

# Real-time Stereo Visual SLAM

Stephen J. Thomas

Department of Engineering and Physical Sciences  
Heriot-Watt University



Department of Computer Architecture and Technology  
Universitat de Girona



Le2i Laboratory  
Universite de Bourgogne



A Thesis Submitted for the Degree of  
MSc Erasmus Mundus in Vision and Robotics (VIBOT)

· 2008 ·

## Abstract

Simultaneous localisation and mapping (SLAM) has been the focus of intensive research in the last decade due to the potential benefits it offers to the field of autonomous mobile robotics. SLAM is concerned with the ability of an autonomous vehicle to navigate through an unexplored environment and incrementally construct a map of the environment and localise itself within this map. This thesis describes an entirely vision-based, large-area, 6DoF SLAM system that was developed specifically for real-time deployment on an autonomous underwater vehicle (AUV) equipped with a calibrated stereo system. This SLAM system is based on the extended Kalman filter (EKF) and incorporates a novel approach to landmark description and data association in which landmarks are essentially local submaps that consist of a cloud of 3D points and their associated SIFT or SURF descriptors. Furthermore, landmarks are sparsely distributed in the constructed map which greatly simplifies and accelerates data association and map updates. In addition to performing localisation based on landmark observations the system also performs visual odometry and predicts vehicle motion using a constant-velocity model. For a simulated 87m long 3D loop trajectory the mean squared localisation error of the system was 3.16 and the maximum absolute error in roll, pitch and yaw angles was  $11.6^\circ$ ,  $24.3^\circ$  and  $24.4^\circ$  respectively when the stereo and landmark correspondences contained Gaussian noise with a standard deviation of 0.1 pixels and 10% of correspondences were outliers. This thesis represents an important contribution to entirely vision-based 6DoF SLAM as very few implementations currently exist, and the approach utilised in this thesis achieves comparable results and has the potential to operate in real-time.

*I have yet to see any problem, however complicated, which, when you looked at it the right way, did not become still more complicated. . . .*

Paul Alderson

# Contents

<b>Acknowledgements</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 What is SLAM?	1
1.2 Why is SLAM Required?	5
1.3 Why Visual SLAM?	5
1.4 Project Goals	5
<b>2 Literature review</b>	<b>7</b>
2.1 Computational Solutions to the SLAM Problem	7
2.1.1 Extended Kalman Filter (EKF-SLAM)	8
2.1.2 Rao-Blackwellised Particle Filter (FastSLAM)	13
2.2 Feature Recognition and Tracking	15
2.2.1 Colour Histograms and KLT Tracker	15
2.2.2 Precise Harris Corner Detector	17
2.2.3 Scale Invariant Feature Transform (SIFT)	17
2.2.4 Image Patches	18
2.2.5 Comparison of Interest Point Detectors	19
2.2.6 Non-geometric Landmarks	19
2.3 Data Association and Loop Closure Techniques	20

2.3.1	Maximum Likelihood and Gated Nearest Neighbour . . . . .	21
2.3.2	Robust Data Association . . . . .	22
2.3.3	Visual Tracking and Loop Closure . . . . .	23
2.4	Vision-based Motion Estimation . . . . .	24
2.5	Real-time 6-DOF vision-based SLAM Implementations . . . . .	27
2.6	Available Resources . . . . .	33
<b>3</b>	<b>Real-time Stereo Visual SLAM</b>	<b>34</b>
3.1	Proposed Solution . . . . .	34
3.2	Implementation . . . . .	35
3.2.1	Landmark Observation . . . . .	35
3.2.2	Data Association . . . . .	39
3.2.3	Motion Estimation . . . . .	41
3.2.4	EKF-SLAM and UKF-SLAM . . . . .	42
3.2.5	Simulation and Testing Environment . . . . .	46
<b>4</b>	<b>Results</b>	<b>48</b>
<b>5</b>	<b>Conclusions</b>	<b>55</b>
	<b>Bibliography</b>	<b>60</b>

# List of Figures

1.1	An AUV moves through an unexplored environment simultaneously estimating its position and landmark locations based on relative measurements of the landmarks [1] . . . . .	2
2.1	Motion Recovery Methods [2] . . . . .	26
3.1	Epipolar geometry - Two 3D points $X_1$ and $X_2$ are projected into the left and right image planes using camera projection matrices $P$ and $P'$ to obtain $x_1, x_2$ and $x_1', x_2'$ respectively. An epipolar plane is defined by each 3D point together with the two camera centres and the associated epipolar lines correspond to the intersection of this plane with the image planes. In the absence of noise any point which lies on the epipolar plane (at different scene depths) will be projected onto the associated epipolar lines in each image plane. [3] . . . . .	36
3.2	Feature matching and outlier removal - The epipolar lines (black) are shown for each matched point (dots) and a correspondence line (red) is shown for the matched left and right points which are sufficiently close to the corresponding epipolar lines . . . . .	37
3.3	Scenario Generator . . . . .	47
4.1	Comparison of EKF and UKF in noiseless conditions . . . . .	49
4.2	Comparison of EKF and UKF in presence of Gaussian noise ( $\sigma = 0.1$ ) . . . . .	50

4.3	Comparison of EKF and UKF in presence of Gaussian noise ( $\sigma = 0.2$ ) . . . . .	51
4.4	Robustness of EKF to outliers . . . . .	53

# List of Tables

- 2.1 Strengths and weaknesses of SLAM solutions . . . . . 16
- 2.2 Strengths and weaknesses of interest point detectors . . . . . 20
- 2.3 Strengths and weaknesses of data association approaches . . . . . 23

# Acknowledgements

I would like to acknowledge the guidance and support of my supervisors Dr. Yvan Petillot and Dr. Joaquim Salvi and also the encouragement and support from my girlfriend Chloe Wharton, good friend Luca Zappella and my loving family.

# Chapter 1

## Introduction

### 1.1 What is SLAM?

Robots are increasingly deployed in diverse fields of service ranging from housekeeping to space exploration. The majority of these applications require the robot to extract useful information from its sensors in order to make its own decisions, therefore these robots exercise a high degree of autonomy. One of the fundamental requirements of an autonomous robot is the ability to navigate within a dynamic, unexplored environment. Navigating successfully from one place to another essentially involves the resolution of three problems:

1. Determining where the robot is (localisation)
2. Determining where the robot would like to go (goal recognition)
3. Determining how the robot should get there (path planning)

In this paper we are primarily concerned with the first problem. To determine their location most modern autonomous robots employ a variety of cheap sensors and efficient computers to construct a map of the explored section of the environment and simultaneously estimate the location of the robot within this map. In most literature this approach is referred to as simultaneous localisation and mapping (SLAM). SLAM is a rather complex problem due to inherent uncertainties associated with the robots motion and measurement sensors. In real world applications SLAM implementations are also complicated by restrictions on available computational power, difficulties in estimating relative motion and the challenge of identifying unique and easily recognisable landmarks in the environment. However, the solution to the SLAM problem is considered the "Holy Grail" of robotics for many researchers as it is the key

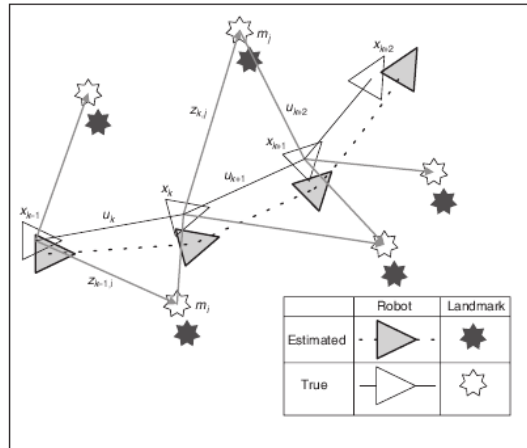


Figure 1.1: An AUV moves through an unexplored environment simultaneously estimating its position and landmark locations based on relative measurements of the landmarks [1]

to producing truly autonomous robots [1]. Consequently, each of these issues have received significant research interest and a diverse range of solutions have been proposed.

The overwhelming majority of successful SLAM algorithms utilise a probabilistic framework to describe the relationship between the locations of observed landmarks and the uncertainty of the individual landmark positions [1]. This probabilistic framework takes advantage of the fact that there is a high degree of correlation between the estimates of individual landmark locations [1]. By expressing mapping and localisation as a joint problem in which the state of the system is composed of both the vehicle pose and every estimated landmark position, it is possible to observe that these correlations grow as the state is updated after each observation [1]. More importantly, it is possible to see that as these correlations grow the system converges towards the true vehicle and landmark locations [1]. To better understand this concept the basic overall framework of the SLAM problem is presented in its simplest Bayesian form.

Consider an autonomous underwater vehicle (AUV) moving through an environment and making relative point measurements of a number of previously unobserved landmarks using a stereo camera system as illustrated in Figure 1.1. At a given time instant  $k$  the vehicle maintains the following variables:

- $x_k$  - A state vector which contains the estimated location and orientation of the vehicle
- $u_k$  - A control vector which contains the control outputs applied at time  $k - 1$  to move the vehicle to state  $x_k$  at time  $k$

- $m_k$  - A landmark vector which contains the estimated location of  $N$  stationary landmarks at time  $k$
- $z_k$  - An observation vector which contains the relative observation of  $n \leq N$  landmarks at time  $k$

In addition, the vehicle also maintains a history of the  $x_k$ ,  $u_k$  and  $z_k$  vectors from time 0 to time  $k$ , which are denoted by  $X_{0:k}$ ,  $U_{0:k}$  and  $Z_{0:k}$  respectively. The probabilistic SLAM framework requires that from these variables the probability distribution:

$$p(x_k, m_k | Z_{0:k}, U_{0:k}, x_0) \quad (1.1)$$

be computed at each time instant. In words, this probability distribution describes the joint posterior density of the estimated landmark positions, vehicle position and vehicle orientation at time  $k$ , given the complete history of landmark observations and control outputs along with the initial state of the vehicle. It is important to realise that given both an observation model and a state transition model (motion model) this probability distribution can be estimated recursively using Bayes theorem. In this case, the observation model is of the form:

$$P(z_k | x_k, m_k) \quad (1.2)$$

and essentially describes the probability of making an observation  $z_k$  given the current estimates of the vehicle state and landmark positions. The motion model is assumed to be a Markov process, thus the next state depends only on the current state and the control output and consequently the motion model is expressed in the form:

$$P(x_k | x_{k-1}, u_k) \quad (1.3)$$

The following prediction and correction procedure based on the motion and observation models can then be recursively applied to obtain the probability distribution of Equation 1.1:

1. Time update (Prediction):

$$P(x_k, m_k | Z_{0:k-1}, U_{0:k}, x_0) = \int P(x_k | x_{k-1}, u_k) P(x_{k-1}, m_k | Z_{0:k-1}, U_{0:k-1}, x_0) dx_{k-1} \quad (1.4)$$

2. Observation update (Correction):

$$P(x_k, m_k | Z_{0:k}, U_{0:k}, x_0) = \frac{P(z_k | x_k, m_k) P(x_k, m_k | Z_{0:k-1}, U_{0:k}, x_0)}{P(z_k | Z_{0:k-1}, U_{0:k})} \quad (1.5)$$

Therefore, assuming that the vehicle location is deterministic at every instant the estimated landmark locations can be obtained by computing the conditional density:

$$P(m_k | X_{0:k}, Z_{0:k}, U_{0:k}) \quad (1.6)$$

Conversely, assuming that the landmark locations are known with absolute certainty an accurate estimate of the vehicle location can be obtained by computing the probability distribution:

$$P(x_k | X_{0:k}, U_{0:k}, m_k) \quad (1.7)$$

However, in reality the landmark locations and vehicle position are never known with absolute certainty. What is known is that much of the error in the estimated landmark locations is common to all landmarks and arises from the error in the estimated vehicle position [1]. Therefore, the error in the landmark location estimates is highly correlated, which means the relative location of landmarks may be known with high accuracy, despite the fact that the absolute location of the landmarks is quite uncertain [1]. This is illustrated in Figure 1.1 where an AUV at location  $x_k$  observes two landmarks,  $m_j$  to its left and  $m_i$  to its right, and is able to make an accurate measurement of the relative location of the two landmarks that is quasi-independent of the vehicle frame [1]. Note that the measurements are not truly independent of the vehicle frame as the measurement error will be correlated through successive vehicle motions [1]. When the AUV arrives at location  $x_{k+1}$  it still observes landmark  $m_j$  but landmark  $m_i$  is not visible. However, due to the high degree of correlation between  $m_j$  and  $m_i$ , when the predicted location of  $m_j$  is updated, the result propagates back to update the unseen landmark  $m_i$  and the correlation continues to grow. Dissanayake et. al. [4] proved for the linear Gaussian case that this correlation increases monotonically as more observations are made, and consequently knowledge of the relative location of landmarks never diverges regardless of vehicle motion. Consequently, in this theoretical framework the localisation accuracy of the vehicle relative to the constructed map is bounded only by the quality of the map and the accuracy of the measurement sensor [1].

In practical SLAM implementations two of the most critical tasks are data association and detecting "Loop closure" [5]. Data association is the process of associating observations with map landmarks and creating new landmarks when necessary [5]. Detecting loop closure is essentially a specific case of the data association problem, and refers to the ability to correctly assert when the vehicle has returned to a previously visited location after completing a large loop [5]. Loop closure is critical as accurately closing a loop can drastically reduce the error in vehicle and landmark location estimates, while conversely, inaccurately closing a loop almost certainly leads to irrecoverable error [5]. As a result, loop closure is often treated as a separate problem to short-term data association [5].

## 1.2 Why is SLAM Required?

Satellite-based global positioning systems (GPS) are widely used for robot localisation, however, they are susceptible to signal loss due to intentional blocking, shading from buildings, trees, mountains and canyons and poor reception indoors [6]. Furthermore, GPS is completely unavailable in underwater and mine environments due to natural attenuation of the signal [6]. However, an accurate mapping of the environment and reliable localisation is essential for almost any task [6]. Therefore, in some cases SLAM is absolutely necessary and in others it is an essential backup to ensure GPS signal loss does result in failure to accomplish a task [6].

## 1.3 Why Visual SLAM?

Almost any type of sensor can be incorporated into the SLAM framework, but the most commonly used sensors can be categorised as being laser, sonar, inertia or vision based [7]. Laser ranging systems are active sensors that are highly accurate, but their point-to-point measurement characteristic limits their ability to perform object recognition and tracking [7]. Sonar-based systems are fast, cheap and have similar measurement and object recognition capabilities as vision, but they tend to produce less accurate and more noisy measurements [7]. Inertial sensors such as odometers are useful, however, a small error in a single measurement can have large effects on later position estimates [7]. In contrast, passive vision-based systems are cheap, light-weight, low power and offer long range, high resolution measurements [7]. Consequently, they are one of the mainstream perception techniques for SLAM systems [7]. Another reason for their popularity is the extensive amount of directly applicable knowledge that exists in the computer vision community for extracting and matching good visual features [8]. The computational complexity of some of these techniques exceeds the capacity of typical embedded computers carried by mobile vehicles so the constraints of real-time operation always need to be considered [8]. However, this represents a minor limitation and many of the most successful SLAM implementations use computer vision and state-of-the-art scene modeling, camera motion analysis and high-level feature extraction and description techniques [7].

## 1.4 Project Goals

The ultimate goal of this project was to implement a real-time visual SLAM algorithm which could be deployed in an autonomous underwater vehicle (AUV) equipped with forward and downward pointing stereo cameras. This system was required to use computer vision techniques for feature tracking and recognition to generate robust three-dimensional feature maps which are embedded within a probabilistic SLAM framework. In order to identify the best possible

approach to solving this problem a survey was compiled of the state-of-the-art in vision-based SLAM for mobile robots which operate in unstructured outdoor environments. From this survey it was evident that the intended application environment is particularly important as it dictates what type of features we can expect to observe and how they are distributed in the environment [7]. For example, indoors we can expect to see a large number of geometrical structures which can be easily tracked, but also many repeating structures which offer no global context [7]. In contrast, in an underwater environment we can expect to observe small regions with significant, distinctive texture such as reefs, but also vast regions which are almost completely devoid of features such as flat seabed [7]. Consequently, the solution that was proposed and has since been implemented was developed with careful consideration of the nature of the application environment.

To present the surveyed literature in a clear and concise way the material has been split into four broad categories:

1. Computational solutions to the SLAM problem: Including conventional approaches which utilise extended Kalman filters (EKF SLAM) and Rao-Blackwellised particle filters (Fast-SLAM) in addition to recent improvements such as linear-time state augmentation, sparsification in information form, partitioned updates, and submapping methods.
2. Feature recognition and tracking: Interest point detectors and feature descriptors for visual SLAM such as image patches, Harris corners, SIFT, SURF and SUSAN.
3. Data association and loop closure techniques.
4. Visual motion estimation techniques.

In the following chapter we will present the state-of-the-art in each of these categories, discuss their strengths and weaknesses and compare their suitability for our intended application. This literature review is then completed by presenting and analysing existing real-time visual SLAM implementations and available resources. In Chapter 3 we provide a brief outline of our initial proposal for a real-time visual SLAM system which combines both state-of-the-art ideas and our own innovations to obtain a SLAM solution which is customised for deployment on an AUV. Following this we provide a detailed description of the complete SLAM system that has been implemented based on this proposal. In Chapter 4 we provide a quantitative analysis of the systems performance in a series of simulated scenarios. Finally, in Chapter 5 we draw some conclusions about the current implementation and suggest directions for future work.

## Chapter 2

# Literature review

### 2.1 Computational Solutions to the SLAM Problem

The state-based formulation of the SLAM problem presented in Chapter 1 involves the estimation of a joint state composed of the vehicle location and pose, and the locations of all the observed stationary landmarks [9]. This inherently means that the size of state grows as new landmarks are observed and as a result the computational complexity of state updates increases rapidly [9]. However, the motion model (state transition model) only affects the vehicle location and pose, and the observation model can be applied independently to each vehicle-landmark pair [9]. Consequently, a wide variety of different techniques have been proposed that exploit this state structure and reformulate the prediction and correction steps to reduce the computational complexity of the SLAM algorithm [9]. These techniques can be broadly categorised into two categories: those which are optimal and those which are conservative [9]. Optimal algorithms reduce the computational complexity but still produce estimates and covariances equal to the full SLAM algorithm presented in Chapter 1. In contrast, conservative algorithms produce estimates that have larger covariances than the full SLAM algorithm, but are considerably more efficient in terms of computational complexity [9]. In this section only optimal algorithms are considered as conservative algorithms are generally considered invalid solutions to the SLAM problem [9].

