Master's Thesis

Localization and Active Exploration in Indoor Underwater Environments

Sudharshan Suresh

CMU-RI-TR-19-61

August 2019

Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Michael Kaess, Chair
David Wettergreen
Maxim Likhachev
Eugene Fang

Submitted in partial fulfillment of the requirements for the degree of Master of Science.

Keywords: SLAM; underwater robotics; active exploration

Abstract

Autonomous underwater vehicles have the potential to inspect and map indoor underwater environments, such as spent nuclear fuel pools and ship ballast tanks. These environments are to be regularly monitored for structural integrity—existing manual methods are expensive, dangerous and slow. Employing an autonomous agent presents distinct challenges in SLAM and exploration. This thesis makes contributions in the domains of visual localization and active SLAM for these environments.

First, we propose a novel through-water method for visual localization using landmarks above the water surface. With dead-reckoning, the vehicle pose estimate drifts and the errors propagate to the resultant map. Adopting methods from multimedia photogrammetry in our localization framework, we model refraction at the water-air interface. To the best of our knowledge, this is the first through-water method for underwater localization. We evaluate our method via both simulation and real-world experiments in a test-tank environment.

The second work presents an active SLAM framework for sonar mapping of these environments. Accurate mapping requires jointly considering the robot trajectory with the state estimation problem. Building on previous work in mapping and planning, we devise an exploration policy that bounds pose uncertainty through revisit actions. A revisit policy is selected based on submap saliency, propagated pose uncertainty, and path information gain. We demonstrate the system in simulation and highlight the advantages over an uncertainty-agnostic framework.

Acknowledgments

I first thank my advisor—Michael Kaess—for his thoughtful guidance, steadfast support, and utmost patience. Over these two years, I have benefited from his counsel and grown as a robotics researcher. He gave me free rein to explore and dabble to my heart's content, while shielding me from anything that isn't research. This would not be possible without the support system that is the robot perception lab (circa 2017-19): Eric[1], Paloma, Ming, Monty, Jack, Nora, Zimo, Jerry, Bing, Akshay, Eric[2], Josh, Wei, Allie, and Chen. They've kept me happy and intellectually stimulated in 2204 despite its dearth of natural light. This body of work is a result of their advice, insights, and encouragement. I am grateful to the rest of my committee—David Wettergreen, Maxim Likhachev, and Eugene Fang—for their time, feedback, and thoughtful questions. I thank Red Whittaker for his guidance and support, which predates my time here as a graduate student. I am indebted to Rachel Burcin and John Dolan—without the RISS program I wouldn't even be here in the first place.

I am incredibly grateful to all my friends across time zones—in Pittsburgh, the USA, and my wonderful hometown of Chennai, India. Finally and most importantly, I thank my family who put my needs before theirs so I could lead a privileged life. It is their love and support that empowers me to follow my dreams.

I further acknowledge funding support from the Department of Energy and Office of Naval Research for this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Autonomous underwater vehicles (AUVs) can conduct inspection tasks in complex underwater environments. They have the potential to create high-fidelity maps of such areas with minimal manual intervention. These include underwater structures like ship-hulls, dams, pipelines, reactor pressure vessels, spent nuclear fuel (SNF) pools, and ship ballast tanks. These environments are required to be regularly inspected for ensuring structural integrity and safety. This body of work focuses on a subset of these environments—*indoor underwater environments*—such as SNF pools and ballast tanks. Examples of these environments are shown in Fig. 1.1.

The prevalent workflow for inspecting SNF storage pools is manual, expensive and dangerous. Human inspectors lower telescopic cameras mounted on poles into the pool to identify structural deficiencies (Fig. 1.1). The process is slow and prohibitively expensive, with the personnel position themselves on platforms above the water surface. There is incomplete coverage, as they perform coarse manipulation of the device and cannot access complex geometries. In ship ballast tanks, the inspections are usually conducted in the dry-docks and can cost up to 800 thousand dollars [21]. Moreover, regular inspection is required as seawater accelerates structural corrosion. AUV inspection can reduce downtime, with inspections during active operation of the ship.

Having established the advantages of robotic inspection, we now consider the sensing capabilities of service AUVs. This generally includes proprioceptive sensors for 6-DoF state estimate, visual information via onboard camera, and 3-D information from sonar. Visual methods are viable due to excellent visibility and absence of open-sea error sources such as surface disturbances. Sonar has been successfully used in challenging scenarios, such as bridge and ship-hull inspection [40, 64]. Attenuation of signals prohibit the use of a global positioning system (GPS), except when resurfacing. It is required that all sensors are subject to radiation hardening.

Despite this high-precision sensing payload, the uncertainty of vehicle dead-reckoning state estimates grows. The accumulated drift consequently affects the quality of the generated map. Fig. 1.2 illustrates separate instances of odometry drift and inconsistent global map during inspection.

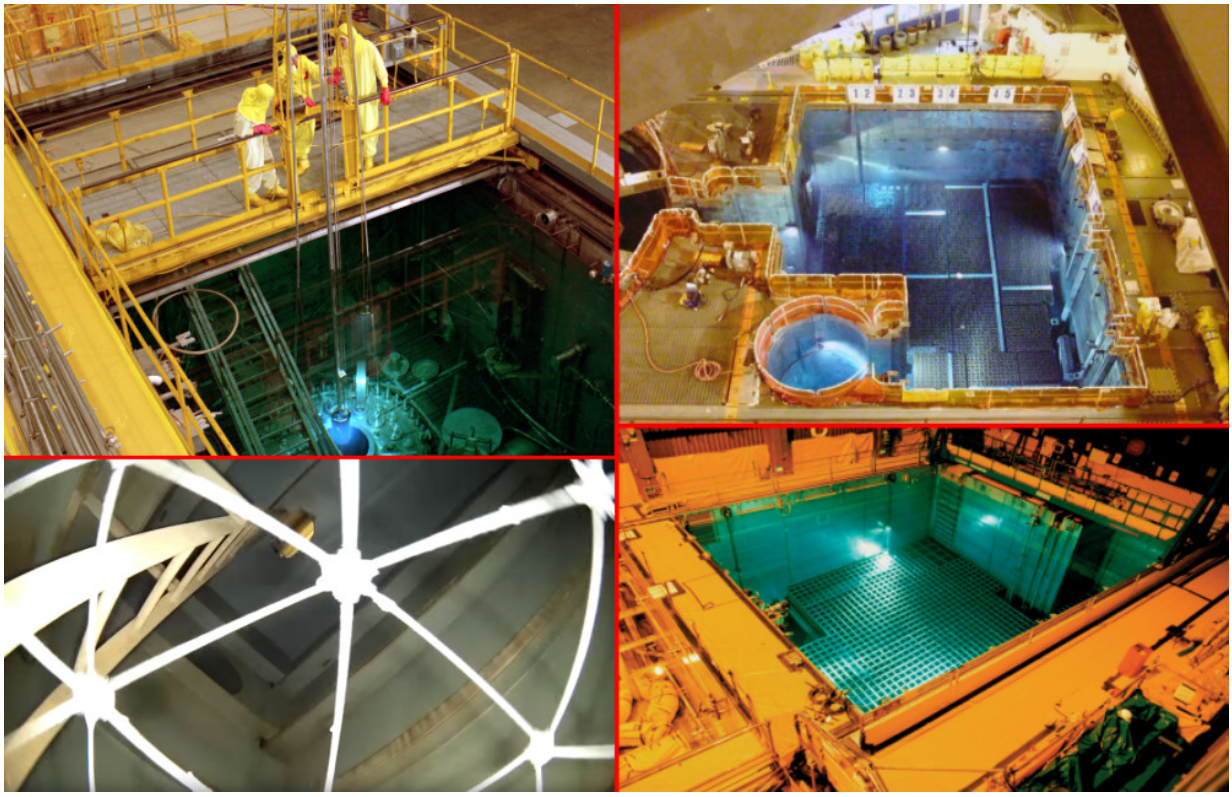


Figure 1.1: Examples of indoor underwater environments that require regular inspection. *(top-left)* Status quo of SNF pool inspection involves human inspectors lowering a camera into the pool [2]. *(top-right)* and *(bottom-right)* show further examples of these environments [3, 4]. *(bottom-left)* shows an inspection solution for ship ballast tanks with a drone [1].

These navigational shortcomings are generally handled by simultaneous localization and mapping (SLAM) frameworks. This is formulated as a probabilistic inference problem over the robot's noisy sensor data, and has shown success in the underwater applications [69, 86].

In addition, teleoperation is difficult in cluttered underwater environments. Instead, the vehicle can perform inspection via an exploration policy for volumetric coverage. In the absence of informative loop closures, vehicle state estimates can drift and give rise to ill-advised behavior. This motivates an active SLAM approach—where the robot performs deliberate actions to complement mapping and localization, while maintaining a safe exploration policy.

1.2 Scope and Approach

In this thesis, we discuss two methods which address underwater visual localization and active exploration respectively. The work analyzes the potential of vision and sonar modalities individually, and motivates their combined use as future work. We develop these methods for the class of hovering autonomous underwater vehicles (HAUVs) [90] (Fig. 2.2).

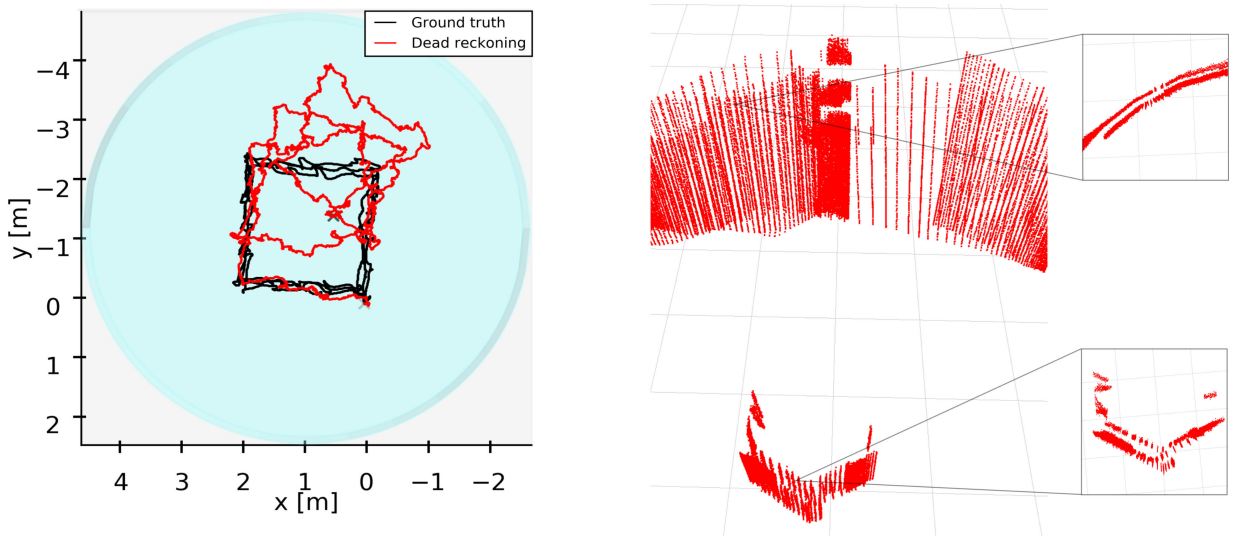


Figure 1.2: *(left)* Gradually drifting dead-reckoning estimates of the underwater vehicle when executing a square trajectory. *(right)* An example of a globally inconsistent map due to drifting odometry—the insets show point cloud misalignment.

