SensorSLAM an investigation into sensor parameter estimation for SLAM

Andrew Hogue

Technical Report CSE-2008-11

December 19, 2008

Department of Computer Science and Engineering

4700 Keele Street Toronto, Ontario M3J 1P3 Canada

SENSORSLAM: AN INVESTIGATION INTO SENSOR PARAMETER ESTIMATION FOR SLAM

ANDREW HOGUE

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO
DECEMBER, 2008

SENSORSLAM: AN INVESTIGATION INTO SENSOR PARAMETER ESTIMATION FOR SLAM

by **Andrew Hogue**

a dissertation submitted to the Faculty of Graduate Studies of York University in partial fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

SENSORSLAM: AN INVESTIGATION INTO SENSOR PARAMETER ESTIMATION FOR SLAM

by **Andrew Hogue**

By virtue of submitting this document electronically, the author certifies that this is a true electronic equivalent of the copy of the dissertation approved by York University for the award of the degree. No alteration of the content has occurred and if there are any minor variations in formatting, they are as a result of the coversion to Adobe Acrobat format (or similar software application).

Examination Committee Members:

1. Dr. M. Jenkin
2. Dr. R. S. Allison
3. Dr. R. Wildes
4. Dr. M. Spetsakis
5. Dr. S. Negahdaripour
6. Dr. R. Lee

Abstract

Mapping technology is an essential component of autonomous robotic systems. The ability of a vehicle to establish its location with respect to some environmental representation allows the vehicle to navigate and reason about the environment. This process of estimating vehicle motion while fusing sensor data to acquire a map of the environment is called simultaneous localization and mapping or SLAM. Existing SLAM formulations are unable to effectively address the problem of non-stationary sensor noise in the sensor model. Yet orientation or location dependent noise sources are commonplace in real-world scenarios. The use of standard noise models (which assume stationary noise) leads to instability or reduced accuracy of the resulting map within traditional SLAM solutions. By parameterizing the non-stationary aspects of the noise model and estimating these parameters simultaneously with the map and sensor location, a stochastic formulation for SLAM is developed.

The general Bayesian framework developed in this dissertation is applicable in many domains, however this work focuses on the underwater environment. Stereo video cameras and inertial sensors are utilized to solve SLAM in the underwater environment. The resulting algorithm is used to recover models of complex underwater structures. Incorporating a non-stationary noise model within a Bayesian SLAM formulation reduces the error in the resulting trajectory and map by simultaneously estimating the location-based (non-stationary) noise in the environment, the trajectory of the sensor, and an accurate map of the environment.

To Ula and Suzy

Acknowledgements

There are so many people that have contributed to the completion of my doctoral research and dissertation. Dr. Michael Jenkin has provided me with this amazing opportunity and I am indebted to him for my entire research career. He has been my guide throughout my graduate experience and has provided me with many opportunities for travel, discussion, freedom to explore my own research interests, and a multitude of funding for which I am extremely grateful. The entire York AQUA team was essential for the completion of my degree. Andrew German, Jim Zacher, Robert Codd, Heather Jenkin, and Jeff Laurence have all been integral in the design of the AQUASensor and during the data collection process. The lack of ability to keep my head underwater without hyperventilating was compensated by all through their tireless diving expeditions. Thank you to Jim, who continued to give his input and provide feedback on hardware designs and who wouldn't give up even after the leaky incidents. Andrew has been especially helpful during the software development of the sensor. His level-headedness, system development and programming skills, as well as his constant vigil of the Mac rumour sites kept me sane throughout the process. I would also like to thank my parents Carolee and René, who never gave up hope that I would one day finish school.

Urszula, the love of my life, my wife, my friend. Thank you for being the one constant in my life. You have helped me more than you can ever imagine. I could not have done this without you. You kept me on track even at times where I was stuck and thought I would quit. The support you have given me has been invaluable. Lastly, I have to mention Suzanne. Without you, my life would not have meaning. Thank you for smiling.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Mapping technology is an essential component of autonomous robotic systems. If a map of the environment is known prior to vehicle deployment, it can be used to represent and accurately estimate the position of the vehicle. The ability of a vehicle to establish its location with respect to its environmental representation enables autonomous navigation and real-time reasoning. In an environment where things do not move or change state it may be possible to use an *a priori* known map. However, the reality is that the positions of objects change, many environments are dynamic and a static representation is rapidly invalidated. In other situations, such as exploration of new environments, it is not possible to have a map on hand prior to deployment. Thus, a critical task for autonomous vehicles is the development of technologies and sensors to enable a vehicle to explore the environment and create a map representation suitable for self-navigation. Maps aid in path planning, in the assessment of possible obstacles, and enable a wide range of autonomous abilities. Autonomous vehicles may operate in a multitude of environments such as outer space, underwater, or on land. Applications for autonomous vehicles include navigation (e.g. DARPA Grand Challenge, autonomous military convoys), inspection (e.g. oil-pipelines, satellites, ship-hulls), and exploration of new environments (e.g. the Mars rover). Often, the goal of robotic exploration is to estimate a map of the environment to be used for measurement purposes or to be used at a later date for localization.

Mapping is accomplished through the use of sensors attached to the robot. For wheeled robots, odometry provides an estimate of the vehicle's motion relative to some previous point in time. Using a series of relative transformations, measurement data can be transformed into a world coordinate frame. Continually updating both the vehicle position and estimates of feature locations in the environment into a common frame of reference allows the robot to maintain its trajectory

while simultaneously estimating the locations of the features in the environment (i.e. the map). This process of simultaneous self-localization and map-building is commonly referred to as *Simultaneous Localization and Mapping* or SLAM (see [Thrun et al., 2005]).

There are many realities of the SLAM problem that make it a difficult problem to solve. Since sensors measure physical quantities, unknown factors in the environment degrade sensor performance. In existing algorithms these error sources are typically assumed to be known, negligible, or stationary such that a calibration phase can be performed prior to deployment to fully characterize the sensor error and performance. Pre-calibration phases use a mathematical model of sensor performance in the presence of these error sources to properly characterize how the error changes in specific situations. Pre-calibration phases are not always suitable (and may even be impossible) for complex environments. Dynamic sources of error cannot be modelled effectively through pre-calibration and noise sources are not necessarily position independent (i.e. some errors are dependent on how you sense the environment, where the vehicle is relative to the object, etc.). Un-modelled dynamic errors affect the quality of the map (since they are constructed through sensor data fusion) and in some cases may lead to algorithmic inconsistency.

This dissertation explores methods of automatic three-dimensional (3D) map creation suitable for autonomous vehicle operation. In particular, the focus of this work is on developing algorithms that permit more general sensor models than assumed in existing approaches. Methods for integrating sensor data are utilized to develop a new framework for simultaneous localization and mapping of a visual sensor-based robot moving in an unknown six-degree-of-freedom environment. Although the primary application domain for this work is the underwater realm, experiments demonstrate the approach is capable of addressing the SLAM problem for a wide range of environments.

1.0.1 Notational Conventions

Throughout this dissertation, subscripts such as s_t denote a particular value of a random variable at time t where superscripts such as s^t denote a set of values.

Vehicle/sensor locations/pose are always denoted by s_t, observations such as range or bearing to landmarks are always denoted as $\{z_1, z_2, \ldots, z_t\} = z^t$. Control inputs, such as odometry, are always denoted as u^t and the map is denoted as Θ. Note that a time-varying map would be denoted as Θ_t and the set of all maps over all time would be denoted as Θ^t. Probabilistic density functions are always considered to be continuous unless otherwise stated and are denoted as $p(x)$ where x is a random variable. The notation $\hat{x}$ denotes an estimated value not the true value of the variable. When discussing sampling approaches to estimation of a probability density function, superscripts such as $x^{(j)}$ are used to denote the value of the j-th particle in the distribution or particle set. When discussing Kalman filtering and recursive estimation, the predicted state at time k is denoted as $\hat{x}_k^-$ and the updated posterior estimate is denoted as $\hat{x}_k^+$.

1.1 SLAM

Smith, Self, and Cheeseman [Smith and Cheeseman, 1986, Smith et al., 1990] were among the first researchers to explore simultaneous self-localization and the estimation of spatial relationships of landmarks. They formulated the problem in a stochastic manner to accommodate for uncertainty in the system. They used a probabilistic representation to model uncertainty in the sensor data; measurements are biased by noise, the sensor has limited resolution, and environmental factors play a role in the accuracy of the sensor data. A stochastic formulation enables the fusion of uncertainty from multiple error sources increasing landmark position accuracy. The basic approach (as described in [Smith and Cheeseman, 1986]) is to estimate the first two moments (mean and covariance) of the joint probability distribution of the robot pose and landmarks in the map.

The SLAM problem can be formally defined — as in [Durrant-Whyte et al., 2003] — in terms of estimating a joint probability distribution over the robot pose s_t and the map feature locations $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$,

$$p(s_t, \Theta).$$

Information about the scene is given in the form of control inputs $u^t =$

$\{u_1, u_2, \ldots, u_t\}$, sensor measurements $z^t = \{z_1, z_2, \ldots, z_t\}$, and data associations (the mapping between each measurement and map feature) $n^t = \{n_1, n_2, \ldots, n_t\}$. Using this data, the joint probability can be described through a conditional relation

$$p(s_t, \Theta | z^t, u^t, n^t).$$

Traditionally, range and bearing to landmarks are provided as measurements to solve the SLAM problem. The control inputs are provided in terms of vehicle odometry; the data associations — typically unknown — are estimated through a separate optimization process [Cox, 1993, Hahnel et al., 2005]. Estimating the joint probability maintains estimates of the landmark locations (mapping) and the vehicle pose (localization) simultaneously. By applying appropriate definitions of conditional probability and incorporating other assumptions — perhaps most importantly the Markov chain assumption — a recursive estimation process is realized. Smith and Cheesman [Smith and Cheeseman, 1986] originally formulated this estimation process using a Kalman filter [Kalman, 1960] framework. In a Kalman filter framework, the algorithm functions by updating a state estimate by optimally weighting the error between the received measurement and what was expected given estimates of the current state. Solving SLAM with a Kalman filter imposes several assumptions on the pdfs and the motion model. The underlying probability distribution is assumed to be a Gaussian, i.e. fully modelled by the mean and covariance estimate of the distribution and both the vehicle dynamic and measurement models are assumed to be described by a linear process. Violating these assumptions can lead to a divergence of the filter, or more commonly, inaccurate estimates resulting from either over or underestimating the state variables. If the measurement and noise processes are realistically modeled, the sensor data can be corrected by compensating for the appropriate amount of error currently affecting the measurement and the state is updated with higher accuracy. If the measurement includes errors that are un-modeled (e.g. the error violates the Kalman assumptions) then the state will be over or under-estimated resulting in a system that has a high amount of uncertainty. Such a situation leads to an inaccurate robot trajectory and uncertain map.

A number of extensions to the Kalman filter algorithm have been developed to address non-linear systems including the extended Kalman filter (EKF, see [Welch and Bishop, 1995]) which estimates non-linearity in the motion and measurement models by estimating the first-order terms of the Taylor series of these processes. The unscented Kalman filter (UKF, see [Julier and Uhlmann, 1997]) addresses the non-linearity by directly using the non-linear models within the framework and re-estimating the mean and covariances through a sampling approach. Other sampling-based representations (commonly known as Monte Carlo estimators or Particle filters, see [Gordon et al., 1993, Doucet et al., 2001, Ristic et al., 2004]) have been developed to address non-linear and non-Gaussian processes at the expense of computational power.

1.1.1 Limitations of current approaches

Current approaches to SLAM can be efficient and reliable in the application domain for which they are intended. They do, however, have limitations that often preclude their direct application to more generalized environments (such as the full 6DOF underwater realm). These limitations include

- The use of an over-simplified model of the environment and vehicle motion. Even in the airborne and underwater realm, existing SLAM approaches often assume a planar (2D) world model with heading (for 3DOF). They often assume altitude (or underwater depth) plus a planar model for mapping 3D environments.

- Sensors are typically treated as black boxes with a probabilistic measurement process. Existing systems typically assume that sensor noise is fully known a priori, is stationary, and independent of landmark and vehicle pose (or trajectory).

- Sensor calibration must be known or pre-tuned prior to deployment.

- Multi-sensor independence. Each sensor (and each measurement) is assumed to be independent of all other sensors (and measurements).

Traditionally, the observation model in SLAM is treated as though the measurements did not come from a dynamic process. The sensor itself is modelled as a stationary stochastic process biased by a zero-mean Gaussian error model. This type of error model is reasonable for sensors that contain very little error on a per-measurement basis and that allow for proper modelling using a conservative Gaussian model. Unfortunately, many sensor error processes are not well modelled using a single stochastic process. It may be that the underlying process is more complex than a simple Gaussian model. For example, the sensor errors may be modelled better by the sum of a Gaussian plus some non-Gaussian outlier process. Sensor measurements are often assumed to be independent of each other, which is not necessarily the case even with "clean" sensors such as laser rangefinders. The sensing process is complex; thus, it is often impossible to compute an explicit closed-form error model, or to have a reasoned mechanism for estimating a Gaussian-like error model for the sensor.

1.1.2 Applications

Modelling the sensor noise parameters successfully within the SLAM framework impacts many application domains. This dissertation concentrates on the application of SLAM in underwater environments. The underwater environment presents numerous challenges for the design of robot vision sensors and algorithms. Yet it is these constraints and challenges that make this environment almost ideal for the development and evaluation of robotic sensing technologies. Vehicles and their sensors operating in this environment must cope with unknown motion due to external forces such as currents, surf, and swells. These actions can produce unwanted and unpredictable 6DOF motion of the sensor. In spite of these complexities, the underwater environment provides a vast range of applications for which environmental reconstruction is desirable including inspection and entertainment applications.

Most importantly in the underwater realm, 6DOF vehicle motion is not well-modelled due to the effect of unknown forces — such as current or surge — acting on the vehicle body. This creates a necessity for the SLAM algorithm to rely heavily on sensor measurements to determine vehicle motion and map the environment.

Yet the sensor measurements that are available in the underwater domain are often more complex than the measurements assumed by traditional SLAM algorithms.

The primary sensor of concern in this dissertation is a stereo vision camera. Stereo-vision algorithms are often used by autonomous robots to extract dense range information enabling the creation of dense 3D models of the environment. Visual odometry is often complimented by other sensors, such as inertial measurement units (IMUs), to provide more precise measurements either at a faster rate or in situations where visual measurements cannot be made. IMUs measure the linear acceleration and angular rates applied on all axes. These measurements are then integrated to provide accurate 3DOF orientation estimates.

This environment and our choice of sensors allows us to examine potential error sources that affect the SLAM estimate in the field. These include errors that are static but unknown (such as fixed intrinsic and extrinsic camera calibration parameters); dynamic but environment-based errors (such as color resolution and lighting which change with underwater depth); dynamic but environment and position based errors (such as the disparity range used in the stereo algorithm which depends on what the world looks like and where the camera is relative to the world).

1.2 Objectives of the dissertation

In this dissertation, it is argued that the proper estimation of the sensor parameters is important for SLAM when using sensors in which there are multiple error sources per measurement. These errors may be induced by unknown, or un-modelled, intrinsic calibration parameters, time-varying biases, or even map/location dependent external factors. Estimation of these parameters should be performed as part of the SLAM algorithm rather than as an offline process. This is especially true for applications where the sensor parameters are a function of the map and vehicle location. Such situations occur in many application domains including the underwater realm which is examined here.

This dissertation contributes to existing research in the following ways,

1. It provides a validation of underwater stereo-inertial integration for trajectory

and 3D model recovery. It shows that the utilization of IMU data improves the accuracy of ego-motion estimation when using 2D/3D feature tracking with underwater stereo images.

2. It develops a theoretical exploration into the use of sensor parameters within the probabilistic SLAM framework.

3. It develops an algorithm (SensorSLAM) to simultaneously estimate sensor calibration parameters, the robot location, and the map.

4. It provides experimental validation of the SensorSLAM algorithm for 6DOF underwater applications.

The main contribution of this dissertation is to explore the integration of sensor parameter estimation into the SLAM framework. Simulations are performed to validate the solution and an algorithm is developed under various assumptions that enable estimation specifically for 3D reconstruction in the underwater domain. Traditionally these error sources are pre-calibrated through a separate calibration methodology however this assumes that the parameters are fixed for the duration of the experiment or uniform and fixed over the capture volume. This work provides a general framework to incorporate the estimation of the parameters, the location, and the map simultaneously. Provided that the parameters are stationary in either of time or space, the general framework is able to capture the other non-stationary noise components of the model. An application relevant to the thesis which can be accommodated by the framework is the use of an amphibious vehicle. Given a vehicle that can perform sensing on land and in the water and be able to transition smoothly between the two environments, the framework is able to capture the map-dependent (non-stationary) sensor noise in either (or both) of the environments. The calibration parameters of visual sensors must be modified depending on whether the robot is being operated on land or in the water. Other situations occur where noise is map/pose dependent which can be modeled using this approach such as areas with fog coverage. As the sensor goes from areas of little or no fog to areas which have a high fog component affecting the visual sensors, different sensor parameters may have to be used to reconstruct the environment and eliminate the

effects induced by the fog. Another example of a situation are areas that transition from very bright lit areas to very dimly lit areas (commonly found in space missions), in this situation the noise parameters must be chosen correctly to operate effectively within some reasonable error bound. The approach developed in this thesis is designed to accommodate such situations.

1.3 Structure of the dissertation

This dissertation is organized into the following chapters:

- Chapter 2 presents previous work and discusses the limitations of current SLAM algorithms in the context of sensor modelling and 3D reconstruction.
- Chapter 3 presents 3D reconstructions obtained using a vision-based approach in a real-world underwater application and illustrates the difficulties in obtaining accurate maps.
- Chapter 4 presents the theoretical foundation for SensorSLAM: an approach to the SLAM problem that provides a more general approach to sensor modelling and auto-calibration.
- Chapter 5 presents a set of experiments that illustrate the effectiveness of the approach and contrast it with the results obtained from the algorithm presented in Chapter 3.
- Chapter 6 discusses the results and provides directions for future work.

1.4 Previously published material

Much of the work described in this dissertation has been previously published at these refereed conferences and journals: [Jenkin et al., 2008, Hogue et al., 2007a, Jenkin et al., 2007, Dudek et al., 2007, Hogue and Jenkin, 2006b, Hogue et al., 2006a], presented as posters at conferences [Hogue and Jenkin, 2003a, Hogue and Jenkin, 2003b, Hogue and Jenkin, 2004a,

Hogue and Jenkin, 2004b, Hogue and Jenkin, 2005a, Hogue and Jenkin, 2005b, Hogue and Jenkin, 2005c, Hogue and Jenkin, 2006a, Hogue et al., 2006b, Hogue et al., 2007b, Hogue and Jenkin, 2007] and a land-based human modeling application using the approach has been developed and published [Hogue et al., 2007c].

CHAPTER 2

Related work

Even though SLAM is a relatively recent development (dating back only to the early 1980's), the relevant research/literature is vast. Consequently, it is not possible to review all of the material related to SLAM here. Detailed surveys of the mapping and localization field can be found in [Thrun, 2000, Thrun, 2003] and in [Thrun et al., 2005]. In this dissertation, the review of SLAM is targetted specifically at the error models used and the requirements of existing algorithms for a priori error models. In addition to SLAM, also relevant to the discussion at hand are topics in photogrammetry and structure from motion (SFM). A review of the relevant literature from these fields is also presented here. In order to put this work in context, this chapter begins with an introduction to some of the problems involved in the development of autonomous underwater vehicles.

2.1 Autonomous underwater vehicles and associated problems

Autonomous Underwater Vehicles (AUVs) are an emerging technology for a range of scientific endeavours [Clarke, 2003]. AUVs are used to explore and collect information about the underwater environment; an environment which can be extremely hazardous to human explorers. Figure 2.1 shows examples of existing autonomous underwater vehicles. Not surprisingly, there has been much work done in the design and control of AUV systems. See [Rodseth and Hallset, 1991, Caccia et al., 1999, Caccia et al., 2000, Bruzzone et al., 2001, Caccia et al., 2001, Caccia et al., 2005, Caccia, 2006] for representative work. Examples of notable AUV projects include Kambara [Reynolds, 1998, Fitzgerald, 1999, Bryant, 2000, Biddle, 2003], Orca [Turner, 1995], the Mako Project [Braunl et al., 2004] and the AQUA project

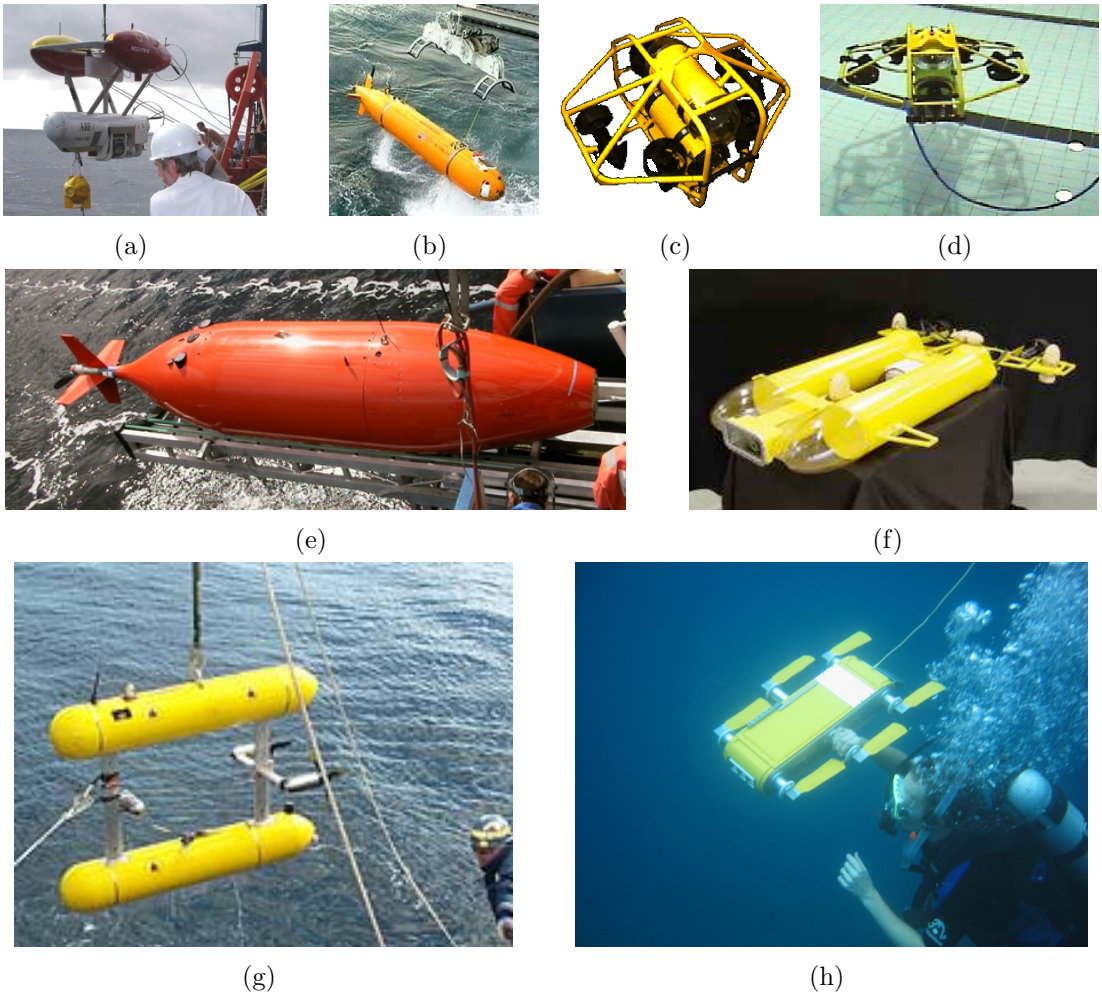


Figure 2.1: Examples of autonomous underwater vehicles (a) Autonomous Benthic Explorer [Jakuba and Yoerger, 2003], (b) Autosub [Brierley et al., 2002], (c) KAMBARA [Wettergreen et al., 1998], (d) Oberon [Majumder, 2001], (e) HUGIN [Hagen et al., 2003], (f) Starbug AUV [Dunbabin et al., 2006], (g) SeaBED AUV [Singh et al., 2004a], (h) The AQUA Robot [Dudek et al., 2007].

[Dudek et al., 2007].

One motivation for the development of AUVs is the need to monitor coral reef health more effectively. Coral reefs are indicators of global climate change. Their health changes rapidly due to small fluctuations in oceanic temperatures. The coral bleaching event in 1998 in Okinawa — where an estimated 80% of the coral died — was due to a global increase in ocean temperature of only a few degrees [Furushima et al., 2004]. To understand and monitor coral health, a survey program called "Reef Check" was instituted globally in 1996 to enlist the aid of the public in the assessment of global coral health [Hodgson, 1999]. Reef Check is an "easy-to-learn" method that systematically assesses the variety of fish, invertebrates and substrate in a given region but can be time consuming and is prone to human error. AUVs designed specifically to perform coral surveys have been deployed [Williams and Mahon, 2004a, Singh et al., 2004a, Dunbabin et al., 2004, Dunbabin et al., 2005a, Dunbabin et al., 2005b, Armstrong et al., 2006, Dunbabin et al., 2006]. In addition to coral monitoring, other applications for AUVs include underwater archaeology [Caiti et al., 2006, Ludvigsen et al., 2006], the evaluation of organism density under sea ice in the Antarctic [Bone et al., 1994, Bono et al., 1998, Bono et al., 1999, Brierley et al., 2002], underwater cable tracking [Ortiz et al., 2002], automatic visual pipeline inspection [Hallset, 1991a, Hallset, 1991b, Hallset, 1992], ship-hull inspection [Negahdaripour and Firoozfam, 2006] and covert environmental assessments for military applications [Hagen et al., 2003]. Acoustical sensors have been used to reconstruct shallow archaeological sites (e.g. boat wrecks and coral reefs) in [Plets et al., 2008] and laser scanners have also been used to extract 3D information of coral reef as in [Holmes, 2008].

A popular method used to acquire high resolution imagery/maps for AUVs is the use of photomosaicking algorithms to stitch and blend sequences of images together [Xu and Negahdaripour, 1997, Singh et al., 2002, Singh et al., 2004b, Singh et al., 2007]. Developing computer vision techniques for use underwater is important for visually guided AUVs (e.g. [Wettergreen et al., 1998] and [Dudek et al., 2007]). Issues related to underwater imaging include (but are not limited to) dynamic lighting, refraction and color resolution that is based upon the sen-

sor depth [Duntley, 1963, Kwon, 1999b, Schechner and Karpel, 2004]. These factors complicate vision-based sensing in the underwater environment. Recent trends and problems with underwater imaging were discussed in [Shortis et al., 2007, Kocak et al., 2008].

Underwater mapping and localization using acoustical (sonar) imaging devices has been widely studied (see [Ribas et al., 2007b, Williams and Mahon, 2004b, Bono et al., 1994, Caccia et al., 1997, Jakuba and Yoerger, 2003, Yoerger et al., 2005, Yoerger et al., 2007]). These techniques have been deployed in marine environments [Ribas et al., 2007a], partially structured environments [Ribas et al., 2006], man made environments [Ribas et al., 2008] and natural underwater caves and tunnels [Fairfield et al., 2006, Fairfield et al., 2007]. Determining an accurate motion model for an autonomous underwater vehicle is a complex task. Other than forces applied by motion commands and kinematics, unknown three-dimensional forces such as current (a uni-directional force) or surge (a sinusoidal-like force) continually affect the vehicle displacement (see [Makarenko et al., 1997] for an example of a hydrodynamic model for a submersible). Propellors or legs associated with the vehicle create vortices that change the vehicle dynamics, and tethers attached to vehicles complicate matters. Many algorithms have been developed to extract vehicle motion from imagery including [Negahdaripour et al., 1990, Yu and Negahdaripour, 1993, Negahdaripour and Lanjing, 1995, Negahdaripour et al., 1996, Negahdaripour et al., 1998, Negahdaripour et al., 1998]. Stereo vision systems are also popular in the underwater research community, for example passive stereo systems [Negahdaripour et al., 1995, Khamene and Negahdaripour, 1999, Negahdaripour et al., 2002, Negahdaripour and Madjidi, 2003, Negahdaripour and Firoozfam, 2006, Hogue and Jenkin, 2006b, Hogue et al., 2007a, Jenkin et al., 2008] as well as active vision systems have been investigated [Strickrott and Negahdaripour, 1997]. The measurement of reef fish lengths is difficult to determine accurately since the typical algorithm is to have divers visually inspect and report the type of fish and their lengths to aid in determining the health of the coral environment. In [Harvey et al., 2001], the diver visual inspection accuracy is determined by

using a passive stereo-video system to extract metric information of the fish lengths. This requires robust calibration algorithms which were described in [Harvey and Shortis, 1998].

2.2 Maps and their representations for 2D/3D environments

Maps are an enabling technology for many human endeavors including autonomous underwater vehicles. Maps are fundamental tools that aid self-navigation and autonomous operation. Thus, it is not surprising that there is a long history of map construction and the development of tools and techniques to support map building. There is evidence that humans have been building maps since the Upper Paleolithic era (10,000 to 40,000 years ago); however, the Babylonians are credited for making the earliest recorded attempt at a reasoned conception of the universe using graphical mapping techniques [Harley and Woodward, 1987]. Although maps had found wide use in land-based and near shore environments, it wasn't until the invention and adoption of navigational tools such as the compass, telescope, and sextant (from the 13th to 17th centuries) that the map-building process was suitable for navigating the oceans.

In the robotics literature, maps are generally thought of as a collection of entities describing the spatial distribution and relationships between objects or features in the environment located on a 2D Cartesian grid (or 3D volume). That being said, there are many different map representations including topological maps [Kuipers and Byun, 1991, Simhon and Dudek, 1998, Thrun, 1998, Choset and Nagatani, 2001], 2D metric occupancy grids [Elfes, 1989b], 3D volumetric grids [Foley et al., 1990] and 3D hierarchical volumes such as Octrees [Gervautz and Purgathofer, 1988].

A commonly adopted map representation for robotics applications is the 2D occupancy grid due to its simplicity both computationally and conceptually. This representation is sufficient for environments that are planar, (or locally planar) such as man-made office spaces, levelled outdoor environments, and even out-

door environments that are locally planar with gentle slopes. However, 2D grids do not sufficiently characterize environments that are naturally 3D. If the vehicle is capable of 6DOF motion through a 3D environment, it is necessary to utilize higher order representations (at higher computational and algorithmic cost). Examples of such 3D environments include outerspace, underwater, and even land-based environments containing 3D structures that do not lend themselves well to 2D projection such as tunnels, or highly variable mountainous regions. Other non-feature based representations for models and maps exist such as scalar fields or implicit functions [Bloomenthal, 1988, Wang et al., 2005], Level-Sets [Bærentzen, 2002, Bischoff and Kobbelt, 2003, Gomes and Faugeras, 1999, Houston et al., 2004], or distance fields [Jones et al., 2006, Sigg et al., 2003, Frisken et al., 2000, Perry and Frisken, 2001] which have yet to be explored in the context of SLAM.

2.3 Localization given a map

Localization — the ability to determine the location of the vehicle — is a common robotics problem. When a suitable map of the environment is known *a priori*, it is sufficient to find the associations between sensor data and known feature locations in the map. Given this knowledge, an accurate transformation can be determined to denote the vehicle location. For example, in an autonomous museum guidance application, museums have a very well defined floor plan that is known accurately. A robotic vehicle can use this prior knowledge of the environment to self-navigate. The map can be acquired in many ways prior to autonomous operation, possibly from a tele-operated session, or manual map creation using blueprints/CAD drawings. The Minerva project (see [Thrun et al., 2000a]) developed such an autonomous museum guidance robotic vehicle.

Many robotic systems have utilized this approach. Leonard and Durrant-Whyte developed a localization method using an extended Kalman Filter (EKF) framework [Leonard and Durrant-Whyte, 1991]. In this work, an EKF was used for robot pose estimation through the observation of known geometric-beacons in the environment. Planes, cylinders and corners were used as distinctive geometric entities that could

be reliably extracted from the sonar sensor data. The map is a set of known beacon locations and were assumed to be static and naturally occurring in the environment. Using the robot location and error model from the EKF, a method was devised to acquire accurate data association using the known beacon locations for the next EKF update. In [Cox, 1991], range data from an optical rangefinder was used to acquire a 360 degree range image around the vehicle. Line segments extracted from the range data were matched through an iterative least squares approach to a known map to determine the vehicle location.

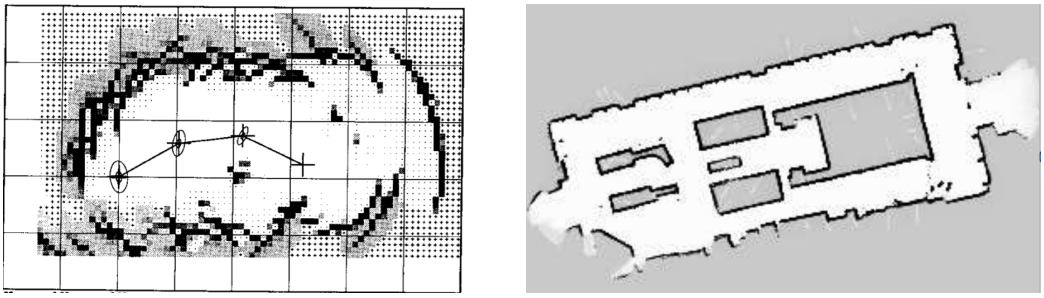
2.4 Mapping given localization

In the reverse problem — knowledge of the robot location in the world coordinate frame — sensor data is fused together to generate a map representation of the environment assuming some independent mechanism to maintain the pose of the vehicle. The occupancy grid introduced by Elfes and Moravec [Moravec and Elfes, 1985, Elfes, 1989b, Elfes, 1989a] (see Figure 2.2) is an example of such an approach. In an occupancy grid, the environment is represented as a fine-grained grid where each cell is in either an occupied, empty, or unknown state. Each cell of the grid maintains a probability representing the degree of certainty that it is occupied. As range data is gathered from the sensors (in this case sonar range sensors are used), empty and occupied cells are identified. The cells between the current robot location and the sensed range are set to an *empty* state due to the assumed line-of-sight properties of sonar technology. Similarly, the cells at the sensed range are updated to an *occupied* state. In Elfes' Thesis [Elfes, 1989a], the process of updating the occupancy grid is described as:

1. The sensor reading is converted into a probability distribution using a stochastic sensor model.

2. Bayes rule is applied to update the cell probabilities.

3. If desired, the maximum a posteriori (MAP) decision rule is applied to label the cell as being either occupied, empty, or unknown.



(a) [Elfes, 1989b] (b) [Thrun et al., 2000a]

Figure 2.2: Examples of Occupancy Grids

For many robotic tasks, it is sufficient to use the probabilistic representation of the occupancy grid; it is rare that the final MAP stage is applied. The occupancy grid representation has received much success in the field robotics community and has also been applied in situations within which localization is not assumed, (see [Buhmann et al., 1995, Guzzoni et al., 1997, Lu and Milios, 1997, Burgard et al., 1999, Thrun et al., 2000a]). It is still used at the core of more recent algorithms such as DP-SLAM (see [Eliazar and Parr, 2003, Eliazar and Parr, 2004]) and is used as a visualization tool for others (see [Montemerlo et al., 2002, Sim et al., 2007]).

2.5 SLAM

Chatila and Laumond in [Chatila and Laumond, 1985] were among the first to develop the idea of simultaneous localization and mapping. They argued that in order for a robot to understand its environment, it must be able to create a consistent world model of the environment and locate itself within this model. The algorithm attempted to create three types of environment models: a *geometric* model using metric sensor data, a *topological* model to express areas that are connected for navigation, and a *semantic* model to add semantic meaning to objects for a higher-level decision making process. They also employed three types of localization. First, absolute positioning was determined using known beacons in the environment. Second, the trajectory of the vehicle was integrated using odometric

and inertial information both of which are subject to drift. Third, relative positioning was performed with respect to objects or other distinctive features that are characteristic of the environment. The authors observed that if an estimate of the robot position is maintained along with appropriate uncertainties derived from the motion and odometric/inertial error model, overlapping sensor information could be used in a position correction stage. As the robot moves in the environment, its sensor information (range to walls in this case) contains an area that has already been seen. Thus, given the approximate (but noisy) estimate of the current robot's motion, the sensor data can be used to update the map and the robot's position. The uncertainties in the measurements are modelled as zero-mean Gaussian error functions and the update method incorporates the uncertainty of the measurements by computing a weighted average. This update scenario is a direct pre-cursor to using a more general Kalman filter formulation.

Building on the principles developed in [Chatila and Laumond, 1985] — using the idea of estimating a map while localizing the vehicle and using proper uncertainty models for the measurements — Smith, Self, and Cheeseman [Smith and Cheeseman, 1986, Smith et al., 1990, Smith et al., 1991] introduced a more formal and general approach to the SLAM problem by developing a probabilistic framework to model the spatial-relationships between landmarks in the environment. They placed the mapping and localization problem within a formally defined stochastic parameter estimation framework. Their framework of choice was a linear least-squares optimal recursive filter, namely the Kalman filter. Smith et al. argued that mapping is an extension to localization and provided a rigourous estimation-based algorithm. They did not describe the important side-effects of this type of SLAM formulation. As Csorba argues in his thesis [Csorba, 1997], the main effect that the probabilistic formulation of SLAM has, is that the errors between the vehicle and feature estimates become fully correlated in the limit as the number of observations grows. He argues that it is these correlations that "capture the essence of the SLAM problem."[1]. Csorba also provided a rigourous theoretical investigation into the SLAM problem and examined the convergence of the solu-

[1][Csorba, 1997], pg 9.

tion. Since then, SLAM has been applied in many domains including underwater [Williams and Mahon, 2004b, Eustice, 2005], underground [Thrun et al., 2004b], on land — indoors [Takezawa et al., 2004] and outdoors [Newman et al., 2006, Wang et al., 2007a] — and in the air [Elfes et al., 1998, Kim and Sukkarieh, 2007, Bryson and Sukkarieh, 2007, George and Sukkarieh, 2007a].