In this section a basic description of SLAM based on the extended Kalman filter is presented along with the state-augmentation method for reducing the complexity of the prediction step and the partitioned update method for reducing the complexity of the correction step. This is also complemented by a brief description of the unscented Kalman filter and sparse extended information filter; and also a discussion of the increasingly popular submapping methods, which

are based on the concept that a map can be broken into regions with local coordinate systems that are arranged in a hierarchical manner. Finally, the section is concluded with a brief presentation of the Rao-Blackwellised particle filtering approach to SLAM. A summary of the strengths of weaknesses of each of these approaches is presented in Table 2.1.2.

### 2.1.1 Extended Kalman Filter (EKF-SLAM)

An overwhelming number of solutions to the SLAM problem have been implemented within the framework of the EKF or one of its variants [7]. A brief description of the theory behind the EKF is presented below using the same variables and syntax as Chapter 1.

The basis for the EKF-SLAM method is to describe the vehicle motion in the form:

$$P(x_k|x_{k-1}, u_k) \iff x_k = f(x_{k-1}, u_k) + w_k \quad (2.1)$$

where  $f()$  is a function which models the vehicle kinematics and  $w_k$  are the additive, zero mean uncorrelated Gaussian motion disturbances with covariance  $Q_k$  which affect the vehicle motion. The observation model is described in the form:

$$P(z_k|x_k, m_k) \iff z_k = h(x_k, m_k) + o_k \quad (2.2)$$

where  $h()$  is a function which models the geometry of the observation and  $o_k$  are the additive, zero mean uncorrelated Gaussian observation errors with covariance  $R_k$  which affect the observations. With these definitions, the standard EKF method can be applied to compute the mean:

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{m}_k \end{bmatrix} = E \left[ \begin{array}{c|c} x_k & Z_{0:k} \\ m_k & \end{array} \right] \quad (2.3)$$

and the covariance:

$$P_{k|k} = \left[ \begin{array}{c|c} \left( x_k - \hat{x}_k \right) & \left( x_k - \hat{x}_k \right)^T \\ m_k - \hat{m}_k & |Z_{0:k} \end{array} \right] \quad (2.4)$$

of the joint posterior distribution  $P(x_k, m_k|Z_{0:k}, U_{0:k}, x_0)$  via an iterative prediction and correction algorithm. The time update (Prediction) step for the mean and covariance is:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k) \quad (2.5)$$

$$P_{k|k-1} = \nabla f P_{k-1|k-1} \nabla f^T + Q_k \quad (2.6)$$

where the notation  $a_{i|j}$  is used to represent the estimate  $a$  based on the prediction update at time  $i$  and observation update at time  $j$ . Therefore,  $\hat{x}_{k|k}$  represents the estimate at time  $k$  after both the prediction and observation updates, and  $\hat{x}_{k|k-1}$  represents the estimate at time  $k$  after the prediction update but before the observation update. In Equation 2.6,  $\nabla f$  is the Jacobian of  $f$  evaluated at the estimate  $\hat{x}_{k-1|k-1}$ . A Jacobian matrix contains all the first order partial derivatives of a vector-valued function, and if the function is differentiable, represents the best linear approximation to the function at the specified point. Therefore, the covariance is updated using a linear approximation of the non-linear model around the estimated operating point, which is the fundamental difference between the extended and linear versions of the Kalman filter. Following this prediction step an observation update (Correction) step is performed:

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{m}_k \end{bmatrix} = [\hat{x}_{k|k-1} \hat{m}_{k-1}] + W_k [z_k - h(\hat{x}_{k|k-1}, \hat{m}_{k-1})] \quad (2.7)$$

$$P_{k|k} = P_{k|k-1} - W_k S_k W_k^T \quad (2.8)$$

where:

$$S_k = \nabla h P_{k|k-1} \nabla h^T + R_k \quad (2.9)$$

$$W_k = P_{k|k-1} \nabla h^T S_k^{-1} \quad (2.10)$$

and where  $\nabla h$  is the Jacobian of  $h$  evaluated at  $\hat{x}_{k|k-1}$  and  $\hat{m}_{k-1}$ . This EKF-SLAM solution inherits many of the same benefits and problems as EKF solutions to standard navigation and tracking problems, including convergence, computational complexity, data association and non-linearities [9]. Under ideal conditions for EKF-SLAM the covariance of the estimates of the vehicle location and pose, and individual landmark positions converge towards zero [9]. However, the computational complexity of the correction step of the algorithm grows quadratically with the number of landmarks, which is almost certainly a problem for practical applications [9]. Moreover, the EKF-SLAM solution is extremely sensitive to incorrect association of landmarks to observations, yet the data association problem is extremely difficult for SLAM as landmarks are re-observed from very different viewpoints [9]. Finally, the inherent linearisation of the nonlinear motion and/or observation models can potentially result in inconsistent solutions [9].

Despite these problems the majority of successful SLAM implementations have employed an EKF [7]. For example, Davison et. al. [10] used an EKF framework in their SLAM using active vision scheme, which was deployed on a vehicle mounted with an active stereo camera [10]. This particular implementation was specifically designed to operate in small, topological, feature-rich

environments, and was capable of traversing a 24m trajectory and localising with centimetre accuracy [10]. Similarly, Castellanos et. al. [11] utilised an EKF-SLAM approach in their system which fused data from a 2D laser rangefinder and trinocular camera system to produce a symmetries and perturbations model of the environment. This dual sensing system proved to be very reliable and by carefully combining the uncertainties of the two sensors they were able to estimate the localisation with 96.2% compatibility with the ground truth [11]. Lacroix et. al. [8] also chose an EKF-SLAM based solution for their unmanned aerial vehicle which was equipped with a stereo camera and an inertial measurement unit (IMU). Comparing their results to ground truth provided by GPS the absolute angular errors were less than  $0.89^\circ$  and the absolute translation errors less than 0.35m for a 70m trajectory with altitude changes between 25 and 30m [8]. For a much larger 270m loop the absolute angular errors were less than  $3.56^\circ$  and the absolute translation errors less than 2.12m [8]. This is just a small selection from the enormous quantity of publications on EKF-SLAM that serves to illustrate the accuracy that can be attained using this approach.

### Linear-time state augmentation

At any time  $k$  the joint state vector of the EKF-SLAM approach contains both the robot location and pose and the set of landmark locations [12]. In a naive implementation the computation of the covariance in the prediction step (Equation 2.6) has cubic complexity in the number of landmarks due to the matrix multiplication of the Jacobian  $\nabla f$  with the whole state [12]. However, in reality only the vehicle position and pose is affected by the motion model so the covariance prediction can be re-written in a form which has linear complexity in the number of landmarks [12]. Furthermore, the addition of new landmarks to the joint state vector can be expressed as a case of state augmentation where new landmarks are initialised as a function of the current robot position and an observation [12]. The prediction step can also be expressed as a special case of state augmentation in which the new pose is augmented and the old pose removed by marginalisation [12]. In this form both the prediction step and addition of new landmarks have a complexity that is linear in the number of landmarks [12].

### Partitioned Updates

Similarly to linear-time state augmentation, partitioned updates takes advantage of the structure of the EKF state vector to reduce the computational complexity, but in this case for the observation update step [12]. In a naive implementation the observation update step corrects all vehicle and landmark states every time an observation is made, which results in quadratic complexity in the number of landmarks [12]. Partitioned updates reduce this computational effort by performing sensor-rate updates only to the landmarks in a small local region, and per-

forming full global map updates as a background task at a much lower frequency [12]. There are two basic types of partitioned update, both of which produce optimal estimates [12]. The first is that used by compressed EKF (CEKF) and the postponement algorithm, which operate in a local region of the global map and maintain globally referenced coordinates [12]. The second is that used by the constrained local submap filter (CLSF) and local map sequencing algorithm, which generate a temporary submap which has its own coordinate frame [12]. This second approach has the advantage that it avoids large global covariances, which reduces the linearisation error introduced by the EKF and hence increases numerical stability [12].

### Submapping Methods

Submapping methods are very similar to partitioned updates and also come in globally and locally referenced varieties [12]. Furthermore, they also define a local coordinate frame and obtain estimates within this frame using the optimal SLAM algorithm on the locally referenced landmarks [12]. However, in the case of submapping methods the resulting submap structures are arranged in a hierarchy which further improves computational efficiency but means the global estimates are no longer optimal [12]. Despite this drawback submapping methods are still widely used as they make real-time SLAM in large environments feasible [12].

A global submap approach is adopted by the relative landmark representation (RLR), hierarchical SLAM and constant time SLAM (CTS) methods [12]. These approaches reduce computational complexity to linear or constant time by maintaining a conservative estimate of the global map [12]. However, the use of a global coordinate frame does not alleviate problems associated with linearisation errors when the vehicle pose uncertainty is large [12].

The relative submap approach was first proposed by Chong et. al. [13], but was further developed in the form of the constrained relative submap filter (CRSF) and network coupled feature maps (NCFM) [12]. Relative submap methods differ from global submaps in that there is no common coordinate frame and the location of any given submap is defined by conservative links to its neighboring submaps [12]. In this way global estimates must be obtained by vector summation along the complete path of submaps [12]. This approach has the advantage that not only is computational complexity reduced but uncertainty within submaps is also reduced which alleviates nonlinearity issues [12]. In addition, by performing updates locally the algorithm attains higher numerical stability and enables batch association between submap frames [12].

### Sparse Extended Information Filtering (SEIF)

The conventional EKF-SLAM approach produces a state estimate  $\hat{x}_k$  and a covariance matrix  $P_k$  which implicitly describe the first two central moments of a Gaussian probability density on the true state  $x_k$  [9]. It is also possible to represent this Gaussian probability density in a

canonical or information form using an information vector  $\hat{y}_k$  and an information matrix  $Y_k$ . These two representations are related by the following transforms:

$$Y_k = P_k^{-1} \quad (2.11)$$

$$\hat{y}_k = Y_k \hat{x}_k \quad (2.12)$$

The advantage of the information form over the moment form is that for large-scale maps the majority of the off-diagonal components of the normalised information matrix are very close to zero [9]. Thrun et al. [14] exploited this fact and developed a sparsification procedure in which the elements of the normalised information matrix that were close to zero were actually set to zero [14]. This forces the information matrix to be sparse, which saves a significant amount of memory and enables the application of very efficient sparse update procedures for information estimates [14]. While this method results in relatively little loss in optimality, it was shown to be inconsistent [12]. However, the concept of information-form SLAM has recently received much attention and optimal and exactly sparse solutions have been proposed by Dellaert [15], Eustice et. al. [3] and Folkesson et. al. [16]. Exact sparsification of the information matrix is achieved by augmenting the state with the new vehicle pose estimate during each update and not removing any of the past vehicle poses [12]. As a result the off-diagonal terms of the matrix are non-zero only for poses and landmarks which are directly related by observations [12]. However, it is not possible to keep the entire pose history and marginalisation is necessary to remove some of the past vehicle poses [12]. Unfortunately, marginalisation introduces links between all state elements that were connected to the removed states, which results in a dense information matrix [12]. Nevertheless, by carefully selecting a small set of vehicle poses that are retained it is possible to decouple different regions of the map and obtain an information matrix that is a reasonably sparse estimate [12]. While the information form appears far more computationally efficient than the moment form the margin is not as large as expected as the mean estimate must be recovered at each update step to perform linearisation of the motion and observation models, and information from the covariance matrix is often required for data association [9]. While the mean estimate can be recovered fairly efficiently using the conjugate gradients method proposed by Eustice et. al. [3] and efficient solutions have also been proposed for simple gating based data association, robust data association generally requires the full covariance matrix, which is recovered with cubic complexity in the number of landmarks [9].

### Unscented Kalman Filter (UKF)

Julier and Uhlmann [17] noted that when the motion and observation models of the EKF are highly non-linear the EKF performs particularly poorly because the mean is propagated

through the non-linearity. To resolve this issue Julier and Uhlmann [17] proposed the unscented Kalman filter (UKF), which uses a deterministic sampling technique known as the unscented transform to select a minimal set of sample points distributed around the mean [17]. All of these sample points are then propagated through the non-linear models and an estimate of the mean and covariance is recovered from the resulting samples [17]. Using Monte Carlo sampling or Taylor series expansion of the posterior statistics, it is possible to verify that this mean and covariance more accurately represent the true mean and covariance than the estimates produced by the EKF [17]. While the propagation of the samples through the non-linear models can be computationally expensive, this approach eliminates the need to compute the model Jacobians [17]. Consequently, in many cases the UKF is able to provide more accurate estimates with little or no increase in computational load.

### 2.1.2 Rao-Blackwellised Particle Filter (FastSLAM)

Particle filters are also known as sequential Monte-Carlo (SMC) methods and are used to estimate Bayesian models with a recursive sampling-based approach [7]. Particle filters, like all sampling-based approaches maintain a set of random point clusters (particles) that approximate the Bayesian posterior [7]. Because this representation is approximate and non-parametric the distributions it can represent are not limited to Gaussians as is the case with parametric filters such as the EKF [7]. Furthermore, this representation is much better suited to modelling nonlinear transformations [7]. This makes particle filters particularly well suited to SLAM problems in which the motion and observation models are highly nonlinear [7]. However, similarly to EKF-based approaches, sampling-based approaches suffer from a rapid growth in computational complexity with increasing state size [7]. Consequently, for real-time SLAM applications in which the state is composed of both the vehicle pose and hundreds of landmarks it is computationally infeasible to directly apply particle filters [7]. However, it is possible to reduce the dimensionality of the state-space by applying Rao-Blackwellisation, which partitions the joint state according to the product rule:

$$P(x_1, x_2) = P(x_2|x_1)P(x_1) \quad (2.13)$$

If  $P(x_2|x_1)$  can be represented analytically only  $P(x_1)$  needs to be sampled. Therefore, the joint distribution is represented by the set  $\{x_1^i, P(x_2|x_1^i)\}_i^N$  where  $x_1^i$  is a sample from  $P(x_1)$ . The marginal  $P(x_2)$  can be obtained statistically from:

$$P(x_2) \approx \frac{1}{N} \sum_i^N P(x_2|x_1^i) \quad (2.14)$$

It should be noted that by partitioning the joint state the marginal is actually obtained with a higher accuracy than when sampling is performed over the whole space. This approach was used by Montemerlo et. al. [18] and is the basis of the FastSLAM algorithm [18], which was the first SLAM implementation to directly estimate the nonlinear motion model and non-Gaussian pose distribution by recursive particle filtering (EKF is still used for the observation model) [18]. In FastSLAM the joint state is factored into a vehicle pose component and a conditional map component:

$$P(X_{0:k}, m|Z_{0:k}, U_{0:k}, x_0) = P(m|X_{0:k}, Z_{0:k})P(X_{0:k}|Z_{0:k}, U_{0:k}, x_0) \quad (2.15)$$

Therefore, the probability distribution is conditioned on the trajectory  $X_{0:k}$  rather than the current pose  $x_k$  and each particle defines a different vehicle trajectory hypothesis [18]. This is the most important innovation of FastSLAM, as it enables the map landmarks to be represented as a set of independent Gaussians which can be processed with linear rather than quadratic complexity [18]. In this form FastSLAM is essentially a Rao-Blackwellised state, where the trajectory is represented by weighted samples and the map is computed analytically [18]. Accordingly, the joint distribution at time k is represented by a set of particle weights, trajectory hypotheses and associated map hypotheses,  $\{w_k^i, X_{0:k}^i, P(m|X_{0:k}^i, Z_{0:k})\}_i^N$  where the maps are composed of a set of independent Gaussian distributions:

$$P(m|X_{0:k}^i, Z_{0:k}) = \prod_j^M P(m_j|X_{0:k}^i, Z_{0:k}) \quad (2.16)$$

Map updates are simple as each observed landmark is processed individually as an EKF observation update from a known pose (specific trajectory particle) and unobserved landmarks are unchanged [18]. Propagation of the pose particles is based on a recursive form of sequential important sampling (SIS) [18]. For each particle a proposal distribution conditioned on the particles history ( $P(x_k|X_{0:k-1}^i, Z_{0:T})$ ) is computed and a sample is drawn from this distribution [18]. These samples are then weighted according to an importance function which is a function of both the motion model and an observation model for which the dependency on the map is marginalised away [18]. However, the approximation error of the proposal distribution grows with both time and increasing variation in the particle weights so a re-sampling step is periodically performed to reinstate uniform weighting [18]. This re-sampling step results in a loss of information regarding particle histories and for this reason it is only possible to apply this approach to systems that "exponentially forget" [18]. Systems which fall into this category are those for which the system process noise causes the state at time k to become increasingly independent of preceding states [18].

There exists two versions of FastSLAM, FastSLAM 1.0 [18] and FastSLAM 2.0, which differ

only in the form of their proposal distribution and importance weighting [1]. In FastSLAM 1.0 the proposal distribution is the motion model and the samples are weighted according to the marginalised observation model [18]. FastSLAM 2.0 is much more efficient as the proposal distribution also incorporates the current observation, which means it is locally optimal in terms of the variance in the assigned importance weights [19]. One major weakness of the FastSLAM approach is that marginalising the observation model introduces a dependence on the pose and observation history, which is a problem as re-sampling erases this history resulting in a loss of accuracy [1]. Despite this seemingly crippling weakness, Montemerlo et. al. [19] experimentally demonstrated that FastSLAM 2.0 provides excellent results with as little as one particle [19]. Operating on the popular benchmark data set collected with an outdoor vehicle in Victoria Park, Sydney, Montemerlo et. al. [19] achieved an accuracy equivalent to EKF-SLAM implementations which have  $O(N^2)$  complexity using only 50 particles and an execution time less than 4% of the vehicle trajectory time [19]. Porta et. al. [20] developed a particle filter based visual SLAM algorithm which also included a novel appearance-based feature representation and map compression method. This implementation also provided excellent results with absolute translation errors less than 25cm and absolute angular errors less than  $5^\circ$  [20].

## 2.2 Feature Recognition and Tracking

Vision-based systems can be used to estimate both the 3D location of environment landmarks (or features) and the pose of the vehicle [7]. Vision-based systems include stereo camera pairs which perform triangulation and also monocular cameras which utilise structure from motion recovery [7]. This section concentrates on approaches to solving the feature recognition and tracking problem, which involves estimating the locations of features in an image sequence and matching these to previously observed features [7]. A summary of the strengths and weaknesses of the methods presented in this Section is provided in Table 2.2.6.

