UNDER-ICE RELATIVE POSE ESTIMATION AND ICE ANOMALY MAPPING WITH AN UNMANNED UNDERWATER VEHICLE

A Dissertation
Presented to
The Academic Faculty

By

Anthony Spears

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2015

UNDER-ICE RELATIVE POSE ESTIMATION AND ICE ANOMALY MAPPING WITH AN UNMANNED UNDERWATER VEHICLE

Approved by:

Dr. Ayanna M. Howard, Advisor
Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Britney Schmidt
Assistant Professor, School of Earth and Atmospheric Sciences
Georgia Institute of Technology

Dr. Linda Wills
Associate Professor, School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Michael West
Senior Research Engineer, Georgia Tech Research Institute
Georgia Institute of Technology

Dr. Thomas Collins
Principal Research Engineer, Georgia Tech Research Institute
Georgia Institute of Technology

Date Approved: October 2015

For my wife, my family, and my friends, whose support made this dissertation possible.

ACKNOWLEDGMENT

This thesis would not have been possible without the guidance and support of my mentors, colleagues, family, and friends.

First, I would like to thank my advisor, Dr. Ayanna Howard, for her patience and insight while guiding me through my research and the Ph.D. process. I have learned many valuable lessons from her guidance in the fields of academic research and field robotics.

I would also like to thank the rest of my committee: Dr. Mick West, Dr. Britney Schmidt, Dr. Tom Collins and Dr. Linda Wills for their incredibly valuable time and input.

I would like to thank Dr. Mick West and GTRI for providing me with a graduate assistantship through most of my PhD program. I would also like to thank Dr. Schmidt for providing the opportunity to work on the Icefin vehicle, and to be part of the 2014 field team in Antarctica.

I would like to thank my HumAnS lab group for their support through this project, with special thanks to Kevin DeMarco, who has provided many invaluable tools for undertaking this field robotics research.

Of course, without the support of my family, this thesis would not have been possible, and I thank them for everything they have done. I cannot begin to repay the many sacrifices made by my wife throughout this journey, and I would not have made it this far without her.

There are many others who have also provided support over the years, and I could write pages on my appreciation of their various contributions.

Without the support of my mentors, colleagues, family, and friends, this thesis would not have been possible. Thank you to everyone who has helped throughout this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Under-ice regions in both the Arctic and Antarctica are of great interest to many domains of science including biology, climate science, and planetary science. Great environmental and technical challenges face researchers when attempting to gather data from the polar under-ice regions of Earth for these applications. The harsh environments encountered both above and below the polar ice limit the use of human divers and manned submersibles in such data collection efforts. However, recent technological advances have provided the means for data collection using unmanned underwater vehicles (UUVs) beneath the ice.

New challenges are encountered using this unmanned technology including deployment, recovery, risk mitigation, navigation, and mapping. Presented in this dissertation are methods developed to help aid in navigation and mapping tasks in these under-ice environments. Specifically, development of computer vision methods using acoustic and optical imaging sensors (especially through sensor fusion) is used to help aid in under-ice UUV motion estimation. In addition to algorithms which can aid in the navigation problem, additional computer vision methods for ice texture and ice anomaly mapping are also developed and presented herein. The methods presented here utilize low-cost sensors already onboard many UUV platforms, and do not require expensive external infrastructure or setup effort.

First, a relative pose estimation method, used to help track a vehicle's position over a trajectory, is presented using a novel combination of optical flow-based computer vision methods with forward-looking sonar data, assuming a rigid motion model between frames. The use of computer vision techniques with sonar data is uncommon due to the high noise levels from this sensor, as well as the changing appearance of objects between frames. In addition to the use of a sonar sensor to aid in vehicle motion estimation, the use of camera sensors, also commonly onboard UUVs, is presented to provide additional independent relative vehicle pose estimates. While monocular camera relative pose methods are commonly applied in feature-rich environments such as terrestrial urban and underwater coral

reef locations, such methods are not common in featureless environments such as that found under the ice. In order to overcome these challenges, a novel adaptation of this approach is presented here using contrast enhancement and low feature detection thresholds, along with robust feature matching to eliminate outliers during motion model estimation.

While relative pose estimation using either a sonar or camera sensor provides valuable information for autonomous navigation in a UUV, fusion of these two complementary sensors can result in a much stronger and more robust trajectory estimate. A novel sonar and camera sensor fusion approach is presented here using a factor graph framework to combine estimates from these noisy but partially redundant sensors. In this case, the camera estimates provide additional degrees of motion not possible using a sonar sensor, but the translational camera estimates contain a scale factor ambiguity inherent to the camera sensor. While the sonar estimates are limited to only three degrees of motion, no scale factor ambiguity is encountered, and absolute translational motion can be estimated. Fusion of these two sensors can be used to leverage the strengths of each sensor to overcome the individual weaknesses and provide much more robust overall vehicle motion estimates.

Finally, it is not only useful to provide an estimate of the location of a UUV during data collection, but also to automatically estimate ice texture and to flag frames with possible anomalies of interest present against the ice background. This can help aid human vehicle operators and scientists in data collection and post analysis of these large, mostly featureless, datasets. A method for estimating ice texture using point features is presented here, along with point feature- and hue-based methods for anomaly detection and mapping.

Methods such as those developed in this dissertation can help aid vehicle operators and scientists in the difficult tasks of navigation and mapping for under-ice exploration, and can eventually provide an autonomous means for such data collection. The algorithms developed here can further the mission capabilities of current under-ice vehicle platforms to enable further exploration of these remote areas of the planet.

CHAPTER 1

INTRODUCTION

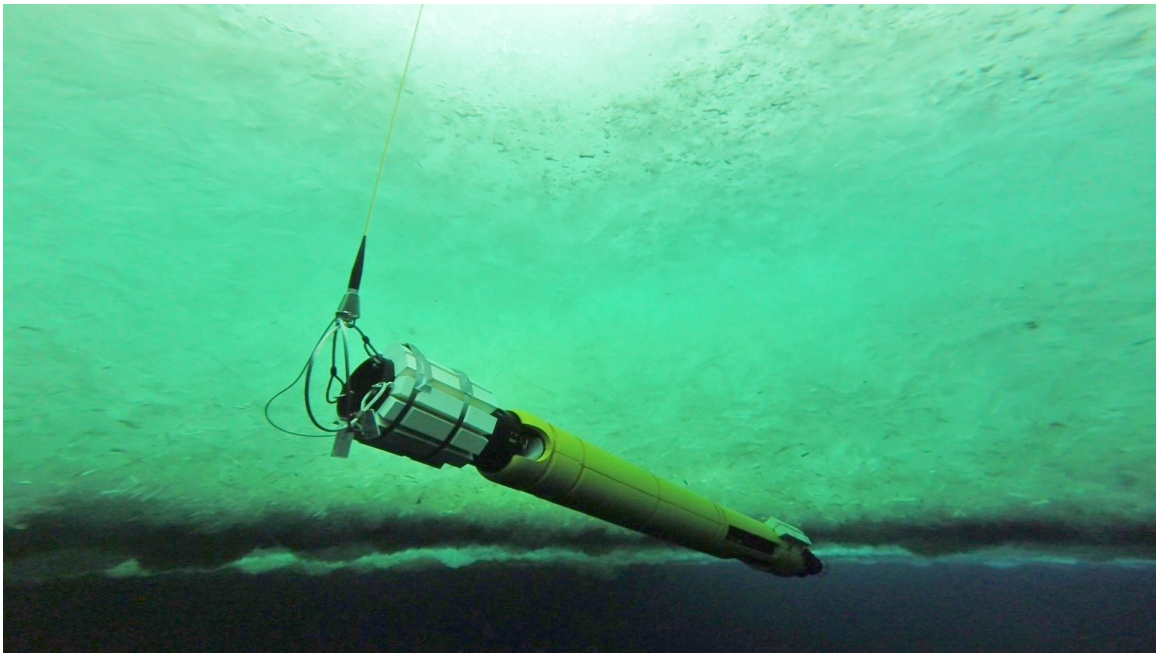


Figure 1: Custom Icefin vehicle under the sea ice near McMurdo, Antarctica

1.1 Under-ice Exploration and Navigation

Under-ice environments (Figure 1) are some of the least explored areas of planet earth due to the harsh elements present and difficulty accessing the water below the ice sheet, which can reach hundreds of meters in thickness. While earlier under-ice exploration in the Arctic and Antarctic was facilitated through the use of submarines and human divers, the past 15 years have seen a drastic increase in the use of unmanned underwater vehicles (UUVs) to perform these dangerous tasks. Localization and navigation through underwater environments is a difficult task for divers, submarines and UUVs alike due to the lack of salient (unique) visual features, poor visibility, limited communications, and absence of GPS signals used as a standard in most terrestrial and airborne applications. Under-ice environments tend to present even fewer salient visual features, less natural light, and more

limited access to the surface. An example of the under-ice environments considered here can be seen in Figure 2 with both permanent ice shelf (left) and thinner sea ice (right) environments.

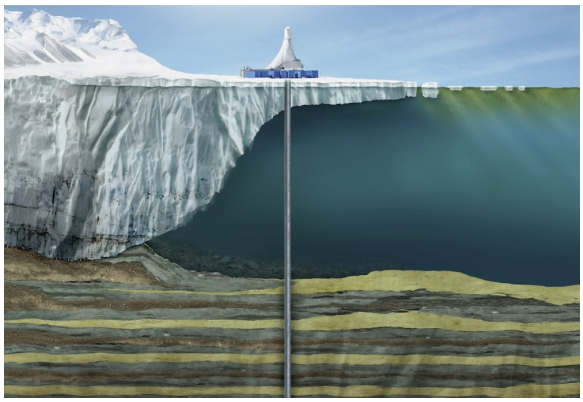


Figure 2: ANDRILL on Ross Ice Shelf (c.o. ANDRILL and NASA), Antarctica

Underwater navigation is largely reduced to an x, y problem due to the availability of highly accurate pressure (to constrain depth, or z-location) and compass (to constrain heading) sensors. The baseline localization method used in almost all open-water and under-ice UUVs is an inertial navigation system (INS) provided with information from an IMU (inertial measurement unit) and gyroscope that can be integrated to obtain position information. While the sole reliance on an INS is prohibitive to most missions due to the unbounded drift rates encountered and the resultant limitations on mission duration, this remains the most common solution for under-ice UUVs due to the unique environmental constraints. Some open water autonomous underwater vehicles (AUVs) have developed a solution to occasionally surface and obtain a GPS baseline, but this is not possible in under-ice environments due to the ice cover. Another solution for underwater localization takes the form of acoustic beacon networks for vehicle triangulation, commonly used with systems deployed in the open ocean. However, deployment of such a system is infeasible in many ice-covered environments due to the infrastructure required to deploy the beacons through the ice. Under-ice environments located in the polar regions of Earth also introduce the additional challenge of limited compass functionality due to the extreme latitudes so close

to the magnetic poles. All of these challenges present a very inhospitable environment for autonomous systems, which can easily get lost as their estimated ego-location accumulates error. Under-ice AUVs cannot be easily recovered after an accumulation of error in the position estimate (as those in open ocean can by simply surfacing), and much greater care must be taken to solve the localization and navigation problem in under-ice environments. A system capable of providing better estimates of localization over an extended trajectory in sensor-deprived, under-ice environments would allow for longer mission durations, wider vehicle trajectories and greater autonomous capabilities for these AUVs.

1.2 Use of Camera and Sonar Sensors Under the Ice

One increasingly common localization and motion estimation approach for autonomous vehicles is termed vision-based relative pose estimation. By matching corresponding feature points detected in consecutive images, the motion of the onboard imaging camera, correlating to movement of the vehicle, can be estimated. Vision-based relative pose estimation has been previously evaluated for use in terrestrial and underwater environments. However, such a method typically requires feature-rich and salient image streams, such as those obtained on coral reefs, but not present under the ice. A camera-based pose estimation method, adapted from use with aerial and terrestrial vehicles, is presented here for use in relatively featureless under-ice environments such as that in Figure 1. Preprocessing of the low-contrast input images and tuning of this pose estimation approach results in a self-contained method for estimating vehicle motion between image frames with six degrees-of-freedom (minus translational scale). This camera-based algorithm provides a method for vehicle pose estimation that is less vulnerable to integration drift error, such as that encountered using an INS, and does not require external infrastructure in contrast to acoustic-beacon-based methods.

Forward-looking multibeam acoustic sonar sensors are commonly found on underwater

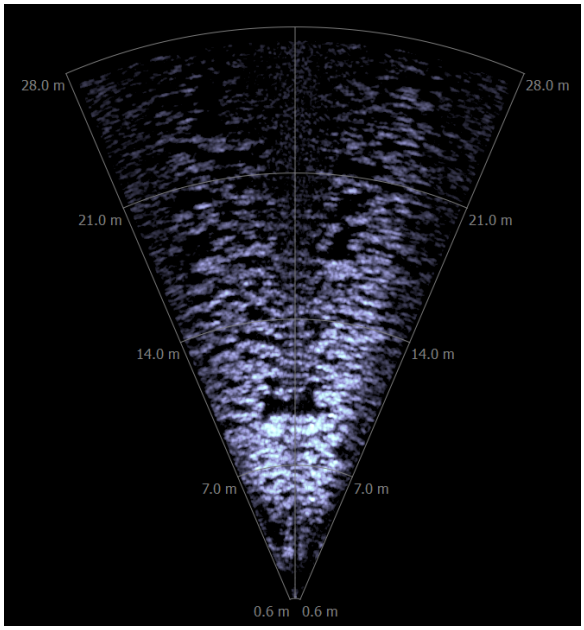


Figure 3: Sonar image of Antarctic ice

vehicles for use in object or obstacle detection. These sensors can provide range and bearing to an object in the path of the vehicle using acoustic waves. Forward-looking acoustic sonar sensors provide the viewer with a two-dimensional representation of the area in front of the sensor, highlighting objects with strong acoustic return, as seen in Figure 3. Sonar sensors do not encounter scale factor ambiguity, as monocular camera systems do, but are limited in the resolution of altitude angle. These sensors have very high noise levels and many current sonar processing applications are limited to the detection of strong intensity blob objects in the data. However, advances in imaging sonar technology over recent years have introduced the possibility of using common image processing algorithms, such as point-feature tracking and optical flow, with these high-resolution and high-frequency sonar images. A novel sonar-based pose estimation algorithm is presented here for use in estimating vehicle motion in relatively featureless under-ice environments. The algorithm uses an optical-flow-based approach to match points between frames along with a robust rigid motion model to estimate movement of the vehicle between frames. This motion model computes the optimal rotation and translation between point sets that minimizes

weighted pixel reprojection error using singular value decomposition and robust estimation. Such an algorithm provides a self-contained, frame-to-frame, relative pose estimation method that utilizes a sensor already present on many underwater vehicles. This algorithm is not subject to integration drift rates such as those encountered with an INS and does not require the external infrastructure of acoustic-beacon-based methods.

One proposed solution to the under-ice localization problem is visual SLAM (simultaneous localization and mapping). SLAM [7] refers to a localization method commonly used in terrestrial robotics where observations of landmarks in the environment are used to concurrently create a map of the environment and localize the vehicle inside that map as the vehicle moves. However, under-ice environments are largely low-contrast and featureless, and comprise mostly of repetitive ice texture. Therefore, only a very limited set of current under-ice vehicles utilize a camera-based localization method. SLAM using a forward-looking sonar sensor has been used in open ocean environments, but typically requires large anomaly features for blob tracking throughout the vehicle trajectory. Due to the limited anomaly features present in the sub-ice topography, the use of such sonar sensors for under-ice localization is also very uncommon.

1.3 Multimodal Sensor Fusion

Both cameras and sonar sensors have limitations in underwater and under-ice environments. Poor visibility, low ambient lighting, and scale ambiguity limit the camera sensor, while high noise levels and altitude angle ambiguity limit the sonar sensor. However, fusion of these datasets can utilize the strengths of both to offset the limitations of the individual sensors. Here, the novel use of both video- and sonar-based relative pose estimation methods is presented to provide additional constraints on the estimated vehicle trajectory and to help bound the drift rates of the INS. Sonar data can be used to obtain a three degree-of-freedom estimate (yaw, surge and sway), while six degrees-of-freedom (modulo translational scale) can be estimated from the camera data. Independent INS, camera-based, and sonar-based

systems each present unique weaknesses and strengths, and each sensor can encounter periods of inaccurate estimation. The use of a factor graph framework [8] [9] is presented here to fuse this information together and obtain a robust location estimate over the vehicle trajectory. This method combines the strengths of each sensor to overcome the independent weaknesses. A factor graph framework is a model well suited to such complex estimation problems where nodes represent unknown random variables to be estimated and the factors between them represent probabilistic information on those nodes (Figure 4). Optimization can then be performed over all available estimates to obtain the optimal vehicle state estimate over the entire trajectory. Here, the use of a GTSAM (Georgia Tech Smoothing and Mapping) [8] [10] [9] framework is used to fuse frame-to-frame sonar and video relative pose estimates. A novel sensor fusion method is presented here which incorporates the number of point features matched between frames into the error model of the estimated vehicle trajectory to more heavily weight the impact of estimates with more matches over those with less matches. In this way, stronger estimates from one sensor can overpower the weaker, more inaccurate estimates from the sensor with less point-matches, resulting in a more robust system. This fusion framework, algorithm and evaluation results are discussed herein.

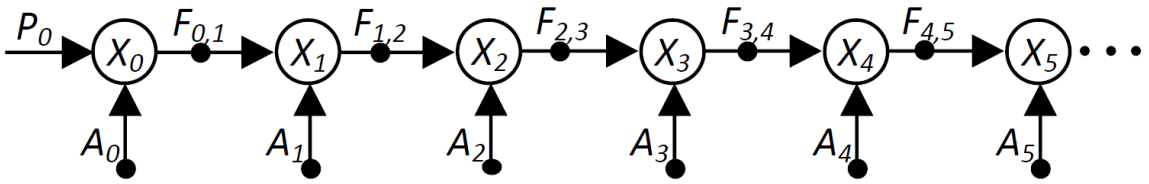
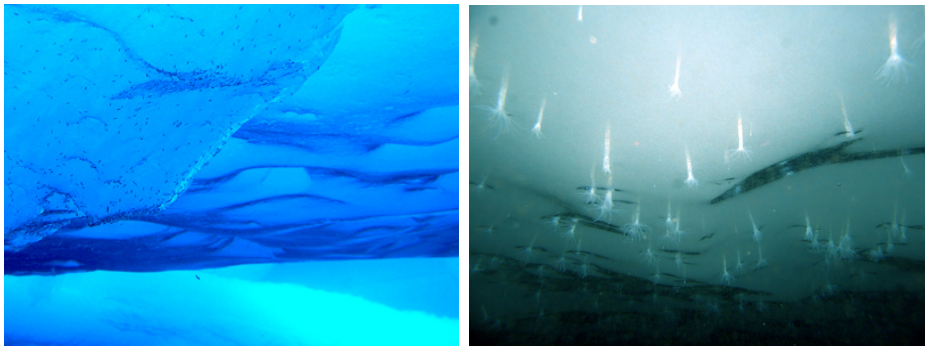


Figure 4: Visualization of a generic factor graph framework

1.4 Texture and Anomaly Mapping

Searching for interesting features under the ice, including animals capable of sustaining life in such harsh environments (Figure 5), is of great interest in both polar (Antarctica)



(a) Arctic Ice Topography (c.o. Arcodiv) (b) Antarctic Anemone (c.o. Rack et al.)

Figure 5: Life present against the ice.

and planetary (Europa) domains. Underwater environments are known to be largely featureless, even at the seafloor which can many times consist only of monochrome sand or rock. Under-ice environments, such as those encountered beneath the Antarctic ice shelves, are even more devoid of features and tend to be monochromatic centered on the blues of the ice. With the advent of remotely operated underwater vehicles (ROVs) and autonomous underwater vehicles (AUVs) has come a large number of video datasets taken in these underwater and under-ice environments. Analysis of these videos can be tedious from both the perspective of human operators (with ROVs) and scientists performing post-processing (with AUVs), as most of the dataset contains no interesting information or unique features. However, all frames must be carefully analyzed to find the few frames of interest. A method for automatically highlighting a relatively feature-rich or uniquely colored frame to bring it to the attention of an analyst is presented here. Using point features (already detected by another algorithm in this application), a novel method for estimating the overall texture of the background ice in an image is also presented. These same point features are used in another algorithm presented here to detect any dense groupings of features to mark as an anomaly candidate. A third algorithm uses a histogram of the hues detected in an image to find any groupings of anomalous hues outside of the blues corresponding to the ice background to mark as anomaly candidates. A final mapping algorithm presented here provides a measurement of the pixels outside the blue hue range as an estimate of non-ice pixels in

the image. A combination of these four approaches can be used to map the texture and anomalous colors of the environment over the vehicle's trajectory, and to map anomalies detected in the globally feature- and color-poor, under-ice datasets of interest.

1.5 Datasets for Algorithm Evaluation

Under-ice sonar and video datasets are required for evaluation of the algorithms in this work, but are not readily available. Therefore, simulated sonar and video data of under-ice environments was obtained along with ground truth position information for additional validation of the algorithms presented here. Real-world, under-ice, concurrent video and sonar datasets were taken using a VideoRay UUV deployed under the ice in a frozen lake in Colorado, as well as the custom Icefin vehicle under the ice in Antarctica (Figure 1). These video and sonar datasets provide means for evaluation of the presented algorithms in the environment of interest. Simulation of the under-ice environments was undertaken using Blender [11] (a visually accurate simulation engine) for camera data and a custom simulation method for sonar data. These simulation methods and datasets are discussed here. Specifications and deployment details of the VideoRay vehicle are also presented in this dissertation. The custom Icefin UUV was designed by a team at the Georgia Institute of Technology specifically to meet the challenges encountered in sub-ice environments. The design and deployment of this vehicle beneath the ice near McMurdo, Antarctica is discussed herein.

1.6 Summary

The sub-ice underwater environment presents unique challenges for navigation and data collection. The objective of the research presented here is to develop algorithms which utilize onboard camera and sonar sensors to enhance the capabilities of UUVs to perform ego-motion estimation and ice anomaly mapping in under-ice environments. A vision-based relative pose estimation algorithm is developed along with a sonar-based relative

pose estimation algorithm to estimate even slight vehicle drift movements due to currents. Fusion of the results from these algorithms using a factor graph framework is used to obtain optimal estimates of the vehicle trajectory during deployment. Algorithms are presented here which use sonar and video datasets to automatically estimate and map the texture of the ice as well as possible locations of marine life anomalies at the water-ice boundary. Results from the evaluation of all algorithms on simulated and real-world under-ice data are presented throughout this dissertation. Use of such methods as those presented here will help further the capabilities of current under-ice vehicle platforms to enable greater exploration of these unique and interesting environments.

The specific contributions of this dissertation are as follows:

1. Development of a novel monocular camera relative pose estimation algorithm for use in low-contrast, featureless, under-ice environments

2. Development of new under-ice texture and anomaly mapping methods

3. Development of an innovative under-ice, sonar-based relative pose estimation algorithm using an optical flow and rigid motion model approach

4. Development of a novel under-ice sonar and video sensor fusion algorithm using a factor graph framework to combine the unique strengths of both sensors

This chapter has provided an introduction to the scope of the problem under consideration in this dissertation, as well as an overview of the research undertaken here to overcome these challenges. A background of previous work in the various technical areas of this dissertation is presented in Chapter 2. The vehicle platforms and field deployments, as well as the datasets produced for algorithm evaluation, are presented in Chapter 3. The simulation methods and corresponding datasets produced for algorithm evaluation are presented in Chapter 4. A camera-based pose estimation algorithm is presented in Chapter 5. The texture and color-based mapping and anomaly detection algorithms are presented in Chapter 6. A sonar-based relative pose estimation algorithm is presented in Chapter 7. A novel

high-level underwater sensor fusion method, based on a factor graph framework, is presented in Chapter 8. Finally a review of the work presented in this dissertation is presented in Chapter 9 along with an overview of proposed areas of interest for future research.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

This dissertation presents methods for under-ice relative pose estimation, mapping, and sensor fusion using sonar and camera sensors on an unmanned underwater vehicle (UUV). To lay the groundwork for the research presented in this dissertation, a survey of relevant literature is provided in this chapter.

2.2 Under-Ice Motivations

Under-ice regions in both the Arctic and Antarctica are of great interest in many domains of science including biology [12], climate science [1], and planetary science [2]. Planetary science, in particular, the study of earth's polar regions and eventually Jupiter's ice-covered, sea-moon Europa [2], is a main motivation for multiple under-ice UUV deployments [3] [4] [5] [6].

Great environmental and technical challenges face researchers when attempting to gather data from the polar under-ice regions of Earth for these applications. Antarctica is Earth's southernmost, coldest, driest, and windiest continent [13] with 98% of its land covered by a thick continental ice sheet [13] and with 44% of its coastline consisting of permanent ice shelves [14]. The average annual temperature at the United States research center (McMurdo station), located south of New Zealand, is -18C and can vary from -50C in the winter to 8C in the summer [15]. In addition to the continental ice sheet and ice shelves, the oceans around Antarctica are extensively covered by both annual and multi-year sea ice (almost 20 million square km area in the winter [16] and up to tens of meters thick). While Antarctica's harsh climate prevents easy colonization, mankind has been driven to explore the continent for almost two centuries. Divers and various observing platforms have conducted some limited exploration below the sea ice and in sub-ice lakes in the Antarctic dry

valleys via auger-drilled access holes in the ice. However, despite scientific interest, exploration of the ocean environment beneath the thick ice shelves has been a nearly impossible feat prior to the advent of new drilling technology, remote sensors, and remotely operated and autonomous underwater vehicles (ROVs and AUVs). Note that in this work, "ROV" refers to a vehicle with no autonomy which requires a human operator, "AUV" refers to an autonomous vehicle with no human operator, and "UUV" is used to refer generically to the entire spectrum of unmanned underwater vehicles. Prior to the introduction of this new technology, under-ice exploration was accomplished with the use of human divers [17] and manned submarines [1]. UUVs greatly facilitate data collection methods in these regions and provide scientists access to previously unexplored areas. UUVs have been deployed under the ice by scientists to collect information on ice topography and thickness [18] [19] as well as chemical composition [20], temperature, and current speed [21] [20] [22] distributions through the sub-ice water column. In many previous missions, this scientific data is collected for studies on climate change and the interactions of arctic waters with other oceans. This includes research on the freshening of seawater [1] and interaction of seawater with the ice shelf [22]. Monitoring under-ice topography is also a primary focus due to the fact that bottom melt is an important indication of ice thinning, where the measured melt rate is proportional to ice thickness [1]. Although many under-ice UUV deployments are motivated by scientific missions, Ferguson et al. [23] [24] [25] demonstrate another motivation for under-ice exploration and seafloor mapping with the political intention of claiming expanded northern borders as part of the United Nations Convention on the Law of the Sea (UNCLOS). Commercial motivations for under-ice exploration include laying transoceanic cables (Theseus vehicle in 1996) as well as offshore oil exploration and monitoring [19] [23].

The technology developed for under-ice UUVs can be extrapolated for planetary exploration missions as well. Search for the possibility of life on Jupiter's moon Europa (Fig. 6)

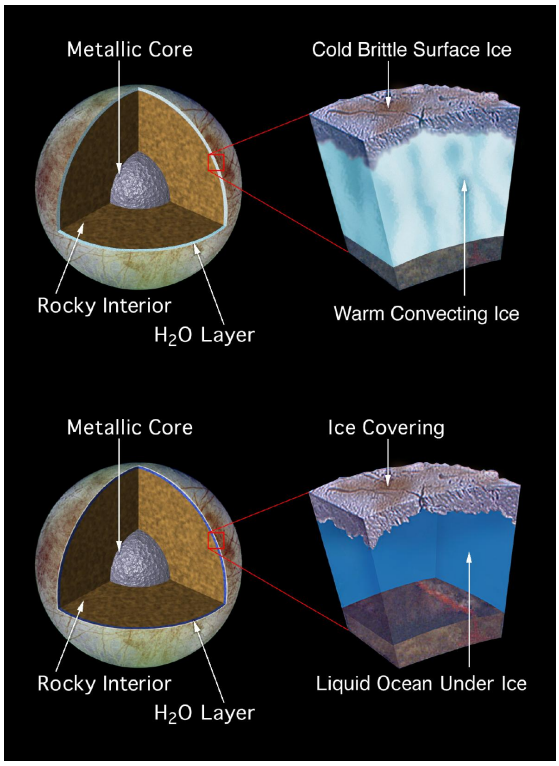


Figure 6: Cutaway of Europa (c.o. JPL and NASA)

began when exploration by the Voyager and Galileo spacecraft suggested that Europa possesses an active ice shell and subsurface ocean [26]. This ocean exists between the moon's outer 3-30km ice shell and its rocky interior. The future exploration of Europa may one day require an autonomous underwater vehicle (AUV) to be able to self-deploy beneath several kilometers of ice to explore possible oceans or water bodies within Europa's ice shell in search of the possibility of life [27] [28] [26] [2]. The closest approximations to such environments that can be found on Earth are located in Antarctica, where large permanent ice shelves form as multiple glacier systems flow off of the continent and stich together [14]. These large, thick ice masses float on the ocean and provide simulated Europa-like environments that are scientifically compelling in and of themselves for their implications for glacial stability and sea level rise. This unique ice-ocean environment presents the possibility of using Antarctic ice shelves as a test location for evaluating the feasibility of proposed Europa missions and technology, while also returning a wealth of data about ice

shelf processes and their stability in a changing climate. Vehicles deployed through the ice in Antarctica can help frame important climate science by obtaining data and imagery of areas on Earth that have been out of reach to other technologies, as well as developing technologies required for the future exploration of Europa.

2.3 Under-ice Vehicle Platforms

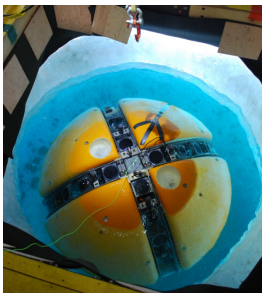
The ice thickness and harsh climate has limited the use of human divers and manned submersibles for sub-ice oceanic exploration. Most data currently collected from beneath the ice shelves is derived from long-lived ocean moorings set through the shelf via hot-water-drilled holes and allowed to freeze into the ice. Mooring instruments and drop-cameras have been successful in obtaining data from the ice and ocean below [29], but are limited to providing data for only a single location during each deployment. Spatially-resolved data over a wide area beneath these shelves, such as that obtained with a UUV, can improve the utility of mooring data that provide a long baseline temporal signal. Such spatially-resolved data can also uniquely constrain how sub-ice and seafloor topography influence ocean circulation and the interaction of ice and the ocean.

The development of ROVs and AUVs over the past few decades has led to advances that enable the exploration of hazardous and extreme environments, such as that found beneath the Antarctic ice shelves and on the ice-moon Europa. One of the main challenges preventing further use of these vehicles is difficulty with navigation and localization given the lack of access to a global positioning system or compass so far beneath the ice and close to the magnetic poles. However, a few AUVs and ROVs have been developed over the past decade specifically for such polar sub-ice oceanic exploration. Under-ice UUVs have been deployed both in the Arctic [25] [19] [21] [30] [31] and Antarctica [3] [31] [18] [6] [22]. While most previous under-ice deployments utilize off the shelf vehicles, custom vehicles have also been designed for this purpose [5] [6] [3]. The custom Icefin vehicle designed and deployed in Antarctica (detailed herein) by a team at Georgia Tech now joins this group.

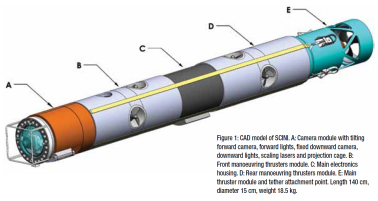
A number of large, complex platforms have made contributions to sub-ice missions: Autosub at 7 meters long and 3266 kg explored beneath the Pine Island Glacier tongue [22], ALTEX at 2.5-6.4 meters with a 53 cm diameter explored beneath ice in the Norwegian Arctic [32], Endurance a 1.2 meter circular AUV at 1089 kg deployed in sub-glacial Lake Bonney, Antarctica [33], Nereid Under Ice (NUI) vehicle [4] (1800kg, 1.8x1.8x3m) deployed in sea ice from a ship in the Arctic, and the Seabed-class vehicles [34] [5] at 2 x 1.5 meters in size (weighing 250 kg) deployed from support ships from the open ocean. The earlier Seabed-class AUVs (Seabed, Jaguar, Puma) were deployed in the Arctic to survey hydrothermal vent sites in 2007 [5] as well as in the Antarctic to survey sea ice draft in 2010 and 2012 [34]. ARTEMIS (heritage from Endurance) will be deployed by SIMPLE in Austral summer 2015 under the McMurdo Ice Shelf [35]. It is worth noting that the NUI vehicle [4] has a single optical fiber tether for use in command and control, similar to that on the Icefin vehicle. However, the NUI tether is expendable and cannot be used for recovery of the vehicle. Some of these vehicle platforms can be seen in Fig. 7.

(a) Seabed Jaguar

(b) Theseus and Explorer



(c) Endurance



(d) SCINI

(e) Autosub

(f) ALTEX

Figure 7: Some Previous Under-ice Vehicle Platforms.

While large AUV and ROV platforms maintain long range traverse capabilities with a variety of scientific sensors, they also carry with them immense logistical requirements,

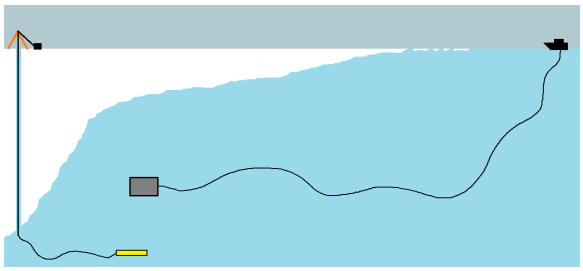


Figure 8: Through ice-shelf deployment method of the Icefin vehicle (left) compared to the ship-based open ocean deployment method of most other under-ice vehicles (right).

requiring deployment via gantry or vessel-based lift systems with large teams and heavy equipment, usually off the side of a ship in open ocean. Because large vehicles like this cannot be deployed directly through the ice shelves, they are forced to traverse many kilometers from the entry point to targets beneath the ice, limiting the penetration range beneath the shelves. Many smaller under-ice vehicles including seagliders [36], and the Bluefin AUV [37] have similar limitations and must also be deployed and recovered from ships. One drawback with using gliders specifically for polar under-ice missions is their difficulty dealing with rapid changes in current and bathymetric conditions as well as their reliance on GPS to navigate (GPS systems have low accuracy at the poles). The Icefin vehicle, capable of deploying directly through the ice shelf, was designed as a human-portable alternative to large, long-range vehicles without the need for such extensive support infrastructure, while increasing the number of onboard science sensors relative to smaller platforms. Figure 8 depicts the advantages of Icefin's novel through ice-shelf deployment over common ship-based open ocean deployments. Design and deployment of the Icefin vehicle is presented in Chapter 3.

Holes drilled through an ice shelf for access to the ocean below are generally kept small to lower the cost of fuel, which adds mass and cost to field deployments. Few state of the art vehicles deploy directly through the ice due to these constraints. Of those that do, even fewer are able to deploy through thick ice shelves and instead deploy through the thinner sea ice. Some early work in this area includes development of the Theseus AUV (10.8m

length, 1.28m diameter, 8,600kg) [23], designed to lay fiber-optic cables in the ice-covered Canadian arctic. Theseus was able to deploy beneath 1.7 meter thick ice through a 2 x 13 meter hole and set records for under-ice mission duration. However, the Theseus vehicle is too large to be deployed and retrieved through a small diameter hole in the ice shelf, as required of Icefin. Small observing UUV platforms have been used to make observations and assist divers below the sea ice, including the Submersible Capable of under-Ice Navigation and Imaging (SCINI) [3]. The SCINI ROV was developed as a small, human-portable, tethered vehicle with a camera for real-time control and exploration of the sub-ice environment by a human operator. The SCINI vehicle was deployed by the SIMPLE project in 2012 and in 2014 along with the Icefin vehicle, and provided some of the heritage for the design of the Icefin vehicle. SCINI is 4ft in length and 20 cm in diameter, weighing 15.9 kg in air. Both the Icefin and SCINI vehicles have the unique capability to be deployed and recovered through a small diameter hole. SCINI has been successfully utilized over several Antarctic field seasons and is a useful tool for initial ice shelf exploration as a roving eye as well as for vehicle-assisted diver operations in sea ice environments. However, SCINI's requirements did not necessitate the additional deep water operation (1500m), greater range (3 km), localization capabilities, or full oceanographic sensor-suite capabilities designed into the larger Icefin vehicle. The SCINI vehicle's man portability, hovering capability, and deployment through small diameter ice-holes have helped influence the development of Georgia Tech's Icefin [38] [39] vehicle design. The SCINI and Theseus vehicle platforms can be seen in Fig. 7.

A few predecessor missions have explored Earth environments as planetary analogs in ways similar to those that may be one day employed on ocean planets like Europa. Among the first was the DEPTH-X vehicle [40] used for deployment and science sampling within deep cenotes between 2006 and 2007. This vehicle was then adapted to become the ENDURANCE vehicle [33] deployed through the ice at Lake Bonney, Antarctica in 2008 and 2009 Austral summers. In 2007, the Woods Hole Oceanographic Institute used AUVs

to survey hydrothermal vent environments and provided extrapolations for deployment of such a mission to Europa [5]. Each of these missions identified unique challenges for future AUV deployment to Europa as they relate to polar operations. Many of the Europa-specific AUV mission elements proposed by Kunz et al. [5] have been incorporated into the design of the Icefin vehicle (deployment through thick ice using a small diameter hole, smaller vehicle size, and robust communication methods). While some initial investigations have taken place using current AUVs to begin to find solutions to these challenges, Icefin is unique in incorporating these Europa-specific mission requirements and previous lessons learned into the design of the vehicle and its deployment methodology.

While a few ROVs and AUVs have been developed over the past 20 years for sub-ice deployment, none couple the human-portability, modularity and small diameter needed for through-ice deep field deployment with a sufficient oceanographic sensor suite and navigational capability required for the missions at hand. We hope to fill this void in the spectrum of available tools with the development of the Icefin vehicle. The Icefin system design has been focused on developing technologies and exploration strategies that can overcome challenges for both polar and planetary operations.

2.4 Under-Ice Sonar Use

The use of sonar to image under-ice topography (Figure 9) is valuable for many applications. The first under-ice profile of ice topography was obtained in 1958 from the USS Nautilus with a single-beam upward-looking sonar [18]. Sidescan sonar was first used to map ice topography in 1976 from the HMS Sovereign, followed by the first use of 3D multibeam sonar in 2004 from the Autosub II AUV [18]. Many UUV under-ice deployments record sonar data for offline analysis [31] [32] [19] [18] [22], while few others actually use the topography to aid in vehicle localization. Iceberg-relative navigation with a UUV [30] was performed by Kimball and Rock using simultaneous localization and mapping (SLAM) [7] with sonar data. SLAM refers to a localization method commonly used

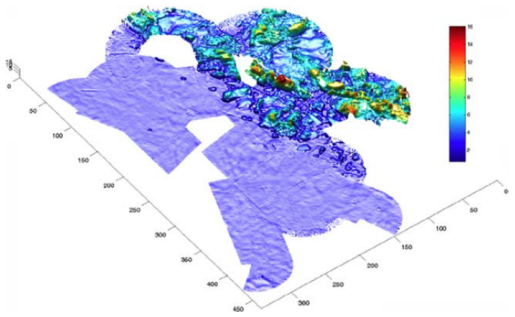


Figure 9: Smooth first-year and rough multi-year ice topography [18]

in terrestrial robotics where observations of landmarks in the environment are used to concurrently create a map of the environment and localize the vehicle inside that map as the vehicle moves. Another SLAM application is proposed by Kunz and Singh [41] and was tested in an under-ice environment. As a result of lack of topographical features present in many under-ice environments to provide significant and unique sonar returns, investigation into the use of sonar as a navigational aid in such environments has been limited in previous work.

2.5 Under-Ice Navigation

Underwater navigation is very difficult due to the lack of access to global positioning systems (GPS) commonly used in terrestrial applications. Navigation and localization under the ice presents additional challenges on top of those encountered in the already difficult underwater navigation and localization problem. The dead reckoning systems that form the baseline navigation solution for almost all under-ice platforms [4] [5] [6] [18] [21] [23] [32] incorporate accelerometer and gyroscope data from an inertial measurement unit (IMU) and velocity data from a Doppler velocity log (DVL) [21] [19] [6] [22] [5] [18], usually through a Kalman filter [19] [6] [21]. However these inertial navigation systems (INSs) are prone to dramatic drifts in position estimates over time, especially through the double integration of accelerometer data. Long baseline (LBL) [19] [4] [5] [3] or ultra-short baseline (USBL) [6] acoustic beacon systems are used with many previously deployed under-ice

vehicles to bound the error drift encountered in dead reckoning INS systems (sometimes down to 2% [42] or 3% [20] distance travelled). These LBL and USBL systems are comprised of acoustic beacons that are used to calculate time of flight and direction information to a transponder on the vehicle for localization of the UUV, but require substantial external infrastructure and setup effort and also limit the deployment area. Compass readings can prove inaccurate [21] [3] at extreme latitudes, further complicating the navigation problem. Some previous under-ice systems have used USBL [31] [6] [18] [19] [42] [23] or a strobe light [6] as a homing beacon to aid in return of the vehicle after accumulated inertial navigation drift.

The use of automated sonar and video processing as a navigational aid, presented here, is not common in under-ice or even underwater systems. Some previous under-ice work has used sonar [41] or monocular camera [6] based algorithms to aid in localization of the vehicle. While there has been some initial research in this area, these SLAM and visual relative pose estimation methods (sonar nor camera-based) are not commonly used with under-ice UUV deployments due to their lack of maturity, lack of unique features in ice topography, and the need for proven, robust and reliable navigation systems in the extreme arctic environment. However, systems utilizing common sensors already onboard a UUV (such as camera and sonar sensors) to aid in localization of the vehicle are highly desirable. A system, such as that presented here, that is capable of utilizing both sonar and video data to aid in vehicle ego-motion estimation and ice anomaly (and texture) mapping in relatively featureless under-ice environments is not present in the current literature. While uncommon, some previous work in the areas of underwater sonar and video pose estimation as well as sensor fusion is discussed in this section and has provided some groundwork for the development of the algorithms in this dissertation.

2.6 Acoustic Sonar Processing

Automated acoustic sonar processing has steadily increased in popularity for applications such as object detection (or obstacle detection) [43], tracking [44] [45] [46], and SLAM [47] [48] [49]. Detection of manmade objects (mostly mines) in sonar data is a popular subject of research and uses strong areas of backscatter return in the data, followed by a shadow (lack of backscatter return) as a detection template [43] [48]. In previous sonar processing applications, blob or cluster detection algorithms are the most prevelant [50] [44] [51] [48] [49] [52] [53] and are used to find concentrated areas of strong backscatter return in the sonar frame corresponding to a possible object or landmark [53], sometimes for use as features for SLAM [49] [53]. Most of these blob-based algorithms [52] [46] [47] use features such as blob centroid, area, perimeter, and second moments to describe and match features between sonar frames. Due to the lack of unique topographical features present to provide such significant and clustered sonar returns in most under-ice environments, investigation into the use of sonar as a navigational aid in such environments has been limited in previous work.

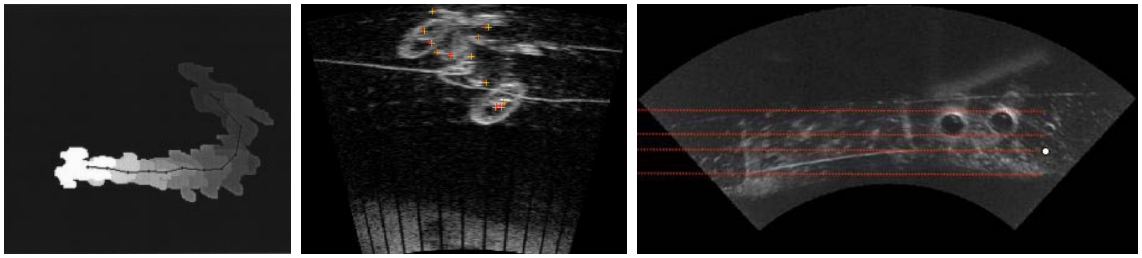
Sonar-based topography measurements can also be used as features for matching, such as in [54] (motivated by an under-ice mapping scenario) and in [30] [55] [56], using terrain relative navigation (TRN). In a similar application to that presented here, iceberg-relative navigation with a UUV [30] was performed by Kimball and Rock using sonar-based SLAM. However, this method requires a previously known map of the environment for localization and uses a terrain-based matching approach instead of the point-feature matching approach used here. Texture can also be used [57] [58] to automatically segment or describe areas in an image. Mallios et al. [59] use a scan-matching method with sonar images for use in SLAM.

Although much of the previous work in automated sonar processing for tracking and detection is limited to blob-based methods, new high-frequency and high-resolution sensors

(i.e. the BlueView P900-45 multibeam sonar [60]) introduce the possibility of using point-feature (i.e. Shi-Tomasi [61], Harris [62], SIFT [63] and SURF [64]) tracking methods, popular in optical image processing. While similar point feature methods have been used in some early sonar processing applications [65] [66] [67], this remains uncommon. Kim et al. [65] use Harris corners with a multi-scale Gaussian pyramid approach for feature detection and matching with the purpose of creating a sonar mosaic. They use an affine transformation as a good approximation of feature movement between frames and examine the projective transformation model (instead of the pinhole camera model commonly used in optical imaging) of the sonar imaging camera in detail. A robust model estimation method similar to RANSAC [68] and Least Median of Squares [69] is used in [65] to obtain estimates of the model parameters for this affine transform. Harris corners are also used as point-features with sonar images in [66] [67] described below. While point features have been used in some early sonar processing research, this is still very uncommon. This is especially true for unstructured environments and for ice texture and vehicle motion estimation applications.

Optical flow [70] refers to a method for determining motion of pixels between successive image frames in a sequence. Optical flow based tracking is not commonly used with sonar images due to high noise levels present and non-conformity of the sonar images to the underlying constant-brightness assumption of optical flow. However, some early sonar optical flow research by Lane et al. [45] [46] presents such a method to obtain motion estimates for objects in the sonar data to constrain search areas to a region of interest in the next frame (Fig. 10). In a similar application to that presented here, a Harris point feature detector and the optical flow based Lucas-Kanade tracking algorithm [71] are used by Sekkati and Negahdaripour [66] to estimate 3D motion estimation from 2D sonar images (Fig. 10). Motion estimation is derived from temporal correlation across successive frames using an affine transformation model with an added dependency on elevation angle (their homography matrix is given in [66]). However, this work assumes planar scene surfaces in

the motion model, which does not hold for an unstructured environment such as the sub-ice topography considered in this dissertation. Negahdaripour et al. [67] present an optical flow based method for registration of multibeam sonar images for use in creating mosaics (Fig. 10). This motion model assumes a fixed affine mapping over the entire image (conformal mapping with scale change, rotation, and x,y shift). While these few pioneering applications introduce the use of sonar-based feature matching and optical flow algorithms and prove their utility, such methods are not currently prevalent in previous literature. Previous work on the use of sonar-based optical flow or point-feature matching methods in applications of real-time ego-motion estimation in feature-poor, unstructured environments do not exist in previous work. This is especially true in the realm of under-ice applications.



(a) Lane et al. (b) Sekkati and Negahdaripour (c) Negahdaripour et al.

Figure 10: Previous sonar processing work

One of the difficulties encountered when working with sonar data is the low signal-to-noise ratio. Sources such as backscatter, reflections, and other acoustic transmitters inject large amounts of noise into the data. Therefore, preprocessing of the sonar images is required prior to analysis [45] [72] [52] [44] [51]. This preprocessing usually involves median [45] [72] or Gaussian [52] [44] [51] filtering to remove noise, followed by thresholding [52] [44] [45]. Morphological filtering [73] (usually opening or closing) is also popular [74] [47] when filtering sonar images. Histogram equalization is used in [75] to compensate for the low dynamic range encountered in sonar datasets. Temporal filtering is suggested in [45], but most previous work avoids such filtering due to the time lag introduced. Due to the noisy nature of the sonar data, most of the previous research encountered in the literature requires structured environments or multiple objects of strong acoustic

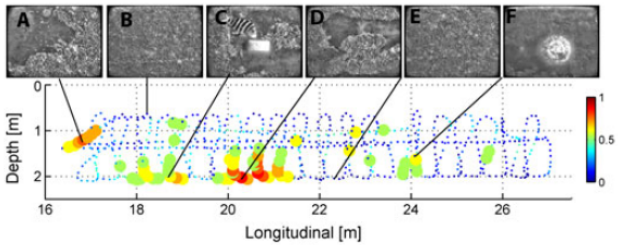
return in the view of the sensor to yield successful results. Investigation into the algorithmic analysis of sonar data obtained in unstructured and relatively featureless environments, such as those here, is not prevalent in the literature.

2.7 Monocular Video Processing

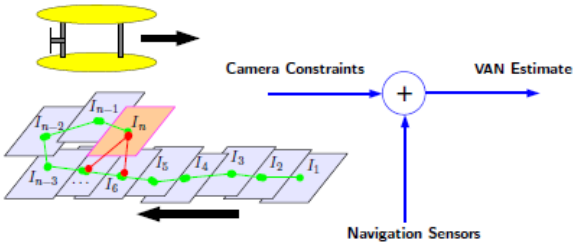
Optical cameras are not commonly used in automated underwater navigation or mapping applications due to the poor visibility [6] [66], lack of light [76] [48] and heavy processing [77] demands encountered with this sensor. However, monocular camera systems have been utilized in some underwater applications of SLAM and ego-motion estimation [76] [52] [78] [79] [72]. Almost all such processing algorithms require an undistorted image from a calibrated camera [78] [76] [72] [80], usually assuming a pinhole camera model [76]. In almost all cases, contrast limited adaptive histogram equalization (CLAHE) is used [79] [80] [72] [81] to compensate for the low contrast encountered with underwater and arctic imagery. CLAHE [82] is used to adaptively adjust the contrast of local pixel neighborhoods based on the intensity histogram distribution in that neighborhood.

SIFT [63] and SURF [64] features are used by Beall et al. [78] [83] to estimate camera movement between frames (Fig. 11). A three-point algorithm employed in a random sample consensus (RANSAC) [68] framework is used to recover the rotation and translation between frames for use in a smoothing and mapping factor graph SLAM application. Monocular vision-based SLAM [7] is performed by Kim and Eustice [79] using bag-of-words [84] methods for matching landmarks using sets of point features in a ship hull inspection application (Fig. 11). Pairwise image registration in a pose graph framework is used to provide a six degree-of-freedom (modulo scale) relative pose between nodes. A pose-constrained correspondence search (PCCS) and RANSAC [68] geometrical model selection framework is introduced for feature robust matching between images using a strong prior on vehicle motion to provide a bound for matching (useful in feature poor imagery). Image saliency (distinctiveness) is used to sparsify the pose graph to keep only beneficial

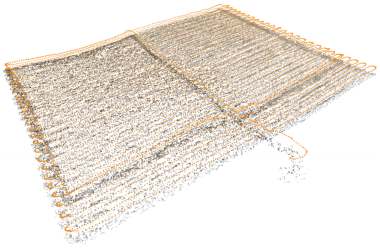
constraints with high feature richness (local saliency) and to determine unique regions of the hull that would be ideal for loop closure (global saliency). Image saliency is calculated based on the bag-of-words [84] histograms for each image and is used to determine link proposal events (such as loop closures). Similar relative camera pose estimation work is presented in [85] using SIFT [63] and Harris features [62]. Eustice [76] presents an implementation of pose graph SLAM with camera-derived motion estimates, termed visually aided navigation (VAN), to provide relative pose constraints for trajectory consistency and loop closure (Fig. 11). Rather than tracking features over time, overlapping image pairs are matched (with at least six Harris and SIFT features) to get relative orientation and direction (essential matrix) between camera poses. A similar method is used by Hover et al. [72] with SIFT [63] features to provide camera-relative pose constraints during ship hull inspection.



(a) Kim and Eustice



(b) Eustice



(c) Beall et al.

Figure 11: Previous underwater vision-based navigation work

Frame-to-frame six degree-of-freedom motion (or odometry) is estimated in [80] by solving for the fundamental matrix [73] between image frames, using an algorithm which requires eight point correspondences and optical flow. Salvi et al. [80] use stereo cameras to create a cloud of 3D feature points with a mix of SIFT [63] and SURF [64] descriptors, where landmarks encompass a spatially grouped set of these points for use in

SLAM [7]. A "video velocity log" is calculated between frames using a pyramidal implementation of the optical flow [70] based Lucas-Kanade [71] feature tracker. The Shi-Tomasi-Kanade [71] [61] tracker and RANSAC [68] is used by Trucco et al. [52] to robustly estimate the fundamental matrix between image frames in an underwater environment. Features are tracked through multiple frames until no longer matched. When enough features have been dropped, the algorithm is reinitialized. Aulinas et al. [86] present a novel method for detection of strong landmarks for SLAM [7] by segmenting the image (with either edge detection or hue channel thresholding) into background and landmark candidates. These candidates are then used as regions of interest to search for SIFT [63] and SURF [64] features. If found to be sufficiently salient, these regions are designated as SLAM landmarks. Although a terrestrial application, it is also worth mentioning the monocular visual pose estimation algorithm presented by Williams [81] for use in feature-devoid arctic environments, as this work has provided some groundwork for the work presented here. Harris [62] and SIFT [63] features are used with Nister's [87] five point algorithm to calculate the essential matrix (camera rotation and translation) [73] between image frames using a robust parameter estimation method termed maximum a posteriori sample consensus (MAPSAC) [88]. Nister provides a terrestrial method for visual odometry motion estimation using a five-point algorithm with RANSAC [89]. Robust parameter estimation is commonly performed to estimate sensor motion between image frames using RANSAC [52] [89] [85] [79] [41], Least Median of Squares [76] [41] [80], Maximum Likelihood Estimation (MLE) or MAPSAC [81]. These robust parameter estimation algorithms are detailed in [68] for RANSAC, [69] for least median of squares (LMedS), [90] for maximum likelihood estimation sample consensus (MLESAC), and [88] for maximum a-posteriori sample consensus (MAPSAC).

Some monocular camera vision-based applications and tools from the literature are presented in this section. While the use of a monocular camera sensor to provide relative pose

constraints between image frames has been established in terrestrial and some underwater applications, most require feature-rich environments to provide sufficient matches for an estimated motion model. The low-contrast, feature-poor, under-ice environment considered here presents unique challenges for the use of such a method. The application of monocular visual pose estimation in the sub-ice environments considered here is not found in previous literature and requires novel tuning of these previously proposed visual odometry methods to provide the desired ego-motion estimates. However, this previous literature has helped form some of the tools and groundwork for the development of the algorithms presented here.

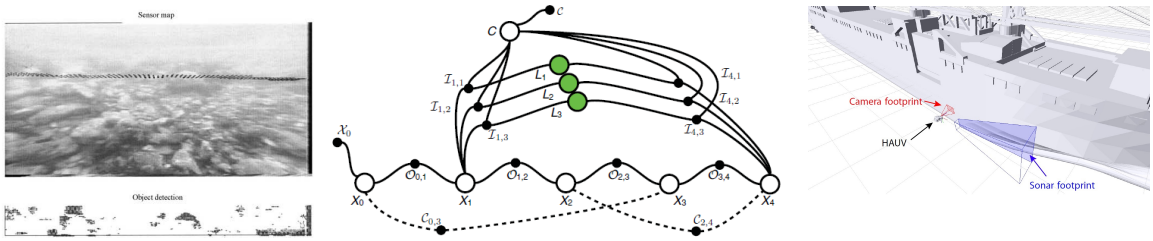
2.8 Sensor Fusion

Exploiting the complementary and independent information obtained between range and optical sensors through fusion has been used in a few previous unmanned terrestrial (ground) [91] [92] [93] [94] [77] [95] and unmanned underwater [96] [41] [50] [97] [98] applications. However, very few systems in the literature investigate fusion of underwater sonar and camera sensors for use in cross-sensor mapping or ego-motion estimation, as presented here. Ground-based systems commonly use either LIDAR [91] [99] [95], laser range scanner [92] [93] [94], or ultrasonic [100] sensors with monocular [91] [93] [94] [77] [98] or stereo [97] [101] camera configurations. Some systems [91] [92] [102] use co-registered range and image data to create 3D point cloud maps for more robust feature extraction and mapping. Many applications [97] [93] [94] [77] of terrestrial sensor fusion match 3D edges and corners (in structured environments), which are visible in both datasets, for improved sensor cross-calibration and for use as landmarks. Many times such landmarks are used for SLAM [7] [97] [94]. Sensor fusion in these cases provides more robust (relative to vision only) and accurate mapping and matching of landmarks due to the accurate depth provided by the ranging sensor. In other applications [99], ranging sensors provide regions of interest to search in the imaging data. Matthies and Elfes [100] fuse stereo vision and

range data at a high level to detect occupied cells for obstacle avoidance and path planning. Zhu et al. [95] fuse 3D visual landmark matching (from 3D LIDAR) with visual odometry estimates to reduce errors from both systems. Song and Tang [77] use ultrasonic sensors for distance measurement and a vision system for more accurate object boundary detection. Zhang and Pless [103] describe how a laser range finder and camera sensor fusion system is calibrated. Neira et al. [104] use a laser range sensor to find walls in front of the robot, and a camera to find vertical edges corresponding to corners or door frames. While these terrestrial sensor fusion methods provide insight into the synergy possible through the use of a range-optical sensor system, such methods are not common in underwater, under-ice, unstructured, or relatively featureless environments such as those considered here.

Some underwater sensor fusion methods have also been investigated. Singh et al. [105] balance the strengths and weaknesses of multiple sub-sea 3D mapping methods (image structure from motion and sonar bathymetric 3D) by fusing the datasets to create an optimal 3D map of an underwater environment. Krout et al. [106] present an underwater acoustic sonar and above-water camera fusion for buoy detection. Cross-calibration of an acoustic camera and an optical camera for shape recovery of underwater targets exploiting epipolar geometry constraints is discussed by Negahdaripour et al. [98] and Sekkati and Negahdaripour [66]. A planar calibration grid visible in both sensors is used for sensor cross calibration. Once cross calibrated, the two sensors can be used to exploit epipolar geometry constraints (similar to stereo optical systems) between them to better determine underwater 3D shape over stereo optical systems. Majumder et al. [50] propose a low level fusion of sonar and camera sensors prior to feature extraction for more reliable mapping and navigation (Fig. 12). These sensors are combined using projection and a perspective transformation to project all sensor data into a common state space prior to feature extraction in an attempt to improve feature detection underwater. Due to the lack of stable corner features underwater, blobs are used as features and texture is determined from color analysis. Williams and Mahon [96] propose fusion through projection of strong sonar cluster

returns into camera images to provide initial search locations of visual point-features used for SLAM on the Great Barrier Reef (a very unstructured environment). The sonar is set to profile mode (in contrast to the methods here) to determine the location of the seafloor, and strong returns are projected into the camera image to initialize high contrast features in the SLAM map. Once features are initialized with sonar data, the vision system (with a Lucas-Kanade [71] tracker) is used to track the feature positions. While these low-level fusion methods exploit the strengths of both camera and sonar sensors, they require consistently reliable input from both sensors (in contrast to higher level fusion methods such as that presented here).



(a) Majumder et al. (b) Kunz and Singh (c) Hover et al.

Figure 12: Previous underwater sensor fusion work

A high-level sonar-camera sensor fusion technique for relative pose estimation is developed in this dissertation. While similar work is present in the literature, none utilize multibeam sonar and camera sensors for navigation in relatively featureless under-ice environments. Monocular camera and multibeam sonar fusion in a pose graph SLAM framework is used by Kunz and Singh [41] to automatically build 3D maps of underwater terrain (Fig. 12). A pose graph framework is used with square root information smoothing and mapping to optimally solve for the vehicle trajectory, map and camera location. Multibeam sub-map matches and visual feature matches are used as relative pose constraints in the graph, but here the calibration of the sensors and extrinsic camera position are also set as special nodes in the pose graph. Therefore, this system does not require pre-calibration of the sensors as it is done as part of the global pose map optimization at run time. A scaled Harris corner [62] detector with Zernike polynomial descriptors and RANSAC [68]

or Least Median of Squares [69] robust parameter estimation is used for image matching to estimate a six degree-of-freedom (modulo scale) model of camera motion. Similar underwater sensor fusion is performed by Hover et al. [72] for vehicle position estimation during ship hull inspection by combining DIDSON sonar and monocular camera constraints using a pose graph framework and iterative smoothing and mapping (Fig. 12). In this case, fusion occurs at a very high level. Sonar and camera clients both add relative pose constraints to the pose graph while iterative smoothing and mapping is used to optimally (least squares estimate) solve for the vehicle trajectory over the entire graph. Features in the sonar data are based on clusters of strong gradients while SIFT [63] features are extracted and matched in the image data. Optical camera image registration provides a five degree-of-freedom constraint between two camera nodes in the graph. Both sonar and video data can provide sequential constraints and loop closure constraints in the pose graph. While these pose graph sensor fusion methods are similar to the work presented here, these applications use ice draft topography [41] (requiring uniquely varying ice draft) and clustered, strong gradient pixel returns [72] as features to provide sonar constraints, in contrast to the point-feature-based odometry constraints used here. These previous pose graph systems use SLAM instead of odometry methods, which require unique landmark features to be present in both datasets that can be saved and robustly matched over the vehicle trajectory, especially during loop closure events. While these systems can obtain very low errors assuming re-observation of unique landmarks, this assumption breaks down in the case of continuously textured but non-uniquely varying environments, such as sand-ripples and the majority of ice topography. While these previous sensor fusion systems have proven the utility of such methods in underwater environments, systems robust to the unique feature-poor, under-ice environments, such as that considered here, do not currently exist in the literature.

2.9 Mapping

The use of point features (pixel locations with large, unique local gradients) such as SURF (Speeded Up Robust Features) [64] features or Shi-Tomasi corners [61] for texture estimation in an image has been explored in some previous work for categorizing images [107]. However, previous methods require extensive processing and have not been applied to underwater or under-ice autonomous systems, such as the application here. Kim and Eustice [79] use point features to estimate saliency (uniqueness) and feature-richness of an image, which requires great processing effort to calculate. Presented here is an algorithm with very low additional computational cost to get a real time estimate of the texture in an image using point features that have already been extracted for use in another algorithm in the system. While a few previous underwater vision applications use point features in video to perform SLAM [79] [76] [52] [78] [79] [72], none currently use the features detected to estimate if there is an anomaly of interest in the image across a globally feature poor dataset (commonly encountered in underwater and under-ice environments). Such a method for detection of anomalies in a largely featureless under-ice environment is presented here. Color is commonly used for dissecting an image into foreground and background or segments [108], even in underwater applications [109]. However, such a color-based approach is not commonly used in under-ice applications due to the nature of the environment as largely monochrome and featureless. Despite these challenges, this unique application requires a means for searching the monochrome ice background for any anomalies (specifically animals) that might be located at the ice-water boundary. Therefore, a color-based algorithm is presented here using hue histograms and clustering to determine outlying colors corresponding to anomalies in under-ice environments.

2.10 Summary

In this chapter was presented a summary of relevant literature in the technical areas of this work. While many prior works have investigated the use of sonar or camera sensors to

aid in localization, the use of such a system in a feature-poor, under-ice environment has not been examined. The use of underwater sonar sensors for localization has been mostly limited to applications with blob features of significant acoustic return or structured environments such as pools. Use of optical flow and point-feature-based algorithms with sonar, such as those considered here, is very uncommon in the literature. Camera-based monocular pose estimation applications remain largely terrestrial-based, with the few underwater applications commonly requiring feature-rich environments for successful results. While range and optical sensor fusion has been explored, use of underwater sonar sensors with camera sensors is very uncommon, especially in featureless and under-ice environments. Sonar- and video-based algorithms have not been previously used for automatic ice texture estimation and ice anomaly detection. The work presented here includes sensor fusion of underwater acoustic sonar data with camera data to aid in ego-motion estimation of a UUV in a feature-poor, under-ice environment. The use of optical flow is presented as a useful tool for analysis of sonar data in these applications. Camera-based monocular pose estimation shows promising results despite the lack of strong features in under-ice environments. Fusion of data from sonar and camera sensors, which are already onboard many underwater vehicle platforms, can provide a solution to aid in ego-motion determination of a UUV, without requiring any additional infrastructure. The use of two sensors also provides a more robust means of estimation, which is useful in such difficult environments. The use of sonar and video sensors for automatic ice texture estimation as well as ice anomaly mapping is also presented. Availability of previous data collection using forward-looking sonar and optical camera sensors in sub-ice environments is limited. Previous sonar and camera simulations of under-ice environments are not readily available from previous work. Therefore, datasets for the evaluation of the algorithms presented here are obtained here directly through simulation and under-ice field deployments. The under-ice UUV frontier is relatively new in and of itself, and the novel use of the algorithms presented here will help to advance capabilities in the field.

CHAPTER 3

DATA COLLECTION AND VEHICLE PLATFORMS

3.1 Introduction

Navigation and mapping in relatively featureless underwater and under-ice environments is a difficult challenge. Using common sensors already onboard many unmanned underwater vehicles (UUVs) to aid in this process provides a self-contained and cost-effective solution. This dissertation presents methods for relative pose estimation, sensor fusion, ice texture estimation, and anomaly detection using camera and sonar sensors in an under-ice environment. To evaluate the algorithms in this work, multiple UUVs are used to gather the required sonar and video data in underwater and under-ice environments. Use of a UUV for data collection allows for remote control of the position and speed of the system, some ground truth information from onboard sensors, and a realistic application environment for the methods under evaluation. The robotic platforms deployed in these field trials include the VideoRay Pro IV and custom Icefin vehicles. Both of these vehicles are equipped with the necessary forward-looking sonar sensor and cameras needed for data collection of the type required for evaluation of the algorithms presented here. The vehicle platforms used and their field deployments to date are detailed in this chapter. To obtain additional under-ice data in a more controlled environment with full ground truth, simulated datasets of the environments of interest were also created. The simulation methods used in the creation of these additional datasets are detailed in Chapter 4.

3.2 VideoRay Pro IV

3.2.1 Platform Overview

The VideoRay Pro IV UUV (Figure 13) is a tethered, commercial off-the-shelf vehicle. The physical dimensions of the VideoRay are approximately 33cm in width, 40cm in length

and 23cm in height. The sensor suite includes an onboard camera and an external forward-looking sonar sensor. This vehicle has a single vertical thruster for control of heave motion, and two more powerful horizontal thrusters for control of yaw and surge motion. The onboard tiltable camera is a Sony WDR380 CCD image sensor, built into the vehicle behind a clear plastic dome, capable of providing a 768x494 pixel resolution. The sonar instrument is a BlueView P900-45 forward-looking sonar sensor mounted to the bottom of the vehicle. To provide support for multiple tilt angles of the sonar sensor, custom running boards were built for the vehicle, with mounting holes at the desired angles.

