Advances in simultaneous localization and mapping in confined underwater environments using sonar and optical imaging

by

Paul Ozog

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in the University of Michigan
2016

Doctoral Committee:

Associate Professor Ryan M. Eustice, Chair
Associate Professor Jason J. Corso
Assistant Professor Matthew Johnson-Roberson
Assistant Professor Shai Revzen

2016

ACKNOWLEDGMENTS

During the summer of 2010, I found myself on a boat in the middle of Lake Huron helping Ryan, Ayoung, and Jeff test two really expensive underwater robots, and, in Ryan's words, "get a feel for what our lab does." I only just moved to Ann Arbor two days prior, and during the course of the day my feelings of excitement and curiosity slowly transitioned into overwhelming sense of dread: by lunchtime I was one sudden jolt away from barfing all over the cabin of our tiny boat. "Paul," I thought to myself, "what *did* you just get yourself into...?"

Well, I am happy to report that I did not hurl that day, and only one other time in the five years since did I almost honk all over the deck while performing field trials with Ryan. What's even better is that what I "got into" turned out to be one of the most intellectually satisfying journeys in my life. I would like to thank Ryan for his guidance through the truly exciting world of mobile robotics and robotic perception.

Thank you to Matt, the cognate member of this committee, who even during his job talk was so helpful and open in sharing his expertise in underwater mapping. Many of the cool-looking figures I made for this thesis are a direct result of his collaboration. Thanks, Matt.

Thank you to the other PeRL students (in order that I met them): Jeff "No-honk-guarantee" Walls [1], Ayoung, Gaurav, Nick, Steve, Ryan, Schuyler, Vittorio, Jie, GT, Enric, Alex, Arash, Steven, Josh, Vozar, and Derrick. I am sure that years from now I will be hard-pressed to think of another time I was able to so easily collaborate with such a fine cohort of engineers and researchers.

Thank you to my EE:Systems friends who got me through the stress of my first two years: Pat, Rob, Eric, Nick, Madison, Matt, and Mitch. You taught me that people can be both frighteningly intelligent and, uh, cool?

To my family: hopefully this document will establish once and for all that I did not get my PhD in fixing your printers and that I did not spend the last five years learning the solution to all Internet problems: unplug the router, wait 10 seconds, and plug it back in. To David: hopefully together we can end the our bloodline's curse of computer illiteracy. I'd say we're on the right track.

To Carrie, my partner throughout this whole ordeal: you had to put up with a lot, and

your encouragement when times got hard was extraordinary. We've learned a lot together, so perhaps the next thing we can learn is how to not work so hard.

References

[1] G. Algar and W. Campbell. On the fraternal demerits of emesis in urban vehicles. *Wayne's World*, volume 1. Feb. 1992.

Funding

This work was funded by the Office of Naval Research under award N00014-12-1-0092 and the American Bureau of Shipping under award N016970-UM-RCMOP.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ACRONYMS

2D two-dimensional

3D three-dimensional

ASFM acoustic structure from motion

AUV autonomous underwater vehicle

BA bundle adjustment

BoW bag-of-words

CAD computer aided design

CCD charged-coupled device

CLT Chow-Liu tree

CNN convolutional neural network

DBSCAN density-based spatial clustering of applications with noise

DCS dynamic covariance scaling

DIDSON Dual frequency IDentification SONar

DOF degree-of-freedom

DR dead-reckoning

DTM digital terrain model

DVL Doppler velocity log

EKF extended Kalman filter

EIF extended information filter

EM expectation-maximization

FAB-MAP fast appearance-based matching

FOV field of view

GICP generalized iterative closest point

GLC generic linear constraints

GMM Gaussian max-mixtures

GP Gaussian process

GPS global positioning system

GPU graphical processing unit

HAUV hovering autonomous underwater vehicle

ICP iterative closest point

IMU inertial measurement unit

iSAM incremental smoothing and mapping

KD k-dimensional

KLD Kullback-Leibler Divergence

LBL long baseline

LED light-emitting diode

LM Levenburg-Marquardt

MAP maximum *a posteriori*

MCMC Markov Chain Monte Carlo

MLE maximum likelihood estimate

PCCS pose-constrained correspondence search

pdf probability density function

RGB-D red, gleen, blue, and depth

RANSAC random sample consensus

ROV remotely operated vehicle

RMS root mean square

SBA sparse bundle adjustment

SBL short baseline

SFM structure from motion

SIFT scale-invariant feature transform

SLAM simultaneous localization and mapping

SSD sum of squared differences

SURF speeded up robust features

SVD singular value decomposition

TSDF truncated signed distance function

UAV unmanned aerial vehicle

USBL ultra-short baseline

UT unscented transform

ABSTRACT

This thesis reports on the incorporation of surface information into a probabilistic simultaneous localization and mapping (SLAM) framework used on an autonomous underwater vehicle (AUV) designed for underwater inspection. AUVs operating in cluttered underwater environments, such as ship hulls or dams, are commonly equipped with Doppler-based sensors, which—in addition to navigation—provide a sparse representation of the environment in the form of a three-dimensional (3D) point cloud. The goal of this thesis is to develop perceptual algorithms that take full advantage of these sparse observations for correcting navigational drift and building a model of the environment. In particular, we focus on three objectives. First, we introduce a novel representation of this 3D point cloud as collections of planar features arranged in a factor graph. This factor graph representation probabalistically infers the spatial arrangement of each planar segment and can effectively model smooth surfaces (such as a ship hull). Second, we show how this technique can produce 3D models that serve as input to our pipeline that produces the first-ever 3D photomosaics using a two-dimensional (2D) imaging sonar. Finally, we propose a model-assisted bundle adjustment (BA) framework that allows for robust registration between surfaces observed from a Doppler sensor and visual features detected from optical images. Throughout this thesis, we show methods that produce 3D photomosaics using a combination of triangular meshes (derived from our SLAM framework or given *a-priori*), optical images, and sonar images. Overall, the contributions of this thesis greatly increase the accuracy, reliability, and utility of in-water ship hull inspection with AUVs despite the challenges they face in underwater environments.

We provide results using the hovering autonomous underwater vehicle (HAUV) for autonomous ship hull inspection, which serves as the primary testbed for the algorithms presented in this thesis. The sensor payload of the HAUV consists primarily of: a Doppler velocity log (DVL) for underwater navigation and ranging, monocular and stereo cameras, and—for some applications—an imaging sonar.

CHAPTER 1

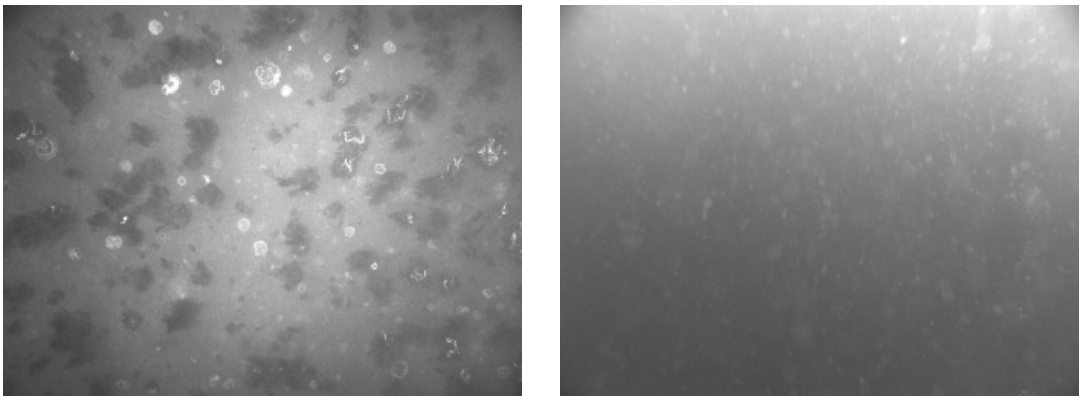
Introduction

1.1 The Importance of SLAM in Autonomous Robotics

Autonomous mobile robots are becoming a promising aid in a wide variety of tasks, such as surveying inaccessible environments, navigating complex roadways, disposing hazardous materials, and inspecting large man-made structures [155]. Though non-autonomous remotely operated vehicles (ROVs) are becoming commonplace in these sorts of situations, piloting them still requires significant expertise and training resources. A central goal of mobile robotics research is therefore to deploy robots that operate fully autonomously.

To this end, truly autonomous robots must answer the fundamental questions: "Where am I?", "Where am I going?", and "How do I get there?" (Leonard and Durrant-Whyte [107]). A popular technique, known as simultaneous localization and mapping (SLAM), answers the first question by bounding the uncertainty of the robot's current and past locations. In addition to its importance in autonomy, SLAM also provides a structural information about the robots' environment. Indeed, SLAM has been a crucial component in a range of autonomous systems, from small-sized flying unmanned aerial vehicles (UAVs) [38], to advanced self-driving cars [179], to autonomous underwater vehicles (AUVs) operating in deep-sea environments [50]. Its importance to modern mobile robotics is undeniable.

Despite the success of SLAM, certain applications remain challenging for robots to robustly perform these tasks. This document will propose various techniques aimed at improving the robustness, accuracy, and utility of SLAM in these sorts of challenging environments. In particular, we report on algorithms that can incorporate surface information in a real-time SLAM context (Chapter 2), an offline sonar-only mosaicing pipeline (Chapter 3), and a surface-assisted bundle adjustment (BA) framework (Chapter 4).



(a) Underwater image with slight backscatter (b) Underwater image with heavy backscatter

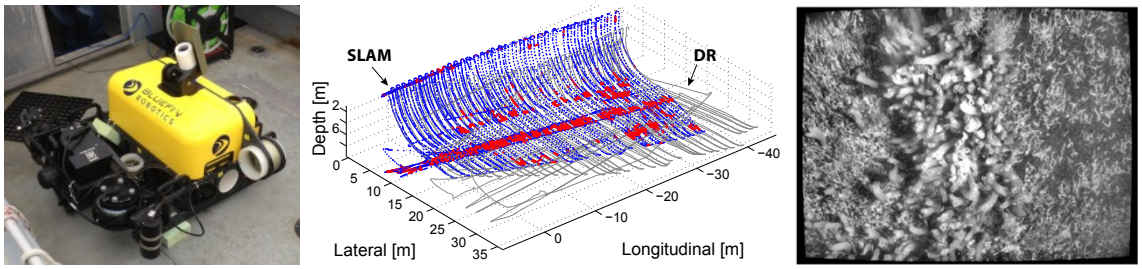
Figure 1.1: Examples of underwater imagery collected by an AUV. Both images are taken from the surface of the same ship hull with a standoff of $1.5\ \mathrm{m}$. In (a), biofoul is clearly visible throughout the image. On the other hand, (b) shows an example image that only captures the water column in between the camera and the ship hull. For high backscatter, visual SLAM becomes difficult, if not impossible, to successfully register overlapping images.

1.1.1 Challenges Posed in Confined Underwater Environments

Underwater robots face several interesting and unique challenges compared to more common terrestrial and indoor robots. There is no global positioning system (GPS) underwater to assist in world-frame localization. In addition, underwater visibility can be extremely limited due to a phenomenon known as *backscatter*, where light waves from an illumination source reflect off of particulates in the water, blocking the visibility of an intended subject [44, 80]. An example of backscatter in underwater imagery is shown in Fig. 1.1. Despite these challenges, AUVs can still perform SLAM either using light-based sensors (cameras) or acoustic-based sensors (sonar).

The difficulties associated with underwater SLAM are compounded for AUVs mapping confined and cluttered environments, such as ship hulls [120]. In particular, water in a high-traffic harbor is typically much more turbid, and consequently more prone to backscatter than open seawater making vision-based navigation extremely difficult. Though a two-dimensional (2D) imaging sonar is impervious to murky water, extracting three-dimensional (3D) information from imaging sonar is challenging due to the low texture detail and geometrical ambiguities of the sensor. Finally, large vessels' hulls induce significant magnetic and acoustic interference for compasses and acoustic-based transponders, making the use of these devices practically impossible.

Besides the navigational challenges that AUVs face while surveying the underwater



(a) HAUV (b) Figure courtesy A. Kim [96]. (c) Example of biofouling

Figure 1.2: Overview of the Bluefin Robotics HAUV for hull inspection. The HAUV provides the experimental evaluation for the methods described in this thesis. An above-water photograph of the vehicle placed on the deck of a small boat is shown in (a), along with its visual SLAM capabilities in (b). An example of a structural defect (in this case, excessive biofouling) is provided in (c).

portions of these vessels, they also should complete their inspection and assessment as quickly as possible. Scheduling constraints may prevent a single AUV from completing a full survey in one session. Therefore it is beneficial to either parallelize inspection tasks with multiple robots, or combine multiple partial surveys taken from a single robot into a common frame of reference. This is challenging because (*i*) the ship may move berth, rendering GPS (available at the surface) useless; (*ii*) establishing the data association between different partial surveys is extremely difficult for underwater environments; and (*iii*) many common SLAM algorithms have computational complexities that grow unbounded in time.

1.1.2 Autonomous Hull Inspection with the HAUV

The challenges described in the previous section are especially prevalent for the hovering autonomous underwater vehicle (HAUV) platform [169] for autonomous in-water ship hull inspection, shown in Fig. 1.2. Since 2007, the University of Michigan, Massachusetts Institute of Technology, and Bluefin Robotics have collaborated extensively on this project.[1] The robot is designed to replace human divers for the dangerous, expensive, and time-consuming task of mapping and inspecting below-water portions of ship hulls and marine structures. This inspection is intended to diagnose any structural wear or corrosion, or to identify foreign objects.

The HAUV is a free-floating vehicle that requires no existing long baseline (LBL) infrastructure for localization [71]. Instead, the HAUV uses a Doppler velocity log (DVL) as its primary navigation sensor, which can operate in a hull-relative or seafloor-relative

[1] Collaboration between MIT and Bluefin started in the early 2000's; the University of Michigan joined afterwards in 2007.

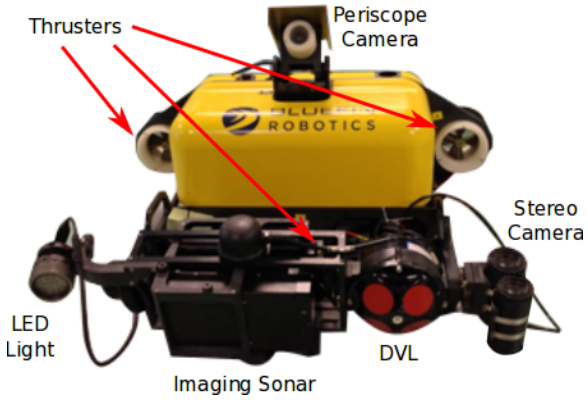


Figure 1.3: Overview of the HAUV sensor configuration.

mode. This allows the robot to inspect a variety of shallow-water infrastructure: ship hulls, pilings, or the seafloor of harbors. For this thesis, however, we are primarily concerned with its use for ship hull inspection.

1.1.2.1 Sensor Payload

An illustration of the HAUV's sensor configuration is shown in Fig. 1.3. The light-emitting diode (LED) light, underwater stereo camera, Dual frequency IDentification SONar (DIDSON) imaging sonar, and DVL are mounted on a sensor tray at the front of the vehicle, while the periscope camera is placed on top. In this configuration, the robot can easily capture either below-water (stereo or monocular) or above-water (monocular periscope) images.

During a mission, the sensor tray is servoed such that both the DVL and camera point nadir to the hull while the robot keeps a fixed distance. In late 2013, we upgraded the underwater monocular camera to a stereo configuration. The periscope camera, on the other hand, is fixed with a static angle allowing the robot to reliably image superstructure and above-water features.

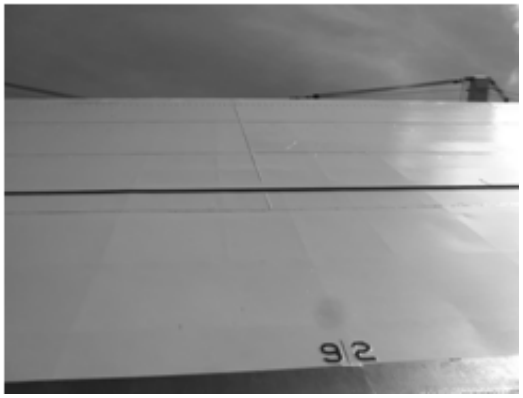
The specifics of the sensor suite are tabulated in Table 1.1 and reported by Hover et al. [72]. The inertial measurement unit (IMU), DVL velocities, and depth sensors are used for dead-reckoning (DR), and the cameras, sonar, and DVL ranges are used for perceptual information. In Fig. 1.4, we provide typical examples of both optical and sonar imagery.

1.1.2.2 Prerequisites of fully-automated hull inspection

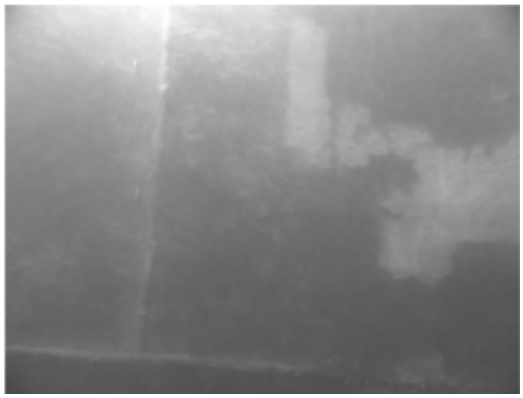
The end goal for the HAUV platform is fully automated ship hull inspection in which a robot can autonomously perform an in-water survey of ship hull with minimal human interaction. Towards this end, there are a variety of problems to be addressed. For example, the HAUV

Table 1.1: Payload characteristics of the HAUV

Prosilica GC1380	12-bit digital stills, fixed-focus, monochrome, 1 Megapixel
Periscope Camera	Monocular Prosilica GC1380 in water-proof housing
Underwater Camera (monocular, pre-2013)	Monocular Prosilica GC1380 in water-proof housing
Underwater Camera (stereo, post-2013)	Two Prosilica GC1380s in separate water-proof bottles, linked via Fast Ethernet
Lighting	520 nm (green) LED
IMU	Honeywell HG1700
Depth	Keller pressure, 1-σ noise at 10 cm
Imaging Sonar	Sound Metrics 1.8 MHz DIDSON
DVL	RDI 1200 kHz Workhorse; also provides four range beams
Thrusting	Five rotor-wound thrusters
Battery	1.5 kWh lithium-ion
Dry Weight	79 kg
Dimensions	1 m $\times$ 1 m $\times$ 0.45 m



(a) Periscope camera



(b) Underwater camera

(c) Imaging sonar

Figure 1.4: Sample periscope, underwater, and sonar imagery.

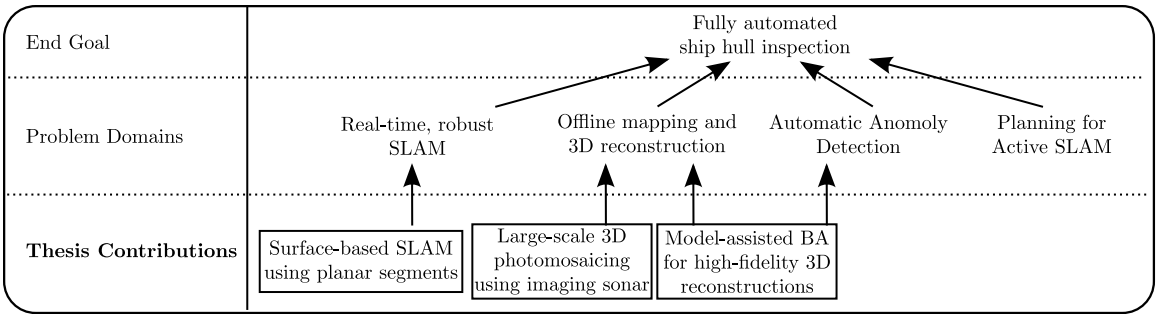


Figure 1.5: Overview of our contributions in the realm of fully automated hull inspection. The specific contributions of this thesis are shown in the bottom row, with connections to the corresponding prerequisite problem domains in the middle row.

may not be able to achieve whole-ship coverage in a single survey. We therefore envision a capability such that the robot can map individual portions of the hull and combine them over time (without human intervention) until the entire ship is covered. The ability for large-scale, repeatable, multi-session, and real-time SLAM performance is thus a prerequisite for truly autonomous hull inspection.

In addition, we envision a 3D reconstruction system such that the HAUV can produce high-fidelity models of the ship hull being surveyed. This reconstruction can serve as an indication of regions of the ship that contain small-scale foreign matter such as biofouling (which compromises fuel efficiency) or limpet explosives (which put human divers at risk). A truly autonomous robot will be able to identify these small-scale structural abnormalities, and offer an accurate and spatially intuitive 3D representation. Furthermore, fusing this 3D information into a computer aided design (CAD) model (if available) will offer an advantageous representation of the environment: one that combines textureless large-scale structure with small-scale details derived from both camera and sonar imaging modalities.

For this thesis, we provide several contributions that enable capabilities such as those described above. In particular, we achieve robust and accurate multi-session SLAM performance in real-time using a piecewise-planar representation of the ship hull (Chapter 2), a novel system for visualizing imaging sonar data in large-scale 3D (Chapter 3), and a framework for building high-fidelity reconstructions that combine a coarse 3D CAD model with camera-derived 3D structure (Chapter 4). These contributions are illustrated in Fig. 1.5, where we show that the end-goal of fully automated ship hull inspection involves a diverse set of problem domains. The specific contributions of this thesis are shown in the bottom row, with arrows encoding the problem domain(s) that pertain to each contribution. Throughout this chapter, we describe related research in the relevant problem domains from this figure.

(a) *USS Saratoga*

	Dim [m]
Length	324
Beam	39.6
Draft	11.3

(b) General characteristics

(c) *SS Curtiss*

	Dim [m]
Length	183
Beam	27
Draft	9.1

(d) General characteristics

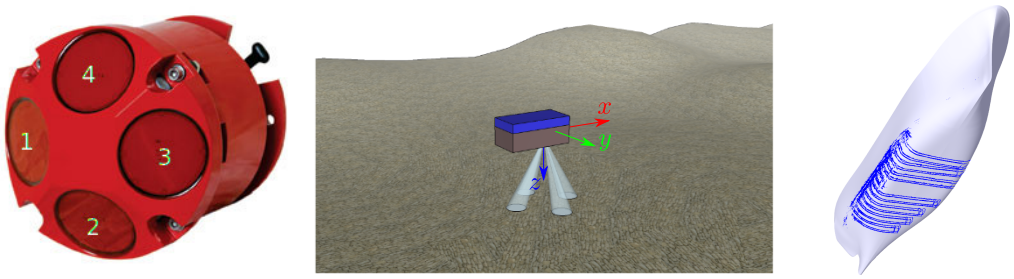
Figure 1.6: Size overview of the *USS Saratoga* (a) and *SS Curtiss* (c). General size characteristics of the vessels are given in the respective tables.

1.1.2.3 Surveyed Vessels

The contributions of this thesis are evaluated using data from two vessels: the *USS Saratoga* and *SS Curtiss*, shown in Fig. 1.6. The *USS Saratoga* is a *Forrestal*-class supercarrier, and was decommissioned in 1994. It is currently dismantled, however during our 2013 field trials it was stored at Newport Naval Station. The *SS Curtiss* is a *Wright*-class Aviation Logistics Support Container Ship that is currently in active service. Field trials for the *SS Curtiss* were performed at the San Diego Naval Base.

1.2 A Review of Doppler Sonar in Underwater Robotics

In this section, we describe the use of a Doppler sonar in underwater robotics. In particular, we describe its use as a *proprioceptive* sensor (i.e., odometry), and compare its advantages and disadvantages to underwater beacons. Finally, we cite some examples of DVL use for *exteroceptive* information (i.e., environmental observations) through the use of a sparse range observations.



(a) Workhorse Navigator DVL (b) Seafloor-relative navigation (c) Hull-relative navigation

Figure 1.7: An overview of the DVL in a Janus configuration [19] is shown in (a). It is commonly used to provide xyz position for an AUV by integrating sensor-frame velocities. Each of the four sonar beam measures the velocity of the vehicle with respect to the seafloor, as projected onto the beam's axis. In (b), these velocities are linearly transformed into inertial-frame velocities (using the orientation sensors), which are then integrated to provide positional measurements (figure courtesy G. Troni [168]). In (c), we show an example of hull-relative navigation with a DVL. Blue points represent the accumulation of range returns.

1.2.1 Proprioceptive Sensing

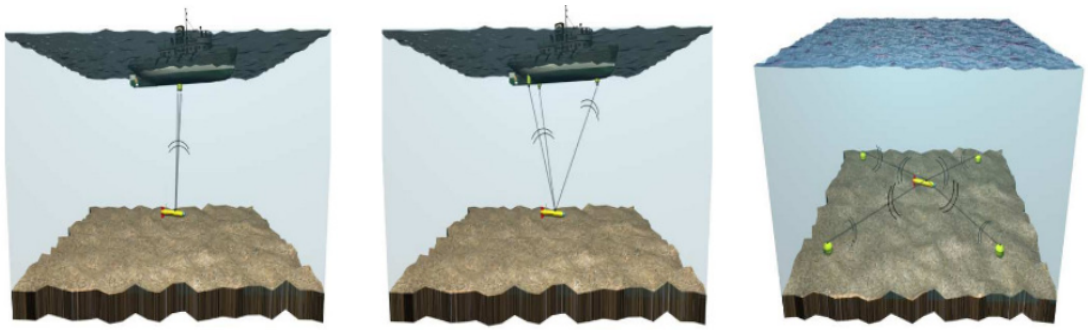
The DVL is a special application of a Doppler sonar that provides state-of-the-art underwater dead-reckoning position estimates. The Janus configuration is particularly popular for AUVs, in which four transducers are pointed 30 degrees around the sensor's vertical axis [19]. This configuration is illustrated in Fig. 1.7. The sensor measures a vector of the four beam-component velocities over time:

$$\mathbf{v}_b(t) = \begin{bmatrix} v_1(t) & v_2(t) & v_3(t) & v_4(t) \end{bmatrix}^\top.$$

Because the four transducers' angles to the vertical axis are known, the sensor-frame velocities can be linearly transformed with a matrix that converts the beam velocities to sensor-frame velocities in xyz and an error signal:

$$\begin{bmatrix} v_{s_x}(t) \\ v_{s_y}(t) \\ v_{s_z}(t) \\ e(t) \end{bmatrix} = \mathrm{M}\mathbf{v}_b(t), \qquad \mathrm{M} = \begin{bmatrix} a & -a & 0 & 0 \\ 0 & 0 & -a & a \\ b & b & b & b \\ d & d & -d & -d \end{bmatrix},$$

where $a = 1/\left(2\sin\left(30^\circ\right)\right) = 1.000$, $b = 1/\left(\cos\left(30^\circ\right)\right) = 0.2887$, and $d = \sqrt{2}/2 = 0.7071$. The resulting sensor-frame velocities must then be rotated into a inertial frame of reference,



(a) USBL (b) SBL (c) LBL

Figure 1.8: Overview of beacon-based navigation. The DVL is a common alternative to acoustic beacons placed on ships (USBL and SBL), or on the seafloor (LBL). Though a DVL is much more portable, it does not provide an *absolute* position reference, so dead-reckoned uncertainty will grow unbounded in time. Figure courtesy Alcocer, Oliveira, and Pascoal [5].

then integrated to compute a DR xyz position of the sensor (provided the vehicle attitude is accurately instrumented [142, 143, 176]). Therefore, IMUs on DVL-equipped robots have relatively low drift rates.

The DVL is most commonly used in bottom-referenced (downward-looking) or water column-referenced (upward-looking) mode. For the HAUV introduced in Section 1.1.2, however, the DVL is servoed to point orthogonal to the ship hull. Doing so allows the robot to navigate in an entirely hull-relative reference frame, thus eliminating the need for a GPS altogether.

1.2.2 Comparison to Acoustic Beacons

Compared to the acoustic beacon layouts in Fig. 1.8, the DVL has the advantages of requiring no pre-existing infrastructure nor initial calibration. However, integrating body-frame velocities over time results in position errors that grow unbounded, whereas a vehicle navigating via acoustic beacons will *always* have bounded position uncertainties [5, 166].

To bound this navigation drift, it is common to use perceptual information (such as the HAUV camera or sonar sensors introduced in Section 1.1.2) to constrain the vehicle's trajectory estimate.

1.2.3 Perceptual Sensing

In addition to measuring the velocity of an AUV, the raw DVL sensor data contains the range of each of the four beams. These ranges $r_1, \ldots r_4$ can easily be converted into 3D

points $\mathbf{p}_1, \ldots \mathbf{p}_4$ in the sensor-frame as follows:

$$\begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} = \begin{bmatrix} -r_1 \sin(30^\circ) & r_2 \sin(30^\circ) & 0 & 0 \\ 0 & 0 & r_3 \sin(30^\circ) & -r_4 \sin(30^\circ) \\ -r_1 \cos(30^\circ) & -r_2 \cos(30^\circ) & -r_3 \cos(30^\circ) & -r_4 \cos(30^\circ) \end{bmatrix}. \tag{1.1}$$

These 3D points provide sparse perceptual information that a few researchers have leveraged in prior work, with a particular focus on terrain-aided localization and bathymetric SLAM [81, 111]. Underwater terrain-aided techniques are typically performed with a multibeam sonar, which is much more dense than a DVL. Despite this trend, Eustice, Camilli, and Singh [48] and Meduna, Rock, and McEwen [111] proposed methods for an AUV equipped with a DVL to localize with respect to a prior bathymetric map derived from a large surface vessel equipped with a multibeam sonar with high spatial resolution. In addition, Bichucher et al. [12] explored the use of submap alignment consisting of sparse range returns from a DVL. They used generalized iterative closest point (GICP) to derive a spatial constraint between two submaps, which is then added as a loop-closing constraint in a bathymetric SLAM framework.

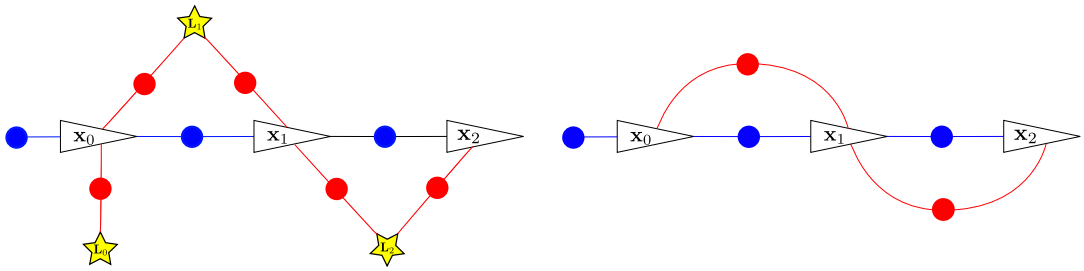
1.3 A Review of Sparse Methods in SLAM

In the simultaneous localization and mapping (SLAM) problem, a robot moves through an unknown environment, where the robot corrects for errors in its proprioceptive sensors using one or more of its perceptual sensors. This document is concerned with *metric* SLAM, where the robot strives to compute geometric relationships between elements in the map, as opposed to *topological* SLAM, where the map consists of abstract *places* without defining a metric. Additionally, there could be hybrid approaches that combine metric and topological representations [116].

The method used to solve SLAM depends on the formulation of the problem. For the *full SLAM* problem, this is formulated as the maximum *a posteriori* (MAP) estimate of all robot poses and landmarks, given all odometry and feature observations:

$$\widehat{\mathbf{X}}, \widehat{\mathbf{L}} = \underset{\mathbf{X},\mathbf{L}}{\operatorname{argmax}}\, p(\mathbf{X}, \mathbf{L} | \mathbf{U}, \mathbf{Z}),$$

where $\mathbf{X}$ represents the vehicle poses and $\mathbf{L}$ are landmarks. Here, $\mathbf{Z}$ are perceptual measurements of features, and $\mathbf{U}$ are proprioceptive odometry measurements.



(a) Full SLAM (b) Pose-graph SLAM

Figure 1.9: Example of SLAM as a factor graph. In the case of SLAM, unknowns (called "variable nodes") are represented using vehicle poses, $\mathbf{x}_i$, and landmarks $\mathbf{l}_j$. Variable nodes are constrained by factor nodes; blue factors correspond to proprioceptive sensors, while red factors correspond to perceptual sensors.

The structure of the full SLAM problem is inherently sparse; this differentiates it from the classic extended Kalman filter (EKF)-based SLAM [158] (which is inherently dense). Several approaches, like the extended information filter (EIF) for example [51], leverage this sparsity by not marginalizing past poses; however for this thesis we use a factor graph representation, discussed below.

Immediately marginalizing landmarks for pairwise egomotion estimates gives rise to what is sometimes referred to as *pose-graph SLAM*, or PoseSLAM. Here, the set $\mathbf{Z}$ contains relative-motion measurements *derived* from features in the environment, rather than explicitly optimizing landmark locations:

$$\widehat{\mathbf{X}} = \underset{\mathbf{X}}{\operatorname{argmax}}\, p(\mathbf{X}|\mathbf{U}, \mathbf{Z}).$$

The metric map in this case is simply the robot trajectory itself. This formulation can be particularly useful when a robot's sensors cannot uniquely observe point-feature landmarks in the environment, as is the case with laser scanners or sonar.

The SLAM problem is often represented as a graphical model [39, 45, 51]. Dellaert and Kaess [39] introduced a factor graph that represents robot poses and landmarks as variable nodes (unknowns), and odometry and landmark measurements as factor nodes together arranged in a bipartite graph. A simple example of this model for the full and pose-graph versions of SLAM is shown in Fig. 1.9. An obvious advantage of factor graphs is that they implicitly encode which variable nodes support the observation model of each odometry or feature measurement. Assuming additive Gaussian noise models, and treating $\mathbf{X}, \mathbf{L}, \mathbf{U}$ and $\mathbf{Z}$ as stacked vectors, the full SLAM problem simplifies to a nonlinear least-squares

problem:

$$\widehat{\mathbf{X}}, \widehat{\mathbf{L}} = \underset{\mathbf{X},\mathbf{L}}{\operatorname{argmax}}\, p(\mathbf{X}, \mathbf{L}|\mathbf{U}, \mathbf{Z}) \tag{1.2}$$

$$= \underset{\mathbf{X},\mathbf{L}}{\operatorname{argmin}}\, \left\| h(\mathbf{X}, \mathbf{L}) - \left[\mathbf{U}^\top, \mathbf{Z}^\top\right]^\top \right\|_\Sigma^2, \tag{1.3}$$

where $h(\mathbf{X}, \mathbf{L})$ is an observation model that computes all predicted feature and odometry measurements given the state, and Σ is the covariance matrix of all additive noise terms in the observations. We can simplify the above expression further if we note that the Jacobian of h with respect to $\mathbf{X}$ and $\mathbf{L}$ is sparse because each measurement depends only on a small subset of poses and landmarks. If all measurements are independent (i.e., Σ is a block-diagonal matrix, as is commonly assumed the case), we can express the optimization problem as minimizing a sum of squared differences (SSD) of all measurement errors:

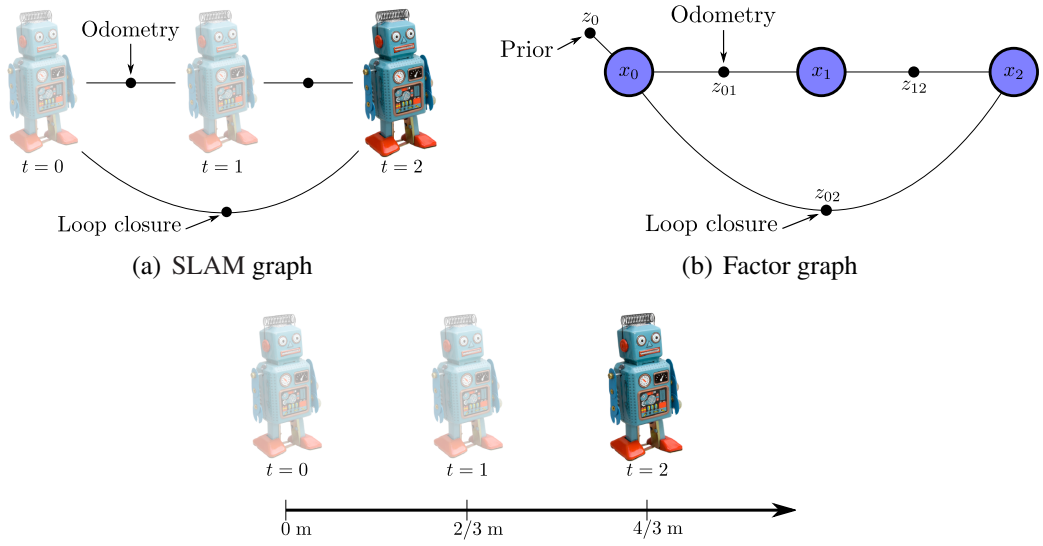
$$\widehat{\mathbf{X}}, \widehat{\mathbf{L}} = \underset{\mathbf{X},\mathbf{L}}{\operatorname{argmin}} \sum_i \|\mathbf{e}_i(\mathbf{S}_i)\|_{\Sigma_i}^2,$$

where $\mathbf{S}_i$ is the set of variable nodes that support (i.e., are connected to) the i^{th} factor node in the factor graph, $\mathbf{e}_i$ is the error between the observation and observation model evaluated at $\mathbf{S}_i$, and Σ_i is the measurement covariance of the i^{th} factor node. In this case, $\|\mathbf{v}\|_\Sigma^2 = \mathbf{v}^\top \Sigma^{-1} \mathbf{v}$ is the squared Mahalanobis distance. Intuitively, each factor potential is the *exponential* of the weighted error between the predicted and observed measurement.

Solving this optimization problem quickly, efficiently, and robustly is a major topic in optimization-based SLAM research [1, 63, 88, 90, 130, 132]. Popular methods to solve this formulation of SLAM fall under two loose categories: (*i*) direct factorization of the information matrix, and (*ii*) gradient descent techniques. Direct factorization techniques solve a linear system derived from the normal equations, which has the form

$$\mathrm{J}_h^\top \Sigma^{-1} \mathrm{J}_h \Delta\mathbf{x} = \mathrm{J}_h^\top \Sigma^{-1} \mathbf{r}, \tag{1.4}$$

where $\mathrm{J}_h^\top \Sigma^{-1} \mathrm{J}_h$ is the sparse information matrix, J_h is the Jacobian with respect to h, and $\mathbf{r}$ is a vector of measurement residuals. For the Gauss-Newton algorithm, this matrix can be decomposed using matrix factorization techniques such as Cholesky decomposition, thereby using forward and back substitution to efficiently solve the linear system. Incremental techniques avoid decomposing the information matrix and instead directly factor the weighted measurement Jacobian using QR factorization. The resulting R factor is the Cholesky factor of the information matrix, and can be incrementally updated by applying Givens rotations as the robot collects new measurements [88].



(a) SLAM graph

(b) Factor graph

(c) Final least-squares estimate

Figure 1.10: Toy example of a factor graph. The SLAM graph in (a) can be easily converted into a factor graph in (b). The final least-squares estimate is shown in (c) using the numerical values from Section 1.3.1.

Iterative linear solvers, on the other hand, do not perform sparse matrix factorization of any kind and instead perform variations of gradient descent on (1.3). Though slow for small problems, these methods are far more parallelizable and memory-efficient, making them suitable for optimizing very large SLAM graphs, or offline structure from motion (SFM) [2, 41].

1.3.1 A Hands-on Introduction to Factor Graph SLAM

In this section we provide an example of a factor graph being used to model the SLAM problem for a one-dimensional robot. Consider the illustration shown in Fig. 1.10(a). First, the robot observes an odometry measurement at time $t = 1$. An odometry measurement is an observation of the relative-motion between the robot pose at two discrete timesteps: $t = 0$ and $t = 1$. In this case, the robot moves forward by 1 m, and the variance of the zero-mean Gaussian noise that corrupts this measurement is $\sigma_{01}^2 = 1\ \text{m}^2$. Similarly, the robot observes an additional odometry measurement between $t = 1$ and $t = 2$ that describes the movement as forward motion by 1 m with noise variance $\sigma_{12}^2 = 1\ \text{m}^2$.

Finally, let us assume that the robot establishes a *loop-closure* between $t = 0$ and $t = 2$ (using, say, a laser range finder). By coincidence, suppose the robot establishes that relative-motion between $t = 0$ and $t = 2$ is again a forward motion by 1 m with a variance $\sigma_{02}^2 = 1\ \text{m}^2$.

The figure from Fig. 1.10(a) shows that this sequence of events is graphical in nature; it is therefore sometimes called the *SLAM graph*. By simply replacing the cartoon robot symbols with so-called *variable nodes*, and the directed edges with so-called *factor nodes*, we arrive at a *factor graph* representation of the SLAM problem shown in Fig. 1.10(b). A factor graph is a bipartite graph that acts as a factorization of the underlying probability density function (pdf) over the robot poses. In this representation, the common practice is to "pin" the first robot pose at the origin to eliminate *gauge freedoms* in the spatial relationship between nodes in the graph. Therefore, at time $t = 0$, the robot invokes a prior measurement, z_0, at the origin (0 m) having low variance ($\sigma_0^2 = 0.001$ m^2).

For this example, the factorization from the factor graph is as follows:

$$\begin{aligned} p(\mathbf{X}) = p(x_0, x_1, x_2) &\propto \prod_{i \in \{0,01,12,02\}} p(z_i|\mathbf{X}) \\ &\propto p(z_0|x_0)p(z_{01}|x_o, x_1)p(z_{12}|x_1, x_2)p(z_{02}|x_0, x_2), \end{aligned}$$

where each factor node corresponds to a factor in the above product.

Under the assumption of zero-mean Gaussian noise (as in this example), maximizing the above factorization over $\mathbf{X}$ simplifies to a nonlinear least-squares problem:

$$\begin{aligned} \underset{\mathbf{X}}{\operatorname{argmax}}\, p(\mathbf{X}) &= \underset{\mathbf{X}}{\operatorname{argmin}}\ -\log p(\mathbf{X}) \\ &= \underset{\mathbf{X}}{\operatorname{argmin}} \sum_{i \in \{0,01,12,02\}} \|h_i(\mathbf{X})\|^2 \end{aligned}$$

where h_i is an observation model corresponding to the factor indexed by i. In this case, the least-squares problem is minimizing the following SSD with respect to x_0, x_1, and x_2:

$$\|z_0 - x_0\|^2_{\sigma^2_{z_0}} + \|z_{01} - (x_1 - x_0)\|^2_{\sigma^2_{z_{01}}} + \|z_{12} - (x_2 - x_1)\|^2_{\sigma^2_{z_{12}}} + \|z_{02} - (x_2 - x_0)\|^2_{\sigma^2_{z_{02}}}.$$

By stacking the observation models into a column vector and assuming the initial guess is $[x_0, x_1, x_2]^\top = [0, 1, 2]^\top$, we have

$$h\left(\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_0 \\ x_1 - x_0 \\ x_2 - x_1 \\ x_2 - x_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \end{bmatrix}.$$

Next, we can set up the normal equations by applying (1.4). The corresponding linearized[2] least-squares problem is:

$$\begin{bmatrix} 1002 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \Delta\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & 1 \end{bmatrix}^\top \left(\begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \end{bmatrix} \right)$$

$$\begin{bmatrix} 1002 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \Delta\mathbf{X} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$\Delta\mathbf{X} = \begin{bmatrix} 0 \\ -1/3 \\ -2/3 \end{bmatrix}.$$

After solving for $\Delta\mathbf{X}$, we update our estimate of $\mathbf{X}$ to be $[0,\ 2/3,\ 4/3]^\top$ and repeat an additional iteration. It can easily be demonstrated that the solution to the normal equations in the second iteration yields $\Delta\mathbf{X} = [0,\ 0,\ 0]^\top$, and we therefore have converged to our final least-squares estimate of $\mathbf{X}$, which is illustrated in Fig. 1.10(c).

Factor graphs are a popular graphical model to represent optimization-based SLAM techniques; we use this representation throughout this thesis because of its tremendous success in the SLAM community. Though popular for the SLAM problem, factor graphs can also model a wide range of tasks in robotics. For example, Dhiman et al. [42] showed that 2D occupancy grid mapping can be accurately and efficiently modeled using a factor graph. They used modern MAP inference methods such as belief propagation and subgradient dual decomposition [92] and showed improved accuracy and performance to previous methods that use Markov Chain Monte Carlo (MCMC) algorithms such as Gibbs sampling [115].

1.3.2 Multi-session and Multi-robot SLAM

For some applications in mobile robotics, a robot may survey a particular area more than once, or multiple robots may be cooperatively exploring a common area. In these cases, it is often desirable to combine each session into a single SLAM graph. This is known as multi-session SLAM.

For this section, we simplify the notation by considering only robot poses as unknowns (i.e., the PoseSLAM problem discussed in Section 1.3). Furthermore, we assume a perfect

[2]The observant reader will note that $h(\cdot)$ is already a linear function of $\mathbf{X}$, so we expect this example to converge in exactly one iteration.

communication channel, where the robot can transmit its entire factor graph to another session (for single-agent multi-session SLAM) or robot (for multi-agent multi-session SLAM).

1.3.2.1 Global Methods

If the unknown vehicle poses in a multi-session SLAM problem are all parameterized with respect to a common world frame, then this is known as a "global" method for multi-session SLAM. A simple example is illustrated in Fig. 1.11.

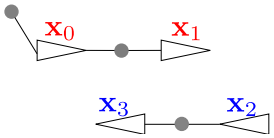
Though the formulation and factor graph topology of global multi-session SLAM are simple, there are several limitations to this method. The first is that before the factor graphs of each session are aligned, they are disjoint. This implies that one of the disjoint graphs is not fully constrained with respect to a common frame and therefore the graph has a low-rank information matrix. A low rank information matrix is problematic because the normal equations do not admit a unique solution, and similarly the covariance matrices will have a determinant of zero and are slow to compute.

Second, by the time the graphs are registered, the previously low-rank graph may have accumulated significant navigational drift, introducing the possibility of the optimizer converging to a local minimum. While these issues are less of a problem for offline processing, they can make this method unusable for real-time applications.

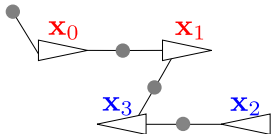
1.3.2.2 Relative Pose-graph Methods

Unlike global methods, it is far more common to parameterize each variable node in the factor graph with respect to the frame of reference of their respective SLAM sessions. Each session has an associated variable node in the factor graph, which provides a transformation from the global frame of reference to the session frame. These nodes are sometimes called "base nodes" [124] or "anchor nodes" [98]. A simple example is illustrated in Fig. 1.12.

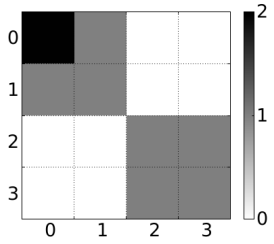
Ni, Steedly, and Dellaert [124] introduced these nodes in a factor graph representation of SLAM graphs, with the aim of increasing the convergence speed of single-robot graphs. Kim et al. [98] adopted the incremental factorization techniques from [88] for these anchor nodes to improve the performance of multi-robot mapping. The advantage of relative pose-graphs over the global parameterization is twofold. First, the graph tends to converge in fewer iterations when using anchor nodes. Second, the graph is always full-rank using this method because each session is full constrained and absent of any gauge freedoms.



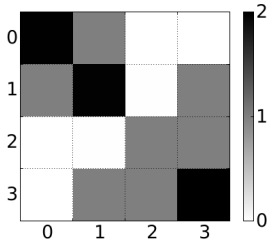
(a) Low-rank pose graph before alignment



(b) Full-rank pose graph after alignment

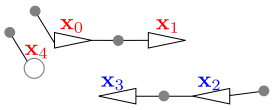


(c) Low-rank information matrix before alignment

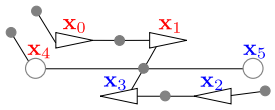


(d) Full-rank information matrix after alignment

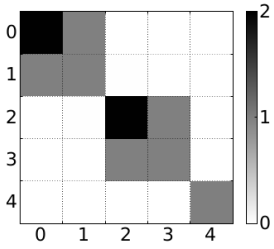
Figure 1.11: Global multi-session SLAM example. In (a) and (b) are the factor graph representations of the global parameterization of multi-session SLAM. Nodes colored in red belong to a different session than nodes colored in blue. Before the first alignment measurement, the information matrix, shown in (c), is low rank because the blue nodes are not fully constrained. After the encounter, shown in (d), the information matrix becomes full-rank and the covariances can be recovered from the backend.



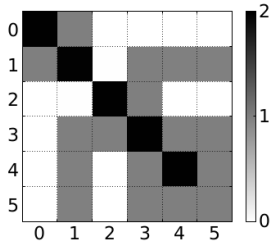
(a) Full-rank pose graph before encounter



(b) Full-rank pose graph after encounter



(c) Full-rank information matrix before encounter



(d) Full-rank information matrix after encounter

Figure 1.12: Relative pose-graph multi-session SLAM example. In (a) and (b) are the factor graph representations of multi-session SLAM using the anchor node paradigm from Kim et al. [98]. The node colors (red, blue) correspond to different SLAM sessions. Before and after an encounter measurement, the information matrices, in (c) and (d), are full-rank and therefore the covariances may be recovered at any time. Nodes x_4 and x_5 are the anchor nodes for the red and blue sessions, respectively.

1.3.2.3 Global Localization

Regardless of the use of global or relative graphical methods, global localization is a prerequisite step in cooperative and multi-sesison SLAM. Localization is a long-studied problem in robotics, and is especially important in GPS-denied environments. Dellaert et al. [40] popularized the Monte-Carlo localization techniques, where a particle filter converges to a likely distributions of particles as current measurements are compared against a prior map. Recently, these methods have attempted to improve particle filter performance by making data associations across multiple sensor modalities (Schwendner, Joyeux, and Kirchner [152]). Indelman et al. [78] solved data association using expectation-maximization (EM) to cluster many alignment hypotheses to an inlier set and outlier set. Their approach was evaluated indoors and is much more difficult to apply to a complex underwater environment. For the HAUV application, high-resolution 3D scanners are not available so we prefer methods that are generalizable to low-resolution (e.g., DVL) or low-cost (e.g., optical camera) sensors used in localization.

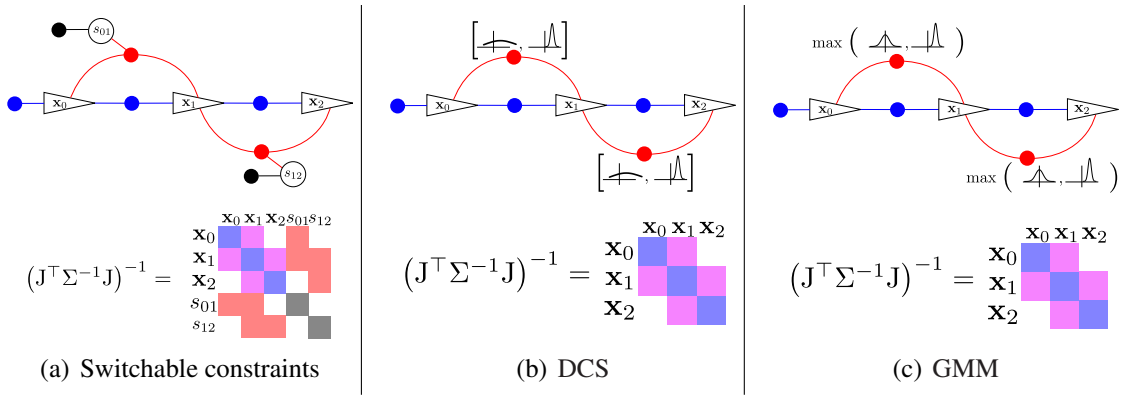
Visual, or appearance-based, localization is an especially active field in robotics. In particular, bag-of-words (BoW) and visual vocabulary tree-based approaches have shown promising computational advantages over geometrically matching visual features for every keyframe [126]. More recent approaches involve efficiently modeling the co-occurrence of words in the vocabulary and efficient data structures that allow for extremely fast localization and appearance-based mapping [35].