### 2.2.1 Colour Histograms and KLT Tracker

Mahon et. al. [21] developed a visual SLAM algorithm for an AUV which operates in an unstructured, underwater environment. The AUV fused information from both sonar and vision sensors to identify and track a sparse set of distinguishable environment landmarks [21]. They identified that in underwater environments where water movement may quickly cause large uncertainty in the vehicle location it is necessary to have a large number of landmarks to ensure multiple landmarks are always visible [21]. They proposed the use of a separate class of features for motion estimation and landmark description, with a small number of highly robust features used for the later [21]. This choice is largely due to the fact that the robust features used for the

Table 2.1: Strengths and weaknesses of SLAM solutions

Filter	Strengths	Weaknesses
Extended Kalman Filter	Linear time observation updates using partitioned updates  Linear time prediction updates using state augmentation  Constant time solution achievable using submapping methods	Assumes Gaussian error which is often untrue  Linearisation of non-linear models can cause divergence
Information Filter	Computationally efficient due to sparseness of information matrix	Cubic complexity to recover covariance matrix if required for data association
Unscented Kalman Filter	More robust and accurate estimation of system state and covariances than EKF  Does not require computation of model Jacobians	Complexity dependent on number samples
Rao-Blackwellised Particle Filter	Simultaneously maintains multiple hypotheses  With large number of samples can efficiently represent non-Gaussian and/or non-linear models	Number particles increases exponentially with state size

landmark description cannot be obtained in the quantity required for motion estimation [21]. A large number of small rectangular patches were tracked using the Lucas-Kanade algorithm to provide estimates of the vehicle motion between frames [21]. However, these transient features are not suitable for describing landmarks as they are generally only robust to the small changes in viewpoint expected between consecutive video frames [21]. To generate landmark descriptions Mahon et. al. [21] segmented the image using region growing and constructed a histogram of each region's pixels normalised red and green colour components [21]. Data association was then achieved by computing a distance metric based on the Bhattacharyya coefficient [21]. To avoid confusion each landmark must be unique, and consequently Mahon et. al. [21] proposed using the average Shannon information content of the pixels' hue to identify the best landmarks. However, this has an inherent bias for selecting small landmark features, therefore only landmarks with large differences according to the distance metric used for data association were retained [21]. Mahon et. al. [21] demonstrated that these landmarks could be successfully associated over small loops. However, they also noted that these landmarks were susceptible to fluctuations in colour due to changes in the lighting and depth of the AUV [21]. In addition,

while new landmarks are inherently distinguishable from all previously stored landmarks, due to the nature of the environment and simplicity of the descriptors it is quite probable that landmarks in other parts of the environment may yield similar descriptors [21]. Moreover, due to the nature of the environment it is extremely difficult to produce a robust segmentation algorithm capable of operating in real-time, which is a prerequisite of this approach [21].

### 2.2.2 Precise Harris Corner Detector

Lacroix et. al. [8] developed a visual SLAM algorithm for an unmanned aerial vehicle (UAV) which operates in both urban and rural environments. The UAV was equipped with a stereo camera system which was used for both landmark detection and motion estimation [8]. Similarly to AUVs, UAVs also require a large number of features to always be visible in order to perform accurate high-speed visual odometry as wind can quickly introduce large uncertainty in the vehicle location [8]. Lacroix et. al. [8] proposed to use the precise version of the Harris detector developed by Mikolajczyk et. al. [22] to identify a large number of interest points. Small image patches are then extracted around each of these interest points and these features are filtered based on the resemblance of their principal curves to eliminate similar features. The 3D position of the remaining features is then computed using stereo matching and the features are clustered into groups which describe individual landmarks [8]. Data association is performed by predicting where each feature in a group is likely to be in a new frame (if visible) based on the estimated motion and then searching in a small window around each of these points in order to find a set of matches [8]. After performing extensive practical trials they showed that not only was matching highly accurate between consecutive frames, but the data association was also robust to large changes in viewpoint, scale and occlusions [8]. Despite this approaches inherent reliance on the accuracy of the motion estimation Lacroix et. al. [8] demonstrated that they were able to successfully close a 270m loop containing 350 landmarks.

### 2.2.3 Scale Invariant Feature Transform (SIFT)

Se et. al. [23] developed a visual SLAM algorithm for a ground-based autonomous mobile vehicle which was equipped with a trinocular camera system. Se et. al. [23] proposed the use of the scale invariant feature transform (SIFT) for both motion estimation and landmark description. SIFT features are invariant to image translation, scaling and rotation, and are not sensitive to illumination changes and affine/perspective projection [23]. Moreover, the local and multi-scale nature of SIFT features makes them insensitive to noise, clutter, and occlusion, yet the detailed local image properties enable highly selective matching [23]. These characteristics make them ideal features for robust SLAM, as landmarks will usually be re-observed from different viewpoints and under different lighting conditions [23]. Se et. al. [23] support this

claim by referring to a recent performance evaluation of various local descriptors, in which SIFT features performed the best and the widely used Harris corner detector performed poorly due to its sensitivity to changes in scale [23]. By using a trinocular system, Se et. al. [23] were able to compute the 3D world coordinates of each feature relative to the vehicle by matching features in all three frames based on the epipolar and disparity constraint and also the SIFT scale and orientation constraints [23]. To reduce computation time the system did not use the conventional SIFT feature vector which is computed from 8 orientations, each sampled over a 4x4 grid of locations to produce a 128 element SIFT key [23]. Instead, a smaller SIFT key was produced from 4 orientations, each sampled over a 2x2 grid of locations to obtain just 16 elements [23]. Feature matching was performed by comparing the scale and orientation of a pair of SIFT features and if required computing the Euclidean distance between their keys to check if this was below a fixed matching threshold [23]. Se et. al. [23] demonstrated that a set of 16 element SIFT keys was sufficiently distinctive for small environments and suggested that simply increasing the size of the SIFT key would enable applications in much larger outdoor environments. Essentially, this was confirmed by Schleicher et. al. [24] who developed a real-time visual SLAM algorithm based on 128 element SIFT keys which was able to close loops larger than 270m when tested on a mobile robot mounted with a wide-angle stereo camera [24].

#### 2.2.4 Image Patches

Blesser et. al. [25] developed a visual SLAM algorithm for a single self-tracking camera operating in a stable natural environment. Blesser et. al. utilised the robust and drift-free method for registration of 2D features developed by Zinßer et al. [26] who combined the well-known Kanade Lucas Tracker (KLT) with inverse compositional image alignment and compensation for affine motion and linear illumination changes. The system estimates the motion between frames and uses this estimate to predict which features should be visible and attempts to match these features [25]. Once the camera has moved and a sufficient baseline is available triangulation is used to compute the 3D position of the features and matching is refined by adding epipolar constraints [25]. Blesser et. al. [25] noted that using the 3D uncertainty to identify the best features is not suitable as this uncertainty depends on the metric and size of the target scene and the distance between the feature and the camera. Furthermore, this uncertainty does not reflect the behaviour of the feature on the 2D level [25]. Instead, Blesser et. al. [25] proposed to maintain a low-pass filtering of the normalised re-projection error for each feature between frames as a feature quality indicator. In this way features which are stable and repeatedly matched can be separated from unstable features which are frequently mismatched or poorly localised in 3D, making the system considerably more robust [25]. Blesser et. al. [25] demonstrated that this approach was able to successfully localise the camera in small, feature-

rich environments. However, the results suggest the closure of large loops may not be possible as the feature matching has a strong dependence on the motion estimation, which is susceptible to drift [25]. Overall, this approach is remarkably similar to that of Montiel et. al. [27] except that Montiel et. al. use simple image patches, represent feature locations by direction and inverse depth and initialise features as semi-infinite lines [27].

### 2.2.5 Comparison of Interest Point Detectors

Obviously it is difficult to compare the strengths and weaknesses of these different approaches when each research group devised a different set of tests and performance measures. Thankfully, Mozos et. al. [28] recently analysed the suitability of several interest point detectors as landmark extractors for vision-based SLAM. The detectors were evaluated according to their repeatability and robustness under changes in viewpoint and scale (not illumination) for both planar objects and 3D scenes as this is the usual requirement in visual SLAM [28]. The evaluation metric was the "survival ratio",  $S_i$  which indicates which percentage of features detected in the first frame were successfully tracked until the  $i^{th}$  frame (interest points lost due to occlusions were omitted) [28]. In this way it is possible to estimate both the stability of the interest points and also how long the particular type of interest points should be tracked before integrating them into the map in order to eliminate spurious points [28]. The following interest point detectors were evaluated according to this criterion: Harris corner detector, Harris-Laplace detector, Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF) and Smallest Univalued Segment Assimilating Nucleus (SUSAN). The results showed that for both changes in scale and viewpoint the Harris corner detector outperformed all the other interest point detectors [28]. SIFT and SURF both performed slightly worse and achieved similar survival ratios for changes in viewpoint, although SURF was much more stable than SIFT under changes in scale [28]. SUSAN achieved the lowest score in all tests and appears to be unsuitable for visual SLAM [28].

### 2.2.6 Non-geometric Landmarks

Perhaps surprisingly, SLAM implementations typically utilise geometric landmarks (often mis-named point landmarks) [12]. However, it is possible to handle landmarks of arbitrary shapes by estimating landmark locations separately from the estimation of the landmarks shape [12]. This is achieved by simply attaching a coordinate frame to the object and placing the origin of this coordinate frame into the SLAM algorithm [12]. An auxiliary process then describes the shape of the landmark relative to this coordinate frame [12]. When the vehicle re-observes the landmark this process attempts to align the shape model with the observation data, and assuming this alignment is approximately Gaussian it should be possible to produce an esti-

mate of the location of the coordinate frame origin, which can then be used as an observation update [12].

Table 2.2: Strengths and weaknesses of interest point detectors

Technique	Strengths	Weaknesses
Colour Histograms	Simple representation enables fast data association  Moderately robust to changes in scale and viewpoint	Colour variation with depth, viewing angle and distance  Image segmentation computationally expensive  Not sufficiently distinctive
KLT	Very accurate tracking for short sequences	Not robust to changes in scale and viewpoint
Harris Corners	Detection of Harris corners very robust to large changes in viewpoint and scale	Matching performance depends on descriptor used
Image Patches	Simple, computationally efficient	Poor robustness to changes in viewpoint and scale without pre-transformation based on estimated motion
SIFT	Invariant to translation, scaling and rotation  Low sensitivity to illumination changes and affine/perspective projection	Computationally expensive feature extraction
SURF	More robust than SIFT (particularly to changes in scale)  Significantly faster feature extraction than SIFT	Typically yields lower number of points than SIFT
SUSAN		Poor robustness to changes in viewpoint and scale

## 2.3 Data Association and Loop Closure Techniques

Data association is essentially a feature correspondence problem, i.e. it relates a set of observed features with features within the map [29]. Data association is the most critical point in obtaining a robust SLAM solution as incorrect data association can cause localisation methods such as the EKF to diverge [29]. One specific example of this problem is known as loop closure and refers to the case when the robot re-observes a landmark after traversing a large loop [7]. Data association during loop closure is particularly difficult as the errors accumulated along

the trajectory mean the constructed map is inconsistent and the true vehicle location is often outside the  $3\sigma$  bound on the estimated vehicle position [5]. Therefore, correctly asserting that the vehicle has returned to a previously visited location is crucial in order to correct the map [5].

At the highest level the data association problem can be solved either by searching in the pose space or searching in the correspondence space [29]. In the pose space only a small set of potential vehicle locations are considered and each is analysed for consistency with the current observation and the previous map [29]. This approach can be used with raw sensor data and is the foundation of Monte Carlo localisation SLAM [29]. In the correspondence space observations are processed to obtain a discrete set of features which are matched against a database of map features, which is the case for visual SLAM [29].

### 2.3.1 Maximum Likelihood and Gated Nearest Neighbour

Consider the case when the problem of finding correspondences is solved using a maximum-likelihood (ML) approach [7]. The algorithm could compare an observation with every map feature, however, the complexity of finding correspondences is exponential on the number of possible pairings [29]. Consequently, this approach is usually not viable and the set of features to compare must be selected based on the predicted vehicle position and orientation [7]. The compatibility of the observation and each feature in this set is then computed and the match with the highest probability is selected [7]. If this probability is above an empirically determined threshold the observation is considered a match to the map feature, otherwise it is considered a new feature [7]. This is essentially equivalent to the widely used gated nearest neighbour (NN) algorithm. In this case, the normalised squared innovation test (Mahalanobis distance) is used to determine compatibility and a threshold or other heuristics is applied to the best match to decide if a new map feature should be created [29]. This approach is very simple conceptually and has been demonstrated to work well when the density of features in the map is low and the precision of the vehicle location estimates and measurement sensor is fairly high [29]. However, Neira et. al. [29] noted that the normalised squared innovation of different observations obtained from the same vehicle position are correlated, and consequently an individual innovation compatibility test easily accepts hypotheses formed by mutually inconsistent matches [29]. Obviously, this is of great concern as incorporating spurious matches into the map leads to rapid divergence in the estimation of the vehicle location [29]. This highlights the weakness of such proactive approaches, which generate all hypotheses at the time a feature is observed (or a fixed number of time steps later) and after this never reconsider the data association decisions made [30].

### 2.3.2 Robust Data Association

A large number of more robust algorithms have been proposed such as multitracking, which obtains hypotheses where temporal coherence is guaranteed but increases complexity as it is necessary to maintain a separate map for each hypothesis [31]. Other approaches exist in which geometric constraints between features are used to obtain hypotheses that are consistent with a large number of compatible pairings, however, pairwise compatibility does not guarantee joint compatibility [31]. Two notable solutions which overcome these problems are the joint compatibility branch and bound algorithm (JCBB) developed by Neira et. al. [31] and the tree-structured Bayesian representation of map posteriors proposed by Hä hnel et. al. [30].

The algorithm developed by Neira et. al. [31] explicitly takes into account the correlation between the innovations of individual pairings and determines their joint compatibility [31]. In this way most spurious matches can be eliminated as the probability that a spurious pairing is jointly compatible with all the pairings of a given hypothesis decreases as the number of pairings in the hypothesis grows [31]. Therefore, the algorithm must search an interpretation tree for the hypothesis which contains the largest number of jointly compatible pairings [31]. Neira et. al. [31] developed an approach for incrementally evaluating the consistency (joint compatibility) of a hypothesis using a joint implicit function. This monotonically nondecreasing criterion was then used to bound the search in the interpretation tree [31]. Furthermore, each node belonging to a hypothesis is assigned a quality value based on the number of nonnull pairings that can be established from that node such that nodes with quality values lower than the best available hypothesis are not explored [31]. This branch and bound algorithm using joint compatibility results in an algorithm which is always more robust than gated NN and can be efficiently computed in real-time if the hypothesis size is limited to 10-12 observations before performing a map update [31]. Using a dataset of a small 60m loop collected by a Labmate mobile robot equipped with a trinocular vision system Neira et. al. [31] demonstrated that the JCBB algorithm was considerably more robust than the incremental ML algorithm.

The algorithm developed by Hä hnel et. al. [30] is based on a similar concept and uses a tree-structured Bayesian representation of map posteriors so that it is possible to revise data association decisions made in the past. Similarly, to Neira et. al. [31] they also proposed a criterion for detecting and repairing poor data association decisions [30]. Every time an observation is made the algorithm reviews the complete set of paths through the data association tree to see whether a different set of data associations may have yielded a map with a higher likelihood [30]. To implement this efficiently alternative assignments are only evaluated if they show promise, the map is condensed into a graphical representation, and equality constraints are employed for alleged data associations [30]. In addition, to make the computation tractable the algorithm is applied to local submaps of approximately five meters in length, and within

these maps it is assumed incremental ML data association is sufficiently robust [30]. Using a dataset of a large loop collected by a vehicle equipped with a laser range finder in an abandoned mine Hä hnel et. al. [30] demonstrated that this "lazy approach" to data association produced more accurate maps than both incremental ML and FastSLAM. A comparison of the relative strengths and weaknesses of these data association algorithms is presented in Table 2.3.2.

Table 2.3: Strengths and weaknesses of data association approaches

Technique	Strengths	Weaknesses
Incremental Maximum Likelihood	Simple, computationally efficient solution  Suitable for low density map and high precision vehicle pose estimates and observations	Never reconsiders matches  Accepts mutually inconsistent matches which leads to inconsistent maps
Multi-Tracking	Most robust data association solution achievable - guarantees temporal coherence	Maintains one map per hypothesis so computationally infeasible
Joint Compatibility Branch and Bound	Guarantees selection of hypothesis containing largest number of jointly consistent pairings  Reconsideration of matchings during fixed interval and before integration into map  Real-time operation achievable if size of hypotheses is limited	
Tree Structured Bayesian Representation of Map	Reconsiders matchings made in the past - selects branch with highest log-likelihood	High computational complexity even when tree depth is limited
FastSLAM (Particle Filtering Based Technique)	Able to generate many hypotheses for each observation and essentially filter over time	Re-sampling can eliminate valid hypothesis  Computational complexity increases with number of particles and hence number of hypotheses

### 2.3.3 Visual Tracking and Loop Closure

Another common approach to data association which is specific to visual SLAM is visual tracking, which can be used to perform highly accurate local data association, but generally fails to solve the loop closing problem [31]. As a result there has been a large volume of research into independent techniques for local data association and loop closure, two of the most notable of

which were developed by Se et. al. [23] and Ho et. al. [5].

Se et al. [23] developed a vision-based loop-closing approach that constructs multiple overlapping 3D submaps which are subsequently merged together. Local registration within submaps is achieved by tracking distinctive SIFT features, and global correlation is achieved by applying the Hough transform and RANSAC to align submaps [23]. When the algorithm detects a significant overlap between landmarks in the current submap and a previously stored submap a global minimisation strategy is applied to perform backward correction on all the submaps in the loop [23]. This enables the algorithm to minimise the error that accumulated along the trajectory and hence obtain a considerably more accurate global map [23]. The proposed method of combining the power of distinctive and reliable features such as SIFT and robust algorithms such as RANSAC appears to be very effective, but experiments with complicated loop closure scenarios have not yet been tested [23].

Ho. et. al. [5] suggests one of the primary reasons many SLAM loop closure algorithms fail is because they rely on internal estimates of the vehicle position, which at the end of a large loop can be in gross error. This lead Ho et. al. to propose a loop closing algorithm which makes no recourse to the internal estimates of the SLAM system [5]. Instead, at regular intervals the algorithm takes a snapshot of the local scene (described by a set of SIFT descriptors from Harris-Affine regions) and encodes the similarity between all possible scene pairings in a similarity matrix [5]. In this way the loop closure problem is solved by simply detecting statistically similar scene sequences within this matrix [5]. This approach implicitly exploits the temporal and spatial proximity of the scene acquisition locations and enables evidence to be integrated over the vehicle trajectory, significantly reducing the occurrence of false positives [5]. Ho et. al. [5] devised an efficient method for analysing and decomposing this similarity matrix in such a way that loops could be reliably detected even in repetitive and visually ambiguous scenes [5]. To demonstrate the robustness of this approach extensive experimentation was carried out over a range of realistic and challenging outdoor settings using visual images [5]. The algorithm is designed to strongly prefer false negatives over false positives and consequently loop closure was never incorrectly asserted [5]. However, for some loops with small overlaps closure was not detected as the accumulated probability did not grow sufficiently fast, and in other cases with repetitive architecture the detection was suppressed due to the high likelihood of false loop closure [5].