Smith et al. formulated SLAM as a recursive state estimation process. The state vector they wished to estimate included the robot pose (position and orientation), as well as the landmark positions. For the robot pose, the 2D robot position and orientation along with its covariance are estimated

$$\hat{\mathbf{x}}_{\text{pose}} = \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{\theta} \end{bmatrix}, \hat{\mathbf{C}}_{\text{pose}} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{yx} & \sigma_y^2 & \sigma_{y\theta} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 \end{bmatrix}.$$

Given N landmarks, the 2D position in the world frame of landmark i is estimated as

$$\hat{\mathbf{x}}_{\mathbf{i}} = \begin{bmatrix} \hat{x}_i \\ \hat{y}_i \end{bmatrix}$$

To simplify notation, the pose state vector and each landmark state vector are concatenated into a $(3 + 2N) \times 1$ state vector, $\mathbf{x}$, that they wished to estimate. To parameterize the state vector distribution, they estimate the mean $\hat{\mathbf{x}}$,

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_{\text{pose}} \\ \hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \\ \vdots \\ \hat{\mathbf{x}}_n \end{bmatrix}.$$

and the jointly estimated covariance matrix, $\hat{\mathbf{C}}$,

$$\hat{\mathbf{C}} = \begin{bmatrix} \hat{C}(x_{\text{pose}}) & \hat{C}(x_{\text{pose}}, x_2) & \cdots & \hat{C}(x_{\text{pose}}, x_n) \\ \hat{C}(x_1, x_{\text{pose}}) & \hat{C}(x_1) & \cdots & \hat{C}(x_1, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{C}(x_n, x_{\text{pose}}) & \hat{C}(x_n, x_2) & \cdots & \hat{C}(x_n) \end{bmatrix}$$

where each $\hat{C}(x_i)$ is a 3×3 covariance matrix and the following relationships hold

$$\hat{\mathbf{C}}(x_i, x_j) = E\left((x_i - \hat{x}_i)(x_j - \hat{x}_j)^T\right),$$
$$\hat{\mathbf{C}}(x_j, x_i) = \hat{C}(x_i, x_j)^T.$$

Of importance are the off-diagonal entries which denote the covariances between landmarks (and also between the landmark and the robot pose). These encode the relationships between each pair of landmarks and also between the robot pose and each landmark. The simplest model to use mathematically to manipulate these random variables is the Gaussian distribution as it only relies on the mean and covariance of the distribution.

The state is estimated recursively through a predict-correct loop implemented using a Kalman filter (see [Kalman, 1960]). The prediction stage incorporates the motion model of the vehicle as

$$\hat{x}_t = f(\hat{x}_{t-1}, u_{t-1}, 0)$$
$$C(\hat{x}_t^-) = A_t C(\hat{x}_{t-1}) A_t^T + W_t Q_{t-1} W_t^T$$

where $f(\cdot)$ is a function that describes the motion of the vehicle. This motion model projects the state forward given the previously estimated state ($\hat{x}_{t-1}$) and the control input (u_{t-1}). Q_{t-1} is the system process noise, W_t is the process noise Jacobian and A_t is the system Jacobian that relates the motion model to the state. For the estimator to be optimal, the motion model must be a linear function however the use of an extended Kalman filter, that linearizes the non-linear motion model, while being suboptimal performs well in practice.

The correction step is given by

$$\hat{x}_t^+ = \hat{x}_t^- + K_t\left[z_t - h_t(\hat{x}_t^-)\right]$$
$$C(\hat{x}_t) = C(\hat{x}_t^-) - K_t H_t C(\hat{x}_t^-)$$
$$K_t = C(\hat{x}_t^-) H_t^T \left[H_t C(\hat{x}_t)^- H_t^T + C(v)_t\right]^{-1}$$

where K_t is the Kalman gain, $\hat{x}_t^-$ is the predicted state at time t, $\hat{x}_t^+$ is the corrected state given the error between the measurements and what was predicted using the measurement function $h_t(\cdot)$, and v represents the measurement noise.

This algorithm makes assumptions — as stated in [Smith et al., 1991] — that are reasonable for most applications. These assumptions can be summarized as

- Functions on the random variables are relatively smooth around the estimated means within one standard deviation.
- It is sufficient to estimate the first two moments of the distribution.

The algorithm by Smith et al. utilizes the extended Kalman filter equations as the robot motion and measurement models are nonlinear and as such is considered to be sub-optimal however it approaches optimality if an iterated extended Kalman filter [Maybeck, 1979] is used.

2.5.1 Formal Bayesian definition

A Bayesian approach to robot mapping was formalized in [Thrun et al., 1998] and [Durrant-Whyte et al., 2003]. Most formulations have since used this notation as a basis. A Bayesian probabilistic framework for SLAM represents each of the robot position and map locations as pdfs. Within this framework, the goal is to estimate the posterior density of maps Θ and poses s_t given knowledge of the observations $z^t = \{z_1, z_2, \ldots, z_t\}$, the control inputs $u^t = \{u_1, u_2, \ldots, u_t\}$ and data associations $n^t = \{n_1, n_2, \ldots, n_t\}$ which represent the mapping between map points in Θ and observations in z^t. The SLAM posterior, as defined in Montemerlo (see [Montemerlo et al., 2002]), is given by

$$p(s_t, \Theta | z^t, u^t, n^t). \tag{2.1}$$

The system dynamics motion model assumes the Markov property

$$p(s_t | s^t, u^t) = p(s_t | s_{t-1}, u_t)$$

and the sensor measurement model is

$$p(z_t | s_t, \Theta, n^t).$$

The derivation of the SLAM posterior is detailed in [Montemerlo et al., 2002] but it is worthwhile to repeat here in detail since it provides insight into the SLAM

problem and how successful solutions are modeled. The first step in deriving the SLAM posterior is to apply Bayes rule. Bayes rule (see [Stark and Woods, 2002]) relates conditional probabilities as

$$\begin{aligned} p(A|B) &= \frac{p(B|A)p(A)}{p(B)} \\ &= \eta p(B|A)p(A) \end{aligned}$$

where $\eta^{-1} = p(B)$ is a normalization factor ensuring that the resulting value is a probability. Also useful is the similar extension to Bayes rule

$$p(A|B,C) = \eta p(B|A,C)p(A,C) \tag{2.2}$$

and by using the substitutions from our problem domain, namely $A = \{s_t, \Theta\}$, $B = z_t$, and $C = \{z^{t-1}, u^t, n^t\}$, Equation 2.2 becomes

$$p(s_t, \Theta|z^t, u^t, n^t) = \eta p(z_t|s_t, \Theta, z^{t-1}, u^t, n^t)p(s_t, \Theta|z^{t-1}, u^t, n^t). \tag{2.3}$$

where η is a normalizing term. The $p(z_t|s_t, \Theta, z^{t-1}, u^t, n^t)$ term can be simplified by assuming the measurements are independent of each other, thus z_t is independent of z_{t-1} and all other measurements. The measurement at this time step is also independent of the control input. This allows for the following simplification

$$p(z_t|s_t, \Theta, z^{t-1}, u^t, n^t) = p(z_t|s_t, \Theta, n_t).$$

and the SLAM posterior (Equation 2.3) can be re-written as

$$p(s_t, \Theta|z^t, u^t, n^t) = \eta \underbrace{p(z_t|s_t, \Theta, n^t)}_{\text{Measurement Model}} p(s_t, \Theta|z^{t-1}, u^t, n^t). \tag{2.4}$$

The next step is to evaluate the last factor in Equation 2.4, $p(s_t, \Theta|z^{t-1}, u^t, n^t)$. The theorem of total probability (see [Stark and Woods, 2002]) enables the use of the prior history of the robot motion. Assuming that the previous state and the odometry contains all of the necessary information required to predict the motion of the vehicle (Markov Assumption). This simplifies the pdf as

$$\begin{aligned} p(s_t, \Theta|z^{t-1}, u^t, n^t) &= \int p(s_t, \Theta, s_{t-1}|z^{t-1}, u^t, n^t)ds_{t-1} \\ &= \int p(s_t|\Theta, s_{t-1}z^{t-1}, u^t, n^t)p(s_{t-1}, \Theta|z^{t-1}, u^t, n^t)ds_{t-1}. \end{aligned}$$

and the SLAM posterior can now be re-written as

$$p(s_t, \Theta | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta, n^t) \int p(s_t | \Theta, s_{t-1}, z^{t-1}, u^t, n^t) \\ p(s_{t-1}, \Theta | z^{t-1}, u^t, n^t) ds_{t-1}.$$

If the map is dynamic then a new map is possible at each time step as denoted by the subscripted Θ_t. In this case, the SLAM posterior to be estimated is given by

$$p(s_t, \Theta_t | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta_t, n^t) \cdot \\ \iint p(s_t | \Theta_t, u_t, s_{t-1}, z^{t-1}, n^t) p(\Theta_t | s_{t-1}, \Theta_{t-1}, z^t, u^t, n^t) \cdot \\ p(s_{t-1}, \Theta_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) ds_{t-1} d\Theta_{t-1}.$$

This is computationally very difficult, so the most common assumption in SLAM algorithms is a static map

$$p(s_t, \Theta | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta, n^t) \cdot \\ \int p(s_t | s_{t-1}, \Theta, u_t, z^{t-1}, n^t) \cdot \\ p(s_{t-1}, \Theta | z^{t-1}, u^{t-1}, n^{t-1}) ds_{t-1}.$$

Under the Markov assumption, the robot pose at time t, denoted by s_t, is conditionally independent of all variables except its own pose in the previous time step s_{t-1} and the current control input

$$p(s_t | s_{t-1}, \Theta, z^{t-1}, u^t, n^t) = p(s_t | s_{t-1}, u_t).$$

which gives the standard robotic motion model. Under this assumption, SLAM can be re-expressed as

$$p(s_t, \Theta | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta, n^t) \\ \int \underbrace{p(s_t | s_{t-1}, u_t)}_{\text{Motion Model}} p(s_{t-1}, \Theta | z^{t-1}, u^t, n^t) ds_{t-1}.$$

Since $u^t = \{u_{0:t-1}, u_t\}$ and $n^t = \{n_{0:t-1}, n_t\}$ and as u_t, n_t do not influence the previous measurement and pose, z_{t-1}, s_{t-1}, they can be eliminated. Finally the

Bayesian filtering equation for SLAM can be written as a probabilistic framework (as in [Montemerlo et al., 2002]) as

$$p(s_t, \Theta | z^t, u^t, n^t) = \eta \underbrace{p(z_t | s_t, \Theta, n^t)}_{\text{Measurement Model}} \int \underbrace{p(s_t | s_{t-1}, u_t)}_{\text{Motion Model}} \underbrace{p(s_{t-1}, \Theta | z^{t-1}, u^{t-1}, n^{t-1})}_{\text{Recursive formulation}} ds_{t-1} \tag{2.5}$$

Note that the second term in the integral — $p(s_{t-1}, \Theta | z^{t-1}, u^{t-1}, n^{t-1})$ — is the SLAM posterior for the previous time step thus realizing a recursive estimation process suited towards Kalman filtering. This is the traditional Bayesian formulation of the SLAM problem as is realized in modern implementations for map-creation. SLAM essentially is defined as estimating the joint probability of the current robot location (s_t) and the map (Θ) given knowledge of the observations ($z^t = \{z_0, z_1, \ldots, z_t\}$), the robot control inputs ($u^t = \{u_1, u_2, \ldots, u_t\}$) and data associations ($n^t = \{n_1, n_2, \ldots, n_t\}$). Equation 2.5 incorporates a probabilistic measurement model $p(z_t | s_t, \Theta, n^t)$ and a system dynamics motion model $p(s_t | s_{t-1}, u^t)$ into a single expression and is suitable for recursive estimation through the use of the Markov assumption for robot locations.

Placing SLAM in this framework involves providing explicit mechanisms for representing the pdf's used in the representation and developing mechanisms for interpreting the resulting pdf's as maps and position estimates. Two traditional approaches to solving this problem are described below.

2.5.2 Kalman filter-based SLAM

One approach to SLAM implements the probabilistic framework using recursive estimation techniques and Gaussian pdfs. The standard approach utilizes an extended Kalman filter, detailed in [Smith and Cheeseman, 1986, Smith et al., 1990, Smith et al., 1991, Newman, 1999]. Each of the pdf's in the above probabilistic formulation are assumed to be Gaussian in nature allowing the use of the Kalman filter framework. The motion model and the measurement model are assumed to be almost linear so that a first-order Taylor-series is a valid approximation that

properly represents the nonlinearity in the system. The ease of implementation is the main advantage to approaching SLAM in this way. The most straightforward implementation uses a Kalman filter with the state vector to estimate the position, and possibly velocity, of the robot and each of the landmarks in the map. If the number of landmarks are known prior to operation, the state stays the same size. If new landmarks must be estimated then the state vector increases in size proportionally to the number of landmarks. This necessitates the usage of an appropriate data structure for the map. The main disadvantage to using the Kalman approach or an extended Kalman filter approach is that the motion and measurement models must be linear (or locally linear) and the noise models must be appropriately modeled using a Gaussian. These two assumptions in the Kalman filter are easily invalidated ([Tanizaki, 2003]) when navigating through real environments. Computational issues plague EKF based SLAM such as the complexity due to matrix inversion[2], however there have been recent advances that reduce standard EKF SLAM updates to $O(n)$ using a divide and conquer strategy [Eustice et al., 2005c, Eustice et al., 2005a] when formulating the SLAM problem in a delayed state information form.

EKF SLAM operates on discrete sets of salient features to represent the map and reduce processing time. This can be detrimental to algorithm performance if saliency is determined more globally rather than locally. Sensors produce a large amount of data about the scene, laser range scanners produce many measurements per second as do stereo cameras creating a large density of range information. Work has been done to use the dense data set within a SLAM framework to incorporate the large amount of data appropriately into the map representation [Mahon and Williams, 2003, Nieto et al., 2006b]. Recently, other map representations such as using B-Splines have also been proposed [Pedraza et al., 2007].

[2]The algorithm presented in [Coppersmith and Winograd, 1987] performs matrix inversion in $O(n^{2.376})$ at each update is approximately $O(n^3)$, but few people use such optimized implementations in practice.

Sparse extended information filters (SEIF)

A practical issue in Kalman filter approaches is how to handle new measurements. The state vector must be dynamically sizable to accommodate a new measurement, however more problematic is the need to dynamically update the size of the covariance of the state vector and update the covariance entries appropriately. Given this, there has been a thrust in developing SLAM solutions using techniques that make the filter update more computationally efficient. Most notably here is the work by Nettleton and others [Maybeck, 1979, Nettleton et al., 2000] using Extended Information Filters (EIF). The EIF algorithm is analogous to the EKF algorithm described previously, however instead of maintaining a covariance matrix, the information matrix (the inverse of the covariance matrix) is maintained and the update equations are changed to accomodate this formulation. The use of information filters for SLAM was proposed in [Frese and Hirzinger, 2001] and implemented in [Thrun et al., 2004a], and is related to the work in [Lu and Milios, 1997]. Due to the structure of the SLAM problem, the landmark estimates become fully correlated over time [Newman, 1999] and the normalized information matrix corresponding to the SLAM solution is almost sparse. The non-zero elements in the information matrix represent constraints on the relative positions of features in the map. The larger the value of the entry in the information matrix, the stronger the link between the two encoded features. Features that are further apart tend to have weaker links between them. Exploiting this information allows for a faster update since only elements that are near the robot need to be updated instead of updating all of the relationships between each landmark in the map. The links also encode information about the robot pose relative to the map. At each step of the algorithm, weak off-diagonal elements are removed from the information matrix to guarantee sparsity. By exploiting the sparsity of the information matrix, the algorithm complexity is reduced from the EKF's $O(n^3)$ to $O(n)$. The most important update equations can be performed in constant time. The time complexity for each update is independent of the number of landmarks being estimated in the map. Even though the algorithm requires constant time for a measurement update, the relaxation process used introduces small errors in the resulting map that affects negatively the

overall map fidelity. Recent advances in this area are due to Eustice's work on Exactly Sparse Delayed State Filters or ESDF [Eustice et al., 2005a]. Eustice showed that if the SLAM problem is re-formulated as a view-based delayed-state information form, the information matrix is exactly sparse and does not require the extra *sparsification* stage. Instead of estimating the map feature locations, a view-based representation is defined and the estimation problem becomes one of estimating the current robot pose and a sampling of the previous poses. The ESDF formulation is closely related to structure from motion algorithms such as bundle adjustment that exploit the sparsity between the camera pose and the 3D scene locations to effectively solve a large system of equations that recover the camera trajectory and 3D locations. The algorithm was shown to be effective in solving for the trajectory of an underwater ROV that surveyed the RMS Titanic in [Eustice et al., 2005b]. A 3D surface model was subsequently extracted by triangulating the tracked features using the ESDF estimated trajectory. Related to these approaches is the D-SLAM algorithm [Wang et al., 2007b]. Wang and colleagues decoupled the SLAM problem into two concurrent but distinct processes. This results in the drawback that the robot location does not inform the mapping process. The advantage is the low computational complexity of the algorithm since they use an SEIF-like approach with a separate mapping process that retains correlations between landmarks.

2.5.3 Particle filter-based SLAM

Kalman filter approaches to SLAM impose restrictions on the way the pdfs are modelled. Gaussian (or Gaussian-like) models are integral to the derivation of the recursive update routine. When error models are not well modelled by a Gaussian, the filter may diverge. Particle filters model the pdf's through a sampling approach (see [Fox et al., 2001]). A large number of samples are used that are intended to be representative of the pdf. Each sample is propagated through the dynamic process and the covariance is estimated for each sample. The samples are weighted by their covariance and the entire pdf is re-sampled from this distribution. This general approach removes the Gaussian restriction in favour of a more computationally complex algorithm. Within the robotics literature, particle filters have been ap-

plied to pure localization (e.g., [Fox, 1998, Thrun et al., 2000b]), and SLAM (e.g., [Eliazar and Parr, 2003]).

The major advantage that particle systems have over Kalman filtering approaches is their ability to represent multi-modal distributions that occur from single or the combination of non-linear functions. Weaknesses include issues related to proper sampling of the pdf. This can lead to particle diversity problems and an inability to effectively track the process. Computational requirements are also a problem as particle filters require many particles to track an unknown distribution. As the distribution to be tracked becomes more complex in nature, i.e., due to the underlying sensing process, the filter may require more samples than can be computed in a reasonable amount of time on modern processors.

Most particle filter systems rely on Rao-Blackwellisation in order to realize efficiency in the sampling process. The term Rao-Blackwellisation (see [Doucet et al., 2000, Murphy and Russell, 2001]) refers to the process of transforming a crude estimator into an optimal estimator. The Rao-Blackwell theorem provides a mechanism to improve on any estimator by integrating out an ancillary statistic (see [Casella and Robert, 1996]). This transformation ensures that the conditional expected value that is estimated is no worse than the untransformed version and typically more convenient to compute. This has the important result of being able to marginalize out some of the variables in the pdf thus reducing the size of the state space.

FastSLAM

The FastSLAM algorithm, introduced by Montemerlo (in [Montemerlo et al., 2002]), takes a Rao-Blackwellized Particle Filter (RBPF) approach to SLAM. In traditional Kalman filter SLAM, the landmark estimates become fully correlated as time approaches infinity. This is due to the inherent correlation between landmarks and successive robot locations. Since the robot views all of the landmarks, the uncertainty of each landmark is tied not only to the robot motion, but also to the other landmarks due to the indirect correlation. If the robot location is known without error, the correlation between landmarks

disappears. In the Dynamic Bayes Network literature the robot path is said to *d-separate* (see [Pearl, 2000]) the landmark nodes in the graph. This implies that if the entire robot path is known, gaining knowledge of one landmark position does not affect the uncertainty of any other landmark in the set. It is sufficient to have knowledge of the robot path in order to properly characterize the distribution. This removes the conditional dependency between all landmarks and as such given the robot path, the landmarks can be estimated separately. Montemerlo (building on work done in [Murphy, 1999]) exploited this fact to marginalize the SLAM posterior over the robot motion. An implementation of this effect can be realized by using a particle filter to sample the entire robot trajectory. Each sample estimates its own version of the map and through the effects of statistical re-sampling, trajectories with lower errors (and higher fidelity maps) survive.

FastSLAM estimates a slightly different posterior than the regular stochastic mapping approach

$$p(s^t, \Theta | z^t, u^t, n^t) \tag{2.6}$$

where $s^t = s_1, s_2, \ldots, s_t$ is an entire robot path or trajectory. Thus, given the robot path, each landmark Θ_i is independent from the rest of the landmarks and can be estimated separately. This enabled Montemerlo to factor the SLAM posterior into the following form

$$p(s^t, \Theta | z^t, u^t, n^t) = \underbrace{p(s^t | z^t, u^t, n^t)}_{\text{path posterior}} \underbrace{\prod_{i=1}^{N} p(\Theta_i | s^t, z^t, u^t, n^t)}_{\text{N landmark estimators}} . \tag{2.7}$$

Due to the decoupling of the pose and the landmarks, SLAM estimation can be separated into a particle filter that estimates the robot pose, followed by N-landmark estimators (a single Kalman filter per landmark). Thus, the SLAM problem is partitioned into two parts: a localization problem and a mapping problem. FastSLAM addresses the localization problem through the use of the particle filter that samples the pdf of the robot trajectory and the map is constructed using a set of independent EKFs, one per landmark.

FastSLAM's strength is its algorithmic simplification of the SLAM framework to multiple independent SLAM instances. Its weaknesses lie in the map representation,

and issues related to particle filters in general such as algorithmic consistency, particle depletion and divergence (see [Bailey et al., 2006, Kwak et al., 2007]). In [Bailey et al., 2006], the authors show that FastSLAM is guaranteed to diverge over time and this divergence is independent of the number of particles used and of the observed landmark density. Recent examples of using FastSLAM like techniques for a vision-based robot can be found in [Barfoot, 2005] and [Sim et al., 2007].

DP-SLAM

DP-SLAM — [Eliazar and Parr, 2003, Eliazar and Parr, 2004] — is a pure particle-filtering approach to solving SLAM. It uses a particle filter to maintain a joint probability density over robot position as well as the possible map configurations. This approach removes the need to maintain separate EKFs for the landmarks as in FastSLAM. DP stands for *Distributed Particle* mapping and allows for efficient maintenance of hundreds of candidate maps and robot poses. The authors noted that since the map requires the most memory, only a single copy of the map should be maintained. They chose to associate the particles with the map instead of the standard approach of associating a map with each particle. The data structure used in DP-SLAM plays a key role in the efficiency of this algorithm. In each grid-cell, a balanced tree is kept of all of the particles that have updated this cell. A second data structure, called the ancestry tree, is also maintained per particle describing the relationship between each particle at time t and the sampled particle at time $t+1$. An update consists of the following; when a particle makes an observation about a particular grid cell, an identification number (ID) of that particle is inserted into the balanced data structure stored in the cell. In order to check the occupancy state of that cell, the particle looks into its ancestry tree and finds an ancestor that has updated this cell previously. If no ancestor is found then the state of the cell is currently unknown to this particle. In this way, each particle is able to efficiently maintain its own version of the map (i.e. thus multiple maps are estimated).

An important advantage of this algorithm over most SLAM solutions is that there is no need for explicit data association or external loop-closing. Since the algorithm maintains multiple maps and robot locations, the proper data associa-

tions are estimated and loops are automatically closed. The algorithmic complexity of this algorithm was analyzed to be log-quadratic in the number of particles [Eliazar and Parr, 2003].

Subsequently, an addendum to the original DP-SLAM algorithm called DP-SLAM 2.0 [Eliazar and Parr, 2004] was developed which proposed improvements to the laser sensor model of DP-SLAM. In the original DP-SLAM algorithm there was an inconsistency with the laser model since they assumed it was a perfect sensor for the mapping problem, but assumed it was noisy for localization. A probabilistic laser penetration model was later developed that is dependent upon the distance the laser has travelled through each grid cell. Using this method, the occupancy grid was refined from storing binary states (*occupied* or *empty*) to a more stochastic representation. Each cell has a probability associated with the laser penetration model, the higher the probability the more likely the cell is to be occupied. The search method for the localization stage was also improved. When an update occurs each particle is required to search through its ancestry to determine if any of its ancestors have updated this cell before. The new method uses a more efficient search method using a batch search of all ancestors at the same time. Simplifications can then be made to the algorithm using sorting methods and the overall time complexity is reduced from log-quadratic to simply quadratic in the number of particles [Eliazar and Parr, 2004]. The disadvantage to using this algorithm is the need for a tremendous number of particles (on the order of thousands to tens of thousands) to estimate a large map thus increasing processing time. The implementation of such an algorithm is complex and requires a tremendous amount of care for memory management.

2.5.4 SLAM with visual sensors

As high resolution video cameras have become less expensive, there has been renewed interest in using imagery and video as input to SLAM as opposed to the laser-like range sensors assumed in algorithms such as FastSLAM and DP-SLAM. Vision-based approaches have been coined *Visual SLAM*. Visual SLAM is related to the structure-from-motion (SFM) field in that algorithms in both of these fields

attempt to address the same issue — creating a map/model of the world and maintain the trajectory of the camera — but in very different ways. Typically in SFM, the motion of the camera is estimated through a series of nonlinear numerical minimizations using geometric constraints on the movement of image features throughout the video stream. SFM algorithms focus on accuracy, while Visual SLAM algorithms tend to focus on realtime performance. Visual SLAM algorithms also immerse the problem in a blanket of probabilistic theory to estimate the uncertainty of the visual features and world model. The uncertainty of the features is highly correlated to the uncertainty of the camera motion, the observation process, and other probabilistic processes.

Visual SLAM algorithms have been developed for both monocular (e.g, [Davison, 2003]) and stereo (e.g, [Takezawa et al., 2004]) camera systems. Differences between Visual SLAM and traditional SLAM with laser ranging or sonars include:

- Cameras are 2D area based sensors while lasers provide 1D range data. Laser ranging sensors need to be swept through the environment in order to acquire a linear range stripe.
- Cameras provide color or visual structure to the scene and can be used to analyze more salient features (rather than simply corners or other features in depth).
- A single camera does not produce a range estimate but rather gives an estimate of bearing to objects.
- Range extracted from stereo cameras are less accurate in general than laser sensors. Stereo camera accuracy is related to the baseline between the cameras, the resolution of the cameras, and the distance of the object.
- Stereo cameras typically produce more noisy range estimates than laser scanners at larger distances.
- Laser noise models are simpler than noise models for stereo and more easily integrated into SLAM.

- Computational requirements are greater for stereo algorithms than for using laser data-based systems.

Monocular camera based SLAM algorithms (e.g. [Davison, 2003, Montiel and Davison, 2006, Civera et al., 2007, Davison et al., 2007, Pinies et al., 2007, George and Sukkarieh, 2007b, Civera et al., 2008]) use the fact that a single camera is not a range device, but rather is a "bearing-only" sensor. Thus, for each interesting feature in the image, the bearing from the camera to the feature is well-known (this assumes accurate knowledge of intrinsic calibration parameters of the camera). As the camera moves through the scene, these salient features are tracked while estimates of the camera pose are generated using the projection error of the predicted image feature locations versus the measured image feature locations. The algorithm presented in [Davison et al., 2007] operates at realtime rates (30Hz) on low power mobile processors with accuracies within a few centimeters over movement around a known one meter square area.

Stereo vision-based SLAM algorithms (e.g. [Takezawa et al., 2004, Garcia and Solanas, 2004, Diebel et al., 2004, Herath et al., 2006, Marzorati et al., 2007, Hogue et al., 2007a, Zhang and Negahdaripour, 2008]) use stereo cameras as ranging devices. Perhaps the first to use stereo in an online Visual SLAM algorithm was Davison and Murray [Davison, 1998, Davison and Murray, 1998, Davison and Murray, 2002]. Their approach to incorporating stereo into the SLAM algorithm is to extract salient features in the images, track their motion temporally and utilize stereo disparity extraction algorithms to estimate the 3D location of the salient features. These salient 3D features are then thrust into a traditional SLAM algorithm with an appropriate error model for the disparity algorithm utilized.

Stereo-vision algorithms are not without their problems. Errors in calibration can cause highly inaccurate results. Each camera has its own intrinsic calibration parameters including the focal length, lens distortion, and camera projection centers must all be known accurately. The extrinsic rotation and translation between cameras must be known in order to rectify the images for efficient image search. Well-known algorithms exist to pre-calibrate these parameters

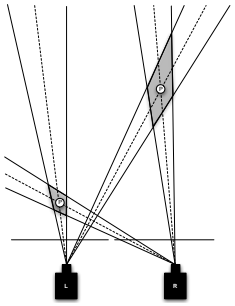


Figure 2.3: Stereo Error modeling illustration. The dotted lines show the intersected actual 3D points at two different depths. The shaded area is the uncertainty bound due to ± 0.5 pixel error. As can be seen, the error bound is not an ellipse but rather a diamond shape that is elongated along the Z-axis. At further depths, the non-symmetric nature of the error becomes more pronounced.

(e.g. [Tsai, 1987, Zhang, 2000]) and excellent treatments of error modeling for stereo vision can be found in [Matthies and Shafer, 1987] and [Murray and Little, 2000]. Another issue with stereo vision algorithms is the existence of depth-dependent errors. When estimating the range to a particular landmark, the light rays that project from a 3D point through the observed image locations for each camera must be intersected to obtain the 3D location. Due to the discrete nature of image projection onto the image sensor, the actual 3D location is bounded by a diamond-like error bound (see [Matthies and Shafer, 1987, Marzorati et al., 2007] and Figure 2.3). This error is typically modeled by a Gaussian, however a Gaussian model fails to capture the longer tail of the distribution. This is almost negligible for points near the cameras, however for more distant points, the tail becomes significant affecting the overall quality of the 3D location. One might think of the error as being map and location dependent since it changes depending upon the pose of the camera and the sensed structure of the environment.

Structure from motion is a vast research area whose goal is to extract metric measurements from image sequences or video. The most typical scenario uses feature extraction and tracking to evaluate the motion of the camera throughout the video sequence. Projective geometric constraints are applied to the feature tracks

and relative motion relationships, namely the Fundamental matrix and Trifocal tensor [Hartley and Zisserman, 2000, Faugeras and Luong, 2001], estimated to denote the relative pose and motion of the camera. The set of features are tracked through as many images as possible. The goal is to estimate the 3D structure (3D locations of the 2D features) and the pose of the camera per frame. The tracked feature set constitutes a (very) large constrained least-squares problem that, with appropriate assumptions, can be solved using a technique called Bundle Adjustment (see [Triggs et al., 2000]). This method essentially solves for the 3D locations of the features and the camera pose simultaneously using standard non-linear minimization algorithms, the most notable being the Levenberg-Marquardt minimization (see [Press et al., 2002]). If the calibration parameters of the camera are known, the 3D reconstruction can be upgraded to a metric reconstruction, or the calibration parameters can be computed through non-linear minimization as well (see [Pollefeys, 1999]).

The strengths of this approach include a basis on well defined and rigourous mathematical and geometric principles. Commercial software such as Autodesk Maya[3], and 2d3's Boujou[4] utilize this approach to accurately extract the camera motion for use in entertainment applications for inserting digital characters into captured video. Algorithmic approaches developed in the photogrammetry literature typically emphasize accuracy over processing speed. Although fully automated systems have been developed that estimate the camera motion, 3D feature locations, and models from video sequences (e.g. [Pollefeys, 1999, Nister, 2001]), these algorithms can take on the order of days to process relatively short sequences. Unfortunately, the computational requirements for these algorithms prohibit their use in robotic applications which require algorithms to provide accurate data in a timely manner for online processing.

[3]Autodesk, http://www.autodesk.com

[4]2d3, http://www.2d3.com

2.5.5 SLAM in large-scale environments

The above methods for addressing the SLAM problem are capable of maintaining valid representations of maps and vehicle pose for environments that are not very large, i.e. indoor applications and fixed size outdoor environments. Kalman filter approaches require that the state contains all of the feature information. In an exploration type of application, the number of landmarks are not known a priori. As the vehicle moves throughout the environment, the number of landmarks increases, thus increasing the size of the Kalman state vector. This poses a computational problem for large environments (i.e. areas larger than about 1000 square meters). Even though some could argue that they have addressed the issue of large scale environments for FastSLAM (e.g., [Hähnel et al., 2003]), any RBPF is bound to have issues with particle depletion and degeneracy (see [Kwak et al., 2007]). There has been much work attempting to address the problem of large-scale environments for SLAM. Perhaps the most effective is the utilization of a sub-map approach. The main idea here is that the world can be subdivided into many large (possibly overlapping) maps. One such algorithm, Atlas [Bosse et al., 2003], generates a hybrid topological-metric map. The advantage of this algorithm is the constant time update per landmark encountered. The algorithm keeps a bounded (or fixed) set of robot pose hypotheses that are used to determine which map is the best fit for the sensor data. As the robot traverses the environment it creates a graph of coordinate frames. Each node in the graph is a local frame of reference where the edges represent traversibility between adjacent frames. Each frame contains a metric map of the environment represented using traditional Kalman filter SLAM approaches. The algorithm uses an efficient map-matching algorithm to detect and close large loops. The local maps are used for reasoning about the world locally (i.e. obstacle detection, local path planning) while the topological map is used for larger scale reasoning. Hierarchical SLAM formulations have also been investigated in [Estrada et al., 2005].

Also related to large scale environments are scan-matching, or scan registration, approaches [Lu and Milios, 1994, Nieto et al., 2006a, Nieto et al., 2007]. The goal of scan registration is to determine the rigid transformation(s) that aligns two or

more sensor scans to minimize an objective error function. It is important to note that the transformation that aligns two scans is directly related to estimating the pose of the sensor that generated the data relative to the previous pose.

The iterative-closest-point algorithm (see [Besl and McKay, 1992]) is popular in the scan-matching literature. The algorithm assumes the existence of two dense sets of 3D points that have significant geometric overlap. The goal is to determine the six-degree-of-freedom rotation and translation that needs to be applied to one scan to minimize the error between the sets of points. The algorithm is iterative and in general requires many iterations to converge, however in practice it can be employed quite effectively if the sensor motion is small.

Alternative methods for scan matching were developed in [Lu and Milios, 1994] and also in [Schiele and Crowley, 1994]. These algorithms iteratively match line segments extracted from the range data with the map to determine correspondences and minimize the global registration error. The major problem with these approaches is the need to iterate over the entire trajectory and incorporate all scans since its goal is to minimize the global error.

2.6 Sensor modelling for SLAM

For the purposes of this thesis, a sensor is defined as an electronic device that measures a physical entity. There are many types of sensors used in robotics literature including acoustic ranging (sonar), laser range finders, cameras, accelerometers, rate gyroscopes, magnetometers (compass), optical joint encoders, GPS units, thermal arrays, and infrared sensors. See Figure 2.4 for examples of range sensors and Figure 2.5 for examples of orientation sensors commonly found in robotic systems. More details of robotic sensors can be found in [Everett, 1995]. It is necessary to transform the raw output of each sensor into a usable value. This requires several types of parameters, such as bias, gain, multiplicative factors, which when set accurately allows the algorithm to transform the raw output into a value that is meaningful. The process of determining these parameters is called calibration. Typically, the calibration process is performed off-line prior to using the sensor.

(a) SICK LMS-200

(b) MESA Imaging SR-3000

(c) Point Grey Bumblebee2

(d) Point Grey XB3

Figure 2.4: Examples of range sensors commonly used in robotics. (a) Laser Scanner courtesy of SICK AG (http://www.sick.com) (b) LIDAR Time of Flight courtesy of MESA Imaging (http://www.mesa-imaging.ch) (c)-(d) Stereo vision courtesy of Point Grey Research Inc. (http://www.ptgrey.com/).

When using multiple sensors fixed together, other parameters such as the rigid-body transformation necessary to bring the sensors into a common reference frame must be known. It is impossible to know the values of the calibration parameters perfectly. It is typical in SLAM to think of sensors as a black-box, where the underlying principles and internal parameters are unknown. This is true in many robotic applications that use a laser range-finder for localization and mapping. Laser range finders provide very accurate data, however it is a common misconception that the data can be used without conditioning. It is advantageous to the algorithm to take into account a statistical error model (determined through empirical means) to gauge the error of each measurement [Thrun et al., 2005]. Other difficult to measure factors influence the accuracy of the data. For example, in the case of laser range-finders, the type of material that the laser is shining upon affects the quality of the measurements. Transparent materials such as glass and reflective surfaces such as mirrors cannot be measured properly. The incidence angle of the laser beam to the surface is important as well; the reflected light pulse may not be detected properly if the incident angle falls outside of some range around the local surface normal.

In the SLAM literature, the measurements are typically assumed to be biased by a white, zero-mean Gaussian noise process. This assumption works in practice in many situations however when the measurement noise process violates the Gaussian assumption, is temperature dependent, time-varying, or position dependent, Kalman filter-based algorithms are prone to failure. There have been efforts to support more general sensor models. For example in [Marzorati et al., 2007], the sensor process is modeled using a particle filter allowing the estimated error of the sensor to take on a non-Gaussian process that is more representative of the actual values.

To characterize the uncertainty of a sensor, an appropriate probabilistic model must be used and known. In [Thrun et al., 2005], a sensor model of a general range sensor is developed. The model is applicable to laser range finders and other ranging devices. The probabilistic model is a combination of four distributions:

- A Gaussian distribution p_{hit} which denotes the uncertainty of a range mea-

(a) Inertiacube2+

(b) MotionNode

(c) Xsens MTi

(d) Xsens MTiG (IMU with GPS)

Figure 2.5: Examples of 3DOF orientation sensors commonly used in robotics. (a) courtesy of Intersense (http://www.isense.com) (b) courtesy of MESA Imaging (http://www.motionnode.com) (c)-(d) courtesy of Xsens Technologies B.V. (http://www.xsens.com/).

surement centered around the reported range value. The mean and covariance to this distribution must be identified through a calibration phase.

- An exponential distribution p_{short} which denotes the probability of sensing an unexpected object which reduces with range to the sensor, i.e. the sensor is more likely to hit an unexpected object if the sensed range is far from the sensor.

- A uniform distribution p_{max} denoting a missed measurement due to specular reflections or sensor failure.

- A uniform distribution p_{rand} which denotes other random noise that affects the measurement.

These four distributions are mixed with appropriate weighting parameters. The weighting parameters, as well as the intrinsic parameters for each distribution, are computed through an iterative maximum likelihood (ML) estimation process, i.e. learned from data offline. This may be effective for stationary error models, however any non-stationary noise source will be detrimental to the algorithm. An example of such an error source is a location-dependent error, i.e., the sensor functions poorly in specific but unknown areas of the environment.