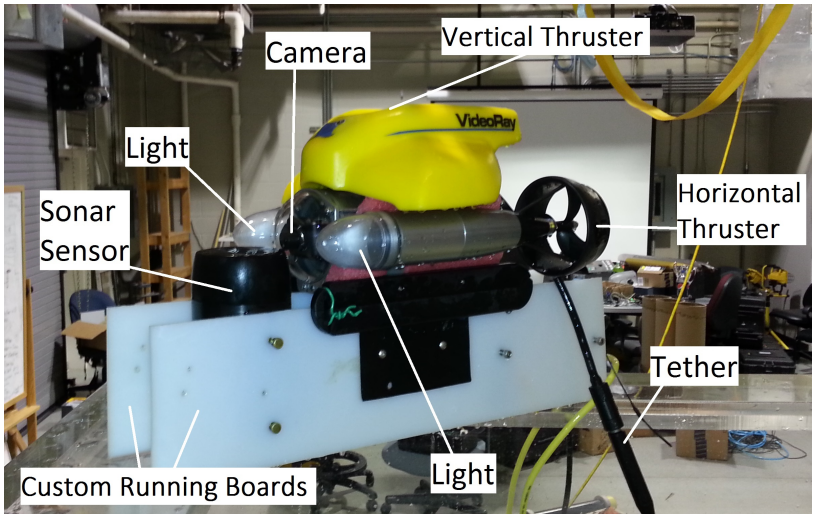


Figure 13: VideoRay Pro IV Vehicle

In addition to the camera, the vehicle also has a built-in compass to provide heading, an inertial measurement unit (IMU) to provide measurements of 3-D acceleration and rotation, and a pressure sensor to provide depth measurements. The IMU of this vehicle was determined to provide insufficient accuracy for absolute localization, but could be used to aid in short duration local navigation. The VideoRay vehicle has built-in low-level processing capabilities including controllers for the actuators (thrusters). However, all high-level processing, user interface, and control is accomplished from the surface computer, with command and control communication over the tether. User interface software allows for control of the vehicle actuators, acquisition of camera and sonar images, control of the

camera tilt angle, and access to the internal vehicle state sensors from a surface computer. Time-stamped and synchronized images from the camera and sonar sensors can be recorded for offline analysis. Compass and IMU data can also be recorded concurrently with each frame for use as ground truth. Using topside control software packages, commands are sent over RS-485 down the tether connected to the vehicle, while status and video data are received back in the same manner, and a separate tether provides sonar data back to the base station. Both heading and depth of the vehicle can be automatically maintained at a desired set point value with the control software. The vehicle can also be controlled manually. A handheld controller can be used to control the vehicle camera, forward thrusters, vertical thruster, and lights. This provides the ability to easily command the vehicle around in three dimensional space. Full specifications on the VideoRay Pro IV vehicle can be found in [110]. While this vehicle has been deployed in an under-ice lake environment, it has limited deployment capabilities in the harsher polar under-ice missions.

3.2.2 Hydrodynamics Model and Dead Reckoning

The hydrodynamics of the VideoRay vehicle were analyzed in [111], allowing for the development here of a vehicle hydrodynamics simulator in software [112]. This vehicle is very stable and thus encounters limited movement and disturbance in the roll and pitch directions, allowing for simplification of the dynamics model. Based on the hydrodynamic characteristics of the vehicle along with the command inputs given to the actuators during a mission, an estimate of the internal vehicle state can be determined over time. This is referred to as dead-reckoning and provides a method for vehicle localization estimation based only on the control inputs to the thrusters. While dead-reckoning estimates are very noisy and prone to drift, they can be fused with other means of localization for more robust positioning estimates. The hydrodynamics simulator developed here is also useful for simulation of a mission prior to deployment of the vehicle. While development of a hydrodynamics simulator and dead reckoning system for the VideoRay Pro IV

vehicle was undertaken in early work here, this is out of the scope of the dissertation presented here. However, a detailed work on this dead reckoning and simulation system can be found in [112]. Dead reckoning provides a valuable tool for sensor-deprived navigation and should be considered for integration into any robust localization and navigation system.

3.3 Icefin

3.3.1 Introduction

Exploration of the furthest reaches of our planet, as well as other planetary bodies, typically requires the use of robotic platforms due to the extreme environments encountered. Some of the harshest conditions on Earth are encountered in Antarctica and require the use of autonomous underwater vehicles (AUVs) to explore the remote and hazardous areas beneath the ice. The future exploration of Jupiter's moon Europa may one day require a similar AUV to be able to self-deploy through several kilometers of Europa's ice cover to explore its ocean or water bodies within its ice shell in search of the possibility of life [27] [28] [26] [2]. The closest approximations to such environments that can be found on Earth are located in Antarctica, where large permanent ice shelves form as multiple glacier systems flow off of the continent and stich together. The custom Icefin under-ice unmanned underwater vehicle (UUV) has been developed for deployment in such ice-covered oceans as those found in Antarctica and the Arctic with the intent of furthering relevant technology for future Europa missions. Lessons learned from Antarctic deployment of the Icefin vehicle can be extrapolated to future polar and planetary AUV design.

The novel Icefin vehicle (Figure 14) is unique within the sub-ice UUV class as it combines human portability with an extensive science sensor suite, a modular design, a fiber optic tether, and a unique deployment and recovery method directly through the thick (hundreds of meters) ice shelf. This relatively small but capable vehicle has relatively low impact on logistics, since it can be broken down into individual modules and transported using a light ground vehicle, boat or helicopter to any deployment site of interest; this is not the case for heavier and larger under-ice vehicles. The Icefin vehicle contains a suite of

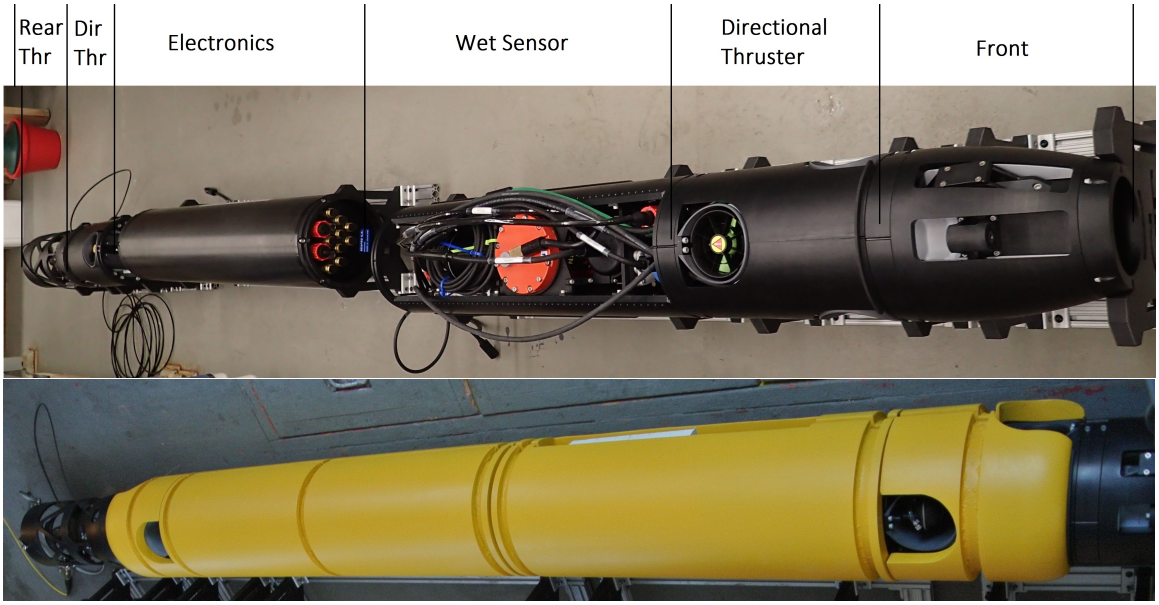


Figure 14: The Icefin modular vehicle in modules (top) and completely assembled with syntactic foam (bottom)

sensors for scientific data collection and navigation, including multiple cameras and sonar sensors for visualization of the environment. Five directional thrusters are used in place of commonly used protruding control planes to facilitate maneuverability and hovering during missions.

The Icefin vehicle is designed in a modular fashion, which allows for ease of operations in the field, future expansion, and mission adaptability. Modularity also provides the Icefin vehicle with the unique ability to face the sensor bay up toward the ice or down toward the ground with a simple 180 degree rotation, possible in the field. Such a capability is not present in any other current state-of-the-art, sub-ice vehicle.

The Icefin vehicle is deployed and recovered vertically through a hole drilled in the ice shelf, which can be hundreds of meters thick, in place of the commonly used methods based from ships in the open ocean. This unique deployment and recovery method provides access to more remote areas and is an important extrapolation for future planetary missions. A small (3.3 mm) diameter optical fiber tether is used for deployment and recovery of the vehicle, as well as for real-time communication and control. This novel lightweight tether

has minimal impact on the dynamics of the vehicle, while providing sufficient bandwidth for communication and control from the surface. Despite its small displacement, the tether also provides a strong mechanical connection (rated at 272 kg) to the surface, which ensures recovery of the vehicle.

3.3.2 Mechanical Design

This section provides an overview of the design of the Icefin vehicle, as well as the challenges encountered due to the unique under-ice mission requirements. The harsh under-ice environment introduces many challenges not encountered by terrestrial and aerial vehicles, or even open-water UUVs. In the specific case of the Icefin vehicle, the mission requirements include a depth rating of 1500 meters, range capability of 3km, temperature rating of -5C, hovering capability, modularity to improve field operations and scientific flexibility, and a small diameter limitation of 26 centimeters. Deployment and retrieval of the vehicle directly through a small diameter hole in the ice shelf introduced many of the unique design challenges. More details on the vehicle design solution are provided in this section and a summary of the under-ice design challenges can be seen in Table 1.

The Icefin vehicle is designed in a modular fashion to retain a human-portable capability for remote deployments. There are six main modules (Figure 15) that fasten together to create a three meter long vehicle (3.5 meters with a drop-weight attached to the front): two directional thruster modules, a front sensor module, a dry electronics module, a wet sensor module and a rear thruster module. Fiberglass-reinforced syntactic foam is placed around the three center modules to provide additional buoyancy. The current configuration of the Icefin vehicle is 26 cm in diameter and weighs 105 kg when fully assembled.

The front module consists of a Kongsberg Light Ring forward-looking camera and a BlueView P900-45 forward-looking sonar sensor. A Neil Brown conductivity-temperature (CT) sensor is also located in the front module to sample the salinity and temperature of undisturbed surrounding sea-water. The front module also contains a Tini Aerospace Frangibolt mechanism [113] that holds and releases a 4.5 kg weight at the front of the

Table 1: Polar Under-ice UUV Design Challenges/Solutions

Under-ice Mission Requirement	**Vehicle Design Solution**
Sub-freezing water temperatures (-5 degrees C)	Low temperature capable batteries Temperature-rated sensors
Hovering capability, non-protruding thrusters or control planes, 6 DOF vehicle control	Four non-protruding directional thrusters
Limited power budget	Ability to switch power to each sensor from the surface control station in software
Remote vehicle control and communication; Deployment and recovery of the vehicle	Kevlar-reinforced single-mode strand of optical fiber for vehicle tether
Small-diameter deployment hole in ice, Difficulty avoiding snags	Constraints on vehicle diameter design No protruding sensors or actuators
1500-meter depth mission requirement	Cylindrical pressure vessel for the main electronics module rated to mission depth, Pressure-rated sensors and actuators
Reliable vehicle navigation and position information, Vehicle control and sensor data management	Greensea Balefire software to create position estimates and record sensor data
Human-portable vehicle	Modular design of vehicle, Size/weight constraints in the vehicle design
Need for both ice and seafloor data collection	Modular sensor bay that can be rotated to face the ice or face the seafloor
Vertical deployment configuration of vehicle, but horizontal mission configuration	Frangibolt drop-weight system at the front of the vehicle

vehicle when it is vertically deployed. This drop-weight shifts the center of mass forward, forcing the vehicle to remain in a vertical position during deployment through the small diameter vertical hole in the ice shelf, which can be only a few centimeters wider than the diameter of the vehicle. The Antarctic ice shelves can be hundreds to thousands of meters thick, and remaining close to a vertical 90 degree pitch prevents the Icefin vehicle from getting stuck during deployment. Upon reaching the required operational depth past the ice-water boundary, the Frangibolt system heats a metal ring around the drop-weight's retaining bolt, which expands and breaks the bolt. With this system, the drop-weight keeping the

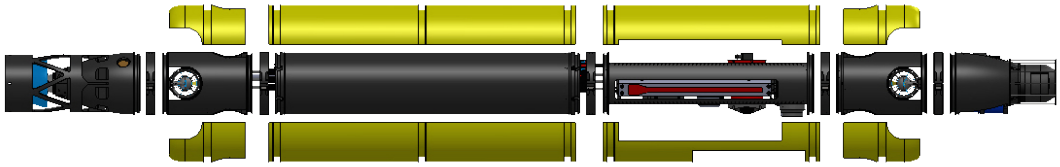


Figure 15: An Icefin vehicle CAD drawing, showing the Icefin disassembled into its individual modules and syntactic foam

vehicle in a vertical pitch orientation is released, and the Icefin vehicle naturally rotates to a horizontal (zero pitch) state, as shown in Figure 16. Internal space limitations on the Icefin vehicle eliminated an internal weight or battery actuation system for pitch control, such as those used in glider AUVs [36]. Use of the Frangibolt mechanism does not require any moving parts (as opposed to a robotic arm) that can freeze at low temperatures, and thus provides a simple and robust system for pitch translation in sub-zero degree seawater. During recovery, the vehicle is first commanded to a depth of tens of meters below the ice-water interface at the deployment hole, and is directed to the hole by pulling in the tether using the winch. Since the vehicle is neutrally buoyant, this provides sufficient time and force for the vehicle to rotate close to a 90 degree pitch as it is pulled by the tether winch system toward the deployment hole at a slow, constant velocity. A hot water drill creates a wide entrance to the deployment hole at the ice-water interface, which also helps to slowly funnel the vehicle to its 90 degree pitch state for safe recovery.

The Icefin vehicle is mission-required to have five controllable degrees-of-freedom (no roll control). Translational surge (x-direction), sway (y-direction), and heave (z-direction) are controlled with respect to the local vehicle frame while rotational pitch (θ) and yaw (ψ) are controlled with respect to the global coordinate frame (Fig. 17). While roll (ϕ) cannot be controlled, rotation around this axis is limited by separation of the vehicle center of buoyancy and center of mass in the vehicle design.

The two directional thruster modules on the Icefin vehicle are located toward the front

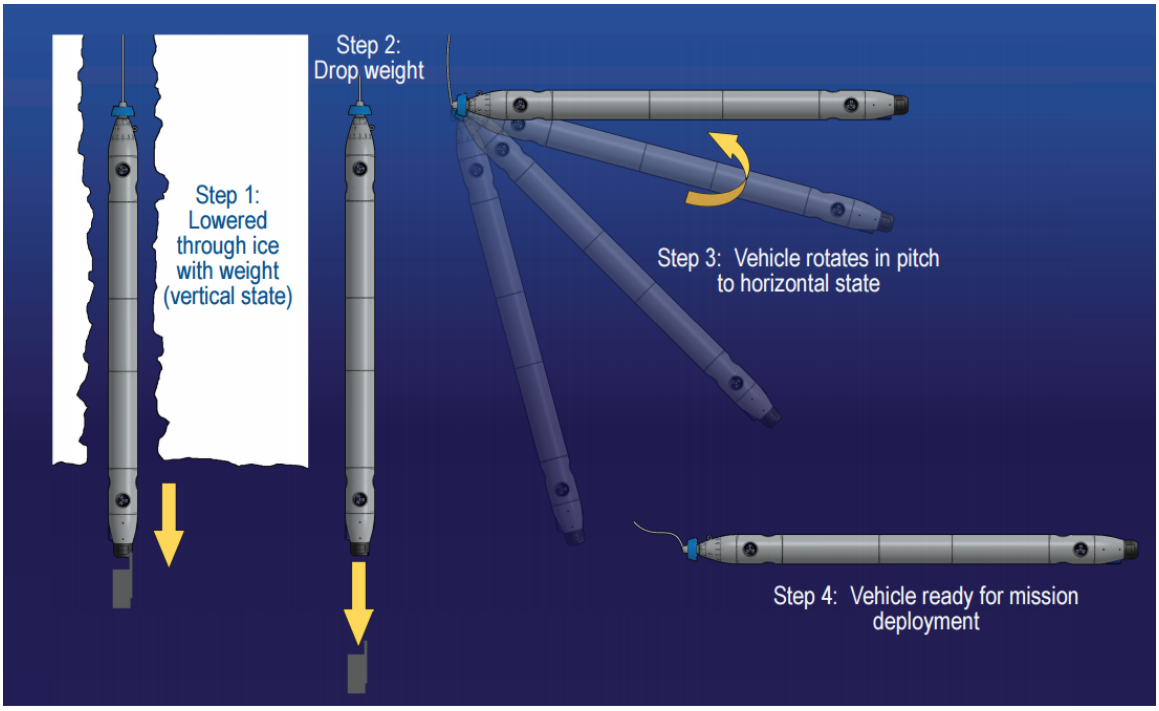


Figure 16: Deployment of the Icefin vehicle.

and rear of the vehicle. These modules are identical, each containing a vertical thruster and a horizontal thruster which can be run in both directions. Each of these four thrusters is mounted into an individual shaft that extends through the diameter of the vehicle, allowing sufficient water flow as the thrusters are engaged. This allows for directional control of the vehicle while retaining its torpedo shape with no external protrusions. Control of five degrees-of-freedom using the five available thrusters is illustrated in Figure 18. Vertical thrusters are used to control pitch and heave; horizontal thrusters for yaw and sway; and the rear thruster for surge.

The downward/upward-facing wet-sensor module contains a suite of both oceanographic and navigational sensors including a high-definition camera, Doppler velocity log and current profiler (ADCP), depth sensor, altimeter, and sidescan sonar. Each of these sensors is contained within independent manufacturer housings rated to the mission-required depth and temperature of the vehicle, and can therefore be mounted in a wet bay module that is

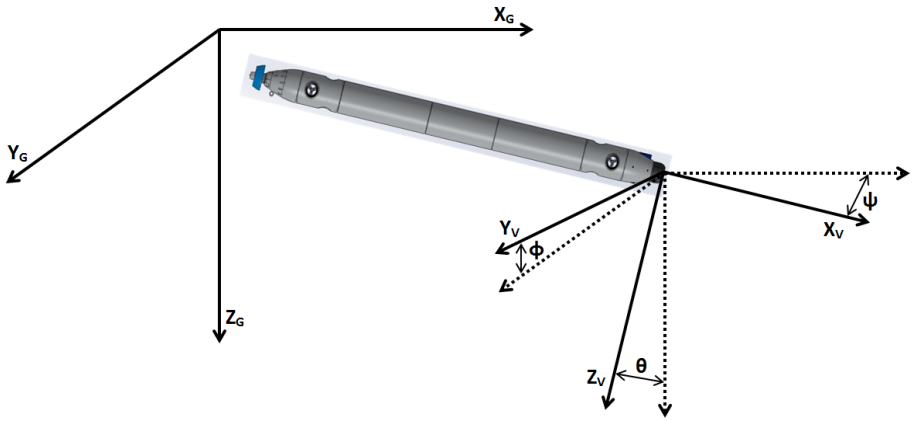
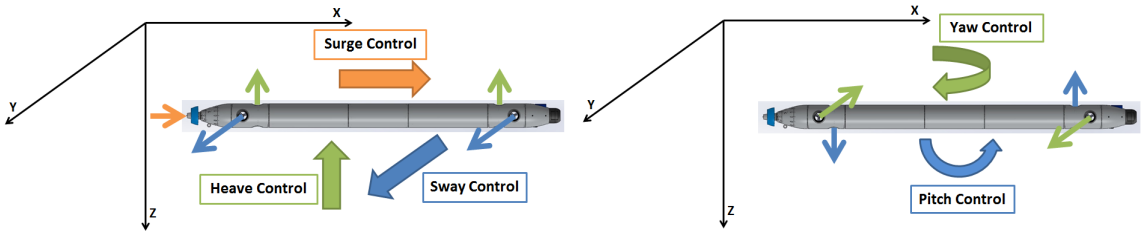


Figure 17: Local and global vehicle coordinate systems.



(a) Surge, sway, and heave

(b) Pitch and yaw

Figure 18: Directional thruster configurations for control of vehicle surge, sway, heave, pitch and yaw

open to the seawater. Leveraging the modularity of the Icefin vehicle, the wet sensor module is designed with the novel capability of operating in two modes: facing up towards the ice or facing down towards the seafloor. This provides the capability to adapt the vehicle to different mission requirements with simple field alterations by rotating the module 180 degrees. This module can also be replaced to satisfy other scientific objectives.

The main electronics module consists of a pressure housing made of 6061 aluminum (hard coat anodized for corrosion resistance) rated to 1500 meters depth, with a bulkhead on each side for external electrical connections. The electronics and batteries required for sensor communication and vehicle control are housed inside this module. The syntactic foam covering this module helps to insulate the sensitive electronics and batteries contained within from the sub-zero external water temperature. Aft of the two directional thruster modules is the rear thruster module. This module consists mainly of a powerful (17.3 kgf)

thruster that provides the surge thrust for the vehicle. The associated maximum forward vehicle speed possible with this thruster was measured to be 2.32 m/s (4.51 knots) on the Icefin vehicle. A protective structure made of 6061 aluminum shields the thruster to prevent propeller damage to the environment, while still allowing sufficient water flow when the thruster is engaged. An overview of the sensors, actuators and electronics in each module can be seen in Figure 19.

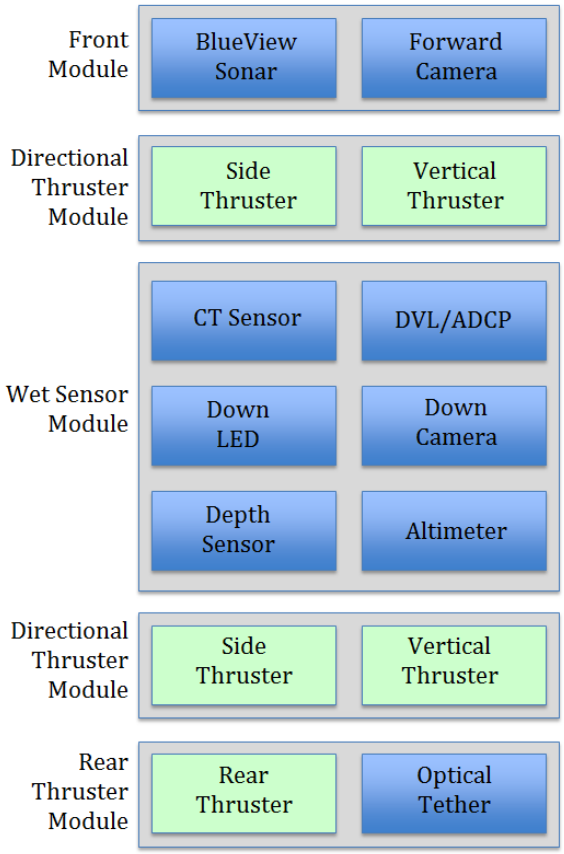


Figure 19: Icefin vehicle modular architecture showing sensors (blue) and actuators (green)

A Kevlar-reinforced optical fiber tether is attached to the rear thruster module for communication and control of the vehicle, as well as for mechanical deployment and recovery. This tether is designed to be only 3.3mm in diameter and lightweight to eliminate almost all adverse effects on the dynamics of the vehicle. The tether is rated to withstand forces up to 272 kg and bend diameters of less than a centimeter without damage. The length of the tether is 3km, providing the vehicle with a radius range of slightly less than that from the

deployment hole, depending on the ice thickness. A secondary steel cable (rated to 907 kg) is attached to the first 30 meters of tether to further mitigate risk of loss and tether damage during deployment and recovery. Both tethers can be seen in Figure 20.

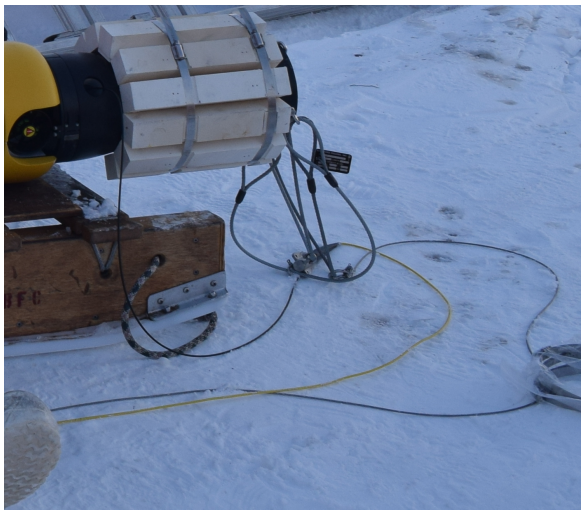


Figure 20: Rear thruster module with the optical fiber tether (yellow) and the steel strength support cable (gray)

Sub-zero temperatures encountered both above and below the ice can result in reduced performance and damage to vehicle electronics and batteries if not considered in the design. All custom mechanical parts, cables, connections, and O-ring seals on the Icefin vehicle were designed to a low temperature rating of -5C to avoid complications. All commercial off-the-shelf (COTS) parts, thrusters and sensors were chosen with ratings to withstand low temperatures down to -5C. Flash memory is used in place of a mechanical hard drive, as this is one of the most common failure points in systems exposed to extreme cold. Precautions are taken during vehicle operation to limit drastic temperature changes inside the electronics pressure housing. Although humidity levels are extremely low in Antarctica, such temperature changes can lead to water condensation, causing shorts and component degradation, and thus a humidity sensor is included on the Icefin vehicle. Extremely low temperatures can increase the internal resistance of lithium-ion batteries, resulting in drastically decreased capacity. To avoid this, heat-radiating components are positioned close to the system's batteries to help keep the battery temperatures in the optimal range above 10

degrees C.

3.3.3 Electrical Design

Most of the vehicle electronics and batteries are housed in the electronics module pressure housing. The main processing element of the vehicle is a small form-factor single-board computer with an Intel Atom processor. This single-board computer has an Ethernet connection and multiple USB connections that are used to communicate with sensors, actuators, and the surface control station. An overview of the electrical design is shown in Figure 21.

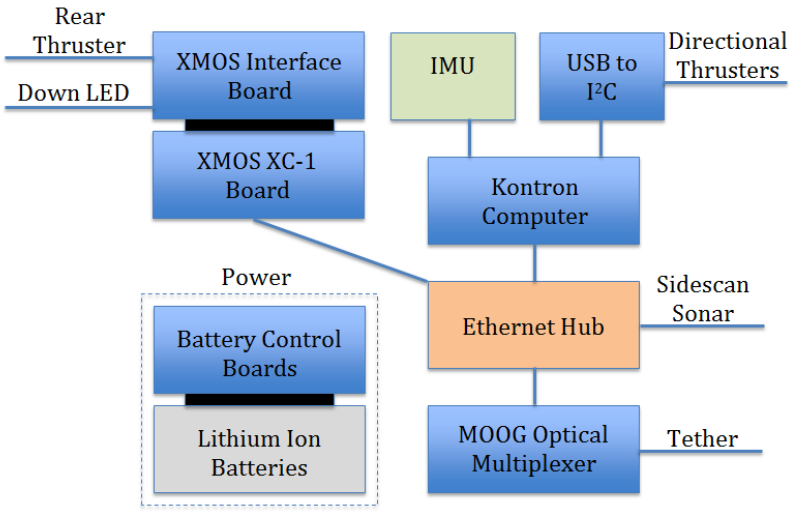


Figure 21: Design of the electronics module in the Icefin vehicle, along with connections between the main components.

For lower-level communication with digital relays, analog sensors, and actuators, the vehicle design incorporates an XMOS XC-2 [114] board, designed for parallel and real-time applications. A custom interface printed circuit board (PCB) was designed and fabricated in-house to provide the necessary translations between the XMOS XC-2 board and the relays, actuators, and sensors on the Icefin vehicle. Most sensors on the vehicle use RS-232, RS-422, or RS-485 for serial communication, which can be translated to TTL levels using the XMOS interface board or translated directly to USB protocol with an FTDI converter. Mechanical relays are used to switch on and off high voltage (24V-48V) power

to each of the vehicle sensors and thrusters independently as a power-saving technique.

Ethernet is used for communication between the vehicle on-board computer, the XMOS board, the BlueView forward-looking sonar, and the tether connection to the surface. As the tether between the vehicle and surface is comprised of a single strand of optical fiber, all data signals are multiplexed using an optical multiplexer, and then demultiplexed at the surface in a similar manner. For this purpose, a Moog 907E mux board is located both inside the electronics module pressure housing as well as at the surface control station. A KVH 1750 fiber optic gyro inertial measurement unit (IMU) is used with a NavQuest 600-micro Doppler velocity log (DVL) to provide inertial navigation data in both the translational and rotational directions. Depth (z-position in vehicle coordinates) is known with a high degree of certainty and zero drift from the Valeport pressure sensor. Distance to the ice cover is obtained with an upward-looking OceanTools altimeter. No compass is currently present onboard due to the polar singularity encountered at such extreme latitudes, proving such magnetic north-seeking methods useless. In this case, a relative coordinate reference system is used based on the initial state (inertial navigation system is zeroed) of the vehicle at the surface prior to deployment beneath the ice. This can be translated to an absolute coordinate system using the GPS coordinates of the deployment hole and initial rotational state of the vehicle at the surface. The navigational sensor outputs (IMU, DVL, depth, and altimeter) are processed directly by the vehicle onboard main computer to enable a real-time navigation solution. The conductivity-temperature sensor data as well as battery status data are not needed for real time processing and are both relayed directly to the surface control station using serial ports present on the Moog optical multiplexer. The four directional thrusters are each controlled independently using the vehicle's main computer via communication over a single FTDI USB to I2C adaptor cable. The parts used in the design of the Icefin vehicle are detailed in Table 2.

To provide power to the electronics, sensors, and thrusters, 15 Inspired Energy NH2054HD31 14.4V 6.8 amp-hour rechargeable lithium-ion batteries (for a total of 1470 watt-hours) are

Table 2: Icefin Vehicle Part List

System Component	Manufacturer	Part
Forward-looking sonar	BlueView	P900-45
Sidescan sonar	Tritech	SeaKing ROV
Inertial Measurement Unit	KVH	1750 IMU
DVL/ADCP	LinkQuest	NavQuest 600u
CT Sensor	Neil Brown	G-CT
Altimeter	OceanTools	MA500D
Depth	Valeport	miniIPS
Downward Camera	DSPL	HD Multi Sea Cam
Forward Camera	Kongsberg	Light Ring Colour Camera
Single Board Computer	Kontron	PITX-SP
Power/Battery Management	Ocean Server	XP-08SR
DC Voltage Regulation	Ocean Server	DC123SR
Batteries	Inspired Energy	NH2054HD31
Ethernet Router	Routerboard	RB951-2n
Rear Thruster	Technadyne	580
Rim Thrusters	SeaBotix	HPDC1502
Tether	Linden	N/A
Optical Multiplexer	Focal Moog	907E Mux, HD-SDI
Low-level Computer	XMOS	XC-2
Sensor Interface Board	Custom PCB	Custom PCB

mounted in the lower portion of the electronics module. Peltier plates were considered in the design of the electronics module to provide additional heat to keep the temperature of the batteries within the optimal operating range above 10 degrees C. However, this additional heat was not required due to the placement of the heat-generating power conversion boards in close proximity to the batteries. The battery temperature during Antarctic field deployment remained above the required 10 degrees C despite the sub-zero exterior conditions; thus battery performance was not impacted. These 15 batteries are divided into three independent banks: an electronics bank providing 3.3V, 5V, 12V, and -12V DC, a thruster bank providing 48V DC to the rear thruster, and a high voltage bank providing 28V and 24V DC for the remaining thrusters and sensors. Separate battery banks isolate the noisy thruster and sensor power from the sensitive electronics. The electronics bank was estimated by the corresponding power controller to have a battery life of approximately eight hours under a nominal computational load in low-temperature, under-ice mission conditions. However, the sensor and thruster banks encounter drastically varying battery life depending on the amount of actuator and sensor engagement. The actual operating duration capability of the vehicle during deployment exceeded eight hours between under-ice mission time and idle time (thrusters not engaged). The main electronics bank is commanded to switch power on and off using an external bulkhead dummy plug. The higher voltage sensor and thruster power banks are designed to be switched on and off through software during the mission due to power-reduction and safety motivations.

3.3.4 Control Architecture and Software Design

The main software component of the Icefin system is comprised of a customized version of Greensea's Balefire software [115]. This software provides a framework for streaming sensor values and vehicle status to a control station, calculating inertial navigation estimates onboard the vehicle in real time, controlling the vehicle actuators in real time, and for transferring high-level control signals from the surface control station to the vehicle during mission operation. While a Linux-based surface computer is used for the main control

station, an additional topside surface computer is also used to concurrently run commercial software (sonar, video, etc.). The control software provides the capability for both real-time human operator control as well as autonomous control of the vehicle. Vehicle state and estimated position information is presented through a graphical interface (GUI) to the operator at the surface control station (Figure 22). The actuator control architecture consists of a low-level front seat driver application running on the vehicle's on-board computer, while a higher-level back seat driver application is run at the surface control station. The front seat driver application controls the actuators directly through low-level hardware commands. The back seat driver application utilizes input from the autopilot or human operator to provide set points for the front seat driver's low-level mixing functionality and proportional-integral-derivative (PID) controllers.

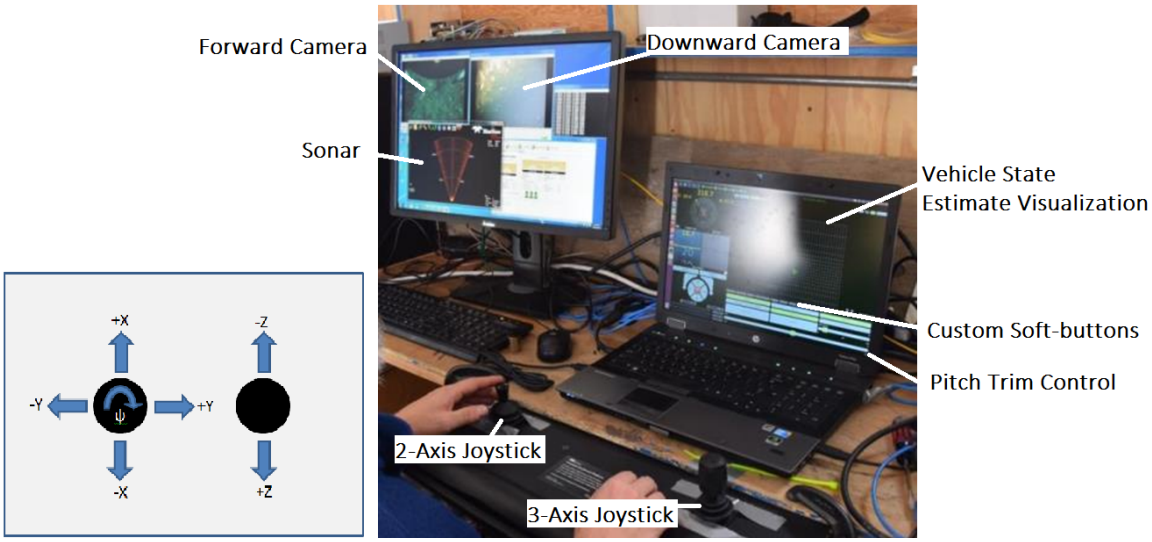


Figure 22: The topside surface control station for the Icefin vehicle (center) with a mapping diagram of joystick control to vehicle dynamics (lower left).

The majority of the processing takes place on a computer at the surface. However, another Balefire node is also running on the vehicle's main onboard computer for real-time processing and input/output control. Communication between these two Balefire nodes takes place through a publish-subscribe architecture built into Balefire itself over an Ethernet (and optical fiber) physical interface. The main processing element of the vehicle is an

Intel Atom-based pITX computer which handles the majority of the high-level control and communication with the surface computer. An XMOS XC-2 [114] computing platform is used to handle the lower-level control of the actuators and contains the ability to multiplex multiple RS232 serial input streams for sensor transmission to the surface computer. A customized Balefire process, named GTRI-Thrust, was developed to provide hardware-specific, low-level control of the vehicle's thrusters. This process translates and mixes the generic commands from the surface control station (joysticks or autopilot) to the required specific low-level thruster hardware commands. Desired effort commands from the joystick user input or autopilot controllers are translated into the range of $+/-100\%$ values and then sent to the GTRI-Thrust application in the form of a five-variable effort data structure. This effort is then mixed, saturated, and translated to the required format for each thruster before being sent to the hardware.

The rear thruster is controlled through the XMOS board, while the directional thrusters are controlled directly from the main vehicle computer through a USB to I2C FTDI interface. Custom XMOS code is run on the XC-2 board to create a Telnet interface, used to communicate with the Balefire software on the main vehicle computer. Other custom XMOS code, to be run in parallel, was also written to directly set the analog control signals (rear thruster speed, LED intensity) as well as the digital control signals (switching of individual sensor and actuator power, leak sensor input, power-board status signal inputs). A custom Balefire process was added to the vehicle computer, named GTRI-Telnet, to provide a communication liaison between the surface control station soft-button GUI commands and the XMOS hardware implementation of these commands. In this process, all commands from the surface control station are converted to the correct Telnet format of "SET" and "GET" variable commands and sent to the XMOS chip over the Ethernet channel. An XMOS-provided Ethernet IP software block was added to the XMOS chip to provide the low-level Ethernet IP interface. An overview of the software system architecture of the Icefin vehicle is shown in Figure 23. It can be seen here that four computing

nodes comprise this system including a Windows commercial software command station and a Linux Balefire command station located at the surface, as well as a Kontron vehicle computer and an XMOS board located onboard the vehicle. Communication between each of these nodes utilizes an Ethernet/Fiber physical interface.

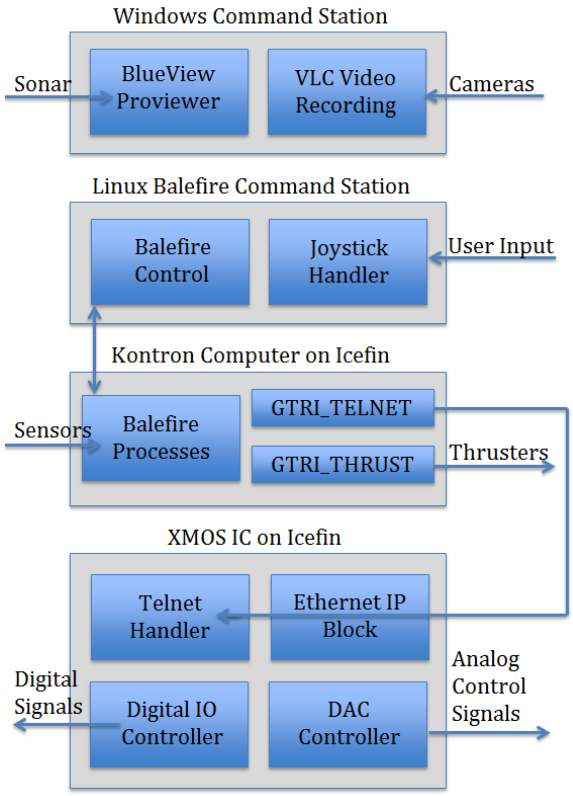


Figure 23: The software/communication architecture diagram for the Icefin vehicle and command and control center. The main software components are listed along with connections between them, broken out into the four main computing components in the system.

For human operation of the vehicle, two joysticks are incorporated into the surface control station for control of the thrusters: one three-axis joystick for control of surge, sway and yaw, and one two-axis joystick for depth control (Figure 22). Video streams from both vehicle cameras and a visual representation of the sonar data is also presented at the control station in real time to aid in operator control.

As recovery of the Icefin vehicle through the thick Antarctic ice shelves requires the use of a tether, the current configuration of the vehicle is designed to remain tethered with

constant communication and control from the surface. However, the autonomy capabilities of the vehicle's control software enable untethered operation if desired, simply requiring the addition of a larger onboard hard drive for data collection. Autonomy capabilities were not utilized during the 2014 Antarctic missions due to the early nature of the deployments, but are fully integrated into the Balefire system and will be used in future deployments.

The forces and torques obtained from each thruster can be modelled as in Equations 1-2 where F is the control force due to the propeller, T is the torque due to the propeller, k is the force coefficient, l is the distance between the thruster and the vehicle's center of gravity and u is the input to the thruster. To correctly mix the independent effects of each of the vehicle's five thrusters, the thruster allocation matrix ($\mathbf{T}$) in Equation 3 is used to obtain the forces and torques on the vehicle (τ). Here, K is the thruster coefficient matrix and u is the input command matrix corresponding to Equation 4.

$$F = k * u \tag{1}$$

$$T = l * F = l * k * u \tag{2}$$

$$\tau = \begin{bmatrix} X \\ Y \\ Z \\ M \\ N \end{bmatrix} = \mathbf{T} * \mathbf{K} * \mathbf{u} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & l_3 & 0 & -l_5 \\ 0 & l_2 & 0 & -l_4 & 0 \end{bmatrix} \begin{bmatrix} k_1 & 0 & 0 & 0 & 0 \\ 0 & k_2 & 0 & 0 & 0 \\ 0 & 0 & k_3 & 0 & 0 \\ 0 & 0 & 0 & k_4 & 0 \\ 0 & 0 & 0 & 0 & k_5 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} RearThruster \\ ForwardLateralThruster \\ ForwardVerticalThruster \\ RearLateralThruster \\ RearVerticalThruster \end{bmatrix} \tag{4}$$

3.3.5 Summary

The Icefin vehicle was designed as a modular unmanned underwater vehicle with an extensive sensor suite and an optical fiber tether for control and communication to meet under-ice deployment challenges while obtaining high science return. A unique combination of human portability and a novel through-ice (small diameter hole) deployment method facilitates deployment of the modular Icefin vehicle out into the field. The vehicle design builds upon a foundation of previous under-ice research to make a unique contribution to the field. The Icefin vehicle is novel among state of the art under-ice vehicles; it contains a scientific and navigational sensor suite, autonomous capabilities, vectored thrusters in place of control planes, and a unique sensor module integration (rotatable by 180 degrees for ice missions). The Icefin vehicle's small size and human portability lowers logistical effort and cost. The Icefin vehicle provides a custom solution to the unique polar and under-ice deployment challenges.

3.4 Sensor Package

3.4.1 Overview

A multibeam forward-looking sonar sensor and an optical camera sensor are both required for the algorithms presented in this dissertation. The methods here are designed to be robust to changes in vehicle platform, requiring only slight changes in sensor to vehicle relative coordinate transformations. The sonar sensor used here with both the VideoRay and Icefin platforms is the BlueView P900-45 acoustic sonar sensor. However the cameras on each vehicle platform are different. The commercial off-the-shelf Videoray Pro IV vehicle contains a built-in camera sensor, while the custom Icefin vehicle design incorporates both an upward and forward camera using commercial components. The sonar and camera sensors used for data collection in the field deployments here are detailed in this section.

3.4.2 Sonar Sensor

Forward-looking multibeam acoustic sonar sensors are commonly found on underwater vehicles for use in object or obstacle detection. These sensors can provide range and bearing to an object in the path of the vehicle using acoustic waves. Forward-looking acoustic sonar sensors provide the viewer with a two-dimensional representation of the area in front of the sensor, highlighting objects with strong acoustic return, as seen in Figure 3. These sonar sensors are limited in the resolution of altitude angle and encounter very high noise levels. The BlueView P900-45 sonar sensor used here provides a 45-degree horizontal field of view with an angular resolution of 256 pixels (beams), and a range resolution of 2.54cm per pixel. Using a tether with Ethernet capabilities, imaging from the sonar sensor can be obtained at the surface in near real time. This sonar sensor was chosen for use in this and future under-ice missions due to its low cost, high resolution and widespread use. Additional specifications for this sonar sensor can be found in [60].

The BlueView P900-45 forward-looking sonar sensor can be mounted to the bottom of the VideoRay Pro IV vehicle and is tethered to the base station computer at the surface. To change the tilt angle of the sonar sensor (to match the angle of the camera), custom running boards were built for the vehicle with mounting holes at the desired angles. This same sonar sensor was also used in the design of the Icefin vehicle. In this case, it is mounted at the nose of the vehicle, facing forward (zero degree tilt), and cannot change angles dynamically. Sensor communication between the Icefin vehicle and the surface is coupled onto the Ethernet to fiber interface built into the Icefin vehicle.

3.4.3 VideoRay Pro IV Camera

The VideoRay Pro IV UUV is a tethered, commercial off-the-shelf vehicle. A camera is built into the vehicle behind a clear plastic dome, and is capable of tilting at angles between 90 and -90 degrees in the altitude direction. The camera has a field of view through the lens and dome of approximately 75 degrees. The camera sensor's effective pixel resolution is 768x494, but the images are digitized to 640x480 and 496x480 pixels by the capture

hardware and software. Full specifications on the camera sensor can be found in [116]. Two lights are located at the front of the vehicle to provide variable magnitude lighting for the camera based on commands from the user interface.

3.4.4 Icefin Upward Camera

The upward-looking camera on the Icefin vehicle is mounted in the sensor bay module of the vehicle. This HD Multi Sea Cam camera is a commercial off-the-shelf sensor made by Deep Sea Power and Light. The field of view available is 85 degrees in the horizontal direction and 50 degrees in the vertical direction. The resolution of the camera is 1920x1080 pixels. More specifications for this sensor can be found in [117]. A SeaLite Sphere LED light made by Deep Sea Power and Light is mounted close to this camera in the sensor bay module of the Icefin vehicle to provide additional lighting if needed.

3.4.5 Icefin Forward Camera

The forward-looking camera on the Icefin vehicle is mounted to the front module of the vehicle in line with the BlueView sonar sensor. This Light Ring Color Camera is a commercial off-the-shelf sensor made by Kongsberg. The field of view available is 36 degrees in the horizontal direction and 27.5 degrees in the vertical direction. The resolution of the sensor is 752x582 pixels, but images are digitized to 640x480 pixels by the capture hardware and software. A ring of LEDs built into the camera housing provide lighting for the camera if needed. More specifications for this sensor can be found in [118].

3.5 Field Trials

During the development of this research, the VideoRay Pro IV and Icefin vehicles were deployed in multiple field tests for data collection. Details of these field deployments, including dates and locations, are presented in this section. The under-ice simulation methods used and corresponding datasets obtained are detailed in Chapter 4.

3.5.1 Open Water Testing, Lake Lanier, GA - August 2013

Lake Lanier (Fig. 24) is a reservoir located north of Atlanta, and provides a natural test location for the vehicle. The VideoRay vehicle was deployed here with the BlueView sonar sensor attached facing straight forward. The visibility in Lake Lanier is extremely low and testing of the VideoRay camera was limited. However, the sonar sensor was successfully used to image the relatively featureless lake bottom (Figure 25a). Sonar (Figure 25b) and camera (Figure 25c) data were also obtained of the square dock buoys clearly visible from both sensors. The sonar data taken of the lake floor was used for development and evaluation of the ego-motion estimation methodology presented here in relatively featureless ground tracking missions, due to the similarity of this environment to under-ice topography.

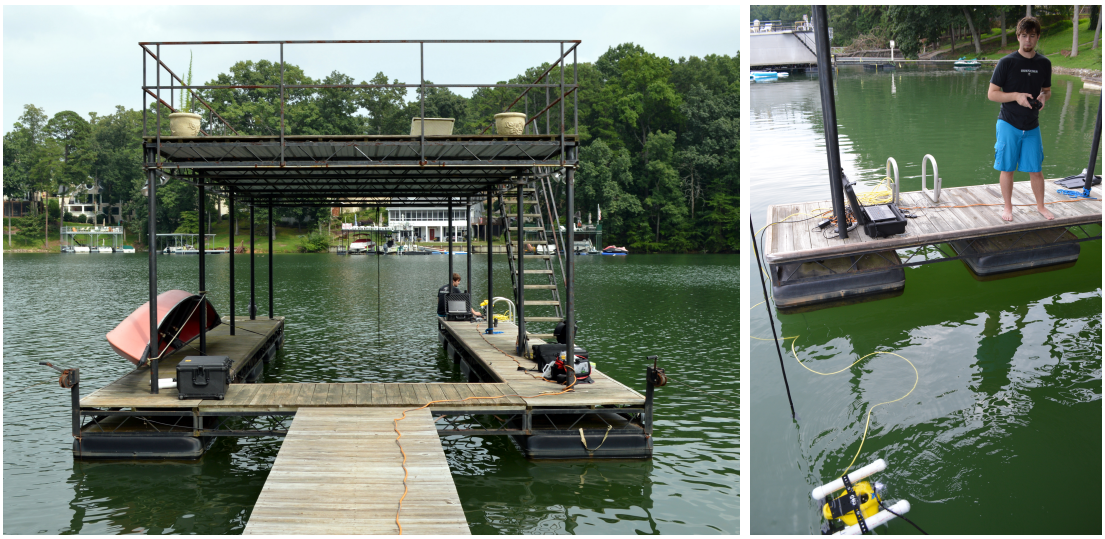
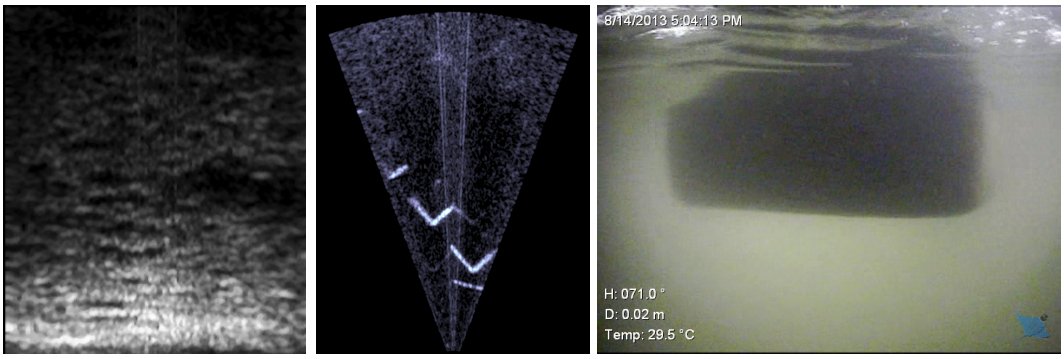


Figure 24: Lake Lanier Test Setup

3.5.2 Outdoor Pool Testing, Atlanta, GA - December 2013

A man-made pool provides a very structured environment where walls and corners are clearly visible in the sonar data. A pool also provides a very high visibility environment to test the vehicle's camera. The VideoRay vehicle with the BlueView sonar was deployed in an outdoor pool to collect concurrent camera and sonar data. This data was used for initial analysis of the synchronicity and cross-calibration of the dual sensor data collection



(a) Sonar - lake bottom (b) Sonar - dock buoys (c) Camera - dock buoy

Figure 25: Sonar and video imagery collected in Lake Lanier

method.

3.5.3 Under-ice Testing, Lake John, CO - January 2014

3.5.3.1 Introduction

In this section is presented some initial under-ice testing and data collection results using the sensor package of interest. This sensing package was deployed with a VideoRay Pro IV unmanned underwater vehicle and includes sonar, video, depth, temperature, and compass instruments. The platform was deployed in an under-ice lake in the Rocky Mountains of Colorado. The experiment and data collection was conducted by deploying the vehicle through a hole cut in the ice. A command center was placed above the ice around the hole for control and deployment of the vehicle (shown in Figure 26). Multiple sensor configurations were tested during data collection to find the most useful configuration for future under-ice deployment. The data obtained from the testing included many deployments of the vehicle through the ice with sonar and image sensors placed at various angles of inclination toward the ice. This enabled the topography of the under-ice structure to be examined using both video and sonar data in various configurations.

3.5.3.2 Approach

Lake John in the Colorado Rocky Mountains was chosen as the testing location due to the lake depth and ice thickness there during the winter testing period. As Lake John is

Figure 26: Lake John under-ice testing location. Tent placed around deployment hole and VideoRay vehicle (in tent) can be seen.

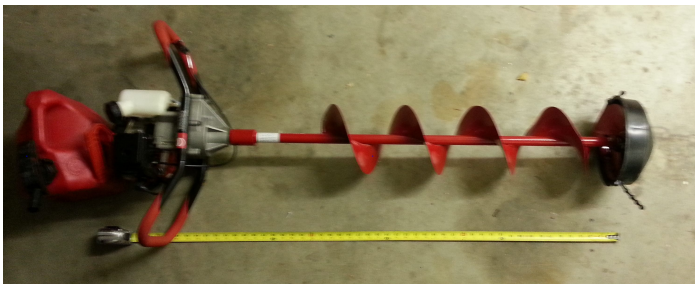


Figure 27: Auger used to drill ice deployment hole and as a feature for the sonar and video sensors

primarily used as a fishing lake, approval was obtained to drill a large hole and perform the desired testing. A combination of ice auger and chainsaw effort was used to cut a rectangular hole in the ice of the required size for vehicle deployment (46cm x 71cm). A tent was placed over the hole to provide protection from the elements for the vehicle, equipment, and personnel. To ensure there was at least a singular feature that could be easily seen in both the sonar and video data, the ice auger (shown in Figure 27) used to drill ice holes was placed through the ice at a sufficient distance from the deployment hole (4.5 meters). This large artificial feature provided a good point of reference when taking data and could be easily seen in both the video and sonar datasets.

The BlueView P900-45 forward-looking sonar sensor with a 45 degree horizontal beam

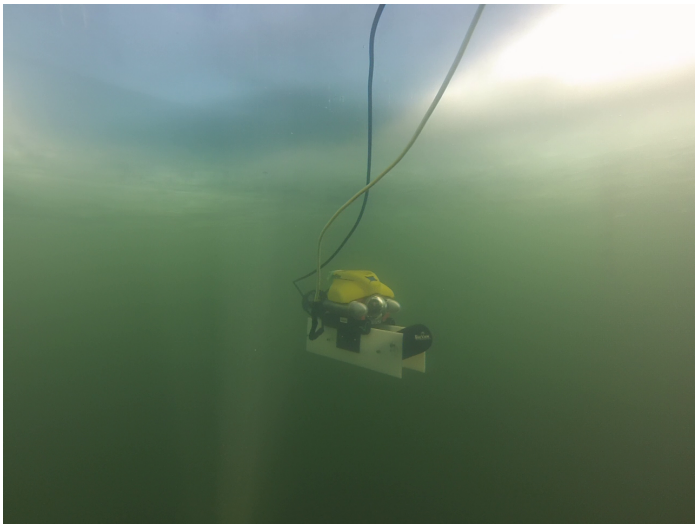


Figure 28: VideoRay vehicle and sonar sensor deployed under the ice in Lake John

width was mechanically connected to the Videoray Pro IV UUV. The assembled vehicle was deployed and recovered multiple times by hand through the hole in the ice. Once the vehicle was placed in the hole, the slightly negative buoyancy of the vehicle carried it slowly down until it was commanded to a certain depth by the operator. The vertical thruster on the vehicle was used to control the depth of the vehicle and prevented it from sinking to the bottom. Control of the vehicle was accomplished using both control software and manual joystick control. Two different forms of control software were used during testing. Windows software packages provided by VideoRay and BlueView as well as a Linux-based Robot Operating System software package were used independently for control and data collection with the vehicle, camera, and sonar sensor. An image of the Videoray vehicle deployed under the ice in Lake John is shown in Figure 28.

Many different independent runs (or missions) were completed during vehicle deployment, as shown in Table 3. In some cases, the vehicle was commanded straight forward and then straight back to end at the start location. In other runs, the vehicle was turned in a 360 degree circle with minimal translational movement. Many runs were also completed with the vehicle starting close to the hole, being commanded in a random path, and returning

Table 3: Lake John navigation scenarios

Navigation Scenario	**Runs**
Deployment through ice	6
360 degree circle	7
Forward and back	10
Examine auger	6
Random path	14

to the start location. In all of these cases, the data recorded for each run was obtained independently from the others. The vehicle was commanded to remain at a constant desired depth for most of the testing. Ground truth vehicle position is always difficult to obtain in underwater environments due to the lack of a GPS system. These run-types were designed to provide as much consistency and ground truth as possible in the data sets. The vehicle was visible from the deployment hole, and data-recording was started and stopped with the vehicle as close to straight under the hole as possible to aid in ground truth analysis of the data. The artificial ice auger feature was used as a known reference point to aid in post-deployment analysis of the data. Many runs contain multiple views of this feature.

Heading and depth readings from the Videoray were utilized to provide ground truth information. Heading from a compass sensor and depth from a pressure sensor provided consistent and fairly accurate readings. Pressure readings were calibrated to translate effectively to depth readings using known lake depth information. Compass heading readings were used to provide yaw truth for each set of associated concurrent sonar pings and camera images. Such data can be used for analysis of the runs in which the vehicle was commanded to turn in a 360 degree yaw circle. Inertial measurement data, often used for dead-reckoning inertial navigation, was also recorded by the vehicle, but requires careful calibration and scaling if used for vehicle positioning.

The camera and sonar sensors were pointed at varying angles up at the ice throughout the testing runs. One such configuration was straight up while the other two were set to

graze the ice at 8 degree and 24 degree angles. Each configuration was used for multiple runs to determine the optimal configuration for the sensors in similar future missions. A single run was taken looking down at the bottom of the lake by manually holding the vehicle and pointing the sensors in the desired direction. Once all required concurrent sonar and video data was collected, the sonar sensor was removed. Multiple runs were undertaken in which the vehicle was commanded in a random path to record only video. The vehicle was much more positively buoyant (remained just below ice level) in this case and had a much larger radius to explore without the limitation of the second sonar tether. All data obtained during the vehicle runs was recorded for future offline analysis.

3.5.3.3 Experimental setup

The vehicle, sonar and command and control systems were tested in a test tank prior to field deployment. Upon validation of all software and hardware involved, the deployment setup was transferred to the field testing location. Due to the extreme temperatures and damp conditions on the ice, a tent was used as a control base-station around the deployment hole during field testing. Equipment cases were used to keep the equipment in the base station dry and off of the ice. Power to the equipment was provided using a small suitcase-sized generator placed outside of the tent. The deployment hole cut into the ice was 45cm by 71cm and cut through approximately 51cm of ice thickness. An image of the test setup is shown in Figure 29. The artificial ice auger feature was placed at a heading of 45 degrees from north and distance of 4.5m from the deployment hole. The depth of the lake at the deployment hole was around 6.1m at the time of deployment.

The Videoray Pro IV vehicle was assembled with additional buoyancy to compensate for the weight of the sonar sensor as well as the balancing weight. Extra-long plastic runners were attached to the bottom of the vehicle to allow attachment of the sonar sensor at the desired angles of 90 degrees (straight up), 30 degrees, and 10 degrees (almost straight forward). The sensors mounted in the 90 degree configuration are shown in Figure 30. A diving weight was attached at the back of the vehicle between these runners to balance the

Figure 29: Lake John under-ice experimental testing setup and base station

pitch effects of the sonar sensor attached at the front. The sonar sensor could be mounted at any of the three angles by moving between three pre-drilled mounting points, and the vehicle camera was mechanically tilted to the desired angle to match the angle of the mounted sonar.

During deployment, one operator was in control of the deployment and retrieval of the vehicle through the hole, as well as management of the tether. A second operator controlled the vehicle and data collection efforts using the laptop control station. Two cables were used to tether the vehicle: one for the control, status and video from the vehicle and another for sonar sensor communication. The sonar tether limited the vehicle range to a radius of approximately 6.1m from the deployment hole. The tethers were secured to a stake drilled in the ice and managed by an operator to reduce strain on the control equipment and prevent control equipment from accidentally being dragged into the water.

Two sets of control and data collection software were used for redundancy and robustness. Off-the-shelf software from Videoray and BlueView was used on a Windows operating system. A Robot Operating System (ROS) package developed for control of the Videoray vehicle and BlueView sonar was used on an Ubuntu Linux operating system as well. The ROS package records a timestamp and single video image for each sonar ping

Figure 30: Videoray sonar and video sensors in 90 degree straight-up configuration

recorded, which provides datasets for the required algorithm validation and analysis. This ROS package also provides vehicle control, vehicle status, and navigational sensor data recording capabilities. The BlueView off-the-shelf software used with Windows was also able to record concurrent sonar and video streams. A GoPro camera was used to take video of the vehicle operating under the ice.

3.5.3.4 Results

During this Lake John deployment, over 40 runs of data were collected across the two sets of system software. The Linux-based ROS package seemed much more responsive for vehicle control, and provided the ability to record vehicle status information (compass heading, depth, IMU, control inputs) concurrently with the sonar and video data for post-analysis. The frame rates of the sonar and video streams differed when recorded using the BlueView software, and required manual post-synchronization using the timestamps in each dataset.

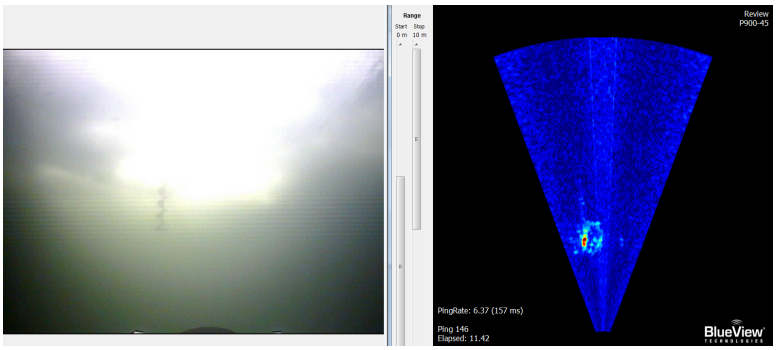
The artificial ice auger feature was clearly visible in both the sonar and video image data, but was only clear in the video data when the vehicle was sufficiently close to the

object, due to the poor visibility in the lake. Figures 31a-31c show concurrent sonar and video image frames of the auger feature and ice cover. The 30 degree sensor angle seemed to provide the best configuration for highlighting features in the ice topography. The 90 degree angle (straight up) configuration (Figure 31c) provided a cutaway view of the ice topography directly above the vehicle, but the small topographical features were lost amidst stronger returns with this straight-on angle. The 10 degree angle (Figure 31a) seemed to provide too small of a grazing angle, and little acoustic return was received by the sonar sensor due to the lack of strong topographical features in the ice. By tilting the sonar at a 30 degree angle from the horizontal up at the ice (Figure 31b), small features were discernible, even in the mostly flat and smooth first-year ice. At this angle, multiple small features can be seen to move upon panning of the sonar sensor, and a human operator can detect a slight uniform shift of many faint features in the sonar data as vehicle motion. Table 4 presents a summary of the analytical results from varying the sensor tilt (altitude) angle. One observation determined from this under-ice sonar data was an ambiguous visual result for the ice location (with the 90 degree configuration) due to propagation of the acoustic sonar waves through the ice (instead of providing a strong return at the ice-water boundary).

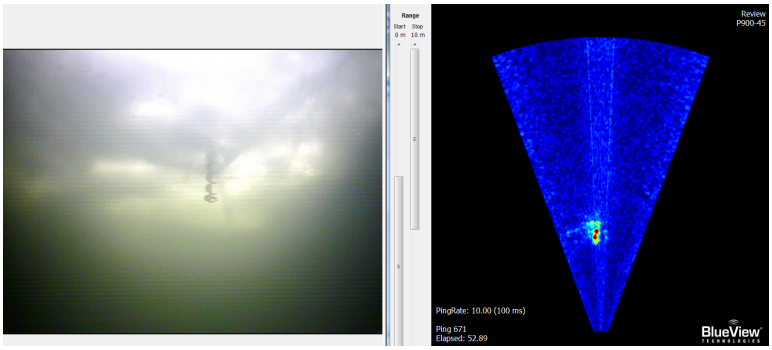
Table 4: VideoRay sensor angles during Lake John deployment

Angle	**Benefits**	**Drawbacks**
90 degrees	Good ice topography cutaway information.	Difficult to distinguish smaller features.
30 degrees	Good for distinguishing smaller features. Cannot determine ice location.	Smaller coverage area.
10 degrees	Can distinguish some smaller features. Large coverage area.	Less resolution for small ice features.

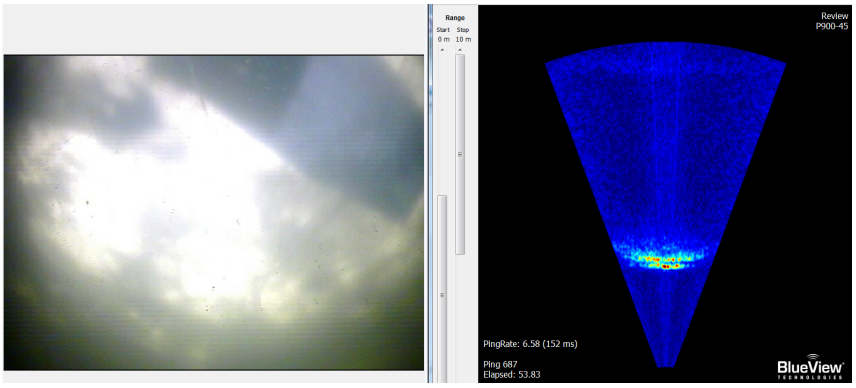
Some features in the ice were visible to the camera including cracks in the ice, snow on top of the ice, and air bubbles caught in the ice, shown in Figure 32. However, many of these features are very small and could not be distinguished from more than a few meters away. The slight topography of the ice could be determined visually when viewed at



(a) 10 degrees



(b) 30 degrees



(c) 90 degrees

Figure 31: Concurrent video (left) and sonar (right) image frames of the auger feature and the surrounding ice taken at a various angles up at the ice.

Figure 32: Close video imaging view of water-ice boundary with visible snow and ice-crack features.

extreme grazing angles with the vehicle very close to the ice. Sunlight projecting through the ice and water reduced visibility of the ice-water boundary in many cases. Sunlight also resulted in saturation of some of the camera data. The ice topography would be better viewed in the absence of sunlight (such as with thicker polar ice), using the cameras on the VideoRay as a sole light source. The ice was mostly clear and transparent, which made it even more difficult to find distinguishable features in the video data. However, this transparency provided the ability to see through the ice layer to the snow pack on top, which was determined to have more features for the camera to pick up.