The first part—on visual localization—explores a novel *through-water* stereo framework for drift-free vehicle pose estimates. As highlighted in Section 1.1, visual perception is viable in these indoor underwater environments. We look through the refractive water-air interface and track visual features in an incremental optimization framework. We establish the theory behind refraction correction, borrowing ideas from multimedia photogrammetry. This is applied to the HAUV in simulation, and in our real-world environment.

The second part—on active SLAM—devises a safe exploration policy for sonar mapping that bounds pose uncertainty. It builds upon the existing virtual occupancy grid map (VOG-Map) framework, which combined pose graph submap SLAM with a sampling-based planner [37]. Our method biases the vehicle towards revisitation if there is large drift in its pose estimate, and towards *next-best-view* [11] exploration otherwise. We introduce the idea of *global submap saliency* (GloSSy) metric for good revisit pose candidates. Further, we evaluate a revisit penalty term based on propagated uncertainty and path information gain. We demonstrate that our method has advantages over a standard exploration policy in simulation experiments with the HAUV in a cluttered underwater tank environment.

1.3 Contributions and Organization

As highlighted in Section 1.2, the thesis can be organized into two chapters. In Chapter 3, we propose a SLAM formulation for AUVs using an onboard upward-facing stereo camera. To the best of our knowledge, this is the first through-water visual localization technique for underwater vehicles. Concisely, our main contributions are[1]:

[1]Supplementary video : `https://youtu.be/fZZTDyLymBs`

- an upward-facing stereo SLAM method for AUV localization with a ceiling feature map,
- a refraction correction module for through-water vision, modeled after prior work in multimedia photogrammetry, and
- evaluation in both simulation and real-world settings.

In Chapter 4, we present an active SLAM method for an AUV in cluttered small-scale underwater environments. Built upon the VOG-Map framework [37], it enables the robot to plan, explore and map an apriori unknown environment while considering pose uncertainty. Here, our contributions are:

- a global submap saliency (GloSSy) metric to identify revisit poses for reliable loop closures,
- a revisit penalty term based on propagated pose uncertainties and view utility gain, and
- experiments in simulation, for an underwater tank environment.

The thesis is organized as follows. Chapter 2 covers the preliminary theory on SLAM, factor graphs, and introduces the platform. Chapter 3 describes the theory and implementation of the through-water SLAM method. Chapter 4 details active submap SLAM for underwater exploration. Finally, Chapter 5 recaps the contributions of the thesis and discusses future research directions.

Chapter 2

Preliminaries

2.1 SLAM and Factor Graphs

This section acts as a primer on SLAM, and introduces the reader to a representation of the inference problem—factor graphs. A robot must perform two concurrent tasks—**(i)** infer where it is in the environment (localization) **(ii)** construct a map of the environment (mapping). This involves getting the best estimate from inference over noisy sensory data and the robot's motion model. Initial probabilistic methods used filtering frameworks—such as the extended Kalman filter (EKF) [83]. While effective, they are limited by their compute cost and linearization errors.

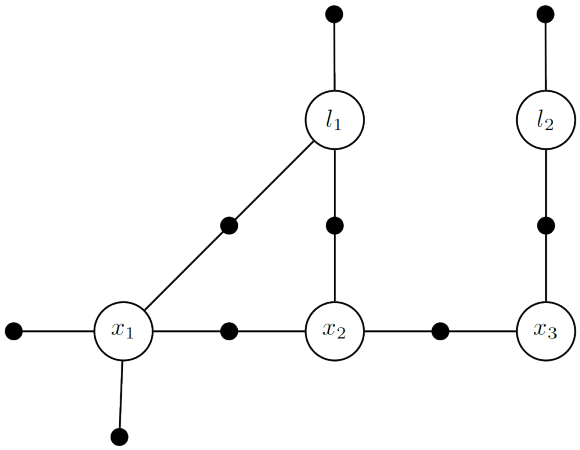


Figure 2.1: A sample factor graph representing poses and landmarks, taken from [23]. Variable nodes are the large circles which are poses (x_i) or landmarks (l_i), while measurement factors are denoted by smaller circles.

Optimization-based SLAM solves a nonlinear least-square problem. This is more accurate than filtering methods as it preserves the history of cost functions over timesteps to compute the optimal solution. Subsequent work by Kaess et al. [48, 49] developed an efficient solving strategy—incremental smoothing and mapping (iSAM). Instead of re-calculating the entire system each time, it updates the previous matrix factorization with the new measurements. The sparse nature of the system (i.e. pose-landmark connectivity) ensures computational efficiency. For a

high-level summary, Cadena et al. [12] provides a comprehensive overview of SLAM, recent progress, and future directions.

Given sensory data and timesteps, we represent the problem as a factor graph—as commonly done in the SLAM literature. A factor graph is a bipartite graph comprised of *variables* to be optimized and *factors* that constrain the system. The variable nodes represent the state we wish to estimate and the factors are the measurements obtained from sensors. An example—taken from [23]—is shown in Fig. 2.1. It represents a SLAM problem where the state $\mathcal{X}$ comprises of poses x_i and landmarks l_j. Measurements $\mathcal{Z}$ between nodes are binary factors (e.g. odometry, camera measurements, and loop closures) and the rest are unary factors (e.g. GPS measurements, pose priors).

We compute the *maximum a posteriori* (MAP) estimate, which predicts variable values that maximally agree with the given measurements:

$$\begin{aligned} \mathcal{X}^* &= \underset{\mathcal{X}}{\operatorname{argmax}} \; p(\mathcal{X}|\mathcal{Z}) \\ &= \underset{\mathcal{X}}{\operatorname{argmax}} \; p(\mathcal{X})\, p(\mathcal{Z}|\mathcal{X}) \\ &= \underset{\mathcal{X}}{\operatorname{argmax}} \; p(\mathcal{X})\, l(\mathcal{X};\mathcal{Z}) \\ &= \underset{\mathcal{X}}{\operatorname{argmax}} \; p(\mathcal{X}) \prod_{i=1}^{N} l(\mathcal{X};\mathbf{z_i}) \end{aligned} \tag{2.1}$$

where $l(\mathcal{X};\mathcal{Z})$ is proportional to $p(\mathcal{Z}|\mathcal{X})$ and denotes the likelihood of state $\mathcal{X}$ given measurements $\mathcal{Z}$. The next step does the same, instead as a product of individual measurements z_i. This makes the assumption of conditional independence of measurements, as encoded in the factor graph (Fig. 2.1). In Section 3.4.1 and 4.2.3, we revisit this math and demonstrate how the Gaussian noise assumption reduces the inference to a nonlinear least-squares problem.

2.2 Hovering Autonomous Underwater Vehicle

The methods in the thesis generalize to the class of hovering autonomous underwater vehicles (HAUVs). They can hover in place and are used for a variety of operations such as inspection, surveying and scientific research. Some examples of these vehicles are shown in Fig. 2.2. In our experiments, we use the HAUV from Bluefin Robotics [90] (Fig. 2.3). It has five thrusters and is controllable in all degrees of freedom, except pitch and roll. The platform has been successfully employed for ship inspection [40, 86], as well as in indoor underwater applications [37, 94].

Vehicle Payload

The vehicle's payload is comprised of a Doppler velocity log (DVL), attitude and heading reference system (AHRS) and depth sensor, with measurements characterized as follows:

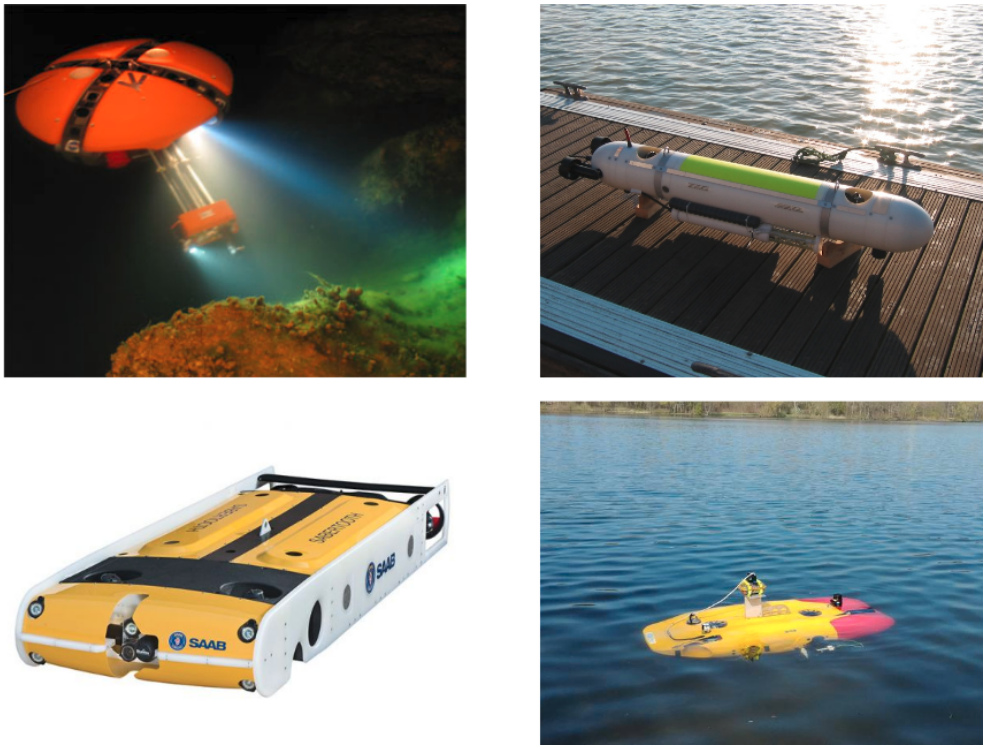


Figure 2.2: The class of hovering autonomous underwater vehicles that the thesis focuses on. These include the DepthX [29], MARES AUV [15], Sabertooth [79], and the AUV-Dagon [16]. Our own vehicle, the Bluefin HAUV, pictured in 2.3.

(i) The depth sensor provides direct measurements of HAUV depth (Z).

(ii) The AHRS observes gravity to give drift-free pitch and roll estimates.