More formally, Thrun models a general range finder sensor measurement model for time t and measurement k as a mixture of the following pdfs

$$p(z_t^k|s_t,\Theta) = \begin{pmatrix} w_{\text{hit}} \\ w_{\text{short}} \\ w_{\text{max}} \\ w_{\text{rand}} \end{pmatrix} \cdot \begin{pmatrix} p_{\text{hit}}(z_t^k|s_t,\Theta) \\ p_{\text{short}}(z_t^k|s_t,\Theta) \\ p_{\text{max}}(z_t^k|s_t,\Theta) \\ p_{\text{rand}}(z_t^k|s_t,\Theta). \end{pmatrix}$$

Here, the output of the measurement model is a weighted sum of the pdfs with weights $w_{\text{hit}} + w_{\text{short}} + w_{\text{max}} + w_{\text{rand}} = 1$. The probability functions above are

defined as

$$p_{\text{hit}}(z_t^k|s_t,\Theta) = \begin{cases} \eta\mathcal{N}(z_t^k; z_t^{k*}, \sigma^2) & \text{if } 0 \le z_t^k \le z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

$$p_{\text{short}}(z_t^k|s_t,\Theta) = \begin{cases} \eta\lambda_{\text{short}} e^{-\lambda_{\text{short}} z_t^k} & \text{if } 0 \le z_t^k \le z_t^{k*} \\ 0 & \text{otherwise} \end{cases}$$

$$p_{\max}(z_t^k|s_t,\Theta) = I(z = z_{\max}) = \begin{cases} 1 & \text{if } z = z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

$$p_{\text{rand}}(z_t^k|s_t,\Theta) = \begin{cases} \frac{1}{z_{\max}} & \text{if } 0 \le z_t^k < z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

The pdf for $p_{\text{hit}}(z_t^k|s_t,\Theta)$ is a Gaussian distribution centered around the mean z_t^{k*} (which is computed by raycasting from s_t within the map Θ) with standard deviation σ^2. The normalizer η in this case is the integral of the distribution over the range of measurments or

$$\eta = \left(\int_0^{z_{\max}} \mathcal{N}(z_t^k; z_t^{k*}, \sigma_{\text{hit}}^2) dz_t^k \right)^{-1}.$$

The pdf for $p_{\text{short}}(z_t^k|s_t,\Theta)$ is an exponential distribution that takes into account short-range dynamic obstacles. The likelihood decreases with distance and is parameterized by λ_{short} for specific sensors. The normalizer η is computed as

$$\eta^{-1} = 1 - e^{-\lambda s_t^{k*}}.$$

Dealing with sensor failures is handled by the pdf $p_{\max}(z_t^k|s_t,\Theta)$ as a point-mass distribution centered around the maximal range. In laser range finders, if the sensor provides erroneous data the maximum value is generated and passed back to the calling function. Finally, the pdf $p_{\text{rand}}(z_t^k|s_t,\Theta)$ is used to explain unexplainable measurements and is denoted as a uniform distribution over the entire measurable range.

Camera calibration

The goal of camera calibration is to determine the parameters necessary to describe the projection of a 3D point to a 2D image location. The world is inherently three-

dimensional. Cameras provide us with a mechanism to capture an instant of time in a two-dimensional form. This is accomplished through projection through a lens onto a CCD or other type of sensor. Cameras and lenses are complex and require several parameters to be defined accurately in order to use the information for metric purposes. Parameters important to using cameras include the focal length, the radial distortion parameters, the skewness of the sensor, and the location of the true image center. For a single camera system, these parameters are sufficient to begin to utilize images for metric evaluation of distances between objects (see [Criminisi et al., 2000]). Projection of a 3D point, (X, Y, Z), onto a pinhole camera image plane is described by a projection matrix that converts 3D world points into 2D image points, (u, v) as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P_{3\times4} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.8}$$

The projection matrix encodes both intrinsic and extrinsic parameters of the camera. Examples of intrinsic parameters typically used are the focal length (f_x, f_y), image center (c_x, c_y), and skewness factor (α). Extrinsic parameters describe the orientation of the camera coordinate frame relative to the world coordinate frame, e.g. rotation matrix $(R_{3\times3})$ and translation vector $(\vec{t}_{3\times1})$ that denote the camera's orientation. For the context of this thesis, these operations can be defined as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R^T_{3\times3} & -R^T_{3\times3}\vec{t}_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.9}$$

Once the 3D world point is transformed into the 2D image plane, lens distortion must be corrected. Physical lenses introduce abberations and distrotions to the resulting image points. It is important to correct these distortions when extracting metric information using single-view photogrammetry, or multi-view stereo algorithms. Commonly, radial and tangential distortions are corrected using the fol-

lowing model. Here $x_n = \begin{bmatrix} x \\ y \end{bmatrix}$ is the undistorted point projected using the pinhole model. The distorted point that is viewed (x_d) can be described by

$$x_d = \begin{bmatrix} u_d \\ v_d \end{bmatrix} = (1 + K1r^2 + K2r^3 + \ldots)x_{\mathrm{n}} \tag{2.10}$$

where $r = x^2 + y^2$ is the radial distance from the image center. There are more complex models for radial distortion that includes tangential distortions and more parameters such as those found in [Heikkila and Silven, 1996, Bouguet, 1999], however for this thesis it is sufficient to describe the distortion completely with the first two radial parameters $K = [K1, K2]$. Applying these parameters is performed by pre-computing an inverse warp look-up table and performing bi-linear interpolation.

Developing robust algorithms to estimate these parameters is a research field on its own but the problem is largely considered to be "solved." Many off-line algorithms exist to provide estimates of these values, namely [Abdel-Aziz and Karara, 1971, Tsai, 1987, Bouguet, 1999, Zhang, 2000]. Multi-camera self-calibration algorithms such as [Hemayed, 2003, Svoboda et al., 2005] also exist.

Refraction

When cameras view environments through a lens into a different medium such as water, light rays tend to bend upon entering/exiting the interface between the mediums. Underwater vision applications are plagued by this problem since you must place the camera in a watertight housing that is then placed underwater. Light rays viewed by the camera must first exit the water make its way into the air surrounding the lens prior to activating the light sensitive CCD in the camera. When light crosses into a different medium, it will bend depending on its angle of incidence described by Snell's Law (see Figure 2.6) which can be described as

$$\eta_1 \sin(\theta_1) = \eta_2 \sin(\theta_2) \tag{2.11}$$

where η_1, η_2 are the index of refraction of both media and θ_1, θ_2 are the angles of the light ray to the normal of the interface between the two media. Thus, given

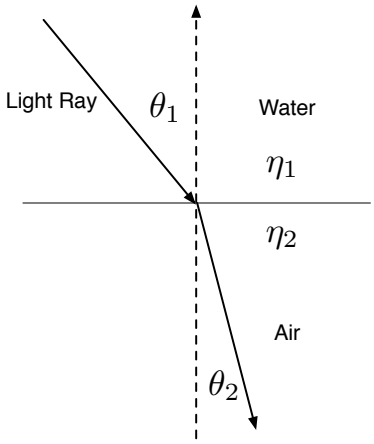


Figure 2.6: Snells' Law. As light enters a new medium, the angle of the light ray changes depending on the ratio of the index of refraction of both media. Diagram shown assumes that light is coming from the underwater medium and entering an air filled tube containing a camera.

the angle of incidence in one medium, the resulting angle may be computed as

$$\theta_2 = \sin^{-1}\left(\frac{\eta_1}{\eta_2}\sin(\theta_1)\right) \tag{2.12}$$

It can be seen that there are a couple of ways this affects the image projection model. First, as points at the same depth from the camera move away from the camera axis (and water-air interface), the angle of incidence will change. Thus the light will refract differently depending on its distance from the center of projection. This creates a radial like distortion near the edges of the image. Second, as the objects change depth, the angle of incidence will also change. Thus the light will also refract differently depending on its distance causing another radial-like distortion (see [Kwon, 1999b]). If the depth from the camera didn't affect the angle of incidence, this would not be problematic for image analysis, however this added distortion is not easily corrected optically nor mathematically. In Figure 2.7, the effects of refraction on image formation can be seen when light travels from the water medium to air before being imaged. Note that these scenarios are

for a single image only. In a stereo-vision situation, the refraction errors become even more dangerous since for a single world point there are two different distortions due to the refraction since there are two centers of projection. Methods for calibrating the distortion have been developed for underwater stereo systems (see [Harvey and Shortis, 1998] for details). The system however utilizes a large control frame whose measurements are precisely known (approximately 2 meters by 2 meters by 1 meter). The frame is required into the viewing volume of the stereo system and rotated to at least four different orientations requiring at least one other diver for calibration deployment. The distortion is approximated as a radial distortion in the image plane which minimizes the error for that particular volume. Similarly, in [Kwon, 1999a, Kwon and Lindley, 2000], a control-frame is used in the viewing volume to approximate the distortion, however as stated in [Kwon, 1999b], using a fixed radial distortion model is erroneous since the amount of distortion is dependent upon object location thus a direct linear transform (DLT) method is used instead.

2.7 Open problems in SLAM

Simultaneous localization and mapping is a highly active field of research. The basis of successful real-time SLAM algorithms is a probabilistic framework which is appropriate since uncertainty is inherent in any robotic system. In the approaches to SLAM described previously, a number of common issues are apparent. The Markovian assumption, and most importantly in terms of this thesis, the measurement noise models for the sensor is assumed to be fixed (stationary) and known prior to the experimental trials. In general, sensor modelling is a challenge that needs to be addressed for SLAM. While in typical situations the standard models work well, in reality there are many situations where these assumptions are violated. Environments such as outer-space or underwater have characteristics that affect sensors in ways that are not modelled well by a Gaussian noise function. It is possible that the sensor used to observe the world changes its parameters over time, or that the current environment map and robot pose influences either the characteristics

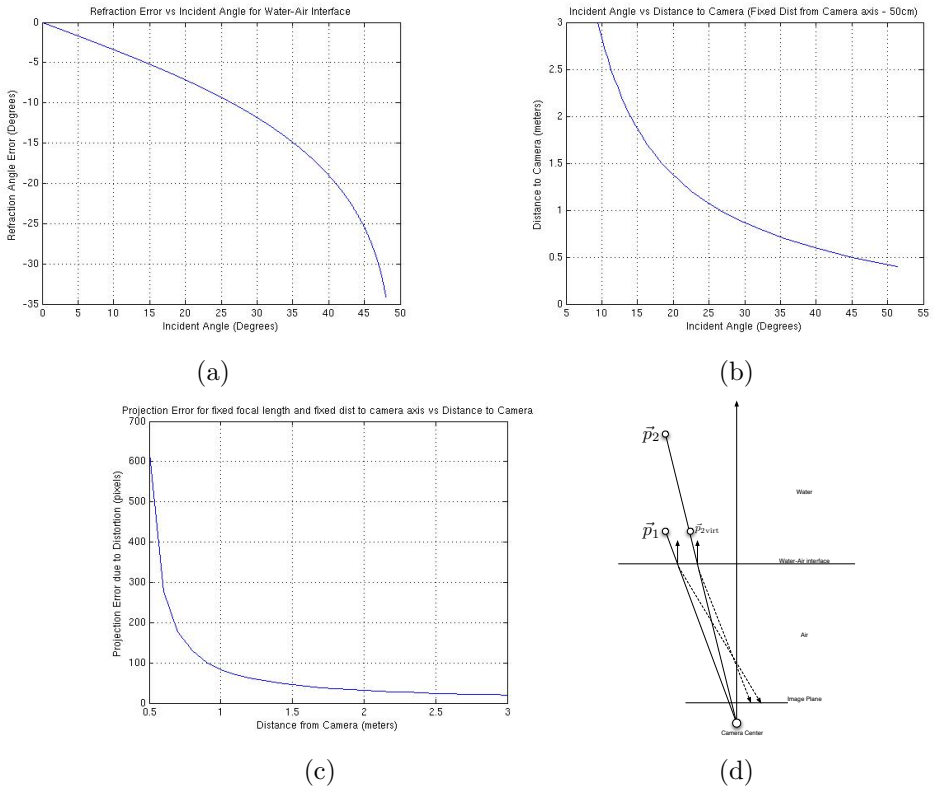


Figure 2.7: Refraction Effects on Imaging. (a) Difference between incident and refracted light rays as a function of incident angle. (b) Shows how the incident angle changes with camera depth for a fixed distance from the camera axis. (c) Shows the error in pixels between the actual projection (using a pinhole model) and the projection of the refracted ray onto the image. As can be see in (c), the projection error ranges from 20 to 600 pixels in the image depending on the depth of the point from the camera center. (d) Refraction of points at different depths from the camera

of the sensor or aid in extracting information from the sensor itself. The lack of a model to account for sufficient malleability in sensor calibration parameters in SLAM can cause failure of the algorithm if the sensor parameters do not fit the assumed error model. As a specific real-world example, consider the existence of magnetic field distortions in the environment and their effect on a magnetometer contained in typical 3DOF IMU sensors which contain accelerometers, gyroscopes and magnetometers.

A common sensor used for earth-based navigation is the compass or magnetometer [Everett, 1995]. Magnetometers measure the strength of the horizontal component of the earths' magnetic field to provide a heading direction. The output of the sensor is a vector denoting the direction to magnetic north. Magnetometers are prone to local distortions due to changes in the local magnetic field that arise from the presence of ferrous material in the vicinity, electrical motors (abundant in robotic applications), electrical cabling, or any other magnetic object.

If the compass is to be used in known locations, the local distortions can be determined through an intensive pre-calibration phase. For navigation in an unknown environment this is not possible. This is especially true in man-made indoor environments where the existence of magnetic material is plentiful. Outdoor environments are less prone to distortion however the existence of dynamic man-made objects such as automobiles or boats can bias the measurement. In [Skvortzov et al., 2007] the authors provide a mechanism to utilize the magnetometer measurements to determine whether the measurement is being affected by local distortions or not. This type of information could be valuable to any robotic platform relying on its compass for heading. Without the knowledge of the local magnetic distortions, SLAM algorithms that do not take this information into account will undoubtedly be inconsistent with the actual environment being traversed since the errors in the magnetometer readings bias the robot orientation estimate, thus through integration these errors will accumulate.

Figure 2.9 and Figure 2.8 illustrate the effect a magnet has on the output of a commonly used 3DOF orientation sensor that uses magnetometers to determine the north direction. The maximal yaw error is upwards of 200 degrees making the output of the orientation sensor entirely unusable at this location. Note that the

error distribution is not elliptical at any of the spatial locations, and also note that the shape is different depending on where the IMU is placed. The noise model is not well-modeled by a Gaussian. The experiment performed was to place the IMU at various locations on a grid and record the orientation estimate provided for a few seconds. The IMU was placed at 4 distinct orientations at each grid point, i.e. at 0, 90, 180, 270 degrees. The data was collected with and without a magnet present in the environment. The magnet was stationary for each data point and was placed at a location $(-6.4, 25)$cm from the origin. To visualize the error, the error between the magnet and non-magnet data was computed for each grid location and a unit vector representing the orientation was scaled by the computed error. A cubic polynomial was fit through the four data points at each grid location and is drawn to show the effect of the magnet with relation to sensor position and orientation. As can be seen, the error is non-Gaussian, not symmetric, and can be quite severe as the sensor moves close to the distortion. This non-Gaussian error can wreak havoc on SLAM algorithms that assume Gaussian noise models and will clearly bias the resulting orientation estimate.

2.8 Summary

There has been a vast amount of research in the robotics community related to simultaneous localization and mapping. The SLAM problem has been formulated in various ways, most successfully using a stochastic formulation. A Bayesian approach is a general way to represent the SLAM problem in terms of probability density functions. Using various common assumptions such as Gaussian models to represent the pdf's allows for the use of the Kalman filter recursive algorithms to solve for both the map and robot location simultaneously. Various adaptations have been studied, such as the use of particle filters — as in DP-SLAM — to represent the pdf's more generally and hybrid solutions such as FastSLAM have been successful in addressing the problem on land.

The underwater domain presents various challenges for SLAM such as the in-

Figure 2.8: Inertiacube^2 orientation error when positioned near magnet. The IMU orientation was sampled at each location in the grid at 4 distinct orientations (0,90,180,270 degrees) with and without a magnet placed at one of the grid corners. The error was calculated and a cubic-polynomial is fit through the scaled orientation vectors to show the estimated error as a function.

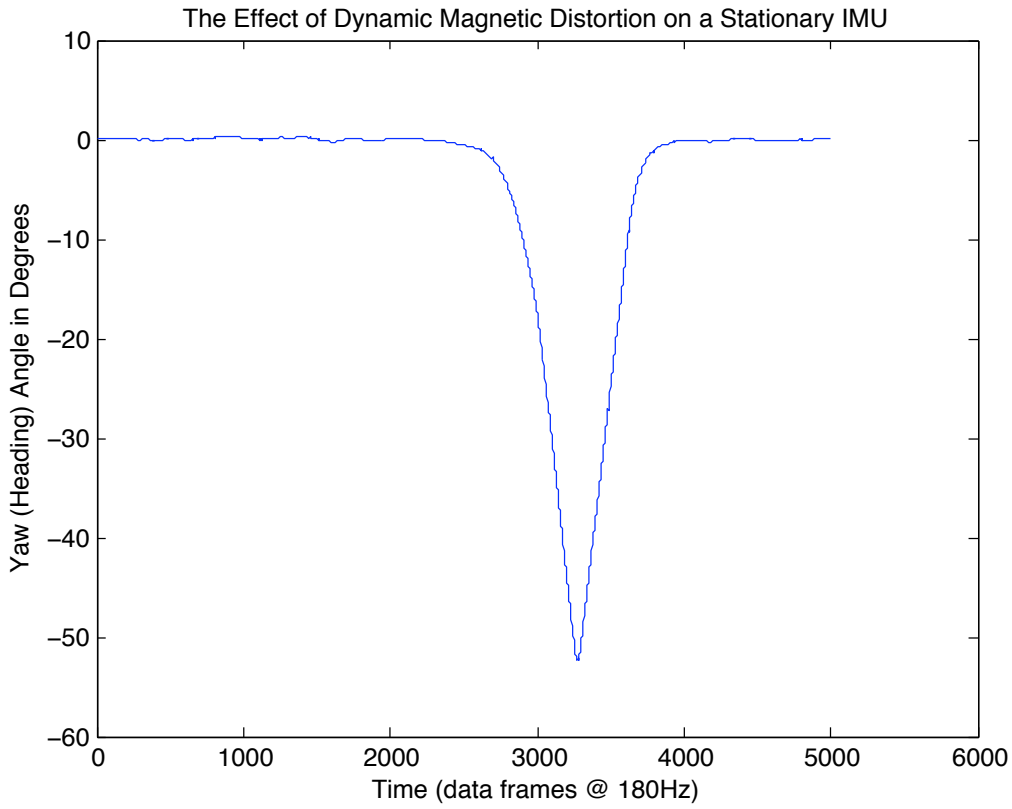


Figure 2.9: The effect of a spatially-varying magnetic source on the reported yaw value of a stationary Intertiacube^2 as it comes near to the IMU and passes it. The graph shows the yaw angle reported back from the IMU over time. Note that for the duration of this experiment, the IMU was stationary. The magnetic source was a hard drive magnet. The reported error peaks at around -53 degrees for this experiment.

troduction of position or map dependent non-stationary noise sources. Without the proper modelling of these sources, traditional algorithms may diverge when mapping in the underwater environment. Additional techniques must be studied to address the interaction between the sensor errors, the pose, and the map. The following chapter describes a stereo video sensor system that was developed to examine the interaction between sensor error, pose, and the map in terrestrial and underwater domains.

CHAPTER 3

A sensor for 3D surface modelling

This chapter investigates the use of stereo vision and inertial sensing in an underwater setting. Both land-based and underwater experiments are provided to show the effectiveness and accuracy of ego-motion estimation using stereo-vision in these domains, and to illustrate the need for proper calibration of the sensor system. A discussion of the development of the AQUASENSOR hardware platform details some of the difficulties in underwater field-robotics. Selected material in this chapter has been published previously in [Jenkin et al., 2008, Hogue et al., 2007a, Jenkin et al., 2007, Hogue and Jenkin, 2006b, Hogue et al., 2006a] and land-based applications based on this work have been published in [Hogue et al., 2007c]. A natural sensing choice for autonomous scene reconstruction is to use cameras and computer vision techniques. In the terrestrial domain, sensing technologies such as stereo vision coupled with good vehicle odometry have been used to construct 2D top-down maps and 3D models of the environment. The lack of such predictable vehicle odometry in the underwater domain necessitates solutions which are more dependent upon sensor information than is traditional in the terrestrial domain. This has prompted recent research in robotic vehicle design, sensing, localization and mapping for underwater vehicles (see [Eustice et al., 2005a, Eustice, 2005, Pizarro et al., 2004, Williams and Mahon, 2004b]).

3.1 Sensor hardware design

The hardware design goal for the stereo-video platform (known as the AQUASENSOR) was to construct a compact, fully-contained unit that is sufficiently portable to be used by a single diver and whose components, suitably repackaged, could later be integrated within the AQUA robot [Dudek et al., 2007]. Range information extracted from the stereo video is integrated with 3DOF orientation from an

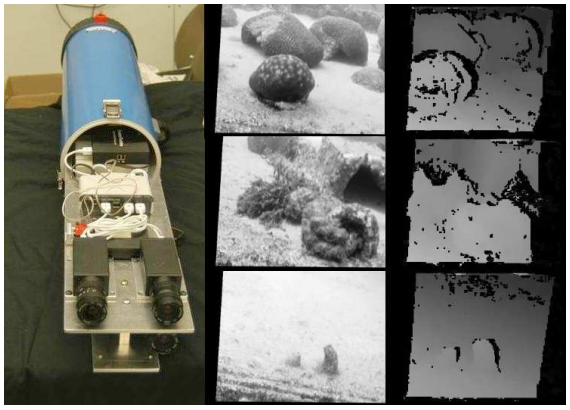


Figure 3.1: AQUASENSOR V1.0. Trinocular stereo rig and example reference images with corresponding dense disparity.

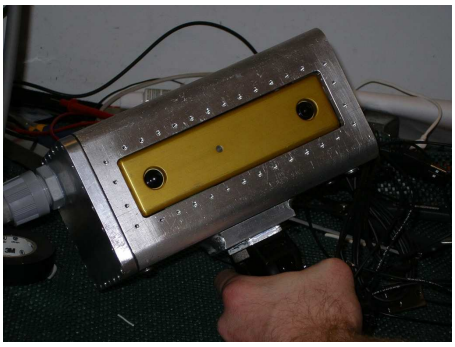
inertial measurement unit (IMU) to simultaneously estimate a dense 3D model of the environment and the sensor trajectory. It is important to note that although the AQUASENSOR does not process the collected data in real-time, the algorithms used to analyze the data were developed with real-time performance in mind.

Developing a sensor for underwater usage is a non-trivial task. There are many challenges for such a system; although ensuring the electronics are protected by a watertight housing is perhaps the most critical. During the course of this work multiple camera housing designs were built and tested, the details of each housing can be found in [Hogue et al., 2006a] and see Figures 3.1 and 3.2 for images of each.

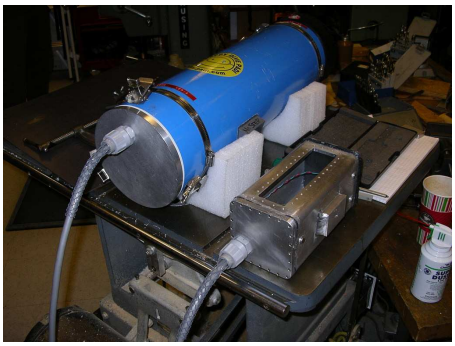
AQUASENSOR V3.0

The third and most recent design of the sensor can be seen in Figure 3.3. A single tube was purchased to house all of the components. Learning from previous hardware designs, smaller and components with lower power requirements were used. An embedded board and lower power cpu was used which reduced the amount of heat it produced alleviating much of the instability issues. A fan was strategically

(a)



(b)



(c)

(d)

(e)

Figure 3.2: AQUASENSOR V2.0 (a-c) and V2.1(d-e). (a) Hand-held unit (b) Hand-held unit with Bumblebee (c) Complete unit (d) Final system (e) Closeup of revised Hand-held unit

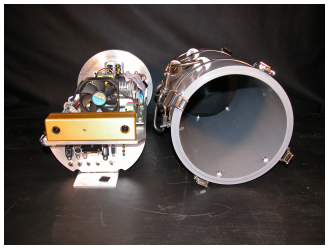
placed inside to circulate the air with the intention of moving hot air away from the CPU and hard disks. This reduced the number of pockets of hot air that would concentrate over the components causing them to fail. The final tube is shorter in length than Version 1.0, fully self contained and is extremely maneuverable by a single diver. In fact, it has been deployed several times by a single diver throughout the course of this work.

3.2 Data processing

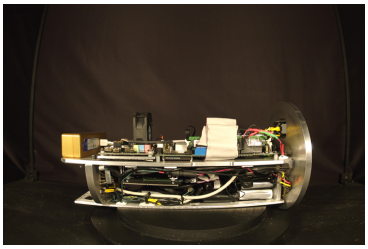
Upon returning to the surface, data from the AQUASENSOR is offloaded to higher performance computers and a larger disk array for processing[5]. Recovering accurate depth from stereo imagery is performed by leveraging the optimized sum-of-absolute-differences (SAD) algorithm implemented in the Point Grey Triclops SDK[6]. This provides a dense set of 3D points per acquired image frame. To estimate the motion of the camera, 3D point sets from different times are registered into a common reference frame. The 6DOF position and orientation (pose) of the sensor is estimated by utilizing the inter-frame 2D image motion and integrating these motion estimates over time. This is accomplished by tracking "interesting" features temporally creating a large set of 2D features containing outliers and errors. This set of 2D feature tracks is pruned to contain only features that correspond to 3D locations. This results in two clouds of 3D points, one from the current frame and another from the previous frame. The 2D feature tracks are used to estimate the registration of these point clouds. A 6DOF rigid-body transformation is estimated between these point clouds using a least-squares algorithm. These delta poses computed per frame are integrated to provide an estimate of the final pose of the sensor. This also means that any error in the estimate will accumulate over time. Also, since the intrinsic calibration parameters for the cameras are not known perfectly, any error caused by calibration error will also accumulate causing the tra-

[5]Throughout this work, the basic stereo-vision algorithm evolved into the version described here. Earlier versions of the AQUASENSOR used slightly simpler versions of the algorithm.

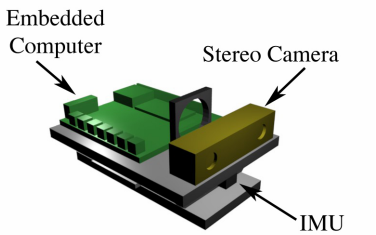
[6]http://www.ptgrey.com



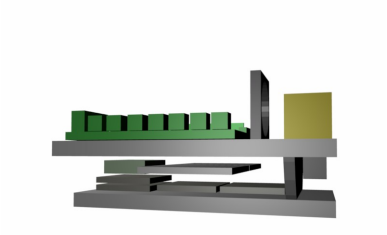
(a) Sensor with housing



(b) Side view of sensor



(c) Schematic



(d) Side view of schematic

(e) Sensor deployed underwater

Figure 3.3: AQUASENSOR V3.0. (a) Front view with underwater housing, (b) Side view of sensor, (c) and (d) show a schematic representation of the device and (e) shows the sensor in use scanning a sunken barge.

jectory to diverge from reality. To reduce this error, a 3DOF inertial measurement unit is used to provide independent complimentary information about the relative orientation of the device between temporal frames.

3.2.1 Visual Ego-motion estimation

First, "good" features are extracted from the reference camera at time t using the Kanade-Lucas-Tomasi feature tracking algorithm (see [Shi and Tomasi, 1994, Lucas and Kanade, 1981]) and are tracked into the subsequent image at time $t+1$. Using the disparity map previously extracted for both time steps, tracked points that do not have a corresponding disparity at both time t and $t+1$ are eliminated. Surviving points are subsequently triangulated to determine the metric 3D points associated with each disparity. Depending on the characteristics of the stereo sensor, a gate or threshold based on the individual 3D point uncertainty (as defined in [Matthies and Shafer, 1987]) may be used to further prune the 3D points.

In underwater scenes, many objects and points are visually similar and thus many of the feature tracks will be incorrect. Dynamic illumination effects, aquatic snow, and moving objects (e.g. fish) increase the number of spurious points that may be tracked from frame-to-frame. To overcome these problems, robust statistical estimation techniques are employed to label the feature tracks as belonging to either a static or non-static world model. This is achieved by estimating a rotation and translation model under the assumption that the scene is stationary. The resulting 3D temporal correspondences are associated with stable scene points for the basis of later processing. A further optimization would be to implement an algorithm such as Epiflow [Zhang and Negahdaripour, 2008] which aids in feature tracking by predicting the epipolar geometry for underwater imaging.

The camera orientation is represented as a quaternion and the position is represented as a 3D vector from the initial frame of reference. A least-squares best-fit rotation and translation is computed for the sequence in a two stage process. First, RANSAC [Fischler and Bolles, 1981] estimates the best linear least squares transformation using Horn's absolute orientation method [Horn, 1987] and is similar to the method presented in [Michaels and Boult, 1999]. Given two 3D point clouds

r_{t_0} and r_{t_1} at times t_0 and t_1 respectively, the rotation and translation required to bring r_{t_1} into accordance with r_{t_0} are estimated. The centroid, $\bar{r}_{t_0}, \bar{r}_{t_1}$, of each point cloud is computed and subtracted from the points to obtain two new point sets, $r'_{t_0} = r_{t_0} - \bar{r}_{t_0}$ and $r'_{t_1} = r_{t_1} - \bar{r}_{t_1}$

To compute a function representing rotation, denoted as $R(\cdot)$ since it may simply be a matrix multiplication or a more complex operation such as a quaternion rotation operator, the error function

$$\sum_{i=1}^{n} ||r'_{t_0,i} - sR(r'_{t_1,i})||^2 \tag{3.1}$$

is minimized. The rotation, $R(\cdot)$, and scale, s, are estimated using a linear least-squares approach (detailed in [Horn, 1987]). After estimating the rotation, the translation is estimated by transforming the centroids into a common frame and subtracting.

Noting that the above method is prone to an imperfect result, the final step is to refine the rotation and translation simultaneously using a nonlinear Levenberg-Marquardt minimization [Press et al., 2002] over six parameters. For this stage the rotation is parameterized as a Rodrigues vector [Weisstein, 2006] and the rotation and translation parameters are estimated by iteratively minimizing the transformation error

$$\sum_{i=1}^{n} ||r_{t_0,i} - (R(r_{t_1,i}) + T)||^2 \tag{3.2}$$

In practice, the minimization takes only a few iterations to reduce the error to acceptable levels and does not necessarily preclude realtime operation. This approach to pose estimation differs from the traditional Bundle-Adjustment approach [Triggs et al., 2000] in the structure-from-motion literature in that it does not refine the 3D locations of the features as well as the trajectory. The 3D structure is not refined during the minimization to limit the number of unknowns and thus provide a solution to the system more quickly.

3.2.2 IMU integration

Egomotion estimation via vision motion introduces at least two sources of error in the estimate of 6DOF pose. First, the point-cloud registration computed from feature tracks can never be perfect. As such, there is always a small residual error in the registration per frame which accumulates over time. Second, the intrinsic camera parameters are not known perfectly and any error in these parameters introduces an error in each 3D point. In particular, the radial distortion estimate of the lens is prone to error and the per-point error is non-uniform over the visual field. This can introduce an artificial surface curvature in the 3D point clouds which is subtle but noticeable when many frames are registered. This effect can be seen in Figure 3.4a. Here, the registration error is small and the points line up very well creating a visually "correct" model when viewed closely, however after registering many frames it can be seen that there is an introduced curvature to the recovered surface. To help counteract this effect, the 3DOF IMU is used to provide more information about the orientation of the device. The IMU provides a quaternion representing the absolute 3DOF orientation in 3D space. The change in orientation as computed from the visual system must be consistent with the change in the absolute orientation of the device. This is accomplished by transforming the IMU orientation change,

$$^{\text{IMU}}q_{\delta} = {}^{\text{IMU}}q^{c}_{t-1} * {}^{\text{IMU}}q_{t} \tag{3.3}$$

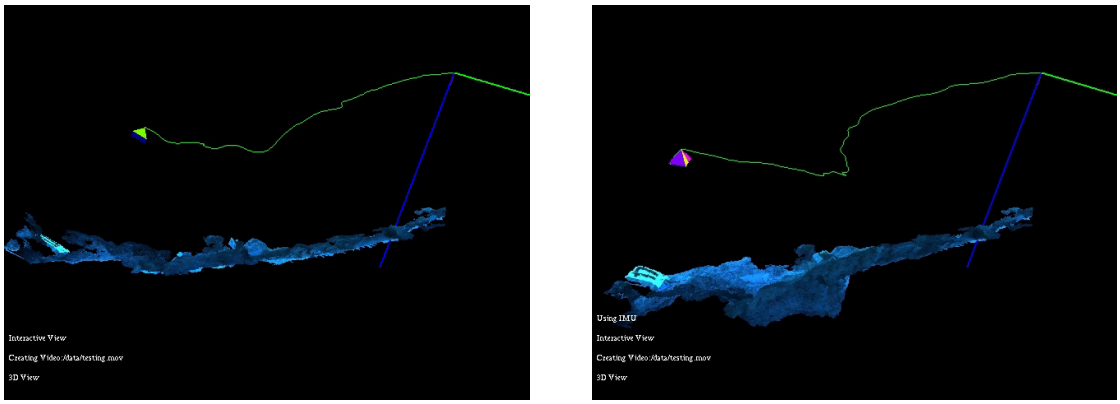
into the estimated camera frame,

$$q_{t} = {}^{\text{CAM}}q_{\delta}{}^{\text{CAM}}q_{t-1}{}^{\text{IMU}}q_{\delta} \tag{3.4}$$

and then performing the Levenberg-Marquardt minimization using this new pose as the initial guess. The effect of utilizing this information can be seen in Figure 3.4b where the curvature has been reduced in the resulting model.

3.2.3 Reconstruction algorithm

The reconstruction algorithm is summarized below. Given stereo images at time t and $t-1$,



(a) Visual motion only (No IMU)

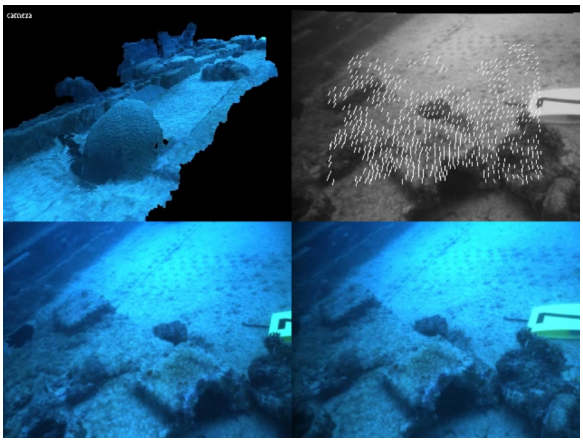
(b) Visual motion coupled with IMU

Figure 3.4: Resulting model with and without IMU information. Here, the 3D model of a 2 metre section of coral bed on a mostly planar barge is shown with the camera trajectory overlaid. It can be seen that the model in (a) exhibits significant curvature due to per-point error while (b) shows a significantly more planar model since the use of IMU information reduces the pose error incurred from the incorrect calibration parameters. Qualitatively, the camera trajectory in (b) also matches the real camera motion more closely.

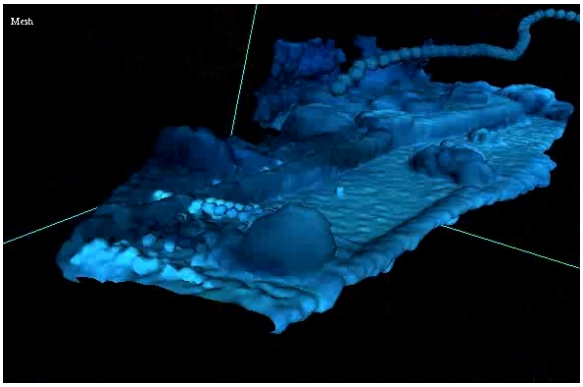
1. Perform the stereo disparity extraction algorithm and estimate 3D point cloud at time t
2. Track salient features in Left image of stereo pairs from time $t-1$ to t
3. Prune 2D feature set to only contain 2D feature matches with corresponding 3D points
4. Estimate 3D vision-only pose change using RANSAC, ${}^{\text{CAM}}q_{\delta}, T_t$
5. Compute ${}^{\text{IMU}}q_{\delta}$ and q_t as above
6. Refine q_t, T_t using Levenberg-Marquardt minimization for only a few iterations
7. Apply pose to point cloud and add to octree data structure
8. *Optional:* Extract mesh using a constrained elastic surface-net algorithm [Gibson, 1998] or the Marching Cubes algorithm [Lorenson, 1987].

3.3 Experimental Validation

Experiments have been performed to both evaluate the accuracy of the reconstruction system and to create 3D models of real-world objects in the field. Results from field experiments near Holetown, Barbados show the reconstruction of a coral bed and sections of a sunken barge lying in Folkstone Marine Reserve. Sample qualitative reconstructions from the underwater sequences are shown in Figure 3.5 and in Figure 3.6. Land-based reconstructions demonstrate the ability to use the sensor to reconstruct terrestrial scenes captured with 6DOF hand-held motion. To evaluate the accuracy of the sensor, experiments were performed and the recovered trajectory compared with ground truth trajectory data. Experiments were performed by estimating the handheld motion using the ego-motion algorithm described above coupled with ground truth motion gathered by an auxiliary sensor

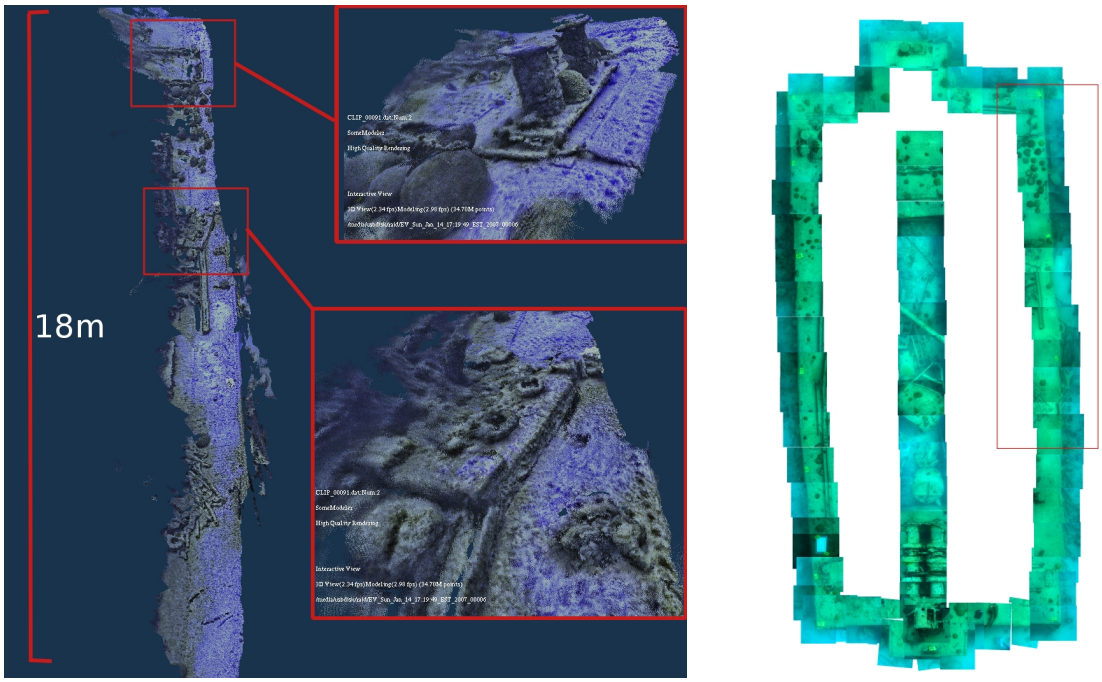


(a) Section of Barge



(b) Polygonal Mesh

Figure 3.5: (a) shows the algorithm in action. The upper left quadrant displays the 3D point cloud representation of the currently estimated model. The upper right quadrant shows the feature tracks for the current frame and the bottom displays the current stereo vision images. (b) shows the corresponding polygonal mesh extracted from the point cloud (untextured).



