduction in the number of non-zeros when a better ordering is found. For SLAM, the worst-case number of non-zeros is $O(n^2)$ in the number poses, however the sparse set of loop-closures in the Ningaloo experiment result in a growth that is not much worse than linear.



(a) Prediction without augmentation



(b) Prediction with augmentation

Figure 5.33: Processing times for prediction operations. In (a), prediction without augmentation is performed in constant time, while in (b), prediction with augmentation is performed in amortised constant time due to resize events. When a matrix or vector needs to be resized, the capacity is doubled, resulting in increasing periods between resize events.

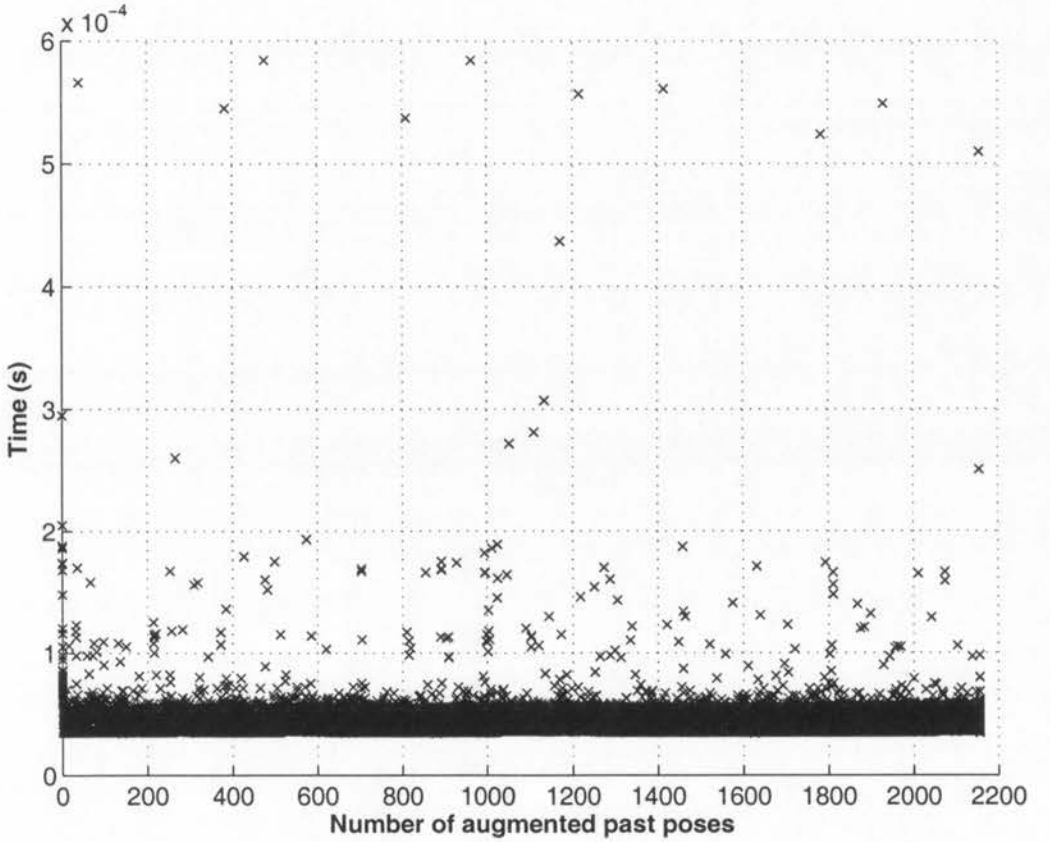


Figure 5.34: Processing times for vehicle state observations. Observations of the current vehicle depth, attitude and velocity are all performed in constant time.

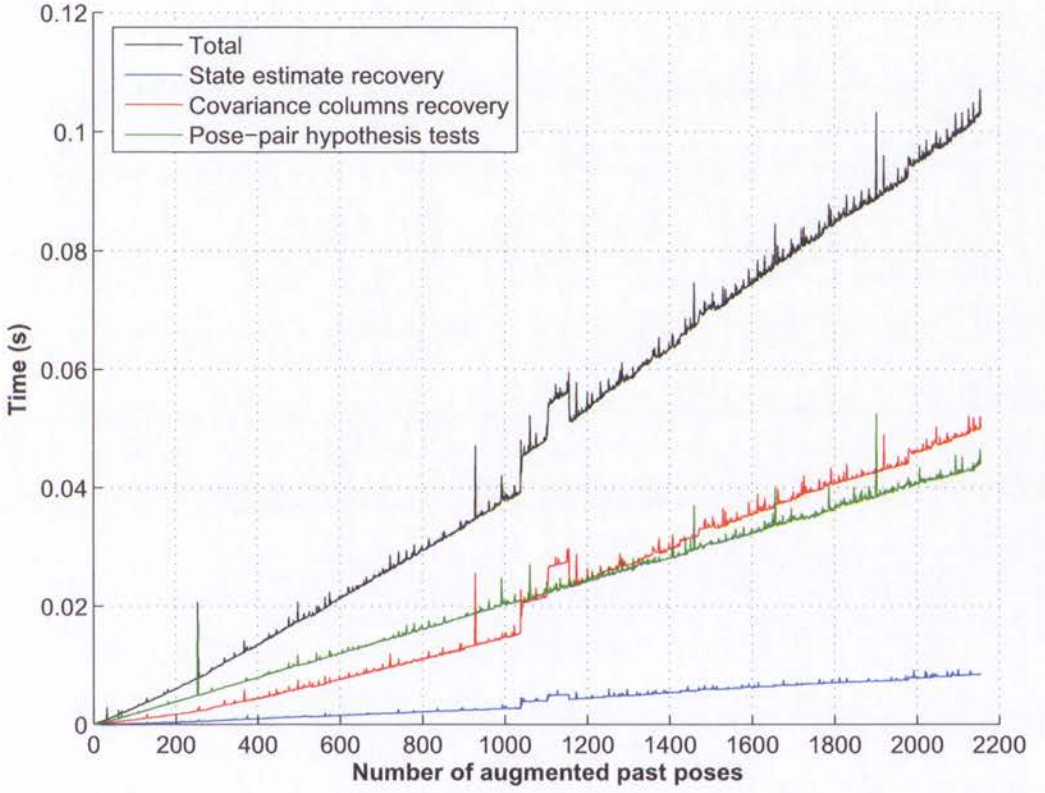


Figure 5.35: Processing times for loop-closure hypothesis generation. The processing time for the pose-pair tests used to identify loop-closure hypotheses grows linearly with the number of poses. The time required to perform the state estimate and covariance column recovery operations is a function of the number of non-zero elements in the Cholesky factor. In general they are $O(n^2)$ operations, however in this experiment their growth is not much worse than linear due to the near linear growth in the number of non-zeros in the Cholesky factor. The processing times in this figure were acquired on a 2.0 GHz Pentium M processor.

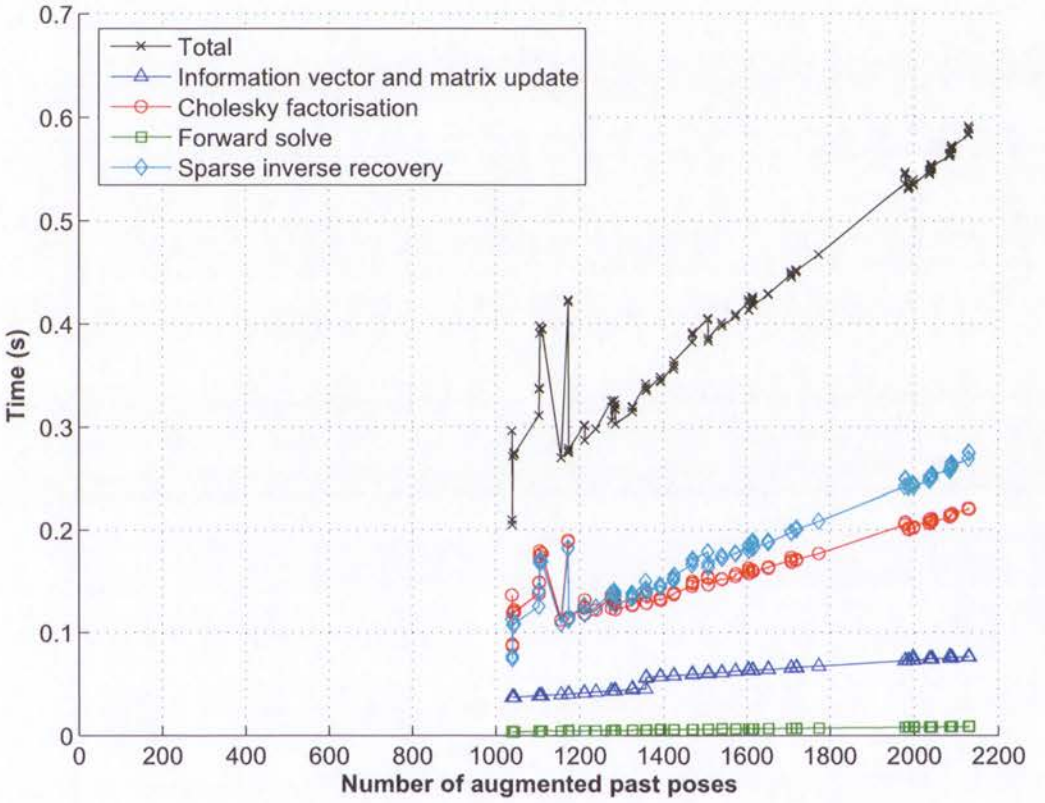


Figure 5.36: Processing times for loop-closure observations. Applying an observation update to the information matrix is a $O(n)$ operation in this application due to the use of a compressed row storage format. In general, the computational complexity of the Cholesky factorisation and sparse inverse recovery operations is $O(n^3)$, and the forward solve operation is $O(n^2)$. In this experiment, the growth in their processing times is not much worse than linear due to the near linear growth in the number of non-zeros in the Cholesky factor.

5.10 Summary

A novel SLAM algorithm based the VAN framework was presented and demonstrated using data acquired by the SeaBED AUV at Ningaloo Marine Park. The stereo relative pose estimation method presented in Chapter 4 has been used to generate loop-closure observations and correct dead-reckoning drift.

The use of Cholesky factorisation modifications to update a decomposition of the information matrix was demonstrated, preventing the need to repeatedly perform the computationally expensive factorisation process each time state estimates and covariances are recovered.

An exact partial state recovery method was presented, allowing prediction and vehicle state observation operations to be performed without corrupting the filter with approximate vehicle state estimates. Through the use of use of Cholesky modifications and an appropriate variable ordering, exact recovery of the vehicle state estimates can be performed in constant time.

An improved method to update conservative pose covariances based on recovery of the sparse inverse was presented. This approach was demonstrated to improve the computational efficiency of the VAN algorithm by reducing the number of loop-closure hypotheses requiring evaluation.

The scalability of the each operation in the VAN algorithm has been investigated. While in the worst-case the complexity of some operations is $O(n^3)$ in the number of augmented poses, processing times for a typical underwater survey experiment suggest an on-line implementation is feasible.

Chapter 6

Conclusions and Future Research

6.1 Introduction

The objective of this thesis has been the development of vision-based navigation techniques for underwater vehicles. Several approaches have been presented, including odometry estimation from the fusion of observations from a monocular camera and a sonar range-finder, odometry estimation using stereo-vision, and a SLAM algorithm using cameras to augment traditional dead-reckoning sensors.

This chapter summarises the contributions of this thesis, and provides suggestions for further research. Section 6.2 provides a summary of contributions towards improving vision-based AUV navigation. Section 6.3 proposes directions for future research.

6.2 Summary of Contributions

Odometry using Sonar and Vision

An algorithm to estimate the odometry of an underwater vehicle through the fusion of sonar and vision observations has been presented. The algorithm has been demonstrated on data collected from a vehicle lacking traditional dead-reckoning sensors such as a DVL.

Novel aspects of this work include:

- The use of a SLAM-style algorithm with a highly dynamic state vector consisting of both temporary vehicle poses and features. A set of temporary natural features tracked using a vision system enables the odometry of the vehicle to be estimated without introducing artificial landmarks or beacons into the environment.

- The use of a sonar range-finder to initialise the depth of features observed in images. Maintaining an estimate of the pose of the vehicle at the time the last image was acquired provides a method to register non-synchronised sonar and vision observations.

Relative Pose Estimation using Stereo-vision

A novel relative pose estimation algorithm using stereo-vision has been presented, and demonstrated in visual odometry and SLAM applications.

Contributions in this work include:

- A survey and evaluation of motion estimation approaches. Algorithms that ignore some or all observation uncertainties were shown to result in biased and noisy estimators.

- A novel approach to efficiently calculate the maximum likelihood motion parameters by registering triangulated 3D feature positions. Summarising stereo-vision observations of a feature with triangulated position estimates and covariances was shown to produce equivalent results to bundle adjustment, while being more computationally efficient due to a more compact observation vector and a less complex observation model.

- A survey and evaluation of outlier rejection frameworks. The iterative outlier rejection approach was found to be unreliable when applied to data containing outliers with large errors, such as those produced by incorrect association of feature observations. The commonly used RANSAC algorithm was shown to accept fewer inliers than could be expected with better motion hypotheses.

- An outlier rejection approach based on a robust estimator. A redescending M-estimator is used, not in its typical role to provide a final motion estimate, but to create a relative pose hypothesis from which an outlier classification test is performed. The maximum likelihood motion parameter estimate and covariance is then calculated from the remaining inlier features. The robust estimator was demonstrated to provide superior motion hypotheses to RANSAC, resulting in an improved outlier classifier and motion estimator.

Simultaneous Localisation and Mapping using Visual Loop-closures

A novel SLAM algorithm based on the VAN framework has been presented. The algorithm has been demonstrated using stereo-vision loop-closure observations to correct drift in the estimated trajectory of an AUV equipped with a DVL.

Contributions include:

- The application of Cholesky modifications to a factorisation of the VAN information matrix. It has been shown that an existing factorisation of the information matrix can be efficiently modified to reflect changes to the filter within prediction and observation operations. If the variable reordering used during the factorisation process is constrained to keep the vehicle states last, the factor modifications required after prediction and vehicle state observation operations can be performed in constant time. Using cholesky modifications avoids performing many computationally expensive factorisation operations, resulting in significant improvements to the efficiency of the VAN filter.

- A method to recover optimal estimates of the current vehicle pose states in constant time. Recovery of the exact vehicle states allows prediction and observation operations to be performed efficiently without corrupting the filter with the approximate estimates used in previous VAN implementations.

- A covariance recovery method based on the 'sparse inverse' matrix. Recovering the covariance of all estimated poses allows less conservative pose covariances to be used when generating loop-closure hypotheses. A reduction in the number of loop-closure hypotheses results in the need to perform fewer computationally expensive image analysis operations.

6.3 Future Research

On-line Demonstration

The vision-based navigation methods presented in this thesis have been demonstrated on data acquired from previous deployments of underwater vehicles. Developing on-line implementations capable of handling real-time constraints is likely to require a significant effort.

A Vision Sensor for Navigation Applications

The cameras used in the experiments presented in this thesis have been selected to meet the needs of human experts such as biologists and geologists who will view the acquired images. The properties of a vision sensor built specifically for navigation applications may be significantly different from those currently used.

Greyscale cameras could be used, since colour is typically ignored when associating features due to large variations that occur under different lighting conditions. A greyscale camera also has the advantage of producing sharper images than a colour camera that interpolates pixel intensities from a Bayer pattern of red, green and blue photosensors.

Cameras that produce smaller images with a larger field of view are likely to be better suited for navigation. Smaller images will improve the efficiency of image processing operations, and a larger field of view allows more features to be observed and tracked over larger distances.

Dedicated hardware could be used to perform feature extraction and calculate the feature descriptions used for association. Several algorithms such as MSER, SIFT and SURF could be implemented to ensure reliable performance in a range of environments. Once feature extraction and description have been performed, images could be discarded to reduce bandwidth and storage requirements.

Camera Calibration and the Assumption of Independent Observations

The assumption of independent visual feature observation errors is only valid if perfect calibration parameters have been acquired. Obtaining a good calibration is difficult, particularly in underwater applications where a camera housing may deform under high pressures. On the Australian SeaBED vehicle, deformation of a temporary camera viewport has been observed in shallow (30 metre) waters. An improved housing design will reduce deformations, however the same effects are likely occur at greater depths.

On-line estimation of the intrinsic camera parameters could be attempted, however the often planar structure of the seafloor may lack the geometric variation required to make the estimation problem observable. Since deformations are likely to be repeatable, perhaps an experiment where the camera-rig is placed inside a pressure chamber with a calibration target could be performed to generate curves for the calibration parameters with varying depth.

The Computational Complexity of SLAM

While the SLAM algorithm presented in this thesis is suitable for many applications, there will be a limit to the number of poses and loop-closure observations that can be handled in real-time.

The most complex operation in the SLAM algorithm is the application of a loop-closure observation to the filter, which requires the estimate and conservative covariances of all

poses to be updated. Perhaps the computational complexity of a loop-closure update could be reduced using sub-maps. Some benefits may also be gained by approximations such as sparsification [27, 34], in which small correlations are ignored.

In this thesis, direct solutions to systems of linear equations have been investigated for state estimate and covariance recovery. Iterative approaches such as the Conjugate Gradient method [27, 95, 105] and multilevel relaxation [35] may provide some benefits.

The size of the estimated state vector could be reduced through intelligent pose management. For example, since a typical AUV deployment involves traversing a predefined survey pattern, poses in areas that will not be revisited could be removed from the filter.

While loop-closure observations are necessary to correct dead-reckoning drift, they also increase the computational complexity of the filter due to fill-in of the Cholesky factor. In situations such as a dense survey of a small region where lots of loop-closure constraints could be generated, many observations are likely to provide little additional information. A principled way to decide which observations should be applied to the filter, based on a trade-off between information gain and computational complexity may be advantageous.

Methods to limit the number of pose-pairs evaluated to generate loop-closure hypotheses should be considered. Perhaps a tree structure could be used to limit the search to poses in the neighbourhood of the current vehicle position.

Combining Visual Odometry and SLAM

In the experiments performed in this thesis, SLAM was performed using a DVL to provide odometry. Combining visual odometry and SLAM could improve the localisation of low-cost vehicles lacking expensive sensors such as a DVL or IMU.

Temporally Stable Features for Repeatable Surveys

An important application for AUVs is monitoring changes to fragile environments such as coral reefs. Ideally this should be performed by repeating the same survey pattern over months or years, ensuring the same locations are observed on each deployment.

A map built from data collected during previous deployments could be used to improve the survey repeatability. The small point features used by the navigation approaches presented in this thesis are unlikely to be stable over long periods of time, and location recognition over a larger spatial scale may be required. Since the shape of the seafloor is likely to be more stable than its appearance, 3D reconstructions from stereo-vision or sonar sensors may form a significant part of a solution to this problem.

Appendix A

Geometry

A.1 Euler Angle Orientation Representation

The orientation of a frame j relative to a frame i is represented by a vector of Euler angles.

$$ {}^{i}\psi_{j} = \begin{bmatrix} {}^{i}\phi_{j} \\ {}^{i}\theta_{j} \\ {}^{i}\psi_{j} \end{bmatrix} \tag{A.1} $$

where ϕ represent a rotation around the x-axis, θ represents a rotation around the y-axis, and ψ a rotation around the z-axis. In this thesis, a XYZ ordering convention is used.

The Euler angle orientation representation suffers from a problem known as gimbal lock, which occurs when the second rotation causes the final axis of rotation to be the same as the first, causing an inability to rotate the frame around a third independent axis. In the selected Euler angle convention, Gimbal lock occurs when $\theta = \pm\frac{\pi}{2}$. This should not pose a problem, since the vehicles used in the experiments presented in this thesis do not operate at such high pitch angles.

A.2 Rotation Matrices

The matrix that rotates frame i to be aligned with frame j is given by

$$
\begin{aligned}
{}^{j}_{i}\mathbf{R} &= \mathbf{R}_x[{}^{i}\phi_j]\mathbf{R}_y[{}^{i}\theta_j]\mathbf{R}_z[{}^{i}\psi_j] \qquad \text{(A.2)}\\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos{}^{i}\phi_j & \sin{}^{i}\phi_j \\ 0 & -\sin{}^{i}\phi_j & \cos{}^{i}\phi_j \end{bmatrix} \begin{bmatrix} \cos{}^{i}\theta_j & 0 & -\sin{}^{i}\theta_j \\ 0 & 1 & 0 \\ \sin{}^{i}\theta_j & 0 & \cos{}^{i}\theta_j \end{bmatrix} \begin{bmatrix} \cos{}^{i}\psi_j & \sin{}^{i}\psi_j & 0 \\ -\sin{}^{i}\psi_j & \cos{}^{i}\psi_j & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} c({}^{i}\psi_j)c({}^{i}\theta_j) & s({}^{i}\psi_j)c({}^{i}\theta_j) & -s({}^{i}\theta_j) \\ c({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) - s({}^{i}\psi_j)c({}^{i}\phi_j) & s({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) + c({}^{i}\psi_j)c({}^{i}\phi_j) & c({}^{i}\theta_j)s({}^{i}\phi_j) \\ c({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) + s({}^{i}\psi_j)s({}^{i}\phi_j) & s({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) - c({}^{i}\psi_j)s({}^{i}\phi_j) & c({}^{i}\theta_j)c({}^{i}\phi_j) \end{bmatrix}
\end{aligned}
$$

where $s(\cdot)$ and $c(\cdot)$ have been used to represent sine and cosine respectively.