(iii) The X, Y and yaw quantities are obtained via dead reckoning, which drift with operation.

Such a configuration is common among underwater vehicles, and is modelled in both our simulation and real-world experiments. Using high-precision navigation sensors, the proprietary odometry of our vehicle exhibits very low drift over the relatively short time frames of operation.

Figure 2.3: Underwater robot used in real-world experiments with its sensing payload visible. The DVL (**D**), stereo camera (**C**), and DIDSON sonar (**S**) are fixed in front, with the camera facing upwards.

In all our experiments, we treat the vehicle odometry as the *ground truth*. We corrupt the relative odometry between poses with significant additive white Gaussian noise. This induces drift in the XY plane to mimic having a less accurate inertial measurement unit (IMU) + DVL payload, as usually seen in underwater applications. This is summarized in Fig. 2.4.

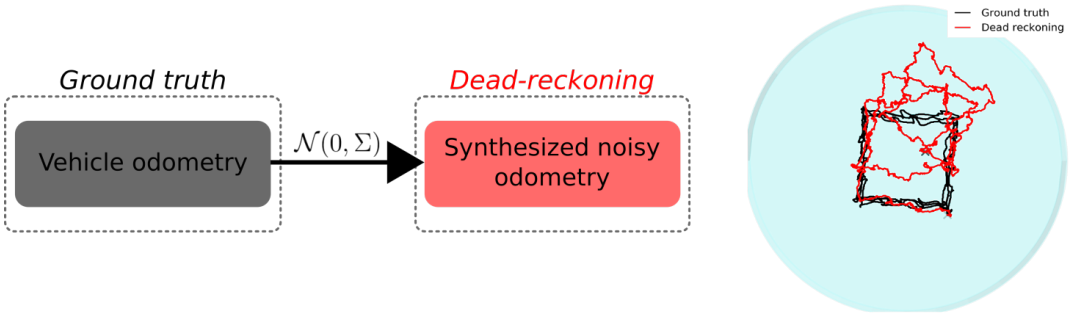


Figure 2.4: *(left)* Schematic depicting noise addition to vehicle odometry to get a corrupted estimate. *(right)* Example square trajectory shows drifting dead-reckoning with operation.

Stereo Camera

The stereo pair consists of two Prosilica GC1380 cameras fixed adjacent to the DVL, oriented upwards (Fig. 2.3). It has a 0.078 m baseline and records 5 fps grayscale images (680×512). We calibrate the stereo camera underwater and manually measure the camera-robot transformation. Images are corrected for radial and tangential distortion.

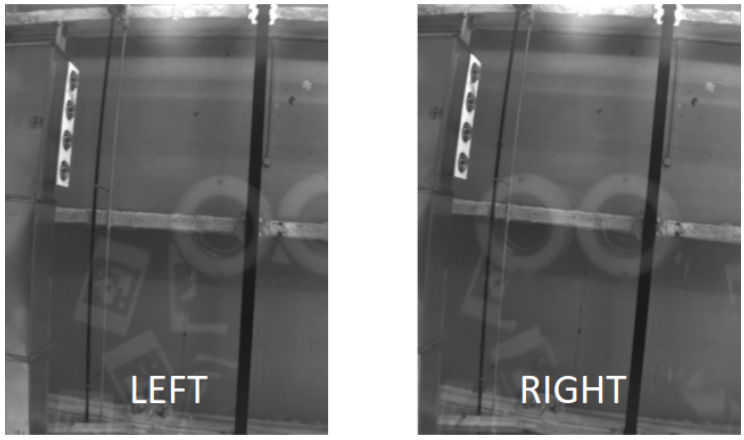


Figure 2.5: Stereo imagery from upward facing stereo camera. We use visual features for localization in Chapter 3.

DIDSON Sonar

Sonar is the primary sensing modality in Chapter 4, and its configuration resembles that in [37, 86]. The sensor has a phased array of transducers, operated in the profiling mode. These transducers produce beams with 14° vertical and 0.3° horizontal width. Every sonar scan sweeps

a horizontal arc, comprising of 96 beams. We further avoid ambiguity in the vertical FoV with a concentrator lens. Teixeira et al. [86] performs post-processing and extracts range from the sonar data, ultimately representing each scan as points on a horizontal plane. The HAUV possesses only one rotational degree of freedom, yaw, thus the sonar is rotated by 90° with respect to the body frame (Fig. 2.6). The vehicle can rotate and translate in the environment, and the sonar sweeps the free-space volume.

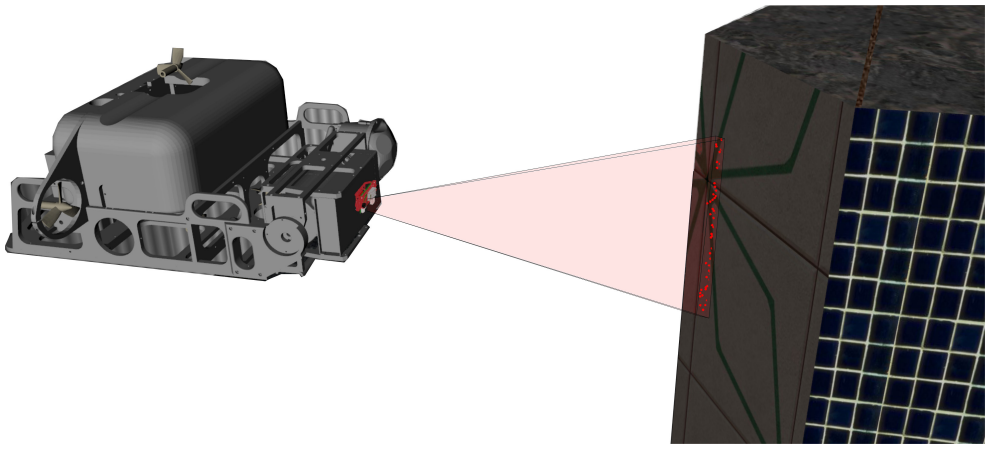


Figure 2.6: Simulated HAUV with DIDSON sonar imaging an object in environment. The red scan line is approximate representation of the sonar beam when filtered by the concentrator lens. This model is used for simulation experiments in Section 4.5

Chapter 3

Localization: Through-water Visual SLAM

3.1 Introduction

There is a need for autonomous inspection and mapping of underwater environments such as SNF pools. However, long-term operation causes drift in pose estimate if it is solely reliant on dead reckoning. Existing line-of-sight and vision-based localization methods are not adaptable to cluttered environments, especially when no modifications can be made to the surroundings. We note that while underwater sections of the pools may not have uniformly distributed visual features, ceilings are structurally rich (or can be modified to be so). There is also good visibility in the pools, as there is little-to-no turbidity. These factors motivate a visual localization method for AUVs that looks through the water surface.

A problem of interest is—how do we model refraction at the water-air interface? This work builds on existing literature in through-water photogrammetry, but, unlike them, considers underwater cameras observing landmarks in air. The scope of our method widens when we consider other applications that can benefit from it. Commercial swimming pool cleaning robots are platforms that have cheaper navigational payloads. Here, our localization framework could work standalone, or act as a fail-safe for existing visual methods [13]. This is also viable for exploration robots in partially submerged caves, with active lighting [92]. To researchers, the method can deliver ground-truth estimates in the absence of expensive underwater motion capture systems.

3.2 Background and Related Work

The problem of underwater localization has received considerable attention over the years. Numerous sensing modalities and algorithms have been explored, as documented by Paull et al. [71]. Recently, Nawaz et al. [65] proposed an acoustic sensor network to localize a robot swarm in a nuclear storage pond (Fig. 3.2), while Rust et al. [77] used visible light to localize an ROV in a nuclear reactor. Both methods suffer from attenuation in cluttered environments with multi path transmission.

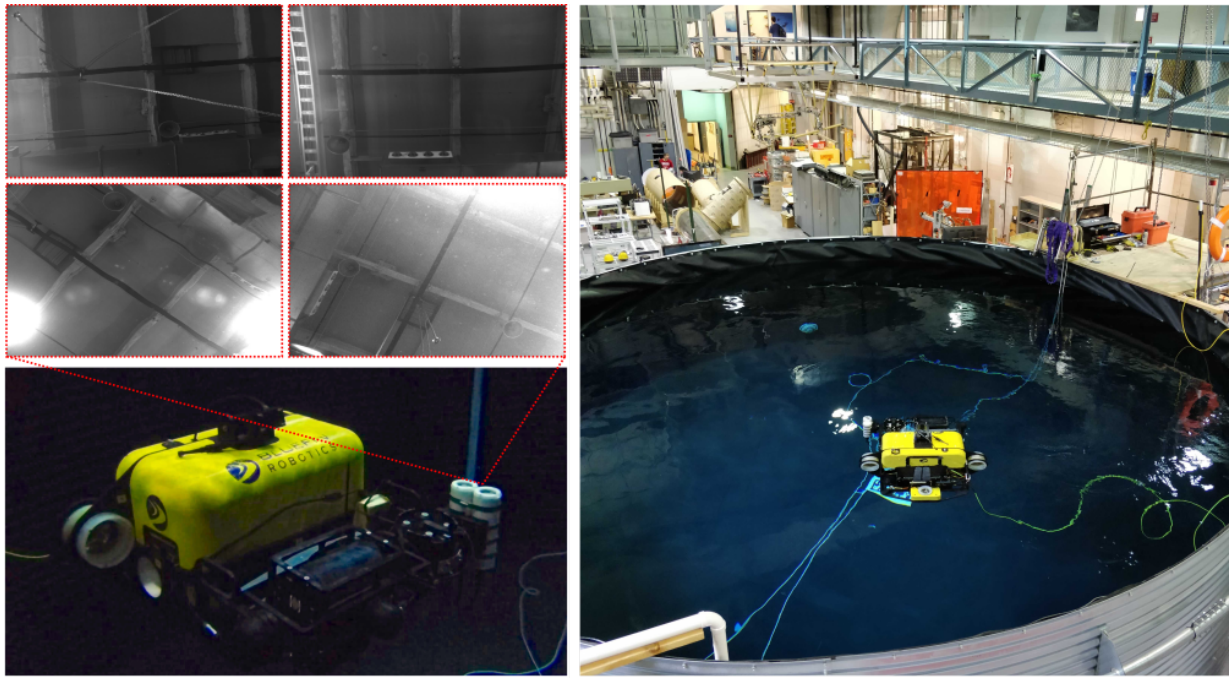


Figure 3.1: *(top-left)* A sampling of ceiling frames taken from the stereo pairs. These highlight challenges for the frontend, including motion blur, light scattering and particulates. *(bottom-left)* An underwater still of our AUV executing a trajectory in the test environment at a depth of 1m. The upward-facing stereo camera views the ceiling through the water interface. *(right)* Our test tank environment, with the vehicle executing a trajectory.