(a) Reconstruction of Sunken Barge

(b) 2D Mosaic of Sunken Barge

Figure 3.6: (a) shows the model of a large section (18 metres) of the barge as a point cloud with inset images to show close-up views of the 3D model that illustrate scale and detail. A 2D mosaic of the barge (created manually) is shown in (d) with the recovered section enclosed in a red box to place the 3D model in context.

(IS-900). In one experiment, a scene with a colleague sitting in a chair is reconstructed and in the second experiment, a more controlled setting with markers that were placed at a known distance was reconstructed. The resulting 3D models can be seen in Figure 3.7 and Figure 3.8. Two yellow markers were placed 63.5 cm apart on a platform and the sensor was moved in approximately a straight line from one marker to the other. Figure 3.8a and Figure 3.8b show the stereo pairs associated with the first and the last image of the sequence. To estimate the error of the reconstructed trajectory, an Intersense IS-900 tracking system was used to provide absolute data and computed the error on a frame to frame basis.

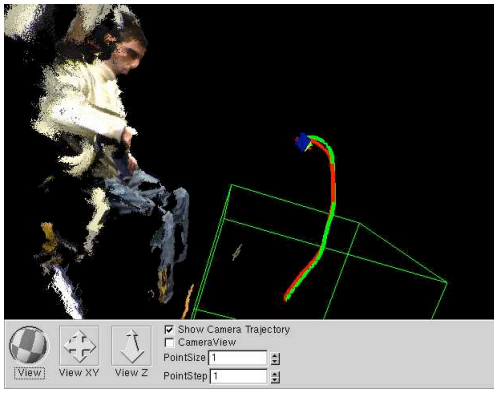
The offset between the IS-900 sensor and the camera coordinate frame was estimated manually to bring the measurements into alignment. The Euclidean distance between the 3D camera positions of the vision-based reconstruction and the IS-900 absolute position was used as an error metric and the resulting error is plotted in Figure 3.9 for the two reconstructions. In the first error plot, the mean error was found to be 2.1cm and in the second the mean error was 1.7cm. The error is computed per frame with the euclidean distance as an error metric, namely

$$\text{err} = \sum ||(p_i - P_i)|| \tag{3.5}$$

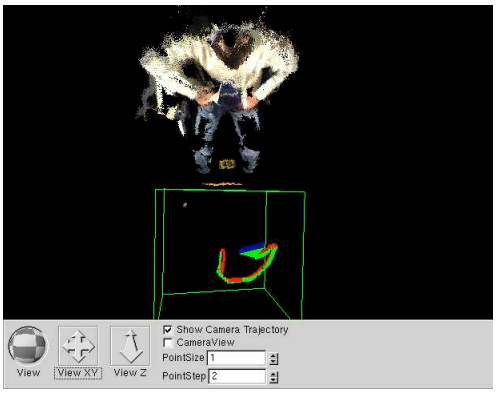
where $p_i = [x_i, y_i, z_i]$ is the position reported from our vision-based sensor and $P_i = [X_i, Y_i, Z_i]$ is the position reported by the IS-900 tracking system. The error reported is quite low over the experiment, however there is a drift that is not compensated for due to the manual alignment of the IS-900 frame.

3.4 Discussion

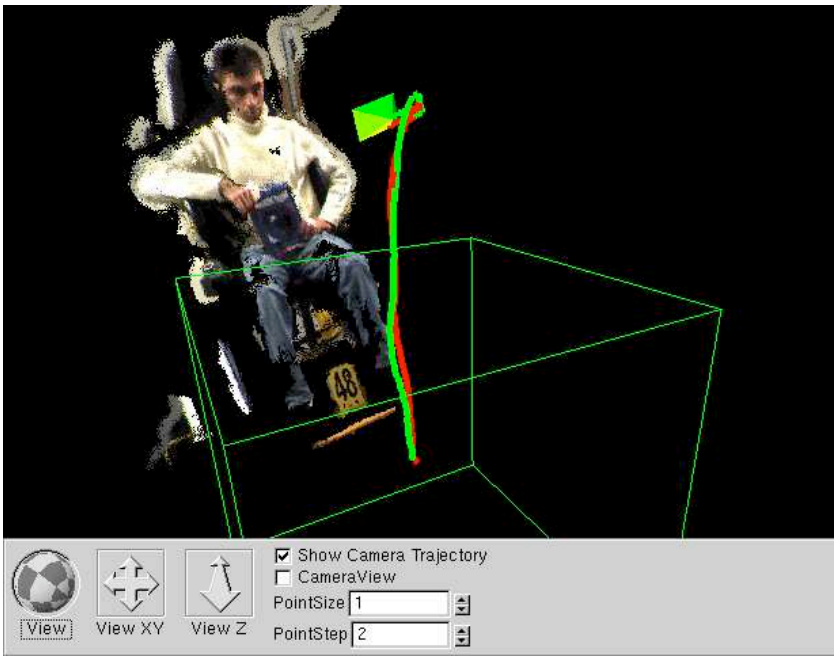
Traditional underwater sensing devices have relied on active sensors (sonar in particular) to recover three-dimensional environmental structure. Advances in stereo sensing and data fusion technologies demonstrates that passive stereo is a sufficiently robust technology to be applied in the aquatic domain as well. Although the underwater domain presents unique challenges to traditional vision



(a)



(b)



(c)

Figure 3.7: Land-based Experiments 1. (a-c) Shows the reconstruction of a scene with a colleague sitting in a chair. In the reconstruction, the computed trajectory is shown in green while the absolute IS900 trajectory is shown in red.

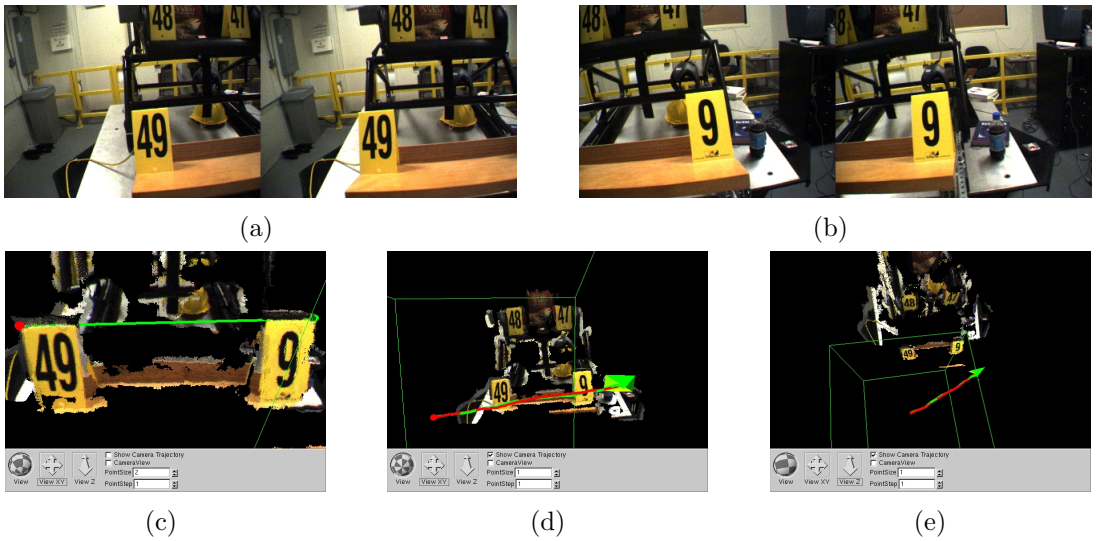
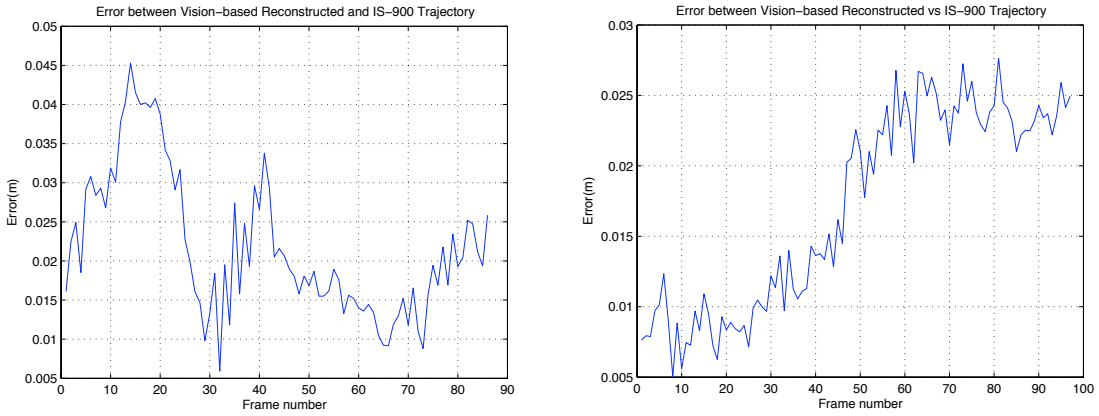


Figure 3.8: Land-based Experiments 2. (a-b) Show reference stereo images from two frames in the sequence. (c) shows the markers and two measured points with a line drawn between them (in green). The distance between these points was manually measured to be 63.5cm while the vision-based reconstruction reported a distance of 63.1cm. (d-e) Show the reconstruction of the scene with the markers placed within it. (f-h) show a second sequence of a colleague sitting in the chair. In all reconstructions, the computed trajectory is shown in green while the absolute IS900 trajectory is shown in red.



(a) (b)

Figure 3.9: Error plots for two trajectory reconstructions. The euclidean distance error (plotted in meters) is shown over time (in frames). The error was computed between the absolute position reported by the IS-900 tracking system and the camera position reported from our vision-based reconstruction algorithm. There is a slow error drift over time due to imprecision in the manual calibration between the IS-900 sensor and the stereo coordinate frames.

algorithms, robust noise-suppression mechanisms can be used to overcome many of these difficulties. Results from land-based experiments demonstrate the accuracy of our reconstruction system. Results from underwater field trials demonstrate the system's ability to reconstruct qualitatively correct 3D models of the aquatic environment. The entire reconstruction system operates at 2-4 frames per second (depending on the stereo algorithm and feature tracking parameter settings). The current speed limitation is primarily due to the stereo algorithm and feature tracking which could be implemented in graphics hardware at near real-time rates. Stereo disparity extraction has been implemented on modern GPUs [Yang and Pollefeys, 2003, Yang et al., 2004], and the KLT and SIFT algorithms have also been accelerated by GPUs [Sinha et al., 2006]. Leveraging the GPU could increase the modeling frame-rate dramatically. Recovery of a long (18 metre) section of sunken barge is shown containing approximately 35 million 3D points. To place the model in context, a manually registered 2D mosaic from images of the same barge is shown with with the area highlighted that corresponds to the recovered 3D model.

The algorithm described in this chapter utilizes traditional computer vision techniques — stereo range estimation plus visual motion registration — to acquire 3D models of the environment and maintain an estimate of the sensor trajectory. Deploying the sensor in a particular environment requires solving **for that environment** a range of calibration and parameter tuning issues. These are described in the following section.

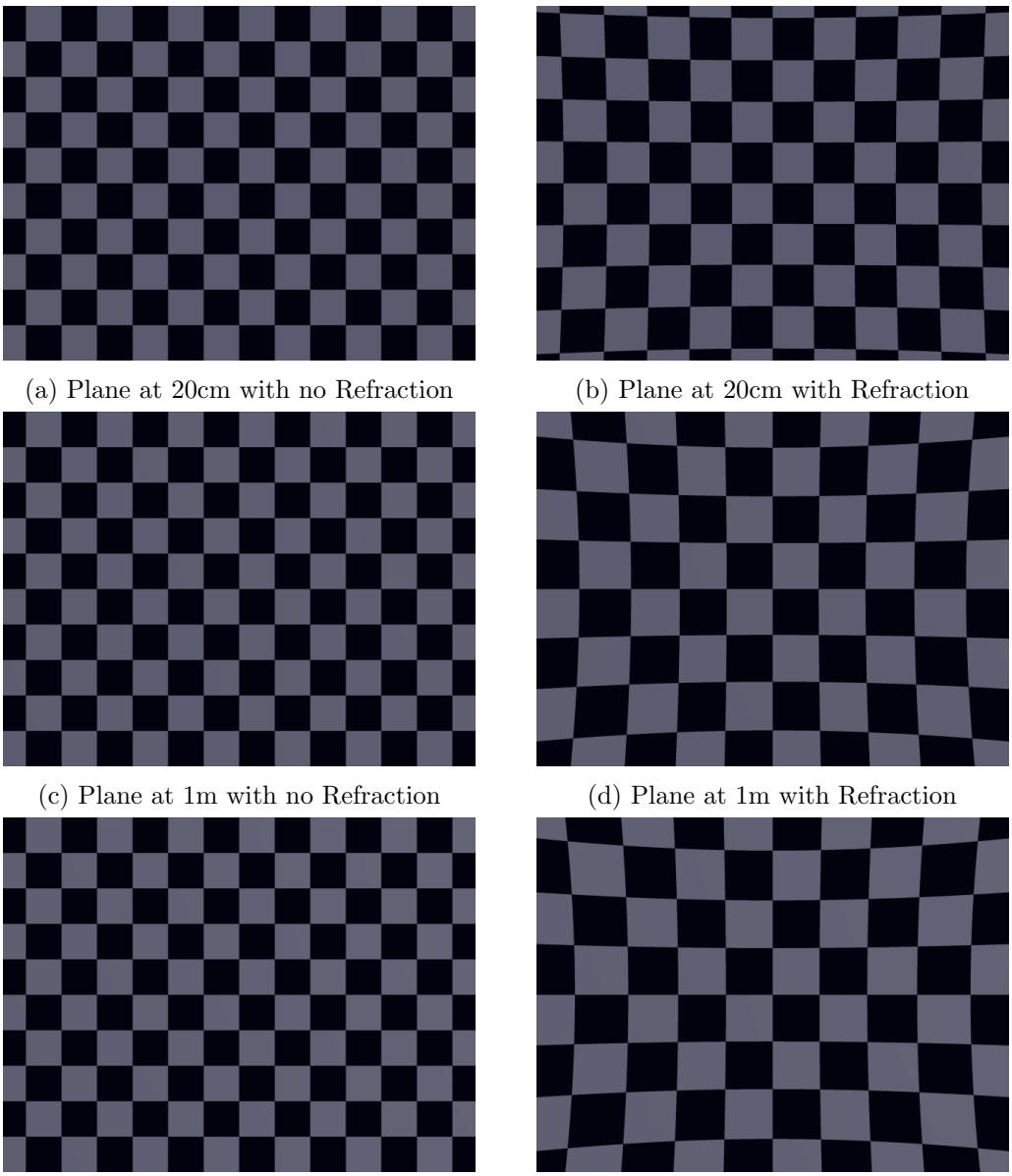
3.5 Issues with underwater stereo-vision

One major issue with using stereo-vision underwater is the need to re-calibrate the sensor(s) when placed in water. Although this only needs to be performed once (assuming that the cameras are fixed), performing calibration underwater is challenging, requires extra diver support and a precision calibration rig [Harvey and Shortis, 1998]. This would seem straightforward, however even with good underwater calibration, it is difficult to determine the parameters with high accuracy due to refraction errors which is typically modelled as a radial distortion

in the image (see [Kwon, 1999b, Treibitz et al., 2008, Kunz and Singh, 2008]).

Underwater, images acquired by the camera are subject to refraction due to the air-water interface. Light rays bouncing off of the objects in the environment refract as they leave the water into the plexiglass (or glass) port of the camera housing before reaching the camera lens. These errors induced by refraction are traditionally treated as radial distortion parameters in the image plane (see [Harvey and Shortis, 1998]) however 3D errors are apparent (for example objects that are occluded may be visible after refraction) [Treibitz et al., 2008]. In a stereo-vision application, this refractive error creates images that violate the epipolar constraint. Thus, a stereo-vision system undergoing this type of refraction may incorrectly match points and increase the error in the disparity estimate. An example of this can be seen in Figure 3.10. These images were created with POV-Ray, a free open-source ray-tracer. In all images, a plane was textured with a checkerboard and moved to various distances from the camera. The air-water interface was placed at 10cm from the camera and extended beyond the working volume. As illustrated, the curvature of the checkerboard lines changes noticeably depending on the distance of the plane from the camera. The refractive error is an example of a sensor parameter that can be modeled as depending on both the pose of the sensor and the structure of the environment. The image error for each point in the scene is different and should be estimated to properly characterize the error of each 3D point. One solution to this problem is to focus on objects within a particular distance volume from the cameras and disregard the rest of the scene. The volume can be calibrated by fixing calibration targets in a known lattice at known distances and a lookup table can be precomputed and applied in the image space as done in [Kwon, 1999a, Kwon, 1999b, Kwon and Lindley, 2000]. This solution is suitable for a pool scenario however it becomes problematic to deploy in the field. The calibration method requires a large structure set up in the underwater environment. Ensuring that this large (possibly 1-2 meters square) structure is stable in the field is very difficult due to current and surge of the open waters. It also becomes a three person job at minimum to deploy the sensor and perform the calibration.

The effects of this type of error is apparent not only in simulation but in real-



(a) Plane at 20cm with no Refraction

(b) Plane at 20cm with Refraction

(c) Plane at 1m with no Refraction

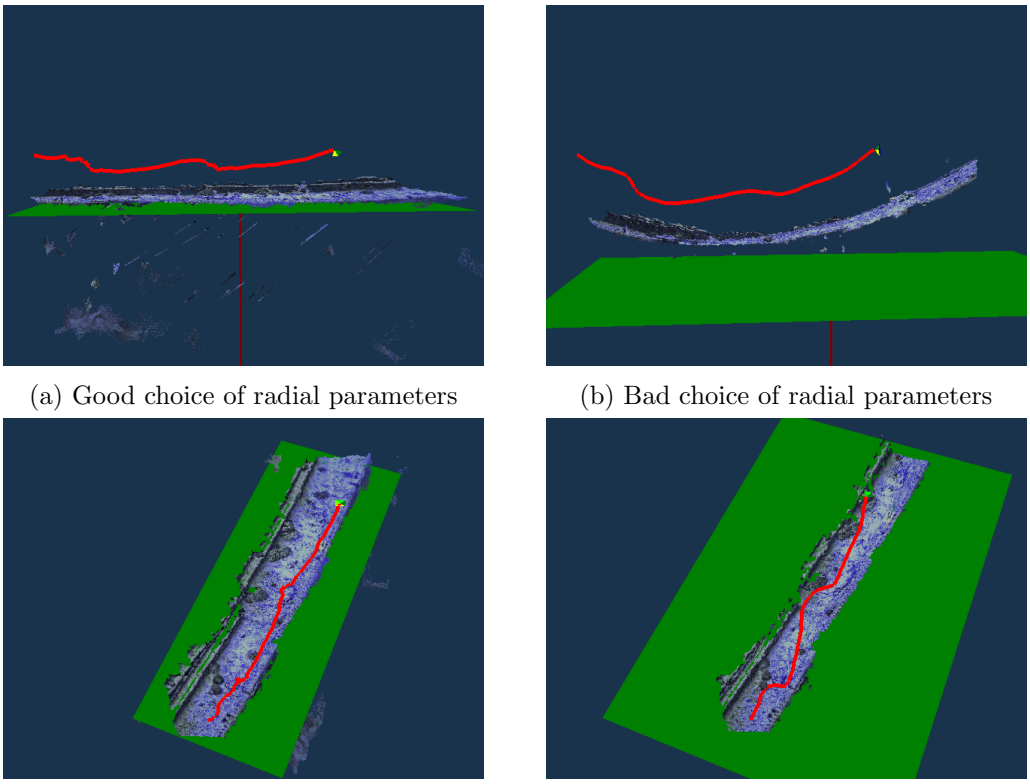
(d) Plane at 1m with Refraction

(e) Plane at 2m with no Refraction

(f) Plane at 2m with Refraction

Figure 3.10: Effects of object depth on image formation. Simulation performed using POV-Ray. (a,c,e) index of refraction = 1 (b,d,f) index of refraction=1.33.

world situations as well. Figure 3.11 shows examples of reconstructions of a relatively planar barge using different sets of radial parameters (found empirically). A better solution is to utilize a method that takes into account the environment model created from the sensor and estimate the parameters online to minimize the effects of the refraction. It may be advantageous to use different sets of these parameters in different situations. For example, for different disparity volumes a different set of parameters may be chosen. In this manner, it would be interesting and useful to develop a method that simultaneously estimates the sensor parameter values, the pose of the sensor, and the map resulting from the measurements. Things to consider when developing such an algorithm are that if the refraction error is modeled as a radial distortion, then the amount of error is non-stationary as the sensor moves in the environment generally. Obviously, if the sensor views the world from exactly the same location this is not a problem. Nor is it a problem if the sensor is always kept at a fixed distance from the environment. These scenarios are uninteresting however as it is unrealistic to assume that the sensor doesn't move or that its motion is constrained in this manner. Thus, the noise model should be kept general. The sensor parameters are thusly allowed to be dynamically changing and the noise is a function of the map and trajectory of the sensor itself. An algorithm that provides such flexibility in its sensing paradigm — SensorSLAM — is developed in the next chapter.



(a) Good choice of radial parameters

(b) Bad choice of radial parameters

(c) Top view of (a)

(d) Top view of (b)

Figure 3.11: Effects of Radial distortion parameters in real-world scenario. The curvature in (b) is due to error in 3D points near the edges of the images. This error is due to refraction effects coupled with the depth of the 3D point. The images are improperly registered due to the improper radial parameters for this volume of space thus violating the epipolar constraints. Matches are found due to similarity in the image but the resulting 3D point is incorrect (shifted in depth). The top views suggest the image registration is correct since the model is smooth and continuous.

CHAPTER 4

SensorSLAM

One issue with existing SLAM formulations is their inability to address the problem of non-stationary sensor noise within their sensor models. If sensor performance were to degrade due to the orientation or position of the vehicle in the environment, then standard noise models (which assume stationary noise) are invalidated and this leads to instability and loss of accuracy in the resulting map and trajectory. With camera sensors in underwater environments, non-standard noise manifests itself in different ways. Here non-stationary is used to encapsulate the possibility of the noise model being dependent on the map or pose of the vehicle but could also be treated as a time-varying model. One could imagine a situation where both issues occur, i.e. the sensor is affected by a moving distortion in the environment. For the remainder of this dissertation however, only map/pose dependent noise sources will be considered. For example, light is attenuated differentially by wavelength. Red wavelengths are absorbed by the water column at a higher rate than blues and greens. If the vision algorithms depend upon consistency in the red color channel for tracking or segmentation, the algorithm accuracy and effectiveness will be degraded differentially as a function of the vehicle's state as the vehicle submerges. The total amount of illumination is also depth-dependent (as well as weather dependent). Another example is the radial-like image distortion when a planar camera model is used underwater. Due to the refraction, light rays entering the sensor at the water-glass or water-air interface are slightly bent. The effects can be alleviated (but not elminated) in a single camera sensor using a properly mounted hemispherical lens that surrounds the camera [Treibitz et al., 2008, Kunz and Singh, 2008] but when using stereo sensors this becomes more difficult. The image distortion appears as a magnification plus a radial component. Insidiously however, the amount of distortion changes as objects move further from the camera. The distortion is dependent on the interaction of map and pose.

Although a range of solutions exist for the calibration of sensor parameters that are static (e.g., [Bouguet, 1999, Hemayed, 2003, Lavest et al., 2003, Habib et al., 2002, Salvi et al., 2002, Personnaz and Horaud, 2002, Zhang, 2000, Chatterjee and Roychowdhury, 2000, Wei and Ma, 1994]), how can we incorporate such calibration when the calibration changes as a function of vehicle or map state? Incorporating the sensor calibration parameters within the SLAM formulation requires solving for the joint probability of the vehicle pose, s_t, the map, Θ, and the sensor parameters, P_t. For the purposes of this work, this formulation is called SensorSLAM. In general, it can be expressed as task of solving for the conditional probability

$$p(s_t, \Theta, P_t | z^t, u^t, n^t). \tag{4.1}$$

Essentially, the idea is to derive the joint pdf over the sensor pose, s_t, the map of the environment, θ, and the current estimate of the sensor parameters, P_t. This enables a recursive filtering algorithm to simultaneously estimate the sensor parameters in the SLAM framework. For the following derivations, the sensor parameters are denoted at time t with a subscript as P_t and the collection of such parameters over time $1 \dots t$ is denoted with a superscript as $P^t = \{P_1, P_2, \dots, P_t\}$. We wish to estimate the joint probability of the robot location s_t, the map $\Theta = \{\Theta_1, \Theta_2, \dots, \Theta_n\}$ which represents the world as a collection of n landmarks, and the current sensor parameters P_t. We also have knowledge of sensor observations $z^t = \{z_1, z_2, \dots, z_t\}$, control inputs $u^t = \{u_1, u_2, \dots, u_t\}$, and data associations $n^t = \{n_1, n_2, \dots, n_t\}$.

The first thing to note about equation (4.1) is that certain forms of the conditional probability of SensorSLAM permit simple solutions that reduce the problem to traditional SLAM. If the sensor parameters are known then we do not have to estimate them as part of the joint distribution as they are conditionally independent. The desired pdf could be factored in terms of

$$p(s_t, \Theta | P_t, z^t, u^t, n^t) p(P_t | z^t, u^t, n^t). \tag{4.2}$$

Specifically, for a conditional distribution function $F(y|x, w)$ over variables y given

$X = x$ and $W = w$ we can marginalize over W as in [Cox and Wermuth, 2003] as

$$F(y|x) = \int F(y|x, w) f(w|x) dw \tag{4.3}$$

This relationship can be used to marginalize over the sensor parameters as

$$p(s_t, \Theta|z^t, u^t, n^t) = \int p(s_t, \Theta|P_t, z^t, u^t, n^t) p(P_t|z^t, u^t, n^t) dP_t \tag{4.4}$$

resulting in the traditional SLAM pdf. Thus, assuming that the sensor parameters are a function of observations only (i.e. not related to the map or pose) reduces the entire problem to SLAM.

Another special case of the problem is to assume that the sensor parameters are independent of both the pose, s_t, and the map, Θ, and only dependent on the measurements. This implies that the pose is also not a function of P_t. Equation 4.1 can thus be factored as

$$p(s_t, \Theta, P_t|z^t, u^t, n^t) = p(s_t, \Theta|z^t, u^t, n^t) p(P_t|z^t, u^t, n^t). \tag{4.5}$$

Notice that under these assumptions the problem reduces to two independent subproblems. First, is the estimation of the traditional SLAM pdf, $p(s_t, \Theta|z^t, u^t, n^t)$, and second is the estimation of the sensor parameter pdf, $p(P_t|z^t, u^t, n^t)$. Furthermore, using the sensor parameter may be simplified since the knowledge of the data associations and controls do not influence the parameters resulting in

$$p(P_t|z^t, u^t, n^t) = p(P_t|z^t) \tag{4.6}$$

Using Bayes rule on Equation 4.6 obtains

$$p(P_t|z^t) = \frac{p(z_t|P_t, z^{t-1}) p(P_t|z^{t-1})}{p(z_t|z^{t-1})} \tag{4.7}$$

Noting that the measurements are independent of each other and the current sensor parameters are independent on the previous measurements, Equation 4.7 becomes

$$p(P_t|z^t) = \frac{p(z_t|P_t) p(P_t)}{p(z_t)} \tag{4.8}$$

$$= \eta p(z_t|P_t) p(P_t) \tag{4.9}$$

where η is a normalizing constant denoting the denominator resulting from Bayes rule that ensures the result is a probability. Combining this result with equation (4.5) gives

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta p(s_t, \Theta | z^t, u^t, n^t) p(z_t | P_t) p(P_t) \tag{4.10}$$

This form however does not encapsulate possible interactions between the vehicle location and the map with the sensor parameters and would be unable to accommodate any dynamic location-based bias in the sensor model. Furthermore, $p(z_t|P_t)$ could be subsumed into the observation model of the traditional SLAM formulation resulting in $p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta p(s_t, \Theta | z^t, u^t, n^t) p(P_t)$ which could be considered equivalent to SLAM generally.

4.1 The general SensorSLAM derivation

Let us now turn to the general form of the problem. Assume that the form of the SensorSLAM conditional pdf is not factorizable at the outset. To be as general as possible, assume a dynamic map process. The desired pdf to estimate is

$$p(s_t, \Theta_t, P_t | z^t, u^t, n^t). \tag{4.11}$$

This is to be contrasted with the traditional SLAM pdf (see [Thrun et al., 2005]) that does not incorporate the sensor parameters (and uses a static map)

$$p(s_t, \Theta | z^t, u^t, n^t) \tag{4.12}$$

The following derivation parallels the derivation of SLAM as described in [Montemerlo, 2003] and shown in Chapter 2. Using Bayes rule, Equation 4.11 can be converted into a posterior probability as

$$p(s_t, \Theta_t, P_t | z^t, u^t, n^t) = \frac{p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t)}{p(z_t | z^{t-1}, u^t, n^t)} \tag{4.13}$$

For completeness this is shown explicitly. Given Bayes rule

$$p(A|B) = \frac{p(A, B)}{p(B)} \tag{4.14}$$

and substituting $A = \{s_t, \Theta_t, P_t\}$, and $B = \{z^t, u^t, n^t\}$, results in the following expression

$$p(s_t, \Theta_t, P_t | z^t, u^t, n^t) = \frac{p(s_t, \Theta_t, P_t, z^t, u^t, n^t)}{p(z^t, u^t, n^t)} \tag{4.15}$$

The numerator can be simplified by using the definition of conditional probability — specifically that $p(A, B) = p(A|B)p(B)$ — to isolate the current measurement by substituting $A = \{z_t\}$ and $B = \{s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t\}$ as

$$p(s_t, \Theta_t, P_t, z^t, u^t, n^t) = p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) \tag{4.16}$$

In the second term, s_t, Θ_t, P_t are isolated in the same manner to obtain

$$p(s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) = p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t) p(z^{t-1}, u^t, n^t) \tag{4.17}$$

Isolating the current measurement in the denominator of equation (4.15) results in a denominator of

$$p(z^t, u^t, n^t) = p(z_t | z^{t-1}, u^t, n^t) p(z^{t-1}, u^t, n^t) \tag{4.18}$$

Substituting equation (4.16), equation (4.17), and equation (4.18) into equation (4.15) and simplifying we get

$$\begin{aligned}
p(s_t, \Theta_t, P_t | z^t, u^t, n^t) =& \frac{p(s_t, \Theta_t, P_t, z^t, u^t, n^t)}{p(z^t, u^t, n^t)} \\
\overset{\text{equation (4.16)}}{=}& \frac{p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t)}{p(z^t, u^t, n^t)} \\
\overset{\text{equation (4.17)}}{=}& \frac{p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t) p(z^{t-1}, u^t, n^t)}{p(z^t, u^t, n^t)} \\
\overset{\text{equation (4.18)}}{=}& \frac{p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t) p(z^{t-1}, u^t, n^t)}{p(z_t | z^{t-1}, u^t, n^t) p(z^{t-1}, u^t, n^t)} \\
=& \frac{p(z_t | s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t) p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t)}{p(z_t | z^{t-1}, u^t, n^t)}
\end{aligned}$$

The denominator does not depend on $\{s_t, \Theta, P_t\}$ and can be considered as a normalizing constant. This can be shown explicitly by using the theorem of total

probability: Given a set of hypotheses A_i that cover the sample space, we can express the probability of B as

$$p(B) = p(B, A_1) + p(B, A_2) + \ldots + p(B, A_n) \tag{4.19}$$

given that the sets denoted by the A_i's are disjoint. This can be expressed compactly as

$$p(B) = \sum_i^n p(B, A_i). \tag{4.20}$$

Through the definition of conditional probability

$$p(B, A_i) = p(B|A_i)p(A_i) \tag{4.21}$$

which allows the above summations to be expressed as

$$p(B) = \sum_i^n p(B|A_i)p(A_i) \tag{4.22}$$

To express the denominator of equation (4.13) in this form, however it is necessary to first show that the following holds

$$p(B|C) = \sum_i^n p(B|C, A_i)p(A_i|C) \tag{4.23}$$

Using the well-known relationship $p(B, C) = p(B|C)p(C)$, equation (4.19) becomes

$$\begin{aligned}
p(B, C) &= p(B, C, A_1) + p(B, C, A_2) + \ldots + p(B, C, A_n) \\
&= p(B|C, A_1)p(C, A_1) + p(B|C, A_2)p(C, A_2) + \ldots + p(B|C, A_n)p(C, A_n) \\
&= p(B|C, A_1)p(A_1|C)p(C) + \ldots + \quad p(B|C, A_n)p(A_n|C)p(C) \\
&= p(C) \cdot [p(B|C, A_1)p(A_1|C) + \ldots + p(B|C, A_n)p(A_n|C)] \\
p(B, C) &= p(C) \sum_i^n p(B|C, A_i)p(A_i|C) \\
\frac{p(B, C)}{p(C)} &= \sum_i^n p(B|C, A_i)p(A_i|C) \\
p(B|C) &= \sum_i^n p(B|C, A_i)p(A_i|C)
\end{aligned}$$

and for continuous probability functions the summation becomes an integral as

$$p(B|C) = \int p(B|C, A_i)p(A_i|C)dA_i$$

Going back to the SLAM formulation and by using the appropriate substitutions, i.e.

$$p(\underbrace{z_t}_{B} | \underbrace{z^{t-1}, u^t, n^t}_{C}) \text{ and } \underbrace{\{s_t, \Theta_t, P_t\}}_{A_i}$$

the denominator of Equation 4.15 becomes

$$\begin{aligned} p(z_t|z^{t-1}, u^t, n^t) &= \int p(B|C, A_i)p(A_i|C)dA_i \\ &= \int p(z_t|A_i, z^{t-1}, u^t, n^t)p(A_i|z^{t-1}, u^t, n^t)dA_i \\ &= \iiint p(z_t|s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t)p(s_t, \Theta_t, P_t|z^{t-1}, u^t, n^t)ds_t d\Theta_t dP_t \\ &= \eta^{-1} \end{aligned}$$

Typically, η^{-1} is thought of as a normalizing constant since it ensures the numerator is a probability (i.e. between 0 and 1). For the remainder of this thesis we will view the denominator η^{-1} as a normalization constant for notational convenience until we are required to compute it directly. Substituting the denominator back into equation (4.13) results in

$$p(s_t, \Theta_t, P_t|z^t, u^t, n^t) = \eta p(z_t|s_t, \Theta_t, P_t, z^{t-1}, u^t, n^t)p(s_t, \Theta_t, P_t|z^{t-1}, u^t, n^t) \quad (4.24)$$

Assuming that each measurement z_t is conditionally independent of the previous measurements $z_{t-1}, z_{t-2}, \ldots, z_1$ given knowledge of s_t, Θ_t, P_t (Markov assumption) and control inputs equation (4.24) simplifies to

$$p(s_t, \Theta_t, P_t|z^t, u^t, n^t) = \eta p(z_t|s_t, \Theta_t, P_t, n_t)p(s_t, \Theta_t, P_t|z^{t-1}, u^t, n^t).$$

Note that $p(z_t|s_t, \Theta_t, P_t, n_t)$ is a probabilistic sensor measurement model that represents the probability density of the sensor measurements. It depends upon the given sensor parameters P_t, the pose of the vehicle s_t and the data associations, n^t.

The second term $p(s_t, \Theta_t, P_t | z^{t-1}, u^t, n^t)$ is a more complex interaction of density functions.

Assuming that the robot pose and sensor parameters are governed by Markov chain processes — they are dependent not on their entire history but rather only over the previous estimate — allows the marginalization over the previous pose and previous sensor parameter estimate as

$$p(s_t, \Theta_t, P_t | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta_t, P_t, n_t) \iiint p(s_t, s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | z^{t-1}, u^t, n^t) ds_{t-1} dP_{t-1} d\Theta_{t-1} \tag{4.25}$$

Using the definition of conditional probability, and using the shorthand $R = \{z^{t-1}, u^t, n^t\}$, to isolate the robot pose term results in

$$p(s_t, s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R) = p(s_t | s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1}, R) \cdot p(s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R)$$

The first term of the right hand side of the equation above does not require information about the previous map and sensor parameters as it has knowledge of the current information. Noticing that Θ_{t-1}, P_{t-1} can be removed as they do not provide any new information and simplifying gives

$$p(s_t, s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R) = p(s_t | s_{t-1}, \Theta_t, P_t, R) \cdot p(s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R) \tag{4.26}$$

Re-arranging the last term of Equation 4.26 gives

$$p(s_t, s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R) = p(s_t | s_{t-1}, \Theta_t, P_t, R) \cdot p(\Theta_t, P_t, s_{t-1}, \Theta_{t-1}, P_{t-1} | R)$$

Using the definition of conditional probability to isolate the current map and sensor parameters results in

$$p(s_t, s_{t-1}, \Theta_t, \Theta_{t-1}, P_t, P_{t-1} | R) = p(s_t | s_{t-1}, \Theta_t, P_t, R) \cdot p(\Theta_t, P_t | s_{t-1}, \Theta_{t-1}, P_{t-1}, R) p(s_{t-1}, \Theta_{t-1}, P_{t-1} | R)$$

Substituting back into equation (4.25) results in the following

$$\begin{aligned} p(s_t, \Theta_t, P_t | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta_t, P_t, n_t) \cdot \\ \iiint p(s_t | s_{t-1}, \Theta_t, P_t, z^{t-1}, u^t, n^t) \cdot \\ p(\Theta_t, P_t | s_{t-1}, \Theta_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t) \cdot \\ p(s_{t-1}, \Theta_{t-1}, P_{t-1} | z^{t-1}, u^t, n^t) ds_{t-1} dP_{t-1} d\Theta_{t-1} \end{aligned} \tag{4.27}$$

One last set of general simplifications can be made to the last term under the integral by noting that a future control and data association cannot affect a past state, thus the current terms u_t, n_t can be dropped which gives

$$\begin{aligned} p(s_t, \Theta_t, P_t | z^t, u^t, n^t) = \eta \underbrace{p(z_t | s_t, \Theta_t, P_t, n_t)}_{\text{Measurement Model}} \cdot \\ \iiint \underbrace{p(s_t | s_{t-1}, \Theta_t, P_t, z^{t-1}, u^t, n^t)}_{\text{Vehicle Motion model}} \cdot \\ \underbrace{p(\Theta_t, P_t | s_{t-1}, \Theta_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Joint Map \& Sensor Parameter pdf}} \cdot \\ \underbrace{p(s_{t-1}, \Theta_{t-1}, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1})}_{\text{Recursive Formulation}} ds_{t-1} dP_{t-1} d\Theta_{t-1} \end{aligned} \tag{4.28}$$

Solving this problem is certainly computationally expensive and infeasible for real-time processing with modern computational systems since the algorithm would have to compute all possible vehicle poses for each possible map given all possible sensor parameters at each update time step. To develop an algorithm that approximates this pdf, several reasonable assumptions must be made.