Instead of appearance-based feature or keyframe matching, other methods have fitted a rigid transformation between two robots' maps using a combination of Delaunay triangulation and random sample consensus (RANSAC) [128]. This approach has been successfully used in distributed and communication-constrained applications, where the robots do not have a perfect communication channel [36].

1.3.3 Robust Techniques for SLAM Backends

The conversion of a factor graph representation of SLAM to a least-squares problem involves the assumption of Gaussian errors, as discussed in Section 1.3. Least-squares problems are well-known to be susceptible to measurement outliers. A number of recent approaches have been proposed to make the backend solver more robust to these outliers. The standard approach in the SFM community is the use of M-estimators [69, 74] to robustify outlier terms in the SSD from (1.3), however these have been shown to work poorly in SLAM pose-graphs and SLAM graphs with relatively few feature nodes [1].

In this section, we provide an overview of several notably popular methods used in



(a) Switchable constraints (b) DCS (c) GMM

Figure 1.13: Illustration of three popular methods for robust graph-based SLAM. Switchable constraints is often the least performant of the three because the information matrix has an extra row and column for each of the switch variables s_{01} and s_{12}. The colors in the information matrix denote the connectivity and information induced by factors having the same color.

modern approaches to SLAM. We cover these methods for two reasons: (*i*) they are an important component in making long-term visual SLAM with the HAUV robust, and (*ii*) in Chapter 4, we will describe a surface-assisted BA framework as a special case of the Gaussian Max-Mixtures technique. An illustration of each method is provided in Fig. 1.13.

1.3.3.1 Switch Variables and Switchable Constraints

One of the early proposed methods in robust SLAM backends in the use of switch variables and switchable constraints [160]. In this approach, the state variable ($\mathbf{X}$ from Section 1.3) contains a set of *switch variables*, which are continuous variables that, when evaluated at a *switch function*, scale the covariance of the corresponding loop-closing factors to which they are connected (red nodes in Fig. 1.13(a)). This switch function can be any function of the switch variable that returns a value between 0 and 1 (typically chosen as a sigmoid function).

This approach has the undesirable properties of adding an additional row and column to the information matrix, in addition to not having a strong probabilistic interpretation.

1.3.3.2 Dynamic Covariance Scaling

Agarwal et al. [1] introduced dynamic covariance scaling (DCS) as a faster and simpler alternative to the switchable constraints method from above. Using DCS, the scale variable is determined *in closed form* of the measurement error (given the corresponding factor's variable nodes), rather than included as an unknown variable node in a factor graph (as

was the case with switchable constraints). Using this approach, the state variable (and hence, the information matrix) *does not grow* with the addition of robust loop-closing factors. Therefore, the complexity of the matrix factorization remains performant as in the non-robust case. Similar to switchable constraints, however, the use of DCS lacks a strong probabilistic interpretation.

1.3.3.3 Gaussian Max-Mixtures

Olson and Agarwal [131] introduced the Gaussian max-mixtures (GMM) method as a probabilistically motivated approach for robust SLAM backends. Using GMM, each loop-closing factor has a corresponding finite set of Gaussian components, each with different means and covariances. When evaluating potentials for these factors, the algorithm chooses the component that is most likely. The most likely component is then used to update the state estimate in a similar fashion to the EM class of algorithms.

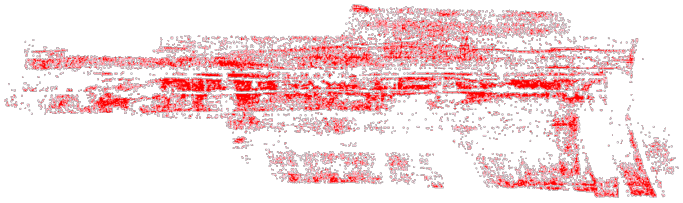
The advantage of this approach is that the underlying algorithm parameters (number of components, their weights, and their mean and covariances) have a more intuitive probabilistic interpretation than dynamically scaling the measurement noise covariance matrix. In addition, Olson and Agarwal [131] showed that this model can provide robust odometry (i.e., addressing the so-called *slip-or-grip* problem in ground robots), whereas switchable constraints and DCS require a well-behaved odometry backbone.

1.4 A Review of Surface Representations in SLAM

As formulated in Section 1.3, one of the ways to represent a map from SLAM is as a set of landmarks or features, which are any distinctive attributes of an environment. One particularly common technique is to represent this map as a sparse collection of point features. Points are easily detected as corners in an image, for instance as a point with large gradient in pixel intensity with respect to both x and y [67].

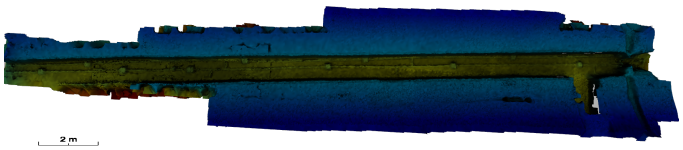
However, more recently there has been considerable efforts on constructing SLAM maps using surface primitives. In this section, we cover various ways of representing these surfaces and cite several applications in robotics. They fall into two loose categories: explicit surfaces, which are described in some functional form, and implicit surfaces (also called an *isosurface*), which are defined as a set of points that evaluate to a certain value (commonly, 0) when mapped by a scalar function. Some examples of SLAM maps containing both point-based and surface-based features are shown in Fig. 1.14.

(a) MHL Tank



(b) Sparse 3D points (monocular or stereo camera)

(c) Planar patches (DVL)



(d) Implicit surface (stereo camera)

Figure 1.14: Examples of different maps derived from SLAM for the same physical environment (the University of Michigan Marine Hydrodynamics Lab's water tank shown in (a)) using data from the HAUV. In (b), sparse features are plotted in their optimal 3D position using stereo camera SLAM. A map using planar patches fitted from a DVL point cloud (as will be discussed in Chapter 2) in (c). Finally, (d) shows the implicit surface from the map in (b), computed offline using dense stereo matching.

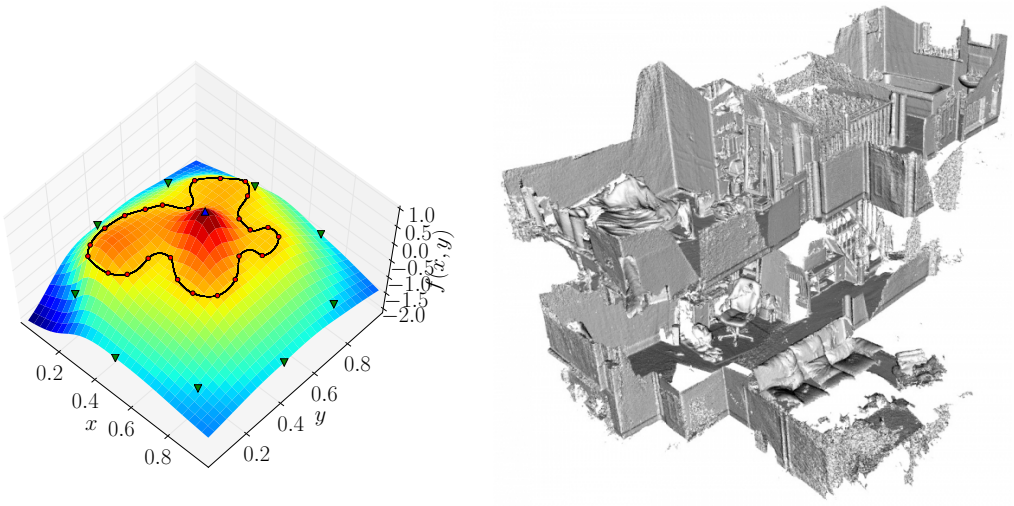
1.4.1 Explicit Surfaces

By far the most common explicit surface used in robotics is a plane (or line, if mapping in 2D). Planes are a natural choice for representing structured environments like office buildings and most other indoor environments. Smith, Reid, and Davison [157] used line primitives detected from a monocular camera to perform SLAM, taking advantage of line-like features that are commonly found in man-made structures. Weingarten and Siegwart [173] used pitch-actuated 2D LIDAR scanners to acquire high-resolution 3D points clouds of an office building. They then segmented these point clouds into a relatively few number of planes. This method introduced the advantage of having a much more compact representation of an indoor environment compared to storing millions of 3D points. More recent works such as Trevor, Rogers, and Christensen [164] and Taguchi et al. [162] followed this trend using handheld RGB-D sensors in real-time systems. To our best knowledge, Trevor, Rogers, and Christensen [164] were the first to use the factor graph model to perform 3D SLAM with planar primitives. In addition, Ruhnke et al. [149] used small planar features ("surflets") in a BA framework, where surflets observed from a dense range image are co-registered. Other underwater approaches, such as the work by Pathak et al. [140], do not generalize to the sparse 3D point clouds collected by a DVL because they require that planar surfaces be *directly observed* from multiple views. In particular, DVL-based survey tracklines in structured environments may often not overlap, because the field of view (FOV) of the imaging sensor may extend well beyond a single trackline.

Besides simple planes, explicit surfaces can take the form of a height map of two spatial variables, such as $z = f(x, y)$. Van Kaick et al. [170] used piecewise bivariate quadratic height maps to mesh a point cloud instead of the commonly-used linear interpolation. These height maps can also be empirically defined, which is useful for mapping unstructured terrain. Whitaker and Juarez-Valdes [175] provided methods to fuse multiple scans to a single model using known scan locations. Kweon and Kanade [104] provided early methods for terrain-based SLAM using laser scanners. The core idea of this approach has been used in several other applications, including the underwater domain using down-looking profiling sonar instead of laser. Aligning local height maps collected from profiling sonar is a common way to perform bathymetric SLAM using submaps [139, 178].

1.4.2 Implicit Surfaces

Explicit surface representations fail when the surface cannot be represented as a height map. For instance, a unit sphere has no explicit representation. The unit-sphere, described by the equation $x^2 + y^2 + z^2 = 1$, cannot be represented explicitly as a height map function of x



(a) Synthetic 2D implicit surface (black line) (b) 3D implicit surface from a depth camera

Figure 1.15: Examples of implicit surfaces. In (a), a 2D implicit surface (black line) is estimated from finite measurements (red dots) using the GP method from Williams and Fitzgibbon [177]. A 3D implicit surface of an indoor environment is shown in (b) (image credit Whelan et al. [174]).

and y. Formally, one can define a $(d-1)$-dimensional manifold $\mathcal{S}$ as the set of all points that evaluate to a certain value when passed to a scalar function $f : \mathbb{R}^d \mapsto \mathbb{R}$:

$$\mathcal{S}_0 = \left\{ \mathbf{x} \in \mathbb{R}^d \middle| f(\mathbf{x}) = 0 \right\}. \tag{1.5}$$

For the unit sphere example, the implicit surface representation is

$$\left\{ [x, y, z]^\top \in \mathbb{R}^3 \middle| x^2 + y^2 + z^2 - 1 = 0 \right\}.$$

Some examples of both 2D and 3D implicit surfaces are shown in Fig. 1.15.

In many applications, f is either the truncated or non-truncated signed distance function. Curless and Levoy [37] developed a method for the accumulation weighted signed distance functions from individual keyframes. The final merged surface "carves out" the initial emptiness of a 3D space, resulting in a seamless model. Johnson-Roberson et al. [84] used this method to integrate underwater stereo-derived 3D meshes from a calibrated stereo camera rig for large-scale texturing and visualization of underwater 3D photomosaics.

Williams and Fitzgibbon [177] use a Gaussian process (GP) to fit a implicit surface from few measurements using a covariance kernel equivalent to the thin plate spline regularizer introduced by Girosi, Jones, and Poggio [59]. Dragiev, Toussaint, and Gienger [43] used this implicit surface technique for a robot to grasp 3D objects using only a few measurements on

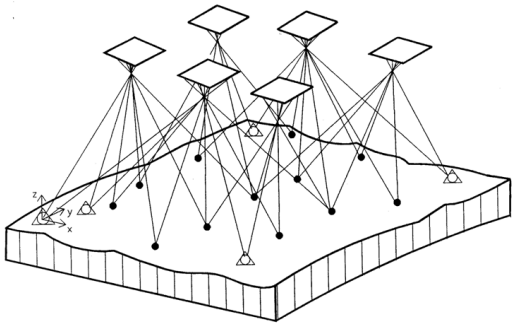


Figure 1.16: Illustration of BA for downward-looking cameras. Squares denote CCD sensors, with circles denoting visual features. The process of minimizing reprojection error is a special case of the SLAM problem discussed in Section 1.3. Image credit Matrix-Geo [110].

the object's surface. In a somewhat similar fashion, Barkby et al. [9] used GPs to perform SLAM on partially-observed bathymetric surface maps from a downward-looking profiling sonar.

The Microsoft Kinect sensor has been a recent focus in many efforts in terrestrial mobile robots. In their seminal work, Newcombe et al. [123] present a method for high-resolution SLAM using a red, gleen, blue, and depth (RGB-D) sensor like the Microsoft Kinect. The truncated signed distance function (TSDF) acts as the map of the environment, and as a means to align successive keyframes by projecting the model into a synthetic image. Though this original work requires a fixed size environment, Whelan et al. [174] extended this work to grow the surface as the RGB-D sensor moves outside the originally-allocated space.

1.5 A Review of Model-assisted Bundle Adjustment

Bundle adjustment (BA) is a special case of the SLAM problem in which light emanating from a 3D features strikes an optical sensor (such as a digital camera's charged-coupled device (CCD) sensor) as illustrated in Fig. 1.16 [165]. The corresponding observation model is often called *reprojection* or *reprojection error*. A fundamental problem with this classic technique is that the feature matching step is subject to outliers, and that the optimizing reprojection is subject to local minima. Effectively, this can result in inconsistencies in the final estimate of the camera poses and 3D structure. To cope with these issues, researchers in the 1990's and early 2000's developed methods that leverage the information in prior models in camera-based 3D reconstruction pipelines, particularly in the domain of human face reconstruction [33, 54, 56, 91, 154].

These methods have widely been overshadowed, however, by modern dense reconstruc-

(a) CAD model of the *SS Curtiss*

(b) CAD model of the *NS Savannah*

Figure 1.17: Examples of 3D CAD models of ship hulls. One of the research goals of this thesis work is to utilize these prior surfaces during the BA step. Doing so will combine the geometric consistency of the CAD model with the fine-level details of the vision-based reconstruction.

tion methods using either high-quality optical cameras or commodity RGB-D cameras discussed in Section 1.4.2 [57, 123, 174]. Recently, however, these model-assisted approaches have re-emerged in certain field robotics applications in challenging environments [58]. This thesis will explore the use of these techniques in autonomous ship hull inspection when a CAD model of the surveyed ship is available. Examples of these CAD models are provided in Fig. 1.17.

1.6 Thesis Outline

1.6.1 Thesis Goals

The problems we consider in this thesis are as follows:

- Given sparse range returns of a locally planar underwater structure (such as a ship hull), we explore how to leverage this information to improve navigational performance in a *real-time* setting. Solving this problem would therefore provide the autonomous robot with an improved state estimate.
- Produce improved 3D visualizations (photomosaics) of perceptual data in an *offline* setting using both perceptual data from both sonar and optical cameras.

To this end, we have achieved the following contributions:

(i) We propose a representation of the underwater portion of a ship hull as a collection of planar features. These features can be constrained in a factor graph representation

without direct overlap by assuming some basic prior information about the ship hull.

(ii) We developed a 3D photomosaicing pipeline using a 2D imaging sonar and DVL. The range returns from the DVL allow us to reconstruct an untextured 3D mesh of the hull, however our method can also localize to a prior CAD-derived mesh, if available. To our knowledge, we are the first to present 2D sonar imagery using a 3D model.

(iii) We propose an EM-derived model-assisted BA algorithm that allows for more tightly-constrained estimates of the surfaces observed from the DVL and visual features observed from the optical camera. We show this algorithm is a special case of the GMM framework [131].

1.6.2 Document Roadmap

Each contribution from above is described in detail in the followings chapters:

Chapter 2 Describes a piecewise-planar relationship of planar nodes in a SLAM graph, and offers significant evaluation in a real-time, multi-session context.

Chapter 3 Describes a post-processing 3D photomosaicing pipeline using a Doppler sonar and 2D imaging sonar. The piecewise-planar representation from Chapter 2 acts as a prerequisite for this technique.

Chapter 4 Describes an offline model-assisted BA framework, in which surfaces observed by a DVL are robustly co-registered to visual features from an optical camera.

Chapter 5 Offers a summary of our contributions and potential directions for future work.

Appendix A Offers an overview of the coordinate frame conventions used in this thesis.

Appendix B Details a minimal planar representation used in this thesis and motivates the advantages of this representation in the context of SLAM. In addition, we geometrically derive planar composition operators used in Chapter 2.

Appendix C Discusses a sensitivity analysis to the user-defined parameters used in the SLAM framework from Chapter 2.

Appendix D Offers an overview of the pinhole camera model, providing details for both monocular and stereo configurations.

Appendix E Provides important properties of the multivariate Gaussian distribution that ease the derivation of GP regression used in Chapter 3.

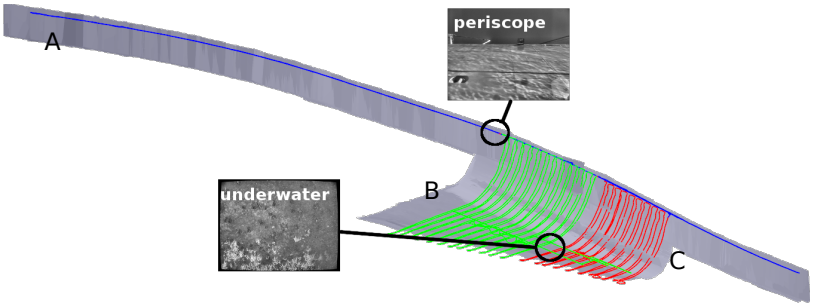
CHAPTER 2

Real-time SLAM with Planar Segments

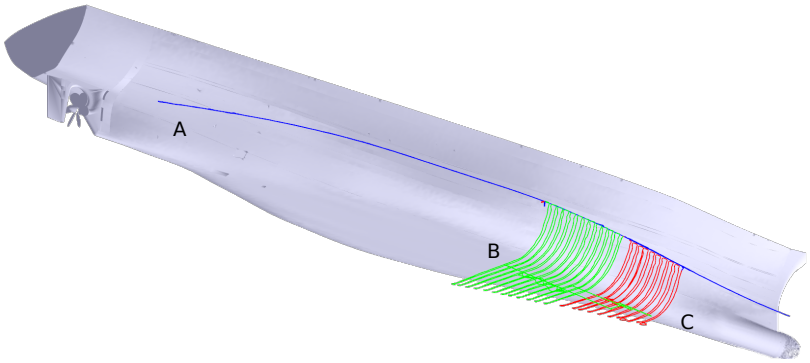
2.1 Introduction

Periodic maintenance, structural defect assessment, and harbor surveillance of large ships is of great importance in the maritime industry because structural wear and defects are a significant source of lost revenue [15]. Until recently, these tasks were carried out by periodic dry-docking, where a vessel is removed from the water for easy access to the entire ship hull. This task has recently evolved to include the deployment of a variety of unmanned underwater robots to map and inspect the hull *in situ*. Due to the periodic nature of the inspection, surveys are often conducted every few months. Having a single autonomous underwater vehicle (AUV) map an entire vessel within a short time frame may be impossible. However, using multi-session mapping techniques would parallelize the inspection process, or allow partial maps to be built as time allows. In addition, the sheer size of the final map would prevent a state-of-the-art simultaneous localization and mapping (SLAM) system from performing in real-time. Therefore, a computationally efficient and compact representation of the map is critical to maintain real-time map-merging performance in the long term. In this chapter, we propose a real-time SLAM pipeline that models the surface of a ship as a collection of many planar segments and show robust and reliable long-term multi-session SLAM performance.

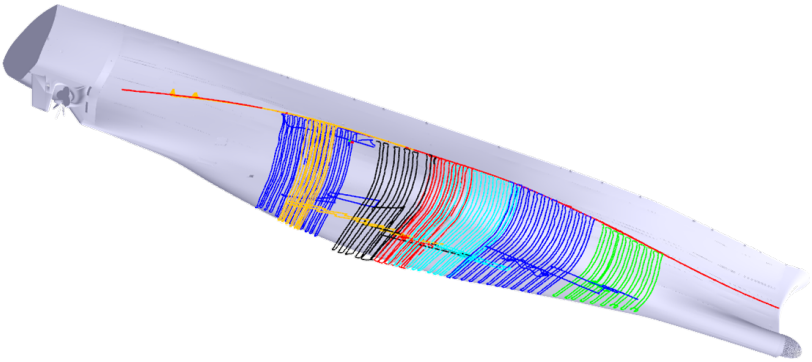
The goal of this chapter is to extend our hovering autonomous underwater vehicle (HAUV) graph-based visual SLAM system to support automated registration from multiple maps via a method called multi-session SLAM. To maintain real-time performance, redundant or unnecessary nodes are marginalized from the factor graph once the sessions are aligned into a common hull-relative reference frame. An overview of our multi-session SLAM framework is illustrated in Fig. 2.1, including a visualization of how the estimated trajectory would align with a computer aided design (CAD) hull model. This framework is consistent, computationally efficient, and leverages recent techniques in SLAM research to



(a) *SS Curtiss* multi-session SLAM result with three sessions (A,B,C)



(b) Three merged graphs overlaid on CAD model



(c) Additional graphs overlaid on CAD model

Figure 2.1: Multi-session SLAM overview. In (a), we show three different sessions aligned to a common frame using our multi-session SLAM framework, with each color denoting a different session. A surface mission ('A'), in blue, uses the periscope camera to capture images of the ship's above-water superstructure. Two underwater missions ('B' and 'C'), in red and green, capture underwater images of the ship hull. The gray patches denote the planar features used in our novel planar constraints framework, described in Section 2.3.4. Two representative images from the periscope and underwater camera are shown. In (b), we manually aligned a low fidelity CAD model of the ship for the sake of visual clarity, with the corresponding sessions colored appropriately. In (c), we show this technique can repeatedly used to cover more and more of the hull. Note that the CAD model is not used in our SLAM system.

create an extensive map. Our method has several advantages over long baseline (LBL) and global positioning system (GPS) navigation methods. In essence, long-term hull-inspection with an AUV demands a hull-relative method for navigation because a vessel's berth may change with time.

This chapter represents the culmination of several of our publications [135–137]. In the sections that follow, we discuss several contributions that arose from our work:

- The introduction of a *piecewise-planar* factor that can constrain planar features that do not directly overlap.
- A full-rank planar representation that allows for efficient covariance recovery and the use of the Gauss-Newton optimization algorithm.
- A raycasting constraint between a plane and individual Doppler velocity log (DVL) beams in the event that not all four beams are available.
- Periscope re-localization with the aid of pose-constrained correspondence search (PCCS).
- Exemplar views to maintain visual diversity in the merged graph.
- The use of generic linear constraints (GLC) in our planar feature-based factor graphs, and a quantitative evaluation of the effect of repeated applications of GLC.
- An extensive evaluation of field data for autonomous hull inspection; the surveys presented in this chapter span three years, with a total path length exceeding $10\ \mathrm{km}$.

2.1.1 Outline

We start with a brief overview of relevant literature in Section 2.2. In Section 2.3, we summarize our single session SLAM framework and introduce our planar factor constraint, which substantially improves our survey results. We describe our overall approach for long-term hull-relative SLAM using the HAUV in Section 2.4. Here, we present how our SLAM system for the HAUV can be adapted for the GLC framework, thus enabling long-term mapping capabilities. Finally, in Section 2.5, we experimentally evaluate our approach using real-world data, and conclude with a discussion in Section 2.6.

2.2 Related Work

We cover three broad areas of related work: (*i*) underwater surveillance using robots and underwater sensors; (*ii*) managing graph complexity for long-term SLAM; and (*iii*) merging

intermediate SLAM graphs into a larger map without a fixed absolute reference frame, for example, as provided by a GPS.

2.2.1 Robotic Underwater Surveillance

Early work by Harris and Slate [68] aimed to assess superficial defects caused by corrosion in large ship hulls using the *Lamp Ray* remotely operated vehicle (ROV). A non-contact underwater ultrasonic thickness gauge measured the plate thickness of the outer hull, and an acoustic beacon positioning system was used for navigation. Since the early 2000's, some focus has shifted from structural defect detection to the identification of foreign objects like limpet mines. Sonar imaging sensors, in particular, have been successfully deployed to detect foreign objects underwater [11]. In work by Trimble and Belcher [166], a similar imaging technique was successfully deployed using the free-floating *CetusII* AUV. Like the *Lamp Ray*, this system used a specifically-designed LBL acoustic beacon system for navigation around ship hulls and similar underwater structures. Additionally, Carvalho et al. [29] used a probability-of-detection metric to assess the effectiveness of these sorts of acoustic-based sensors. Regardless of the specific robot used, the significant amount of infrastructure required for beacon-based navigation [117] hinders the widespread use of these robotic systems in large shipping ports.

More recently, techniques in computer vision and image processing have been applied to various applications in underwater inspection. Negahdaripour and Firoozfam [121] used a stereo vision system to aid in the navigation of a ROV, and provide coarse disparity maps of underwater structure. Ridao et al. [145] used both a camera and sonar to inspect dams using an ROV, and employed bundle-adjustment techniques and image blending algorithms to create mosaics of wall-like structures of a hydroelectric dam. Some efforts have even attempted to identify physical defects only from a single image. Bonnín-Pascual and Ortiz [14] applied various image processing techniques to automatically identify the portions of a single image that appear to contain corrosion.

In addition to free-floating vehicles, there are some approaches that use robots that are physically attached to the ship hull itself. Menegaldo et al. [113, 114] use magnetic tracks to attach the robot to the hull, but their hardware is limited to above-water portions of the hull. By contrast, Ishizu et al. [79] developed a robot that uses underwater, magnetically adhesive spider-like arms to traverse the hull. Like previously mentioned free-floating vehicles, this robot employs a stereo camera to assess the amount of corrosion. Hull inspection with high frequency profiling sonar is presented by VanMiddlesworth et al. [171] where they build a 3D map and the entire pose-graph is optimized using submaps.

Although the main focus of this chapter is on in-water hull inspection, there are other applications for long-term AUV-based surveillance and mapping in large area underwater environments. Autonomous vehicles present significant potential for efficient, accurate and quantitative mapping in Arctic exploration [103], Mariana Trench exploration [18], pipeline inspection [65] and coral reef habitat characterization [156]. These studies were presented within the context of a single-session SLAM application, however, the repeated monitoring and multi-session SLAM framework we present here could be beneficial in those application domains as well.

2.2.2 Long-term SLAM for Extensive Mapping

Graph-based SLAM solvers [39, 62, 88, 102] are popular tools in SLAM, however they are limited by computational complexity in many long-term applications. As a robot explores an area and localizes within its map, pose nodes are continually added to the graph. Therefore, optimizing a graph of modest spatial extent becomes intractable if the duration of the robot's exploration becomes sufficiently large. A number of proposed methods attempt to address this problem by removing nodes from the graph, thus reducing the graph's size. Recent emphasis has been placed on measurement composition using full-rank factors, such as laser scan matching or odometry [45, 99, 100]. Measurement composition synthesizes virtual constraints by compounding full-rank 6-degree-of-freedom (DOF) transformations, as shown in Fig. 2.2(a), however this operation is ill-defined for low-rank measurements, such as those produced by a monocular camera, shown in Fig. 2.2(b). This is one of the primary motivations for the development of the factor graph marginalization framework described by Carlevaris-Bianco and Eustice [25], called generic linear constraints (GLC), which address the low-rank problem.

The GLC method avoids measurement composition and works equally well using full-rank and low-rank loop-closing measurements. This is a critical requirement for the HAUV application because low-rank factors from either a monocular camera or an imaging sonar are the primary means of correcting navigational error from the DVL. Furthermore, measurement composition tends to re-use measurements when synthesizing virtual constraints. Doing so double-counts information and does not produce the correct marginalization in the reduced SLAM graph.

Previous studies of large-scale SLAM suggest solving SLAM using multiple maps to efficiently and consistently optimize very large graphs. The first method for constructing a large-scale map involves the use of submapping techniques, i.e., building local maps and then stitching the local maps together to build larger maps. This type of submap technique

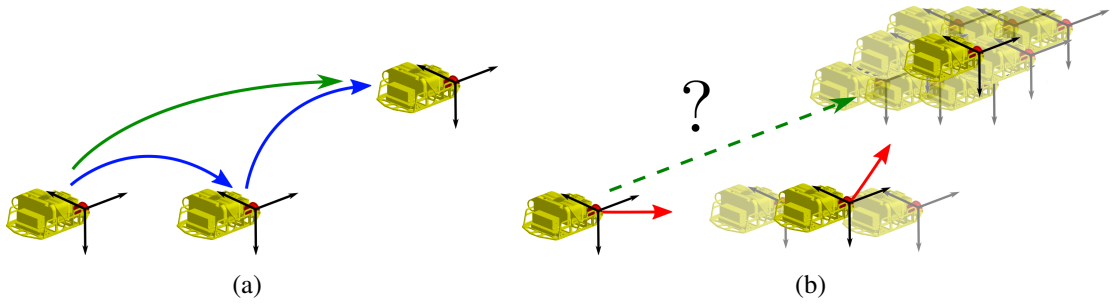
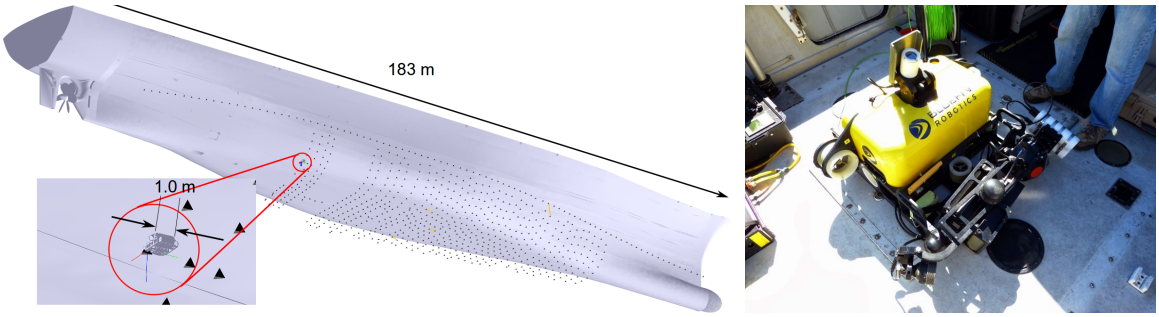


Figure 2.2: Illustration of measurement composition. This figure shows that the measurement composition technique, often used in variable marginalization in graph-based SLAM, is undefined for low-rank monocular camera measurements, as used in this work. For generating the green end-to-end 6-DOF transformation in (a), one simply has to compound the intermediate 6-DOF relative-pose transformations, in blue. This cannot be done for the intermediate 5-DOF measurements shown as red arrows in (b). In this case, computing the end-to-end baseline direction of motion is ill-posed because it depends on the scale at each camera measurement, which is not known.

was originally proposed by Leonard and Feder [106] to reduce the computational cost by decoupling the larger maps into submaps. This work was further developed into the *Atlas* framework by Bosse et al. [17] where each submap is represented as a coordinate system in a global graph. For example, Bosse and Zlot [16] developed a two-level loop-closing algorithm based on the *Atlas* framework, while Neira and Tardos [122] introduced a way to measure joint compatibility, which resulted in optimal data association. Examples of underwater submap applications include vision [141] and bathymetry maps from acoustic sensors [146]. Kim et al. [98] used a similar submapping approach, except in the context of a factor graph representation of cooperative SLAM. They used so-called *anchor nodes*, which are variable nodes representing the transformation from a common reference frame to the reference frame of each vehicle's SLAM graph. Anchor nodes are nearly identical to *base nodes* introduced by Ni, Steedly, and Dellaert [124]. They are a very simple way to extend observation models to perform submap alignment, which is why our multi-session SLAM approach (discussed in Section 2.4) formulates the session alignment using anchor nodes.

To efficiently perform multi-session SLAM in the absence of a readily-available absolute position reference requires an initial data-association step. By doing so, each map can be expressed into a common frame of reference, thus bounding the search space for identifying additional data associations. Recently, visual information has been exploited in data association by providing an independent loop-closing ability based upon appearance. Nistér and Stewénius [126] introduced a framework that used a bag-of-words (BoW) representation of



(a) (b)

Figure 2.3: A size comparison of the HAUV compared to a typical ship hull is shown in (a). A photograph of the robot on the deck of a small boat is shown in (b).

images to quickly compare the histogram of words between two images using vocabulary trees. Cummins and Newman [34] used a generative model to describe the co-occurrence of words in the training data. They developed the popular fast appearance-based matching (FAB-MAP) algorithm, which frames the problem as a recursive Bayes filter and showed that their system can leverage temporal information with or without a topological motion prior. Both of these methods have become a common practice for loop-closure detection in real-time SLAM front-ends.

Our approach also shares some similarity with methods that maintain visual diversity in long-term SLAM graphs. GraphSLAM by Thrun and Montemerlo [163] presents some early work on a large urban area with visual features. The method, however, was an offline approach with conventional optimization methods. Our work has more connection to recent online approaches such as the one introduced by Konolige and Bowman [99]. They represent a place in the SLAM graph as a collection of exemplar views. This encourages visually diverse keyframes in the SLAM graph, thus increasing potential matchability as more images are collected by the robot. Furthermore, this approach limits the number of views to be less than a user-defined parameter, which keeps the graph size bounded.

2.3 Underwater Visual SLAM with Planar Constraints

In this section, we introduce the overall system, including the SLAM back-end, and camera and plane constraints for the HAUV platform developed by Bluefin Robotics, shown in Fig. 2.3. A more detailed description of the HAUV can be found in Section 1.1.2. The illustration of the pose-graph framework is given in Fig. 2.4. The robot is provided with camera, planar, odometry measurements together with absolute measurements on depth and

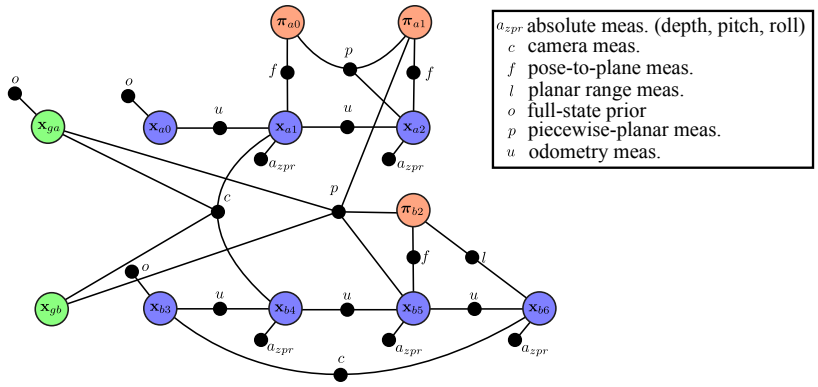


Figure 2.4: An illustration of a multi-session factor graph with camera, planar and other measurements. Orange, blue, and green variable nodes correspond to planes, vehicle poses, and anchor nodes [98], respectively. The top half of the nodes correspond to a prior session (reference frame a), while the bottom half belong to a subsequent session (reference frame b). Factor nodes are shown as black circles, along with the type of observation encoded as a script character. The use of odometry and prior factors is common-practice in pose-graph SLAM so these are not covered in detail in this chapter. The camera and planar measurements are discussed in more detail in Section 2.3.3 and Section 2.3.4, respectively.

attitude. Following a brief illustration of each module, we specifically focus on the planar constraints that enable substantially improved mapping and localization performance.

2.3.1 State Representation

For the remainder of this section, let $\mathbf{x}_{ij} = [x_{ij}, y_{ij}, z_{ij}, \phi_{ij}, \theta_{ij}, \psi_{ij}]^\top$ be the 6-DOF relative-pose of frame j as expressed in frame i, where x, y, z are the Cartesian translation components, and ϕ_{ij}, θ_{ij}, and ψ_{ij} denote the roll (x-axis), pitch (y-axis), and yaw (z-axis) Euler angles, respectively. ${}^i_j\mathrm{R}$ is the rotation matrix that rotates a vector in j's frame to a vector in i's frame, and ${}^i\mathbf{t}_{ij}$ is the translation from i to j as expressed in i's frame. Finally, g will refer to the global, or common, reference frame.

2.3.2 SLAM Back-end

The HAUV is equipped with a variety of sensors: a DVL and inertial measurement unit (IMU) for odometry, a pressure depth sensor, two cameras for 3D bearing measurements, and an imaging sonar. The use of these sensors for SLAM is described in detail by Johannsson et al. [82] and Kim and Eustice [95]. SLAM aims to correct for any accumulated navigational error when the robot senses environmental features it has seen previously. The DVL measures four beam ranges, along with four body-frame velocities. These range

measurements can also be converted to a sparse 3D point set, which allows us to fit local planar patches to the DVL data.

The SLAM system used on the HAUV has evolved from an offline filtering-based approach with manual data association [172], to a real-time factor graph optimization framework that supports both sonar and monocular camera measurements automatically [72]. Currently, our SLAM system uses the incremental smoothing and mapping (iSAM) algorithm as the optimization back-end [87–89].

2.3.3 Camera Constraints

The previously-mentioned underwater and periscope cameras can both be used to derive spatial constraints from two-view feature correspondence. The underwater environment is particularly challenging because it does not always contain visually useful features for camera-derived measurements. To alleviate this issue, Kim and Eustice [96] allowed for greatly increased computational performance by only considering visually salient keyframes in the visual SLAM pipeline. They derived a local saliency score using a coarse BoW representation of each keyframe from speeded up robust features (SURF) descriptors [10]. To derive two-view camera constraints from overlapping images, the system searches for correspondence between keyframes using the scale-invariant feature transform (SIFT) descriptor [109]. A scene depth prior based upon the DVL and a relative-pose prior based on the SLAM estimate offer a constrained putative correspondence search that is faster and more consistent than typical appearance-only random sample consensus (RANSAC)-based approaches [24, 52].

From two overlapping images taken from our monocular camera, we can derive a spatial constraint, modulo scale, between the vehicle poses from which those images where taken. This measurement, $h^{5\text{dof}}$, between two poses i and j therefore has five DOFs: three rotations, and a vector representing the direction of translation that is parameterized by azimuth and elevation angles. We denote this measurements as $h^{5\text{dof}}\left(\mathbf{x}_{gi}, \mathbf{x}_{gj}\right) = [\alpha_{ij}, \beta_{ij}, \phi_{ij}, \theta_{ij}, \psi_{ij}]^\top$, consisting of the baseline direction of motion azimuth α_{ij}, elevation β_{ij}, and the relative Euler angles ϕ_{ij}, θ_{ij}, ψ_{ij}.

We have found that adding camera measurements with low baseline can potentially produce outlier measurements. These outliers are occasionally not identified with a Mahalanobis distance check prior to their addition to the SLAM graph. We therefore use the robust back-end method known as dynamic covariance scaling (DCS) [1]. This method is based on the work by Sunderhauf and Protzel [160], which employs switching variables to turn on and off measurements by scaling the information matrix for loop-closing measurements.

DCS does this in closed form, which avoids having to increase the dimensionality of the state to estimate the scale variables. Thus, the robust monocular camera factor between poses $\mathbf{x}_{gi}$ and $\mathbf{x}_{gj}$ is:

$$\chi^2_{DCS_{ij}} = \left\| \mathbf{z}_{ij}^{5\text{dof}} - h^{5\text{dof}}\left(\mathbf{x}_{gi}, \mathbf{x}_{gj}\right) \right\|^2_{\frac{1}{s_{ij}^2}\Xi_{\mathbf{z}_{ij}}}, \tag{2.1}$$

where $\frac{1}{s_{ij}^2}\Xi_{\mathbf{z}_{ij}}$ is the DCS scaled measurement covariance matrix. Agarwal et al. [1] show that the covariance scale variable, s_{ij}, is given by

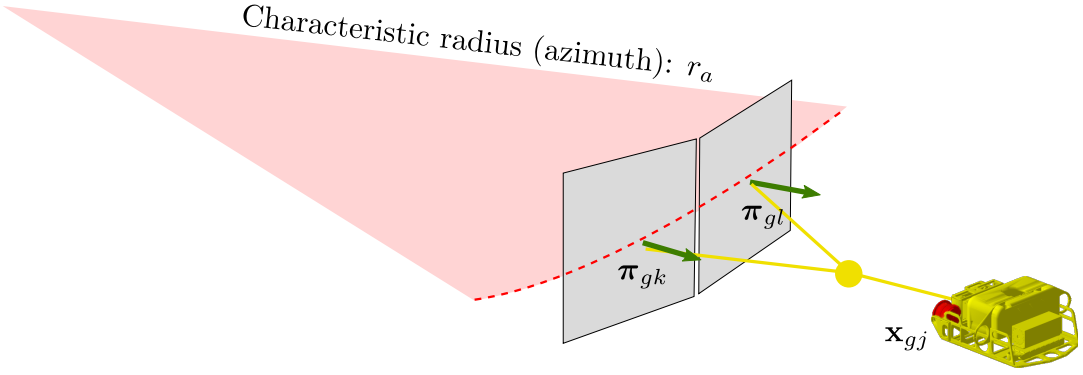
$$s_{ij} = \min\left(1, \frac{2\Phi}{\Phi + \chi_l^2}\right),$$

where χ_l^2 is the unscaled chi-squared error, computed from (2.1) using the unscaled measurement covariance, $\Xi_{\mathbf{z}_{ij}}$. Higher values of the parameter Φ make it more likely to accept loop closures. For our field trials, we choose $\Phi = 5$; however, values between 1 and 10 produce nearly identical results.

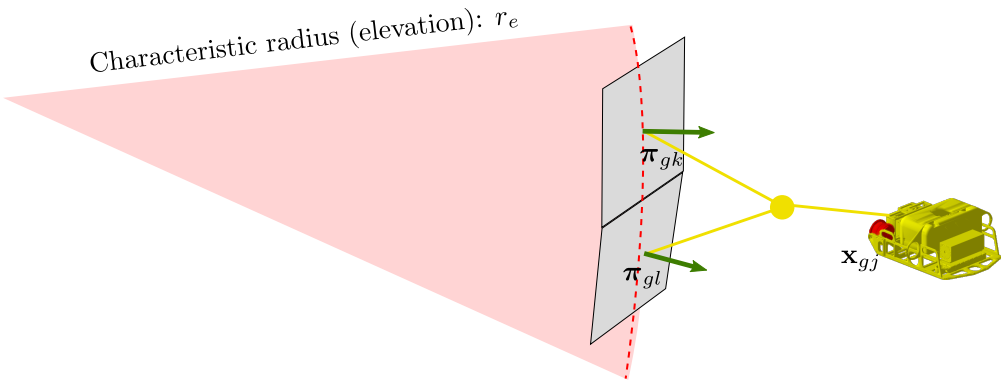
2.3.4 SLAM with Planar Segments

To further constrain the vehicle's trajectory, we use planar patches as nodes in the SLAM graph using a piecewise-planar model [135], which is summarized in Fig. 2.5. This method provides an explicit representation of the ship hull surface in the factor graph, and produces accurate and efficient maps using only a sparse 3D point cloud (such as one produced by the underwater DVL navigation sensor). Furthermore, this method preserves the overall curvature of the ship's exterior hull, assuming that the robot has coarse prior knowledge on the side-to-side and top-to-bottom characteristic curvature of the surface. This distinguishes our work from other planar SLAM approaches [140, 164, 173] in two ways. First, other methods make use of a much more data-rich 3D laser scanners or depth cameras. Second, these other approaches are experimentally evaluated in structured environments, like office buildings, that contain many planar walls at sharp angles to each other. Unfortunately, these approaches require that the planes be directly observed from two or more views. However, this is often not achievable in ship hull inspection because (*i*) the hull is smoothly curved, and (*ii*) the DVL's field of view is both more narrow and more sparse than a typical laser scanner or depth camera. In addition, because the DVL returns are very sparse, this differentiates our work from other underwater SLAM methods that use a 3D multibeam sonar [22, 167].

In this section, we explain the plane constraints from a single session point of view, but



(a)



(b)

Figure 2.5: Characteristic radii overview. We represent the ship hull surface as a collection of locally planar feature patches (gray), with a simple assumed prior characteristic curvature model. The residual error of two co-registered planes are weighted in such a way so that only errors that extend outside of the deviation explained by the characteristic curvature are significant. The method relies on two characteristic radii parameters: one for the azimuth (side-to-side variation, shown in (a)), and one for elevation (top-down variation, shown in (b)). In Appendix C we offer a sensitivity analysis of these user-defined parameters through simulation.

another significant strength of these constraints are in robust registration that allows for multi-session SLAM, which will be discussed in Section 2.4.

2.3.4.1 Plane Composition Operators

The planar observation models used in this chapter involve expressing planes in multiple frames of reference. In this section, we describe these operations in detail. Let $\boldsymbol{\pi}_{ik}$ be the plane indexed by k, expressed with respect to frame i. When i is the global frame, g, this plane corresponds to a variable node in our factor graph. We parametrize the plane $\boldsymbol{\pi}_{gk} = [n^x_{gk}, n^y_{gk}, n^z_{gk}]^\top$ using three numbers, where n^x_{gk}, n^y_{gk}, and n^z_{gk} are the x, y, and z components of the (non-unit) normal vector to the plane k, expressed in the global-frame, g. We do not use the popular Hessian normal form simply because it is parameterized by four numbers, despite having only *three* degrees of freedom. This over-parameterization would therefore result in a singular covariance matrix. The Hessian normal form is given by: $\widehat{\mathbf{n}}^\top \mathbf{x} = -p$ for any point $\mathbf{x}$ that lies on the plane, where $\widehat{\mathbf{n}}$ is the unit normal of the plane, and p is the distance of the plane to the origin. Conversely, our parameterization uses only three numbers. We simply scale the unit-normal by the distance to the origin, yielding what we denote as the $\boldsymbol{\pi}$ symbol. With our parameterization, we have $\boldsymbol{\pi}^\top \mathbf{x} = -\|\boldsymbol{\pi}\|^2$ for any point $\mathbf{x}$ on the plane.

The frame of reference of the plane is denoted with a subscript whereby the plane indexed by k as expressed with respect to some frame i is denoted $\boldsymbol{\pi}_{ik}$. If one wishes to express a plane in a different frame of reference, one simply applies the $\boxminus$ and $\boxplus$ operators, which take as input a 6-DOF pose and 3-DOF plane. Let $\mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik} = \boldsymbol{\pi}_{jk}$, where

$$\boldsymbol{\pi}_{jk} = \mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik} = \frac{\left({}^i\mathbf{t}_{ij}{}^\top \boldsymbol{\pi}_{ik} + \|\boldsymbol{\pi}_{ik}\|^2\right) \mathrm{R}^i_j \boldsymbol{\pi}_{ik}}{\|\boldsymbol{\pi}_{ik}\|^2}.$$

The $\boxminus$ operator is commonly used to evaluate factor potentials, discussed in the next section. We also define the $\boxplus$ operator, where $\mathbf{x}_{ij} \boxplus \boldsymbol{\pi}_{jk} = \boldsymbol{\pi}_{ik}$. This can be expressed in terms of the $\boxminus$ operator by

$$\mathbf{x}_{ij} \boxplus \boldsymbol{\pi}_{jk} = \ominus \mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{jk},$$

where $\ominus$ is the pose-inversion operation [158]. The $\boxplus$ operator is commonly used to transform plane measurements from the vehicle frame to the global, but not to evaluate factor potentials.

More details regarding the choice of planar parameterization, along with a derivation of the composition operator can be found in Appendix B.

2.3.4.2 Planar Constraints between Poses and other Planes

In this section we describe observation models used to add measurements to our SLAM graph. A binary factor potential of the form

$$\Psi(\mathbf{x}_{gi}, \boldsymbol{\pi}_{gl}; \mathbf{z}_{\boldsymbol{\pi}_{il}}, \Sigma_{\mathbf{z}_{\boldsymbol{\pi}_{il}}}) = \| \mathbf{z}_{\boldsymbol{\pi}_{il}} - (\mathbf{x}_{gi} \boxminus \boldsymbol{\pi}_{gl}) \|^2_{\Sigma_{\mathbf{z}_{\boldsymbol{\pi}_{il}}}} \tag{2.2}$$

is used when the DVL sensor determines a good planar fit of a sliding time-window of 3D points. The covariance matrix, $\Sigma_{\mathbf{z}_{\boldsymbol{\pi}_{il}}}$, is determined by first-order approximation of a function that fits a plane from the 3D points, which are assumed to be corrupted by Gaussian noise.

A piecewise-planar factor potential is similarly used to constrain the surface normals of two plane nodes with respect to a common pose node. It is given by the expression

$$\begin{aligned} &\Omega(\mathbf{x}_{gi}, \boldsymbol{\pi}_{gk}, \boldsymbol{\pi}_{gl}; \mathrm{W}_{ijkl}) = \\ &\quad \| (\mathbf{x}_{gi} \boxminus \boldsymbol{\pi}_{gk}) - (\mathbf{x}_{gi} \boxminus \boldsymbol{\pi}_{gl}) \|_{\mathrm{W}_{ijkl}}. \end{aligned} \tag{2.3}$$

A weight matrix, W_{ijkl}, is based on the characteristic curvature of the ship hull. It is a function of two vehicle poses from which the planes k and l were observed (that is, i and j, respectively):

$$\mathrm{W}_{ijkl} = \mathrm{diag}\left(\Delta_x^2, \Delta_y^2, \Delta_z^2\right) + \mathrm{C}\Sigma_{ij}\mathrm{C}^\top,$$

where Σ_{ij} is the marginal covariance between poses i and j. C is the Jacobian of the function $c(\cdot)$ that is a simple curvature model of the surface being observed. Since the two measured planes will not be perfectly coplanar due to surface curvature, it is a simple way to take this into account when computing error from the $\boxminus$ operator described in Section 2.3.4. This approach is illustrated in Fig. 2.5, and the mathematical formulation of the curvature model is

$$\begin{bmatrix} \Delta_x \\ \Delta_y \\ \Delta_z \end{bmatrix} = c\left(\mathbf{x}_{gi}, \mathbf{x}_{gj}; \boldsymbol{\pi}_{ik}, \boldsymbol{\pi}_{jl}, r_a, r_e\right) = \boldsymbol{\pi}_{jl} - \left(\mathbf{x}_{ij} \boxminus \mathrm{trans}\left(\mathrm{dir}\left(\boldsymbol{\pi}_{ik}\right) + \begin{bmatrix} t^i_{ij_y}/r_a \\ t^i_{ij_z}/r_e \\ 0 \end{bmatrix} \right) \right),$$

where $\mathrm{dir}(\cdot)$ converts a 3-dimensional Cartesian vector to one expressed in 3-dimensional spherical coordinates: azimuth, elevation, and distance to the origin. Conversely, $\mathrm{trans}(\cdot)$ converts spherical coordinates to Cartesian. See Appendix B.6 for the full definition of these functions. The user-defined characteristic radii for azimuth and elevation are denoted r_a

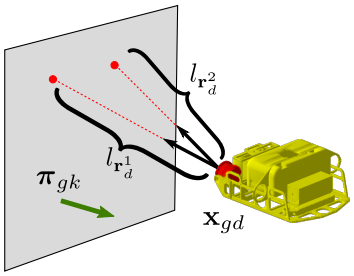


Figure 2.6: Raycasting overview. Given plane k and DVL pose d in the global frame, the points of intersection for each ray can be computed so long as the ray origin and direction are known using (2.4). We can compare the predicted range to the DVL-measured range, and use this as a spatial constraint between the pose and plane. This is particularly useful when the points detected by the plane are ill-conditioned for fitting a plane (such as only two DVL beams that will always lie on a line, as in this figure).

and r_e, respectively. For the experimental results in this chapter, we use $r_a = 322$ m and $r_e = 7$ m. Similar to what is described in Ozog and Eustice [135], the results presented in this chapter are almost identical so long as r_a and r_e are within the same order of magnitude as these provided values.