Jung et al. [44, 45] developed visual localization methods for AUVs by installing fiducials underwater. However, spent nuclear pools cannot be modified due to radioactivity and often have nuclear waste and thick sludge deposition at the bottom. Cho et al. [18] performed 3-DoF state estimation of a robot in a reactor vessel through an external camera, by viewing LEDs on the vehicle frame (Fig. 3.2). Later, Lee et al. [54] used a submerged camera and prior map to obtain a full 6-DoF state estimate through fiducial tracking (Fig. 3.2). Both methods, however, are affected by clutter in the line-of-sight between the camera and robot. Consequently, they do not scale to larger environments.

The field-of-view of an upward-facing camera on an AUV is not obstructed while navigating these cluttered environments (Fig. 3.1). Ceilings in most such environments have robust structural cues for localization, with several notable examples for ground vehicles [30, 42]. In this problem formulation, feature points triangulate to landmarks in air, viewed through the water-air interface. Refraction causes light to bend at the interface, and generates a systematic error in heights calculated from stereo correspondences. This creates large geometric errors in the global map, and affects the optimized trajectory estimate. To achieve the true SLAM solution, we must explicitly model for the refraction.

Refraction correction was first explored in aerial photogrammetry for shallow water [60, 87]. These works obtain actual water depth from an analytic plotter by applying a correction factor. Fryer et al. later demonstrated that two-camera photogrammetry of submerged objects can only be

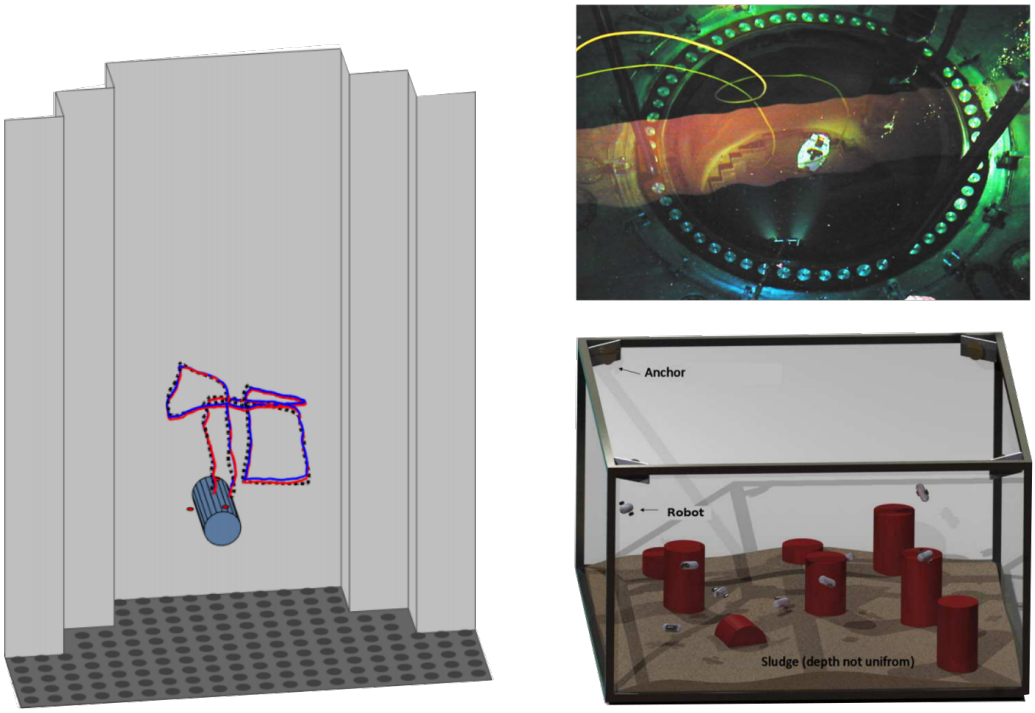


Figure 3.2: *(left)* Full 6-DoF state estimation with a submerged camera, monitoring a robot in a nuclear reactor [54]. *(top-right)* An external camera performs 3-DoF state estimation of a robot in a reactor vessel [18]. *(bottom-right)* Acoustic sensor network for a nuclear storage pond [65].

approximated, as rays from an object have different incident angles with the cameras [31]. This procedure was used to create underwater topographical maps of river beds and reefs [63, 93]. All prior through-water methods (i) are not in the context of SLAM and (ii) consider aerial cameras observing underwater objects.

3.3 Refraction Observation Model

A routine operation in any visual SLAM framework is the projection of 3-D points to image pixels and the corresponding backprojection of 2-D image points to 3-D points. When operating in a single medium, this is a trivial operation given camera intrinsics and extrinsics. We require compensation factors that enable these operations in a dual-media setting. Thus, based on previous work, we introduce methods for refraction-corrected stereo triangulation (Section 3.3.2) and refraction-corrected projection (Section 3.3.3). However, the prior algorithms were for aerial photogrammetry through shallow water. We adapt them to the inverse problem of an underwater camera observing points in air. Our SLAM framework (Section 3.4) uses this module at every stage for true landmark positions.

A fixed-baseline stereo camera is calibrated underwater and has known, constant camera-robot extrinsics. When a light ray enters the camera housing, it passes from $water \rightarrow glass \rightarrow air$. The single viewpoint (SVP) pinhole camera model is found to be theoretically inaccurate due

to refraction at the camera housing [89]. Agrawal et al. later show that these cameras can be expressed in terms of an axial model [6] . This implies that light rays intersect an axial line rather than a single point. The validity of any approximation thus depends on the length of this axial line. Luczynski et al. study this closely to conclude that an SVP pinhole model is a valid approximation if the center of projection and flat port and very close to each other [57]. This is always the case for underwater housings—the camera is placed very near the flat port.

Thus, we adopt the pinhole camera model—refraction at the camera's housing is accounted for in the lens distortion parameters. This approximation has worked well for prior work in underwater imaging [10, 43, 66]. The camera viewing direction is not required to be perpendicular to the water surface. Lacking this assumption, we cannot model refraction at the water interface as a radial distortion [80]. We establish a sign-convention for the Z direction: the water surface is the XY plane, points in air are negative and points underwater are positive. The *apparent* landmark is that triangulated without considering refraction at the interface. The *true* landmark is that obtained from explicitly modeling this refraction.

3.3.1 Assumptions

Sections 3.3.2 and 3.3.3 make some simplifications, and we discuss them here for completeness:

(i) We assume the water surface is planar, which is commonly done in through-water methods. This allows us to apply the laws of refraction to the problem. Prior work makes this simplification not only for indoor environments, but also in reefs with minor waves [63, 87].

(ii) It is assumed that we know the pose of the cameras. This is valid as we possess knowledge of our vehicle's pose, or more precisely, the pose estimate from the factor graph optimization. We transform this by the known, fixed extrinsics to get the camera poses.

(iii) We know the refractive indices of the media, $\mu_w = 1.33$ for water and $\mu_a = 1$ for air. These values can be modified depending on the media we consider (e.g. saline water).

(iv) We are given the pixel correspondences for the landmarks. This is obtained from our feature detector (Section 3.4.2).

3.3.2 Refraction-corrected Stereo Triangulation

Given pixel correspondences in an image pair, we wish to calculate the *true* position of a landmark. Fig. 3.3 (a) illustrates the geometry for a single point landmark observed by a stereo pair. The cameras are at positions c_1 and c_2, having depths H_1 and H_2 below the water surface respectively. The *apparent* landmark P' has a height h', while the *true* landmark P has a depressed height h. c_1' and c_2' are the interface intercept points obtained by tracing the rays from P' to c_1 and c_2 respectively. For rays from P to c_1 and c_2, the incidence angles with the interface are i_1, i_2 and refracted angles are r_1, r_2.

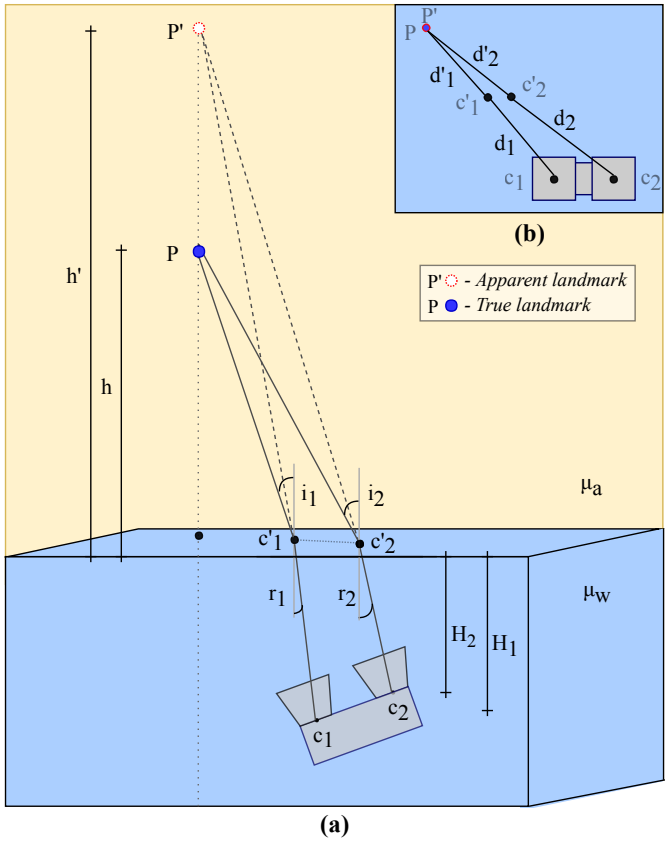


Figure 3.3: **(a)** Geometry of refraction-corrected stereo triangulation for a single landmark in air. While the stereo pair incorrectly triangulates a measurement to *apparent* position P', we perform correction to obtain the *true* position P. **(b) (inset)** top view of the geometry, showing directly observable quantities in the XY plane.

Fig. 3.3 (b) is the top view of Fig. 3.3 (a) showing the distances in the XY dimensions:

$$\begin{aligned} d_1 &= \|c_{1\mathbf{xy}} - c'_{1\mathbf{xy}}\| & d_2 &= \|c_{2\mathbf{xy}} - c'_{2\mathbf{xy}}\| \\ d'_1 &= \|c'_{1\mathbf{xy}} - P'_{\mathbf{xy}}\| & d'_2 &= \|c'_{2\mathbf{xy}} - P'_{\mathbf{xy}}\| \end{aligned} \tag{3.1}$$

From Fig. 3.3, r_1 and r_2 are:

$$r_1 = \tan^{-1}\left(\frac{d_1}{H_1}\right) \qquad r_2 = \tan^{-1}\left(\frac{d_2}{H_2}\right) \tag{3.2}$$

Snell's law relates the refractive indices of media with the direction of light propagation. Further, i_1 and i_2 are:

$$\frac{\sin i_1}{\sin r_1} = \frac{\sin i_2}{\sin r_2} = \frac{\mu_w}{\mu_a} \tag{3.3}$$

Knowing the angles of incidence, we obtain the corrected height of the landmark for each camera (h_{c_1}, h_{c_2}). From Fig. 3.3, in a similar fashion to Equation 3.2, we have:

$$h_{c_1} = d'_1 / \tan(i_1) \qquad h_{c_2} = d'_2 / \tan(i_2) \tag{3.4}$$

They are found to be slightly different, as no unique solution exists when rays from the *true* 3-D point landmark have different incident angles with the cameras [31]. However, an approximate solution suffices for landmark initialization. We take the average, giving the final corrected height:

$$h = (h_{c_1} + h_{c_2})/2 \tag{3.5}$$

Thus, refractive triangulation gives us the *true* position of landmarks. This ensures consistent triangulation regardless of robot location and assures geometrically accurate maps.