## 2.4 Vision-based Motion Estimation

The estimation of camera motion between consecutive frames (egomotion) is an old problem in computer vision [2]. In the past homography-based methods have been widely used for

the estimation of mobile vehicle motion [32]. For example, Fleischer [33] developed a real-time motion estimation technique based on a visual map which consists of a collection of key images that represent places the vehicle has visited. Upon revisiting any of these places an image registration is performed between the current image and the stored image to obtain 2D translational constraints between the camera poses [33]. Negahdaripour et al. [34] developed a mosaic-based motion estimation technique based on the optical flow and the generalised brightness consistency constraint. This algorithm estimates inter-frame motion directly from the optical flow and integrates this over time to estimate the vehicle position. To constrain the error growth to the accuracy of the mosaic, the algorithm periodically generates an expected image based on the position estimate and constructed mosaic, from which a homography is obtained and used to update both the estimated position and mosaic [34]. Garcia, et al. [35] and Lots, et al. [36] also utilised homographies to estimate the motion of underwater vehicles for mosaicking and station keeping. All of these solutions require robust techniques for estimating the homography between two images, and methods to recover metric properties of the motion and scene from these homographies, which are discussed in detail by Hartley and Zisserman [37]. While these techniques have been applied with success over small, relatively flat areas, homography-based motion estimation fails when the seafloor deviates from the planar assumption [32]. This makes the techniques inapplicable for environments with significant 3D structure, such as surveys over coral reefs or archaeological sites.

A large number of approaches based on both the discrete and the differential epipolar constraint have been proposed [2]. The discrete constraint is typically used in self-calibrated stereoscopic systems in which motion is recovered from a set of 3D point correspondences [2]. The differential constraint is usually applied when using a single camera for the motion estimation and is based on optical flow [2]. Table 2.1 classifies a number of well known motion recovery methods according to this division and also according to whether they utilise linear or non-linear techniques [2]

Armangue et. al. [2] presents a complete survey of techniques for the differential case including those which explicitly use the differential epipolar constraint and also those which are directly based on the optical flow [2]. The survey compared the relative performance of the 6DoF motion recovery methods on synthetic data with Gaussian noise and outliers, and it was observed that all the 6DoF motion recovery methods were quite sensitive to noise [2]. The surveyed methods which performed the most poorly were the modified iteratively re-weighted least squares (MIRLS) and the least median squares (LMedS) algorithms due to their lack of convergence [2]. The other surveyed methods all performed comparably with respect to the estimation of translation, where an angular error of approximately  $20^\circ$  is present due to the addition of Gaussian noise with a standard deviation of just 0.3 pixels [2]. However, under more favourable conditions in which the camera field of view is decreased or the linear/angular velocity

Discrete case	Differential case
<i>Linear techniques</i>	
Longuet-Higgins [8]	Zhuang, Huang, Ahuja, Haralick [21,22] <sup>a</sup>
Tsai and Huang [23]	Heeger and Jepson [24–26]
Toscani and Faugeras [27]	Kanatani [28,29] <sup>a</sup>
Tomasi and Kanade [30]	Tomasi and Shi [31,32]
	Brooks, Chojnacki, Hengel and Baumela [33] <sup>a</sup>
	Seven points
	Least squares
	Iteratively reweighted least squares
	Modified iteratively reweighted least squares
	Least median squares
	Ma, Košecká and Sastry [18,34] <sup>a</sup>
	Baumela, Agapito, Bustos and Reid [35] <sup>a</sup>
<i>Nonlinear techniques</i>	
Horn [36]	Prazdny [37,38]
Weng, Ahuja and Huang [39]	Bruss and Horn [40]
Taylor and Kriegman [41]	Zhang and Tomasi [42]
Scatto and Brockett [43]	
Ma, Košecká and Sastry [44]	

Figure 2.1: Motion Recovery Methods [2]

coefficient is increased the algorithms performed considerably better [2]. The method proposed by Kanatani [38] which performs a pre-processing step to normalise the data was observed to obtain by far the best results in angular velocity estimation and provided an estimation in less than 0.1s on an 800 MHz Pentium III computer [2]. The LMedS solution proposed by the authors is also of interest as it optimises the solution once the outliers have been removed and provided an estimation in less than 0.5s [2].

As a prologue to their proposed approach to egomotion estimation Dell’Acqua et. al. [39] presented a brief survey of the state-of-the-art in techniques that are based on the geometry of image features. Dell’Acqua et. al. [39] stressed the fact that while the theory behind structure and motion estimation methods is well-established, egomotion estimation is still extremely challenging in practical applications [39]. One of the main reasons for this is the reliance on high quality (accurate, long lifespan and stable) feature localisation and tracking, which is rather difficult as features are observed from different viewpoints and frequently occluded [39]. In addition, to solve for the large number of unknowns it is generally necessary to use as many features and constraints as possible in order to avoid ill-conditioning [39]. Dell’Acqua surveyed a large number of solutions to the egomotion estimation problem and divided them into two broad categories: those that exploit multi-view geometric constraints between pairs or triplets of views from the sequence (batch techniques), and those that estimate the iterative, online motion of a camera [39]. Only the later category of solutions are of interest here as these causal and recursive algorithms can be directly applied to real-time camera tracking applications as they only require information from the past [39]. These approaches generally refine camera motion and 3D structure estimates by employing a Kalman filter and performing

an update each time a new frame arrives, and hence have time complexity linear in the number of images [39] [32]. These techniques assume a set of sparse two dimensional point features are tracked through the image sequence, which in many cases is achieved using the Lucas-Kanade tracker as it has shown good performance in comparative studies [32]. For good examples of this approach see Azarbayejani and Pentland [40], [41], Chiuso et. al. [42], [43] and [44] and Kim and Chung [45]. Note that this approach only recovers motion and structure estimates up to a scale, and for autonomous navigation this ambiguity must be removed by augmenting this method with additional information from other sensors [32]. The performance of this approach was compared to the homography-based approach by Adams et. al. [32]. The results revealed that structure and motion methods were more than twice as computationally expensive as the homography-based approaches that are combined with RANSAC [32]. Both techniques performed comparably for planar terrain, and while all the homography-based methods failed in truly 3D scenes in many cases when the terrain was mostly planar but with sparse non-trivial extrusions the homography-based approach that uses RANSAC still achieved an accuracy approximately equivalent to the structure and motion method [32].

## 2.5 Real-time 6-DOF vision-based SLAM Implementations

The number of researchers currently developing real-time SLAM implementations is overwhelming, so in this section the scope of the material reviewed has been reduced to encompass only those implementations that operate with 6 degrees of freedom (DOF). Of particular interest are those implementations that combine vision-based sensors and inertial measurement units (IMU) as both of these sensors are independent of the platforms kinematics, which means the system can easily be ported to autonomous aerial, underwater and all-terrain vehicles with very little or no modifications [46].

Inertial measurement units (IMU) sense the acceleration and rotation rates of a platform at high frequencies [46]. Inertial navigation systems (INS) integrate and process this information in order to estimate the position, velocity and attitude of the platform [46]. However, like all dead-reckoning systems the estimates produced by the INS drift with time, and for an INS the drift rate is typically a cubic function of time [46]. This is due to the fact that even a small error in the gyros will be accumulated in angle estimates (roll and pitch), which in turn misrepresent gravitational acceleration as vehicle acceleration, thus resulting in quadratic velocity (and cubic position) errors [46]. As a result absolute sensory information is required to constrain this drift and to obtain what is known as an aided inertial navigation system (AINS) [46]. This review only considers on AINS in which the absolute sensory information is provided by vision-based observation of natural landmarks.

SLAM is a computationally expensive process and 6DoF implementations have an additional burden as the state size is increased to incorporate the full description of the vehicle pose ( $x,y,z$  position and roll, pitch and yaw angles) [47]. Computational complexity is increased further when using inertial sensors as they require high sampling rates [47]. As a result, 6DoF implementations often employ novel techniques to reduce computational complexity and these have been highlighted in the description of the following implementations.

### Underwater SLAM

Very few papers have been published regarding visual SLAM for underwater environments and to the best of our knowledge none exist that are capable of operating in real-time. However, to indicate the current state-of-the-art three algorithms which operate offline have been presented.

Saez et. al. [48] developed a 6DoF SLAM algorithm based on Entropy Minimisation, which is able to create dense 3D visual maps in underwater environments. The algorithm utilises a stereo system to obtain dense 3D information, and performs robust egomotion estimation and global-rectification using an optimisation approach [48]. Saez et. al. noted many challenges that are unique or far more extreme in the subsea environment and must be considered including:

- Unpredictable currents and surge acting upon the vehicle body to produce motion in any direction.
- Highly dynamic lighting conditions which add to the complexity of motion tracking algorithms.
- Decreasing visibility with depth and turbidity of the water.
- Marine life produces spurious sensor measurements.
- Aquatic snow complicates even static measurements.
- The environment is visually unstructured - even man-made objects degrade or become covered in marine growth, which causes almost certain failure of algorithms which assume corners or planarity.

The approach of Saez et. al. [48] was to utilise exclusively 2D image features and 3D data from the stereo system to compute the inter-frame motion (egomotion). A global rectification strategy based on entropy minimisation was then applied to the dense 3D information to maintain the global consistency of the vehicle trajectory [48]. The algorithm was tested offline on sequences captured in practical environments and it was observed that the egomotion algorithm was very robust and produced trajectory estimates that were so close to the minimal energy configuration that the SLAM algorithm only had to fine tune the localisation [48]. Unfortunately,

while the egomotion is able to operate at near real-time speeds the global rectification process has a computational complexity proportional to the volume of the map, thus considerable work is required to achieve real-time operation [48].

Mahon et. al. [21] developed an EKF based SLAM algorithm which utilised a pencil beam scanning sonar and high resolution stereo vision system to perform both motion estimation and localisation. The feature selection process used in the implementation is described in detail in Section 2.2. While the algorithm is relatively simple, real-time operation was not achieved as robust segmentation of the underwater images was computationally expensive [21]. However, the implementation was tested offline on sequences captured in practical environments and the algorithm was able to provide adequate motion and localisation estimates over short time periods [21].

In his thesis Eustice [49] developed an information filter based SLAM algorithm that was exactly sparse and operates completely in the information space, thus obtaining linear complexity in the number of landmarks. In conjunction with Pizarro et. al. [50] and other authors [51], Eustice demonstrated the ability to produce large scale 3D reconstructions of underwater environments from monocular video and IMU data logged on several ocean surveys. The solution uses a modified Harris corner detector to select interest point locations and extracts an elliptical patch around each point that is invariant to affine geometric transformations [51]. These patches are then represented compactly using Zernike moments with full complex moments (invariance to rotation compensated using attitude sensors) [51]. Motion estimation is based on a robust estimate of the essential matrix which is computed from a set of tracked features which are carefully selected by constraining correspondences based on predicted motion uncertainty and applying RANSAC [51]. These features are stored in a local submap which is closed and bundle adjusted once a specified number of points have been observed [51]. The global map is then refined by searching for spatial relationships between submaps and applying iterative closest point (ICP) techniques to register the sets of 3D points and refine the alignment [51]. This approach was demonstrated to produce consistent reconstructions for both small-scale and large-scale environments, but to date has only been demonstrated offline despite the seemingly low computational complexity [51].

### **Aerial SLAM**

Similarly to underwater SLAM, aerial SLAM suffers from several difficulties in terms of computational complexity and loop closure; and high nonlinearity in both vehicle dynamics and observations [46]. To the best of our knowledge only four real-time solutions have been published. The first was implemented on a low-dynamic blimp-type platform equipped with a stereo vision system [52]. The second was implemented for small UAVs and was demonstrated

in a laboratory experiment [53]. The third is presented in a series of publications by Kim et. al. [54], [47] and [46] and describes the implementation of EKF-SLAM on a fixed-wing UAV with inertial sensors and a monocular vision system. The final was by Euston and Kim [6] and investigated the potential of a novel Rao-Blackwellised approach on the same platform.

Jung et. al. [52] developed a real-time 6DoF EKF-SLAM algorithm which was deployed on a unmanned blimp in order to construct high resolution digital elevation maps from a sequence of unregistered low altitude stereo images. The approach relies solely on visual motion estimation based on the observed landmarks to determine the vehicle motion between consecutive frames [52]. An EKF then estimates the 6 vehicle pose parameters and also the 3D location of all the stored landmarks [52]. Experimental results over a 60m trajectory showed that the precision of the method enables the construction of large spatially consistent maps [52]

Langelan et. al. [53] developed a method for passive GPS-free navigation of a small UAV equipped with only an IMU and a single camera. Langelan et. al. [53] noticed that when sensing is limited to inertial measurements and bearing measurements EKF-SLAM approaches are often divergent. Therefore, they chose to employ an unscented Kalman filter, which leads to consistent unbiased estimates of both vehicle and landmark states [53]. The algorithm was tested in a simulated 20m low altitude (5m) loop through randomly distributed trees, and a larger 1hour long loop with the vehicle flying between the trees on a circular trajectory with a 200m radius [53]. For both simulations error growth during exploration of new terrain was demonstrated to be roughly constant at 1.7% of the distance travelled [53]. The algorithm was then tested in real-time in a laboratory environment for a curved trajectory of approximately 8m in length and the error was observed to grow to 3.25% of the distance travelled [53].

Kim et. al. [54] developed a 6DoF SLAM algorithm to compliment a GNSS/INS based navigation system and support operation in environments where GPS signals are not available. The 6DoF EKF-SLAM algorithm incorporates an IMU as its core dead-reckoning sensor and uses a monochrome camera to constrain the INS errors by providing range, bearing, and elevation measurements based on environment landmarks. This work was later improved by Kim et. al. [47] and the computational complexity of the solution was reduced by recasting the SLAM filter into a linearised error state form and treating the map as an external database to remove the need for computationally expensive SLAM map updates when GNSS is available [47]. The system was verified on a fixed-wing UAV which followed three race-horse trajectories and localised successfully even when GNSS was unavailable for extended periods of time [47]. Kim et. al. [46] published a more detailed description of this solution and the testing performed in [46] and identified that the most significant source of error was in the range calculation in undulating 3D terrain and that the difference between the SLAM estimated trajectory and GNSS/Inertial trajectory was largely due to errors in the initial parameters and height deviation [46]. While this is probably the state-of-the-art in real-time 6DoF SLAM several issues

remain to be solved [46]. Comparing the estimates produced by the algorithm to the ground truth it is evident that the SLAM filter maintains over-confident covariance estimates [46]. While this could be addressed by analysing the errors arising from the inertial sensors, increasing the process noise makes the vehicle attitude estimates more sensitive to observation updates [46]. Therefore, work is required to either partition the attitude updates from the velocity/position updates or to fuse them with an external aiding source such as an air-data system [46]. Other theoretical, technical and practical issues such as evaluating consistency, synchronising vision and INS, performing robust natural feature detection and description, and incorporating sub-map techniques are required in order to enable long-term and/or large-scale deployment [46]. A related publication by Bryson et. al. [55] utilised the same framework but investigated the potential of "active SLAM", in which the vehicle trajectory is actively modified while exploring unknown terrain in order to re-observe landmarks sufficiently often to keep the uncertainty in the vehicle pose within acceptable limits.

Kim and Sukkarieh's [46] EKF-SLAM approach presented above inherently suffers from quadratically increasing complexity in the number of landmarks and consequently Euston and Kim [6] investigated the potential of a Rao-Blackwellised particle filtering (RBPF) based solution. The RBPF approach cannot be directly applied to 6DoF with inertial-based measurements due to the high dimensionality of the state, which typically has position, velocity, and attitude states, as well as sensor error states for gyroscopes and accelerometers [6]. This leads to a dimensionality of 15, compared to just three for land based robots which generally only estimate x-y position and heading [6]. Because the number of particles required for RBPF increases exponentially with the state dimension, direct application to 6DoF with inertial-based measurements is computationally infeasible. [6]. As a result, Euston and Kim [6] proposed a novel technique in which the full state is partitioned into an external INS state (vehicle pose), an internal INS state (navigation and sensor calibration) and map states, with a particle filter being applied only to the external INS state and the remainder updated via parallel Kalman filters. However, unlike the conditionally independent map states, the internal INS states are dependent on the pose particles [6]. Therefore, Kim and Euston [6] utilise a hybrid approach for vehicle pose estimation which uses both a full INS EKF and a pose-sampled RBPF. At present the algorithm has only been tested offline on simulated data and was shown to perform reliably with 50 particles, which to the best of my knowledge represents the state-of-the-art in RBPF approaches to 6DoF SLAM [6].

### Monocular SLAM

Strangely almost all of the research into 6DoF SLAM for land-based applications has focused on cost effective solutions that utilise a single camera. Davison [56] provided the impetus for

this line of research when he demonstrated the ability to perform SLAM with a single camera in indoor environments using the EKF. Davison [57] later extended this algorithm to initially represent features with a particle distribution, rather than incorporate them immediately into the extended Kalman filter, which overcomes problems with the initial range uncertainty due to the use of a single camera. This algorithm was able to operate in unknown environments and generate a map of 3D point features in real-time [57]. This work was further refined by Davison et. al. [58] and research in conjunction with Civera et. al. [59] showed that the fundamental computer vision problem of structure from motion with a single camera can be tackled using the sequential, probabilistic methodology of monocular SLAM. Civera et. al. [60] also implemented a fully Bayesian Interacting Multiple Models (IMM) framework which is capable of automatically switching between parameter sets in a dimensionless formulation of monocular SLAM. Works by various other authors have also incorporated and compared the types of features used for landmark identification.

Burchka et. al. [61] developed a monocular camera SLAM algorithm for robots designed to operate off-road. The algorithm tracks a set of natural landmarks at video frame-rate and pre-processes the images to compensate for occlusions and decreasing reconstruction accuracies as the distance to the landmark increases [61]. Off-road robots are generally equipped with soft tyres to absorb vibrations, but these also cause the vehicle to bounce and tilt in an unpredictable fashion, making odometry information invalid [61]. To counter this problem the vision system is coupled with an IMU capable of providing complete 6DoF motion estimates [61]. The system estimates the vehicle pose using an image-based approach which compares the 2D projections of the internal 3D map (estimated up to a scale) between images and relates the deviation to position errors in 3D space [61]. In this way the algorithm always operates in image coordinates and the 3D map can be represented in a way that does not require any knowledge about the three-dimensional position in the world to register the reconstructions to each other [61]. This pose estimation method is capable of operation under significant pose variations between the reference and actual position provided that at least 3 landmarks can be matched [61]. To avoid the convergence problems normally associated with approaches based on image Jacobian matrices, Burchka et. al. [61] use a recursive model-adaptation method operating in the 2D image space [61]. The system has not yet been tested on autonomous robots, but is currently used for camera localisation in endoscopic procedures where the reconstructed 3D features are registered with an anatomic CT-scan of the patient [61].