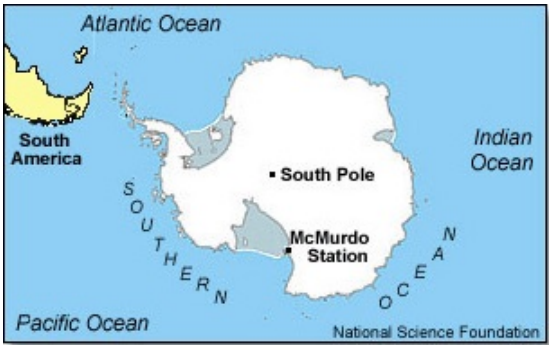
Many lessons were learned during this under-ice field deployment. It was determined that the sonar and video streams collected using the BlueView software have different offsets and frame rates, which must be taken into account during sensor fusion post-analysis. The light saturation, due to the sun shining through the ice, should be avoided if possible to obtain more feature-rich video data of the under-ice topography. A 30 degree tilt from the horizontal toward the ice was determined to provide the best sonar views of the features in the ice topography. Lastly, the low-temperature ratings of the vehicle platform and sonar sensor proved reliable, and no equipment problems or failures were encountered despite the harsh deployment environment.

3.5.3.5 Summary

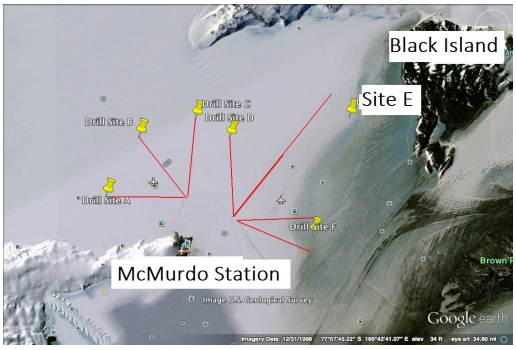
This under-ice field deployment in Lake John, Colorado using the VideoRay Pro IV vehicle and BlueView P900-45 forward-looking sonar provided a wealth of data for use in algorithm development and analysis. Video and sonar data was collected for over 40 runs of the vehicle using multiple software suites for redundancy. The sensors were tilted at three different angles toward the ice to determine the optimal configuration for viewing the under-ice topography. The qualitative results from this sensor angle evaluation was used for development of the Icefin vehicle. The data collected and lessons learned here facilitated development of the Icefin custom polar under-ice vehicle, as well as development of the algorithms presented in this dissertation.

3.5.4 Under-ice Testing, McMurdo, Antarctica - November 2014

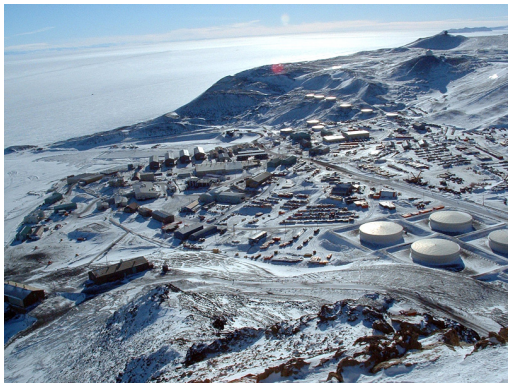
3.5.4.1 Introduction



(a) Map of Antarctica (c.o. NSF)



(b) McMurdo and deployment site (c.o. Google)



(c) McMurdo Station

Figure 33: Maps and images of the deployment area in Antarctica. McMurdo station and the relative deployment location (SIMPLE site E) can be seen.

Figure 34: 2014 Antarctica SIMPLE field deployment team.

The Icefin vehicle's maiden sub-ice voyage occurred as part of the 2014 Austral summer deployment of the NASA-funded project Sub-Ice Marine and Planetary-analog Ecosystems (SIMPLE) to McMurdo station, Antarctica (Figure 33). The goal of the deployment was to support a NASA scientific expedition, while providing an assessment of this new under-ice vehicle and deployment strategy to help lay the initial groundwork for both terrestrial and planetary exploration. The 2014 deployment team can be seen in Figure 34. Final Icefin integration took place in Austral spring, 2014 and the vehicle's maiden voyage beneath the ice took place in November, 2014. The Icefin vehicle was first deployed from a dive jetty on the sea ice, close to McMurdo station, for system testing. Following validation of the vehicle's control and communication systems, the vehicle was deployed through the ice at SIMPLE site E (Figure 33b) on the McMurdo Ice Shelf. Video, sonar and other scientific data was collected for analysis by scientists and engineers on the team. The Icefin vehicle returned data of the previously unexplored under-ice and seafloor environment adjacent to Black Island in McMurdo Sound. The field deployment to Antarctica ended in December, 2014. This section provides an overview of the 2014 Icefin deployment field trials in Antarctica and lessons learned, with extrapolations to future polar and planetary exploration missions.

3.5.4.2 Testing and Validation

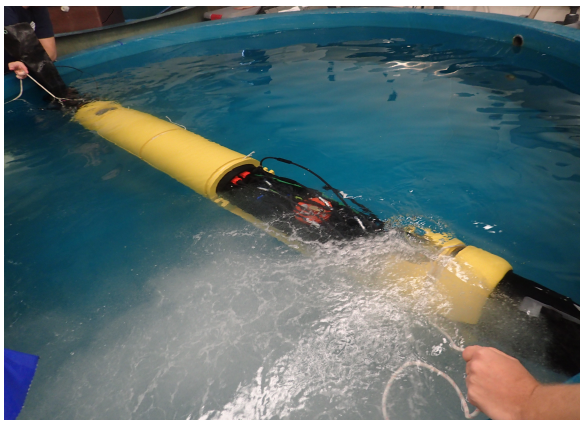


Figure 35: Icefin vehicle deployed by hand in the test tank for actuator testing.

Prior to deployment in an under-ice environment, lab and tank-based testing (Figure 35) was required to reduce the probability of operational failures with the Icefin vehicle. The Icefin vehicle was transported over 15,000 km by plane to McMurdo station in Antarctica. A small shared laboratory space was provided at the bottom of McMurdo Station's Crary science lab for final Icefin vehicle integration and testing (Figure 35). The control software present on the vehicle computer, XMOS, and surface control station computer were tested, debugged and validated. A ten-meter diameter test tank was provided in the lab, and the vehicle was deployed by hand into this tank multiple times for in-water testing. Sensors, actuators, communication and control over the optical fiber tether were tested in this tank along with mechanical aspects such as buoyancy and center of mass. Additional syntactic foam was added the fore and aft of the vehicle during tank testing to compensate for unanticipated additional weight in the electronics module, and to bring the vehicle to the desired slightly negative overall buoyancy with zero pitch. The thrusters were validated by sending gradually increasing speed commands to the thrusters in the sway, surge, heave, and yaw directions, and verifying the desired dynamic response of the vehicle.

(a) Dive jetty and Pisten Bully (b) Icefin vehicle deployed in jetty

Figure 36: The dive jetty near McMurdo Station, Antarctica used for pre-field testing of the Icefin vehicle and the location of the vehicle's initial under-ice deployment. The dive jetty shack is shown on the right and a Pisten Bully used for transportation of the vehicle equipment is shown to the left in (a). The Icefin vehicle being deployed through the dive-jetty ice-hole in (b).

3.5.4.3 Field Deployment

Ice cover is difficult to recreate in testing chambers, thus extensive under-ice testing in a relatively controlled environment was required by the science and engineering teams prior to certifying the vehicle ready for science operations through the ice shelf. The Icefin vehicle's maiden deployment beneath the ice took place in November, 2014 at McMurdo station's sea-ice jetty (Figure 36), commonly used by human divers. The dive jetty as well as the Pisten Bully vehicle, used for vehicle field transport, can be seen in Figure 36a. This location was utilized due to its close proximity to McMurdo, existing hut infrastructure setup, and large pre-drilled ice-hole. The goal of this initial sub-ice deployment was to fully validate the vehicle system and deployment setup in a more controlled environment prior to full deep-field deployment. The dive jetty provided a much larger diameter hole and thinner ice than would be possible with ice shelf deployment. These aspects eased Icefin operations as the vehicle could be deployed by hand and recovered quickly if necessary, with a greatly reduced probability of loss with respect to ice shelf operations. Deployment in the dive jetty was also beneficial to evaluate the vehicle drop-weight system, and provided the unique ability to view the vehicle during deployment through direct human observation and with a standalone external camera in real time. Such observation was used to directly evaluate

the vehicle's behavior and dynamics during testing. An image of the Icefin vehicle under the sea ice is shown in Figure 37. The vehicle was deployed in the dive jetty four times for a total of approximately five hours. During the final jetty deployment, the vehicle was fully operational and performed as desired for approximately 45 minutes of human operator joystick control before testing was considered to be complete. Successful deployment of the Icefin vehicle in the McMurdo sea-ice dive jetty proved the vehicle ready for mission deployment through the deep-field ice shelves.

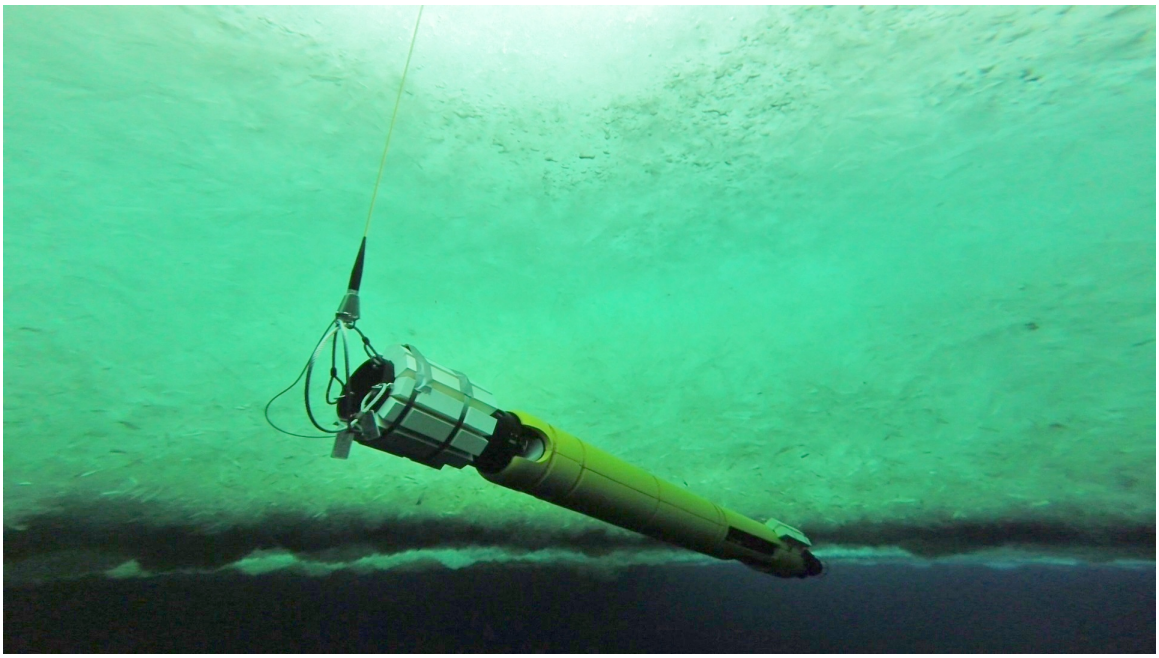


Figure 37: The Icefin vehicle beneath the sea-ice, deployed through the dive jetty ice-hole near McMurdo Station, Antarctica.

After the initial sub-ice deployment, the Icefin vehicle was transported from McMurdo station out to SIMPLE Site E on the McMurdo Ice Shelf (30 km away), as seen in Figure 33. The disassembled Icefin modules and support equipment were loaded into sleds and tracked vehicles (Pisten Bullies) and driven four hours out to the deployment location on the ice shelf. A team of hot water ice-drillers had spent the prior day to the vehicle's arrival drilling a 38 cm diameter hole for the vehicle deployment, extending vertically through the 14.6 meters of ice thickness present at the Site E location. A 4.5 meter tall tripod was set up

and positioned above the hole, while the vehicle tether and winch were set up two meters away for use during deployment. The Icefin surface control station was set up inside a tent approximately five meters from the deployment hole. This deep-field camp setup can be seen in Figure 38. Upon connection of the Icefin vehicle modules, the tether was attached to the rear module and the vehicle was raised to a vertical pitch state above the deployment hole using the tether, tripod and winch.

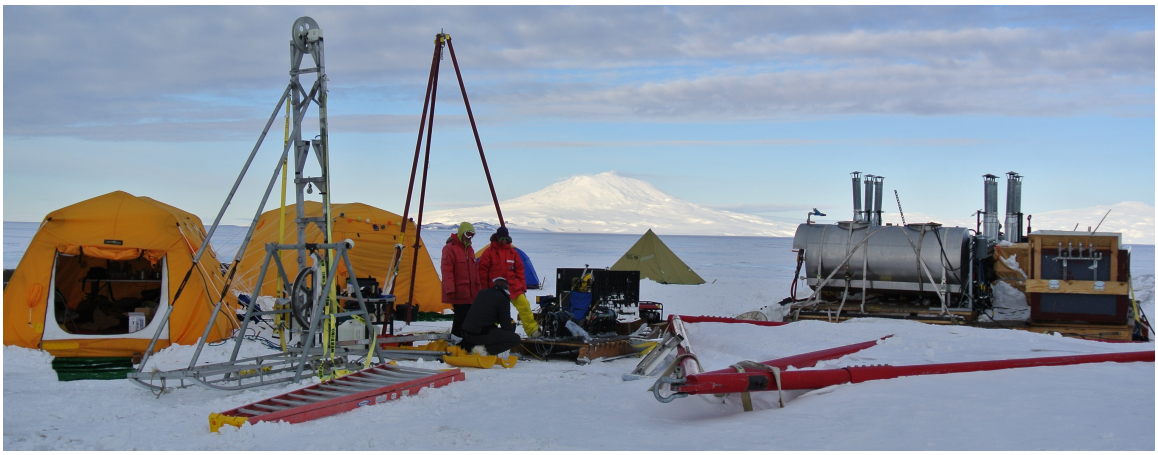


Figure 38: The field camp for the Icefin deep-field deployment through the McMurdo Ice Shelf. The hot water drill used to drill holes through the ice shelf is can be seen on the right. The vehicle tripod is shown in the middle with the surface control station tent on the left.

The day following departure from McMurdo Station for the deep field, the Icefin vehicle was successfully deployed at SIMPLE's site E location near Black Island. After setup and system initialization, the vehicle was lowered vertically by the tether through the ice shelf deployment hole using the winch. Upon reaching the bottom of the hole, the ice-water interface was examined (with vehicle still in vertical orientation). During this time, the platelet ice (top of Figure 39) found at the ice-water boundary was examined through real-time camera streams. The vehicle was then commanded directly down to the seafloor, while collecting water column profile measurements. At approximately 480 meters depth (less than five meters from the seafloor), the drop weight at the fore of the vehicle was released using the Frangibolt system, resulting in vehicle rotation to a horizontal pitch state, as

desired. Currently, the drop-weight is released upon receipt of a human operator command from the surface. However, this capability is integrated into the autonomy infrastructure of the control software, and can be completed autonomously at a preset desired depth in future missions. The main goal of the deep-field Antarctic deployment described herein was to obtain characteristic data of the water column extending from the surface to the seafloor, as well as images, video and sonar imagery of the ice-water boundary and the sub-ice seafloor (Figure 39). Upon reaching the seafloor and release of the drop-weight, the vehicle was commanded in a random exploratory pattern (mostly yaw and surge) by the human operator in search of interesting features and life forms (bottom of Figure 39). A summary of testing and deployment results is presented in Table 5.

Table 5: Icefin field testing/deployment results and lessons learned

Power-up and quick validation of sensor and actuator functionality	Current draw, communication interfacing, sensor and actuator behavior determined Changed directional thrusters from RS422 to SPI interface.
Pressure testing of pressure bottle and bulkheads.	Pressure rating validated to 1500 meters.
Testing of main electronics components and computing elements	Software bugs were fixed and minor electrical and heating problems were resolved.
Light dB loss test for 4000 meter optical fiber tether to validate optical continuity.	No breaks detected, despite being wound around the winch multiple times. Very low dB loss through fiber.
Tank test to validate buoyancy and leak resistance.	Added more buoyancy to make the vehicle only slightly negative. Removed sidescan sonar.
Communications test between surface control station and the vehicle through the multiplexer boards and a patch optical fiber cable.	Multiplexer boards were very easy to use and robust. Communications and video streams established successfully over single fiber.
Tank test to test buoyancy, communications between surface and vehicle, actuator and light functionality.	Problems with directional thrusters determined to be shorting of the I2C to USB chip. Protection circuitry added to both the directional and the rear thruster control. External power switch and battery charging solution added to bulkhead.
Full software, electronics, and sensor bench testing.	Small changes made to complete the full required system functionality.
Full tank testing of electronics, sensors, and actuators as well as surface/vehicle communications.	Validated system readiness for under-ice deployment.
Under-ice deployment of the vehicle in the McMurdo dive jetty. Weight-release system validation. Full system validation.	Further debug of directional thruster issue and changed lubricant for wet-mate connector. Actuator tuning, data collection.
Field deployment to McMurdo Ice Shelf.	Data collection mission
Under-ice deployment of vehicle in the McMurdo dive jetty.	Further data collection. Intermittent electronics system problem debugged.

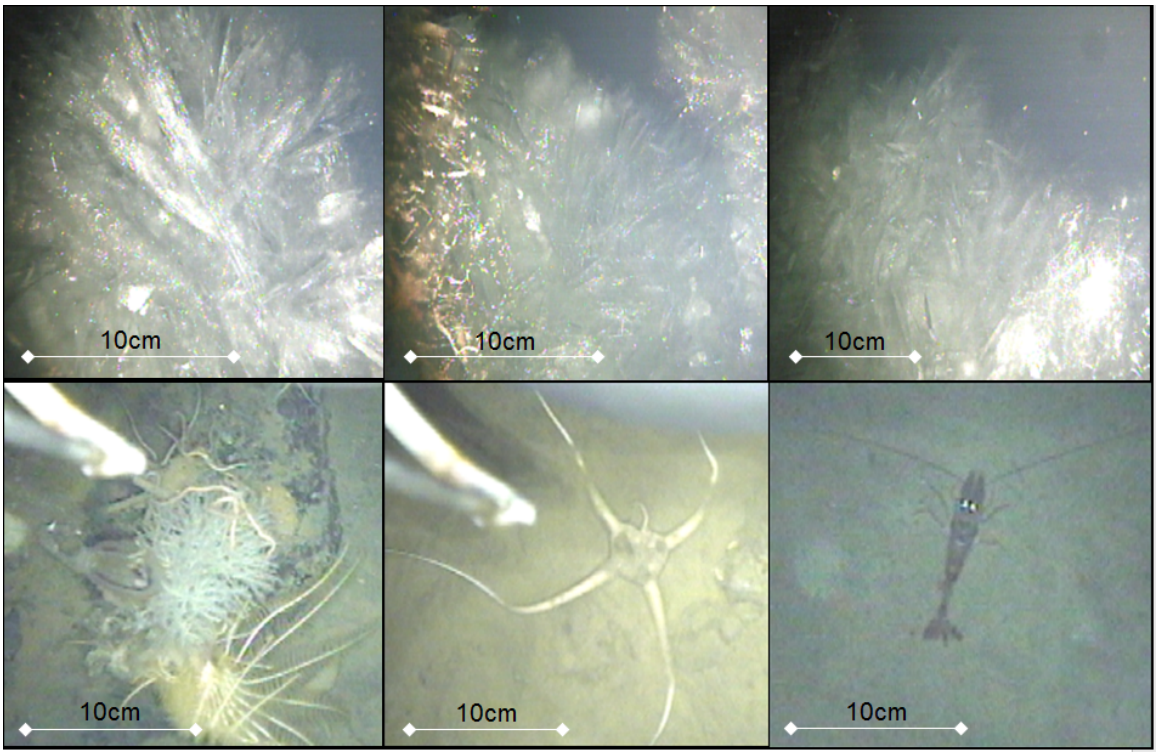


Figure 39: Images captured by the Icefin vehicle during its Antarctic deep field deployment. The top row of images show the crystal-like platelet ice found at the ice-water boundary under the McMurdo Ice Shelf formed by slow freezing of the seawater. The bottom row of images show a variety of life found on the seafloor 482 meters below the ice, including anemones, brittle stars, and crustaceans. Scale is estimated from sonar range and camera field of view.

Sonar and video imagery, as well as water column salinity, temperature, altimetry and depth data, was recorded during deployment for post-analysis. Video imagery obtained by the Icefin vehicle of the seafloor includes views of sea stars, brittle stars, crustaceans, anemones, and sponges (Figure 39). The ocean floor at this location was previously unexplored prior to this Icefin voyage. Observations and imagery of life that is able to sustain itself in such harsh, sub-ice environments is of interest to biologists. A summary of the datasets collected during field deployment is presented in Table 6.

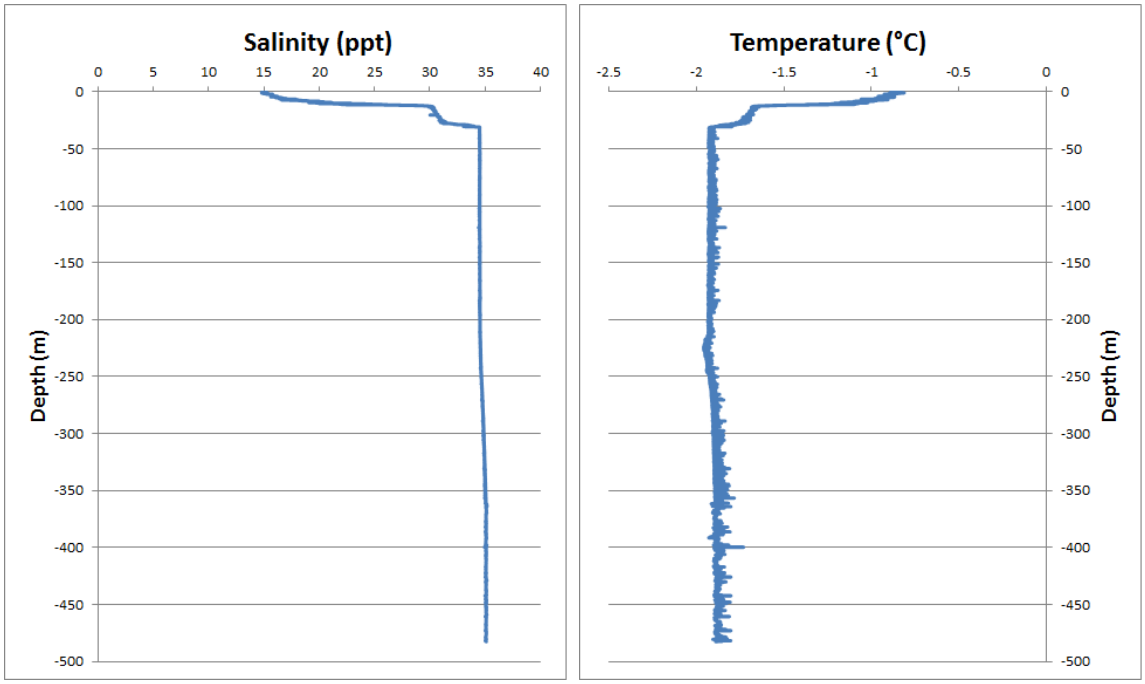
Ocean currents and water properties affect how the ice shelf and ocean interact, including whether ice is melted or accreted and what kind of ice forms. Salinity and temperature versus depth plots of the water column from initial analysis of the CT sensor data, collected

Table 6: Datasets recorded during deployment

Data Type	Location	Total Time	Number of Files
External Camera	Dive Jetty	0:54:19	8
Down-looking Camera	Dive Jetty	7:38:54	10
Down-looking Camera	Ice Shelf (Field)	1:00:34	37
Front-looking Camera	Dive Jetty	7:08:06	10
Front-looking Camera	Ice Shelf (Field)	1:07:42	42
Sonar	Dive Jetty	2:41:10	20
Sonar	Ice Shelf (Field)	0:53:44	5
Telemetry	Dive Jetty	5:18:06	7
Telemetry	Ice Shelf (Field)	0:00:00 (Corrupted)	1

during this deployment, is shown in Figure 40. In this case, the distinct water profile within the ice-hole (first 15 meters) is evident from lower salinity and a higher temperature seen in these plots. The water column below the ice shelf is seen to be well-mixed to the seafloor, consistent with the formation of platelet ice, seen in the top row of Figure 39.

After a mission consisting of two hours and 45 minutes of data collection beneath the ice, the winch was used to recover the Icefin vehicle through retraction of the tether. The vehicle rotated to a -90 degree pitch upon intersection with the deployment hole, as designed, and was raised at a constant velocity by the tether through the deployment hole using the winch. Upon completion of the deep field deployment, the Icefin vehicle was broken down and returned to McMurdo station. The vehicle was subsequently returned to the dive jetty for two more sub-ice missions of sonar and video imagery collection. Throughout these jetty missions the vehicle was commanded over singular decoupled degrees-of-freedom (surge, sway, and yaw), as well as over random exploratory paths, for a total duration of four hours and 22 minutes. Sonar and video datasets were collected from beneath the sea-ice for use in algorithm development and evaluation. An example of the sonar and video



(a) Salinity versus depth

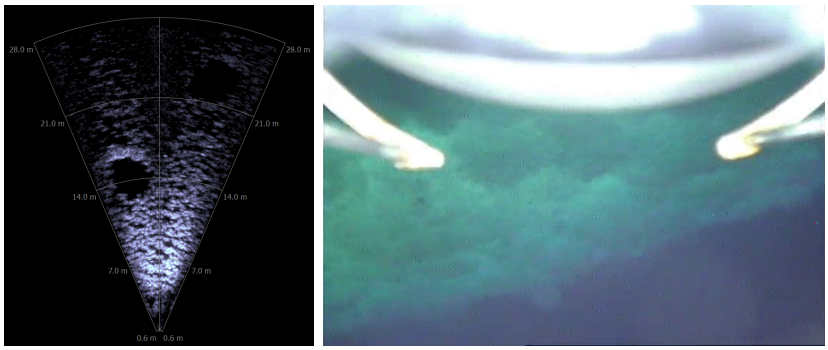
(b) Temperature versus depth

Figure 40: Salinity (left) and temperature (right) from initial data analysis of a vertical profile taken during the ice shelf deployment, showing the water column profile with depth.

imagery from the sea-ice deployment can be seen in Figure 41. In mid-December of 2014, the vehicle and support equipment was broken down and shipped back to Atlanta, GA.

3.5.4.4 Lessons learned from Antarctic deployment

Design and deployment of the Icefin vehicle in the Antarctic presented unique challenges. Many lessons were learned from this work that can be passed on to future sub-ice missions. One lesson learned during tank testing was the need for an external vehicle power-switching and battery-charging solution to reduce wasted energy expenditure during deployment lead up time and to facilitate faster mission turnaround with limited vehicle tear-down in-between. Relatively small screws are currently used to connect the modules prior to deployment, which presents a challenge in the field with sub-zero air temperatures. A module connection system requiring less fine motor work is suggested for future designs. It is also suggested that such a system incorporate a single standard bulkhead connector between all modules to reduce the setup effort required to connect many independent



(a) Sonar (b) Video

Figure 41: Sonar and video images collected during the Icefin vehicle's under-ice dive jetty deployments. The textured frazil ice and ice-hole features can be seen here.

bulkhead cables between modules. During deployment, it was discovered that the vertical hovering thrusters disturb the seafloor and reduce visibility at close proximity. Any design incorporating vertical thrusters is encouraged to design a control system to engage these thrusters minimally when close to the seafloor in areas of loose sediment. While a magnetic compass cannot be used for polar operations, it is suggested that this sensor be added to any polar vehicle design, with the option to ignore sensor data in the control software. These sensors are small and low cost and can increase the utility of the vehicle in the likely case that the vehicle is used for missions at less extreme latitudes. During post-analysis of the sonar and video data obtained by the Icefin vehicle during Antarctic deployment, it was learned that sufficient texture and features in the ice-cover were present to be considered for use in vision-based algorithms to aid in navigation, currently under development.

The use of internal vectored thrusters in place of protruding control planes in the vehicle design proved successful, but requires additional energy expenditure. A hybrid of retractable control planes and vectored thrusters is suggested for consideration with future through-ice-shelf designs. While the deployment and recovery methodology used here was successful, internal actuators for movement of batteries (similar to seagliders) is suggested as a method to consider, in place of a drop weight, for moving the center of mass to control vehicle pitch. Provided there is sufficient space in the vehicle (which was not the case here), such internal mass actuation would allow the vehicle to be pitched vertically for recovery

as well as deployment.

3.5.4.5 Extrapolation for Future AUV Planetary Exploration

Many of the lessons learned from the design and deployment of the unique Icefin vehicle can be extrapolated for use in future Europa and other planetary exploration missions. Use of a hot water drill to provide a small-diameter hole through thick ice for vehicle deployment proved effective. However, this drilling method would need to be replaced by onboard melting or drilling for Europa missions. Deployment and recovery of the vehicle vertically through a small diameter hole (as well as the use of a drop-weight system for pitch translation) proved successful in Antarctica and is a key technology that can be extrapolated to future planetary AUV missions. Such a method might be considered for use in an autonomous Europa mission, although more safeguards would be needed to help guide the vehicle back into the hole in the unlikely case that vehicle recovery is a requirement.

The use of internal thrusters and lack of protruding control planes is suggested for consideration in future Antarctic and planetary AUV designs deployed through small diameter ice holes. Such a self-contained design lowers the probability of damage to fragile but necessary external components. However, additional energy is required to run these thrusters, and thus a tradeoff is presented. A hybrid system might be considered for planetary missions with a combination of directional thrusters and retractable control planes. An inertial navigation system proved to be effective for tethered Antarctic missions, but a method less prone to drift (such as Simultaneous Localization and Mapping [7]) is suggested for longer duration planetary missions.

While the Icefin vehicle does not currently contain sufficient autonomy in deployment and control for planetary deployment, and is not rated for some of the extreme conditions encountered in space, many of the lessons learned in the design of the vehicle and deployment in Antarctica can be extended to aid the design of future planetary AUVs. A system capable of self-deployment to Europa must be fully autonomous, capable of landing on Europa, capable of drilling and deploying through hundreds of meters of ice, should

have a robust underwater-surface-Earth communication system, and will be very limited in power, size and weight. This is a tall order to fill, but with recent advances in the field, the possibility of such a capable system now seems to be on the horizon.

3.5.4.6 Summary

Deep-field Antarctic deployment of a complex under-ice vehicle system is accompanied by many unique challenges. The Icefin vehicle was designed as a modular unmanned underwater vehicle with an extensive sensor suite and an optical fiber tether for control and communication to meet these challenges while obtaining high science return. The vehicle was deployed to McMurdo station in Antarctica from October to December of 2014. Designed for the harsh under-ice environment encountered on the McMurdo Ice Shelf, the Icefin vehicle was successfully deployed over multiple sub-ice exploration missions to obtain data of previously unexplored areas beneath the ice shelf. The Icefin vehicle's small size and human-portability lowered logistical effort and cost, as expected. Vertical deployment through a small diameter hole in a thick ice shelf was verified as a tenable approach. The vehicle's maiden sub-ice voyage took place in November of 2014, and was followed by additional missions out in the deep field. Further sub-ice deployment of the vehicle is planned for Greenland in 2016 and a return to Antarctica in future seasons.

3.6 Summary

Evaluation of the algorithms presented in this dissertation requires sonar and video data obtained in sub-ice environments. The VideoRay Pro IV and custom Icefin vehicles were used for data collection beneath the ice in Lake John, Colorado and McMurdo, Antarctica respectively. These real-world under-ice datasets were used for development and validation of the algorithms presented in the chapters below. The field deployments, as well as the vehicle platforms and sensor packages used in this work, were discussed in this chapter.

CHAPTER 4

SIMULATION

4.1 Introduction

The algorithms presented in this dissertation include methods for motion estimation, sensor fusion, navigational facilitation, ice texture estimation, and ice anomaly detection and mapping with an under-ice UUV. In order to robustly develop, test, and evaluate these algorithms, under-ice datasets similar to those encountered during full mission deployment of the algorithms are required. Real-world, under-ice dataset collection presents many challenges and requires a prohibitive amount of time, cost and support to complete. Such environments are also very hazardous and force human operators into dangerous situations. A method for simulation of such under-ice environments is presented here for use in vision-based algorithm development and evaluation. Simulation-based methods for data collection are beneficial for many reasons: very low-cost, tight control of the environment and sensor trajectory, knowledge of ground truth information, and faster results. These benefits provide motivation for the creation of the under-ice simulation systems presented here. The algorithms presented in this dissertation require forward-looking sonar as well as camera sensor datasets of sufficient visual accuracy for algorithm evaluation. Simulated under-ice environments are created here for both sonar and camera sensors for this purpose. A variety of datasets are obtained over multiple vehicle trajectories and sensor angles. Development of the simulation methods used here is detailed in this section, and the resultant datasets are presented.

4.2 Camera Simulation

Cameras are present on almost all unmanned underwater vehicles due to the desire of human operators and scientists to view the mission environment. Under-ice UUVs are no exception to this rule. The Icefin under-ice vehicle considered here has two onboard cameras;

one upward-facing and one front-facing. A method for simulation of these two unique camera views in an under-ice environment is presented here. Using Blender [11], realistic and visually consistent scenes of an under-ice environment can be created and imaged. Blender has been used in multiple previous works, including some in arctic environments [81], for visually accurate imagery simulation. Motion of the camera can be simulated over the environment to imitate a realistic vehicle trajectory. The output of such a Blender simulation provides a stream of sequential simulated images as a video file. Motion of the camera between frames can be clearly seen in the resultant output video as relative world motion. Upon creation of a sufficiently accurate under-ice environment in Blender, videos can be obtained simulating realistic motion of an under-ice vehicle over this environment. These output videos can be used for evaluation of under-ice vision-based algorithms and should yield equivalent results to real-world video datasets.

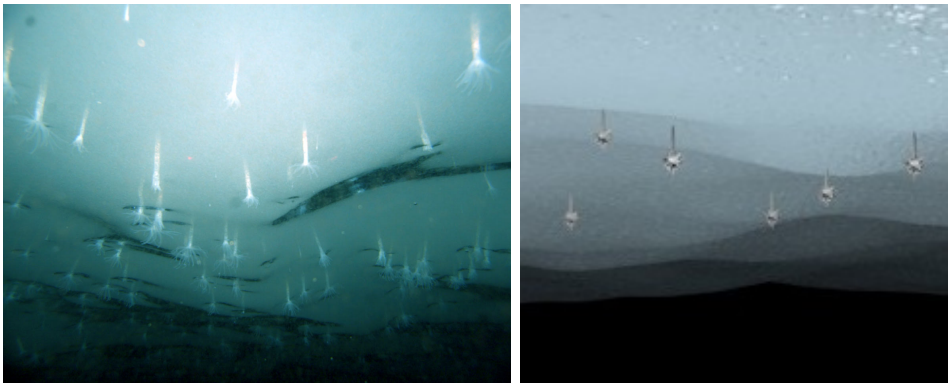
Ice can vary drastically in texture from first year ice that can be only a few centimeters thick and very smooth with cracks, to multi-year ice that is smooth but varies greatly in topography, to platelet ice that is comprised of a textured mix of small ice fragments at the ice-water boundary, to drastically textured frazil ice comprised of half-formed crystals of ice at the ice-water boundary. In some sub-ice regions, multiple types of ice can be encountered during a single UUV mission. In Antarctica, some under-ice locations encounter large currents where any frazil or platelet ice is swept away to leave only the smooth ice beneath, while other locations are sheltered from these currents and platelet or frazil ice can accumulate multiple meters in thickness. Here, an under-ice environment was created in Blender containing flat and smooth first-year ice, topographically varying but smooth multi-year ice, textured platelet ice in a wave-like formation (from currents), and drastically textured and crystal-like frazil ice. A summary of each ice type and its formation details in Blender is presented in Table 7.

Anomalous objects present at the ice-water interface are often of interest to scientists for analysis as well as to navigational systems to provide unique visual landmarks. Cracks

Table 7: Simulated ice types and corresponding Blender configuration settings

Ice Type	Blender Settings
All (ICE Material)	**Diffuse:** Intensity=0.7, R=0.5,G=0.7, B=0.8, **Specular:** Intensity=0.1, R=0.72,G=0.72,B=0.72, Hardness=100, **Shading:** Emit=0, Translucency=0, Transparency=FALSE, Shadow Cast = FALSE
Brash/Platelet Ice	**Displacement Modifier:** Texture=Dunes, Strength=1.0, **Displacement Modifier:** Texture=Clouds, Strength=0.4, **Displacement Modifier:** Texture=Noise, Strength=0.2, **Subsurface Modifier:** View=1, Render=1 **Displacement Modifier:** Texture=Noise, Strength=0.1
Multi-year Smooth Ice	**Displacement Modifier:** Texture=Voronoi, Size=0.2 **Displacement Modifier:** Texture=Clouds, Strength=0.02 **Subsurface Modifier:** View=1, Render=1 **Displacement Modifier:** Texture=Noise, Strength=0.008
Frazil Ice	**Displacement Modifier:** Texture=Noise, Strength=1.0 **Displacement Modifier:** Texture=Clouds, Strength=1.0 **Subsurface Modifier:** View=1, Render=1 **Displacement Modifier:** Texture=Noise, Strength=0.1
First-year Smooth Ice	**Displacement Modifier:** Texture=Noise, Strength=0.1 **Subsurface Modifier:** View=1, Render=2 **Displacement Modifier:** Texture=Noise, Strength=0.01

in the ice, holes in the ice, ice deformations, and even life such as anemones [12] can been found in previous under-ice video datasets. Similar anomalies were added to regions of each type of ice in the simulated environment presented here, to include ice cracks, topographical ice deformations, sections of green ice, anemone-like objects, sponge-like objects, and other animal-like objects. While these plant- and animal-like anomalies do not need to be completely visually consistent with real anomaly images for the algorithms developed here, it is important that they provide a unique color and texture against the ice background, as would be the case with a real ice anomaly. A summary of these anomalies introduced into the under-ice Blender environment is presented in Table 8.



(a) (b)

Figure 42: Real-world under-ice camera images (left) along with the resultant simulated under-ice camera images (right) obtained using the simulation methods presented here.

Using Blender, the ground truth of vehicle position over the entire trajectory as well as the ground truth map of anomaly locations is known exactly and can be used for algorithm validation. A simulated camera is used to produce video datasets as the camera moves over the simulated ice environment, just as a camera on a UUV would. A view looking straight up (pitch=90 degrees) at the ice is simulated in addition to a forward view (pitch=0 degrees) from the front of the simulated vehicle. This accurately simulates the two cameras available on the Icefin UUV platform being simulated, which was used for real under-ice data collection in McMurdo, Antarctica. The results of the simulated under-ice environment presented here provide realistic, visually consistent, under-ice images. Examples of

such images can be seen below, compared with a corresponding real-world, under-ice image. The anemones found under the ice in Antarctica [12][3] can be seen in Figure 42a, along with simulated anemones in Figure 42b. A patch of green ice located beneath the Antarctic sea ice can be seen in Figure 43c, along with simulated version of such anomalous ice in Figure 43d. The four different ice types simulated here can be seen in Figure 43b (frazil ice), Figure 43f (smooth multi-year ice), Figure 43d (Brash or platelet ice), and Figure 43h (smooth first-year ice). Real-world images from under-ice datasets are used for comparison of these simulated images; Figures 43a, 43e, 43c (collected in Antarctica with the Icefin vehicle) and Figure 43g (collected in Lake John, Colorado using the VideoRay vehicle) respectively. Views from both simulated camera viewpoints can be seen in these images as well. For example, the forward-looking camera view is used to synthesize Figure 43d, while the upward-looking camera view is used for Figure 43f.

Table 8: Simulated under-ice anomalies and corresponding Blender configuration settings and components

Anomaly	Anomaly Description	Blender Settings and Components
Ice crack	Crack in the ice. Visual but not largely topographical.	Shape=Icosphere, Hardness=511, Diffuse Intensity=0.5, Specular Intensity=1.0, Transparency=TRUE, Alpha=0.2, **Displacement Modifier:** Texture=Noise, Strength=0.4
Ice deformation	Topographical protruding feature in the ice	Material=ICE (see above), Shape=Icosphere, **Displacement Modifier:** Texture=Noise, Strength=1.0
Anemone	White color, complex shape	Shape=Multiple connected icospheres, Hardness=1, Diffuse intensity=0.8, Specular intensity=0, Color(R,G,B)=(1,1,1)
Sea star	White color, star shape	Shape=Multiple connected icospheres, Diffuse intensity=0.8, Specular intensity=0, Color(R,G,B)=(1,1,1)
Green ice patch	Green color, circular but flat shape	Shape=Icosphere, Color(R,G,B)=(0.1,0.3,0.1), Hardness=50, Diffuse intensity=0.8, Specular intensity=0.5, **Displacement Modifier:** Texture=Noise, Strength=0.1
Sponge	Various colors, complex and jagged structure	Shape=Icosphere, Diffuse intensity=0.8, Specular intensity=0, Color(R,G,B)=(0.8,0.8,0.3), Color(R,G,B)=(0.6,0.1,0.6) **Displacement Modifier:** Texture=Noise, Strength=10
Sea slug	Dark color, rounded circular shape, antennae	Shape=Multiple connected icospheres, Diffuse intensity=0.8, Specular intensity=0, Color(R,G,B)=(0,0,0)
Shrimp	Dark color, complex shape	Shape=Multiple connected icospheres, Diffuse intensity=0.8, Specular intensity=0, Color(R,G,B)=(0,0,0)

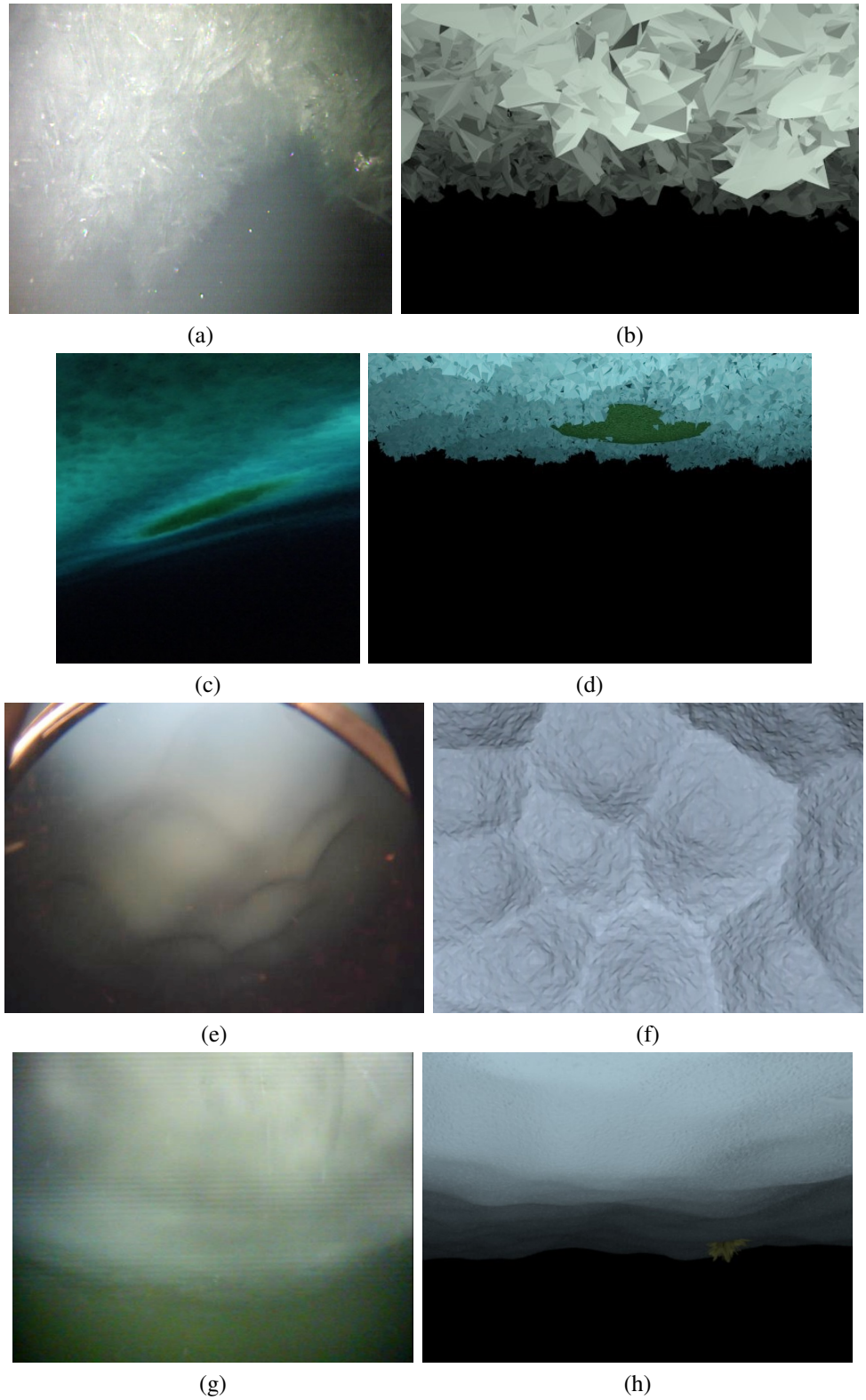


Figure 43: Real-world under-ice camera images (left) along with the resultant simulated under-ice camera images (right) obtained using the simulation methods presented here.

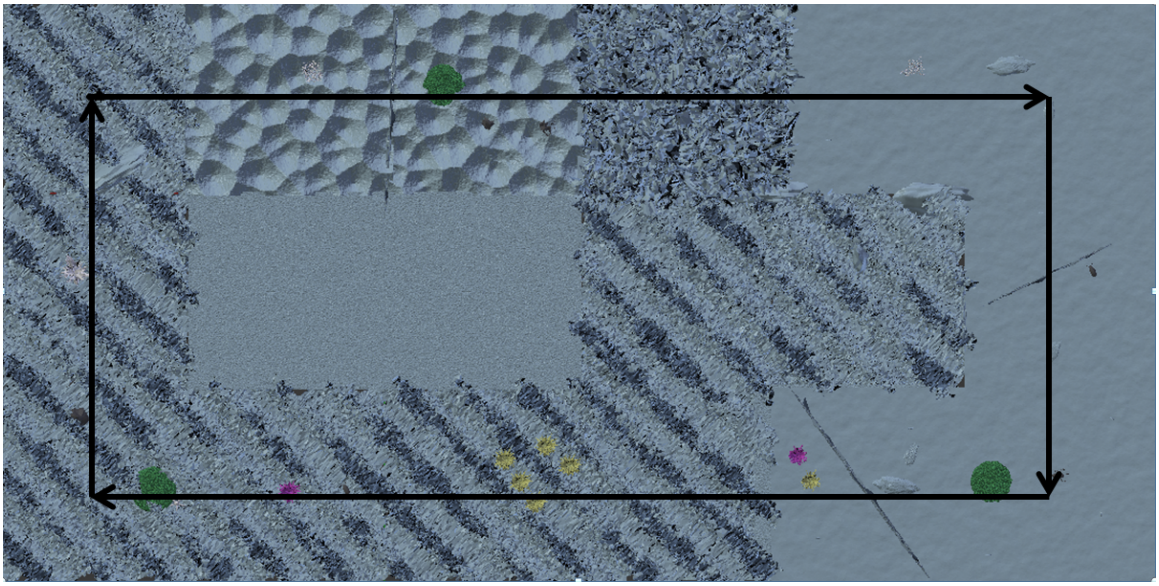


Figure 44: Ground truth map of the all-ice rectangular simulated under-ice dataset. This view is looking up from beneath the ice at the entire environment with the various ice types and ice anomalies.

In order to obtain useful video datasets for vision-based algorithm development, a variety of camera, or vehicle, trajectories were simulated over these simulated under-ice environments. The simulated cameras were first moved in a translational (purely surge and sway) trajectory as well as a rotational (yaw or heading) pattern around the vehicle's z-axis to obtain decoupled translational and rotational datasets for unit-testing of algorithm functionality. This was done for each ice type individually as well as over both camera angles. A summary of these decoupled simulation trajectories is presented in Table 9. In addition to these rotation- and translation-only trajectories over a single ice type, more complex camera trajectories were also simulated. A surge-only trajectory as well as a surge and sway rectangular trajectory over all ice types and both camera angles was simulated to present a dataset containing various ice textures and shapes. An overview of the rectangular all-ice environment is can be seen in Figure 44. While these trajectories are also decoupled from rotation, a final trajectory was simulated with an s-curve trajectory which couples translation and rotation over both camera angles for full vision-based motion estimation algorithm validation. A summary of all simulated, under-ice camera datasets is presented in Table 9.

Ground truth (x, y, z) position and heading are recorded for each video frame as ground truth over each trajectory. These simulated datasets provide the most accurate means for evaluation of many vision-based algorithms as they contain the most reliable ground truth.

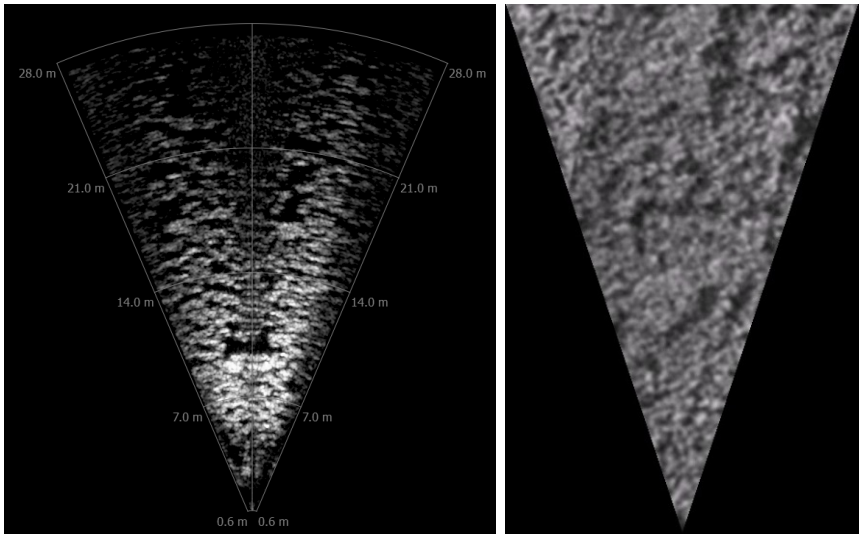
Table 9: Camera Simulated Datasets

Ice Type	**Camera Angle**	**Trajectory**
First-year Smooth	Up	Yaw-only
		Forward-only (Surge)
	Front	Yaw-only
		Forward-only (Surge)
Brash/Platelet	Up	Yaw-only
		Forward-only (Surge)
	Front	Yaw-only
		Forward-only (Surge)
Multi-year Smooth	Up	Yaw-only
		Forward-only (Surge)
	Front	Yaw-only
		Forward-only (Surge)
Frazil	Up	Yaw-only
		Forward-only (Surge)
	Front	Yaw-only
		Forward-only (Surge)
All Ice Types	Up	Forward-only (Surge)
		Rectangular (Surge/Sway)
		S-Curve (Coupled Surge, Sway, and Yaw)
	Front	Forward-only (Surge)
		S-Curve (Coupled Surge, Sway, and Yaw)

4.3 Forward-Looking Sonar Simulation

Forward-looking multibeam sonar sensors, such as the BlueView sensor [60] considered here, are used in multiple algorithms presented in this dissertation. These sensors are considered to be "imaging sonars" and present the viewer with a relatively high-definition, two-dimensional representation of the area in front of the sensor, up to a certain range and

bearing field of view. An example of such a sonar image taken of the sea ice near McMurdo, Antarctica can be seen in Figure 45a. In order to simulate such sonar images of an under-ice environment with sufficient accuracy for vision-based algorithm evaluation, a large image with similar texture to that found under the ice is used as a representation of the under-ice environment. In order to imitate the limited range and field of view of the sensor, a triangular (45 degree) bounding frame is translated and rotated over the large world image to obtain simulated local views over a trajectory. An example of such a bounded-frame simulated sonar image can be seen in Figure 45b in comparison with an actual under-ice sonar image in Figure 45a.



(a) (b)

Figure 45: Under-ice sonar data (a) obtained of the sea ice near McMurdo, Antarctica along with a simulated version (b) of this sonar image.

The texture in these images represents formation of the semi-frozen platelet, brash or frazil ice located at the ice-water boundary. Anomalies in the ice (such as that seen in the lower-center of Figure 45a) present strong returns in the image and are also present in smooth ice (no semi-frozen ice layer). An example of a strong return from an anomaly deformation in the ice can be seen in Figure 46a, along with a simulated high-return object

in Figure 46b. Contiguous areas of low return are also present in both smooth and non-smooth ice types, and represent holes in the ice or acoustic shadows following large, high-return objects. An example of such a low-return area can be seen in Figure 46c, which represents a hole drilled in the sea-ice near McMurdo, Antarctica. A simulated sonar image with such a feature is shown in Figure 46d for comparison.

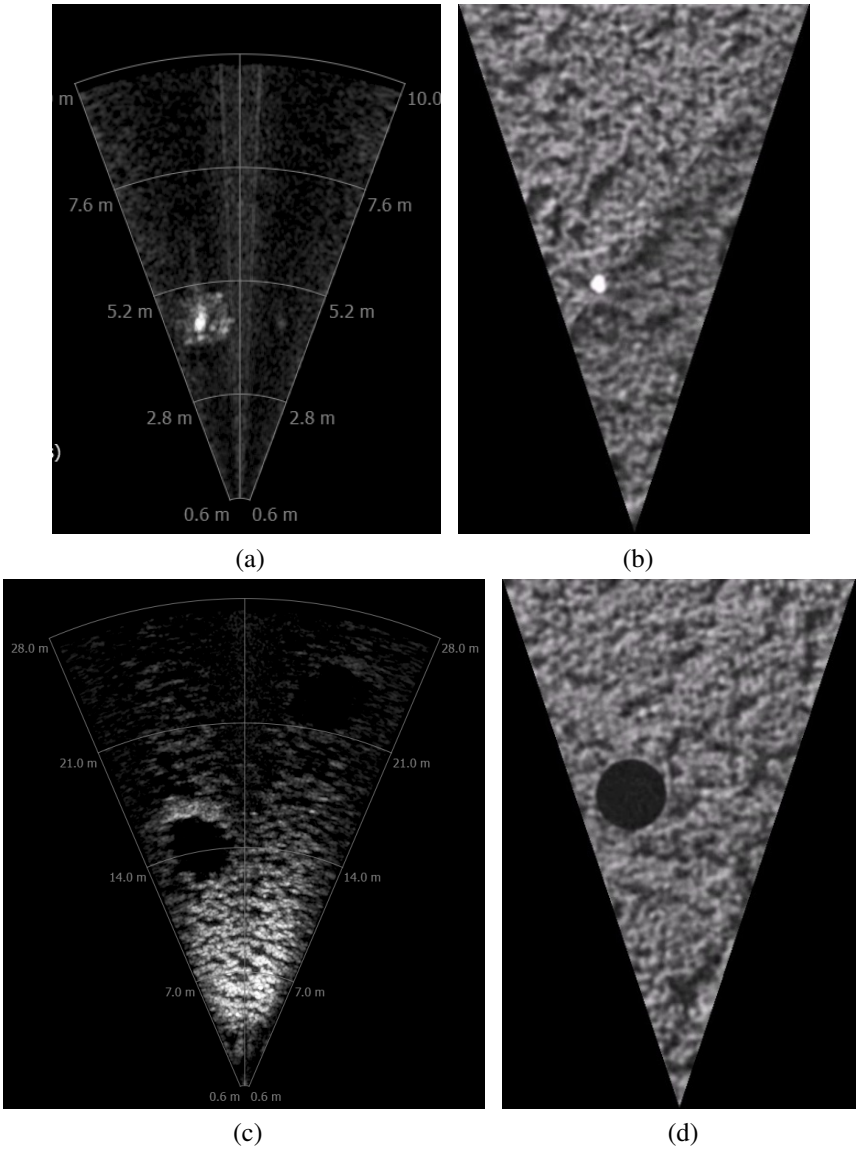


Figure 46: Under-ice sonar data (left) obtained of the sea ice near McMurdo, Antarctica along with simulated versions of this sonar data (right). Areas of strong return in the sonar can be seen in (a)-(b) while low-return areas can be seen in (c)-(d).

Many different trajectories are simulated using this under-ice sonar simulation method.

Table 10: Sonar simulated datasets

Trajectory	Description
Forward	Forward-only (surge) decoupled motion
Circle	Rotation-only (yaw) decoupled motion over 360 degrees
Rectangle	Surge and sway motion in a rectangular trajectory returning to the initial point
S-Curve	Coupled rotation (yaw) and translation (surge/sway) to form alternating circular trajectories which create an s-curve shape

These trajectories are designed to match those discussed in the camera simulation section above. Such complementary datasets provide the means for evaluation of sensor fusion algorithms. In the case of the sonar sensor (as opposed to the camera sensor), only a single sensor angle is considered (forward-looking) as this is the common sensor configuration. Decoupled rotation- and translation-only trajectories are simulated, as well as a rectangular surge/sway trajectory. An s-curve trajectory coupling rotation and translation (similar to that in the camera simulation section above) is also simulated for full validation of motion estimation algorithms. A summary of all trajectories simulated over the under-ice sonar datasets created is presented in Table 10.

4.4 Summary

The difficulties encountered in obtaining under-ice sonar and camera datasets due to the infrastructure required and the harshness of the environments can be very prohibitive to such data collection missions. Development of an under-ice environment simulation method is presented here as a low-cost, controlled, and time-effective approach for creation of under-ice datasets for evaluation of vision-based algorithms such as those considered here. These simulation methods are detailed herein and results from such under-ice environment simulation is presented as images along with real-world imagery for comparison. These simulation methods prove effective for low-impact dataset creation for the evaluation of the vision-based algorithms considered here.

CHAPTER 5

UNDER-ICE MONOCULAR CAMERA RELATIVE POSE ESTIMATION

5.1 Introduction

Localization and navigation through underwater environments is a difficult task for divers, submarines and unmanned underwater vehicles (UUVs) alike due to the lack of salient visual features, poor visibility, limited communications, and absence of GPS signals used as a standard in most terrestrial and airborne applications. Under-ice environments tend to present even fewer salient visual features, less natural light, and increased difficulty deploying acoustic beacon based localization methods commonly used in the open ocean. Most current under-ice vehicles use an inertial navigation system for position estimation, despite the large error drifts encountered, which limit mission duration. One increasingly common localization and motion estimation approach used with unmanned vehicles is termed vision-based relative pose estimation. Using corresponding feature points detected in consecutive images, the motion of a camera between these frames can be estimated. Vision-based relative pose estimation has been previously evaluated for use in underwater environments, but typically requires feature rich and salient image streams not present under the ice.

Adaptation of a camera-based relative pose estimation method previously used on aerial and terrestrial vehicles is presented here for use in relatively featureless under-ice environments. Preprocessing of the low-contrast input images and adaptation of this relative pose estimation approach for this unique environment results in a self-contained method for estimation of under-ice vehicle motion between image frames in six degrees-of-freedom. This camera-based relative pose estimation algorithm is evaluated using both simulated and real-world, under-ice datasets with positive results. This algorithm provides a method for vehicle-relative pose estimation that is less vulnerable to integration-based drift error, such as that encountered using an inertial navigation system (INS), and does not require external

infrastructure, in contrast to acoustic beacon-based methods.

5.2 Camera Model and Intrinsic Parameter Estimation

In order to relate pixels in a camera image to points in the outside world from which they were obtained, a camera model for how an image is formed must be considered. The model used here is an extension of the simple pinhole camera model with added distortion. Almost all cameras have some amount of tangential and radial (fisheye) distortion. Equations 5 – 6 (radial distortion) and 7 – 8 (tangential distortion) can be used to model this distortion and correct for it [119] [120]. In these equations, x and y represent the pixel coordinates and r represents the radius from the center of the image, while k_1-k_3 and p_1-p_2 represent the radial and tangential distortion coefficients respectively.

$$x_{corrected} = x * (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{5}$$

$$y_{corrected} = y * (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{6}$$

$$x_{corrected} = x + [2p_1 xy + p_2(r^2 + 2x^2)] \tag{7}$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy] \tag{8}$$

A camera calibration process is used to estimate both the pinhole camera model parameters (focal lengths and principal point) and the distortion coefficients. Camera calibration is a process that uses multiple images (from different viewpoints) of a known object to obtain an estimate of the camera's intrinsic parameters (focal length, principal point and distortion coefficients) that minimizes the reprojection error of the model. In this case, a planar chessboard pattern with known dimensions is imaged, the corner pixel locations are extracted, and these pixel locations are compared with the known model of the chessboard to determine the intrinsic parameters.

Calibrated models for the cameras used here for data collection do not exist in the literature and were thus obtained through testing. The VideoRay camera encounters a significant amount of fisheye distortion due to the lens and the large, clear plastic dome that acts as

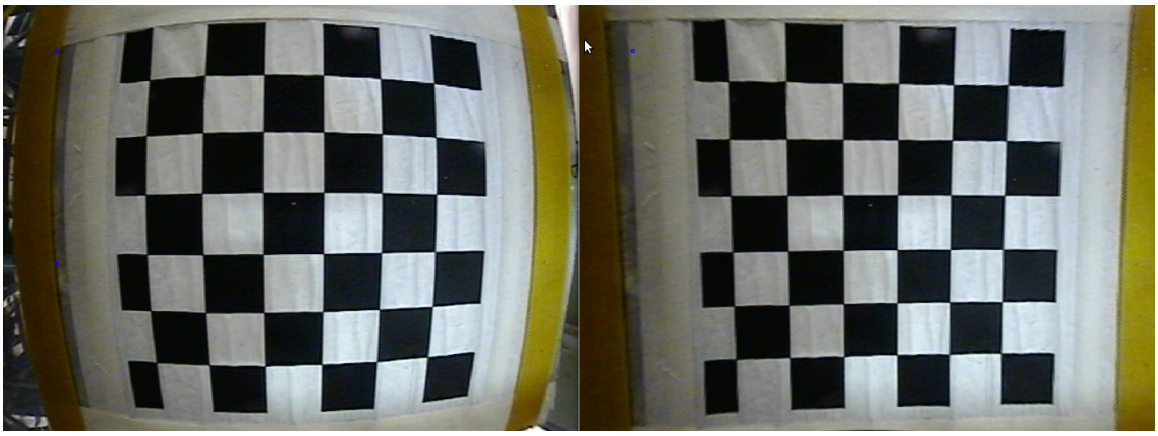


Figure 47: Uncalibrated (left) and calibrated (right) images from the VideoRay camera.

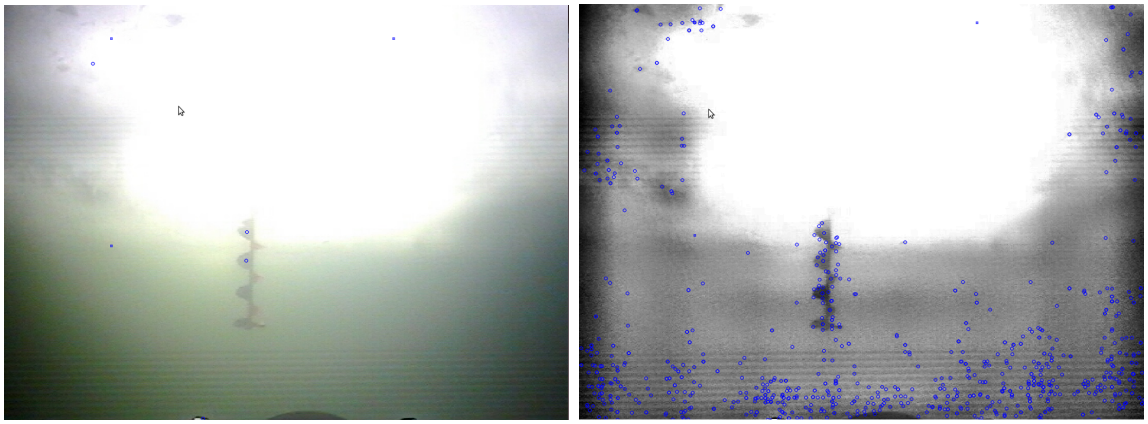
an additional optical element. An example of a VideoRay chessboard calibration image both before and after removing distortion is shown in Fig. 47. The VideoRay camera hardware provides the capability to pan up and down inside the plastic dome, but movement of the camera induces different distortion effects, and each tilt angle must be calibrated separately. Using the VideoRay Pro IV camera, separate calibration datasets with images of the chessboard pattern were obtained underwater for each sensor angle used when obtaining the under-ice dataset in Lake John, Colorado. Chessboard calibration datasets were also obtained for the forward and upward cameras present on the Icefin vehicle. The simulated (Blender) camera provides control over the intrinsic parameters, and did not require calibration. The algorithm used for calculation of intrinsic and extrinsic parameters is that presented in [73] based on [119] and [120]. A summary of the calibration results obtained for the camera parameters is shown in Fig. 48. The average pixel reprojection error is calculated by taking the arithmetical mean of the absolute norms between all projected points in the model and the corresponding detected corner locations. Following calibration, distortion can be removed to produce images assumed to satisfy an ideal pinhole camera model.

Camera Setup	Camera Model Parameters						Camera Distortion Coefficients				
	Img Size (pix)		*fx*	*fy*	*cx*	*cy*	*k1*	*k2*	*p1*	*p2*	*k3*
HP Laptop	640	480	677.7	676.6	319.5	239.5	0.058	-0.391	0.016	-0.009	0.521
VideoRay 0° Win.	720	540	450.2	452.5	359.5	269.5	-0.297	0.113	0.003	0.002	-0.025
VideoRay 0° Win.	720	540	474.3	474.0	359.5	269.5	-0.321	0.126	0.002	0.002	-0.031
VideoRay 0° Win.	720	540	471.3	468.3	359.5	269.5	-0.318	0.137	0.005	0.002	-0.040
VideoRay 0° Win.	720	540	471.6	477.5	359.5	269.5	-0.320	0.129	0.002	0.000	-0.027
VideoRay 0° Lin.	496	480	312.6	413.6	247.5	239.5	-0.310	0.136	0.005	0.001	-0.041
VideoRay 17° Lin.	496	480	319.2	421.6	247.5	239.5	-0.320	0.140	0.003	0.001	-0.038
VideoRay 30° Lin.	496	480	326.6	432.6	247.5	239.5	-0.333	0.144	0.004	-0.003	-0.038
VideoRay 68° Lin.	496	480	316.4	416.0	247.5	239.5	-0.294	0.081	0.001	-0.001	-0.008
GoPro	1280	960	601.6	599.2	639.5	479.5	-0.269	0.087	-0.001	-0.002	-0.014
Icefin Forward	640	480	811.5	806.2	319.5	239.5	-0.433	0.197	-0.001	-0.001	0.078
Icefin Down	640	480	315.6	415.0	319.5	239.5	-0.323	0.128	0.000	-0.004	-0.025

Figure 48: Table of underwater calibrated camera intrinsic parameters along with the average pixel reprojection error of the estimated models. Here f_x and f_y are the camera focal lengths and c_x and c_y represent the optical center in pixel coordinates.