3.3.3 Refraction-corrected Projection

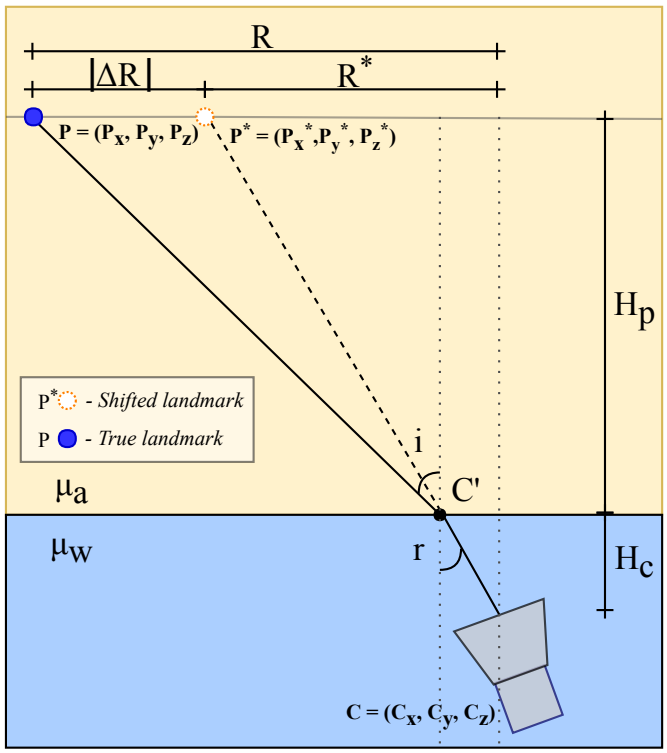


Figure 3.4: Radial shift geometry for the estimated *true* position of a landmark P with respect to camera center C. An iterative procedure converges to P^*, which is the *shifted* position of the landmark. This allows us to trivially project the 3-D landmark into the camera similar to a single-medium setting.

Given the *true* position of a landmark—P—we wish to project it to image coordinates. First, we calculate the *shifted* position P^* the camera views the landmark at. This is done by radially shifting it parallel to the water surface. P^* lies on the ray joining the *apparent* landmark position P' with the camera center (Fig. 3.3). Due to bending of light at the interface, *true* landmark position and camera center are no longer collinear. We radially shift the landmark with respect to the camera center before projection [58].

Fig. 3.4 shows the problem geometry when viewed perpendicular to the direction of the ray. The *true* landmark to be imaged is P, at a height H_p. The projection center of the camera viewing the landmark is C, at depth H_c. The incident and refracted rays make angles i, r with the interface. There is no closed form solution as C' is unknown—we follow an iterative method. This process is formalized in Algorithm 1.

We initialize the *shifted* radial distance R^* to the *true* radial distance R itself. Knowing its position and applying Snell's law, we can compute the angles i and r. The radial shift, ΔR, is

Algorithm 1 Iterative radial-shift for refraction correction.

$R^* = R = \sqrt{(P_x - C_x)^2 + (P_y - C_y)^2}$
repeat
  $r = \tan^{-1} \frac{R^*}{C_z + P_z}$
  $i = \sin^{-1} \left(\frac{\mu_w}{\mu_a} \sin r \right)$
  $\Delta R = R^* - (H_p \tan i + H_c \tan r)$
  $R^* \leftarrow R^* + \Delta R$
until $(|\Delta R| < \epsilon)$
$P_x^* = C_x + \frac{R^*}{R}(P_x - C_x)$
$P_y^* = C_y + \frac{R^*}{R}(P_y - C_y)$
$P_z^* = P_z$

computed as $\Delta R = R^* - R$. From Fig. 3.4:

$$\begin{aligned} R &= H_p \tan i + H_c \tan r \\ R^* &= (H_p + H_c) \tan r \end{aligned} \tag{3.6}$$

These are directly evident from the right triangles that i and r are part of. Using Equation 3.6, we compute ΔR at every iteration and radially shift the point until convergence (i.e. $|\Delta R| < \epsilon$). We convert the expression to Cartesian coordinates to get the *shifted* landmark that we can project trivially, as in the single-media case. In initial tests, we get convergence to within a few *cm* from ground truth in 10 iterations.

We implement the correction equations from Mass et al. without any modifications for convergence [58]. Maas et al. also mentions the introduction of an *overcompensation factor* as a way to accelerate the convergence of the iterative equation set. They do not, however, implement it as the choice of such a value c is non-trivial. It is dependent on the two refractive indices, the ratio of the path lengths in the two media, and the incidence angle. A constant overcompensation factor c cannot guarantee convergence, it must instead be varying and recomputed every time. Thus we settle for the vanilla method without an overcompensation factor. Future work could include constructing a lookup table to speed-up this operation.

3.4 Proposed SLAM Formulation

3.4.1 Factor Graph Representation

We represent the problem as a factor graph optimization, as explained in Section 2.1. This is graphically represented in Fig. 3.5. As discussed in Section 2.2, AUVs generally have a pressure sensor that directly observes depth (Z). Detecting the direction of gravity allows the IMU to provide absolute pitch and roll measurements. The remaining degrees of freedom—X, Y and

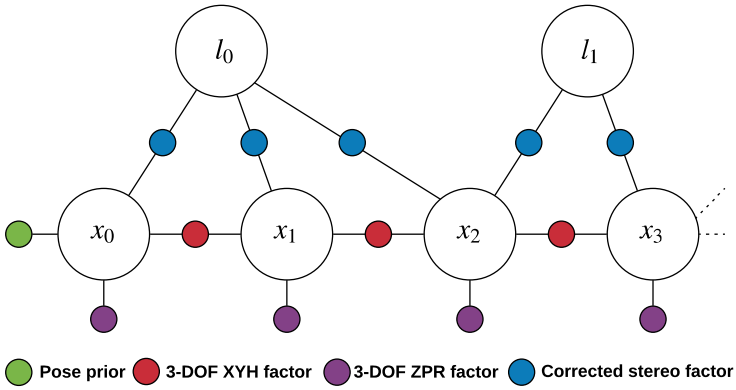


Figure 3.5: Factor graph representing our SLAM formulation. Variable nodes are the large circles that represent either poses (x_i) or landmarks (l_i). Measurement factors are denoted by smaller, colored circles. As opposed to conventional landmark-based stereo SLAM, our method incorporates a refraction-corrected stereo factor between poses and landmarks.

yaw—are obtained via dead reckoning and are subject to drift. This gives accurate measurements locally, but the pose estimate drifts over long dives. We represent the vehicle pose as:

$$x_i = [\underbrace{t_{i,x}, t_{i,y}, t_{i,z}}_{\text{translational components}}, \underbrace{\phi_i, \theta_i, \psi_i}_{\text{yaw, pitch and roll angles}}]^T \tag{3.7}$$

Thus, we can split the vehicle odometry into two independent constraints, similar to [94]: **(i)** a 3-DoF pose-to-pose relative odometry constraint on XYH (X, Y, yaw) and **(ii)** a 3-DoF unary constraint on ZPR (Z, pitch, roll). At a given timestep i: an XYH factor u_{i-1} is added between x_{i-1} and x_i, and a ZPR factor v_i is added to vehicle pose x_i. A corrected stereo measurement factor m_k joins any observed 3-D point landmark l_j with pose x_i. This factor m_k is the corrected stereo landmark pixel observations, which is a four-vector for unrectified stereo. We attach a pose prior measurement p_0 to x_0 to bind the entire trajectory to a global coordinate frame. The state and measurement vectors are:

$$\begin{aligned} \mathcal{X} &= \{x_0, \ldots, l_0, \ldots\} \\ \mathcal{Z} &= \{p_0, u_0, \ldots, v_0, \ldots, m_0, \ldots\} \end{aligned} \tag{3.8}$$

We expand the MAP estimate defined in Section 2.1 as follows:

$$\begin{aligned} \mathcal{X}^* &= \underset{\mathcal{X}}{\text{argmax}} \; p(\mathcal{X}|\mathcal{Z}) \\ &= \underset{\mathcal{X}}{\text{argmax}} \; p(\mathcal{X})p(\mathcal{Z}|\mathcal{X}) \\ &= \underset{\mathcal{X}}{\text{argmax}} \; \underbrace{p(x_0)}_{\text{prior}} \prod_{i=1}^{n} \underbrace{p(u_i|x_{i-1}, x_i)}_{\text{XYH}} \underbrace{p(v_i|x_i)}_{\text{ZPR}} \\ &\qquad \prod_{k=1}^{m} \underbrace{p(m_k|x_i, l_j)}_{\text{corr. stereo factor}} \end{aligned} \tag{3.9}$$

We consider all four measurements as normally distributed random variables with covariances Σ_0, Ψ_i, Φ_i, Γ_k:

$$\begin{aligned} p(x_0) &= \mathcal{N}(p_0, \Sigma_0) \\ p(u_i|x_{i-1}, x_i) &= \mathcal{N}(\mathcal{U}(x_{i-1}, x_i), \Psi_i) \\ p(v_i|x_i) &= \mathcal{N}(\mathcal{V}(x_i), \Phi_i) \\ p(m_k|x_i, l_j) &= \mathcal{N}(\mathcal{M}(x_i, l_j), \Gamma_k) \end{aligned} \tag{3.10}$$

In Equation 3.10:

(i) p_0 represents the pose prior.

(ii) $\mathcal{U}(x_{i-1}, x_i)$ represents the relative transform between consecutive poses in $[t_{i,x}, t_{i,y}, \phi_i]$.

(iii) $\mathcal{V}(x_i)$ is the direct measurement of $[t_{i,z}, \theta_i, \psi_i]$.

(iv) $\mathcal{M}(x_{i_k}, l_{j_k})$ is the refraction-corrected stereo measurement function. It projects l_j into the stereo cameras at vehicle pose x_i while accounting for refraction. The output is a four-vector of stereo pixel measurements.