Pinies et. al. [62] investigated the benefits of using a low cost IMU to aid an implementation of inverse depth monocular SLAM similar to that of Davison et. al. [58]. They demonstrated that the inertial observations constrain the uncertainty of the camera pose which improves the accuracy of the estimated trajectory [62]. When using a single camera the scale of the scene is unobservable, so the scale of the map generated depends on the prior used to initialise the

depth of the features and this may drift as loops grow larger [62]. However, the results showed that the inertial observations also reduced the variation in the scale factor between features in the map and consequently a more consistent map with less uncertainty is obtained [62]. Finally, it was noted that the use of an IMU enables more accurate predictions of camera egomotion, which enables more accurate prediction of the location of features in the next image, which leads to improvements in robustness and the efficiency of data association [62].

## 2.6 Available Resources

During the survey the openCV, Bayes++ and Scene open-source and GPL licensed software libraries were identified as containing material that could be directly applied to our problem.

The openCV library is targeted at real time computer vision applications and contains functions that are designed for human-computer interaction, object identification, segmentation, face recognition, gesture recognition, motion tracking, ego-motion, structure from motion (SFM) and mobile robotics. All of the core libraries are written in C/C++.

The Bayes++ open source library provides classes that implement a wide variety of numerical algorithms for Bayesian Filtering of discrete systems. The classes utilise tested and consistent numerical methods and the class hierarchy explicitly represents a variety of filtering algorithms and system model types including the extended Kalman filter and information filter.

The Scene open source library is a flexible software library for SLAM in C++, which provides a generic SLAM framework which can theoretically be applied to almost any SLAM problem. Scene achieves this by separating the core probabilistic SLAM framework from the details of a particular application. Therefore, the user must implement model classes that represent the specifics of a practical system and plug these into the framework.

## Chapter 3

# Real-time Stereo Visual SLAM

### 3.1 Proposed Solution

The ultimate goal of this project is to implement a real-time visual SLAM algorithm which can be deployed on an 6-DoF autonomous underwater vehicle (AUV). Due to the availability of low cost underwater cameras a stereo camera setup will be employed as this eliminates the problems associated with feature initialisation and measurement present in monocular SLAM. A wide baseline will be used in order to achieve the desired accuracy ( $\leq 5\text{cm}$  error) over the anticipated measurement range (0.5 - 10m) and the cameras will be calibrated offline before mounting them on the vehicle. Based on the survey of feature descriptors presented in Section 2.2 a variable mixture of SIFT and SURF features will be used, as this should enable a dynamic trade-off between quantity, accuracy and computational complexity and take advantage of their fundamentally similar descriptors. Landmarks will in turn be described by local submaps that consist of clouds of 3D points and their corresponding 2D feature descriptors. Consequently, landmarks will be sparsely distributed and extremely distinctive, which should enable simple, low complexity maximum likelihood data association based on pairwise matching of the 2D descriptors and registration of the corresponding 3D points. This registration will also yield an estimate of the rotation and translation of the camera between successive observations of the same landmark which should enable frequent estimations of inter-frame vehicle motion. Due to the low computational complexity of EKF-SLAM approaches which utilise partitioned updates and state augmentation and the large number of improvements that may be directly incorporated in the future (such as submapping and robust data association) an EKF based approach is proposed for this project. The Bayes++ library will be utilised as it provides a well defined, hierarchical framework for implementing filters and models using the Boost uBLAS

matrix library, and is sufficiently flexible to enable almost any state based implementation. A (non-linear) constant velocity prediction model is proposed in order to provide maximum portability, however, if performance is not satisfactory then the full dynamic model of the vehicle will be incorporated. In order to identify bottlenecks and achieve real-time operation the average and worst-case execution times of the algorithms will be tested on the development platform, which has a 2Ghz Intel Centrino processor and 1Gb of memory.

## 3.2 Implementation

A complete EKF-SLAM implementation has been produced in C++ along with a Matlab simulation environment and auxiliary tools to aid testing and development such as an XML configuration file parser and results analyser. This SLAM implementation employs a novel approach to landmark description and data association and if deployed on the Nessie II AUV for the European SAUC-E competition would be to the best of our knowledge the first deployment of a real-time implementation of vision-based SLAM in an underwater environment.

### 3.2.1 Landmark Observation

The system utilises a calibrated stereo camera setup, which consists of two CCD underwater cameras with a baseline of approximately 0.5m and a tilt of approximately  $15^\circ$  on the right camera. The left and right cameras are synchronised using a frame grabber and the captured images are converted to greyscale and undistorted using the radial and tangential distortion estimates obtained from the calibration. The user is able to extract both SIFT and SURF features which are both fundamentally similar and rely upon condensing the large amount of image information into a small number of distinctive descriptors. The first step in both algorithms is to construct a scale-space, Hessian for SURF and Gaussian for SIFT, and to locate maxima (keypoints) within this scale space. For each of the 360x288 pixel images the implementation requires an average of 104ms to generate a Gaussian scale space with 6 octaves and extract the maxima, compared to 31ms for the equivalent Hessian scale space. While it was mentioned in Section 2.2 that the Harris corner detector provides keypoints that are more robust to changes in viewpoint than both SIFT and SURF keypoints, the real power of SIFT and SURF lies in their compact, robust and distinctive descriptors. The SIFT descriptor for each keypoint is generated using the SIFT++ algorithm which is available free online and is capable of producing a single descriptor approximately every 1.9ms. Similarly, the SURF descriptors are generated using the SURF algorithm which is available free online and is capable of producing a single descriptor approximately every 1.5ms. While SURF may not seem considerably faster, if we consider a pair of images containing just 200 features each, the relative execution times based on

these averages would be 662ms and 968ms for SURF and SIFT respectively. Consequently, the fact that the SIFT algorithm is able to produce more keypoints is not always an advantage, as in many cases the number of keypoints retained must be reduced in order to obtain satisfactory execution times. However, at present the full 128 element SIFT descriptor is used, whereas SURF uses a 64 element descriptor, thus one possibility for accelerating the SIFT descriptor generation would be to reduce the descriptor size.

Once features have been extracted from each image the next step is to establish their putative correspondences. The current implementation computes the sum of squared differences (SSD) between each pair of descriptors and selects only those pairings for which the ratio of the SSD with the nearest neighbour to the SSD to next nearest neighbour is below a fixed threshold. This matching approach is extremely fast and for the data tested up to 398 features were compared in less than 1ms, therefore feature matching time is negligible. This approach generally yields a large number of matches which can then be filtered to remove outliers by applying epipolar constraints. The fundamental matrix,  $F$  is a  $3 \times 3$  matrix obtained from the camera calibration which describes the relative camera pose and defines a bi-linear constraint between the coordinates of corresponding image points. Using the fundamental matrix it is possible to map any point in one camera to a corresponding line (epipolar line) in the opposite camera which represents all possible locations at which the same ray could be projected taking into account all possible scene depths as illustrated in Figure 3.1.

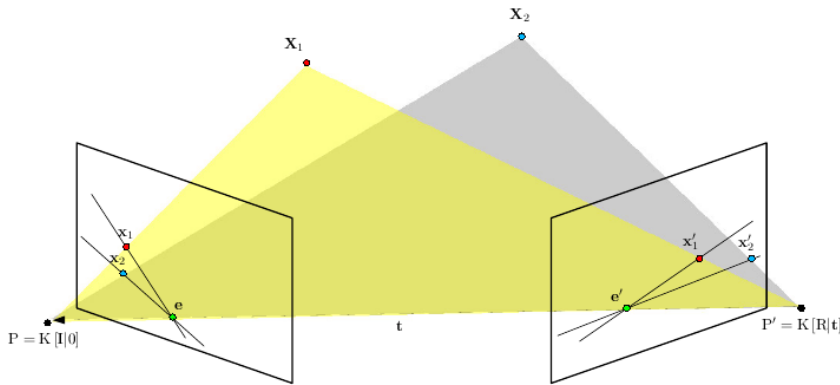


Figure 3.1: Epipolar geometry - Two 3D points  $X_1$  and  $X_2$  are projected into the left and right image planes using camera projection matrices  $P$  and  $P'$  to obtain  $x_1, x_2$  and  $x'_1, x'_2$  respectively. An epipolar plane is defined by each 3D point together with the two camera centres and the associated epipolar lines correspond to the intersection of this plane with the image planes. In the absence of noise any point which lies on the epipolar plane (at different scene depths) will be projected onto the associated epipolar lines in each image plane. [3]

The epipolar line,  $l'$  in the right image for a point  $x$  in the left image is given by  $l' = Fx$  and conversely the epipolar line,  $l$  in the left image for the corresponding point  $x'$  in the right image is given by  $l = F^T x'$ . Because in each case the corresponding point must lie on the epipolar line we must have  $x \cdot l = 0$  and  $x'^T \cdot l' = 0$ , which means the epipolar constraint can be written as  $x'^T Fx = 0$ . Applying this constraint, for each matched feature in the left image we can compute the deviation of the matched feature in the right image from the corresponding epipolar line and eliminate the majority of outliers by applying a simple threshold. This is illustrated for a pair of real images in Figure 3.2. The optimal threshold can be empirically determined by analysing the variance in the point to line distance for the calibration data.

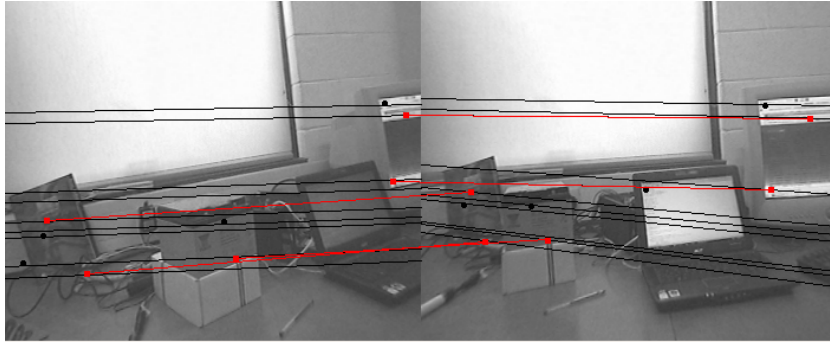


Figure 3.2: Feature matching and outlier removal - The epipolar lines (black) are shown for each matched point (dots) and a correspondence line (red) is shown for the matched left and right points which are sufficiently close to the corresponding epipolar lines

It is possible to determine the depth of each matched point in the scene by triangulation provided that the metric rotation and translation from one camera to the other is known. The essential matrix  $E$  encapsulates this information is computed from the fundamental matrix and the left and right camera calibration matrices  $K$  and  $K'$  according to the equation:

$$E = K'^T F K \quad (3.1)$$

The multiplication with the camera calibration matrices removes the dependence on the camera parameters and the rotation and translation can be determined by factoring the essential matrix. If we consider the singular value decomposition  $E = USV^T$  the rotation will be given by:

$$R = UWV^T \quad \text{or} \quad R = UW^T V^T \quad \text{where} \quad W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

and the translation,  $t$  by either:

$$t = u_3 \quad \text{or} \quad t = -u_3 \quad (3.3)$$

where  $u_3$  is the third column of  $U$ . Therefore, there are four possible combinations of  $R$  and  $t$ , but only one correctly describes the geometry of the setup. This combination corresponds to the case in which the data points are in front of both cameras, which results in a positive depth for both cameras. As shown in Figure 3.1 the pose matrix of the left camera is  $P = KP_{cam} = K[I|0]$  and the pose matrix of the right camera is  $P' = K'P'_{cam} = K'[R|t]$ . If the matched point in each image is converted to metrics according to  $\tilde{x}_{cam} = K^{-1}x_i$  and  $\tilde{x}'_{cam} = K'^{-1}x'_i$  then the following relationships exist with the 3D point  $\tilde{X}_i$  up to a scale:  $\tilde{x}_{cam} = P\tilde{X}_i$  and  $\tilde{x}'_{cam} = P'\tilde{X}_i$ . Expanding and combining these equations for both cameras we obtain:

$$\begin{bmatrix} P_3x_i & - & P_1 \\ P_3y_i & - & P_2 \\ P'_3x'_i & - & P'_1 \\ P'_3y'_i & - & P'_2 \end{bmatrix} \tilde{X}_i = 0 \quad (3.4)$$

where  $P_i$  is the  $i^{th}$  row of the  $P$  matrix. This problem has the form  $Ax = 0$  and can be easily solved by least squares or singular value decomposition. It is possible to show that  $\tilde{X}_i$  is the eigenvector corresponding to the smallest eigenvalue of  $A^T A$ , therefore if we compute  $A = USV^T$  then  $\tilde{X}_i$  is equivalent to the last column of  $V$ . Finally, we normalise this eigenvector by the last element to obtain the 3D scene point in homogeneous coordinates. These 3D points and their corresponding 2D points and feature descriptors are then stored together as a local submap. During this step the 3D points are transformed from the camera coordinate system to the coordinate system of the vehicle using a fixed homography. This permits the local submap to contain features observed from different sensors all referenced to a single coordinate system. In this implementation we represent landmarks as a single, complete local submap and use the centre of gravity of the cloud of 3D points as an anchor point in the global map. Therefore, each landmark has an arbitrary shape (non-geometric landmarks) and combines many robust SIFT or SURF features, which results in extremely distinctive and very robust landmark descriptions. To the best of our knowledge no existing SLAM implementation utilises a similar approach, and almost all employ significantly less descriptive features such as edges, contours, individual corners or individual geometric descriptors.

### 3.2.2 Data Association

Low overlap imagery is common for AUVs due to the fact that vehicle speeds are moderate, video frame rates are usually low, and attenuation, scattering and limited illumination restrict visibility and dictate that the vehicle must stay relatively close to the seabed which reduces the effective field of view for each image. Taking this into account and also the fact that landmarks are represented by large sparsely distributed submaps, it is reasonable to assume that usually only one landmark will be visible at any point in time. As a result the current approach to data association simply involves identifying the maximum likelihood match between the current observation and the set of map landmarks based on both the 3D points and 2D descriptors.

Firstly, the 2D descriptors of the current observation are compared to the set of map landmarks that are identified as potential matches based on the distance between their estimated global position and the current estimate of the vehicle global position. This matching utilises the same gated nearest neighbour algorithm described in Section 3.2.1. These matches generally contain a large number of outliers, even when re-observing the same landmark, due to the changes in viewpoint and indistinct descriptors associated with surfaces such as sand, mud and rock. Consequently, the epipolar constraint is applied in the same manner as described in Section 3.2.1 to eliminate as many outliers as possible. However, in this case the fundamental matrix is unknown and depends on the rotation and translation relative to the first observation of the landmark. Therefore, the algorithm attempts to robustly estimate the fundamental matrix using the normalised 8-point algorithm and least median squares (LMedS) method.

The normalised 8-point algorithm was described by Hartley [63] and is an improvement to the 8-point algorithm developed by Christopher Longuet-Higgins in 1981, which is capable of estimating the fundamental matrix from a set,  $p$  of at least eight image points and a set,  $p'$  of corresponding image points. Hartley [63] recognised that the original algorithm produced incorrect matrices when the image points (in homogeneous coordinates) are poorly distributed in space. Hartley [63] realised the reason for this was that all the vectors described by the homogeneous image coordinates pointed in almost the same direction. Accordingly, Hartley [63] proposed that the coordinate systems of both images should be independently transformed in order to better distribute the vectors. This transformation involves shifting the origin of the coordinate system to the centre of gravity of the image points and then uniformly scaling the image coordinates such that their mean distance to the origin is  $\sqrt{2}$ . If we denote the combined translation and scaling transforms by  $T$  and  $T'$ , the new sets of image points  $\bar{p}$  and  $\bar{p}'$  are obtained from  $p$  and  $p'$  respectively according to:

$$\bar{p} = Tp \quad \text{and} \quad \bar{p}' = T'p' \quad (3.5)$$

These normalised image coordinates are then directly used in the 8-point algorithm and the

resulting fundamental matrix,  $\bar{F}$  is denormalised to obtain the true fundamental matrix,  $F$  according to:

$$F = (T')^T \bar{F} T \quad (3.6)$$

Generally the resulting fundamental matrix is much more accurate than the fundamental matrix obtained using unnormalised coordinates [63]. The original 8-point algorithm is based on the fact that for each pair of corresponding points  $\bar{x} \in \bar{p}$  and  $\bar{x}' \in \bar{p}'$  we have a single linear constraint on  $\bar{F}$  provided by the epipolar constraint  $\bar{x}'^T \bar{F} \bar{x} = 0$ . This allows  $\bar{F}$  to be estimated linearly (up to an arbitrary scale factor) from 8 independent correspondences by constructing a linear system of the form  $A \bar{f} = 0$ , where  $\bar{f}$  is a vector containing the 9 coefficients of  $\bar{F}$ , and each row of  $A$  contains the coordinates  $\bar{x}$  and  $\bar{x}'$  of a single correspondence. The coefficients of  $\bar{F}$  are then estimated by expressing the relationship as a least squares problem and finding  $\bar{f}$  which minimises  $\|\bar{f}^T A\|^2$  for  $\|\bar{f}\| = 1$ . In fact, this is equivalent to choosing  $\bar{f}$  to be the eigenvector associated with the smallest eigenvalue of the  $9 \times 9$  symmetric, positive semidefinite (PSD) matrix  $A^T A$ . When the resulting  $\bar{f}$  vector is rearranged to be  $3 \times 3$  the resulting matrix may not satisfy the rank 2 constraint of the fundamental matrix. Therefore, the singular value decomposition of the matrix is computed and the smallest eigenvalue is forced to zero before recomposing the matrix to obtain a rank 2 estimate of  $\bar{F}$ .