2.3.4.3 Range Constraints between Planes and Poses

In the event that fitting a plane to a point cloud is ill-conditioned, as is the case when all points are colinear, or when there is only a single point, we use a simple ray-tracing method to constrain a plane with these ill-conditioned DVL beams. This is illustrated in Fig. 2.6.

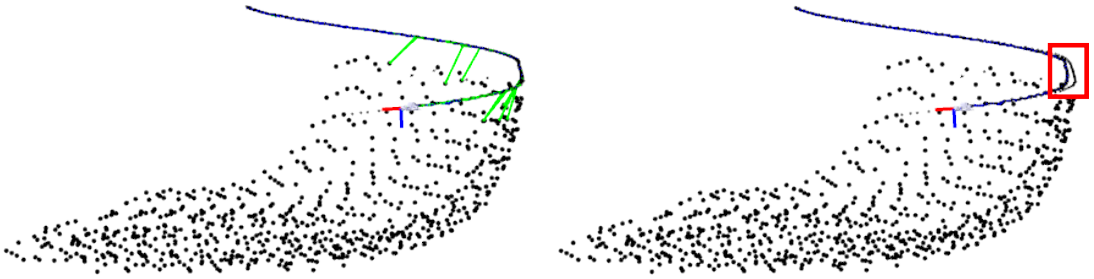
Let ${}^i\mathbf{r}_n$ be the n-th unit-norm ray expressed in pose frame i. If ${}^i\mathbf{r}_n$ is known to intersect with plane $\boldsymbol{\pi}_{ik}$, then we can compute the predicted ray length, $\widehat{l}_{i_{\mathbf{r}_n}}$, by tracing the ray until it intersects with the surface. Because the surface is planar, we can derive a closed-form expression for the length:

$$\widehat{l}_{i_{\mathbf{r}_n}}\left(\mathbf{x}_{ij}, \boldsymbol{\pi}_{ik}; {}^i\mathbf{r}_n\right) = \frac{\|\mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik}\|}{{}^i\mathbf{r}_n{}^\top \left(\mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik}\right)}. \tag{2.4}$$

The resulting factor potential between pose j and plane k is computed by weighting the difference between the observed beam range, $z_{l_{i_{\mathbf{r}_n}}}$, and the predicted range from above:

$$\Phi\left(\mathbf{x}_{ij}, \boldsymbol{\pi}_{ik}; z_{l_{i_{\mathbf{r}_n}}}, \sigma_{z_{l_n}}, {}^i\mathbf{r}_n\right) = \left(\frac{\widehat{l}_{i_{\mathbf{r}_n}}\left(\mathbf{x}_{ij}, \boldsymbol{\pi}_{ik}; {}^i\mathbf{r}_n\right) - z_{l_{i_{\mathbf{r}_n}}}}{\sigma_{z_{l_n}}}\right)^2, \tag{2.5}$$

where $\sigma_{z_{l_n}}$ is the measurement noise variance of the range reading from the n^{th} beam.



(a) With range constraints (b) Without range constraints

Figure 2.7: Example of the utility of raycasting factors. In practice, we find the HAUV may only successfully measure the top two DVL returns — especially when surfaced. Here, we show the benefits of leveraging this information in the context of SLAM. In each figure, the viewpoint is from stern-to-bow, and green lines encode the invocation of a range constraint. In (a), two surface missions are well-aligned, whereas in (b), there is some noticeable drift, highlighted in red. In each figure, black dots denote pose nodes in a sparsified graph using the GLC method discussed in Section 2.4.1.

Intuitively, this factor potential is less informative than the piecewise planar factor defined in (2.3). However, in the event that a plane is not observable from the DVL's point cloud, it still provides a useful constraint for the SLAM trajectory if a local planar patch has been previously established. In practice, we use this constraint in the event that the incident angle for the bottom two DVL beams is too large, which we find can happen when the HAUV is at the surface. An example of the benefit of this information is shown in Fig. 2.7.

2.4 Multi-Session SLAM for Long-term Hull Inspection

The typical use case for the SLAM system discussed above is for single-session surveys. When the robot begins a new survey minutes or days after the previous survey, the navigation is reset and all the information from the previous map is no longer used in navigation correction. For long-term or multi-robot applications, this is detrimental for two reasons. First, there is less information for the robot to correct its navigation. Second, each session has a different local reference frame. If a particular area for one of these surveys demands attention, it requires a human-in-the-loop to identify where the area of interest lies in a hull-relative reference frame. It is therefore beneficial, though challenging, to have the surveying robot build each map with respect to the same reference frame. In this section, we present: (*i*) an accurate and efficient global localization system to make the initial alignment; (*ii*) surface-based SLAM techniques to make the real-time alignment robust and accurate; and (*iii*) a framework for managing the computational complexity of optimizing

long-term SLAM graphs.

To illustrate how these techniques are used in practice during real-world operations, the following is a high-level outline of our multi-session SLAM system: (*i*) load past sessions as a SLAM graph; (*ii*) sparsify the graph with GLC; (*iii*) deploy the HAUV and start building a separate, unaligned pose-graph; (*iv*) localize to the past session with an initial guess of the alignment; and (*v*) refine alignment using monocular camera measurements and piecewise-planar constraints on anchor nodes. These steps are illustrated in Fig. 2.8.

GLC has been thoroughly-evaluated on standard SLAM graphs containing pose nodes and landmark nodes. However, the graphs used in this work are significantly more nonstandard because they contain anchor nodes, planar nodes, and planar observation models. This section will also describe how GLC can be used with these aspects of our SLAM system.

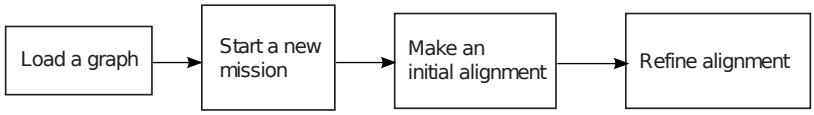
2.4.1 GLC-based Approximate Node Removal

A detailed derivation and computational performance analysis of GLC node marginalization and removal can be found in Carlevaris-Bianco and Eustice [25, 26]. GLC-based graph sparsification starts with an n-ary factor that captures the information within the elimination clique following marginalization. In short, this n-ary factor is computed by considering all the measurements contained in the Markov blanket of the node that is to be marginalized. To handle low-rank measurements, this method computes eigenvalue-decomposition of the induced target information, Λ_t, and forms a generic linear observation model for the n-ary factor:

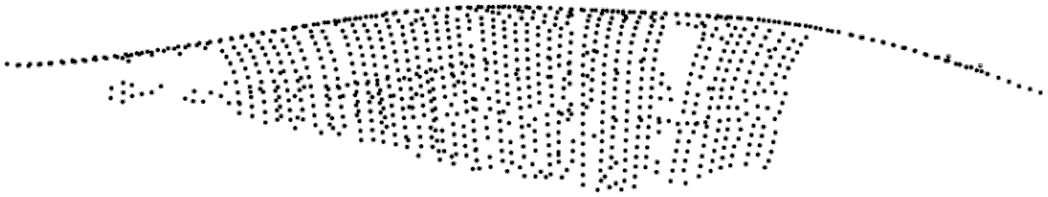
$$\mathbf{z}_{glc} = \mathrm{G}\mathbf{x}_c + \mathbf{w}', \tag{2.6}$$

where $\mathbf{w}' \sim \mathcal{N}(\mathbf{0}, \mathrm{I}_{q\times q})$, $\mathrm{G} = \mathrm{D}^{1/2}\mathrm{U}^\top$, $\Lambda_t = \mathrm{UDU}^\top$, q is the rank of Λ_t, and $\mathbf{x}_c$ is the current linearization of nodes contained in the elimination clique. $\mathrm{UDU}^\top$ is the Eigendecomposition of Λ_t, where U is a $p \times q$ matrix of Eigenvectors and D is a $q \times q$ diagonal matrix of Eigenvalues. To preserve sparsity in the graph, the target information Λ_t is approximated using a Chow-Liu tree (CLT) structure, where the CLT's unary and binary potentials are represented as GLC factors. These resulting unary and binary factors replace the node's original surrounding factors.

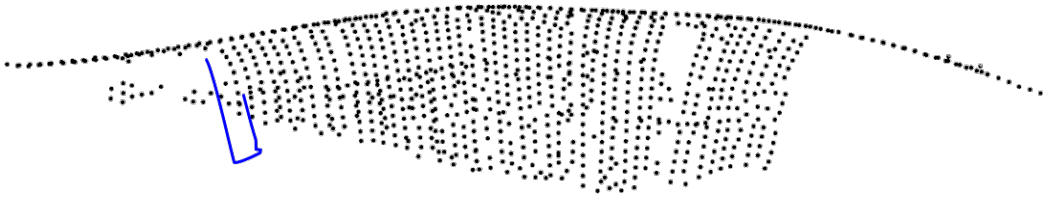
GLC optionally supports node reparameterization into a local reference frame around $\mathbf{x}_c$ when evaluating the observation model from (2.6). This avoids committing to a world-frame linearization if the nodes are not well-optimized. Instead, nodes are linearized about a local relative-frame transformation. This relative transformation arbitrarily picks a single pose node in the elimination clique as the common frame. For planar nodes, we use the $\boxminus$



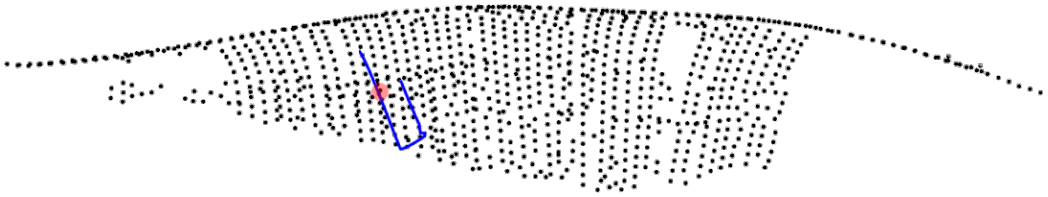
(a) Multi-session SLAM diagram



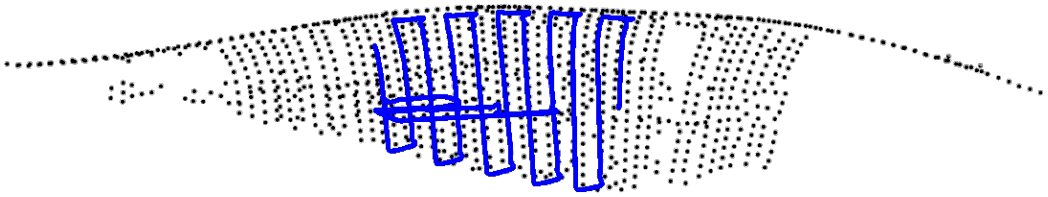
(b) Sparse graph loaded (factors not rendered)



(c) Unaligned pose-graph

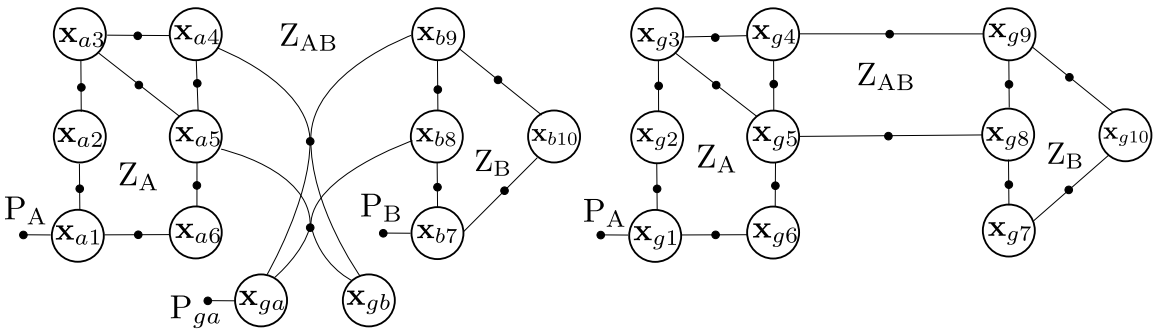


(d) Initial alignment to sparse graph



(e) Refine alignment and complete survey

Figure 2.8: Depiction of multi-session SLAM using the techniques provided in Section 2.4. The red region in (d) denotes the point of map reacquisition, where the robot determines a rough initial alignment to the graph from (b). We refine this alignment in real-time using the same techniques as our single-session system from previous work. Once the session from (e) is complete, we sparsify and repeat the process with additional sessions.



(a) factor graph with anchor nodes (b) factor graph with global reference frame, g

Figure 2.9: A factor graph topology for a simple SLAM example involving two sessions, A (left side) and B (right side). In (a), the graph encodes the distribution from (2.7), where each session has an associated anchor node. This graph can be converted to a global representation in (b) without any loss of information using (2.9). The only discarded factors are the full-state prior factors in session B and on the anchor node $\mathbf{x}_{ga}$. These prior factors have little probabilistic importance and function primarily to keep the SLAM information matrix non-singular.

operator to express planes with respect to a root node. In the next section, we describe how to support anchor nodes in GLC's reparameterization step.

2.4.2 Global Multi-Session SLAM from Anchor Nodes

For a robot or multiple robots performing multi-session SLAM, a pose-graph containing anchor nodes is a popular method to align multiple relative graphs [98]. In this approach, each robot session has an associated anchor node. An anchor node is a node containing the transformation from a global reference frame to the corresponding session's reference frame. The important advantages of anchor nodes over global multi-session SLAM, where *all* sessions are expressed in a common frame, are (*i*) faster convergence of the nonlinear least-squares solver, and (*ii*) individual sessions can optimize their pose graphs before any constraints between them are observed.

For a two-session case consisting of sessions A and B, the factor graph containing anchor nodes, from Fig. 2.9(a), encodes a probability distribution of the form

$$\begin{aligned} &p(X \mid Z_{\mathrm{A}}, Z_{\mathrm{B}}, Z_{\mathrm{AB}}, P_{\mathrm{A}}, P_{\mathrm{B}}, P_{ga}) = \\ &p(\mathbf{x}_{ga}, \mathbf{x}_{a1_{\mathrm{A}}}, \mathbf{x}_{a2_{\mathrm{A}}}, \ldots, \mathbf{x}_{aN_{\mathrm{A}}}, \\ &\quad \mathbf{x}_{gb}, \mathbf{x}_{b1_{\mathrm{B}}}, \mathbf{x}_{b2_{\mathrm{B}}}, \ldots, \mathbf{x}_{bM_{\mathrm{B}}} \mid Z_{\mathrm{A}}, Z_{\mathrm{B}}, Z_{\mathrm{AB}}, P_{\mathrm{A}}, P_{\mathrm{B}}, P_{ga}), \end{aligned} \tag{2.7}$$

where $\mathbf{x}_{ga}$ is an anchor node representing the 6-DOF transformation from the global-frame,

g, to the reference frame a of session A. $\mathbf{x}_{ai_\mathrm{A}}$ is the 6-DOF relative-pose from frame a to frame i of session A. X denotes the set of variable nodes in the factor graph (i.e., the unknowns). Z_A and Z_B denote the sensor and odometry measurements contained in sessions A and B, respectively, and Z_AB denotes the sensor measurements between nodes in sessions A and B. P_A and P_B denote full-state priors in sessions A and B, and P_{ga} denotes the full-state prior on the anchor node, $\mathbf{x}_{ga}$. Finally, N and M are the number of nodes in sessions A and B, respectively, not including the anchor node (Fig. 2.9).

The parameterization from (2.7) is somewhat inconvenient because in order to place the nodes into a common frame (when visualizing the distribution, for instance, or when computing the expected information gain of a measurement between two nodes), a sparse nonlinear function, f, must be applied to the distribution:

$$\mu_{f(X)} = \begin{bmatrix} \mathbf{x}_{ga} \oplus \mathbf{x}_{a1_\mathrm{A}} \\ \vdots \\ \mathbf{x}_{ga} \oplus \mathbf{x}_{aN_\mathrm{A}} \\ \mathbf{x}_{gb} \oplus \mathbf{x}_{b1_\mathrm{B}} \\ \vdots \\ \mathbf{x}_{gb} \oplus \mathbf{x}_{aM_\mathrm{B}} \end{bmatrix}, \tag{2.8}$$

where $\oplus$ is defined in Smith, Self, and Cheeseman [158] as the compounding operation. The covariance of the resulting distribution is computed to first order as

$$\Sigma_{f(X)} = \mathrm{J}_f \Sigma_\mathrm{X} \mathrm{J}_f^\top,$$

where J_f is the Jacobian of $\mu_{f(X)}$.

Extending this analysis to more than two sessions is trivial. It is also straightforward when X contains plane nodes. In this case, the sparse nonlinear function f contains the $\boxplus$ operator discussed in Section 2.3.4.

For convenience, we alter the distribution so that all nodes are in a common frame. Along with simplifying path planning, visualization, and hypothesizing informative measurements, doing so allows for trivial application of the GLC reparameterization suggested by Carlevaris-Bianco and Eustice [26]. This alteration is illustrated in Fig. 2.9. First, we reduce the state variables using (2.8), effectively removing the anchor nodes from the factor graph. Next, we remove the full-state priors for nodes in session B and the anchor node in session A, leaving

us with the distribution

$$
\begin{aligned}
&p(X' \mid Z_{\mathrm{A}}, Z_{\mathrm{B}}, Z_{\mathrm{AB}}, P_{\mathrm{A}}) = \\
&p(\mathbf{x}_{g1_{\mathrm{A}}}, \mathbf{x}_{g2_{\mathrm{A}}} \ldots, \mathbf{x}_{gN_{\mathrm{A}}}, \\
&\quad \mathbf{x}_{g1_{\mathrm{B}}}, \mathbf{x}_{g2_{\mathrm{B}}} \ldots, \mathbf{x}_{gM_{\mathrm{B}}} \mid Z_{\mathrm{A}}, Z_{\mathrm{B}}, Z_{\mathrm{AB}}, P_{\mathrm{A}}),
\end{aligned} \tag{2.9}
$$

In this way, none of the sensor measurements are discarded when converting the relative pose-graphs to the global-frame. In practice, we perform this operation immediately after the robot completes a survey, and before performing offline GLC-sparsification.

2.4.3 Reacquisition to Sparsified Graph by Particle Filtering

This section describes how we accomplish the initial alignment step illustrated in Fig. 2.8(d). Once a graph has been sparsified using GLC, we use a particle filter to estimate a distribution of likely poses in the reference frame of the past session. Our particle filter is based on a classic Monte-Carlo localization framework [40]. We use odometry, depth, pitch, and roll measurements derived from sensors to propagate particles to their next state. We use planar measurements to compute importance weights for each particle, which guides the resampling step. Our particle filter approach is summarized in Fig. 2.10.

2.4.3.1 Weighting particles from planar measurements

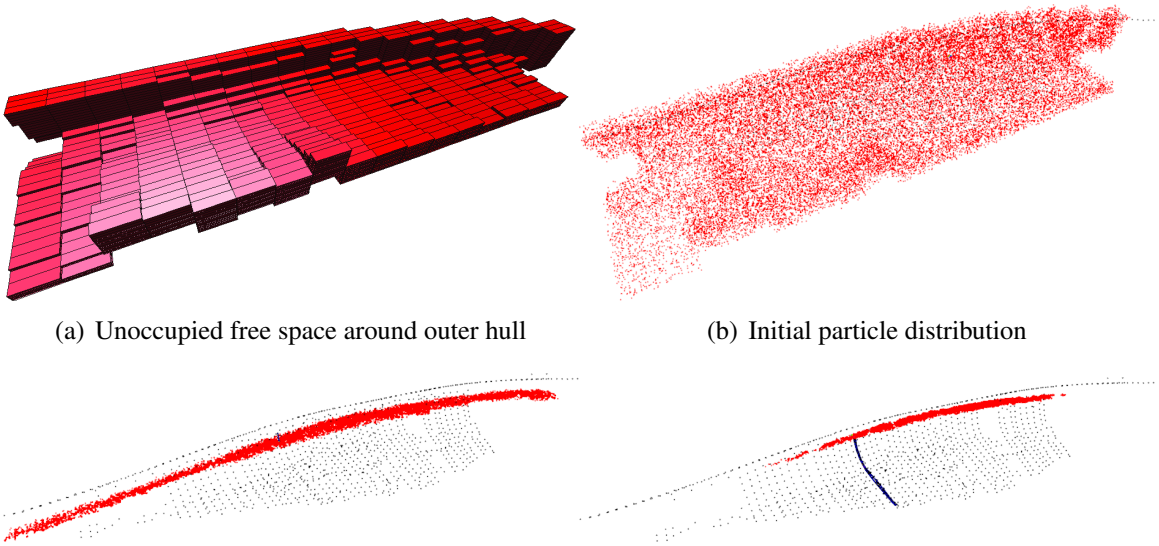
To weight the particles as new plane measurements are observed, we use a method similar to the computation of potentials used in our factor graph, described in Section 2.3.4. When a plane-fitting measurement, $\mathbf{z}_{\boldsymbol{\pi}_{ik}}$ from (2.2), is received by the particle filter, it finds the nearest neighboring pose node i according to

$$
i = \underset{i \in \mathrm{I}_{\Pi}}{\operatorname{argmin}} \|\mathbf{t}^{g}_{gp_i} - \mathbf{t}^{g}_{gi}\|,
$$

where I_{Π} is the set of pose indices in the GLC-sparsified graph that have a corresponding plane. Finding the minimum over all I_{Π} is quite slow for large graphs, so this operation is approximated using a k-dimensional (KD)-tree from the `FLANN` library [119].

Next, we take i' to be the index of the plane that is observed from pose i. Finally, we set the weight, w_{p_i}, by computing the Mahalanobis distance between the observed and expected planar measurement:

$$
w_{p_i} = \left\| \mathbf{z}_{\boldsymbol{\pi}_{p_i k}} - \left(\mathbf{x}_{gi} \boxminus \mathbf{x}_{gi'}\right) \right\|^2_{\Sigma_{ii'}}.
$$



(a) Unoccupied free space around outer hull

(b) Initial particle distribution

(c) Particle distribution at start of first trackline

(d) Particle distribution at end of first trackline

Figure 2.10: Particle filter reacquisition into a previous session SLAM graph. We improve the efficiency of our particle filter by computing a simple occupancy grid, shown in (a), based upon the planar features of the sparsified graph. Only the grid cells that lie less than $3\ \mathrm{m}$ from the surface of the nearest plane may contain particles from the initial distribution. We uniformly distribute particles, shown as red dots in (b), over the cells from (a). When the new survey begins, the particles are weighted based on their compatibility with planar measurements. In (c) and (d), particles are overlaid on the pose-graph where the black dots are graph nodes. Over time, the distribution of particles will tend toward a smaller set of candidate keyframes to search.

Algorithm 1 Match current keyframe to candidate keyframes based on particle distribution

1: **Input:** Particles $p_1 \dots p_N$, current keyframe k, set of all pose indices in GLC-sparsified graph with corresponding keyframe I_K
2: $\mathrm{S} \leftarrow \varnothing$
3: **for** $p_i \in \{p_1 \dots p_N\}$ **do**
4: $\quad \mathrm{S} \leftarrow \mathrm{S} \cup \texttt{NeighestNeighbor}(p_i, \mathrm{I}_K)$ // Uses kd-tree
5: **end for**
6: **Output:** $\texttt{FindBestMatch}(k, \mathrm{S})$ // Uses GPU

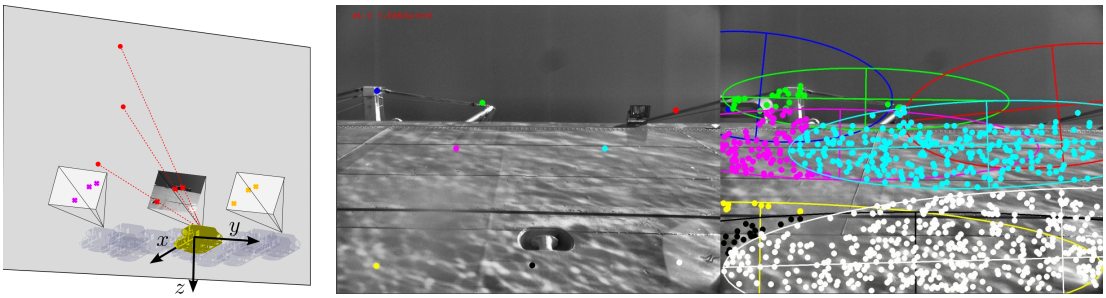
To get the initial distribution of particles, we compute a simple binary 3D occupancy grid from the GLC-sparsified graph, and remove any particles that are assigned to occupied cells. For each cell, if it lies outside the nearest pose-plane pair, that cell is marked as "not occupied." Otherwise, the cell is marked as "occupied."

We find that the particle distribution tends toward a narrow band at the proper depth. With the addition of planar measurements, this distribution can cull out ends of the ship that do not agree with the structural information observed by the robot. This behavior is shown in Fig. 2.10(c) and Fig. 2.10(d).

2.4.3.2 Keyframe Matching with SIFT and RANSAC

If after resampling, the distribution of particles is sufficiently small in the x, y plane, we find the set of all images contained in the particle distribution. We then initialize a keyframe matching algorithm that searches for an image from previous sessions that match an image from the current session.

More specifically, we build a set of candidate images when the square-root of the determinant of the marginal x, y covariance is less than $250\ \mathrm{m}^2$. This threshold tends to produce a keyframe set that contains about 100 images and takes roughly three to four seconds to perform a brute-force search. This step, denoted as FINDBESTMATCH in Algorithm 1, uses a graphical processing unit (GPU) for SIFT descriptor extraction and matching, and finally rejects outliers by using an eight-point RANSAC algorithm to fit a fundamental matrix between the current keyframe and a candidate image extracted from the set. The keyframe with the most inliers above a threshold is selected to be the best match. For our experiments, we choose a threshold of 12 inliers to avoid false positives. If no matches are found, the search is repeated when the robot moves approximately one meter from the point at which the previous search was attempted. Doing so prevents redundant searching.



(a) Planar scene and relative-pose prior for upward-looking periscope

(b) 99% confidence regions for 2D point transfer for real imagery

Figure 2.11: Constrained descriptor matching using periscope images. When searching for putative correspondence, the geometry of the periscope images can be exploited since the variation within relative-poses from the same place is mostly in the y-axis ("side-to-side"). First, we can raytrace the points from the prior vehicle pose assuming a planar scene, as shown in (a). Second, we model the pose uncertainty as a Gaussian ellipsoid and project this uncertainty as 2D ellipses in the candidate keyframe's image plane, shown in (b). We can search for putative SIFT correspondences within these ellipses, rather than searching the entire image.

2.4.3.3 Planar Scene Prior for Putative Correspondence in Periscope Images

The RANSAC approach from the previous section assumes no scene prior when determining putative matches due to the small baseline of overlapping underwater images. However, for above-water images where the vehicle is floating, we assume a coarse prior as follows. The uncertainty between two matching above-water poses can be modeled as a covariance ellipsoid with high variation in the side-to-side axis. The other dimensions are well-instrumented with depth and IMU sensors. This can be probabalistically modeled by assuming a zero-mean transformation between the two candidate poses with an appropriately-constructed covariance matrix.

If we further approximate the periscope scene as purely planar, we can back-project points from the first candidate camera pose onto a known plane in 3D space. We can then re-project these points into the image of the second candidate camera, and propagate the covariance terms using an unscented transform (UT) [85, 134]. In Appendix D we discuss this formulation in more detail.

This process is illustrated in Fig. 2.11, and is a special case of the PCCS described by Eustice, Pizarro, and Singh [52]. Note that this technique is only useful for the periscope images, where the scene is large as compared to the underwater images. Effectively, this constrains corresponding features in periscope images to share similar pixel row indices.

Algorithm 2 Sparsification algorithm

1: **Input**: Alignment of two SLAM sessions, A+B. A is the pre-sparsified prior session and B is the current session.
2: **Output**: Set of all nodes to marginalize, S. Updates exemplar views.
3: $\mathrm{N} = \mathrm{SpatiallyRedundantNodes(B)}$
4: **for** node in B and not in N **do**
5: views $\leftarrow \mathrm{NearestExemplarClusterOrNew(A}, node)$
6: **if** AnyViewCloseInTime(views, node) or not VisuallySalient(node) **then**
7: S $\leftarrow$ S $\cup$ node
8: **else**
9: S $\leftarrow$ S $\cup$ Push(views, node) // "views" is fixed-size FIFO. If views is full, Push() returns popped node.
10: **end if**
11: **end for**
12: $\mathrm{GlcRemove(A + B, S \cup N)}$

By contrast, correspondences in underwater images could have very different row indices. Therefore, for the underwater case, we revert to a purely putative correspondence search because it runs faster on a GPU.

2.4.3.4 Maintaining Keyframe Diversity in Merged Graph

Maintaining diversity in the keyframes associated with vehicle pose nodes in the SLAM graph is critically important for the particle filter to successfully localize. To maintain diversity of the images associated with nodes in our merged SLAM graph, we use a simple algorithm that is inspired from the work by Konolige and Bowman [99]. In their approach, a robot maintains a maximum number of recent exemplar views using a simple image closeness measure and a least-recently used cache algorithm. By contrast, our algorithm only adds additional exemplar images if the keyframe is visually salient and sufficiently distant in time to all others from the same neighborhood (lines 5 and 6 in Algorithm 2). This approach produces graphs whose sizes are closely bound to the size of the ship hull and maintains nodes with high utility for multi-session SLAM.

In practice, Algorithm 2 tends to preserves nodes that are both visually salient (for potential camera measurements), and useful for their locally planar structure (for piecewise planar or planar range measurements). Intuitively, this algorithm simply keeps the most recently-used views in a fixed-size neighborhood. An example of the how we maintain image diversity for the graphs used in the experimental results section is shown in Fig. 2.12.

This algorithm has various user-defined parameters. For the HAUV datasets, we chose a threshold of 1.5 m to decide if a node is spatially redundant (line 3 in Algorithm 2), and the maximum number of views per neighborhood as three (line 9 in Algorithm 2). We used the

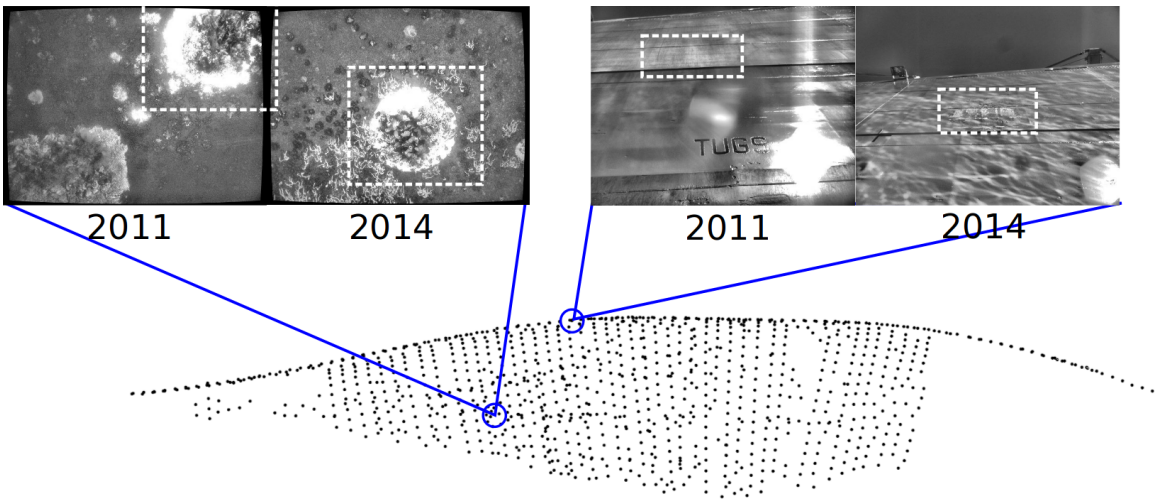


Figure 2.12: Illustration of exemplar views for the GLC *SS Curtiss* graphs. Two example view neighborhoods are circled in blue. Exemplar views within these neighborhoods, shown above, are noticeably different between 2011 and 2014. We manually outlined common regions in white for illustration. In the left example, different bio-foul patterns are growing on the same circular structure on the hull bottom. In the right example, SIFT features are detected on the pitting from the 2014 data, however, this pitting is not present in the 2011 data.

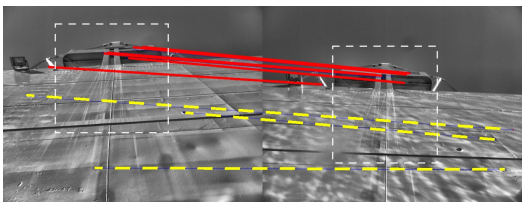
local saliency metric described by Kim and Eustice [96] as the method for determining if an image is visually salient.

2.5 Experimental Trials

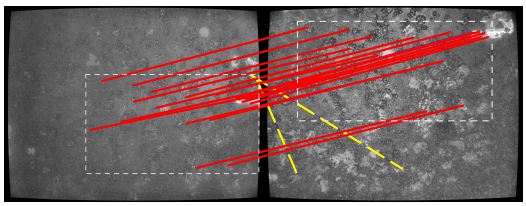
This section describes experimental trials with the HAUV performing automated inspections on the *USS Saratoga* and *SS Curtiss*, where we present an extensive multi-session SLAM graph for each vessel. The HAUV and vessels used for this evaluation are described in detail in Section 1.1.2. Further analysis on the proposed method is followed in terms of complexity, comparison to other vision-based loop-closure methods, and map accuracy in sparsification.

2.5.1 Experimental Setup

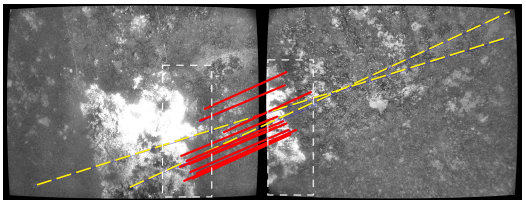
Table 2.1 gives an overview of how the onboard cameras were used in each dataset, along with the survey date. Note that our system does not require that the time sequence of surveys be causal. For instance, we can build a map of the ship hull from 2014 before co-registering surveys from 2011.



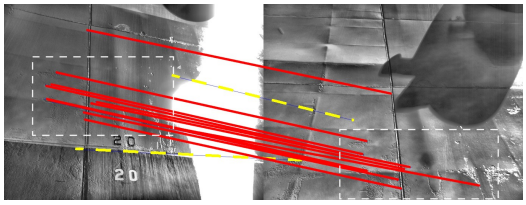
(a) Sunlight reflections from water



(b) Changes in illumination

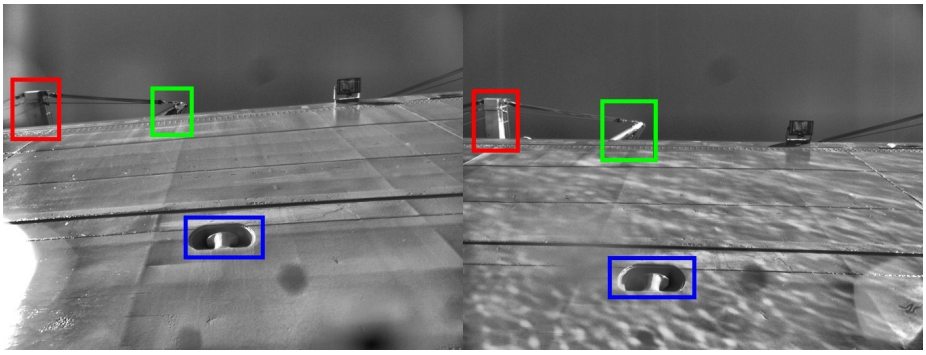


(c) Low overlap

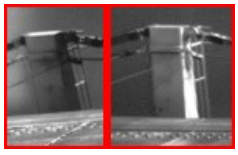


(d) Sunlight reflections and shadows on hull, 2011 to 2014

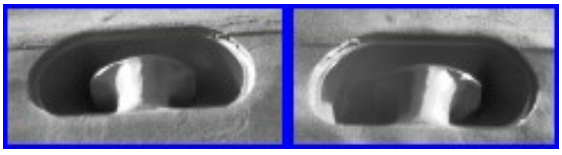
Figure 2.13: Notable examples of our localization system succeeding in challenging conditions. The scenarios in (b), (c), and (d) are especially difficult for a human to identify from a set of candidate keyframes. The feature correspondences shown are taken directly from RANSAC. Because purely epipolar-based feature matching across two views is prone to outliers (even with RANSAC), we manually annotated each figure with true inliers (solid red lines) and false inliers (dotted yellow lines). Common regions are denoted with dotted white boxes.



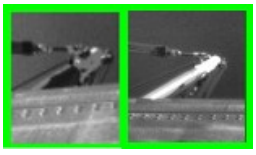
(a)



(b)



(c)



(d)

Figure 2.14: Example of a matching keyframe that our place recognition system fails to detect. We manually colored boxes around key areas to emphasize that the pixel intensity gradients in each patch are very different. The local features within these patches do not match because the SIFT descriptor is not invariant to strong changes in pixel gradient.

Table 2.1: Summary of HAUV field trials

Session ID	Date	Cameras Used	Survey Type	Trajectory Length (m)
		USS Saratoga		
1	May 21, 2013	Periscope	Surface	239
2	May 21, 2013	Periscope	Surface	136
3	May 21, 2013	Periscope + Underwater	Underwater + Surface	334
4	May 22, 2013	Periscope + Underwater	Underwater + Surface	490
5	May 22, 2013	Periscope + Underwater	Underwater + Surface	528
6	May 23, 2013	Periscope + Underwater	Underwater + Surface	493
7	May 23, 2013	Periscope + Underwater	Underwater + Surface	256
8	Aug. 08, 2013	Periscope + Underwater	Underwater + Surface	143
				2619 Total
		SS Curtiss		
1	Mar. 08, 2014	Periscope	Surface	341
2	Mar. 08, 2014	Periscope + Underwater	Underwater + Surface	367
3	Mar. 08, 2014	Periscope + Underwater	Underwater + Surface	768
4	Mar. 08, 2014	Periscope + Underwater	Underwater + Surface	719
5	Mar. 08, 2014	Periscope + Underwater	Underwater + Surface	497
6	Mar. 09, 2014	Periscope + Underwater	Underwater + Surface	587
7	Mar. 10, 2014	Periscope + Underwater	Underwater + Surface	449
8	Mar. 11, 2014	Periscope + Underwater	Underwater + Surface	934
9	Feb. 09, 2011	Periscope	Surface	148
10	Feb. 09, 2011	Periscope	Underwater + Surface	1125
11	Feb. 05, 2011	Underwater	Underwater	426
12	Feb. 05, 2011	Underwater	Underwater	1179
				7540 Total

The HAUV executes a survey using one or both of the underwater and periscope cameras. When both are used, the vehicle switches between the two as the vehicle approaches the surface, and as it submerges again for an additional trackline.

2.5.2 Multi-Session SLAM Results

We performed the proposed multi-session SLAM on two vessels, *USS Saratoga* and *SS Curtiss*. In particular, the *SS Curtiss* was surveyed over a three year period as shown in Table 2.1. While the current mission executes, it is localized to a GLC-sparsified graph using Algorithm 1. Examples of matches using this approach are shown in Fig. 2.13. Though the algorithm shows some resilience to lighting, sunlight, and shadows, this is not always

Algorithm 3 Match current keyframe to feasible keyframes from visual place-recognition system

1: **Input:** Set of all keyframes, K, in GLC-sparsified graph, current keyframe k, belief threshold τ, ship-specific vocabulary, V
2: $\mathrm{S} \leftarrow$ `FAB_MAPv2`$(k, \mathrm{K}, \mathrm{V}, \tau)$
3: **Output:** `FindBestMatch`(k, S) // Uses GPU

the case (see Fig. 2.14 for one such example). In total, eight missions on *USS Saratoga* and twelve missions on *SS Curtiss* are merged as in Fig. 2.15. These missions are accumulated into a common hull-relative reference frame, and sparsified after each session. The end result is a far more extensive map of each ship hull, as compared to the single session case.

2.5.3 Graph Complexity Over Time

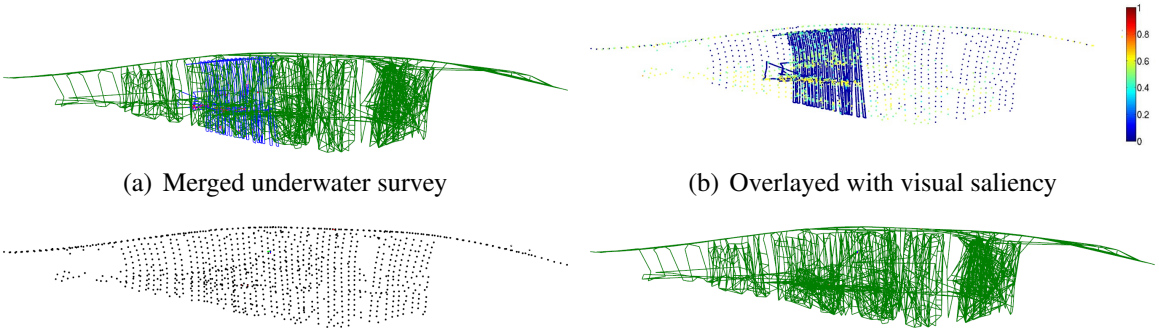
For our evaluation, we take the graph complexity to be the total number of variable nodes and factor nodes in the SLAM graph. The graph complexities over time using our framework for each vessel are shown in Fig. 2.16 over each successive session. By the end of the third session for each vessel, we lose the ability for real-time performance unless we apply GLC between sessions. Conversely, the use of our method for graph sparsification preserves real-time performance over the course of many more sessions.

2.5.4 Comparison to Bag-of-Words Place Recognition

To baseline the performance of our particle filter, we use an open-source implementation of FAB-MAP version 2.0 [60] as an appearance-only method for place recognition, which represents each keyframe as a visual BoW using a vocabulary that is learned offline. This algorithm is summarized in Algorithm 3, and can be used in place of Algorithm 1 for initial alignment. FAB-MAP provides the belief that the live image is taken from the same location as a place in a prior map. If a match is detected with significant probability, a threshold is used to determine if the SLAM front-end should accept the match. This threshold is typically set very high to avoid false-positives, but we use a very low threshold of 0.0001 for our application to ensure that FAB-MAP returns as many candidates as possible. These candidates are geometrically verified using RANSAC in the final step to reject outliers.

For FAB-MAP, we learn a separate SIFT vocabulary and CLT over the distribution of codewords for each vessel. In practice, this is impractical because it requires a training stage before multi-session SLAM can be attempted. Even so, based upon our experiments, FAB-MAP's performance is acceptable when matching images of the ship's above-water superstructure, like the ones shown in Fig. 2.13(a) or Fig. 2.17, top row. However, for the underwater images, FAB-MAP performs poorly, and we were not able to successfully

USS Curtiss



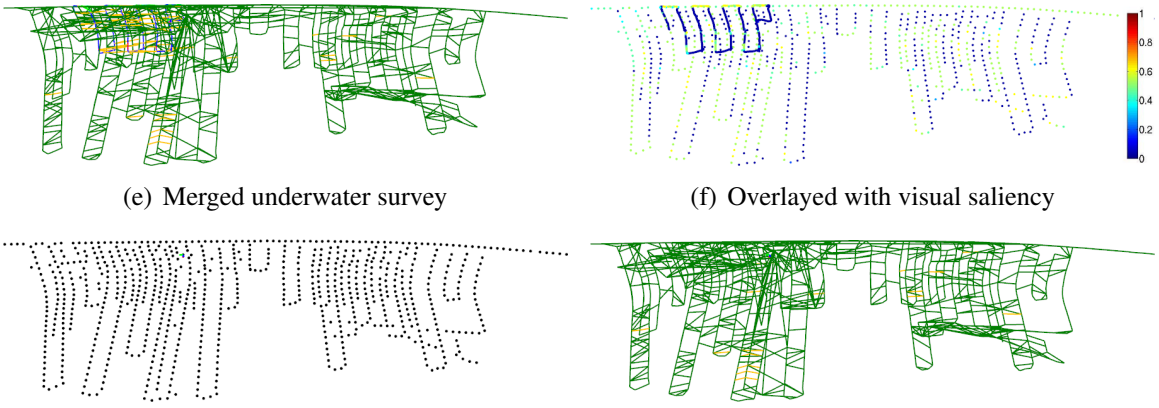
(a) Merged underwater survey

(b) Overlayed with visual saliency

(c) Preserved nodes after sparsification

(d) Final sparsified graph

USS Saratoga



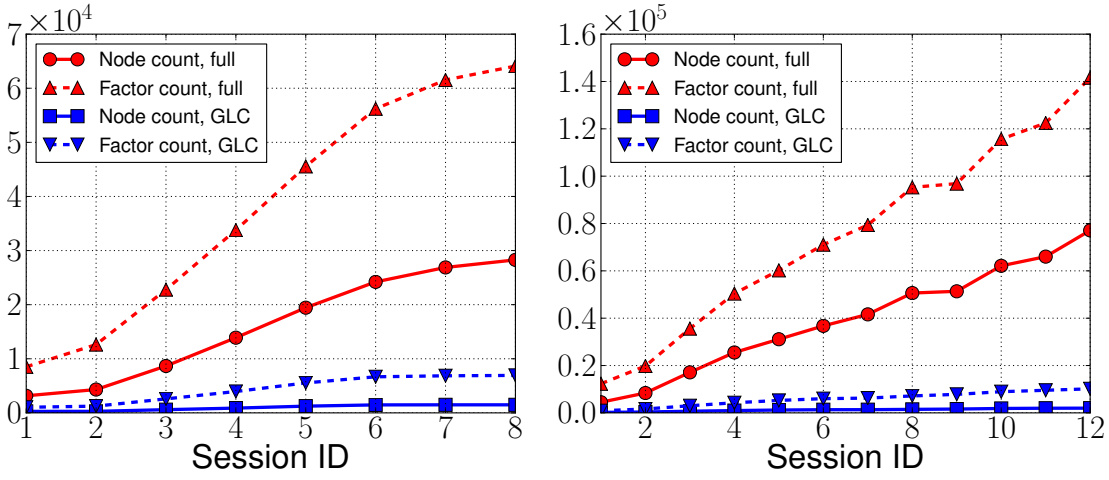
(e) Merged underwater survey

(f) Overlayed with visual saliency

(g) Preserved nodes after sparsification

(h) Final sparsified graph

Figure 2.15: Example of underwater surveys that are automatically aligned with a GLC-sparsified graph. The top and bottom halves show examples from the *SS Curtiss* and the *USS Saratoga*, respectively. In these figures, we disable rendering the planar patches and factors for the sake of visual clarity. In (b) and (f), we overlay the nodes with their local saliency score to show that low scores, in blue, will be marginalized while nodes with high scores, shown in yellow, will be kept so as to provide a more visually informative set of nodes for future localization (see Algorithm 2). Despite the increased spatial extent of the green GLCs, the graphs in (d) and (h) have significantly fewer edges than the graphs in (a) and (e), as also shown in Fig. 2.16.



(a) USS Saratoga (b) SS Curtiss

Figure 2.16: Graph complexity over multiple sessions for the *USS Saratoga*, in (a), and the *SS Curtiss*, in (b). Without managing the graph complexity, the number of nodes and factors grows unbounded with time. However, the size of the graph using the GLC framework is much more closely bounded to the size of the ship hull. For the *USS Saratoga*, sessions 1 through 7 occur within a week of each other, but session 8 occurs four months later. For the *SS Curtiss*, sessions 5 through 12 occur approximately three years after sessions 1 through 4.

identify an underwater loop-closure in any of our experiments, even with a very low loop-closure probability threshold. For underwater images, FAB-MAP consistently identifies the live underwater image as being taken from an unseen place. Two typical cases for the periscope and underwater cameras are shown visually in Fig. 2.17.

Where our method excels over FAB-MAP is the ability re-localize into a previous session while underwater. We consider three scenarios: (*i*) starting the survey with the robot at the surface, (*ii*) starting submerged, and (*iii*) a survey conducted entirely underwater. The results summarized in Table 2.2 are as follows: both methods are comparable when at the surface (first row), but FAB-MAP is unable to re-localize while starting underwater, and only when the robot surfaces can it find a good match (second row). For an entirely underwater survey (third row), FAB-MAP is unable to localize. As a whole, our system, which is tailored for the sensor payload of the HAUV, can more quickly and reliably localize to our long-term SLAM graphs.