Jacobians of the rotation matrix with respect to the Euler angles are

$$
\frac{\partial({}^{j}_{i}\mathbf{R})}{\partial({}^{i}\phi_j)} = \begin{bmatrix} 0 & 0 & 0 \\ c({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) + s({}^{i}\psi_j)s({}^{i}\phi_j) & s({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) - c({}^{i}\psi_j)s({}^{i}\phi_j) & c({}^{i}\theta_j)c({}^{i}\phi_j) \\ -c({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) + s({}^{i}\psi_j)c({}^{i}\phi_j) & -s({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) - c({}^{i}\psi_j)c({}^{i}\phi_j) & -c({}^{i}\theta_j)s({}^{i}\phi_j) \end{bmatrix} \qquad \text{(A.3)}
$$

$$
\frac{\partial({}^{i}_{j}\mathbf{R})}{\partial({}^{i}\theta_j)} = \begin{bmatrix} -c({}^{i}\psi_j)s({}^{i}\theta_j) & -s({}^{i}\psi_j)s({}^{i}\theta_j) & -c({}^{i}\theta_j) \\ c({}^{i}\psi_j)c({}^{i}\theta_j)s({}^{i}\phi_j) & s({}^{i}\psi_j)c({}^{i}\theta_j)s({}^{i}\phi_j) & -s({}^{i}\theta_j)s({}^{i}\phi_j) \\ c({}^{i}\psi_j)c({}^{i}\theta_j)c({}^{i}\psi_j) & s({}^{i}\psi_j)c({}^{i}\theta_j)c({}^{i}\phi_j) & -s({}^{i}\theta_j)c({}^{i}\phi_j) \end{bmatrix} \qquad \text{(A.4)}
$$

$$
\frac{\partial({}^{i}_{j}\mathbf{R})}{\partial({}^{i}\psi_j)} = \begin{bmatrix} -s({}^{i}\psi_j)c({}^{i}\theta_j) & c({}^{i}\psi_j)c({}^{i}\theta_j) & 0 \\ -s({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) - c({}^{i}\psi_j)c({}^{i}\theta_j) & c({}^{i}\psi_j)s({}^{i}\theta_j)s({}^{i}\phi_j) - s({}^{i}\psi_j)c({}^{i}\phi_j) & 0 \\ -s({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) + c({}^{i}\psi_j)s({}^{i}\phi_j) & c({}^{i}\psi_j)s({}^{i}\theta_j)c({}^{i}\phi_j) + s({}^{i}\psi_j)s({}^{i}\phi_j) & 0 \end{bmatrix} \qquad \text{(A.5)}
$$

The matrix that rotates frame j back to be aligned with frame i is given by

$$
\begin{aligned}
{}^{i}_{j}\mathbf{R} &= {}^{j}_{i}\mathbf{R}^{-1} \qquad \text{(A.6)}\\
&= {}^{j}_{i}\mathbf{R}^{\mathsf{T}}
\end{aligned}
$$

Similarly

$$\frac{\partial({}^j_i\mathbf{R})}{\partial({}^i\phi_j)} = \left(\frac{\partial({}^i_j\mathbf{R})}{\partial({}^i\phi_j)}\right)^{\mathsf{T}} \tag{A.7}$$

$$\frac{\partial({}^j_i\mathbf{R})}{\partial({}^i\theta_j)} = \left(\frac{\partial({}^i_j\mathbf{R})}{\partial({}^i\theta_j)}\right)^{\mathsf{T}} \tag{A.8}$$

$$\frac{\partial({}^j_i\mathbf{R})}{\partial({}^i\psi_j)} = \left(\frac{\partial({}^i_j\mathbf{R})}{\partial({}^i\psi_j)}\right)^{\mathsf{T}} \tag{A.9}$$

The Euler angle can be extracted from the rotation matrix orientation representation with the function

$$\begin{aligned} {}^i\psi_j &= \mathbf{matrixToEuler}[{}^j_i\mathbf{R}] \\ &= \begin{bmatrix} \text{atan2}({}^j_i\mathbf{R}_{3,1}\sin{}^i\psi_j - {}^j_i\mathbf{R}_{3,2}\cos{}^i\psi_j, -{}^j_i\mathbf{R}_{2,1}\sin{}^i\psi_j + {}^j_i\mathbf{R}_{2,2}\cos{}^i\psi_j) \\ \text{atan2}(-{}^j_i\mathbf{R}_{1,3}, {}^j_i\mathbf{R}_{1,1}\cos{}^i\psi_j + {}^j_i\mathbf{R}_{1,2}\sin{}^i\psi_j) \\ \text{atan2}({}^j_i\mathbf{R}_{1,2}, {}^j_i\mathbf{R}_{1,1}) \end{bmatrix} \end{aligned} \tag{A.10}$$

A.3 Converting Local Body Rotation Rates to Euler Rates

The local rotations of frame j are represented by

$$\boldsymbol{\omega}_j = \begin{bmatrix} p_j \\ q_j \\ r_j \end{bmatrix} \tag{A.11}$$

where p, q and r are the rotation rates around frame j's x-axis, y-axis and z-axis respectively.

The local body rates of frame j can be converted to Euler rates in frame i by

$${}^i\dot{\boldsymbol{\psi}}_j = {}^i_j\mathbf{E}\boldsymbol{\omega}_j \tag{A.12}$$

where

$${}^i_j\mathbf{E} = \begin{bmatrix} 1 & \sin({}^i\phi_j)\tan({}^i\theta_j) & \cos({}^i\phi_j)\tan({}^i\theta_j) \\ 0 & \cos({}^i\phi_j) & -\sin({}^i\phi_j) \\ 0 & \sin({}^i\phi_j)\sec({}^i\theta_j) & \cos({}^i\phi_j)\sec({}^i\theta_j) \end{bmatrix}. \tag{A.13}$$

The derivatives of the conversion matrix with respect to the Euler angles representing the

orientation of the body are

$$\frac{\partial({}^i_j\mathbf{E})}{\partial({}^i\phi_j)} = \begin{bmatrix} 0 & \cos({}^i\phi_j)\tan({}^i\theta_j) & -\sin({}^i\phi_j)\tan({}^i\theta_j) \\ 0 & -\sin({}^i\phi_j) & -\cos({}^i\phi_j) \\ 0 & \cos({}^i\phi_j)\sec({}^i\theta_j) & -\sin({}^i\phi_j)\sec({}^i\theta_j) \end{bmatrix} \tag{A.14}$$

$$\frac{\partial({}^i_j\mathbf{E})}{\partial({}^i\theta_j)} = \begin{bmatrix} 0 & \sin({}^i\phi_j)\sec^2({}^i\theta_j) & \cos({}^i\phi_j)\sec^2({}^i\theta_j) \\ 0 & 0 & 0 \\ 0 & \sin({}^i\phi_j)\sec\theta\tan({}^i\theta_j) & \cos({}^i\phi_j)\sec({}^i\theta_j)\tan({}^i\theta_j) \end{bmatrix} \tag{A.15}$$

$$\frac{\partial({}^i_j\mathbf{E})}{\partial({}^i\psi_j)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{A.16}$$

A.4 Pose Representation

The pose of frame j relative to frame i is represented by a vector containing translation and Euler angles orientation states

$$\begin{aligned} {}^i\mathbf{p}_j &= \begin{bmatrix} {}^i\mathbf{t}_j \\ {}^i\psi_j \end{bmatrix} \\ &= \begin{bmatrix} {}^ix_j & {}^iy_j & {}^iz_j & {}^i\phi_j & {}^i\theta_j & {}^i\psi_j \end{bmatrix}^\mathsf{T} \end{aligned} \tag{A.17}$$

A.5 Reference Frame Conversions

The position of x relative to frame i can be transformed to be relative to frame j using the equations

$${}^j\mathbf{t}_x = {}^j_i\mathbf{R}\,{}^i\mathbf{t}_x + {}^j\mathbf{t}_i \tag{A.18}$$

$${}^j\mathbf{t}_x = {}^j_i\mathbf{R}({}^i\mathbf{t}_x - {}^i\mathbf{t}_j) \tag{A.19}$$

Equation A.18 is useful when ${}^j\mathbf{p}_i$ (the pose of frame i relative to j) is known, while A.19 is useful when ${}^i\mathbf{p}_j$ (the pose of frame j relative to i) is known.

The Jacobian of the transformed coordinates ${}^j\mathbf{t}_x$ with respect to the original coordinates is ${}^i\mathbf{t}_x$

$$\frac{\delta{}^j\mathbf{t}_x}{\delta{}^i\mathbf{t}_x} = {}^j_i\mathbf{R} \tag{A.20}$$

The Jacobian of the transformed coordinates ${}^j\mathbf{t}_x$ with respect to the pose of frame i relative to frame j is

$$\frac{\delta^j \mathbf{t}_x}{\delta^j \mathbf{p}_i} = \begin{bmatrix} \mathbf{I} & \frac{\delta^j_i \mathbf{R}}{\delta^j \phi_i} {}^i\mathbf{t}_x & \frac{\delta^j_i \mathbf{R}}{\delta^j \theta_i} {}^i\mathbf{t}_x & \frac{\delta^j_i \mathbf{R}}{\delta^j \psi_i} {}^i\mathbf{t}_x \end{bmatrix} \tag{A.21}$$

The Jacobian of the transformed coordinates ${}^j\mathbf{t}_x$ with respect to the pose of frame j relative to frame i is

$$\frac{\delta^j \mathbf{t}_x}{\delta^i \mathbf{p}_j} = \begin{bmatrix} -{}^j_i\mathbf{R} & \frac{\delta^j_i \mathbf{R}}{\delta^i \phi_j}({}^i\mathbf{t}_x - {}^i\mathbf{t}_j) & \frac{\delta^j_i \mathbf{R}}{\delta^i \theta_j}({}^i\mathbf{t}_x - {}^i\mathbf{t}_j) & \frac{\delta^j_i \mathbf{R}}{\delta^i \psi_j}({}^i\mathbf{t}_x - {}^i\mathbf{t}_j) \end{bmatrix} \tag{A.22}$$

A.6 Pose Composition Operations

Two dimensional pose inverse, head to tail and tail to tail composition operations were first presented in [101]. The six degree of freedom versions listed here have been adapted from [27]. To illustrate the pose composition operations, three reference frames i, j and k are shown in Figure A.1.

Inverse

If the pose of frame j relative to i is known, the pose of frame i relative to j can be obtained using the pose inverse operation defined by

$$\begin{aligned} {}^j\mathbf{p}_i &= \ominus {}^i\mathbf{p}_j \\ &= \begin{bmatrix} -{}^j_i\mathbf{R}\,{}^i\mathbf{t}_j \\ \mathbf{matrixToEuler}[{}^j_i\mathbf{R}] \end{bmatrix} \end{aligned} \tag{A.23}$$

The Jacobian of the pose inverse operation is

$$\begin{aligned} \mathbf{J}_\ominus &= \frac{\partial^j \mathbf{p}_i}{\partial^i \mathbf{p}_j} \\ &= \begin{bmatrix} -{}^j_i\mathbf{R} & \mathbf{N} \\ \mathbf{0}_{(3\times 3)} & \mathbf{Q} \end{bmatrix} \end{aligned} \tag{A.24}$$

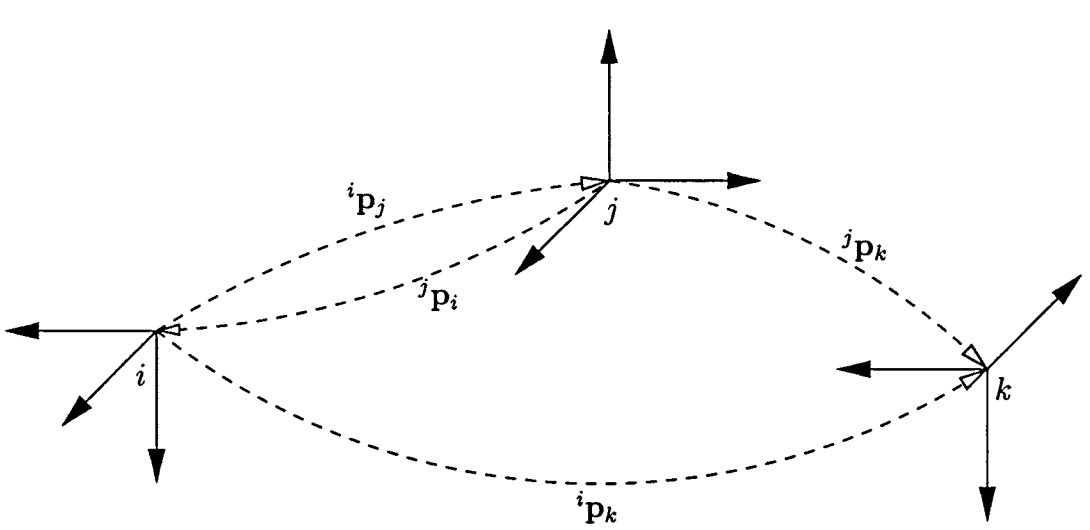


Figure A.1: Reference frames demonstrating pose composition operations. The coordinate systems of three reference frames i, j and k are shown, along with relative poses represented by dotted lines.

where

$$\mathbf{N} = \begin{bmatrix} 0 & -{}^j_i\mathbf{R}_{1,3}({}^ix_j\cos{}^i\phi_j + {}^iy_j\sin{}^i\phi_j) + {}^iz_j\cos{}^i\theta_j & {}^j_i\mathbf{R}_{1,2}{}^ix_j - {}^j_i\mathbf{R}_{1,1}{}^iy_j \\ {}^jz_i & -{}^j_i\mathbf{R}_{2,3}({}^ix_j\cos{}^i\phi_j + {}^iy_j\sin{}^i\phi_j) + {}^iz_j\sin{}^i\theta_j\sin{}^i\phi_j & {}^j_i\mathbf{R}_{2,2}{}^ix_j - {}^j_i\mathbf{R}_{2,1}{}^iy_j \\ -{}^jy_i & -{}^j_i\mathbf{R}_{3,3}({}^ix_j\cos{}^i\phi_j + {}^iy_j\sin{}^i\phi_j) + {}^iz_j\cos{}^i\theta_j\cos{}^i\phi_j & {}^j_i\mathbf{R}_{3,2}{}^ix_j - {}^j_i\mathbf{R}_{3,1}{}^iy_j \end{bmatrix} \tag{A.25}$$

$$\mathbf{Q} = \frac{1}{(1-{}^i_j\mathbf{R}^2_{1,3})}\begin{bmatrix} -{}^j_i\mathbf{R}_{1,1} & -{}^j_i\mathbf{R}_{2,1}\cos{}^i\phi_j & {}^j_i\mathbf{R}_{3,1}{}^j_i\mathbf{R}_{3,3} \\ {}^j_i\mathbf{R}_{2,1}\sqrt{(1-{}^i_j\mathbf{R}^2_{1,3})} & -{}^j_i\mathbf{R}_{3,3}\cos{}^i\phi_j\sqrt{(1-{}^j_i\mathbf{R}^2_{3,1})} & {}^j_i\mathbf{R}_{3,2}\sqrt{(1-{}^j_i\mathbf{R}^2_{3,1})} \\ {}^j_i\mathbf{R}_{3,1} & -{}^j_i\mathbf{R}_{3,2}\cos{}^i\phi_j & -{}^j_i\mathbf{R}_{3,3} \end{bmatrix} \tag{A.26}$$

Head To Tail

If the pose of frame j relative to i, and the pose of frame k relative to j are known, the pose of k relative to i can be obtained using the pose head to tail operation. As the name of the operation suggests, in Figure A.1 the head of the arrow representing ${}^i\mathbf{p}_j$ is located at the tail of the arrow of ${}^i\mathbf{p}_k$.

The pose head to tail operation is defined as

$$
\begin{aligned}
{}^{i}\mathbf{p}_k &= {}^{i}\mathbf{p}_j \oplus {}^{j}\mathbf{p}_k \\
&= \begin{bmatrix} {}^{i}_{j}\mathbf{R}\,{}^{j}\mathbf{t}_k + {}^{i}\mathbf{t}_j \\ \mathbf{matrixToEuler}[{}^{i}_{k}\mathbf{R}] \end{bmatrix}
\end{aligned} \tag{A.27}
$$

where ${}^{i}_{k}\mathbf{R} = {}^{i}_{j}\mathbf{R}\,{}^{j}_{k}\mathbf{R}$.

The Jacobian of the head to tail operation with respect to ${}^{i}\mathbf{p}_j$ and ${}^{j}\mathbf{p}_k$ is given by

$$
\begin{aligned}
\mathbf{J}_{\oplus} &= \frac{\partial {}^{i}\mathbf{p}_k}{\partial({}^{i}\mathbf{p}_j, {}^{j}\mathbf{p}_k)} \\
&= \begin{bmatrix} \mathbf{J}_{\oplus 1} & \mathbf{J}_{\oplus 2} \end{bmatrix} \\
&= \left[\begin{array}{cc|cc} \mathbf{I}_{(3\times 3)} & \mathbf{M} & {}^{i}_{j}\mathbf{R} & \mathbf{0}_{(3\times 3)} \\ \mathbf{0}_{(3\times 3)} & \mathbf{K}_1 & \mathbf{0}_{(3\times 3)} & \mathbf{K}_2 \end{array}\right]
\end{aligned} \tag{A.28}
$$

where

$$
\mathbf{M} = \begin{bmatrix} {}^{i}_{j}\mathbf{R}_{1,3}\,{}^{j}y_k - {}^{i}_{j}\mathbf{R}_{1,2}\,{}^{j}z_k & ({}^{i}z_k - {}^{i}z_j)\cos{}^{i}\psi_j & -({}^{i}y_k - {}^{i}y_j) \\ {}^{i}_{j}\mathbf{R}_{2,3}\,{}^{j}y_k - {}^{i}_{j}\mathbf{R}_{2,2}\,{}^{j}z_k & ({}^{i}z_k - {}^{i}z_j)\sin{}^{i}\psi_j & ({}^{i}x_k - {}^{i}x_j) \\ {}^{i}_{j}\mathbf{R}_{3,3}\,{}^{j}y_k - {}^{i}_{j}\mathbf{R}_{3,2}\,{}^{j}z_k & -{}^{j}x_k\cos{}^{i}\theta_j - ({}^{j}y_k\sin{}^{i}\theta_j + {}^{j}z_k\cos{}^{i}\theta_j)\sin{}^{i}\theta_j & 0 \end{bmatrix} \tag{A.29}
$$

$$
\mathbf{K1} = \begin{bmatrix} \cos{}^{i}\theta_j\cos({}^{i}\psi_k - {}^{i}\psi_j)\sec{}^{i}\theta_k & \sin({}^{i}\psi_k - {}^{i}\psi_j)\sec{}^{i}\theta_k & 0 \\ -\cos{}^{i}\theta_j\sin({}^{i}\psi_k - {}^{i}\psi_j) & \cos({}^{i}\psi_k - {}^{i}\psi_j) & 0 \\ ({}^{j}_{k}\mathbf{R}_{1,2}\sin{}^{i}\phi_k + {}^{j}_{k}\mathbf{R}_{1,3}\cos{}^{i}\phi_k)\sec{}^{i}\theta_k & \sin({}^{i}\psi_k - {}^{i}\psi_j)\tan{}^{i}\theta_k & 1 \end{bmatrix} \tag{A.30}
$$

$$
\mathbf{K2} = \begin{bmatrix} 1 & \sin({}^{i}\psi_k - {}^{j}\psi_k)\tan{}^{i}\theta_k & ({}^{j}_{k}\mathbf{R}_{1,3}\sin{}^{i}\phi_k + {}^{j}_{k}\mathbf{R}_{2,3}\cos{}^{i}\phi_k)\sec{}^{i}\theta_k \\ 0 & \cos({}^{i}\psi_k - {}^{j}\psi_k) & -\cos{}^{j}\theta_k\sin({}^{i}\psi_k - {}^{j}\psi_k) \\ 0 & \sin({}^{i}\psi_k - {}^{j}\psi_k)\sec{}^{i}\theta_k & \cos{}^{j}\theta_k\cos({}^{i}\psi_k - {}^{j}\psi_k)\sec{}^{i}\theta_k \end{bmatrix} \tag{A.31}
$$

Tail To Tail

If the poses of frame j and k relative to frame i are known, the pose of k relative to j can be obtained using the tail to tail operation. As the name of the operation suggests, the tails of the arrows representing ${}^{i}\mathbf{p}_j$ and ${}^{i}\mathbf{p}_k$ in Figure A.1 are located together.