5.3 Contrast Limited Adaptive Histogram Equalization (CLAHE)

Due to the low contrast encountered in under-ice images, use of a preprocessing technique called contrast limited adaptive histogram equalization (CLAHE) [82] was used to adaptively adjust the contrast of local pixel neighborhoods based on the intensity histogram distribution in that neighborhood. This method results in globally contrast-enhanced images, and allows for improved point-feature detection. The CLAHE mapping function maximizes the contrast of the most common intensity values. This mapping function is shown in Equations 9 – 10. Here i represents intensity, $C_{clahe}(i)$ is the contrast of a specific intensity value, $f_{clahe}(i)$ is the mapping function, α is a constant representing the display range divided by the region size, and C_{max} is the clip limit on the contrast. An example of CLAHE applied to an under-ice image is shown in Figure 49. Point-features can be automatically extracted from images to provide strong points for tracking. Detected point-features are shown as small circles in Figure 49 to show the increased performance (more features detected) of such point-feature detection algorithms on CLAHE contrast enhanced



(a) Raw original image (b) CLAHE preprocessed image

Figure 49: SIFT feature detections in an original (a) and CLAHE preprocessed (b) image.

images. Preprocessing of low-contrast, feature-poor, under-ice images using CLAHE enhances the capabilities of computer vision methods, such as point-feature algorithms, in this environment. Therefore, vision-based relative pose estimation methods, commonly used with feature-rich imagery, can now be utilized in this much harsher environment.

$$C_{clahe}(i) = \begin{cases} histogram(i), & histogram(i) < C_{max} \\ C_{max}, & histogram(i) >= C_{max} \end{cases} \tag{9}$$

$$f_{clahe}(i) = \alpha \int_0^i C_{clahe}(\tau)\mathrm{d}\tau \tag{10}$$

5.4 Vision-Based Relative Pose Estimation Algorithm

In order to estimate relative motion of the vehicle between camera frames, application of a vision-based relative pose estimation method, commonly used in feature-rich environments, is presented here. A combination of preprocessing, low feature detection thresholds, and robust matching methods is presented in order to successfully apply this method to feature-poor, low-contrast under-ice environments. The method presented here for visual relative pose motion estimation utilizes a five-point algorithm with random sampling and

consensus (RANSAC) [68] provided by Nister [89]. The required point-features for matching between images and use in relative pose estimation methods can be extracted once the camera images in the system here have been preprocessed, distortion has been removed, and histogram equalization (CLAHE) has been applied. Matching corresponding points between images requires point-features that can be easily found and reliably matched, such as the common Shi-Tomasi [61] or SURF [64] point-features. Shi-Tomasi features are a modified version of Harris corner point-features, which are based on derivatives (the second order moment matrix) [62] or local gradients of the neighborhood surrounding a pixel. SURF features can be extracted very fast and use an approximation of the determinant of Hessian blob detector. Once a set of point-features have been detected and a description for them has been extracted, the point-features between two images can be matched.

The method presented here for estimating corresponding point sets between images utilizes optical flow theory. Optical flow [70] is a pixel-based method often used to estimate motion of pixel points between images in a sequence. This differential algorithm works on the assumption of a brightness constancy constraint (Equation 11), similar motion between neighboring pixels (smoothness), and small temporal movement between the images. In Equation 11, u and v refer to the x and y motion vector magnitudes respectively, while E_x, E_y and E_t are the partial derivatives of image brightness with respect to x, y and time respectively. The Lucas-Kanade algorithm [71] used here provides an approximation for optical flow of 3x3 pixel patches [73] in the images by using least squares minimization to find an estimate for the gradient constraint equation at a sparse set of Shi-Tomasi [61] or SURF points. Specifically, a pyramidal implementation of this method [121] is used in order to increase robustness over scale-space. A candidate set of matched points between images results from the application of this Lucas-Kanade method. While point-feature and optical flow algorithms are common in computer vision applications, underwater and under-ice relatively featureless environments, such as that here, present much less common and more challenging applications.

$$uE_x + vE_y + E_t = 0 \tag{11}$$

Vision-based relative pose estimation methods are not commonly used in underwater or under-ice applications due to low contrast and limited features encountered. In order to apply such a method to the feature-poor environments of interest here, point-feature detection and matching thresholds must be set low to provide the maximum number of candidate matches between images. However, this dramatically increases the number of false detections and matches, and a robust model estimation method is used here to remove these false matches from the inlier set. The method presented here utilizes adaptive contrast enhancement (CLAHE), as well as a system with maximum point-feature detections on the front end and robust model estimation on the back end, to provide the robustness required for use in under-ice environments. Provided sufficient candidate corresponding points matched between images, it is possible to estimate the relative pose of the camera (and therefore the vehicle) between successive images. In computer vision, the essential matrix (**E**) encodes complete information on the 3-D rotation and translation between two camera viewpoints. Nister's five-point algorithm [87] provides a method for estimation of the essential matrix with only five point correspondences between images, using the epipolar constraint equation (Equation 12). Here x_1 and x_2 are points in the image plane and **E** is the essential matrix. When more than five candidate point correspondences are available, robust parameter estimation techniques such as random sample consensus (RANSAC) [68] can be used to improve the parameter accuracy. RANSAC iteratively finds a minimal random sample set, calculates the motion model for this set, and then computes the resultant error from this model over all match points and repeats this process until a satisfactory model (with sufficient inlier points) is produced. All matches within a certain error threshold of the final model are considered "inliers" while any matches with model error above this threshold are considered "outliers". RANSAC is used here to eliminate false matches from the motion model calculation, and allows for lower detection and matching thresholds, and therefore

more candidate matches, on the front end.

$$\hat{\mathbf{x}_2}^T \cdot \mathbf{E} \cdot \hat{\mathbf{x}_1} = 0 \tag{12}$$

In the vision-based pose estimation algorithm presented here, the essential matrix between two camera viewpoints is extracted from corresponding point sets between two consecutive images, as detailed above. The essential matrix can be decomposed into two possible rotation matrices and a translation vector using singular value decomposition (SVD) [122] with Equations 13–17. In these equations $\mathbf{E}$ is the essential matrix, $\mathbf{t}$ is the translation vector, $\mathbf{R}$ is the rotation matrix, $\mathbf{U}, \mathbf{V}, \mathbf{\Sigma}$ are the decomposed SVD matrices, $\mathbf{R_1}$ and $\mathbf{R_2}$ are two possible rotation matrices, and $\mathbf{W}$ is the matrix in Equation 17. SVD decomposition provides four possible sets of rotation and translation ($\mathbf{R}_1 \mid \mathbf{t}$, $\mathbf{R}_2 \mid \mathbf{t}$, $\mathbf{R}_1 \mid -\mathbf{t}$, $\mathbf{R}_2 \mid -\mathbf{t}$). The two possible rotations are related through a 180 degree rotation around the translation vector [123], and the roll and pitch of the vehicle platforms used here are assumed to be zero. Therefore, the rotation vector chosen for the relative pose estimate, in the algorithm presented here, is that with the smallest roll and pitch Euler angles. The scale and sign of the translation vector is not recoverable from this method.

$$\mathbf{E} = [\mathbf{t}]_X \mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{13}$$

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}^T\mathbf{V}^T \tag{14}$$

$$\mathbf{R}_2 = \mathbf{U}\mathbf{W}\mathbf{V}^T \tag{15}$$

$$\mathbf{t} = \mathbf{U}(0, 0, 1)^T \tag{16}$$

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{17}$$

The Euler angles of rotation can be extracted from the rotation matrices using Equations 19–21, assuming that the rotation matrix is of the form shown in Equation 18. In these equations ϕ, θ and ψ are angles around the x, y and z axes respectively. $\mathbf{R}_x$, $\mathbf{R}_y$, and $\mathbf{R}_z$ are

the individual rotation matrices for these angles, and c_x, c_y, c_z, s_x, s_y and s_z are the cosine and sine values of these angles.

$$R_{zyx} = R_z R_y R_x = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} c_z c_y & c_z s_y s_x - s_z c_x & c_z s_y c_x + s_z s_x \\ s_z c_y & s_z s_y s_x + c_z c_x & s_z s_y c_x - c_z s_x \\ -s_y & c_y s_x & c_y c_x \end{bmatrix} \tag{18}$$

$$\phi = tan^{-1}(r_{32}/r_{33}) \tag{19}$$

$$\theta = tan^{-1}(-r_{31}/\sqrt{r_{32}^2 + r_{33}^2}) \tag{20}$$

$$\psi = tan^{-1}(r_{21}/r_{11}) \tag{21}$$

In order to obtain the correct motion model estimates between frames in the vehicle coordinate system instead of the camera coordinate system, the result must be multiplied by the rotation matrix $R_{image2vehicle}$ in Equation 22 to convert from image coordinates to vehicle coordinates (Figure 50). The camera tilt angle is accounted for by further multiplying the model by the rotation matrix R_{sensor} in Equation 23, where θ corresponds to the camera tilt angle. The resulting output motion model provides relative pose estimates of the camera between image frames. The number of inliers in the estimated model is also presented at the output to indicate the strength of the estimate. Pseudo code for the camera-based relative pose algorithm developed here is presented in Figure 51.

$$R_{image2vehicle} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{22}$$

$$R_{sensor} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \tag{23}$$

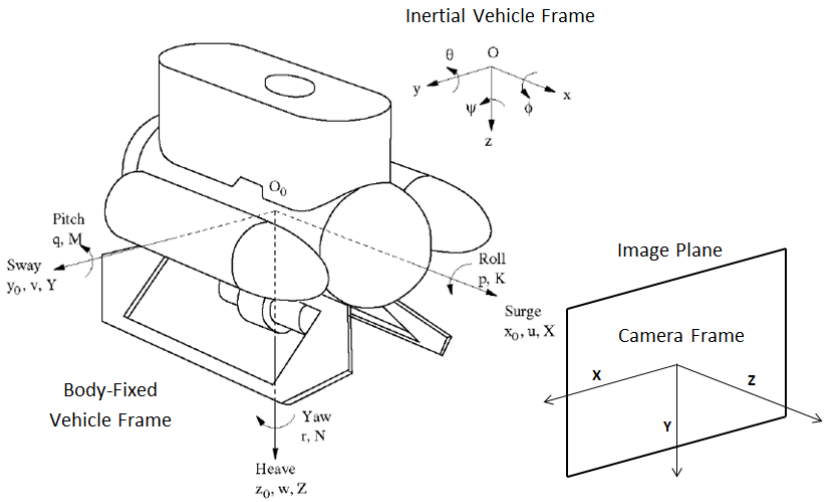


Figure 50: Vehicle and camera coordinate systems.

Pseudo code for camera-based relative pose algorithm main loop:

Capture camera image at time t_0 and t_1
Convert both images to grayscale
Apply CLAHE to both images
Remove lens distortion from both images (camera calibration)
Extract 1000 strongest Shi-Tomasi or SURF features in t_0 image
Extract 1000 strongest Shi-Tomasi or SURF features in t_1 image
Bouguet's pyramidal Lucas-Kanade optical flow matching of features between images
Estimate 5.5 DOF motion model essential matrix using Nister's five-point method and RANSAC
Decompose essential matrix into translation and rotation matrices using SVD
Convert relative pose estimate to vehicle-base coordinate system
Output estimated motion model and number of inliers

Figure 51: Pseudo code for camera-based relative pose algorithm

5.5 Algorithm Results

The vision-based relative pose estimation algorithm presented here was first evaluated using simulated camera data, due to the controllability and inherent ground truth available. These simulated video datasets provided a variety of vehicle trajectories for full evaluation of the algorithm. The camera-based relative pose estimation algorithm presented here provides a six degree-of-freedom (minus scale) motion estimate between frames. The first trajectories considered during testing were single degree-of-freedom trajectories with decoupled rotational and translational motion. A surge-only trajectory was used for evaluation of the translational estimation accuracy. In this case, the simulated camera sensor was commanded in a surge-only (directly forward) path for 600 meters. Between each frame, the simulated vehicle was moved forward a distance of one meter. The results of this testing can be seen in Figure 52 along with the ground truth. It can be seen that a small drift is accumulated (clearly visible in the y-direction), but represents an error of less than one percent of the distance travelled. In this case, the maximum accumulated x-error over 600 meters is measured at 7.24m, while the maximum accumulated y-error is measured at 5.75m. This corresponds to error percentages of 1.2% and 0.96% respectively. In contrast to INS systems, this accumulated error is positional in nature and does not affect future calculations. Any error drift in INS acceleration-based methods for position estimation result in velocity offset error, further resulting in a steady state position error, which effects all future calculations. This illustrates the benefit of a direct positional difference method over an integrated acceleration method in tracking motion or odometry.

In addition to the surge-only simulated dataset described above, an additional decoupled, single degree-of-freedom dataset was also used to evaluate the rotational estimation performance of the algorithm. In this case, yaw motion of the vehicle was simulated at a rate of one degree per frame, over a full 360 degree circle. Surge and sway translational motion was kept at zero during this rotational motion. The results from one of the simulated yaw-only camera datasets can be seen in Figure 53. In this case, it is clear that the

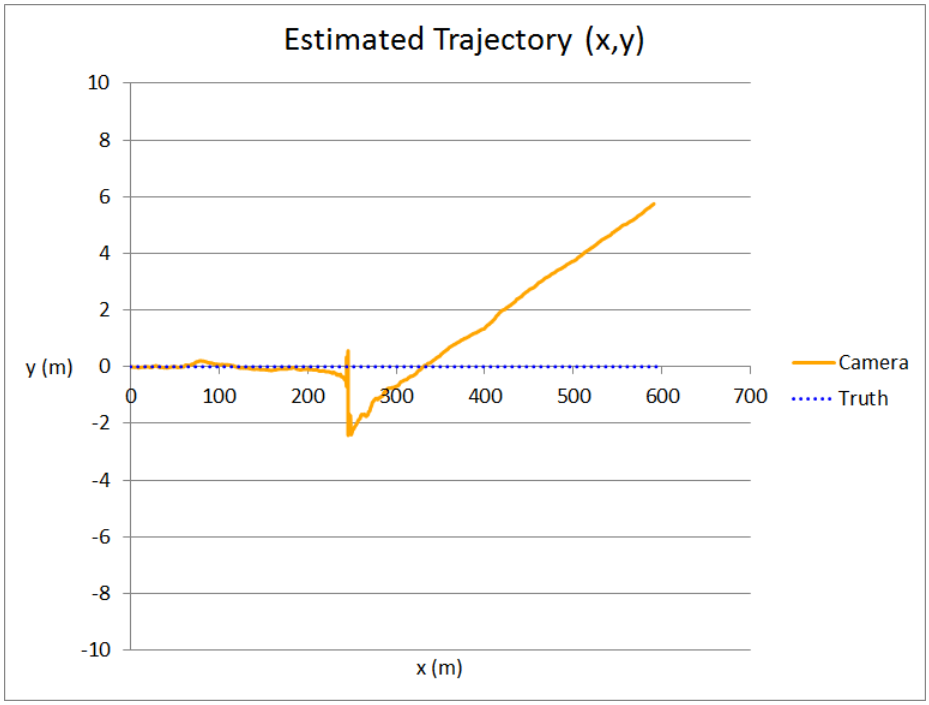


Figure 52: Translational (surge-only) estimated trajectory with ground truth

accumulated error rates are very low (less below one percent of the distance travelled). The maximum accumulated yaw error is calculated at 0.8 degrees, corresponding to 0.2% of the total 360 degree rotation. Qualitatively, it can be seen from Figure 53 that the accumulated yaw-estimate result tracks very well with the ground truth. These translational and rotational, decoupled, single degree-of-freedom evaluations show that the camera-based relative pose algorithm performs as expected, with sufficiently low error rates.

A final decoupled, simulated camera dataset was tested with the vision-based relative pose estimation algorithm before advancing to more complicated trajectory evaluations. In this case, the simulated vehicle was commanded in a piecewise sway- and surge-only translational trajectory to form a rectangular path. Yaw was again held constant to decouple the translational and rotational effects. The resultant estimate, along with the ground truth trajectory, can be seen in Figure 54. It is clear that the vehicle begins at the origin (0,0) and moves in the negative sway direction, then the negative surge direction, followed by the positive sway direction and then finally the positive surge direction to return to the

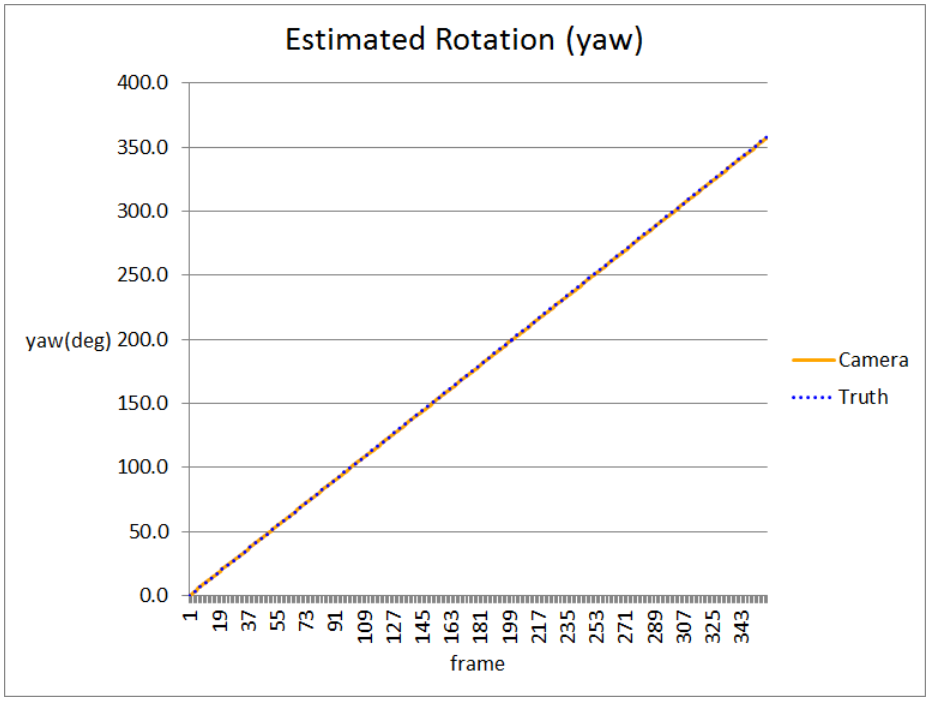


Figure 53: Rotational (yaw-only) estimated yaw with ground truth

origin. A slight accumulated error can be seen between the camera estimate and the ground truth in this figure. The maximum error calculated over this trajectory was 7.8 meters in the x-direction and 12.45 meters in the y-direction, over a distance of 600 meters. This corresponds to an accumulated error percentage of 1.3% and 2.1% of the distance travelled respectively.

The positive results from the evaluation of this algorithm over these three initial, decoupled trajectories proves the utility of the camera-based relative pose algorithm presented here. However, further evaluation is required using a dataset which contains coupled rotational and translational motion if the algorithm is to be used in realistic applications. Estimation of coupled rotational and translational motion presents a much more difficult problem, but such motion is commonly encountered when using real vehicle systems. In order to evaluate the performance of the algorithm presented here with coupled translational and rotational motion, an s-curve trajectory is used to create an additional simulated camera dataset. Over this s-curve trajectory, the vehicle follows alternating hemispherical

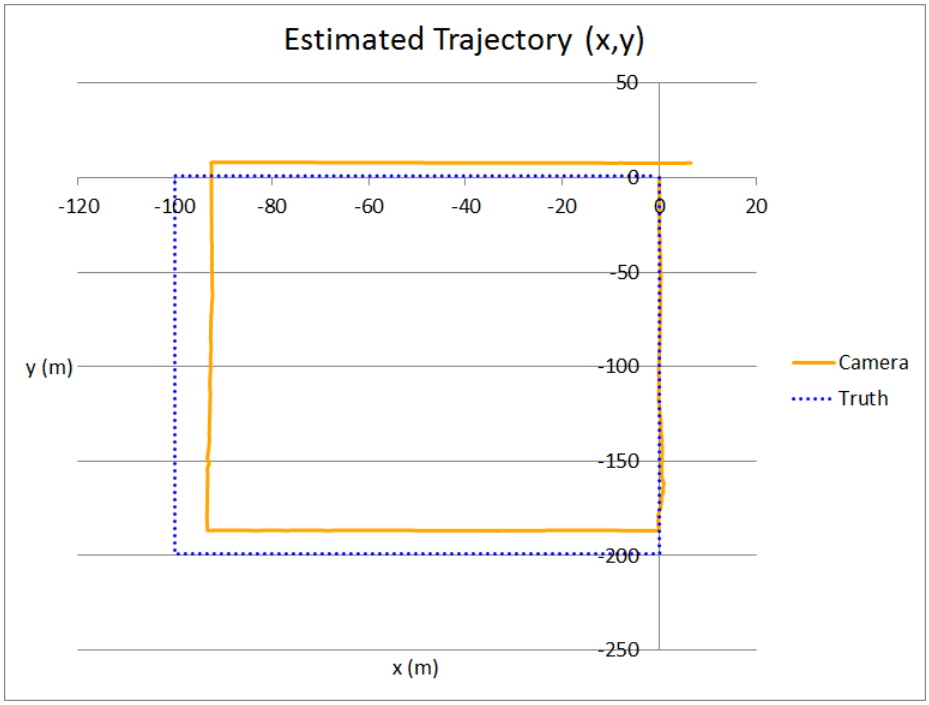


Figure 54: Rectangular (surge and sway) estimated trajectory with ground truth

paths, with surge and yaw motion between each frame. The ground truth s-curve path can be seen in Figure 55a along with the resultant output estimate from the vision-based relative pose algorithm, scaled from unit vector form for visual comparison. In this case it can be seen qualitatively that the estimated trajectory tracks well with the ground truth. While quantitative error values are not available for the translational results in this case, due to the camera scale factor ambiguity, the absolute rotational yaw error values can be used for algorithm validation. The estimated s-curve trajectory encounters a maximum rotational error of 2.12 degrees, corresponding to 0.3%, over the total 720 degrees of rotation (Figure 55b). A summary of all maximum accumulated errors and average frame-to-frame errors over all simulated camera dataset trajectories (detailed in Section 4.2) can be seen in Table 11 and 12 respectively.

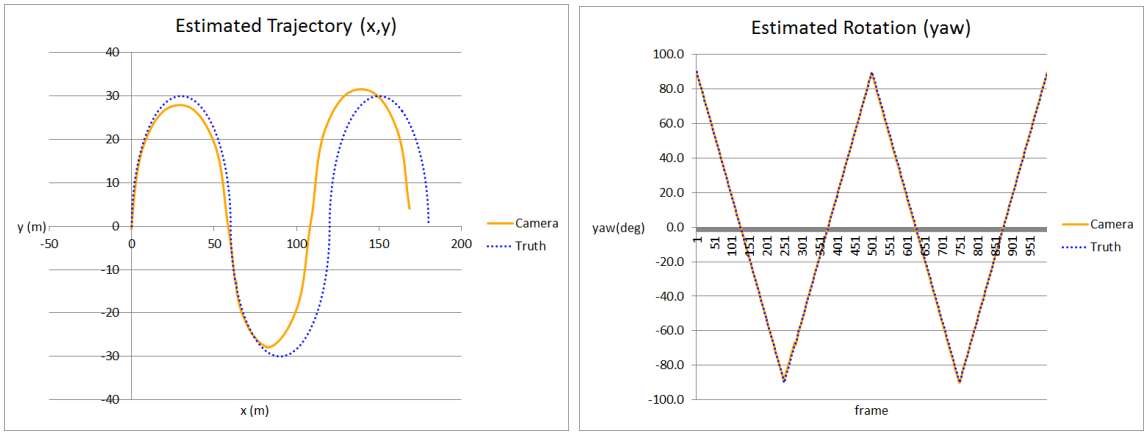
Once the algorithm performance was validated using simulated camera datasets, further evaluation was undertaken using real under-ice datasets taken in Colorado and Antarctica.

Table 11: Maximum accumulated error results (all simulated data)

Trajectory	**Ice type**	**Angle**	**x (%, m)**	**y (%, m)**	**ψ (%, °)**
Surge-sway translation (Fig. 54)	All	Upward	1.30%, 7.80m	2.08%, 12.45m	-
Yaw-only (Fig. 53)	Brash	Upward	-	-	0.22%, 0.80°
Yaw-only (Fig. 53)	First-year	Upward	-	-	0.39%, 1.40°
Yaw-only (Fig. 53)	Frazil	Upward	-	-	0.10%, 0.37°
Yaw-only (Fig. 53)	Multi-year	Upward	-	-	0.06%, 0.20°
Surge-only (Fig. 52)	All	Forward	1.21%, 7.24m	0.96%, 5.75m	-
Yaw-only (Fig. 53)	Brash	Forward	-	-	0.24%, 0.85°
Surge-only (Fig. 52)	Brash	Forward	0.04%, 0.07m	0.11%, 0.22m	-
Yaw-only (Fig. 53)	First-year	Forward	-	-	0.26%, 0.93°
Surge-only (Fig. 52)	First-year	Forward	0.07%, 0.14m	0.19%, 0.37m	-
Yaw-only (Fig. 53)	Frazil	Forward	-	-	0.30%, 1.09°
Surge-only (Fig. 52)	Frazil	Forward	0.01%, 0.02m	0.08%, 0.15m	-
S-curve (Fig. 55)	All	Forward	-	-	0.29%, 2.12°
Yaw-only (Fig. 53)	Multi-year	Forward	-	-	0.25%, 0.91°
Surge-only (Fig. 52)	Multi-year	Forward	16.22%, 32.43m	0.16%, 0.32m	-
Surge-only (Fig. 52)	All	Upward	11.08%, 66.47m	3.93%, 23.58m	-
S-curve (Fig. 55)	All	Upward	-	-	0.33%, 2.34°
Surge-sway translation (Fig. 54)	All	Upward	1.30%, 7.80m	2.08%, 12.45m	-

Table 12: Average frame-to-frame error results (all simulated data)

Trajectory	**Ice type**	**Angle**	**x (%, m)**	**y (%, m)**	**ψ (%, °)**
Surge-sway translation (Fig. 54)	All	Upward	3.00%, 0.03m	4.00%, 0.04m	-
Yaw-only (Fig. 53)	Brash	Upward	-	-	1.00%, 0.01°
Yaw-only (Fig. 53)	First-year	Upward	-	-	1.00%, 0.01°
Yaw-only (Fig. 53)	Frazil	Upward	-	-	0.00%, 0.00°
Yaw-only (Fig. 53)	Multi-year	Upward	-	-	0.00%, 0.00°
Surge-only (Fig. 52)	All	Forward	1.00%, 0.01m	3.00%, 0.03m	-
Yaw-only (Fig. 53)	Brash	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 52)	Brash	Forward	0.00%, 0.00m	1.00%, 0.01m	-
Yaw-only (Fig. 53)	First-year	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 52)	First-year	Forward	0.00%, 0.00m	2.00%, 0.02m	-
Yaw-only (Fig. 53)	Frazil	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 52)	Frazil	Forward	0.00%, 0.00m	1.00%, 0.01m	-
S-curve (Fig. 55)	All	Forward	-	-	2.78%, 0.02°
Yaw-only (Fig. 53)	Multi-year	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 52)	Multi-year	Forward	16.00%, 0.16m	1.00%, 0.01m	-
Surge-only (Fig. 52)	All	Upward	11.00%, 0.11m	8.00%, 0.08m	-
S-curve (Fig. 55)	All	Upward	-	-	4.17%, 0.03°
Surge-sway translation (Fig. 54)	All	Upward	3.00%, 0.03m	4.00%, 0.04m	-



(a) x,y trajectory (b) Yaw

Figure 55: S-curve (coupled rotation and translation) estimated trajectory with ground truth (left), as well as estimated yaw with ground truth (right)

These real datasets provide difficult benchmarks due to the disturbances of human operator control, and more realistic autonomous missions would present much smoother inputs with the ice consistently fixed in the frame. Compass and gyroscope data was used for ground truth in the yaw direction for the Colorado and Antarctica datasets respectively. Translational motion in the datasets was validated using approximated vehicle trajectories as qualitative ground truth. During rotational data collection, the vehicle was commanded to rotate around the z-axis (yaw) with minimal translation. The results for the estimated vehicle rotation, along with compass yaw ground truth, for one rotational Colorado dataset is shown in Figure 56. In this case, SURF feature-points were tracked between frames, providing an estimate that follows close to the ground truth. The maximum accumulated yaw error for this dataset is 107.29°, and the final yaw error is 32.08° over 550 total degrees of rotation. These error rates correspond to 19.51% and 5.83% respectively, and mostly result from periods of time when the ice is not visible in the frame. It is important to note that absolute camera translational trajectory estimates are shown for the simulation datasets above, despite the fact that these estimates are composed of a unit vector without scale. Using simulated datasets, the magnitude of the motion between each frame is known, which provides the ability to plot absolute trajectories. However, these magnitude values are not

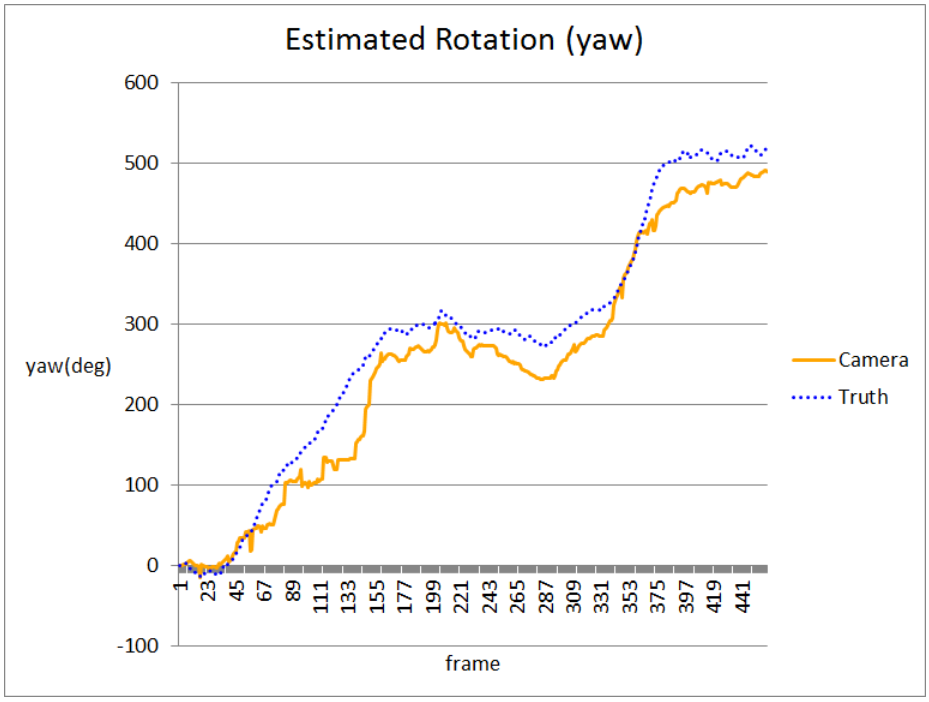


Figure 56: Yaw rotation estimates from the algorithm using SURF features along with compass ground truth data for an under-ice rotation run through 550 degrees

known in real-world datasets, and therefore the absolute translational trajectories cannot be plotted. Evaluation of the translational results, in these cases, is limited to qualitative analysis. However, absolute rotational information is estimated, and can be compared with the available ground truth for real-world validation of the algorithm. Qualitative analysis of the Colorado out-and-back translational dataset estimate, shown in Figure 57, shows that the vehicle encounters positive sway for the first half of the trajectory, followed by negative sway for the remainder, as expected with such a trajectory. In this case, magnitude in the plot is not significant, and it is only important to note that the plot shows an increase, followed by a decrease in accumulation over a symmetrical number of frames.

Translational ground truth is not available for the Antarctic datasets used here. Therefore, the yaw ground truth from the gyroscope and INS was used to validate the performance of the camera-based pose estimation algorithm presented in this dissertation. It can be seen from the estimated output plots in Figure 58 that this algorithm tracks well with the ground truth. The maximum accumulated error in this case is 70.98° and the final error

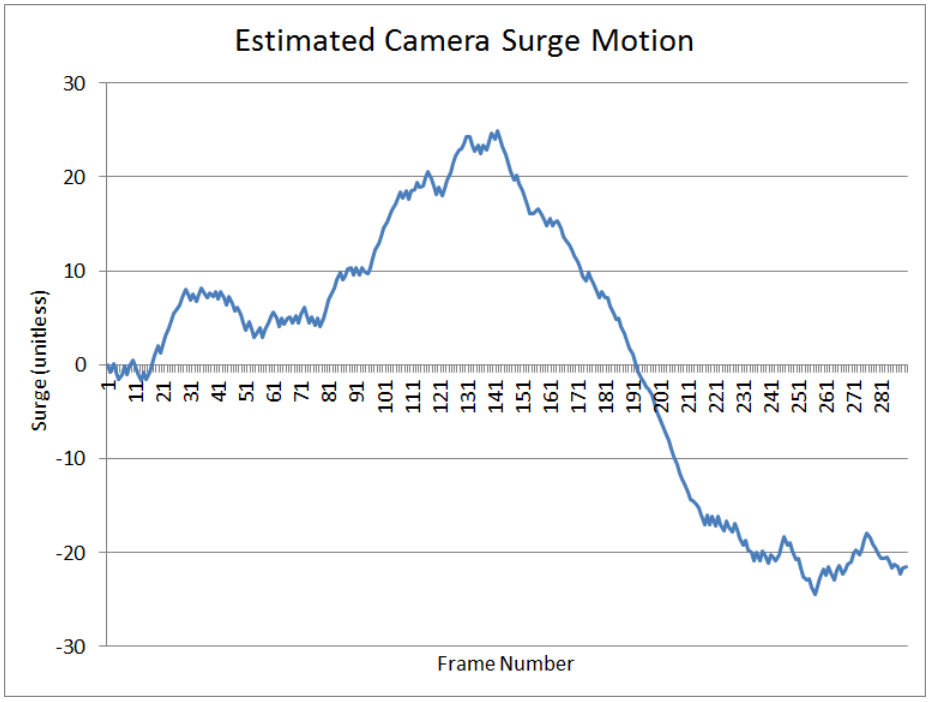


Figure 57: Surge motion estimate representation using a summation of surge unit vector components for a Colorado out-and-back trajectory

is 50.46°, corresponding to 26.29% and 18.69% respectively, over 270 total degrees. The majority of this accumulated error results from periods of time when the ice is not visible in the frame. These promising results using real-world, under-ice datasets proves the utility of this vision-based relative pose estimation algorithm, despite such low-contrast, feature-poor environments.

While translational ground truth was not available for the Antarctic datasets, qualitative analysis of the estimated trajectories can be performed. Figure 59 presents the estimated trajectory obtained with this algorithm using one of the Antarctic datasets, in which the vehicle was commanded in a mostly forward (surge) trajectory. This is apparent in Figure 59, as the vehicle maintains a mostly steady course over the estimated trajectory. Again, it should be noted that care must be taken during analysis of this plot, as it does not contain absolute position estimates, but instead an accumulation of unit vectors with the inherent assumption of constant magnitude. Another qualitative visualization of a mostly-surge trajectory from another Antarctic dataset can be seen in Figure 60. In this case, the magnitude

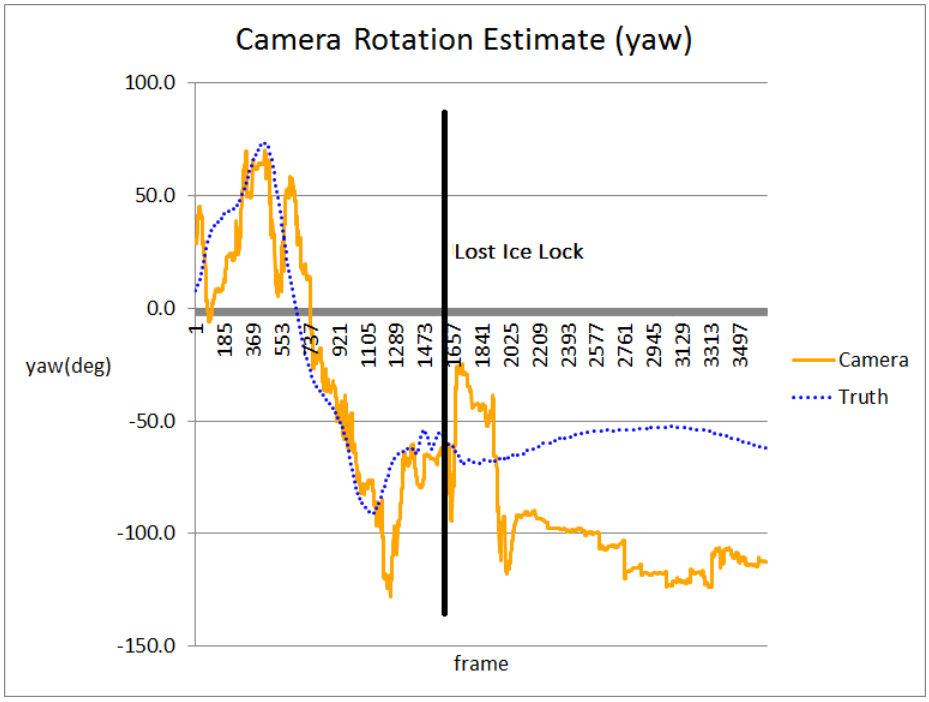


Figure 58: Yaw rotation estimates from the algorithm using SURF features along with compass ground truth data for an Antarctic under-ice rotation run over 270 degrees

of the plot is not of interest, as it is simply a summation of surge unit vector components over the trajectory. The important take-away from this plot is the positive slope of the surge accumulation value over the entire trajectory, as expected from the mostly positive surge trajectory considered here. Additional results can be found in Appendix A.

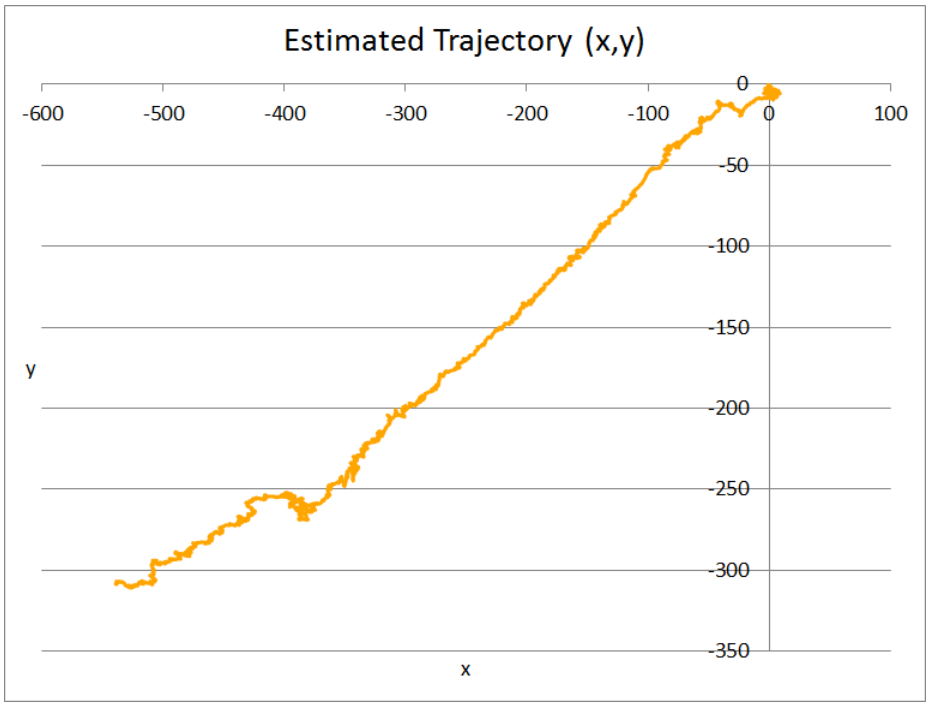


Figure 59: Estimated vehicle trajectory (modulo frame-to-frame unit vector scale) in an Antarctic dataset with mostly surge-based motion

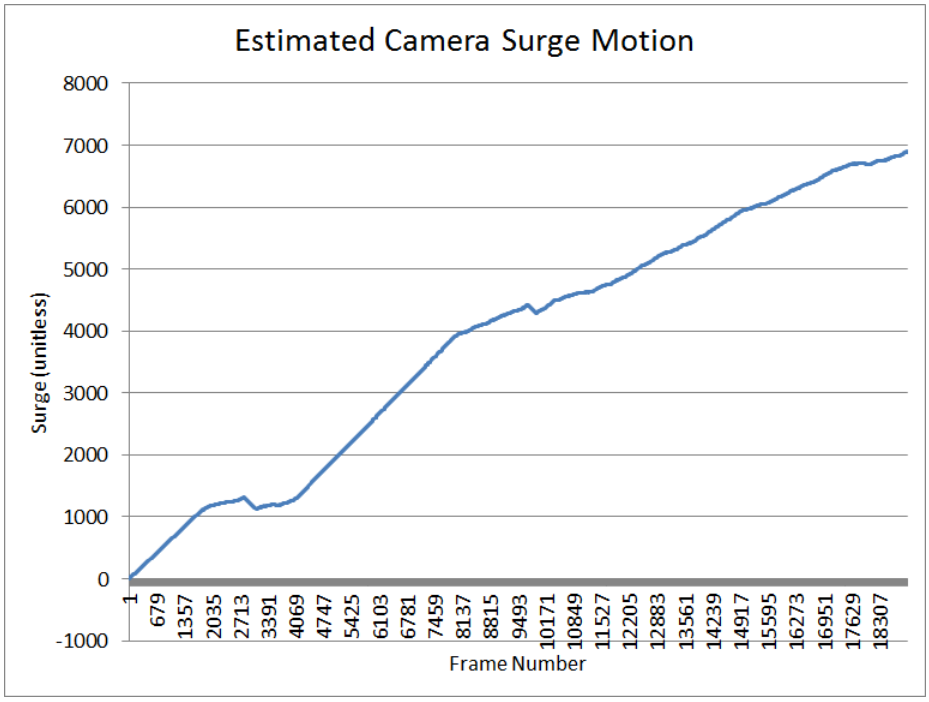


Figure 60: Surge motion estimate representation using a summation of surge unit vector components for an Antarctic positive surge trajectory

5.6 Summary

A vision-based relative pose estimation algorithm was presented in this chapter for use in low-contrast and feature-poor under-ice environments. This method utilizes a five-point algorithm for estimating relative camera pose between frames. Using adaptive contrast enhancement preprocessing and liberal feature-detection and matching thresholds on the front end, with a robust model estimation method on the back end, this relative pose algorithm, commonly used with feature-rich environments, can be adapted for application in the much harsher environments considered here. The algorithm is detailed in this chapter, along with evaluation results over simulated and real-world, under-ice datasets. In the next chapter, a novel method using the point-features, already extracted in the algorithm from this chapter, for sub-ice texture estimation and anomaly mapping, is presented. Both of these vision-based algorithms can provide enhanced capabilities for under-ice UUVs.

CHAPTER 6

TEXTURE AND ANOMALY MAPPING

6.1 Introduction

In this chapter, the focus moves from an under-ice navigation problem to an under-ice mapping problem, but remains consistent in the use of camera sensors. Much of the computational effort expended for vision-based relative pose estimation, such as point-feature extraction, can be leveraged for the equally important task of mapping the sub-ice environment. Searching for interesting features under the ice, including animals capable of sustaining life in such harsh environments, is of great interest in both polar (Antarctica) and planetary (Europa) domains. Underwater environments are known to be largely featureless, even at the seafloor which can many times consist only of monochrome sand or rock. Under-ice environments such as those encountered beneath the Antarctic ice shelves are even more devoid of features and tend to be monochromatic centered on the blues of the ice. With the advent of remotely operated underwater vehicles (ROVs) and autonomous underwater vehicles (AUVs) has come a large number of video datasets taken in these underwater and under-ice environments. Analyzing these videos can be tedious from both the perspective of human operators (ROVs) and scientists performing post-processing (AUVs), as most of the dataset contains no "interesting" information or unique features. However, all frames must be carefully analyzed to find the few frames of interest. A method is presented here for automatically highlighting a relatively feature-rich or uniquely colored frame to bring it to the attention of an analyst. If an estimate of the vehicle location is available, these frames or anomalies of interest can be mapped to create a global view of these anomalies over the vehicle's trajectory. Using point features (already extracted using the algorithm in the previous chapter running in parallel), a novel method for estimating the overall texture of the background ice in an image is presented here. Point features, along with image color histogram information, are also used to detect and map the ice texture and

any candidate anomalies detected in the globally feature- and color-poor under-ice datasets of interest. The methods developed here are evaluated using both simulated data of sub-ice environments and real under-ice data obtained from Lake John, Colorado and McMurdo, Antarctica.

6.2 Algorithms

6.2.1 Texture Estimation Algorithm

An estimation of texture in an image can give insight into the content of the image with a single quantitative value. Mapping the estimated texture of video frames over a large vehicle trajectory can indicate areas of interest during post processing, as well as give insight into the type and texture of background ice present over a wide area. Ice can vary drastically in texture from first year ice that can be only a few centimeters thick and very smooth with cracks, to multi-year ice that is smooth but varies greatly in topography, to platelet ice that is comprised of a textured mix of small ice fragments at the ice-water boundary, to drastically textured frazil ice comprised of half-formed crystals of ice at the ice-water boundary. A good example of the contrast between first-year and multi-year ice can be seen in Figure 9. In some sub-ice regions, multiple types of ice can be encountered during a single AUV mission. In Antarctica some under-ice locations encounter large currents where any frazil or platelet ice is swept away to leave only the smooth ice beneath, while other locations are sheltered from these currents and platelet ice can accumulate multiple meters in thickness. A mapping of estimated ice texture throughout under-ice mission video datasets can provide a global view of the ice types encountered over the mission. This mapping can also give scientists an idea of what to expect from a frame during post-analysis.

The algorithm presented here uses point-features already extracted from a video frame (during relative pose estimation) to calculate an estimate of ice texture. The model used here for a high texture value is the case of a large number of detected point features distributed evenly over the image plane. Given the point features extracted from the image,

Pseudo Code (Texture Estimation Loop):

```
Get input frame and initialize masks to all zeroes
Extract SURF and GFTT Feature Locations
Draw coverage circles for each feature (radius=r, value=1)
maskImg = and(SURF mask, GFTT mask)
pointCover = sum(maskImg)
pointCoverPct = pointCover/(maskImg.width*maskImg.height)
```

Figure 61: Pseudo code for main loop of texture estimation algorithm

the algorithm creates a coverage area for each pixel, which includes all pixels within a certain radius (r) of the point feature. A binary mask with the dimensions of the input image is then created from the summation of each of these individual coverage areas, where a pixel inside at least one coverage area takes the value of one, and any pixel not covered by any point feature coverage area takes the value of zero. From this binary mask, much information can be gleaned about the distribution of the point feature pixels. In the case of this algorithm, the total number of covered pixels can be easily and quickly determined with a summation over all pixels in the mask image. The ratio of covered pixels over total image pixels gives an estimate of texture which is normalized over image size. The pseudo code for the main loop of this algorithm is shown in Figure 61. Both Shi-Tomasi GFTT (Good Features to Track) [61] features and SURF (Speeded Up Robust Features) [64] features were used during the development of this algorithm. Shi-Tomasi features and SURF features each performed well on different datasets and seem to complement each other well in this application. A combined system using both Shi-Tomasi and SURF features is used in this algorithm to provide a more robust system.

To see how this algorithm obtains an estimate of ice texture, the reader should consider the following three examples. In the first case, a drastically textured image would have many point detections throughout the image plane that are fairly evenly distributed. This would lead to a very small amount of overlap between the coverage zones of the point features, and therefore result in a high value for total coverage and texture. In the case of an anomaly located in the middle of smooth and featureless ice, almost all point features

detected would be tightly grouped around the anomaly, with little coverage over the rest of the image. This would lead to a large amount of overlap between coverage areas, as the point-feature pixels are densely grouped, and therefore a small value would be obtained for a coverage and texture estimate. In the third case, mostly smooth ice with no anomalies would only have a few point-feature detections, which would be fairly evenly distributed. Despite the minimal overlap in coverage areas, the small number of features would lead to a low value for total coverage and texture. A high value for texture can only be obtained when a large number of features are detected, and those features are evenly distributed over the image plane. This ensures the algorithm is robust against a large number of densely grouped features, such as in the case of an anomaly in the ice. This algorithm provides the means for quickly calculating an estimate of the background texture in an under-ice image based on point features and their distribution over the image plane.

6.2.2 Point-Feature Anomaly Detection Algorithm

The distribution of detected point features in the image plane can not only give an idea about the texture over the entire image, but can also indicate the presence of an object of interest, or anomaly, in the image. Modelling such a small object or anomaly against a mostly featureless background as a tight grouping of point features in a small area is presented here. The algorithm developed here to detect such anomalies in a frame of video uses point-features, already extracted for relative pose estimation in the system, as input. The algorithm first determines groupings for the detected features using a k-means [124] based method which partitions the points (both Shi-Tomasi and SURF) into k clusters, where each point is part of the group with the nearest mean. The number of clusters (k) varies here between each frame and is calculated as in Equation 24, where n is the number of features detected and $N_{cluster}$ is a constant. Ideally, a closely grouped set of features (such as that corresponding to an anomaly) will form at least one group by itself due to the relative local density of the points. In contrast, a textured image with evenly distributed feature points will yield widely distributed groupings, and a mostly featureless image will

Pseudo Code (Texture-based Anomaly Detection Loop):

Get input frame
Extract SURF/GFTT feature locations
k-means clustering of SURF/GFTT features (k from Eq. 24)
$N_{fmin} = k * T_{cluster}$ (Eq. 25)
FOR i=0; i <k; i++ {
    $n_{box} = SUM$ (cluster[i] points within centroid box)
    IF $n_{box} > N_{fmin}$
        Anomaly detected }

Figure 62: Pseudo code for main loop of texture-based anomaly algorithm

have very few points per group where these points will be widely spread out. In order to determine if a group of extracted feature points form an anomaly or object of interest, the number of features within an S x S square pixel area, surrounding the group mean, is considered. In order to normalize the threshold of anomaly detection over the total number of features detected in the image, the ratio of feature points within the centroid square over the total number of group features is used as the metric for determining if a group comprises an anomaly in the image. In the algorithm, this threshold is represented by $T_{cluster}$, which requires the corresponding percentage of the total feature points in the group to be within that group's centroid box to be classified as an area of interest (Equation 25, where N_{fmin} is the minimum points for a feature, k is the number of clusters, and $T_{cluster}$ is the cluster threshold). In this manner, density of feature points is used to estimate whether an object of interest, or anomaly, is present in each video frame. The pseudo code for the main loop of this algorithm is shown in Figure 62.

$$k = n/N_{cluster} \tag{24}$$

$$N_{fmin} = k * T_{cluster} \tag{25}$$

6.2.3 Hue Mapping Algorithm

In addition to the use of point feature distribution as a means for detecting anomalies or areas of interest in a frame, a second method is presented here, which uses hue to find such anomalies. The model of hue distribution in an under-ice image used here consists

mainly of blues corresponding to the ice background (determined from experimental testing). Therefore, any pixel groupings outside of the main blue hues of the ice can be considered as anomaly candidates. This hue mapping algorithm first splits the input video frame into hue, saturation, and value channels, each consisting of a single matrix of 8 bit values with the same size as the input image. Hue represents the actual color of the image, while value represents brightness, and saturation represents color strength. The hue channel is the only one used in this algorithm as it provides information about the color of each pixel, which is used to find anomalous colors in the image. In theory, hue values take the form of degrees in a circle, and can range from zero back around to 360, both of which correspond to red (Figure 63). However, because 8 bit values (0:255) are used here, this spectrum is halved to fit between 0 and 179 and stay below the 255 maximum value.

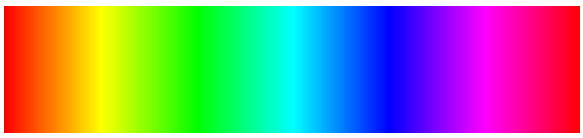


Figure 63: Hue spectrum which typically varies from 0 to 359, but here is divided by two to fit into an 8-bit representation, and so varies from 0 to 179.

An algorithm to obtain a single quantitative measurement for color in a video frame is presented here, which provides an indication of how much anomalous color is present in each frame throughout an under-ice video dataset. To obtain this value, each frame is first filtered to eliminate all pixels with a hue corresponding to the blue range ($T_{BL} - T_{BH}$ here). Then all remaining pixels are summed to obtain the total number of non-blue pixels in the frame. This value is normalized over the total number of pixels in the frame to obtain a percentage of pixels that are not within the blue band of hue values. The pseudo code for this algorithm is shown in Figure 64. During analysis, this value can be mapped over the dataset to provide indications of frames of interest for post-analysis located at local maximums, where an abnormal amount of non-blue color is present in the image.

Pseudo Code (Color Estimation Loop):
frame = Get input frame
threshBlueImg1 = THRESHOLD(frame, values above T_{BL})
threshBlueImg2 = THRESHOLD(frame, values below T_{BH})
threshBlueImg = $threshBlueImg1 + threshBlueImg2$
pixNotBlue = $SUM(threshBlueImg)/255$
pctNotBlue = $pixNotBlue/(frame.width * frame.height)$

Figure 64: Pseudo code for main loop of color estimation algorithm

6.2.4 Hue-based Anomaly Detection Algorithm

In addition to mapping a quantitative value for relative colorfulness of a video frame over a dataset, an algorithm which uses hue to detect anomalies in an under-ice image for mapping is also presented here. In this algorithm, a histogram is created from the input image over all possible values of hue with a bin width of one. The local maximums of the histogram are used as candidates for anomalous colors in the image, as the model used assumes a Gaussian-like distribution around at least one hue value for any color grouping in the image. This expected distribution can be seen in Figure 65 and Figure 66, where a large Gaussian-like distribution is centered on the blue hue values, and smaller Gaussian-like distributions are found outside the blue hues as candidates for anomalies.

Two normalized thresholds are used to eliminate hues with small and large histogram values, assumed to correspond to noise and background (ice or water) respectively. These thresholds are set at T_{HL} and T_{HH} to eliminate from consideration any hue local maximum that represents over T_{HL}% of the total pixels or under T_{HL}% of the total image pixels. In order to reduce the chance that local maximums corresponding to the blues of water or ice are considered to be anomalous, all hue values between T_{BL} and T_{BH} are ignored, in a form of band-stop filtering. The values of 0 and 179 (on a 180 scale) are also ignored, as they most often correspond to the whites and blacks in the image. While these values also correspond to the red hue, they can be safely ignored, as red is very uncommon in underwater imagery. Hues with fewer than T_{GL} or more than T_{GH} pixels in the image are also eliminated from anomaly consideration, and are assumed to correspond to noise and

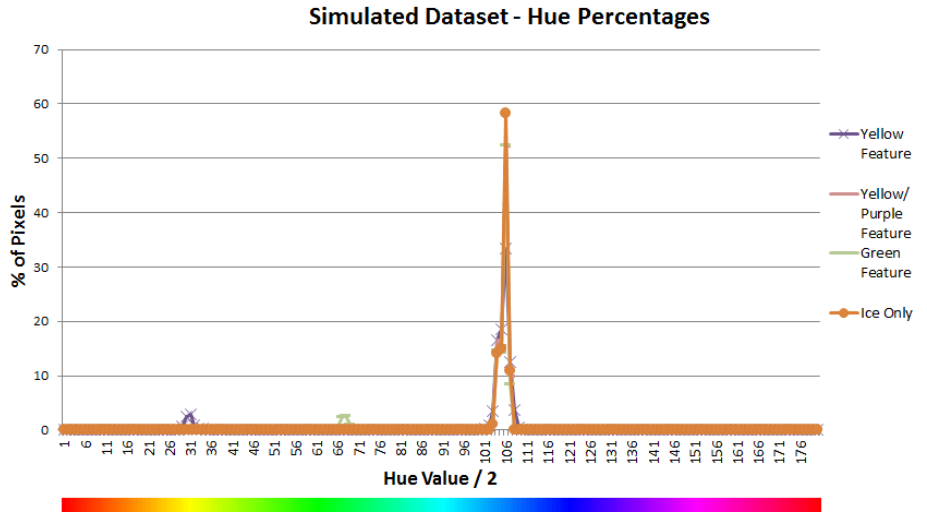


Figure 65: Histogram representing hue values for four simulated under-ice images. Most pixels are clustered around blue hue values (105) but two other much smaller groupings can be seen around yellow (30) and green (68) representing yellow sponges and green ice respectively.

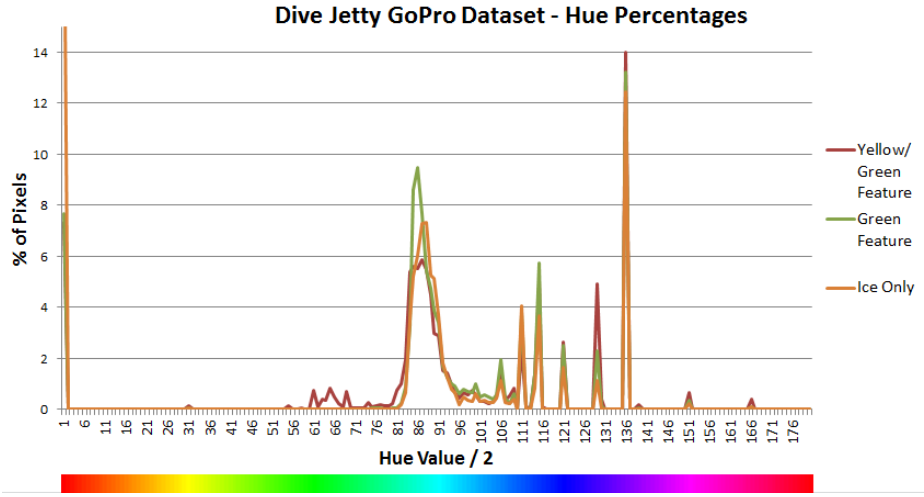


Figure 66: Histogram representing hue values for three real-world under-ice images. Most pixels are clustered around blue hue values (87) with some consistent false anomalies (115, 120, 130, 135, 150) that represent black pixels (ignored). Image with green ice and yellow Icefin vehicle has more pixels centered around green (57-80) and yellow (30). Image with green ice has slightly more green hue pixels. Image with only ice pixels is more centered around blue hues.

background respectively. The remaining local maximums in the hue histogram are used to add the corresponding hue pixels to an output mask, along with any pixels of the hue values on either side of the local maximum. The result of this algorithm is a mask of binary values the same size as the input image, where a value of one represents a pixel that is of a hue considered to be possibly anomalous and a zero represents background. The pixels of the output mask are then clustered into groups using a connected components method, where a connected component is considered to be a continuous group of touching pixels. A normalized threshold is used here to eliminate small (noise) groupings of pixels. This threshold, represented by T_A in Figure 67, is used to eliminate any pixel groupings with an area smaller than $T_A\%$ of the image size. Pseudo code for this algorithm is shown in Figure 67. The output result is a number of candidate components in the image that are considered anomalous in hue to the rest of the image, which mostly comprises of the blues of the ice. This provides a method for automatic detection and mapping of any ice anomalies, such as animals, present at the ice-water boundary.

The hue-based anomaly detection algorithm presented above uses only local maximums of hue pixel count in the calculations. However, this limits the amount of information obtained about a detected anomaly candidate, as only the pixels with hues at a local maximum and the two surrounding hue values are added to the output mask. Most anomaly objects contain a distribution of colors that is much more widely spread than a simple three-point local maximum. Therefore, during development of this hue-based anomaly detection algorithm, an additional method for anomaly detection was developed which considers all hues except those eliminated in the blue hue band described above, instead of only local maximums. The pseudo code for this algorithm is shown in Figure 68. This method results in more false detections, but gives much more detail about any candidate anomalies that are detected (size, shape, area). Both of these hue-based anomaly mapping algorithms are evaluated using the datasets described below.

Explicit quantitative values, obtained through experimentation, were applied to the

Pseudo Code (Local Max Hue-based Anomaly Detection):

frame = Get input frame
hueHistogram[180] = calculateHistogram(frame)
FOR i=0; $i < 180$; i++
    IF ($i < T_{BH}$) AND ($i > T_{BL}$)
        CONTINUE
    ELSE IF ($hueHistogram[i] > T_{GH}$)
        OR ($hueHistogram[i] < T_{GL}$)
            CONTINUE
    ELSE IF ($hueHistogram[i] > hueHistogram[i-1]$)
        AND ($hueHistogram[i] > hueHistogram[i+1]$)
            Add pixels in hueHistogram[i] to maskImg
dilate(maskImg, 4 times)
erode(maskImg, 4 times)
components[] = find connected components (threshBlueImg)
FOR i=0; $i < components.size()$; i++ {
    IF $component[i].area() > T_A$
        Anomaly detected }
Clear maskImg to all zeroes

Figure 67: Pseudo code (main loop) for local-max hue-based anomaly detection

Pseudo Code (Blue-thresh Hue-based Anomaly Detection):

frame = Get Frame
threshBlueImg1 = THRESHOLD(frame, values above T_{BH})
threshBlueImg1 = THRESHOLD(frame, values below T_{BL})
threshBlueImg = threshBlueImg1 + threshBlueImg2
threshBlueImg = dilate(threshBlueImg, 4 times)
threshBlueImg = erode(threshBlueImg, 4 times)
components[] = find connected components (threshBlueImg)
FOR i=0; i <components.size(); i++ {
    IF $component[i].area() > T_A$
        Anomaly detected }

Figure 68: Pseudo code (main loop) for blue-thresh, hue anomaly detection.

thresholds and constants introduced in this section during the evaluation of the algorithms presented here. A summary of these threshold and constant value assignments is presented in Table 13.

Table 13: Threshold and constant value assignments for the mapping algorithms determined through experimental testing

Constant	Value Assignment
r	20 pixels
$N_{cluster}$	20
S	30 pixels
$T_{cluster}$	0.8
T_{BL}	82
T_{BL}	151
T_{HL}	0.0001
T_{HH}	0.1
T_A	0.03

6.3 Algorithm Results

In order to evaluate these algorithms, simulated video data of under-ice environments (with inherent ground truth) was used, along with real under-ice data collected in Lake John, Colorado and McMurdo, Antarctica. Other real-world, under-ice animal imagery previously collected [12] is also used to further validate the algorithms on real data with anomalies. The simulated datasets provide the most accurate means for evaluation of the algorithms presented here, as they contain the most reliable ground truth. The main simulated dataset used for validation of the algorithms here is named the "Upward All-Ice Simulated Dataset". This simulated ice environment spans a very large area and contains regions of first-year ice, multi-year ice, frazil ice, and platelet ice. A ground truth map of this simulated ice environment is shown in Figure 69. Ice anomalies were introduced across the simulated ice in the form of ice deformations, ice cracks, and various colored animal shapes, similar to those encountered under the ice in Antarctica. The camera path was a translational rectangle of surge and sway (no rotational motion), where the vehicle returns

to the starting point at the end of the trajectory. Because an estimate (ground truth in this case) of the camera location for each frame is known, maps can be created over the trajectory to show estimated textures as well as the location of ice anomalies using the camera field of view and known altimetry (distance to ice). This dataset provides a very difficult benchmark for the algorithms presented here, as it contains a larger number and variety of anomalies and textures than would normally be encountered in a real-world under-ice data collection mission.

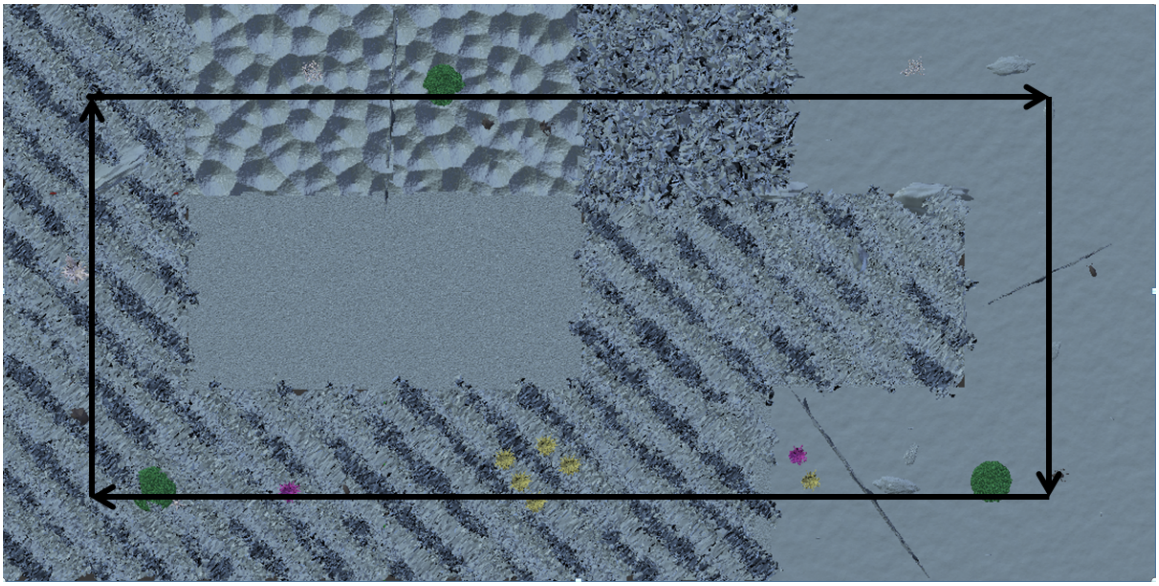


Figure 69: Ground truth map of the main simulated under-ice dataset and vehicle trajectory. This view is looking up from beneath the ice at the entire environment with the various ice types and ice anomalies.

6.3.1 Texture Estimation Results

The results from evaluation of the texture estimation algorithm on simulated under-ice data are presented in Table 14. In this table, the estimated texture values correspond to the percent point cover of the algorithm, and therefore provide an estimate of how textured an image is (high values correspond to highly textured backgrounds). Many representative single-frame examples can be seen in Figure 70 and Figure 71. In these figures, the left column shows the original image, with point feature detections as colored dot groups, while the right column shows a visualization of the point cover result mask, used to estimate

Table 14: Texture estimation results and relative rankings (1 = most textured)

Platform	Image Contents	Estimated Texture (%)	Expected Relative Rank	Estimated Relative Rank
Simulated	First-year Ice	0.179	5	5
	Voronoi Ice	48.038	3	3
	Platelet Ice	48.647	2	2
	Frazil Ice	94.551	1	1
	First-year ice with features	8.331	4	4
	Front camera, platelet and green ice	16.709	-	-
SCINI	Multi-year ice, Anemones	12.840	-	-
GoPro	Jetty Ice, Platelet ice and green ice	24.914	1 (tied)	1 (tied)
	Jetty Ice, Platelet/green ice, Icefin	2.313	3	3
	Platelet ice and green ice	25.640	1 (tied)	1 (tied)
	Platelet ice	24.996	1 (tied)	1 (tied)
	Platelet ice and green ice	24.996	1 (tied)	1 (tied)
	Platelet ice, green ice, Icefin	9.739	2	2
Icefin Front	Jetty Ice, Nothing just blue	12.423	4	4
	Jetty Ice, Large ice crack	16.341	2	3
	Jetty Ice, Ice crack and ice hole	23.442	1	1
	Jetty Ice, Ice crack	21.025	3	2

texture. A point cover image with more white pixels corresponds to a highly textured image while a point cover image with mostly black pixels corresponds to an image with a low texture value. As expected, the first-year ice, shown in Figure 70a, yields only a few, closely grouped point features, resulting in a low value for point cover (0.179%). At the other end of the spectrum, the frazil ice image (Figure 70g) yields many point-feature detections distributed evenly over the image, leading to a point cover image of mostly white pixels (94.55%), which correctly corresponds to a highly textured image. In between these two extremes are the multi-year ice (Figure 70c) and platelet ice (Figure 70e) images which have point cover values somewhere in the middle (48% for both). It can be seen from Table 14 that the sample simulated images are correctly ranked by estimated texture values. Simulation data (Figure 71a and Figure 71b) with this algorithm yields very similar results to real data obtained from under the ice in Antarctica (Figure 71c and Figure 71d).