Using either the particle filter or FAB-MAP, we strive to keep the recall of candidate images as high as possible. Reasonably low precision is not a primary concern because we can geometrically verify candidate matches relatively quickly. If we compute the precision and recall of the set S from particle filtering (Algorithm 1) and FAB-MAP (Algorithm 3), we find that the particle filter produces a set of candidate images with a lower average precision

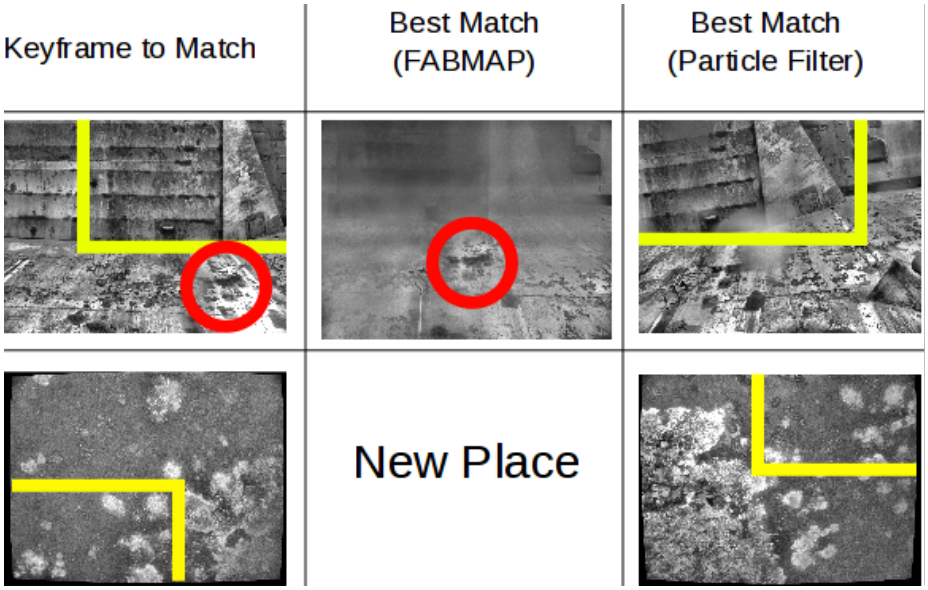


Figure 2.17: Comparison to FAB-MAP. Two representative attempts at aligning the current survey into past sessions using FAB-MAP and our particle filter. Corresponding image regions, where identified, are highlighted. We allow FAB-MAP to return as many candidates as possible by setting the loop-closure probability threshold low. By contrast, the number of candidate matches identified by our particle filter depends on how discriminative the planar measurements are. Because the image matching algorithm is done on the GPU, each candidate match takes roughly 70 ms when using a consumer-grade GPU. FAB-MAP performs adequately for surveys consisting of periscope images (top row) but fails using underwater images (bottom row), where it consistently assigns the "new place" label.

of 3.0%, but high average recall of 100.0%. FAB-MAP, on the other hand, produces a higher average precision of 12.1%, but a much lower average recall of only 27.7%. In other words, the particle filter always contains the correct place within the candidate set, but requires geometrically verifying more keyframes. With FAB-MAP, the candidate set is smaller and therefore faster to process, however, localization will often fail because a true match is not contained in the set of candidate images.

2.5.5 Comparison to Full Graph

To assess the accuracy of the GLC-based marginalization tool, we tracked the graph produced by the repeated application of sparse-approximate GLC after each session is completed. This graph was directly compared to a synthetically-created full graph, with no marginalized nodes, that contains the exact set of measurements that the HAUV accumulated over the course of each session. For the *SS Curtiss*, we show the full graph over all twelve sessions reported in this chapter in Fig. 2.18.

Table 2.2: Time to localize to past GLC graph

	Time until alignment (sec)	
Session	FAB-MAP	Particle Filter Search
USS Saratoga 2013 Session 7, starting from surface	7.1	5.2
USS Saratoga 2013 Session 7, starting underwater	143.2	20.6
SS Curtiss 2011 Session 12, underwater-only	N/A	15.3

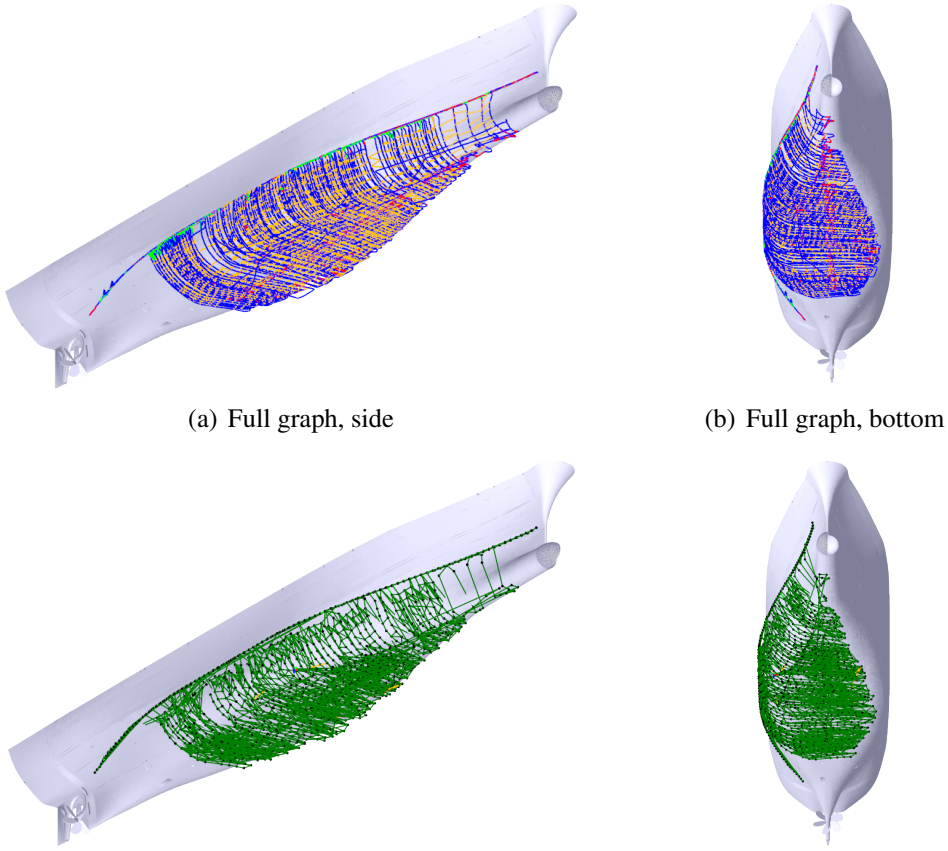
These comparisons are shown in Fig. 2.19 for the *USS Saratoga* and in Fig. 2.20 for the *SS Curtiss*. Each measure is computed only for the vehicle pose nodes for the sake of clarity. For each session, we compute the following: (*i*) marginal Kullback-Leibler Divergence (KLD) over each node (a measure of the similarity between two probability distributions); (*ii*) absolute positional distance between each corresponding node; (*iii*) absolute attitude difference between each node, defined as the $l2$-norm of the difference of the poses' Euler angles; and (*iv*) ratio of the determinants of marginal covariances of each node. For the KLD calculation, we use the well-known closed-form expression for KLD between two Gaussian distributions [101].

As a whole, the accuracy of our system is quite comparable to the full graph, showing typical errors well within a meter despite a total path length of multiple kilometers. These errors are absolute, however, and they do not take into account the strong correlations between neighborhoods of nodes. In addition, as shown in Fig. 2.21, the marginal ellipses of the approximated poses are consistent with the full graph. Though repeated sparsification may eventually result in inconsistent estimates, recent works include Carlevaris-Bianco and Eustice [26, 27], which guarantee that the GLC approximation is conservative to counter these inconsistencies. Seeing how these methods affect the success rate of data association is an interesting topic that remains to be explored. For the long-term datasets covered here, inconsistent approximations were not an issue; we were able to successfully establish camera measurements across multiple sessions throughout the field trials.

2.5.6 Comparison to CAD Model with Planar Constraints

We provide some qualitative results in Fig. 2.22 to show that the effectiveness of the method depends strongly on the use of planar constraints. In this figure, we disabled both the piecewise-planar factor potential from (2.3) and the planar range factors from (2.4). To keep the comparison fair, we provided the same particle filter localizations used in Fig. 2.18.

Additionally, we used the ground truth CAD model to assess the quality of the SLAM



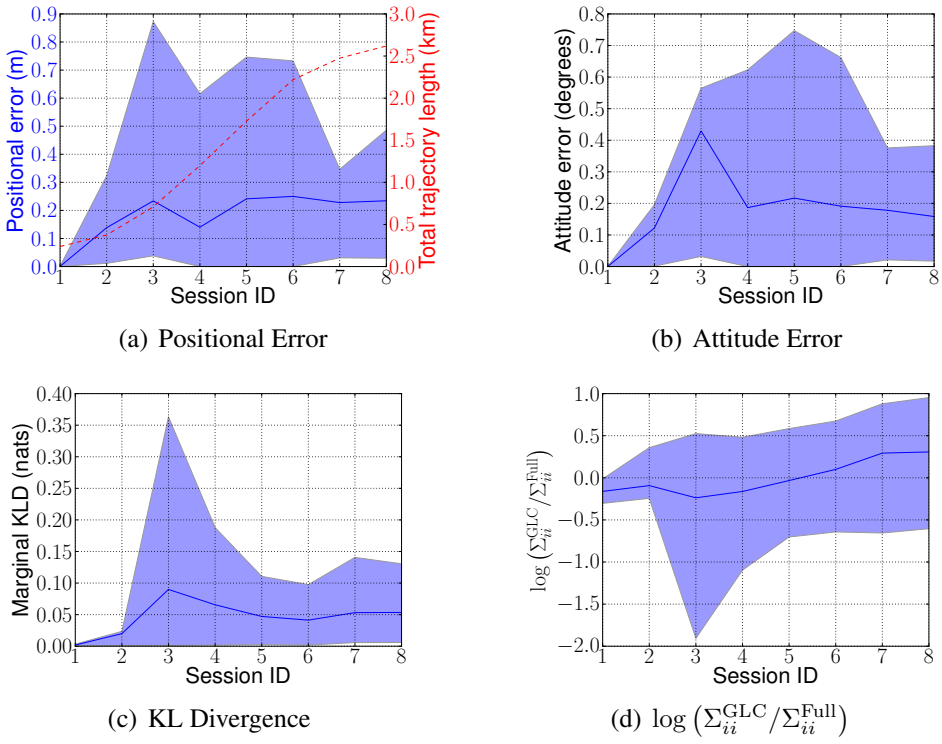
(a) Full graph, side

(b) Full graph, bottom

(c) GLC graph, side

(d) GLC graph, bottom

Figure 2.18: Comparison to full graph. The full, unsparsified graph of the *SS Curtiss* after twelve automatically-aligned sessions that spans from February, 2011 to March, 2014. The CAD model is aligned to provide visual clarity and size reference — it is not used in the SLAM system. The non-sparsified factor graphs in (a) and (b) consist of odometry factors (blue), camera links (red), piecewise-planar factors from (2.3) (yellow), and planar range factors from (2.5) (lime green). The sparsified graphs in (c) and (d) consist mostly of GLC factors (green). These graphs serve as the comparison for the results in Fig. 2.20 for $\text{SessionID} = 12$.



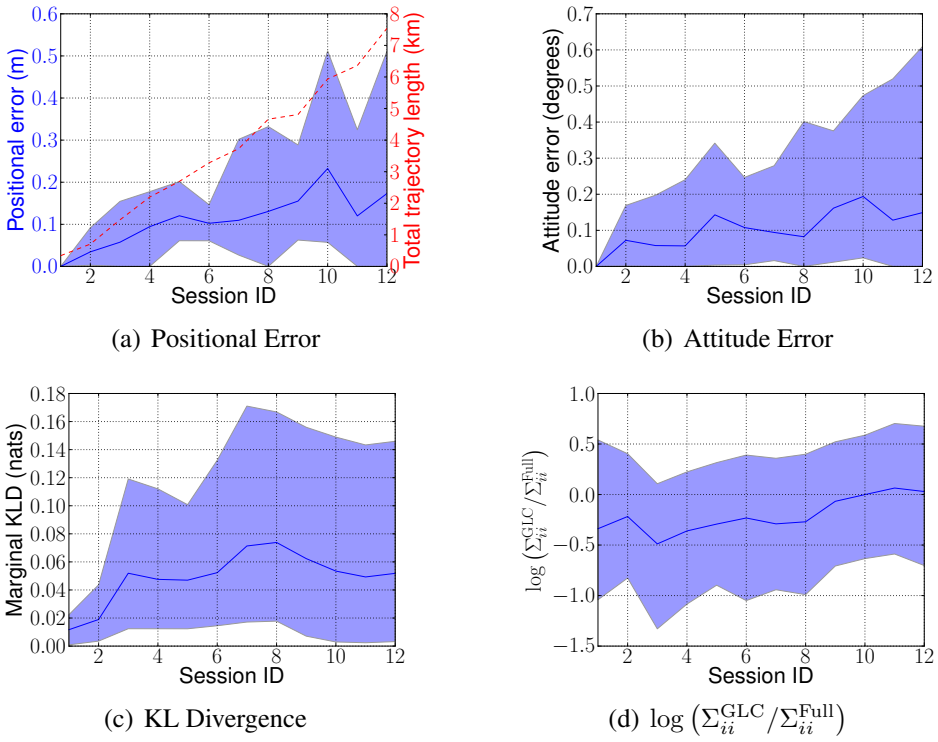
(a) Positional Error

(b) Attitude Error

(c) KL Divergence

(d) $\log\left(\Sigma_{ii}^{\mathrm{GLC}}/\Sigma_{ii}^{\mathrm{Full}}\right)$

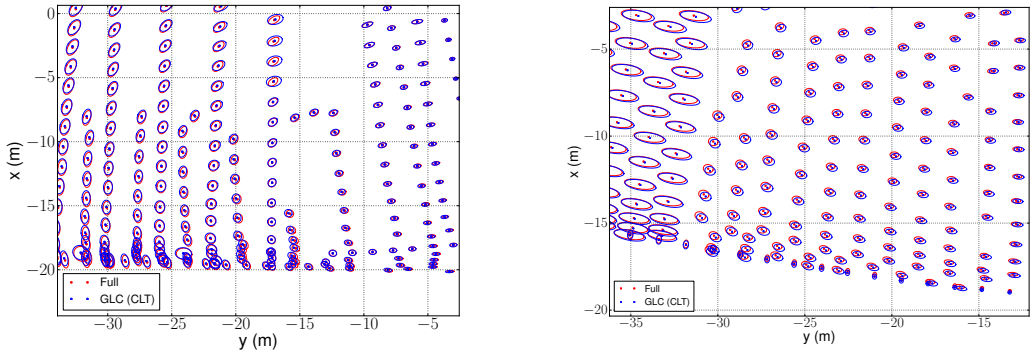
Figure 2.19: GLC results for the *USS Saratoga*. Positional error and trajectory length (a), attitude error (b), average KLD (c), and log-ratio of marginal covariances (d) as a timeseries for the 2014 *USS Saratoga* datasets. The average value is shown as solid blue line. Percentile bounds are shown in the shaded region, from 5% to 95%.



(a) Positional Error (b) Attitude Error

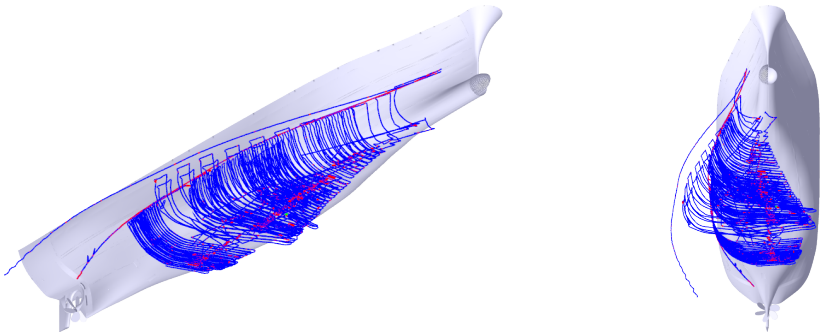
(c) KL Divergence (d) $\log\left(\Sigma_{ii}^{\mathrm{GLC}}/\Sigma_{ii}^{\mathrm{Full}}\right)$

Figure 2.20: GLC results for the *USS Curtiss*. Comparisons to the full graph for each session for the *SS Curtiss* datasets. We show positional error (a), attitude error (b), average KLD (c), and log-ratio of marginal covariances (d) as a timeseries. The average value is shown as solid blue line. Percentile bounds are shown in the shaded region, from 5% to 95%.



(a) USS Saratoga (b) SS Curtiss

Figure 2.21: Visualizations of 3-σ positional covariances for the full graph are shown in red, and the corresponding ellipses from the GLC-sparsified graph are shown in blue. Our system marginalizes approximately 95% of nodes from each session so that the system can perform in real-time, yet the probabilistic deviation from the full graph is small. The results for the *USS Saratoga* and *SS Curtiss* are shown in (a) and (b), respectively.



(a) Full graph, side

(b) Full graph, bottom

Figure 2.22: Results with planar information omitted. The quality of the SLAM estimate suffers significantly if the planar-based constraints from Section 2.3.4.2 and Section 2.3.4.3 are left out of the factor graph. These constraints are encoded as yellow and green lines in Fig. 2.18(a). The CAD model is once again provided for visual reference, and to show that there are obvious discrepancies between it and the SLAM estimate. These discrepancies are quantified in Fig. 2.23.

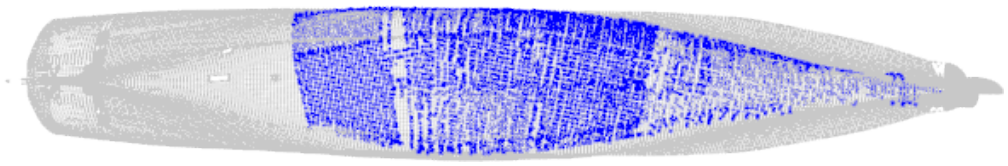
trajectory, with and without planar constraints. We converted the DVL range returns into a global-frame point cloud using the estimated SLAM poses and down-sampled them with a voxel grid filter. Then, we rigidly aligned these points to the CAD mesh with generalized iterative closest point (GICP) [153]. Finally, for each point in the DVL point cloud we found the nearest vertex in the CAD mesh and computed the Euclidean distance.

This method serves as a sufficient proxy for the structural consistency of the SLAM estimate as compared to ground truth. The results of this comparison are shown in Fig. 2.23. For the SLAM estimate with no planar constraints, the error distribution had a mean of $1.31\ \mathrm{m}$ and standard deviation of $1.38\ \mathrm{m}$. Furthermore, 20% of the DVL points had an error of more than $1.5\ \mathrm{m}$.

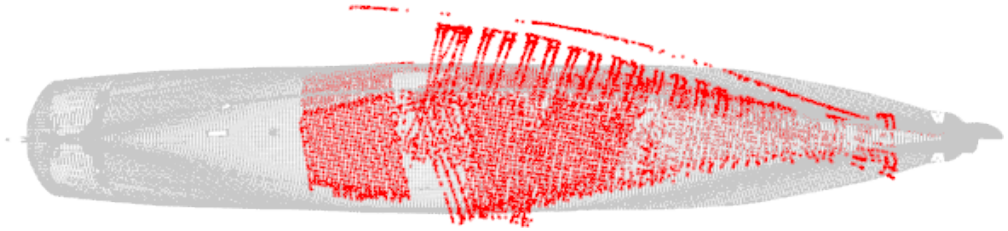
Conversely, the use of planar constraints brought the mean and standard deviation to $0.45\ \mathrm{m}$ and $0.19\ \mathrm{m}$, respectfully. In addition, there were no points that had an error of more than $1.5\ \mathrm{m}$. With these results, we conclude that camera constraints alone are not sufficient for long-term hull inspection, and we have shown that the results are greatly improved if the ship hull surface itself is included in the SLAM pipeline.

2.6 Conclusion

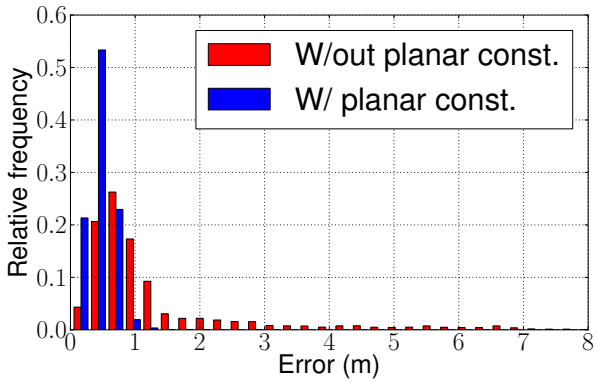
We provided an overview of how to adapt our visual SLAM algorithm for long-term use on large ship hulls. We proposed planar observation models that do not require overlap between



(a) DVL alignment with planar constraints



(b) DVL alignment without planar constraints



(c) Error histogram for (a) and (b)

Figure 2.23: Comparison to ground-truth CAD model. For the *SS Curtiss*, we were able to quantify the error with respect to the ground truth CAD model by registering the DVL returns (red and blue point clouds) to the CAD mesh vertices (gray point cloud). The DVL point cloud in (a) was derived from the SLAM estimate in Fig. 2.18 (with planar constraints), while the point cloud in (b) was derived from Fig. 2.22 (without planar constraints). The Euclidean error between corresponding vertices in the CAD model mesh is summarized in (c), where the planar constraints provide a much tighter distribution of error.

planes, and applied this model to factor graphs (for SLAM) and particle filters (for global localization). We have shown that it is well-suited for modeling the smooth hull form of large ships.

We use the GLC framework to remove redundant or unneeded nodes from a factor graph. Doing so involves supporting a node reparameterization (root-shift) operation to avoid unnecessary error induced by world-frame linearization. Furthermore, we described an efficient particle filtering algorithm that globally localizes to past sessions using imagery from the current session.

We showed results from our localization algorithm automatically aligning SLAM sessions separated in time by days, months, and years. Once sessions were aligned to a past graph, the result was sparsified and the process was repeated. Using simple sparsification criteria, we show that the complexity of our factor graphs remain more closely bounded to the ship hull area, rather than growing unbounded with time. Furthermore, despite the repeated applications of graph sparsification with GLC, the errors between the sparsified and non-sparsified graphs are reasonably small.

CHAPTER 3

3D Photomosaicing using 2D Imaging Sonar and a Doppler Velocity Log

3.1 Introduction

Several tasks in ocean exploration require the visualization of a large set of images to understand underwater phenomena at a broad spatial scale. In recent years, techniques have been developed that allow for the reconstruction of visually rich three-dimensional (3D) mosaics of the seafloor from thousands of optical images [83, 125]. Though these methods have been successfully applied in large-scale marine environments, underwater optical cameras have several limitations. For example, turbid water makes identification of visual features difficult, light attenuates much more in water than in air, and underwater cameras typically must provide their own light source. Acoustics are the preferred sensor modality for underwater robotics because they overcome several of those limitations. However, there are several challenges with this approach that need to be solved. These challenges include: (*i*) sonar technology is typically more expensive than optical cameras, (*ii*) the sensor's vantage point strongly affects signal intensity, and (*iii*) high field of view (FOV) imaging sonars have non-standard geometrical properties of their projection from 3D to 2D. This chapter will explore the third challenge: we present a method to create a textured 3D mosaic from an imaging sonar, coupled with a typical sensor payload on a small autonomous underwater vehicle (AUV). An overview of our method is illustrated in Fig. 3.1.

3.1.1 Related Work

In terms of attenuation, sonar is by far the preferred sensor for surveying the ocean seafloor [8], and imaging sonars are a more prevalent alternative to underwater cameras. Recent work related to imaging sonars has focused on the registration problem, where two overlapping images are geometrically matched. Solutions to this problem use either spectral

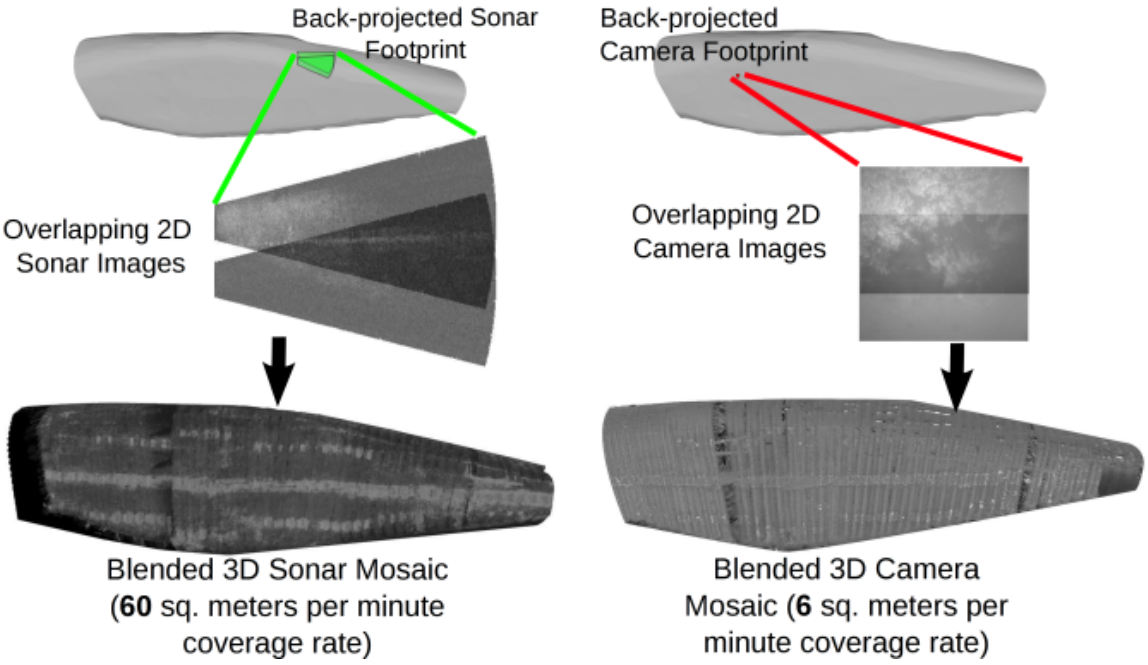


Figure 3.1: Coverage rate comparison between camera and sonar. In the context of ship hull inspection, one key benefit of our sonar-based mosaicing pipeline (left) over our camera-based mosaicing pipeline (right) is coverage rate. In this example, a 3D surface, shown above in gray, is either reconstructed from a typical underwater SLAM sensor payload or given as a prior CAD model. Two overlapping sonar images back-project onto the 3D surface in the green region, which has a much larger footprint than an underwater camera's, shown in red. Mesh faces within the overlapping region are blended to avoid the presence of seams in the final mosaic.

methods [22, 77], or feature-based methods [7]. In either case, these techniques have several applications, such as underwater simultaneous localization and mapping (SLAM) [82, 144] and photomosaicing [75, 76]. However, previous work produces strictly 2D mosaics or 3-degree-of-freedom (DOF) motion estimates (relative x, y, and heading). Recently, Negahdaripour lifted feature tracking and motion estimation to 3D, but did not explore the applications to 3D mosaicing [120]. Their method additionally improves the results of full 3D pose estimation by zeroing higher dimensional pose parameters like pitch and roll (as in the 3-DOF case).

Despite their limitations in marine environments, optical cameras have become a popular modality for the creation of underwater mosaics. Historically, most applications have used 2D mosaics for vision-aided navigation [61]. Johnson-Roberson et al. [83] argue that the rugged terrain of the seafloor necessitates projecting the imagery onto 3D models to properly account for the geometry, rather than force-fitting a plane to a non-planar environment.

Similar reasoning suggests that 3D mosaicing methods are a better choice for building mosaics of large man-made structures, which include dams, harbors, pipelines, and ship hulls [145]. We are particularly interested in autonomous ship hull inspection, and we believe that 3D mosaicing will be of great benefit for improved efficiency in maintenance, assessment, and security. 3D mosaics would help robots' human supervisors to easily visualize the data, assist in cooperation between robots, or aid in automated tracking of structural changes or anomalies over time. Novel techniques for the generation of acoustic 3D mosaics is therefore the focus of this chapter.

3.1.2 Outline

This chapter is organized as follows. In Section 3.2 we describe our 3D mosaicing approach, where a surface mesh is reconstructed from SLAM-corrected poses, and an empirical reprojection operation is used to assign images to triangles for texturing. In Section 3.3 we describe our experimental setup and we offer several evaluations of the method's performance. Section 3.4 summarizes and offers concluding remarks.

3.2 Approach

3.2.1 Correcting Navigation Drift with SLAM

A prerequisite of our 3D mosaicing pipeline is that the vehicle's trajectory is already corrected from SLAM. There are several methods to accomplish this. Our past work

primarily focused on camera-based techniques [96], but our recent work has shifted some attention on relaxing the reliance of an underwater camera [135]. As discussed in Chapter 2, by estimating surface normals from Doppler velocity log (DVL) range returns, we can constrain the normals of nearby planar patches and produce more self-consistent maps. In addition, this approach has tremendous benefits for performing long-term SLAM since it can be effectively combined with recent developments in graph sparsification techniques [28, 136].

This method, which we will refer to as "piecewise-planar SLAM" in this chapter, is of particular relevance to sonar mosaicing because it can be generalized to other AUVs that do not have a camera. Furthermore, the geometrical information provided by planes is beneficial to other applications besides ship hull inspection, such as underwater trenches or dam inspection.

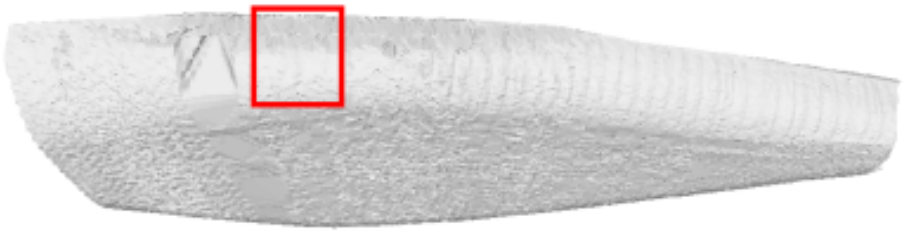
3.2.2 Surface Reconstruction via DVL Ranges

The first step in creating a 3D mosaic is to estimate a surface reconstruction. Traditionally, surface reconstruction either relied on a stereo camera to merge individual keyframe meshes into a large 3D model [84], or from camera-derived point sets that contain large amounts of features [23]. Since we are interested in mosaics derived from imaging sonar, our work instead reconstructs the surface using the DVL range returns. Doing so we can create suitable models of ship hulls and relax the need for underwater cameras.

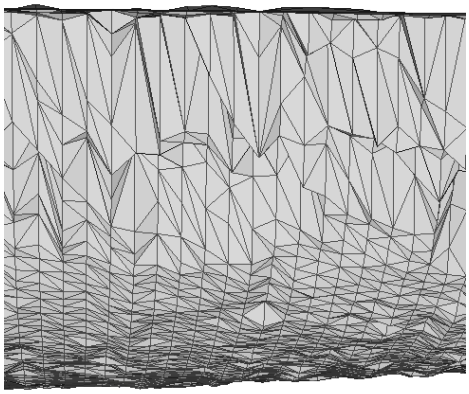
To this end we have developed two approaches: one using linear interpolation and the other using Gaussian process (GP) regression. Each has certain advantages over the other, which will be discussed in Section 3.3.3.

3.2.2.1 Surface Estimation using Linear Interpolation

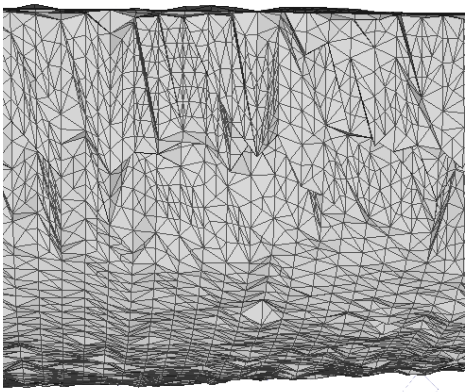
For this approach, we begin by converting range returns from all DVL poses into a point cloud in the global frame. We automatically reject outliers using the method described by Rusu et al. [150], which involves rejecting points that fall outside one standard deviation from a neighborhood consisting of 30 neighboring points. Next, we use the inlier set to linearly interpolate the z-coordinates over an evenly-spaced grid in the global frame's x, y plane. We apply Delaunay triangulation to these points, producing a height-map. A well-known limitation with this technique is that it exposes stretched triangles in the near-vertical portions of the surface that will not fit in a single camera or sonar image. To mitigate this effect, we recursively inscribe triangles within triangles until all edge lengths in the mesh are below a threshold (0.1 m for the results shown in Section 3.3). This is sometimes called



(a)



(b)



(c)

Figure 3.2: Surface reconstruction using linear interpolation. A simple midpoint resampling method prevents pixel stretch for the Delaunay reconstruction of the mesh shown in (a). In (b), there are triangles that are ill-suited for texture blending. By recursively splitting the edge of these triangles at their midpoint, shown in (c), these triangles are divided into smaller faces while preserving the overall shape of the mesh.

the "midpoint method" for resampling triangular meshes.

An overview of this linear interpolation-based reconstruction is shown in Fig. 3.2. The result is more jagged than the true ship hull, suggesting areas for improvement. In addition, due to the sparsity of the DVL returns, a simple linear interpolation may not generalize to some non-ship hull applications. However, more advanced techniques exist that can interpolate DVL returns in more varied structure, like underwater terrain [9].

3.2.2.2 Surface Estimation using Gaussian Process Regression

A technique known as GP regression is a popular method for estimating smooth curves from finite observations. The generally smooth profile of large ship hulls suggests favorable performance using GP regression over the linear interpolation method discussed above.

Similar to the previous section, we estimate z depth values of the ship over a xy-grid. However, rather than linearly interpolating between samples we consider data from a wide range of DVL returns using GP regression. First, we assume the depth of the ship hull is generated with Gaussian white noise around the continuous function f:

$$z_i = f(\mathbf{x}_i) + \epsilon_i,$$

where $\epsilon_i \sim \mathcal{N}(0, \beta^{-1})$ represents white Gaussian noise with variance β^{-1} and $\mathbf{x}_i = [x_i, y_i]^\top$ is a point in the xy-plane. Therefore, the likelihood over all depths is given by

$$p(\mathbf{z}|\mathbf{x}_1 \dots \mathbf{x}_N) = \mathcal{N}(\mathbf{f}, \beta^{-1}\mathrm{I}).$$

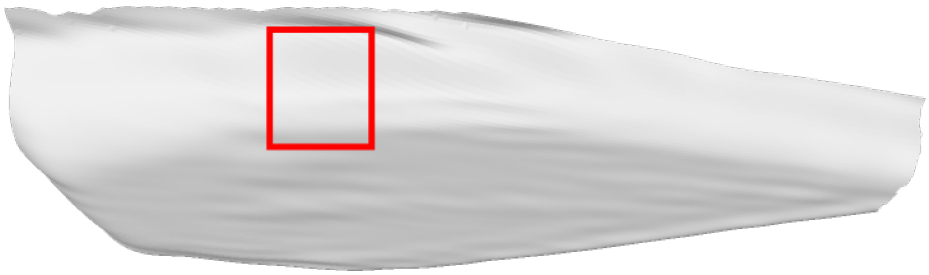
The goal of regression is to estimate $\mathbf{f} = [f_1(\mathbf{x}_i), \dots, f_N(\mathbf{x}_N)]^\top$. Since $\mathbf{f}$ can be thought of as a parameter with infinite dimension, this is sometimes called *non-parametric Bayesian estimation* [133].

By assuming a prior on $p(\mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathrm{K})$, where K is the so-called *Grammian* matrix, we can compute the marginal likelihood of $\mathbf{z}$ using properties of the Gaussian distribution (shown in Appendix E.1):

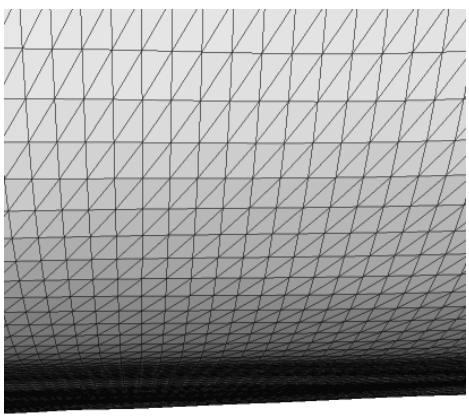
$$\begin{aligned} p(\mathbf{z}) &= \int p(\mathbf{z}|\mathbf{f})p(\mathbf{f})d\mathbf{f} \\ &= \mathcal{N}(\mathbf{0}, \mathrm{K} + \beta^{-1}\mathrm{I}). \end{aligned} \tag{3.1}$$

Each element of the Grammian matrix K_{ij} is computed from a kernel function with respect to input variables $\mathbf{x}_i$ and $\mathbf{x}_j$.

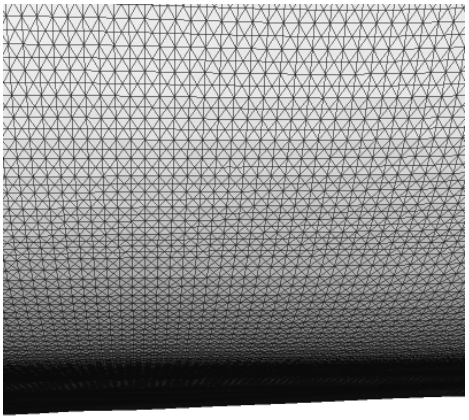
Consider N training input and output pairs $(\mathbf{x}_i, y_i)$, where $i = 1, \dots, N$, and a test input



(a)



(b)



(c)

Figure 3.3: Surface surface reconstruction using GP regression. Similar to Fig. 3.2, we use the midpoint method to keep each triangle below a set size, however the GP offers a much smoother reconstruction. To keep this method computational tractable, we use a sparse kernel when building the Grammian matrix over training data.

$\mathbf{x}_t$. The joint training and test marginal likelihood is given by

$$p(\mathbf{z}, z_t) = \mathcal{N}(\mathbf{0}, \mathrm{K}_{N+1} + \beta^{-1}\mathrm{I}), \qquad \mathrm{K}_{N+1} = \begin{bmatrix} \mathrm{K}_N & \mathbf{k}_{Nt} \\ \mathbf{k}_{Nt}^\top & k_t \end{bmatrix}.$$

Using the property from Appendix E.2, we find that the conditional probability distribution $p(z_t|\mathbf{z}) = \mathcal{N}(\mu_t, \sigma_t^2)$ is given by

$$\mu_t = \mathbf{k}_{Nt}^\top \left(\mathrm{K}_N + \beta^{-1}\mathrm{I}\right)^{-1} \mathbf{z} \tag{3.2}$$

$$\sigma_t^2 = k_t - \mathbf{k}_{Nt}^\top \left(\mathrm{K}_N + \beta^{-1}\mathrm{I}\right)^{-1} \mathbf{k}_{Nt} + \beta^{-1}. \tag{3.3}$$

For a large dataset, this can be computationally intractable because the Grammian matrix K_N may be large and dense. Therefore, we use the kernel developed by Melkumyan and Ramos [112], which preserves sparsity in the Grammian while enabling exact GP inference.

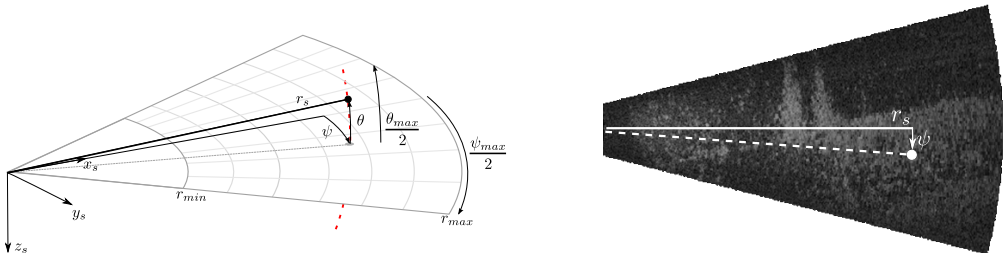


Figure 3.4: Illustration of the zero-degree plane of the DIDSON. These illustrations show how a 3D point is projected into the DIDSON frame from its spherical coordinates (). All points not contained in the volume bounded by $r_{min}, r_{max}, \theta_{max}$, and ψ_{max} are not visible to the sonar.

Since the $\mathbf{x}_i$ training samples above are multivariate, we use the multivariate version of their kernel:

$$k(\mathbf{x}, \mathbf{x}') = \begin{cases} \sigma_0 \left[\frac{2+\cos(2\pi r)}{3} (1-r) + \frac{\sin(2\pi r)}{2\pi} \right], & r < 1 \\ 0, & r \geq 1 \end{cases}$$

$$r = \sqrt{(\mathbf{x} - \mathbf{x}') \, \Omega \, (\mathbf{x} - \mathbf{x}')}$$

$$\Omega = \text{diag} \left(\frac{1}{l_x^2}, \frac{1}{l_y^2} \right),$$

where σ_0, l_x, l_y are hyperparameters of the sparse kernel. We tune these manually to reflect the geometry of typical ship hulls. Intuitively, l_x and l_y are the characteristic lengths of the kernel for the x and y axis, respectively. Since the ship hull is aligned more-or-less parallel to the y-axis, we choose $l_y > l_x$. For our experiments, we choose $l_x = 3.0$ m and $l_y = 15.0$ m.

Finally, once the regression estimates the z depth of the hull over the entire xy-grid, we convert those points to a triangular mesh using Delaunay triangulation and subdivide the triangles to be no more than 0.1 m in length using the midpoint method.

3.2.3 Projecting Mesh Vertices into Image Coordinates

When the surface is reconstructed, we must then project each vertex into the sonar images where the vertex is visible. Unlike a calibrated projective camera, which has a simple closed-form expression for the projection from 3D to 2D pixel coordinates, we empirically compute this relationship using several sensor-specific parameters.

The imaging sonar has a discrete number of beams, N_b, each containing a discrete number of ranges, N_r, as illustrated in Fig. 3.4. Let r_{min} and r_{max} be the minimum and

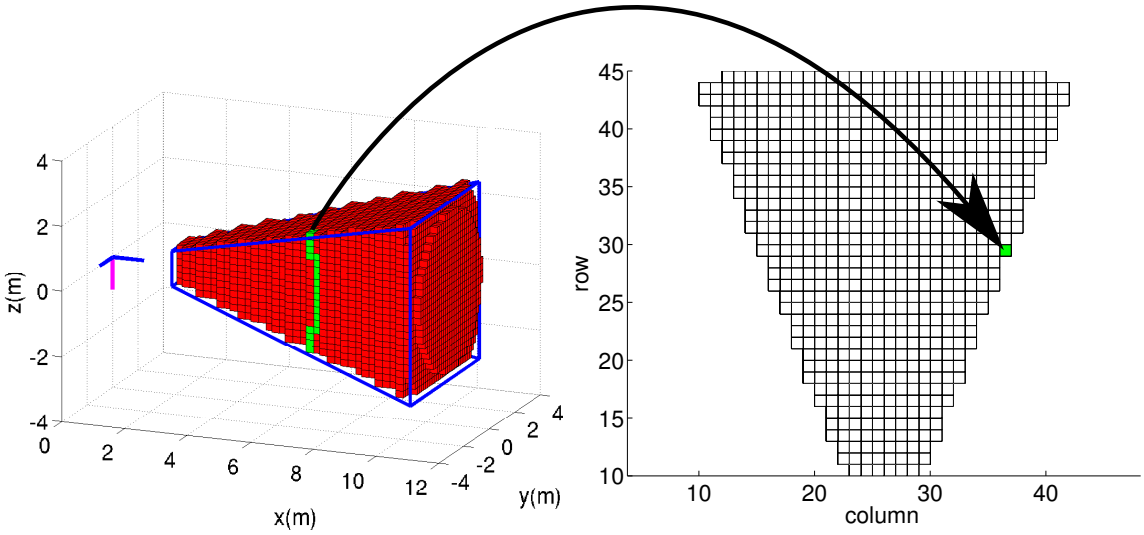


Figure 3.5: Projection of 3D coordinates to 2D pixels in a sonar image. We discretize the Cartesian volume contained in the sonar's field-of-view frustum as voxels. These voxels are mapped into a corresponding pixel in the 2D image with a simple look-up table. This lookup table is computed in the sonar's frame, so it only must be stored once. In this example, all 3D points contained in the green arc of voxels (left) map to the single green pixel in the sonar's 8-bit image (right). We over-pixelated this diagram for the sake of clarity.

maximum ranges that are observable by the sonar. For a given u, v pixel coordinate in the sonar image (such as the synthetic one in Fig. 3.5, right), the corresponding sensor-frame Cartesian coordinates, x_s, y_s, are given by:

$$x_s = \frac{u - \frac{w}{2}}{\gamma}$$
$$y_s = r_{max} - \frac{v}{\gamma},$$

where w is the sonar image width, h is the height, u is the column index of the pixel, v is the row index of the pixel, and

$$\gamma = \frac{w}{2r_{max}\sin(\psi_{max}/2)}$$

is a constant. Here, ψ_{max} and θ_{max} are the maximum field of view along the sonar's z and y axes, respectively.

We convert these sensor-frame coordinates into range and bearing values in the sensor

frame (assuming that $z = 0$) as follows:

$$r_s = \sqrt{x_s^2 + y_s^2}$$
$$\psi_s = \text{atan2}\left(x_s, y_s\right)$$

Finally, we assign these continuous range and bearings to one of the sensor's discrete range bins and beam number:

$$n_b = \frac{\left(r_s - r_{min}\right)\left(N_r - 1\right)}{r_{max} - r_{min}}$$
$$n_r = M_4\left(\psi_s\right)^\top \mathbf{a},$$

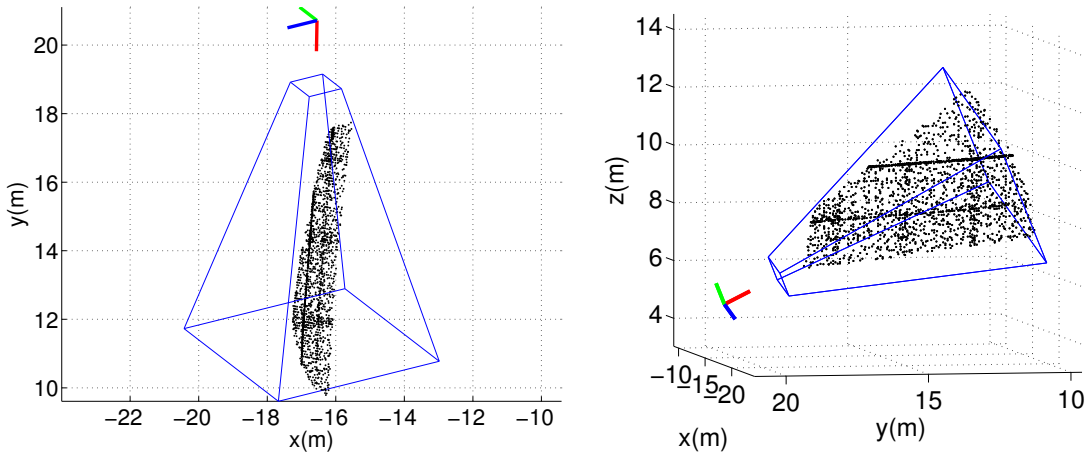
where n_b is the beam number, n_r is the range bin number, $M_4(\psi) = [1, \psi, \psi^2, \psi^3]$ is a fourth-degree vector of monomial bases and $\mathbf{a}$ is a vector of sensor-specific coefficients provided by the manufacturer. To compute the inverse mapping from n_r and n_b to 3D Cartesian coordinates, we simply discretize the volume inside the sonar's frustum into voxels, apply the above set of equations, and store the inverse map. For a given x_s, y_s coordinate, the voxels where $z \neq 0$ project to the same pixel as the corresponding voxel where $z = 0$. This is illustrated in Fig. 3.5.

3.2.4 Blending Step

Our image blending pipeline is based on the previous work by Johnson-Roberson et al. [83] for creating large-scale 3D mosaics of seafloor environments using a stereo camera. The approach works by assigning a fixed number of sonar images to every face in the mesh (for the experimental results, shown in Section 3.3, we use a maximum of four images per face). For each mesh face, we compute the set of sonar poses such that the face is visible. Fig. 3.6 shows an example of which face vertices are visible. From this set, we pick the four best views of the face using a user-defined proxy for image quality. For underwater cameras, popular heuristics include choosing the smallest distance of the projected face to the center, or choosing the most orthogonal camera poses to the face's surface normal.

Choosing the correct heuristic for an imaging sonar is a subjective matter, and we have found from our experimental results that picking images where the face projection is closest to the pixel coordinates $u_{\text{best}} = w/2$ and $v_{\text{best}} = 3h/4$ works well. For the ship hull mosaics presented in Section 3.3, we have found this typically corresponds to a face normal of approximately 7 degrees from orthogonal to the sonar frame's x-axis.

Once we determine the four best image patches, we weight each pixel contained in the



(a) Top-down view

(b) Side view

Figure 3.6: Example subset of mesh vertices within the sonar's FOV. The frustum, computed from the method in Section 3.2.3, is shown in blue. Vertices from the surface reconstruction that lie within the frustum are visible to the sonar, and will be included in the blending step. These vertices are shown as black dots.

i^{th} mesh face by the distance r to the pixel coordinate $(u_{\text{best}}, v_{\text{best}})$. This weight is determined from the expression

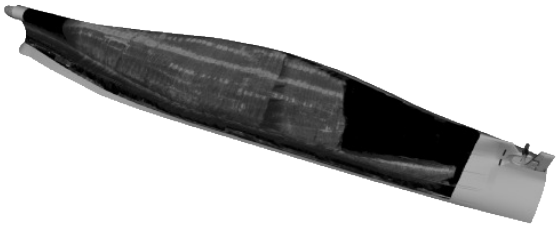
$$r = \sqrt{(u_{\text{best}} - u)^2 + (v_{\text{best}} - v)^2}$$
$$B_k^i(r) = \frac{e^{-k\frac{r}{R}}}{1 + e^{-2k\frac{r}{R}}},$$

where R is a reference distance (typically the maximum distance to be considered). This process is determined for three different resolution bands, where $k = 5, 10, 50$ is an appropriately chosen coefficient for each resolution band. Larger k indicates a sharper drop-off as r increases, and is useful for bands with higher resolution. During the actual pixel blending computation, the four blending weights are normalized so that they sum to one [83].

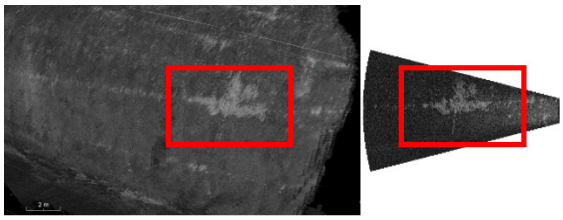
3.3 Experimental Evaluation

3.3.1 Robot Platform

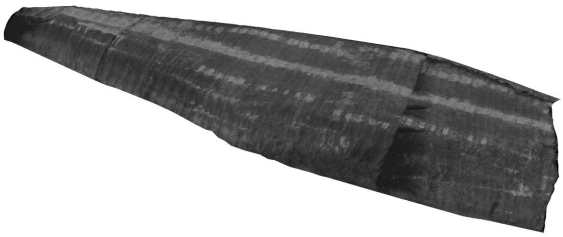
We use data collected from the hovering autonomous underwater vehicle (HAUV) to experimentally evaluate our method for creating 3D mosaics. The imaging sonar used on the HAUV is a Sound Metrics Dual frequency IDentification SONar (DIDSON) [159]. Other relevant sensors on the HAUV are shown in Section 1.1.2. Though the HAUV has a



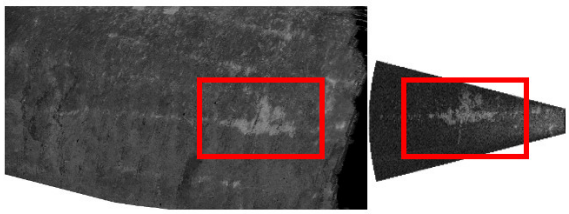
(a) Mosaic derived from bundle adjustment and CAD



(b) Close-up of (a) (left) and raw sonar image (right)



(c) Mosaic derived from piecewise-planar SLAM



(d) Close-up of (c) (left) and raw sonar image (right)

Figure 3.7: Qualitative results of imaging sonar mosaics. The CAD model was used to generate (a) and (b). Even without a prior CAD model, as shown in (c) and (d), we can still produce a 3D mosaic that appears nearly identical in texture to the model from (a). The most apparent difference is that the model in (c) has a smaller surface area. The HAUV was not always able to measure valid DVL returns near the water surface. This explains why those portions of the model are missing.

underwater stereo camera, it was only used for evaluation purposes and our method does not require it [135]. The periscope camera, however, was used to globally localize successive surveys into a single common hull-relative reference frame.