Static world model assumption

The first simplification is to assume the world model is static. This is a typical assumption in SLAM and is valid for many applications. In situations where the map is filled with highly dynamic objects this would not hold, however if the scene contains a large set of static objects the assumption holds. If a small number of objects are dynamic — such as small numbers of fish swimming in the scene — a

separate, but more computationally feasible (and sub-optimal), method could be used to label measurements as being either static or dynamic as in [Wang, 2004, Wang et al., 2007a]. Incorporating the assumption of a static map/world model reduces the problem to the following form

$$\begin{aligned} p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta \underbrace{p(z_t | s_t, \Theta, P_t, n_t)}_{\text{Measurement Model}} \cdot \\ \iint \underbrace{p(s_t | s_{t-1}, \Theta, P_t, z^{t-1}, u^t, n^t)}_{\text{motion model}} \cdot \\ \underbrace{p(\Theta, P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Joint Map \& Sensor Parameter pdf}} \cdot \\ \underbrace{p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1})}_{\text{Recursive Formulation}} ds_{t-1} dP_{t-1} \end{aligned} \tag{4.29}$$

Note that in the above simplification, the sensor parameters are still assumed to be dynamic. An example of such a situation would be when the sensor parameters vary on the position and/or orientation of the sensor relative to the map. One such parameter might be an estimate of the current heading bias of a magnetometer in an environment containing local magnetic distortions. Also, if the sensor parameter model is time-varying, the appropriate model could be incorporated here as well.

Static map + constant but unknown sensor parameters

One further assumption would be to assume that the sensor parameters are constant but are unknown. An example of such a situation includes modelling static but unknown intrinsic camera parameters of the camera, e.g. the focal length. This

would further simplify equation (4.29) to

$$
\begin{aligned}
p(s_t, \Theta, P | z^t, u^t, n^t) = \eta \underbrace{p(z_t | s_t, \Theta, P, n_t)}_{\text{Measurement Model}} \cdot \\
\int \underbrace{p(s_t | s_{t-1}, \Theta, P, z^{t-1}, u^t, n^t)}_{\text{motion model}} \cdot \\
\underbrace{p(\Theta, P | s_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Joint Map \& Sensor Parameter pdf}} \cdot \\
\underbrace{p(s_{t-1}, \Theta, P | z^{t-1}, u^{t-1}, n^{t-1})}_{\text{Recursive Formulation}} ds_{t-1}
\end{aligned}
\tag{4.30}
$$

In traditional SLAM, if errors occur due to sensor parameters, these will be assigned as noise to the map prior pdf. In scenarios where the parameters are non-linear or have a probabilistic model that is separate from the landmark error, this explicit form allows for a more accurate map since the sensor error is represented explicitly. This formulation decouples the error introduced by the sensor parameter noise from the map feature uncertainty resulting in a higher accuracy map.

Static map + pose/map-dependent parameters

Given equation (4.29), the joint map and parameter pdf can be simplified in two ways depending on the situation. First, splitting $p(\Theta, P_t | s_{t-1}, z^{t-1}, u^t, n^t)$ into a map prior and a sensor pdf results in

$$
\begin{aligned}
p(\Theta, P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t) = p(P_t | \Theta, s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t) \cdot \\
p(\Theta, | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)
\end{aligned}
\tag{4.31}
$$

This form implies that the current state of the sensor parameters is dependent upon both motion history and the map as well as its own history. Also, the map is dependent only upon the previous state of the sensor parameters, measurements and motion. Alternatively, the parameters can be isolated from the map as

$$
\begin{aligned}
p(\Theta, P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t) = p(\Theta | P_t, s_{t-1}, z^{t-1}, u^t, n^t) \cdot \\
p(P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)
\end{aligned}
\tag{4.32}
$$

which implies that the sensor parameters depend only on its own history, the previous vehicle location, measurements and controls and does not depend on the map. The map thus depends on the current state of the sensor parameters which suggests that as the sensor parameters increase in accuracy, the map uncertainty decreases.

One way to think about equation (4.32) is to think of the problem as estimating two types of maps spread out over the traversed area. The map and parameters are tightly coupled so they can be represented in similar fashions. One environmental "map" represents the feature locations and their uncertainties while the second "sensorial parameter map" represents the current value and uncertainty of the sensor parameters. At any point in time, the full map can be extracted from these representations simply by multiplying the respective map elements together. To illustrate this point, an example is developed that underlines this approach.

Example

As an example, take a robot that relies heavily on the use of a magnetometer for its absolute heading in the world. The world contains a large amount of ferrous material in some area thus creating static magnetic distortion biases that are location dependent. The distortion is spread over the travelled area, thus the current estimate of the bias changes depending on some model that incorporates the current map and vehicle location. Odometry can be relied upon for a short period of time and as such the motion model in the SensorSLAM framework (Equation 4.29) collapses to the traditional vehicle motion model, namely $p(s_t|s_{t-1}, \Theta, P, z^{t-1}, u^t, n^t) = p(s_t|s_{t-1}, u_t)$ resulting in a posterior

$$\begin{aligned} p(s_t, \Theta, P_t|z^t, u^t, n^t) = \eta p(z_t|s_t, \Theta, P_t, n_t)\cdot \\ \iint p(s_t|s_{t-1}, u_t)p(\Theta, P_t|s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)\cdot \\ p(s_{t-1}, \Theta, P_{t-1}|z^{t-1}, u^{t-1}, n^{t-1})ds_{t-1}dP_{t-1} \end{aligned} \tag{4.33}$$

The motion model is independent of the sensor parameters. Thus the motion model can be moved out of the interior integral as

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta, P_t, n_t) \cdot \int p(s_t | s_{t-1}, u_t) \int p(\Theta, P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t) \cdot p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) dP_{t-1} ds_{t-1} \tag{4.34}$$

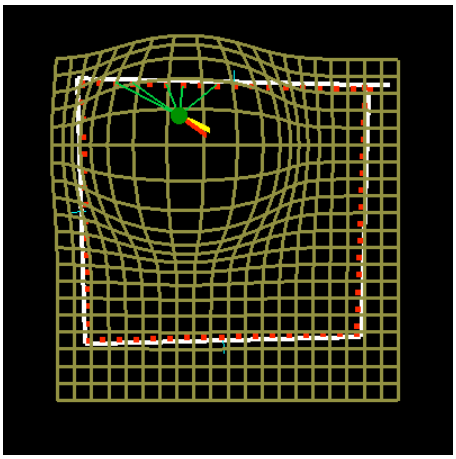
Equation 4.32 explicitly denotes the relationship between the map and sensor parameter model

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta \underbrace{p(z_t | s_t, \Theta, P_t, n_t)}_{\text{Measurement Model}} \cdot \int \underbrace{p(s_t | s_{t-1}, u_t)}_{\text{Motion Model}} \int \underbrace{p(\Theta | P_t, s_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \underbrace{p(P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \cdot \underbrace{p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1})}_{\text{Recursive Formulation}} dP_{t-1} ds_{t-1} \tag{4.35}$$

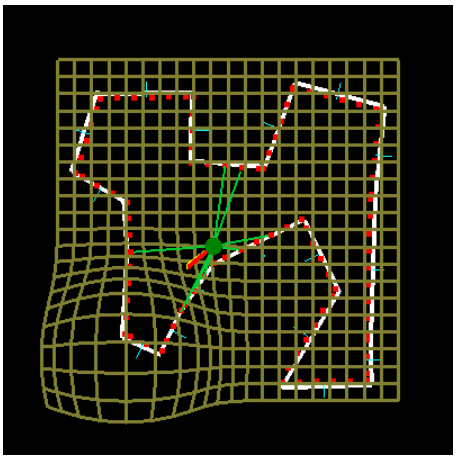
To simulate this scenario an OpenGL simulator was developed that uses a standard kinematic model for robotic motion (see Figure 4.1). Beacons (landmarks) can be interactively placed in the environment that follow the set of walls. A simulated laser beam measures the distance to beacons near the vehicle. Beacons beyond a given distance threshold are not observed. The robot is controlled by the user. While moving, the odometry, simulated laser observations and beacon IDs are stored in a log file. A magnetic distortion source can be placed within the environment (represented as a distortion on the grid) and its strength can be modified and controlled. The simulator uses a Gaussian error model for the magnetic distortion.

The SLAM implementation used as the *gold-standard* solution is the Matlab EKF implementation by Tim Bailey[7]. Figure 4.2a shows the results of the SLAM

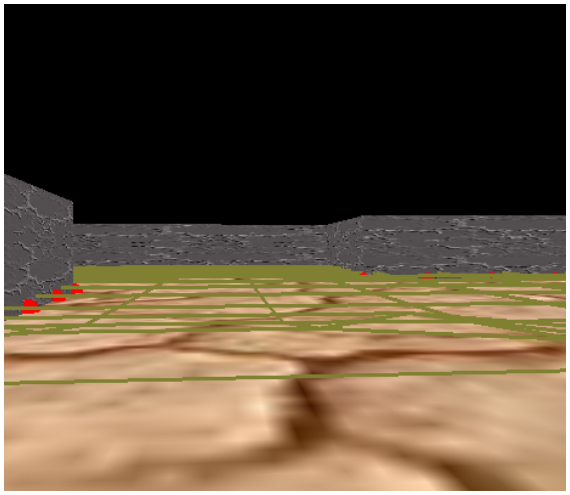
[7]http://openslam.org/



(a) Top-Down View

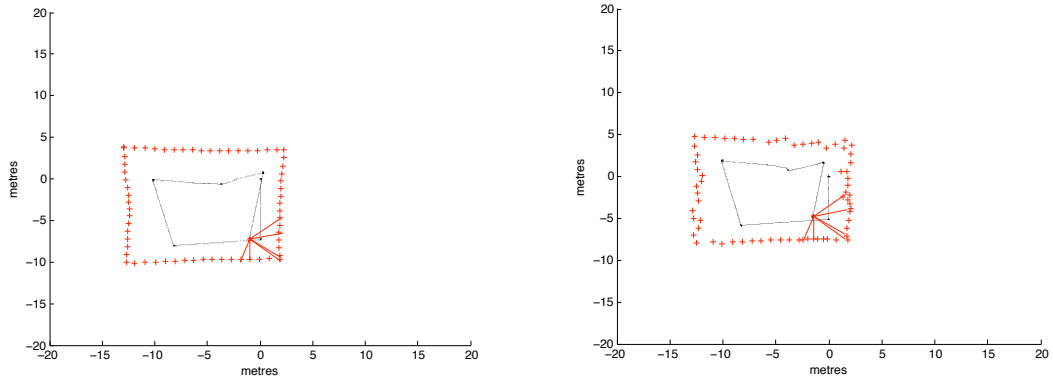


(b) Top-Down View



(c) 3D View

Figure 4.1: SLAM Simulator Images. Red dots are beacons, white lines are walls, green lines denote laser beams. (a) shows a top down view of the environment, the grid shows the amount of distortion applied to the sensor at each location (b) another top-down view of a different environment with different distortion amounts (c) the view from the simulated robot.

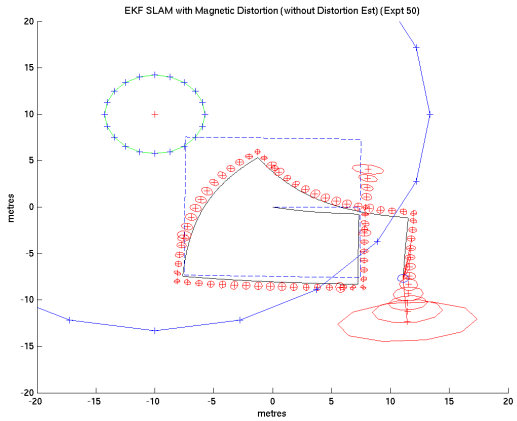


(a) Gold Standard Method with known data associations.

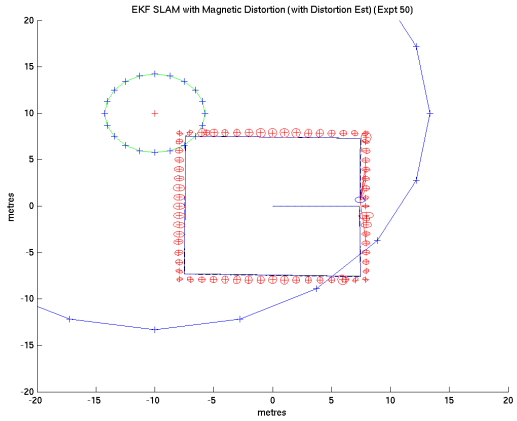
(b) EKF SLAM without known data associations.

Figure 4.2: Comparison of standard EKF-SLAM (a) with and (b) without known data associations. Magnetic distortion is not applied to the measurements. Red-plus signs (+) are the estimated beacon locations, red lines represent the current laser beam to the measurements and black lines represent the estimated vehicle location.

algorithm when run with the data and known data associations (the beacon identifiers were used directly to constrain the problem) from a single traversal of a rectangular area. Figure 4.2b shows the results of traditional SLAM **without** magnetic distortion but with **unknown** data assocations. A gated nearest neighbour search was used to estimate the associations. The result is similar to the ground truth scenario. Adding the magnetic distortion to the compass and re-running the SLAM algorithm (without known associations) produces a result as shown in Figure 4.3a. The map is incorrect and the trajectory is lost when the compass has a significant bias in the measurements. Figure 4.3b shows the same data run with full knowledge of the distortion at each pose. When the distortion parameters are fully known, the SLAM algorithm returns the map to an accurate result. In this simple illustration, it can be seen that knowledge of the distortion is beneficial to the map-making



(a) EKF SLAM with unknown distortion



(b) EKF SLAM with known distortion

Figure 4.3: Comparison of results from standard EKF SLAM when a magnetic distortion exists in the environment and is (a) unknown and (b) known to the algorithm. Distortion is located at (-10,10) and iso-contours at 1-sigma indicate the strength of the distortion. Beacon locations are shown (+ signs) with their corresponding uncertainty ellipses. Estimated trajectory is shown with black lines and ground-truth trajectory shown in dotted blue.

process.

4.2 SensorSLAM for location-based magnetic distortion bias

The algorithm developed here (and in the previous example) implements the Bayesian formulation in Equation 4.35 to approximate the SensorSLAM posterior for the previous example

$$\begin{aligned} p(s_t, \Theta, P_t | z^t, u^t, n^t) = \eta p(z_t | s_t, \Theta, P_t, n_t) \cdot \\ \int p(s_t | s_{t-1}, u_t) \int \underbrace{p(\Theta | P_t, s_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \\ \underbrace{p(P_t | s_{t-1}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \quad \cdot \\ p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) dP_{t-1} ds_{t-1} \end{aligned} \tag{4.36}$$

Adopting a hybrid particle-Kalman filter formulation enables the approximation of this pdf as before using the stochastic integration method. Here we perform this operation more generally so it can be used in an underwater environment with a stereo vision sensor.

It would be quite difficult to compute the interior integral over the entire continuous spectrum for the sensor parameters. As an approximation, this example uses a *stochastic integration*-like method. Assume that there is a finite number of possible locations for the magnetic distortion source. This allows for the integration over all possible values of the sensor parameters.

Given the following integral

$$\int p(P_t | s_{t-1}, P_{t-1}) dP_{t-1}$$

and given knowledge that there are only two possible values for P_{t-1}, $\{\text{state}_0, \text{state}_1\}$, the integral can be computed (now a summation since it is dis-

crete) as

$$\sum_{P_{t-1}} p(P_t|s_{t-1}, P_{t-1}) = p(P_t|s_{t-1}, \mathbf{P}_{t-1} = \text{state}_0) + p(P_t|s_{t-1}, \mathbf{P}_{t-1} = \text{state}_1)$$

The integral can be approximated by summing over all possible discrete values of the previous sensor parameter values. In the case of a discrete set of possible magnetic source locations, the set of possible parameter values is also discrete.

An approximation can be made to alleviate the necessity to integrate over multiple parameters. Obviously, doing so removes information that may be valuable for modelling every interaction of the random variables, however it lowers the computational burden. Assume for the map and sensor parameter model, that there exists a fixed value for the previous vehicle location. At each iteration we collapse the conditional probabilities in the map and sensor parameter pdf's to a single value for the vehicle pose $s_{t-1} = \mu_{s_{t-1}}$ which is the mean of the distribution calculated in the previous iteration resulting in the following

$$\begin{aligned} p(s_t, \Theta, P_t|z^t, u^t, n^t) \approx \eta p(z_t|s_t, \Theta, P_t, n_t) \cdot \\ \int p(s_t|s_{t-1}, u_t) \int \underbrace{p(\Theta|P_t, s_{t-1} = \mu_{s_{t-1}}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \\ \underbrace{p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \cdot \\ p(s_{t-1}, \Theta, P_{t-1}|z^{t-1}, u^{t-1}, n^{t-1}) dP_{t-1} ds_{t-1}. \end{aligned} \tag{4.37}$$

Since the map and sensor parameter pdf's are being approximated using a fixed

previous robot pose, the integrals can be rearranged as

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) \approx \eta p(z_t | s_t, \Theta, P_t, n_t) \cdot \int \underbrace{p(\Theta | P_t, s_{t-1} = \mu_{s_{t-1}}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \underbrace{p(P_t | s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \cdot \int p(s_t | s_{t-1}, u_t) p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) ds_{t-1} dP_{t-1} \tag{4.38}$$

In essence, this approach to SensorSLAM embeds the traditional SLAM formulation within a sensor parameter estimation framework as

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) \approx \eta p(z_t | s_t, \Theta, P_t, n_t) \cdot \int \underbrace{p(\Theta | P_t, s_{t-1} = \mu_{s_{t-1}}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \underbrace{p(P_t | s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \cdot \underbrace{\left[\int p(s_t | s_{t-1}, u_t) p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) ds_{t-1} \right]}_{\text{“traditional” SLAM sub-problem}} dP_{t-1} \tag{4.39}$$

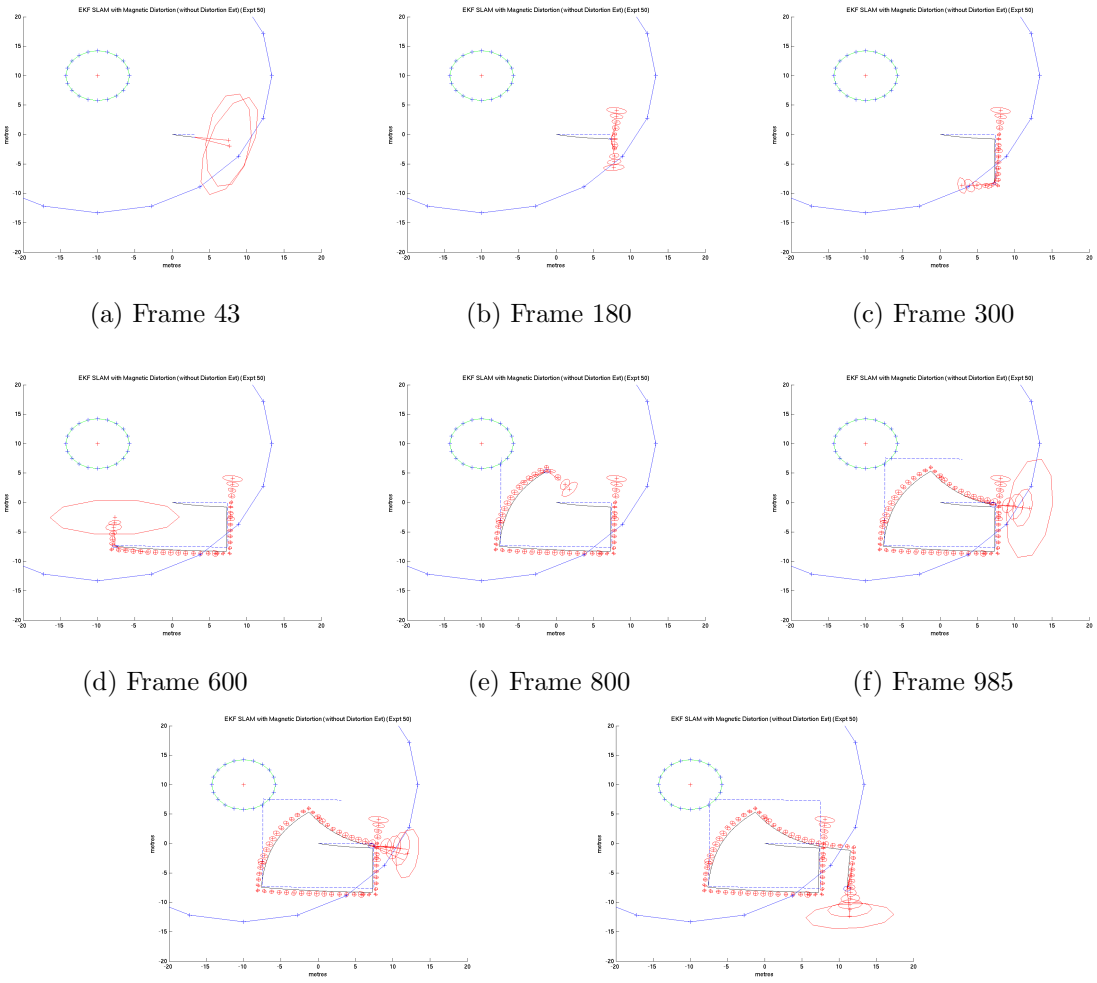
The pdfs, $p(\Theta | P_t, s_{t-1} = \mu_{s_{t-1}}, z^{t-1}, u^t, n^t) \cdot p(P_t | s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$, are maintained as the two separate maps and multiplied together when required. $p(P_t | s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$ is approximated as a set of sensorial maps, each one representative of a different magnetic distortion location. This is a discrete set of samples and the pdf can be estimated using a particle filtering algorithm.

To illustrate the effectiveness of the approach, simulated experiments were performed in Matlab using a standard implementation of EKF SLAM[8]. Without prior

[8]Tim Bailey's EKF slam implementation from the OpenSLAM initiative http://www.openslam.org.

knowledge of the sensor parameters the algorithm performs quite poorly in the presence of a magnetic distortion. The compass heading is biased differently throughout the environment increasing the error in the resulting map and trajectory (see Figure 4.4). When using SensorSLAM to estimate the sensor parameter pdf (see Figure 4.5) — in this example the parameter affecting the sensor is the magnetic distortion location — it can be seen that the resulting map and trajectory follow the ground truth and that the appropriate magnetic distortion location is estimated correctly. The particle error in these experiments was computed simply as the covariance of the current SLAM estimate with the idea that the SLAM estimate will have a higher error when the magnetic distortion affects the model and lower variance when the model is unaffected by the distortions. During the run, self-consistency or trusting the internal representation of the map too much can be detrimental to the overall quality of the map. This can be seen in Frames 800-985. Subsequently the map is self-consitent with the observations even without fully compensating for the distortion. However, the final map quality is higher in the SensorSLAM implementation than in the EKF-SLAM version. Fortunately, the loop closure event that occurs in Frame 986 reduces the error enough to bring the map and trajectory into alignment with the correct world view.

In a second experiment (see Figure 4.6), the error function was changed to take into account the relationship between the vehicle and the landmarks. The error model per particle was improved to be a Gaussian error model for each particle centered around the estimated landmark positions and uses the covariance matrix determined by the EKF. This error is used to weight the particle. Even though this approach favors an incorrect solution at the beginning of the run, it quickly (at Frame 405) corrects itself and continues robustly with that solution for the duration of the experiment.

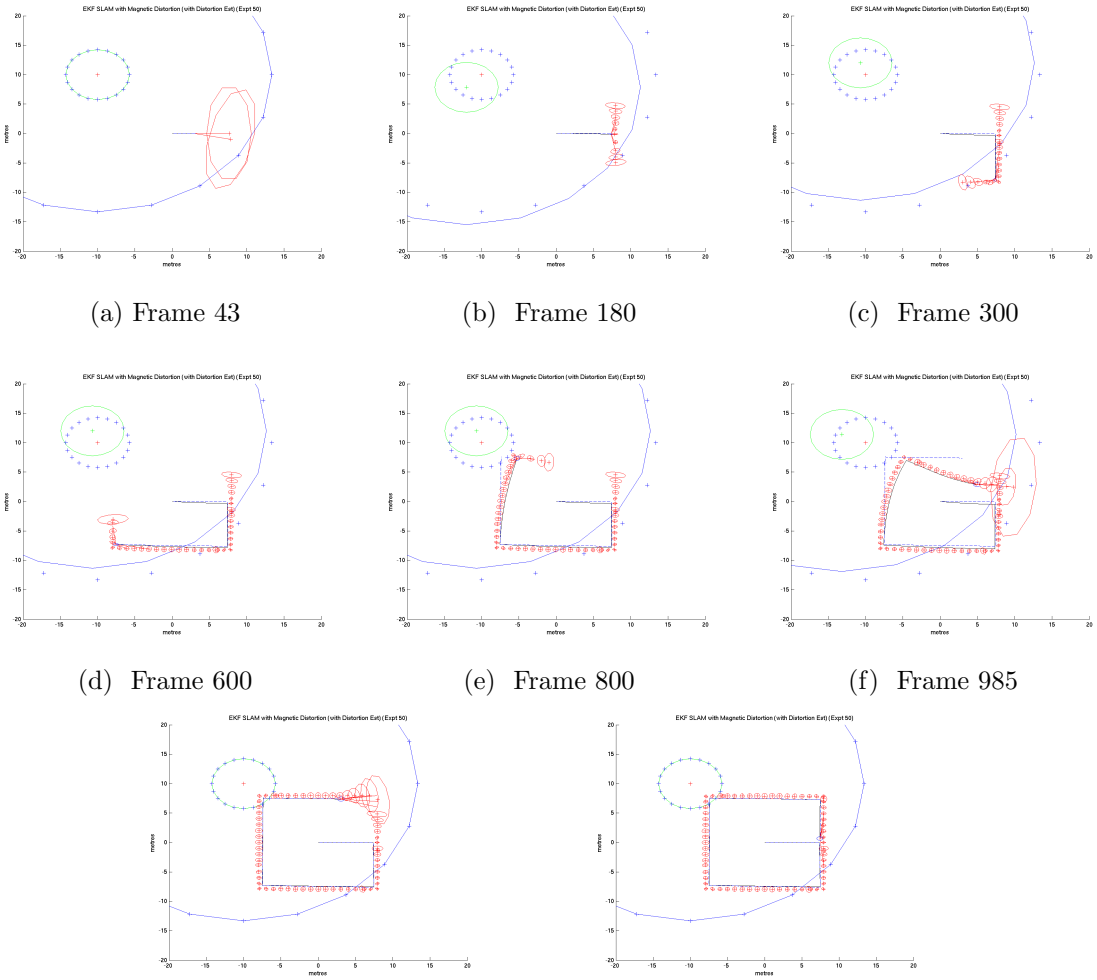


(a) Frame 43 (b) Frame 180 (c) Frame 300

(d) Frame 600 (e) Frame 800 (f) Frame 985

(g) Frame 986 (h) Frame 1200 (Final)

Figure 4.4: Example of simulation without distortion estimation using EKF SLAM. The magnetic distortion is illustrated but its parameters and position are not estimated during the run. Note the significant errors in the resulting map and trajectory. See the legend of Figure 4.3 for more details.

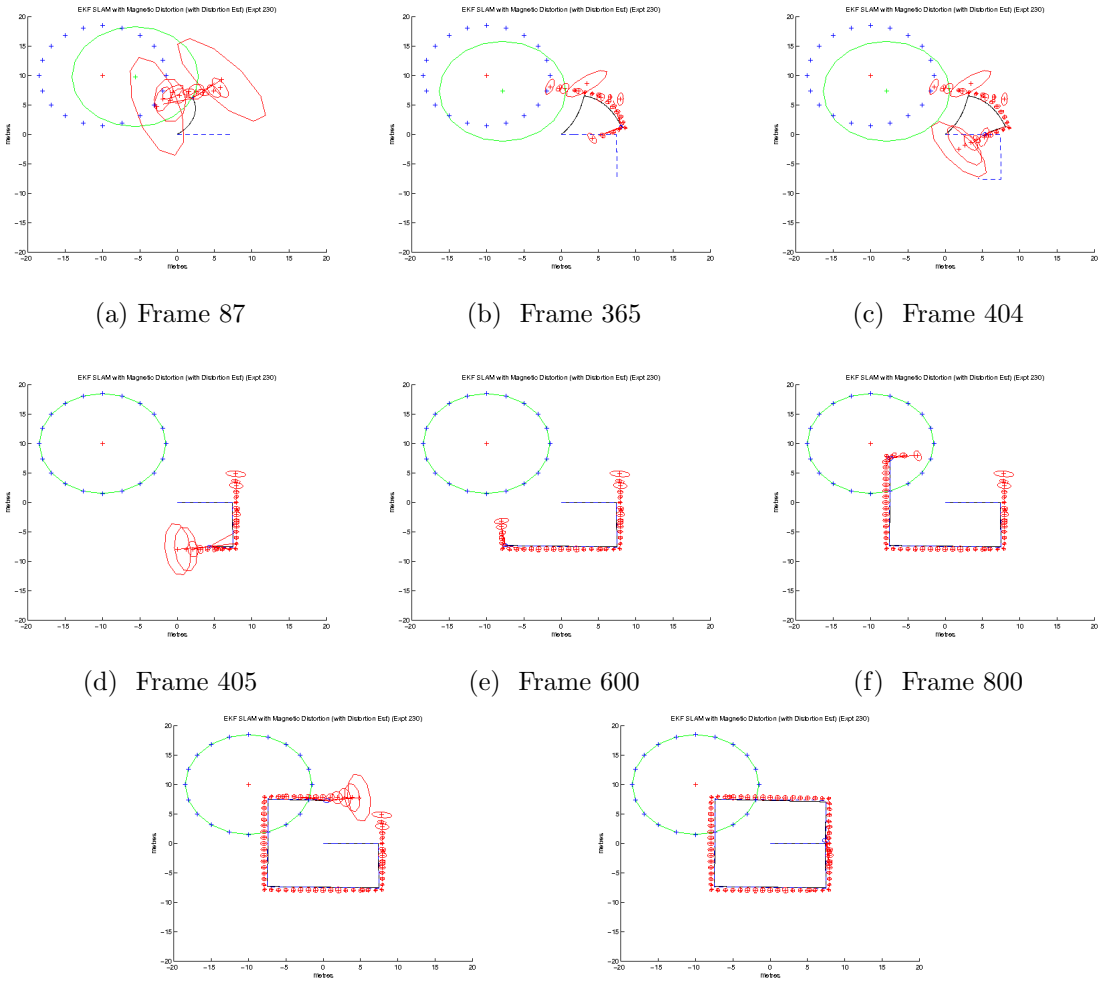


(a) Frame 43 (b) Frame 180 (c) Frame 300

(d) Frame 600 (e) Frame 800 (f) Frame 985

(g) Frame 986 (Loop Closure) (h) Frame 1200 (Final)

Figure 4.5: Example of simulation shown in Figure 4.4 using SensorSLAM. The sensor parameter (magnetic distortion location) is estimated during the run. The final map and trajectory follows the ground truth trajectory. The position of current estimated distortion location is shown with dotted blue circles. The final position of the magnetic distortion is estimated correctly in Frames 986-1200. See the legend of Figure 4.3 for more details.



(a) Frame 87 (b) Frame 365 (c) Frame 404

(d) Frame 405 (e) Frame 600 (f) Frame 800

(g) Frame 960 (h) Frame 1200 (Final)

Figure 4.6: Example of simulation with higher amount of distortion in the environment. Shown with distortion estimation using a Gaussian error model. Convergence happens at frame 405 and is stable throughout the rest of the simulation. See the legend of Figure 4.5 for more details.

4.3 Summary

This chapter developed an algorithm (SensorSLAM) for simultaneously estimating — possibly dynamic — sensor parameters that affect the quality of the estimated map and vehicle location. A general Bayesian formalism was developed and a special case was developed for applying the formulation to a specific example. A simulated example showing the necessity of such an algorithm was developed. SensorSLAM was shown to be effective at characterizing the sensor parameter dynamic model for the example. The following chapters applies the SensorSLAM algorithm to an underwater scenario using stereo-video as the primary sensing mechanism.

CHAPTER 5

Experiments

This chapter details experiments performed with data from underwater field trials to verify the approach and discuss real-world limitations of SensorSLAM. The data used in this chapter was collected during 2007 and 2008 during the AQUA field trials in Holetown, Barbados. The goal of the field trials was to collect stereo imagery of a sunken barge in sufficient detail to enable the 3D reconstruction of the entire structure. Building on the theoretical model developed in Chapter 4, this chapter provides details of the SensorSLAM algorithm used to analyze the imagery followed by validation against brute force methods showing the validity of the results.

5.1 Algorithm overview

In order to estimate the SensorSLAM pdf, $p(s_t, \Theta, P_t | z^t, u^t, n^t)$, a hybrid particle-filter EKF approach is developed. As discussed previously, with various assumptions, the SensorSLAM pdf (Equation 4.39) is described as

$$p(s_t, \Theta, P_t | z^t, u^t, n^t) \approx \eta p(z_t | s_t, \Theta, P_t, n_t) \cdot \int \underbrace{p(\Theta | P_t, s_{t-1} = \mu_{s_{t-1}}, z^{t-1}, u^t, n^t)}_{\text{Map in Global Ref. frame}} \cdot \underbrace{p(P_t | s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)}_{\text{Est. Local Sensor Bias in Map Ref. frame}} \cdot \underbrace{\left[\int p(s_t | s_{t-1}, u_t) p(s_{t-1}, \Theta, P_{t-1} | z^{t-1}, u^{t-1}, n^{t-1}) ds_{t-1} \right]}_{\text{“traditional” SLAM sub-problem}} dP_{t-1} \tag{5.1}$$

Under varying assumptions, the problem may be separated into two related sub-problems. Namely,

1. A traditional SLAM sub-problem using a fixed set of sensor parameters over each time interval.

2. Map and sensor parameter estimation using the current solution to the SLAM sub-problem.

By solving these two sub-problems, the SensorSLAM solution provides higher accuracy map and trajectory estimates since there is a model for varying (dynamically changing) sensor parameters. This is accomplished by estimating the sensor parameter pdf $p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$ using particle filtering techniques. Particle filtering samples this pdf with a discrete set of independent samples to characterize the distribution of the sensor parameters. The resampling stage in the particle filter prunes out parameters that are underperforming, keeping only good particles for the next iteration. A parameter model specific to the application is used to draw new samples from the parameter pdf.

Each sample drawn from $p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$ represents a fixed sensor parameter trajectory. Each particle contains an instance of the SLAM sub-problem which treats the sensor parameters as fixed for the duration of its lifetime (one time-step). Since there are many particles (each one representing a different sensor parameter), the system contains multiple separate instances of the SLAM sub-problem. At each step of the algorithm, the set of particles are evaluated as to their effectiveness at characterizing the map and trajectory. Particles that perform well are chosen to survive for the next iteration through a statistical resampling stage. Each time the particle sets are resampled, the parameters per particle are re-estimated based on their current error model. Resampling involves copying the appropriate data (including the map and trajectory) that is specific to the particle and in essence updating the trajectory of the parameter over time. As such, the new set of particles share a history with the ones that performed well prior to the resampling. If the sensor parameter pdf is sampled with k samples, there are k individual sub-solutions to SLAM that need to be estimated. The general problem is reduced to picking the "correct" solution at each iteration of the algorithm and utilizing the appropriate data (i.e. map and trajectory) per particle. This particular implementation uses a view-based delayed state extended Kalman filter (DSEKF)

as discussed below to provide the SLAM error estimate per sample in the particle filter.

5.1.1 Sensor Parameters

To illustrate the effectiveness of the algorithm, a single sensor parameter was chosen that can be thought of as location and map dependent. The approach in this chapter is by no means the only way to model important sensor parameters but rather the results suggest that the algorithm is robust enough to deal with situations where the sensor parameter noise-model is non-stationary (map/pose dependent). It is possible that the distortion may be reduced significantly through the use of specialized optics and hardware design, however dealing with this distortion using SensorSLAM allows for a suitable avenue to discuss the effectiveness of the algorithm. Throughout this discussion, the sensor parameter used is the first radial parameter of the camera model, $K1$. The models reconstructed in this section are of an approximately planar region of a barge (the Jolly Rogers) lying in 35 feet of water off of Holetown, Barbados. This portion of the barge was chosen as it illustrates the impact of incorrect modelling of the radial distortion parameters on the resulting 3D model. Gross deviation from a planar model is due to errors in the sensor parameter. Under a planar projective model this parameter can be estimated in a pre-calibration phase for land-based applications by viewing an appropriate calibration target. This pre-calibration approach is prone to error due to the nature of refraction underwater (see Chapter 2, Chapter 3 and [Kwon and Lindley, 2000, Kwon, 1999a, Kwon, 1999b]); the refractive index invalidates the planar projection model (see [Kunz and Singh, 2008, Treibitz et al., 2008]). This distortion may be reduced by using a hemispherical lens, but the placement of the camera relative to the lens becomes increasingly important. Any misplacement of the camera center from the center of the hemispherical lens results in an added distortion that is dependent on the distance to environmental structure, i.e., that is map and location dependent. As such, it is extremely difficult to develop a hardware solution that would work for an off-the-shelf stereo camera over a range of target distances as was used to collect the data in the AQUA field trials. The AQUASENSOR uses

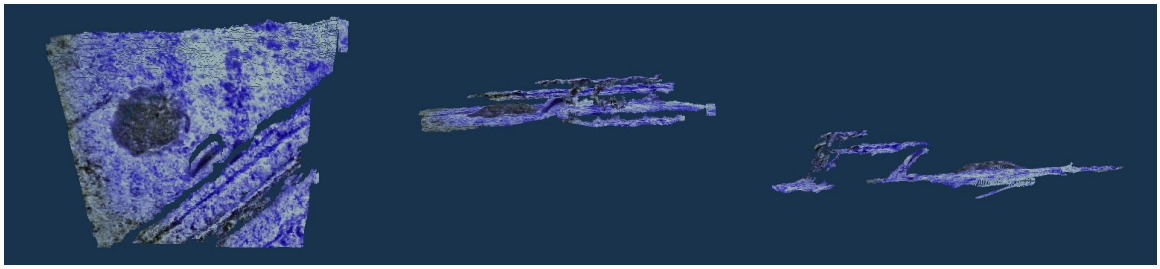
a PointGrey Bumblebee2[9] as a turnkey solution to stereo vision. The camera is pre-calibrated for land applications and the cameras are at a fixed distance apart (and therefore cannot move/rotate) maintaining the validity of the calibration information acquired prior to its use. Calibration of various sensor parameters may be performed in a pool prior to its use in the ocean however there are applications where this is not possible, for example outer-space applications. In situations where the camera hardware/lenses may distort due to pressure, temperature, or even due to shock (i.e. changing the stereo baseline) in between the calibration and deployment stages, the pool or lab calibration may become invalidated. The values of these calibrations may be a good estimate, however an online estimation/refinement of the parameters would be an asset to any autonomous vehicle.