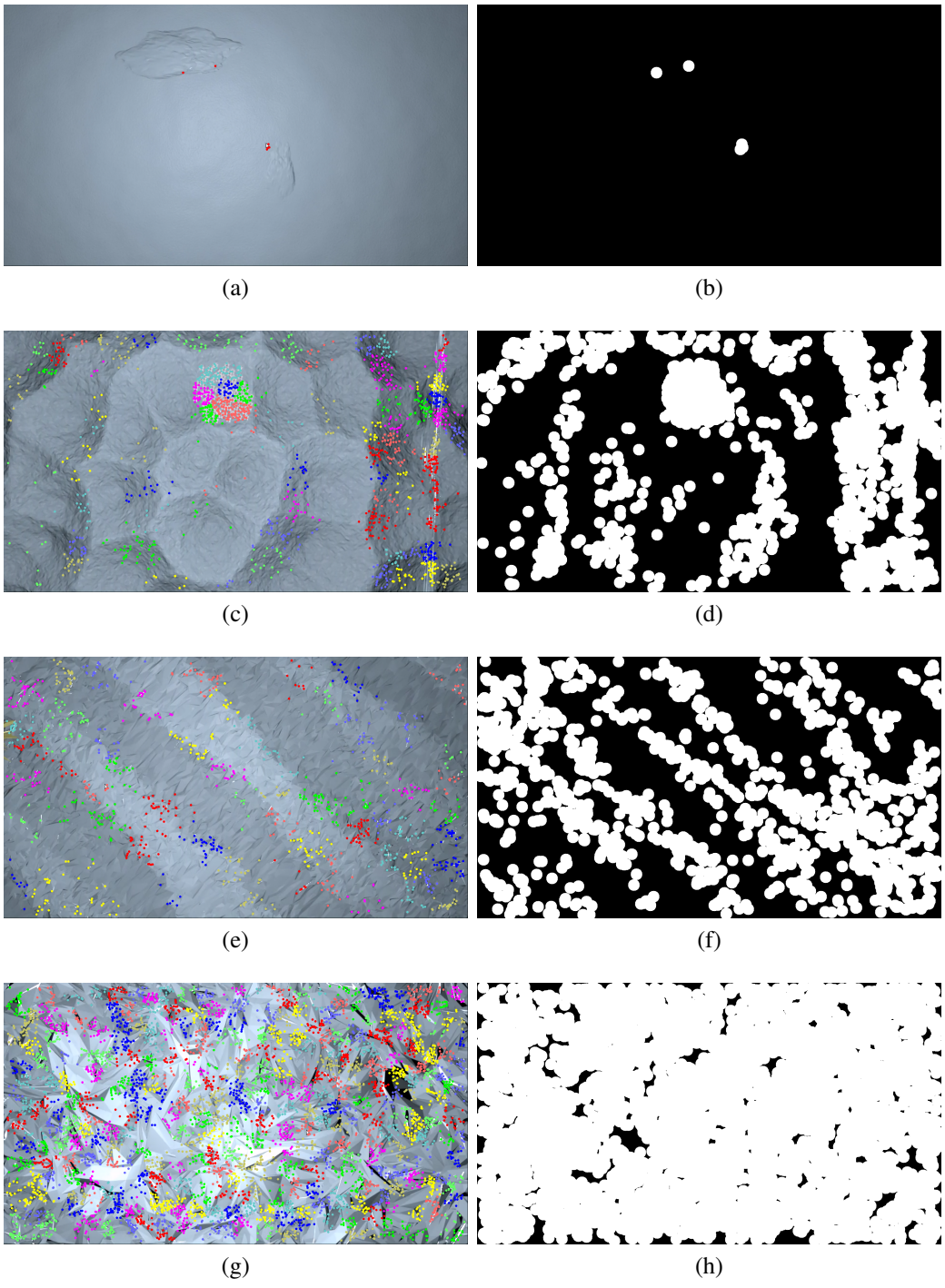
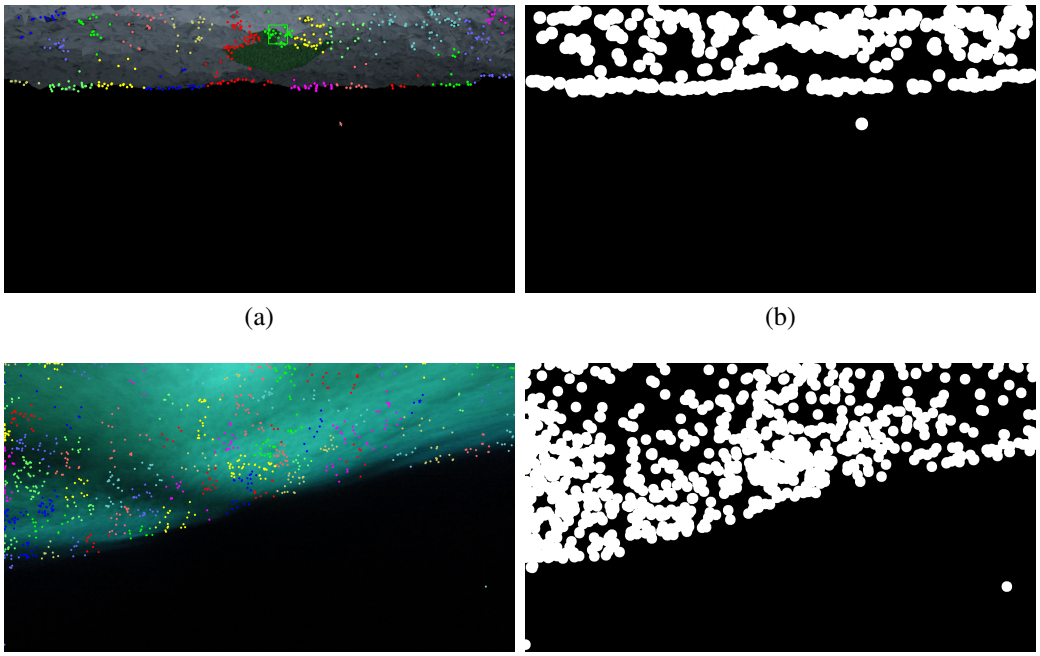


Figure 70: Results from texture estimation algorithm with the simulated under-ice dataset showing each ice type (a-b: first-year, c-d: multi-year, e-f: platelet, g-h: frazil). On the left are input images with point feature detections as dots with color determined by k-means grouping. On the right are visual representations of the point cover mask used to calculate estimated texture.



(a) (b)

(c) (d)

Figure 71: Results from the texture estimation algorithm with the simulated under-ice dataset (a-b) compared to real-world results (c-d). On the left are the input images with point feature detections as dots with color determined by k-means grouping. On the right is a visual representation of the point cover mask used to calculate estimated texture.

The resultant texture estimation map of the simulated data over the vehicle trajectory is shown in Figure 72, where white represents more textured areas and black represents no texture. The quantitative value for estimated texture can be visualized in the point cover images (the right side of Figure 70) as the percentage of non-zero valued pixels (white pixels here). It can be seen from this output map that the expected texture is obtained with the algorithm for each type of ice: platelet ice is on the left hand side, the top left corner, and the bottom left corner and shows a medium texture value (gray); multi-year ice is on the top to the right of the platelet ice and shows a much lower texture; frazil ice is on the top to the right of the multi-year ice and is the most textured (white); first-year ice is located from the end of the frazil ice clockwise until reaching the platelet ice on the bottom left corner, and shows a very low value for texture (black), as expected.

The texture estimation results can also be represented versus time (instead of x, y position) over the dataset duration (Figure 73), which does not require an estimate of the vehicle

Figure 72: Texture estimation using point features. The vehicle trajectory is plotted along with the estimated texture at each point. Here the texture measurement is represented as a scale from white (most textured) to black (least textured) with a gray scale in between. It can be seen that the most textured points in this case are located at the frazil ice while the least textured points are located at the first-year ice, as expected. Ground truth can be seen in Figure 69.

location. From this representation it can be seen that the global maximum is located around frame 120, which corresponds to the frazil ice, while the frames corresponding to first-year ice (frames 150-350) are at the global minimum - as expected. Local maximums (i.e. frames 240, 410) mostly correspond to frames with features (cracks, animals) in the ice. There is a single frame of incorrect estimation (frame 217) in the first-year data resulting from many false feature detections. In areas where there are sharp transitions between ice types (i.e. between frames 120 and 160), the estimated texture value makes a linear translation, as would be expected. The texture estimation algorithm estimates texture values consistent with the corresponding ice types for 98.9% of the "Upward All-ice Simulated Dataset". Table 15 presents a summary of these texture estimation results. The bad frame estimates in this case include frames with ice deformations that are not detected due to their smooth transitions (which is acceptable), as well as the single frame of false feature detections in the first-year ice. The algorithm is successfully able to produce higher texture

Table 15: Texture estimation results for upward all ice dataset

Ice type	Number of frames	% Correct texture estimate
Brash	250	100
Frazil	25	100
First-year	200	97
Multi-year	75	100

estimates from the more evenly distributed point sets. This can be seen between the image of simulated platelet ice only in Figure 70f (232181 pixel cover / 469 SURF points = 495 pixels / point spread) and the image of simulated first-year ice with features in Figure 75b (42664 pixel cover / 132 SURF points = 323 pixels / point spread). Here the spread of each point-feature is much greater for the evenly distributed points in the platelet ice, as expected, due to the limited overlap between point covers.

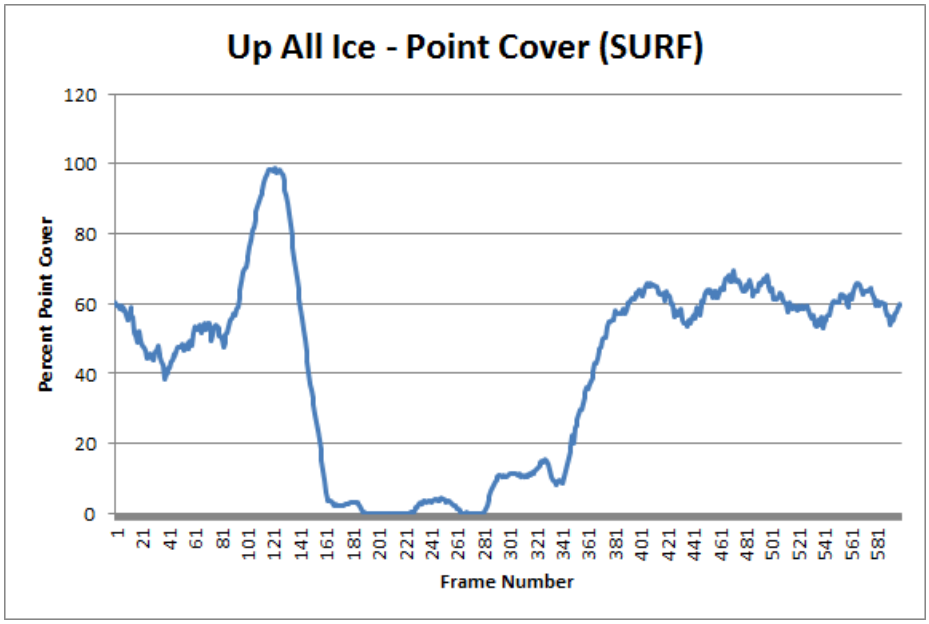


Figure 73: Texture estimation results plotted versus frame number for the simulated all ice dataset shown in Figure 69. As expected, the global maximum (approx. frame 120) corresponds to the most textured frazil ice, while the global minimums (between frames 150-350) correspond to the least textured first-year ice.

The texture estimation algorithm was also evaluated using other simulated data with similar results, as well as on real under-ice imagery obtained under the ice near McMurdo,

Antarctica. Results from this testing is shown in Table 14. In this case, the algorithm uses a combination of Shi-Tomasi and SURF point-features for more robust results and better comparison across datasets. It can be seen from this table that the four types of ice simulated (first-year, multi-year, platelet, and frazil) are correctly ranked by texture value. It can also be seen that in the "Icefin Dive Jetty Dataset", the frame with nothing but blue water has an estimated texture of only 42935 pixels, while any image with ice present has a much higher value (at least 53256 pixels). In the image where an ice hole and ice crack feature are visible, the texture estimate is almost double (81014 pixels) that of the empty frame. The GoPro Jetty dataset shows consistent texture estimation across all images, except those with the Icefin vehicle present in the frame, as expected. This algorithm shows promising results in attempting to estimate ice texture throughout an under-ice video dataset.

6.3.2 Texture-based Anomaly Detection Results

The texture-based anomaly mapping algorithm using point feature methods presented here was first tested on the simulated all ice dataset described above, given the inherent ground truth benefits. This dataset represents a large simulated under-ice environment comprised of multiple sections of different types of ice (platelet, multi-year, frazil, and first-year) as well as anomalies in the ice (cracks, deformations, and animals of multiple colors seen under the ice in Antarctica). The location of the camera throughout the trajectory is known, so the locations of detected objects can be used to create a map using the camera field of view and the altimetry (distance to the ice). The results from this algorithm over the simulated all ice dataset can be seen in the form of the mapped positions of detected ice anomalies (seen as blue blobs) in Figure 74. The ground truth map can be seen in Figure 69.

The five yellow sponges (bottom center), green ice (bottom right, bottom left, top middle), and anemones (top mid-left) were detected successfully as well as cracks (long straight features in lower right quadrant) and ice deformations (right side) that cannot be detected

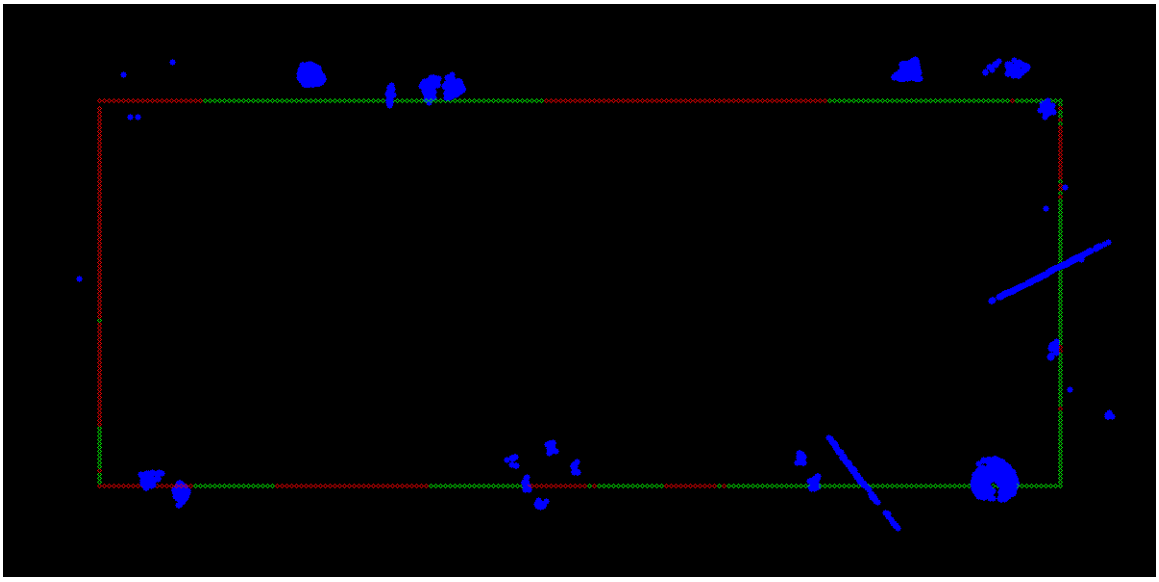


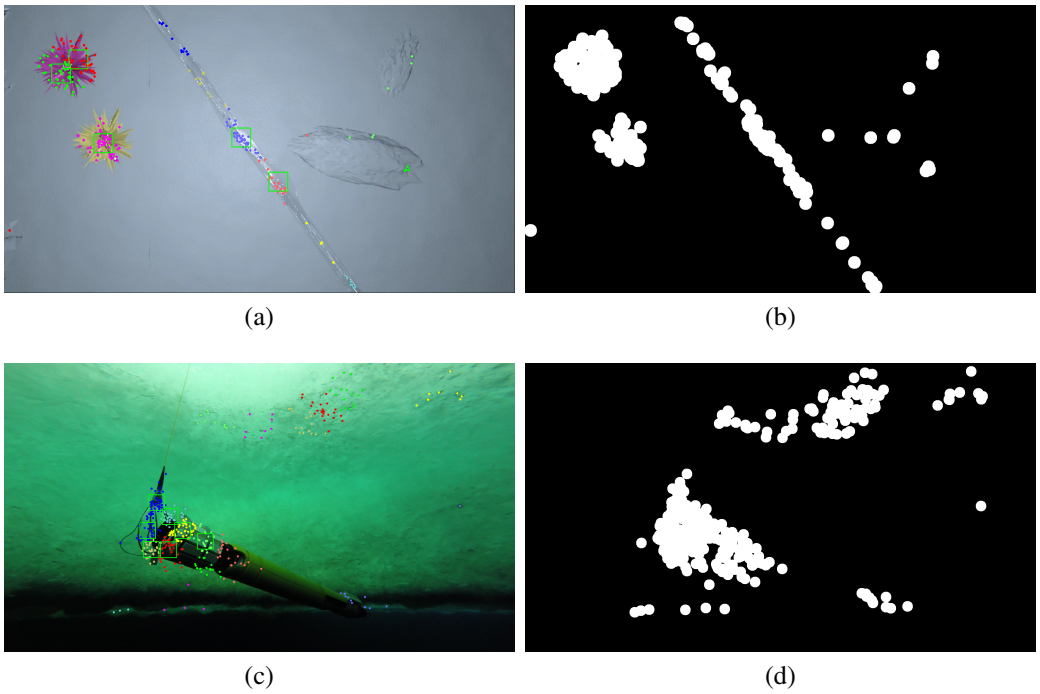
Figure 74: Resultant map from the point-feature-based under-ice anomaly detection algorithm. Here 20 of 24 anomalies were detected (six misses) with only six false frame detections over the 600 frame dataset.

using color-based methods. Overall, 20 of the 24 anomalies were successfully detected (four misses for a 16% miss rate) with only six false frame detections over the 600 frame dataset (1% false detection error). All misses represent anomalies of similar texture to the background ice, showing one benefit of the fused use of both a texture and color detection method. An example anomaly detection frame can be seen in Figure 75a with the detection of both colorful features (yellow and purple anemones) as well as an ice crack feature. Local maximums in the estimated texture plot (Figure 73) (i.e. frame 240 crack in the ice; frame 410 five yellow anemones) mostly correspond to frames with anomalies in the ice, and can be used to indicate frames of interest in post-analysis.

This algorithm was also tested on other simulation and real-world data with similarly promising results, as summarized in Table 16. Examples of successful detections, successful empty frames, and missed detections are presented in this table across multiple datasets. In the dataset obtained by the Icefin vehicle under the sea ice in Antarctica, there are many false anomaly detections of the vehicle's structure (seen in Figure 76a), where there are anomaly detections despite a mostly blue frame. However, these false detections can be

Table 16: Texture-based anomaly detection results

Platform	Image Contents	Qualitative Results	Frame count
Simulated	Platelet ice	Success no detections	34
	Platelet ice and green ice	Success green ice	46
	Platelet ice and 5 sponges	Fail misses sponges	31
	FY ice, anemone group	Success - anemones	31
	FY ice, ice features, sponges	Success - crack, sponges	24
	FY ice, green ice, shrimp	Success - shrimp, green ice	33
	FY ice, ice crack, sea slug	Success - slug, crack	24
	Frazil ice	Success no detections	36
	MY ice, crack, anemones	Fail misses anemones	35
SCINI	MY ice and anemones	Success - detects anemones	1
GoPro in Jetty	Platelet ice, green ice	Success no detections	1
	Platelet ice, green ice, Icefin	Success - detects Icefin	1
	Platelet ice, green ice, tether	Success - no detections	1
	Platelet ice	Success - no detections	1
	Icefin, green ice background	Success - detects Icefin	1
Icefin in Jetty	Blue frame	Success no detections	1
	Jetty platelet ice	Success no detections	1
	Platelet ice, small ice crack	Fail -misses small ice crack	1
	Platelet ice, ice crack	Success - detects ice crack	1
	Platelet ice, large ice hole	Success - detects ice hole	1
	Platelet ice, spec feature	Success - detects spec	1
	Platelet ice, small ice hole	Fail misses small ice hole	1
	Platelet ice, ice crack	Success - detects ice crack	1
	Platelet ice, ice crack/hole	Fail - misses ice crack	1
	Platelet ice, large ice hole	Success - detects ice hole	1
	Platelet ice, ice hole	Success - detects ice hole	1



(a) (b)

(c) (d)

Figure 75: Results from the texture estimation algorithm (right) and feature-based anomaly detection algorithm (left). It can be seen that in both the simulated (a) and real (c) datasets, anomalies are successfully detected (green boxes).

easily ignored as the location of the vehicle structure in the image is constant and can be masked out. True detections using the algorithm can also be seen, including specs in front of the camera, ice cracks (Figure 76c) and holes in the ice.

In the GoPro dataset taken under the sea ice in McMurdo, Antarctica, the only texture anomaly feature present was the Icefin vehicle itself, which is successfully detected in Figure 75c. Images from this dataset with only the small diameter tether in view yield clustered point groupings around the tether, indicating an object that is difficult to see with human perception, but more easily found in post-processed images. It is interesting to point out that while the patches of green ice are considered ice anomalies here, they are not detected with the texture-based algorithm, as they are the same texture as the background ice surroundings. This indicates the benefit of a fusion of color- and texture-based detection methods. The images from this Antarctica GoPro dataset that do not have anomalies present yield no detections despite the large number of point features, as the features are

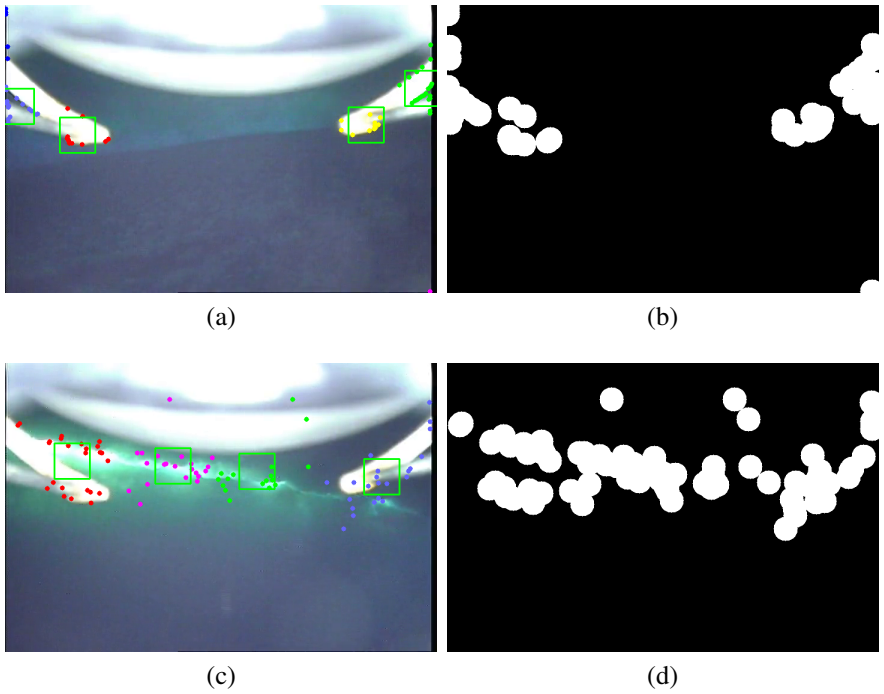


Figure 76: Results from the texture estimation (right) and feature-based anomaly detection (left) algorithms. It can be seen that only the vehicle body is detected in an empty frame (a) while an ice crack features is detected in (c).

evenly distributed and not sufficiently dense to indicate an anomaly. This is the desired behavior of the algorithm in such cases, as it is able to avoid false detections in highly textured images, which cannot be accomplished with a simple point-feature count for anomaly detection. The missed detection failures encountered in the simulated datasets result from similarity between textures of the anomalies and the ice background, which presents another motivation for a fused texture- and hue-based detection algorithm. Most of the missed detection failures encountered in the Icefin Antarctica dataset result from anomalies that are too small to accumulate a sufficient grouping of point features.

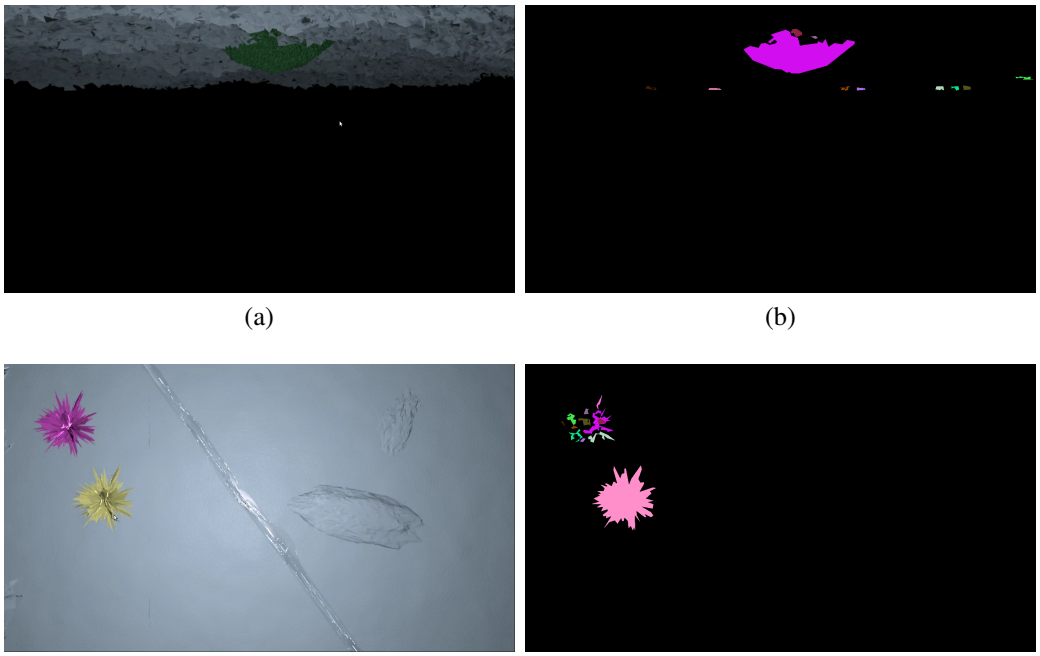
6.3.3 Hue-based Anomaly Detection Results

The hue-based algorithms presented here were also evaluated using the same simulated and real-world under-ice datasets entailed above for the texture-based algorithms. The upward all-ice simulated dataset was again used for primary validation of the algorithm, as ground truth was known with the highest accuracy. Table 17 presents a summary of the results

Table 17: Hue-based anomaly detection results for upward all ice dataset

Method	Correct detections (#, %)	False detections (frames, %)
Local maximum	20 anomalies, 100%	4 frames, 0.67%
Blue threshold	20 anomalies, 100%	42 frames, 7%

of the hue anomaly detection algorithms on this dataset. An example output of the blue-thresholding and component-grouping algorithm is shown in Figure 77 and Figure 79. In Figure 77b, the main detection in the image is of the green ice, along with other small false detections at the edge of the ice (which are filtered out based on their small size). In Figure 77d, the yellow sponge anomaly is detected completely, while the purple sponge yields multiple fragmented detections due to the close proximity of the color purple to the blue hue band. However, both colorful sponge anomalies are successfully detected, while the ice features of blue color are not - as expected from a hue-based algorithm. This provides an argument for a texture and color fusion method for anomaly detection, as these ice features are detected using the texture-based method. The mapped results of the hue-based anomaly detection algorithm indicate highlighted objects of interest over the trajectory, given an estimate of the vehicle position over time as well as the distance to the ice and the camera field of view. The simulation datasets contain ground truth for these values, and thus the estimated output map for the upward all-ice simulated dataset can be seen in Figure 78, where the trajectory is plotted in green (anomaly detected) and red (nothing detected) and objects of interest are marked with blue blobs. In this case, the green ice and the anemone group anomalies are successfully detected and mapped. It can be seen from comparing the output maps (Figure 78) to the ground truth map (Figure 69) that all 20 non-blue anomalies are detected with zero misses for both local-maximum and blue-thresholding methods. One false detection was found with the local-maximum method, while ten false detections (eight of which were in the frazil ice) were found with the blue-thresholding method. Over the 600 frames of the simulated dataset, the local maximum method was 88.5 percent accurate



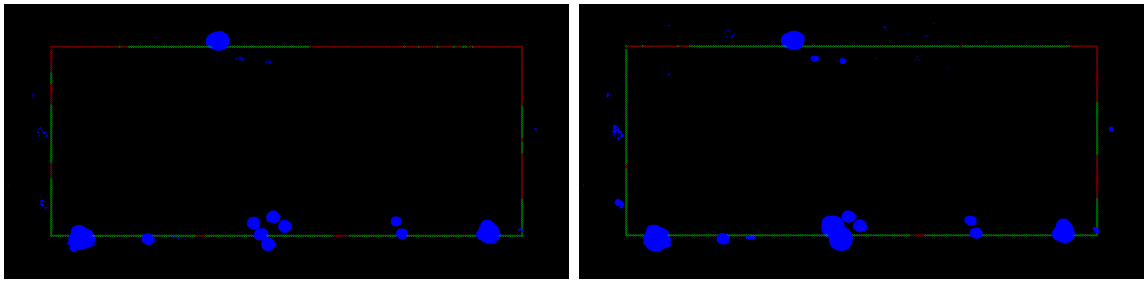
(a) (b)

(c) (d)

Figure 77: Results from the blue thresholding hue-based anomaly detection algorithm with simulated under-ice data. Both green ice anomalies (a-b) and yellow/purple sponge anomalies (c-d) are successfully detected.

in detection of any colorful anomaly in the frame. Specifically, 27 frames encountered misses and 42 frames encountered false detections. These results are summarized in Table 17.

While the simulated datasets provided reliable ground truth for evaluating the accuracy of the algorithm, further insight was gained with the evaluation of the algorithm on real-world data obtained from beneath the ice near McMurdo station. An image from the SCINI dataset [12] of anemones present against the ice shelf resulted in the detection of six of the nine anemones in the foreground (Figure 79a-79b). An image containing the Icefin vehicle and green ice as anomalies from the GoPro Jetty dataset shows correct detections of both the Icefin vehicle and the patch of green ice (Figure 79c–79d). An image from the Icefin Jetty dataset with an ice-hole anomaly is shown in Figure 79e–79f, where the hole itself was not detected, but instead the green ice around it. In this image, the false detection of the Icefin vehicle frame can be seen. This is masked out, as the location is known and constant



(a) (b)

Figure 78: Output map result from the hue-based anomaly detection algorithm where the blue blobs represent detected anomaly candidates (Left: Local maximum method, Right: Blue threshold method). Using the local maximum method, 100% of the 20 non-ice anomalies were detected with one false detection. Using the blue thresholding method, 100% of the 20 non-ice anomalies were detected with ten false detections (eight of which are in the frazil ice).

Table 18: Hue-based color estimation results (local maximums) for upward all ice dataset

Local maximum frame range with anomaly present	**Contents**	**Found anomaly?**
50 – 100	Green ice	Yes - correct
280 – 330	Green ice	Yes - correct
330 – 360	Two yellow/purple sponges	Yes - correct
400 – 420	Five yellow sponges	Yes - correct
440 – 460	Single purple sponge	Yes - correct
460 – 520	Green ice	Yes - correct

over time.

6.3.4 Hue-based color estimation results

In addition to the hue-based filtering and clustering method for anomaly detection in a frame, presented in the previous section, a quantitative value for percent “interesting” color (i.e. non-ice) is obtained by another algorithm, for each frame over a dataset. When these values are mapped over the vehicle trajectory, local maximums can be used during post processing to indicate candidate frames of interest to examine for anomalies. Mapping of these non-blue hue percentage values can give insight into the color contents of the video frames, and therefore indicate areas of interest over the vehicle’s trajectory.

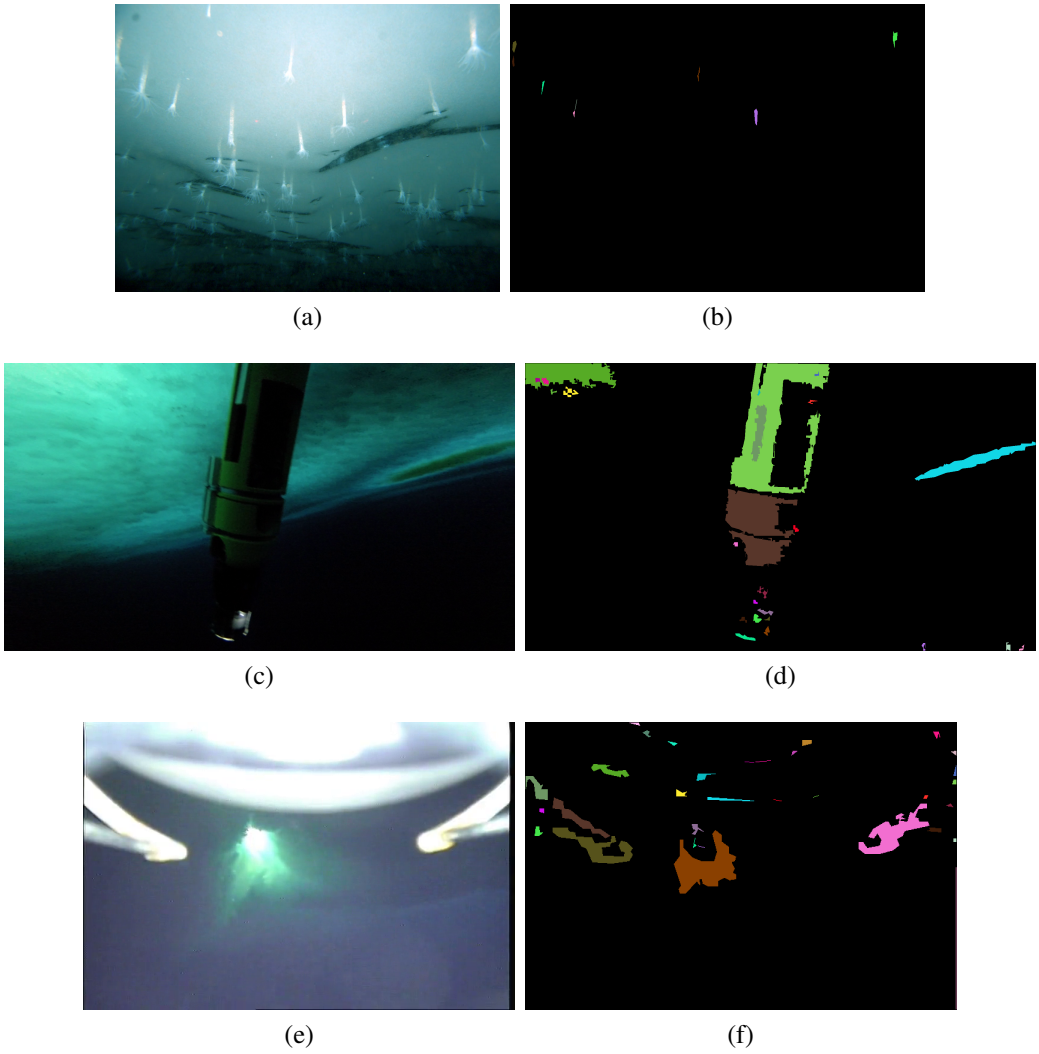


Figure 79: Results from the blue thresholding hue-based anomaly detection algorithm with real-world under-ice datasets. Anemones (a-b), the Icefin vehicle (c-d), green ice (c-d), and a hole in the ice (e-f) are all successfully detected. False detections of the Icefin vehicle body can also be seen (f), but can be filtered out in practice as the locations are known and constant.

This algorithm was evaluated using the upward all-ice simulated dataset (with ground truth), and plots of the resultant hue values are shown in Figure 80 and Figure 81. The local maximums in these plots, corresponding to estimated areas of interest, are summarized in Table 18. Most of the frames with color values around zero percent represent those with only ice in the frame across the various ice types (Figure 70). Using an estimate of vehicle location, a map of the vehicle trajectory along with estimated color can be obtained (Figure 80), where brighter values represent more non-ice color and darker values represent a smaller percentage of non-ice color present in a frame. It can be seen from this map that the algorithm outputs high color values for areas of large non-blue groupings, such as the green ice and five yellow sponges, while it has low values for ice-only areas or areas of ice with only white and black features. The use of a quantitative non-blue percentage value for each frame in a dataset yields an accurate estimate for 95% of the simulated all ice dataset, where almost all of the 30 incorrect estimates result from small false detections in the frazil ice.

In addition to evaluation with simulated data, this hue-based color estimation algorithm was tested with real-world under-ice datasets obtained in Antarctica. A summary of the results is shown in Table 19, where "Baseline" represents an image with only ice background present (no anomaly). In the simulation dataset, it can be seen that images of just platelet ice yield a value of zero (as expected), while images with green ice and other colorful anomalies yield values between seven and nine percent (corresponding to the local maximums in Figure 81). The simulated forward-looking camera datasets encounter much lower relative values for colorful detections (less than two percent), as over half of the image pixels are black, corresponding to the open water taking up most of the frame (which is ignored). In the Icefin Jetty dataset, a much smaller variation is observed in the values, with a baseline of around three percent, representing the amount of vehicle frame always present in the images. However, the platelet ice with an ice-crack anomaly yields a local maximum at 0.4% above the baseline with only water in the image. One image in this

Figure 80: Output map from the hue-based color estimation algorithm. The vehicle trajectory is plotted along with the estimated amount of non-ice color at each point. Here the color measurement is represented as a scale from white (most color) to black (least color) with a gray scale in between. It can be seen that the highest valued locations in this case are located around the large green ice patches and groups of colorful features while the lowest valued points are located in locations with no colorful features (black, white, and blue), as expected. Ground truth can be seen in Figure 69.

dataset shows a box at the seafloor, which results in a color value of 82%. Such a high value can be expected in this case, as there is no ice in the frame and the box is not of a blue hue. The anemones in the image taken from the SCINI vehicle in Antarctica [12] represent 0.4% of the image. In the GoPro Jetty dataset, no vehicle frame is present and the baseline for the values in this dataset is around 0.375%. Various amounts of green ice across the dataset increase this value to 0.4-4.1%. When the Icefin vehicle is present in the image in addition to the patch of green ice, this output value increases to 8.2%. In the frame containing the Icefin vehicle against a patch of green ice, this value increases drastically to 83%. This value, representing percent of pixels outside of the blue hue range, can be mapped to provide a relative indication within a dataset of the likelihood that a frame has a colorful anomaly present.

After evaluation of both anomaly detection methods presented here with the upward

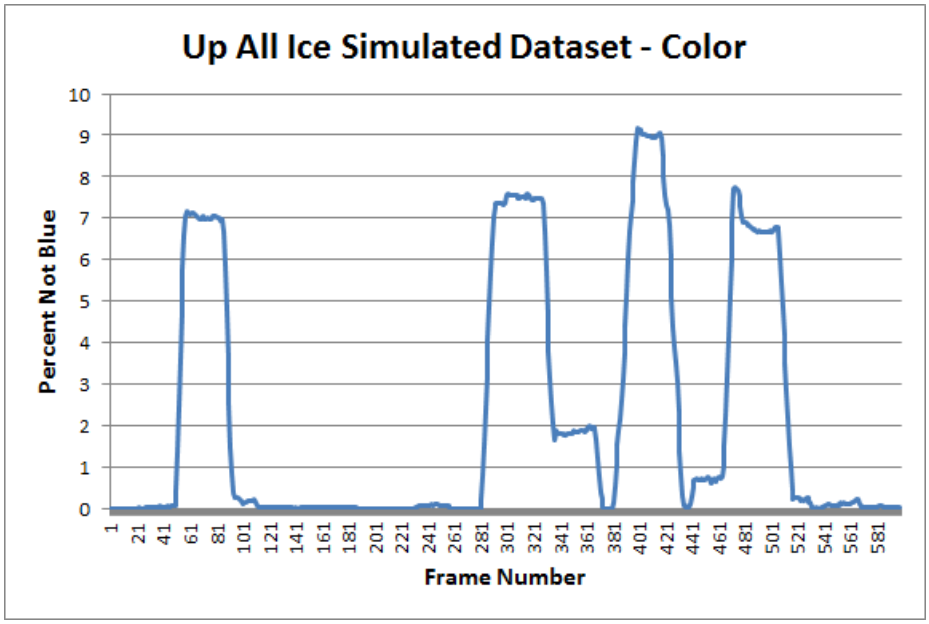


Figure 81: Output results from the local maximum hue estimation algorithm plotted versus frame number for the simulated all ice dataset. This measure represents the percentage of pixels in each frame that are outside of the blue hue range. The local maximums here represent patches of green ice and a large group of yellow sponges.

all-ice simulated dataset, results were gathered for evaluation of algorithm detection performance. The ground truth used for this evaluation was human annotation of each frame in the dataset. The results are shown Table 20 for both the hue-based and feature-based detection algorithms. As expected, the first-year and multi-year ice show the best results, as they contain a smoother ice background. There are some missed detections and false detections of anomalies in the ice, but most of these scenarios occur at the abrupt boundaries between ice types (not common in real-world datasets). However, these false detections at sharp transitions between ice types are acceptable, as they would represent areas of interest for post-analysis. The hue-based methods did not perform well with the frazil ice due to false positives stemming from small blobs of artificial colors introduced in the simulated data with such a jagged ice structure. These small false detections would be eliminated in a real-world dataset by raising the threshold for component size, due to the high texture of frazil ice. The low detection score for the feature-based method with platelet ice is caused by the high texture of the ice, which masks the texture of the anomalies (mostly misses

Table 19: Hue estimation example results (single images)

Platform	Image Contents	% Anomaly Pixels	Result
Simulated	Platelet Ice	0.000	Baseline
	MY ice with green ice and features	7.028	Detection
	Platelet ice with yellow sponges	9.000	Detection
	Platelet/green ice and colorful features	6.949	Detection
	Platelet ice and green ice	1.933	Detection
Icefin in Jetty	Platelet ice	3.562	Baseline
	Nothing just blue	3.264	Baseline
	Seafloor, Box feature	82.45	Detection
	Platelet ice and ice crack	3.682	Miss
SCINI	Multi-year ice and Anemones	0.410	-
GoPro in Jetty	Platelet ice only	0.375	Baseline
	Platelet ice and green ice	0.710	Detection
	Platelet ice, green ice and Icefin	8.192	Detection
	Platelet ice and green ice	0.844	Detection
	Platelet ice and small amount of green ice	0.411	Miss
	Platelet ice and green ice	2.367	Detection
	Platelet ice and large amount of green ice	4.103	Detection
	Platelet ice, green ice, Icefin	83.426	Detection

here). In the case of highly textured platelet and frazil ice, the hue-based algorithm should commonly perform with better results. In practice, the algorithm could automatically favor the hue-based method in a situation of high background texture detection, using the point cover texture estimation detailed above. The main problem encountered with the hue-based algorithm was missed detections with the white and black features due to the fact that these hues are filtered out during pre-processing. The main issue with the feature-based algorithm was both false detections and misses due to similarity in textures between anomalies and background ice. This leads to the desire for a fusion algorithm that uses both hue and texture to more robustly find anomalies, which is out of the scope of this work.

Table 20: Anomaly Detection Results (Simulated Dataset)

Ice Type	**Correct Detection Percentage**		
	Feature-based (SURF)	**Hue-based**	**Hue-based (blue-thresh)**
First-year	93.89	98.33	93.89
Multi-year	95.83	100	100
Platelet	57.21	80.18	81.53
Frazil	100	0	0

6.4 Summary

Post processing analysis of under-ice video datasets can be incredibly tedious. The four algorithms presented here aid in this process by mapping automatic estimates of the texture, non-ice color percentage, and anomaly candidates in each video frame over the dataset. The texture estimation method presented here provides a quantitative estimate for the number and spread of the detected point features in a frame, corresponding to the texture of the background ice. Any dense groupings of these extracted features indicate anomaly candidates for post-processing by a human analyst. Hue, or color, is also used to estimate the percentage of pixels in the image that do not correspond to ice background yielding a good relative estimate of how "interesting" the colors in a frame are. Any pixel groupings outside of the blue hues corresponding to the ice that are found from this hue analysis provide candidates for ice anomalies in post processing. Each of these algorithms show promising results across both simulated and real under-ice datasets and provide useful tools in the mapping and post-processing of these large datasets. These mapping algorithms can be run in parallel with the navigational algorithms also presented in this dissertation to best utilize the limited computational budget available on the vehicle platforms. This chapter, as well as the previous chapter, present methods to aid in navigation and mapping during under-ice UUV missions. While cameras are present on almost all UUVs, sonar sensors are just as common, and provide much greater visibility in many cases. Use of such a sonar sensor is presented in the following chapter for use in relative pose estimation as another possible

sub-ice navigational aid.

CHAPTER 7

MULTIBEAM SONAR RELATIVE POSE ESTIMATION

7.1 Introduction

Forward-looking multibeam acoustic sonar sensors are commonly found on underwater vehicles for use in object or obstacle detection. These sensors can provide range and bearing to an object in the path of the vehicle using acoustic sound waves. Forward-looking acoustic sonar sensors provide the viewer with a two-dimensional representation of the area in front of the sensor, highlighting objects with strong acoustic return. Sonar sensors do not encounter scale factor ambiguity, in contrast to the monocular vision systems used in the previous chapters. However, these sensors encounter very high noise levels, and many current sonar processing applications are limited to detecting strong-intensity blob objects in the data. The increase in quality of imaging sonars in recent years has introduced the possibility of using common image processing algorithms, such as point-feature tracking and optical flow, with these sonar images. A novel sonar-based pose estimation algorithm is presented here for use in estimating vehicle motion in relatively featureless under-ice environments. This algorithm uses an optical-flow-based approach to match points between sonar frames, along with a robust, rigid motion model to estimate motion of the vehicle. This motion model computes the optimal rotation and translation between point sets that minimizes weighted pixel reprojection error using singular value decomposition and robust estimation. The algorithm was tested with both simulated and real-world, under-ice sonar datasets with positive results. Such an algorithm provides a self-contained, frame-to-frame pose estimation method that utilizes a sensor already present on many unmanned underwater vehicles (UUVs). This method is not subject to integration-based drift rates, such as those encountered with an inertial navigation system (INS), and does not require the external infrastructure of external acoustic-beacon-based methods. Use of a sonar-based relative pose estimation method to aid in navigation under the ice complements the camera-based

algorithm presented in Chapter 5. While sonar sensors cannot provide the resolution and information-richness of camera images, the range of these sensors is much larger in low-visibility environments, such as those considered here.

7.2 Formation of Sonar Images

The most commonly used forms of perception in underwater applications rely on acoustic sensing. A form of active sonar, in which the sensor transmits acoustic pulses into the environment and then listens for the echo to return, can be used to form images of the sensor's surroundings. This is accomplished using an array of acoustic transmit and receive transducers that are able to effectively steer the sonar beam to different angles [125]. In this way, sonar sensors can be used to create a map, or image, of the surroundings by steering the beam across a number of angles and splicing together the angle slices into an image of a continuous sweep of the area in front of the vehicle. Based on the elapsed time taken for the echo to be received by the sensor, distance to an object can also be determined. Sonar data can be represented as an image of intensity return values (pixels), with range (R) from the sensor on the vertical axis, and azimuth angle (θ) from the sensor on the horizontal axis (but can be easily converted to Cartesian x- and y-coordinates). Multibeam forward-looking sonar can be used to create 1-D, 2-D, or even 3-D range maps of the surrounding environment, and is typically used for obstacle avoidance. Often the sensor is tilted at a slight angle so that the area insonified by the sensor is limited to the seafloor (or ice-sheet in this case). A good illustration of the formation of sonar images can be found in [60], and is shown here in Figure 82.

One benefit realized with sonar sensors is the lack of scale factor ambiguity, which is a problem in the case of optical monocular vision. However, while the angular bearing resolution in the horizontal azimuth (yaw in this case) direction is inversely proportional to the number of beams, there is ambiguity in the altitude direction due to a single beam resolution with a relatively large beam width (20 degrees here). This ambiguity disallows

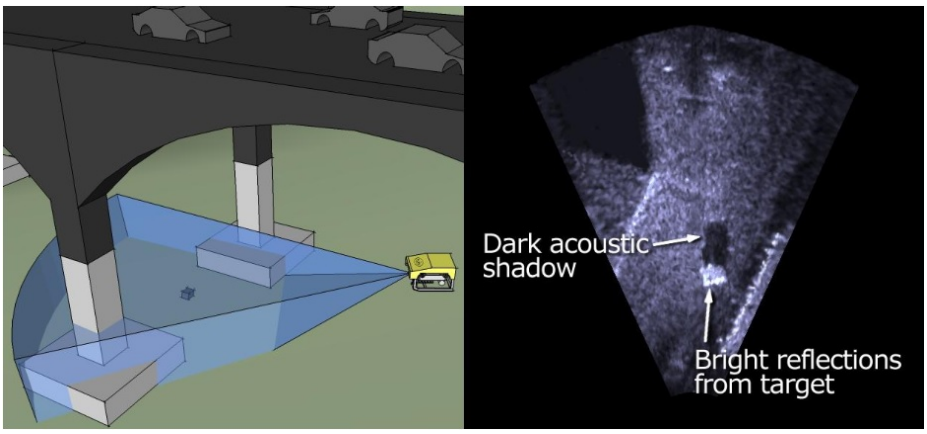


Figure 82: Illustration depicting the formation of a sonar image [60]

resolution of the z (Cartesian) or elevation angle (polar) degree-of-freedom, and reduces the three-dimensional world to a two-dimensional representation with ambiguity. In the ice-survey applications considered here, it is reasonable to assume a constant elevation angle (or z-distance) to the mostly flat ice surface of interest. It is also reasonable to assume that the ice topography will comprise almost all acoustic returns found in the sonar data. Therefore, the elevation angle ambiguity can be ignored, and a 2-D sonar image can be used to provide a full picture of the local environment. While absolute information on the unknown elevation angle to the ice sheet can be obtained using an altimeter sensor present on the vehicle, this is not needed to calculate relative range and bearing (or x and y) shifts between successive sonar images. Therefore, this z-range distance to the ice is considered to be constant here and is factored out, as altimeter and depth measurements can be used by control software to keep the vehicle at a constant distance from the ice.

7.3 Preliminary Investigation of Optical Flow with Sonar Data

This section presents a preliminary investigation into the use of optical flow methods for determining vehicle ego-motion based on sequences of acoustic sonar images. If a sonar sensor has a lock on the seafloor or ice-cover, features encountered in the resultant images can be used to extract motion between frames. While under-ice environments contain very

few and indistinguishable features, the model here makes the assumption that all environmental features are static, and all movements are correlated between sonar frames. This assumed model can be used with machine vision techniques to determine vehicle motion between frames. Statistical analysis of the estimated movement of all detected features from one frame to the next is assumed here to provide an accurate estimate of vehicle motion, as the vehicle is the only object assumed to move between frames. In this way, a shift detected in the sonar image of a few pixels can translate directly to vehicle movement of a certain distance or angle. In this initial investigation of the use of optical flow with sonar data, the motion model is assumed to have only surge translation and yaw rotation. This simple model was used for initial validation of the use of optical flow with sonar imagery. Following positive results in this preliminary investigation, a final motion model was developed, and is presented in Section 7.4, which also incorporates sway motion. Range-bearing sonar image representations were used in this preliminary investigation, but were abandoned in favor of the Cartesian representation for the final algorithm. The performance of a sparse, as well as a dense, algorithm for calculating optical flow are compared for use in this application, along with a common algorithm for the estimation of the rigid transform between images. An investigation into sonar noise reduction methods is also presented in this section. Multiple preprocessing filters were tested in an attempt to transform the sonar data into a form more attractive for robust feature detection and tracking. This preliminary investigation provided the groundwork for development of the final sonar-based relative pose estimation algorithm, presented in Section 7.4.

7.3.1 Noise in Sonar Images

While sonar provides a common method used for viewing the underwater environment, sonar data is extremely noisy, and requires interpretation by experienced sonar experts in most applications. Sonar data may be corrupted with a much higher noise level than that seen in optical images [66]. Other acoustic sources, such as surface noise (including waves), biological noise from fish and other living entities, noise from surface ships and

propellers, and noise due to interference from other man-made sources such as acoustic modems, all introduce undesirable noise into the sonar return data. The sonar sensor itself can also add noise due to multipath returns and overlapping echoes. It is very difficult to perform useful computations, such as automatic feature extraction, on this noisy sonar data [51]. It is also difficult to characterize the inherent noise of most sonar sensors due to the dynamic nature of the sensor gains. Many times the magnitude of the noise in the image coincides with the magnitude of object pixels in the image, thus noise cannot be easily eliminated without also removing useful parts of the image.

Most previous research in the area of acoustic image processing uses a smoothing filter in an attempt to eliminate some of the noise encountered in the sensor data. Here, mean smoothing, median smoothing, Gaussian smoothing, morphological opening, thresholding, adaptive thresholding, and histogram equalization were investigated as candidate sonar preprocessing approaches to reduce noise and increase the utility of image processing techniques on these images. A range-bearing sonar image, with these various preprocessing techniques applied, can be seen in Figure 83. Histogram equalization was used for preprocessing in this initial investigation, but was eliminated from the final algorithm as it changes the pixel brightness distribution between frames, and is not well suited to optical flow methods. A Gaussian smoothing filter was determined to best reduce noise in the image without altering the nature of the raw sonar intensity returns. This smoothing filter is used in the final implementation of the sonar-based relative pose estimation algorithm presented in Section 7.4.

7.3.2 Optical Flow with Sonar Images

Optical flow [70] is a pixel-based computer vision method often used to estimate motion of pixel points between images in a sequence. This differential algorithm works on the assumption of a brightness constancy constraint (Equation 26), similar motion between neighboring pixels (smoothness), and small temporal movement between the images. In Equation 26, u and v refer to the x and y motion vector magnitudes respectively, while

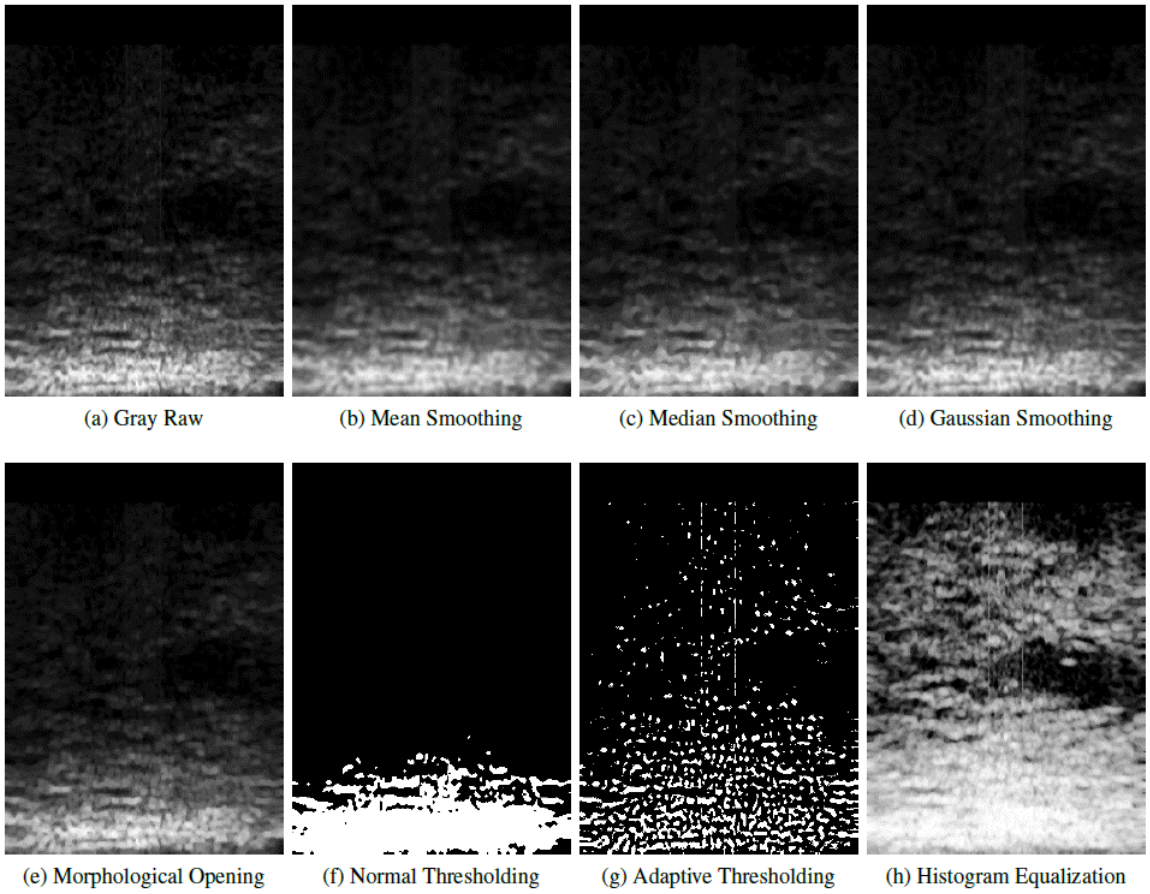


Figure 83: Outputs of Sonar Preprocessing Filters.

E_x, E_y and E_t are the partial derivatives of image brightness with respect to x, y and time respectively.

$$uE_x + vE_y + E_t = 0 \tag{26}$$

Optical flow is differential in nature and is therefore limited in noisy applications such as sonar image processing. Optical flow is not often used with sonar images due to their noisy nature and low frame rates [52]. Thus, most previous sonar image tracking algorithms favor the use of blob-based methods instead of gradient-based methods such as optical flow. Another problem encountered with sonar images that violates the optical flow assumptions occurs when the movement of the sonar sensor causes the target to appear differently in successive images. Movement of the source (acoustic in this case) can cause the image

brightness values to change at a given object point, which violates the constant brightness assumption of optical flow, and can cause errors in motion estimation. Despite the challenges encountered in using optical flow with sonar images, some preliminary research has shown the utility of such methods for object tracking using sonar data. Provided there is a sufficient number of features identified and matched between images, and the frame rate is high enough, it is possible to estimate relative motion of the sonar sensor using optical flow, despite these challenges. Because optical flow is a pixel-based, dense algorithm, it is very computationally intensive and difficult to use in real-time applications. However, sparse optical flow techniques have been developed to reduce the computational demand of this approach. The Lucas-Kanade algorithm [71] provides an approximation for optical flow of 3x3 pixel patches in the images by using least squares minimization to find an estimate for the gradient constraint equation at a sparse set of Shi-Tomasi [61] points. Specifically, a pyramidal implementation of this method [121] is evaluated here in order to increase robustness over scale-space.

7.3.3 Investigation Method

A comparison of performance between sparse and dense optical flow methods with sonar data is undertaken in this section. The dense optical flow method evaluated in this research is that of Gunnar Farneback [126]. Another commonly used optical flow method is the Lucas-Kanade method [71], which was designed as a dense algorithm, but has been adapted for use as a sparse algorithm performed on predetermined feature points. Using a pixel down-sampled pyramid of images, the method presented by Bouguet in [121] is able to relieve the breakdown encountered with large jumps between images. The Lucas-Kanade method is applied to the pyramid of images, starting with the lowest resolution first, and each optical flow estimate is used successively as an initial estimate for optical flow at the next level down in the pyramid. The features chosen from the sonar images for use in the Lucas-Kanade algorithm result from the feature detection algorithm presented by Shi and Tomasi [61]. In the initial investigation presented here, features obtained using

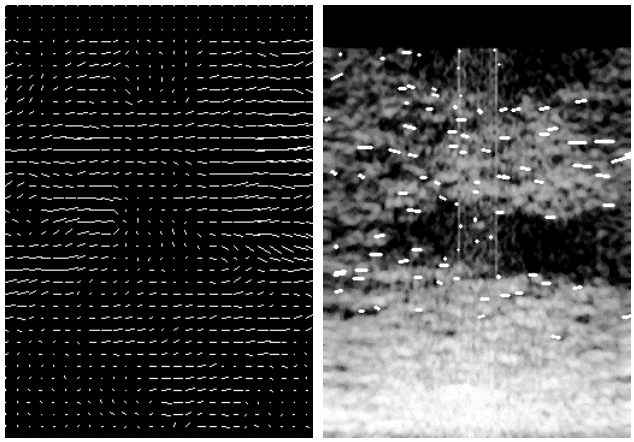


Figure 84: Visualization of sparse (right) and dense (left) optical flow results.

the Shi-Tomasi algorithm are used for sparse optical flow determination with a five level image pyramid. Sub-pixel resolution for object position estimation is then obtained. A third method for motion estimation evaluated here is an algorithm to estimate the rigid transform between two images by finding the image shift that minimizes the difference between the images using least squares minimization [73]. This least squares transformation estimate is shown in Equation 27 where A is the scale and rotation transform (not considered here) and b is the (x,y) shift. Due to the lower computational cost and better performance, sparse optical flow using the Lucas-Kanade method is chosen for the final sonar-based relative pose estimation algorithm presented in Section 7.4. A visualization of the results obtained with the two optical flow algorithms are shown in Figure 84.

$$[A^*|b^*] = \arg\min_{[A|b]} \sum_i ||dst[i] - Asrc[i]^T - b||^2 \tag{27}$$

7.3.4 Investigation Results

Sonar data collected using the BlueView sonar sensor on the VideoRay Pro IV vehicle, deployed in Lake Lanier, was used for this preliminary investigation. In this case, the surface area being insonified by the sonar sensor was a portion of the lake bottom, which is mostly planar and largely featureless. In order to determine vehicle or sensor movement,

small and sparse features such as sand ripples and rocks can be used. It is assumed that the only movement in the image should be a common feature shift in the horizontal and vertical directions of the range-bearing sonar data, corresponding to a yaw rotation and surge translation respectively. Therefore any translation of the image corresponds to movement of the vehicle in the corresponding direction, and can be directly translated as so. While optical flow methods can encounter problems tracking certain individual features in these noisy sonar images, the assumption that all interesting features should move with the same speed and direction, relaxes the computations. Under this simplifying assumption, it is only required that the majority of the interesting features are tracked correctly between frames, providing a robust method for motion estimation. The motion of the vehicle is estimated using statistical analysis of the optical flow results between frames. Each of the three methods for motion determination described previously are tested using sonar image sequences from the Lake Lanier dataset, and the performance of each method is evaluated. The computational demands of the three methods are compared using algorithm execution times, providing insight into the feasibility of algorithm implementation in a real-time embedded control system, such as that on an autonomous underwater vehicle. The ground truth image motion between frames is determined manually from human-annotated measurements of various feature locations. The results from testing both reliable and false shift data are presented as percent correct estimates in Table 21. The standard deviation values of the estimates can be used as confidence values for the shift values returned by the algorithms, where smaller standard deviation values indicate a tighter distribution, and therefore a more confident estimate.

The first data set contains a sequence of ten sonar images with the vehicle moving at a relatively constant rotation rate, two of which are shown in Figure 85. This rotation corresponds to a horizontal translation in the range-bearing sonar image space. The translation between each of the images in this sequence can be consistently measured at around 7-10 pixels per frame. The estimation results from the sparse optical flow method, the

Dataset	ERT	Sparse Mean	Sparse Mode	Dense Mean	Dense Mode
Yaw-shift	40%	90%	100%	40%	0%
Surge-shift	60%	100%	100%	80%	100%
Yaw-shift False	92%	92%	92%	100%	100%
Surge-shift False	92%	92%	100%	92%	100%

Table 21: Percent accurate movement estimates (top) and percent accurate rejection rates (bottom)

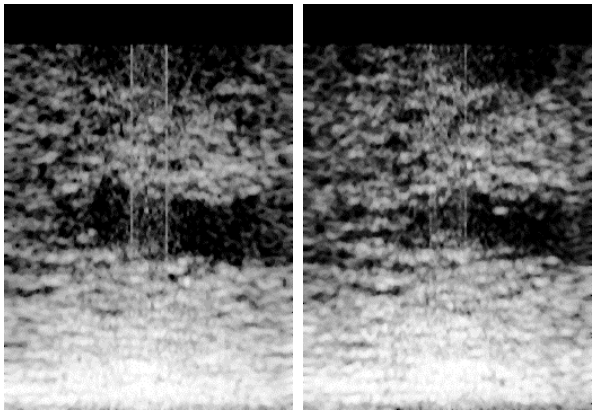


Figure 85: Example of movement between six sonar frames

dense optical flow method, and the rigid transform estimation method are plotted in Figure 86, along with the human-annotated truth measurements. The rigid transform estimation method shows a very wide distribution and drastically varying amount of error, which can be seen in all datasets. The mode value for dense optical flow calculations is always zero, and the mean dense flow estimates tend to show larger magnitude error than those produced with the sparse method. The most accurate estimates are obtained using mean or mode values from the sparse optical flow algorithm.

The second dataset used for evaluation contains a sequence of ten images with the vehicle moving in the vertical (forward) direction at a relatively constant speed of around one to two pixels per frame. The performance results from the three motion estimation methods are plotted in Figure 87. Similar to the first data set, the estimated rigid transform method produces inconsistent estimates of the shift, while the sparse optical flow method

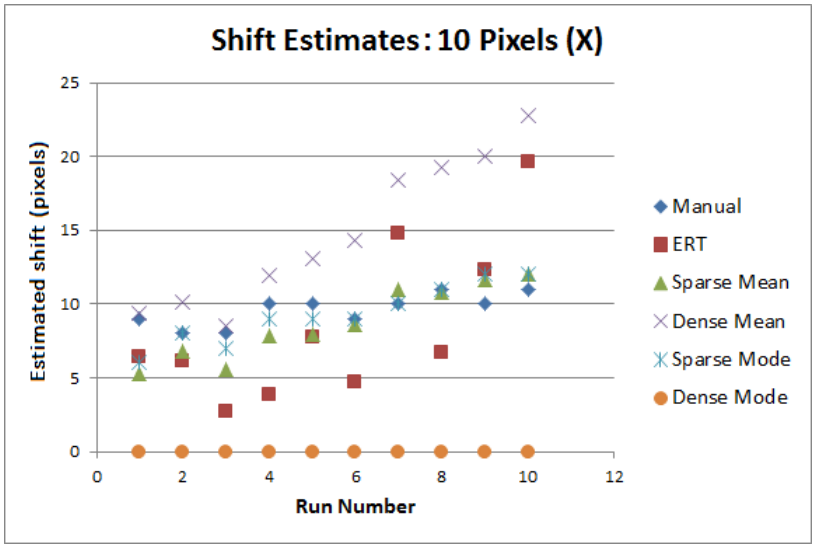


Figure 86: Shift estimates for movement in x-direction

remains the most accurate. The magnitude of movement between images is much smaller in this case (one pixel instead of ten) resulting in an increase in performance for both the sparse and dense optical flow methods.