The datasets used in this section were collected on the 180 m *SS Curtiss* vessel shown in Section 1.1.2.3. The mosaics presented in Section 3.3.3 consist of eight individual surveys, all aligned to a common hull-relative frame of reference. Sample mosaics, with and without the use of a prior computer aided design (CAD) model, are shown in Fig. 3.7.

3.3.2 Evaluated SLAM Techniques

The input to our 3D mosaicing pipeline is a set of SLAM-corrected sonar poses. We applied our 3D mosaicing pipeline using four different techniques of varying computational efficiency and practicality to assess our approach in each setting. In particular, we investigate using (*i*) piecewise-planar surface SLAM, (*ii*) piecewise-planar surface constraints with underwater camera constraints, (*iii*) bundle-adjusted underwater camera poses, and (*iv*) bundle-adjusted underwater camera poses rigidly aligned to a CAD model. We provide more details for each method in the following sections.

3.3.2.1 Piecewise-planar SLAM

This approach (introduced in Chapter 2) models the ship hull as a collection of many planar features. Co-registered planar patches are constrained so that their normals are similar, but not to the extreme that the curvature of the hull is lost in the representation. Essentially, this deviation measures the orthogonal distance from a point on one mesh to the closest face on the mesh to which it is being compared. This method has the most practical significance since it does not require an underwater camera for navigation correction.

Despite not relying on an underwater camera, this technique is applied across multiple SLAM sessions, where the initial global localization must be determined with a periscope camera. For the HAUV application, a global positioning system (GPS) will not suffice since we desire the SLAM maps to be expressed in a hull-relative reference frame. This requirement can be relaxed for other applications, where GPS or beacons provide world-frame localization.

3.3.2.2 Piecewise-planar SLAM with underwater camera

A major benefit of the piecewise-planar SLAM technique discussed in the previous section is its usability in an underwater visual SLAM framework. In particular, we explore the use of piecewise planar SLAM techniques with the saliency-informed visual SLAM approach developed by Kim and Eustice [96]. This method does not require full camera coverage, but can still constrain navigational drift even if the space between survey tracks results in low or zero image overlap.

3.3.2.3 Bundle adjustment using underwater camera

This approach minimizes the reprojection error of visual features in a calibrated camera [165]. In our application, we use a stereo underwater camera with 100% coverage of the hull surface. This approach is computed offline, and uses scale-invariant feature transform (SIFT) features to make visual correspondences [109]. Outliers are rejected by using random sample consensus (RANSAC) [55] with a least-squares point cloud alignment algorithm originally proposed by Arun, Huang, and Blostein [6]. Along with optimizing a sparse set of SIFT features, we also include odometry measurements from the DVL and absolute constraints on depth, pitch, and roll from the HAUV's depth sensor and inertial measurement unit (IMU).

This approach is impractical for sonar mosaicing because it relies on 100% camera coverage, which is time-consuming and not possible in turbid water. However, we include this method so that we can compare its image blending consistency against the more practical approaches, discussed above.

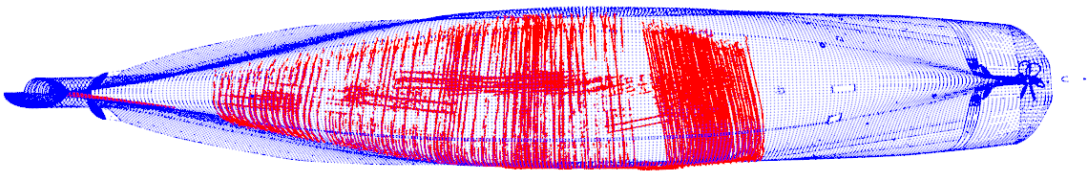


Figure 3.8: Rigid alignment between CAD model vertices (shown in blue) and the DVL point cloud computed from the bundle-adjusted underwater camera poses (shown in red). To perform the alignment, we use the outlier rejection approach described in Section 3.2.2 and find the optimal rigid transformation using the GICP algorithm [153].

3.3.2.4 Rigidly aligning bundle-adjusted poses to CAD model

For certain applications, like underwater ship hull inspection, a CAD model of the surveyed vessel may be available. Having ground-truth is a unique capability in marine robotics, especially for field experiments. We therefore draw attention to using bundle adjustment to take full advantage of this prior CAD model as a way to assess the quality of the proposed technique. In particular, the CAD model makes it possible to analyze the structural similarity of our 3D sonar mosaics to ground-truth. It should be noted that this method's reliance on camera-based bundle adjustment makes it impracticable for mosaicing with an imaging sonar. In addition, the CAD model is not a perfect indication of ground-truth due to structural wear, biofouling, and fluctuations in the hull form due to the ship's own weight.

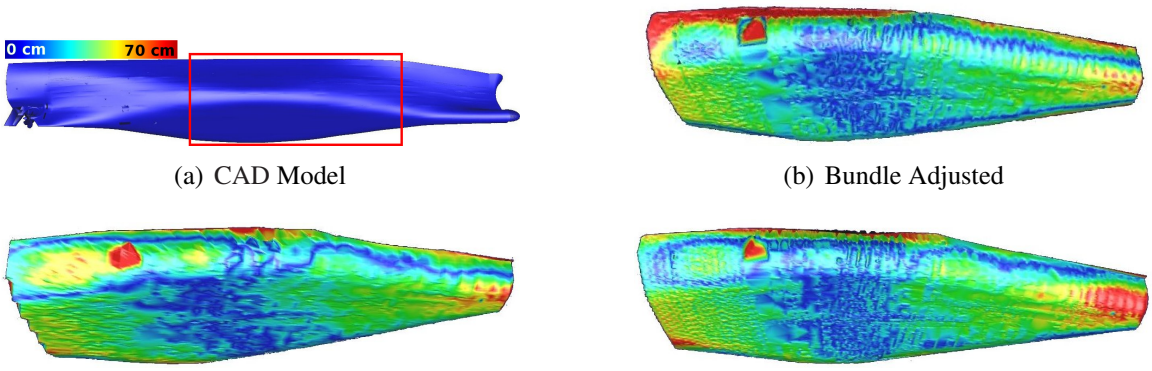
For this approach, we substitute the CAD model surface in place of the one derived from the SLAM methods described previously. We align the reference frame of the CAD model to the SLAM reference frame using the well-known generalized iterative closest point (GICP) algorithm [153]. The alignment between the bundle-adjusted DVL point cloud and *SS Curtiss* CAD model is shown in Fig. 3.8.

3.3.3 3D Mosaic Quality

We measure the quality of the 3D mosaic in two different ways. First, we consider the structural deviation from the mosaic's surface to the ground-truth CAD model. Second, we measure the consistency of the images used in the blending step described in Section 3.2.4.

For each of the SLAM techniques described in Section 3.3.2, we evaluate the structural similarity of the 3D surface to the ground-truth CAD model. This method uses the "attribute deviation metric" developed by Roy, Foufou, and Truchetet [148]. The false color visualization of this metric is shown in Fig. 3.9 for the reconstructions using linear interpolation, and in Fig. 3.10 for the reconstructions derived form GP regression.

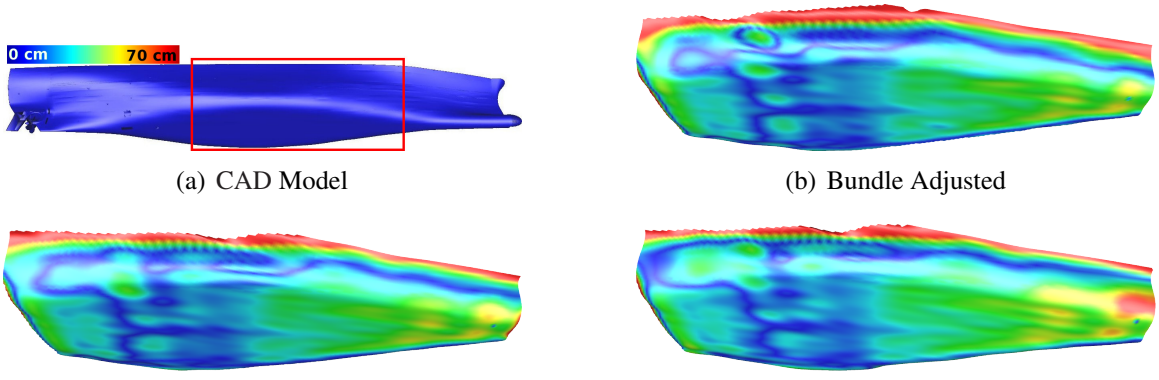
The mesh deviation results demonstrate that our method can produce accurate models



(a) CAD Model (b) Bundle Adjusted

(c) Piecewise-Planar+Visual SLAM (d) Piecewise-Planar SLAM

Figure 3.9: Deviation to ground-truth CAD model (a) using linear interpolation. The color from each figure denotes the amount of error, ranging from 0 cm to 70 cm (shown in the colorbar from (a)). The RMS error for bundle adjustment is actually the highest because the ship hull surface itself is left out of the optimization. However, (c) and (d) account for this and therefore have less error compared to the CAD model.



(a) CAD Model (b) Bundle Adjusted

(c) Piecewise-Planar+Visual SLAM (d) Piecewise-Planar SLAM

Figure 3.10: Deviation to ground-truth using GP regression. Compared to linear interpolation, the deviation is much improved along the side and bottom of the hull, however the deviation is significantly larger along the waterline (i.e., the top of each subfigure).

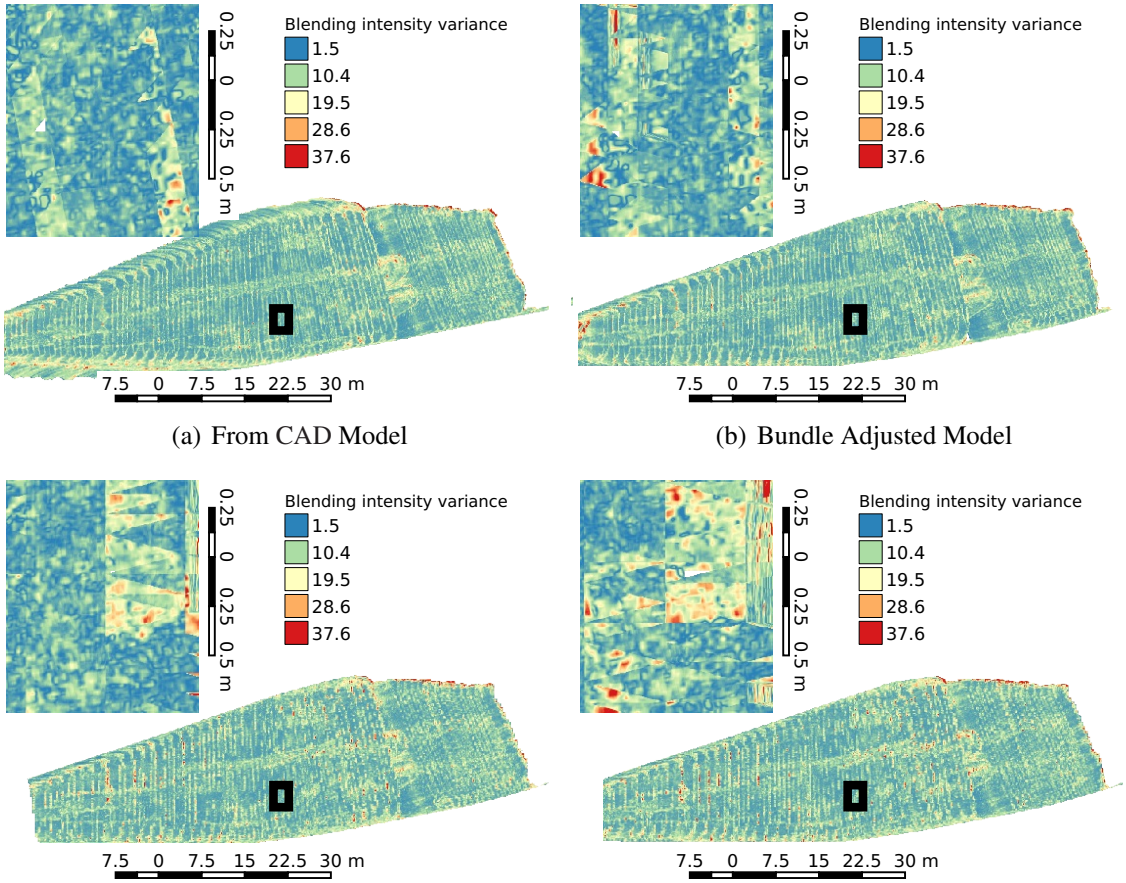
without the use of an underwater camera. Indeed, the methods leveraging our prior work with piecewise-planar SLAM outperforms bundle adjustment. This should not be surprising because our SLAM technique includes the ship hull surface itself as part of the optimization. Bundle adjustment, as implemented for this comparison, only uses visual feature correspondence to constrain the SLAM estimate.

In addition to the results shown in Fig. 3.9, Fig. 3.10, the *texture quality* of our sonar mosaics will clearly be sensitive to the quality of poses from SLAM. To quantify this, we computed the variance of weighted pixel intensities during the image blending step described in Section 3.2.4 for every pixel in the output mesh. These pixel intensities range from $[0, 255]$, corresponding to the value of an eight-bit unsigned integer. We provide a visualization of these variances in Fig. 3.11, where we highlight an area that was only correctly blended using a bundle adjustment step.

A histogram of these variances is shown in Fig. 3.12. Though the results are quite similar between each method, the results taken from poses that were *not* bundle-adjusted with a camera have noticeably heavier tails in the variance distribution (for variances greater than 20). That being said, these results show that, by and large, our mosaic blends together images with relatively similar pixel intensities. Considering that the globally bundle-adjusted techniques do perform better, this suggests that our method will see improvement if we add a step to globally match features across sonar images and include these constraints in a SLAM framework.

3.4 Conclusion

In summary, we have demonstrated a novel system to create 3D models using an imaging sonar, a DVL, and a small AUV. We provided a convenient empirical method to project 3D points onto a 2D pixel coordinate in a sonar image. We have shown that our mosaic pipeline can properly handle 3D geometry rather than requiring the environment to be entirely planar. We offered two ways to measure the quality of mosaic: structural deviation from ground-truth and variance over pixel intensities that are chosen for blending. We provided quantitative evaluations for our approach using both classic and recently-introduced SLAM techniques, such as bundle adjustment and modeling smoothly curved surfaces as piecewise planar. These evaluations were experimentally analyzed using field data from an AUV performing ship hull inspection.



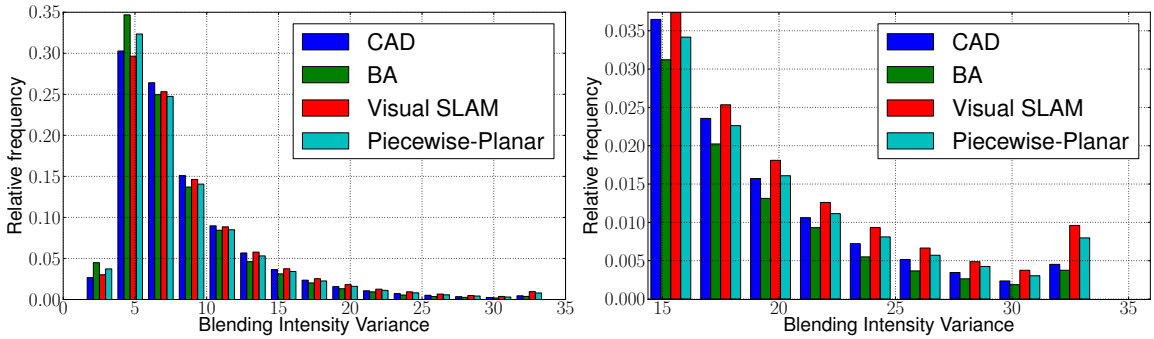
(a) From CAD Model

(b) Bundle Adjusted Model

(c) From Piecewise-Planar+Visual SLAM

(d) From Piecewise-Planar SLAM

Figure 3.11: Overview of the variance of weighted pixel intensities used during the blending step. Blue values denote consistent pixel intensities, while red shows relatively large variation. For certain small-scale regions, the blending consistency is noticeably compromised if the poses are not bundle adjusted. As a whole, however, the large-scale features are captured in all cases, as shown in Fig. 3.7.



(a) Full distribution

(b) Tail of distribution

Figure 3.12: Histogram of intensity variances for the results shown in Fig. 3.11. The tails in (b) are purposely clamped to emphasize that SLAM approaches that are not globally bundle adjusted do not perform as well as those that are. Importantly, however, the full histogram from (a) shows that as a whole our mosaic blends together images with relatively similar pixel intensities, even when using the piecewise-planar SLAM technique, which does not rely on constraints from an underwater camera.

CHAPTER 4

Model-Assisted Bundle Adjustment and Underwater Visual Mapping

4.1 Introduction

In the previous chapter, we explored a 3D photomosaicing pipeline using a two-dimensional (2D) imaging sonar. One of the capabilities we analyzed was the use of a low-fidelity prior model (derived from computer aided design (CAD) drawings), to provide the surface to which we apply texture. In this chapter, we propose methods to leverage this prior model into a 3D reconstruction pipeline consisting of a bundle adjustment (BA) step followed by a mapping step.

BA is a special case of the simultaneous localization and mapping (SLAM) problem; it is an estimation problem whose unknowns consist of camera poses and the positions of visually-observed features. Thus, BA is a reconstruction technique that seeks to estimate the three-dimensional (3D) structure of the scene and the egomotion of the camera. It is a widespread technique used throughout computer vision and mobile robotics, due mainly to the low cost and high reconstruction quality of digital cameras [2, 21, 38]. A major drawback of BA using optical cameras in robotic perception is the susceptibility to environmental noise and poor lighting conditions. Despite these challenges, the main benefit of vision-based 3D reconstruction is high spatial resolution and cost savings as compared to laser and acoustic-based sensing.

To mitigate the challenges of vision-based 3D reconstruction, researchers have previously proposed modifications to BA that leverage 3D models of the scene (provided *a priori*). This practice is sometimes referred to as *model-assisted bundle adjustment*. The reconstruction of human faces has been a particularly prevalent application domain, however these techniques have certain shortcomings that are ill-suited for their application in large-scale robotic surveillance. For instance, mobile robots typically survey areas that are much larger than

themselves, unlike the relative sizes of a camera and human faces. In addition, the scene will almost surely consist of 3D structure that is absent from the provided model.

Underwater environments are especially challenging for optical imaging and 3D reconstruction. In particular, underwater light attenuation and a phenomenon known as *back-scatter* are issues that researchers must consider when deploying autonomous underwater vehicles (AUVs) that are equipped with optical cameras [4, 23, 44, 80]. Despite these challenges, the benefits of the aforementioned model-assisted BA have yet to be explored in a large scale underwater setting. The facial reconstruction methods are designed for standard in-air imaging so these methods do not easily transition to underwater environments.

To address these challenges, we have developed an improved model-assisted BA framework that is easily applicable to underwater ship hull inspection, as shown in Fig. 4.1. In addition, we have leveraged this BA framework in a mapping pipeline that can identify foreign 3D structure and fuse it back into the prior model, as shown in Fig. 4.2. The contributions of Section 4.2 are as follows:

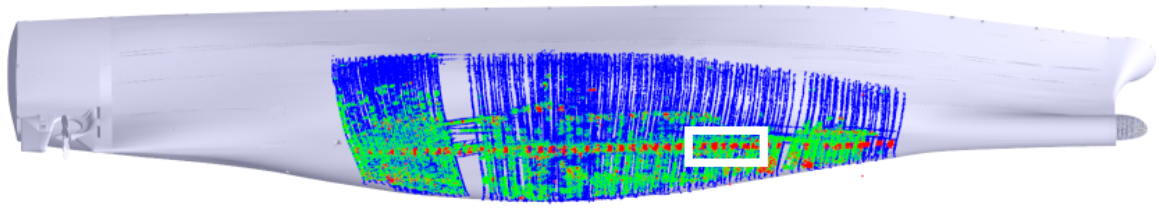
- We propose a expectation-maximization (EM) algorithm that assigns hard binary labels to each visual feature indicating if a feature lies on the nominal surface of the prior mesh. Given these labels, we solve for the optimal 3D locations of cameras and features accordingly. This approach is therefore capable of identifying 3D structure that is absent from the prior model.

- We show that this algorithm is a special case of the Gaussian max-mixture framework, which was originally intended for handling non-Gaussian error models in graphical SLAM [131].

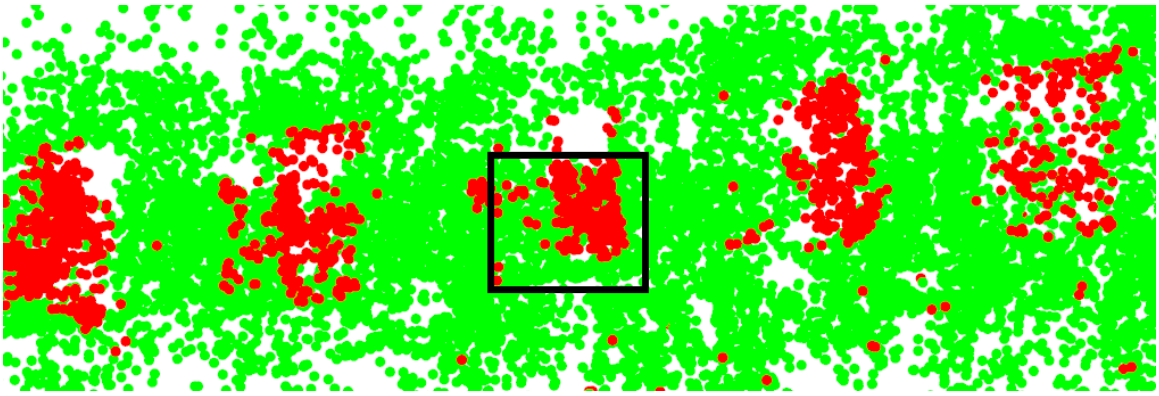
- To our best knowledge, we provide the largest evaluation of real-world model-assisted BA, both in physical scale and number of images.

In addition, in Section 4.3 we explore mapping techniques that fuse the model-assisted bundle-adjusted structure (in addition to texture, similar to our method presented in [138]) back into the low-fidelity prior mesh. In this chapter, we are interested in identifying structural differences detected from the underwater camera. We build upon this previous work by annotating a prior model with SLAM-derived structure. We show experimental results taken from the Bluefin Robotics hovering autonomous underwater vehicle (HAUV) platform for automated ship hull inspection [71]. The contributions of Section 4.3 allow our AUV to:

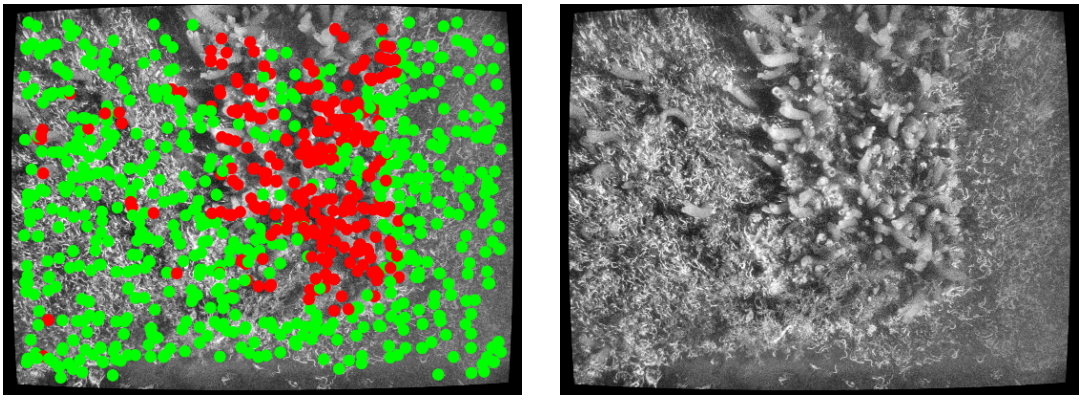
- Label visually-derived 3D shapes based on their deviation from the nominal *a priori* mesh.



(a)



(b)



(c) (d)

Figure 4.1: Overview of model-assisted bundle adjustment. (a) DVL ranges, shown in blue, allow us to localize to the prior model of the ship being inspected, shown in gray. In (b) and (c), visual features that are hypothesized to lie on the nominal surface of the prior model are shown in green. Features that correspond to 3D structure that is absent from the model are shown in red. In this example, red features correspond to tubular biogrowth emanating from docking blocks along the hull's centerline, in (d).

- Augment the nominal mesh with visually-derived 3D information, producing a high-fidelity map.

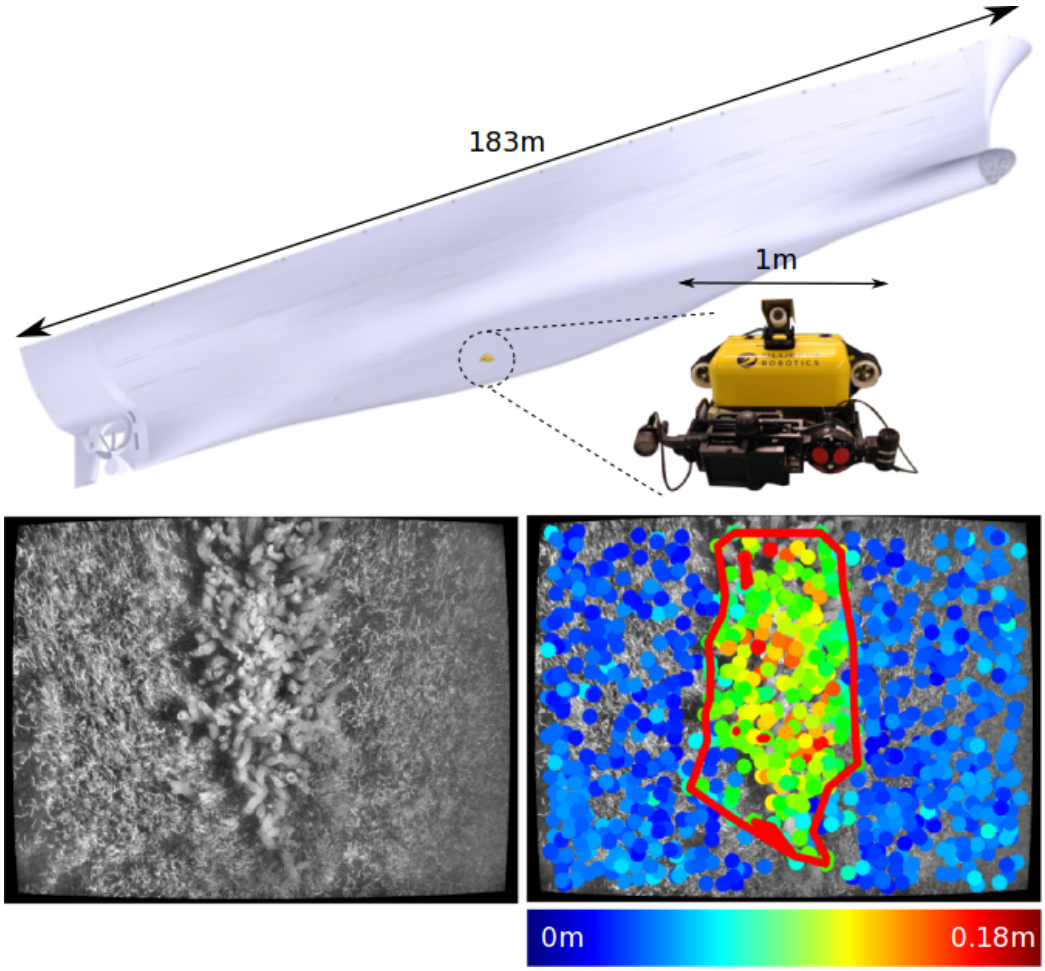


Figure 4.2: Overview of remeshing prior CAD model. In addition to an improved BA framework, we propose a mapping technique that can identify 3D shapes in the imagery (bottom) and fuse these back into a low-fidelity prior model derived from CAD drawings (top). The color of each feature encodes the structural deviation from the CAD model. An outline of clusters of these features is shown in red, computed using DBSCAN.

4.1.1 Related Work: Model-Assisted BA

Model-assisted visual reconstruction methods were particularly popular during the late 1990's and early 2000's, particularly in the domain of human face reconstruction [33, 54, 56, 91, 154]. Works by Fua [56] and Kang and Jones [91] are similar to traditional BA: a least-squares minimization over reprojection error. However, they introduce regularization terms that essentially enforce triangulated points lying close to the model's surface. Shan, Liu, and Zhang [154] introduce an optimization problem over a set of model parameters—rather than a regularization over features—that allow the generic face model to more closely match the geometry of the subject's face. Fidaleo and Medioni [54] noted that these methods are

rarely able to integrate 3D structure present in the subject that is absent from the model (such as facial hair and piercings). Instead, their approach used a prior model strictly for pose estimation, but the reconstruction of the face was entirely data-driven.

The primary application domain of these methods is in the reconstruction of human faces, however they have largely been overshadowed by modern, highly accurate, dense reconstruction methods that use either commodity depth cameras [123, 174], patch-based multiview stereopsis using high-quality imagery [57], or photometric stereo reconstruction techniques [94, 147]. These more recent methods have shown impressive reconstructions of both small-scale objects (human faces), and large scale objects (indoor environments and outdoor structures).

Recently, however, model-assisted methods have seen some re-emergence in particularly challenging areas of mobile robotics, such as the work by Geva et al. [58] in which an unmanned aerial vehicle (UAV) surveys a remote area. They used digital terrain models (DTMs) to regularize the position of 3D features observed from the camera mounted on the UAV, in a very similar fashion to the work in [56, 91]. These DTMs are freely available from the Shuttle Radar Topography project [53], and act as the prior model used in their approach. This approach is most similar to ours, however we differentiate our approach in three important ways: (*i*) our approach is capable of incorporating visual information that is absent from the nominal *a priori* model by assigning a hidden binary random variable for each visual feature; (*ii*) we use an orthogonal signed distance, rather than raycasting, to evaluate a feature's surface constraint likelihood; (*iii*) we evaluate our approach on a dataset with several orders of magnitude more bundle-adjusted keyframes.

4.1.2 Related Work: Underwater Visual Mapping

Early work in ship hull inspection includes the use of long-baseline navigation, where a robot localizes to a ship hull using manually-deployed acoustic pingers [166]. More recently, researchers have instead used underwater visual perception and SLAM techniques, rather than acoustic localization beacons, for AUV navigation. A survey of underwater visual sensing modalities was provided by [13]. Some examples of field robots that use visual perception include work by Negahdaripour and Firoozfam [121], in which they used a stereo camera rig on a remotely operated vehicle (ROV) to inspect the underwater portion of a ship hull. Visual mapping of underwater infrastructure was also explored by Ridao et al. [145] using an AUV with a calibrated underwater monocular vision system. In addition to mapping tasks, several researchers have explored automated object identification (such as corrosion or underwater mines) using both visual sensors [14] and acoustic sensors [11].

The computer vision and graphics community have studied fusing optical range measurements to form a reconstruction of a 3D surface for several decades [37, 64, 93]. The seminal work by Curless and Levoy [37] used running averages to fuse range measurements into an implicit surface. This simple approach is still used in state-of-the-art surface reconstruction and pose tracking algorithms using a commodity depth camera [123, 174]. We differentiate our work in three ways. First, we assume the availability of a nominal mesh of the surface being constructed and that the camera pose with respect to this mesh is unknown. Second, we assume that the object can only be observed at a very close distance—i.e., the observations are locally planar and so using iterative closest point (ICP) (or its variants) to estimate relative poses between keyframes is ill-constrained [32, 153, 180]. Third, we do not assume the availability of dense depth images during the pose estimation stage. Though we use a stereo camera for some of our experimental analysis (from which a dense disparity map can be easily converted to a dense depth image), we instead use sparse feature-based registration so that this approach is also applicable to monocular (bearing-only) cameras.

4.1.3 Outline

This chapter is organized into two major components: a model-assisted BA framework discussed in Section 4.2 and a underwater visual mapping pipeline discussed in Section 4.3. In Section 4.2, we describe the mathematical model for a robust model-assisted optimization framework, and we show that the approach is a special case of the Gaussian max-mixture models from Olson and Agarwal [131]. In Section 4.3, we describe a mapping framework that fuses the BA result back into the prior model in the form of triangulated 3D shapes derived from a clustering algorithm. Results and discussion are provided in Section 4.4. In Section 4.5 we offer some concluding remarks.

4.2 Model-Assisted Bundle Adjustment

4.2.1 Notation

We denote the set of all unknowns, $\mathbf{X}$, as consisting of N_p poses, $\mathbf{x}_{g1} \dots \mathbf{x}_{gN_p}$, the relative transformation to the model frame, $\mathbf{x}_{g\mathcal{M}}$, and N_l landmarks, $\mathbf{l}_1 \dots \mathbf{l}_{N_l}$

$$\mathbf{X} = \{\underbrace{\mathbf{x}_{g1} \dots \mathbf{x}_{gN_p}}_{\text{robot poses}}, \underbrace{\mathbf{x}_{g\mathcal{M}}}_{\text{model pose}}, \underbrace{\mathbf{l}_1 \dots \mathbf{l}_{N_l}}_{\text{visual landmarks (features)}}\},$$

where $\mathbf{x}_{ij}$ denotes the 6-degree-of-freedom (DOF) relative-pose between frames i and j. The common, or global frame, is denoted as g. Visually-derived features, denoted as $\mathbf{l}_i$, are the 3D positions of features as expressed in the global frame. Finally, $\mathcal{M}_{\text{prior}}$ denotes a prior triangular mesh consisting of a set of vertices, edges between vertices, and triangular faces.

Note that $\mathbf{X}$ may consist of additional variables, such as extrinsic parameters of the robot sensors. We omit these values now for the sake of clarity, however we re-introduce them in Section 4.2.6.

Let $\mathbf{Z}$ denote the set of all measurements, which consists of all odometry measurements, priors, surface range measurements (e.g., from an active range scanner), visual feature detections, and surface constraints (which will be described in Section 4.2.3),

$$\mathbf{Z} = \{\mathcal{Z}_{\text{odo}}, \mathcal{Z}_{\text{prior}}, \mathcal{Z}_{\text{range}}, \mathcal{Z}_{\text{feat}}, \mathcal{Z}_{\text{surf}}\}.$$

We assign a hidden binary latent variable to each visual feature,

$$\mathbf{\Lambda} = \{\lambda_1 \ldots \lambda_{N_l}\}, \lambda_i \in \{0, 1\},$$

where a value of one encodes that a visually-derived feature lies on the nominal surface of $\mathcal{M}_{\text{prior}}$. A value of zero encodes that the visually-derived feature corresponds to physical structural that is absent from $\mathcal{M}_{\text{prior}}$.

4.2.2 Formulation as EM

The goal of our work is to estimate $\mathbf{X}$ using a simplified variant of the EM algorithm, known as *hard EM*:

1. Initialize $\mathbf{X}$
2. Repeat the following until $p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X})$ converges:
   (a) $\mathbf{\Lambda}^* = \underset{\mathbf{\Lambda}}{\operatorname{argmax}}\, p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X})$
   (b) $\mathbf{X}^* = \underset{\mathbf{X}}{\operatorname{argmax}}\, p(\mathbf{Z}, \mathbf{\Lambda}^*|\mathbf{X})$

Similar to previous work, we introduce a set of prior measurements, $\mathcal{Z}_{\text{surf}}$, that regularize the positions of 3D visual features so that they lie on the surface of $\mathcal{M}_{\text{prior}}$. We expand the likelihood function using Bayes' rule and note that the odometry, prior, and feature detection observations are independent of the feature labels (and conditionally independent of each

other):

$$p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X}) = p(\mathbf{Z}|\mathbf{\Lambda}, \mathbf{X})p(\mathbf{\Lambda}|\mathbf{X})$$
$$= p(\mathcal{Z}_{\text{odo}}, \mathcal{Z}_{\text{prior}}, \mathcal{Z}_{\text{range}}, \mathcal{Z}_{\text{feat}}|\mathbf{X})p(\mathcal{Z}_{\text{surf}}|\mathbf{\Lambda}, \mathbf{X})p(\mathbf{\Lambda}|\mathbf{X}). \quad (4.1)$$

If we conservatively assume that $p(\lambda_i|\mathbf{X})$ is uninformative, then we can express the likelihood as proportional to a simpler expression:

$$p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X}) \propto p(\mathcal{Z}_{\text{odo}}, \mathcal{Z}_{\text{prior}}, \mathcal{Z}_{\text{range}}, \mathcal{Z}_{\text{feat}}|\mathbf{X})p(\mathcal{Z}_{\text{surf}}|\mathbf{\Lambda}, \mathbf{X}).$$

Therefore, Step 2(a) in the hard EM algorithm simplifies to

$$\underset{\mathbf{\Lambda}}{\text{argmax}}\, p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X}) = \underset{\mathbf{\Lambda}}{\text{argmax}}\, p(\mathcal{Z}_{\text{surf}}|\mathbf{\Lambda}, \mathbf{X}), \quad (4.2)$$

where $p(\mathcal{Z}_{\text{surf}}|\mathbf{\Lambda}, \mathbf{X})$ is described in Section 4.2.3. In addition, Step 2(b) simplifies to

$$\underset{\mathbf{X}}{\text{argmax}}\, p(\mathbf{Z}, \mathbf{\Lambda}|\mathbf{X}) =$$
$$\underset{\mathbf{X}}{\text{argmax}}\, p(\mathcal{Z}_{\text{odo}}, \mathcal{Z}_{\text{prior}}, \mathcal{Z}_{\text{range}}, \mathcal{Z}_{\text{feat}}|\mathbf{X})p(\mathcal{Z}_{\text{surf}}|\mathbf{\Lambda}, \mathbf{X}), \quad (4.3)$$

which is equivalent to a least-squares optimization problem when the measurements are corrupted by additive Gaussian noise.

4.2.3 Modeling the Surface Constraint

Consider the set of all surface constraints $\mathcal{Z}_{\text{surf}} = \{z_{s_1} \ldots z_{s_{N_l}}\}$, we model the conditional distribution of these constraints as Gaussian:

$$p(z_{s_i}|\lambda_i, \mathbf{X}) = \begin{cases} \mathcal{N}(h\left(\mathbf{x}_{g\mathcal{M}}, \mathbf{l}_i\right), \sigma_0^2), & \lambda_i = 0 \\ \mathcal{N}\left(h\left(\mathbf{x}_{g\mathcal{M}}, \mathbf{l}_i\right), \sigma_1^2\right), & \lambda_i = 1 \end{cases}, \quad (4.4)$$

where $h(\,\cdot\,)$ computes the orthogonal signed distance of the i^{th} feature to the model. The values σ_0^2 and σ_1^2 denote the variance of the surface constraint when λ_i is 0 or 1, respectively. Intuitively, these variances are chosen such that $\sigma_1^2 \ll \sigma_0^2$, i.e., features that lie close to the model surface are more tightly pulled toward it, while features that lie away from the model are free to vary with approximately zero cost. To constrain visual features to lie on the surface, we assign $z_{s_i} = 0$ for all of the features. If we desired the surfaces to tend toward lying inside or outside the surface by distance d, we would assign z_{s_i} to $-d$ or d,

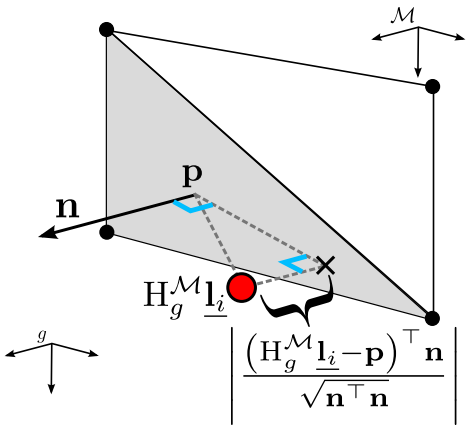


Figure 4.3: Overview of the surface constraint using a simple triangular mesh $\mathcal{M}$ consisting of two triangles. The constraint converts the distance to the closest face, d_{s_i}, to a signed distance (depending on if the feature is inside or outside the triangular face).

respectively.

The orthogonal signed distance function $h(\,\cdot\,)$ is a nonlinear function of the pose of the model and the position of the visual feature:

$$h\left(\mathbf{x}_{g\mathcal{M}}, \mathbf{l}_i\right) = \frac{\left(\mathrm{H}_g^{\mathcal{M}}\underline{\mathbf{l}}_i - \mathbf{p}\right)^\top \mathbf{n}}{\sqrt{\mathbf{n}^\top\mathbf{n}}}, \tag{4.5}$$

where $\mathrm{H}_g^{\mathcal{M}} = \left[{}_g^{\mathcal{M}}\mathrm{R} \,|\, {}^{\mathcal{M}}\mathbf{t}_{\mathcal{M}g}\right]$ is the transformation (orthonormal rotation matrix, ${}_g^{\mathcal{M}}\mathrm{R}$, followed by three-vector translation, ${}^{\mathcal{M}}\mathbf{t}_{\mathcal{M}g}$) of points in the global frame to points in the model frame and $\underline{\mathbf{u}} = [\mathbf{u}^\top 1]^\top$ represents a vector expressed in homogeneous coordinates. Intuitively, $h(\,\cdot\,)$ returns the orthogonal signed distance of a visual feature $\mathbf{l}_i$ to the surface of the closest triangular face in $\mathcal{M}$. This triangle is characterized by any point, $\mathbf{p}$, that lies on the surface of the triangle, and its surface normal, $\mathbf{n}$. This calculation is illustrated in Fig. 4.3.

4.2.4 Relation to Gaussian Max-Mixture Models

In this section, we show how the previous formulation is a special case of Gaussian max-mixture models proposed by Olson and Agarwal [131]. This approach was mainly introduced in the area of robust SLAM backends as a probabilistically motivated approach to rejecting incorrect loop closures [1, 131, 160] and detecting wheel slippage in ground robots [131]. More recently, it has been applied in learning robust models for consumer-grade global positioning system (GPS) measurements that can reject outliers [118].

First, we note that the surface constraint likelihood $p(z_{s_i}|\mathbf{X})$ is not Gaussian (due to the

absence of λ_i). Even so, we can write the conditional distribution of the unknowns given the measurements as

$$\log p(\mathbf{X}|\mathbf{Z}) \propto \log \prod_i p(z_i|\mathbf{X}), \tag{4.6}$$

where z_i denotes the i^{th} measurement; either an odometry, prior, range, feature, or surface constraint. By maximizing this distribution, we arrive at a maximum *a posteriori* (MAP) estimate for $\mathbf{X}$, as shown by [39].

Though the labels, $\mathbf{\Lambda}$, are absent from (4.6), we can re-introduce them by adapting the Gaussian max-mixture distribution proposed by Olson and Agarwal [131]. The surface constraint likelihood then takes the form

$$p(z_{s_i}|\mathbf{X}) = \eta \max_{\lambda_i} p(z_{s_i}|\lambda_i, \mathbf{X}). \tag{4.7}$$

The logarithm can be brought inside the product from (4.6), and again inside the $\max$ operator from (4.7). This distribution can therefore be thought of as a binary Gaussian max-mixture with equal weights for each component of the mixture.

This conditional distribution essentially combines Steps 2(a) and 2(b) from the hard EM algorithm so that the labels are determined whenever the likelihood term is evaluated. The distribution from (4.7) is therefore equivalent to a binary max-mixture of Gaussians with equal weights. This conforms to our earlier formulation from Section 4.2.2 that assigns equal prior probability to a feature lying on or off the mesh's surface. The only two parameters used in our approach are therefore σ_0^2 and σ_1^2 from (4.4). We illustrate this distribution in Fig. 4.4 using typical values for these parameters.

Note that the distribution from (4.7) contains an unknown normalization constant, η, that ensures a valid probability distribution. However, for the purposes of maximizing the likelihood, computing the specific value of this scale factor is not necessary [131]. Additionally, we represent the distribution from (4.6) using a factor graph [39], as shown in Fig. 4.5. To solve the corresponding least-squares problem, we use the freely-available Ceres library [3].

4.2.5 Localizing to the Prior Model

Our approach, like all model-assisted BA frameworks, require a good initial guess of the alignment between the camera trajectory and the prior model. This is typically done using triangulated features from the camera imagery, but for autonomous ship hull inspection there are many portions of the ship where no visual features can be detected because the

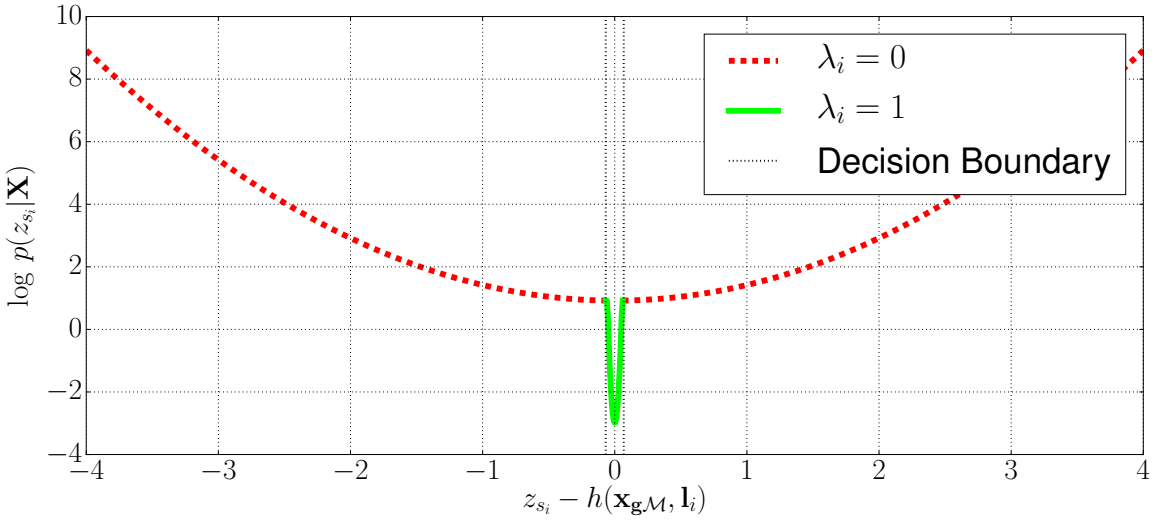


Figure 4.4: Decision boundary for $\sigma_0 = 1$ m, $\sigma_1 = 0.02$ m overlayed on the log probability (i.e., cost computed during optimization). If desired, features could be biased toward the "inside" or "outside" of the model by assigning a nonzero value to z_{s_i} to shift this curve left and right, respectively. For our experiments, however, we assign $z_{s_i} = 0$ for all features.

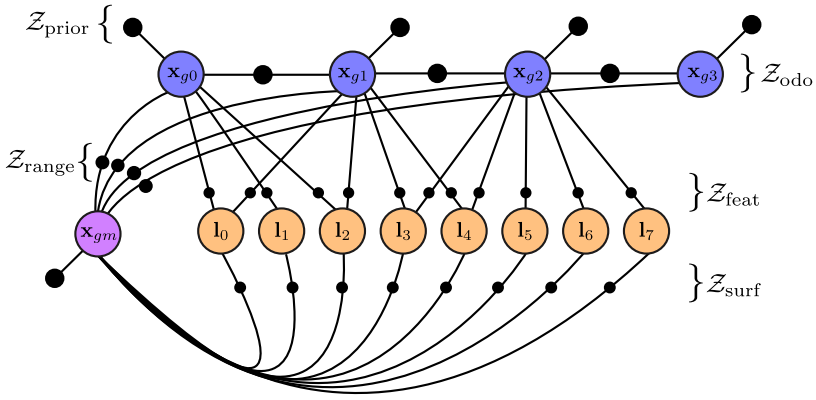
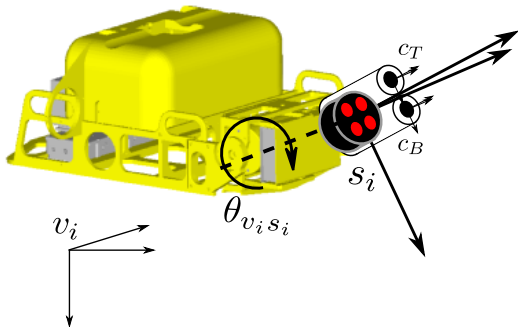


Figure 4.5: Representation of our method as a factor graph. The factor nodes denoted with $\mathcal{Z}_{\text{surf}}$ denote the surface constraints, which are implemented using binary Gaussian max-mixtures distributions from Section 4.2.4. These factors constrain the pose of the prior model $\mathbf{x}_{gm}$ and the location of visual features $\mathbf{l}_i$.

hull is not uniformly salient [30, 96, 108].

In our case, the underwater robot observed sparse range measurements using a Doppler velocity log (DVL). These range returns are rigidly aligned to the prior model using generalized iterative closest point (GICP) [153], which serves as an initial guess (i.e., the prior factor connected to node $\mathbf{x}_{gm}$ in Fig. 4.5). In our case, one point cloud consists of vertices in $\mathcal{M}_{\text{prior}}$, while the other point cloud consists of DVL range returns expressed



(a)

Figure 4.6: Illustration of the various reference frames at time i with respect to the vehicle frame, v_i. The vehicle has a sensor tray that houses both the DVL and vertically-oriented stereo rig. An onboard servo rotates the servo frame, s_i, which in turn rotates the DVL and camera. The vehicle controls this angle so that these sensors point approximately orthogonal to the ship hull surface. This angle is instrumented, but must be treated as uncertain in our estimation framework due to the mechanical slop in the servo.

in the global frame, which are SLAM-corrected in real-time using the method described in [137]. Individual poses can be further optimized using raycasting techniques to compute the likelihood of $\mathcal{Z}_{\text{range}}$ and the surface constraint measurements $\mathcal{Z}_{\text{surf}}$ from Section 4.2.3.

4.2.6 Application to the HAUV

In Section 4.2.3 we described the general evaluation of the surface constraint measurements, $\mathcal{Z}_{\text{surf}}$. In this section we describe in detail the measurement models used on the HAUV and relate them to the factors illustrated in Fig. 4.5. Each subsection will describe $\mathcal{Z}_{\text{prior}}$, $\mathcal{Z}_{\text{odo}}$, $\mathcal{Z}_{\text{feat}}$, and $\mathcal{Z}_{\text{range}}$ in terms of the conditional distributions of the observations given the unknowns. The covariance matrix for each factor is assumed known, as is standard practice.