Assuming Gaussian noise reduces the inference to a nonlinear least squares optimization [23]:

$$\begin{aligned} \mathcal{X}^* &= \underset{\mathcal{X}}{\operatorname{argmin}} - \log \left(p(x_0) \prod_{i=1}^{n} p(u_i|x_{i-1}, x_i) p(v_i|x_i) \right. \\ &\qquad \left. \prod_{k=1}^{m} p(m_k|x_i, l_j) \right) \\ &= \underset{\mathcal{X}}{\operatorname{argmin}} \|p_0 \ominus x_0\|^2_{\Sigma_0} + \sum_{k=1}^{m} \|m_k - \mathcal{M}(x_i, l_j)\|^2_{\Gamma_k} \\ &\qquad + \sum_{i=1}^{n} \left(\|u_i - \mathcal{U}(x_{i-1}, x_i)\|^2_{\Psi_i} + \|v_i - \mathcal{V}(x_i)\|^2_{\Phi_i} \right) \end{aligned} \tag{3.11}$$

The 6-DoF pose prior is in the $SE\,(3)$ Lie group, and $\ominus$ represents the logarithm map of the relative transformation between the elements [7]. The notation of the form $\|w\|^2_\Lambda = w^T \Lambda^{-1} w$ is the Mahalanobis distance of w.

We use incremental methods to obtain optimized vehicle pose and landmark estimates at every timestep [48, 49]. Instead of re-calculating the entire system each time, it updates the previous matrix factorization with the new measurements. The sparse nature of the system (i.e. pose-landmark connectivity) assures computational efficiency.

3.4.2 Feature Extraction

Our technique uses sparse stereo feature points. Existing benchmarks for feature detectors underwater focus on repeatability in turbid environments [33], which is not required in our clear conditions. Our preliminary investigation demonstrated no discernible upside to using other feature detectors such as SIFT, SURF, or MSER versus the ORB detector. We use ORB based on the following:

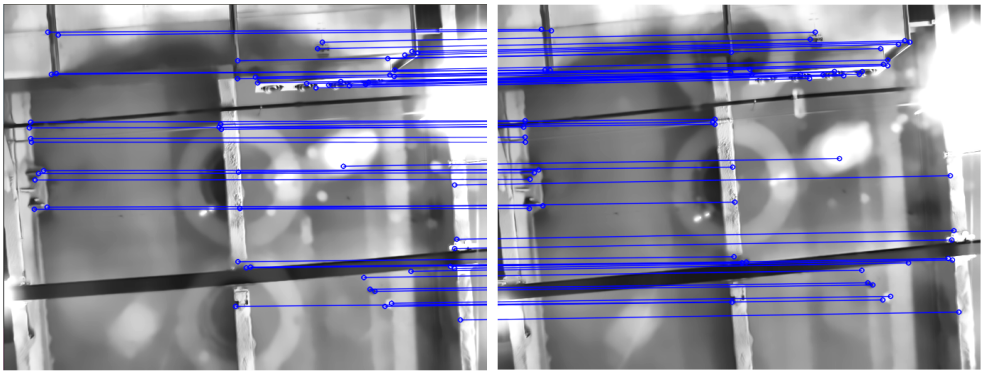


Figure 3.6: Feature matching between a stereo pair of images from our real-world dataset (Section 3.5.4). Adaptive non-maximal suppression prevents clustering of feature points and gives good spatial distribution. In these frames, the vehicle is just below the water surface and the reflection of the stereo pair at the water interface is faintly visible.

Environment

The studies [19, 33] consider Harris, Hessian, Laplacian, DoG and other detectors in underwater environments of varying turbidity. Ferrera et al. [28] later performed similar evaluations with ORB features on the TURBID dataset [19]. SIFT and SURF perform better in these scenarios as compared to ORB. However, our method considers underwater environments with little-to-no turbidity. The configurations of the experiments are also different—while the above prior work images objects present *inside* turbid water, we look directly *out* of clear water into air. We thus believe directly adapting their results would not be appropriate.

Speed

In stereo visual SLAM, we require features that can be computed and matched extremely fast. Prior literature documents feature extraction times per frame of—ORB ($\approx$ 20ms), SURF ($\approx$ 200ms) and SIFT ($\approx$5000ms) [76].

Empirical considerations

Our preliminary tests comprised of an empirical evaluation of ORB, SIFT, SURF and MSER (region-based) features. In the presence of water surface disturbances, we believe ORB features tended to be more stable. ORB features are also proven to perform better in the presence of image noise [76]. In these experiments, they were less sensitive to washout and lens flares. These artifacts are commonly seen in our datasets (Fig. 3.1), but do not occur in the TURBID dataset [19]. Region-based methods like MSER were found to yield resulting blobs that were highly unstable.

In summary–first, we valued the efficiency of ORB features for close-to-real-time implementation. Second, our preliminary investigation demonstrated no discernible upside to using blob-based feature detectors.

We detect a large number of ORB feature points and prune them through adaptive non-maximal suppression [8], selectively choosing keypoints based on corner strength and spatial location. This prevents clustering, degeneracy, and speeds up computation. We establish matches between the stereo pairs based on the Hamming distance between their binary descriptors. To remove ambiguous matches, we perform the distance-ratio test [56] and further select the inliers of a RANSAC homography computation. Fig. 3.6 shows feature matches between a stereo pair from our real-world dataset.

3.4.3 Data Association

Wrong correspondences affect the accuracy of the state estimate and landmark map. Thus, we need a reliable data association framework. Two operations—map update and landmark initialization—are explained below:

Map update

The estimated positions of landmarks in the optimization are first corrected to their *apparent* positions for the current camera poses (Section 3.3.3). They are then projected into the cameras of the stereo pair. A landmark is temporally matched with a stereo keypoint if its corresponding projection lies within an empirical gating threshold g_t in both cameras ($g_t = 5$ pixels). In the case of multiple matches, the closest projected landmark is considered.

Landmark initialization

If a stereo keypoint does not correspond to an existing landmark, it is considered for initialization as a new landmark. We only initialize landmarks when they are supported by observations from multiple viewpoints, similar to the distant stereo point triangulation in [62]. We triangulate a stereo keypoint upon first viewing it, but do not add it to the optimization yet. If a stereo keypoint lies within g_t of the projected landmark for the N consecutive frames, we add this landmark to the global map. We initialize it by triangulating over all the N views. It is only then that the landmark and its corresponding measurements are added to the optimization. The value of $N = 5$ is empirically selected, but this is often reduced in difficult visibility conditions.

3.4.4 Implementation

Our framework uses the Georgia Tech smoothing and mapping (GTSAM) library [22] for factor-graph optimization. We use iSAM2 [49] for an efficient incremental solution using the Powell's dog-leg optimization algorithm. The experiments (Section 3.5.3 and 3.5.4) are run offline on an Intel Core i7-7820HQ CPU @ 2.90GHz and 32GB RAM without GPU parallelization.

3.5 Experimental Results

3.5.1 Trajectory Metrics

Given the estimated, noisy and ground truth trajectories of the vehicle, we wish to evaluate the quality of the SLAM solution. In simulation, we also compute the mean and median absolute landmark error (ALE) of the final landmark map. We follow the relative pose error (RPE) and absolute trajectory error (ATE) metrics [85]. Consider an estimated vehicle trajectory $P_1 \dots P_n \in \mathbf{SE}(3)$ and ground truth trajectory $Q_1 \dots Q_n \in \mathbf{SE}(3)$. Both sequences are assumed to be time-synced and have the same number of readings. In general, both the metrics are correlated and we expand them below:

Relative Pose Error

This is a measure of how locally accurate a trajectory is over a time interval t. The RPE thus sufficiently captures drift of vehicle pose.

$$E_i := \left(Q_i^{-1} \cdot Q_{i+t}\right)^{-1} \left(P_i^{-1} \cdot P_{i+t}\right) \tag{3.12}$$

From these errors, we compute the root mean square error (RMSE) of only the translational component. This is generally sufficient as rotational errors manifest themselves as translational errors. We empirically select $t = 1$ sec for the RPE evaluation.

Absolute Trajectory Error

This metric quantifies the global consistency of the estimated trajectory. At a high-level, this is a comparison of the estimated and ground truth trajectory. Before we do so, we use Horn's algorithm [38] to compute a transformation that aligns the trajectories. The error at timestep i is:

$$F_i := Q_i^{-1} S P_i \tag{3.13}$$

Similar to the RPE case, we compute the root mean square error (RMSE) of the translational components of F_i.

Table 3.1: Covariance matrices (defined in Section 3.4.1) used in simulation and real-world experiments. They are diagonal square matrices of the form $\mathrm{diag}({M_0}^2, {M_1}^2, \dots)$. The units for translation, rotation and image measurements are meters, radians and pixels respectively.

Covariances	**Type**	**Square roots of matrix diagonal elements** (M)
$\mathbf{\Sigma_0}$	6-DoF pose prior	10^{-4} m, 10^{-4} m, 10^{-4} m, 10^{-4} m, 10^{-4} m, 10^{-4} m
$\mathbf{\Psi_i}$	3-DoF XYH odometry	0.01 m, 0.01 m, 0.01 rad
$\mathbf{\Phi_i}$	3-DoF ZPR measurement	0.01 m, 0.005 rad, 0.005 rad
$\mathbf{\Gamma_k}$	Corrected stereo factor	1 pix, 1 pix, 1 pix, 1 pix

3.5.2 Noise and Covariance

Noise is added in the XYH directions at every frame, with standard deviations $\sigma_x = \sigma_y = 0.01$ m and $\sigma_\phi = 0.01$ rad. In Section 3.5.3 and 3.5.4, we compare this synthesized dead reckoning with our SLAM solution. The covariance values for all the factors are shown in Table 3.1.

3.5.3 Simulated Experiments

For preliminary analysis, we run simulations with generated vehicle motions and assume known data association. We randomly initialize landmarks in space above the water surface, spread across the XY plane and between 4–5 m in the Z direction. We add Gaussian noise ($\sigma = 1$ pixel) to stereo landmark measurements. When projecting ground truth landmarks, we apply our refractive model to simulate looking through the water surface. Two scenarios are analyzed: a *square* and *corkscrew* trajectory. While *square* does not include motion in the Z direction or yaw rotation, *corkscrew* exercises all these degrees of freedom. To emulate the HAUV, we constantly vary the pitch and roll over the $\pm 5^\circ$ range. Each dataset has 1200 poses, executing 7 loops of radius 2.5m in *corkscrew* and 10 loops of side length 3m in *square*.

In Table 3.2, we quantitatively compare the dead reckoning and SLAM estimate trajectories against ground truth. It can be seen that we achieve substantial reduction in ATE and RPE with our framework for both trajectories. While the mean ALE is higher for `corkscrew`, the median verifies that it is due to outliers. Fig. 3.7 qualitatively compares both trajectories and estimated landmarks. The dead reckoning trajectory drifts significantly over time, while our solution roughly overlaps with the ground truth. In Table 3.4, we further compare these results with a modified implementation that does not account for refraction.