For most applications this approach is not sufficiently robust as the set of correspondences may contain outliers and the correspondence points will almost certainly be corrupted by noise. Therefore, the algorithm utilises the least median squares (LMedS) framework to reduce the effect of outliers by using random sampling to identify inliers in the data set. In each iteration of the LMedS algorithm a random subset of 8 correspondences is selected, an estimate of the fundamental matrix is computed using the normalised 8-point algorithm, and the median of the symmetric epipolar distances for the complete data set is computed. At the end of each iteration the algorithm estimates the confidence level that the fundamental matrix is correct and terminates once this is above a specified level or continues until it finds the matrix which minimises this median. Therefore, by adjusting the desired level of confidence it is possible to control the tradeoff between the number of iterations of the algorithm and the likelihood of obtaining an incorrect fundamental matrix. Due to the fact that the current implementation never reconsiders matchings, false negatives are preferred over false positives and it is acceptable to sacrifice some execution time in order to ensure the probability of accepting outliers is minimised. As a result the level of confidence must be  $> 95\%$  and the threshold which specifies the maximum distance between each point and its corresponding epipolar line, beyond which the point is considered an outlier must be  $< 1$  pixel. It is not possible to quantify the execution time of the LMedS algorithm as it is non-deterministic and dependent on the number of matches,

projection noise and percentage of outliers. However, to give an indication, for a simulation in which the projection noise is Gaussian with a standard deviation of 1 pixel, 20% of the matchings are outliers, and the average number of matches per observation is 209 the algorithm requires on average 33ms to compute the fundamental matrix.

When the LMedS algorithm completes the set of matched points that are considered inliers according to the epipolar constraint are already known. Therefore, the next step in the algorithm is to register the corresponding sets of 3D points in order to measure their compatibility and accurately estimate the relative rotation and translation. Using this rotation and translation it is possible to find the position of the landmark centre of gravity even when a subset of the landmark points are re-observed. In addition, the compatibility of the observation with the landmark can also be evaluated based on the least mean squared (LMS) registration error. Therefore, an observation is matched to a landmark if and only if a sufficient number of matched points are inliers and the LMS error falls below a fixed threshold, otherwise, a new landmark is created provided the new landmark is sufficiently distant from all other landmarks.

### 3.2.3 Motion Estimation

The final step of the data association algorithm described in Section 3.2.2 involves a registration of the set of observed 3D points with a map landmark, which provides an estimate of the relative rotation and translation between the first and current observation of the landmark. Therefore, by recording the estimated position and pose of the vehicle when the landmark is created it is possible to perform fairly robust inter-frame motion estimation with almost no additional computational burden. However, due to the fact that only one landmark is observed at any given time, continuous visual odometry that is independent of the Kalman filter state is not attainable. This limitation arises from the fact that when the algorithm switches from one landmark to another (a new landmark or different map landmark) it is not possible to recover the rotation and translation during this switch-over period. Consequently, while this motion estimation approach can significantly reduce the error in the estimation of the vehicle location and pose due to its high accuracy, estimates may be obtained infrequently and inherit the error accumulated in the Kalman filter state during the switch-over period. Therefore, it may be beneficial to utilise this approach in combination with a motion estimation technique which provides more frequent (but probably less accurate) estimates of inter-frame motion. Accordingly, the optical flow of a sparse set of KLT features was computed using the iterative Lucas-Kanade pyramidal algorithm available in the OpenCV library. Assuming that the terrain is roughly planar it should be possible to estimate the 6DoF motion by robustly computing the inter-frame homographies of the tracked features using RANSAC. However, in practise it was observed that without constraining the number of degrees of freedom the error in the motion

estimation was too high to merit the additional computational load (approximately 150ms per frame). Consequently, this approach has not been incorporated into the final implementation. However, parallel work by other students is investigating the potential of a similar approach based on the motion vectors of the MPEG compression when only planar translation is estimated and depth and orientation are recovered using a depth sensor and IMU.

### 3.2.4 EKF-SLAM and UKF-SLAM

An EKF-SLAM algorithm was implemented using the Bayes++ open source library. By implementing the SLAM algorithm, models, and customised filter functionality within this well defined hierarchical framework it is possible to switch between fundamentally similar state based filters such as the extended Kalman filter and unscented Kalman filter with minimal work. Furthermore, the Bayes++ library is built around and used in conjunction with the Boost uBLAS matrix library, which provides an excellent selection of matrix operations and when used with an optimised compiler is exceptionally fast. In addition, the Bayes++ library performs automatic checks to detect numerical failures and ill conditioned matrices and also maintains the symmetry of matrices which ensures the algorithm does not continue to trust state estimates that are undoubtedly incorrect.

In this implementation the Kalman filter state contains 6 vehicle states: x, y and z position and roll, pitch and yaw angles all relative to a global coordinate system. In addition, the Kalman filter state contains the x, y and z position of the centre of gravity of  $N$  landmarks with respect to the global coordinate system. Therefore, the full EKF state  $x_k$  has the form:

$$\hat{x}_k = \left[ \hat{v}_{px} \quad \hat{v}_{py} \quad \hat{v}_{pz} \quad \hat{v}_{rr} \quad \hat{v}_{rp} \quad \hat{v}_{ry} \quad \hat{m}_{0x} \quad \hat{m}_{0y} \quad \hat{m}_{0z} \quad \dots \quad \hat{m}_{N-1x} \quad \hat{m}_{N-1y} \quad \hat{m}_{N-1z} \right]^T$$

where  $\hat{v}$  is a 6 element vector containing an estimate of the vehicle position and pose and  $m_i$  is a 3 element vector containing the estimated global position of the  $i^{th}$  landmark. Accordingly, the full EKF covariance  $P_k$  has the form:

$$P_k = \begin{bmatrix} P_{vv} & P_{vm} \\ P_{vm}^T & P_{mm} \end{bmatrix} \quad (3.7)$$

Due to the structure, size and distribution of landmarks, the number of landmarks contained in the filter is significantly lower than typical SLAM implementations and provides considerable reductions in prediction and observation update times for comparatively sized maps.

Using the common scheme interface provided by the Bayes++ library it is possible to quickly shift to estimating the covariance using the unscented Kalman filter approach which propagates

samples through the non-linear models rather than computing the Jacobians as described in Section 2.1.1. Consequently, using the current implementation we were able to evaluate the relative performance of both schemes with respect to computational complexity and accuracy.

### Prediction Update

The current prediction update is based on a non-linear constant velocity model, which provides a generic, platform independent solution. As described in Section 2.1.1 only the vehicle position and pose are affected by the prediction model so the update can be performed in linear time using the augmented state approach. In this case, the prediction update of the state is trivial and can be expressed as:

$$\hat{x}_{k|k-1} = \begin{bmatrix} f_v(\hat{v}_{k-1|k-1}, u_k) \\ \hat{m} \end{bmatrix} \quad (3.8)$$

where  $\hat{v}$  is the current estimate of the 6 vehicle states contained within the full EKF state, and the function  $f_v(\hat{v}_{k-1|k-1}, u_k)$  is the augmented state version of Equation 2.5 which models the motion kinematics and is described by:

$$\begin{aligned} \hat{v}_{px,k|k-1} &= \hat{v}_{px,k-1|k-1} + cv * \cos(\hat{v}_{rp,k-1|k-1})\cos(\hat{x}_{ry,k-1|k-1}) \\ \hat{v}_{py,k|k-1} &= \hat{v}_{py,k-1|k-1} + cv * \cos(\hat{v}_{rp,k-1|k-1})\sin(\hat{x}_{ry,k-1|k-1}) \\ \hat{v}_{pz,k|k-1} &= \hat{v}_{pz,k-1|k-1} - cv * \sin(\hat{v}_{rp,k-1|k-1}) \\ \hat{v}_{rr,k|k-1} &= \hat{v}_{rr,k-1|k-1} \\ \hat{v}_{rp,k|k-1} &= \hat{v}_{rp,k-1|k-1} \\ \hat{v}_{ry,k|k-1} &= \hat{v}_{ry,k-1|k-1} \end{aligned}$$

where  $cv$  is a parameter which specifies the constant velocity of the system and  $u_k = 0$  as there are no control inputs. The prediction update of the covariance described by Equation 2.6 has cubic complexity in the number of landmarks, whereas the augmented state version has linear complexity in the number of landmarks and has the form:

$$P_{k|k-1} = \begin{bmatrix} \nabla f_v P_{vv} \nabla f_v^T + Q_k & \nabla f_v P_{vm} \\ P_{vm}^T \nabla f_v^T & P_{mm} \end{bmatrix} \quad (3.9)$$

where  $\nabla f_v$  is the Jacobian of  $f_v(\cdot)$  evaluated at the estimate  $\hat{v}_{k-1|k-1}$  and is described by:

$$\nabla f_v = \begin{bmatrix} 1 & 0 & 0 & 0 & -cv * \sin(\hat{v}_{rp,k-1})\cos(\hat{v}_{ry,k-1}) & -cv * \cos(\hat{v}_{rp,k-1})\sin(\hat{v}_{ry,k-1}) \\ 0 & 1 & 0 & 0 & -cv * \sin(\hat{v}_{rp,k-1})\sin(\hat{v}_{ry,k-1}) & -cv * \cos(\hat{v}_{rp,k-1})\cos(\hat{v}_{ry,k-1}) \\ 0 & 0 & 1 & 0 & -cv * \cos(\hat{v}_{rp,k-1}) & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and the zero mean uncorrelated Gaussian errors  $w_k$  which affect the motion observation and have covariance  $Q_k$  are specified in the configuration file and must be tuned according to the environment. Studying the structure of the augmented state prediction update described above it is evident that the mean and covariance prediction update can be performed efficiently according to the following steps:

1. Construct a new EKF which has a state equivalent to the current estimate of the 6 vehicle states from the full EKF and a covariance matrix equivalent to the associated 6x6 subsection of the full EKF covariance matrix
2. Perform the prediction update on this new EKF and copy the predicted state and covariance to the full EKF
3. Update the full EKF covariance between the vehicle states and landmarks by computing  $\nabla f_v P_{vm}$  and also copying this result to the associated locations in the symmetric matrix.

and the result is equivalent to the naive prediction update defined by Equations 2.5 and 2.6.

### Observation Updates

The motion observations are already relative to the global coordinate system and consequently only a simple linear observation model is required in which the predicted observation is equal to the current estimate of the vehicle state. However, care must be taken to ensure that the roll, pitch and yaw angles remain in the range  $[-\pi \pi]$  and that the EKF innovation is computed correctly when the observed and state angles lie on opposite sides of the  $\pm\pi$  boundary. The landmark observations are considerably more complicated due to the fact that landmarks are observed in the camera coordinate frame but are stored in the global coordinate frame. To simplify the process and utilise a common observation model for all sensor measurements, all observations are pre-transformed to a common vehicle coordinate frame using fixed homographies (assume that sensors are stationary). Given a translation  $T$ , roll  $RR$ , pitch  $RP$  and yaw  $RY$  which describe the position and pose of the vehicle relative to the global coordinate system:

$$T = \begin{bmatrix} \hat{v}_{px,k|k-1} \\ \hat{v}_{py,k|k-1} \\ \hat{v}_{pz,k|k-1} \end{bmatrix} \quad RR = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\hat{v}_{rr,k|k-1}) & -\sin(\hat{v}_{rr,k|k-1}) \\ 0 & \sin(\hat{v}_{rr,k|k-1}) & \cos(\hat{v}_{rr,k|k-1}) \end{bmatrix}$$

$$RP = \begin{bmatrix} \cos(\hat{v}_{rp,k|k-1}) & 0 & \sin(\hat{v}_{rp,k|k-1}) \\ 0 & 1 & 0 \\ -\sin(\hat{v}_{rp,k|k-1}) & 0 & \cos(\hat{v}_{rp,k|k-1}) \end{bmatrix} \quad RY = \begin{bmatrix} \cos(\hat{v}_{ry,k|k-1}) & -\sin(\hat{v}_{ry,k|k-1}) & 0 \\ \sin(\hat{v}_{ry,k|k-1}) & \cos(\hat{v}_{ry,k|k-1}) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = RY * RP * RR$$

the homogeneous transformations between the base and world coordinate systems are given by:

$${}^{world}T_{base} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad {}^{base}T_{world} = \begin{bmatrix} R^T & -R^T T \\ 0 & 1 \end{bmatrix}$$

Therefore, the predicted observation  $\hat{z}_{i,k}$  for a single landmark  $m_i$  is given by:

$$\hat{z}_{i,k} = h(\hat{v}_{k|k-1}, \hat{m}_{i,k-1}) = {}^{base}T_{world} \begin{bmatrix} \hat{m}_{i,k-1} \\ 1 \end{bmatrix} \quad (3.10)$$

and the corresponding Jacobian  $\nabla h$  is given by:

$$\left[ \begin{array}{ccc|ccc|ccc} \frac{\delta \hat{z}_{ix,k}}{\delta v_{px,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta v_{py,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta v_{pz,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta v_{rr,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta v_{rp,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta v_{ry,k-1}} & \frac{\delta \hat{z}_{ix,k}}{\delta m_{ix,k-1}} & \frac{\hat{z}_{ix,k}}{\delta m_{iy,k-1}} & \frac{\hat{z}_{ix,k}}{\delta m_{iz,k-1}} \\ \frac{\delta \hat{z}_{iy,k}}{\delta v_{px,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta v_{py,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta v_{pz,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta v_{rr,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta v_{rp,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta v_{ry,k-1}} & \frac{\delta \hat{z}_{iy,k}}{\delta m_{ix,k-1}} & \frac{\hat{z}_{iy,k}}{\delta m_{iy,k-1}} & \frac{\hat{z}_{iy,k}}{\delta m_{iz,k-1}} \\ \frac{\delta \hat{z}_{iz,k}}{\delta v_{px,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta v_{py,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta v_{pz,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta v_{rr,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta v_{rp,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta v_{ry,k-1}} & \frac{\delta \hat{z}_{iz,k}}{\delta m_{ix,k-1}} & \frac{\hat{z}_{iz,k}}{\delta m_{iy,k-1}} & \frac{\hat{z}_{iz,k}}{\delta m_{iz,k-1}} \end{array} \right]$$

Note that the actual matrices are not shown as they are exceedingly large and that  $3 \times i$  zeros must be inserted at the | position when when landmark  $i \in [0, (N-1)]$  is observed. Each time a landmark is observed the predicted observation for the landmark is computed, the corresponding Jacobian is updated based on the current state estimate and these two results are directly substituted into Equations 2.7 and 2.8 to perform the mean and covariance observation update. The additive, zero mean uncorrelated Gaussian errors  $o_k$  which affect the observations and have covariance  $R_k$  are specified in the configuration file and must be tuned according to the environment. When a new landmark is observed the state and covariance of the EKF is expanded using the augmented state approach, which requires the implementation of an inverse landmark observation model. In this model the estimated position of a new landmark  $m_j$  is given by a function  $g(\hat{v}_{k|k-1}, z_k)$  which is essentially the inverse of  $h(\hat{v}_{k|k-1}, \hat{m}_{i,k-1})$ :

$$\hat{m}_{j,k} = g(\hat{v}_{k|k-1}, z_k) = {}^{world}T_{base} \begin{bmatrix} z_k \\ 1 \end{bmatrix} \quad (3.11)$$

and the addition of the landmark to the state vector is trivial:

$$\hat{x}_{k|k} = \begin{bmatrix} \hat{v}_{k|k-1} & \hat{m} & g(\hat{v}_{k|k-1}, z_k) \end{bmatrix}^T \quad (3.12)$$

Computing the Jacobian  $\nabla g$  of  $g()$  at  $\hat{v}_{k|k-1}$  the new landmark is added to the covariance matrix by computing:

$$P_{k|k-1} = \begin{bmatrix} P_{pp} & P_{pr} & P_{pm} & P_{pp}\nabla g^T \\ P_{pr}^T & P_{rr} & P_{rm} & P_{pr}\nabla g^T \\ P_{pm}^T & P_{rm}^T & P_{mm} & P_{pm}\nabla g^T \\ \nabla g P_{pp} & \nabla g P_{pr}^T & \nabla g P_{pm}^T & \nabla g P_{pp}\nabla g^T + R_k \end{bmatrix} \quad (3.13)$$

where  $P_{pp}$  and  $P_{rr}$  are the  $3 \times 3$  sub matrices of the covariance matrix which correspond to the x, y, z position, and roll, pitch, yaw angles respectively, and  $R_k$  is the covariance of the additive, zero mean uncorrelated Gaussian observation errors.

### 3.2.5 Simulation and Testing Environment

In order to evaluate the systems performance a C++/Matlab simulation and testing environment was implemented. A C++ XML configuration file parser was developed so that the calibration matrices, vectors, and system parameters could be stored in an intuitive format and tuned without recompiling. A scenario generator was then developed in Matlab based on an existing function for constructing a 3D surface from an arbitrary number of Gaussians with user specified variance as illustrated in Figure 3.3(a). The scenario generator then requests the user to introduce surface features either by placing sets of Gaussian distributed points with the mouse as illustrated in Figure 3.3(b), or by specifying the density at which features should be randomly distributed. Finally, the scenario generator prompts the user to define the vehicle trajectory by placing successive waypoints in the map and specifying the depth and roll at each point as illustrated in Figure 3.3(c). The vehicle position and pose is then computed at fixed intervals along the trajectory by linking the waypoints with cubic spline interpolations to obtain a smooth trajectory as illustrated in Figure 3.3(d). Finally, the trajectory, features and surface points are written to individual files in a space delimited array format.

In C++ an inherited class replaces the normal class which grabs or loads video frames, finds matches and performs triangulation as described in Section 3.2.1. This class reads and stores the trajectory, features and surface points from the files specified by the user and each time

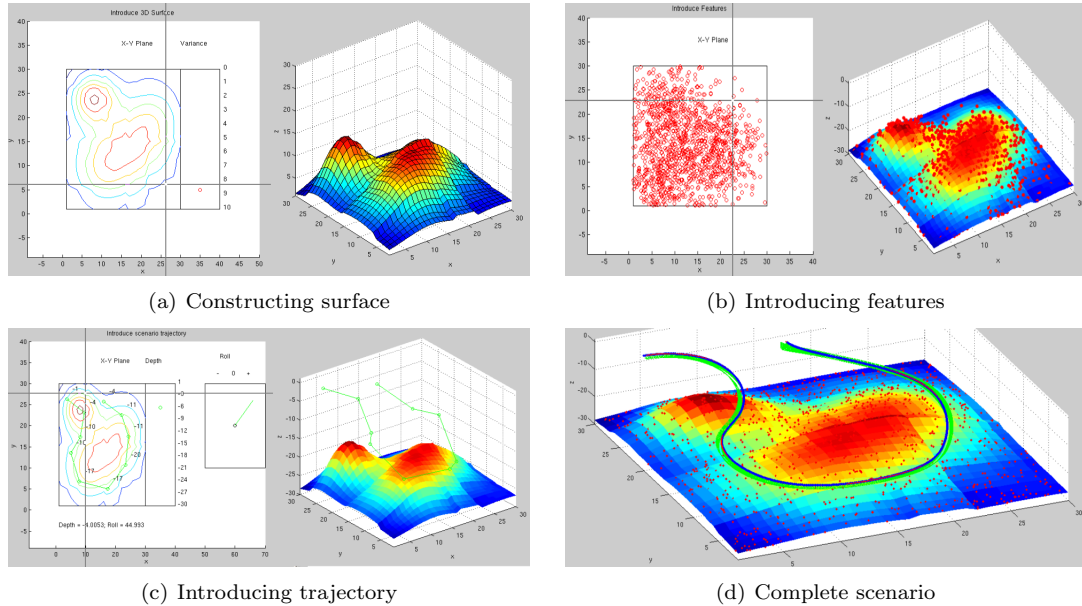


Figure 3.3: Scenario Generator

an observation is requested uses this data to generate synthetic measurements. This involves placing the cameras with respect to the map based on the ground truth trajectory and projecting features and surface points with positive depth into the image planes. Feature points which fall outside the image boundaries or are occluded by surface points (in one or both cameras) are then rejected. Finally, Gaussian noise of user specified variance is added to the remaining set of correspondence points and outliers are also added with user specified probability.