As the radial distortion parameters change, this affects the quality (and correctness) of the 3D reconstruction as seen in Figure 5.1 and the respective final 3D models in Figure 5.2. Figure 5.1 illustrates the qualitative error that an incorrect sensor parameter can cause in a single stereo image frame. Even if the radial parameters are estimated at a higher accuracy, there are small local errors in each 3D frame due to the refraction distortion that leads to a slowly increasing global error in the model. These type of errors are insidious since the inter-frame registration error is small. That is, when computing the rotation and translation to bring the current local frame into alignment with the global model, the error in computing these values is sufficiently small until they result in a visually appealing local registration. Even though the curvature is unnoticeable in Figure 5.1a, the accumulation of the error is apparent as seen in Figure 5.2a.

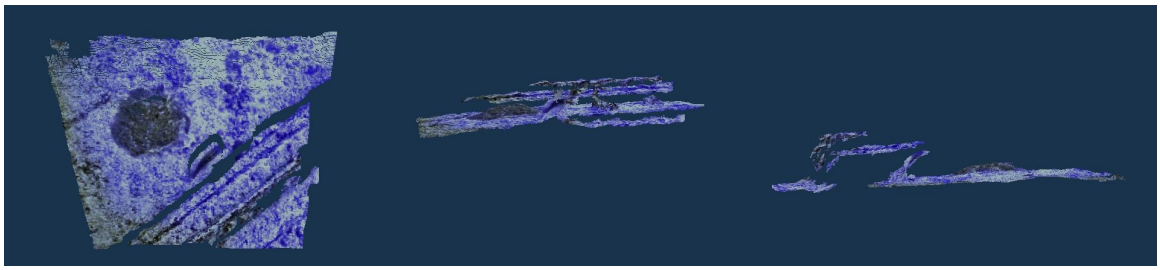
5.1.2 Particle Filtering

A particle filter is used to represent the sensor parameter pdf $p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$. The pdf is estimated by using a sequential importance sampling/resampling algorithm (SIS) as discussed in [Arulampalam et al., 2002,

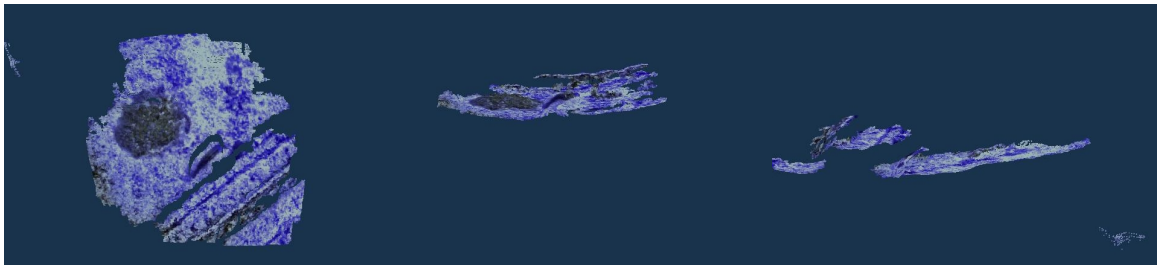
[9] http://www.ptgrey.com



(a) Top (b) Front (c) Side

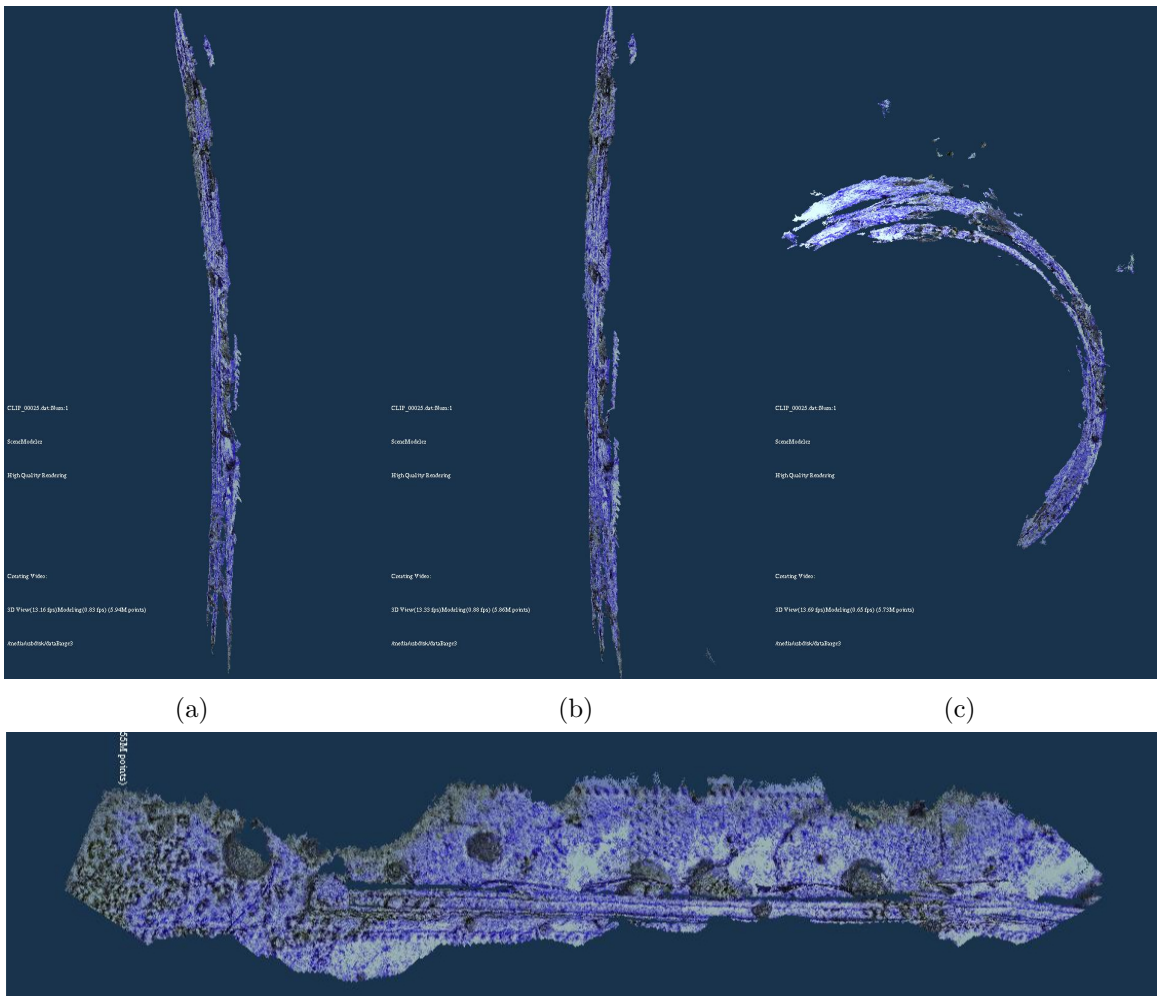


(d) Top (e) Front (f) Side



(g) Top (h) Front (i) Side

Figure 5.1: The effect of radial parameter change on 3D reconstruction. K1 is the radial parameter in left image. This image shows the first frame of the reconstruction only. Top two rows are similar and error is not noticeable locally. Bottom row contains noticeable curvature. (a-c) K1=-0.162218 (d-f) K1=-0.127218 (g-i) K1=-0.358218.



(a) (b) (c)

(d)

Figure 5.2: The effect of radial parameter change on the 3D reconstruction (315 frames) (Side view). These images show examples of a longer 3D reconstruction using different sensor parameters. Models created using fixed parameters as given in the legend of Figure 5.1. Here (a) corresponds to K1=-0.162218 (b) the second row K1=-0.127218 (c) the third row K1=-0.358218. (d) The top view of (b).

Doucet et al., 2001, Liu et al., 2001]. Given a set of m weighted random samples, $\{(x^{(j)}, w^{(j)})\}_{j=1}^{m}$, of the pdf, $p(x)$, to be estimated, an approximation of the pdf, $\hat{p}(x)$, is given as

$$\hat{p}(x) = \frac{1}{W} \sum_{j=1}^{m} w^{(j)} h(x^{(j)}) \tag{5.2}$$

where $W = \sum_{j=1}^{m} w^{(j)}$ and $h(\cdot)$ is an arbitrary integrable function.

The pdf that needs to be estimated in this manner is the sensor parameter model $p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}, z^{t-1}, u^t, n^t)$. Therefore, any sample of the particle filter is to be drawn from the distribution

$$p_t^{(j)} \sim p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t) \tag{5.3}$$

obtained by the probabilistic sensor parameter model. This model applies a non-linear function to the current values of the sensor parameters to provide an estimate of the parameter in the next time instant. Applying this to m particles gives a sampling of the desired pdf at time t. After propagating the set of particles drawn from this pdf, a new set must be sampled from the particles to properly characterize the new distribution. The new set is sampled by picking at random with probability proportional to the importance-factor $w_t^{(j)}$ of the distribution.

The desired pdf (or target distribution) is actually

$$p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^t, u^t, n^t) \tag{5.4}$$

However, at each iteration, only this pdf is known

$$p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t) \tag{5.5}$$

which is typically called the proposal distribution. In order to determine the actual value of the target (desired) pdf, an importance weight must be calculated. This is accomplished using a sequential importance sampling algorithm (SIS) as described in [Arulampalam et al., 2002, Doucet et al., 2001, Liu et al., 2001]. The importance weight is calculated as the ratio of the target and proposal distribution

pdfs

$$w_t^{(j)} = \frac{p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^t, u^t, n^t)}{p(P_t|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t)} \tag{5.6}$$

$$\overset{Bayes}{\propto} \frac{p(z_t|P_t^{(j)}, s_{t-1}, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t)p(P_t^{(j)}|s_{t-1}, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t)}{p(P_t^{(j)}|P_{t-1}^{(j)}, s_{t-1}, z^{t-1}, u^t, n^{t-1})} \tag{5.7}$$

$$\propto p(z_t|P_t^{(j)}, s_t, P_{t-1}^{(j)}, z^{t-1}, u^t, n^t) \tag{5.8}$$

$$= p(z_t|P_t^{(j)}, s_t, z^{t-1}, u^t, n^t) \tag{5.9}$$

The weight is proportional to the measurement model of the particle given the current estimate of the sensor parameters, sensor location, measurement history and motion history. In the underwater stereo-vision scenario, each measurement is the series of 3D points used to update a 3D model in the world frame. The map/model should be taken into account in order to determine the proper weight of the particle. This is accomplished by marginalizing over the model as follows‘

$$w_t^j \propto p(z_t|P_t^{(j)}, s_t, z^{t-1}, u^t, n^t) \tag{5.10}$$

$$= \int p(z_t|\Theta, P_t^{(j)}, s_t, z^{t-1}, u^t, n^t)p(\Theta|P_t^{(j)}, s_t, z^{t-1}, u^t, n^t)d\Theta \tag{5.11}$$

$$\overset{Markov}{\propto} \int p(z_t|\Theta, P_t^{(j)}, s_t)p(\Theta|P_t^{(j)}, s_{t-1}, z^{t-1}, u^{t-1}n^{t-1})d\Theta \tag{5.12}$$

where $p(\Theta|P_t^{(j)}, s_{t-1}, z^{t-1}, u^{t-1}n^{t-1})$ is computed as the planar error of the entire map and $p(z_t|\Theta, P_t^{(j)}, s_t)$ is a measurement model given from an EKF stored within the particle.

Given a plane denoted by $\Pi = \{\vec{n}, \vec{p}\}$ where $\vec{n} = [x, y, z]^T$ is a unit vector representing the normal direction of the plane and $\vec{p} = [x, y, z]^T$ is a point on the plane, and a particular 3D point in the map/model, $\Theta_i = [x, y, z]^T$, the planar error of the model is computed as

$$p(\Theta|P_t^{(j)}, s_{t-1}, z^{t-1}, u^{t-1}, n^{t-1}) \propto \sum_i^N (\vec{n} \cdot (\Theta_i - \vec{p}))^2 \tag{5.13}$$

Issues with particle filters include problems with particle diversity. Since re-sampling the distribution involves the copying of particle data and redistributing

the sensor parameter around its new mean, it is possible that the set of particles may become too similar to one another. This would eliminate the possibility of loop closing over large areas or returning to a sensor parameter that was eliminated from the particle set previously. To alleviate this problem, the particle set is not resampled at each step of the algorithm, but rather only when the number of "effective" particles decreases below some set threshold as specified in [Grisetti et al., 2005, Doucet et al., 2001]. The particles are re-distributed and sampled according to the weight distribution using a stratified random approach. The particle filtering algorithm employed is stated in Algorithm 1.

Algorithm 1: SensorSLAM particle filter algorithm

Input: Number of Particles, N, Threshold, N_{thresh}.

Output: Set of Particles, $\{p_k^{(j)}, w_k^{(j)}\}_{j=1}^{N}$, Best Particle, $\{p_{\text{best}}, w_{\max}\}$

foreach $p_k^{(j)}$ **do**

Sample PDF: $p_k^{(j)} \sim p(P_t^{(j)}|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^t, u^t, n^t)$

Update particle as in Algorithms 2 and 3

Compute particle weight:

$w_k^{(j)} \propto \int p(z_t|\Theta, P_t^{(j)}, s_t) p(\Theta|P_t^{(j)}, s_{t-1}, z^{t-1}, u_t, n^{t-1}) d\Theta$

end

Calculate total weight of distribution: $T = \sum_{j=1}^{N} w_k^{(j)}$

for $j = 1 \ldots N$ **do**

Normalize Weights: $w_k^{(j)} = \frac{w_k^{(j)}}{T}$

end

Calculate $N_{\text{eff}} = \sum_{j=1}^{N} \frac{1}{(w_k^{(j)})^2}$

if $N_{eff} < N_{thresh}$ **then**

$\{p_k^{(j)}, w_k^{(j)}\}^*|_{j=1}^{N} = \text{ResampleSIS}\{p_k^{(j)}, w_k^{(j)}\}_{j=1}^{N}$

$\{p_{\text{best}}, w_{\max}\} = \text{MAX}_w\{p_k^{(j)}, w_k^{(j)}\}$

end

$\{p_k^{(j)}, w_k^{(j)}\}_{j=1}^{N} = \{p_k^{(j)}, w_k^{(j)}\}^*|_{j=1}^{N}$

.

Each particle is required to estimate its own weight which is a function of the current map and trajectory. Since the map and trajectory are highly dependent upon the choice of sensor parameters, each particle must maintain its own estimate of the map and trajectory. Thus, within each particle is a complete sub-SLAM solution, namely the most recent estimate of s_t, Θ. This is accomplished through the use of a view-based delayed state extended Kalman filter described in the next section.

5.1.3 Delayed State EKF

Each particle implements its own SLAM algorithm which maintains the map and trajectory given a fixed/known sensor parameter value. The SLAM solution gathers the information from the stereo-video and uses the egomotion as described in Chapter 3 to estimate the delta pose (δ_{pose}) at each frame. This δ_{pose} is used as a measurement in a view-based delayed state extended Kalman filter (DSEKF) [Newman et al., 2006]. The delayed state EKF (DSEKF) was chosen for the implementation of the algorithm to minimize the amount of memory used by the EKF. The number of 3D features estimated at each step of the ego-motion algorithm is large (i.e. each stereo frame may contain upwards of $76,800$ 3D points for disparity images of resolution 320×240). Thus, for implementation of the SLAM sub-problem for SensorSLAM was chosen not to be "feature-based." The feature locations and covariances are not estimated within the SLAM state. Rather than estimating the error of each "landmark" and how it corresponds to each pose, the EKF maintains an estimate of the entire trajectory. The measurement is an estimate of how the vehicle/sensor pose has changed due to the dense set of 3D correspondences given by the stereo algorithm. An error may be computed based on the map if necessary. Using this type of EKF can be thought of as a recursive bayes filter approximation to [Lu and Milios, 1997]. The DSEKF also enables future implementations that may add large loop closing to the reconstruction system fairly easily.

Essentially, a DSEKF is simply an extended Kalman filter estimating the entire history of poses in the trajectory. The following equations follow the standard DSEKF formulation as described in [Newman et al., 2006]. Given the current pose

$\vec{p} = \left[\vec{t}, \hat{q}\right]$, where $\vec{t}$ is the world translation vector of the sensor and $\hat{q}$ is the world orientation of the sensor as represented with a quaternion, then the filter estimates the state

$$\hat{x} = [\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_{t-1}, \vec{p}_t] \tag{5.14}$$

and covariance matrix

$$\hat{P} = \begin{bmatrix} \hat{P}_{p_1p_1} & \hat{P}_{p_1p_2} & \cdots & \hat{P}_{p_1p_t} \\ \vdots & \ddots & \ddots & \vdots \\ \hat{P}_{p_tp_1} & \cdots & \cdots & \hat{P}_{p_tp_t} \end{bmatrix} \tag{5.15}$$

encodes the relationships between the poses in space. Note that the exactly Sparse Extended Information Filter (SEIF) developed in [Eustice et al., 2005a] is highly related to this algorithm and could be used in future implementations to reduce computation.

The vehicle dynamics are estimated using the control input as

$$\hat{x}^-_{t+1} = \hat{x}_t \oplus u_t \tag{5.16}$$

where $\hat{x}^-_{t+1}$ is the vehicle pose at time $t+1$ given the previous vehicle position at time t, $\hat{x}_t$ is the current vehicle pose (at time t), $\oplus$ is the composition operator defined in [Newman et al., 2006] and u_t) is the control input/odometry that is affecting the pose of the vehicle. Similarly, the covariance matrix of the distribution is estimated as

$$\hat{P}^-_{t+1} = J_1(\hat{x}_t, u_t)\hat{P}_t J_1(\hat{x}_t, u_t)^T + J_2(\hat{x}_t, u_t)U_t J_2(\hat{x}_t, u_t)^T \tag{5.17}$$

where U_T is the control input covariance matrix, u_t is the control input, $\hat{x}_t$ is the current estimated pose of the vehicle, $\hat{P}_t$ is the current estimated vehicle covariance matrix, and J_1, J_2 are the matrices of partial derivatives of the composition operator with respect to the pose and control inputs respectively. These are defined as in [Smith et al., 1990] as follows

$$J_\oplus = \frac{\partial(x_{ij} \oplus x_{jk})}{\partial(x_{ij}, x_{jk})} = \begin{bmatrix} J_1(x_{ij}, x_{jk}) & J_2(x_{ij}, x_{jk}) \end{bmatrix} \tag{5.18}$$

$$J_1(x_1, x_2) = \frac{\partial(x_1 \oplus x_2)}{\partial x_1} \tag{5.19}$$

$$J_2(x_1, x_2) = \frac{\partial(x_1 \oplus x_2)}{\partial x_2} \tag{5.20}$$

During the time update stage of the filter, the new pose is predicted by augmenting the state vector as

$$\hat{\mathbf{x}}_{t+1}^{-} = \begin{bmatrix} \hat{\mathbf{x}}_t \\ \hat{x}_t \oplus u_t \end{bmatrix} \tag{5.21}$$

and the covariance matrix is augmented as

$$\hat{\mathbf{P}}_{t+1}^{-} = \begin{bmatrix} \hat{\mathbf{P}}_t & \hat{\mathbf{P}}_t J_1(\hat{x}_t, u_t)^T \\ J_1(\hat{x}_t, u_t)\hat{\mathbf{P}}_t^T & \hat{P}_{t+1}^{-} \end{bmatrix} \tag{5.22}$$

The measurement update is the correction phase. It updates the augmented state to reflect the measurement. The standard EKF update equations can be used to update the predicted state using the measurement. This uses a measurement function $h(\cdot)$ which transforms the current state into the measurement space and an error is computed. The new state is computed as a linear sum of the previous state and the error weighted by the Kalman gain as

$$\hat{\mathbf{x}}_{t+1}^{+} = \hat{\mathbf{x}}_{t+1}^{-} + K_t(z_t - \mathbf{h}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.23}$$

and the covariance matrix is updated as

$$\hat{\mathbf{P}}_{t+1}^{+} = (I - K_t H_t)\hat{\mathbf{P}}_{t+1}^{-} \tag{5.24}$$

where the Kalman gain K_t is defined as

$$K_t = \hat{\mathbf{P}}_{t+1}^{-} H_t^T (H_t \hat{\mathbf{P}}_{t+1}^{-} H_t^T + V_t R_t V_t^T)^{-1} \tag{5.25}$$

Here, R_t is a matrix representing the measurement noise, V_t is the Jacobian of the measurement model with respect to the measurement noise, and H_t is the Jacobian of the measurement model with respect to the state. Formally, these are defined as

$$H_t = \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{x}}}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.26}$$

$$V_t = \frac{\partial \mathbf{h}}{\partial v}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.27}$$

5.1.3.1 DSEKF Implementation

In the implementation of the DSEKF used for the experiments, the control input is non-existent as there is no easy way in an underwater scenario to acquire odometry (neither for land-based handheld motion). This simplifies much of the above equations. In the time update, the following simplifications are made:

$$\hat{x}_{t+1}^{-} = \hat{x}_t \oplus u_t = \hat{x}_t \tag{5.28}$$

$$J_{\oplus} = \begin{bmatrix} J_1 & J_2 \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \tag{5.29}$$

In the measurement update, the measurement z_k used is the $\delta_{\text{pose}} = [\delta_{\vec{t}}, \delta_{\hat{q}}]$ given by the 3D registration as acquired from the stereo-video reconstruction algorithm described in Chapter 3. Thus, the observation function $h(\mathbf{x})$ must transform the current state estimate into a δ_{pose}. This can be done by assuming that the delta pose should be related to the current and previous frames and as such, the relative transformation can be computed to bring the previous frame into the current frame. This can be computed as follows

$$h(\hat{\mathbf{x}}_{t+1}^{-}, 0) = h(\hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t-1}) = \begin{bmatrix} \delta\vec{t} \\ \delta\hat{q} \end{bmatrix} = \begin{bmatrix} \vec{t}_t - \vec{t}_{t-1} \\ \hat{q}_t \hat{q}_{t-1}^{c} \end{bmatrix} \tag{5.30}$$

The measurement Jacobian matrix determines how the observation function has affected the state. The $h(\cdot)$ function uses two poses, namely the current and previous time frames, to compute the delta pose. Thus the function does not utilize the rest of poses in the state. As such, most of the measurement Jacobian are zeroes except for the final 14×14 sub-matrix. Finally, the derivatives may be computed as

$$H_t = \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{x}}}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.31}$$

$$= \begin{bmatrix} \frac{\partial h_1}{\partial x_0} & \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_{i-1}} & \frac{\partial h_1}{\partial x_i} \\ \frac{\partial h_2}{\partial x_0} & \frac{\partial h_2}{\partial x_1} & \cdots & \frac{\partial h_2}{\partial x_{i-1}} & \frac{\partial h_2}{\partial x_i} \end{bmatrix} \tag{5.32}$$

$$= \begin{bmatrix} \mathbf{0}_{7\times7} & \mathbf{0}_{7\times7} & \cdots & \frac{\partial h}{\partial x_{i-1}} & \frac{\partial h}{\partial x_i} \end{bmatrix} \tag{5.33}$$

$$\frac{\partial h}{\partial x}_{7\times7} = \begin{bmatrix} \frac{\partial h_1}{\partial x}_{3\times7} \\ \frac{\partial h_2}{\partial x}_{4\times7} \end{bmatrix} \tag{5.34}$$

$$\frac{\partial h_1}{\partial x_{i-1}} = \begin{bmatrix} -I_{3\times3} & \mathbf{0}_{3\times4} \end{bmatrix} \qquad \frac{\partial h_1}{\partial x_i} = \begin{bmatrix} I_{3\times3} & \mathbf{0}_{3\times4} \end{bmatrix} \tag{5.35}$$

$$\frac{\partial h_2}{\partial x_{i-1}} = \begin{bmatrix} I_{3\times3} & \mathbf{0}_{3\times4} \\ \mathbf{0}_{4\times3} & \frac{\partial h}{\partial \hat{q}_{i-1}} \end{bmatrix} = \begin{bmatrix} q_w^{(i)} & q_x^{(i)} & q_y^{(i)} & q_z^{(i)} \\ q_x^{(i)} & -q_w^{(i)} & q_z^{(i)} & -q_y^{(i)} \\ q_y^{(i)} & -q_z^{(i)} & -q_w^{(i)} & q_x^{(i)} \\ q_z^{(i)} & q_y^{(i)} & q_x^{(i)} & -q_w^{(i)} \end{bmatrix} \tag{5.36}$$

$$\frac{\partial h_2}{\partial x_i} = \begin{bmatrix} I_{3\times3} & \mathbf{0}_{3\times4} \\ \mathbf{0}_{4\times3} & \frac{\partial h}{\partial \hat{q}_i} \end{bmatrix} = \begin{bmatrix} q_w^{(i-1)} & q_x^{(i-1)} & q_y^{(i-1)} & q_z^{(i-1)} \\ q_x^{(i-1)} & -q_w^{(i-1)} & q_z^{(i-1)} & -q_y^{(i-1)} \\ q_y^{(i-1)} & -q_z^{(i-1)} & -q_w^{(i-1)} & q_x^{(i-1)} \\ q_z^{(i-1)} & q_y^{(i-1)} & q_x^{(i-1)} & -q_w^{(i-1)} \end{bmatrix} \tag{5.37}$$

The DSEKF algorithm proceeds by first predicting the state, then correcting the predicted state with an observed measurement by alternating the time update equations shown in Algorithm 2 with the measurement update equations shown in Algorithm 3.

Algorithm 2: SensorSLAM DSEKF algorithm (time update/Prediction)

Input: Delta Pose, $\delta_{\text{pose}} = [\vec{t}, \hat{q}]$

Output: Predicted State, $\hat{\mathbf{x}}_{t+1}^{-}$, and Covariance, $\hat{\mathbf{P}}_{t+1}^{-}$.

Augment State with Prediction:

$$\hat{x}_{t+1}^{-} = \hat{x}_t \oplus u_t = \hat{x}_t$$

$$\hat{\mathbf{x}}_{t+1}^{-} = \begin{bmatrix} \hat{\mathbf{x}}_t \\ \hat{x}_{t+1}^{-} \end{bmatrix}$$

Compute Jacobians:

$$J_1(\hat{x}_t, u_t) = \frac{\partial(\hat{x}_t \oplus u_t)}{\partial \hat{x}_t} \qquad J_2(\hat{x}_t, u_t) = \frac{\partial(\hat{x}_t \oplus u_t)}{\partial u_t}$$

Augment Covariance Matrix with Prediction:

$$\hat{P}_{t+1}^{-} = J_1(\hat{x}_t, u_t)\hat{P}_t J_1(\hat{x}_t, u_t)^T + J_2(\hat{x}_t, u_t) U_t J_2(\hat{x}_t, u_t)^T$$

$$\hat{\mathbf{P}}_{t+1}^{-} = \begin{bmatrix} \hat{\mathbf{P}}_t & \hat{\mathbf{P}}_t J_1(\hat{x}_t, u_t)^T \\ J_1(\hat{x}_t, u_t)\hat{\mathbf{P}}_t^T & P_{t+1}^{-} \end{bmatrix}$$

Algorithm 3: SensorSLAM DSEKF algorithm (measurement update)

Input: $\hat{\mathbf{x}}_{t+1}^{-}$, $\hat{\mathbf{P}}_{t+1}^{-}$, $z_k = \delta_{\text{pose}}$

Output: $\hat{\mathbf{x}}_{t+1}^{+}$, $\hat{\mathbf{P}}_{t+1}^{+}$.

Compute Measurement Function:

$$h(\hat{\mathbf{x}}_{t+1}^{-}, 0) = h(\hat{x}_t, \hat{x}_{t-1}) \tag{5.38}$$

Compute Measurement Jacobians:

$$H_t = \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{x}}}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.39}$$

$$V_t = \frac{\partial \mathbf{h}}{\partial v}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.40}$$

Compute Kalman Gain:

$$K_t = \hat{\mathbf{P}}_{t+1}^{-} H_t^T (H_t \hat{\mathbf{P}}_{t+1}^{-} H_t^T + V_t R_t V_t^T)^{-1} \tag{5.41}$$

Update:

$$\hat{\mathbf{x}}_{t+1}^{+} = \hat{\mathbf{x}}_{t+1}^{-} + K_t(z_t - \mathbf{h}(\hat{\mathbf{x}}_{t+1}^{-}, 0) \tag{5.42}$$

$$\hat{\mathbf{P}}_{t+1}^{+} = (I - K_t H_t)\hat{\mathbf{P}}_{t+1}^{-} \tag{5.43}$$

foreach $\hat{x}_i = [\vec{t_i}, \hat{q}_i]$ *in* $\hat{\mathbf{x}}_{t+1}^{+}$ **do**
    Normalize($\hat{q}_i$)
end

5.1.4 SensorSLAM Implementation Details

Due to the nature of the particle filter, the algorithm is extremely parallizable. The algorithm was developed to take advantage of this from an implementation perspective. The system was implemented using a client-server model over a local network of nine linux computers. The server manages the particle filtering algorithm and sends commands to each client for synchronization. Each client represents a sample in the particle filtering algorithm. Upon initialization, the server samples

the parameter pdf model and distributes the parameters to connected clients. Each client is instructed to perform a single frame of the reconstruction algorithm using this model and report back with its error using the desired error model. In these examples, the deviation from a planar world model is used since it is known that errors in the radial parameters will affect the 3D model by introducing a non-planarity curvature. The server collects the individual errors from each sample and resamples the pdf. Each client is responsible for its local solution and all of the details of the SLAM sub-problem. Thus, it initializes its sensor model with the parameters given by the server, loads the stereo images from disk, performs the stereo reconstruction algorithm as discussed in Chapter 3 and uses the delta pose as the measurement to the DSEKF which estimates the trajectory history. When instructed, the client is informed as to which particle information it is required to copy (during the resampling) and a new set of sensor parameters is given to it by the server. When the sample copies another sample, the new map/3D model and trajectory are loaded and the previous models/trajectory are backed up for visualization. This is mitigated by using a shared disk space between all computers on the local network to reduce the need to send huge amounts of model data over a LAN. This also impedes the performance of the system. The development of this system is not optimized for real-time performance and much information is saved at each step of the algorithm for visualization purposes that would not be required in an online scenario. One issue that plagues this algorithm is the need to have multiple 3D models (one per particle). When run on a single CPU rather than distributing over a LAN, the memory consumption would be overly large due to the choice of model data structure. This could be alleviated if the point cloud models were only used as an intermediate representation and a mesh-based data structure used as the final stored model.

Algorithm 4: SensorSLAM Server algorithm

Initialize all values

$\{c_j\}_{j=1}^{N}$ = Wait for incoming client connections

while *All Clients,* c_j*, in* $\{c_j\}_{j=1}^{N}$ *are connected* **do**

  Synchronize

  foreach c_j *in* $\{c_j\}_{j=1}^{N}$ **do**

    Draw $c_j = \{P_t^{(j)}, w_j\}$ from $p(P_t^{(j)}|s_{t-1} = \mu_{s_{t-1}}, P_{t-1}^{(j)}, z^t, u^t, n^t)$

    SEND(InitializePacket) = $\{P_t\}$ to c_j

    SEND(StartIterationPacket) to c_j

  end

  Synchronize

  $\{w_j, \text{Error}_j\}$ =CollectResults($\{c_j\}_{i=1}^{N}$)

  $\{P_t^{(j)}, w_j\}$ = DoParticleFilterAlgorithm($\{P_t^{(j)}, w_j\}$)

  Re-distribute particles to clients

end

Algorithm 5: SensorSLAM Client algorithm

Connect to Server

while *!finished sequence* **do**

  Packet = GetPacket(Server);

  if *Packet is InitPacket* **then**

    $\{P_t\}$ = GetParametersFromPacket(InitPacket);

    InitializeReconstructionWithParameters(P_t);

  else if *Packet is StartIterationPacket* **then**

    currentFrame = GetNextStereoFrame();

    $\{\delta_{\text{pose}}\}$ = ComputeEgoMotion(currentFrame, previousFrame);

    $\{\hat{\mathbf{x}}_{t+1}^{-}, \hat{\mathbf{P}}_{t+1}^{-}\}$ = DSEKF→TimeUpdate();

    $\{\hat{\mathbf{x}}_{t+1}^{+}, \hat{\mathbf{P}}_{t+1}^{+}\}$ = DSEKF→MeasurementUpdate($\hat{\mathbf{x}}_{t+1}^{-}, \hat{\mathbf{P}}_{t+1}^{-}, \delta_{\text{pose}}$);

  else if *Packet is CollectResultsPacket* **then**

    $\{w, \text{Error}\}$ = ComputeError($\Theta, \hat{\mathbf{x}}_{t+1}^{+}, \hat{\mathbf{P}}_{t+1}^{+}$);

    SEND($\{w, \text{Error}\}$) to Server

  end

end

5.2 Results

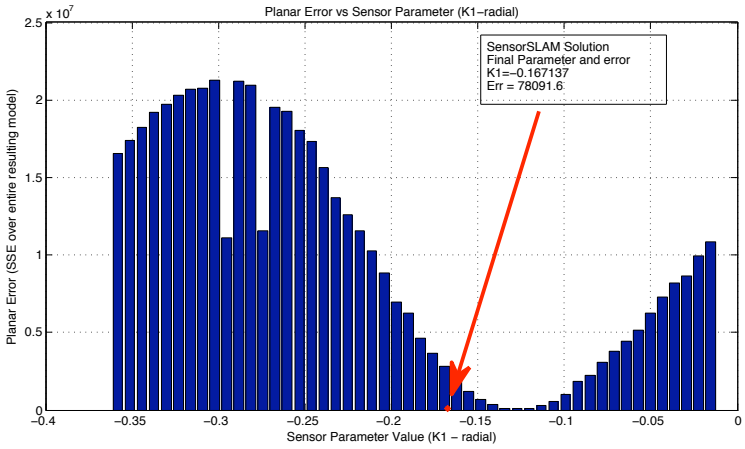
Data of a relatively planar section of the barge was chosen to mitigate the choice of error estimate per particle for these experiments. A user-specified plane is used to evaluate the data and the planar residual of each point in the model is computed as the sum of squared errors. In general, any error metric model may be used to estimate the error of the model with respect to the parameter. In this case, a planar error model was chosen to illustrate the algorithm. More complex error models that integrate smoothness of the trajectory and map or how it maps to a user supplied model (possibly recovered through a previous run that extracted a mosaic) could also be used. The covariance matrix of the SLAM solution is used to compute the normalized gaussian error of the pose with respect to the current trajectory and is weighted by the planar error as the final weight of the particle. This allows the system to favour trajectories that have maps that are more planar than others. To show how sensitive the resulting map is on the sensor parameter, 50 uniformly distributed radial parameters were chosen and used as fixed parameters throughout the entire modelling process. For this run, SensorSLAM was not used, but rather the standard reconstruction process with a fixed fixed value of the radial parameter K1 as described previously in Chapter 3 was used to extract a model after 451 image frames. At the end of each run, the planar error was estimated and recorded shown in Figure 5.3. This figure shows the distribution of the errors of the resulting model versus the sensor parameter value. The error is not a linear function of sensor parameter value and it can be seen that there are local minima in the distribution.

The SensorSLAM algorithm was deployed on the same data, over the same range of sensor parameter values. At the end of the SensorSLAM experiment, the planar error of the "best" particle was computed and recorded. This value is plotted in Figure 5.3a. As can be seen, the SensorSLAM error comes very close to the global minimum error encoded by the parameter values indicating that the algorithm performs very well with respect to the fixed parameter runs. Note that in this experiment, the SensorSLAM algorithm is allowed to modify the sensor parameter over time while the brute force methods use a fixed sensor parameter over the

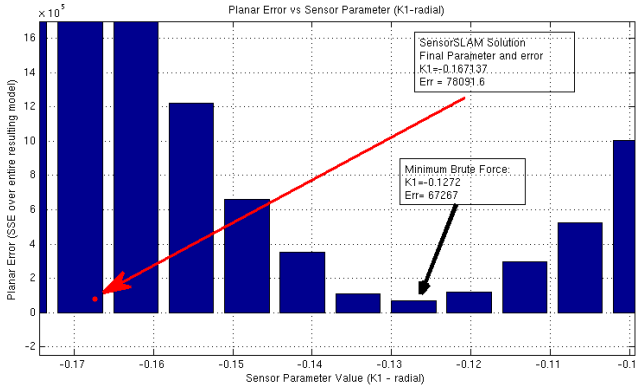
entire sequence. Tables of camera information, tracking estimation parameters, experimental parameters, and stereo parameters can be found at the end of the chapter in Tables 5.1,5.2,5.3, and 5.4. The final error plotted for SensorSLAM is the planar error of the final resulting model hence only a single value is plotted.

The resulting models from the experiment from the best particle can be seen in Figure 6.1 which shows a top and side view of the reconstructed barge. To contrast this final result, a sequence of images is shown in Figure 5.6 which shows the progression of the algorithm of a particular particle in the distribution. Figures 5.6a,c,e show the model just before the particle distribution is resampled showing the effects of choosing incorrect sensor parameters on the visual quality of the model. Figures 5.6b,d,f show the model just after the resampling stage correcting the distortion in the map and trajectory and choosing a new set of sensor parameters from the distribution.