The pose tail to tail operation is defined as

$$
\begin{aligned}
{}^{j}\mathbf{p}_k &= \ominus {}^{i}\mathbf{p}_j \oplus {}^{i}\mathbf{p}_k \\
&= {}^{j}\mathbf{p}_i \oplus {}^{i}\mathbf{p}_k
\end{aligned} \tag{A.32}
$$

The Jacobian of the tail to tail operation with respect to each pose is

$$
\begin{aligned}
{}_{\ominus}\mathbf{J}_{\oplus} &= \frac{\partial {}^{j}\mathbf{p}_k}{\partial({}^{i}\mathbf{p}_j, {}^{i}\mathbf{p}_k)} \\
&= \frac{\partial {}^{j}\mathbf{p}_k}{\partial({}^{j}\mathbf{p}_i, {}^{i}\mathbf{p}_k)} \cdot \frac{\partial({}^{j}\mathbf{p}_i, {}^{i}\mathbf{p}_k)}{\partial({}^{i}\mathbf{p}_j, {}^{i}\mathbf{p}_k)} \\
&= \mathbf{J}_{\oplus} \cdot \begin{bmatrix} \mathbf{J}_{\ominus} & \mathbf{0}_{(6\times6)} \\ \mathbf{0}_{(6\times6)} & \mathbf{I}_{(6\times6)} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{J}_{\oplus 1}\mathbf{J}_{\ominus} & \mathbf{J}_{\oplus 2} \end{bmatrix}
\end{aligned}
\tag{A.33}
$$

Appendix B

Platforms, Sensors and Models

This appendix describes the two underwater vehicles that have been used in experiments presented in this thesis. A vehicle model for both platforms is presented along with observation models for their sensors.

B.1 Platforms

B.1.1 The Oberon ROV

Oberon, shown in Figure B.1, is a mid-sized ROV designed and built at the Australian Centre for Field Robotics to test novel sensing and control methods. A summary of the vehicle and sensor specifications are provided in Table B.1. A detailed description of the vehicle is presented in [113].

The vehicle is designed around two connected waterproof chambers containing electronics and sensors. A 30 metre tether supplies power from a supporting ship and enable remote control of the vehicle. A video link to a forward-looking camera is provided to monitor the vehicle, and an Ethernet connection is used for communications.

Oberon's sensor suite consists of a depth sensor, a fibre-optic gyroscope measuring the vehicle's yaw rate, two low-frequency scanning sonars, an integrated tilt and compass sensor and a colour CCD camera oriented to look down towards the seafloor. Propulsion is performed by five thrusters, three of which are oriented vertically to control of the vehicle's roll, pitch and vertical speed, while the remaining two are oriented horizontally to control yaw and forwards velocity.

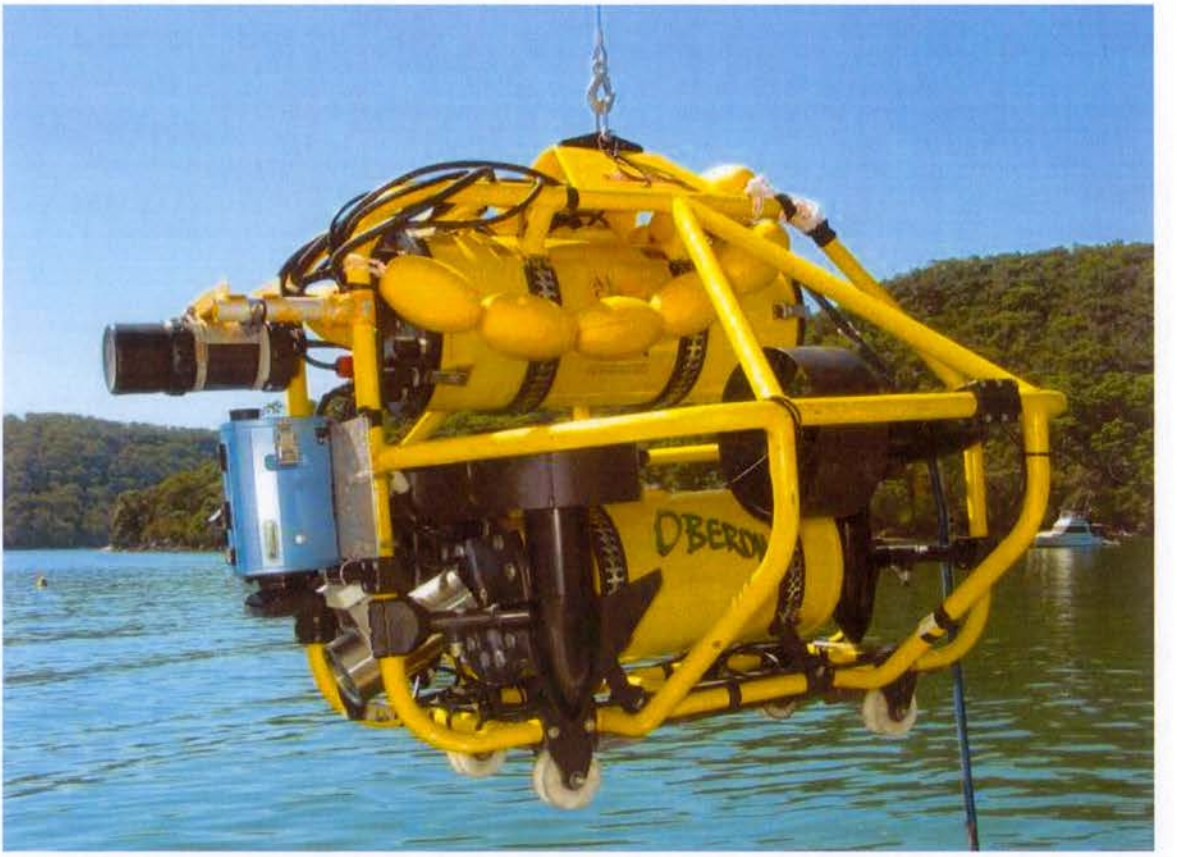


Figure B.1: The Oberon ROV

Maximum depth	20 m
Size	1.2 m (L) × 1.0 m (H) × 1.2 m (W)
Mass	110 kg
Power	Supplied through tether
Propulsion	Five thrusters (3 horizontal, 2 vertical)
Depth sensor	Data Instruments 30 psig pressure transducer
Cameras	Sony 720×576 colour CCD (down-looking)
	Pulnix color CCD (forward-looking)
Imaging sonars	Tritech SeaKing 600/1200 kHz pencil-beam
	Imagenex 640 kHz fan-beam
Attitude	PNI TCM2 magnetometer/tilt sensor
Yaw-rate	Andrews fibre-optic gyroscope

Table B.1: Specifications of the Oberon ROV

B.1.2 The SeaBED AUV

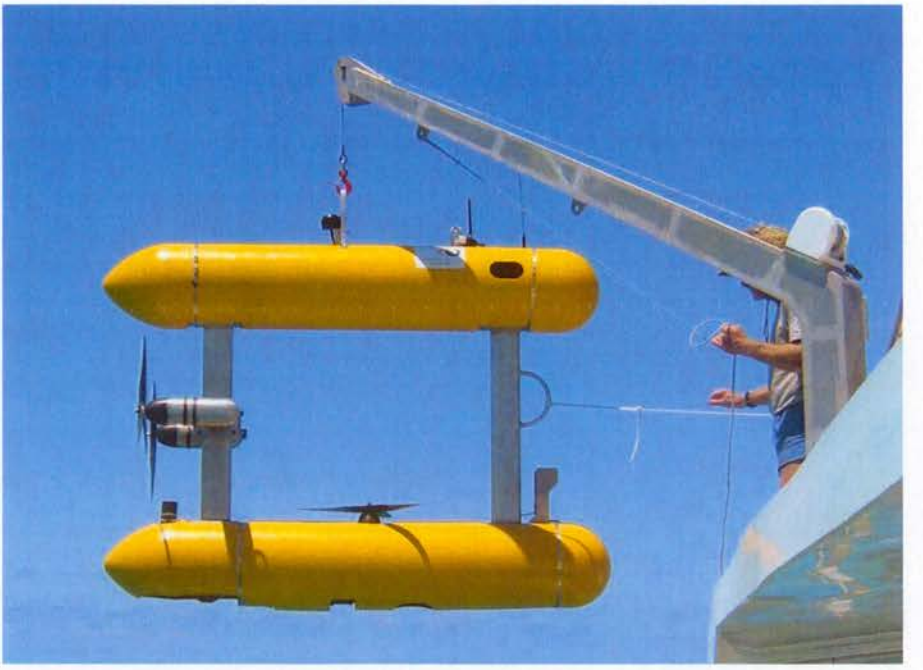
The SeaBED AUV, shown in Figure B.2, was deigned and built at the Woods Hole Oceanographic Institution for high resolution optical and acoustic sensing. The original SeaBED vehicle is described in [98, 99]. Specifications for the slightly modified Australian SeaBED are listed in Table B.2.

The vehicle design consists of two 1.9 metre long hulls arranged vertically 1.1 metres apart. The top hull contains syntactic foam flotation and an electronics housing, while the lower hull contains batteries and sensors. This design provides a naturally stable platform in roll and pitch, while three thrusters allow control of depth, heading, and forward velocity.

The main navigation sensor is a DVL. Stereo cameras and a mechanically scanned sonar image the seafloor.

Maximum depth	700 m
Maximum speed	1.2 m/s
Size	2.0 m (L) × 1.5 m (H) × 1.5 m (W)
Mass	200 kg
Power	1.6 kWh Li-ion battery pack
Propulsion	Three 150 W brushless DC thrusters
Depth sensor	Paroscientific pressure sensor
Conductivity and temperature	Seabird 37SBI
DVL	RDI 1200 kHz Navigator ADCP
Camera	Stereo Prosilica 12bit 1360×1024 CCD
Lighting	Two PerkinElmer MVS-5000 strobes
Imaging sonar	Tritech SeaKing DFS
Obstacle avoidance sonar	Imagenex model 852
Surface communications	Freewave RF Modem and wireless Ethernet
GPS	Ashtech receiver

Table B.2: Specifications of the Australian SeaBED AUV



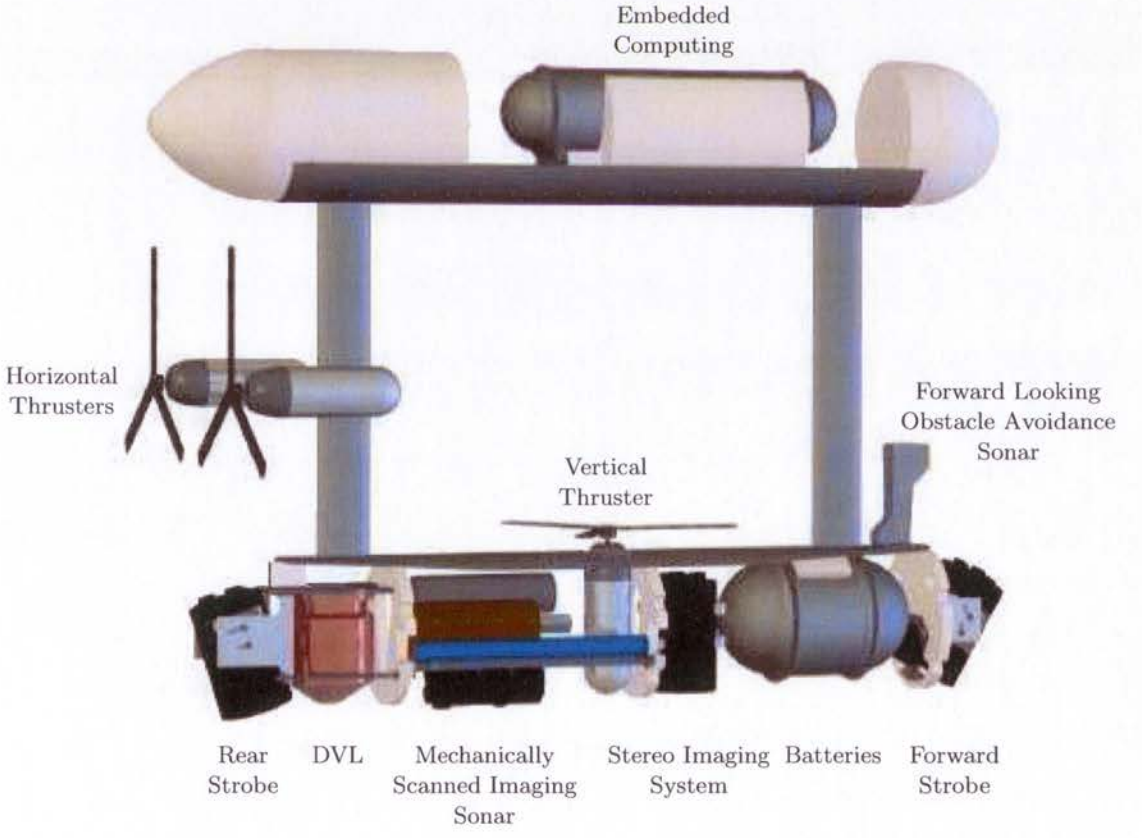


Figure B.2: The SeaBED AUV

B.2 Vehicle Models

B.2.1 Continuous Vehicle Model

This section details the development of a continuous vehicle model of the form

$$\dot{\mathbf{x}}_v(t) = \boldsymbol{f}_v[\mathbf{x}_v(t)] + \boldsymbol{G}_v(t)\boldsymbol{w}_v(t) \tag{B.1}$$

where $\mathbf{x}_v(t)$ is the vector of vehicle states, $\dot{\mathbf{x}}_v(t)$ is a vector of vehicle states derivatives, $\boldsymbol{f}_v[\cdot]$ is the vehicle state transition function, $\boldsymbol{G}_v(t)$ is the noise transition matrix and $\boldsymbol{w}_v(t)$ is the vector of the vehicle model errors, which is assumed to have covariance $\boldsymbol{Q}_v(t)$.

A constant velocity model is used, producing the continuous vehicle model

$$\begin{bmatrix} {}^n\dot{\mathbf{t}}_v(t) \\ {}^n\dot{\boldsymbol{\psi}}_v(t) \\ {}^v\dot{\mathbf{v}}_v(t) \\ \dot{\boldsymbol{\omega}}_v(t) \end{bmatrix} = \begin{bmatrix} {}^n_v\mathbf{R}(t)\,{}^n\mathbf{t}_v(t) \\ {}^n_v\mathbf{E}(t)\,{}^n\boldsymbol{\psi}_v(t) \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\dot{\mathbf{v}}}(t) \\ \mathbf{w}_{\dot{\boldsymbol{\omega}}}(t) \end{bmatrix} \tag{B.2}$$

in which $\mathbf{w}_{\dot{\mathbf{v}}}(t)$ and $\mathbf{w}_{\dot{\boldsymbol{\omega}}}(t)$ represent translational and angular acceleration disturbances to the model at time t.

The continuous model covariance has the form

$$\boldsymbol{Q}_v(t) = \begin{bmatrix} \boldsymbol{\sigma}^2_{\dot{\mathbf{v}}}(t) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}^2_{\dot{\boldsymbol{\omega}}}(t) \end{bmatrix} \tag{B.3}$$

in which

$$\boldsymbol{\sigma}^2_{\dot{\mathbf{v}}}(t) = \begin{bmatrix} \sigma^2_{\dot{v}_x}(t) & 0 & 0 \\ 0 & \sigma^2_{\dot{v}_y}(t) & 0 \\ 0 & 0 & \sigma^2_{\dot{v}_z}(t) \end{bmatrix} \tag{B.4}$$

and

$$\boldsymbol{\sigma}^2_{\dot{\boldsymbol{\omega}}}(t) = \begin{bmatrix} \sigma^2_{\dot{p}}(t) & 0 & 0 \\ 0 & \sigma^2_{\dot{q}}(t) & 0 \\ 0 & 0 & \sigma^2_{\dot{r}}(t) \end{bmatrix} \tag{B.5}$$

B.2.2 Discrete Vehicle Model

To derive a discreet vehicle model, the continuous model will be approximated with a function of the form

$$\dot{\mathbf{x}}_v(t) = \boldsymbol{F}(t)\mathbf{x}(t) + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{B.6}$$

Assuming the change in the orientation states over the sampling period $[t_{k-1}, t_k)$ is small, a suitable approximation to the continuous vehicle model of Equation B.2 is

$$\begin{bmatrix} {}^n\dot{\mathbf{t}}_v(t) \\ {}^n\dot{\boldsymbol{\psi}}_v(t) \\ {}^v\dot{\mathbf{v}}_v(t) \\ \dot{\boldsymbol{\omega}}_v(t) \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & {}^n_v\mathbf{R}(t_{k-1}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & {}^n_v\mathbf{E}(t_{k-1}) \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^n\mathbf{t}_v(t) \\ {}^n\boldsymbol{\psi}_v(t) \\ {}^v\mathbf{v}_v(t) \\ \boldsymbol{\omega}_v(t) \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\dot{\mathbf{v}}}(t) \\ \mathbf{w}_{\dot{\boldsymbol{\omega}}}(t) \end{bmatrix} \tag{B.7}$$

Models in the form of Equation B.6 have a solution

$$\mathbf{x}(t_k) = \mathbf{F}(t_k)\mathbf{x}(t_{k-1}) + \mathbf{v}(t_k) \tag{B.8}$$

where

$$\mathbf{F}(t_k) = \boldsymbol{\Phi}[t_k, t_{k-1}] \tag{B.9}$$

$$\mathbf{v}(t_k) = \int_{t_{k-1}}^{t_k} \boldsymbol{\Phi}[t_k, \tau]\boldsymbol{G}(\tau)\mathbf{v}(\tau)d\tau \tag{B.10}$$

$$\mathbf{Q}(t_k) = \int_{t_{k-1}}^{t_k} \boldsymbol{\Phi}[t_k, \tau]\boldsymbol{G}(\tau)\mathbf{Q}(\tau)\boldsymbol{G}^{\mathsf{T}}(\tau)\boldsymbol{\Phi}^{\mathsf{T}}[t_k, \tau]d\tau \tag{B.11}$$

and $\boldsymbol{\Phi}[\cdot, \cdot]$ is the state transition matrix satisfying the matrix differential equation

$$\dot{\boldsymbol{\Phi}}[t, t_0] = \boldsymbol{F}(t)\boldsymbol{\Phi}[t, t_0], \qquad \boldsymbol{\Phi}[t_0, t_0] = 1 \tag{B.12}$$

For constant $\boldsymbol{F}(t)$, the state transition matrix can be calculated with

$$\boldsymbol{\Phi}[t, t_0] = e^{\boldsymbol{F}(t)\Delta t} \tag{B.13}$$

For the approximated continuous model in Equation B.7, the discreet model components

are

$$\mathbf{F}_v(t_k) = \begin{bmatrix} \mathbf{I} & \mathbf{0} & {}^n_v\mathbf{R}(t_{k-1})\Delta t(t_k) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & {}^n_v\mathbf{E}(t_{k-1})\Delta t(t_k) \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{B.14}$$

$$\mathbf{v}_v(t_k) = \begin{bmatrix} {}^n_v\mathbf{R}(t_{k-1})\mathbf{w}_{\dot{\mathbf{v}}}(t_k)\frac{\Delta t(t_k)^2}{2} \\ {}^n_v\mathbf{E}(t_{k-1})\mathbf{w}_{\dot{\boldsymbol{\omega}}}(t_k)\frac{\Delta t(t_k)^2}{2} \\ \mathbf{w}_{\dot{\mathbf{v}}}(t_k)\Delta t(t_k) \\ \mathbf{w}_{\dot{\boldsymbol{\omega}}}(t_k)\Delta t(t_k) \end{bmatrix} \tag{B.15}$$

$$\mathbf{Q}_v(t_k) = \begin{bmatrix} {}^n_v\mathbf{R}(t_{k-1})\sigma^2_{\dot{\mathbf{v}}}{}^n_v\mathbf{R}^\mathsf{T}(t_{k-1})\frac{\Delta t(t_k)^3}{3} & \mathbf{0} & {}^n_v\mathbf{R}(t_{k-1})\sigma^2_{\dot{\mathbf{v}}}\frac{\Delta t(t_k)^2}{2} & \mathbf{0} \\ \mathbf{0} & {}^n_v\mathbf{E}\sigma^2_{\dot{\boldsymbol{\omega}}}{}^n_v\mathbf{E}^\mathsf{T}\frac{\Delta t(t_k)^3}{3} & \mathbf{0} & {}^n_v\mathbf{E}(t_{k-1})\sigma^2_{\dot{\boldsymbol{\omega}}}\frac{\Delta t(t_k)^2}{2} \\ \sigma^2_{\dot{\mathbf{v}}}{}^n_v\mathbf{R}^\mathsf{T}(t_{k-1})\frac{\Delta t(t_k)^2}{2} & \mathbf{0} & \sigma^2_{\dot{\mathbf{v}}}\Delta t(t_k) & \mathbf{0} \\ \mathbf{0} & \sigma^2_{\dot{\boldsymbol{\omega}}}{}^n_v\mathbf{E}^\mathsf{T}(t_{k-1})\frac{\Delta t(t_k)^2}{2} & \mathbf{0} & \sigma^2_{\dot{\boldsymbol{\omega}}}\Delta t(t_k) \end{bmatrix} \tag{B.16}$$

The complete discrete vehicle model is

$$\begin{bmatrix} {}^n\mathbf{t}_v(t_k) \\ {}^n\boldsymbol{\psi}_v(t_k) \\ {}^v\mathbf{v}_v(t_k) \\ \boldsymbol{\omega}_v(t_k) \end{bmatrix} = \begin{bmatrix} {}^n\mathbf{t}_v(t_{k-1}) + {}^n_v\mathbf{R}(t_{k-1}){}^v\mathbf{v}_v(t_{k-1})\Delta t(t_k) \\ {}^n\boldsymbol{\psi}_v(t_{k-1}) + {}^n_v\mathbf{E}(t_{k-1})\boldsymbol{\omega}_v(t_{k-1})\Delta t(t_k) \\ {}^v\mathbf{v}_v(t_{k-1}) \\ \boldsymbol{\omega}_v(t_{k-1}) \end{bmatrix} + \begin{bmatrix} {}^n_v\mathbf{R}(t_{k-1})\mathbf{w}_{\dot{\mathbf{v}}}(t_k)\frac{\Delta t(t_k)^2}{2} \\ {}^n_v\mathbf{E}(t_{k-1})\mathbf{w}_{\dot{\boldsymbol{\omega}}}(t_k)\frac{\Delta t(t_k)^2}{2} \\ \mathbf{w}_{\dot{\mathbf{v}}}(t_k)\Delta t(t_k) \\ \mathbf{w}_{\dot{\boldsymbol{\omega}}}(t_k)\Delta t(t_k) \end{bmatrix} \tag{B.17}$$

The Jacobian of the vehicle model with respect to the vehicle states is

$$\nabla_x \mathbf{f}_v(t_k) = \begin{bmatrix} \mathbf{I} & \mathbf{A} & {}^n_v\mathbf{R}(t_{k-1})\Delta t(t_k) & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} & {}^n_v\mathbf{E}(t_{k-1})\Delta t(t_k) \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{B.18}$$

where

$$\mathbf{A} = \begin{bmatrix} \frac{\partial^n_v\mathbf{R}(t_{k-1})}{\partial^n\phi_v} & \frac{\partial^n_v\mathbf{R}(t_{k-1})}{\partial^n\theta_v} & \frac{\partial^n_v\mathbf{R}(t_{k-1})}{\partial^n\psi_v} \end{bmatrix} {}^v\mathbf{v}_v(t_{k-1})\Delta t(t_k) \tag{B.19}$$

$$\mathbf{B} = \mathbf{I} + \begin{bmatrix} \frac{\partial^n_v\mathbf{E}(t_{k-1})}{\partial^n\phi_v} & \frac{\partial^n_v\mathbf{E}(t_{k-1})}{\partial^n\theta_v} & \frac{\partial^n_v\mathbf{E}(t_{k-1})}{\partial^n\psi_v} \end{bmatrix} \boldsymbol{\omega}_v(t_{k-1})\Delta t(t_k) \tag{B.20}$$

B.3 Sensor Models

B.3.1 DVL Observation Model

The observation model is required to have the form

$$\mathbf{z}_d(t_k) = \mathbf{h}_d[\mathbf{x}(t_k)] + \mathbf{v}_d(t_k) \tag{B.21}$$

The DVL observes velocity relative to the seafloor, measured in its own reference frame. Following the model in [27], the observed velocity including the effect of the distance between the vehicle and sensor frames is

$$\begin{aligned}\mathbf{h}_d[\mathbf{x}(t_k)] &= {}^d_v\mathbf{R}(t_k)\left({}^v\mathbf{v}_v(t_k) + \boldsymbol{\omega}_v(t_k) \times {}^v\mathbf{t}_d\right) \\ &= {}^d_v\mathbf{R}(t_k)\left({}^v\mathbf{v}_v(t_k) - [{}^v\mathbf{t}_d]_\times \boldsymbol{\omega}_v(t_k)\right)\end{aligned} \tag{B.22}$$

in which ${}^v\mathbf{t}_d$ is the position of the DVL relative to the vehicle frame, which should be known from measurement or calibration.