As suggested earlier, we also include extrinsic sensor parameters in the factor graph formulation. In particular, we assume that the time-varying servo angle that actuates the HAUV's sensor tray is instrumented but uncertain. Similarly, the static servo-to-camera transform is also not precisely calibrated and treated as uncertain. Therefore, we denote these unknowns as $\theta_{v_i s_i}$ and $\mathbf{x}_{sc_T}$, respectively, and include them as variable nodes in our factor graph implementation. These additional unknowns are illustrated in Fig. 4.6(a).

4.2.6.1 Prior factors

A full-state prior on all six degrees of freedom for a particular variable node, $\mathbf{x}_{ij}$, is given by the conditional distribution of the measurement $\mathbf{z}^{\text{full}}_{\mathbf{x}_{ij}}$:

$$p\left(\mathbf{z}^{\text{full}}_{\mathbf{x}_{ij}}\middle|\,\mathbf{x}_{ij}\right) = \mathcal{N}\left(\mathbf{x}_{ij}, \Sigma_{\mathbf{z}^{\text{full}}_{\mathbf{x}_{ij}}}\right). \tag{4.8}$$

The initial guess for the pose of the prior model, $\mathbf{x}_{gm}$, is determined using the GICP algorithm for aligning two point clouds [153], as discussed in Section 4.2.5. This GICP alignment is added as a prior factor on $\mathbf{x}_{gm}$.

The onboard depth and inertial measurement unit (IMU) sensors allow us to directly observe a bounded-error measurement of the vehicle's depth, pitch, and roll. This observation, denoted $\mathbf{z}^{\text{zpr}}_{\mathbf{x}_{ij}}$, has the following conditional distribution:

$$p\left(\mathbf{z}^{\text{zpr}}_{\mathbf{x}_{ij}}\middle|\,\mathbf{x}_{ij}\right) = \mathcal{N}\left(\left[z^{i}_{ij}, \phi_{ij}, \theta_{ij}\right]^{\top}, \Sigma_{\mathbf{z}^{\text{zpr}}_{\mathbf{x}_{ij}}}\right). \tag{4.9}$$

Finally, we model the servo angle at time i as being directly observed with a prior factor. The corresponding observation model is simply:

$$p\left(z_{v_i s_i}\middle|\,\theta_{v_i s_i}\right) = \mathcal{N}\left(\theta_{v_i s_i}, \sigma^2_{z_{s_i}}\right). \tag{4.10}$$

4.2.6.2 Odometry factors

Our factor graph formulation models odometry measurements as a sequential relative-pose observation, $\mathbf{z}^{\text{odo}}_{i(i+1)}$. The conditional distribution of this measurement is

$$p\left(\mathbf{z}^{\text{odo}}_{i(i+1)}\middle|\,\mathbf{x}_{gi}, \mathbf{x}_{g(i+1)}\right) = \mathcal{N}\left(\ominus\mathbf{x}_{gi} \oplus \mathbf{x}_{g(i+1)}, \Sigma_{\mathbf{z}^{\text{odo}}_{i(i+1)}}\right), \tag{4.11}$$

where $\oplus$ and $\ominus$ are pose composition operators following the conventions of Smith, Self, and Cheeseman [158].

4.2.6.3 Stereo camera factors

The observed pixel locations at time i corresponding to the k^{th} feature are denoted as $\mathbf{z}^{T}_{ik}$ and $\mathbf{z}^{B}_{ik}$ for the top and bottom cameras, respectively:

$$p\left(\left[\mathbf{z}^{T\top}_{ik}\ \ \mathbf{z}^{B\top}_{ik}\right]^{\top}\middle|\,\mathbf{x}_{gv_i}, \mathbf{x}_{sc_T}, \theta_{v_i s_i}, \mathbf{l}_k\right) = \mathcal{N}\left(h_c\left(\mathbf{x}_{gv_i}, \mathbf{x}_{sc_T}, \theta_{v_i s_i}, \mathbf{l}_k\right), \sigma^2_c \mathrm{I}_{4\times 4}\right). \tag{4.12}$$

The observation model, h_c, corresponds to two pinhole cameras in a calibrated and rectified vertical stereo configuration (from Fig. 4.6):

$$h_c\left(\mathbf{x}_{gv_i}, \mathbf{x}_{sc_T}, \theta_{v_i s_i}, \mathbf{l}_k\right) = \begin{bmatrix} \mathrm{K}_{c_T}\left[{}^{c_T}_{g}\mathrm{R} \,|\, {}^{c_T}\mathbf{t}_{c_T g}\right]\underline{\mathbf{l}_k} \\ \mathrm{K}_{c_B}\left[{}^{c_B}_{g}\mathrm{R} \,|\, {}^{c_B}\mathbf{t}_{c_B g}\right]\underline{\mathbf{l}_k} \end{bmatrix}.$$

In general, $\left[{}^{j}_{i}\mathrm{R} \,|\, {}^{j}\mathbf{t}_{ji}\right]$ denotes the transformation of a point from frame i to frame j. In this case, it can represent the transformation of points from the global frame, g, to the top camera, c_T (i.e., the rotation and translation of the composed pose $\mathbf{x}_{gv_i} \oplus \mathbf{x}_{v_i s_i} \oplus \mathbf{x}_{sc_T}$, where $\mathbf{x}_{v_i s_i} = [0, 0, 0, 0, \theta_{v_i s_i}, 0]^\top$). It also can describe the transformation of points in the global frame to the bottom camera, c_B (i.e., the rotation and translation of the composed pose $\mathbf{x}_{gv_i} \oplus \mathbf{x}_{v_i s_i} \oplus \mathbf{x}_{sc_T} \oplus \mathbf{x}_{c_T c_B}$, where $\mathbf{x}_{c_T c_B}$ is the transformation from the top camera frame to the bottom camera frame). This transformation is taken from stereo camera calibration.

Note that our notation for $h_c\left(\,\cdot\,\right)$ omits the dehomogenization of each camera projection for the sake of clarity.

4.2.6.4 Monocular camera factors

In the event a stereo camera is not available, our framework can also use a calibrated monocular camera. Similar to the above stereo camera factor from (4.12), we have

$$p\left(\mathbf{z}^c_{ik} \middle|\, \mathbf{x}_{gv_i}, \mathbf{x}_{sc_T}, \theta_{v_i s_i}, \mathbf{l}_k\right) = \mathcal{N}\left(\mathrm{K}\left[{}^{c}_{g}\mathrm{R} \,|\, {}^{c}\mathbf{t}_{cg}\right]\mathbf{l}_k, \sigma_c^2 \mathrm{I}_{2\times 2}\right). \tag{4.13}$$

Similar to (4.12), we omit the extra dehomogenization step for the sake of clarity.

4.2.6.5 DVL raycast factors

A critical component of our factor graph formulation are factors modeling the intersection of DVL beams to the prior mesh—doing so allows us to significantly constrain both the unknown vehicle poses and servo angles. The conditional distribution takes the form:

$$p\left(z_{r_{in}} \middle|\, \mathbf{x}_{gm}, \mathbf{x}_{gv_i}, \theta_{v_i s_i}\right) = \mathcal{N}\left(h_{rn}\left(\mathbf{x}_{gm}, \mathbf{x}_{gv_i}, \theta_{v_i s_i}; \mathcal{M}_{\text{prior}}\right), \sigma^2_{z_{r_{in}}}\right), \tag{4.14}$$

where h_{rn} corresponds to raycasting the n^{th} beam for a DVL in a four-beam Janus configuration [19]. This observation model is illustrated in Fig. 4.7. Since the prior mesh may consist of hundreds of thousands of triangles, we use an efficient octree-based raycast implementation [151]. In addition, when evaluating the Gaussian distribution from (4.12) and (4.14) we apply a Huber M-estimator to the squared loss term to automatically reject

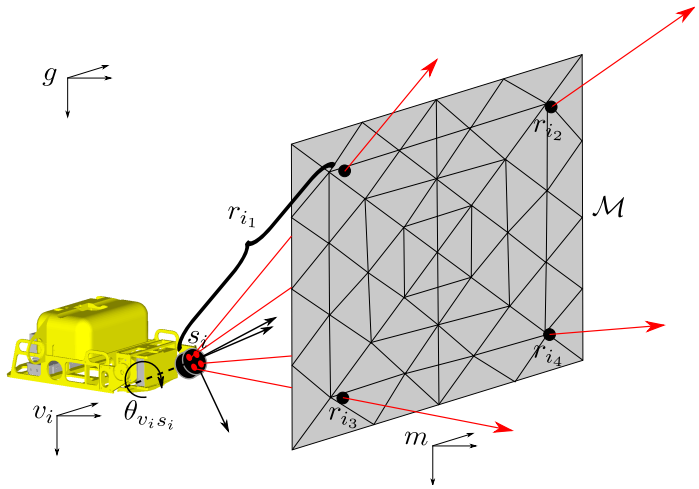


Figure 4.7: Illustration of ray-casting constraint. Given pose of the vehicle frame at time i, $\mathbf{x}_{gv_i}$, the servo angle, $\theta_{v_i s_i}$, the pose of the prior mesh frame, $\mathbf{x}_{gm}$, and the prior mesh, $\mathcal{M}_{\text{prior}}$, the four DVL range returns can be computed with an efficient octree-based ray-casting approach. At time i, the four ranges are predicted as r_{i_1}, r_{i_2}, r_{i_3}, and r_{i_4}.

outliers [74].

4.2.7 Frontend Details

The focus of this chapter is on the optimization backend so we purposefully omit some details of the visual frontend that establishes feature correspondences between keyframes. However, the approach we use is quite standard: a random sample consensus (RANSAC)-based algorithm to reject outliers established during putative feature descriptor matching. In the case of a stereo camera, we use a standard triangulated point-cloud alignment technique to ensure inlier feature matches are consistent (in a Euclidean sense) under a 6-DOF rigid transformation [6, 127]. For a monocular camera, we fit an essential matrix and measure the consistency using Sampson (i.e., epipolar) distance.

We used the scale-invariant feature transform (SIFT) feature descriptor [109] to assign putative visual correspondences. All feature detection and putative matching are done on a graphical processing unit (GPU) to keep the frontend performant.

4.3 Model-Assisted Visual Mapping

4.3.1 Notation

In addition to the notation introduced in Section 4.2.1, we will let $\mathcal{M}_{\text{new}}$ denote the updated mesh (i.e., the fusion of camera-derived structure with the prior model, $\mathcal{M}_{\text{prior}}$). This is the final output of our algorithm.

4.3.2 Identifying Shapes by Clustering Features

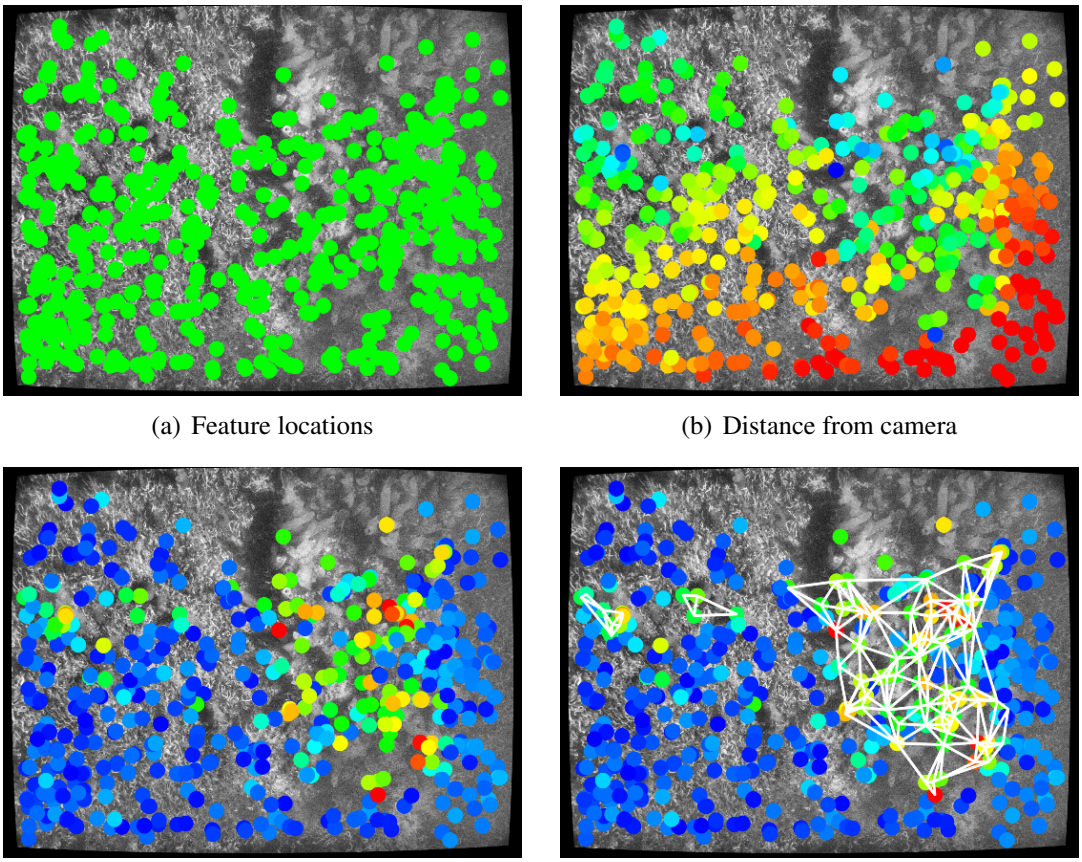
Once we have an estimate of all unknowns in the factor graph formulation from Section 4.2, we can easily compute the structural deviation from the prior model using the formula for signed distance that is provided in (4.5).

Algorithm 4 Detect shapes at a given camera pose at time i

Require: Camera pose (pose$_i$), visible features ($\mathcal{F}_{vi}$), and mesh ($\mathcal{M}_{\text{prior}}$)

$\mathcal{P}_i = \emptyset$ // Set of points to cluster.
$\mathcal{C}_i = \emptyset$ // Set of clusters from DBSCAN.
$\mathcal{S}_i = \emptyset$ // Set of detected shapes.
for feature $\mathbf{l} = [l_x, l_y, l_z]$ in $\mathcal{F}_{vi}$, expressed in global frame **do**
  $d = |h(\mathbf{x}_{g\mathcal{M}}, \mathbf{l})|$ // See Eqn. (4.5).
  if $d > \tau$ **then**
    $[f_x\ f_y\ f_z]^\top = \mathrm{H}^g_{c_T}\underline{\mathbf{l}}$ // Transform to camera frame
    $\mathcal{P}_i = \mathcal{P}_i \cup \{f_x, f_y, d\}$
  end if
end for
$\mathcal{C}_i$ = `DBSCAN`$(\mathcal{P}_i)$
for cluster in $\mathcal{C}_i$ **do**
  $\mathcal{M}_{S_i}$ = `alpha_shape`(cluster)
  $\mathcal{S}_i = \mathcal{S}_i \cup \mathcal{M}_{S_i}$
end for
return $\mathcal{S}_i$

Once we establish the deviations of each feature, we apply density-based spatial clustering of applications with noise (DBSCAN) to nonlinearly separate the features' positions into clusters [47]. These clusters of points are converted to shapes using a simple extension to Delaunay triangulation known as *alpha-shapes* [46]. This algorithm is summarized in Algorithm 4, with an accompanying example in Fig. 4.8. As shown in Fig. 4.8(d), the detected shapes can be projected as two-dimensional triangular meshes in the camera imagery. Note that a single physical object on the hull will have multiple shapes associated with it. In Algorithm 5, these shapes are combined across multiple views and fused into $\mathcal{M}_{\text{prior}}$.



(a) Feature locations

(b) Distance from camera

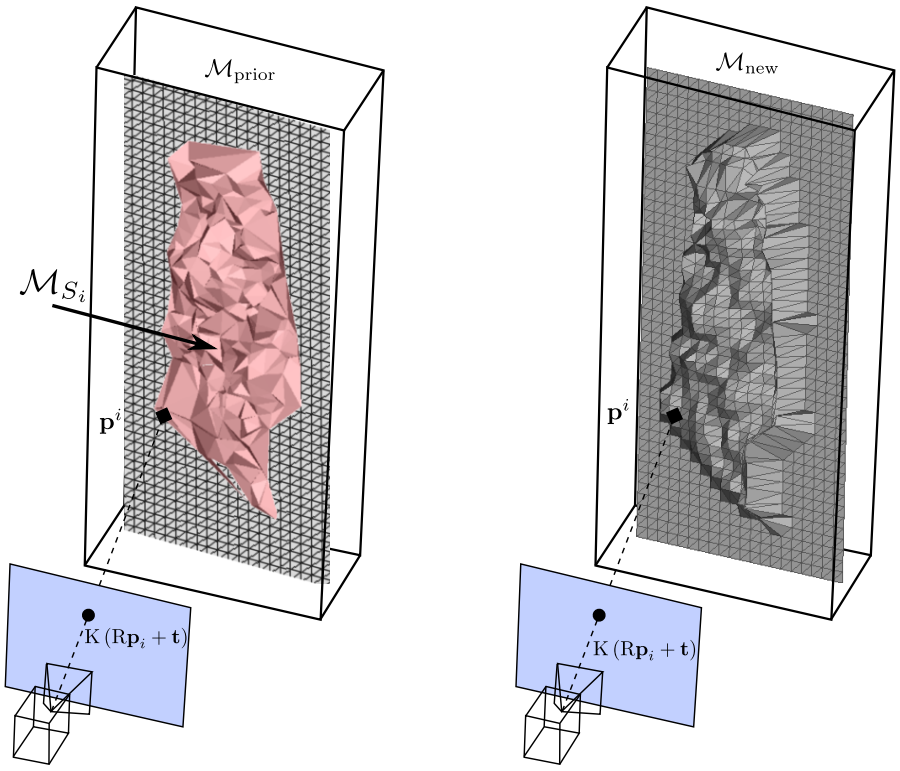
(c) Deviation to $\mathcal{M}_{\text{prior}}$

(d) Alpha-shaped clusters

Figure 4.8: Visual overview of Algorithm 4. For a particular keyframe, the bundle adjusted features, (a) and (b), are assigned a deviation by computing the intersection of the ray to the prior model (c). These values are clustered using DBSCAN, and meshed using alpha-shapes. In (d), the three detected clusters have their alpha-shapes shown as white triangular meshes.

4.3.3 Model Remeshing Step

The final step of our approach is to fuse the shapes detected in Algorithm 4 with the prior mesh, $\mathcal{M}_{\text{prior}}$, resulting in a new mesh, $\mathcal{M}_{\text{new}}$. To this end, we compute a ray originating from the top camera's center and extending toward a vertex in $\mathcal{M}_{\text{prior}}$. Like the DVL range observation model from (4.14), we use a raycasting approach to compute the intersection with any detected shapes. Once the intersection point corresponding to the i^{th} vertex in $\mathcal{M}_{\text{prior}}$, $\mathbf{p}^i$, is calculated, we update the corresponding vertex in $\mathcal{M}_{\text{new}}$, $\widehat{\mathbf{v}}^i$, with a recursive



(a) DBSCAN mesh overlayed on prior mesh

(b) Augmented CAD mesh

Figure 4.9: Visualization of Algorithm 5. A 3D shape, $\mathcal{S}$, derived from Algorithm 4 can be fused into the prior mesh by intersecting rays from the camera frame. For a particular camera pose, we choose a mesh vertex, $\mathcal{V}_n[i]$, and then compute the intersection of the camera-to-vertex ray as $\mathbf{p}^i$. The various intersections at different camera poses are fused into a new mesh, $\mathcal{M}_{\text{new}}$, using a moving-average filter.

Algorithm 5 Fuse shapes into prior mesh

1: $\mathcal{M}_{\text{new}} = \mathcal{M}_{\text{prior}}$ // Make a deep copy of the prior mesh
2: **for** pose in poses **do**
3: $\mathcal{F}_v = \texttt{is\_visible}(\text{pose}, \mathcal{F})$ // Visible features
4: $\mathcal{S}_d = \texttt{Algorithm1}(\text{pose}, \mathcal{F}_v, \mathcal{M}_{\text{prior}})$ // Detected shapes
5: $\mathcal{V}_n = \texttt{nearby\_vertices}(\text{pose}, \texttt{vertices}(\mathcal{M}_{\text{prior}}))$
6: **for** shape $\mathcal{M}_s$ in $\mathcal{S}_d$ **do**
7: **for** vertex $\mathbf{v}^i \in \mathcal{V}_n$ indexed by i **do**
8: ray = $\texttt{make\_ray}(\text{pose}, \mathcal{V}_n[i])$
9: **if** ray.$\texttt{intersects\_with}(\mathcal{M}_s)$ **then**
10: $\mathbf{p}^i$ = ray.$\texttt{intersection}(\mathcal{M}_s)$
11: $\texttt{moving\_avg}(\mathcal{M}_{\text{new}}, i, \mathbf{p}^i)$ // Changes the location of i^{th} vertex using Eqn. (4.15)
12: **end if**
13: **end for**
14: **end for**
15: **end for**
16: **return** $\mathcal{M}_{\text{new}}$

moving average filter:

$$\widehat{\mathbf{v}}^i_{n+1} = \frac{\mathbf{p}^i + n\widehat{\mathbf{v}}^i_n}{n+1} \tag{4.15}$$
$$\widehat{\mathbf{v}}^i_0 = \texttt{get\_ith\_vertex}(\mathcal{M}_{\text{prior}}, i),$$

where the i^{th} vertex's counter, n, is incremented after every evaluation of line 11 from Algorithm 5.

This process is repeated for every pose. A summary of this algorithm is provided in Algorithm 5. Note that we conservatively and efficiently determine the visible features and nearby vertices (lines 3 and 5 of Algorithm 5), using a k-dimensional (KD) tree-based radius search.

4.4 Results

The field data used in our experimental evaluation is taken from the Bluefin Robotics HAUV surveying the *SS Curtiss*. We provide an overview of the sensor configuration and characteristics of the surveyed vessel in Section 1.1.2. A 3D triangular mesh, shown previously in Fig. 4.1 and Fig. 4.2, was derived from computer aided design (CAD) drawings, and serves as the prior model in our model-assisted framework. The size of the bundle adjustment problem is shown in Table 4.1.

In this section, we evaluate the performance of three approaches when processing this single large dataset:

	2011 (monocular)	2014 (stereo)
HAUV trajectory length	1,828 m	4,661 m
Number of images	9,249	44,868
Number of DVL raycasts	36,996	96,944
Number of feature reprojections	201,194	974,144
Number of features	100,597	243,536

Table 4.1: Size characteristics of the 2011 field data (taken from a monocular camera) and the 2014 field data (taken with a stereo rig).

1. A naive BA framework where the measurements consists of $\mathcal{Z}_{\text{prior}}$, $\mathcal{Z}_{\text{odo}}$, $\mathcal{Z}_{\text{range}}$, and $\mathcal{Z}_{\text{feat}}$. All surface constraints are disabled, i.e., $\lambda_i = 0$ for every feature.

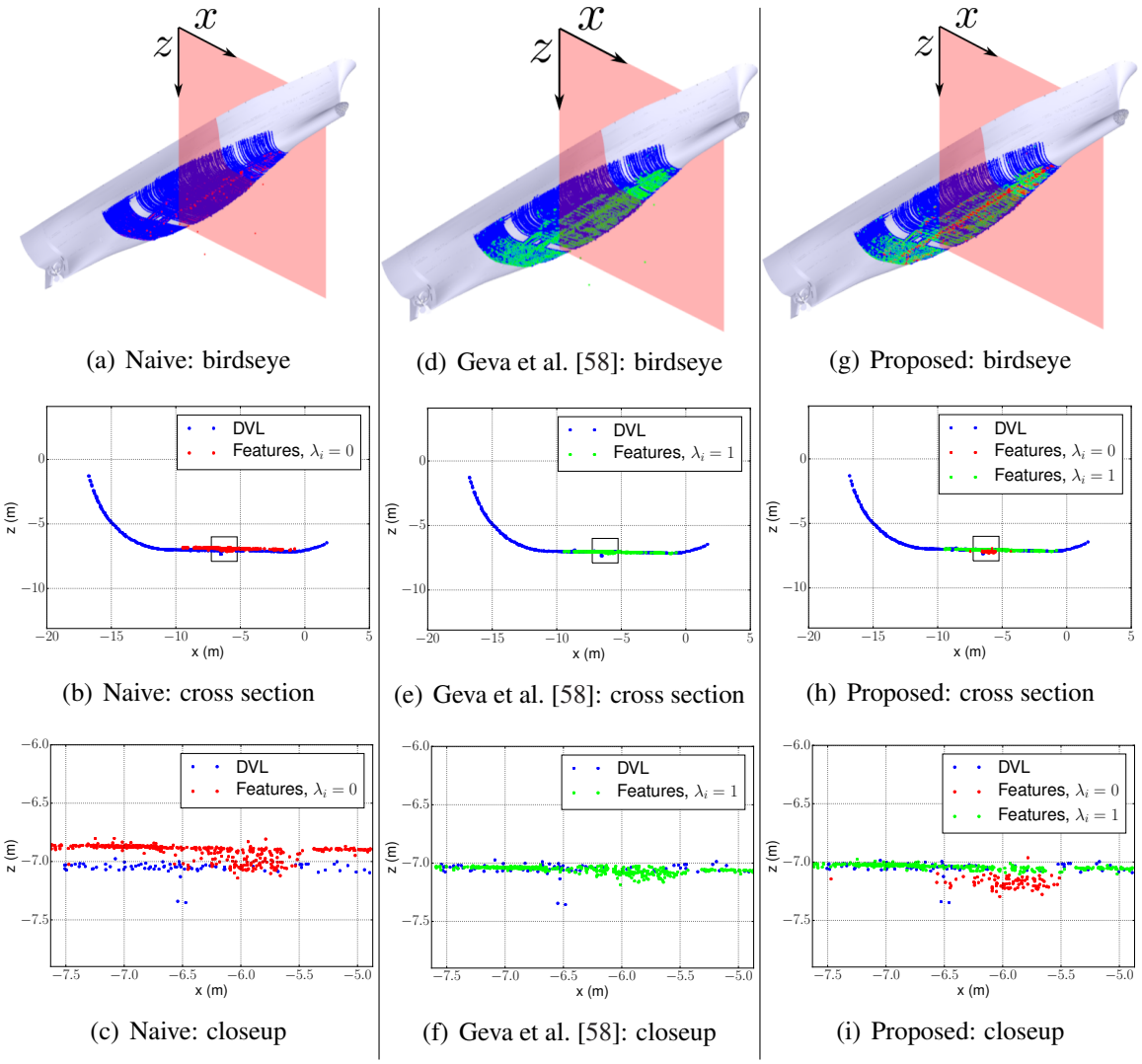
2. The approach based on Geva et al. [58], which consists of the measurements $\mathcal{Z}_{\text{prior}}$, $\mathcal{Z}_{\text{odo}}$, $\mathcal{Z}_{\text{range}}$, and $\mathcal{Z}_{\text{feat}}$, in addition to surface constraints, $\mathcal{Z}_{\text{surf}}$, such that $\lambda_i = 1$ for every feature.

3. The proposed algorithm discussed in Section 4.2.2, implemented using Gaussian max-mixtures, where each hidden label λ_i is assigned from (4.7).

4.4.1 Model-Assisted BA

We provide a visualization of the reconstruction that highlights the advantages of our approach in Fig. 4.10 and Fig. 4.11. These plots show cross sections of the ship hull, from starboard-to-port, to highlight the relevant portions of the visual features and range returns from the DVL. Because of the raycasting factors from (4.14), these range returns act as a proxy for the true profile of the hull form. We achieve this as follows: in the top rows of Fig. 4.10 and Fig. 4.11, we plot a narrow cross-sectional 3D rectangle in pink. In the second row, we only plot the DVL and visual features that are contained within this pink rectangle. Finally, the third row shows a zoomed-in portion of the second row to highlight small-scale details.

Fig. 4.10 shows the reconstruction using 2014 field data, which uses an underwater stereo camera. The portion of the ship hull that is captured in the bottom row of Fig. 4.10 corresponds to the biofouling detection shown in Fig. 4.1. Fig. 4.11 is taken from a 2011 survey, during which the HAUV was equipped with a monocular underwater camera. This scene corresponds to a cylindrical shape with known dimensions. This shape will be shown later in Fig. 4.18. The factor graph representations (discussed in Section 4.2.6) between the stereo and monocular datasets are quite similar, except that the reprojection error is

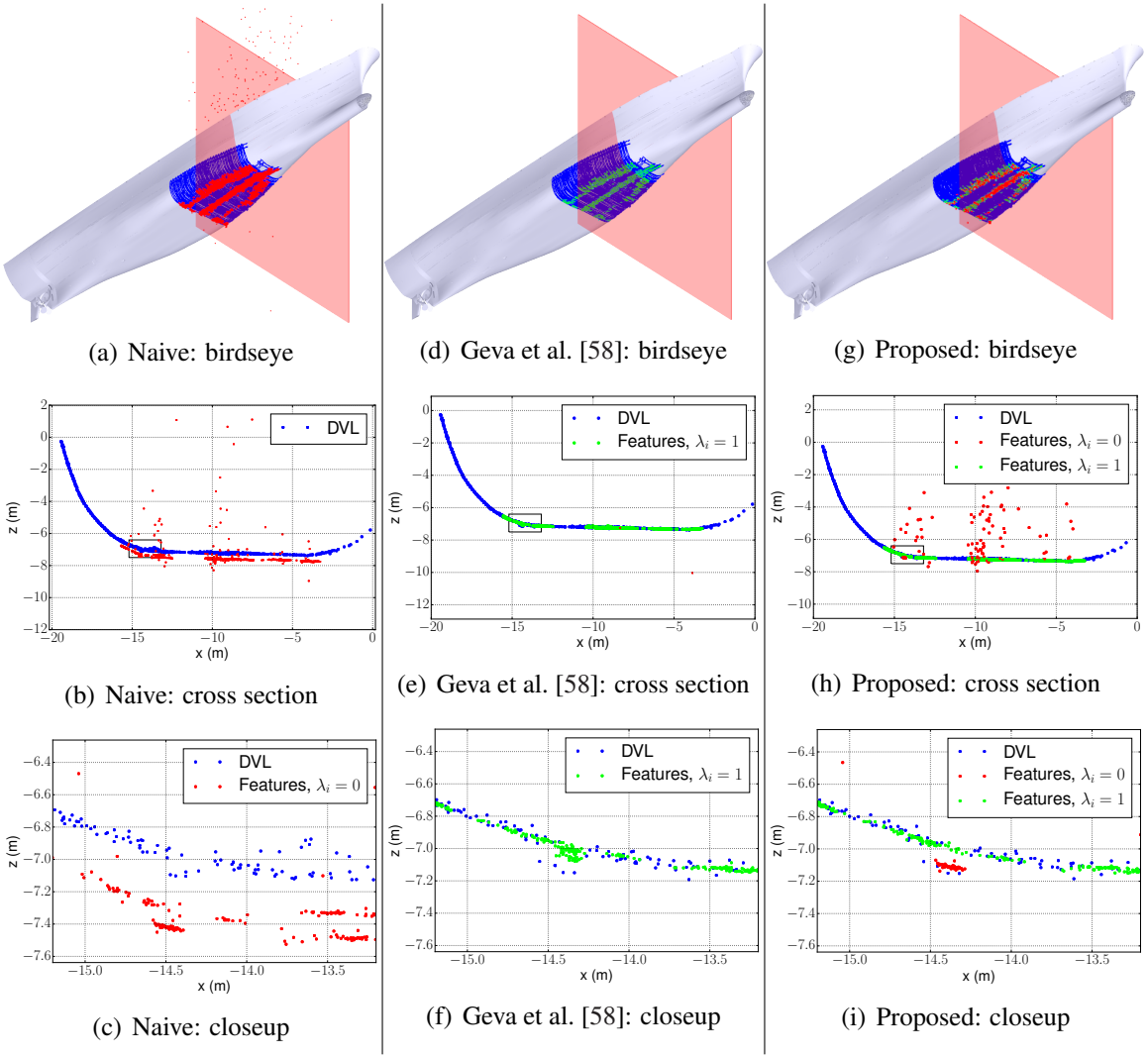


(a) Naive: birdseye (d) Geva et al. [58]: birdseye (g) Proposed: birdseye

(b) Naive: cross section (e) Geva et al. [58]: cross section (h) Proposed: cross section

(c) Naive: closeup (f) Geva et al. [58]: closeup (i) Proposed: closeup

Figure 4.10: Comparison of different BA approaches using the 2014 stereo dataset. Each column represents a different method, with each column showing the same representative cross section of the reconstruction. For the naive approach shown in (a) through (c), there are no factors constraining the visual features and prior model, resulting in a noticeable misregistration with the DVL. Using the method from Geva et al. [58], shown in (d) through (f), the features and DVL ranges are well-registered, but the biofouling is visibly "squished" onto the surface. Our method, shown in (g) through (i), combines the favorable qualities of each method, aligning the visual features and DVL returns while also preserving 3D structure that is absent from the prior model (i.e., the red points in (i)).

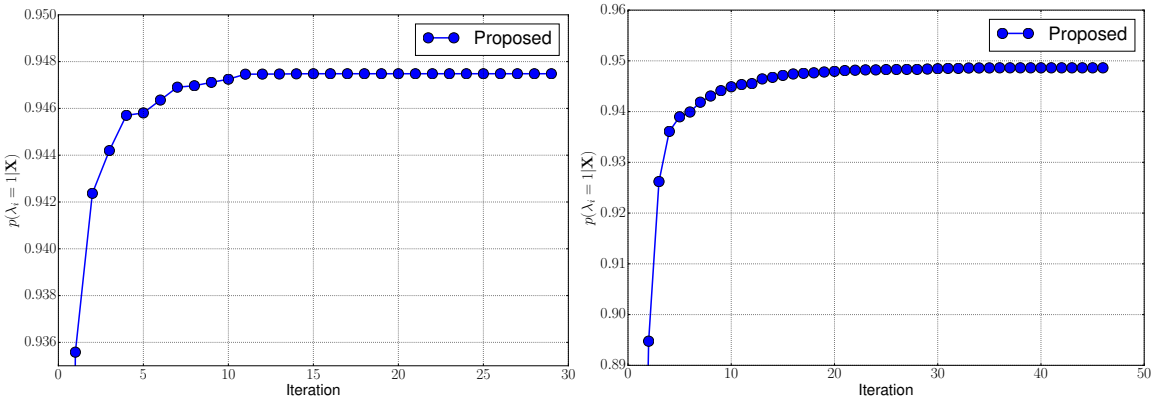


(a) Naive: birdseye (d) Geva et al. [58]: birdseye (g) Proposed: birdseye

(b) Naive: cross section (e) Geva et al. [58]: cross section (h) Proposed: cross section

(c) Naive: closeup (f) Geva et al. [58]: closeup (i) Proposed: closeup

Figure 4.11: Comparison of different BA approaches using the 2011 monocular dataset. The general trends in this result are nearly identical to the trends shown in Fig. 4.10. In this case, the known cylindrical shape's 3D structure is preserved in our method (shown in (i)), while it is *not* well-preserved using the approach from Geva et al. [58] (shown in (f)).



(a) 2014 (stereo) (b) 2011 (monocular)

Figure 4.12: Relative frequency of all features with $\lambda_i = 1$ as a function of the current least-squares iteration. The left and right plots corresponds to the right columns of Fig. 4.10 and Fig. 4.11, respectively. Note the difference in scale for each plots y-axis, and that the first several datapoints were omitted to highlight the subtle changes in subsequent iterations. The initial probabilities for (a) and (b) are 0.82 and 0.34, respectively.

computed using (4.12) in the case of a stereo camera, and (4.13) in the case of a monocular camera.

In these figures, we see some general trends.

1. In the reconstruction derived from the naive approach, the visual features do not lie on the same surface as the range returns from the DVL. The features are underconstrained in the naive case because there is zero information (i.e., connected factors) between the pose of the prior model and the position of the visual features.

2. Using the approach from Geva et al. [58], the visual features lie on the same surface as the DVL range returns, as expected. Because *all* features are constrained to lie on the surface, the algorithm does not capture 3D structure present on the actual ship hull that is not present in the prior model (e.g., the docking blocks along the bottom of the hull, or manually-placed cylindrical shapes).

3. Our approach combines the benefits of both approaches: the visual features and DVL-derived point cloud lie on the same surface, and our visual reconstruction yields 3D structure that would have been heavily regularized using the approach from Geva et al. [58].

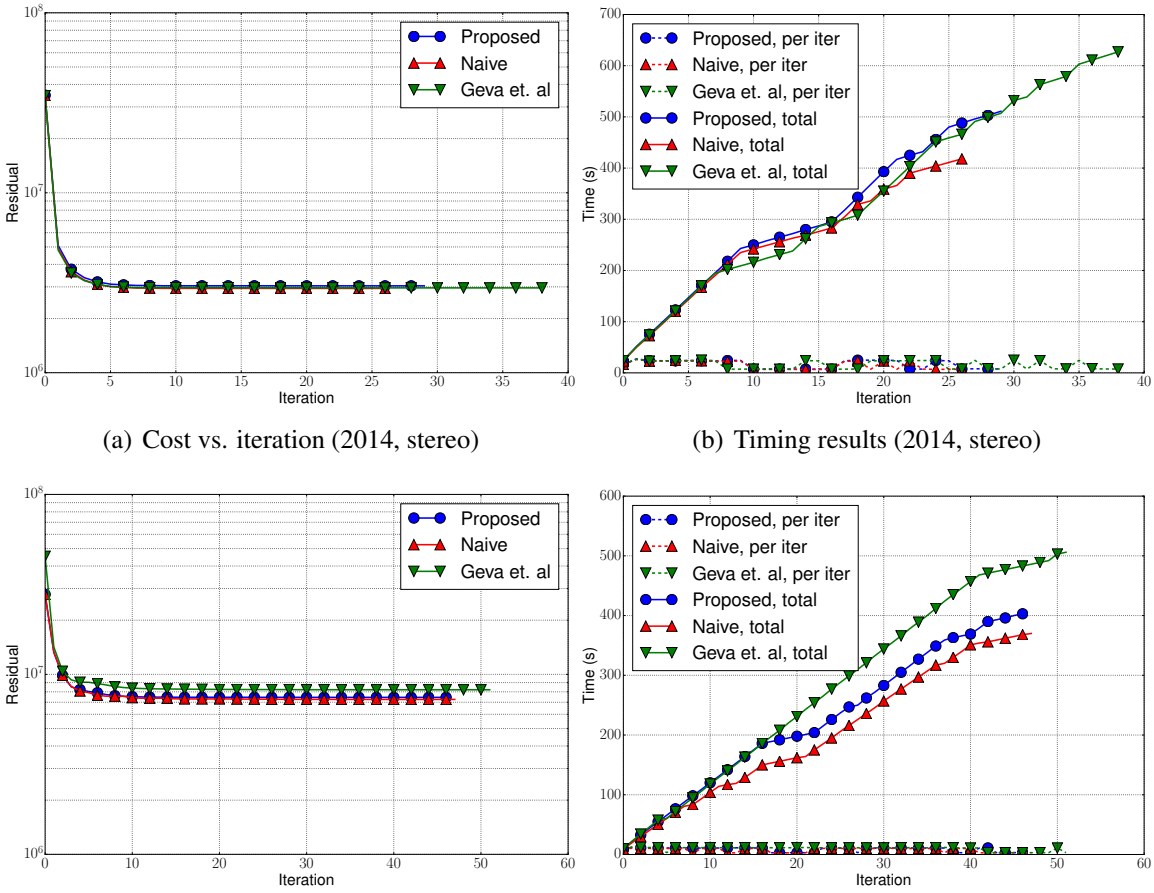
In addition, there exists several outlier features in the sparse monocular reconstruction compared to the stereo reconstruction, as shown in Fig. 4.10(h) and Fig. 4.11(h). This is

not surprising; the geometric verification step is relatively weak for monocular cameras as compared to stereo (as briefly mentioned in Section 4.2.7). This has little to no effect on our model-assisted BA framework; because they are outliers (i.e., the lie far from the surface of the prior mesh) they impose very little cost when evaluating the surface constraint likelihood from (4.4).

The results in Fig. 4.12 suggest the identification of feature labels stabilizes after about twenty iterations. Clearly, for this application, the vast majority of features lie on the prior model, suggesting that the weights in each mixture (or, equivalently, the last multiplicand in (4.1)) can be optionally tuned to reflect this trend, rather than assigning an uninformative prior probability for $p(\lambda_i = 1)$. However, we prefer the latter approach because it is more appropriate for conservatively detecting foreign objects (an important capability for automated ship hull inspection).

We assessed the computational performance by performing timed trials on a consumer-grade four-core 3.30 GHz processor. We report the timing results of each of the three methods in Fig. 4.13. From this plot we draw two conclusions: (*i*) the computational costs of our method impose total performance loss of 22.3% (stereo) and 8.9% (monocular) compared to the naive approach; (*ii*) the computational costs of the approach from Geva et al. [58] imposes a performance loss of 50.0% (stereo) and 36.8% (monocular) compared to the naive approach. This behavior can be explained by the optimizer having to perform more iterations until convergence for the approach from Geva et al. [58]. Intuitively, by forcing visual features that protrude from the prior model to lie flush, the optimizer must perform more iterations to satisfy the corresponding reprojection errors. Even though our method devotes additional processing time when evaluating (4.7), this is overcome by the added cost performing additional iterations.

Finally, if we examine the consistency of DVL range measurements, we can see a noticeable improvement using our model-assisted BA framework. From Fig. 4.14(a) and Fig. 4.14(b), we can see several inconsistencies particularly on the side of the hull. By examining the distribution of residuals in Fig. 4.14(c), we quantitatively confirm that the alignment taken from GICP alone yields a relatively large error distribution of DVL returns, however our model-based BA framework can significantly tighten the distributions of these residuals. The main source of error is the slop in the servo that rotates the DVL. By solving for this angle, we can have a much tighter distribution of error, as shown in Fig. 4.14(c).



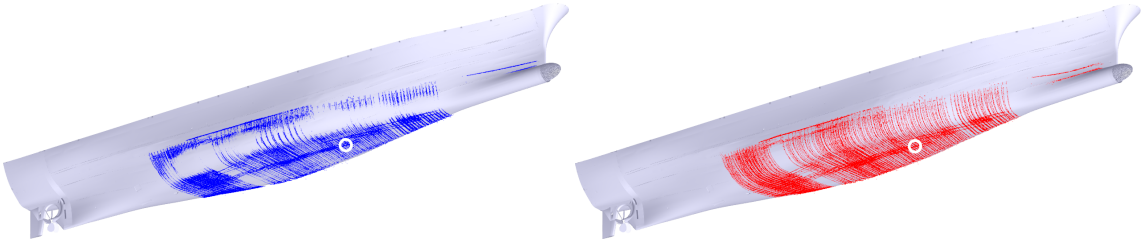
(a) Cost vs. iteration (2014, stereo)

(b) Timing results (2014, stereo)

(c) Cost vs. iteration (2011, monocular)

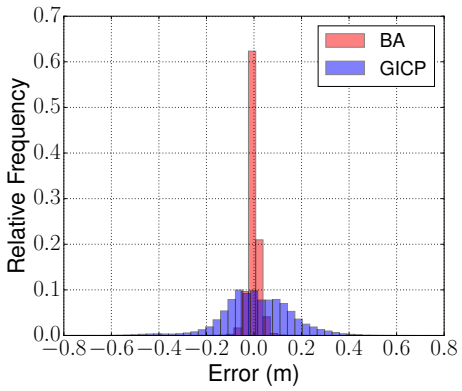
(d) Timing results (2011, monocular)

Figure 4.13: Computational comparison for each method. Our algorithm imposes some extra computation time compared to the naive approach, but the overall execution time is comparable. For the stereo dataset, the naive, Geva et al. [58], and proposed approaches converged in 26, 38, and 29 iterations, respectively. For the monocular dataset, each approach converged in 47, 46, and 51 iterations, respectively.



(a) DVL consistency: GICP only

(b) DVL consistency: BA



(c) DVL beam residuals: (a) vs (b)

Figure 4.14: The distribution of residual error for DVL range returns has a significantly smaller variance using our model-assisted BA framework.

4.4.2 Model-Assisted Mapping

Our evaluation of our model-assisted mapping pipeline consists of a metric evaluation of the structures detected from Algorithm 4 and the utility of the remeshed prior model from Algorithm 5. These will be discussed in Section 4.4.2.1 and Section 4.4.2.2, respectively.

4.4.2.1 Shape Detection and 3D accuracy

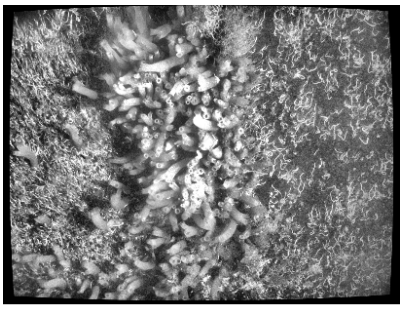
Illustrative examples of our shape detection approach from Algorithm 4 are shown in Fig. 4.15 (stereo) and Fig. 4.16 (monocular). The former shows biofouling emanating from a docking block, while the later shows a manually-placed cylindrical shape. These examples correspond to the cross-sections shown in Fig. 4.10 and Fig. 4.11. The features contained in the shape detected in Fig. 4.15(b) have deviations ranging between 0.06 m and 0.18 m. Though we do not have ground-truth, we can use a dense stereo matching algorithm to get a rough sense of how much the biofouling protrudes compared to the rest of the scene. In Fig. 4.15(c), we see that the biofouling is about 0.10 m to 0.20 m closer to the camera than the rest of the scene. Though this comparison is relatively coarse, this serves as a promising indication that the 3D structure inferred using our approach is reasonably indicative of ground-truth.

The monocular example we show in Fig. 4.16(a) indicates a cluster of features centered on the cylindrical shape with a mean deviation of 0.10 m to the CAD model. In this case we know that the ground-truth height of the cylindrical shape is 0.11 m, which suggests a reconstruction error of approximately 0.01 m. In addition, in this figure there are several outlying features (their corresponding red colors were capped at 0.18 m). Because DBSCAN requires a minimum number of datapoints per cluster (in our case, we choose 3), these features are not detected as foreign shapes.

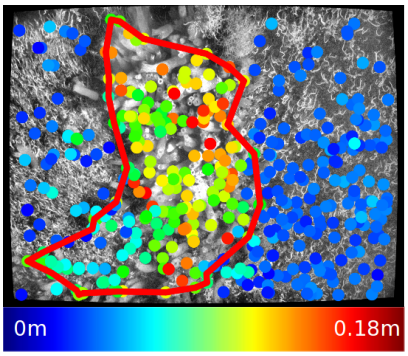
4.4.2.2 Remeshing Results

Algorithm 5 provides us with a remeshed CAD model for a visual survey of the *SS Curtiss*, which is shown in Fig. 4.17. We present results for two different values of τ, the threshold used for determining the eligibility of features to be clustered with DBSCAN. For this application, the preferred approach is to keep the threshold zero (Fig. 4.17(b)), however for certain applications where false positives are a concern, this can be raised (Fig. 4.17(c)). Like the other visualizations for this dataset, rectangular-like foreign 3D structures shown in Fig. 4.17(c) correspond to biofouling along the centerline.

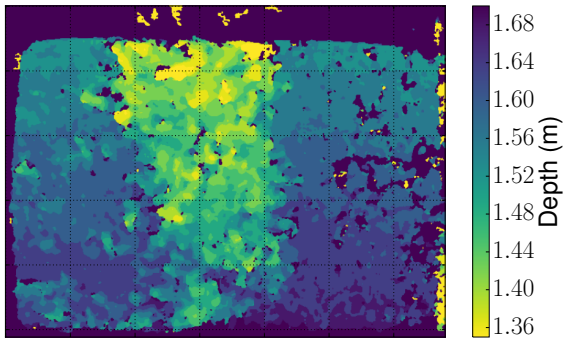
In addition to providing visually intuitive false color maps, the remeshed model can easily be used in a start-of-the-art 3D photomosaicing framework [83]. An example of this



(a) Sample raw image

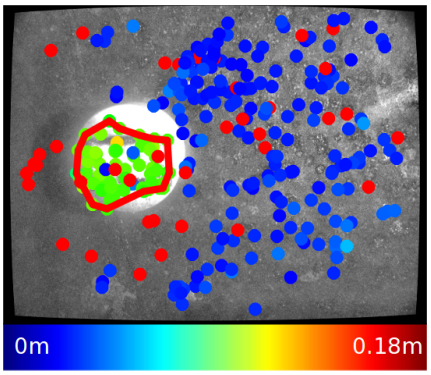


(b) Shape detection (red)

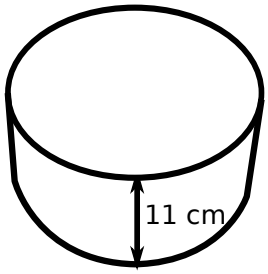


(c) Camera-relative depth (dense stereo)

Figure 4.15: Shape detection example from the 2014 stereo dataset. In this example, the foreign object corresponds to biofouling along the ship hull centerline.

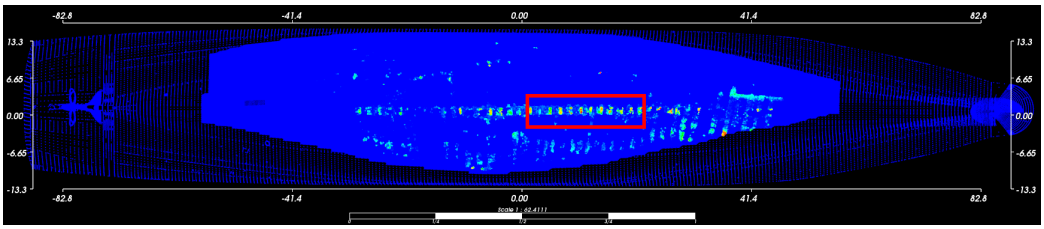


(a) Shape detection (red)

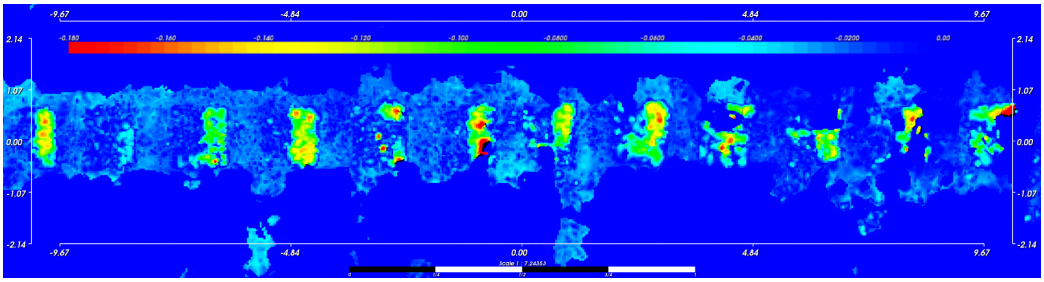


(b) Ground-truth height

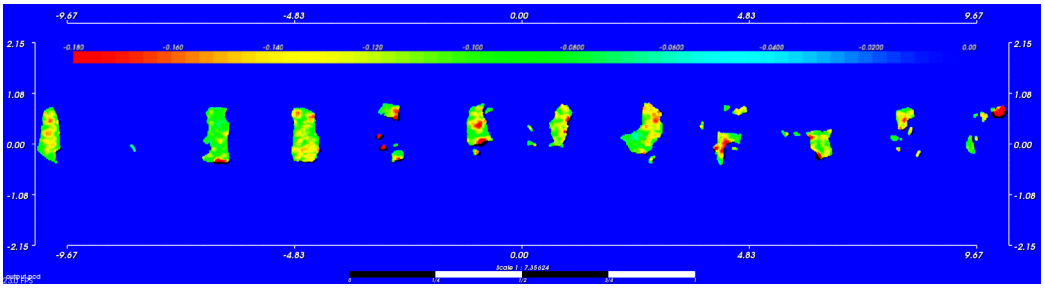
Figure 4.16: Shape detection example from the 2011 monocular dataset. There is approximately 1 cm of error between the height of the detected 3D structure (i.e., green cluster in (a)) and the ground-truth height of the cylindrical shape in (b).