3.5.4 Real-world Experiments

Our SLAM framework is evaluated using the HAUV in an indoor test-tank. The tank has a depth of 3m and radius of 3.5m. Regions of the ceiling are not at the same height from the water surface due to piping, air ducts and girders. On measurement with survey equipment, they are found to be between 3.6–5.8m. Fig. 3.8 shows the ceiling and tank setup.

We log 12 datasets for evaluation that encompass a wide range of scenarios the vehicle may encounter. They vary between 100–686 seconds in length and all but one execute pre-programmed

Table 3.2: Mean absolute trajectory error (ATE) and relative pose error (RPE) for the two simulation trajectories. Mean and median absolute landmark error (ALE) are also shown. We see a significant decrease in error in the SLAM solution as compared to the dead reckoning trajectory.

Dataset	Dead reckoning			SLAM solution				
	ATE (m)	$\text{RPE}_{\text{trans}}$ (m)	RPE_{rot} (°)	ATE (m)	$\text{RPE}_{\text{trans}}$ (m)	RPE_{rot} (°)	ALE_{mean} (m)	$\text{ALE}_{\text{median}}$ (m)
`square`	0.458	0.661	16.444	0.012	0.018	0.130	0.015	0.008
`corkscrew`	0.415	0.593	10.931	0.011	0.017	0.112	0.107	0.005

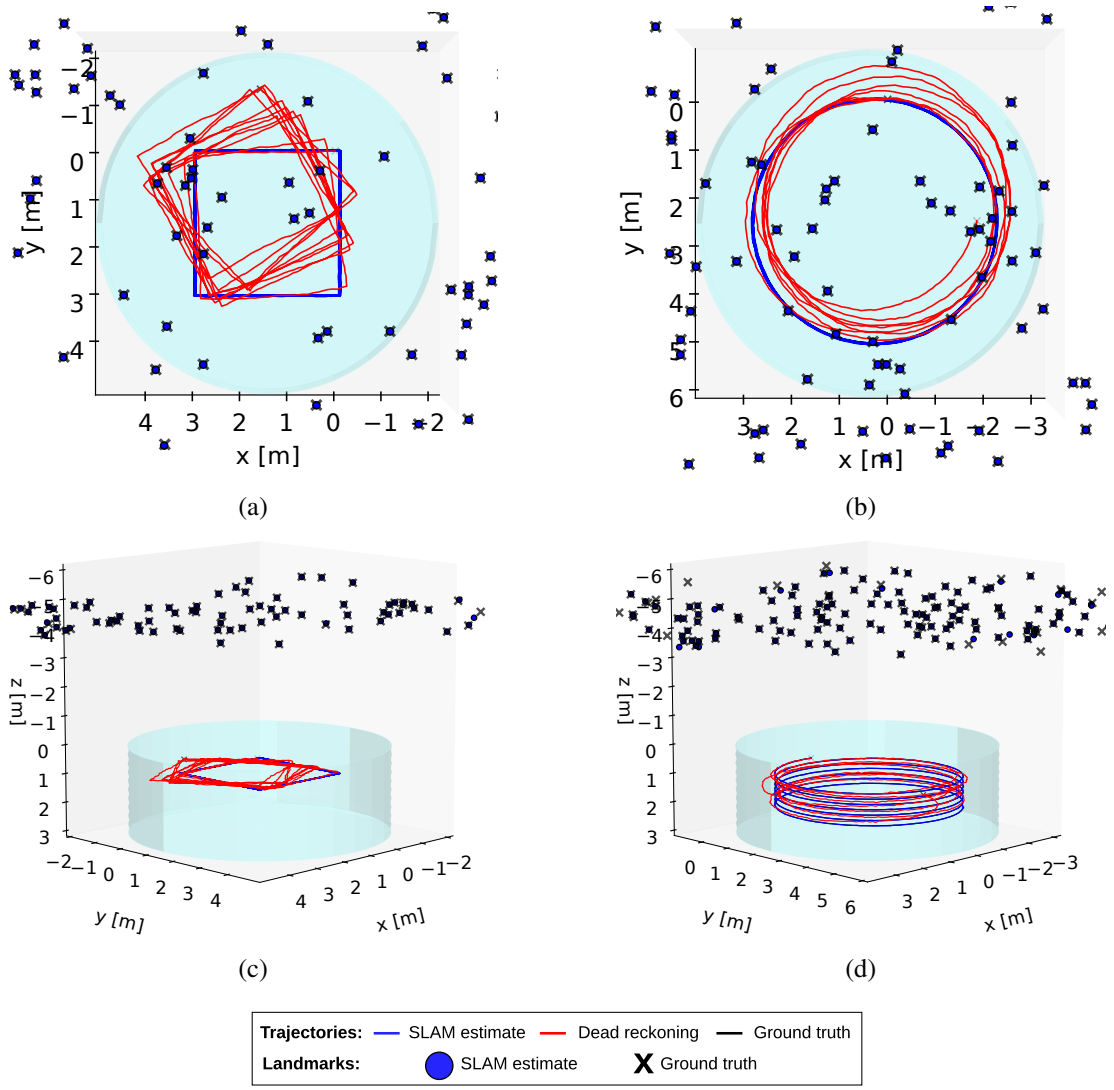


Figure 3.7: Visualization of the SLAM trajectory and landmark estimates from simulation, overlaid with the tank environment. **(a)** and **(b)** show top-views while **(c)** and **(d)** are from the side. The SLAM solution coincides (and thus obscures) the ground truth, while the dead reckoning drifts. The estimated landmarks converge to near their ground truth positions.

loops in the tank. The vehicle translates in the X and Y directions at a fixed depth, along with rotation about the Z-axis (yaw rotation). The pitch and roll directions of the vehicle cannot be controlled, but fluctuate mildly underwater nevertheless. Upon receiving a valid pair of stereo frames, we use its timestamp to interpolate a state estimate. Challenges that can degrade the SLAM solution include water surface disturbances, motion blur, suspended particulates, light scattering and image washout (Fig. 3.1). The value of N (refer Section 3.4.3) is reduced to 2 in datasets with larger disturbances. The datasets incorporate all these conditions *(brackets denotes number of such datasets)*:

Depth: Just below surface *(4)*, 1m *(4)* and 2m depth *(4)*.

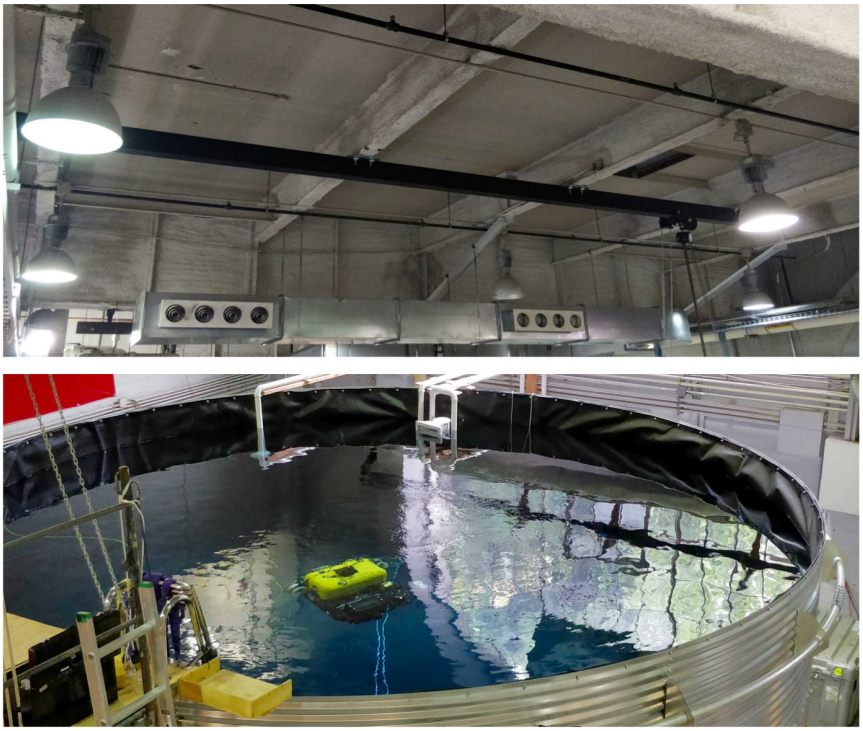


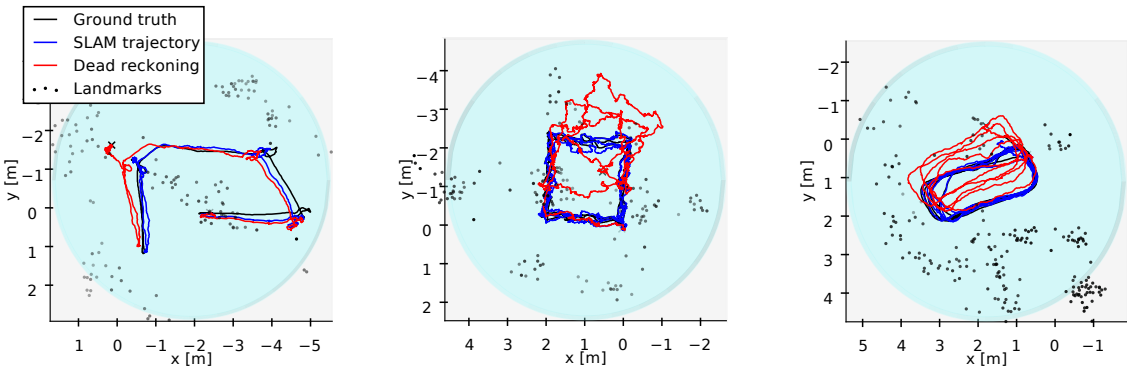
Figure 3.8: *(top)* Ceiling present over the tank. Objects in the vehicle's field-of-view are between 3.6–5.8m in height from the water surface. *(bottom)* Tank setup with vehicle executing a trajectory at 1m depth.

Visibility: With *(8)* and without *(4)* suspended particulates.
Lighting: With *(3)* and without *(9)* ceiling lights.

Table 3.3 lists the evaluation metrics for the dead reckoning and SLAM solution for all 12 datasets. We choose one representative dataset from each depth level—datasets **03**, **08** and **09**—and plot the trajectory estimates (Fig. 3.9). Our proposed method significantly reduces drift in all cases, as seen in the ATE and RPE metrics. This is most apparent in the longer datasets, **08** (Fig. 3.9 (b)) and 10.

We also compare the results from our real-world and simulation dataset with a modified implementation that does not account for refraction (refer Table 3.4). The results show reduced error when we account for refraction (RC), which reinforces our method. We also see a significant difference between the final landmark maps of both cases. Fig. 3.10 visualizes this result for dataset **08**.