The Jacobian of the DVL observation function model with respect to the vehicle states is

$$\nabla_{x_v}\mathbf{h}_d(t_k) = \begin{bmatrix}\mathbf{0}_{(3\times3)} & \mathbf{0}_{(3\times3)} & {}^d_v\mathbf{R}(t_k) & -{}^d_v\mathbf{R}(t_k)[{}^v\mathbf{t}_d]_\times\end{bmatrix} \tag{B.23}$$

The observation covariance has the form

$$\mathbf{R}_d(t_k) = \begin{bmatrix}\sigma^2_{v_x}(t_k) & 0 & 0 \\ 0 & \sigma^2_{v_y}(t_k) & 0 \\ 0 & 0 & \sigma^2_{v_z}(t_k)\end{bmatrix} \tag{B.24}$$

B.3.2 Attitude Sensor Observation Model

The observation model is required to have the form

$$\mathbf{z}_a(t_k) = \mathbf{h}_a[\mathbf{x}(t_k)] + \mathbf{v}_a(t_k) \tag{B.25}$$

The attitude sensor observes its observation relative to some reference frame (frame r).

Following the model in [27], the observed angles are

$$\begin{aligned}\mathbf{h}_a[\mathbf{x}(t_k)] &= \mathbf{A}\,{}^r\mathbf{p}_a(t_k) \\ &= \mathbf{A}({}^r\mathbf{p}_n \oplus {}^n\mathbf{p}_a(t_k)) \\ &= \mathbf{A}({}^r\mathbf{p}_n \oplus ({}^n\mathbf{p}_v(t_k) \oplus {}^v\mathbf{p}_a))\end{aligned} \tag{B.26}$$

where the matrix $\mathbf{A} = [\mathbf{0}_{(3\times3)}\ \mathbf{I}_{(3\times3)}]$ extracts the attitude states from a pose vector.

The Jacobian of the observation function relative to the vehicle states is

$$\nabla_{x_v}\mathbf{h}_a(t_k) = \mathbf{A}\left(\mathbf{J}_{\oplus 2}|_{({}^r\mathbf{p}_n,\,{}^n\mathbf{p}_a(t_k))}\ \mathbf{J}_{\oplus 1}|_{({}^n\mathbf{p}_v(t_k),\,{}^v\mathbf{p}_a)}\right)$$

The observation covariance has the form

$$\mathbf{R}_a(t_k) = \begin{bmatrix}\sigma_\phi^2(t_k) & 0 & 0 \\ 0 & \sigma_\theta^2(t_k) & 0 \\ 0 & 0 & \sigma_\psi^2(t_k)\end{bmatrix} \tag{B.27}$$

B.3.3 Depth Sensor Observation Model

The observation model is required to have the form

$$\mathbf{z}_p(t_k) = \mathbf{h}_p[\mathbf{x}(t_k)] + \mathbf{v}_p(t_k) \tag{B.28}$$

The position of the depth sensor in the navigation frame is given by

$${}^n\mathbf{t}_p(t_k) = {}^n\mathbf{t}_v(t_k) + {}^n_v\mathbf{R}(t_k)\,{}^v\mathbf{t}_p \tag{B.29}$$

where ${}^v\mathbf{t}_p$ is the position of the depth sensor in the vehicle pose, which should be known from measurement or calibration.

The pressure transducer measures the z-axis position of the depth sensor in the navigation frame

$$\mathbf{h}_p[\mathbf{x}(t_k)] = \mathbf{A}\left({}^n\mathbf{t}_v(t_k) + {}^n_v\mathbf{R}(t_k)\,{}^v\mathbf{t}_p\right)$$

in which the matrix $\mathbf{A} = [0\ 0\ 1]$ extracts the z-coordinate from a position vector.

The Jacobian of the depth observation function with respect to the vehicle states is

$$\nabla_{x_v}\mathbf{h}_p(t_k) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \mathbf{A}\left(\frac{\partial_v^n \mathbf{R}}{\partial^n \phi_v}\right){}^v\mathbf{t}_p \\ \mathbf{A}\left(\frac{\partial_v^n \mathbf{R}}{\partial^n \theta_v}\right){}^v\mathbf{t}_p \\ 0 \\ \mathbf{0}_{(3\times 1)} \\ \mathbf{0}_{(3\times 1)} \end{bmatrix} \tag{B.30}$$

The depth observation covariance is

$$\mathbf{R}_p(t_k) = \left[\sigma_p^2(t_k)\right] \tag{B.31}$$

Appendix C

Least Squares Problems and Solutions

This appendix describes the least squares problem and several solutions. Least-squares problems arise in this thesis when calculating maximum likelihood model parameters to fit a set of observations. More information on the algorithms listed here can be found in [50] which focuses on computer vision applications, and [84] which provides a general overview of numerical optimisation.

C.1 The Least Squares Problem

The states to be estimated are represented by a parameter vector $\mathbf{x} \in \Re^N$. Some measured quantities related to the parameters are represented by an observation vector $\mathbf{z} \in \Re^M$ with covariance $\mathbf{\Sigma_z}$. A function $\mathbf{f} : \Re^N \rightarrow \Re^M$ maps the parameter vector to an estimated observation vector

$$\hat{\mathbf{z}} = \mathbf{f}(\mathbf{x}) \tag{C.1}$$

The difference between the actual and expected observation is the residual

$$\begin{aligned} \boldsymbol{\epsilon} &= \mathbf{z} - \hat{\mathbf{z}} \\ &= \mathbf{z} - \mathbf{f}(\mathbf{x}) \end{aligned} \tag{C.2}$$

C.1.1 Weighted Least Squares

The weighted least squares problem involves finding the optimal parameters that minimise the the squared residual Mahalanobis distance

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\, \boldsymbol{\epsilon}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{z}}^{-1} \boldsymbol{\epsilon} \tag{C.3}$$

The covariance of the parameter estimate vector is

$$\boldsymbol{\Sigma}_{\mathbf{x}} = \left(\boldsymbol{\nabla}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{z}}^{-1} \boldsymbol{\nabla} \right)^{-1} \tag{C.4}$$

where $\boldsymbol{\nabla}$ is the Jacobian of the observation function

$$\boldsymbol{\nabla} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}} \tag{C.5}$$

C.1.2 Ordinary Least Squares

If the observation covariance is proportional to the identity matrix, Equation C.3 simplifies to minimising the squared residual Euclidean distance, which is typically called ordinary least squares.

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\, \boldsymbol{\epsilon}^{\mathsf{T}} \boldsymbol{\epsilon} \tag{C.6}$$

C.1.3 Converting Weighted Least Squares to Ordinary Least Squares

A weighted least squares problem can be transformed into an ordinary least squares problem by weighting the observation and predicted observation by the square root of the inverse observation covariance matrix.

The standardised observation and standardised predicted observation are

$$\mathbf{z}^{\diamond} = \sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\mathbf{z} \tag{C.7}$$

$$\hat{\mathbf{z}}^{\diamond} = \sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\hat{\mathbf{z}} \tag{C.8}$$

The standardised residual is then

$$\boldsymbol{\epsilon}^{\diamond} = \sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\boldsymbol{\epsilon} \tag{C.9}$$

When the standardised residual is substituted into the ordinary least squares objective

function in Equation C.6 it becomes

$$\begin{aligned}(\epsilon^{\diamond})^{\mathsf{T}}\epsilon^{\diamond} &= \left(\sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\epsilon\right)^{\mathsf{T}}\left(\sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\epsilon\right) \\ &= \epsilon^{\mathsf{T}}\left(\sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\right)^{\mathsf{T}}\sqrt{(\boldsymbol{\Sigma}_{\mathbf{z}}^{-1})}\epsilon \\ &= \epsilon^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\epsilon\end{aligned} \tag{C.10}$$

which is the weighted least squares objective function in Equation C.3

C.2 A Solution for Linear Problems

If the function $\mathbf{f}[\cdot]$ defined in Section C.1 is linear, the predicted observation of Equation C.1 can be simplified to

$$\hat{\mathbf{z}} = \mathbf{F}\mathbf{x} \tag{C.11}$$

The objective function of Equation C.3 is then

$$\begin{aligned}\mathbf{o}[\mathbf{x}] &= (\mathbf{z}-\mathbf{F}\mathbf{x})^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}(\mathbf{z}-\mathbf{F}\mathbf{x}) \\ &= \mathbf{z}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{z} - 2\mathbf{x}^{\mathsf{T}}\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{z} + \mathbf{x}^{\mathsf{T}}\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F}\mathbf{x}\end{aligned} \tag{C.12}$$

The derivative of the objective function with respect to the parameters is

$$\frac{\partial \mathbf{o}[\mathbf{x}]}{\partial \mathbf{x}} = 2\mathbf{x}^{\mathsf{T}}\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F} - 2\mathbf{x}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F} \tag{C.13}$$

The solution to the least squares problem occurs when the derivative is zero, which produces the normal equation

$$\left(\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F}\right)\hat{\mathbf{x}} = \mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{z} \tag{C.14}$$

If the matrix $\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F}$ is invertible, the solution of Equation C.14 can be found with

$$\hat{\mathbf{x}} = \left(\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\mathbf{z} \tag{C.15}$$

however methods to solve the linear system of Equation C.14 using Cholesky, QR and SVD matrix decomposition are favoured for numerical stability [84].

C.3 Solutions for Non-Linear Problems

C.3.1 The Gauss-Newton Algorithm

The residual is linearised around a parameter estimate $\mathbf{x}_0$

$$\begin{aligned} \boldsymbol{\epsilon} &= \mathbf{z} - \mathbf{f}(\mathbf{x}) \\ &\approx \mathbf{z} - \mathbf{f}(\mathbf{x}_0) - \boldsymbol{\nabla}\boldsymbol{\delta}_{\mathbf{x}} \end{aligned} \tag{C.16}$$

where $\boldsymbol{\delta}_{\mathbf{x}} = \mathbf{x} - \mathbf{x}_0$. The parameter update $\boldsymbol{\delta}_{\mathbf{x}}$ that minimised the linearised least squares system is given by the normal equation

$$\left(\boldsymbol{\nabla}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\boldsymbol{\nabla}\right)\boldsymbol{\delta}_{\mathbf{x}} = \boldsymbol{\nabla}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\boldsymbol{\epsilon} \tag{C.17}$$

The psuedocode for the Gauss-Newton implementation used in this thesis is listed in Algorithm C.1. The Gauss-Newton method provides the basis for the Levenberg-Marquardt algorithm presented in Section C.3.2.

C.3.2 The Levenberg-Marquardt Algorithm

The Gauss-Newton method converges quickly when initialised near a minimum, but often fails when initialised with a poor parameter estimate. The Levenberg-Marquardt algorithm updates the state estimated with a linear combination of Gauss-Newton and Steepest Descent directions. Each iteration involves solving the augmented normal equations

$$\left(\boldsymbol{\nabla}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\boldsymbol{\nabla} + \mu\mathbf{I}\right)\boldsymbol{\delta}_{\mathbf{x}} = \boldsymbol{\nabla}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{-1}\boldsymbol{\epsilon} \tag{C.18}$$

for some value of the damping factor μ. When μ is large, the algorithm approximates the steepest descent method, which displays slow but stable convergence when far from a minimum. When μ is small, Levenberg-Marquardt approximates the Gauss-Newton method for fast convergence.

Algorithm C.2 lists the psuedocode for the Levenberg-Marquardt implementation used on this thesis, based on the algorithms presented in [68] and [64]. For details on the evolution of the damping factor μ see [68, 79].

C.3.3 A Sparse Levenberg-Marquardt Algorithm

A naive implementation of the Levenberg-Marquardt algorithm as described in Section C.3.2 has a complexity of $O(n^3)$. Typical problems are somewhat sparse however, where

observations are a function of a small subset of all parameters.

The sparse Levenberg-Marquardt algorithm described in this section is typically used to estimate camera motion and feature parameters (a problem known as bundle adjustment), and is described in more detail in [50, 65].

If the parameter vector can be partitioned in the form

$$\mathbf{x} = \begin{bmatrix} \mathbf{a}^\mathsf{T} & \mathbf{b}_1^\mathsf{T} & \mathbf{b}_2^\mathsf{T} & \dots & \mathbf{b}_n^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{C.19}$$

and the predicted observation and residual vectors can similarly be partitioned

$$\hat{\mathbf{z}} = \begin{bmatrix} \hat{\mathbf{z}}_1^\mathsf{T} & \hat{\mathbf{z}}_2^\mathsf{T} & \dots & \hat{\mathbf{z}}_n^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{C.20}$$

$$\epsilon = \begin{bmatrix} \epsilon_1^\mathsf{T} & \epsilon_2^\mathsf{T} & \dots & \epsilon_n^\mathsf{T} \end{bmatrix}^\mathsf{T} \tag{C.21}$$

and each $\hat{\mathbf{z}}_i$ is dependent on $\mathbf{a}$ and $\mathbf{b}_i$ only, the predicted observation function Jacobian has the sparse form

$$\nabla = \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_2 & \mathbf{0} & \mathbf{B}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{A}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_n \end{bmatrix} \tag{C.22}$$

where $\mathbf{A}_i = \frac{\partial \hat{\mathbf{z}}_i}{\partial \mathbf{a}}$ and $\mathbf{B}_i = \frac{\partial \hat{\mathbf{z}}_i}{\partial \mathbf{b}_i}$.

It is also assumed that observations are independent, meaning the observation covariance matrix has is block diagonal of the form

$$\Sigma_\mathbf{z} = \begin{bmatrix} \Sigma_{\mathbf{z}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Sigma_{\mathbf{z}_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Sigma_{\mathbf{z}_n} \end{bmatrix} \tag{C.23}$$

In this thesis problems satisfying these assumptions arise when the vector $\mathbf{a}$ represents the poses of a vision rig, $\mathbf{b}_i$ represents the position of feature i and $\hat{\mathbf{z}}_i$ represents the predicted observations of feature i.

Using the sparse Jacobian, the component $\nabla^\mathsf{T} \Sigma_\mathbf{z}^{-1} \nabla$ in the left hand side of the Levenberg-

Marquardt augmented normal equations in Equation C.18 is

$$\nabla^{\mathsf{T}}\Sigma_{\mathbf{z}}^{-1}\nabla = \begin{bmatrix} \mathbf{U} & \mathbf{W}_1 & \mathbf{W}_2 & \dots & \mathbf{W}_n \\ \mathbf{W}_1^{\mathsf{T}} & \mathbf{V}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{W}_2^{\mathsf{T}} & \mathbf{0} & \mathbf{V}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{W}_n^{\mathsf{T}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{V}_n \end{bmatrix} \tag{C.24}$$

where

$$\mathbf{U} = \sum_{i}^{n} \mathbf{A}_i^{\mathsf{T}}\Sigma_{\mathbf{z}_i}^{-1}\mathbf{A}_i \tag{C.25}$$

$$\mathbf{V}_i = \mathbf{B}_i^{\mathsf{T}}\Sigma_{\mathbf{z}_i}^{-1}\mathbf{B}_i \tag{C.26}$$

$$\mathbf{W}_i = \mathbf{A}_i^{\mathsf{T}}\Sigma_{\mathbf{z}_i}^{-1}\mathbf{B}_i \tag{C.27}$$

Substituting the sparse Jacobian into the right hand side of Equation C.18 gives

$$\nabla^{\mathsf{T}}\Sigma_{\mathbf{z}}^{-1}\epsilon = \begin{bmatrix} \epsilon_a \\ \epsilon_{b_1} \\ \epsilon_{b_2} \\ \vdots \\ \epsilon_{b_n} \end{bmatrix} \tag{C.28}$$

where

$$\epsilon_a = \sum_{i}^{n} \mathbf{A}_i^{\mathsf{T}}\Sigma_{\mathbf{z}_i}^{-1}\epsilon_i \tag{C.29}$$

$$\epsilon_{b_i} = \mathbf{B}_i^{\mathsf{T}}\Sigma_{\mathbf{z}_i}^{-1}\epsilon_i \tag{C.30}$$

Using the results of Equations C.24 and C.28, the Levenberg-Marquardt augmented normal equations can be represented as

$$\begin{bmatrix} (\mathbf{U} + \mu\mathbf{I}) & \mathbf{W} \\ \mathbf{W}^{\mathsf{T}} & (\mathbf{V} + \mu\mathbf{I}) \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_b \end{bmatrix} = \begin{bmatrix} \epsilon_a \\ \epsilon_b \end{bmatrix} \tag{C.31}$$

where

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{V}_n \end{bmatrix} \tag{C.32}$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 & \dots & \mathbf{W}_n \end{bmatrix} \tag{C.33}$$

$$\epsilon_b = \begin{bmatrix} \epsilon_{b_1} \\ \epsilon_{b_2} \\ \vdots \\ \epsilon_{b_n} \end{bmatrix} \tag{C.34}$$

Premultiplying each side of Equation C.31 by the matrix $\begin{bmatrix} \mathbf{I} & \mathbf{W}(\mathbf{V}+\mu\mathbf{I})^{-1} \\ \mathbf{W}^\mathsf{T} & \mathbf{I} \end{bmatrix}$ gives

$$\begin{bmatrix} (\mathbf{U}+\mu\mathbf{I}) - \mathbf{W}(\mathbf{V}+\mu\mathbf{I})^{-1}\mathbf{W}^\mathsf{T} & \mathbf{0} \\ \mathbf{W}^\mathsf{T} & (\mathbf{V}+\mu\mathbf{I}) \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_b \end{bmatrix} = \begin{bmatrix} \epsilon_a - \mathbf{W}(\mathbf{V}+\mu\mathbf{I})^{-1}\epsilon_b \\ \epsilon_b \end{bmatrix} \tag{C.35}$$