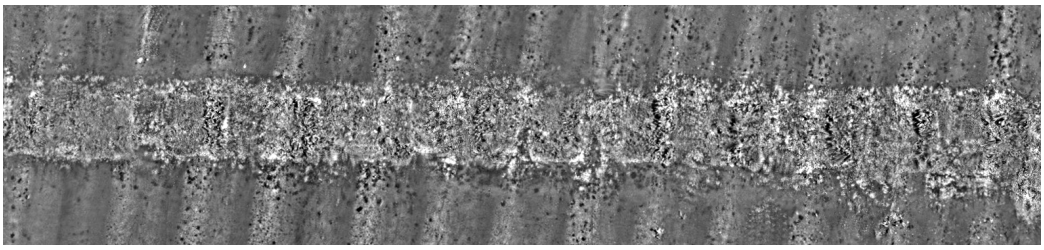
(a)



(b) $\tau = 0.0$ m

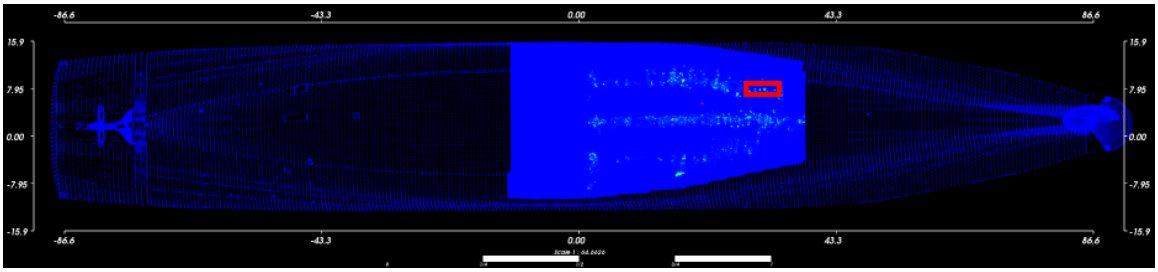


(c) $\tau = 0.06$ m

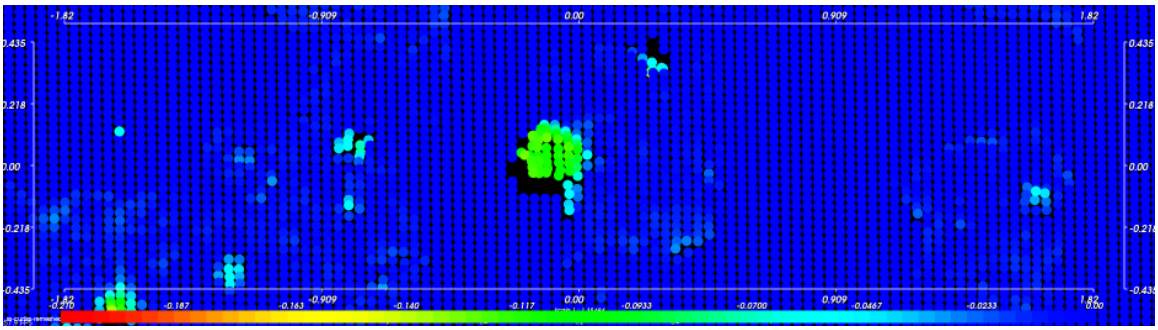


(d) Portion of photomosaic corresponding to (b) and (c)

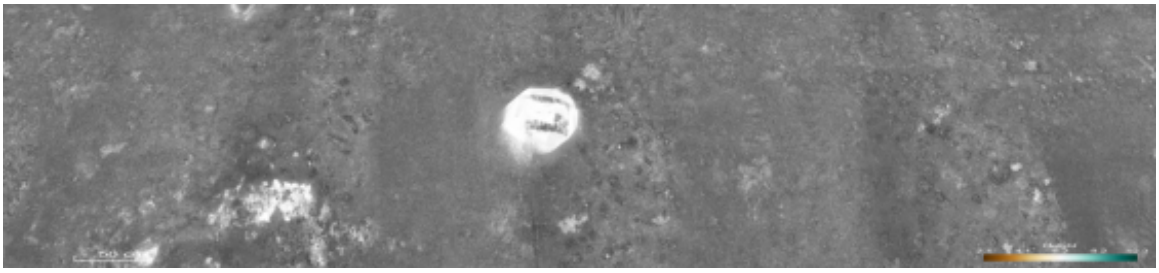
Figure 4.17: Overview of remeshed CAD using stereo. In (a), we show a heatmap of the remeshed CAD vertices. The red region is expanded in (b) and (c). In (b), the clustering threshold from Algorithm 5, τ, is zero while in (c) it is relatively high. Lowering τ provides more details but potentially introduces false positives. We can see the red rectangular region from (a) corresponds to a strip of biofouling at the ship's centerline, shown in (d).



(a)

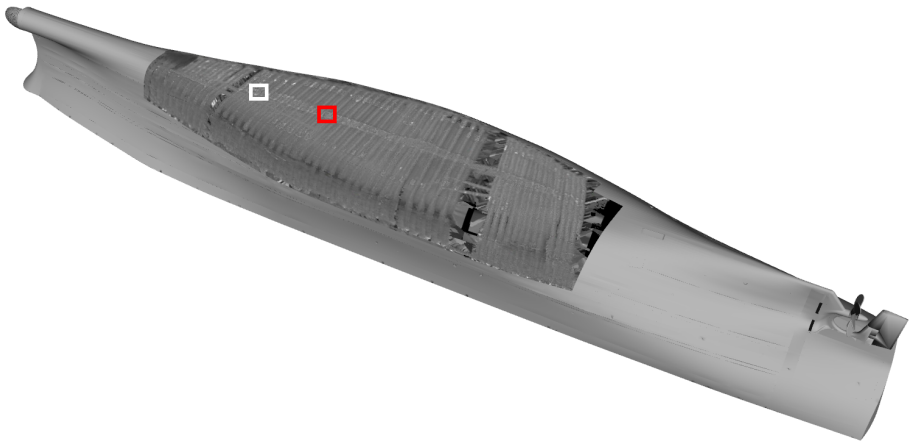


(b) $\tau = 0.0$ m

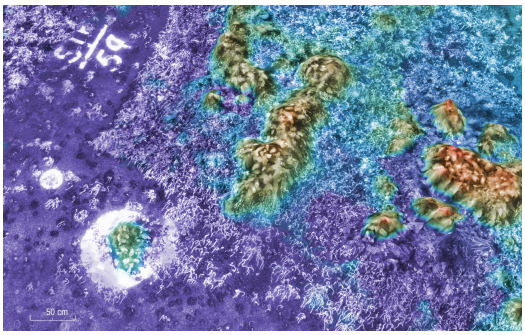


(c) Portion of photomosaic corresponding to (b)

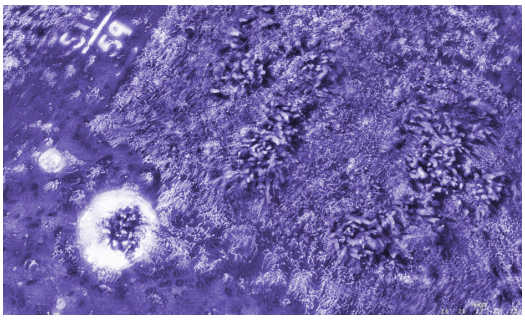
Figure 4.18: Overview of remeshed CAD using a monocular camera. The red region in (a) is expanded in (b), were we show the 3D structure corresponding to the cylindrical shape shown in Fig. 4.16. In (c) we show the corresponding region in 3D photomosac.



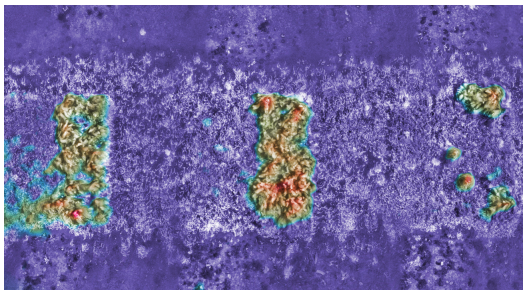
(a) Birds-eye view: (b) and (c) correspond to white region, (d) and (e) correspond to the red region.



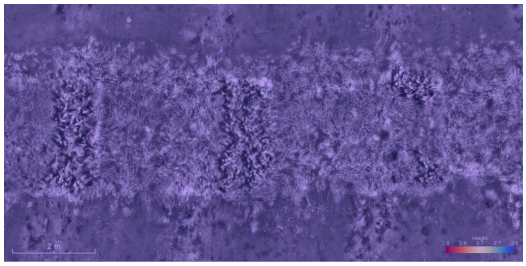
(b) Close-up: proposed method



(c) Close-up: method from [138]



(d) Close-up: proposed method



(e) Close-up: method from [138]

Figure 4.19: Application to large-scale 3D photomosaicing. Our approach allows 3D photomosaicing approaches to combine large-scale consistency in (a) with small-scale detail in (b) and (d). In (b) and (d), the mosaic is shaded according to height. Using the approach from [138], where a CAD model is used for photomosaicing, the small-scale details are lost as evidenced by the regions in (c) and (e) being near-perfectly flat.

application is provided in Fig. 4.19. Unlike our previous work in [138] that applied texture to the ship's CAD model, this work allows additional structural details at a relatively small scale. In Fig. 4.19, we shade the regions of the 3D photomosaic according to height in the z-direction. Clearly, the approach proposed in this chapter captures significantly more information that is otherwise discarded if the ship hull is assumed to match the CAD model shape exactly.

4.5 Conclusion

We proposed a model-assisted bundle adjustment framework that assigns binary labels to each visual feature. Using an EM algorithm with hard hidden variable assignments, we iteratively update these variables and update the current state estimate. We show that this algorithm is a special case of the Gaussian max-mixtures framework from earlier work in robust pose graph optimization. We compared our approach to recent work in model-assisted methods, and showed our algorithm has favorable properties when evaluated in the context of autonomous ship hull inspection.

In addition, we propose a shape identification and mapping algorithm that provides precise capabilities for identifying visually-observed 3D structure that is absent from the CAD model. The mapping algorithm fuses these shapes into the prior mesh, resulting in a newly remeshed model. This newly remeshed model has several important benefits for data visualization. In particular, the false-color figures shown in this chapter offer an intuitive visualization that is easier to discern than from image mosaics alone. In addition, the remeshed model can easily be used in a 3D photomosaicing framework such that the overall consistency of the ship hull reconstruction is preserved, but captures details at a small scale.

We evaluated these techniques using field data collected from the HAUV hull inspection robot—the largest model-assisted BA evaluation to date. We have shown that our BA framework introduces only slight computational overhead, while producing accurate reconstructions for both stereo and monocular camera sensors.

CHAPTER 5

Conclusions and Future Work

This thesis incorporates surface information derived from man-made structure into an in-water underwater inspection system for real-time surveying and offline three-dimensional (3D) reconstruction using the hovering autonomous underwater vehicle (HAUV). We have shown several contributions that leverage this information in the context of the simultaneous localization and mapping (SLAM) problem; both in real-time and post-processing. In this chapter, we succinctly summarize the contributions of this thesis and describe additional directions to further pursue our work.

5.1 Contributions

The specific contributions of this thesis include:

- We have developed a factor graph representation of a smooth surface as a collection of many planar features. By assuming coarse prior knowledge of the curvature, we can constrain planar patches that *do not overlap* with piecewise-planar factor potentials to achieve real-time performance in an underwater SLAM system. We have shown that this representation can be kept computationally tractable in the long-term using the recent generic linear constraints (GLC) framework for sparse-approximate graph marginalization. To our knowledge, our minimal plane parameterization is the first to support a full-rank information matrix within a factor graph representation of SLAM, allowing for incremental updates with incremental smoothing and mapping (iSAM), fast batch optimization algorithms like Gauss-Newton, and fast covariance recovery algorithms.
- We have demonstrated the first-ever 3D photomosaicing framework using 2D imaging sonar data and range returns from a Doppler sonar. These range returns are converted into a triangular surface mesh using Delaunay triangulation. This mesh acts as a surface onto which imagery can be back-projected.

- We have developed a novel model-assisted bundle adjustment (BA) framework that induces strong correlation between visually-derived structure and surface information observed from a Doppler sonar while preserving 3D structure that is not present in the prior model. Our framework is an expectation-maximization (EM) algorithm that is a special case of the Gaussian max-mixtures (GMM) framework (originally developed for robust odometry and automatic loop closure rejection).

5.2 Future Work

This thesis lays a strong foundation for additional work in underwater perception and applied autonomous underwater vehicle (AUV) research. In this section, we briefly summarize several improvements to our autonomous underwater SLAM framework that merit additional research.

Real-time SLAM capabilities

The real-time SLAM system we developed in this thesis is a *passive* process; the robot makes no effort to intelligently plan a path so as to balance the trade-off between coverage rate and navigational uncertainty. *Active SLAM* has seen considerable attention in the research community, including experimental evaluation using the HAUV platform [30, 31, 97]. The focus of this work is in acquiring camera measurements, however as-of-yet there has been little work to incorporate planar information in an active SLAM framework despite the significant navigational aid we demonstrated in this thesis.

Global feature registration with sonar

The photomosaicing work from Chapter 3 currently does not establish feature correspondences between views; rather, it mainly uses range returns from the Doppler velocity log (DVL) as in Chapter 2 or, if available, camera information. Because the Dual frequency IDentification SONar (DIDSON) sonar is servoed independently of the DVL and camera, we must treat the actuator angle as known.

If feature correspondences are known, feature reprojections can be minimized in a similar fashion to BA. In fact, recent work from Huang and Kaess [73] has explored this technique (the so-called acoustic structure from motion (ASFM) problem) using the same reprojection model used in Chapter 3 with manual feature correspondences. If establishing feature correspondences were automated, we expect significant improvements in the consistency of the sonar mosaics presented in this thesis, including the potential capability of inferring

3D information. In addition, the model-asisted methods proposed in this thesis are likely applicable to the ASFM problem.

Feature learning in sonar images

The long-term, multi-session capability we introduced in Chapter 2 coupled with the robust large-scale BA capabilities in Chapter 4 allow for approximate ground-truthing through offline processing. These capabilities introduce large collections of spatially-registered data, which have recently received some attention in the machine learning community. In particular, convolutional neural networks (CNNs) have enabled researchers to train descriptors that capture image similarity on large training sets, rather than hand-crafted features such as scale-invariant feature transform (SIFT), speeded up robust features (SURF), and others [105, 161]. The work in this thesis has provided several such datasets, and we expect significant efforts in learning features for sonar, camera, and perhaps registration *between* the two modalities.

APPENDIX A

Transformations of Points in 3D

This appendix briefly introduces the conventions for coordinate frame composition and affine transformations used throughout this thesis. Let $\mathbf{x}_{ij}$ be the 6-degree-of-freedom (DOF) pose of frame j with respect to frame i following the conventions of Smith, Self, and Cheeseman [158].

$$\mathbf{x}_{ij} = \left[{}^{i}\mathbf{t}_{ij}{}^{\top}, \mathbf{\Theta}_{ij}^{\top}\right]^{\top} = [x_{ij}, y_{ij}, z_{ij}, \phi_{ij}, \theta_{ij}, \psi_{ij}]^{\top}.$$

Here, ${}^{i}\mathbf{t}_{ij}$ is a translation vector from frame i to frame j as expressed in frame i, and ϕ_{ij}, θ_{ij}, ψ_{ij} are the Euler angles about the x, y, and z axes, respectively.

The rotation matrix ${}^{i}_{j}\mathrm{R}$ rotates frame j into frame i. Converting from $\mathbf{\Theta}_{ij}$ to a rotation matrix requires defining an *Euler rotation sequence*, which selects the order to apply rotations to the x, y, and z axes. In this thesis, we adopt the following sequence for all of our rotations:

$$\begin{aligned}
{}^{i}_{j}\mathrm{R} &= \mathrm{rotxyz}(\mathbf{\Theta}_{ij}) \\
&= \mathrm{rotz}(\psi_{ij})^{\top}\,\mathrm{roty}(\theta_{ij})^{\top}\,\mathrm{rotx}(\phi_{ij})^{\top} \\
&= \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}^{\top} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}^{\top} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}^{\top} \\
&= \begin{bmatrix} \cos\psi\cos\theta & -\sin\psi\cos\phi + \cos\psi\sin\theta\sin\phi & \sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi \\ \sin\psi\cos\theta & \cos\psi\cos\phi + \sin\psi\sin\theta\sin\phi & -\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}.
\end{aligned}$$

Therefore, to transform a three-dimensional (3D) point expressed in frame j, ${}^{j}\mathbf{X}$, to a 3D point expressed in frame i, ${}^{i}\mathbf{X}$, we first rotate frame j into frame i, then translate from i

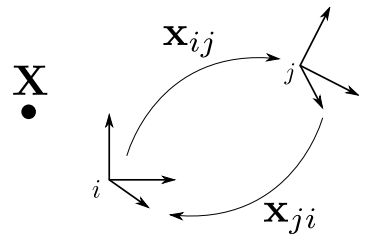


Figure A.1: Illustration of coordinate frames relationships. The 6-DOF relative-pose between two coordinate frames, i and j, are related by either $\mathbf{x}_{ij}$ or $\mathbf{x}_{ji}$, depending on the frame of reference. The 3D point $\mathbf{X}$ can easily be represented with respect to either i or j using the corresponding affine transformation matrix.

to j as expressed in i. More compactly:

$$ {}^{j}\mathbf{X} = {}^{i}_{j}\mathrm{R}\mathbf{X}_i + {}^{i}\mathbf{t}_{ij}. $$

We can adopt homogeneous coordinates to represent this as a linear relationship using an affine transformation matrix ${}^{i}_{j}\mathrm{H}$:

$$ \begin{aligned} {}^{i}\mathbf{X} &= \left[{}^{i}_{j}\mathrm{R} \,|\, {}^{i}\mathbf{t}_{ij}\right] {}^{i}\underline{\mathbf{X}} \\ {}^{i}\underline{\mathbf{X}} &= \begin{bmatrix} {}^{i}_{j}\mathrm{R} & {}^{i}\mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix} {}^{j}\underline{\mathbf{X}} \\ {}^{i}\underline{\mathbf{X}} &= {}^{i}_{j}\mathrm{H}\,{}^{j}\underline{\mathbf{X}}, \end{aligned} $$

where normalized homogeneous vectors are denoted using $\underline{\mathbf{v}} = [\mathbf{v}^\top, 1]^\top$ for some column vector $\mathbf{v}$. In addition, one can go the opposite direction: suppose we wish to express ${}^{i}\mathbf{X}$ with respect to frame j:

$$ \begin{aligned} {}^{j}\underline{\mathbf{X}} &= {}^{i}_{j}\mathrm{H}^{-1}\,{}^{i}\underline{\mathbf{X}} \\ &= {}^{j}_{i}\mathrm{H}\,{}^{i}\underline{\mathbf{X}} \\ &= \begin{bmatrix} {}^{j}_{i}\mathrm{R} & {}^{j}\mathbf{t}_{ji} \\ \mathbf{0}^\top & 1 \end{bmatrix} {}^{i}\underline{\mathbf{X}} \\ &= \begin{bmatrix} {}^{i}_{j}\mathrm{R}^\top & -{}^{i}_{j}\mathrm{R}^\top\,{}^{i}\mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix} {}^{i}\underline{\mathbf{X}} \end{aligned} $$

This operation is illustrated in Fig. A.1.

There are well-known closed-form expressions for linearizing the relation between 6-DOF pose form and the corresponding affine transformation matrix [49, 158]. However, for

the work in this thesis we use either automatic [3] or numerical differentiation for simplicity at the cost of slight performance degradation.

APPENDIX B

Planar Parameterization for SLAM

The choice of planar parameterization is important for simultaneous localization and mapping (SLAM) applications because fast optimization algorithms, incremental updates, and efficient covariance recovery algorithms require a full-rank parameterization of all unknowns [86, 88]. In this chapter we identify several parameterizations of planes in 3D and motivate the parameterization used in this thesis.

Regardless of the underlying parameterization, the notation used to denote the frame of reference for a given plane is as follows. The frame of reference (or "origin") of the plane is denoted with a subscript, e.g. the plane, indexed by k, as expressed with respect to some frame i is denoted $\boldsymbol{\pi}_{ik}$. If one wishes to express a plane in a different frame of reference, one may apply the $\boxminus$ and $\boxplus$ operators, which are defined in Section B.5. For example, given $\boldsymbol{\pi}_{ik}$ and $\mathbf{x}_{ij}$, the plane k may be expressed with respect to frame j as follows:

$$\boldsymbol{\pi}_{jk} = \mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik}.$$

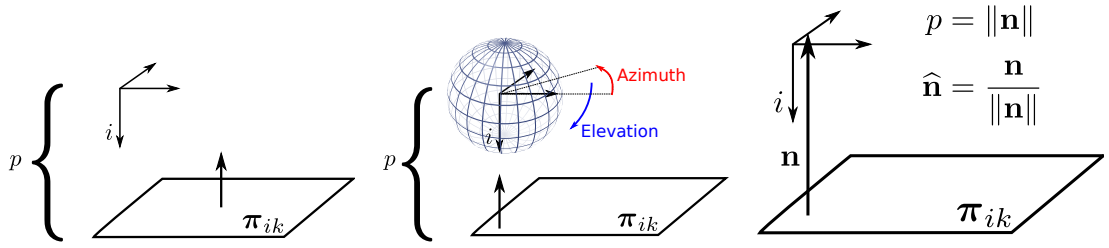
Similarly, given $\boldsymbol{\pi}_{jk}$ and $\mathbf{x}_{ji}$, the plane k may be expressed with respect to frame j as:

$$\boldsymbol{\pi}_{ik} = \mathbf{x}_{ij} \boxplus \boldsymbol{\pi}_{jk}.$$

All of the parameterizations illustrated in Fig. B.1 encode a surface normal of the plane. The convention used in this thesis has the normal extending *from* the plane's surface *to* the coordinate frame in which it is represented, i.e., the surface normal points *outward*, as opposed to *inward*.

B.1 Hessian Normal Form

By far the most common parameterization of planes in 3D is the so-called *Hessian normal* form. Using this form, a plane is parameterized by four numbers: the unit-normal of the



(a) Hessian normal (b) Spherical coordinates (c) Cartesian coordinates

Figure B.1: Comparison of different planar parameterizations. The Hessian normal form is overparameterized but has no degenerate representations. The spherical coordinate and Cartesian coordinate representations are minimally parameterized, but suffer from singularities in certain configurations.

plane, $\widehat{\mathbf{n}} = [n_x, n_y, n_z]^\top$, and the distance of the plane to the origin, p. The Hessian normal form is given by:

$$\widehat{\mathbf{n}} \cdot \mathbf{x} = -p$$

for any point $\mathbf{x}$ that lies on the plane. Despite planes having three DOFs, the Hessian normal form is parameterized by *four* numbers. Therefore, if a plane $\boldsymbol{\pi}_{ik}$ is a Gaussian random vector, its 4×4 covariance would be *singular*, making incremental updates and fast covariance recovery impossible in modern SLAM solvers. This problem can be masked by using algorithms such as Levenburg-Marquardt (LM), however this is neither computationally efficient nor the intended use of LM.

B.2 Spherical Coordinates

Since the unit normal used in the Hessian normal form has two degrees of freedom, one possible alternative would be to convert the unit normal into two angular quantities: azimuth and elevation. By definition, the unit normal is length 1, so one can easily invoke the conversion from Cartesian to spherical coordinates discussed in Appendix B.6. The distance-to-origin, p, from the Hessian normal form is equal to the radius in spherical coordinates. Thus, the spherical representation has a minimal representation with three numbers.

Because we take azimuth to be the rotation about the z axis, this representation yields a singularity for planes whose normal is parallel to the frame of reference's z axis, because the value of the azimuth angle is arbitrary. Unfortunately, this is a common occurrence with the hovering autonomous underwater vehicle (HAUV) inspection robot used extensively in this thesis so this parameterization is not desirable.

B.3 Cartesian Coordinates of Surface Normal

Instead of using angular quantities, we can simply encode the length of the surface normal, p, by scaling the unit normal by p:

$$\mathbf{n} = \begin{bmatrix} pn_x & pn_y & pn_z \end{bmatrix}^\top.$$

The parameterization is simply the three elements of $\mathbf{n}$, and is therefore minimal. Additionally, converting to the Hessian normal form from this parameterization is simple:

$$\begin{aligned} p &= \|\mathbf{n}\| \\ \widehat{\mathbf{n}} &= \frac{\mathbf{n}}{\|\mathbf{n}\|}. \end{aligned}$$

This representation yields a singularity for planes such that the origin lies on the surface of the plane. Because $\|\mathbf{n}\| = 0$ in this case, there are infinitely many planes that have the same representation. For the HAUV inspection robot, this parameterization is rarely encountered so long as the global frame of reference is inboard of the ship being surveyed.

B.4 Avoiding Singularities

To cope with degenerate configurations of planes in SLAM, researchers have avoided globally-referenced planes altogether and instead represent them with respect to a well-behaved reference frame (e.g., the robot pose from which a plane is first observed). This "trick" has been used by Kaess [86] and in our earlier work from [135].

B.5 Plane Composition Derivations

In Section 2.3.4.1, we introduced the planar composition operator $\boxminus$, defined as

$$\boldsymbol{\pi}_{jk} = \mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{ik} = \frac{\left({}^i\mathbf{t}_{ij}{}^\top \boldsymbol{\pi}_{ik} + \|\boldsymbol{\pi}_{ik}\|^2\right) {}^j_i\mathrm{R}\boldsymbol{\pi}_{ik}}{\|\boldsymbol{\pi}_{ik}\|^2},$$

which expresses a plane k with respect to frame j. We derive this relationship as follows. First, the line segment labeled $\mathcal{L}$ in Fig. B.2 has length

$${}^i\mathbf{t}_{ij}{}^\top \frac{\boldsymbol{\pi}_{ik}}{\|\boldsymbol{\pi}_{ik}\|} + \|\boldsymbol{\pi}_{ik}\|$$

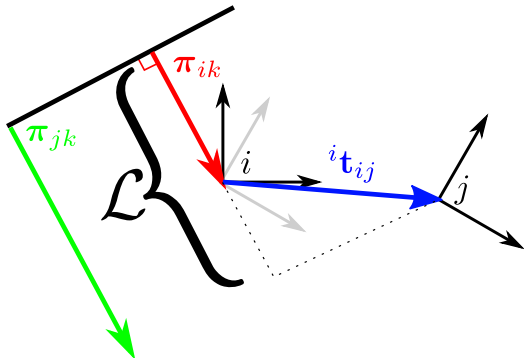


Figure B.2: Illustration of the derivation of the plane composition operator $\boxminus$.

Expressing this length as a fraction of the normal $\boldsymbol{\pi}_{ik}$ yields

$$\frac{{}^{i}\mathbf{t}_{ij}{}^{\top}\frac{\boldsymbol{\pi}_{ik}}{\|\boldsymbol{\pi}_{ik}\|} + \|\boldsymbol{\pi}_{ik}\|}{\|\boldsymbol{\pi}_{ik}\|} = \frac{{}^{i}\mathbf{t}_{ij}{}^{\top}\boldsymbol{\pi}_{ik} + \|\boldsymbol{\pi}_{ij}\|^2}{\|\boldsymbol{\pi}_{ik}\|^2}. \tag{B.1}$$

Next, we express the normal vector $\boldsymbol{\pi}_{ik}$ as rotated into frame j's orientation using ${}^{j}_{i}\mathrm{R}\boldsymbol{\pi}_{ik}$. Finally, we extend this rotated normal by the length derived in (B.1) to arrive at plane k as expressed in frame j:

$$\boldsymbol{\pi}_{jk} = \frac{\left({}^{i}\mathbf{t}_{ij}{}^{\top}\boldsymbol{\pi}_{ik} + \|\boldsymbol{\pi}_{ik}\|^2\right){}^{j}_{i}\mathrm{R}\boldsymbol{\pi}_{ik}}{\|\boldsymbol{\pi}_{ik}\|^2}.$$

By invoking the $\ominus$ operator from Smith, Self, and Cheeseman [158] where $\mathbf{x}_{ji} = \ominus\mathbf{x}_{ij}$, we can define an additional plane composition operator

$$\begin{aligned}\boldsymbol{\pi}_{ik} &= \mathbf{x}_{ji} \boxminus \boldsymbol{\pi}_{jk} \\ &= \ominus\mathbf{x}_{ij} \boxminus \boldsymbol{\pi}_{jk} \\ &= \mathbf{x}_{ij} \boxplus \boldsymbol{\pi}_{jk}\end{aligned}$$

B.6 Spherical to Cartesian Coordinates

The characteristic curvature model used in Chapter 2 lends itself to surface normals expressed in spherical coordinates. To convert the xyz normal vector to a stacked vector of

normal azimuth, a, elevation, e, and magnitude, m, we use the $\mathrm{dir}(\,\cdot\,)$ function:

$$\mathrm{dir}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} \mathrm{atan2}(y, x) \\ \mathrm{atan2}\left(z, \sqrt{x^2 + y^2}\right) \\ \sqrt{x^2 + y^2 + z^2} \end{bmatrix}.$$

Similarly, to convert from spherical coordinates back to Cartesian, we use the $\mathrm{trans}(\,\cdot\,)$ function, defined as:

$$\mathrm{trans}\left(\begin{bmatrix} a \\ e \\ m \end{bmatrix}\right) = m \begin{bmatrix} \cos(e)\cos(a) \\ \cos(e)\sin(a) \\ \sin(e) \end{bmatrix}.$$

APPENDIX C

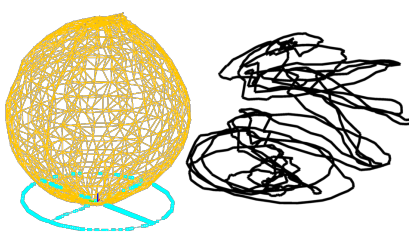
Sensitivity Analysis of Characteristic Radii

The piecewise planar factors introduced in Chapter 2 use tunable characteristic radii that describe how the robot expects the observed surface to curve. In this appendix, we analyze the sensitivity of this value through simulation. We therefore simulated a robot with a sparse 3D scanner surveying a sphere with radius $r = 8$ m at a standoff of 1 m. The robot traverses the surface of the sphere in a spiral-like pattern in such a way that the robot never observes the sphere from the same pose more than once. We corrupt the ground-truth to provide odometry measurements, but have the z, pitch, and roll measurement uncertainties bounded (as would be the case from a pressure depth sensor and gravity-derived roll/pitch measurements in our application). The robot observes one planar patch for each pose.

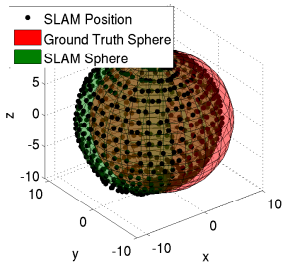
Using our approach, the robot is able to reasonably reconstruct an estimate of its trajectory and surface of the sphere. Interestingly, the robot accumulates most of the pose uncertainty at the top and bottom of the sphere, as shown in Fig. C.1(c). At these portions of the sphere, the surface normal measurements are unable to correct for lateral and heading (x, y, yaw) error because the surface normals align with gravity, which bounds the z, pitch, and roll uncertainties. However, once the robot maps a sphere, the robot is well-localized relative to the sphere, and uncertainty in a global sense is bounded. This suggests the best approach for robot mapping would be to minimize time spent accumulating error in regions where the surface normals are not providing any way to correct odometry error.

Because initial odometry error in the simulation cannot be corrected with planar measurements, we avoid evaluating one-to-one error between ground-truth poses and SLAM poses. For this simulation, we instead evaluate the closeness of the SLAM map to ground-truth by fitting a least-squares sphere for the estimated robot trajectory.

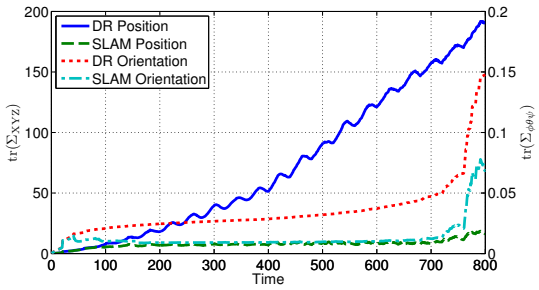
The residuals of these least-squares spheres are shown for various choices of characteristic radii in Fig. C.2. The results from this figure suggest that so long as the tunable radius is somewhere between 4 m and 20 m, the SLAM method produces a reconstruction that is very similar to ground-truth.



(a) SLAM (left) vs. DR (right)



(b) SLAM vs. ground-truth



(c) Covariance over time

Figure C.1: Results of the SLAM simulation using a spherical surface and a 3D robot. A comparison to SLAM and DR is shown in (a), and a comparison to ground-truth is shown in (b). Finally, the trace of the pose covariance is shown as a timeseries in (c).

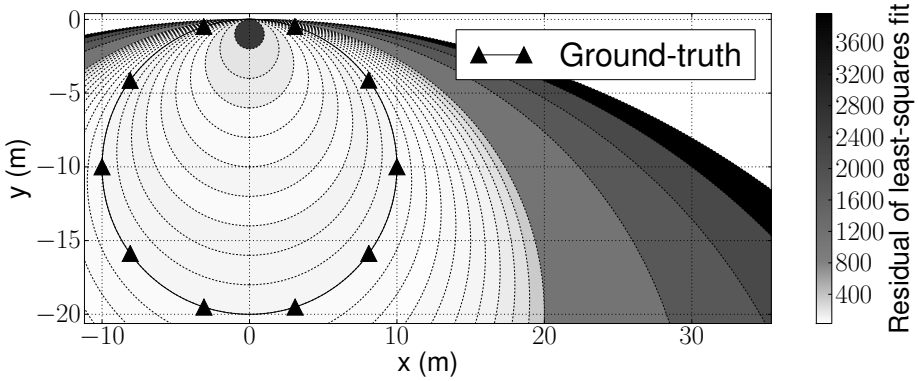


Figure C.2: Comparison to ground-truth for various choices of the characteristic radius. The results of our simulation suggest that our approach produces good SLAM estimates even if the characteristic curvature is not the ground-truth value. We chose a variety of radii for our simulation, and we shaded the corresponding circular regions above according to how well the ground-truth sphere from Fig. C.1(b) fits the estimated trajectory.

APPENDIX D

Pinhole Camera Model

D.1 Calibrated Monocular

In this thesis we use a calibrated pinhole camera with known intrinsic calibration. The projection from a 3D point in the global frame, $\mathbf{X}$, to two-dimensional (2D) pixel coordinates, $\mathbf{u}$, given the pose of the camera and the calibration is computed as follows. First, we transform $\mathbf{X}$ to be expressed in the camera's coordinate frame using the conventions described in Appendix A. Next, we multiply by the camera calibration matrix, K, to arrive at homogeneous pixel coordinates:

$$\underline{\mathbf{u}} = \mathrm{K}\left[{}^{c}_{g}\mathrm{R} \,|\, {}^{c}\mathbf{t}_{cg}\right]\underline{\mathbf{X}}. \tag{D.1}$$

Here, $\underline{\mathbf{u}} = [u, v, 1]^\top$ is the pixel coordinate in normalized homogeneous coordinates and $\mathbf{X} = [X, Y, Z]^\top$ is a point expressed in the global frame. Here, $\mathrm{K} = \left[\begin{smallmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{smallmatrix}\right]$ is a camera calibration matrix consisting of focal lengths f_x and f_y and center of projection c_x and c_y. This model is illustrated in Fig. D.1.

In addition, we formulate a projection to 3D using the planar parameterization discussed in Appendix B. Given the pose of a calibrated camera, c, a feature's pixel coordinate, $\mathbf{u}$, and a planar surface k, with respect to the camera, one can raycast the feature to intersect the surface of the plane. The length of the ray that intersects with the planar surface is given by

$$d = \frac{-\|\boldsymbol{\pi}_{gk}\|^2}{\left(\left[{}^{g}_{c}\mathrm{R} \,|\, {}^{g}\mathbf{t}_{gc}\right]\underline{\mathrm{K}^{-1}\mathbf{u}} - {}^{g}\mathbf{t}_{gc}\right)^\top \boldsymbol{\pi}_{gk}}.$$

Therefore, the 3D coordinate of the intersection (as expressed in the global frame) is given by scaling the unit-length line segment from the camera center of projection to the normalized

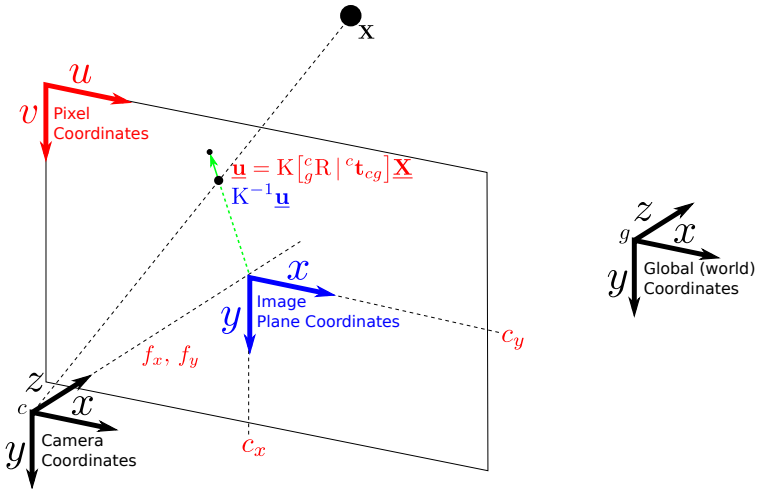


Figure D.1: Pinhole camera model overview. Red denotes pixel-valued quantities, and blue denotes normalized image coordinates. We also use a radial distortion model in this thesis.

image coordinate by d:

$$d\left(\left[{}^{g}_{c}\mathrm{R} \,|\, {}^{g}\mathbf{t}_{gc}\right]\underline{\mathrm{K}^{-1}\underline{\mathbf{u}}} - {}^{g}\mathbf{t}_{gc}\right) + {}^{g}\mathbf{t}_{gc}. \tag{D.2}$$

D.1.1 Two-View Relative-Pose Covariance

In general, bundle adjustment minimizes the weighted squared error of measurements and those predicted by some nonlinear projection model, f, as in

$$\widehat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \|\mathbf{X} - f(\boldsymbol{\Theta})\|^2_{\Sigma_{\mathbf{X}}}, \tag{D.3}$$

where $\|\cdot\|^2_{\Sigma}$ denotes the squared Mahalanobis distance according to the covariance Σ. $\boldsymbol{\Theta}$ is a vector of M unknown parameters containing the relative-pose and 3D structure. This optimization problem forms the maximum likelihood estimate (MLE) of $\boldsymbol{\Theta}$, assuming the measurements, $\mathbf{X}$, are corrupted by additive Gaussian noise. If the measurements are taken as corresponding feature detections from two cameras, and if the Jacobian of f with respect to $\boldsymbol{\Theta}$ is sparse, this optimization problem is known as two-view sparse bundle adjustment (SBA).

In this section, we are interested in recovering two quantities. First, we wish to estimate the relative-pose estimate between cameras 1 and 2:

$$\mathbf{z}_{21} = \begin{bmatrix} \alpha_{21} & \beta_{21} & \phi_{21} & \theta_{21} & \psi_{21} \end{bmatrix}^{\top},$$

consisting of the azimuth, α_{21}, and elevation, β_{21}, of the baseline direction of motion, and the relative Euler orientations, ϕ_{21}, θ_{21}, ψ_{21}.

Second, we wish to recover the covariance of $\mathbf{z}_{21}$. For this section, we have

$$\boldsymbol{\Theta} = \begin{bmatrix} \mathbf{z}_{21} & {}^1\mathbf{P}_1 & \dots & {}^1\mathbf{P}_{M-5} \end{bmatrix},$$

where ${}^1\mathbf{P}_i$ is a the i^{th} 3D point as expressed with respect to camera 1. Because (D.3) is a special case of the SLAM problem, we know from intuition that the covariance of $\boldsymbol{\Theta}$ can be obtained by inverting the information matrix[1], yielding

$$\Sigma_{\boldsymbol{\Theta}} = \left(\mathrm{J}^\top \Sigma_{\mathbf{X}}^{-1} \mathrm{J}\right)^{-1}.$$

However, for this section will derive this using the method from Haralick [66].

The SBA algorithm solves optimization problems in the form of (D.3)) by linearizing $f(\,\cdot\,)$ around the current estimate of $\boldsymbol{\Theta}$, applying a damped Gauss-Newton iteration to update $\boldsymbol{\Theta}$, and repeating the process until convergence. Applying this to (D.3) and using the framework from Haralick [66], we have a scalar-valued cost function that can be written as

$$F(\mathbf{X}, \boldsymbol{\Theta}) = \|\mathbf{X}_u - \mathrm{J}\boldsymbol{\Theta}\|^2_{\Sigma_{\mathbf{X}_u}}, \tag{D.4}$$

where $\mathrm{J} = \frac{\partial f}{\partial \boldsymbol{\Theta}}$ is the Jacobian of $f(\,\cdot\,)$. Then, to first-order,

$$\Sigma_{\boldsymbol{\Theta}} = \left(\frac{\partial g}{\partial \boldsymbol{\Theta}}\right)^{-1} \frac{\partial g}{\partial \mathbf{X}}^\top \Sigma_{\mathbf{X}} \frac{\partial g}{\partial \mathbf{X}} \left(\frac{\partial g}{\partial \boldsymbol{\Theta}}\right)^{-1}, \tag{D.5}$$

In this case, the derivatives are as follows:

$$\begin{aligned} g(\mathbf{X}, \boldsymbol{\Theta}) &= 2\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}\boldsymbol{\Theta} - 2\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathbf{X}_u, \\ \frac{\partial g}{\partial \boldsymbol{\Theta}} &= 2\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}, \\ \frac{\partial g}{\partial \mathbf{X}} &= -2\Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}. \end{aligned}$$

[1] A more performant approach would only be to solve for the upper 5×5 sub-block of $\Sigma_{\boldsymbol{\Theta}}$ corresponding to $\mathbf{z}_{21}$, rather than inverting the entire information matrix followed by marginalizing the 3D points.

Expanding confirms the expected result

$$\begin{aligned}\Sigma_{\boldsymbol{\Theta}} &= \left(2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}\left(-2\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^\top\Sigma_{\mathbf{X}}\left(-2\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)\left(2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}\\ &= \left(2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}\left(-2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\Sigma_{\mathbf{X}}\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)\left(2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}\\ &= 2\left(\mathrm{J}^\top\Sigma_{\mathbf{X}}\mathrm{J}\right)^{-1}\left(\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)\left(2\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}\\ &= \left(\mathrm{J}^\top\Sigma_{\mathbf{X}}^{-1}\mathrm{J}\right)^{-1}.\end{aligned}$$

D.1.2 What if the Calibration is Uncertain?

This thesis assumes a known calibration matrix, but an interesting question arises if the calibration is known with some amount of uncertainty. In this case, we may wish to use maximum *a posteriori* (MAP) estimation to refine our estimate of the calibration, however this is not desirable because the imagery will most likely be noisy in underwater environments and solving for the calibration values would induce dense connectivity in the pose-graph.

Instead, we can place the calibration values in measurement space and use the method from Haralick [66]. In the case of two-view SBA, the optimization problem from (D.3) is modified so that $f(\,\cdot\,)$ is parametrized by calibration values $\mathbf{X}_c$ as follows:

$$\widehat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \|\mathbf{X}_u - f(\boldsymbol{\Theta}; \mathbf{X}_c)\|^2_{\Sigma_{\mathbf{X}_u}}, \tag{D.6}$$

where $\mathbf{X} = [\mathbf{X}_u, \mathbf{X}_c]^\top$ is the new measurement vector, consisting of feature measurements $\mathbf{X}_u$ and calibration values $\mathbf{X}_c$. These calibrations may consist of intrinsic parameters such as focal length, image center coordinates, and lens distortion parameters.

Assuming feature noise and calibration uncertainty are independent, Haralick's approach is given by

$$\Sigma_{\boldsymbol{\Theta}} = \left(\frac{\partial g}{\partial \boldsymbol{\Theta}}\right)^{-1}\frac{\partial g}{\partial \mathbf{X}}^\top\begin{bmatrix}\Sigma_{\mathbf{X}_u} & 0\\ 0 & \Sigma_{\mathbf{X}_c}\end{bmatrix}\frac{\partial g}{\partial \mathbf{X}}\left(\frac{\partial g}{\partial \boldsymbol{\Theta}}\right)^{-1}, \tag{D.7}$$

where $g(\mathbf{X}, \boldsymbol{\Theta}) = \frac{\partial F}{\partial \boldsymbol{\Theta}}$. Applying this technique to (D.4) we have

$$g(\mathbf{X}, \boldsymbol{\Theta}) = 2\mathrm{J}^\top\Sigma_{\mathbf{X}_u}^{-1}\mathrm{J}\boldsymbol{\Theta} - 2\mathrm{J}^\top\Sigma_{\mathbf{X}_u}^{-1}\mathbf{X}_u,$$

with partials

$$\begin{aligned}\frac{\partial g}{\partial \boldsymbol{\Theta}} &= 2\mathrm{J}^\top\Sigma_{\mathbf{X}_u}^{-1}\mathrm{J},\\ \frac{\partial g}{\partial \mathbf{X}} &= \begin{bmatrix}\frac{\partial g}{\partial \mathbf{X}_u}\\ \frac{\partial g}{\partial \mathbf{X}_c}\end{bmatrix} = \begin{bmatrix}-2\Sigma_{\mathbf{X}_u}^{-1}\mathrm{J}\\ \mathrm{A}\end{bmatrix},\end{aligned}$$

where $\mathrm{A} \in \mathbb{R}^{9\times M}$ is a dense matrix with no easily computed closed-form expression. Thus, one can compute A using numerical differentiation. After substituting these values into (D.5), it is not difficult to simplify the covariance estimate to

$$\begin{aligned}\Sigma_{\boldsymbol{\Theta}} &= \left(\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}\right)^{-1} + \mathrm{B}\Sigma_{\mathbf{X}_c}\mathrm{B}^\top \\ &= \Sigma_{\text{HZ}} \qquad\qquad + \Sigma_{\text{Fwd}}\end{aligned} \tag{D.8}$$

where $\mathrm{B} = \frac{1}{2}\left(\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}\right)^{-1} \mathrm{A}^\top$. This covariance estimate may be thought of as the addition of the classic first-order backward propagation of feature covariance, Σ_{HZ}, from Hartley and Zisserman [69] and a forward propagation of calibration covariance, Σ_{Fwd}.

D.1.3 Extension to Unscented Transform

There are some concerns for using Haralick's method for relative-pose uncertainty. First, the elements of A must be individually computed, which is computationally costly when numerically differentiating Brown's distortion model [20]. Further, because A's elements are second-order derivatives, the error from finite differencing can be large and the differentiation step size needs to be tuned depending on the scene and relative-pose. Finally, linearization error from differentiating the nonlinear lens distortion model can be a significant source of inaccuracy in the covariance estimate.

The form of (D.8) suggests that instead the unscented transform can be used to model the forward propagation of calibration uncertainty, rather than linearizing the standard camera projection models [85]. Essentially, we are replacing the $\mathrm{B}\Sigma_{\mathbf{X}_c}\mathrm{B}^\top$ term from (D.8) with one computed from the unscented transform (UT) by approximating the distribution of camera calibration values with $2 \times 9 + 1 = 19$ sigma points, $\boldsymbol{\mathcal{X}}_i$ (with corresponding weights W_i). By doing so, this method avoids inducing linearization error and instead approximates the distribution itself. Our proposed estimate takes the form

$$\Sigma_{\boldsymbol{\Theta}} = \Sigma_{\text{HZ}} + \Sigma_{\text{UT}}. \tag{D.9}$$

Like (D.8), this partitions the relative-pose covariance into two additive terms: a first-order backward propagation of feature covariance, Σ_{HZ}, and a UT-based model of the forward-propagation of camera uncertainty, Σ_{UT}. This method is described in more detail in Algorithm 6.

Our earlier work from [134] analyzes the performances of this method, and we show improved results using low-noise features such as those detected from fiducial markers like calibration targets or AprilTags [129]. However, in the case of using natural features

Algorithm 6 Covariance estimate for two-view SBA with uncertain calibration

Input: Feature locations $\mathbf{X}_u$, camera calibration values $\mathbf{X}_c$, initial guess $\mathbf{\Theta}_0$, covariance matrices $\Sigma_{\mathbf{X}_u}, \Sigma_{\mathbf{X}_c}$
$\widehat{\mathbf{\Theta}} \leftarrow \mathtt{sba}(f, \mathbf{X}_u, \Sigma_{\mathbf{X}_u}, \mathbf{X}_c, \mathbf{\Theta}_0)$
$\mathrm{J} \leftarrow \frac{\partial f}{\partial \mathbf{\Theta}}|_{\mathbf{\Theta}=\widehat{\mathbf{\Theta}}}$
$\Sigma_{\mathrm{HZ}} \leftarrow \left(\mathrm{J}^\top \Sigma_{\mathbf{X}_u}^{-1} \mathrm{J}\right)^{-1}$
$(\boldsymbol{\mathcal{X}}_1, \ldots, \boldsymbol{\mathcal{X}}_{19}, \mathbf{W}) \leftarrow \mathtt{unscentedXfm}(\mathbf{X}_c, \Sigma_{\mathbf{X}_c})$
for $i = 1 : 19$ **do**
    $\boldsymbol{\mathcal{Y}}_\mathbf{i} \leftarrow \mathtt{sba}(f, \mathbf{X}_u, \Sigma_{\mathbf{X}_u}, \boldsymbol{\mathcal{X}}_i, \widehat{\mathbf{\Theta}})$
end for
$\bar{\mathbf{y}} \leftarrow \sum_{i=1}^{p} W_i \boldsymbol{\mathcal{Y}}_i$ // Use weighted "mean of angles" formula
$\Sigma_{\mathrm{UT}} = \sum_{i=1}^{p} W_i \left(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}}\right)\left(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}}\right)^\top$ // Use min. angle for circular quantities
Output: $\Sigma_{\mathrm{HZ}} + \Sigma_{\mathrm{UT}}$

such as those detected by scale-invariant feature transform (SIFT) and speeded up robust features (SURF), the feature noise dominates the calibration uncertainty and so performing Algorithm 6 is not worth the extra computational cost.