As can be seen from these results, the estimate produced by the SensorSLAM algorithm is favorable when compared with the brute force algorithm. This is due to the ability of SensorSLAM to modify the current estimate of the sensor parameter at each iteration to minimize the planar error. Results from a second experiment on different data can be seen in Figure 5.7 and selected frames of the reconstruction can be seen in Figure 5.8. A planar error model is used here as well for the top of the barge, however due to the highly non-planarity of the environment, the algorithm retains a few artifacts. The resulting model however is usable and visually appealing.

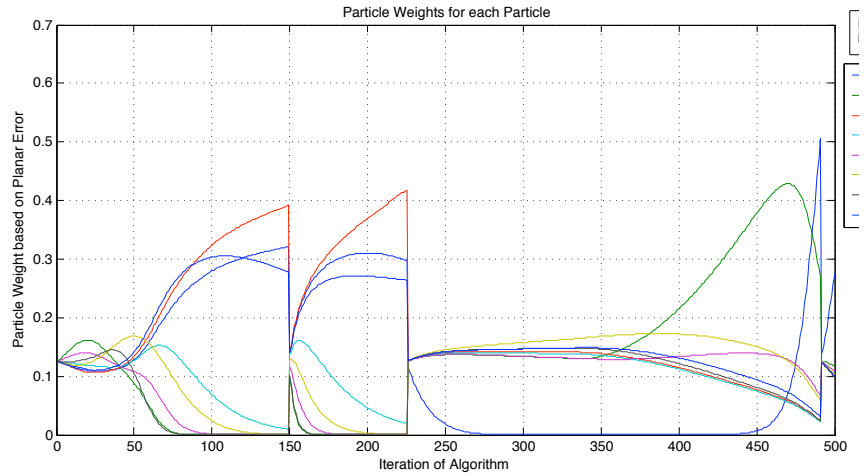


(a) Result from SensorSLAM overlaid on results from 50 runs with fixed sensor parameters.

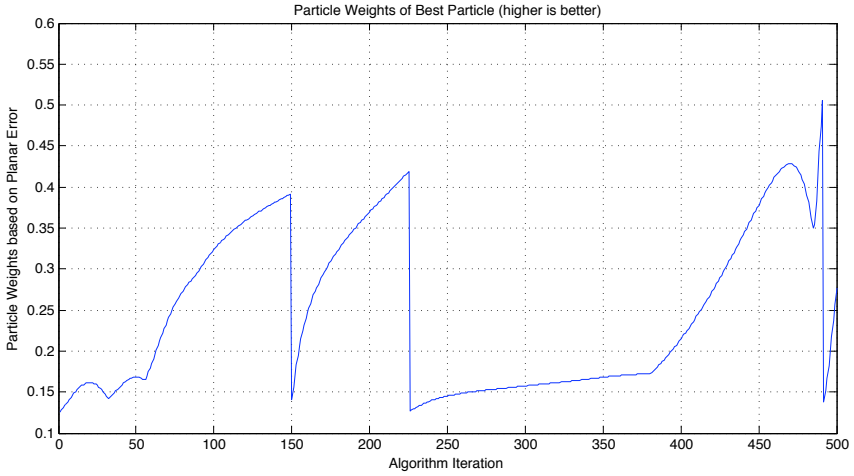


(b) Closeup of (a)

Figure 5.3: Sensitivity of 3D Model error vs choice of sensor parameter. Error is the sum of squared distances from a plane representing the barge surface of all 3D points in the final model. (a) shows the brute force approach with final SensorSLAM result overlaid. Note the local minima in the final errors. (b) Closeup of (a). Note: The SensorSLAM result was obtained over the same data and the error is computed for the final resulting model.

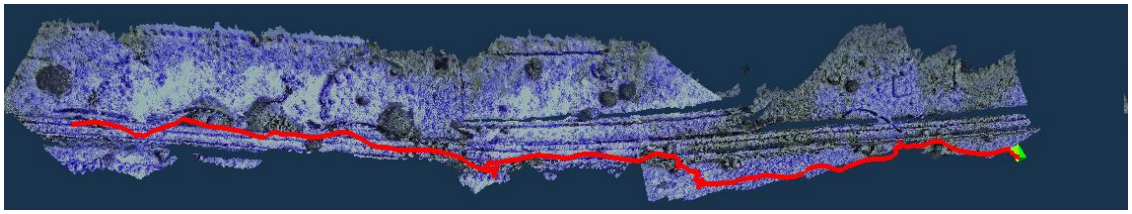


(a) Particle weights vs time for all particles.

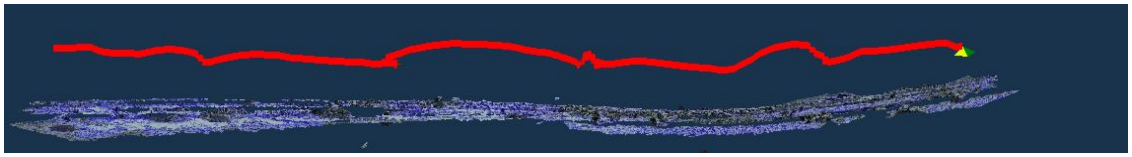


(b) Particle weight vs time of the "best" particle at each iteration.

Figure 5.4: The particle weights are an indicator of how well the sensor parameter, map and trajectory fit the planar barge model. A particle weight is between zero and one and represents a probability in the distribution, thus higher is better. Resampling stages can be clearly seen as spikes since the particle weights are normalized.



(a) Top View



(b) Side View

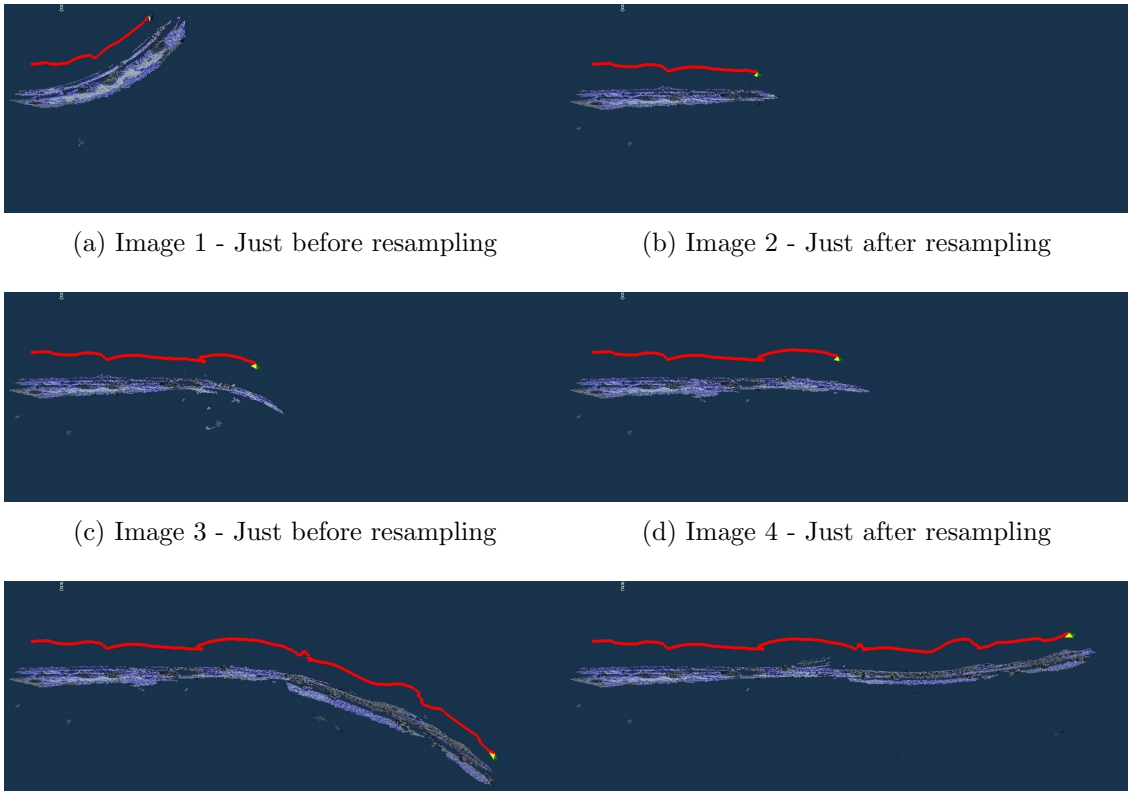
Figure 5.5: (a-b) Results of the 3D map and trajectory of the "best" particle from the SensorSLAM algorithm. Experiment was run over 451 image frames and the distance travelled was reported as 7.7 meters.

Camera Model:	Bumblebee2 Stereo Camera
Manufacturer:	Point Grey Research Inc.
Lens:	4mm
Raw Image Resolution:	640×480 pixels
Frames/second (Capture):	30fps
Field of View in Water:	H(78.78°),V(49.58°)
Focal Length:	194.85 pixels
Image Center:	$(183.88, 128.31)$pixels
Stereo Baseline:	0.119649 meters

Table 5.1: Stereo Camera Information

Disparity Image Resolution:	320×240
Frame Stride:	3 (10fps effective)
Frames/second (Ego-motion):	4fps
Frames/second (SensorSLAM):	~ 0.06 fps
Number of Features (Search):	500
Number of Features (Tracked):	263 per frame (average)
Minimum Distance between Features:	5
Feature Tracking Search Window:	5×5 pixels
Maximum Num Iterations (Tracking Refinement):	100

Table 5.2: SensorSLAM & Ego-motion Algorithm Processing Parameters



(a) Image 1 - Just before resampling

(b) Image 2 - Just after resampling

(c) Image 3 - Just before resampling

(d) Image 4 - Just after resampling

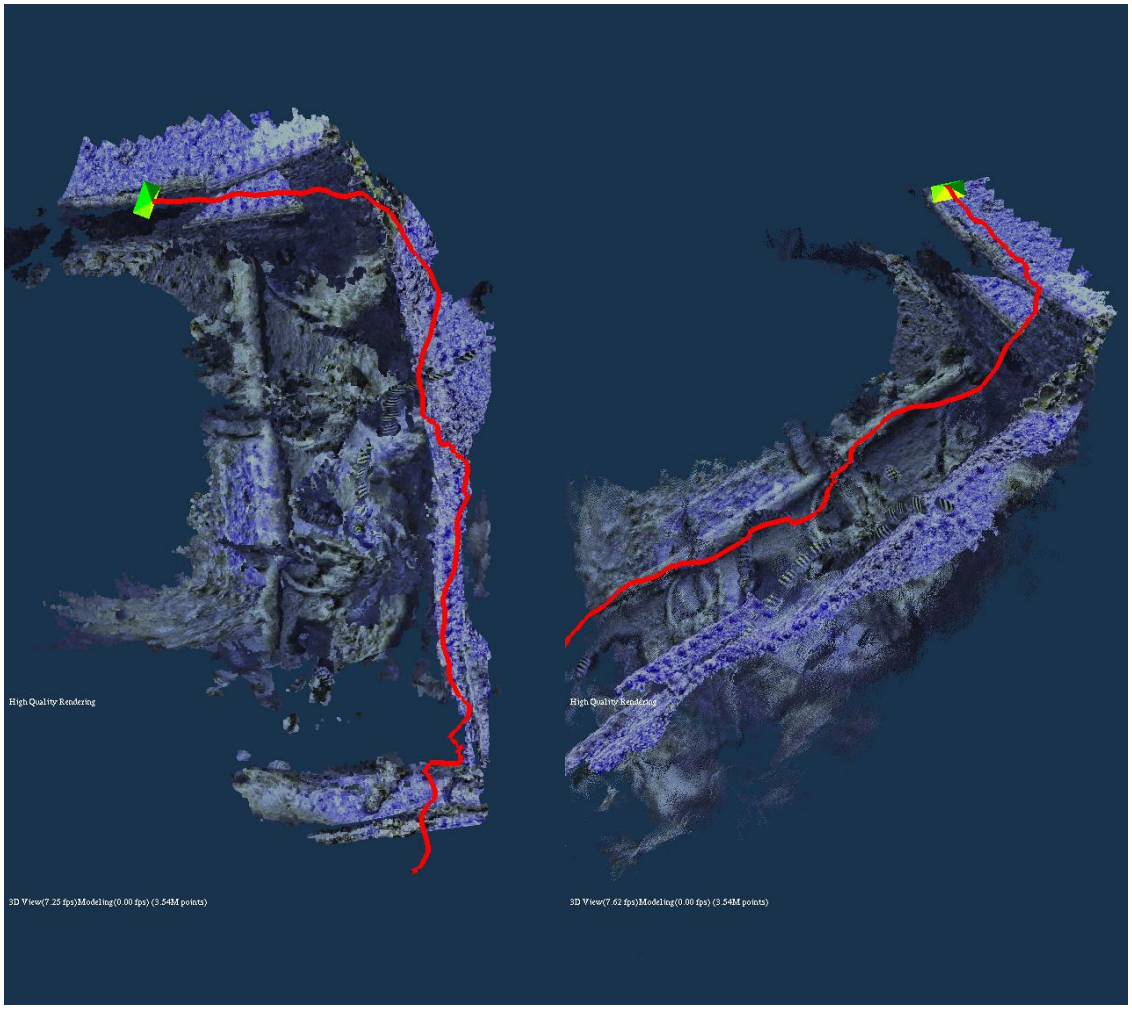
(e) Image 5 - Just before resampling

(f) Image 6 - Final model

Figure 5.6: Side view evolution of the 3D map of a single particle during the SensorSLAM algorithm. This shows the effects that resampling the sensor parameters has on a single particle. At each resampling, a new sensor parameter is chosen from the distribution and the history of the "best-so-far" particle is used as its own history. The final model shows the removal of the curvature from the map.

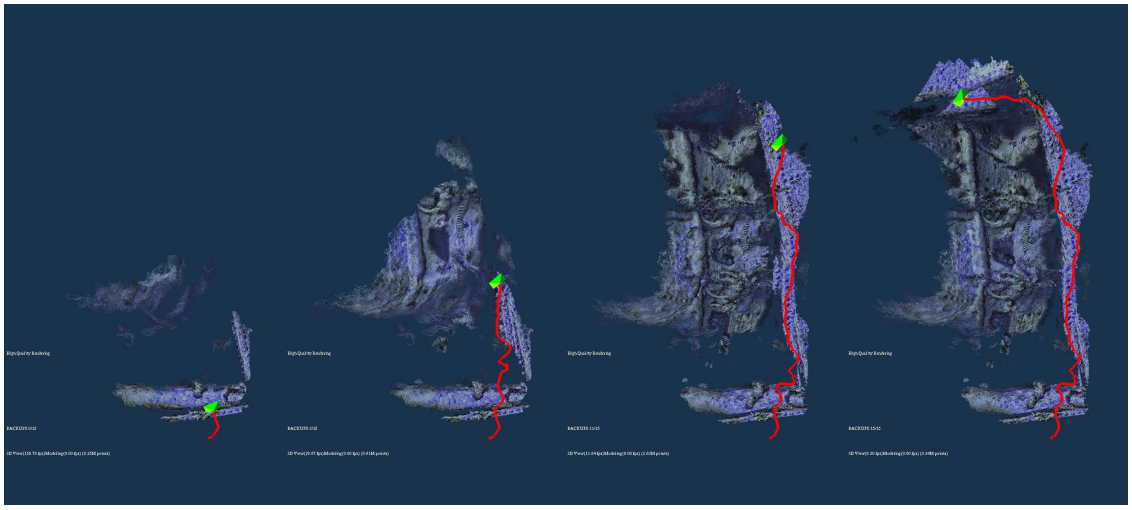
Average Distance to target:	2meters
Average Size of pixels:	$\sim$ 1.03cm/pixel
Number of Frames (Expt. 1):	451
Number of Frames (Expt. 2. nonplanar):	210

Table 5.3: Experimental Parameters



(a) Top View (b) Perspective View

Figure 5.7: Results from another experiment to show that the algorithm works for highly-non-planar environments. Due to the nature of the planar error function a few artifacts are present in the resulting model.



(a) (b) (c) (d)

Figure 5.8: Top view of selected frames in experiment from Figure 5.7.

Edge Correlation:	ON
Back-Forth Validation:	ON
Lowpass Disparity Filter:	ON
Surface Validation:	ON
Texture Validation:	OFF
Uniqueness Validation:	OFF
Subpixel Estimation:	ON
Strict Subpixel:	ON
Rectification Quality:	TriRectQlty_FAST
Stereo Quality:	TriStereoQlty_STANDARD
Stereo Mask Size:	15
Edge Mask Size:	15
Texture Validation Threshold:	0.40
Uniqueness Validation Threshold:	0.80
Surface Validation Size:	100
Surface Validation Difference:	0.50
Disparity Range:	Min:10 Max:240 Offset:0

Table 5.4: Triclops Stereo SDK Parameters

CHAPTER 6

Discussion

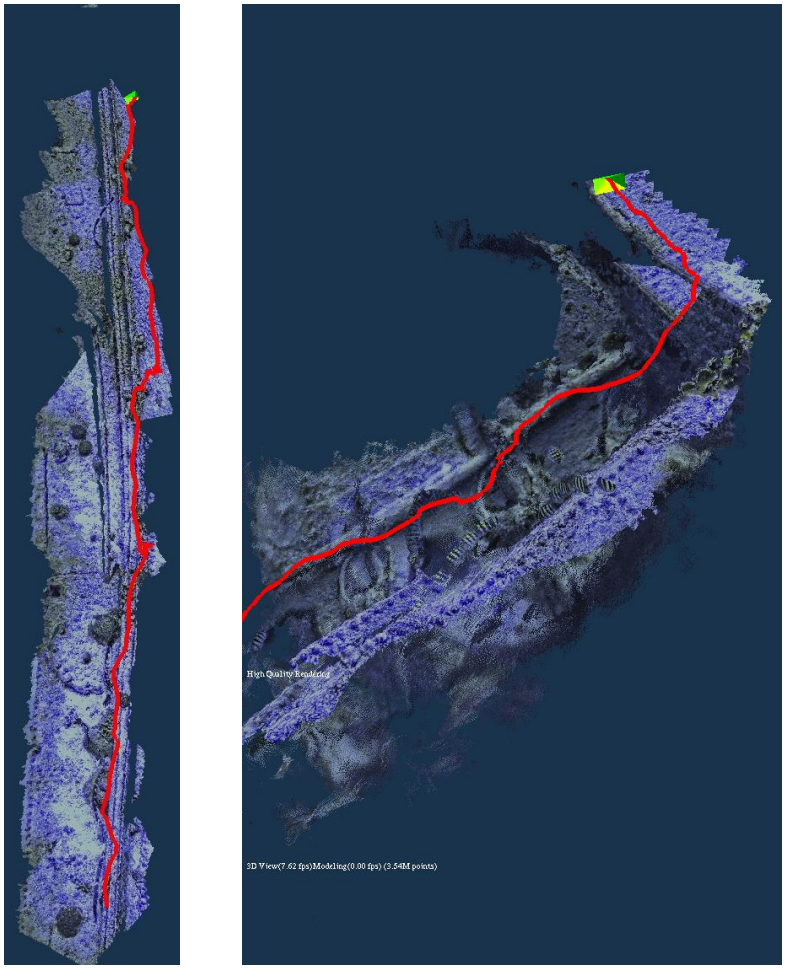
A critical limiting assumption with existing SLAM formulations is their inability to address the problem of non-stationary sensor noise. If sensor performance degrades due to the orientation or position of the vehicle in the environment then standard noise models (which assume stationary noise) are invalidated and this leads to instability and loss of accuracy in the resulting map. The standard approach to modeling sensors within the SLAM framework is to develop a static sensor parameter model that includes many different factors that affect the measurements. Each probabilistic density function is tuned to correspond to a particular type of error such as local measurement noise, unexpected objects appearing in the scene, sensor failure, and random measurements.

This dissertation developed the thesis incorporates sensor parameter estimation into the SLAM formulation providing increased robustness in the presence of dynamic location-based sensor errors. This is useful when utilizing sensors which provide erroneous data whose distribution changes when measuring the process from different locations, orientations, or are prone to errors due to unforeseeable environmental factors. The algorithm has been shown to be effective using simulated data as well as when applied to real data obtained in an underwater robotics setting for performing dense reconstruction of underwater environments.

6.1 Directions for future work

The algorithm and underwater implementation are not without their assumptions that could be relaxed in future research. Assumptions of this implementation of SensorSLAM are

- *Static world assumption.* The environment is assumed to be stationary. To



(a) Expt 1. (b) Expt 2.

Figure 6.1: Final resulting 3D models from two separate experiments.

overcome these limitations, a more general model, such as the ones discussed in [Wang, 2004, Kodagoda et al., 2006, Wang et al., 2007a], that incorporates the estimation of moving objects should be developed and integrated into the framework. This would help in situations where a large number of fish enter the scene and impact the the ego-motion estimate.

- *No control input.* Vehicle dynamics are not incorporated into the current implementation. This is due to the fact that the robot platform was not used for data collection but rather a handheld sensor was used. Thus there is no information as to what was the intended motion of the vehicle or as to kinematic or dynamic constraints on the motion of the sensor. To aid this process, a more sophisticated hydrodynamic model could be developed that utilizes more sensors to better estimate the motion of the vehicle.

- *Map error based on planar model.* For the purposes of this dissertation, the error model used for estimating the effectiveness of each particle is based on a planar residual. This is quite limiting in the type of environments that can be represented. A more generalized error model should be developed that enables a more generalized set of geometry. Given prior knowledge of the environment, obtained through the use of lower accuracy complimentary sensors, or through a manual 3D model generated my hand, this information could be incorporated into the geometric error function. At each step of the particle filter, the algorithm may compute the likelihood of being in a location in this a priori map given the sensor parameters, observations and robot location estimates. This likelihood would be used to weight the particles mitigating the effectiveness of the resampling algorithm. Without prior knowledge, saliency information (such as the operators described in [Lowe, 2004] or [Nicosevici et al., 2007]) may be incorporated to estimate the error of each particle identifying possible loop-closing events. This would allow the algorithm to weight particles that close loops so they are sampled more often during the resampling stage.

Issues that could be approached in future developments could include

- Real-time performance. There are several bottlenecks in the current implementation. Due to the sheer volume of data that is needed during a single iteration of the algorithm, the algorithm was distributed over a LAN of nine computers. Network communication and synchronization of the slaves was over-engineered to ensure that all particles contained the most recent information. This made the algorithm run much slower than it could with the more efficient synchronization. Also, a large disk array was used as the main method for sharing data between particles rather than sending gigabytes over the network each frame. As such, the algorithm slows down as the models grow in size since each particle saves, moves, and copies upwards of 2GB of data per iteration as it nears completion. Competing for disk access causes the particles to stall lowering the performance.

- Data structure and data management. Since there is a very large amount of data used in the process of this algorithm, a more efficient data structure should be used. Tests were performed with triangular meshs, and octrees throughout the development of this implementation, however for display purposes, the most qualitative data structure was an unorganized point cloud. More flexible data structures should be incorporated directly into the SLAM framework to aid in the error reduction per particle. Such an implementation might utilize triangular meshes, [Gibson, 1998], implicit surfaces, [Bloomenthal, 1988], distance fields, [Jones et al., 2006], or hierarchical level sets, [Houston et al., 2006], or a piecewise planar representation as described in [Nicosevici et al., 2005, Nicosevici et al., 2007] as the base data structure. This would allow for dimensionality reduction and the possibility of estimating the parameters of these surfaces directly in the SLAM framework.

- Loop Closing. Loop closing is beyond the scope of this thesis, however the implementation currently supports this in theory. It wouldn't be difficult to incorporate loop closing if an algorithm was used to determine whether the robot has been in this location previously. This could be accomplished by using SIFT features [Lowe, 2004] to represent salient areas (although underwater SIFT features are not as salient as one might think) as used in

[Se et al., 2002]. Once it is known that the robot has been in a particular location previously, the DSEKF can be updated simply by computing the appropriate $J_{\oplus}$ matrix for each of the poses in the trajectory. This would be very similar to a single iteration of the well-known scan-matching algorithm of Lu and Milios [Lu and Milios, 1994].

- Informed stereo sensing algorithm. To fully incorporate the SensorSLAM ideas into the entire process, the stereo algorithm could be informed and the parameters that influence the disparity extraction could also be refined. This could lead to a more accurate set of disparities that more appropriately reflects the environment. In areas that have overlap or have already been explored, the existing map could be used as an initial guess for disparities. Thus, as the map becomes populated, the time it takes to extract 3D information could be improved. This may also be useful for re-evaluation of the environment in applications such as ship-hull inspection or the determination of coral-reef health, as the differences in 3D information can be an indication of temporal changes in the environment.

- Integration into the AQUA robot platform. The algorithm was employed off-line on data collected with the AQUASENSOR rather than running on the actual robot. Future work that increased performance of the SensorSLAM algorithm could integrate it directly into the robot control architecture. This would mitigate the autonomous control as an accurate map/trajectory of the environment is essential.

Bibliography

[Abdel-Aziz and Karara, 1971] Abdel-Aziz, Y. I. and Karara, H. M. (1971). Direct linear transformation into object space coordinates in close-range photogrammetry. In *Proc. of the Symposium on Close-Range Photogrammetry*, pages 1–18, Urbana, Illinois.

[Armstrong et al., 2006] Armstrong, R., Singh, H., Torres, J., Nemeth, R., Can, A., Roman, C., Eustice, R., Riggs, L., and Garcia-Moliner, G. (2006). Characterizing the deep insular shelf coral reef habitat of the Hind Bank marine conservation district (US Virgin Islands) using the Seabed autonomous underwater vehicle. *Continental Shelf Research*, 26(2):194–205.

[Arulampalam et al., 2002] Arulampalam, S., Maskell, S., Gordon, N., and Clapp, T. (2002). A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188.

[Bærentzen, 2002] Bærentzen, J. A. (2002). *Manipulation of volumetric solids with applications to sculpting*. PhD thesis, Informatics and Mathematical Modelling, Technical University of Denmark, Copenhagen, Denmark. Supervised by Niels Jørgen Christensen, IMM.

[Bailey et al., 2006] Bailey, T., Nieto, J., and Nebot, E. (2006). Consistency of the FastSLAM algorithm. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA '06)*, pages 424–429.

[Barfoot, 2005] Barfoot, T. (2005). Online visual motion estimation using FastSLAM with SIFT features. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, pages 579–585.

[Besl and McKay, 1992] Besl, P. and McKay, N. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI'92)*, 14(2):239–256.

[Biddle, 2003] Biddle, D. (2003). Inertial Based Control on the Kambara Project. Undergraduate Thesis, The Australian National University, Canberra, Australia.

[Bischoff and Kobbelt, 2003] Bischoff, S. and Kobbelt, L. (2003). Sub-voxel topology control for level-set surfaces. *Computer Graphics Forum*, 22(8):273–280.

[Bloomenthal, 1988] Bloomenthal, J. (1988). Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5(4):341–355.

[Bone et al., 1994] Bone, R., Bruzzone, G., Caccia, M., Grassia, F., Spirandelli, E., and Veruggio, G. (1994). ROBY goes to Antarctica. In *Proc. of OCEANS '94*, volume 3, pages 621–625.

[Bono et al., 1998] Bono, R., Bruzzone, G., Caccia, M., Spirandelli, E., and Veruggio, G. (1998). Romeo goes to Antarctica. In *Proc. of OCEANS '98*, volume 3, pages 1568–1572.

[Bono et al., 1999] Bono, R., Caccia, M., Spirandelli, E., and Veruggie, G. (1999). ROV exploration of the keel of the Campbell Ice Tongue in Antarctica. In *Proc. of MTS/IEEE OCEANS '99*, volume 2, pages 563–566.

[Bono et al., 1994] Bono, R., Caccia, M., and Veruggio, G. (1994). Reconstructing 2-D maps from multiple sonar scans. In *Proc. of OCEANS '94*, volume 1, pages 164–169.

[Bosse et al., 2003] Bosse, M., Newman, P., Leonard, J., and Teller, S. (2003). An Atlas Framework for Scalable Mapping. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'03)*, volume 2, pages 1899–1906.

[Bouguet, 1999] Bouguet, J.-Y. (1999). *Visual methods for three-dimensional modeling*. PhD thesis, California Institute of Technology, Pasadena, California, USA. Supervisor-Pietro Perona.

[Braunl et al., 2004] Braunl, T., Boeing, A., Gonzales, L., Koestler, A., Nguyen, M., and Petitt, J. (2004). The autonomous underwater vehicle initiative - project Mako. In *Proc. of the IEEE Conference on Robotics, Automation and Mechatronics*, volume 1, pages 446–451.

[Brierley et al., 2002] Brierley, A. S., Fernandes, P. G., Brandon, M. A., Armstrong, F., Millard, N. W., McPhail, S. D., Stevenson, P., Pebody, M., Perrett, J., Squires, M., Bone, D. G., and Griffiths, G. (2002). Antarctic Krill under Sea Ice: Elevated Abundance in a Narrow Band Just South of Ice Edge. *Science*, 295(5561):1890–1892.

[Bruzzone et al., 2001] Bruzzone, G., Bono, R., Caccia, M., and Veruggio, G. (2001). A simulation environment for unmanned underwater vehicles development. In *Proc. of MTS/IEEE OCEANS '01*, volume 2, pages 1066–1072.

[Bryant, 2000] Bryant, M. (2000). A vision system for an autonomous underwater vehicle. Undergraduate Thesis, The Australian National University, Canberra, Australia.

[Bryson and Sukkarieh, 2007] Bryson, M. and Sukkarieh, S. (2007). Building a Robust Implementation of Bearing-only Inertial SLAM for a UAV. *Journal of Field Robotics*, 24(1-2):113–143.

[Buhmann et al., 1995] Buhmann, J., Burgard, W., Cremers, A., Fox, D., Hofmann, T., Schneider, F., Strikos, J., and Thrun, S. (1995). The Mobile Robot Rhino. *AI Magazine*, 16(1):31–38.

[Burgard et al., 1999] Burgard, W., Cremers, A., Fox, D., Hahnel, D., Lakemeyer, G., Schulz, D., Steiner, W., and Thrun, S. (1999). Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2):3–55.

[Caccia, 2006] Caccia, M. (2006). Laser-Triangulation Optical-Correlation Sensor for ROV Slow Motion Estimation. *IEEE Journal of Oceanic Engineering*, 31(3):711–727.

[Caccia et al., 2001] Caccia, M., Bono, R., Bruzzone, G., Spirandelli, E., and Veruggio, G. (2001). Integration and sea trials of ARAMIS with the Romeo ROV. In *Proc. of MTS/IEEE OCEANS '01*, volume 2, pages 1129–1136.

[Caccia et al., 2005] Caccia, M., Bono, R., Bruzzone, G., Spirandelli, E., Veruggio, G., Stortini, A., and Capodaglio, G. (2005). Sampling sea surfaces with

SESAMO: an autonomous craft for the study of sea-air interactions. *IEEE Robotics and Automation Magazine*, 12(3):95–105.

[Caccia et al., 1999] Caccia, M., Bono, R., Bruzzone, G., and Veruggio, G. (1999). Variable-configuration UUVs for marine science applications. *IEEE Robotics and Automation Magazine*, 6(2):22–32.

[Caccia et al., 2000] Caccia, M., Indiveri, G., and Veruggio, G. (2000). Modeling and identification of open-frame variable configuration unmanned underwater vehicles. *IEEE Journal of Oceanic Engineering*, 25(2):227–240.

[Caccia et al., 1997] Caccia, M., Veruggio, G., Casalino, G., Alloisio, S., Grosso, C., and Cristi, R. (1997). Sonar-based bottom estimation in UUVs adopting a multi-hypothesis extended Kalman filter. In *Proc. of the 8th International Conference on Advanced Robotics (ICAR'97)*, pages 745–750.

[Caiti et al., 2006] Caiti, A., Casalino, G., Conte, G., and Zanoli, S. M. (2006). Innovative technologies in underwater archaeology: field experience, open problems, and research lines. *Chemistry and Ecology*, 22(4 supp 1):383–396.

[Casella and Robert, 1996] Casella, G. and Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94.

[Chatila and Laumond, 1985] Chatila, R. and Laumond, J. (1985). Position Referencing and Consistent World Modeling for Mobile Robots. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'85)*, pages 138–145.

[Chatterjee and Roychowdhury, 2000] Chatterjee, C. and Roychowdhury, V. (2000). Algorithms for coplanar camera calibration. *Machine Vision and Applications*, 12(2):84–97.

[Choset and Nagatani, 2001] Choset, H. and Nagatani, K. (2001). Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137.

[Civera et al., 2007] Civera, J., Davison, A. J., and Montiel, J. M. M. (2007). Dimensionless monocular slam. In Martí, J., Benedí, J.-M., Mendonça, A. M., and Serrat, J., editors, *Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'07)*, volume 4478 of *Lecture Notes in Computer Science*, pages 412–419. Springer.

[Civera et al., 2008] Civera, J., Davison, A. J., and Montiel, J. M. M. (2008). Interacting multiple model monocular slam. In *Proc. of the IEEE/RSJ International Conference on Robots and Automation (ICRA'08)*, pages 3704–3709, Pasadena, California.

[Clarke, 2003] Clarke, T. (2003). Oceanography: Robots in the deep. *Nature*, 421(6922):p468.

[Coppersmith and Winograd, 1987] Coppersmith, D. and Winograd, S. (1987). Matrix multiplication via arithmetic progressions. In *Proc. of the nineteenth annual ACM symposium on Theory of computing (STOC '87)*, pages 1–6, New York, NY, USA. ACM.

[Cox and Wermuth, 2003] Cox, D. and Wermuth, N. (November 2003). A general condition for avoiding effect reversal after marginalization. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(5):937–941.

[Cox, 1991] Cox, I. (1991). Blanche - an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204.

[Cox, 1993] Cox, I. (1993). A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10(1):53–66.

[Criminisi et al., 2000] Criminisi, A., Reid, I., and Zisserman, A. (2000). Single view metrology. *International Journal of Computer Vision*, 40(2):123–148.

[Csorba, 1997] Csorba, M. (1997). *Simultaneous Localisation and Map Building*. PhD thesis, Robotics Research Group, Department of Engineering Science, University of Oxford, Oxford, UK.

[Davison, 1998] Davison, A. (1998). *Mobile Robot Navigation using Active Vision*. PhD thesis, Robotics Research Group, Department of Engineering Science, University of Oxford, Oxford, UK.

[Davison, 2003] Davison, A. (2003). Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *Proc. of the 9th International Conference on Computer Vision (ICCV'03)*, pages 1403–1410.

[Davison and Murray, 1998] Davison, A. J. and Murray, D. W. (1998). Mobile robot localisation using active vision. In *Proc. of the 5th European Conference on Computer Vision (ECCV '98)*, volume 2, pages 809–825, London, UK. Springer-Verlag.

[Davison and Murray, 2002] Davison, A. J. and Murray, D. W. (2002). Simultaneous localization and map-building using active vision. *IEEE Tranactions on Pattern Analysis Machine Intelligence*, 24(7):865–880.

[Davison et al., 2007] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067.

[Diebel et al., 2004] Diebel, J., Reutersward, K., Thrun, S., Davis, J., and Gupta, R. (2004). Simultaneous Localization and Mapping with Active Stereo Vision. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'04)*, volume 4, pages 3436–3443.

[Doucet et al., 2001] Doucet, A., Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer-Verlag.

[Doucet et al., 2000] Doucet, A., Freitas, N., Murphy, K., and Russell, S. (2000). Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'00)*, pages 176–183.

[Dudek et al., 2007] Dudek, G., Giguere, P., Prahacs, C., Saunderson, S., Sattar, J., Torres, L., Jenkin, M., German, A., Hogue, A., Ripsman, A., Zacher, J.,

Milios, E., Liu, H., Zhang, P., Buehler, M., and Georgiades, C. (2007). AQUA: An Amphibious Autonomous Robot. *IEEE Computer*, 40(1):46–53.

[Dunbabin et al., 2004] Dunbabin, M., Roberts, J., Usher, K., and Corke, P. (2004). A New Robot for Environmental Monitoring on the Great Barrier Reef. In Barnes, N. and Austin, D., editors, *Proc. of the Australasian Conference on Robotics and Automation (ACRA'04)*, Canberra, Australia.

[Dunbabin et al., 2005a] Dunbabin, M., Roberts, J., Usher, K., and Corke, P. (2005a). A Hybrid AUV Design for Shallow Water Reef Navigation. In *Proc. of the International Conference on Robotics and Automation (ICRA'05)*, pages 2105–2110.

[Dunbabin et al., 2006] Dunbabin, M., Roberts, J., Usher, K., and Winstanley, G. (2006). A vision for saving the reef - the Starbug AUV. In *Video Proc. of the IEEE International Conference on Robotics and Automation (ICRA'06)*, Florida, USA. [video].

[Dunbabin et al., 2005b] Dunbabin, M., Usher, K., and Corke, P. (2005b). Visual motion estimation for an autonomous underwater reef monitoring robot. In *Proc. of the International Conference on Field and Service Robotics*, pages 57–68, Port Douglas, Australia.

[Duntley, 1963] Duntley, S. (1963). *Light in the Sea*, volume 53 of *Journal of the Optical Society of America*, pages 214–233. Optical Society of America, 2nd edition.

[Durrant-Whyte et al., 2003] Durrant-Whyte, H., Majumder, S., Thrun, S., de Battista, M., and Scheding, S. (2003). *A Bayesian Algorithm for Simultaneous Localisation and Map Building*, volume 6 of *Springer Tracts in Advanced Robotics*, pages 49–60. Springer Berlin / Heidelberg.

[Elfes, 1989a] Elfes, A. (1989a). *Occupancy grids: a probabilistic framework for robot perception and navigation.* PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

[Elfes, 1989b] Elfes, A. (1989b). Using Occupancy Grids for Mobile Robot Perception and Navigation. *IEEE Computer*, 22(6):46–57.

[Elfes et al., 1998] Elfes, A., Bueno, S., Bergerman, M., Ramos, J., and Gomes, S. (1998). AURORA: development of an autonomous unmanned remote monitoring robotic airship. *Journal of the Brazilian Computer Society*, 4(3):70–78.

[Eliazar and Parr, 2003] Eliazar, A. and Parr, R. (2003). DP-SLAM: Fast, Robust Simultaneous Localization and Mapping Without Predetermined Landmarks. In *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, pages 1135–1142.

[Eliazar and Parr, 2004] Eliazar, A. and Parr, R. (2004). DP-SLAM 2.0. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'04)*, pages 1314–1320.

[Estrada et al., 2005] Estrada, C., Neira, J., and Tardos, J. (2005). Hierarchical slam: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596.

[Eustice, 2005] Eustice, R. (2005). *Large-Area Visually Augmented Navigation for Autonomous Underwater Vehicles*. PhD thesis, MIT - Woods Hole Oceanographic Institute, Woods Hole, MA, USA.

[Eustice et al., 2005a] Eustice, R., Singh, H., and Leonard, J. (2005a). Exactly Sparse Delayed-State Filters. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'05)*, pages 2417–2424.