The mode value obtained from the dense optical flow method is much more accurate in this case, and estimates two pixels each time, while the manual estimate is between one and two. The mean dense estimate is also more accurate, but still shows poor performance. The best estimates are obtained using either the mean or mode value of sparse optical flow estimates. The distributions of optical flow estimates obtained using this dataset show even higher certainty (lower standard deviation values) than the first dataset due to the smaller shift between images.

Another dataset was obtained using four sets of consecutive images with only noise in the field of view, and are used to determine the system's ability to reject noise as false targets. In the absence of targets, the best solution would be for the system to determine that no movement was made, or to reject the estimate completely and use information gleaned from other sensors or previous more reliable estimates. Therefore, the highest performing algorithm would yield estimates close to zero shift, or low confidence (high standard deviation) values, allowing the system to automatically reject them. The mode

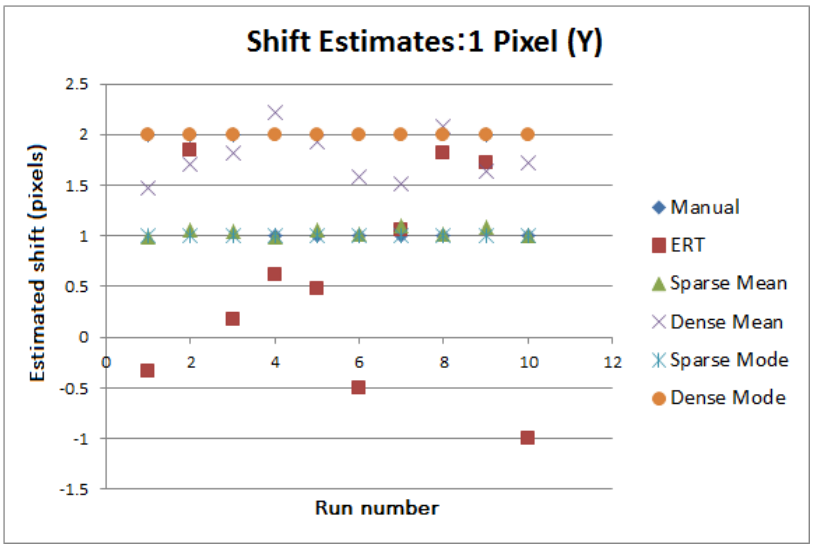
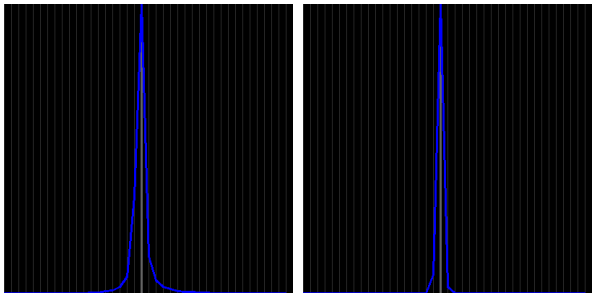


Figure 87: Shift estimates for movement in y-direction

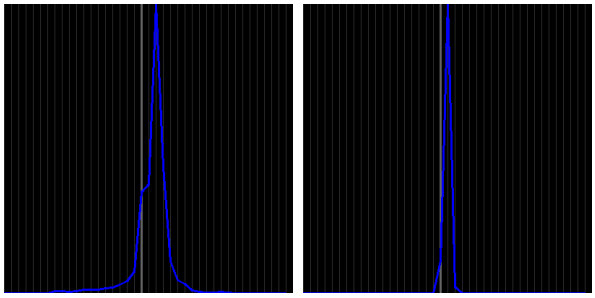
values for both sparse and dense optical flow are consistently zero, as desired. The sparse mean values show better estimate performance than the dense mean values. In almost all cases, the poorer estimates correspond to the highest standard deviation values, which should lead these estimates to be rejected automatically, as desired.

In order to ensure that optical flow algorithms are able to estimate zero movement between frames when the vehicle stays in place, a dataset was obtained using two consecutive images with no feature or target location changes between them. In this case both the sparse and dense algorithms estimated very small shifts with very high certainty, further proving that a smaller amount of movement between frames leads to more accurate estimates of shift. The best performing estimation method is again the sparse optical flow algorithm. The value obtained for a shift estimate in this case is a fraction of a pixel with incredibly high certainty (standard deviation less than 0.5), as desired. A comparison of the distributions of sparse and dense optical flow estimates for a case with no movement and a case with one pixel movement is shown in Figure 88.

The final test of robustness for the algorithms was undertaken using sonar image sequences that are not sequential, and sometimes not even from the same sonar data run. Ideally, these false image sequences should be rejected completely by the algorithms with



(a) Dense No Move (b) Sparse No Move



(c) Dense One Pix Move (d) Sparse One Pix Move

Figure 88: Histogram distributions of shift estimates where each vertical line represents one pixel

a very low number of feature matches and high standard deviation values. The results for these test sequences are shown in Figure 89. This is the only case where the estimate rigid transform method performs sufficiently well, as it rejects almost all false estimates. The best method for rejection of bad data uses mean estimates from the sparse optical flow algorithm, as almost all values of standard deviation returned from this algorithm were higher than ten, and would therefore be automatically rejected by the system. The sparse mode estimate values performed poorly as the higher mode estimate values correspond to the higher confidence values, while the sparse mean values were very low (as desired) in almost all of these cases. Dense algorithm standard deviation values were relatively high as well, and therefore all three shift estimation methods would consistently reject almost all bad image data.

The three methods for estimation of image shift show widely varying execution times. The rigid transform estimation method performed consistently the fastest (median= 7ms, mode=7ms), while the sparse optical flow algorithm ran between two and seven times

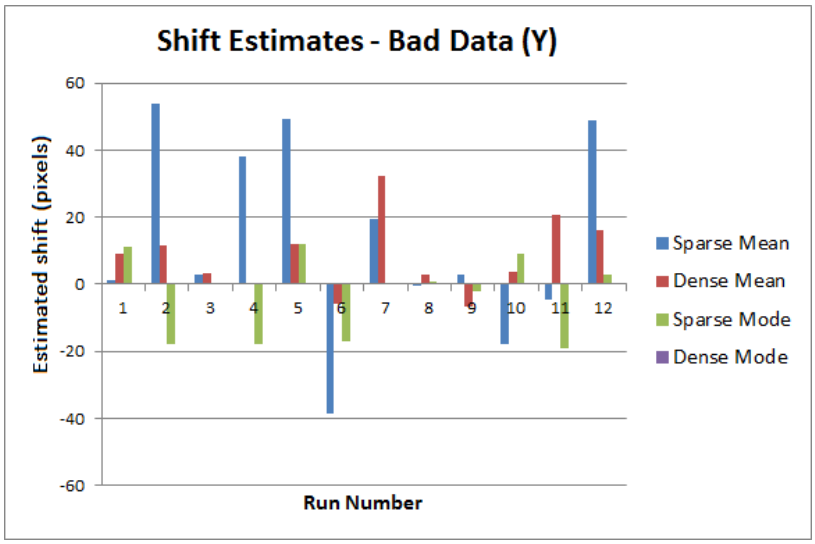


Figure 89: Shift estimates for false data

slower (median=35ms, mode=33ms), and the dense optical flow algorithm method ran between nine and fifteen times slower (median=89ms, mode=87ms). The dense algorithm is known to be very computationally intensive [73], and is typically avoided in real-world applications. The sparse method for optical flow used in this case would be able to satisfy the timing constraints of a real-time sonar processing system. While the dense optical flow algorithm is the most computationally expensive of the three considered, it obtains worse performance results than the sparse method here, due to the fact that most pixels in the sonar data represent noise or background.

7.3.5 Discussion

From the results obtained in this initial investigation, the Lucas-Kanade method was shown to be the most promising for optical flow analysis of sonar images. This sparse optical flow algorithm is able to most accurately estimate the shifts between images in a sequence, reject images with only noise or false data, and accurately determine if there has been no movement between frames. The final system has a high enough frame rate to keep the image movement between frames as low as possible, yielding the best algorithm performance.

The sparse optical flow algorithm obtains estimates of the pixel shift made by the one hundred best features in the images. This maximum feature number seems to be sufficient to predict accurate shifts without producing a lot of false tracks. The mean value of these estimates can be used to predict the vehicle movement between frames in the sonar data, as each pixel in the sonar data corresponds to a constant value of distance and angle in the real world. Therefore, an estimated value of shift in both the vertical and horizontal directions of the sonar data can be directly translated to vehicle movement, aiding in navigational accuracy. The underlying assumption is that the only movement seen in the images in these applications corresponds to a seafloor (or ice-sheet) plane shift, and all feature motion has the same magnitude and direction. In this manner, a vehicle platform can track its position change between frames of sonar data with relatively high accuracy and small drift. Other sensors used for vehicle motion determination, such as inertial navigation systems (INS), measure acceleration instead of position change directly, and must integrate to obtain the change in position. Such a system is very sensitive to drifts in accumulated position and velocity, as well as errors from the sensors.

In order to robustly reject bad data or images with only noise and no target returns, the system should reject estimate values with a standard deviation value that is higher than a predetermined threshold. This threshold can be determined using methods similar to those presented. If an estimate is returned from the sparse optical flow algorithm with a standard deviation above the threshold, it should be rejected, and previous estimate data, or data from other sensors, should be used instead. The results presented here give evidence of the robustness of this system in the rejection of bad estimates. While optical flow algorithms tend to perform poorly in noisy images, such as those produced by acoustic sonar sensors, statistical analysis and the underlying assumption of tightly coupled movement of all features in the image, provide a robust method for using optical flow for vehicle motion estimation in a UUV navigation system. This preliminary investigation of the use of optical flow with sonar data proves the utility of such a method, and provides the groundwork

for development of the final sonar-based relative pose estimation algorithm presented in Section 7.4.

7.3.6 Summary

A preliminary investigation into the use of optical flow with sonar data was presented in this section. Such gradient-based methods are not commonly used with sonar images due to the high noise levels encountered and the variation in object appearance between frames. However, this initial investigation proves the utility of using such an optical flow method with sonar data in a robust framework with a rigid motion model. Dense and sparse implementations of optical flow are compared using the Lake Lanier datasets, and the Lucas-Kanade sparse implementation yields the best performance. This method is used in the implementation of the final sonar-based relative pose estimation algorithm presented in Section 7.4. It was determined in this preliminary investigation that statistical analysis of the optical flow results can be used to estimate a model for vehicle motion. However, a more complex and robust method is required for estimation of the motion model over simple averaging. The final motion model estimation technique developed here is presented in the following section.

7.4 Sonar-Based Relative Pose Estimation Algorithm

The sonar model presented in Section 7.2 above allows for the use of a two-dimensional rigid motion model assumption for movement between frames. Motion of the vehicle in the roll, pitch, and heave (z) directions is assumed constant in the algorithm. Therefore, any valid motion (x, y, or yaw) of the vehicle relative to the ice sheet can be detected through examination of successive sonar images. A rigid motion model allows for movement in these three degrees-of-freedom only, and does not allow a change in scale. While coupling of both rotational and translational motion is difficult to resolve, singular value decomposition is used here with a least squares method for resolution of all three valid degrees-of-freedom [127]. In this method, the optimal rotation ($\mathbf{R}$) and translation ($\mathbf{t}$) are

selected to minimize Equation 28. In this case, n represents the total number of corresponding points found between images, while w_i represents the weight of each point match, $\mathbf{R}$ represents the rotation matrix between points, $\mathbf{t}$ represents the translation vector between points, and $\mathbf{p}_i$ and $\mathbf{q}_i$ represent the corresponding points in each matching set. To find the optimal least-squares solution, the weighted centroid of each set of points is first calculated (Equations 29–30) followed by the resultant centered vectors (Equations 31–32). The covariance matrix $\mathbf{S}$ is then computed as in Equation 33 where $\mathbf{X}$ and $\mathbf{Y}$ are matrices with x_i and y_i as their columns, and $\mathbf{W}$ is as in Equation 34. The singular value decomposition of this covariance matrix ($\mathbf{S}$) is then computed (Equation 35) to find the desired rotation matrix solution (Equation 36). Once the optimal translation result is obtained, the optimal corresponding translation is estimated as in Equation 37.

$$\sum_{i=1}^{n} w_i \|(\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i\|^2 \tag{28}$$

$$\bar{\mathbf{p}} = \frac{\sum_{i=1}^{n} w_i \mathbf{p}_i}{\sum_{i=1}^{n} w_i} \tag{29}$$

$$\bar{\mathbf{q}} = \frac{\sum_{i=1}^{n} w_i \mathbf{q}_i}{\sum_{i=1}^{n} w_i} \tag{30}$$

$$\mathbf{x}_i = \mathbf{p}_i - \bar{\mathbf{p}}, i = 1, 2, ..., n \tag{31}$$

$$\mathbf{y}_i = \mathbf{q}_i - \bar{\mathbf{q}}, i = 1, 2, ..., n \tag{32}$$

$$\mathbf{S} = \mathbf{X}\mathbf{W}\mathbf{Y}^T \tag{33}$$

$$\mathbf{W} = diag(w_1, w_2..., w_n) \tag{34}$$

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{35}$$

$$\mathbf{R} = \mathbf{V}\begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & det(\mathbf{V}\mathbf{U}^T) \end{pmatrix}\mathbf{U}^T \tag{36}$$

$$\mathbf{t} = \bar{\mathbf{q}} - \mathbf{R}\bar{\mathbf{p}} \tag{37}$$

In order to find the corresponding point matches between sonar images, required for computation of the rigid motion model described above, an approach is presented here using an optical-flow based method commonly used with optical camera images, as discussed in section 7.3. While this optical flow based tracking method presents a set of candidate point matches with sub-pixel accuracy, a robust sampling method is presented here to find a motion model with a maximum inlier set and minimized error. Any poor matches that do not fit this model are removed from the resultant inlier set, and are not considered in the calculation of the frame-to-frame motion model. While multiple robust sampling methods were tested, RANSAC [68] proved to be most effective in this case, and is used in the final model estimation algorithm. RANSAC iteratively finds a minimal random sample set, calculates the motion model for this set, and then computes the resultant error from this model over all match points and repeats this process until a satisfactory model (with enough inlier points) is produced. In the case of the rigid-body motion model used here, three feature points matched between sonar images provide sufficient information to estimate the transformation. All matches within a certain error threshold of the final model are considered "inliers" while any matches with model error above this threshold are considered "outliers".

The novel sonar-based motion estimation algorithm implementation begins with a preprocessing step. One of the difficulties encountered when working with sonar data is the

Pseudo code for sonar-based relative pose algorithm main loop:

Capture sonar image at time t_0 and t_1
Apply Gaussian blurring to both images
Find 1000 strongest GFTT features in t0 image
Find 1000 strongest GFTT features in t1 image
Bouguet's pyramidal Lucas-Kanade optical flow matching of features between images
Estimate rigid motion model using least-square, SVD and RANSAC
Convert motion model to vehicle-based coordinate system
Take into account altitude angle to remove foreshortening
Output estimated motion model and number of inliers

Figure 90: Pseudo code for sonar-based relative pose algorithm

low signal-to-noise ratio. Sources such as backscatter, reflections, and other acoustic transmitters inject large amounts of noise into the data, requiring preprocessing of sonar images before analysis. In the method presented here, Gaussian blurring is applied to each corresponding sonar image to reduce the effect of any pixel noise present. This is followed by extraction of Shi-Tomasi features, as described previously. Bouguet's pyramidal Lucas-Kanade optical flow based tracking algorithm is then used to match feature points between sonar images to a sub-pixel accuracy. This set of candidate feature matches is then input to the novel RANSAC implementation of the rigid motion model estimation method described previously. Finally, the altitude angle of the sonar must then be taken into account to remove foreshortening effects. The result from this algorithm is an estimated x, y, and yaw motion model of the vehicle between frames, as well as the number of inliers found for the corresponding motion model to indicate estimate strength. The pseudo-code for the sonar-based relative pose estimation algorithm presented here is shown in Figure 90.

7.5 Algorithm Results

The sonar-based relative pose estimation algorithm presented here was first evaluated using simulated sonar data, due to the controllability and inherent ground truth available. These simulated sonar datasets provided a variety of vehicle trajectories for full evaluation of the algorithm. The algorithm provides an estimate of surge (x), sway (y), and yaw (ψ) motion.

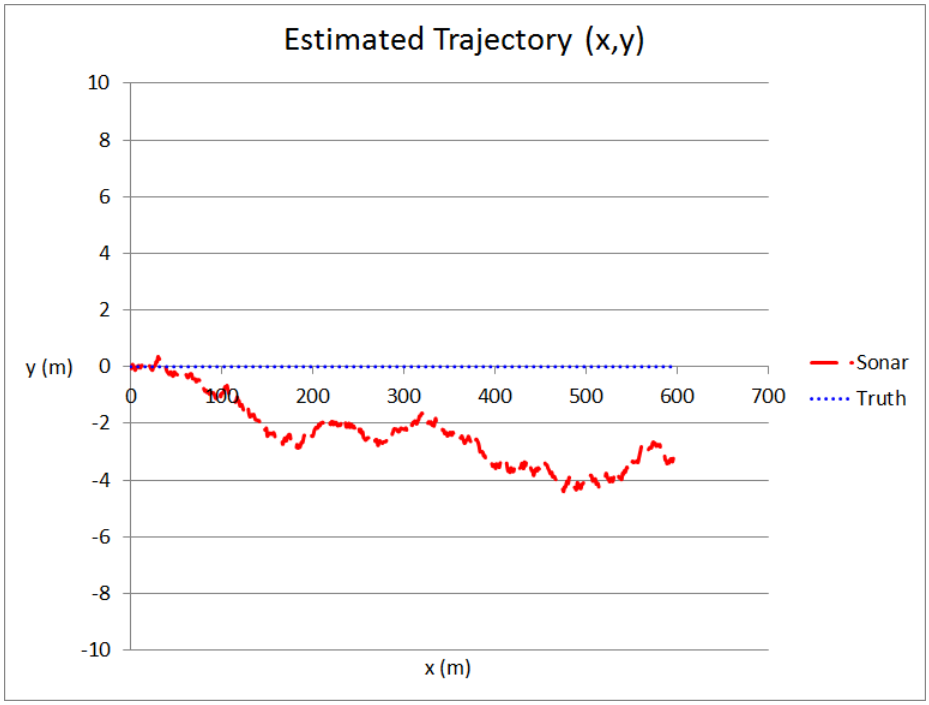


Figure 91: Forward (surge-only) estimated trajectory with ground truth

The first trajectories considered were single degree-of-freedom trajectories with decoupled rotational and translational motion. A surge-only trajectory was used for evaluation of the translational estimation accuracy of the algorithm. In this case, the simulated sonar sensor was commanded in an surge-only (directly forward) path for 600 frames. Between each frame, the simulated vehicle was moved forward the equivalent distance of one pixel in the sonar data. In the case of the simulated sonar sensor, this corresponds to one meter (resolution = one meter per pixel). The results of this evaluation can be seen in Figure 91 along with the ground truth. It can be seen that a slight drift is accumulated (clearly visible in the y-direction), but represents an error of less than one percent of the distance travelled. In this case, the maximum accumulated x-error over 600 meters is measured at 2.4m, while the maximum accumulated y-error is measured at 4.4m. This corresponds to error percentages of 0.4% and 0.7% respectively. In contrast to INS systems, this accumulated error is positional in nature and does not affect future calculations. Any error drift in INS acceleration-based methods for position estimation result in a velocity offset

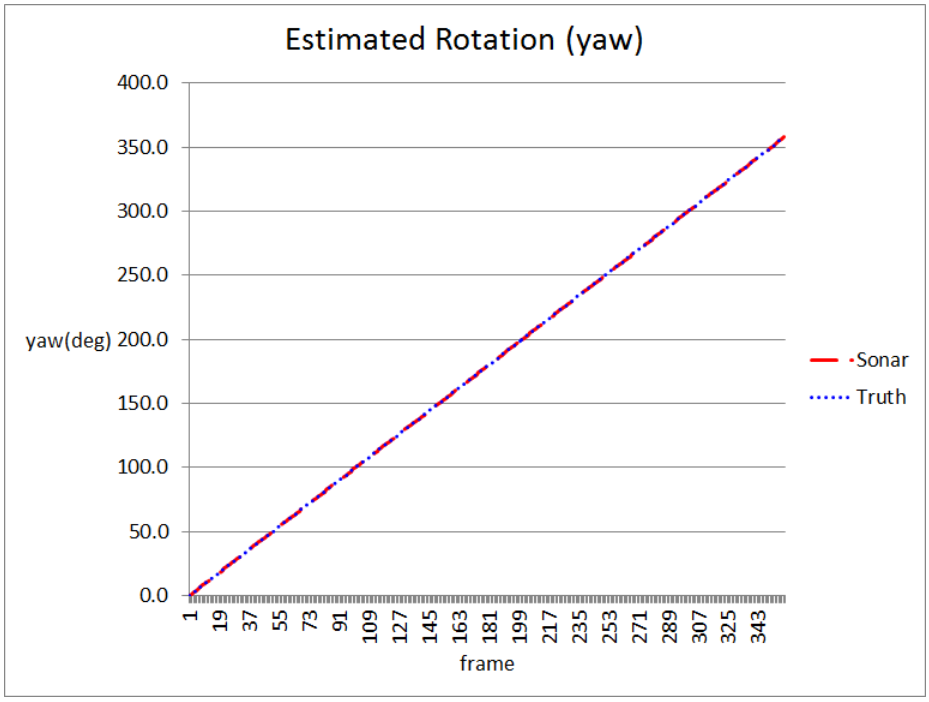


Figure 92: Rotational (yaw-only) estimated trajectory with ground truth

error, further resulting in a steady state position error, which effects all future calculations. This illustrates the benefit of a positional difference estimation method over an integrated acceleration method in tracking motion or odometry.

In addition to the surge-only simulated dataset described above, an additional decoupled, single degree-of-freedom dataset was also used to evaluate the rotational estimation performance of the algorithm. In this case, yaw motion of the vehicle was simulated at a rate of one degree per frame, over a full 360 degree circle. Surge and sway translational motion was kept at zero during this rotational motion. The results from one of the simulated yaw-only sonar datasets can be seen in Figure 92. In this case, it can be seen that accumulated error rates are very low (below one percent). The maximum accumulated yaw error is calculated at 0.22 degrees, corresponding to 0.06% of the total 360 degree rotation. Qualitatively, it can be seen from Figure 92 that the accumulated yaw-estimate result tracks very well with the ground truth. From these translational and rotational, decoupled, single degree-of-freedom evaluations, it is clear that the sonar-based relative pose algorithm

presented here performs as expected, with sufficiently low error rates.

A final decoupled simulated sonar dataset was tested with the sonar-based relative pose estimation algorithm before advancing to more complicated trajectory evaluations. In this case, the simulated vehicle was commanded in a piecewise sway- and surge-only translational trajectory to form a rectangular path. Vehicle yaw was again held constant to decouple the translational and rotational effects. The resultant estimate along with the ground truth trajectory can be seen in Figure 93. The vehicle begins at the origin (0,0) and moves in the negative sway direction, then the negative surge direction, followed by the positive sway direction and then finally the positive surge direction to return to the origin. A slight accumulated error can be seen between the sonar estimate and the ground truth in this figure. The maximum error calculated over this trajectory was 0.57 meters in the x-direction and 1.41 meters in the y-direction. This corresponds to an accumulated error percentage of 0.1% and 0.23% of the distance travelled respectively.

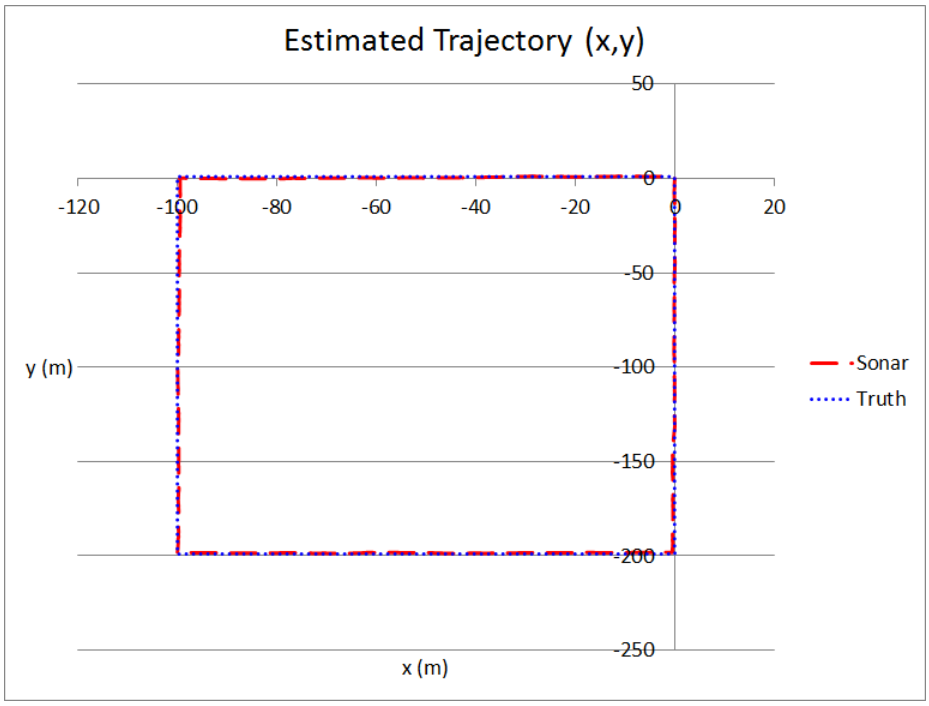


Figure 93: Rectangular (surge and sway) estimated trajectory with ground truth

The positive results from the evaluation of this algorithm over these three initial, decoupled trajectories prove the utility of the sonar-based relative pose algorithm presented here. However, further evaluation is required using a dataset which contains coupled rotational and translational motion if the algorithm is to be used in realistic applications. Estimation of coupled rotational and translational motion presents a much more difficult problem, but such motion is commonly encountered when using real vehicle systems. In order to evaluate the performance of the algorithm presented here on coupled motion, an s-curve trajectory is used to create an additional simulated sonar dataset. Over this s-curve trajectory, the vehicle follows alternating hemispherical paths, with surge and yaw motion between each frame. The ground truth s-curve path can be seen in Figure 94a along with the resultant output estimate from the sonar-based relative pose algorithm. In this case, the sonar estimate tracks well with the ground truth over the entire 377 meter trajectory, with maximum errors of only 2.06m and 1.8m in the x- and y-directions, corresponding to 0.55% and 0.48% accumulated error respectively. The accumulated yaw estimate over this s-curve trajectory can be seen in Figure 94b. This also tracks well with the ground truth yaw, and has a maximum error of 0.31°, corresponding to 0.04%, over the total 720 degrees of rotation. A summary of all maximum accumulated errors and average frame-to-frame errors over all simulated dataset trajectories (detailed in Section 4.3) can be seen in Table 22 and Table 23 respectively.

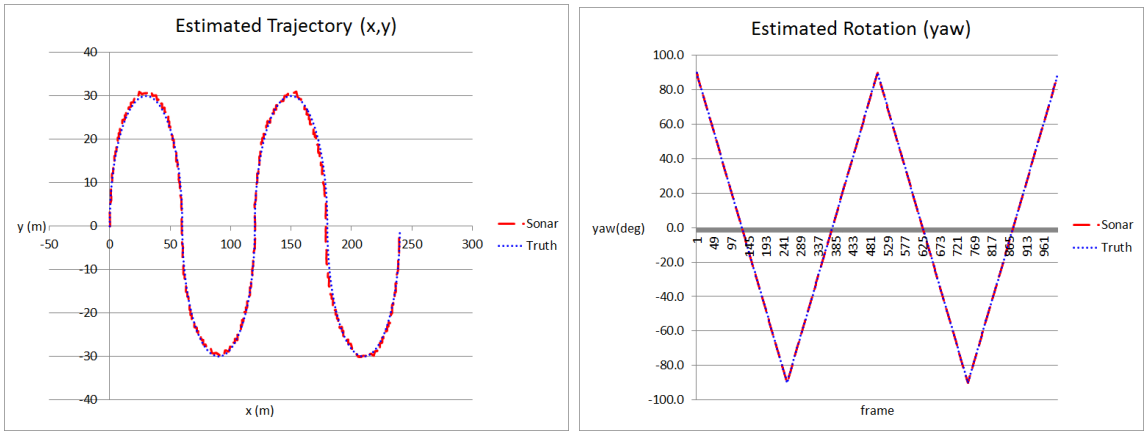
Once the algorithm performance was validated using simulated sonar datasets, further evaluation was undertaken using real under-ice datasets taken in Colorado and Antarctica. These real datasets provide difficult benchmarks due to the disturbances of human operator control, and more realistic autonomous missions would present much smoother inputs with the ice consistently fixed in the frame. In both Colorado and Antarctica datasets, a BlueView P900-45 sonar sensor was used for data collection. Compass and gyroscope data was used for ground truth in the yaw direction for the Colorado and Antarctica datasets

Table 22: Maximum accumulated error results (all simulated data)

Trajectory	**Ice type**	**Angle**	**x (%, m)**	**y (%, m)**	**ψ (%, °)**
Surge-sway translation (Fig. 93)	All	Upward	0.10%, 0.57m	0.24%, 1.41m	-
Yaw-only (Fig. 92)	Brash	Upward	-	-	0.06%, 0.22°
Yaw-only (Fig. 92)	First-year	Upward	-	-	0.06%, 0.22°
Yaw-only (Fig. 92)	Frazil	Upward	-	-	0.06%, 0.22°
Yaw-only (Fig. 92)	Multi-year	Upward	-	-	0.06%, 0.22°
Surge-only (Fig. 91)	All	Forward	0.40%, 2.40m	0.73%, 4.40m	-
Yaw-only (Fig. 92)	Brash	Forward	-	-	0.06%, 0.22°
Surge-only (Fig. 91)	Brash	Forward	0.29%, 0.57m	0.61%, 1.21m	-
Yaw-only (Fig. 92)	First-year	Forward	-	-	0.06%, 0.22°
Surge-only (Fig. 91)	First-year	Forward	0.29%, 0.57m	0.61%, 1.21m	-
Yaw-only (Fig. 92)	Frazil	Forward	-	-	0.06%, 0.22°
Surge-only (Fig. 91)	Frazil	Forward	0.29%, 0.57m	0.61%, 1.21m	-
S-curve (Fig. 94)	All	Forward	0.55%, 2.06m	0.48%, 1.80m	0.04%, 0.31°
Yaw-only (Fig. 92)	Multi-year	Forward	-	-	0.06%, 0.22°
Surge-only (Fig. 91)	Multi-year	Forward	0.29%, 0.57m	0.61%, 1.21m	-
Surge-only (Fig. 91)	All	Upward	0.40%, 2.40m	0.73%, 4.40m	-
S-curve (Fig. 94)	All	Upward	0.55%, 2.06m	0.48%, 1.80m	0.04%, 0.31°
Surge-sway translation (Fig. 93)	All	Upward	2.54%, 15.24m	1.23%, 7.39m	-

Table 23: Average frame-to-frame error results (all simulated data)

Trajectory	**Ice type**	**Angle**	**x (%, m)**	**y (%, m)**	**ψ (%, °)**
Surge-sway translation (Fig. 93)	All	Upward	2.00%, 0.02m	7.00%, 0.07m	-
Yaw-only (Fig. 92)	Brash	Upward	-	-	1.00%, 0.01°
Yaw-only (Fig. 92)	First-year	Upward	-	-	1.00%, 0.01°
Yaw-only (Fig. 92)	Frazil	Upward	-	-	1.00%, 0.01°
Yaw-only (Fig. 92)	Multi-year	Upward	-	-	1.00%, 0.01°
Surge-only (Fig. 91)	All	Forward	2.00%, 0.02m	7.00%, 0.07m	-
Yaw-only (Fig. 92)	Brash	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 91)	Brash	Forward	2.00%, 0.02m	7.00%, 0.07m	-
Yaw-only (Fig. 92)	First-year	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 91)	First-year	Forward	2.00%, 0.02m	7.00%, 0.07m	-
Yaw-only (Fig. 92)	Frazil	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 91)	Frazil	Forward	2.00%, 0.02m	7.00%, 0.07m	-
S-curve (Fig. 94)	All	Forward	-	-	1.39%, 0.01°
Yaw-only (Fig. 92)	Multi-year	Forward	-	-	1.00%, 0.01°
Surge-only (Fig. 91)	Multi-year	Forward	2.00%, 0.02m	7.00%, 0.07m	-
Surge-only (Fig. 91)	All	Upward	2.00%, 0.02m	7.00%, 0.07m	-
S-curve (Fig. 94)	All	Upward	-	-	1.39%, 0.01°
Surge-sway translation (Fig. 93)	All	Upward	4.00%, 0.04m	7.00%, 0.07m	-



(a) x,y trajectory (b) Yaw

Figure 94: S-curve (coupled rotation and translation) estimated trajectory with ground truth (left), as well as estimated yaw with ground truth (right)

respectively. Translational motion in these datasets was validated using approximated vehicle trajectories as qualitative ground truth. During the collection of some datasets, the vehicle was commanded directly forward (surge-only) while minimizing yaw motion, and then returned in the reverse manner to the origin. The resultant vehicle path should show positive surge motion, followed by an approximately equivalent negative surge motion. This expected path can be seen in the estimated surge result from a Lake John dataset in Figure 95.

Translational ground truth was not available for the Antarctic datasets. Therefore, the yaw ground truth from the gyroscope and INS was used to quantitatively validate the performance of the sonar-based pose estimation algorithm presented in this dissertation. It can be seen from the estimated output plots in Figure 96 that this algorithm tracks well with the ground truth, even in real-world applications. In the first case (left), the maximum accumulated error is 71.31° and the final error is 55.75°, corresponding to 8.49% and 6.64% respectively, over 840 total degrees of rotation. In the second case (right), the maximum accumulated error is 29.56° and the final error is 21.18°, corresponding to 10.95% and 7.84% respectively, over 270 total degrees of rotation. Most of the accumulated error in these

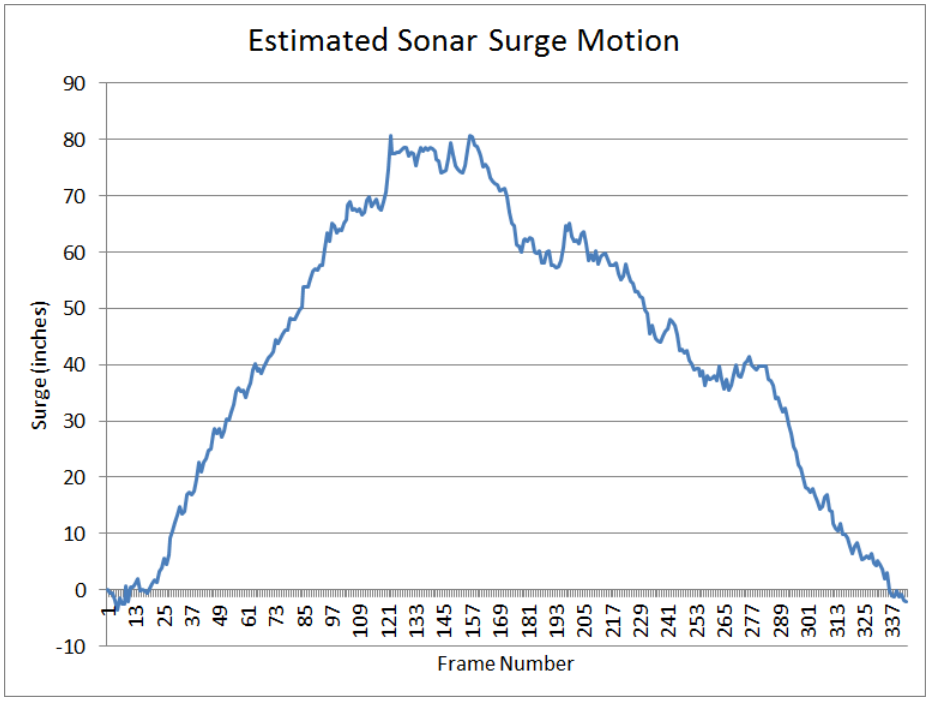


Figure 95: Estimated surge (x-motion) of the vehicle through an out-and-back dataset from the Lake John, Colorado deployment

cases results from periods of time when the ice is not visible in the frame. These promising results using real-world, under-ice datasets prove the utility of this sonar-based relative pose estimation algorithm, in addition to the underlying use of optical flow methods with sonar data.

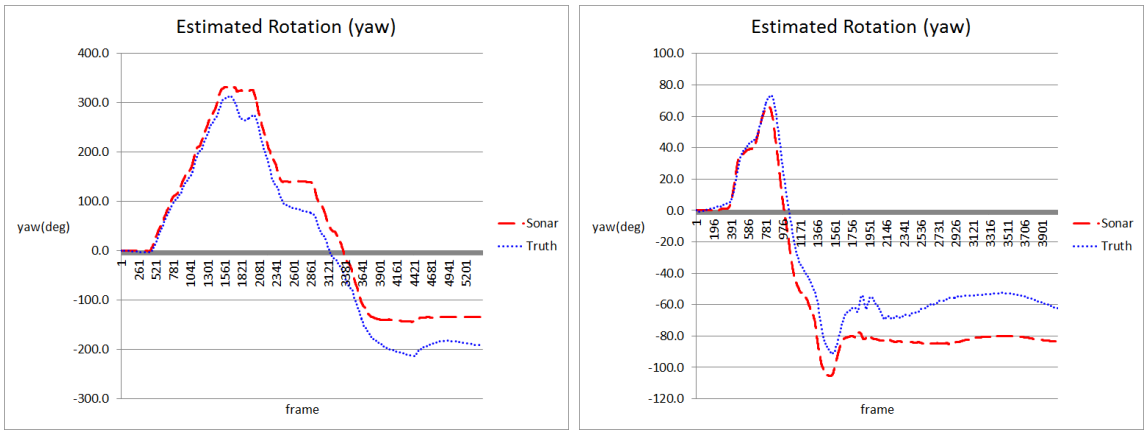


Figure 96: Estimated yaw rotation over multiple Antarctic datasets with ground truth

While translational ground truth was not available for the Antarctic datasets, qualitative

analysis of the estimated trajectories can be performed. Figure 97 presents a qualitative visualization obtained with this algorithm using one of the Antarctic datasets, in which the vehicle was commanded in a mostly forward (surge) trajectory. In this case, the magnitude of the plot is not of interest, as it is simply a summation of estimated surge motion over the trajectory. The important take-away from this plot is the positive slope of the surge accumulation value over the entire trajectory, as expected from the mostly positive surge trajectory considered here. Additional results can be found in Appendix A.

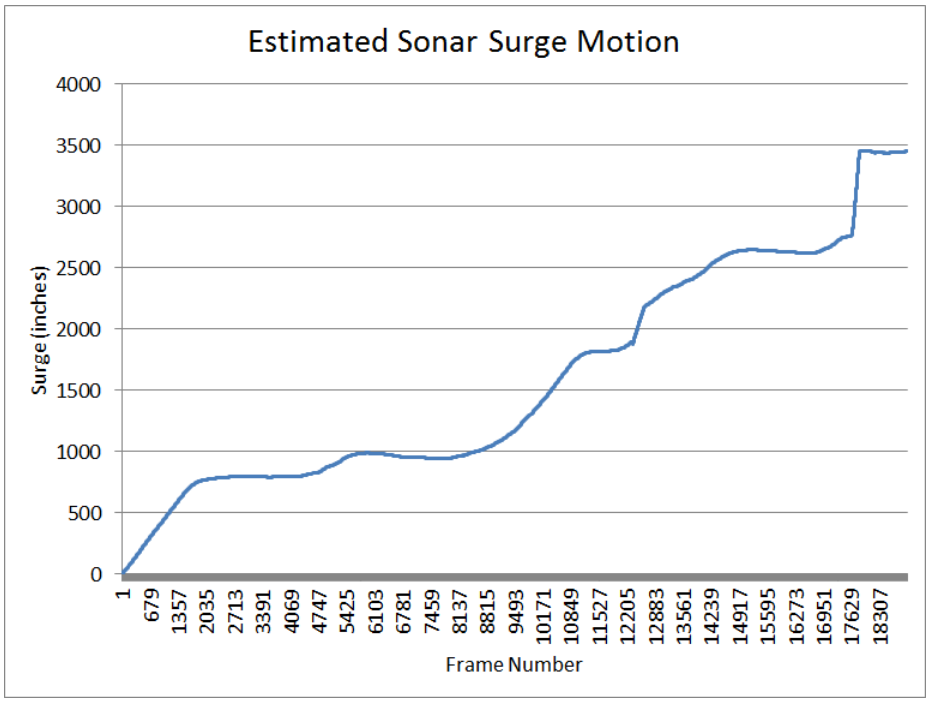


Figure 97: Surge motion estimate representation using a summation of estimated surge shifts for an Antarctic positive surge trajectory

7.6 Summary

A sonar-based relative pose estimation algorithm is presented is this chapter. The use of optical flow and point-feature computer vision methods is not common with sonar imagery. However, the application of such methodology to sonar datasets obtained in simulated and real-world under-ice datasets, presented in this chapter, proves the utility of such an approach. A preliminary investigation into the use of optical flow methods with sonar data was presented in this chapter, along with the final sonar-based relative pose estimation algorithm. Results from evaluation of these methods using simulated and real-world sonar data was presented in this chapter as well. While the sonar-based vision processing algorithms developed here, and the camera-based vision processing methods presented in previous chapters, provide valuable results, there are many drawbacks to methods which depend only on a single sensor. Fusion of information from multiple independent sensors can produce a more robust and capable system. Such sensor fusion methods, using underwater sonar and camera sensors, are investigated in the final chapter of this dissertation.

CHAPTER 8

FACTOR GRAPH SENSOR FUSION

8.1 Introduction

The baseline localization method used with almost all open-water and under-ice unmanned underwater vehicles (UUVs) is an inertial navigation system (INS), provided with information from an IMU (inertial measurement unit) and gyroscope that can be integrated to estimate position over a trajectory. While the sole reliance on an INS is prohibitive to most missions due to the unbounded drift rates encountered, this remains the most common solution for under-ice UUVs due to the unique environmental constraints. One proposed solution to the under-ice localization problem is visual SLAM (simultaneous localization and mapping). SLAM [7] refers to a localization method commonly used in terrestrial robotics where observations of landmarks in the environment are used to concurrently create a map of the environment and localize the vehicle inside that map as the vehicle moves. However, under-ice environments are largely low-contrast and featureless, and comprise mostly of repetitive ice texture; thus re-observation of unique landmarks for loop-closure constraints in SLAM is very challenging. Loop-closure is required, in the case of SLAM, in order to bound localization uncertainty drift, as re-observation of a landmark gives much insight into the relative vehicle motion encountered between viewings. Therefore, only a very limited set of current under-ice vehicles utilize a camera-based localization method. SLAM using a forward-looking sonar sensor (present on many underwater vehicles) has been used with UUVs in open ocean environments (no ice cover), but typically requires large anomaly features for blob tracking throughout the vehicle trajectory. Due to the limited anomaly features present in the sub-ice topography, the use of such sonar sensors for under-ice navigation is also very uncommon.

Here, the novel use of both video- and sonar-based relative pose estimation methods is presented to provide additional constraints on the estimated vehicle trajectory, and to help

bound the drift rates of the INS. In Chapter 7 a method was developed for use of a forward-looking sonar sensor with feature-based optical flow methods to track three degree-of-freedom ego-motion, even in relatively featureless environments such as sand or ice. Such a method extracts feature points [61] (pixels with unique and high-gradient neighborhoods), to match between successive frames, which can be used to estimate a world motion model between those frames. A method for camera-based relative pose estimation, even from the low contrast, relatively featureless environments encountered under the ice, was also presented here in Chapter 5. Sonar data can be used to obtain a three degree-of-freedom motion estimate (yaw, surge and sway), while six degrees-of-freedom (minus translational scale) can be estimated from the camera data. Each of these sensors has unique limitations when used independently. In addition, some individual frame-to-frame estimates over the trajectory can be inaccurate, due to a low number of feature points detected in a frame, a low number of point matches between frames, or inaccurate point matching.

From these insights, a method is developed in this chapter for fusion of both sonar- and camera-based estimation of vehicle motion between sensor frames. Independent INS, camera-based, and sonar-based systems each present unique weaknesses and strengths, and each sensor can encounter periods of inaccurate estimation. The use of a factor graph framework [8] [9] is presented here to fuse this information together and obtain a robust position estimate over the vehicle trajectory, which combines the strengths of each sensor to overcome the independent weaknesses. A factor graph framework is a model well suited to complex estimation problems, where nodes represent unknown random variables to be estimated and the factors between them represent probabilistic information on those variables (Figure 98).

Optimization can be performed over all available estimates in the factor graph framework to obtain the optimal vehicle state estimate over the entire trajectory. Here, the use of a GTSAM (Georgia Tech Smoothing and Mapping) [8] [10] [9] framework is presented to fuse frame-to-frame sonar and video relative pose estimates. A novel sensor fusion method

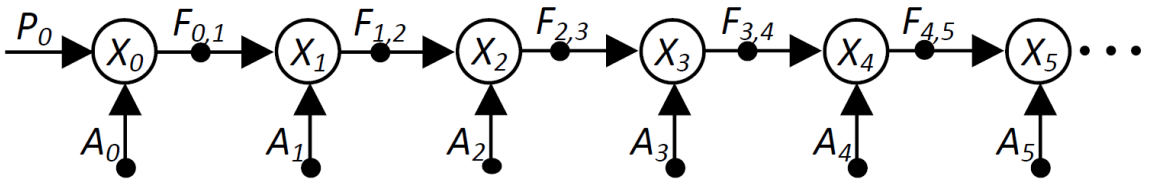


Figure 98: Visualization of a generic factor graph framework, where variable nodes are shown as large open circles (X_n), a prior node is shown as P_0, factor nodes are shown as small solid circles ($F_{n,m}$ for binary and A_n for unary, or absolute, factors) and where n,m $\in$ 0,1,2.... Movement left to right here represents passage of time.

is presented here which incorporates the number of features matched between frames into the error model of the estimated vehicle trajectory, in order to more heavily weight the impact of estimates with more matches over those with less matches. In this way, stronger estimates from one sensor can overpower the weaker, less accurate estimates from the sensor with fewer point matches, resulting in a more robust system. This sensor fusion algorithm is evaluated using simulated data with inherent ground truth, as well as real-world data taken under the ice near McMurdo station, Antarctica (with INS estimates as truth). The sensor fusion method developed here shows promising results for aiding in under-ice UUV relative pose estimation by robustly fusing estimates from both sonar and video sensors. The fusion framework, algorithm and evaluation results are discussed in this chapter.

8.2 Factor Graphs and GTSAM

Under-ice environments can be very feature-poor and low-contrast. Therefore, relative pose estimates obtained from one sensor (i.e. camera or sonar) can be prone to occasional inaccuracies. Fusion of estimates from multiple sensors can provide much more robust estimates. The type of high-level sensor fusion presented here uses a factor graph framework to optimally combine noisy but partially redundant estimates of relative pose from sonar, camera and other navigation sensors. Factor graphs are abstract representations of variables and constraints in an optimization problem [9]. The algorithm presented here utilizes a factor graph framework inside the Georgia Tech Smoothing and Mapping Library

(GTSAM) presented by Dellaert and Kaess [8] [10] [9]. A factor graph is a bipartite graphical model $G = (F, Q, E)$ containing variable nodes $q_j \in$ Q (unknown random variables in the estimation problem), factor nodes $f_i \in$ F (probabilistic information on those variables derived from measurements or prior knowledge) and edges $e_{ij} \in E$ (encode sparse structure between q_j and f_i). A factor graph structure defines the function $f(Q)$ as in Equation 38, where Q_i represents all variables connected to factor f_i, and the goal is to find the variable assignment Q^* that maximizes this equation, as shown in Equation 39. Assuming Gaussian measurement models, this yields a nonlinear least-squares problem. A visualization of a generic pose graph framework is shown in Figure 98. Here variable nodes are shown as large open circles (X_n), while factor nodes are shown as small solid circles. In this figure, the prior factor P_0 encapsulates the prior distribution on the first node X_0. Binary factors ($F_{n,m}$) represent pose constraints between two variable nodes and unary factors (A_n) represent external absolute measurement pose constraints (n,m $\in$ 0,1,2...t).

$$f(Q) = \prod_i f_i(Q_i) \tag{38}$$

$$Q^* = argmax_Q f(Q) \tag{39}$$

The GTSAM library provides efficient, real-time implementations of many factor graph tools commonly used for robot pose estimation and mapping. In applications of factor graphs for robot pose estimation (such as that presented here), variables are used to represent the vehicle pose at each point over the trajectory, while factors encode relative pose estimates between two variables in the graph. The graph itself represents an error function between the measured and predicted trajectory. Each factor represents a part of this overall error function, and contains a relative pose measurement, as well as a matrix (information matrix) to weight the contribution of the factor in the global error function relative to other factors. Measurement error (noise) can be represented as a probability density function on each factor. Note that in this work, the idea of high uncertainty (variance) in a variable's

value is referred to as "high noise" as well as "low confidence". The pose estimation problem using factor graphs is nonlinear in nature due to the rotation matrices and Euler angles used to represent angular pose.

Optimization of the factor graph to obtain the maximum a-posteriori (MAP) assignment for a vehicle trajectory can be performed by the square root information Smoothing and Mapping algorithm [8], using the iSAM incremental real-time implementation [10]. This implementation exploits the sparsity of the pose estimation problem to increase efficiency and performance. In order to find the optimal variable assignment, the overall error function (Equation 40) is minimized over all factors in the graph (where F is a factor, e_F is the error for that factor, and ΛF is the information matrix for the factor used to weight each factor's error). GTSAM uses the Levenberg-Marquardt [128] nonlinear optimization method to solve the factor graph and provide this most likely trajectory over all past and present poses. This optimization method requires initial values to seed the algorithm in order to increase the likelihood that the global minimum is found, instead of a false local minimum (or none at all). One of the advantages of such a "smoothing" method over filtering is that the factor graph embodies the joint probability function P($X|Z$) (X represents variables, Z represents measurements) over the entire trajectory, instead of only the last pose, and optimization is performed for all variables using all available measurements, resulting in a "smoothed" out trajectory [9]. A prior factor can be used to anchor the pose graph to the global or relative coordinate system, by encoding the pose of the vehicle prior to the addition of other factors to the pose graph.

$$argmin(\sum \mathbf{e}_F^T \mathbf{\Lambda}_F \mathbf{e}_F) \tag{40}$$

In a single-sensor odometry application, a dead reckoning solution can be found by adding the current odometry measurement to the previous pose at each step. In this case the pose graph will not be over-constrained and the linearized error matrix will be square and full rank. In the case of multiple sensor inputs (presented here), some variables become

over-constrained and the linearized error matrix becomes rectangular. The pose graph optimization method can then be used to smooth over all nodes in the trajectory to obtain the least square error result between contradictory inputs. The information matrix used to weight each factor in the graph is a diagonal matrix (6x6 for a six degree-of-freedom factor node) with values approximating the inverse of the variance (σ^2) based on the noise in the measurement model. While most similar applications use a constant variance model for each sensor, a novel adaptive noise model is used here, which incorporates a per-measurement estimate of accuracy into the resultant information matrix. The application presented here does not include the absolute pose information (depth, magnetic heading) or inertial navigation system (INS) estimates, available in most underwater vehicles, in order to limit the scope of the problem at hand and to use such information as ground truth. However, it is important to note that these absolute and inertial measurements can be easily incorporated into the factor graph framework as additional factors to increase system accuracy and performance in the final system.

The problem presented here can be represented in the form of a pose graph for real-time sensor fusion of camera and sonar relative pose estimates on an under-ice vehicle. A pose graph framework tailored for the sensor fusion method presented here is shown in Figure 99, where the variable assignments are the same as in Figure 98. It can be seen that, despite operating with different measurement rates, both sonar and camera sensor factors can be represented in a single factor graph, due to the sparse nature of such a framework.

8.3 Factor Graph Sensor Fusion Algorithm

The factor graph sensor fusion algorithm presented here requires processed frame-to-frame relative pose estimates from both an acoustic sonar sensor and an optical camera sensor as inputs. These two sensors provide concurrent and independent vehicle motion estimate information, which is partially redundant but also complimentary. Sonar estimates provide absolute shift estimates (measured in meters and degrees) but are only capable of resolving

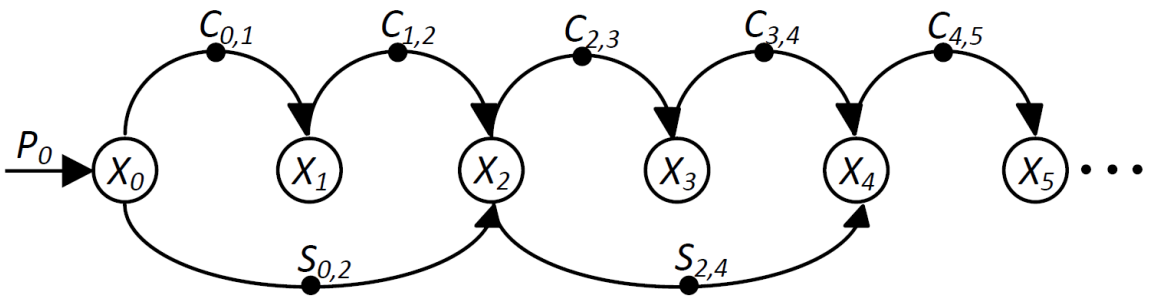


Figure 99: Pose graph framework visualization for the sensor fusion application presented here. Here, the six degree freedom vehicle pose is represented by X_n, the initial vehicle pose estimate is represented by P_0, $C_{n,m}$ represents camera pose constraints between nodes n and m, and $S_{n,m}$ represents a sonar pose constraint between nodes n and m (n,m ∈ 0,1,2...t).

motion in the vehicle's x, y, and yaw directions (Equation 41). Camera estimates provide a more complete six degree-of-freedom picture with the addition of z (depth or heave), roll, and pitch information (Equation 42). However the translational (x, y, and z) estimates are limited to a unit vector (due to a scale ambiguity), providing direction of the translational shift without magnitude values. When processed concurrently, fusion of motion estimates from these two independent sensor streams can leverage the strengths of each sensor to provide a full and robust estimate of vehicle motion.

$$\mathbf{\Delta X}_{son} = \begin{bmatrix} \Delta x & \Delta y & \Delta \psi \end{bmatrix} \tag{41}$$

$$\mathbf{\Delta X}_{cam} = \begin{bmatrix} \Delta \hat{x} & \Delta \hat{y} & \Delta \hat{z} & \Delta \phi & \Delta \theta & \Delta \psi \end{bmatrix} \tag{42}$$

Sonar frames are obtained from the sensor for processing at a rate that can vary from the advertised minimum of 67ms, to over a second in the case of the BlueView P900-45 used here (depending on the range distance). In almost all cases, this frame rate is much lower than that of the camera in the system (30 fps). The fusion algorithm presented here utilizes all frames available, even those obtained at vastly different frame rates, without the need for prior synchronization of the sensors. A pose graph is used to encompass all estimate information over the vehicle trajectory. The rate of node creation in the fusion

graph is set by the sensor with the higher frame rate (camera in this case). A variable node and two factor nodes are created in the fusion graph corresponding to each camera frame timestamp. For each set of successive camera frames, two constraints are added to the fusion graph between the nodes corresponding to the timestamps of the two camera frames. The first constraint contains the sonar motion estimate (Equation 41), which has been subdivided in time to match the camera frame timestamps (Equations 43–44). The second constraint provides the camera motion estimate (Equation 42), where an absolute translational motion estimate is obtained by multiplying the unit vector from the camera estimate by the total magnitude translation estimate obtained by the sonar sensor (Equations 45–47). In these equations, $\mathbf{\Delta X}$ represents the estimated pose vector between frames, Δt is the time difference between frames, $\mathbf{v}$ represents a velocity vector, and $\|\mathbf{v}_{son}\|$ represents the magnitude of the translational sonar velocity vector. A fusion factor graph containing such variable nodes and factor constraints can be optimized (provided an initial estimate) to provide an optimal trajectory of the vehicle motion over the entire trajectory. This optimal result utilizes all estimate information provided by the two sensors to create a robust and complete estimate of the vehicle's motion.

$$\mathbf{v}_{son} = \frac{1}{\Delta t_{son}} * \begin{bmatrix} \Delta x_{son} & \Delta y_{son} & \Delta \psi_{son} \end{bmatrix} \tag{43}$$

$$\mathbf{\Delta X}_{son} = \Delta t_{cam} * \mathbf{v}_{son} \tag{44}$$

$$\|\mathbf{v}_{son}\| = \begin{cases} \frac{v_{sonx}}{\hat{x}_{cam}}, & \hat{y} < 0.1 \\ \frac{v_{sony}}{\hat{y}_{cam}}, & \hat{x} < 0.1 \\ \frac{v_{sonx}}{2*\hat{x}_{cam}} + \frac{v_{sony}}{2*\hat{y}_{cam}}, & otherwise \end{cases} \tag{45}$$

$$\mathbf{v}_{cam} = \|\mathbf{v}_{son}\| * \begin{bmatrix} \Delta \hat{x}_{cam} & \Delta \hat{y}_{cam} & \Delta \hat{z}_{cam} \end{bmatrix} \tag{46}$$

$$\Delta \mathbf{X}_{cam} = \left[\Delta t_{cam} * \mathbf{v}_{cam} \quad | \quad \Delta\phi \quad \Delta\theta \quad \Delta\psi \right] \tag{47}$$

The sensor fusion algorithm presented here was designed for use in both real-time applications as well as during post-processing analysis. Therefore, the algorithm design takes into account the unique input data flow. The algorithm is presented in pseudo code form in Figure 100. A GTSAM nonlinear factor graph is initialized during startup to provide the framework for the fusion algorithm. An initial prior factor node is added to the factor graph to encode the prior estimate of vehicle state, along with the confidence (or noise) of the prior estimate. An initial value array is also instantiated to provide a framework for the initial guess at each graph node, which is required for factor graph optimization. An initial value is added to the beginning of the initial value array, corresponding to the prior factor node added to the factor graph.

After initialization, the main loop begins with data inflow from camera and sonar frame-to-frame shift estimates. Due to the higher frame rate, camera shift estimates are buffered until a corresponding sonar estimate arrives. Upon arrival of the next sonar shift estimate, the time difference between frames (Δt_{son}) is recorded for use in later calculations. For both sonar and video frames, $T_{inliers}$ (set at 200 here through experimentation) point-feature matches is considered sufficient to provide a robust estimate of relative pose. This robust estimate can be used in future calculations in cases where more current sonar data is unreliable or non-existent. This robust estimate is also used as input to an accumulator which keeps a running sum of frame-to-frame estimates to be used for nodes in the initial value array. In order to synchronize the sensor inputs across different frame rates and arrival times, velocity estimates (for x, y and yaw) are calculated from the sonar shift estimates and Δt_{son}, as shown in Equation 43. These velocity values normalize the shift estimates over time, and are used in later calculations.

The confidence (noise or variance) value corresponding to the frame-to-frame sonar estimate is then calculated, as it is required when the factor is added to the factor graph.

Pseudo code for factor graph sensor fusion algorithm:

```
Create factorGraph
Create initialValueArray
Add priorFactor to factorGraph
Add prior value to initialValueArray
//Main (Sonar) Loop:
While(input data available) {
    While(no sonar frame available)
        Buffer camera frame
    If(Sonar frame arrives){
        If(n_soninliers > T_inliers)
            Update robust sonar estimate
        Δt_son=frame_1.time-frame_0.time
        Calculate v_son (Eq. 43)
        Calculate sonar noise model
        //Secondary (Camera) Loop:
        For(all buffered camera frames) {
            If(n_caminliers > T_inliers)
                Update robust camera estimate
            Δt_cam=frame_1.time-frame_0.time
            Subdivide sonar estimate (Eq. 44)
            Calculate | v_son | (Eq. 45)
            Calculate v_cam (Eq. 46)
            Calculate camera estimate (Eq. 47)
            Calculate camera noise model
            factorGraph.add(sonarEst)
            factorGraph.add(cameraEst)
            initialValue = avg(sonarEst, cameraEst)
            initialValueArr.add(initialValue)
} } }
Result = Optimize(factorGraph, initialValueArr)
```

Figure 100: Pseudo code for factor graph sensor fusion algorithm

This confidence value is calculated using the number of inlier matches between the sonar frames (Equation 48 and Table 24). In these equations, K represents the noise multiplier, $n_{inliers}$ represents the number of inlier matches, $T_{inliers}$ represents the inlier threshold, and $\sigma^2_{sonTransl}$ represents the translational noise value calculated for the sonar data. If there are at least $T_{inliers}$ matching inlier points between sonar frames, the confidence is a constant for both translation ($\sigma^2_{sonTranslMin}$) and rotation ($\sigma^2_{sonRotMin}$). In cases with less than $T_{inliers}$ inlier matches, the sonar noise value model is linear with a slope equal to -1/$T_{inliers}$ and a maximum at two orders of magnitude above the high confidence values. In the case of zero inlier matches, this results in a noise value of $\sigma^2_{sonTranslMax}$ for translation and $\sigma^2_{sonRotMax}$ for rotation. All minimum and maximum variance values used here (determined through experimentation) can be seen in Table 25. The model for sonar noise versus number of inlier matches is shown in Figure 101 (translational) and Figure 102 (rotational).

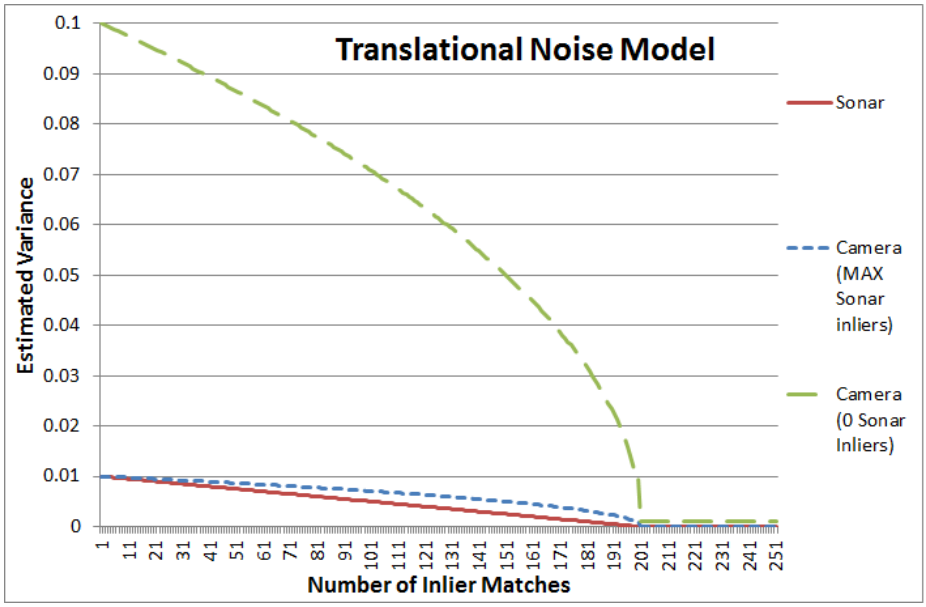


Figure 101: Translational noise model showing variance values used for the noise model on a factor versus the number of inlier point matches used to calculate the relative pose model. The maximum variance occurs with zero matches while the minimum variance occurs with over $T_{inliers}$ matches.

$$K = 1 - \frac{n_{inliers}}{T_{inliers}} \tag{48}$$

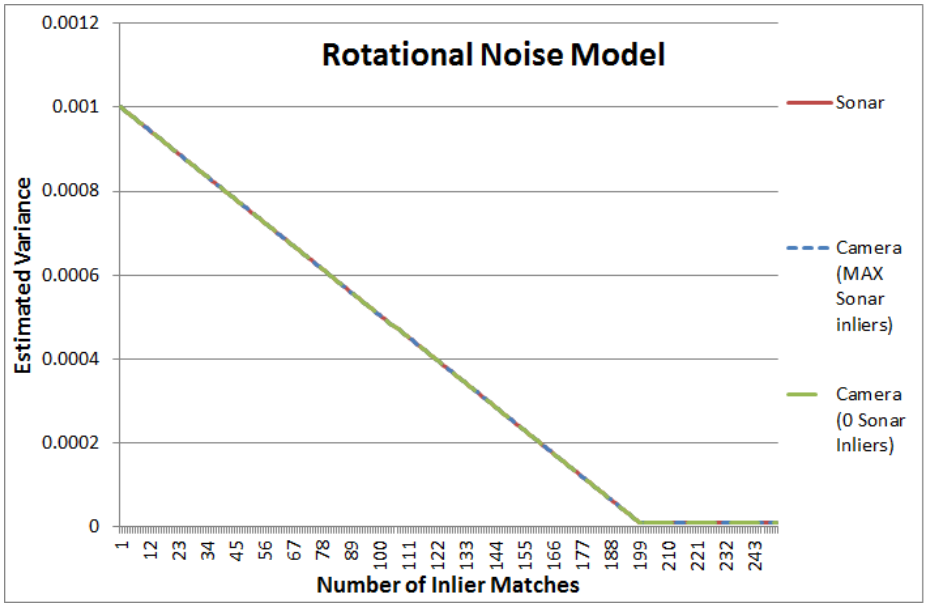


Figure 102: Rotational noise model showing variance values used for the noise model on a factor versus the number of inlier point matches used to calculate the relative pose model. The maximum variance occurs with zero matches while the minimum variance occurs with over $T_{inliers}$ matches.