Noting that

$$\mathbf{W}(\mathbf{V}+\mu\mathbf{I})^{-1}\mathbf{W}^\mathsf{T} = \sum_{i=1}^{n} \mathbf{W}_i(\mathbf{V}_i+\mu\mathbf{I})^{-1}\mathbf{W}_i^\mathsf{T} \tag{C.36}$$

and

$$\mathbf{W}(\mathbf{V}+\mu\mathbf{I})^{-1}\epsilon_b = \sum_{i=1}^{n} \mathbf{W}_i(\mathbf{V}_i+\mu\mathbf{I})^{-1}\epsilon_{b_i} \tag{C.37}$$

and substituting $\mathbf{Y}_i = \mathbf{W}_i(\mathbf{V}_i+\mu\mathbf{I})^{-1}$, Equation C.35 simplifies to

$$\begin{bmatrix} (\mathbf{U}+\mu\mathbf{I}) - \sum_{i=1}^{n}(\mathbf{Y}_i\mathbf{W}_i^\mathsf{T}) & \mathbf{0} \\ \mathbf{W}^\mathsf{T} & (\mathbf{V}+\mu\mathbf{I}) \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_b \end{bmatrix} = \begin{bmatrix} \epsilon_a - \sum_{i=1}^{n}(\mathbf{Y}_i\epsilon_{b_i}) \\ \epsilon_b \end{bmatrix} \tag{C.38}$$

The parameter update steps δ_a and δ_b can therefore be found by solving

$$\left((\mathbf{U}+\mu\mathbf{I}) - \sum_{i=1}^{n}(\mathbf{Y}_i\mathbf{W}_i^\mathsf{T}) \right) \delta_a = \epsilon_a - \sum_{i=1}^{n}(\mathbf{Y}_i\epsilon_{b_i}) \tag{C.39}$$

and

$$\delta_{b_i} = (\mathbf{V}_i+\mu\mathbf{I})^{-1}(\epsilon_{b_i} - \mathbf{W}_i^\mathsf{T}\delta_a) \tag{C.40}$$

Input:

- An initial parameter estimate $\mathbf{x}_0$
- A observation vector $\mathbf{z}$ and observation covariance $\mathbf{\Sigma_z}$
- A maximum number of iterations k_{max}

Output:

- An optimised parameter estimate $\hat{\mathbf{x}}$ that minimises $\epsilon^T \mathbf{\Sigma_z}^{-1} \epsilon$
- The parameter covariance $\mathbf{\Sigma_x}$

Procedure:

$\mathbf{x} \leftarrow \mathbf{x}_0$
$\epsilon \leftarrow \mathbf{z} - \mathbf{f}(\mathbf{x})$
$\mathbf{A} \leftarrow \nabla^T \mathbf{\Sigma_z}^{-1} \nabla$
$\mathbf{g} \leftarrow \nabla^T \mathbf{\Sigma_z}^{-1} \epsilon$
$k \leftarrow 0$
$stop \leftarrow false$
while $(stop = false$ **and** $k < k_{max})$ **do**
  $\delta_\mathbf{x} \leftarrow$ **solve**$(\mathbf{A}\delta_\mathbf{x} = \mathbf{g})$
  if ($\|\delta_\mathbf{x}\| \leq \epsilon_2 \|\mathbf{x}\|$) **then**
    $stop \leftarrow true$
  else
    $\mathbf{x}_{new} \leftarrow \mathbf{x} + \delta_\mathbf{x}$
    $\epsilon_{new} \leftarrow \mathbf{z} - \mathbf{f}(\mathbf{x}_{new})$
    if $(\epsilon_{new}^T \mathbf{\Sigma_z}^{-1} \epsilon_{new} < \epsilon^T \mathbf{\Sigma_z}^{-1} \epsilon)$ **then**
      $\mathbf{x} \leftarrow \mathbf{x}_{new}$
      $\epsilon \leftarrow \epsilon_{new}$
      $\mathbf{A} \leftarrow \nabla^T \mathbf{\Sigma_z}^{-1} \nabla$
      $\mathbf{g} \leftarrow \nabla^T \mathbf{\Sigma_z}^{-1} \epsilon$
    else
      $stop \leftarrow true$
    end if
  end if
  $k \leftarrow k + 1$
end while
$\hat{\mathbf{x}} \leftarrow \mathbf{x}$
$\mathbf{\Sigma_x} \leftarrow \mathbf{A}^{-1}$

Algorithm C.1: The Gauss-Newton algorithm for non-linear weighted least squares.

Input:

- An initial parameter estimate $\mathbf{x}_0$
- A observation vector $\mathbf{z}$ and observation covariance $\mathbf{\Sigma_z}$
- An initial damping factor μ_0
- A maximum number of iterations k_{max}
- A minimum gradient threshold ϵ_1
- A minimum step-size ratio threshold ϵ_2

Output:

- An optimised parameter estimate $\hat{\mathbf{x}}$ that minimises $\epsilon^\mathsf{T}\mathbf{\Sigma_z}^{-1}\epsilon$
- The parameter covariance $\mathbf{\Sigma_x}$

Procedure:

$\mathbf{x} \leftarrow \mathbf{x}_0$
$\epsilon \leftarrow \mathbf{z} - \mathbf{f}(\mathbf{x})$
$\mathbf{A} \leftarrow \nabla^\mathsf{T}\mathbf{\Sigma_z}^{-1}\nabla$
$\mathbf{g} \leftarrow \nabla^\mathsf{T}\mathbf{\Sigma_z}^{-1}\epsilon$
$k \leftarrow 0$
$\mu \leftarrow \mu_0 \times \max\{\mathbf{A}_{11}, \mathbf{A}_{22}, ..., \mathbf{A}_{nn}\}$
$\nu \leftarrow 2$
$stop \leftarrow false$
while $(stop = false)$ **and** $(k < k_{max})$ **and** $(\|\mathbf{g}\|_\infty > \epsilon_1)$ **do**
  $\delta_\mathbf{x} \leftarrow$ **solve** $((\mathbf{A} + \mu\mathbf{I})\,\delta_\mathbf{x} = \mathbf{g})$
  if $(\ \|\delta_\mathbf{x}\| \leq \epsilon_2\|\mathbf{x}\|\ )$ **then**
    $stop \leftarrow true$
  else
    $\mathbf{x}_{new} \leftarrow \mathbf{x} + \delta_\mathbf{x}$
    $\epsilon_{new} \leftarrow \mathbf{z} - \mathbf{f}(\mathbf{x}_{new})$
    $\rho \leftarrow \left((\epsilon_{new}^\mathsf{T}\mathbf{\Sigma_z}^{-1}\epsilon_{new} - \epsilon^\mathsf{T}\mathbf{\Sigma_z}^{-1}\epsilon)\right) / (\delta_\mathbf{x}^\mathsf{T}(\mu\delta_\mathbf{x} + \mathbf{g}))$
    if $(\ \rho > 0\ )$ **then**
      $\mathbf{x} \leftarrow \mathbf{x}_{new}$
      $\epsilon \leftarrow \epsilon_{new}$
      $\mathbf{A} \leftarrow \nabla^\mathsf{T}\mathbf{\Sigma_z}^{-1}\nabla$
      $\mathbf{g} \leftarrow \nabla^\mathsf{T}\mathbf{\Sigma_z}^{-1}\epsilon$
      $\mu \leftarrow \mu \times \max\{\frac{1}{3}, 1 - (2\rho - 1)^3\}$
      $\nu \leftarrow 2$
    else
      $\mu \leftarrow \nu \times \mu$
      $\nu \leftarrow 2 \times \nu$
    end if
  end if
  $k \leftarrow k + 1$
end while
$\hat{\mathbf{x}} \leftarrow \mathbf{x}$
$\mathbf{\Sigma_x} \leftarrow \mathbf{A}^{-1}$

Algorithm C.2: The Levenberg-Marquardt algorithm for non-linear weighted least squares.

Appendix D

Takahashi's Equations for Matrix Inversion

The Takahashi equations [25, 80] provide a method to calculate the inverse of a matrix using an LDU factorisation. In chapter 5, the Takahashi equations are used to calculate a sparse set of elements of a covariance matrix from a sparse information matrix.

The first of the Takahashi equations can be produced by starting with the matrix inverse relationship $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$ and substituting in the factorisation $\mathbf{A} = \mathbf{LDU}$ where $\mathbf{D}$ is a diagonal matrix, and $\mathbf{L}$ and $\mathbf{U}$ are lower and upper triangular matrices respectively that contain ones on their diagonals.

$$\begin{aligned}
\mathbf{A}^{-1}\mathbf{A} &= \mathbf{I} \\
\mathbf{A}^{-1}\mathbf{LDU} &= \mathbf{I} \\
\mathbf{A}^{-1}\mathbf{L} &= \mathbf{U}^{-1}\mathbf{D}^{-1} \\
\mathbf{A}^{-1}(\mathbf{L}-\mathbf{I}) + \mathbf{A}^{-1} &= \mathbf{U}^{-1}\mathbf{D}^{-1} \\
\mathbf{A}^{-1} &= \mathbf{U}^{-1}\mathbf{D}^{-1} - \mathbf{A}^{-1}(\mathbf{L}-\mathbf{I})
\end{aligned} \tag{D.1}$$

Another relation can be derived through similar manipulation of the equation $\mathbf{AA}^{-1} = \mathbf{I}$

$$\begin{aligned}
\mathbf{AA}^{-1} &= \mathbf{I} \\
\mathbf{LDUA}^{-1} &= \mathbf{I} \\
\mathbf{UA}^{-1} &= \mathbf{D}^{-1}\mathbf{L}^{-1} \\
(\mathbf{U}-\mathbf{I})\mathbf{A}^{-1} + \mathbf{A}^{-1} &= \mathbf{D}^{-1}\mathbf{L}^{-1} \\
\mathbf{A}^{-1} &= \mathbf{D}^{-1}\mathbf{L}^{-1} - (\mathbf{U}-\mathbf{I})\mathbf{A}^{-1}
\end{aligned} \tag{D.2}$$

In Equations D.1 and D.2, the components $\mathbf{U}^{-1}\mathbf{D}^{-1}$ and $\mathbf{D}^{-1}\mathbf{L}^{-1}$ are upper and lower triangular matrices respectively, that contain the elements of $\mathbf{D}^{-1}$ on the diagonal. Therefore Equation D.1 allows the lower triangular elements of $\mathbf{A}^{-1}$ to be computed without evaluating $\mathbf{U}^{-1}$

$$[\mathbf{A}^{-1}]_{ij} = [\mathbf{D}^{-1}]_{ij} - \sum_{k=i+1}^{n} [\mathbf{A}^{-1}]_{ik}\, \mathbf{L}_{kj} \quad \text{for } i \geq j \tag{D.3}$$

Similarly, Equation D.2 can be used to calculate the upper triangle of $\mathbf{A}^{-1}$ without $\mathbf{L}^{-1}$.

$$[\mathbf{A}^{-1}]_{ij} = [\mathbf{D}^{-1}]_{ij} - \sum_{k=i+1}^{n} \mathbf{U}_{ik}\, [\mathbf{A}^{-1}]_{kj} \quad \text{for } i \leq j \tag{D.4}$$

If the matrix $\mathbf{A}$ is symmetric, $\mathbf{U} = \mathbf{L}^{\mathsf{T}}$, and $\mathbf{A}^{-1}$ is also symmetric. Equations D.1 and D.2 become

$$\mathbf{A}^{-1} = (\mathbf{L}^{\mathsf{T}})^{-1}\mathbf{D}^{-1} - \mathbf{A}^{-1}(\mathbf{L} - \mathbf{I}) \tag{D.5}$$

$$\mathbf{A}^{-1} = \mathbf{D}^{-1}\mathbf{L}^{-1} - (\mathbf{L}^{\mathsf{T}} - \mathbf{I})\mathbf{A}^{-1} \tag{D.6}$$

Appendix E

Stereo-Vision Relative Pose Estimation Simulation Results

In Chapter 4, several cost functions for estimating the motion of a stereo-rig are evaluated in simulations approximating odometry and loop-closure situations. This appendix presents a set of graphs showing the complete results of the odometry and loop-closure simulations that were previously summarised in Tables 4.1, 4.2, 4.3 and 4.4. The odometry simulation results are shown in Section E.1, and the loop-closure simulations are presented in Section E.2.

The following observations can be made from the graphs presented in this appendix:

- The estimator resulting from minimising the Euclidean 3D registration errors is noisy.
- Minimising the Mahalanobis 3D registration errors produces accurate motion estimates when the correct linearisation is used. However, when calculating the motion parameter covariance, evaluating the observation Jacobian at the initial triangulated feature positions produces suboptimal results. This is best observed in the loop-closure simulations of Section E.2 where the uncertainty bounds produced by this method diverge from those of bundle-adjustment.
- Minimising the image reprojection errors at only one pose after triangulating the feature positions at the previous pose results in a biased estimator.
- The maximum likelihood 3D estimator produces equivalent results to bundle adjustment.

E.1 Odometry Simulations

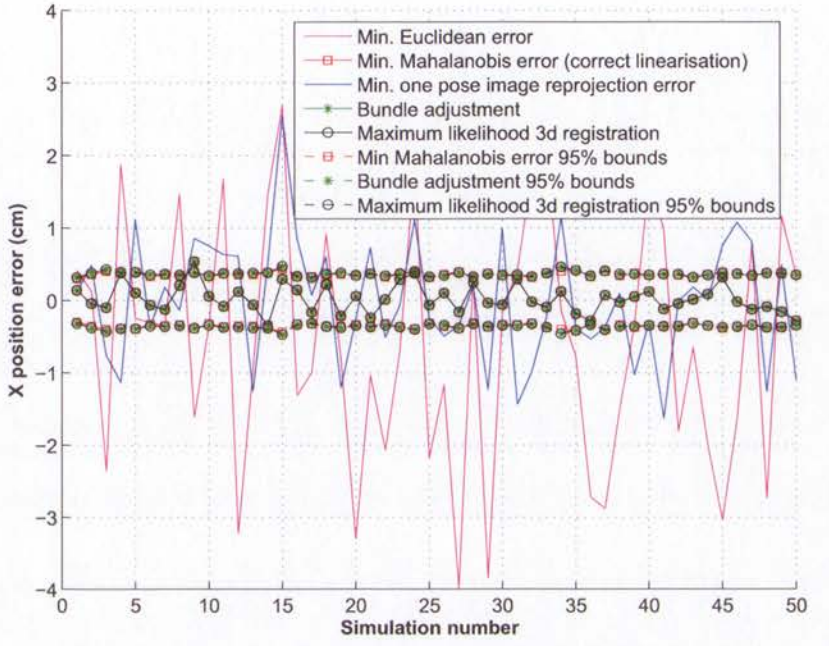


Figure E.1: X-position errors in 50 odometry simulations.

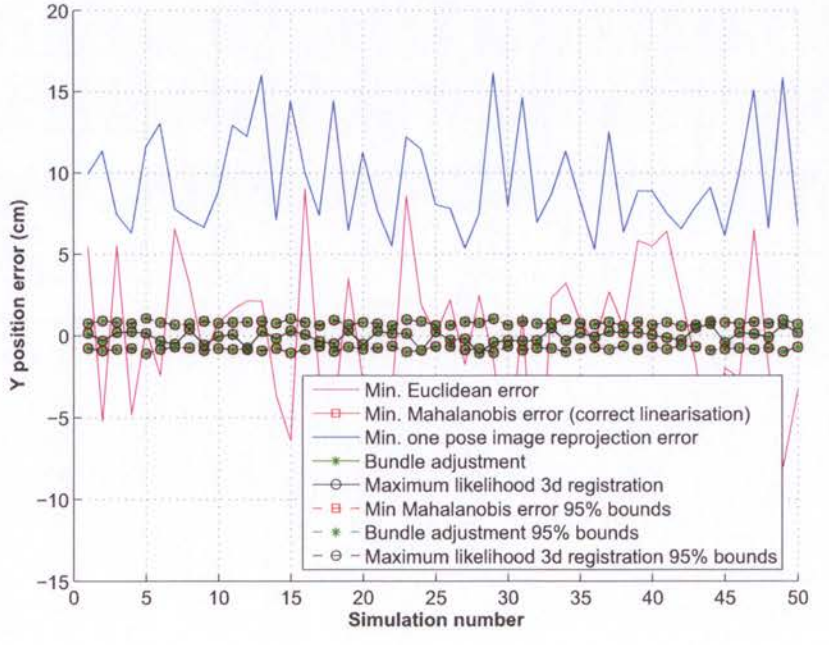


Figure E.2: Y-position errors in 50 odometry simulations.

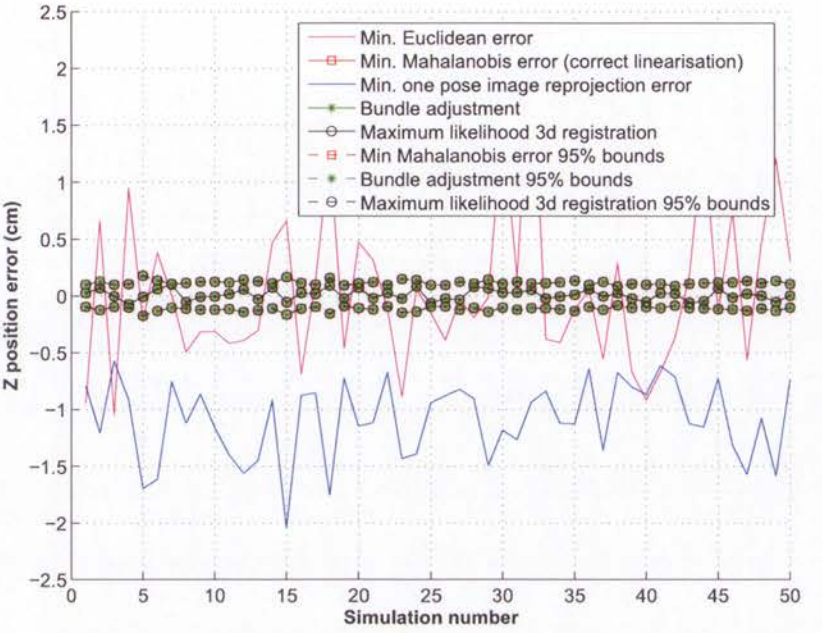


Figure E.3: Z-position errors in 50 odometry simulations.

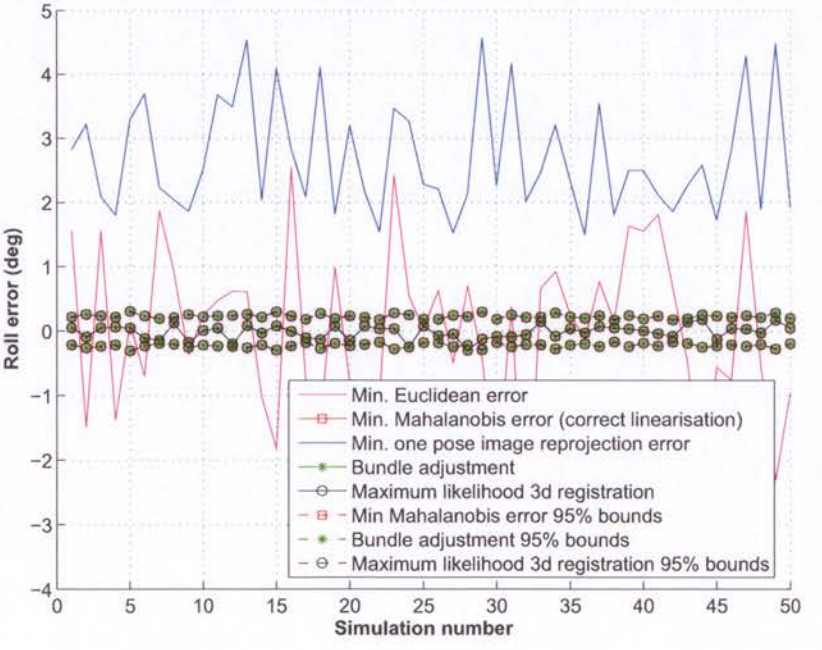


Figure E.4: Roll Euler angle errors in 50 odometry simulations.

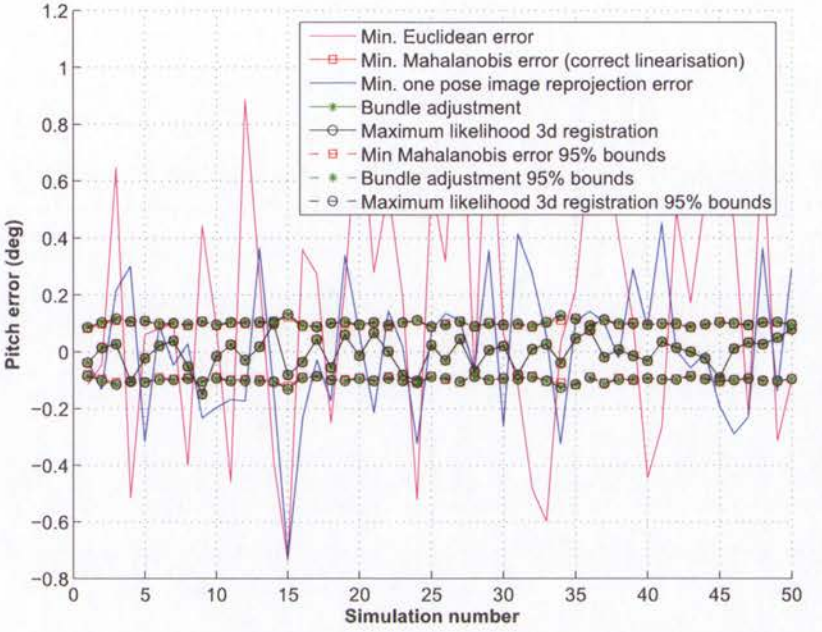


Figure E.5: Pitch Euler angle errors in 50 odometry simulations.