D.2 Calibrated Stereo

A calibrated stereo camera can be thought of as two calibrated monocular cameras with a static and known relative transformation between the two. Stereo-calibration is a common task in mobile robotics so we will not cover the specifics here. For this thesis, we are concerned with the three important quantities from stereo camera calibration:

1. The intrinsic parameters of the top camera.
2. The intrinsic parameters of the bottom camera.
3. The extrinsic parameters describing the transformation from the top to the bottom camera.

Thus, we can model the projection of a 3D point $\mathbf{X}$ onto the image plane of the first camera using a trivial extension of the monocular model from Section D.1:

$$\underline{\mathbf{u}_{c_T}} = \mathrm{K}\left[{}^{c_T}_{g}\mathrm{R} \,|\, {}^{c_T}\mathbf{t}_{c_T g}\right]\underline{\mathbf{X}},$$

and the projection onto the bottom camera is given by

$$\underline{\mathbf{u}_{c_B}} = \mathrm{K}\left[{}^{c_B}_{g}\mathrm{R} \,|\, {}^{c_B}\mathbf{t}_{c_B g}\right]\underline{\mathbf{X}},$$

where $\left[{}^{c_B}_{g}\mathrm{R} \,|\, {}^{c_B}\mathbf{t}_{c_B g}\right]$ are the rotation and translation from the global frame to the bottom camera, computed by compounding the pose of the top camera and the extrinsic parameters from item 3 above. Using the notation from Smith, Self, and Cheeseman [158], this corresponds to the relative-pose $\mathbf{x}_{gc_T} \oplus \mathbf{x}_{c_T c_B}$. Note that in this thesis we are using a *vertical* stereo configuration, however these methods easily transition to *horizontal* stereo configurations.

D.2.1 Establishing Feature Correspondence within a Rectified Stereo Pair

A standard approach for establishing feature matches between two cameras (e.g., two monocular cameras or a single stereo pair) is the epipolar constraint:

$$\underline{\mathbf{u}_T}^\top \mathrm{F}_{TB} \underline{\mathbf{u}_B} < \tau, \tag{D.10}$$

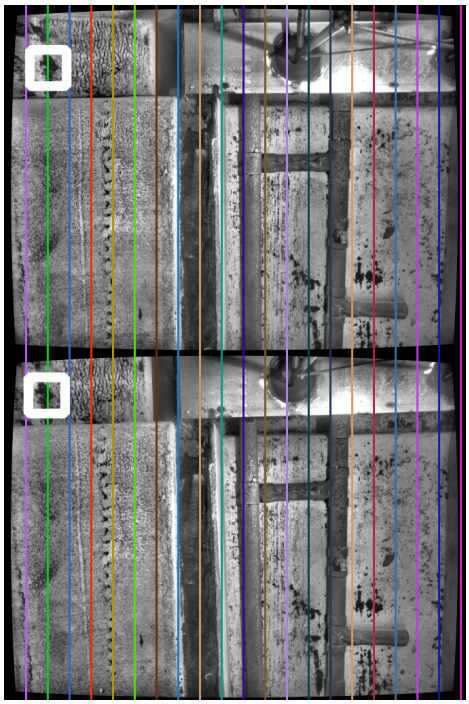
where F_{TB} is the fundamental matrix that converts pixel coordinates in the bottom camera to epilines in the top camera, and τ is a distance threshold (in pixels squared). Because F_{TB} is known for a calibrated stereo camera, this offers a way to check the consistency of candidate matches between the top and bottom camera of a vertical stereo rig. The expression from (D.10) can be thought of the squared distance of a feature to its corresponding epiline as computed using the feature coordinate in the top camera.

A calibrated stereo rig, however, offers a more convenient approach such that the epilines between the top and bottom cameras are perfectly vertical. In this case, (D.10) simplifies to checking the difference of the column indices of $\mathbf{u}_T$ and $\mathbf{u}_B$. In other words, potential feature matches from the top and bottom camera can be geometrically verified simply by comparing the column index of the features' pixel coordinates. This is illustrated using real data in Fig. D.2.

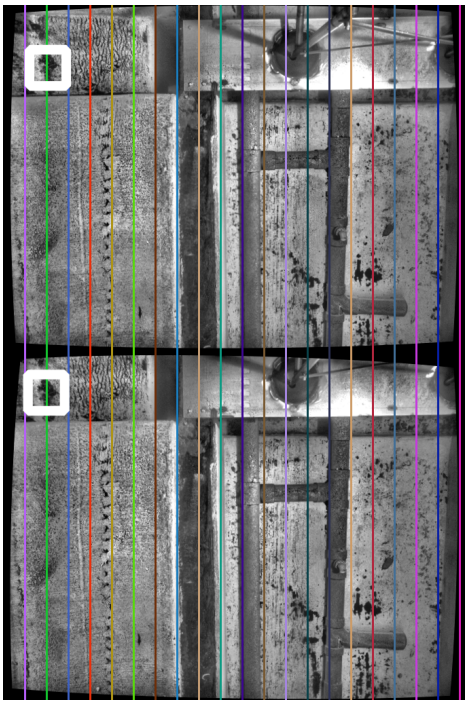
Because the epilines are perfectly vertical in a rectified stereo pair, the relative-pose from the top to bottom camera only involves translation along the y-axis (i.e., the downward-pointing direction). For example, suppose the relative-pose from the bottom camera to the top camera determined from camera calibration is

$$x_{c_B c_T} = \begin{bmatrix} -1.12 & -77.6 & -0.65 & 0.0131 & 0.0034 & -0.0212 \end{bmatrix}^\top,$$

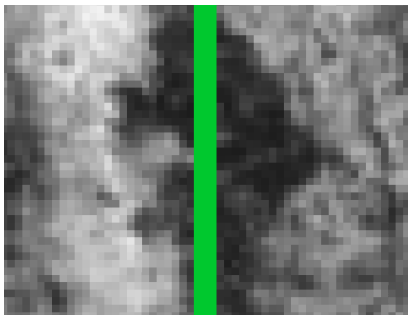
where each translational unit is provided in millimeters and the rotational units are provided in radians. Once the images are rectified, the projection operation from Appendix D.2



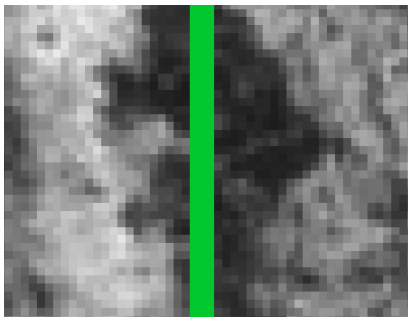
(a) Calibrated, unrectified



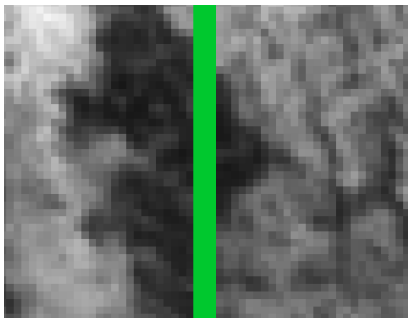
(b) Calibrated, rectified



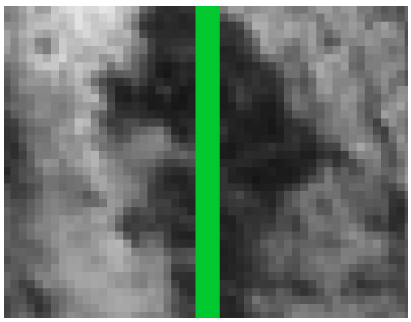
(c) White region from (a), top



(d) White region from (b), top



(e) White region from (a), bottom



(f) White region from (b), bottom

Figure D.2: Rectified stereo pair example. Vertical lines placed uniformly on a stereo pair that is not rectified (left) and one that is (right). In the rectified case, these vertical lines correspond to the epilines, so that corresponding pixels between the top and bottom camera lie on the same line.

behaves as if the relative-pose is:

$$x'_{c_B c_T} = \begin{bmatrix} 0 & -77.6 & 0 & 0 & 0 & 0 \end{bmatrix}^\top.$$

D.2.2 Triangulation for a Rectified Stereo Pair

Once feature matches between the top and bottom are established, as in the previous section, the rectification allows us to compute the disparity between two matching feature coordinates as the difference between the row indices in the top and bottom cameras.

Using a calibrated perspective transformation matrix, Q, we can project a feature to 3D (with respect to the top camera) using the following

$$\underline{\mathbf{X}} = \mathrm{Q} \begin{bmatrix} u_T & v_T & (v_T - v_B) & 1 \end{bmatrix}^\top, \tag{D.11}$$

where u_T and v_T are column and row indices, respectively, of the pixel location of a point in the top camera's image plane, and v_B is the row index of the corresponding point in the bottom camera's image plane. Here, $\mathbf{X}$ denotes the 3D coordinate of the feature as expressed in the top camera frame.

For a rectified stereo pair, Q, takes the form $\mathrm{Q} = \begin{bmatrix} 1 & 0 & 0 & q_{14} \\ 0 & 1 & 0 & q_{24} \\ 0 & 0 & 0 & q_{34} \\ 0 & 0 & q_{43} & 0 \end{bmatrix}$. The entries q_{14} and q_{24} can be thought of as the rectified camera center of the top camera with respect to the bottom camera center. The entry q_{34} is the focal length of the top camera, and q_{43} scales the disparity during the projection to 3D, discussed below.

This matrix serves two purposes:

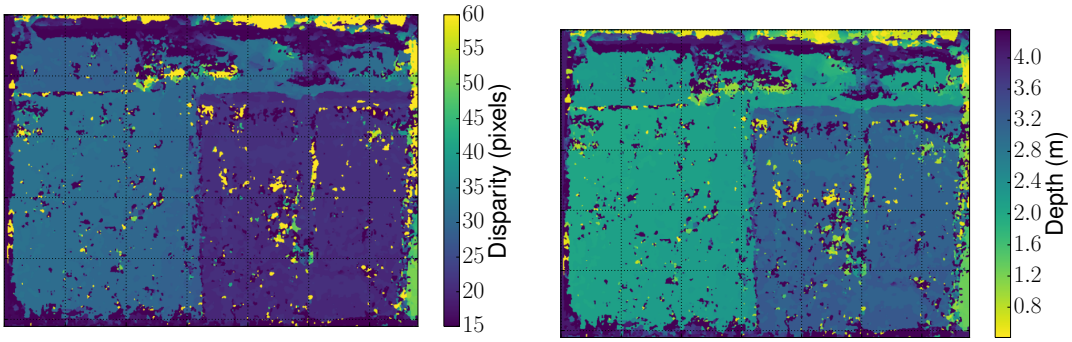
1. Triangulate *sparse* 2D feature detections (such as SIFT) to 3D points using (D.11), and

2. project *dense* disparity images (such as block-matching algorithms [70]) to 3D depth images.

For the latter case, a disparity value directly translates to camera-relative depth, z, using the following formula

$$z = \frac{q_{34}}{(v_T - v_B)\, q_{43}},$$

where $(v_T - v_B)$ is the pixel disparity. An example of this conversion is shown in Fig. D.3.



(a) Disparity (b) Depth

Figure D.3: Disparity and range images for the example shown in Fig. D.2.

D.2.3 Matching Features Between Two Rectified Stereo Pairs

In the previous two sections, we discussed the tools necessary to compute corresponding features between two cameras in a *single* stereo pair, and to project that correspondence to camera-relative 3D coordinates. In this section, we discuss how to establish correspondences between *two* stereo pairs using random sample consensus (RANSAC).

The approach to align two stereo pairs, i and j, is as follows:

1. Use the method in Appendix D.2.1 to establish the correspondences between the top and bottom camera in pair i, and project those correspondences to 3D using Appendix D.2.2.

2. Repeat for stereo pair j.

3. Find the nearest-neighbor putative descriptor matches between the features in the top camera of pair i, and the top camera of pair j. If a good SLAM prior is available, use pose-constrained correspondence search (PCCS) to mask away unlikely matches.

4. Of the matches from step 3, discard those that do have no assigned corresponding feature in the bottom camera.

5. Use RANSAC and least-squares alignment [6] to solve for the relative-pose between the two cameras and discard outliers in the putative matches.

The least-squares alignment from step 5 above is sometimes called the *3-point algorithm* [127]. It uses singular value decomposition (SVD) to fit a least-squares rotation to two sets of corresponding points (each containing a minimum of three points), and uses the point centroids to fit a least-squares translation. By transforming all putative matches using

the candidate alignment, one can use RANSAC to compute the best alignment and discard any outliers.

Since we use the high-dimensional SIFT descriptor in this thesis, a graphical processing unit (GPU) implementation of nearest-neighbor search is recommended for steps 1 and 3 above.

D.3 PCCS: Validity of Gaussian Approximation

Feature correspondences in a two-view visual reconstruction are more easily determined using a prior from SLAM. In this section, we briefly demonstrate that the underlying assumption of pose-constrained correspondence search (PCCS): that propagating features between two cameras results in a Gaussian distribution of pixel coordinates.

Consider a planar scene shown in Fig. D.4(a) with a plane $\boldsymbol{\pi}_{ik}$ and relative-pose $\mathbf{x}_{ij}$ between cameras i and j. Using the monocular pinhole camera model from Appendix D, we can back-project a feature in camera i onto plane $\boldsymbol{\pi}$, and back into camera j. Within the PCCS framework, this is called the point transfer between two images.

If we assume that the distribution of the relative-pose $\mathbf{x}_{ij}$ is Gaussian according to

$$\mathbf{x}_{ij} \sim \mathcal{N}\left(\boldsymbol{\mu}_{ij}, \Sigma_0\right),$$

and that the plane $\boldsymbol{\pi}_{ik}$ is also Gaussian, then we can formulate a function that projects a point with pixel coordinates $\langle u_i, v_j \rangle$ onto the image plane of camera j. For simplicity, we assume that camera i is the origin; therefore ${}^{c}_{g}\mathrm{R} = \mathrm{I}_{3\times3}$ and ${}^{g}\mathbf{t}_{gc} = \mathbf{0}$. Using the pinhole camera model and the raycasting formulation from (D.2), we have

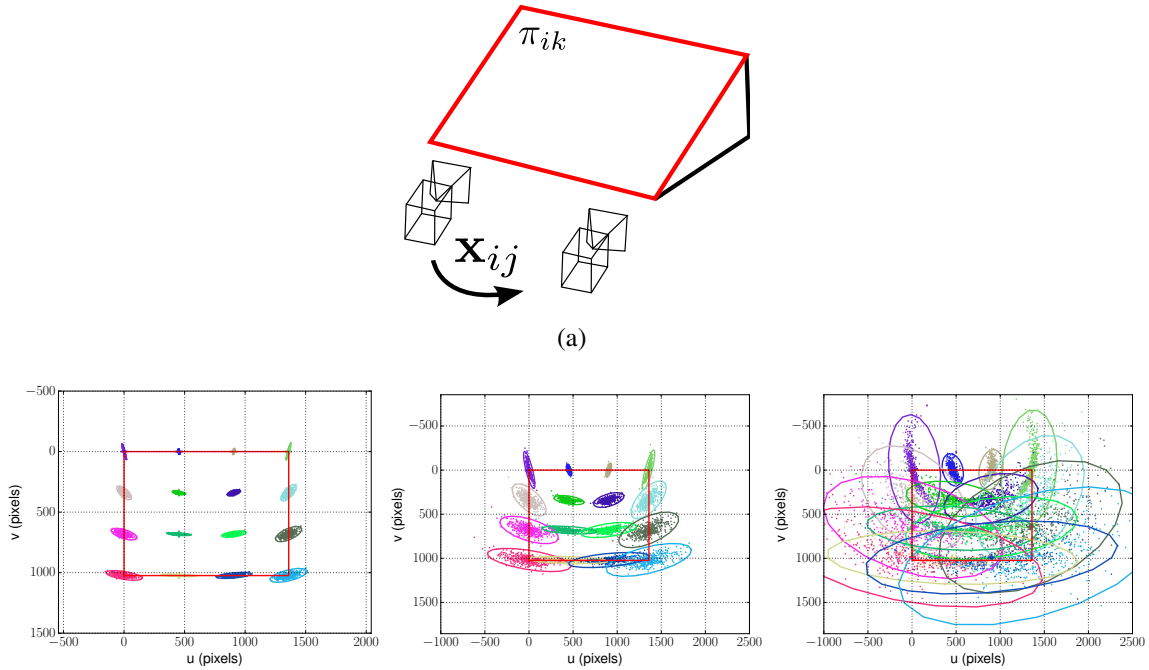
$$\underline{\mathbf{u}_j} = \mathrm{K}\left[{}^{j}_{i}\mathrm{R} \,|\, {}^{j}\mathbf{t}_{ji}\right] \underline{\begin{pmatrix} -\|\boldsymbol{\pi}_{1k}\|^2 \left(\mathrm{K}^{-1}\underline{\mathbf{u}}_i\right) \\ \left(\mathrm{K}^{-1}\underline{\mathbf{u}}_i\right)^{\top} \boldsymbol{\pi}_{1k} \end{pmatrix}} = f\left(\begin{bmatrix} \mathbf{x}_{ij} \\ \boldsymbol{\pi}_{ik} \\ \mathbf{u}_i \end{bmatrix}\right). \tag{D.12}$$

We assume that $\mathbf{x}_{ij}$, $\boldsymbol{\pi}_{ik}$, and $\mathbf{u}_i$ are independent Gaussians according to

$$\begin{aligned} \mathbf{x}_{ij} &\sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_{ij}}, \Sigma_{\mathbf{x}_{ij}}\right) \\ \boldsymbol{\pi}_{ik} &\sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\pi}_{ik}}, \Sigma_{\boldsymbol{\pi}_{ik}}\right) \\ \mathbf{u}_i &\sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{u}_i}, \Sigma_{\mathbf{u}_i}\right), \end{aligned}$$

and we let $\Sigma_0 = \mathrm{diag}\left(\Sigma_{\mathbf{x}_{ij}}, \Sigma_{\boldsymbol{\pi}_{ik}}, \Sigma_{\mathbf{u}_i}\right)$.

We evaluate this point transfer for two methods: Monte-Carlo simulation and an using an



(a)

(b) $\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_{ij}}, \Sigma_0\right)$ (c) $\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_{ij}}, \Sigma_0 \times 10\right)$ (d) $\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_{ij}}, \Sigma_0 \times 100\right)$

Figure D.4: Toy example for analyzing error distribution of (D.12). The geometry of the example is shown in (a), while the results for various covariances are shown in (b) through (d). In each subfigure, we plot the point-transfer for a 4×4 grid of pixel coordinates. The distribution of the point transfer is shown with scattered points (Monte-Carlo) and 2D ellipses with a 99% confidence threshold (UT).

unscented transform of $\begin{bmatrix} \mathbf{x}_{ij} \\ \boldsymbol{\pi}_{ik} \\ \mathbf{u}_i \end{bmatrix}$. As shown in Fig. D.4, the transfer of points to the image plane of camera j is fairly Gaussian except when Σ_0 becomes large. In this case, because (D.12) is very nonlinear, the Gaussian assumption is perhaps no longer valid. However, even for large Σ_0 this approach has some silver-lining for visual SLAM: the predicted covariance ellipses capture a large image area so the recall of putative feature correspondence is large.

APPENDIX E

Properties of the Gaussian Distribution Used in GP Regression

Certain properties of the multivariate Gaussian distribution are used to derive the Gaussian Process regression method described in Chapter 3 and are discussed here.

E.1 Marginal and Conditional Multivariate Gaussian Distribution

Given a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution for $\mathbf{y}$ given $\mathbf{x}$,

$$p(\mathbf{x}) = \mathcal{N}\left(\boldsymbol{\mu}, \Lambda^{-1}\right)$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}\left(\mathrm{A}\mathbf{x} + \mathbf{b}, \mathrm{L}^{-1}\right),$$

then the marginal distribution of $\mathbf{y}$ and the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ are given by

$$p(\mathbf{y}) = \mathcal{N}\left(\mathrm{A}\boldsymbol{\mu} + \mathbf{b}, \mathrm{L}^{-1} + \mathrm{A}\Lambda^{-1}\mathrm{A}^{\top}\right)$$
$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}\left(\Sigma\left[\mathrm{A}^{\top}\mathrm{L}\left(\mathbf{y} - \mathbf{b}\right) + \Lambda\boldsymbol{\mu}\right], \Sigma\right),$$

where $\Sigma = \left(\Lambda + \mathrm{A}^{\top}\mathrm{L}\mathrm{A}\right)^{-1}$.

E.2 Conditional Multivariate Gaussian Using Schur Complement

Given a multivariate Gaussian distribution for $\mathbf{x} = [\mathbf{x}_a^\top \ \mathbf{x}_b^\top]^\top$, the corresponding partitions of the mean and covariance are

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix}$$
$$\Sigma = \begin{bmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{bmatrix}.$$

Then the conditional distribution is also Gaussian, i.e., $p(\mathbf{x}_a|\mathbf{x}_b) = \mathcal{N}(\boldsymbol{\mu}_{a|b}, \Sigma_{a|b})$, where

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \Sigma_{ab}\Sigma_{bb}^{-1}\left(\mathbf{x}_b - \boldsymbol{\mu}_b\right)$$
$$\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}.$$

E.3 Correlations in Prediction

Gaussian process (GP) regression commonly provides only the marginal variance for each prediction, which can easily be computed using Appendix E.2. However, for some applications, the correlations between predictions in GP regression may be useful.

Like Appendix 3.2.2, consider N training input and output pairs $(\mathbf{x}_i, z_i)$ and M test inputs $[\mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_M}]$. Now that our testing samples are vector, the joint training and test marginal likelihood takes the form

$$p(\mathbf{y}, \mathbf{z}_t) = \mathcal{N}\left(\mathbf{0}, \mathrm{K}_{N+M} + \beta^{-1}\mathrm{I}\right), \quad \mathrm{K}_{N+M} = \begin{bmatrix} \mathrm{K}_N & \mathrm{K}_{NM} \\ \mathrm{K}_{MN} & \mathrm{K}_M \end{bmatrix}.$$

Again, using the properties of the multivariate Gaussian distribution, $p(\mathbf{z}_t|\mathbf{z}) = \mathcal{N}(\mu_t, \sigma_t^2)$ is given by

$$\boldsymbol{\mu}_{\mathbf{z}_t} = \mathrm{K}_{MN}\left(\mathrm{K}_N + \beta^{-1}\mathrm{I}\right)^{-1}\mathbf{z}$$
$$\Sigma_{\mathbf{z}_t} = \mathrm{K}_M - \mathrm{K}_{MN}\left(\mathrm{K}_N + \beta^{-1}\mathrm{I}\right)^{-1}\mathrm{K}_{NM} + \beta^{-1}\mathrm{I}.$$

This can simply be thought of as stacking M query points into columns of a matrix, and performing GP regression using the same technique as the case when $M = 1$.

BIBLIOGRAPHY

[1] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust map optimization using dynamic covariance scaling. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3626–3631, Karlsruhe, Germany, May 2013.

[2] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 72–79, Kyoto, Japan, Oct. 2009.

[3] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org, 2014.

[4] F. Aguirre, J. Boucher, and J. Jacq. Underwater navigation by video sequence analysis. In *Proceedings of the International Conference Pattern Recognition*, volume 2, pages 537–539, Atlantic City, NJ, USA, June 1990.

[5] A. Alcocer, P. Oliveira, and A. Pascoal. Underwater acoustic positioning systems based on buoys with GPS. In *Proceedings of the European Conference on Underwater Acoustics*, pages 1–8, Carvoeiro, Portugal, June 2006.

[6] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.

[7] M. D. Aykin and S. Negahdaripour. On feature matching and image registration for two-dimensional forward-scan sonar imaging. *Journal of Field Robotics*, 30(4): 602–623, 2013.

[8] R. D. Ballard, L. E. Stager, D. Master, D. Yoerger, D. Mindell, L. L. Whitcomb, H. Singh, and D. Piechota. Iron age shipwrecks in deep water off Ashkelon, Israel. *American Journal of Archaeology*, pages 151–168, 2002.

[9] S. Barkby, S. Williams, O. Pizarro, and M. Jakuba. Bathymetric SLAM with no map overlap using Gaussian Processes. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1242–1248, San Fransisco, CA, USA, Sept. 2011.

[10] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision*, pages 404–417, Graz, Austria, May 2006. Springer.

[11] E. Belcher, B. Matsuyama, and G. Trimble. Object identification with acoustic lenses. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 6–11, Kona, HI, USA, Sept. 2001.

[12] V. Bichucher, J. M. Walls, P. Ozog, K. A. Skinner, and R. M. Eustice. Bathymetric factor graph SLAM with sparse point cloud alignment. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, Washington, D.C., USA, Oct. 2015. Accepted, To Appear.

[13] F. Bonin, A. Burguera, and G. Oliver. Imaging systems for advanced underwater vehicles. *Journal of Maritime Research*, 8:65–86, 2011.

[14] F. Bonnín-Pascual and A. Ortiz. Detection of cracks and corrosion for automated vessels visual inspection. In *Proceedings of the International Conference of the Catalan Association for Artificial Intelligence*, pages 111–120, Tarragona, Spain, Oct. 2010.

[15] B. Boon, F. Brennan, Y. Garbatov, C. Ji, J. Parunov, T. Rahman, C. Rizzo, A. Rouhan, C. Shin, and N. Yamamoto. Condition assessment of aged ships and offshore structures. In *International Ship and Offshore Structures Congress*, volume 2, pages 313–365, Seoul, Korea, Aug. 2009.

[16] M. Bosse and R. Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *International Journal of Robotics Research*, 27 (6):667–691, 2008.

[17] M. Bosse, P. Newman, J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *International Journal of Robotics Research*, 23(12):1113–1139, Dec. 2004.

[18] A. D. Bowen, D. R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J. C. Kinsey, M. Heintz, G. McDonald, D. B. Peters, J. Bailey, E. Bors, T. Shank, L. L. Whitcomb, S. C. Martin, S. E. Webster, M. V. Jakuba, B. Fletcher, C. Young, J. Buescher, P. Fryer, and S. Hulme. Field trials of the Nereus hybrid underwater robotic vehicle in the challenger deep of the Mariana Trench. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1–10, Biloxi, MS, USA, Oct. 2009.

[19] N. Brokloff. Matrix algorithm for Doppler sonar navigation. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, volume 3, pages 378–383, Brest, France, Sept. 1994.

[20] D. C. Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3): 444–462, 1966.

[21] M. Bryson, M. Johnson-Roberson, O. Pizarro, and S. Williams. Colour-consistent structure-from-motion models using underwater imagery. In *Proceedings of the Robotics: Science & Systems Conference*, Sydney, Australia, July 2012.

[22] H. Bülow and A. Birk. Spectral registration of noisy sonar data for underwater 3D mapping. *Autonomous Robots*, 30(3):307–331, 2011.

[23] R. Campos, R. Garcia, P. Alliez, and M. Yvinec. A surface reconstruction method for in-detail underwater 3D optical mapping. *International Journal of Robotics Research*, 34(1):64–89, 2015.

[24] N. Carlevaris-Bianco and R. M. Eustice. Multi-view registration for feature-poor underwater imagery. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 423–430, Shanghai, China, May 2011.

[25] N. Carlevaris-Bianco and R. M. Eustice. Generic factor-based node marginalization and edge sparsification for pose-graph SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5728–5735, Karlsruhe, Germany, May 2013.

[26] N. Carlevaris-Bianco and R. M. Eustice. Long-term simultaneous localization and mapping with generic linear constraint node removal. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1034–1041, Tokyo, Japan, Nov. 2013.

[27] N. Carlevaris-Bianco and R. M. Eustice. Conservative edge sparsification for graph SLAM node removal. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 854–860, Hong Kong, China, June 2014.

[28] N. Carlevaris-Bianco, M. Kaess, and R. M. Eustice. Generic node removal for factor-graph SLAM. *IEEE Transactions on Robotics*, 30(6):1371–1385, 2014.

[29] A. Carvalho, L. Sagrilo, I. Silva, J. Rebello, and R. Carneval. On the reliability of an automated ultrasonic system for hull inspection in ship-based oil production units. *Applied Ocean Research*, 25(5):235 – 241, 2003.

[30] S. M. Chaves, A. Kim, and R. M. Eustice. Opportunistic sampling-based planning for active visual SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3073–3080, Chicago, IL, USA, Sept. 2014.

[31] S. M. Chaves, J. M. Walls, E. Galceran, and R. M. Eustice. Risk aversion in belief-space planning under measurement acquisition uncertainty. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2079–2086, Hamburg, Germany, Sept. 2015.

[32] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2724–2729, Nice, France, May 1991.

[33] A. K. R. Chowdhury and R. Chellappa. Face reconstruction from monocular video using uncertainty analysis and a generic model. *Computer Vision and Image Understanding*, 91(12):188–213, 2003.

[34] M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 27(6):647–665, June 2008.

[35] M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research*, 30(9):1100–1123, Nov. 2010.

[36] A. Cunningham, V. Indelman, and F. Dellaert. DDF-SAM 2.0: Consistent distributed smoothing and mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5220–5227, Karlsruhe, Germany, May 2013.

[37] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312, New Orleans, LA, Aug. 1996.

[38] S. Daftry, C. Hoppe, and H. Bischof. Building with drones: Accurate 3D facade reconstruction using MAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3487–3494, Seattle, WA, USA, May 2015.

[39] F. Dellaert and M. Kaess. Square root SAM: simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[40] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo localization for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1322–1328, Detroit, MI, USA, May 1999.

[41] F. Dellaert, J. Carlson, V. Ila, K. Ni, and C. E. Thorpe. Subgraph-preconditioned conjugate gradients for large scale SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2566–2571, Taipei, Taiwan, Oct. 2010.

[42] V. Dhiman, A. Kundu, F. Dellaert, and J. Corso. Modern MAP inference methods for accurate and fast occupancy grid mapping on higher order factor graphs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2037–2044, Hong Kong, China, May 2014.

[43] S. Dragiev, M. Toussaint, and M. Gienger. Gaussian process implicit surfaces for shape estimation and grasping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2845–2850, Shanghai, China, May 2011.

[44] S. Duntley. Light in the sea. *Journal of the Optical Society of America*, 53(2): 214–233, Feb. 1963.

[45] E. Eade, P. Fong, and M. Munich. Monocular graph SLAM with complexity reduction. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3017–3024, Taipei, Taiwan, 2010.

[46] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, 1994.

[47] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, OR, USA, 1996.

[48] R. Eustice, R. Camilli, and H. Singh. Towards bathymetry-optimized Doppler renavigation for AUVs. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1430–1436, Washington, DC, USA, Sept. 2005.

[49] R. M. Eustice. *Large-area visually augmented navigation for autonomous underwater vehicles*. PhD thesis, Department of Ocean Engineering, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution Joint Program, Cambridge, MA, USA, June 2005.

[50] R. M. Eustice and H. Singh. Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters. *International Journal of Robotics Research*, 25(12):1223–1242, 2006.

[51] R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, Dec. 2006.

[52] R. M. Eustice, O. Pizarro, and H. Singh. Visually augmented navigation for autonomous underwater vehicles. *IEEE Journal of Ocean Engineering*, 33(2):103–122, Apr. 2008.

[53] T. G. Farr, P. A. Rosen, E. Caro, R. Crippen, R. Duren, S. Hensley, M. Kobrick, M. Paller, E. Rodriguez, and L. Roth. The shuttle radar topography mission. *Reviews of Geophysics*, 45(2), 2007.

[54] D. Fidaleo and G. Medioni. Model-assisted 3D face reconstruction from video. In *Proceedings of the IEEE Workshop on Analysis and Modeling of Faces and Gestures*, pages 124–138, Rio de Janero, Brazil, Oct. 2007.

[55] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[56] P. Fua. Using model-driven bundle-adjustment to model heads from raw video sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 46–53, Kerkyra, Greece, Sept. 1999.

[57] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[58] A. Geva, G. Briskin, E. Rivlin, and H. Rotstein. Estimating camera pose using bundle adjustment and digital terrain model constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4000–4005, Seattle, WA, USA, May 2015.

[59] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7(2):219–269, 1995.

[60] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth. OpenFABMAP: An open source toolbox for appearance-based loop closure detection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4730–4735, St. Paul, MN, USA, 2012.

[61] N. Gracias and J. Santos-Victor. Underwater mosaicing and trajectory reconstruction using global alignment. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 2557–2563, Honolulu, HI, USA, Nov. 2001.

[62] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. *Proceedings of the Robotics: Science & Systems Conference*, June 2007.

[63] G. Grisetti, C. Stachniss, and W. Burgard. Nonlinear constraint network optimization for efficient map learning. *IEEE Transactions on Intelligent Transportation Systems*, 10(3):428–439, 2009.

[64] E. Grosso, G. Sandini, and C. Frigato. Extraction of 3-D information and volumetric uncertainty from multiple stereo images. In *Proceedings of the European Conference on Artificial Intelligence*, pages 683–688, Munich, Germany, Aug. 1988.

[65] E. Gustafson, B. Jalving, ystein Engelhardtsen, and N. Burchill. HUGIN 1000 Arctic class AUV. In *The Polar Petroleum Potential Conference & Exhibition*, pages 714–721, Halifax, Nova Scotia, Sept. 2011.

[66] R. Haralick. Propagating covariance in computer vision. In *Proceedings of the International Conference Pattern Recognition*, pages 493–498, Jerusalem, Israel, Oct. 1994.

[67] C. Harris and M. Stephens. A Combined Corner and Edge Detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, Manchester, England, Aug. 1988.

[68] S. Harris and E. Slate. Lamp Ray: Ship hull assessment for value, safety and readiness. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 493–500, Seattle, WA, USA, Sept. 1999.

[69] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004.

[70] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008.

[71] F. S. Hover, J. Vaganay, M. Elkins, S. Willcox, V. Polidoro, J. Morash, R. Damus, and S. Desset. A vehicle system for autonomous relative survey of in-water ships. *Marine Technology Society Journal*, 41(2):44–55, 2007.

[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. *International Journal of Robotics Research*, 31(12):1445–1464, 2012.

[73] T. Huang and M. Kaess. Towards acoustic structure from motion for imaging sonar. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 758–765, Hamburg, Germany, Sept. 2015.

[74] P. J. Huber. *Robust Statistics*. Wiley, New York, 2011.

[75] N. Hurtós, X. Cufı, and J. Salvi. A novel blending technique for two-dimensional forward-looking sonar mosaicing. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 23–27, San Diego, CA, USA, Sept. 2013.

[76] N. Hurtós, S. Nagappa, N. Palomeras, and J. Salvi. Real-time mosaicing with two-dimensional forward-looking sonar. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 601–606, Hong Kong, China, June 2014.

[77] N. Hurtós, D. Ribas, X. Cufı, Y. Petillot, and J. Salvi. Fourier-based registration for robust forward-looking sonar in low-visibility underwater environments. *Journal of Field Robotics*, 32(1):123–151, 2015.

[78] V. Indelman, E. Nelsony, N. Michaely, and F. Dellaert. Multi-robot pose graph localization and data association from unknown initial relative poses via expectation maximization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 593–600, Hong Kong, China, June 2014. Accepted, To Appear.

[79] K. Ishizu, N. Sakagami, K. Ishimaru, M. Shibata, H. Onishi, S. Murakami, and S. Kawamura. Ship hull inspection using a small underwater robot with a mechanical contact mechanism. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1–6, Yeosu, South Korea, May 2012.

[80] J. Jaffe. Computer modeling and the design of optimal underwater imaging systems. *IEEE Journal of Oceanic Engineering*, 15(2):101–111, 1990.

[81] B. Jalving, M. Mandt, O. K. Hagen, and F. Pøhner. Terrain referenced navigation of AUVs and submarines using multibeam echo sounders. In *Proceedings of the European Conference on Undersea Defense Technology*, Nice, France, June 2004.

[82] H. Johannsson, M. Kaess, B. Englot, F. Hover, and J. J. Leonard. Imaging sonar-aided navigation for autonomous underwater harbor surveillance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4396–4403, Taipei, Taiwan, Oct. 2010.

[83] M. Johnson-Roberson, O. Pizarro, S. B. Williams, and I. Mahon. Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. *Journal of Field Robotics*, 27(1):21–51, 2010.

[84] M. Johnson-Roberson, M. Bryson, B. Douillard, O. Pizarro, and S. B. Williams. Out-of-core efficient blending for underwater georeferenced textured 3D maps. In *Proceedings of the International Conference on IEEE Computing for Geospatial Research and Application*, pages 8–15, San Jose, CA, July 2013.

[85] S. Julier. The scaled unscented transformation. In *Proceedings of the American Control Conference*, volume 6, pages 4555–4559, Anchorage, AK, USA, May 2002.

[86] M. Kaess. Simultaneous localization and mapping with infinite planes. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4605–4611, Seattle, WA, USA, May 2015.

[87] M. Kaess and F. Dellaert. Covariance recovery from a square root information matrix for data association. *Robotics and Autonomous Systems*, 57(12):1198–1210, 2009.

[88] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[89] M. Kaess, H. Johannsson, D. Rosen, N. Carlevaris-Bianco, and J. Leonard. Open source implementation of iSAM. http://people.csail.mit.edu/kaess/isam, 2010.

[90] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 31(2):216–235, 2012.

[91] S. B. Kang and M. Jones. Appearance-based structure from motion using linear classes of 3D models. *International Journal of Computer Vision*, 49(1):5–22, 2002.

[92] J. Kappes, B. Andres, F. Hamprecht, C. Schnorr, S. Nowozin, D. Batra, S. Kim, B. Kausler, J. Lellmann, N. Komodakis, and C. Rother. A comparative study of modern inference techniques for discrete energy minimization problems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1328–1335, Portland, Oregon, June 2013.

[93] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Symposium on Geometry Processing*, volume 7, pages 61–70, Cagliari, Italy, June 2006.

[94] I. Kemelmacher-Shlizerman and S. M. Seitz. Face reconstruction in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1746–1753, Barcelona, Spain, Nov. 2011.

[95] A. Kim and R. M. Eustice. Pose-graph visual SLAM with geometric model selection for autonomous underwater ship hull inspection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1559–1565, St. Louis, MO, USA, Oct. 2009.

[96] A. Kim and R. M. Eustice. Real-time visual SLAM for autonomous underwater hull inspection using visual saliency. *IEEE Transactions on Robotics*, 29(3):719–733, June 2013.

[97] A. Kim and R. M. Eustice. Active visual SLAM for robotic area coverage: Theory and experiment. *International Journal of Robotics Research*, 34(4-5):457–475, 2015.

[98] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple relative pose graphs for robust cooperative mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3185–3192, Anchorage, Alaska, May 2010.

[99] K. Konolige and J. Bowman. Towards lifelong visual maps. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1156–1163, St. Louis, MO, USA, 2009.

[100] H. Kretzschmar and C. Stachniss. Information-theoretic compression of pose graphs for laser-based SLAM. *International Journal of Robotics Research*, 31:1219–1230, 2012.

[101] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[102] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3607–3613, Shanghai, China, May 2011.

[103] C. Kunz, C. Murphy, H. Singh, C. Pontbriand, R. A. Sohn, S. Singh, T. Sato, C. Roman, K. Nakamura, M. Jakuba, R. Eustice, R. Camilli, and J. Bailey. Toward extraplanetary under-ice exploration: Robotic steps in the Arctic. *Journal of Field Robotics*, 26(4):411–429, 2009.

[104] I. S. Kweon and T. Kanade. High resolution terrain map from multiple sensor data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 127–134 vol.1, July 1990.

[105] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the Annual International Conference on Machine Learning*, pages 609–616, Montreal, Canada, June 2009.

[106] J. Leonard and H. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In H. J. and D. Koditschek, editors, *Proceedings of the International Symposium on Robotics Research*, Salt Lake City, Utah, Oct. 1999.

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

[108] J. Li, R. M. Eustice, and M. Johnson-Roberson. High-level visual features for underwater place recognition. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3652–3659, Seattle, WA, USA, May 2015.

[109] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[110] Matrix-Geo. Aerial photogrammetry. http://www.matrix-geo.com/photogrammetry.html, 2013.

[111] D. Meduna, S. Rock, and R. McEwen. Low-cost terrain relative navigation for long-range AUVs. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1–7, Woods Hole, MA, USA, Oct. 2008.

[112] A. Melkumyan and F. Ramos. A sparse covariance function for exact Gaussian process inference in large datasets. In *International Joint Conferences on Artificial Intelligence*, pages 1936–1942, Pasadena, CA, USA, June 2009.

[113] L. L. Menegaldo, M. Santos, G. A. N. Ferreira, R. G. Siqueira, and L. Moscato. SIRUS: A mobile robot for floating production storage and offloading (FPSO) ship hull inspection. In *Proceedings of the IEEE International Workshop on Advanced Motion Control*, pages 27–32, Trento, Italy, Mar. 2008.

[114] L. L. Menegaldo, G. Ferreira, M. F. Santos, and R. S. Guerato. Development and navigation of a mobile robot for floating production storage and offloading ship hull inspection. *IEEE Transactions on Industrial Electronics*, 56(9):3717–3722, 2009.

[115] R. Merali and T. Barfoot. Occupancy grid mapping with Markov Chain Monte Carlo Gibbs sampling. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3183–3189, Karlsruhe, Germany, May 2013.

[116] M. J. Milford and G. F. Wyeth. Mapping a suburb with a single camera using a biologically inspired SLAM system. *IEEE Transactions on Robotics*, 24(5):1038–1053, Oct. 2008.

[117] P. Milne. *Underwater acoustic positioning systems*. Gulf Publishing Company, Houston, 1983.

[118] R. Morton and E. Olson. Robust sensor characterization via max-mixture models: GPS sensors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 528–533, Tokyo, Japan, Nov. 2013.

[119] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application*, pages 331–340, 2009.

[120] S. Negahdaripour. On 3D motion estimation from feature tracks in 2D FS sonar video. *IEEE Transactions on Robotics*, 29(4):1016–1030, 2013.

[121] S. Negahdaripour and P. Firoozfam. An ROV stereovision system for ship-hull inspection. *IEEE Journal of Oceanic Engineering*, 31(3):551–564, 2006.

[122] J. Neira and J. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, Dec. 2001.

[123] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, Basel, Switzerland, Oct. 2011.

[124] K. Ni, D. Steedly, and F. Dellaert. Tectonic SAM: Exact, out-of-core, submap-based SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1678–1685, Roma, Italy, Apr. 2007.

[125] T. Nicosevici, N. Gracias, S. Negahdaripour, and R. Garcia. Efficient three-dimensional scene modeling and mosaicing. *Journal of Field Robotics*, 26(10): 759–788, 2009.

[126] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, New York, NY, USA, June 2006.

[127] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006.

[128] H. Ogawa. Labeled point pattern matching by Delaunay triangulation and maximal cliques. *Pattern Recognition*, 19(1):35–40, Jan. 1986.

[129] E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3400–3407, Shanghai, China, May 2011.

[130] E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. In *Proceedings of the Robotics: Science & Systems Conference*, Sydney, Australia, July 2012.

[131] E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. *International Journal of Robotics Research*, 32(7):826–840, 2013.

[132] E. Olson, M. Walter, S. Teller, and J. J. Leonard. Single-cluster spectral graph partitioning for robotics applications. In *Proceedings of the Robotics: Science & Systems Conference*, pages 265–272, Cambridge, MA, USA, June 2005.

[133] P. Orbanz and Y. W. Teh. Bayesian nonparametric models. In *Encyclopedia of Machine Learning*, pages 81–89. Springer, 2010.

[134] P. Ozog and R. M. Eustice. On the importance of modeling camera calibration uncertainty in visual SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3762–3769, Karlsruhe, Germany, May 2013.

[135] P. Ozog and R. M. Eustice. Real-time SLAM with piecewise-planar surface models and sparse 3D point clouds. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1042–1049, Tokyo, Japan, Nov. 2013.

[136] P. Ozog and R. M. Eustice. Toward long-term, automated ship hull inspection with visual SLAM, explicit surface optimization, and generic graph-sparsification. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3832–3839, Hong Kong, China, June 2014.

[137] P. Ozog, N. Carlevaris-Bianco, A. Kim, and R. M. Eustice. Long-term mapping techniques for ship hull inspection and surveillance using an autonomous underwater vehicle. *Journal of Field Robotics*, 2015. In Press.

[138] P. Ozog, G. Troni, M. Kaess, R. M. Eustice, and M. Johnson-Roberson. Building 3D mosaics from an autonomous underwater vehicle, Doppler velocity log, and 2D imaging sonar. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1137–1143, Seattle, WA, USA, May 2015.

[139] J. Padial, S. G. Dektor, and S. M. Rock. Correlation of sidescan sonar acoustic shadows and bathymetry for terrain-relative navigation. In *Unmanned Untethered Submersible Technology*, Portsmouth, NH, USA, Sept. 2013.

[140] K. Pathak, A. Birk, N. Vaskevicius, M. Pfingsthorn, S. Schwertfeger, and J. Poppinga. Online three-dimensional SLAM by registration of large planar surface segments and closed-form pose-graph relaxation. *Journal of Field Robotics*, 27(1):52–84, 2010.

[141] O. Pizarro, R. M. Eustice, and H. Singh. Large area 3-D reconstructions from underwater optical surveys. *IEEE Journal of Oceanic Engineering*, 34(2):150–169, 2009.

[142] *Acoustic Doppler current profiler: principles of operation: a practical primer*. RD Instruments, San Diego, CA, USA, 2 edition, Jan. 1996. P/N 951-6069-00.

[143] *ADCP coordinate transformation: formulas and calculations*. RD Instruments, San Diego, CA, USA, July 1998. P/N 951-6079-00.

[144] D. Ribas, P. Ridao, and J. Neira. *Underwater SLAM for structured environments using an imaging sonar*, volume 65 of *Springer Tracts in Advanced Robotics*. Springer, 2010.

[145] P. Ridao, M. Carreras, D. Ribas, and R. Garcia. Visual inspection of hydroelectric dams using an autonomous underwater vehicle. *Journal of Field Robotics*, 27(6): 759–778, 2010.

[146] C. Roman and H. Singh. Improved vehicle based multibeam bathymetry using sub-maps and SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2422–2429, Edmonton, Alberta, Canada, Aug. 2005.

[147] J. Roth, Y. Tong, and X. Liu. Unconstrained 3D face reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2606–2615, Boston, MA, USA, June 2015.

[148] M. Roy, S. Foufou, and F. Truchetet. Mesh comparison using attribute deviation metric. *International Journal of Image and Graphics*, 4(1):127–140, 2004.

[149] M. Ruhnke, R. Kummerle, G. Grisetti, and W. Burgard. Highly accurate 3D surface models by sparse surface adjustment. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 751–757, Shanghai, China, May 2012.

[150] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

[151] W. J. Schroeder, B. Lorensen, and K. Martin. The visualization toolkit. `http://vtk.org`, 2004.

[152] J. Schwendner, S. Joyeux, and F. Kirchner. Using embodied data for localization and mapping. *Journal of Field Robotics*, 31(2), Apr. 2013.

[153] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proceedings of the Robotics: Science & Systems Conference*, Seattle, WA, USA, June 2009.

[154] Y. Shan, Z. Liu, and Z. Zhang. Model-based bundle adjustment with application to face modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 644–651, Vancouver, Canada, 2001.

[155] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to autonomous mobile robots*. MIT Press, 2011.

[156] H. Singh, R. Armstrong, F. Gilbes, R. Eustice, C. Roman, O. Pizarro, and J. Torres. Imaging coral I: Imaging coral habitats with the SeaBED AUV. *The Journal for Subsurface Sensing Technologies and Applications*, 5(1):25–42, 2004.

[157] P. Smith, I. D. Reid, and A. J. Davison. Real-time monocular SLAM with straight lines. In *Proceedings of the British Machine Vision Conference*, pages 17–26, Edinburgh, Scotland, Sept. 2006.

[158] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Proceedings of Uncertainty in AI*, pages 435–461. Elsevier, 1986.

[159] Sound Metrics Corp. Didson 300 m Imaging Sonar. Specification sheet and documentations Available at "http://www.soundmetrics.com", 2014.

[160] N. Sunderhauf and P. Protzel. Switchable constraints for robust pose graph SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1879–1884, Algarve, Portugal, Oct. 2012.

[161] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[162] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. Point-plane SLAM for hand-held 3D sensors. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5182–5189, Karlsruhe, Germany, May 2013.

[163] S. Thrun and M. Montemerlo. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research*, 25(5-6):403–429, May 2006.

[164] A. J. B. Trevor, J. G. Rogers, and H. I. Christensen. Planar surface SLAM with 3D and 2D sensors. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3041–3048, St. Paul, MN, USA, 2012.

[165] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 298–372, Corfu, Greece, 1999.

[166] G. Trimble and E. Belcher. Ship berthing and hull inspection using the CetusII AUV and MIRIS high-resolution sonar. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1172–1175, Biloxi, MS, USA, 2002.

[167] Tritech. Eclipse 3d multibeam imaging sonar. http://tritech.co.uk/roduct-category/multibeams, 2015.

[168] G. Troni. *Advances in precision navigation of underwater vehicles*. PhD thesis, Deparment of Mechanical Engineering, Johns Hopkins University, Baltimore, MD, 2013.

[169] J. Vaganay, M. Elkins, D. Esposito, W. O'Halloran, F. Hover, and M. Kokko. Ship hull inspection with the HAUV: US Navy and NATO demonstrations results. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, pages 1–6, Boston, MA, Sept. 2006.

[170] O. M. Van Kaick, M. V. G. da Silva, W. Schwartz, and H. Pedrini. Fitting smooth surfaces to scattered 3D data using piecewise quadratic approximation. In *Proceedings of the International Conference on Image Processing*, pages 493–496, Rochester, NY, Sept. 2002.

[171] M. VanMiddlesworth, M. Kaess, F. Hover, and J. Leonard. Mapping 3D underwater environments with smoothed submaps. In *Proceedings of the International Conference on Field and Service Robotics*, pages 17–30, Brisbane, Australia, Dec. 2013.

[172] M. Walter, F. Hover, and J. Leonard. SLAM for ship hull inspection using exactly sparse extended information filters. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1463–1470, Pasadena, CA, USA, May 2008.

[173] J. Weingarten and R. Siegwart. 3D SLAM using planar segments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3062–3067, Beijing, China, Oct. 2006.

[174] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, J. Mcdonald, and J. J. Leonard. Kintinuous : Spatially extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Sydney, Australia, July 2012.

[175] R. T. Whitaker and E. L. Juarez-Valdes. On the reconstruction of height functions and terrain maps from dense range data. *IEEE Transactions on Image Processing*, 11 (7):704–716, 2002.

[176] L. Whitcomb, D. Yoerger, and H. Singh. Advances in Doppler-based navigation of underwater robotic vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 399–406, Detroit, MI, USA, May 1999.

[177] O. Williams and A. Fitzgibbon. Gaussian process implicit surfaces. In *Gaussian Processes in Practice Workshop*, Bletchley Park, United Kingdom, Apr. 2007.

[178] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. Efficient simultaneous localisation and mapping using local submaps. In *Proceedings of the Australian Conference on Robotics and Automation*, pages 128–134, Sydney, Australia, Nov. 2001.

[179] R. W. Wolcott and R. M. Eustice. Visual localization within LIDAR maps for automated urban driving. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183, Chicago, IL, USA, Sept. 2014.

[180] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.