[Eustice et al., 2005b] Eustice, R., Singh, H., Leonard, J., Walter, M., and Ballard, R. (2005b). Visually Navigating the RMS Titanic with SLAM Information Filters. In *Proc. of Robotics: Science and Systems (RSS'05)*, pages 54–64, Cambridge, USA.

[Eustice et al., 2005c] Eustice, R., Walter, M., and Leonard, J. (2005c). Sparse Extended Information Filters: Insights into Sparsification. In *Proc. of the IEEE*

International Conference on Robotics and Automation (ICRA'05), pages 641–648.

[Everett, 1995] Everett, H. R. (1995). *Sensors for mobile robots: theory and application*. A. K. Peters, Ltd., Natick, MA, USA.

[Fairfield et al., 2006] Fairfield, N., Kantor, G., and Wettergreen, D. (2006). Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'06)*, pages 3575–3580.

[Fairfield et al., 2007] Fairfield, N., Kantor, G., and Wettergreen, D. (2007). Real-Time SLAM with Octree Evidence Grids for Exploration in Underwater Tunnels. *Journal of Field Robotics*, 24(1-2):3–21.

[Faugeras and Luong, 2001] Faugeras, O. and Luong, Q.-T. (2001). *The Geometry of Multiple Images*. The MIT Press, Cambridge, UK.

[Fischler and Bolles, 1981] Fischler, M. and Bolles, R. (1981). Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–385.

[Fitzgerald, 1999] Fitzgerald, I. (1999). A vision system for an autonomous underwater vehicle. Undergraduate Thesis, The Australian National University, Canberra, Australia.

[Foley et al., 1990] Foley, J., van Dam, A., Feiner, S., and Hughes, J. (1990). *Computer Graphics: Principles and Practice, second edition*. Addison-Wesley Professional.

[Fox, 1998] Fox, D. (1998). *Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Navigation*. PhD thesis, Institute of Computer Science III, University of Bonn, Bonn, Germany.

[Fox et al., 2001] Fox, D., Thrun, S., Dellaert, F., and Burgard, W. (2001). Particle filters for mobile robot localization. In Doucet, A., de Freitas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, New York. Springer.

[Frese and Hirzinger, 2001] Frese, U. and Hirzinger, G. (2001). Simultaneous Localization and Mapping - A Discussion. In *Proc. of the International Joint Conferences on Artificial Intelligence (IJCAI) Workshop on Reasoning with Uncertainty in Robotics*.

[Frisken et al., 2000] Frisken, S. F., Perry, R. N., Rockwood, A. P., and Jones, T. R. (2000). Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proc. of ACM SIGGRAPH '00*, pages 249–254, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[Furushima et al., 2004] Furushima, Y., Yamamoto, H., Maruyama, T., Ohyagi, T., Yamamura, Y., Imanaga, S., Fujishima, S., Nakazawa, Y., and Shimamura, A. (2004). Necessity of bottom topography measurements in coral reef regions. In *Proc. of MTS/IEEE OCEANS '04*, pages 930–935.

[Garcia and Solanas, 2004] Garcia, M. and Solanas, A. (2004). 3D simultaneous localization and modeling from stereo vision. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'04)*, pages 847–853. IEEE Computer Society Press.

[George and Sukkarieh, 2007a] George, M. and Sukkarieh, S. (2007a). Camera aided inertial navigation in poor gps environments. *IEEE Aerospace Conference*, pages 1–12.

[George and Sukkarieh, 2007b] George, M. and Sukkarieh, S. (2007b). Inertial navigation aided by monocular camera observations of unknown features. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'07)*, pages 3558–3564.

[Gervautz and Purgathofer, 1988] Gervautz, M. and Purgathofer, W. (1988). A simple method for color quanitization: octree quantization. In *New Trends in Computer Graphics*. Springer Verlag, Berlin.

[Gibson, 1998] Gibson, S. (1998). Constrained Elastic SurfaceNets: Generating Smooth Models from Binary Segmented Data. In *Proc. of the International*

Conference on Medical Image Computing and Computer-Assisted Intervention, pages 888–898.

[Gomes and Faugeras, 1999] Gomes, J. and Faugeras, O. D. (1999). Reconciling distance functions and level sets. In *Proc. of the Second International Conference on Scale-Space Theories in Computer Vision (SCALE-SPACE '99)*, pages 70–81, London, UK. Springer-Verlag.

[Gordon et al., 1993] Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEEE Proceedings for Radar and Signal Processing*, volume 140, pages 107–113.

[Grisetti et al., 2005] Grisetti, G., Stachniss, C., and Burgard, W. (2005). Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'05)*, pages 2432–2437. IEEE Computer Society Press.

[Guzzoni et al., 1997] Guzzoni, D., Cheyer, A., Julia, L., and Konolige, K. (1997). Many robots make short work. *AI Magazine*, 18(1):55–64.

[Habib et al., 2002] Habib, A., Shin, S., and Morgan, M. (2002). New Approach for Calibrating Off-the-Shelf Digital Cameras. In *Proc. of the ISPRS Commision III Symposium on Photogrammatic Computer Vision*, page A: 144.

[Hagen et al., 2003] Hagen, P., Storkersen, N., Vestgard, K., and Kartvedt, P. (2003). The HUGIN 1000 autonomous underwater vehicle for military applications. In *Proc. of OCEANS '03*, volume 2, pages 1141–1145.

[Hähnel et al., 2003] Hähnel, D., Burgard, W., Fox, D., and Thrun, S. (2003). A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*, volume 1, pages 206–211.

[Hahnel et al., 2005] Hahnel, D., Burgard, W., Wegbreit, B., and Thrun, S. (2005). *Towards Lazy Data Association in SLAM*, volume 15 of *Springer Tracts in Advanced Robotics*, pages 421–431. Springer Berlin / Heidelberg, Sienna, Italy, robotics research edition.

[Hallset, 1991a] Hallset, J. (1991a). A prototype autonomous underwater vehicle for pipeline inspection. *Fifth International Conference on Advanced Robotics (ICAR'91)*, 2:1565–1568.

[Hallset, 1991b] Hallset, J. (1991b). Simple vision tracking of pipelines for an autonomous underwater vehicle. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'91)*, volume 3, pages 2767–2772.

[Hallset, 1992] Hallset, J. (1992). A vision system for an autonomous underwater vehicle. In *Proc. of the 11th IAPR International Conference on Pattern Recognition (ICPR'92)*, volume 1, pages 320–323.

[Harley and Woodward, 1987] Harley, J. and Woodward, D., editors (1987). *The History of Cartography*, volume 1. The University of Chicago Press, Chicago, IL.

[Hartley and Zisserman, 2000] Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[Harvey et al., 2001] Harvey, E. S., Fletcher, D., and Shortis, M. R. (2001). A comparison of the precision and accuracy of estimates of reef-fish lengths determined visually by divers with estimates produced by a stereo-video system. *Fishery Bulletin*, 99:63–71.

[Harvey and Shortis, 1998] Harvey, E. S. and Shortis, M. R. (1998). Calibration Stability of an Underwater Stereo Video System: Implications for Measurement Accuracy and Precision. *Marine Technology Society Journal*, 32(2):3–17.

[Heikkila and Silven, 1996] Heikkila, J. and Silven, O. (1996). Calibration Procedure for Short Focal Length Off-the-Shelf CCD-Cameras. In *Proc. of the 13th International Conference on Pattern Recognition (ICPR'96)*, pages 166–170.

[Hemayed, 2003] Hemayed, E. (2003). A survey of camera self-calibration. In *Proc. of the IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 351–357.

[Herath et al., 2006] Herath, D. C., Kodagoda, S., and Dissanayake, G. (2006). Simultaneous localisation and mapping: A stereo vision based approach. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'06)*, pages 922–927.

[Hodgson, 1999] Hodgson, G. (1999). A Global Assessment of Human Effects on Coral Reefs. *Marine Pollution Bulletin*, 38(5):345–355.

[Hogue et al., 2007a] Hogue, A., German, A., and Jenkin, M. (2007a). Underwater environment reconstruction using stereo and inertial data. In *Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC'07)*, pages 2372–2377.

[Hogue et al., 2006a] Hogue, A., German, A., Zacher, J., and Jenkin, M. (2006a). Underwater 3D Mapping: Experiences and Lessons Learned. In *Proc. of the Third Canadian Conference on Computer and Robot Vision (CRV'06)*.

[Hogue et al., 2007b] Hogue, A., German, A., Zacher, J., and Jenkin, M. (2007b). AQUASENSOR: a Stereo-Inertial Tool for Underwater 3D Reconstruction. Poster. Space Vision and Advanced Robotics (SVAR), May, Brampton, Ontario, Canada. [Poster+Demo].

[Hogue et al., 2006b] Hogue, A., Gill, S., and Jenkin, M. (2006b). AQUASENSOR: a 3D reconstruction tool for Security and Entertainment. 16th Annual International Conference hosted by IBM Centers for Advanced Studies (CASCON). October 16–19. Markham, Ontario, Canada. [Poster+Demo].

[Hogue et al., 2007c] Hogue, A., Gill, S., and Jenkin, M. (2007c). Automated avatar creation for 3d games. In *Proc. of the ACM Conference on Future Play*, pages 174–180, Toronto, Ontario, Canada.

[Hogue and Jenkin, 2003a] Hogue, A. and Jenkin, M. (2003a). Fusion of Trinocular Stereo and Inertial Data for Underwater Robotic Navigation: Preliminary Results. Poster. IS2003: 13th Annual Conference on Intelligent Systems, June 8-10, Halifax, Nova Scotia, Canada. [Poster].

[Hogue and Jenkin, 2003b] Hogue, A. and Jenkin, M. (2003b). Fusion of Trinocular Stereo and Inertial Data for Underwater Robotic Navigation: Preliminary Results. Poster. Crestech's Innovation Networking Conference, October 1, Toronto, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2004a] Hogue, A. and Jenkin, M. (2004a). The AQUA Trinocular Stereo Package. Poster. Space Vision and Advanced Robotics (SVAR), May 26, Brampton, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2004b] Hogue, A. and Jenkin, M. (2004b). The AQUA Trinocular Stereo Package. Poster. IS2004: 14th Annual Conference on Intelligent Systems, June 6-8, Ottawa, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2005a] Hogue, A. and Jenkin, M. (2005a). Visual and Inertial 6DOF Pose Estimation and 3D Mapping for AQUA. Poster. Space Vision and Advanced Robotics (SVAR), May 19, Brampton, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2005b] Hogue, A. and Jenkin, M. (2005b). Visual and Inertial 6DOF Pose Estimation and 3D Mapping for AQUA. Poster. IS2005: 15th Annual Conference on Intelligent Systems, June 5-7, Quebec City, Quebec, Canada. [Poster].

[Hogue and Jenkin, 2005c] Hogue, A. and Jenkin, M. (2005c). Visual and Inertial 6DOF Pose Estimation and 3D Mapping for AQUA. Poster. CVR 2005: Computational Vision in Neural and Machine Systems. June 15-18, York University, Toronto, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2006a] Hogue, A. and Jenkin, M. (2006a). AQUASENSOR: Underwater 3D Mapping for the AQUA Robot Experiences and Lessons Learned.

Poster. Space Vision and Advanced Robotics (SVAR), May 26, Brampton, Ontario, Canada. [Poster].

[Hogue and Jenkin, 2006b] Hogue, A. and Jenkin, M. (2006b). Development of an Underwater Vision Sensor for 3D Reef Mapping. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'06)*, pages 5351–5356.

[Hogue and Jenkin, 2007] Hogue, A. and Jenkin, M. (2007). AQUASENSOR: Underwater visual 3D Mapping for the AQUA Robot. Poster. CVR 2007: Cortical Mechanisms of Vision, June 19-23, York University, Toronto, Ontario, Canada. [Poster].

[Holmes, 2008] Holmes, G. (2008). Estimating three-dimensional surface areas on coral reefs. *Journal of Experimental Marine Biology and Ecology*, 365(1):67–73.

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

[Houston et al., 2006] Houston, B., Nielsen, M. B., Batty, C., Nilsson, O., and Museth, K. (2006). Hierarchical rle level set: A compact and versatile deformable surface representation. *ACM Transactions on Graphics*, 25(1):151–175.

[Houston et al., 2004] Houston, B., Wiebe, M., and Batty, C. (2004). Rle sparse level sets. In *Proc. of the ACM SIGGRAPH 2004 Conference on Sketches & Applications*, page 137.

[Jakuba and Yoerger, 2003] Jakuba, M. and Yoerger, D. (2003). High-resolution multibeam sonar mapping with the autonomous benthic explorer (abe). In *Proc. of the 13th International Symposium on Unmanned Untethered Submersible Technology (UUST'03)*, Durham, NH.

[Jenkin et al., 2007] Jenkin, M., Hogue, A., German, A., Gill, S., Topol, A., and Wilson, S. (2007). Underwater surface recovery and segmentation. In *Proc. of the 6th IEEE International Conference on Cognitive Informatics (ICCI'07)*, pages 373–380.

[Jenkin et al., 2008] Jenkin, M., Hogue, A., German, A., Gill, S., Topol, A., and Wilson, S. (2008). Modelling underwater structures. *International Journal of Cognitive Informatics*.

[Jones et al., 2006] Jones, M., Bærentzen, A., and Sramek, M. (2006). 3D distance fields: A survey of techniques and applications. *Transactions on Visualization and Computer Graphics*, 12(4):581–599.

[Julier and Uhlmann, 1997] Julier, S. and Uhlmann, J. (1997). A new extension of the Kalman filter to nonlinear systems. In *Proc. of the International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, pages 182–193.

[Kalman, 1960] Kalman, R. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME– Journal of Basic Engineering*, 82(D):35–45.

[Khamene and Negahdaripour, 1999] Khamene, A. and Negahdaripour, S. (1999). Building 3d elevation maps of sea-floor scenes from underwater stereo images. In *Proc. of MTS/IEEE OCEANS '99*, volume 1, pages 64–70.

[Kim and Sukkarieh, 2007] Kim, J. and Sukkarieh, S. (2007). Real-time implementation of airborne inertial-slam. *Robotics and Autonomous Systems*, 55(1):62–71.

[Kocak et al., 2008] Kocak, D. M., Dalgleish, F. R., Caimi, F. M., and Schechner, Y. Y. (2008). A focus on recent developments and trends in underwater imaging. *Marine Technology Society Journal*, 42(1):52–67.

[Kodagoda et al., 2006] Kodagoda, K., Alempijevic, A., Underwood, J., Kumar, S., and Dissanayake, G. (2006). Sensor registration and calibration using moving targets. In *Proc. of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV'06)*, pages 1–6.

[Kuipers and Byun, 1991] Kuipers, B. and Byun, Y. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8:47–63.

[Kunz and Singh, 2008] Kunz, C. and Singh, H. (2008). Hemispherical refraction and camera calibration in underwater vision. In *Proc. of MTS/IEEE OCEANS '08*, Quebec City, Quebec, Canada.

[Kwak et al., 2007] Kwak, N., Kim, I.-K., Lee, H.-C., and Lee, B.-H. (2007). Analysis of resampling process for the particle depletion problem in fastslam. In *Proc. of the 16th IEEE International Symposium on Robot and Human interactive Communication (RO-MAN '07)*, pages 200–205.

[Kwon, 1999a] Kwon, Y.-H. (1999a). A camera calibration algorithm for the underwater motion analysis. In Sanders, R. and Gibson, B., editors, *Proc. of the Scientific Proceedings of the XVII International Symposium on Biomechanics in Sports*, pages 257–260, Perth, Australia: Edith Cowan University.

[Kwon, 1999b] Kwon, Y.-H. (1999b). Object plane deformation due to refraction in 2-dimensional underwater motion analysis. *Journal of Applied Biomechanics*, 15:396–403.

[Kwon and Lindley, 2000] Kwon, Y.-H. and Lindley, S. (2000). Applicability of the localized-calibration methods in underwater motion analysis. In *Proc. of the XVIII International Symposium on Biomechanics in Sports*.

[Lavest et al., 2003] Lavest, J., Rives, G., and Lapreste, J. (2003). Dry camera calibration for underwater applications. *Machine Vision and Applications*, 13(5–6):245–253.

[Leonard and Durrant-Whyte, 1991] Leonard, J. and Durrant-Whyte, H. (1991). Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382.

[Liu et al., 2001] Liu, J., Chen, R., and Logvinenko, T. (2001). *A Theoretical Framework for Sequential Importance Sampling and Resampling*, pages 225–246. Sequential Monte Carlo Methods in Practice. Springer, New York, NY, USA.

[Lorenson, 1987] Lorenson, W. (1987). Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21(4):163–169.

[Lowe, 2004] Lowe, D. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[Lu and Milios, 1994] Lu, F. and Milios, E. (1994). Robot pose estimation in unknown environments by matching 2D range scans. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 935–938. IEEE Computer Society Press.

[Lu and Milios, 1997] Lu, F. and Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349.

[Lucas and Kanade, 1981] Lucas, B. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. of the International Joint Conferences on Artificial Intelligence (IJCAI'81)*, pages 674–679.

[Ludvigsen et al., 2006] Ludvigsen, M., Eustice, R., and Singh, H. (2006). Photogrammetric models for marine archaeology. In *Proc. of IEEE/MTS OCEANS '06*, pages 1–6, Boston, MA.

[Mahon and Williams, 2003] Mahon, I. and Williams, S. (2003). Three-dimensional robotic mapping. In *Proc. of the Australasian Conference on Robotics and Automation (ACRA'03)*, Presented at Australiasian Conference on Robotics and Automation, Brisbane, QLD.

[Majumder, 2001] Majumder, S. (2001). *Sensor Fusion and Feature-Based Navigation for Subsea Robots*. PhD thesis, Australian Centre for Field Robotics, School of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, Sydney, Australia.

[Makarenko et al., 1997] Makarenko, A., Poddubnyi, V., and Kholopova, V. (1997). Refined model for studies of the nonlinear dynamics of self-propelled submersibles with allowance for the effect of the cable. *International Applied Mechanics*, 33(9):747–753.

[Marzorati et al., 2007] Marzorati, D., Matteucci, M., and Sorrenti, D. (2007). Particle-based Sensor Modeling for 3D-Vision SLAM. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'07)*, pages 4801–4806.

[Matthies and Shafer, 1987] Matthies, L. and Shafer, S. (1987). Error modeling in stereo navigation. *IEEE Journal of Robotics and Automation*, 3(3):239–248.

[Maybeck, 1979] Maybeck, P. (1979). *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, Inc.

[Michaels and Boult, 1999] Michaels, R. and Boult, T. (1999). Increasing robustness in self-localiation and pose estimation. In *Proceedings of the SPIE Conference on Mobile Robots, SPIE.*

[Montemerlo, 2003] Montemerlo, M. (2003). *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association.* CMU-RI-TR-03-28, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

[Montemerlo et al., 2002] Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *Proc. of the AAAI National Conference on Artificial Intelligence (AAAI'02)*, pages 593–598, Edmonton, Canada.

[Montiel and Davison, 2006] Montiel, J. and Davison, A. (2006). A visual compass based on slam. In *Proc. of the IEEE Conference on Robotics and Automation (ICRA'06)*, pages 1917–1922.

[Moravec and Elfes, 1985] Moravec, H. and Elfes, A. (1985). High-Resolution Maps from Wide-Angle Sonar. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'85)*, volume 2, pages 116–121, Los Alamitos, Calif. CS Press.

[Murphy, 1999] Murphy, K. (1999). Bayesian Map Learning in Dynamic Environments. In *Proc. of the 13th Conference on Neural Information Processing Systems (NIPS'99)*, pages 1015–1021.

[Murphy and Russell, 2001] Murphy, K. and Russell, S. (2001). Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In Doucet, A., de Frietas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*. Springer Verlag, New York.

[Murray and Little, 2000] Murray, D. and Little, J. (2000). Using Real-Time Stereo Vision for Mobile Robot Navigation. *Autonomous Robots*, 8(2):161–171.

[Negahdaripour and Firoozfam, 2006] Negahdaripour, S. and Firoozfam, P. (2006). An ROV Stereovision System for Ship-Hull Inspection. *IEEE Journal of Oceanic Engineering*, 31(3):551–564.

[Negahdaripour et al., 1995] Negahdaripour, S., Hayashi, B., and Aloimonos, Y. (1995). Direct motion stereo for passive navigation. *IEEE Transactions on Robotics and Automation*, 11(6):829–843.

[Negahdaripour et al., 1996] Negahdaripour, S., Jin, L., Xu, X., Tsukamoto, C., and Yuh, J. (1996). A real-time vision-based 3d motion estimation system for positioning and trajectory following. In *Proc. of the 3rd IEEE Workshop on Applications of Computer Vision (WACV '96)*, pages 264–269.

[Negahdaripour and Lanjing, 1995] Negahdaripour, S. and Lanjing, J. (1995). Direct recovery of motion and range from images of scenes with time-varying illumination. In *Proc. of the International Symposium on Computer Vision*, pages 467–472.

[Negahdaripour and Madjidi, 2003] Negahdaripour, S. and Madjidi, H. (2003). Stereovision maging on submersible platforms for 3-D mapping of benthic habitats and sea-floor structures. *IEEE Journal of Oceanic Engineering*, 28(4):625–650.

[Negahdaripour et al., 2002] Negahdaripour, S., Madjidi, H., and Khamene, A. (2002). Stereovision imageing to map seafloor and benthic habitats - A promising technology to go from 2d to 3d subsea mapping, already in operation in terrestrial and planetary applications. *Sea Technology*, 43(8):35–+.

[Negahdaripour et al., 1990] Negahdaripour, S., Yu, C., and Shokrollahi, A. (1990). Recovering shape and motion from undersea images. *IEEE Journal of Oceanic Engineering*, 15(3):189–198.

[Negahdaripour et al., 1998] Negahdaripour, S., Zhang, S., Xun, X., and Khamene, A. (1998). On shape and motion recovery from underwater imagery for 3D mapping and motion-based video compression. In *Proc. of OCEANS '98*, volume 1, pages 277–281.

[Nettleton et al., 2000] Nettleton, E., Gibbens, P., and Durrant-Whyte, H. (2000). Closed form solutions to the multiple platform simultaneous localisation and map building (SLAM) problem. In Dasarathy, B. V., editor, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, pages 428–437.

[Newman, 1999] Newman, P. (1999). *On the Structure and Solution of the Simultaneous Localisation and Map Building Problem*. PhD thesis, Australian Centre for Field Robotics, School of Aerospace, Mechanical and Mechatronic Engineering, University of Sydney, Sydney, Australia.

[Newman et al., 2006] Newman, P., Cole, D., and Ho, K. (2006). Outdoor SLAM using Visual Appearance and Laser Ranging. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'06)*, pages 1180–1187.

[Nicosevici et al., 2007] Nicosevici, T., Garcia, R., and Negahdaripour, S. (2007). Identification of suitable interest points using geometric and radiometric cues in motion video for efficient 3-D environmental modeling. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA'07)*, pages 4969–4974, Roma, Italy.

[Nicosevici et al., 2005] Nicosevici, T., Negahdaripour, S., and Garcia, R. (2005). Monocular-based 3-D seafloor reconstruction and ortho-mosaicing by piecewise planar representation. In *Proc. of IEEE OCEANS '05*, volume 2, pages 1279–1286, Washington, DC, USA.

[Nieto et al., 2006a] Nieto, J., Bailey, T., and Nebot, E. (2006a). *Scan-SLAM: Combining EKF-SLAM and Scan Correlation*, volume 25 of *Springer Tracts in Advanced Robotics*, pages 167–178. Springer Berlin / Heidelberg, Port Douglas, Australia.

[Nieto et al., 2007] Nieto, J., Bailey, T., and Nebot, E. (2007). Recursive scan-matching slam. *Robotics and Autonomous Systems*, 55(1):39–49.

[Nieto et al., 2006b] Nieto, J., Guivant, J., and Nebot, E. (2006b). DenseSLAM: Simultaneous Localization and Dense Mapping. *International Journal of Robotics Research*, 25(8):711–744.

[Nister, 2001] Nister, D. (2001). *Automatic Dense Reconstruction from uncalibrated video sequences*. PhD thesis, Royal Institute KTH, Stockholm, Sweden.

[Ortiz et al., 2002] Ortiz, A., Simó, M., and Oliver, G. (2002). A vision system for an underwater cable tracker. *Machine Vision and Applications (MVA'02)*, 13(3):129–140.

[Pearl, 2000] Pearl, J. (2000). *Causality*. Cambridge University Press.

[Pedraza et al., 2007] Pedraza, L., Dissanayake, G., Miro, J. V., Rodriguez-Losada, D., and Matia, F. (2007). BS-SLAM: Shaping the World. In *Proc. of Robotics: Science and Systems (RSS'07)*, Atlanta, GA, USA.

[Perry and Frisken, 2001] Perry, R. N. and Frisken, S. F. (2001). Kizamu: a system for sculpting digital characters. In *Proc. of the ACM SIGGRAPH 2001 conference on Computer graphics and interactive techniques*, pages 47–56, New York, NY, USA. ACM.

[Personnaz and Horaud, 2002] Personnaz, M. and Horaud, R. (2002). Camera calibration: estimation, validation and software. Technical Report RT-0258, INRIA Rhone Alpes, Grenoble.

[Pinies et al., 2007] Pinies, P., Lupton, T., Sukkarieh, S., and Tardos, J. (2007). Inertial aiding of inverse depth slam using a monocular camera. In *Proc. of the*

IEEE International Conference on Robotics and Automation (ICRA'07), pages 2797–2802.

[Pizarro et al., 2004] Pizarro, O., Eustice, R., and Singh, H. (2004). Large Area 3D Reconstructions from Underwater Surveys. In *Proc. of MTS/IEEE OCEANS '04*, pages 678–687. Kobe, Japan.

[Plets et al., 2008] Plets, R., Dix, J., Adams, J., and Best, A. (2008). 3d reconstruction of a shallow archaeological site from high-resolution acoustic imagery: The grace dieu. *Applied Acoustics*, 69(5):399–411. The detection of buried marine targets.

[Pollefeys, 1999] Pollefeys, M. (1999). *Self-calibration and metric 3D recontruction from uncalibrated image sequences*. PhD thesis, ESAT-PSI, K.U. Leuven, Leuven, Belgium. Scientific Prize BARCO 1999.

[Press et al., 2002] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (2002). *Numerical Recipes in C*. Cambridge University Press.

[Reynolds, 1998] Reynolds, J. (1998). Autonomous underwater vehicle: Vision system. Undergraduate Thesis, The Australian National University, Canberra, Australia.

[Ribas et al., 2007a] Ribas, D., Ridao, P., Domingo Tardos, J., and Neira, J. (2007a). Underwater slam in a marina environment. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, pages 1455–1460.

[Ribas et al., 2007b] Ribas, D., Ridao, P., Neira, J., and Tardos, J. (2007b). A method for extracting lines and their uncertainty from acoustic underwater images for SLAM. In *Proc. of the 6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV'07)*, Toulouse, France.

[Ribas et al., 2006] Ribas, D., Ridao, P., Neira, J., and Tardos, J. D. (2006). SLAM using an Imaging Sonar for Partially Structured Underwater Environments. In

Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'06), pages 5040–5045.

[Ribas et al., 2008] Ribas, D., Ridao, P., Tardós, J., and Neira, J. (2008). Underwater SLAM in man made structured environments. *Journal of Field Robotics*, Accepted for publication.

[Ristic et al., 2004] Ristic, B., Arulampalam, S., and Gordon, N. (2004). *Beyond the Kalman Filter - Particle Filters for Tracking Applications*. Artech House, Boston, MA.

[Rodseth and Hallset, 1991] Rodseth, O. and Hallset, J. (1991). Rov90 - a pragmatic approach to autonomous underwater vehicle design. In *Proc. of OCEANS '91*, volume 2, pages 1075–1081.

[Salvi et al., 2002] Salvi, J., Armangue, X., and Batlle, J. (2002). A comparative review of camera calibrating methods with accuracy evaluation. *Pattern Recognition*, 35(7):1617–1635.

[Schechner and Karpel, 2004] Schechner, Y. and Karpel, N. (2004). Clear underwater vision. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04)*, volume 1, pages 536–543.

[Schiele and Crowley, 1994] Schiele, B. and Crowley, J. (1994). A Comparison of Position Estimation Techniques using Occupancy Grids. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'94)*, pages 1628–1634. IEEE Computer Society Press.

[Se et al., 2002] Se, S., Lowe, D., and Little, J. (2002). Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks. *International Journal of Robotics Research*, 21(8):735–758.

[Shi and Tomasi, 1994] Shi, J. and Tomasi, C. (1994). Good Features to Track. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593–600.

[Shortis et al., 2007] Shortis, M. R., Harvey, E. S., and Seager, J. (2007). A review of the status and trends in underwater videometric measurement. In *In Proceedings of the SPIE Conference 6491, Videometrics IX, IS&T/SPIE Electronic Imaging*.

[Sigg et al., 2003] Sigg, C., Peikert, R., and Gross, M. (2003). Signed distance transform using graphics hardware. *IEEE Visualization*, pages 83–90.

[Sim et al., 2007] Sim, R., Elinas, P., and Little, J. J. (2007). A study of the Rao-Blackwellised Particle Filter for Efficient and Accurate Vision-based SLAM. *International Journal of Computer Vision*, 74(3):303–318.

[Simhon and Dudek, 1998] Simhon, S. and Dudek, G. (1998). A Global Topological Map formed by Local Metric Maps. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robotic Systems (IROS'98)*, pages 1708–1714.

[Singh et al., 2004a] Singh, H., Armstrong, R., Gilbes, F., Eustice, R., Roman, C., Pizarro, O., and Torres, J. (2004a). Imaging Coral I: Imaging Coral Habitats with the SeaBED AUV. *Subsurface Sensing Technologies and Applications*, 5(1):25–42.

[Singh et al., 2004b] Singh, H., Can, A., Eustice, R., Lerner, S., McPhee, N., Pizarro, O., and Roman, C. (2004b). Seabed AUV offers new platform for high-resolution imaging. *EOS, Transactions of the American Geophysical Union*, 85(31):289,294–295.

[Singh et al., 2002] Singh, H., Eustice, R., Roman, C., and Pizarro, O. (2002). The Seabed AUV - a platform for high resolution imaging. In *Proc. of the Unmanned Underwater Vehicle Showcase (UUVS'02)*, Southampton Oceanography Centre, UK.

[Singh et al., 2007] Singh, H., Roman, C., Pizarro, O., and Eustice, R. (2007). *Advances in high-resolution imaging from underwater vehicles*, volume 28 of *Springer Tracts in Advanced Robotics*, pages 430–448. Springer Berlin / Heidelberg.

[Sinha et al., 2006] Sinha, S., Frahm, J.-M., Pollefeys, M., and Genc, Y. (2006). Feature tracking and matching in video using programmable graphics hardware. *Machine Vision and Applications*.

[Skvortzov et al., 2007] Skvortzov, V., Lee, H.-K., Bang, S., and Lee, Y. (2007). Application of electronic compass for mobile robot in an indoor environment. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'07)*, pages 2963–2970.

[Smith and Cheeseman, 1986] Smith, R. and Cheeseman, P. (1986). On the Representation and Estimation of Spatial Uncertainty. *International Journal of Robotics Research*, 5(4):56–68.

[Smith et al., 1990] Smith, R., Self, M., and Cheeseman, P. (1990). Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, pages 167–193.

[Smith et al., 1991] Smith, R., Self, M., and Cheeseman, P. (1991). A Stochastic Map for Uncertain Spatial Relationships. In Iyengar, S. and Elfes, A., editors, *Autonomous Mobile Robots: Perception, Mapping, and Navigation*, pages 323–330. IEEE Computer Society Press.

[Stark and Woods, 2002] Stark, H. and Woods, J. (2002). *Probability and Random Processes with Applications to Signal Processing*. Prentice-Hall.

[Strickrott and Negahdaripour, 1997] Strickrott, J. and Negahdaripour, S. (1997). On the development of an active vision system for 3-D scene reconstruction and analysis from underwater images. In *Proc. of MTS/IEEE OCEANS '97*, volume 1, pages 626–633.

[Svoboda et al., 2005] Svoboda, T., Martinec, D., and Pajdla, T. (2005). A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422.

[Takezawa et al., 2004] Takezawa, S., Herath, D., and Dissanayake, G. (2004). SLAM in indoor environments with stereo vision. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'04)*, volume 2, pages 1866–1871.

[Tanizaki, 2003] Tanizaki, H. (2003). Nonlinear and non-gaussian state-space modeling with monte carlo techniques: A survey and comparative study. In *In Rao, C., & Shanbhag, D. (Eds.), Handbook of Statistics*, volume 21, pages 871–929. North-Holland.

[Thrun, 1998] Thrun, S. (1998). Learning Metric-Topological Maps for Indoor Mobile Robot Navigation. *Artificial Intelligence*, 99:1:21–71.

[Thrun, 2000] Thrun, S. (2000). Probabilistic Algorithms in Robotics. *AI Magazine*, 21(4):93–109.

[Thrun, 2003] Thrun, S. (2003). Robotic Mapping: A Survey. In Lakemeyer, G. and Nebel, B., editors, *Exploring Artificial Intelligence in the New Millenium*, pages 1–36. Morgan Kaufmann.

[Thrun et al., 2000a] Thrun, S., Beetz, M., Bennewitz, M., Burgard, W., Cremers, A., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., and Schulz, D. (2000a). Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva. *International Journal of Robotics Research*, 19(11):972–999.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press, Cambridge, Massachusetts.

[Thrun et al., 1998] Thrun, S., Fox, D., and Burgard, W. (1998). A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Machine Learning*, 31:29–53. also appeared in Autonomous Robots 5, 253-271 (joint issue).

[Thrun et al., 2000b] Thrun, S., Fox, D., Burgard, W., and Dellaert, F. (2000b). Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 128(1-2):99–141.

[Thrun et al., 2004a] Thrun, S., Liu, Y., Koller, D., Ng, A., and Ghahramani, Z. (2004a). Simultaneous Localization and Mapping with Sparse Extended Information Filters. *International Journal of Robotics Research*, 23:693–716.

[Thrun et al., 2004b] Thrun, S., Thayer, S., Whittaker, W., Baker, C., Burgard, W., Ferguson, D., Hahnel, D., Montemerlo, M., Morris, A., Omohundro, Z., and Reverte, C. (2004b). Autonomous Exploration and Mapping of Abandoned Mines. *IEEE Robotics and Automation Magazine*, 11(4):79–91.

[Treibitz et al., 2008] Treibitz, T., Schechner, Y., and Singh, H. (2008). Flat refractive geometry. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8.

[Triggs et al., 2000] Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (2000). *Bundle Adjustment - A Modern Synthesis*, pages 278–375. Lecture Notes in Computer Science. Springer-Verlag.

[Tsai, 1987] Tsai, R. Y. (1987). A Versatile Camera Calibration Technique for Highaccuracy 3D Machine Vision Metrology Using off-the-shelf TV Cameras and Lenses. *IEEE Journal of Robotics and Automation*, 3:323–344.

[Turner, 1995] Turner, R. (1995). Intelligent control of autonomous underwater vehicles: the orca project. In *Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC'95)*, volume 2, pages 1717–1722.

[Wang, 2004] Wang, C. (2004). *Simultaneous Localization, Mapping and Moving Object Tracking*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

[Wang et al., 2007a] Wang, C.-C., Thorpe, C., Thrun, S., Hebert, M., and Durrant-Whyte, H. (2007a). Simultaneous localization, mapping and moving object tracking. *International Journal of Robotics Research*, 26(9):889–916.

[Wang et al., 2005] Wang, J., Oliveira, M., Xie, H., and Kaufman, A. (2005). Surface reconstruction using oriented charges. *Computer Graphics International*, pages 122–128.

[Wang et al., 2007b] Wang, Z., Huang, S., and Dissanayake, G. (2007b). D-slam: A decoupled solution to simultaneous localization and mapping. *International Journal of Robotics Research*, 26(2):187–204.

[Wei and Ma, 1994] Wei, G. and Ma, S. (1994). Implicit and Explicit Camera Calibration: Theory and Experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI'94)*, 16(5):469–480.

[Weisstein, 2006] Weisstein, E. (2006). Rodrigues' Rotation Formula. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/RodriguesRotationFormula.html.

[Welch and Bishop, 1995] Welch, G. and Bishop, G. (1995). An Introduction to the Kalman Filter. Technical Report TR95-041, University of North Carolina at Chapel Hill (UNC).

[Wettergreen et al., 1998] Wettergreen, D., Gaskett, C., and Zelinsky, A. (1998). Development of a visually-guided autonomous underwater vehicle. In *Proc. of OCEANS '98*, volume 2, pages 1200–1204.

[Williams and Mahon, 2004a] Williams, S. and Mahon, I. (2004a). Design of an unmanned underwater vehicle for reef surveying. In *Proc. of the 3rd IFAC Symposium on Mechatronic Systems*, Presented at 3rd IFAC Symposium on Mechatronic Systems, Manly NSW, Australia.

[Williams and Mahon, 2004b] Williams, S. and Mahon, I. (2004b). Simultaneous Localisation and Mapping on the Great Barrier Reef. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'04)*, volume 2, pages 1771–1776.

[Xu and Negahdaripour, 1997] Xu, X. and Negahdaripour, S. (1997). Vision-based motion sensing for underwater navigation and mosaicing of ocean floor images. In *Proc. of MTS/IEEE OCEANS '97*, volume 2, pages 1412–1417.

[Yang and Pollefeys, 2003] Yang, R. and Pollefeys, M. (2003). Multi-resolution real-time stereo on commodity graphics hardware. In *Proc. of the IEEE*

Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03), volume 1, pages 211–217.

[Yang et al., 2004] Yang, R., Pollefeys, M., and Li, S. (2004). Improved real-time stereo on commodity graphics hardware. In *Proc. of the Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '04)*, pages 36–36.

[Yoerger et al., 2005] Yoerger, D., Jakuba, M., Bradley, A., and Bingham, B. (2005). Techniques for deep sea near bottom survey using an autonomous vehicle. In *Proc. of the 12th International Symposium of Robotics Research (ISRR'05)*, San Francisco, CA.

[Yoerger et al., 2007] Yoerger, D. R., Jakuba, M., Bradley, A. M., and Bingham, B. (2007). Techniques for Deep Sea Near Bottom Survey Using an Autonomous Underwater Vehicle. *International Journal of Robotics Research*, 26(1):41–54.

[Yu and Negahdaripour, 1993] Yu, C.-H. and Negahdaripour, S. (1993). Underwater experiments for orientation and motion recovery from video images. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'93)*, pages 93–98.

[Zhang and Negahdaripour, 2008] Zhang, H. and Negahdaripour, S. (2008). Epiflow - a paradigm for tracking stereo correspondences. *Computer Vision and Image Understanding*, 111(3):307–328.

[Zhang, 2000] Zhang, Z. (2000). A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(11):1330–1334.