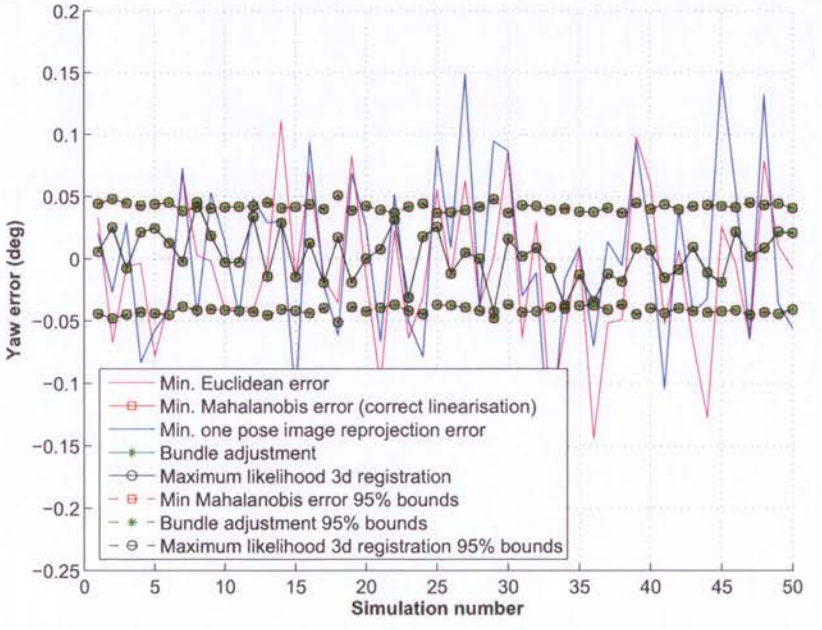


Figure E.6: Yaw Euler angle errors in 50 odometry simulations.

E.2 Loop-closure Simulations

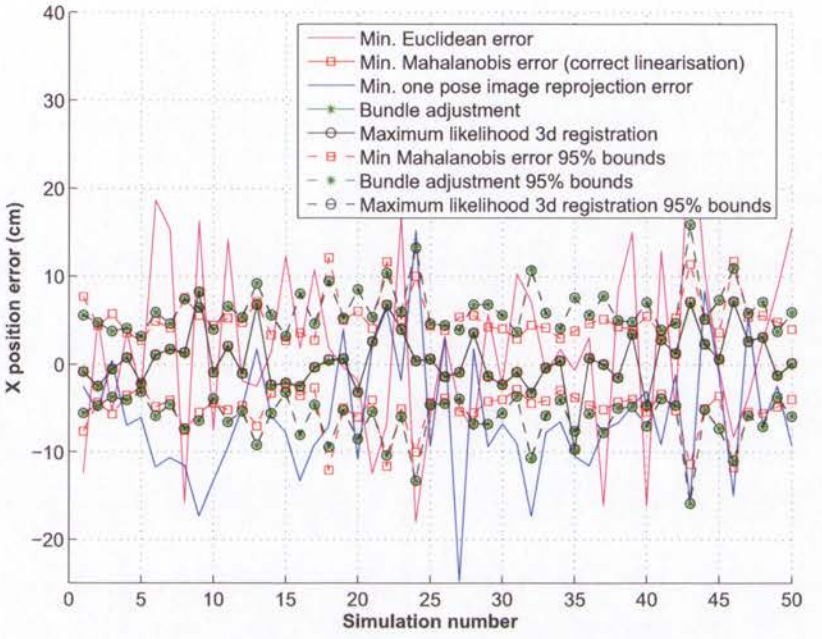


Figure E.7: X-position errors in 50 loop-closure simulations.

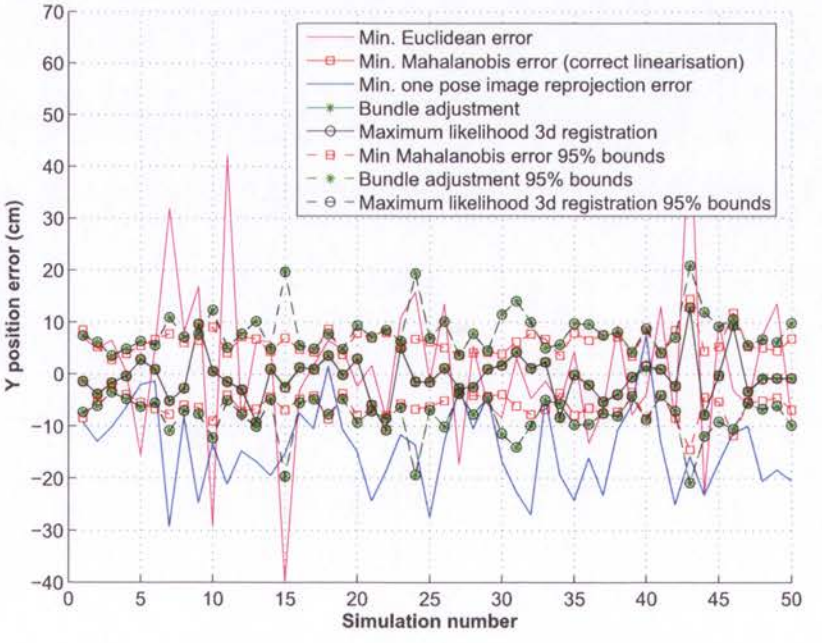


Figure E.8: Y-position errors in 50 loop-closure simulations.

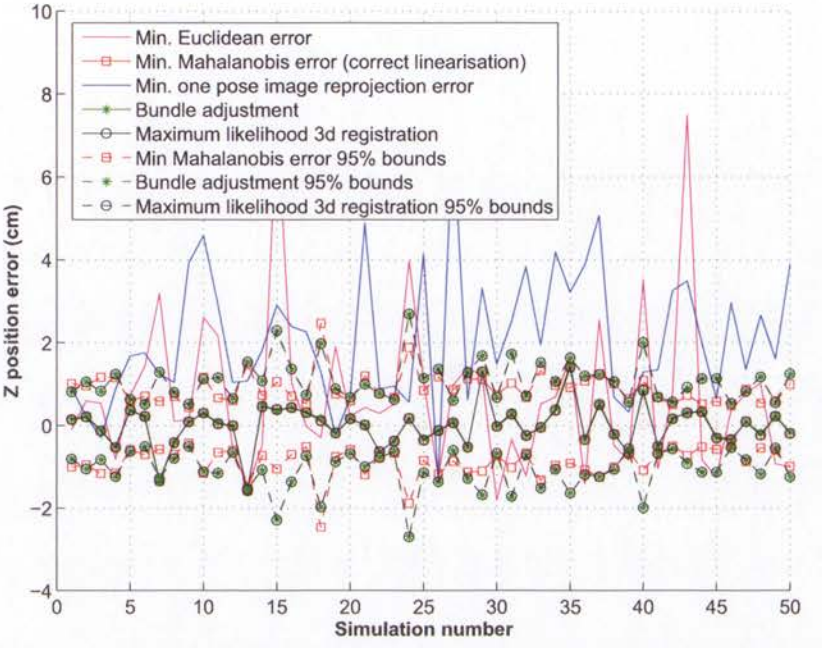


Figure E.9: Z-position errors in 50 loop-closure simulations.

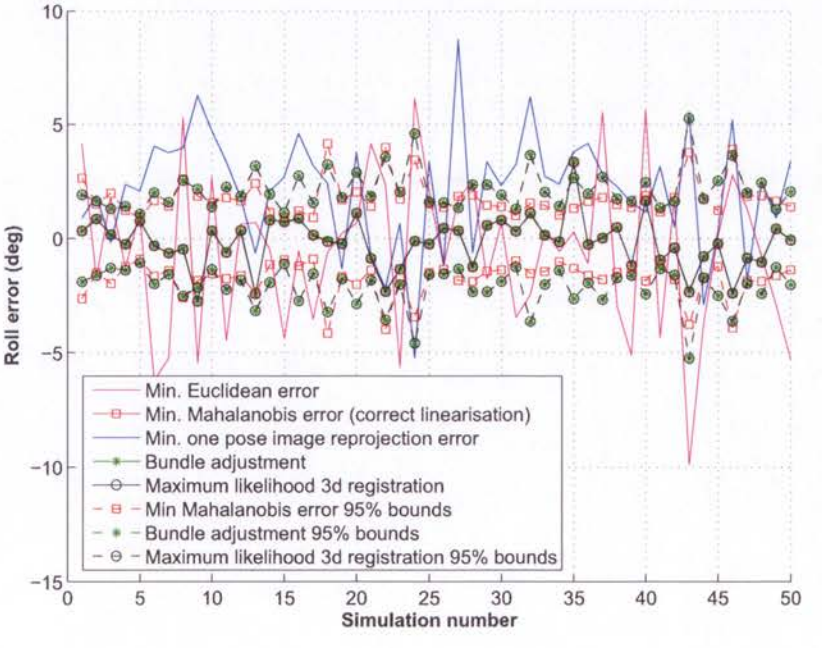


Figure E.10: Roll Euler angle errors in 50 loop-closure simulations.

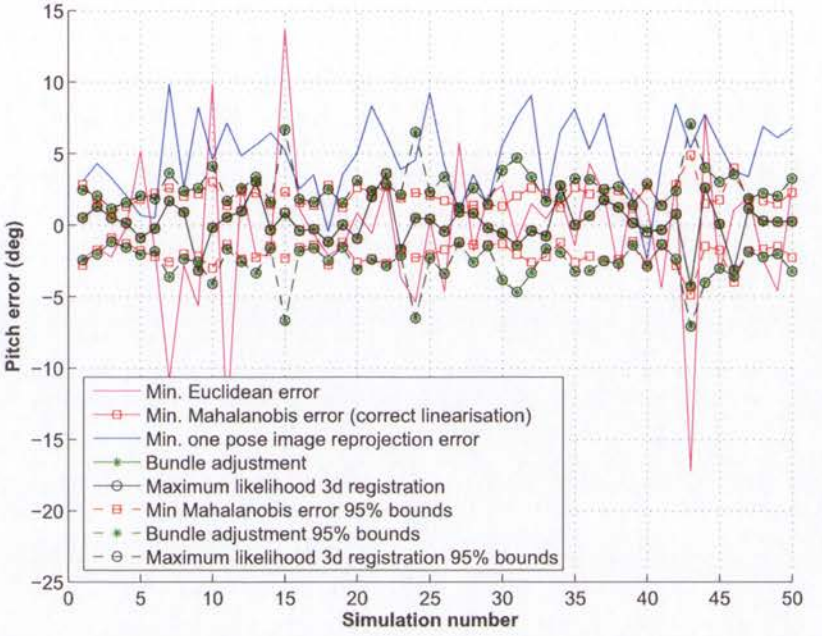


Figure E.11: Pitch Euler angle errors in 50 loop-closure simulations.

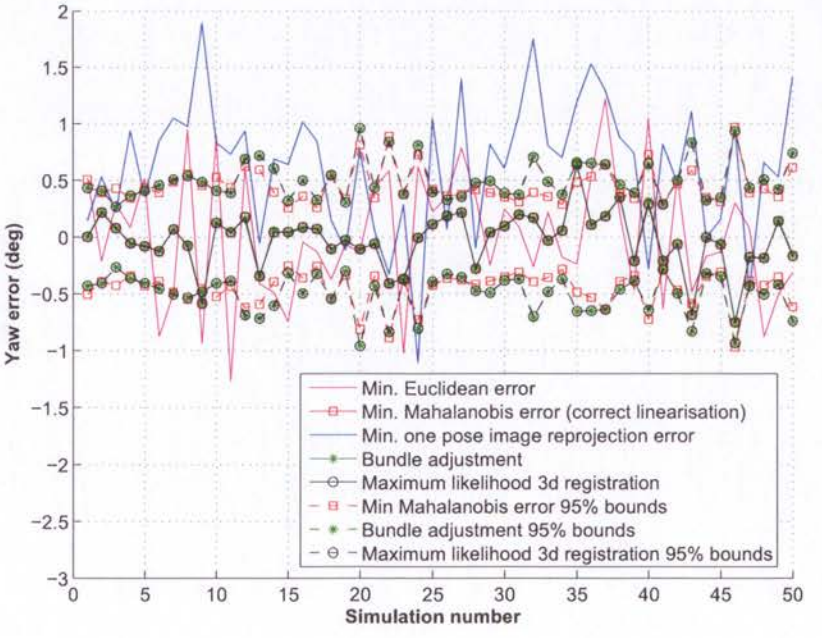


Figure E.12: Yaw Euler angle errors in 50 loop-closure simulations.

Appendix F

Stereo-Vision Outlier Rejection Simulation Results

In Chapter 4, several outlier rejection frameworks are evaluated in simulations approximating stereo-vision odometry and loop-closure scenarios. This appendix presents a set of graphs showing the errors in the final motion estimates calculated from the features classified as inliers by each outlier rejection approach. A summary of this data was previously presented in Tables 4.9, 4.10, 4.11 and 4.12.

The odometry simulation results are shown in Section F.1, and the loop-closure simulations are presented in Section F.2. In the loop-closure simulation plots, the results of the iterative rejection approach are shown separately due to the large errors in many motion estimates. Additionally, gaps in the errors for the iterative rejection approach are present in 17 of the 50 simulations where the algorithm failed to converge to a minimum set of three inlier features required to generate a motion hypothesis.

The following observations can be made from the graphs presented in this appendix:

- The iterative rejection approach works well in the odometry simulations where outliers with only small errors are present. However, in the loop-closure simulations containing outliers with large errors, the motion hypotheses are significantly corrupted causing catastrophic failure in the classification of outliers.
- The RANSAC and robust methods produce similar results, however the more accurate motion hypotheses generated by the robust estimator lead to an increased number of accepted inliers, resulting in an estimator with lower variance.

F.1 Odometry Simulations

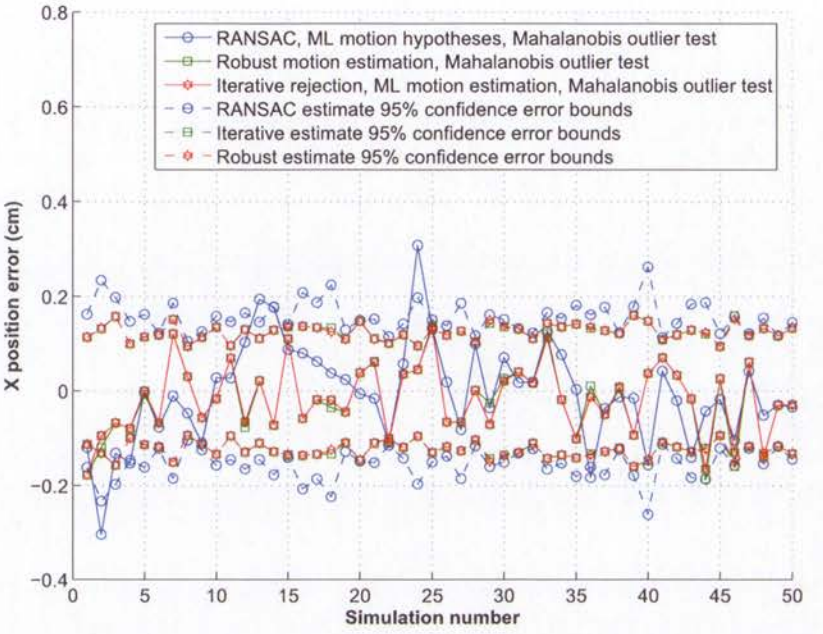


Figure F.1: X-position errors in 50 odometry simulations with outliers.

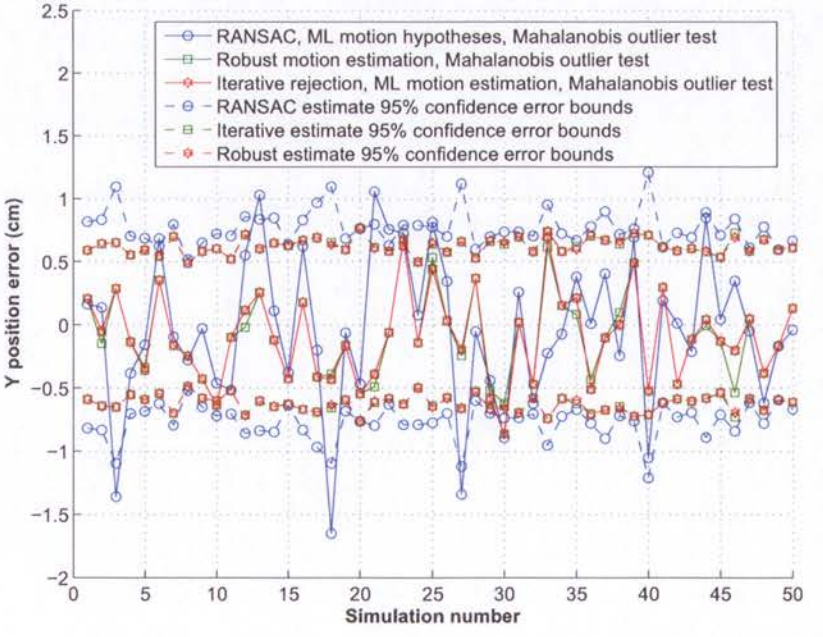


Figure F.2: Y-position errors in 50 odometry simulations with outliers.

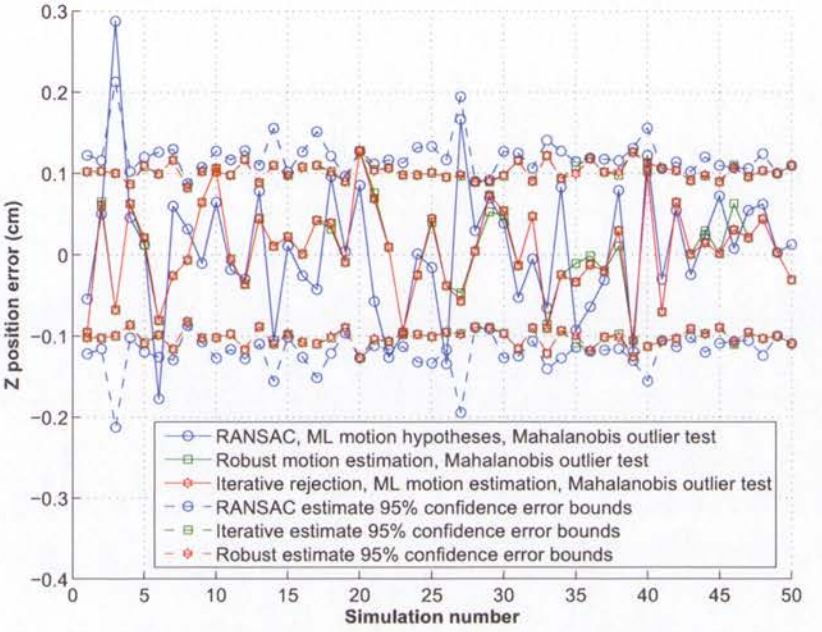


Figure F.3: Z-position errors in 50 odometry simulations with outliers.

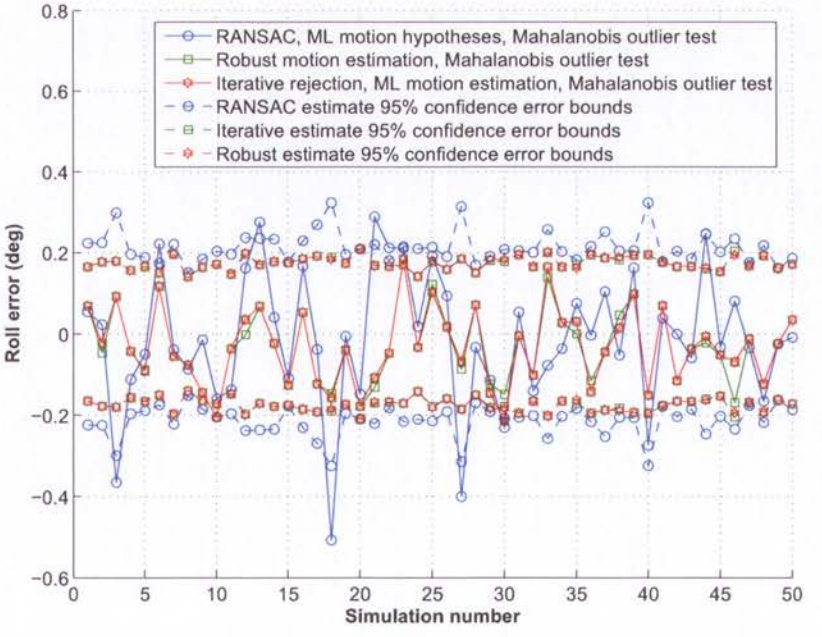


Figure F.4: Roll Euler angle errors in 50 odometry simulations with outliers.