At present each synthetic features descriptor is a single integer, therefore the function for finding matches is also overloaded, however, all other system functionality remains unchanged. At the end of each iteration of the SLAM algorithm the state and diagonal of the covariance matrix is written to file. A Matlab script has been implemented to read this data and playback the simulation step by step, showing the position and pose of the vehicle, uncertainty in the vehicles position, position of landmarks, and uncertainty in the landmarks position. In addition, the script also provides a time series plot of the absolute error in the position and pose and computes the maximum absolute error in the roll, pitch and yaw angles and the mean squared error in the position to enable a quantitative assessment of the performance. The average and worst case execution time of key stages of the algorithm is also recorded, however, during development and testing the code was compiled without compiler optimisations and consequently achievable execution times are expected to be lower than stated.

## Chapter 4

# Results

The system was evaluated on a number of different scenarios in a  $30 \times 30 \times 30m$  simulated environment which has randomly distributed features with a uniform density of  $50/m^2$  with the exception of a  $5 \times 5m$  section which has no features. The results below are presented for a loop trajectory which is 87m long and contains sections of ascent, descent, maintained depth and roll turbulence. For all trials observations were performed at  $0.05m$  intervals.

In the first trial the EKF and UKF variants were tested in noiseless conditions with zero outliers. The constraints for accepting matches were quite relaxed with a 95% level of confidence for the fundamental matrix computation, a maximum deviation of 1.0 pixel between each point and its corresponding epipolar line before it is considered an outlier, at least 8 inliers required to accept an observation and a maximum LMS registration error of 0.35. The orientation and position process noises were 0.05 and 0.01 respectively and the landmark, orientation and position measurement noises were 0.01, 0.25 and 0.15 respectively. The estimated trajectory of the UKF and EKF system variants can be qualitatively compared in Figures 4.1(a) and 4.1(b) and the time series plots of absolute x, y and yaw error in Figures 4.1(c) and 4.1(d). The rapid growth in uncertainty in the error plots corresponds to the section of the trajectory in which there are no features, and consequently no observation updates. For the complete trajectory the mean squared position error for the EKF variant was 0.065 and the UKF variant was 0.466. For the EKF the maximum absolute error in the roll, pitch and yaw angles was  $1.28^\circ$ ,  $14.6^\circ$  and  $12.9^\circ$  respectively. For the UKF the maximum absolute error in the roll, pitch and yaw angles was  $4.30^\circ$ ,  $13.1^\circ$  and  $14.5^\circ$  respectively. The EKF execution time was 70.8 seconds for the 1739 observations, giving an average time of 40.7ms per observation, with a maximum observation time of 1029ms. In comparison, the UKF execution time was 98.4 seconds for the 1739 observations, giving an average time of 56.6ms per observation, with a maximum observation time of 983ms. Both the EKF and UKF added 11 landmarks to the

filter state while traversing the complete trajectory. Based on these results it appears that in the noiseless case the EKF slightly outperforms the UKF in terms of both estimation accuracy and computational load.

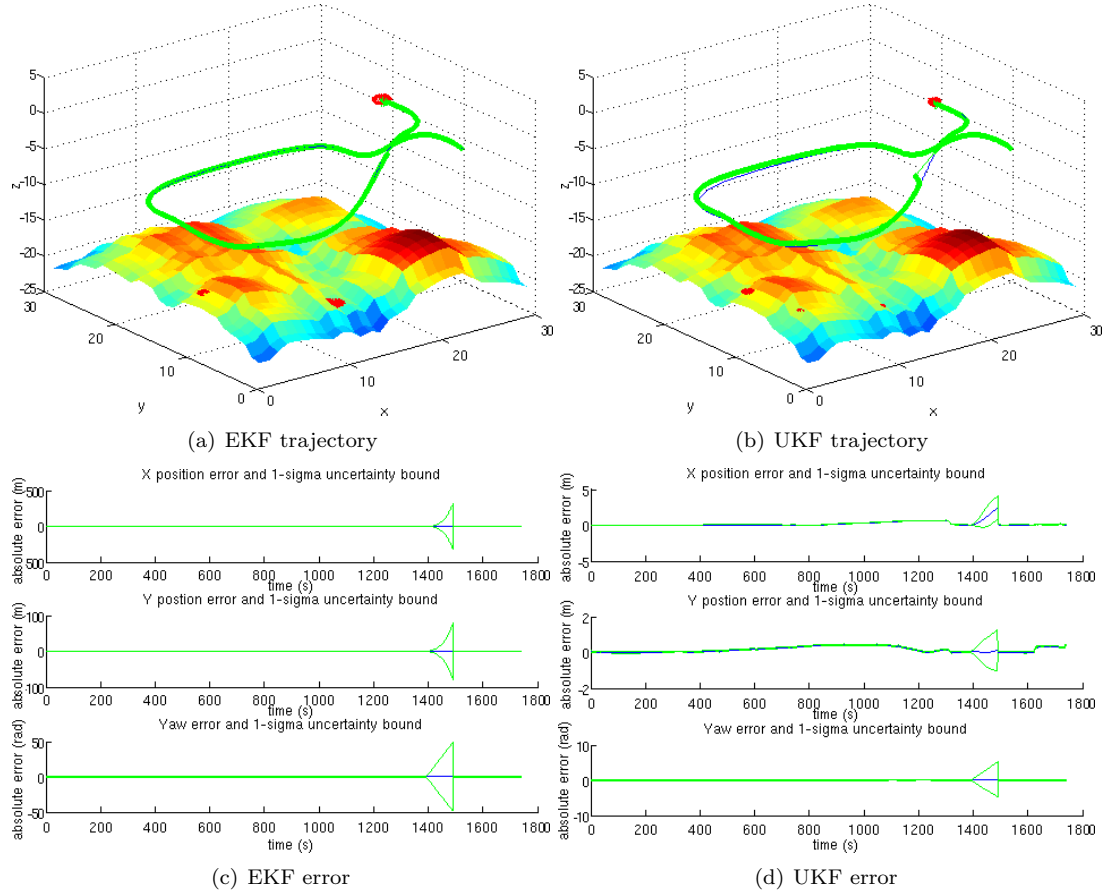


Figure 4.1: Comparison of EKF and UKF in noiseless conditions

In the second trial the EKF and UKF variants were tested when Gaussian noise with a standard deviation of 0.1 pixels is added to the corresponding points in the left and right image planes. In this trial the system parameters were unchanged from the values given above. The estimated trajectory of the UKF and EKF system variants can be qualitatively compared in Figures 4.2(a) and 4.2(b) and the time series plots of the absolute x, y and yaw error in Figures 4.2(c) and 4.2(d). In this case, the mean squared position error for the EKF and UKF were 1.049 and 1.640 respectively. In this trial, the maximum absolute error in the roll, pitch and yaw angles for the EKF was  $7.38^\circ$ ,  $13.9^\circ$  and  $8.85^\circ$  respectively. The maximum absolute error

in the roll, pitch and yaw angles for the UKF were similar at  $7.73^\circ$ ,  $13.2^\circ$  and  $9.23^\circ$  respectively. The EKF execution time was 75.2 seconds for the 1739 observations, giving an average time of 43.2ms per observation, with a maximum observation time of 1002ms. In comparison, the UKF execution time was 111.9 seconds for the 1739 observations, giving an average time of 64.3ms per observation, with a maximum observation time of 1100ms. The EKF utilised 11 landmarks while traversing the complete trajectory, whereas the UKF utilised 12. Similarly to the noiseless case these results suggest that the EKF is slightly better than the UKF in terms of robustness, estimation accuracy, computational load and the number of landmarks required.

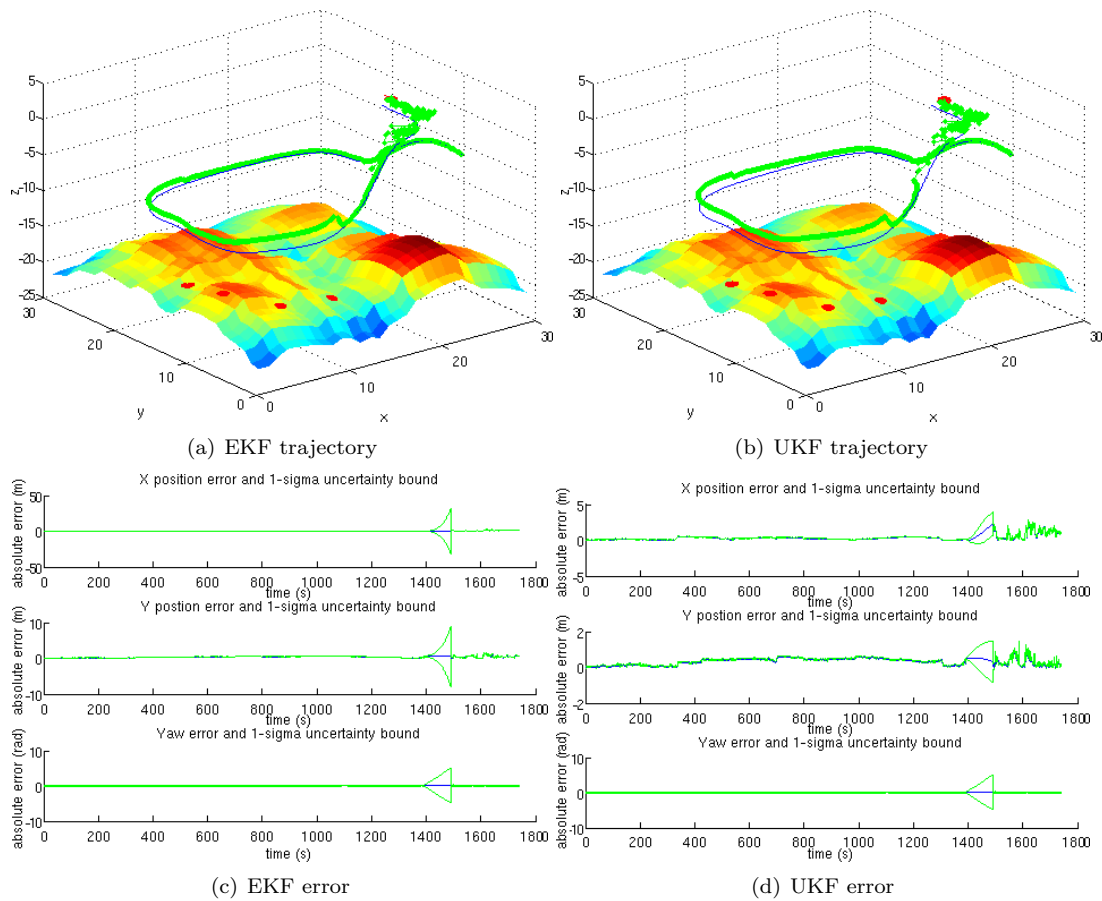


Figure 4.2: Comparison of EKF and UKF in presence of Gaussian noise ( $\sigma = 0.1$ )

The standard deviation of the Gaussian noise added to the corresponding points was then increased by 0.025 in each subsequent trial. For the 0.2 pixel standard deviation case the best results were obtained by relaxing the constraints for accepting matches to allow a maximum LMS

registration error of 0.7. This ensures observations are accepted frequently enough to detect changes in pitch and yaw, otherwise the error introduced in the prediction step accumulates rapidly and the filter diverges. In addition, the landmark, orientation and position measurement noises were increased to 0.1, 0.35 and 0.25 respectively. The estimated trajectory of the UKF and EKF system variants can be qualitatively compared in Figures 4.3(a) and 4.3(b) and the time series plots of the absolute x, y and yaw error in Figures 4.3(c) and 4.3(d).

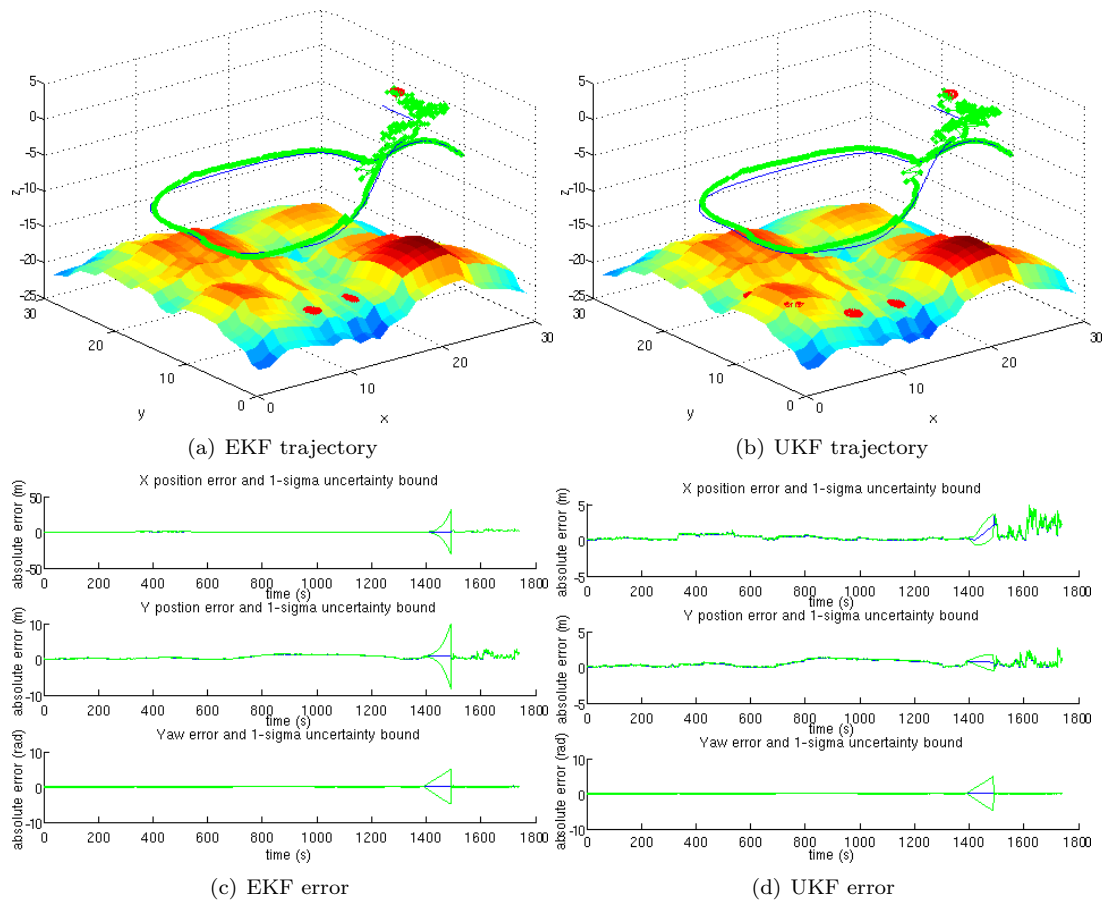


Figure 4.3: Comparison of EKF and UKF in presence of Gaussian noise ( $\sigma = 0.2$ )

For the complete trajectory the mean squared position error for the EKF was 1.90 and the maximum absolute error in the roll, pitch and yaw angles was  $27.4^\circ$ ,  $16.0^\circ$  and  $14.9^\circ$  respectively. In comparison the mean squared position error for the UKF was 2.22 and the maximum absolute error in the roll, pitch and yaw angles was  $26.9^\circ$ ,  $15.2^\circ$  and  $12.8^\circ$  respectively. The EKF execution time was 79.4 seconds for the 1739 observations, giving an average time of 45.7ms

per observation, with a maximum observation time of 1012ms. The UKF execution time was 118.0 seconds for the 1739 observations, giving an average time of 67.8ms per observation, with a maximum observation time of 1071ms. While traversing the loop the EKF added 12 landmarks to the filter whereas the UKF added 13. This result was representative of the full set of trials with respect to the fact that as the level of noise was increased the EKF position error remained marginally better than the UKF, however, the UKF absolute angular error was typically lower than the EKF. Furthermore, across the full set of trials the EKF retained its speed advantage over the UKF and on average the execution time was 36% lower. The failure point of the EKF and UKF was detected by continually increasing the noise level until the maximum absolute position error exceeded 7% of the distance travelled or the maximum absolute angular error for the roll, pitch or yaw exceeded  $30^\circ$ . Based on these conditions the EKF and UKF failed when the noise standard deviation reached 0.25 pixels and 0.225 pixels respectively.

Finally, the EKF and UKF variants were tested for their robustness to outliers. It was hoped that the outliers introduced would be eliminated by the countermeasures described in Section 3.2.2 and consequently the optimal system parameters would be close to those of the trial with the equivalent noise but no outliers. This theory was experimentally confirmed by examining system performance for a large range of system parameters when the standard deviation of the Gaussian noise added to the corresponding points was fixed at 0.1 and 0.15 pixels and the percentage of outliers was increased by 2.5% in each subsequent trial. However, as expected the introduction of outliers had a considerable impact on system performance due to the fact that the frequency of motion observations was drastically reduced. Based on the failure conditions stated earlier the EKF was able to cope with up to 12.5% outliers. In contrast, the UKF proved completely unrobust to outliers as the covariance matrix become ill-conditioned with just 2.5% outliers regardless of the system parameter values. Figures 4.4(a) and 4.4(b) illustrate the estimated trajectory of the EKF with 5% and 10% outliers. The time series plots of the absolute x, y and yaw error for these trajectories are provided in Figures 4.4(c) and 4.4(d). For the 5% outliers case the mean squared position error was 2.56 and the maximum absolute error in roll, pitch and yaw angles was  $9.69^\circ$ ,  $13.4^\circ$  and  $21.0^\circ$  respectively. A total of 32 landmarks were used and the execution time was 366.7 seconds for the 1739 observations, giving an average time of 210.9ms per observation, with a maximum observation time of 1251ms. For the 10% outliers case the mean squared position error was 3.16 and the maximum absolute error in roll, pitch and yaw angles was  $11.6^\circ$ ,  $24.3^\circ$  and  $24.4^\circ$  respectively. A total of 42 landmarks were used and the execution time was 467.9 seconds for the 1739 observations, giving an average time of 269.1ms per observation, with a maximum observation time of 1321ms.