Table 24: Noise Model Equations

Sensor	**Transl/Rot**	**Noise (Variance) Model Equations**		
		Inliers = 0	**Inliers <$T_{inliers}$**	**Inliers ≥ $T_{inliers}$**
Sonar	Transl	0.01	$0.01*K_{son}$	0.0001
	Rotation	0.001	$0.001*K_{son}$	0.00001
Camera	Transl	$\sqrt{\sigma^2_{sonTransl}}$	$\sqrt{K_{cam} * \sigma^2_{sonTransl}}$	$\sqrt{0.0001 * \sigma^2_{sonTransl}}$
	Rotation	0.001	$0.001*K_{cam}$	0.00001

Once all calculations are complete following the arrival of a new sonar frame-to-frame estimate, the previously buffered camera estimates obtained between the last sonar arrival and the current arrival are considered. A second nested loop is used to calculate and add each camera factor along with a corresponding sonar factor to the factor graph. For each camera estimate, the time difference between frames is calculated (Δt_{cam}). In a similar manner to the sonar processing method, the robust camera estimate is updated with the current shift estimate if the number of inlier matches exceeds the threshold of $T_{inliers}$. This previous robust estimate is used for future calculations in cases where the current shift

Table 25: Minimum/maximum noise variance variables and corresponding values used here

	Translational			**Rotational**	
Noise	**Sonar**	**Camera (Sonar=MAX)**	**Camera (Sonar=MIN)**	**Sonar**	**Camera**
Max	$\sigma^2_{sonTranslMax}$ 0.01	$\sigma^2_{camSonTranslMax1}$ 0.1	$\sigma^2_{camSonTranslMax2}$ 0.01	$\sigma^2_{sonRotMax}$ 0.001	$\sigma^2_{camRotMax}$ 0.001
Min	$\sigma^2_{sonTranslMin}$ $1 * 10^{-4}$	$\sigma^2_{camSonTranslMin1}$ 0.001	$\sigma^2_{camSonTranslMin2}$ $1 * 10^{-4}$	$\sigma^2_{sonRotMin}$ $1 * 10^{-5}$	$\sigma^2_{camRotMin}$ $1 * 10^{-5}$

estimate is unreliable or non-existent, yielding a more robust system. An estimate for total magnitude translational camera velocity is calculated from both the robust sonar (x and y) estimate magnitudes and the camera (x, y and z) estimate unit vector, due to the lack of magnitude information available from the camera estimates (Equation 45). This calculation assumes that the robust sonar translational direction and camera translational direction are in agreement in order to extrapolate magnitude information from the sonar estimate to the camera estimate. The translational velocity magnitude is first obtained by dividing the robust sonar velocities by the corresponding unit vector components (Equation 45). The total estimated camera translational shift magnitude can then be calculated from this velocity magnitude estimate (Equation 47).

Values for rotational and translational noise (variance) are also obtained for each camera factor. Because the previous robust sonar estimate is used in the calculation of the translational camera estimates, the noise of this sonar estimate must be incorporated into the overall translational noise estimate of the camera factor. A geometric mean (Table 24) is used to combine the two translational noise estimates ($\sigma^2_{sonTransl}$ and $\sigma^2_{camTransl}$) into a resultant fused translational sonar-camera noise estimate represented by $\sigma^2_{camSonTransl}$. As rotational camera noise does not depend on the sonar estimates, such an averaging technique is not required for this calculation. The strictly camera-based portion of the noise estimate ($\sigma^2_{camTransl}$ and σ^2_{camRot}) is calculated in a similar manner to that of the sonar noise estimates described previously. These variance values are calculated using the number of

inlier matches between the camera frames with the same approach described previously for sonar frames (Equation 48 and Table 24). If there are at least $T_{inliers}$ matching inlier points, the confidence is a constant for both translation ($\sigma^2_{camTranslMin2}$) and rotation ($\sigma^2_{camRotMin}$). For cases where there are less than $T_{inliers}$ inlier matches, the confidence value model is non-linear, with a maximum of two orders of magnitude more than the high confidence values. In the case of zero inlier matches, this results in a value of $\sigma^2_{camTranslMax2}$ for translation noise and $\sigma^2_{camRotMax}$ for rotation noise. The resultant camera-only noise estimate ($\sigma^2_{camTransl}$) is then combined with the most recent sonar-only robust noise estimate ($\sigma^2_{sonTransl}$) using a geometric mean (Table 24), as described above. A geometric mean is used to add robustness to the common case where the two estimates differ by multiple orders of magnitude. In these cases, a geometric mean method normalizes the weight of each estimate to have a more equivalent effect on the result, in contrast with the use of an arithmetic mean approach. In the case of a strong sonar translational noise value ($\sigma^2_{sonTranslMin}$), the range of values for the $\sigma^2_{camSonTransl}$ variable spans from $\sigma^2_{camSonTranslMin2}$ to $\sigma^2_{camSonTranslMax2}$. At the other extreme, a minimum sonar translation variance ($\sigma^2_{sonTranslMin}$) using a sonar estimate with zero inliers yields a span of resultant camera noise an order of magnitude higher ($\sigma^2_{camSonTranslMin1}$ to $\sigma^2_{camSonTranslMax1}$). An order of magnitude difference between sonar and camera translational variances is used to ensure equal noise values following the geometric sum in the case of full feature match inlier sets (high confidence) with both sensors. The model for camera noise versus number of inlier matches is shown in Figure 101 (translational) and Figure 102 (rotational) for both the case of a full sonar inlier set ($> T_{inliers}$) and zero sonar inliers.

Upon calculation of the relative pose estimate and noise model for each set of sonar and camera factors, these factors are added to the initialized nonlinear factor graph as BetweenFactor<Pose3> nodes. A single sonar constraint and a single camera constraint are added for each pass through the secondary loop (corresponding to the arrival of a camera frame), while one sonar frame is processed during each pass through the overarching

main loop. The populated factor graph representation used in the algorithm can be visualized as in 103. Here the six degree-of-freedom vehicle pose is represented by X_n, the initial vehicle pose constraint is represented by P_0, $C_{n,m}$ represents a camera pose constraint between nodes n and m, and $S_{n,m}$ represents a sonar pose constraint between nodes n and m (n,m ∈ 0,1,2...t). As in the algorithm presented here, each pose node and camera factor is added to the factor graph in Figure 103 corresponding to the arrival of a camera frame. It can be seen that a corresponding sonar node is also added for each camera frame, which is subdivided across multiple camera frames to provide a synchronous system, despite the differing frame rates of the sensors.

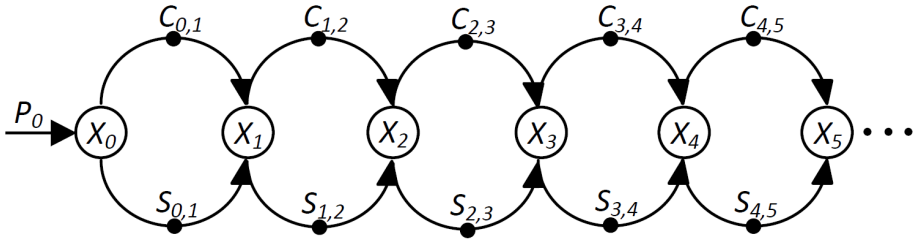


Figure 103: A visualization of the pose graph framework used in the algorithm developed here. Here the six degree-of-freedom vehicle pose is represented by X_n, the initial vehicle pose constraint is represented by P_0, $C_{n,m}$ represents a camera pose constraint between nodes n and m, and $S_{n,m}$ represents a sonar pose constraint between nodes n and m (n,m ∈ 0,1,2...t).

As the algorithm runs through the main loop, accumulator sums of both sonar and camera estimates are maintained for use in calculation of an array of initial values used to seed the optimization algorithm at each node point. In the case of a unique optimization solution, these initial values can be set arbitrarily. For more complex cases in which multiple optimization minimums exist, these values must be estimated close to the optimal solution for the Levenberg-Marquardt optimization algorithm to converge to this desired solution. In the application presented here, these initial values are calculated as an arithmetic average between sonar and camera inputs at each node. In this way, the optimization algorithm begins with a solution that lies equidistant from each independent sensor solution, regardless of the confidences on the estimates. During optimization, the confidence value on each

estimate is then utilized to pull the result toward one sensor estimate or the other over the entire trajectory.

Once a nonlinear factor graph is created and populated with a prior factor node, as well as all between-node factors (or constraints), the graph and initial value array can be presented as input to the Levenberg-Marquardt optimization algorithm. The resultant array of value node assignments represents the vehicle six degree-of-freedom, globally optimal trajectory, taking into account all frame-to-frame estimates from each sensor, and the confidence values on those estimates. This optimal result is smoothed over the entire trajectory using the iterative smoothing methodology [8] [10]. This method smooths in both the forward and backward directions (uses all nodes available) to reduce the effect of short-duration sharp inconstancies or uncertainties at the input. This output result can be used to both estimate the current vehicle state (at the present node), and to estimate the vehicle's entire past trajectory, for use in navigation or post-processing.

8.4 Algorithm Results

8.4.1 Quantitative Overview of Sensor Fusion Benefits

In order to evaluate the utility of the sensor fusion algorithm presented in this chapter, the effect of this sensor fusion on the variance (noise or confidence) of the optimized assigned variable values is first discussed. In order to obtain insightful quantitative values, the scope of this discussion is limited to the simplified cases of minimum (zero inlier matches) and maximum ($> T_{inliers}$ matches) variance noise models with constant relative pose estimates. Limiting the discussion in this way allows for the simplification of the fusion model, since no smoothing is required, and the result is a simple, linearly compounding odometry case, where only the current factor estimate and noise is used in calculations. However, this can be extrapolated over all noise models in the range between the minimum and maximum variance values with the addition of a smoothing factor, which results from the use of all factors in the graph during the optimization process to eliminate large jumps. The results from these middle cases lie on a continuous spectrum between the minimum and maximum

variance values, and share the same benefits discussed below, as well as the additional "smoothing" benefit. The fusion equations for this simplified model below (Equations 49–50) give an idea of how over-constrained inputs are weighted in the factor graph framework used here. In these cases σ^2 represents variance and $\mathbf{X}$ represents vehicle state.

$$\sigma^2_{fused} = \frac{\sigma^2_{son} * \sigma^2_{cam}}{\sigma^2_{son} + \sigma^2_{cam}} \tag{49}$$

$$\mathbf{X}_{fused} = \mathbf{X}_{son}\frac{\sigma^2_{cam}}{\sigma^2_{son} + \sigma^2_{cam}} + \mathbf{X}_{cam}\frac{\sigma^2_{son}}{\sigma^2_{son} + \sigma^2_{cam}} \tag{50}$$

Table 26 presents a summary of quantitative and qualitative sensor fusion benefits in the application considered here. In cases where both sensors provide reliable ($> T_{inliers}$ inlier matches, minimum variances) and equal relative pose estimates, the confidence of the estimates doubles (variance halves) over independent estimates. The benefit of sensor fusion in this case is to more strongly (factor of two) weight the corresponding fused output due to the additional information encoded, as well as to provide the end user with twice the assurance of the assigned value over independent sensor methods. Qualitatively, a combination of two independent but equal measurements of the same variable provides valuable information on the accuracy of the measurement (reduces variance), and allows for increased dependence on the result. In all combinations of sonar and camera variances with equal pose estimates, fusion using the factor graph framework discussed here simply results in increased levels of confidence (reduced variance). The case in which both sensors provide reliable estimates results in a decreased variance of 50% over each independent sensor estimate. The other three combinational cases considered here also see decreased variances for both sensors, the amounts of which can be seen in Table 26.

In more common cases where the two sensors provide differing pose estimates, the benefits of this sensor fusion method are much clearer. In the case where both sensors provide reliable but differing pose estimates, an average will result with a 50% decrease in

Table 26: Factor Graph Sensor Fusion Advantages

Est. Values	**Variances (Son,Cam)**	**Transl. Fusion Variance**	**Factor Weights (Son, Cam)**	**Fusion Benefit (Son, Cam)**
All	All	N/A	N/A	Added translation magnitude to camera. Addition of z, roll and pitch to sonar.
Equal	MAX, MAX	0.009091	0.9091, 0.09091	Lower variance (9.1%, 90.9%)
Equal	MIN, MIN	0.00005	0.5,0.5	Decreased variance (50%, 50%)
Equal	MAX, MIN	0.000909	0.09091, 0.9091	Decreased variance (90.9%, 9.1%)
Equal	MIN, MAX	0.000099	0.99, 0.0099	Decreased variance (1%, 99.01%)
Not equal	MAX, MAX	0.009091	0.9091, 0.09091	Weighted average. Decreased variance (9.1%, 90.91%)
Not equal	MIN, MIN	0.00005	0.5,0.5	Robust average Decreased variance (50%, 50%)
Not equal	MAX, MIN	0.000909	0.09091, 0.9091	Heavily weight low noise value. Ignore high noise val. Decreased variance (90.9%, 9.1%)
Not equal	MIN, MAX	0.000099	0.99, 0.0099	Heavily weight low noise value. Ignore high noise val. Decreased variance (1%, 99.01%)

variance over both individual sensor inputs. This results in a more robust estimate equidistant between the two equally likely pose estimates given. In the case where both sensors provide unreliable pose estimates (zero inlier matches, maximum variances), the output will be a weighted average, inherently favoring the sonar estimate as it is used for both translational pose estimate calculations. Lastly, in the most applicable cases for the sensor fusion method presented here, if one sensor presents a reliable pose estimate and the other presents an unreliable estimate, the factor graph will heavily weight the reliable estimate, while essentially ignoring the unreliable estimate. Quantitative values for these weights can be seen in Table 26. The reliable sonar data is weighted more heavily here over the reliable camera case due to the inherent use of sonar translation estimates in both sensor calculations. If the reliable estimates are assumed to be accurate and the unreliable estimates are assumed to be incorrect, these weights translate directly to decreases in error (99% over a camera-only system in the reliable sonar case, 91% over a sonar-only system in the reliable camera case).

In addition to the quantitative sensor fusion benefits described above based on reduced variance and weighted averaging, two important qualitative benefits should be noted. Sonar pose estimates contain information on x, y, and yaw motion but none on z, roll or pitch. Fusion of a complimentary camera sensor provides this z, roll, and pitch information that is not available using only a sonar sensor. Camera pose estimates are limited to a unit vector (no magnitude scale) x, y, z translational shift in addition to full roll, pitch, and yaw estimates. While magnitude of the translational shift cannot be recovered using only a camera sensor, fusion with a sonar sensor provides an estimate of this information. A summary of the quantitative and qualitative benefits of sensor fusion can be found in Table 26.

8.4.2 Synthetic Data Results

Multiple examples using realistic (but synthetic) input shift estimates are now considered to better present the utility of the sensor fusion algorithm presented here. In the first case,

consistently equal and reliable pose estimates from both sensors results in the translational output variance plot seen in Figure 104. It can be seen here that while the variance of each independent sensor, as well as the variance of the fused output, compound linearly with time, the fused output variance consistently remains at half the value of either sensor alone.

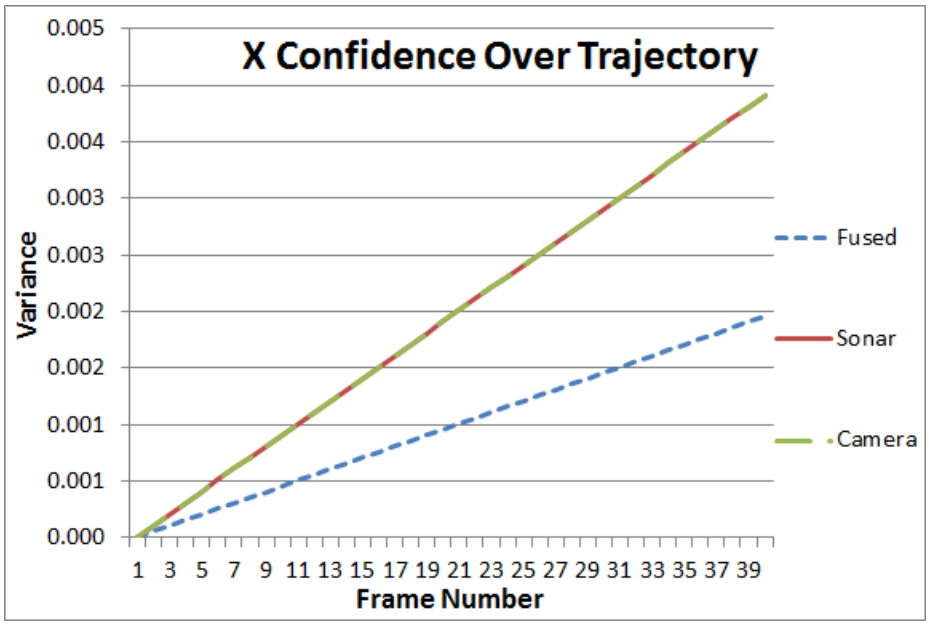


Figure 104: Variance plot of equally confident sonar and camera estimates in the fusion algorithm developed here. Independent sonar and camera results can be seen to be twice as noisy as the fused result.

In the second, more interesting case, synthetic sonar and camera pose estimates are input with alternating periods of unreliable data. Initially both sensor estimates are reliable; then the camera sensor becomes unreliable while the sonar sensor remains reliable; then the sonar sensor becomes unreliable while the camera sensor becomes reliable; etc. The variance plot shown in Figure 105 shows the utility of the factor graph sensor fusion method presented here. While the variance of individual sensors increases dramatically upon reaching a period of unreliability, the fusion variance output remains consistently low and is not subject to such drastic effects from individual sensor unreliability periods. This stems from the fact that when one sensor's pose estimate is unreliable, sensor fusion provides another independent source of information to rely on, and a more robust, low noise

result can still be obtained.

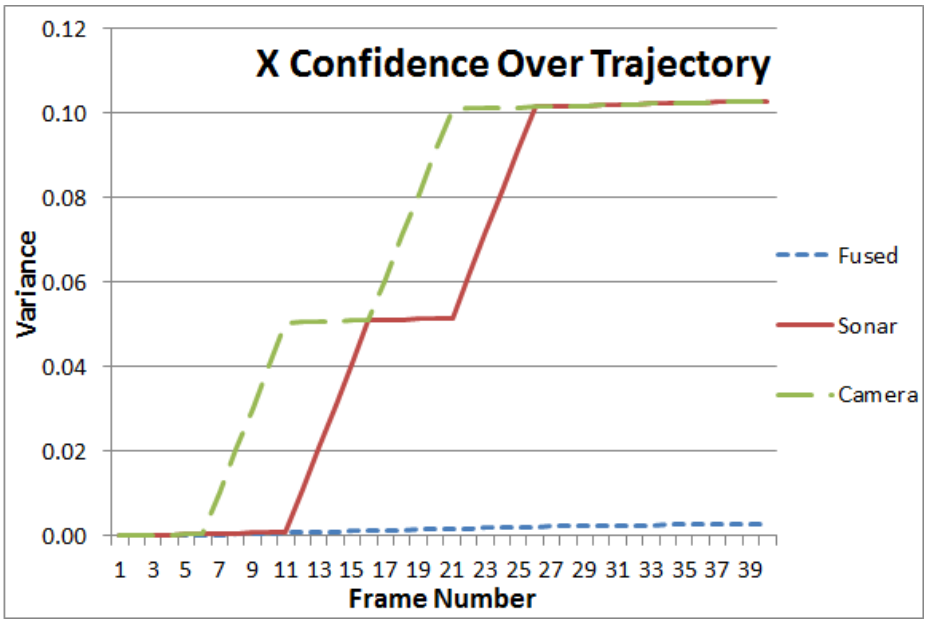


Figure 105: Variances of the estimates from camera, sonar as well as the fusion estimate, showing the decreased variance (decreased noise and increased confidence) of the sensor fused over the individual results. It can be seen that when one sensor (sonar between frames 11 and 15, 21 and 25, camera between frames 6 and 10, 16 and 20) drops out, the combined estimate does not suffer.

8.4.3 Simulated Dataset Results

Initial evaluation of the algorithm utilized synthetic input data (comma separated value files imitating the outputs of the shift estimation algorithms) in order to provide more control for unit testing, development, and algorithm evaluation. Following this stage in the development process, the algorithm was further evaluated using outputs from the actual sonar and camera shift estimation algorithms with simulated under-ice sonar and video data. This simulated data provides a good compromise between control of the inputs and realistic outputs, and has the additional benefit of containing absolute ground truth.

The first cases tested with the fusion algorithm presented here included an x-only (surge) translational case, and a yaw-only rotational case. The results from these single degree-of-freedom tests can be seen in Figure 106 and Figure 107 below. In the translational case (Figure 106), it can be clearly seen that over a translational distance of 600

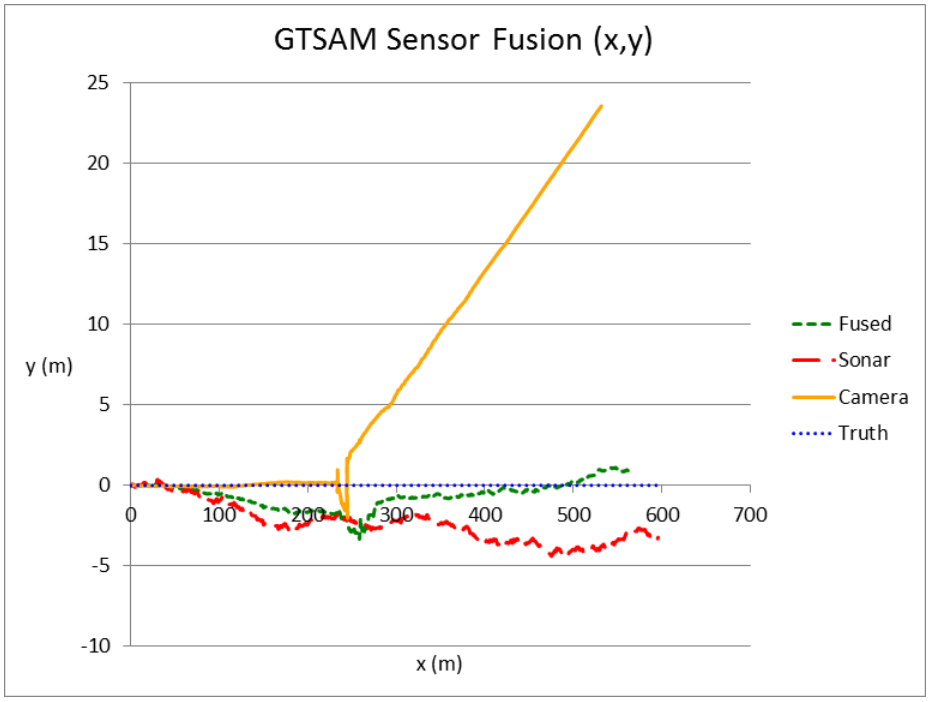


Figure 106: Translational result of fusion algorithm showing sonar, camera, fused, and truth trajectories. The truth trajectory shows decoupled, surge-only motion over 600m.

meters, the sonar y-estimate error drifts to a maximum of 4.4 meters (0.73%), while the camera y-estimate error drifts linearly to a maximum of 23.58 meters (3.93%). The larger drift in the camera data results from an unreliable estimate period (x=230m to 250m) where the estimate of yaw drifts a few degrees, causing a linear x, y drift from that point forward. While both sensors encounter drift error here, as is commonly the case, the fusion method produces a result with less error than either sensor independently. Most notably, the large drift rate in the camera data is greatly reduced while the sonar estimates are favored due to the higher confidence values, as expected. It is important to note that a camera-only translational trajectory estimate is shown despite the fact that these estimates are composed of a unit vector without scale. For all simulated data, a frame-to-frame magnitude shift of one meter can safely be assumed for plotting purposes, as this ground truth is known a priori. In the yaw-only rotational case (Figure 107), it can be seen that all results are essentially equivalent and track well with the ground truth (almost zero error).

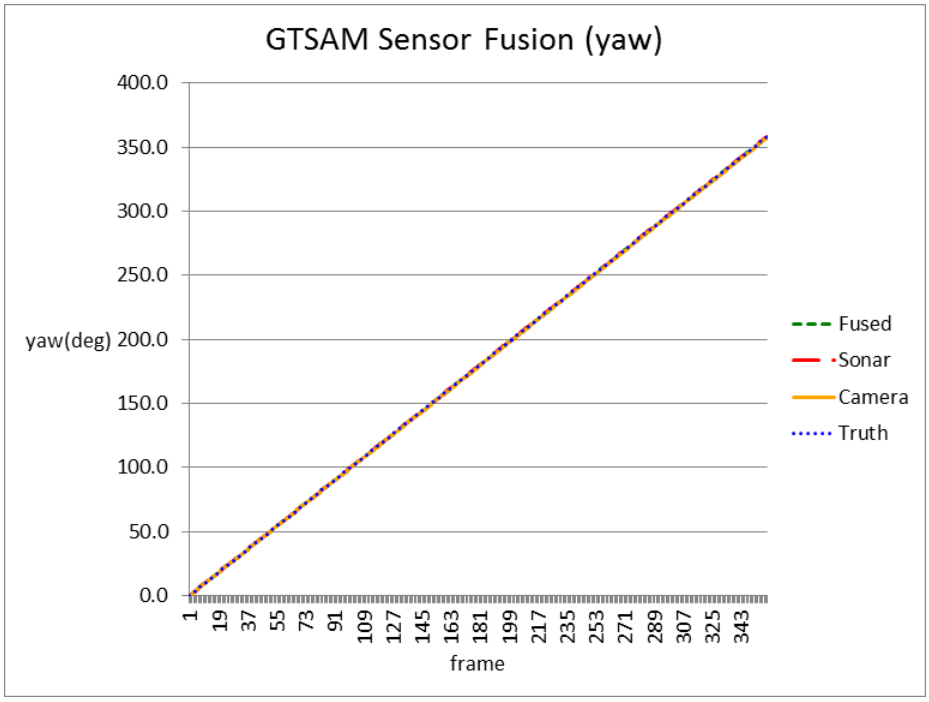


Figure 107: Rotational yaw-only result of sensor fusion algorithm showing sonar, camera, fusion, and truth trajectories. The truth trajectory shows decoupled yaw-only translation over 360.

In order to increase the complexity of the input datasets for further evaluation of the algorithm, a purely translational, but rectangular (surge and sway) trajectory is used to create simulated under-ice datasets. The output of the fusion algorithm on this rectangular input trajectory is shown in Figure 108, along with the independent sonar and camera results and ground truth. The surge (x) error plot for this trajectory can be seen in Figure 109. In this case, it can be seen that the sonar estimate drifts in the negative surge direction with a maximum error of 2.54% (15.24 meters), which results in an under-shoot of this amount at the end of the trajectory. It can be seen that the camera estimates encounter a drift in both the x- and y-directions, with a maximum x-error of 1.3% (7.8 meters), which results in an overshoot of that much at the end of the trajectory. The fused result produces a final x-error of only 0.72% (4.34 meters) over the 600 meter trajectory. This is a representative example showing the power of the fusion algorithm presented here, as the resultant error is much smaller in magnitude than either of the independent sensor errors.

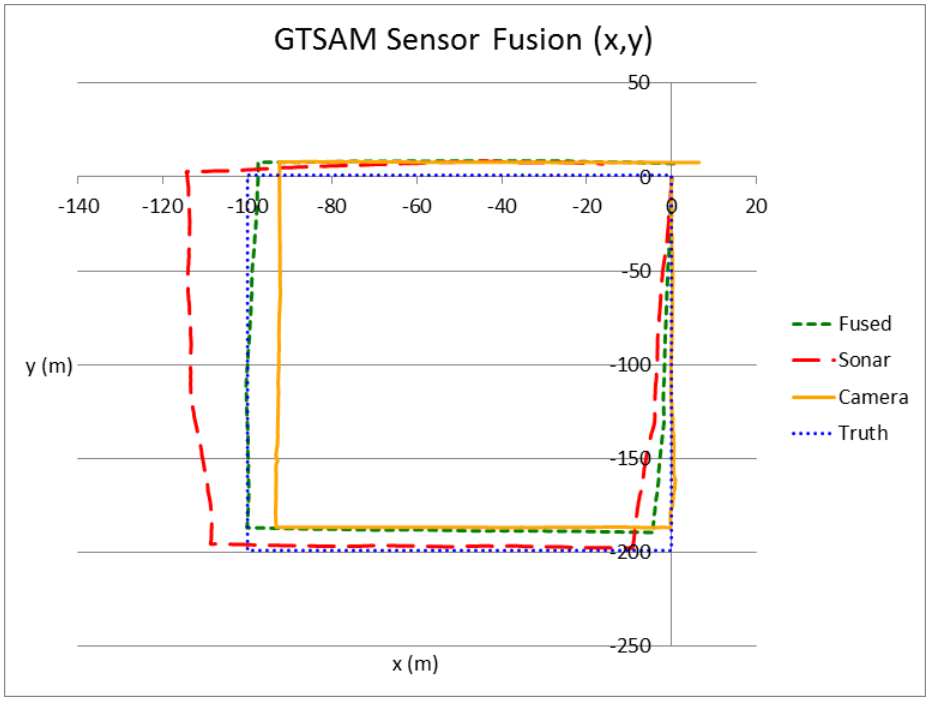


Figure 108: Translational results from the sensor fusion method on a vehicle trajectory with surge- and sway-only motion. The truth trajectory shows 200 meters of positive sway, then 100 meters of negative surge, then 200 meters of negative sway followed by 100 meters of positive surge.

A final, more complex simulated dataset was used to fully validate the algorithm prior to testing with real-world data. This dataset contains simulated sonar and video data of an s-curve trajectory, in which rotational and translational movement is coupled. The fusion result in this case is shown in Figure 110 (translational) and Figure 111 (rotational) along with the sonar-only, camera-only and truth trajectory plots. It is important to note that a camera-only translational trajectory estimate is shown despite the fact that these estimates are composed of a unit vector without scale. For this more complex case of simulated data, a frame-to-frame magnitude shift of one meter is assumed for plotting purposes. However, this is not the case here, and care should be taken during analysis of this trajectory to only consider the frame-to-frame vector shape and not the accumulated magnitude or offset. This is clear from the larger magnitude of the initial curve in the camera data in Figure 110. It can be seen from the translational case in Figure 110 that a fusion of the two sensor estimates results in a combinational result which favors the more confident sonar data, but

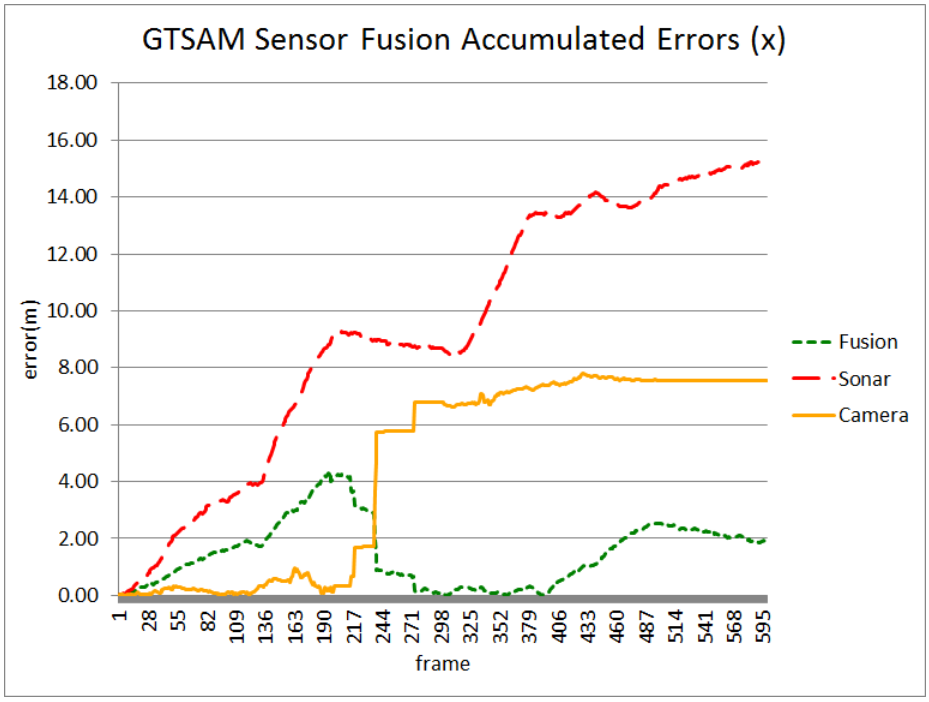


Figure 109: Accumulated error plot for the rectangular (surge and sway) vehicle trajectory in Figure 108. Plots of camera, sonar, and fused error over the inherent ground truth are shown.

also incorporates the camera directional estimates. Analysis of the yaw-only rotational result (Figure 111) is more straight-forward, as absolute angular information is known for both sensors. It can be seen from this yaw plot that sonar-only, camera-only and fusion estimates track well with ground truth. The camera maximum accumulated yaw error is measured at 0.29% and the sonar maximum error is 0.04%, while the fusion error is in between the two at 0.14% over the 377 meter trajectory, as expected. Additional results can be found in Appendix A.

A summary of the maximum accumulated translational and rotational trajectory errors from the inherent ground truth is shown in Table 27 for all simulated datasets (detailed in Chapter 4). The average frame-to-frame error results are also presented for these simulated datasets in Table 28. It can be seen in these tables that the error is reduced using fusion in many cases (e.g. the surge-only, all-ice forward dataset), and the utility of a fusion combination can be seen to provide a more robust estimate over independent sensor systems.

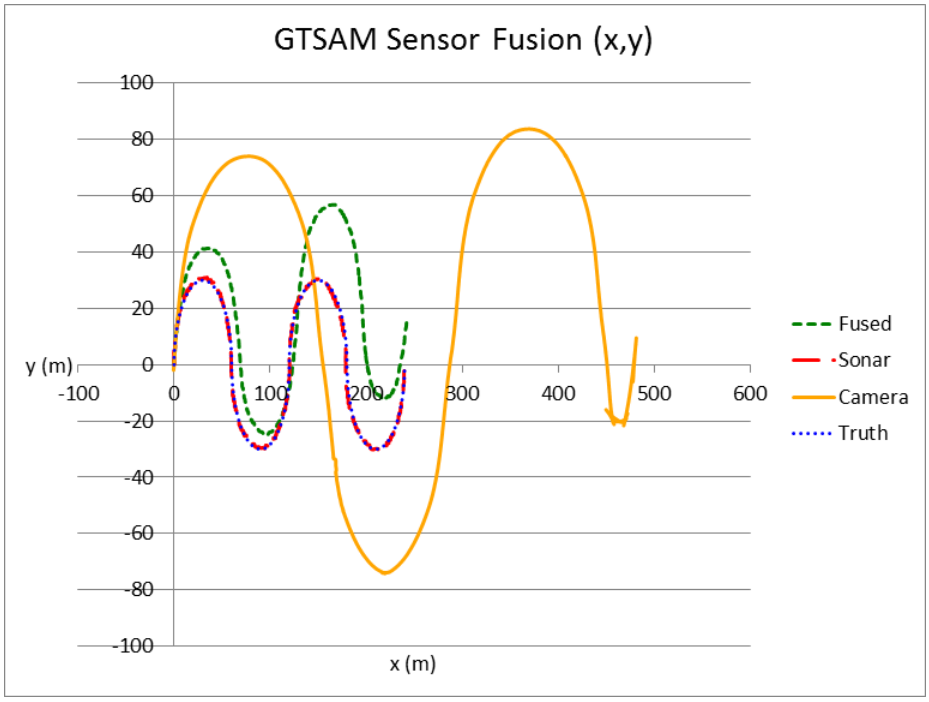


Figure 110: Translational result of the sensor fusion algorithm showing sonar, camera, fusion, and truth over an s-curve trajectory. The truth trajectory shows coupled rotation and translation over an alternating circular pattern with a radius of 30 meters.

In the case of the s-curve forward dataset results detailed previously, the maximum yaw-error is reduced in the fusion case over the camera independent sensor estimates, and lies between the sonar and camera errors. The average frame-to-frame error results from this same s-curve dataset also show that the fusion error rate is equivalent to the more accurate sonar error rate (1.4%), robust to the the camera error values that are twice as large. These examples prove the utility and robustness of the sensor fusion method presented here.

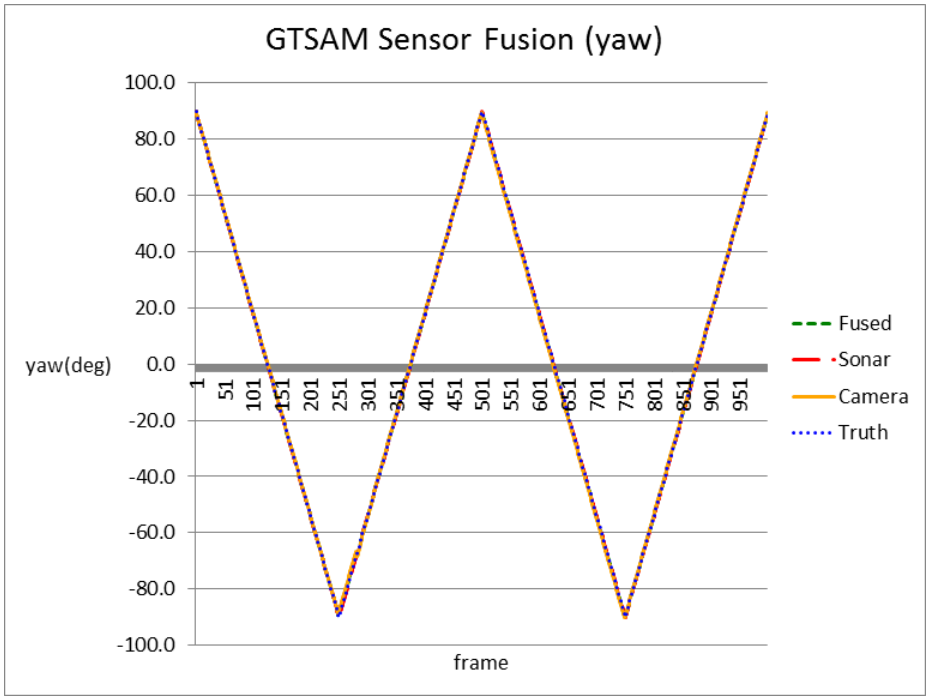


Figure 111: Rotational result of the sensor fusion algorithm showing sonar, camera, fusion, and truth over an s-curve trajectory. The truth plot shows a pattern of alternating yaw rotation between -90 and +90.

Table 27: Maximum accumulated fusion error results (all simulated data)

Trajectory	Ice type	Angle	Camera (% Error)			Sonar (% Error)			Fusion (% Error)		
			x	y	ψ	x	y	ψ	x	y	ψ
Surge-sway translation (Fig. 108)	All	Upward	1.30	2.08	-	0.10	0.24	-	0.80	1.60	-
Yaw-only (Fig. 107)	Brash	Upward	-	-	0.22	-	-	0.06	-	-	0.11
Yaw-only (Fig. 107)	First-year	Upward	-	-	0.39	-	-	0.06	-	-	0.19
Yaw-only (Fig. 107)	Frazil	Upward	-	-	0.10	-	-	0.06	-	-	0.06
Yaw-only (Fig. 107)	Multi-year	Upward	-	-	0.06	-	-	0.06	-	-	0.05
Surge-only (Fig. 106)	All	Forward	1.21	0.96	-	0.40	0.73	-	0.98	0.42	-
Yaw-only (Fig. 107)	Brash	Forward	-	-	0.24	-	-	0.06	-	-	0.13
Surge-only (Fig. 106)	Brash	Forward	0.04	0.11	-	0.29	0.61	-	0.30	0.31	-
Yaw-only (Fig. 107)	First-year	Forward	-	-	0.26	-	-	0.06	-	-	0.13
Surge-only (Fig. 106)	First-year	Forward	0.07	0.19	-	0.29	0.61	-	0.31	0.28	-
Yaw-only (Fig. 107)	Frazil	Forward	-	-	0.30	-	-	0.06	-	-	0.15
Surge-only (Fig. 106)	Frazil	Forward	0.01	0.08	-	0.29	0.61	-	0.29	0.29	-
S-curve (Fig. 110)	All	Forward	-	-	0.29	0.55	0.48	0.04	5.12	7.24	0.14
Yaw-only (Fig. 107)	Multi-year	Forward	-	-	0.25	-	-	0.06	-	-	0.13
Surge-only (Fig. 106)	Multi-year	Forward	16.22	0.16	-	0.29	0.61	-	8.41	0.31	-
Surge-only (Fig. 106)	All	Upward	11.08	3.93	-	0.40	0.73	-	5.90	0.56	-
S-curve (Fig. 110)	All	Upward	-	-	0.33	0.55	0.48	0.04	2.61	3.49	0.18
Surge-sway translation (Fig. 108)	All	Upward	1.30	2.08	-	2.54	1.23	-	0.72	2.02	-

Table 28: Average frame-to-frame fusion error results (all simulated data)

Trajectory	**Ice type**	**Angle**	**Camera (% Error)**			**Sonar (% Error)**			**Fusion (% Error)**		
			x	**y**	ψ	**x**	**y**	ψ	**x**	**y**	ψ
Surge-sway translation (Fig. 108)	All	Upward	3.0	4.0	-	2.0	7.0	-	2.0	7.0	-
Yaw-only (Fig. 107)	Brash	Upward	-	-	1.0	-	-	1.0	-	-	1.0
Yaw-only (Fig. 107)	First-year	Upward	-	-	1.0	-	-	1.0	-	-	1.0
Yaw-only (Fig. 107)	Frazil	Upward	-	-	0.0	-	-	1.0	-	-	1.0
Yaw-only (Fig. 107)	Multi-year	Upward	-	-	0.0	-	-	1.0	-	-	0.0
Surge-only (Fig. 106)	All	Forward	1.0	3.0	-	2.0	7.0	-	3.0	4.0	-
Yaw-only (Fig. 107)	Brash	Forward	-	-	1.0	-	-	1.0	-	-	1.0
Surge-only (Fig. 106)	Brash	Forward	0.0	1.0	-	2.0	7.0	-	2.0	4.0	-
Yaw-only (Fig. 107)	First-year	Forward	-	-	1.0	-	-	1.0	-	-	1.0
Surge-only (Fig. 106)	First-year	Forward	0.0	2.0	-	2.0	7.0	-	2.0	4.0	-
Yaw-only (Fig. 107)	Frazil	Forward	-	-	1.0	-	-	1.0	-	-	1.0
Surge-only (Fig. 106)	Frazil	Forward	0.0	1.0	-	2.0	7.0	-	2.0	4.0	-
S-curve (Fig. 110)	All	Forward	-	-	2.8	-	-	1.4	-	-	1.4
Yaw-only (Fig. 107)	Multi-year	Forward	-	-	1.0	-	-	1.0	-	-	1.0
Surge-only (Fig. 106)	Multi-year	Forward	16.0	1.0	-	2.0	7.0	-	10.0	4.0	-
Surge-only (Fig. 106)	All	Upward	11.0	8.0	-	2.0	7.0	-	7.0	5.0	-
S-curve (Fig. 110)	All	Upward	-	-	4.2	-	-	1.4	-	-	1.4
Surge-sway translation (Fig. 108)	All	Upward	3.0	4.0	-	4.0	7.0	-	3.0	8.0	-

8.4.4 Real-World Dataset Results

The fusion algorithm presented here was evaluated on real-world, under-ice data using datasets from the Colorado and Antarctic vehicle deployments. Ground truth for the Colorado datasets is provided by compass data for rotational yaw. Quantitative translational ground truth is not available, but qualitative analysis of the translational results can be performed based on the expected trajectory type over the dataset. For validation of the fusion algorithm on the Colorado dataset, results from yaw estimation is the primary form of analysis. One example from these Colorado datasets can be seen in Figure 112. Although both sonar and camera estimates encountered large amounts of error in this case, qualitative analysis of the plots in Figure 112 clearly shows the desired effect of the sensor fusion approach, as the fusion plot shows influence from both independent sensor estimates.

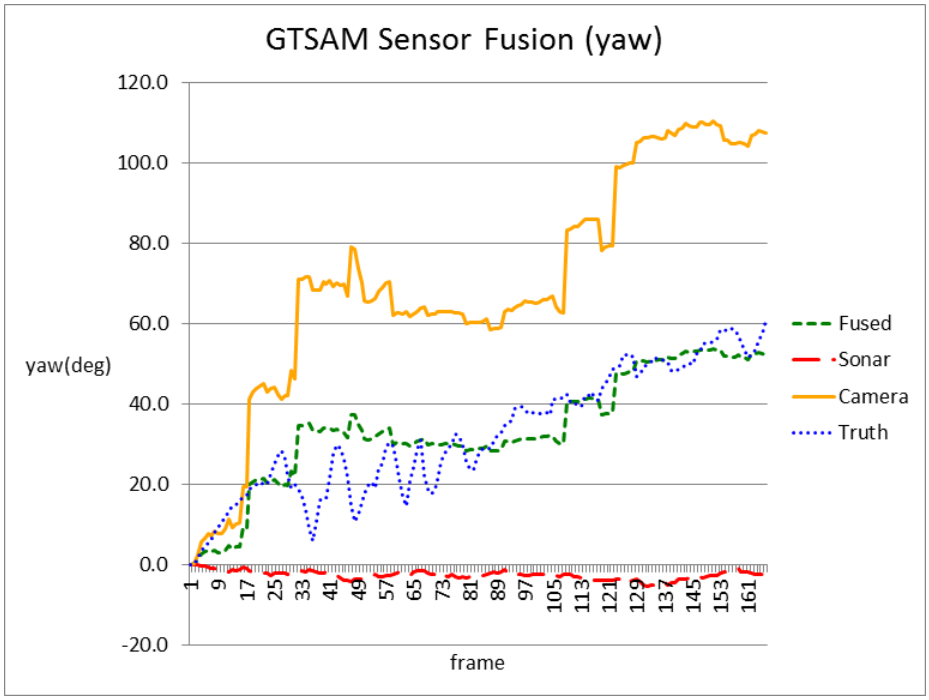


Figure 112: Yaw-only results of the fusion method showing the sonar, camera, and fused result in comparison to the ground truth estimated from the compass.

Full ground truth is not available with the Antarctic vehicle platform, so onboard inertial sensors were used to provide an estimate of all six degrees-of-freedom. The translational

degrees-of-freedom (surge, sway, heave) are subject to large drifts in the accelerometer-based INS. Therefore, these translational INS estimates do not provide useful information in many cases, as the vehicle was running for long durations prior to data collection, resulting in the accumulation of drift error. For validation of the sensor fusion algorithm on real-world data, estimated yaw from an onboard gyroscope is used, as it provides more accurate truth over the translational estimates. For a qualitative translational validation, intra-consistency is analyzed between the sonar and video datasets. Qualitative analysis of the translational results can also be undertaken based on examination of the sonar- and video-only trajectories versus the fusion output trajectory, to ensure that the fusion method performs as desired.

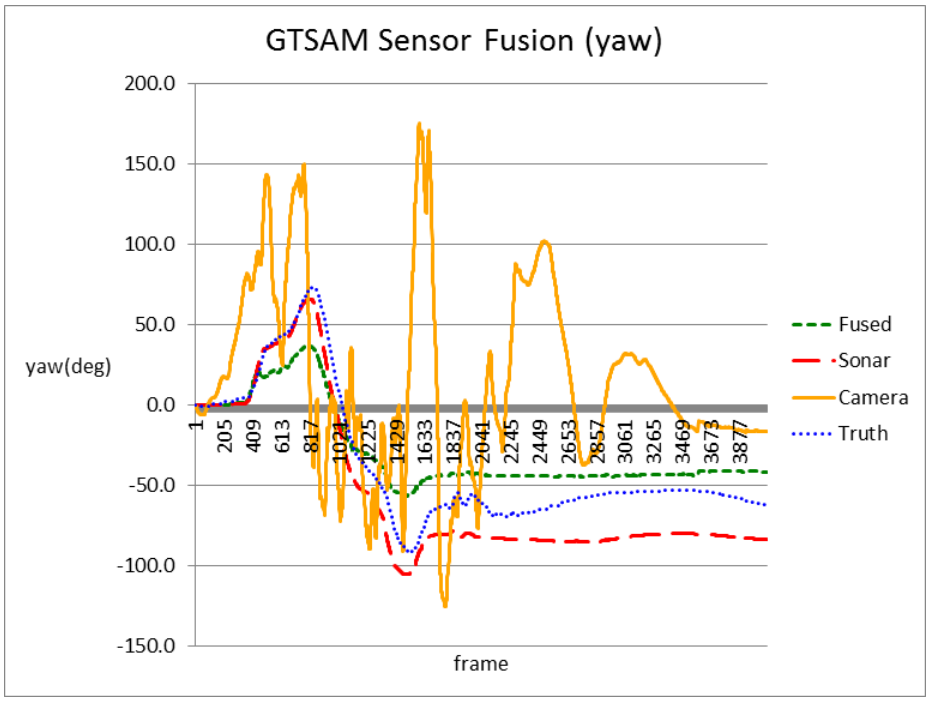


Figure 113: Yaw-only results of the fusion method showing the sonar, camera, and fused result in comparison to the ground truth estimated from the INS.

A plot of yaw rotation from an Antarctic dataset is presented in Figure 113. It can be seen from this plot that the sonar data is relatively accurate, while the camera data is noisy but follows the correct trend. The maximum accumulated error in the camera data is measured at 96.04% and the maximum sonar error is much lower at 10.95% of

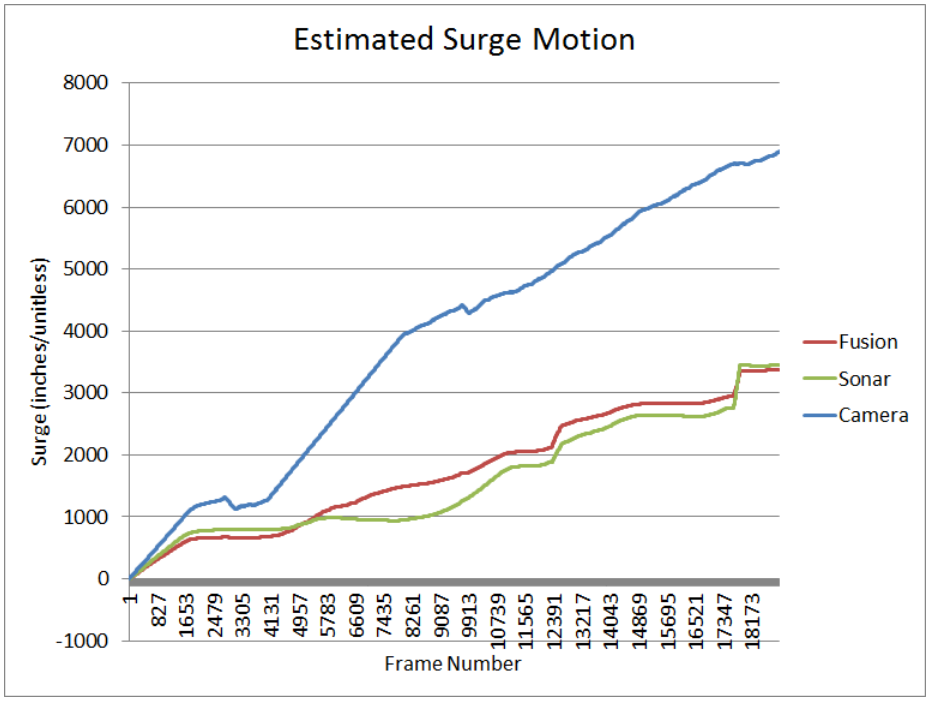


Figure 114: Surge estimation results of the fusion method showing the sonar, camera, and fused result for this Antarctic out and back dataset (camera results are not scaled)

the total 270 degrees of rotation. The fusion of these two inputs with the sensor fusion algorithm presented here results in an output favoring the more stable sonar data, with reduced noise over the camera data, that tracks well with the ground truth. In the fusion case, the maximum accumulated error is measured at only 14.93%, clearly rejecting the majority of the poor camera estimates in favor of the more reliable sonar estimates.

While ground truth translational position information is not available, it can be seen from the surge estimation plot in Figure 114 that the fused trajectory is influenced by both the sonar and camera estimates. In this case, the trajectory is comprised of mostly forward surge motion, clearly visible in the results as a steady increase in accumulation value. It is important to note that a camera-only surge estimate is shown despite the fact that these estimates are composed of a summation of unit vector components without scale. Here, a frame-to-frame magnitude shift of one inch is assumed for plotting purposes; therefore care must be taken when comparing the camera trajectory to the others. Additional results can be found in Appendix A.

8.5 Summary

Navigation under the ice is a difficult challenge. A factor graph sensor-fusion method for utilizing camera and sonar sensors already onboard many underwater vehicles to help bound drift rates encountered with inertial navigation systems is presented here. The work presented in this chapter utilizes the sonar- and camera-based relative pose estimation algorithms presented in Chapter 7 and Chapter 5 respectively. Using relative pose estimates between successive sensor frames, a combination of sonar- and camera-based information can provide a robust and full estimate of the vehicle state over the trajectory. A factor graph framework is used here to incorporate all available estimates in order to obtain an optimized trajectory result. Fusion of these two independent and complementary sensors combines the strengths of each sensor to overcome independent weaknesses, yielding a more robust and powerful system over independent sensor systems. The fusion method developed here was detailed in this chapter, and results were presented from testing over a variety of simulated and real-world under-ice datasets. Such a sensor fusion method can be used to reduce the error in under-ice navigation systems and increase mission capabilities.

CHAPTER 9

CONCLUSIONS

9.1 Summary and Conclusions

Exploration of under-ice environments, such as those found in Antarctica, has been the driving force in advances in the field of sub-ice vehicle development. Early exploration of these harsh regions was accomplished using human divers and manned submersibles. While these approaches are still used occasionally today, there has been a shift to favor the use of unmanned underwater vehicles (UUVs). The technology available with these tools has advanced drastically over the past two decades, and state-of-the-art UUVs can now provide amazing capabilities, while reducing the risk placed on human operators. While low-impact under-ice exploration is now possible using UUVs, these sub-ice environments still present many unique challenges to the vehicle platforms used in these deployments. GPS is not available for use under the ice, and thus navigation and localization is extremely difficult. Deployment of these vehicles beneath the ice presents a feat in and of itself, due to the harsh environments and logistical challenges. Once deployed, the environment encountered beneath the ice is difficult to explore. Almost no sunlight is transmitted through the thick ice cover, resulting in a lack of external light available for any onboard vehicle camera sensors. Even with a self-contained light solution on the vehicle, visibility can be very limited in these environments. Life under the ice is very uncommon, and the ice topography and seafloor is largely featureless and consists mostly of a simple repetitive-texture. This presents a major challenge for camera-based navigation, either by a human operator or a computer-vision algorithm. With no unique and re-observable features present, navigation methods based on landmarks is extremely difficult. For similar reasons, data collected from under the ice in these regions is largely featureless, and difficult to analyze. Most of these datasets provide hours of empty water column, ice-background, or void seafloor images. However, most vehicle deployments are motivated by exploration of these regions in search

of life, or interesting features amidst these vast, mostly empty environments. Therefore, careful analysis of each frame in the datasets must be tediously undertaken. These challenges encountered in the deployment of UUVs under the ice present unique opportunities for novel computer vision techniques to ease these difficulties.

This dissertation presented novel computer vision based algorithms for use in under-ice UUV deployments. Camera and sonar sensors are already present in the design of almost all UUVs. Camera-based and sonar-based relative pose estimation algorithms were presented to aid in the navigation problem encountered in these environments. A method for mapping ice texture and locations of any anomalies present against the ice was also presented here to ease in the scientific analysis of the under-ice video datasets. Novel methods for sensor fusion, using camera and sonar sensors, was presented in this dissertation to provide more robust techniques for landmark detection and relative pose estimation than those possible with single sensor systems. The algorithms developed here were evaluated using simulated under-ice datasets created for this purpose, as well as real-world under-ice datasets collected in Lake John, Colorado and McMurdo, Antarctica. These simulation methods and data collection deployments were detailed herein. The Icefin custom under-ice vehicle deployed for data collection in Antarctica was detailed in this dissertation. This platform represents a novel UUV design, specifically for polar under-ice deployments. The novel algorithms developed as part of this dissertation, and results from the evaluation of these methods using the simulated and real-world under-ice datasets, were discussed throughout this dissertation.

The specific contributions of this dissertation are as follows:

1. Development of a novel monocular camera relative pose estimation algorithm for use in low-contrast, featureless, under-ice environments

2. Development of new under-ice texture and anomaly mapping methods

3. Development of an innovative under-ice, sonar-based relative pose estimation algorithm using an optical flow and rigid motion model approach

4. Development of a novel under-ice sonar and video sensor fusion algorithm using a factor graph framework to combine the unique strengths of both sensors

9.1.1 Previous Work

A background of previous work in the various technical areas of this dissertation was presented in Chapter 2. While many prior works have investigated the use of sonar or camera sensors to aid in localization, the use of such a system in a feature-poor, under-ice environment has not been examined. The use of underwater sonar sensors for localization has been mostly limited to applications with blob features of significant acoustic return, or structured environments such as pools. Use of optical flow and point-feature-based algorithms with sonar, such as those considered here, is very uncommon in the literature. Camera-based monocular pose estimation applications remain largely terrestrial-based, with the few underwater applications commonly requiring feature-rich environments for successful results. While range and optical sensor fusion has been explored, use of underwater sonar sensors with camera sensors is very uncommon, especially in featureless and under-ice environments. Sonar- and video-based algorithms have not been previously used for automatic ice texture estimation and ice anomaly detection. Availability of previous data collection using forward-looking sonar and optical camera sensors in sub-ice environments is limited. Previous sonar and camera simulations of under-ice environments are not readily available from previous work. Therefore, datasets for the evaluation of the algorithms presented here are obtained here directly through simulation and under-ice field deployments. The under-ice UUV frontier is relatively new in and of itself, and the novel use of the algorithms presented here will help to advance capabilities in the field.

9.1.2 Under-Ice Data Collection and Vehicle Platforms

The simulation methods, vehicle platforms and field deployments, as well as the datasets produced for algorithm evaluation, were presented in Chapter 3. Evaluation of the algorithms presented in this dissertation requires sonar and video data obtained in sub-ice environments. The VideoRay Pro IV and custom Icefin vehicles were used for data collection beneath the ice in Lake John, Colorado and McMurdo, Antarctica respectively. Simulation methods were also developed here to produce additional datasets with full controllability and ground truth available. These simulated and real-world under-ice datasets were used for development and validation of the algorithms presented throughout this dissertation. The simulated datasets provided inherent ground truth, along with fine controllability, and thus were used for performance analysis of the algorithms presented in this dissertation.

The under-ice field deployment in Lake John, Colorado using the VideoRay Pro IV vehicle and BlueView P900-45 forward-looking sonar provided a wealth of data for use in algorithm development and analysis. Video and sonar data was collected for over 40 runs of the vehicle, using multiple software suites for redundancy. The qualitative results from this sensor angle evaluation were used for development of the Icefin vehicle.

The Icefin vehicle was designed as a modular unmanned underwater vehicle, with an extensive sensor suite, and an optical fiber tether for control and communication, in order to meet under-ice deployment challenges while obtaining high science return. A unique combination of human portability and a novel through-ice (small diameter hole) deployment method facilitates deployment of the modular Icefin vehicle out into the field. The Icefin vehicle is novel among state of the art under-ice vehicles; it contains a scientific and navigational sensor suite, autonomous capabilities, vectored thrusters in place of control planes, and a unique sensor module integration (rotatable by 180 degrees for ice missions). The Icefin vehicle's small size and human portability lowers logistical effort and cost. The Icefin vehicle provides a custom solution to the unique polar and under-ice deployment

challenges. The Icefin vehicle was deployed to McMurdo station in Antarctica from October to December of 2014. Designed for the harsh under-ice environment encountered on the McMurdo Ice Shelf, the Icefin vehicle was successfully deployed over multiple sub-ice exploration missions to obtain data of previously unexplored areas beneath the ice shelf. The data collected during deployment of the Icefin vehicle provided valuable real-world datasets for full evaluation of the algorithms presented in this dissertation.

9.1.3 Under-Ice Monocular Camera Relative Pose Estimation

A vision-based relative pose estimation algorithm was presented in Chapter 5 for use in low-contrast and feature-poor under-ice environments. This method utilizes a five-point algorithm for estimating relative camera pose between frames. Using adaptive contrast enhancement preprocessing and liberal feature-detection and matching thresholds on the front end, with a robust model estimation method on the back end, this relative pose algorithm, commonly used with feature-rich environments, was adapted for application in the much harsher environments considered here. This vision-based relative pose estimation algorithm was first evaluated using simulation datasets. The algorithm was also evaluated using real-world datasets, and the results track well with the ground truth compass data (in a Colorado dataset) over 550 degrees of rotation. An Antarctic dataset is presented in the results for qualitative validation of the algorithm, which tracks a mostly-surge trajectory with a steady course, as expected. The results from this camera-based relative pose estimation algorithm show promise for the use of such an approach in under-ice environments, despite the challenges encountered.

9.1.4 Texture Estimation and Anomaly Mapping

The texture and color-based mapping and anomaly detection algorithms were presented in Chapter 6. The texture estimation method presented here provides a quantitative estimate for the number and spread of the detected point features in a frame, corresponding to the texture of the background ice. Hue, or color, is also used to estimate the percentage of

pixels in the image that do not correspond to ice background yielding a good relative estimate of how "interesting" the colors in a frame are. Qualitative and quantitative analysis of the texture estimation algorithm with simulation data proves its utility, as four varying ice textures are correctly ranked by the estimated texture values. Qualitative analysis of the hue-based mapping algorithm shows consistency with the ground truth of the main simulated dataset. All of these algorithms also showed similar positive results with real-world datasets. These mapping algorithms can be run in parallel with the navigational algorithms also presented in this dissertation to best utilize the limited computational budget available on the vehicle platforms.

9.1.5 Sonar-based Relative Pose Estimation

A sonar-based relative pose estimation algorithm was presented in Chapter 7. The use of optical flow and point-feature computer vision methods is not common with sonar imagery. However, the application of such methodology to sonar datasets obtained in simulated and real-world under-ice datasets with positive results, presented in this chapter, proves the utility of such an approach. A preliminary investigation into the use of optical flow methods with sonar data was presented in this chapter, along with the final sonar-based relative pose estimation algorithm. Results from evaluation of these methods using simulated and real-world sonar data was presented in this chapter as well. The preliminary investigation into the use of optical flow methods with sonar data proved the utility of such an approach through statistical analysis of the optical flow results over a sonar dataset with human-annotated ground truth. Optical flow methods were able to effectively estimate direction and magnitude of any shift between frames, while rejecting false data. These promising results led to further development of a more robust model for motion estimation between sonar frames. Multiple preprocessing and optical flow techniques were evaluated with sonar data during this initial investigation, and it was determined that Gaussian smoothing should be applied to the sonar images to reduce noise prior to evaluation, and that the Lucas-Kanade optical flow based algorithm performed the best with such unique datasets.

Once the final sonar-based relative pose estimation algorithm was developed, it was evaluated using the simulated and real-world under-ice datasets obtained here. The effective use of optical flow with sonar data in featureless environments presents a novel contribution with promising results. The application of such a method for relative pose estimation can help aid in navigation in challenging under-ice environments.

9.1.6 Factor Graph Sensor Fusion

A factor graph sensor-fusion method for utilizing camera and sonar sensors already onboard many underwater vehicles to help bound drift rates encountered with inertial navigation systems was presented in Chapter 8. The work presented in this chapter utilizes the sonar- and camera-based relative pose estimation algorithms presented in Chapter 7 and Chapter 5 respectively. Using relative pose estimates between successive sensor frames, a combination of sonar- and camera-based information can provide a robust and full estimate of the vehicle state over the trajectory. A factor graph framework is used here to incorporate all available estimates in order to obtain an optimized trajectory result. Fusion of these two independent and complementary sensors combines the strengths of each sensor to overcome independent weaknesses, yielding a more robust and powerful system over independent sensor systems. The fusion method developed here was detailed in this chapter, and results were presented from testing over a variety of simulated and real-world under-ice datasets. Results from this analysis showed a decrease in error for multiple fused output results, over both independent sensor estimates. In almost all other cases, reduced error amounts, over the least accurate single-sensor input, are produced by the sensor fusion estimate. Qualitative analysis of the sensor fusion algorithm with real-world datasets, for both rotational and translation cases, shows the expected influence of both sonar and camera estimates on the fusion output result. The novel factor graph sensor fusion method developed in this dissertation presents a robust algorithm for incorporation of multiple complementary but partially redundant sensor datasets, to provide a more capable navigational aid. Such a sensor fusion method can be used to help bound the error in under-ice navigation systems,

and increase mission capabilities.

9.2 Contributions

The unique contributions of this work include development of multiple algorithms to aid in navigation and mapping in under-ice environments. An under-ice simulation environment is developed for creation of both camera and sonar datasets. The design of a custom under-ice vehicle is presented, along with novel lessons learned from Antarctic deployment, and extrapolation of such a deployment to future planetary exploration. The adaptation of a vision-based relative pose estimation algorithm is presented for use in low-contrast and mostly featureless under-ice environments. The novel use of optical flow with sonar data is shown to be a useful approach, despite its uncommon use. One main contribution of this thesis is the development of a relative pose estimation method using a forward-looking multibeam sonar sensor. The final main contribution of this dissertation is the novel factor graph sensor fusion method presented here for robust relative pose estimation in under-ice environments using sonar and camera sensors, already onboard many UUVs. The contributions of this dissertation will help advance the capabilities of under-ice UUVs to provide greater mission potential in these challenging environments.

9.3 Possible Topics For Future Research

There are many additional interesting topics for investigation in this area, which are out of the scope of this dissertation. Presented in this final section are some of these topics as open investigation problems for future development.

Development of the custom Icefin vehicle for polar sub-ice deployment provided a valuable investigation into the unique challenges of field robotics in such a harsh environment. The lessons learned from the 2014 Antarctic deployment of this vehicle should be exploited in the future development of new under-ice vehicles. Future development of the Icefin vehicle will include additional sensors and modules, tuning of the vehicle, and refinement

of the inertial navigation system. Investigations into the use of power-over-fiber could provide greater mission durations for the vehicle and reduce the dependence on batteries. Development of a similar under-ice vehicle for planetary deployment to Europa is a main open research problem presented here. Lessons learned from vehicles such as Icefin can be extrapolated to the more challenging problem faced with the development of such a planetary exploration platform. Further sub-ice deployment of the Icefin vehicle is planned for Greenland in 2016 and a return to Antarctica in future seasons.

The use of a monocular vision system for sub-ice relative pose estimation, which showed promising results here, should be investigated as a navigational aid for use in similar low-contrast, feature-poor environments. Additional vision-based relative pose estimation approaches could be investigated for use in these under-ice environments, as a more robust and accurate approach may be available in the future. The use of a multiple-camera approach (instead of a monocular system) for under-ice navigation is presented here as an open research problem, and could provide more accurate and robust results if two cameras are available on the UUV.

Mapping methods commonly used with open-water UUVs should be investigated for use in under-ice environments. Methods such as structure-from-motion might be considered to create three-dimensional representations of the under-ice environment. Additional vision-based mapping techniques should also be investigated to provide other valuable information for scientific investigations. Additional future research in this area could also include a fusion of the two anomaly detection algorithms presented here, to increase robustness.