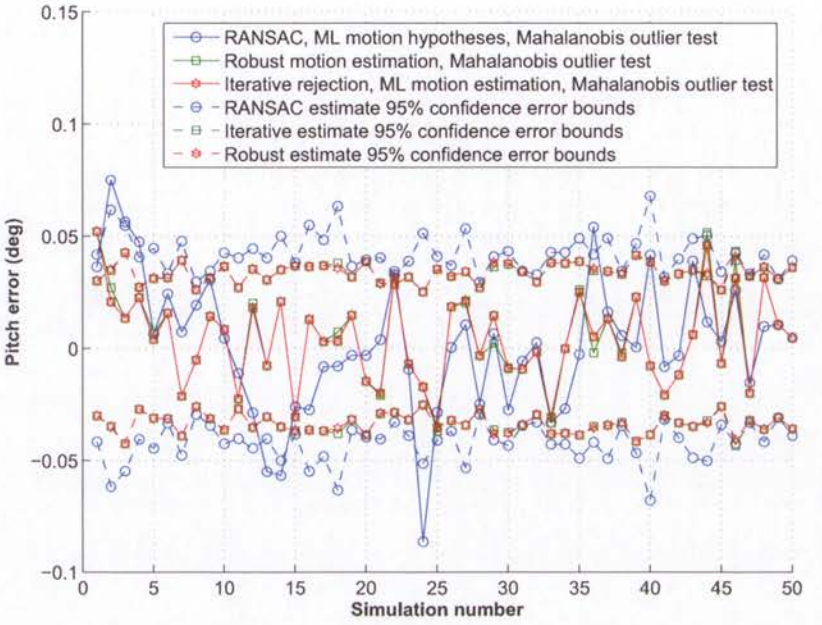


Figure F.5: Pitch Euler angle errors in 50 odometry simulations with outliers.

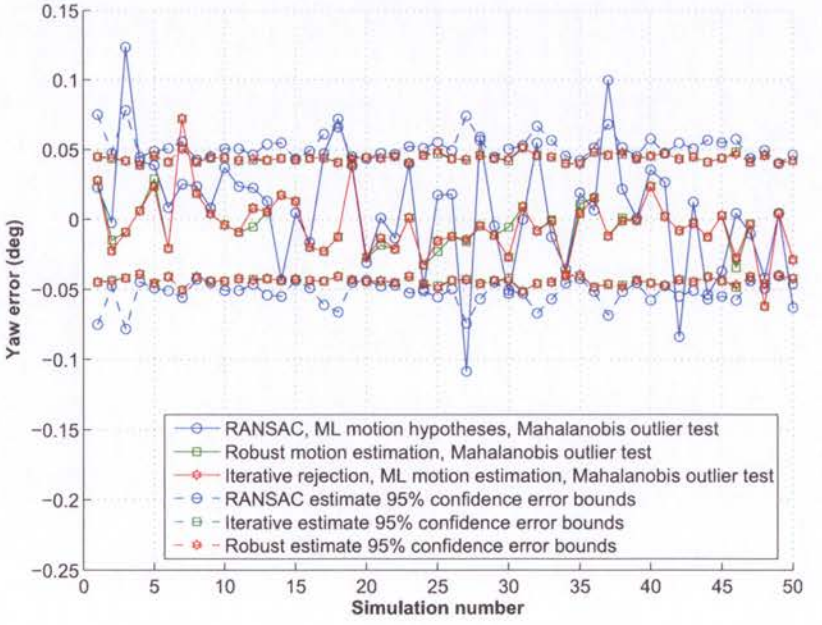


Figure F.6: Yaw Euler angle errors in 50 odometry simulations with outliers.

F.2 Loop-closure Simulations

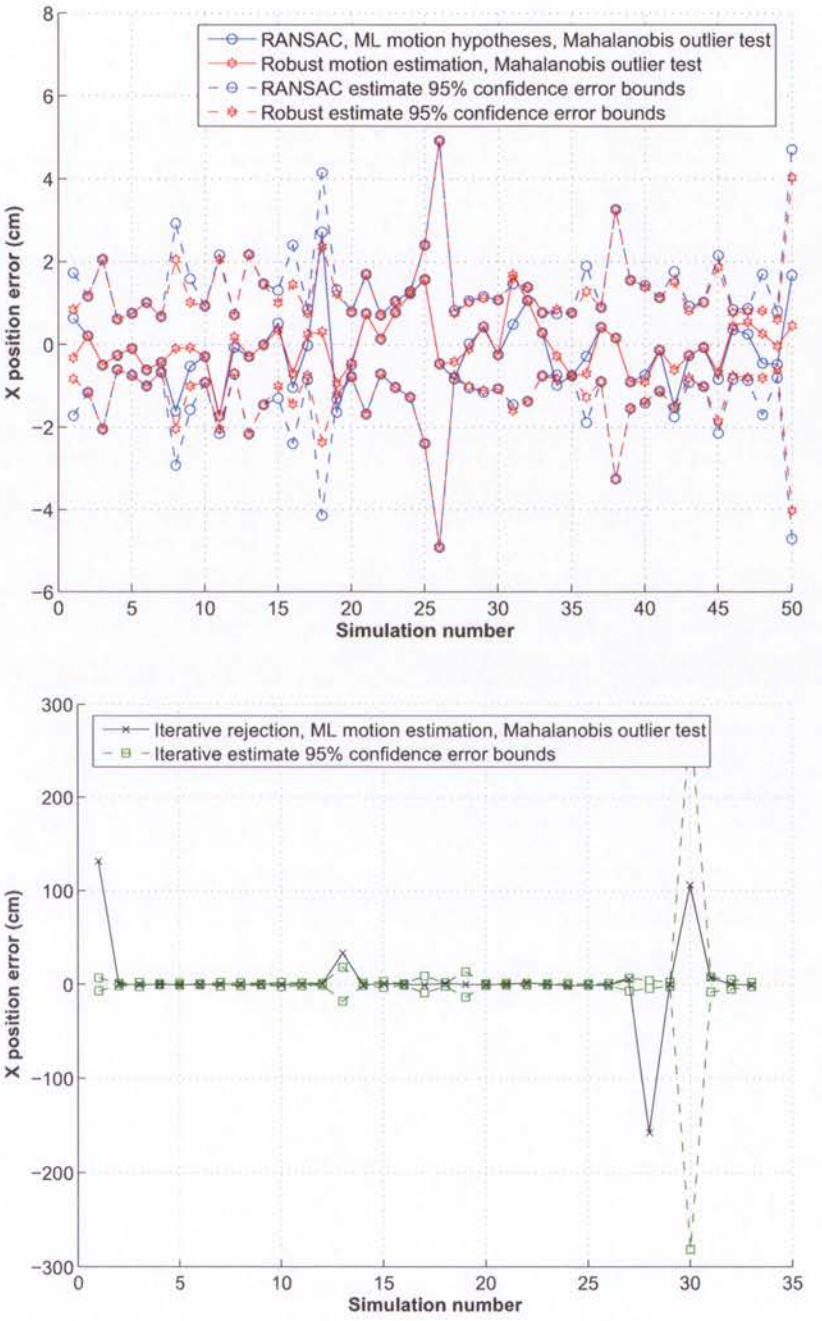


Figure F.7: X-position errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

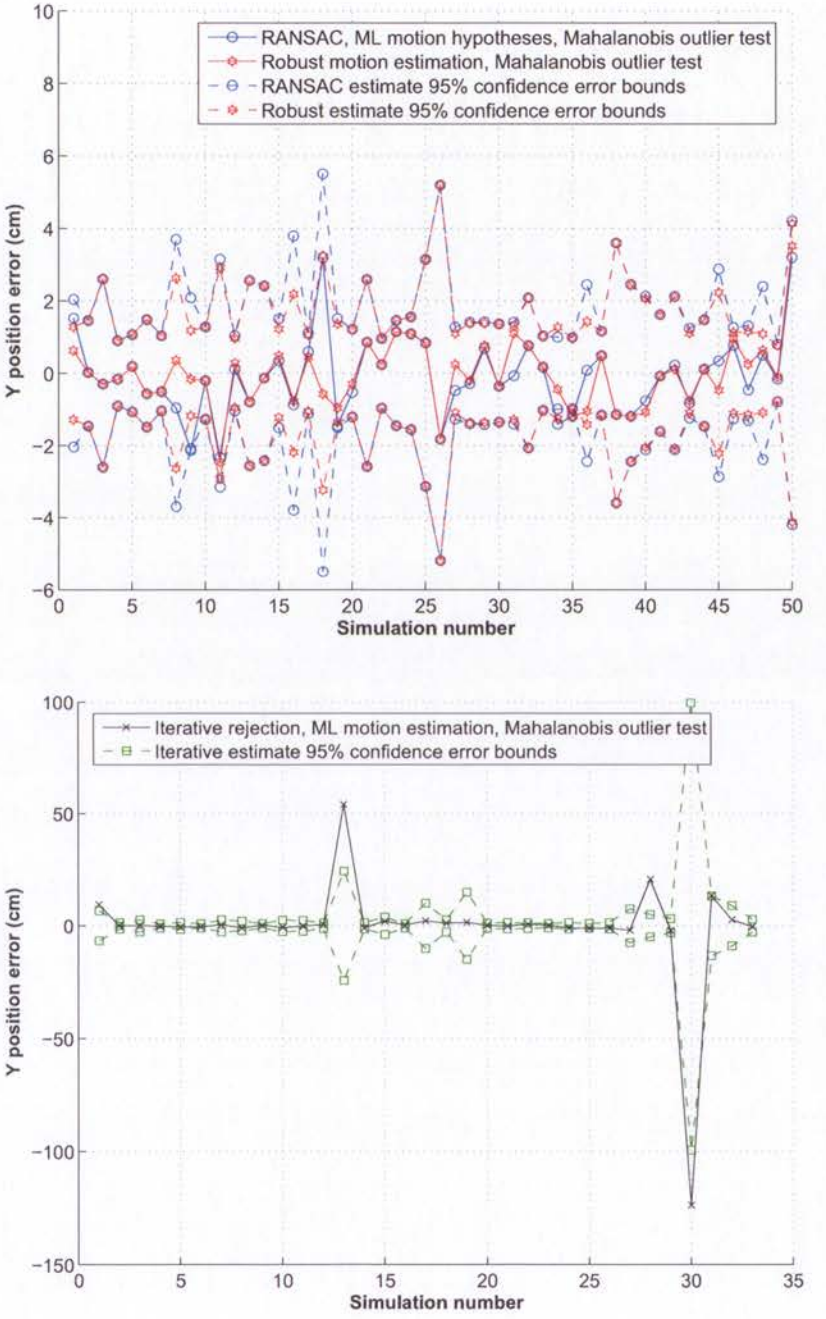


Figure F.8: Y-position errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

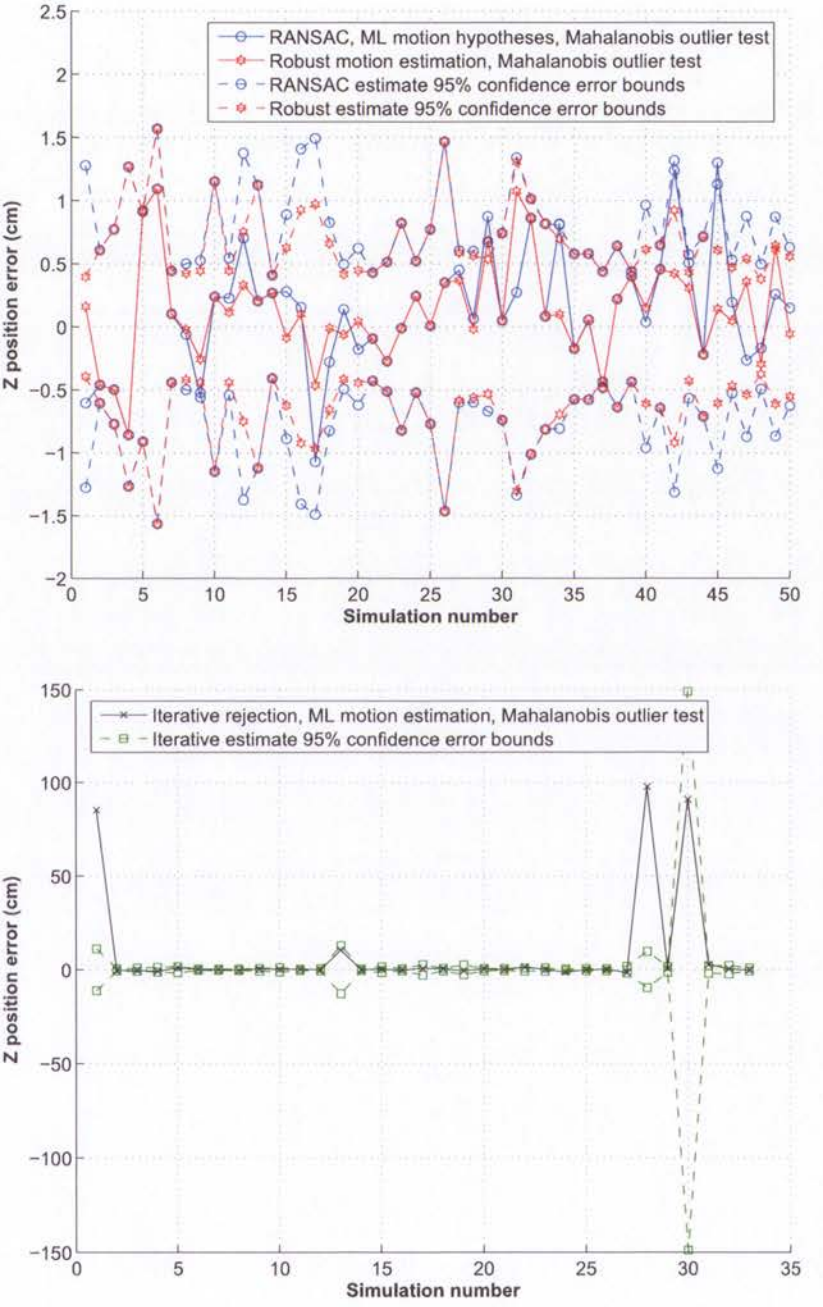


Figure F.9: Z-position errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

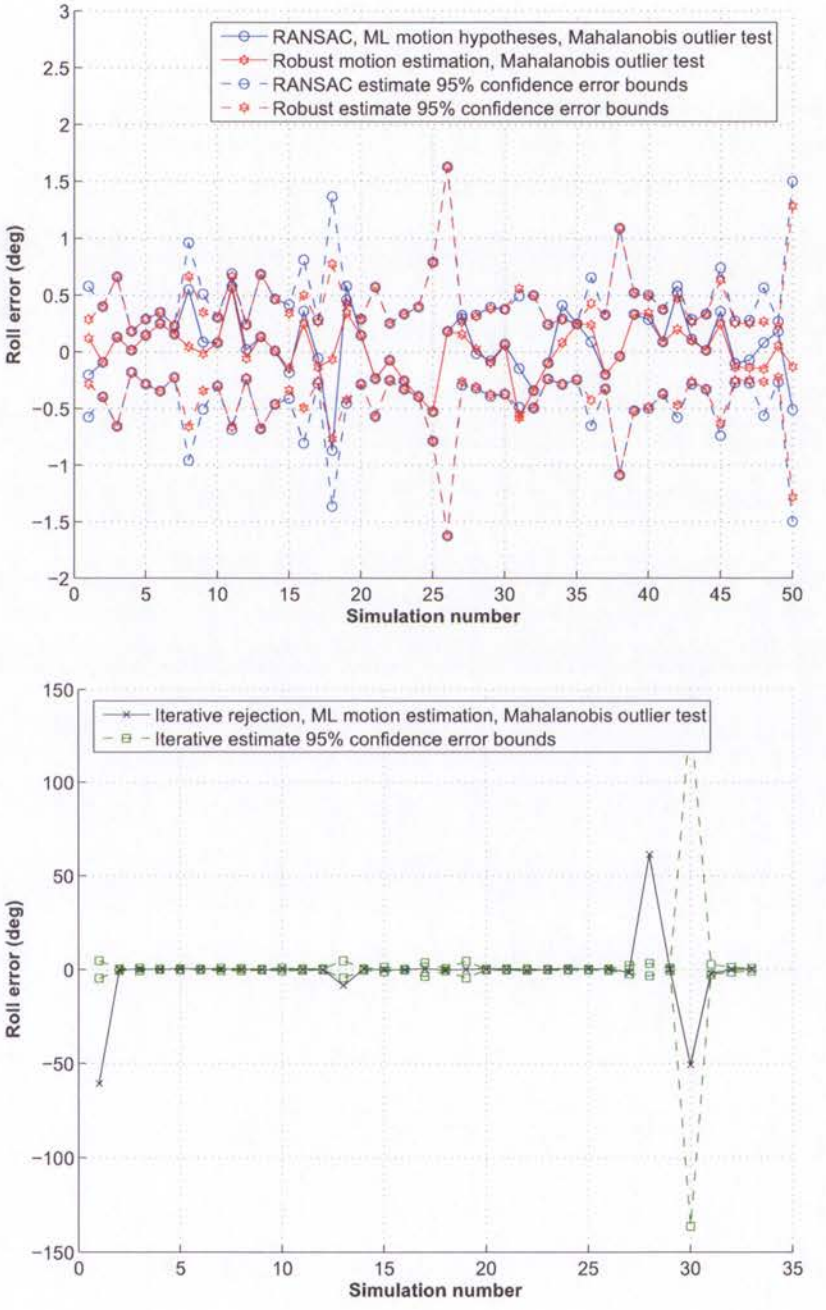


Figure F.10: Roll Euler angle errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

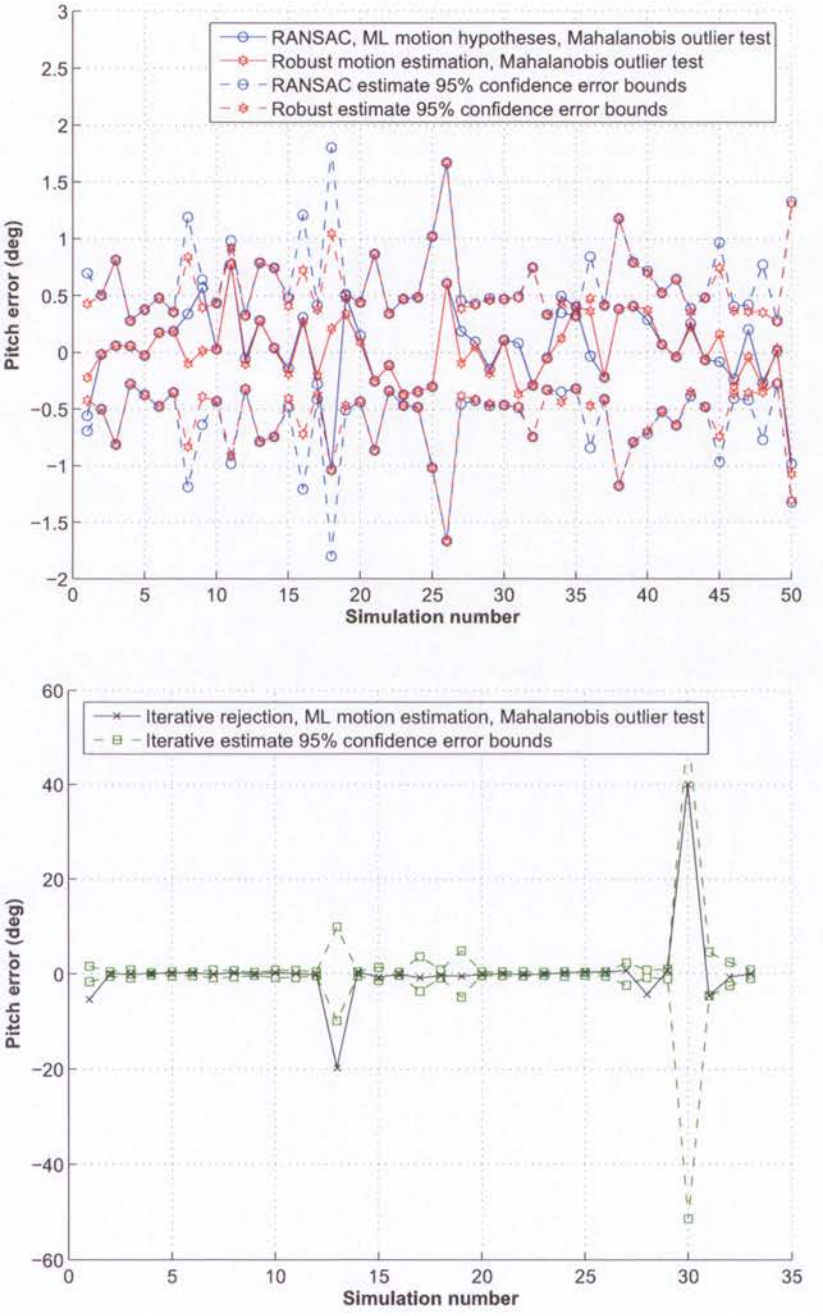


Figure F.11: Pitch Euler angle errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

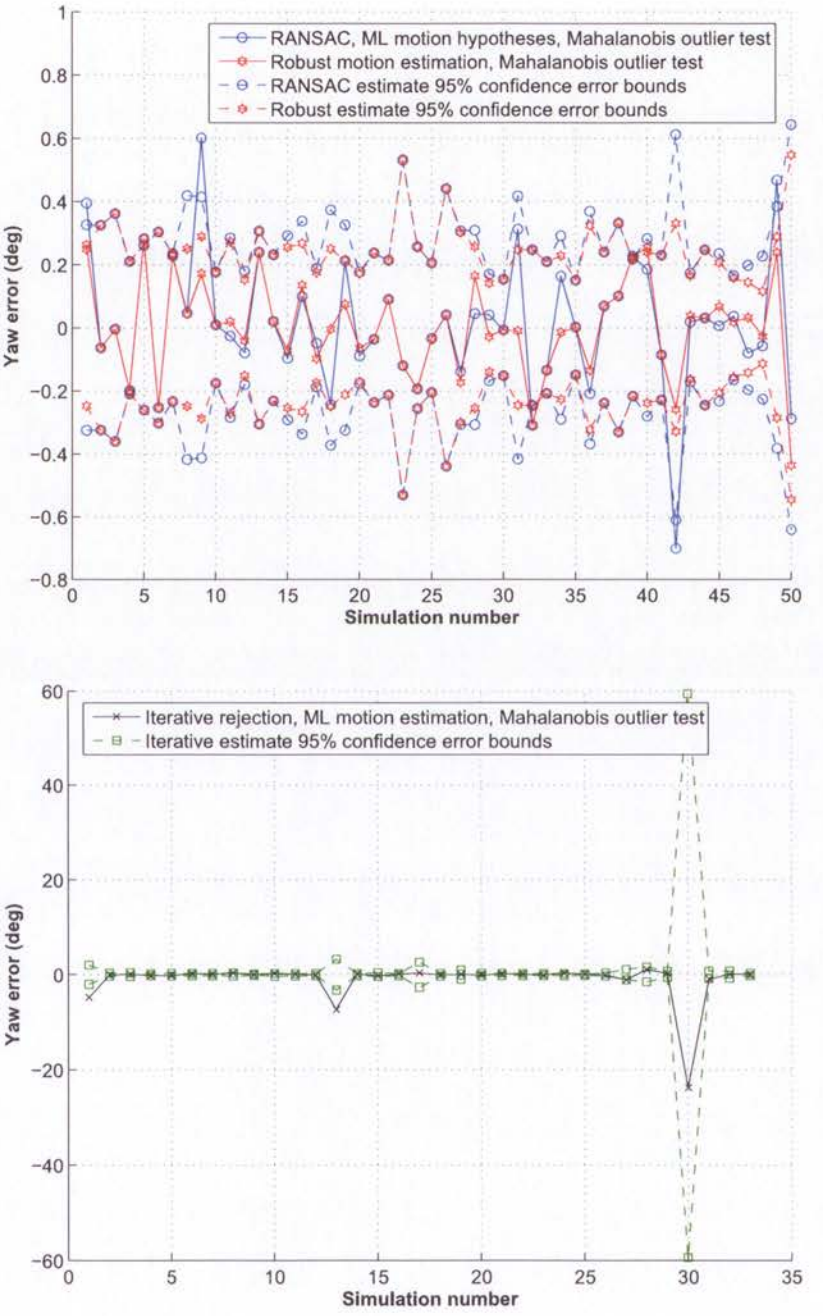


Figure F.12: Yaw Euler angle errors in 50 loop-closure simulations with outliers. Note that different scales are used in the upper and lower plots.