From all the trials it is evident that the EKF outperforms the UKF with respect to localisation accuracy, the number of landmarks required, execution time and stability. It was also observed that as the level of noise was increased the maximum observation time fluctuated,

but did not increase as expected. Further analysis showed that the maximum observation time always occurred following the addition of selected landmarks and appears to be linked to cases in which the covariance matrix and/or state vector cannot be expanded without copying the entire array to a different section of memory.

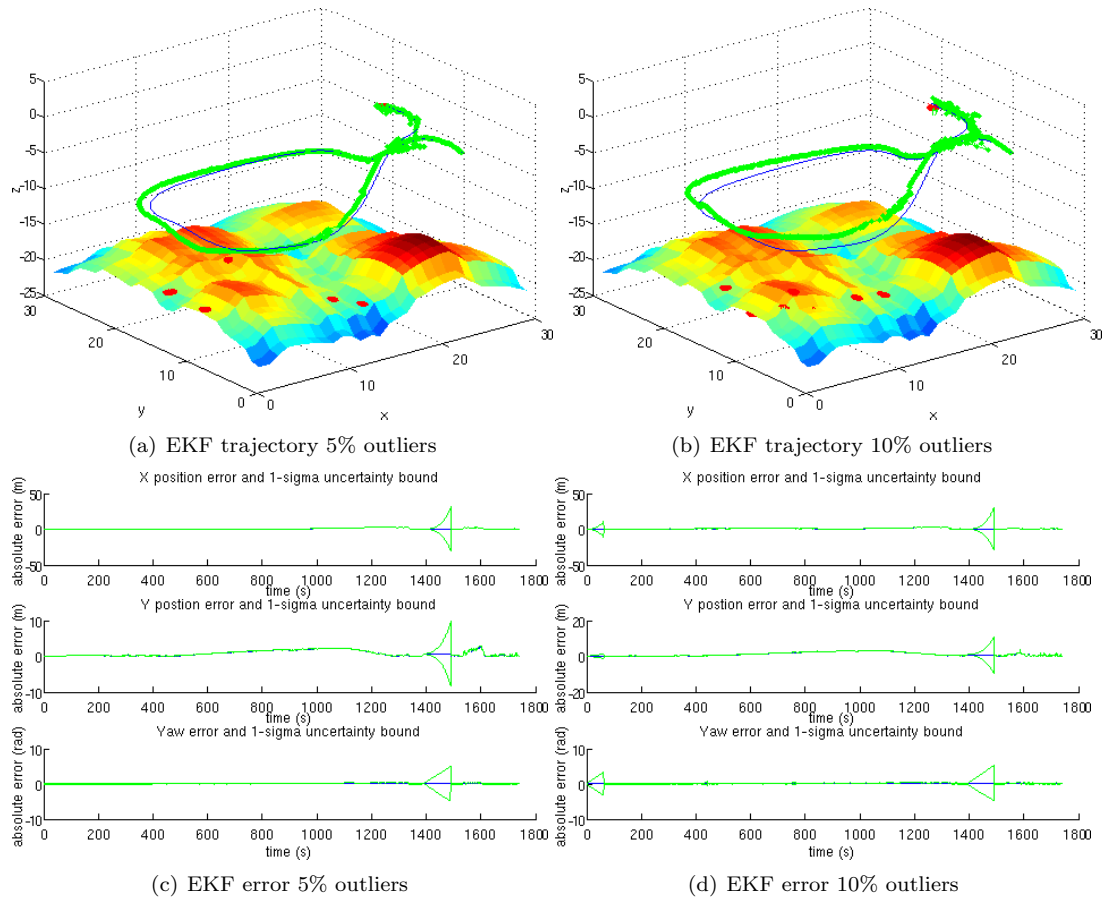


Figure 4.4: Robustness of EKF to outliers

Consequently, a reduction in execution time may be possible if sufficient space is pre-allocated based on an estimate of the number of landmarks required. To predict the execution time of the EKF-SLAM system in real scenarios the number of features extracted in each iteration was recorded and an estimate was computed based on the average values described in Section 3.2.1. For the scenario with 0.1 pixel standard deviation Gaussian noise and 10% outliers 171440 features were extracted during the 1739 observations giving a total execution time of  $467.9 + 311.1 = 779$  seconds for SURF and  $467.9 + 506.6 = 974.5$  seconds for SIFT. Based on the

average time per observation this gives an update frequency of 2.23Hz for SURF and 1.78Hz for SIFT. Therefore, without further optimisation real-time operation will only be possible if the rotation and translation velocities of the vehicle are limited. However, the primary bottleneck of the system is the feature extraction stage, therefore by tuning the SIFT and SURF parameters it may be possible to obtain a more acceptable execution time by utilising a minimal number of features.

From these trials it is observed that as the level of noise is increased the LMS error threshold must also be increased in order to perform observation updates sufficiently often. However, as the LMS error threshold is increased it is observed that while the average error in the position and angles is reduced (due to the increased frequency of observations), the maximum absolute error increases considerably as a result of infrequent but highly inaccurate observations. In addition, it was noted that while the relative motion estimates computed by registering the observation and landmark sub-maps were accurate for small changes in viewpoint, when the change in viewpoint was larger the error in the translation increased slightly and the error in the angles increased moderately. Therefore, a considerable increase in performance may be possible by improving the method for estimating the relative motion. One potential solution that would also improve computational efficiency would be to extract the relative motion from the essential matrix (Equation 3.1).

This SLAM implementation has a significant advantage over most SLAM implementations as it observes the vehicle pose in the global frame using landmark based visual odometry, which means localisation uncertainty does not grow during exploration. However, from the error plots it is clear that the current implementation maintains over-confident estimates due to the fact that the visual odometry is not continuous. It is observed that when visual odometry is not available error accumulates in the state and the uncertainty grows as expected. However, when the visual odometry resumes it is based on this erroneous state estimate and consequently the uncertainty immediately decreases to the uncertainty associated with a motion observation. Therefore, despite the fact that the inter-frame motion estimation is fairly accurate, the error accumulated during each 'blackout' period generally causes the true vehicle position to fall outside the  $3\sigma$  uncertainty bounds. The severity of this problem could be reduced by forcing landmark submaps to overlap, which would eliminate blackout periods while switching between adjacent landmarks. However, to resolve this problem completely, following a blackout visual odometry should only be resumed when the vehicle re-observes a landmark for which the odometry is known, which would require an active approach to navigation to work successfully.

## Chapter 5

# Conclusions

A novel EKF-SLAM system has been implemented which should be capable of operating in real-time on a slow moving AUV equipped with a calibrated stereo system. The map constructed by the system contains a set of sparsely distributed landmarks which are described by a set of observed 3D points and their associated SIFT/SURF descriptors. By utilising such distinctive landmark descriptions the system is able to employ maximum likelihood data association based on the 3D points and 2D descriptors with little risk of incorrectly associating landmarks. To identify stereo and landmark correspondences the system compares the 2D descriptors and attempts to eliminate outliers by applying epipolar constraints. For the landmark matching the fundamental matrix is unknown and must be robustly estimated using the 8-point and least-median squares methods, and the set of 3D point correspondences are also registered to measure their compatibility and determine the relative rotation and translation between the first and current observation of the landmark. These relative motion estimates are then used to perform landmark-based visual odometry. The EKF incorporates a linear-time augmented-state constant-velocity prediction model, a non-linear landmark observation model and a linear motion observation model. Simulated trials demonstrated that when no outliers are present the EKF localisation error is acceptable ( $< 7\%$  distance travelled) when Gaussian noise with  $\sigma \leq 0.25$  pixels is added to the stereo correspondences. In the presence of Gaussian noise with  $\sigma = 0.1$  pixels the EKF was able to operate with up to 12.5% outliers. For an 87m long simulated loop trajectory which contains sections of ascent, descent, maintained depth, and roll turbulence, the mean squared position error was 3.16 and the maximum absolute error in roll, pitch and yaw angles was  $11.6^\circ$ ,  $24.3^\circ$  and  $24.4^\circ$  respectively for the  $\sigma = 0.1$  pixels and 10% outliers case. To reduce the computational load and improve the localisation accuracy of the system further work is required to accelerate feature extraction, discard weak features, improve the robustness of the relative motion estimation and ensure the visual odometry is continuous.

# Bibliography

- [1] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: Part 1,” *IEEE Robotics and Automation Magazine*, pp. 108–117, September 2006.
- [2] X. Armangué, H. Araujo, and J. Salvi, “A review on egomotion by means of differential epipolar geometry applied to the movement of a mobile robot,” *Pattern Recognition*, vol. 36, pp. 2927–2944, December 2003.
- [3] R. Eustice, H. Singh, and J. Leonard, “Exactly sparse filters,” in *IEEE Int. Conf. Robotics Automation*, 2005, pp. 2417–2424.
- [4] G. Dissanayake, P. Newman, H. Durrant-Whyte, S. Clark, , and M. Csobor, “A solution to the simultaneous localisation and mapping (slam) problem,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.
- [5] K. L. Ho and P. Newman, “Detecting loop closure with scene sequences,” *International journal of computer vision*, vol. 74, no. 3, pp. 261–286, September 2007.
- [6] M. Euston and J. Kim, “Rao-blackwellised inertial-slam with partitioned vehicle subspace,” in *ACRA07*, 2007.
- [7] Z. Chen, J. Samarabandu, and R. Rodrigo, “Recent advances in simultaneous localization and map-building using computer vision,” *Advanced Robotics*, vol. 21, pp. 233–265, 2007.
- [8] S. Lacroix, A. Mallet, I.-K. Jung, T. Lemaire, and J. Sola, “Vision-based slam,” *SLAM Summer School 2006, Oxford*, 2006.
- [9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.
- [10] A. J. Davison and D. W. Murray, “Simultaneous localization and map-building using active vision,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 865–880, July 2002.

- 
- [11] J. N. J. Castellanos and J. Tardos, "Multisensor fusion for simultaneous localization and map building," in *IEEE Trans. Robotics Automat.*, vol. 17, 2001, p. 908914.
- [12] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part 2," *IEEE Robotics and Automation Magazine*, pp. 108–117, September 2006.
- [13] K. Chong and L. Kleeman, "Feature-based mapping in real, large scale environments using an ultrasonic array," *Int. J. Robot. Res.*, vol. 18, no. 1, pp. 3–19, 1999.
- [14] S. Thrun, Y. Liu, D. Koller, A. Ng, and H. Durrant-Whyte, "Simultaneous localisation and mapping with sparse extended information filters," *Int. J. Robot. Res.*, vol. 23, no. 7-8, pp. 693–716, 2004.
- [15] F. Dellaert, "Square root sam: Simultaneous location and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.
- [16] J. Folkesson and H. Christensen, "Graphical slam a selfcorrecting map," in *IEEE Int. Conf. Robotics Automation*, 2004, p. 791798.
- [17] S. Julier and J. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, 1997.
- [18] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fast-slam: A factored solution to the SLAM problem," in *AAAI Nat. Conf. Artif. Intell.*, 2002, p. 593598.
- [19] —, "Fast-slam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Int. Joint Conf. Artif. Intell.*, 2003, p. 11511156.
- [20] J. Porta, J. Verbeek, and B. Krose, "Active appearance-based robot localization using stereo vision," *Autonomous Robots*, vol. 18, p. 5980, 2005.
- [21] I. Mahon and S. Williams, "Slam using natural features in an underwater environment," in *IEEE Control, Automation, Robotics and Vision Conference*, vol. 3, December 2004.
- [22] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.
- [23] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," in *IEEE transactions on robotics*, vol. 21, no. 3, June 2005, pp. 364–375.
- [24] D. Schleicher, L. M. Bergasa, M. Ocaa, R. Barea, and E. Lpez, "Real-time stereo visual slam in large-scale environments based on sift fingerprints," *LNCS 4739*, pp. 648–691, 2007.

- 
- [25] G. Bleser, M. Becker, and D. Stricker, “Real-time vision-based tracking and reconstruction,” *Journal of real-time image processing*, vol. 2, pp. 161–175, 2007.
- [26] T. Z. er, C. G. l, and H. Niemann, “Efficient feature tracking for long video sequences,” in *Symposium der Deutschen Arbeitsgemeinschaft fr Mustererkennung (DAGM)*, 2004.
- [27] J. Montiel, J. Civera, and A. J. Davison, “Unified inverse depth parametrization for monocular slam,” in *Robotics: Science and Systems*, 2006.
- [28] O. M. Mozos, A. Gil, M. Ballesta, and O. Reinoso, “Interest point detectors for visual slam,” *LNAI 4788*, pp. 170–179, 2007.
- [29] J. Neira, J. D. Tardos, and J. A. Castellanos, “Linear time vehicle relocation in slam,” in *IEEE International Conference on Robotics and Automation*, September 2003.
- [30] D. Hahnel, W. Burgard, B. Wegbreit, and S. Thrun, “Towards lazy data association in slam,” in *11th Int. Symp. of Robotics Research*, 2003, p. 421431.
- [31] J. Neira and J. D. Tardos, “Data association in stochastic mapping using the joint compatibility test,” in *IEEE Trans. Robotics Automat.*, vol. 17, 2001, pp. 890–897.
- [32] H. Adams, S. Singh, and D. Strelow, “An empirical comparison of methods for image-based motion estimation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2002, pp. 123–128.
- [33] S. Fleischer, “Bounded-error vision-based navigation of autonomous underwater vehicles,” Ph.D. dissertation, Stanford University, 2000.
- [34] S. Negahdaripour and X. Xun, “Mosaic-based positioning and improved motion estimation methods for automatic navigation of submersible vehicles,” *IEEE Journal of Oceanic Engineering*, vol. 27, no. 1, pp. 79–99, January 2002.
- [35] R. Garcia, J. Battle, and X. Cufi, “Positioning an underwater vehicle through image mosaicking,” in *IEEE International Conference on Robotics and Automation*, 2001.
- [36] J. Lots, D. Lane, E. Trucco, and F. Chaumette, “2-D visual servoing for underwater vehicle station keeping,” in *IEEE International Conference on Robotics and Automation*, 2001.
- [37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.
- [38] K. Kanatani, “Unbiased estimation and statistical analysis of 3-d rigid form two views,” in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 1, 1993, pp. 37–50.

- 
- [39] A. Dell'Acqua, A. Sarti, and S. Tubaro, "3d motion from structures of points, lines and planes," *Image and Vision Computing*, vol. 26, p. 529549, 2008.
- [40] A. Azarbayejani and A. Pentland, "Recursive estimation of motion, structure, and focal length," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 6, 1995, pp. 562–575.
- [41] —, "3d structure from 2d motion," *IEEE Signal Processing Magazine*, vol. 16, no. 3, pp. 66–84, 1998.
- [42] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "3-d motion and structure causally integrated over time: theory (stability) and practice (occlusions)," ESSRL Technical Report 99-003, Tech. Rep., October 1999.
- [43] —, "3-d motion and structure from 2-d motion causally integrated over time: implementation," in *European Conference on Computer Vision, Lecture Notes in Computer Science*, vol. 1842, 2000, pp. 735–750.
- [44] —, "Structure from motion causally integrated over time," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, 2002, pp. 509–522.
- [45] J. Kim and M. Chung, "Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases," *Pattern Recognition*, vol. 39, no. 9, pp. 1649–1661, September 2006.
- [46] J. Kim and S. Sukkarieh, "Real-time implementation of airborne inertial-slam," *Robotics and Autonomous Systems*, vol. 55, pp. 62–71, 2007.
- [47] —, "6dof slam aided gnss/ins navigation in gnss denied and unknown environments," *Journal of Global Positioning Systems*, vol. 4, no. 1-2, pp. 120–128, 2005.
- [48] J. M. Saez, A. Hogue, F. Escolano, and M. Jenkin, "Underwater 3d slam through entropy minimization," in *IEEE International Conference on Robotics and Automation*, May 2006, pp. 3562–3567.
- [49] R. Eustice, "Large-area visually augmented navigation for autonomous underwater vehicles," Ph.D. dissertation, June 2005.
- [50] O. Pizarro, R. Eustice, and H. Singh, "Large area 3d reconstructions from underwater surveys," in *OCEANS 2004 MTS/IEEE Conference and Exhibition*, vol. 2, November 2004, pp. 678–687.

- 
- [51] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard, "Visually navigating the RMS titanic with SLAM information filters," in *Proceedings of Robotics: Science and Systems (RSS)*, June 2005.
- [52] I.-K. Jung and S. Lacroix, "High resolution terrain mapping using low altitude aerial stereo imagery," in *IEEE International Conference on Computer Vision*, 2003.
- [53] J. Langelaan and S. Rock, "Passive gps-free navigation for small uavs," in *IEEE Aerospace Conference*, March 2005, pp. 1–9.
- [54] J. Kim and S. Sukkarieh, "Autonomous airborne navigation in unknown terrain environments," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, no. 3, 2004.
- [55] M. Bryson and S. Sukkarieh, "Active airborne localisation and exploration in unknown environments using inertial slam," in *IEEE Aerospace Conference*, March 2006, pp. 13–.
- [56] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Int. Conf. on Computer Vision*, 2003, pp. 1403–1416.
- [57] A. Davison, Y. G. Cid, and N. Kita, "Real-time 3-d slam with wide-angle vision," in *IFAC Symp. on Intelligent Autonomous Vehicles*, 2004.
- [58] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, June 2007.
- [59] J. Civera, A. Davison, and J. Montiel, "Inverse depth to depth conversion for monocular slam," in *IEEE International Conference on Robotics and Automation*, April 2007.
- [60] —, "Dimensionless monocular slam," in *Iberian Conference of Pattern Recognition and Image Analysis*, June 2007.
- [61] D. Burschka and G. D. Hager, "V-gps(slam): Vision-based inertial system for mobile robots," in *IEEE International Conference on Robotics and Automation*, April 2004.
- [62] P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardos, "Inertial aiding of inverse depth slam using a monocular camera," in *IEEE International Conference on Robotics and Automation*, April 2007, pp. 2797–2802.
- [63] R. I. Hartley, "In defense of the eight-point algorithm," in *IEEE Transaction on Pattern Recognition and Machine Intelligence*, vol. 19, no. 6, June 1997, pp. 580–593.