The initial development of a sonar-based relative pose estimation method, presented here, proves the utility of such optical flow and feature-based methods with sonar data. This opens up a large area of research on the application of other computer-vision methods with sonar datasets. The sonar-based relative pose estimation algorithm presented here assumes use in planar, ice-covered environments. However, this could be extrapolated in

future research to additional applications, such as deployment in areas of drastically varying topography.

The factor graph sensor fusion method presented in this dissertation proves the utility of the integration of sonar- and camera-based relative pose estimation into the navigational framework of a UUV. However, other available positional information was used as ground truth for algorithm evaluation here, and not integrated into the system. Future research should investigate the fusion of other available navigational sensors into this factor graph framework approach. The loop-closure problem, not possible in repetitive-texture environments with no unique features, should be further investigated in these under-ice environments. Loop-closure constraints can be easily integrated into the factor graph sensor fusion framework developed here, and can yield substantial reductions in error drift rates.

All of the computer vision methods developed in this dissertation should be integrated and evaluated in real-time embedded UUV platforms. While the theory of these applications is presented here, application of such methods during deployment of UUVs would aid in navigation and data collection. Adaptation of the algorithms presented here for a parallel architecture is presented as an open investigation problem, to provide a more capable, efficient implementation. These final algorithms should be integrated into the Icefin vehicle control and data collection software architecture in future research. Further evaluation and adaptation of these algorithms to an even wider range of environments is suggested to provide further utility and robustness to the methods.

APPENDIX A

ADDITIONAL RELATIVE POSE ESTIMATION RESULTS

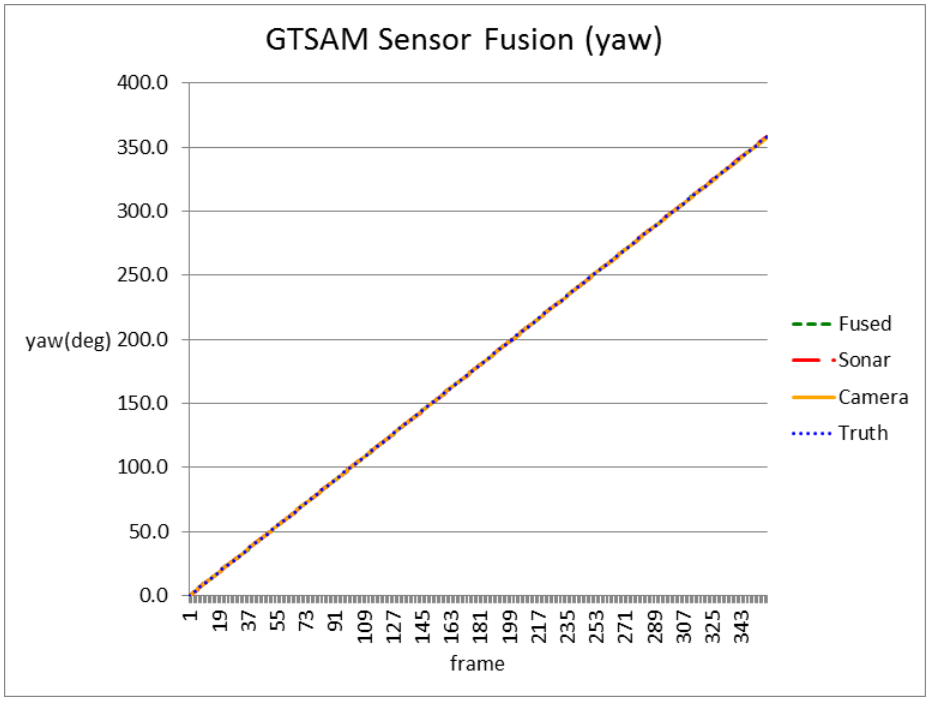


Figure 115: Simulated rotation dataset: estimated yaw over 360 degrees (maximum accumulated error = 0.22% (camera), 0.06% (sonar), 0.11% (fusion))

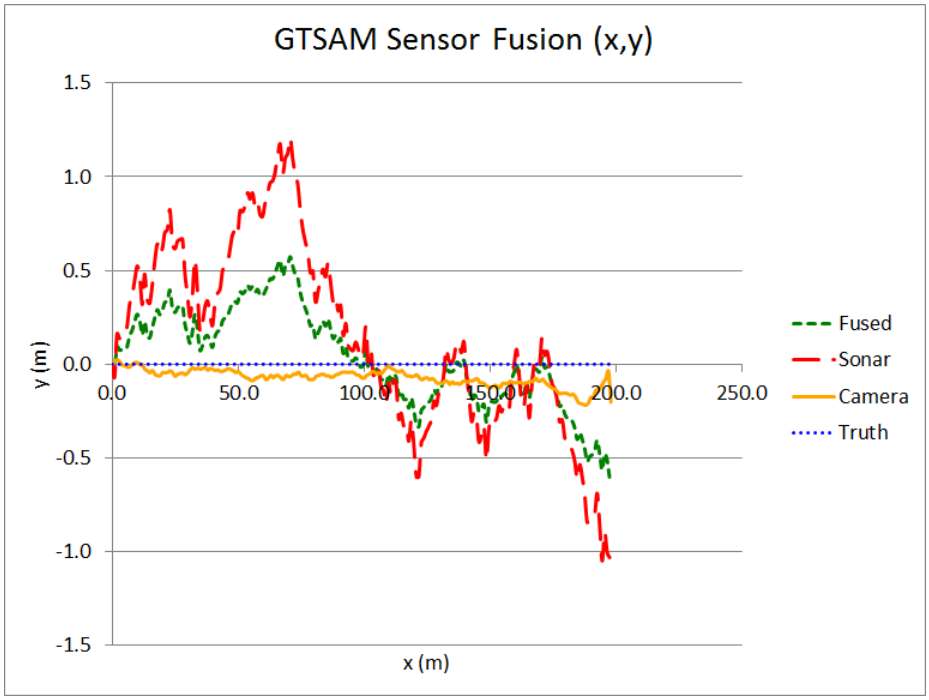


Figure 116: Simulated surge-only dataset: estimated trajectory over 200 meters (maximum accumulated y-error = 0.11% (camera), 0.61% (sonar), 0.31% (fusion))

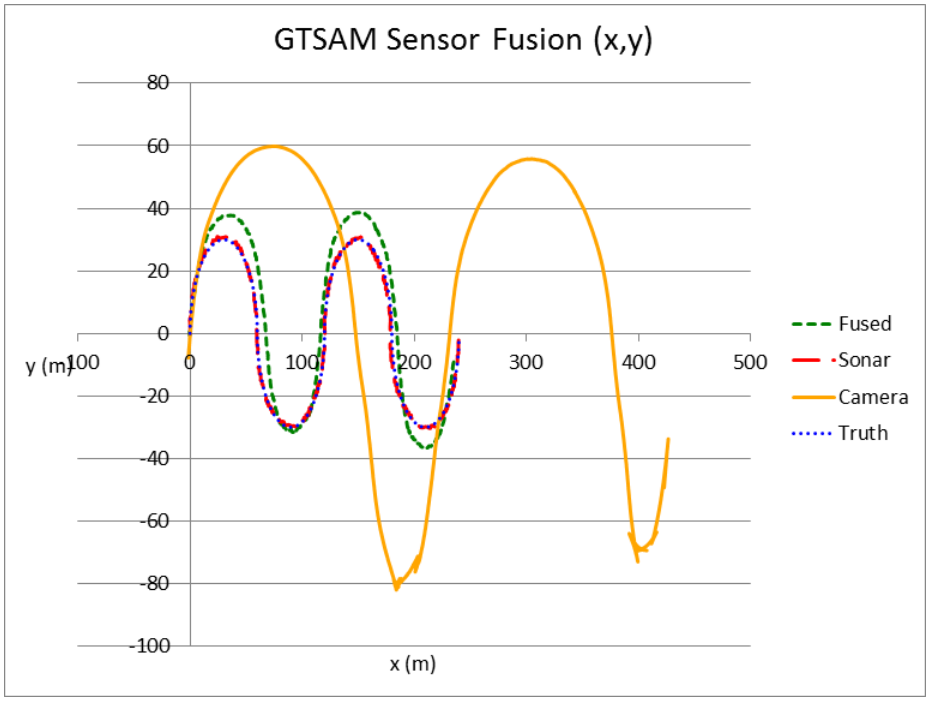


Figure 117: Simulated s-curve estimated trajectory over 377 meters (maximum accumulated error = N/A (camera), 0.55% (sonar), 2.61% (fusion))

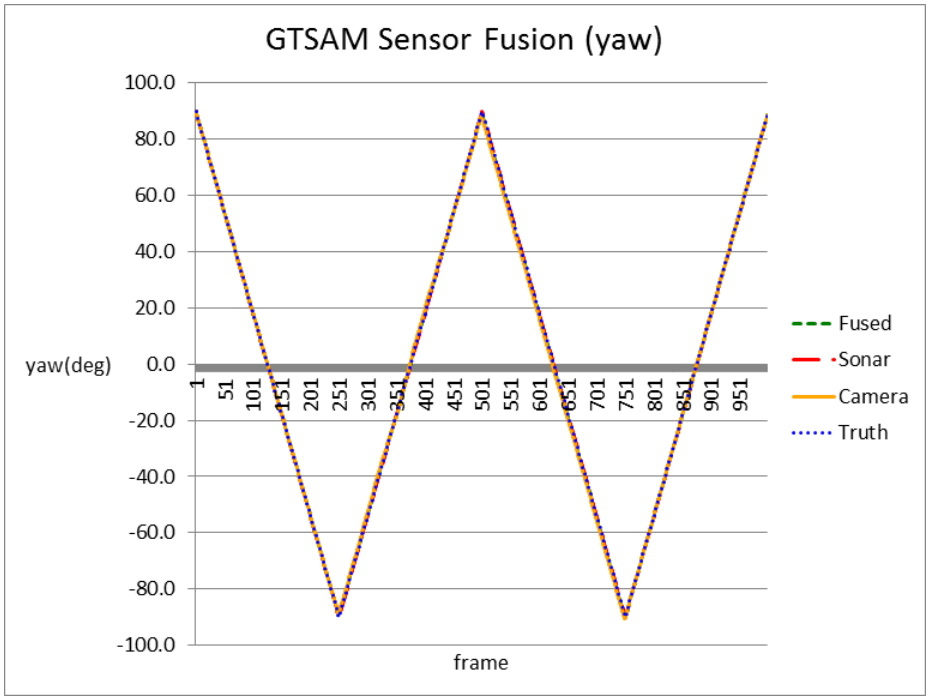


Figure 118: Simulated s-curve estimated yaw over 720 degrees (maximum accumulated error = 0.33% (camera), 0.04% (sonar), 0.18% (fusion))

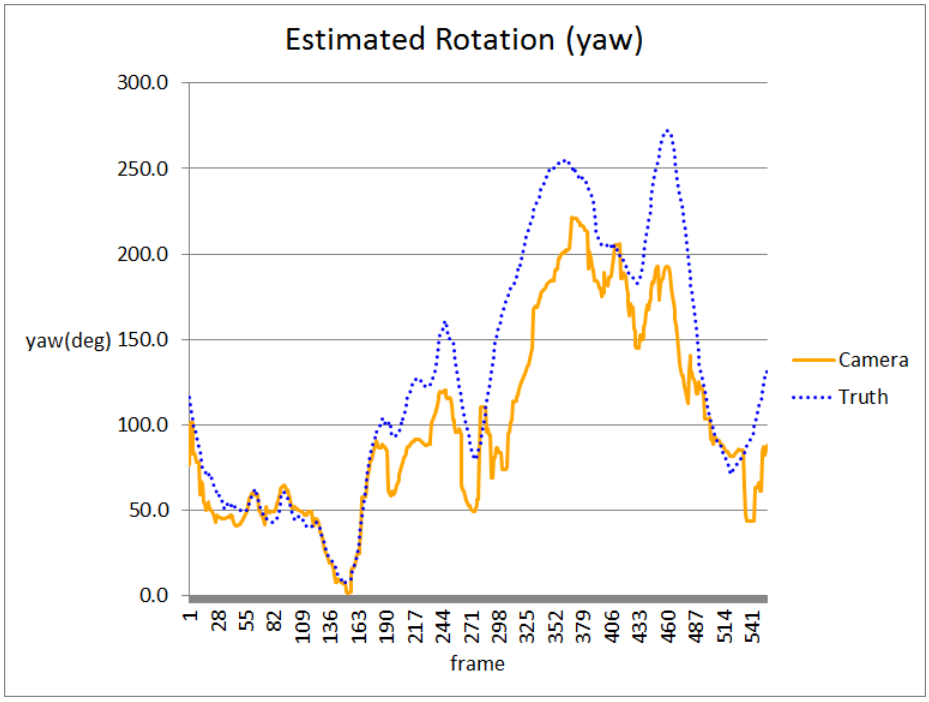


Figure 119: Colorado dataset: camera estimated yaw over 850 degrees (maximum accumulated error = 11.77%, final accumulated error = 5.24%)

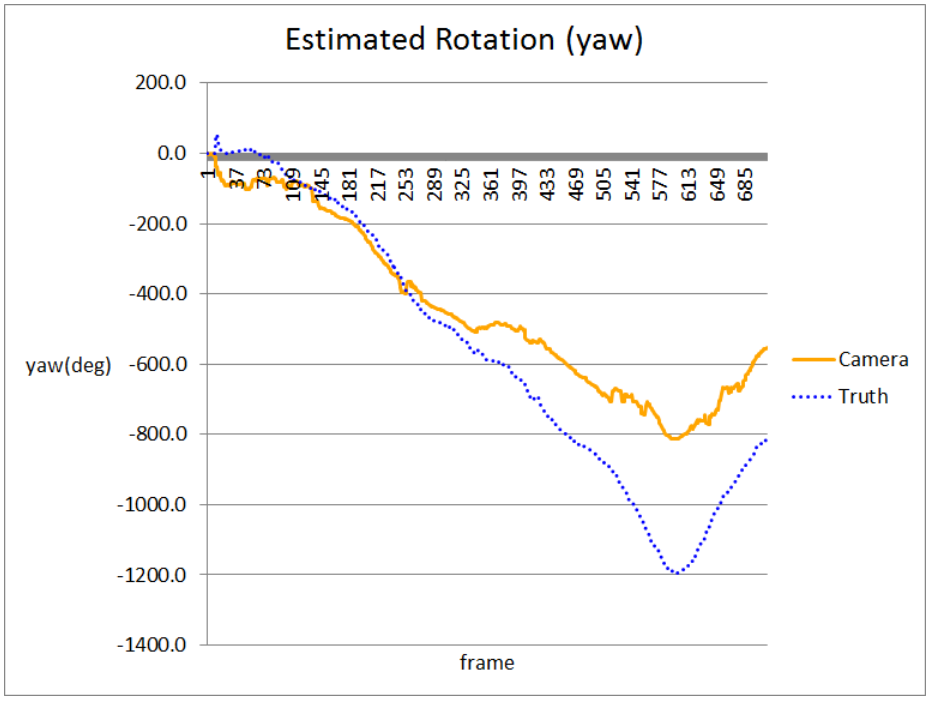


Figure 120: Colorado dataset: camera estimated yaw over 1600 degrees (maximum accumulated error = 24.3%, final accumulated error = 16.28%)

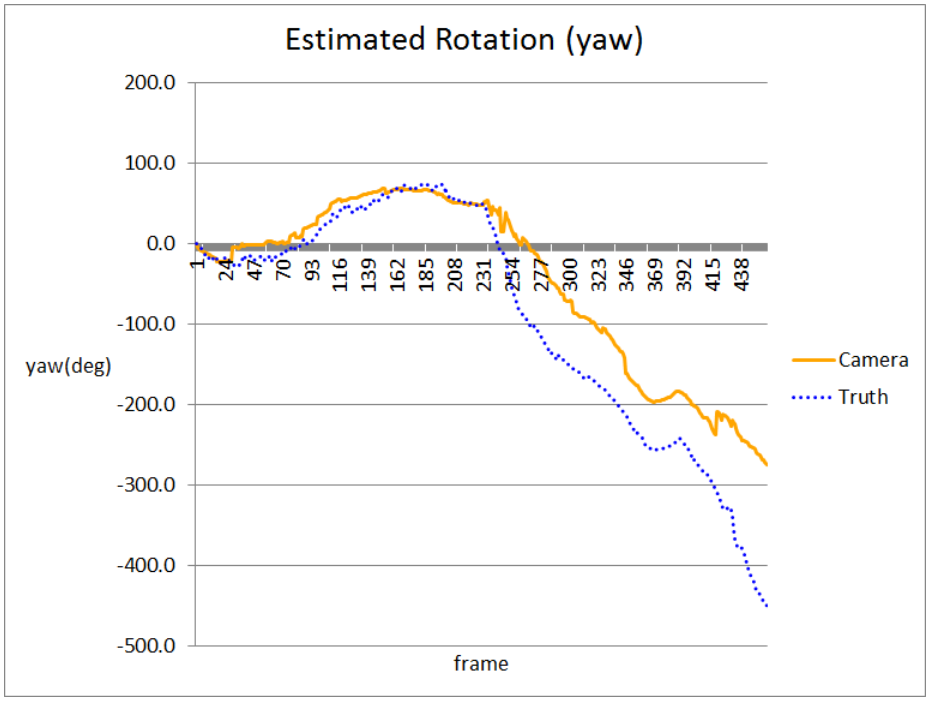


Figure 121: Colorado dataset: camera estimated yaw over 550 degrees (maximum accumulated error = 32.1%, final accumulated error = 31.96%)

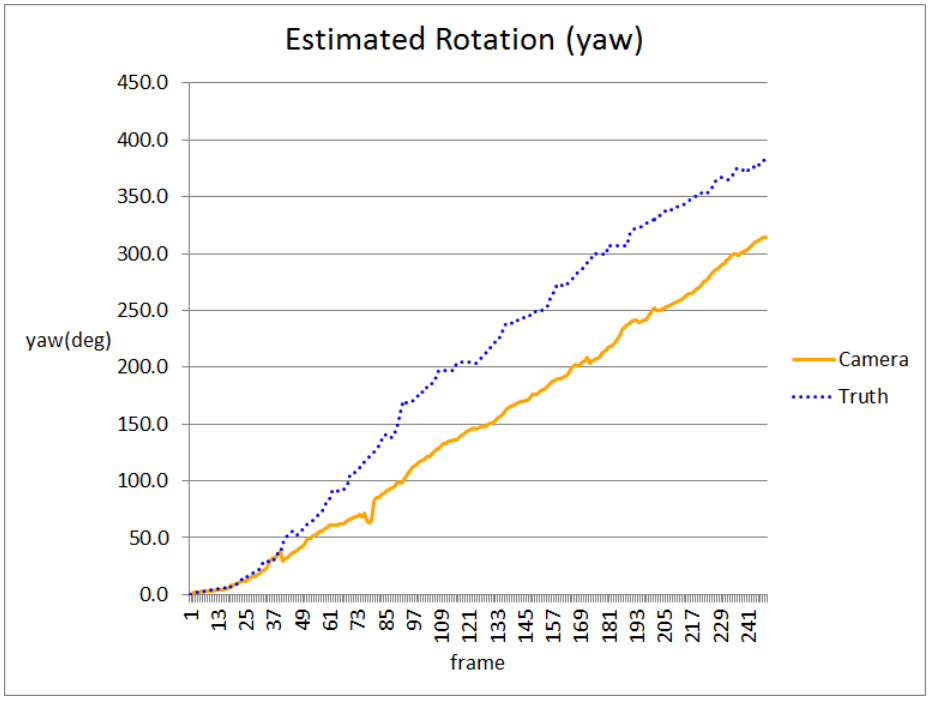


Figure 122: Colorado dataset: camera estimated yaw over 375 degrees (maximum accumulated error = 24.95%, final accumulated error = 18.46%)

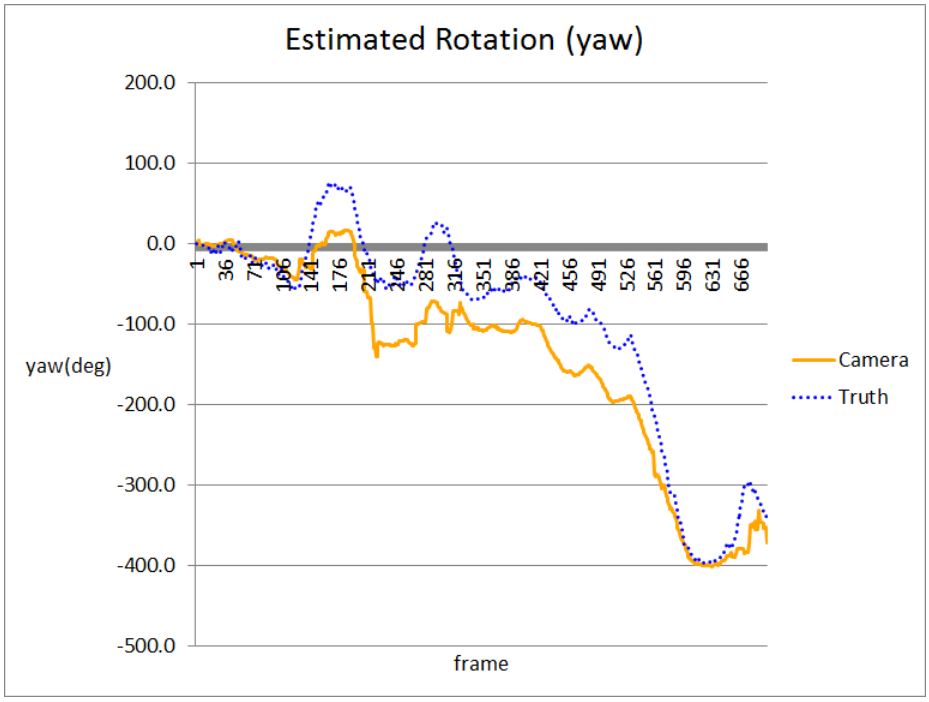


Figure 123: Colorado dataset: camera estimated yaw over 1000 degrees (maximum accumulated error = 12.7%, final accumulated error = 3.23%)

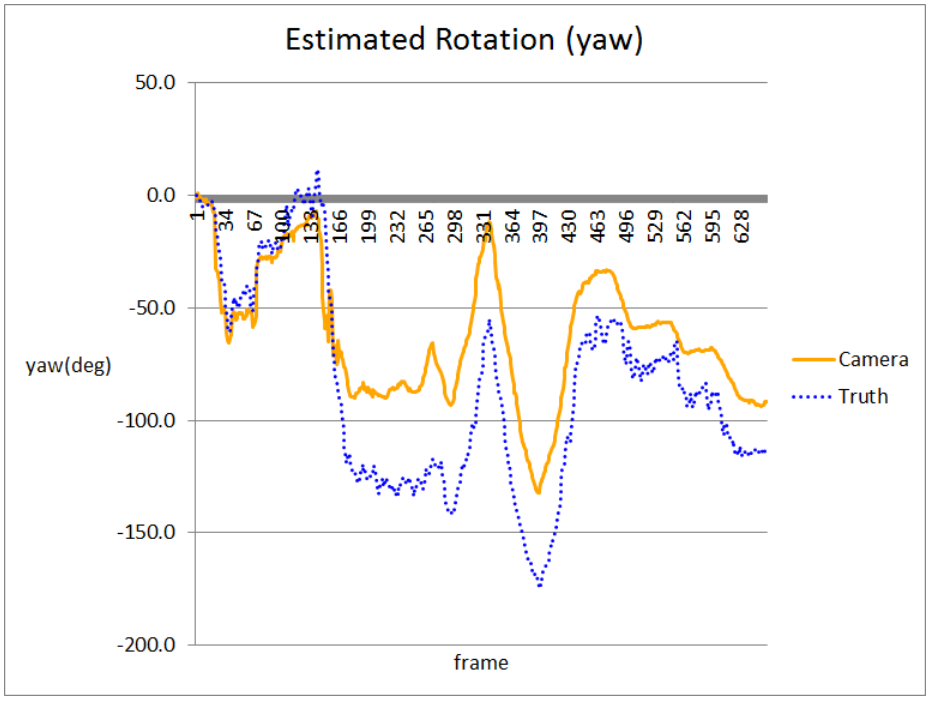


Figure 124: Colorado dataset: camera estimated yaw over 550 degrees (maximum accumulated error = 11.19%, final accumulated error = 4.14%)

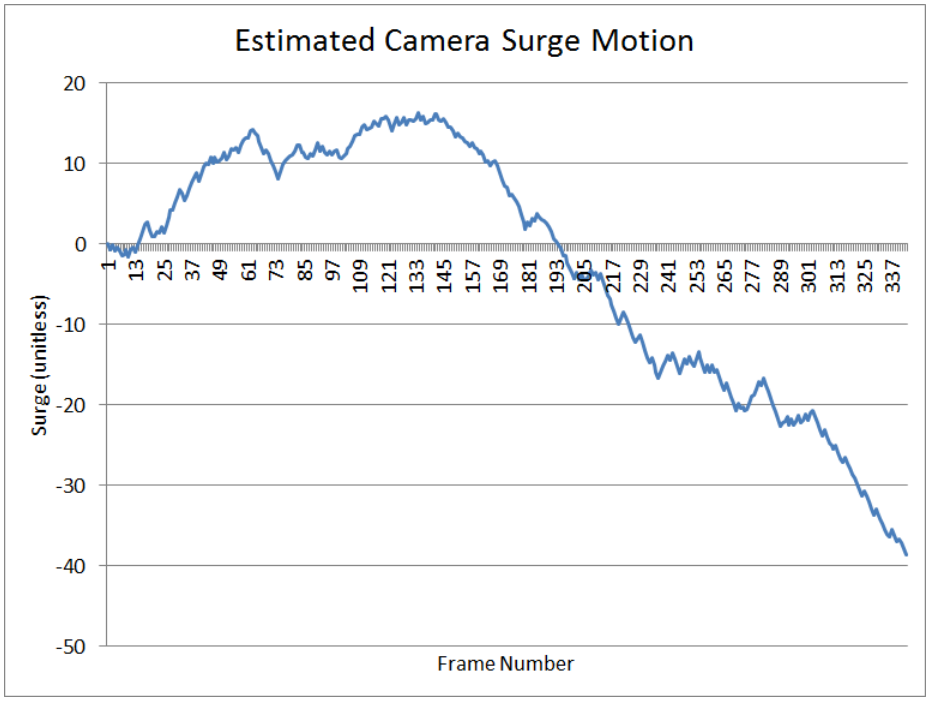


Figure 125: Colorado dataset: camera estimated surge representation for out and back trajectory (qualitative)

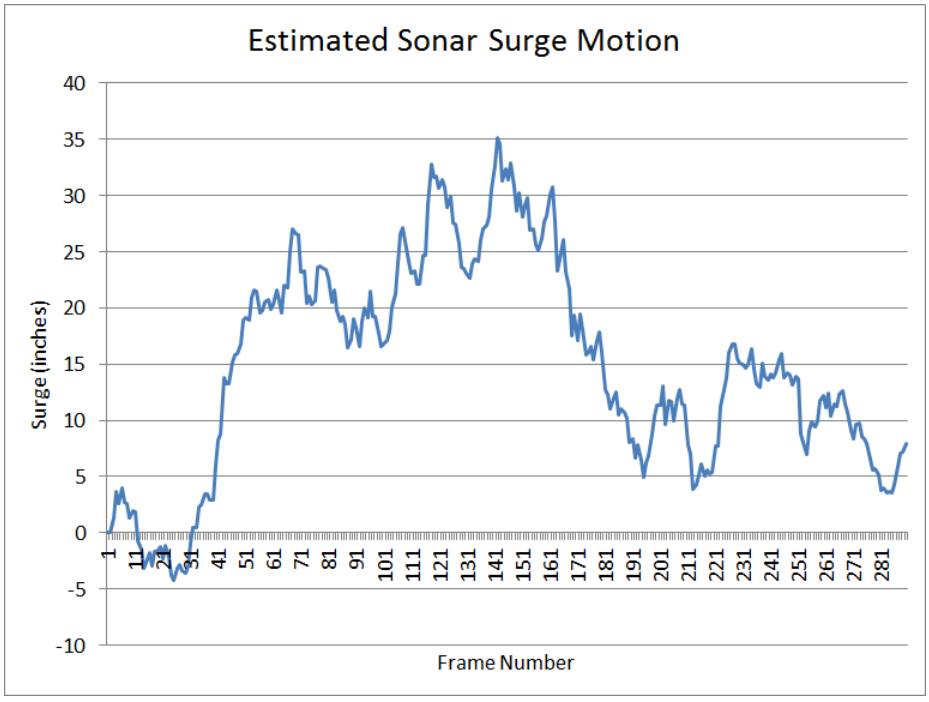


Figure 126: Colorado dataset: sonar estimated surge representation for out and back trajectory (qualitative)

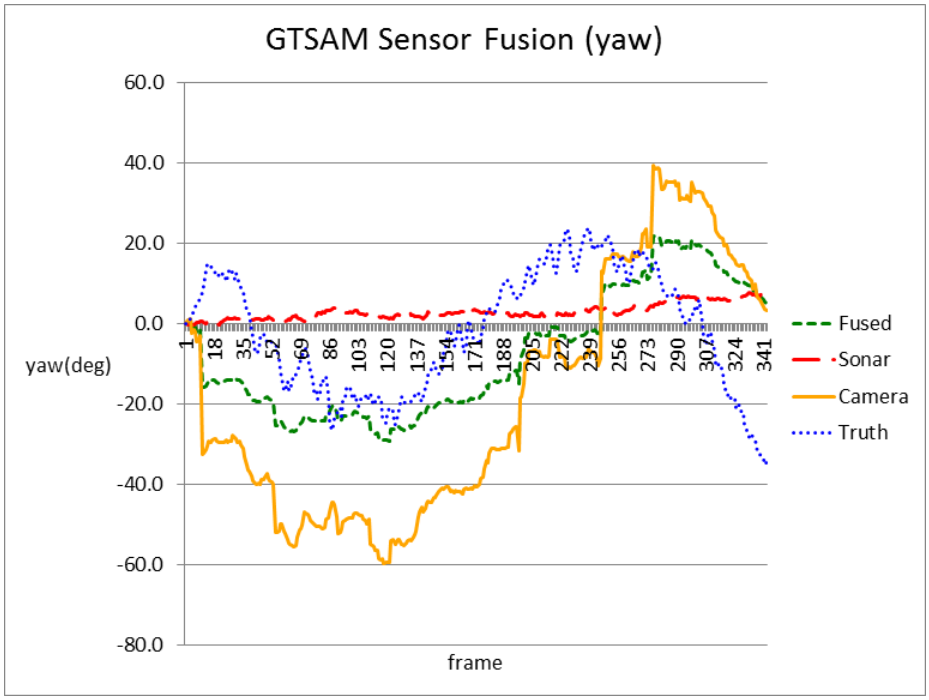


Figure 127: Colorado dataset: estimated yaw over 135 degrees (maximum accumulated error = 54.7% (camera), 57.7% (sonar), 56.15% (fusion))

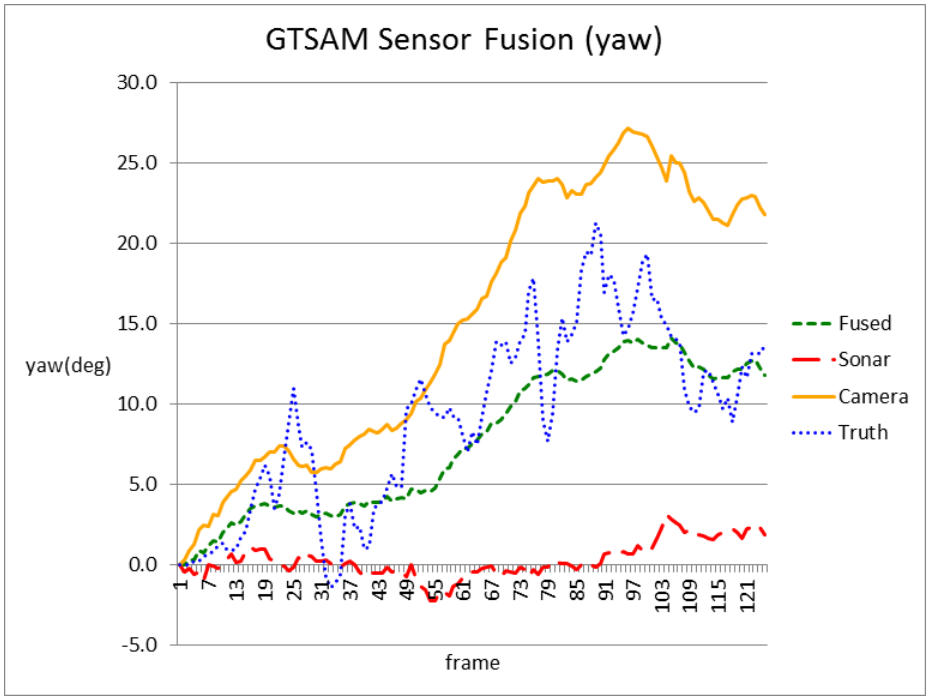


Figure 128: Colorado dataset: estimated yaw over 70 degrees (maximum accumulated error = 23.2% (camera), 30.7% (sonar), 13.31% (fusion), final accumulated error = 11.6% (camera), 16.9% (sonar), 2.64% (fusion))

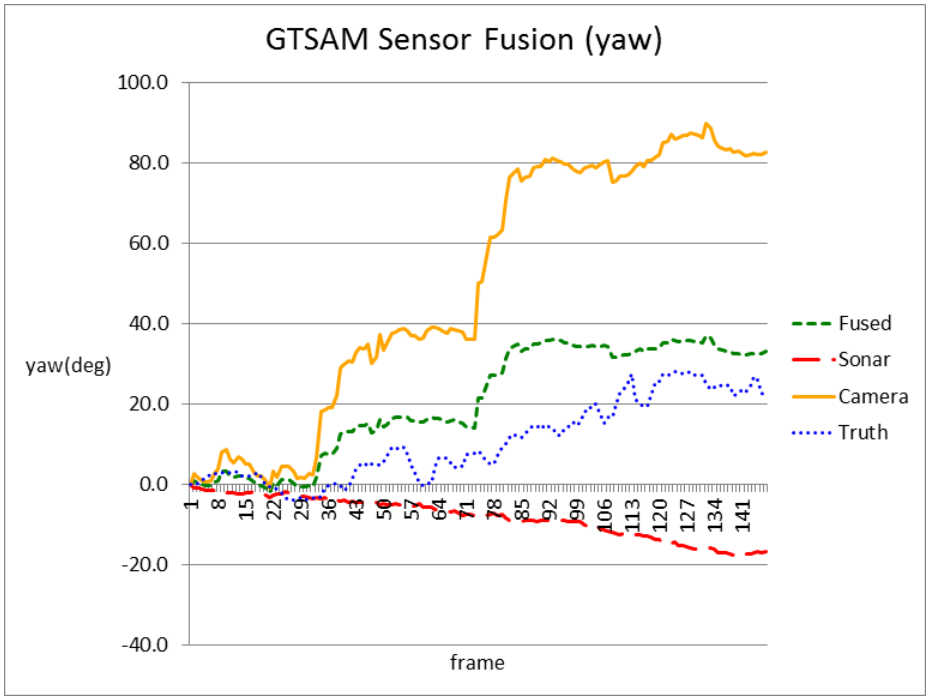


Figure 129: Colorado dataset: estimated yaw over 20 degrees (maximum accumulated error = 341.65% (camera), 219.85% (sonar), 117.85% (fusion), final accumulated error = 308.1% (camera), 189.35% (sonar), 59.4% (fusion))

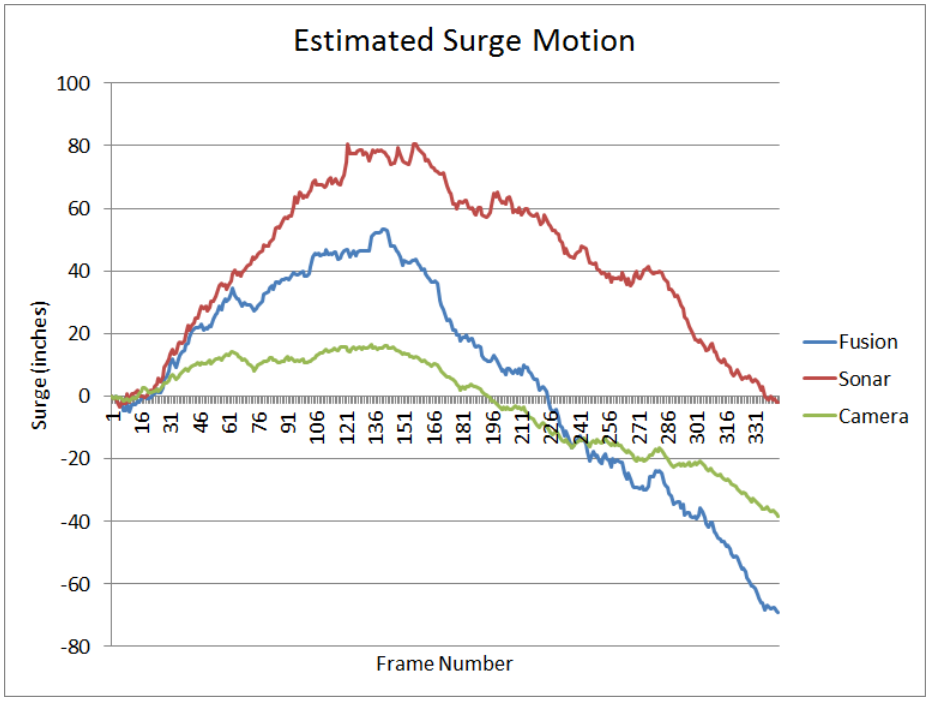


Figure 130: Colorado dataset: estimated surge representation for out and back trajectory (qualitative)

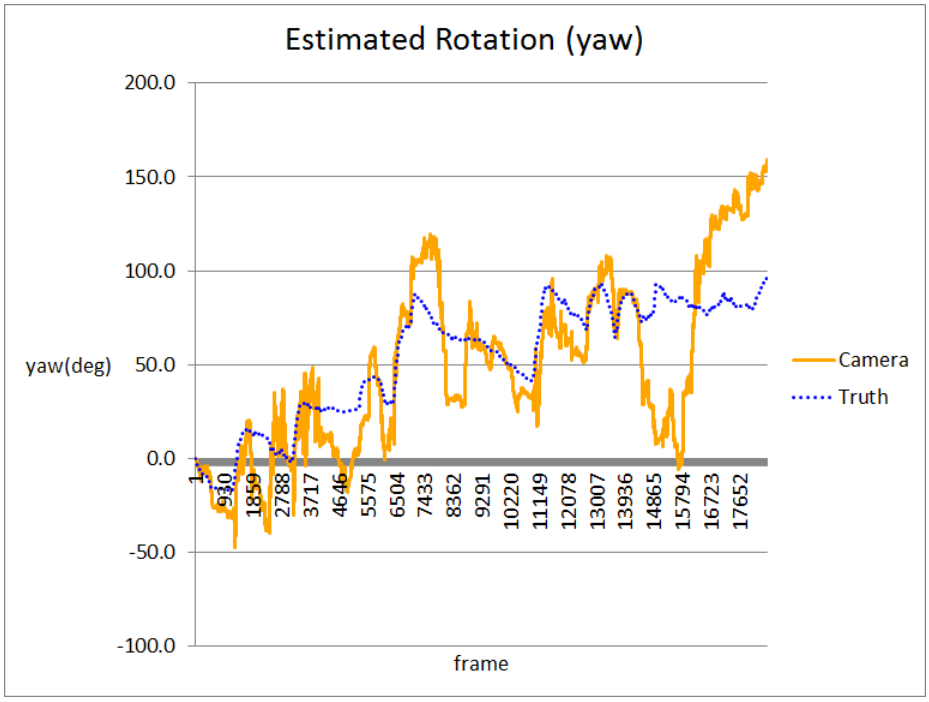


Figure 131: Antarctica dataset: camera estimated yaw over 275 degrees (maximum accumulated error = 32.75%, final accumulated error = 23.11%)

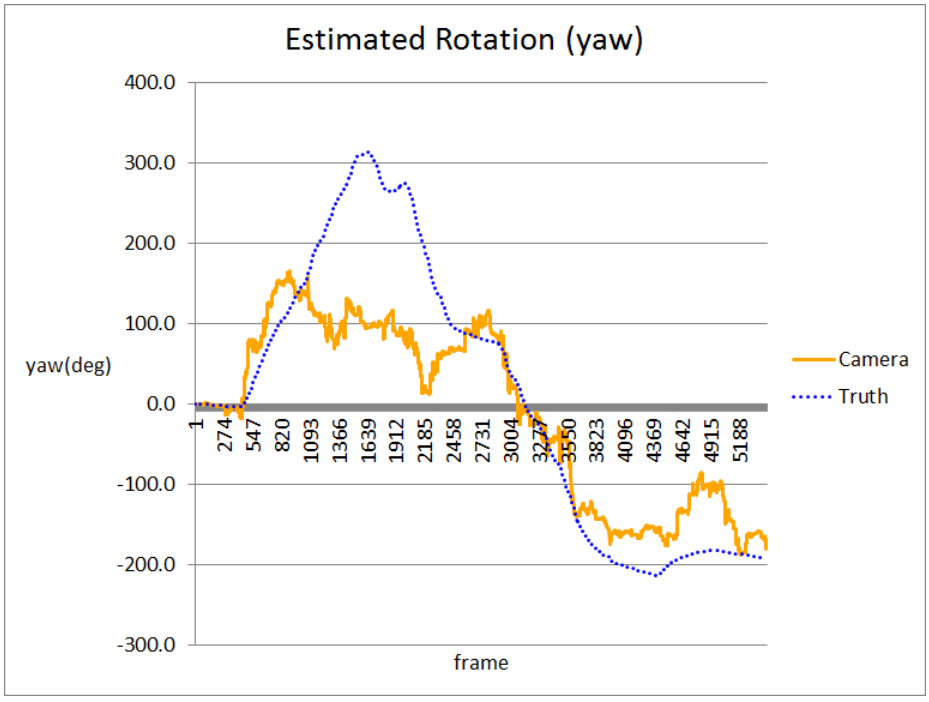


Figure 132: Antarctica dataset: camera estimated yaw over 800 degrees (maximum accumulated error = 27.37%, final accumulated error = 2.18%)

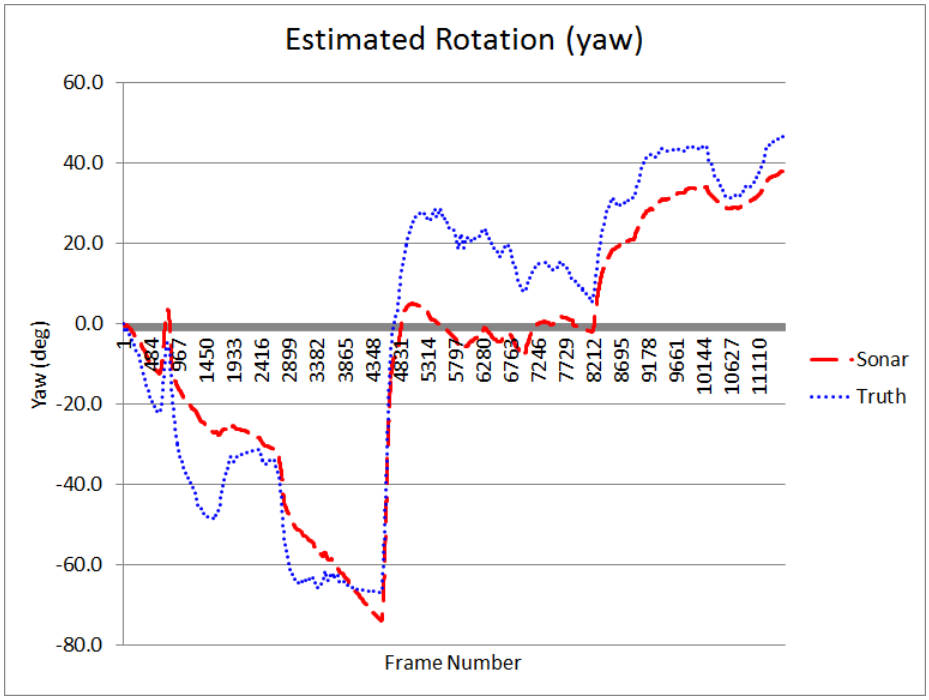


Figure 133: Antarctica dataset: sonar estimated yaw over 300 degrees (maximum accumulated error = 9.59%, final accumulated error = 2.85%)

REFERENCES

[1] P. Wadhams, "Arctic ice cover, ice thickness and tipping points," *Ambio*, vol. 41, no. 1, pp. 23–33, 2012.

[2] B. Schmidt, D. Blankenship, G. Patterson, and P. Schenk, "Active formation of chaos terrain over shallow subsurface water on Europa," *Nature*, vol. 479, no. 7374, pp. 502–505, 2011.

[3] F. Cazenave, R. Zook, D. Carroll, M. Flagg, and S. Kim, "Development of the ROV SCINI and deployment in McMurdo sound, Antarctica.," *Journal of Ocean Technology*, vol. 6, no. 3, 2011.

[4] A. D. Bowen, D. R. Yoerger, C. C. German, J. C. Kinsey, M. V. Jakuba, D. Gomez-Ibanez, C. L. Taylor, C. Machado, J. C. Howland, C. L. Kaiser, *et al.*, "Design of nereid-ui: A remotely operated underwater vehicle for oceanographic access under ice," in *Oceans-St. John's, 2014*, pp. 1–6, IEEE, 2014.

[5] C. Kunz, C. Murphy, H. Singh, C. Pontbriand, R. A. Sohn, S. Singh, T. Sato, C. Roman, K.-i. Nakamura, M. Jakuba, *et al.*, "Toward extraplanetary under-ice exploration: Robotic steps in the Arctic," *Journal of Field Robotics*, vol. 26, no. 4, pp. 411–429, 2009.

[6] W. Stone, B. Hogan, C. Flesher, S. Gulati, K. Richmond, A. Murarka, G. Kuhlman, M. Sridharan, V. Siegel, R. Price, *et al.*, "Design and deployment of a four-degrees-of-freedom hovering autonomous underwater vehicle for sub-ice exploration and mapping," *Engineering for the Maritime Environment*, vol. 224, no. 4, pp. 341–361, 2010.

[7] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 2, pp. 99–110, 2006.

[8] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[9] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," 2012.

[10] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1365–1378, 2008.

[11] U. Blender.org, "Blender," *Retrieved February*, vol. 4, p. 2015, 2015.

[12] M. Daly, F. Rack, and R. Zook, "Edwardsiella andrillae, a New Species of Sea Anemone from Antarctic Ice," *PloS one*, vol. 8, no. 12, p. e83476, 2013.

[13] U. CIA, "The world factbook," *Retrieved February*, vol. 4, p. 2015, 2015.

[14] D. Drewry, S. Jordan, and E. Jankowski, "Measured properties of the antarctic ice sheet: surface configuration, ice thickness, volume and bedrock characteristics," *Annals of Glaciology*, vol. 3, pp. 83–91, 1983.

[15] USAP, "Mcmurdo station weather," *Retrieved February*, vol. 4, p. 2015, 2015.

[16] NSIDC, "Arctic sea ice news and analysis," *Retrieved February*, vol. 4, p. 2015, 2015.

[17] P. Brueggeman, "Diving under Antarctic ice: a history," *Scripps Institution of Oceanography*, 2003.

[18] P. Wadhams, "The use of autonomous underwater vehicles to map the variability of under-ice topography," *Ocean Dynamics*, vol. 62, no. 3, pp. 439–447, 2012.

[19] M. J. Doble, A. L. Forrest, P. Wadhams, and B. E. Laval, "Through-ice AUV deployment: Operational and technical experience from two seasons of Arctic fieldwork," *Cold Regions Science and Technology*, vol. 56, no. 2, pp. 90–97, 2009.

[20] A. Plueddemann, G. Packard, J. Lord, and S. Whelan, "Observing Arctic coastal hydrography using the REMUS AUV," in *Autonomous Underwater Vehicles, 2008. AUV 2008. IEEE/OES*, pp. 1–4, IEEE, 2008.

[21] R. McEwen, H. Thomas, D. Weber, and F. Psota, "Performance of an AUV navigation system at Arctic latitudes," *Oceanic Engineering, IEEE Journal of*, vol. 30, no. 2, pp. 443–454, 2005.

[22] S. D. McPhail, M. E. Furlong, M. Pebody, J. Perrett, P. Stevenson, A. Webb, and D. White, "Exploring beneath the PIG Ice Shelf with the Autosub3 AUV," in *OCEANS 2009-EUROPE*, pp. 1–8, IEEE, 2009.

[23] J. Ferguson, "Adapting AUVs for use in Under-ice Scientific Missions," in *OCEANS, 2008. OCEANS 2008. IEEE*, pp. 1–5, IEEE, 2008.

[24] J. Ferguson, "1000 km Under Ice with an AUV - Setting the Stage for Future Achievement," in *Symposium on and Workshop on Scientific Use of Submarine Cables and Related Technologies (SSC)*, IEEE, 2011.

[25] T. Crees, C. Kaminski, J. Ferguson, J. M. Laframboise, A. Forrest, J. Williams, E. MacNeil, D. Hopkin, and R. Pederson, "UNCLOS under ice survey-an historic AUV deployment in the canadian high arctic," in *OCEANS 2010*, pp. 1–8, IEEE, 2010.

[26] M. G. Kivelson, K. K. Khurana, C. T. Russell, M. Volwerk, R. J. Walker, and C. Zimmer, "Galileo magnetometer measurements: A stronger case for a subsurface ocean at europa," *Science*, vol. 289, no. 5483, pp. 1340–1343, 2000.

[27] E. S. Team *et al.*, "Europa study 2012 report," *National Aeronautics and Space Administration*, 2012.

[28] R. Pappalardo, J. Head, R. Greeley, R. Sullivan, C. Pilcher, G. Schubert, W. Moore, M. Carr, J. Moore, M. Belton, *et al.*, "Geological evidence for solid-state convection in europa's ice shell," *Nature*, vol. 391, no. 6665, pp. 365–368, 1998.

[29] M. Craven, I. Allison, H. A. Fricker, and R. Warner, "Properties of a marine ice layer under the amery ice shelf, east antarctica," *Journal of Glaciology*, vol. 55, no. 192, pp. 717–728, 2009.

[30] P. Kimball and S. Rock, "Sonar-based iceberg-relative AUV navigation," in *Autonomous Underwater Vehicles, 2008. AUV 2008. IEEE/OES*, pp. 1–6, IEEE, 2008.

[31] P. Wadhams and M. Doble, "Digital terrain mapping of the underside of sea ice from a small AUV," *Geophysical Research Letters*, vol. 35, no. 1, 2008.

[32] J. Bellingham, E. D. Cokelet, and W. J. Kirkwood, "Observation of warm water transport and mixing in the Arctic basin with the ALTEX AUV," in *Autonomous Underwater Vehicles, 2008. AUV 2008. IEEE/OES*, pp. 1–5, IEEE, 2008.

[33] K. Richmond, S. Gulati, C. Flesher, B. P. Hogan, and W. C. Stone, "Navigation, control, and recovery of the ENDURANCE under-ice hovering AUV," in *International Symposium on Unmanned Untethered Submersible Technology UUST*, Citeseer, 2009.

[34] M. V. Jakuba, C. N. Roman, H. Singh, C. Murphy, C. Kunz, C. Willis, T. Sato, and R. A. Sohn, "Long-baseline acoustic navigation for under-ice autonomous underwater vehicle operations," *Journal of Field Robotics*, vol. 25, pp. 861–879, 2008.

[35] V. Siegel, W. Stone, B. Hogan, S. Lelievre, and C. Flesher, "Development and testing of a laser-powered cryobot for outer planet icy moon exploration," in *AGU Fall Meeting Abstracts*, vol. 1, p. 11, 2013.

[36] S. E. Webster, C. M. Lee, J. Gobat, *et al.*, "Preliminary results in under-ice acoustic navigation for seagliders in davis strait," in *Oceans-St. John's, 2014*, pp. 1–5, IEEE, 2014.

[37] T. Wulff, S. Lehmenhecker, E. Bauerfeind, U. Hoge, K. Shurn, and M. Klages, "Biogeochemical research with an autonomous underwater vehicle: Payload structure and arctic operations," in *OCEANS-Bergen, 2013 MTS/IEEE*, pp. 1–10, IEEE, 2013.

[38] A. Spears, M. West, B. Schmidt, T. Collins, and A. Howard, "Modification of the Yellowfin Autonomous Underwater Vehicle for use in Under-ice Missions," in *AUVSIs Unmanned Systems Conference 2014*, pp. 1–9, AUVSI, 2014.

[39] A. Spears, M. Meister, B. Schmidt, M. West, T. Collins, and A. Howard, "Design and Development of an Under-Ice Autonomous Underwater Vehicle for use in Polar Regions," in *OCEANS 2014*, pp. 1–6, IEEE, 2014.

[40] W. Stone, D. Wettergreen, G. Kantor, M. Stevens, E. Hui, E. Franke, and B. Hogan, "Depthx (deep phreatic thermal explorer)," in *Proceedings of the 14th International Symposium on Unmanned Untethered Submersible Technology (UUST)*, 2005.

[41] C. Kunz and H. Singh, "Map building fusing acoustic and visual information using autonomous underwater vehicles," *Journal of Field Robotics*, vol. 30, no. 5, pp. 763–783, 2013.

[42] N. Pelavas, C. E. Lucas, and G. J. Heard, "Short range localization of Autonomous Underwater Vehicles," in *OCEANS 2011*, pp. 1–5, IEEE, 2011.

[43] C. M. Ciany and J. Huang, "Computer aided detection/computer aided classification and data fusion algorithms for automated detection and classification of underwater mines," in *OCEANS 2000*, vol. 1, pp. 277–284, IEEE, 2000.

[44] Y. Petillot, I. Tena Ruiz, and D. M. Lane, "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar," *Oceanic Engineering, IEEE Journal of*, vol. 26, no. 2, pp. 240–251, 2001.

[45] D. M. Lane, M. J. Chantler, and D. Dai, "Robust tracking of multiple objects in sector-scan sonar image sequences using optical flow motion estimation," *Oceanic Engineering, IEEE Journal of*, vol. 23, no. 1, pp. 31–46, 1998.

[46] D. M. Lane, M. Chantler, D. Y. Dai, and I. Tena Ruiz, "Tracking and classification of multiple objects in multibeam sector scan sonar image sequences," in *Underwater Technology 1998*, pp. 269–273, IEEE, 1998.

[47] J. L. Leonard, R. N. Carpenter, and H. J. S. Feder, "Stochastic mapping using forward look sonar," *Robotica*, vol. 19, no. 05, pp. 467–480, 2001.

[48] J. Folkesson and J. Leonard, "Autonomy through SLAM for an Underwater Robot," in *Robotics Research*, pp. 55–70, Springer, 2011.

[49] D. Ribas, P. Ridao, J. D. Tardos, and J. Neira, "Underwater SLAM in a marina environment," in *Intelligent Robots and Systems (IROS)*, pp. 1455–1460, IEEE, 2007.

[50] S. Majumder, S. Scheding, and H. F. Durrant-Whyte, "Multisensor data fusion for underwater navigation," *Robotics and Autonomous Systems*, vol. 35, no. 2, pp. 97–108, 2001.

[51] T. Masek, *Acoustic image models for navigation with forward-looking sonars*. PhD thesis, Monterey, California. Naval Postgraduate School, 2008.

[52] E. Trucco, Y. R. Petillot, I. T. Ruiz, K. Plakas, and D. M. Lane, "Feature tracking in video and sonar subsea sequences with applications," *Computer Vision and Image Understanding*, vol. 79, no. 1, pp. 92–122, 2000.

[53] J. Folkesson, J. Leonard, J. Leederkerken, and R. Williams, "Feature tracking for underwater navigation using sonar," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 3678–3684, IEEE, 2007.

[54] A. A. Bennett and J. J. Leonard, "A behavior-based approach to adaptive feature detection and following with autonomous underwater vehicles," *Oceanic Engineering, IEEE Journal of*, vol. 25, no. 2, pp. 213–226, 2000.

[55] S. Barkby, *Efficient and Featureless Approaches to Bathymetric Simultaneous Localisation and Mapping*. PhD thesis, University of Sydney., 2011.

[56] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "A featureless approach to efficient bathymetric SLAM using distributed particle mapping," *Journal of Field Robotics*, vol. 28, no. 1, pp. 19–39, 2011.

[57] B. Calder, L. Linnett, and D. Carmichael, "Bayesian approach to object detection in sidescan sonar," in *Vision, Image and Signal Processing, IEE Proceedings-*, vol. 145, pp. 221–228, IET, 1998.

[58] J. Tegowski, A. Zielinski, and A. Kruss, "Parametrical and Textural Analysis of Sidescan Sonar Images of the Seafloor," in *OCEANS 2007*, pp. 1–6, IEEE, 2007.

[59] A. Mallios, P. Ridao, D. Ribas, and E. Hernández, "Scan matching SLAM in underwater environments," *Autonomous Robots*, vol. 36, no. 3, pp. 181–198, 2014.

[60] B. Technologies, "P900 Series 2D Imaging Sonar Data Sheet," 2011.

[61] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[62] C. Harris and M. Stephens, "A combined corner and edge detector.," in *Alvey vision conference*, vol. 15, p. 50, Manchester, UK, 1988.

[63] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[64] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[65] K. Kim, N. Neretti, and N. Intrator, "Mosaicing of acoustic camera images," in *Radar, Sonar and Navigation, IEE Proceedings-*, vol. 152, pp. 263–270, IET, 2005.

[66] H. Sekkati and S. Negahdaripour, "3-D motion estimation for positioning from 2-D acoustic video imagery," in *Pattern Recognition and Image Analysis*, pp. 80–88, Springer, 2007.

[67] S. Negahdaripour, P. Firoozfam, and P. Sabzmeydani, "On processing and registration of forward-scan acoustic video imagery," in *Computer and Robot Vision, 2005. Proceedings. The 2nd Canadian Conference on*, pp. 452–459, IEEE, 2005.

[68] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[69] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American statistical association*, vol. 79, no. 388, pp. 871–880, 1984.

[70] B. K. Horn and B. G. Schunck, "Determining optical flow," in *1981 Technical Symposium East*, pp. 319–331, International Society for Optics and Photonics, 1981.

[71] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision.," in *IJCAI*, vol. 81, pp. 674–679, 1981.

[72] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, 2012.

[73] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[74] S. G. Johnson and M. Deaett, "The application of automated recognition techniques to side-scan sonar imagery," *Oceanic Engineering, IEEE Journal of*, vol. 19, no. 1, pp. 138–144, 1994.

[75] P. Cervenka and C. de Moustier, "Sidescan sonar image processing techniques," *Oceanic Engineering, IEEE Journal of*, vol. 18, no. 2, pp. 108–122, 1993.

[76] R. M. Eustice, *Large-area visually augmented navigation for autonomous underwater vehicles*. PhD thesis, MIT and Woods Hole Oceanographic Institution, 2005.

[77] K.-T. Song and W.-H. Tang, "Environment perception for a mobile robot using double ultrasonic sensors and a CCD camera," *Industrial Electronics, IEEE Transactions on*, vol. 43, no. 3, pp. 372–379, 1996.

[78] C. Beall, B. J. Lawrence, V. Ila, and F. Dellaert, "3D reconstruction of underwater structures," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 4418–4423, IEEE, 2010.

[79] A. Kim and R. M. Eustice, "Real-time visual SLAM for autonomous underwater hull inspection using visual saliency," in *Conference on Intelligent Robots and Systems*, vol. 1, p. 2, 2013.

[80] J. Salvi, Y. Petillot, S. Thomas, and J. Aulinas, "Visual slam for underwater vehicles using video velocity log and natural landmarks," in *OCEANS*, pp. 1–6, IEEE, 2008.

[81] S. Williams and A. M. Howard, "Developing monocular visual pose estimation for arctic environments," *Journal of Field Robotics*, vol. 27, no. 2, pp. 145–157, 2010.

[82] K. Zuiderveld, "Contrast limited adaptive histogram equalization," in *Graphics gems IV*, pp. 474–485, Academic Press Professional, Inc., 1994.

[83] C. Beall, F. Dellaert, I. Mahon, and S. B. Williams, "Bundle adjustment in large-scale 3d reconstructions based on underwater robotic surveys," in *OCEANS, 2011 IEEE-Spain*, pp. 1–6, IEEE, 2011.

[84] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, pp. 524–531, IEEE, 2005.

[85] A. Kim and R. Eustice, "Pose-graph visual SLAM with geometric model selection for autonomous underwater ship hull inspection," in *Intelligent Robots and Systems. IROS 2009. IEEE/RSJ International Conference on*, pp. 1559–1565, IEEE, 2009.

[86] J. Aulinas, M. Carreras, X. Llado, J. Salvi, R. Garcia, R. Prados, and Y. R. Petillot, "Feature extraction for underwater visual SLAM," in *OCEANS, 2011 IEEE-Spain*, pp. 1–7, IEEE, 2011.

[87] D. Nistér, "An efficient solution to the five-point relative pose problem," *Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–770, 2004.

[88] P. H. S. Torr, "Bayesian model estimation and selection for epipolar geometry and generic manifold fitting," *International Journal of Computer Vision*, vol. 50, no. 1, pp. 35–61, 2002.

[89] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, pp. I–652, IEEE, 2004.

[90] P. H. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[91] N. Carlevaris-Bianco, A. Mohan, J. R. McBride, and R. M. Eustice, "Visual localization in fused image and laser range data," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 4378–4385, IEEE, 2011.

[92] M. Häselich, M. Arends, D. Lang, and D. Paulus, "Terrain Classification with Markov Random Fields on fused Camera and 3D Laser Range Data.," in *ECMR*, pp. 153–158, 2011.

[93] X. Zhang, A. B. Rad, and Y.-K. Wong, "Sensor fusion of monocular cameras and laser rangefinders for line-based simultaneous localization and mapping (SLAM) tasks in autonomous mobile robots," *Sensors*, vol. 12, no. 1, pp. 429–452, 2012.

[94] F. Sun, Y. Zhou, C. Li, and Y. Huang, "Research on active SLAM with fusion of monocular vision and laser range data," in *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pp. 6550–6554, IEEE, 2010.

[95] Z. Zhu, H.-P. Chiu, T. Oskiper, S. Ali, R. Hadsell, S. Samarasekera, and R. Kumar, "High-precision localization using visual landmarks fused with range data," in *Computer Vision and Pattern Recognition, Conference on*, pp. 81–88, IEEE, 2011.

[96] S. Williams and I. Mahon, "Simultaneous localisation and mapping on the great barrier reef," in *Robotics and Automation, 2004. International Conference on*, vol. 2, pp. 1771–1776, IEEE, 2004.

[97] M. Johnson-Roberson, O. Pizarro, and S. Willams, "Towards large scale optical and acoustic sensor integration for visualization," in *OCEANS*, pp. 1–4, IEEE, 2009.

[98] S. Negahdaripour, H. Sekkati, and H. Pirsiavash, "Opti-acoustic stereo imaging: On system calibration and 3-D target reconstruction," *Image Processing, IEEE Transactions on*, vol. 18, no. 6, pp. 1203–1214, 2009.

[99] M. Mahlisch, R. Schweiger, W. Ritter, and K. Dietmayer, "Sensorfusion using spatio-temporal aligned video and lidar for improved vehicle detection," in *Intelligent Vehicles Symposium, 2006 IEEE*, pp. 424–429, IEEE, 2006.

[100] L. Matthies and A. Elfes, "Integration of sonar and stereo range data using a grid-based representation," in *Robotics and Automation*, pp. 727–733, IEEE, 1988.

[101] P. Weckesser, R. Dillmann, and U. Rembold, "Sensor-fusion of intensity-and laser range-images," in *Multisensor Fusion and Integration for Intelligent Systems, 1996. IEEE/SICE/RSJ International Conference on*, pp. 501–508, IEEE, 1996.

[102] M. Hebert, "Outdoor scene analysis using range data," in *Robotics and Automation. Proceedings. IEEE International Conference on*, vol. 3, pp. 1426–1432, IEEE, 1986.

[103] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. International Conference on*, vol. 3, pp. 2301–2306, IEEE, 2004.

[104] J. Neira, J. D. Tardós, J. Horn, and G. Schmidt, "Fusing range and intensity images for mobile robot localization," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 1, pp. 76–84, 1999.

[105] H. Singh, G. Salgian, R. Eustice, and R. Mandelbaum, "Sensor fusion of structure-from-motion, bathymetric 3D, and beacon-based navigation modalities," in *Robotics and Automation, 2002. International Conference on*, pp. 4024–4031, IEEE, 2002.

[106] D. W. Krout, G. Okopal, and E. Hanusa, "Video data and sonar data: Real world data fusion example.," in *FUSION*, pp. 1–5, 2011.

[107] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *International journal of computer vision*, vol. 73, no. 2, pp. 213–238, 2007.

[108] W. Skarbek, A. Koschan, T. Bericht, Z. Veroffentlichung, *et al.*, "Colour image segmentation-a survey," 1994.

[109] J. Sattar and G. Dudek, "On the performance of color tracking algorithms for underwater robots under varying lighting and visibility," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 3550–3555, IEEE, 2006.

[110] V. LLC, "VideoRay Pro 4 Datasheet," 2014.

[111] W. Wang, *Autonomous control of a differential thrust micro ROV*. PhD thesis, Waterloo, Canada. University of Waterloo, 2007.

[112] A. Spears, M. West, and T. Collins, "Autonomous Control and Simulation of the VideoRay Pro III Vehicle Using MOOS and IvP Helm," in *OCEANS 2013*, pp. 1–10, IEEE, 2013.

[113] T. Aerospace, "Frangibolt," *Retrieved February*, 2015.

[114] XMOS, "Xmos xc-2 hardware manual," vol. 1.4, 2015.

[115] G. Inc., "Balefire product sheet," *Retrieved February*, 2015.

[116] L. KT&C co., "WDR380 Operation Manual," 2008.

[117] D. Power and Light, "HD Multi SeaCam Specifications," 2014.

[118] K. Maritime, "OE14 Light Ring Colour Camera Datasheet," 2015.

[119] Z. Zhang, "A flexible new technique for camera calibration," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 11, pp. 1330–1334, 2000.

[120] J.-Y. Bouguet, "Camera calibration toolbox for Matlab," 2004.

[121] J.-Y. Bouguet, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel Corporation*, vol. 5, 2001.

[122] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[123] B. K. Horn, "Recovering baseline and orientation from essential matrix," *J. Optical Society of America*, 1990.

[124] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[125] A. Waite, *Sonar for Practising Engineers*. 2002.

[126] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Image Analysis*, pp. 363–370, Springer, 2003.

[127] O. Sorkine, "Least-squares rigid motion using SVD," *Technical notes*, vol. 120, p. 3, 2009.

[128] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis*, pp. 105–116, Springer, 1978.