Bibliography

[1] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive GPS. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 3, pages 1063–1068, 2006.

[2] T. Bailey. Constrained initialisation for bearing-only SLAM. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 2, pages 1966–1971. IEEE, 2003.

[3] R.D. Ballard, A.M. McCann, D. Yoerger, L. Whitcomb, D. Mindell, J. Oleson, H. Singh, B. Foley, J. Adams, D. Piechota, and C. Giangrande. The discovery of ancient history in the deep sea using advanced deep submergence technology. *Deep Sea Research Part I: Oceanographic Research Papers*, 47(9):1591–1620, September 2000.

[4] R.D. Ballard, L.E. Stager, D. Master, D. Yoerger, D. Mindell, L.L. Whitcomb, H. Singh, and D. Piechota. Iron age shipwrecks in deep water off Ashkelon, Israel. *American Journal of Archaeology*, 106(2):151–168, 2002.

[5] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. Ninth European Conference on Computer Vision*, pages 404–417. Springer, 2006.

[6] P.J. Besl and N.D. MacKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992.

[7] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 2, pages 1899–1906. IEEE, 2003.

[8] N.A. Brokloff. Matrix algorithm for doppler sonar navigation. In *Proc. IEEE/MTS OCEANS*, volume 3, pages 378–383, September 1994.

[9] N.A. Brokloff. Dead reckoning with an ADCP and current extrapolation. In *Proc. MTS/IEEE OCEANS Conference*, volume 2, pages 994–1000, October 1997.

[10] D. C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37: 855–866, 1971.

[11] M. Bryson and S. Sukkarieh. Bearing-only SLAM for an airborne vehicle. In *Proc. Australian Conf. on Robotics and Automation*. Australian Robotics Association, 2005.

[12] Y. Chen, T.A. Davis, W.W. Hager, and S. Rajamanickam. Algorithm 8xx: Cholmod, supernodal sparse cholesky factorization and update/downdate. Technical Report TR-2006-005, Department of Computer and Information Science and Engineering, University of Florida.

[13] Y. Cheng, M. Maimone, and L. Matthies. Visual odometry on the Mars exploration rovers. In *IEEE Conference on Systems, Man and Cybernetics*, volume 1, pages 903–910, 2005.

[14] T.A. Davis. *Direct Methods for Sparse Linear Systems.* SIAM, 2006.

[15] T.A. Davis and W.W. Hager. Row modifications of a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 26(3):621–639, 2005.

[16] T.A. Davis and W.W. Hager. Modifying a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 20(3):606–627, 1999.

[17] F. Dellaert. Square root SAM. In *Proceedings of Robotics: Science and Systems*, June 2005.

[18] F. Dellaert and M. Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *Intl. Journal of Robotics Research*, 25(12): 1281–1203, 2006.

[19] M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csobra. A solution to the simultaneous localization and map building (SLAM) problem. In *IEEE Transactions on Robotics and Automation*, volume 17(3), pages 229–241, Sydney, Australia, June 2001.

[20] G. Dudek, M. Jenkin, C. Parahacs, A. Hogue, J. Sattar, P. Giguere, A. German, H. Liu, S. Sanderson, A. Ripsman, S. Simhon, L.A. Torres, E. Milios, P. Zhang, and I. Rekletis. A visually guided swimming robot. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3604 – 3609. IEEE/RSJ, 2005.

[21] M. Dunbabin, P. Corke, and G. Buskey. Low-cost vision-based AUV guidance system for reef navigation. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 7–12, New Orleans, April 2004.

[22] M. Dunbabin, J. Roberts, K. Usher, and P. Corke. A new robot for environmental monitoring on the great barrier reef. In *Proc. Australian Conf. on Robotics and Automation*, December 2004.

[23] M. Dunbabin, K. Usher, and P. Corke. Visual motion estimation for an autonomous underwater reef monitoring robot. In *Proceedings of the International Conference on Field and Service Robotics*, pages 57–68, Port Douglas, Australia, July 2005.

[24] C. Engels, H. Stewénius, and D. Nistér. Bundle adjustment rules. In *Photogrammetric Computer Vision*, September 2006.

[25] A.M. Erisman and W.F. Tinney. On computing certain elements of the inverse of a sparse matrix. *Communications of the ACM*, 18(3):177–179, 1975.

[26] R. Eustice, O. Pizarro, and H. Singh. Visually augmented navigation in an unstructured environment using a delayed state history. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 1, pages 25–32. IEEE, 2004.

[27] R.M. Eustice. *Large-Area Visually Augmented Navigation for Autonomous Underwater Vehicles.* PhD thesis, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution, 2005.

[28] R.M. Eustice, H. Singh, and J.J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, 2006.

[29] R.M. Eustice, H. Singh, J.J. Leonard, and M.R. Walter. Visually mapping the RMS Titanic: conservative covariance estimates for SLAM information filters. *Intl. J. Robotics Research*, 25(12):1223–1242, 2006.

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

[31] T. Fitzgibbons and E. Nebot. Bearing-only SLAM using colour-based feature tracking. In *Proc. Australian Conf. on Robotics and Automation.* Australian Robotics Association, 2002.

[32] S. Fleischer. *Bounded-Error Vision-Based Navigation of Autonomous Underwater Vehicles.* PhD thesis, Stanford University, 2000.

[33] S. Fleischer, S.M. Rock, and R. Burton. Global position determination and vehicle path estimation from a vision sensor for real-time video mosaicking and navigation. In *Proc. MTS/IEEE OCEANS*, volume 1, pages 641–647, 1997.

[34] U. Frese. A proof for the approximate sparsity of SLAM information matrices. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 329–335. IEEE, 2005.

[35] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, 2005.

[36] C. Georgiades, A. Hogue, H. Liu, A. Ripsman, R. Sim, L.A. Torres, P. Zhang, C. Prahacs, M. Buehler, G. Dudek, M. Jenkin, and E. Milios. Aqua: an aquatic walking robot. Technical report, Faculty of Computer Science, Dalhousie University, 2003.

[37] C. Georgiades, A. German, A. Hogue, H. Liu, C. Prahacs, A. Ripsman, R. Sim, L.A. Torres, P. Zhang, M. Buehler, G. Dudek, M. Jenkin, and E. Milios. Aqua: an aquatic walking robot. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3525–3531. IEEE/RSJ, 2004.

[38] N. Gracias and J. Santos-Victor. Underwater video mosaics as visual navigation maps. *Computer Vision and Image Understanding*, 79(1):66–91, July 2000.

[39] N. Gracias, S. van der Zwaan, A. Bernardino, and J. Santos-Victor. Results on underwater mosaic-based navigation. In *Proc. MTS/IEEE OCEANS*, volume 3, pages 1588–1594, October 2002.

[40] N. Gracias, S. van der Zwaan, A. Bernardino, and J. Santos-Victor. Mosaic based navigation for autonomous underwater vehicles. *IEEE Journal of Oceanic Engineering*, 28(4):609–624, 2003.

[41] J. Guivant and E.M. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. In *IEEE Transactions on Robotics and Automation*, volume 17, pages 242–257, 2001.

[42] P.E. Hagen and N. Storkersen. Network centric warfare with autonomous underwater vehicles - results from experimentation with HUGIN 1000. In *Proc. MTS/IEEE OCEANS*, volume 2, pages 2772–2775, 2005.

[43] P.E. Hagen, N. Storkersen, and K. Vestgard. The HUGIN underwater vehicle for forward mine hunting operations. In *Proc. UDT Europe*, 2001.

[44] P.E. Hagen, N. Storkersen, K. Vestgard, P. Kartvedt, and G. Sten. AUV based mine hunting demonstrated from MCMV. In *Proc. UDT Europe*, 2002.

[45] P.E. Hagen, N. Storkersen, K. Vestgard, and P. Kartvedt. The HUGIN 1000 autonomous underwater vehicle for military applications. In *Proc. MTS/IEEE OCEANS Conference*, volume 2, pages 1141–1145, 2003.

[46] P.E. Hagen, N. Storkersen, K. Vestgard, P. Kartvedt, and G. Sten. Operational military use of the HUGIN AUV in Norway. In *Proc. UDT Europe*, 2003.

[47] P.E. Hagen, N Storkersen, B.-E. Marthinsen, G G. Sten, and K. Vestgard. Military operations with HUGIN AUVs: Lessons learned and the way ahead. In *Proc. IEEE OCEANS Europe*, volume 2, pages 810–813, 2005.

[48] R.M. Haralick, H. Joo, C.N. Lee, X. Zhuang, V.G. Vaidya, and M.B. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(6):1426–1446, November 1989.

[49] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[50] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

[51] R. Henthorn, D.W. Caress, H. Thomas, R. Mcewen, W.J. Kirkwood, C.K. Paull, and R. Keaten. High-resolution multibeam and subbottom surveys of submarine canyons, deep-sea fan channels, and gas seeps using the MBARI mapping AUV. In *Proc. MTS/IEEE OCEANS*, 2006.

[52] D.C. Hoaglin, F. Mosteller, and J.W. Tukey. *Understanding Robust and Exploratory Data Analysis.* John Wiley & Sons, 1983.

[53] A. Hogue and M. Jenkin. Development of an underwater vision sensor for 3D reef mapping. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5351–5356. IEEE/RSJ, 2006.

[54] A. Hogue, A. German, J. Zacher, and M. Jenkin. Underwater 3D mapping: Experiences and lessons learned. In *Proc. Canadian Conference on Computer and Robot Vision*. IEEE, 2006.

[55] P.J. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

[56] S. Ingram. Minimum degree reordering algorithms: A tutorial. Online Only, 2006. URL `http://www.cs.ubc.ca/~sfingram/cs517_final.pdf`.

[57] R. Karlsson and F. Gustafsson. Particle filter for underwater terrain navigation. In *Proc. IEEE Workshop on Statistical Signal Processing*, pages 526–529, 2003.

[58] R. Karlsson, F. Gustafsson, and T. Karlsson. Particle filtering and Cramer-Rao lower bound for underwater navigation. In *Internal Report LiTH-ISY-R-2474*. Dept. of Electrical Engineering, Linkoping University, 2002.

[59] J.C. Kinsey, R.M. Eustice, and L.L. Whitcomb. A survey of underwater vehicle navigation: Recent advances and new challenges. In *Proc. of the IFAC Conference of Manoeuvring and Control of Marine Craft*, 2006.

[60] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, and B. Gerkey. Outdoor mapping and navigation using stereo vision. In *Proceedings of the International Symposium on Experimental Robotics*, 2006.

[61] S. Lacroix, A. Mallet, and R. Chatila. Rover self localization in planetary-like environments. In *5th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, pages 443–440, 1999.

[62] J.J. Leonard, A.A. Bennett, C.M. Smith, and H.J.S. Feder. Autonomous underwater vehicle navigation. Technical Report 98-1, MIT Marine Robotics Laboratory.

[63] J.J. Leonard, H.F. Durrant-Whyte, and I.J. Cox. Dynamic map building for an autonomous mobile robot. *Intl. Journal of Robotics Research*, 11(4):286– 298, 1992.

[64] M.I.A. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. `http://www.ics.forth.gr/~lourakis/levmar/`. Last Accessed on 29 September 2006.

[65] M.I.A. Lourakis and A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical Report 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece, Aug. 2004. Available from `http://www.ics.forth.gr/~lourakis/sba`.

[66] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[67] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, 1981.

[68] K. Madsen, H.B. Nielsen, and O. Tingleff. Methods for non-linear least squares problems (2nd ed.), 2004.

[69] T. Maki, H. Kondo, T. Ura, and T. Sakamaki. Navigation of an autonomous underwater vehicle for photo mosaicing of shallow vent areas. In *Proc. IEEE OCEANS Asia*, 2006.

[70] T. Maki, H. Kondo, T. Ura, and T. Sakamaki. Photo mosaicing of Tagiri shallow vent area by the AUV "Tri-Dog 1" using a SLAM based navigation scheme. In *Proc. MTS/IEEE OCEANS*, 2006.

[71] A. Mallet, S. Lacroix, and L. Gallo. Position estimation in outdoor environments using pixel tracking and stereovision. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 3519–3524, 2000.

[72] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. British Machine Vision Conference*, pages 384–392. British Machine Vision Association, 2002.

[73] L. Matthies and S.A. Shafer. Error modeling in stereo navigation. *IEEE Transactions on Robotics and Automation*, 3(3):239–248, 1987.

[74] K. Mikolajczk and C. Schmid. A performance evaluation of local descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 257–263. IEEE, 2003.

[75] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. 65(1-2):43–72, 2005.

[76] A. Milella and R. Siegwart. Stereo-based ego-motion estimation using pixel tracking and iterative closest point. In *Proc. IEEE International Conference on Computer Vision Systems*, 2006.

[77] D. Mindell and B. Bingham. New archaeological uses of autonomous underwater vehicles. In *MTS/IEEE OCEANS Conference and Exhibition*, volume 1, pages 555–558, 2001.

[78] H.P. Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover.* PhD thesis, Stanford University, 1980.

[79] H.B. Nielsen. Damping parameter in Marquardt's method. Technical Report IMM-REP-1999-05, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, April 1999.

[80] H. Niessner and K. Reichert. On computing the inverse of a sparse matrix. *International Journal for Numerical Methods In Engineering*, 19(10):1513–1526, 1983.

[81] D. Nistér. Preemptive RANSAC for live structure and motion estimation. In *Proc. Ninth International Conference on Computer Vision*, volume 1, pages 199–206, 2003.

[82] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 652–659, 2004.

[83] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1), 2006.

[84] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 1999.

[85] C. Olson, L. Matthies, M. Schoppers, and M. Maimone. Robust stereo ego-motion for long distance navigation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 453–458, 2000.

[86] C. Olson, H. Abi-Rached, M. Ye, and J.P. Hendrich. Wide-baseline stereo vision for Mars rovers. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1302–1307, 2003.

[87] C. Olson, L. Matthies, M. Schoppers, and M. Maimone. Rover navigation using stereo ego-motion. *Robotics and Autonomous Systems*, 43(4):215–229, 2003.

[88] R.B. Porter and N.W. Bergmann. A generic implementation framework for FPGA based stereo matching. In *Proc. IEEE Region 10 Annual Conference Speech and Image Technologies for Computing and Telecommunications*, volume 2, pages 461–464. IEEE, 1997.

[89] W.J.J. Rey. *Introduction to Robust and Quasi-Robust Statistical Methods.* Springer-Verlag, 1983.

[90] P. Rigby and S.B. Williams. Adaptive sensing for localisation of an autonomous underwater vehicle. In *Proc. Australian Conf. on Robotics and Automation.* Australian Robotics Association, 2005.

[91] P. Rigby, O. Pizarro, and S.B. Williams. Towards geo-referenced AUV navigation through fusion of USBL and DVL measurements. In *Proc. MTS/IEEE OCEANS*, 2006.

[92] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.

[93] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[94] C. Schmid, R. Mohr, and C. Bauckhage. Comparing and evaluating interest points. In *Sixth International Conference on Computer Vision*, pages 230–235, 1998.

[95] J.R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University, August 1994.

[96] M.R. Shortis and E.S. Harvey. Design and calibration of an underwater stereo video system for the monitoring of marine fauna populations. In *International Archives Photogrammetry and Remote Sensing*, volume 32, pages 792–799, 1998.

[97] H. Singh, J. Catipovic, R. Eastwood, L. Freitag, H. Henriksen, F. Hover, D. Yoerger, J.G. Bellingham, and B.A. Moran. An integrated approach to multiple AUV communications, navigation and docking. In *Proc. IEEE Oceanic Engineering Society OCEANS*, pages 59–64, 1996.

[98] H. Singh, R. Armstrong, F. Gilbes, R. Eustice, C. Roman, O. Pizarro, and J. Torres. Imaging coral I: Imaging coral habitats with the SeaBED AUV. *Subsurface Sensing Technologies and Applications*, 5(1):25–42, January 2004.

[99] H. Singh, A. Can, R. Eustice, S. Lerner, N. McPhee, O. Pizarro, and C. Roman. Seabed AUV offers new platform for high-resolution imaging. In *Eos Transactions of the American Geophysical Union*, volume 85, pages 289,294–295, November 2004.

[100] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. *Autonomous Mobile Robots : Perception, Mapping and Navigation*, 1:323–330, 1987.

[101] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, pages 167–193, 1990.

[102] S.M. Smith and D. Kronen. Experimental results of an inexpensive short baseline acoustic positioning system for AUV navigation. In *Proc. MTS/IEEE OCEANS Conference*, volume 1, pages 714–720, 1997.

[103] N. Sünderhauf and P. Protzel. Towards using sparse bundle adjustment for robust stereo odometry in outdoor terrain. In *Proc. Towards Autonomous Robotic Systems*, pages 206–213, 2006.

[104] N. Sünderhauf, K. Konolige, S. Laxroix, and P. Protzel. Visual odometry using sparse bundle adjustment on an autonomous outdoor vehicle. *Autonome Mobile Systeme 2005*, pages 157–163, 2005.

[105] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Intl. Journal of Robotics Research*, 23(7-8):693–716, 2004.

[106] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – A modern synthesis. In *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[107] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 (4):376–380, April 1991.

[108] L. Whitcomb, D. Yoerger, H. Singh, and D. Mindell. Towards precision robotic maneuvering, survey, and manipulation in unstructured undersea environments. *The Eighth International Symposium on Robotics Research*, pages 346–353, 1998.

[109] L. Whitcomb, D. Yoerger, and H. Singh. Combined doppler/LBL based navigation of underwater vehicles. *Proc. of the 11th International Symposium on Unmanned Untethered Submersible Technology*, August 1999.

[110] L. Whitcomb, D. Yoerger, H. Singh, and J. Howland. Advances in underwater robot vehicles for deep ocean exploration: Navigation, control and survey operations. *The Ninth International Symposium on Robotics Research*, pages 346–353, 1999.

[111] S.B. Williams. A terrain-aided tracking algorithm for marine systems. In *Proc. International Conference on Field and Service Robotics*, pages 55–60, 2003.

[112] S.B. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems.* PhD thesis, University of Sydney, Australian Centre for Field Robotics, 2001.

[113] S.B. Williams and I. Mahon. Design of an unmanned underwater vehicle for reef surveying. In *Proc. of the 3rd IFAC Symposium on Mechatronic System*, 2004.

[114] S.B. Williams, G. Dissanayake, and H.F. Durrant-Whyte. Towards terrain-aided navigation for underwater robotics. *Advanced Robotics*, 15(5):533–550, 2001.

[115] S.B. Williams, M.W.M.G. Dissanayake, and H.F. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 1, pages 406–411, 2002.

[116] D.R. Yoerger, A.M. Bradley, M-H. Cormier, W.B.F. Ryan, and B.B. Walden. Fine-scale seafloor survey in rugged deep-ocean terrain with an autonomous robot. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 2, pages 1787–1792. IEEE, 2000.