The solve time for each dataset (Table 3.3) depends on how densely connected the underlying factor graph is. Most of the execution time is devoted to the optimization; the refraction module requires only a smaller proportion of the compute time. For example, the entire optimization for dataset **08** (724 seconds dataset duration, 144 landmarks) takes 884.3 seconds to solve with the refraction module, and 789.4 seconds without. We can achieve real-time performance through keyframing or fixed-lag smoothing.



(a) Dataset **03** (0m depth) (b) Dataset **08** (1m depth) (c) Dataset **09** (2m depth)

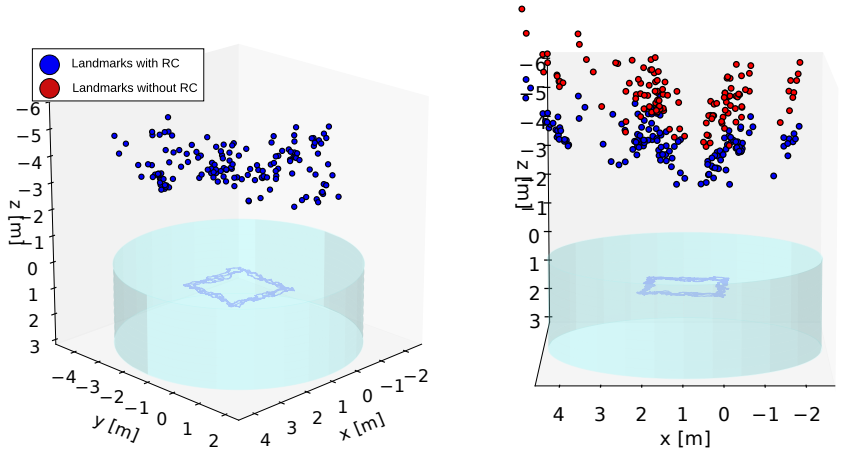
Figure 3.9: Qualitative comparison of trajectories from the representative datasets. We observe strong correspondence between our SLAM trajectory and the ground truth, while the dead reckoning trajectory drifts over time. The global coordinates (in the X and Y) vary between trajectories as the origin is defined by the vehicle start position prior to recording.

Table 3.3: Mean ATE and RPE for the 12 underwater datasets. Details about each dataset—operation depth, runtime duration and solve time—are shown. 0m indicates a depth *just* below the water surface. Datasets in **bold** are the representative datasets, which further appear in Fig. 3.9 and Table 3.4.

Dataset				**Dead reckoning**			**SLAM solution**		
#	depth (m)	duration (s)	solve (s)	ATE (m)	RPE_t (m)	RPE_r (°)	ATE (m)	RPE_t (m)	RPE_r (°)
01	1	133.8	476.3	0.069	0.112	4.572	0.053	0.072	2.198
02	0	99.6	188.7	0.122	0.149	3.301	0.067	0.079	1.480
03	0	202.2	505.5	0.280	0.370	5.976	0.115	0.090	2.539
04	2	121.8	63.0	0.103	0.145	4.121	0.058	0.125	2.822
05	1	192.6	23.4	0.076	0.112	2.600	0.046	0.062	1.273
06	2	203	13.5	0.095	0.137	2.328	0.051	0.068	1.520
07	1	238.8	329.9	0.181	0.248	5.839	0.074	0.096	2.886
08	1	724.0	884.3	0.568	0.818	21.451	0.073	0.098	2.267
09	2	260.0	449.2	0.265	0.343	5.696	0.086	0.105	2.216
10	2	686.2	1409.0	0.327	0.402	20.583	0.068	0.082	1.365
11	0	446.8	2088.0	0.259	0.329	9.050	0.037	0.051	1.175
12	2	200.0	91.4	0.096	0.160	2.972	0.050	0.065	1.164

Table 3.4: ATE of real-world *(left)* and simulation datasets *(right)* with/without refraction correction (RC). It reduces when RC is present in the framework.

Dataset	01	02	**03**	04	05	06	07	**08**	**09**	10	11	12	*mean*	`sq`	`cork`
ATE_{RC}	0.053	0.067	0.115	0.058	0.046	0.051	0.074	0.073	0.086	0.068	0.037	0.050	*0.065*	0.012	0.011
$ATE_{no\ RC}$	0.057	0.067	0.155	0.060	0.047	0.053	0.075	0.074	0.102	0.071	0.041	0.058	*0.072*	0.015	0.014



(a) (b)

Figure 3.10: (a) Final landmark map of dataset **08**. (b) The landmark map with refraction correction is compared with that without refraction correction.

Chapter 4

Mapping and Exploration: Active Submap SLAM

4.1 Introduction

The SLAM problem is generally trajectory agnostic—it assumes that the chosen path is sufficient for mapping and localization. Within the context of our environment, this raises two problems. First, precise teleoperation is difficult in the cluttered environments, such as those pictured in Figure 4.1. Second, the performance of the SLAM solution is dependent on the choice of trajectory—they must be jointly considered. This is considered as *active SLAM*, which dates back to seminal work on active perception by Bajcsy et al. [9].

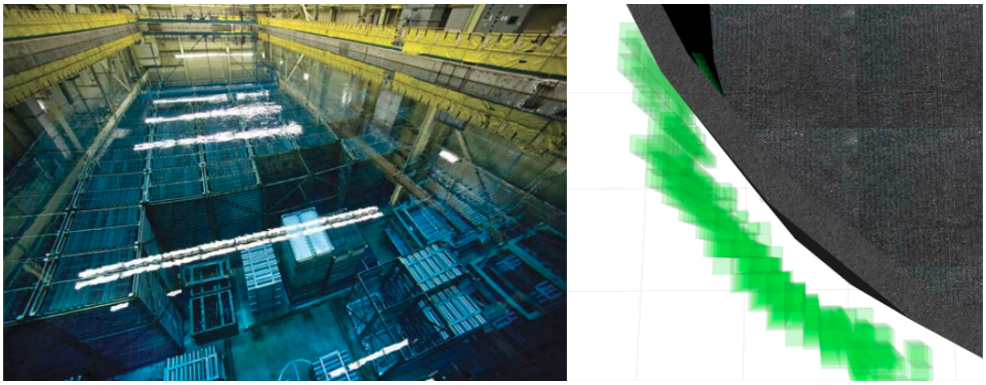


Figure 4.1: *(left)* Example of cluttered SNF pool that is difficult to teleoperate in [5]. *(right)* Drifting state estimate creates erroneous occupancy representation, which can lead to ill-advised trajectories.

Active SLAM in marine environments was demonstrated by Fairfield et al. for re-localizing an AUV [25]. Here, a deep-operating vehicle constructs bathymetric maps with sonar and utilizes it to re-localize its gradually drifting pose. The authors later used a submap representation and selected entropy-reducing actions for active loop closing [26]. Kim et al. used visual saliency for loop-closing camera registrations in ship-hull inspection runs [51]. This method attempts to balance the trade-off between area coverage and revisit actions for a preplanned trajectory. Chaves

et al. extended this work for coverage-efficient revisit policies [17].

This chapter instead presents an active SLAM framework for mapping *indoor* underwater environments. Here, the vehicle must perform volumetric exploration while maintaining bounded pose uncertainty. While it is not handled in conjunction to the localization strategy (Chapter 3), their combination is considered future work (Section 5). Below the water level, the visual measurements are sparse and not uniformly distributed. Instead, shape information from sonar can be used for pose-to-pose constraints.

We look at navigational decision-making—can we devise a balanced exploration strategy that uses sonar information to identify candidate revisit poses? Additionally, how do we decide what are good poses to revisit? Executing revisit actions bounds pose uncertainty through loop closures, but comes at a cost. The revisit policy results in extra distance travelled and redundant volumetric coverage. This necessitates a heuristic for the exploration policy, based on pose uncertainty and path information gain.

This work builds upon VOG-Map [37], which combines pose-graph submap SLAM with sampling-based planning for underwater environments. VOG-Map prevents vehicle drift via iterative closest point (ICP) submap loop closures, and its exploration policy is based on maximizing information gain. The vehicle has a notion of free and occupied space for planning, which can be deformed based on loop closures. It also creates a dense global map from collating local sonar submap point clouds. The system is explained in detail in Section 4.2.3, and its components are shown in Fig. 4.2.

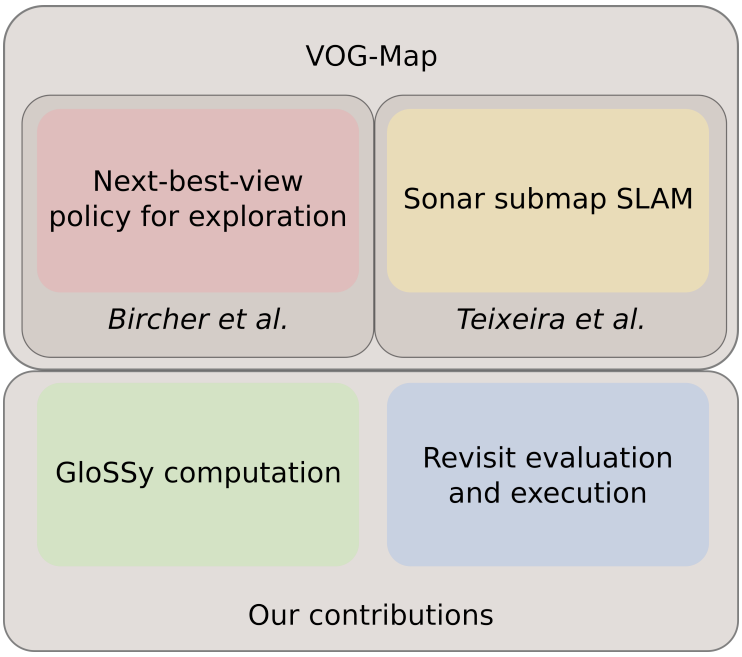


Figure 4.2: The block diagram of the active exploration method, with its different components. We build on the mapping framework by Ho et al. [37], and we add the capabilities for revisit to it.

4.2 Background and Related Work

4.2.1 Unknown Space Exploration

To plan an exploration trajectory, the robot must have a notion of free-space in the environment. An occupancy grid map is a powerful volumetric space representation amenable to motion planning. It reduces the map to uniformly spaced field of binary random variables, each indicating free, occupied or unknown space at the given location. The OctoMap uses an octree data structure, resulting in a tractable representation for robotics problems [39].

The rapidly exploring random tree (RRT) planner is a sampling-based algorithm for space exploration [53]. It grows a tree in configuration space, and in every iteration a newly sampled node is attached to the tree. The sampled node is connected to an existing node in the tree that is closest to it. However prior to connection: **(i)** the node is scaled in configuration space to be within a maximum extension range of the existing tree **(ii)** the new edge is verified to be collision-free via ray-tracing. The RRT extend operation is shown in Fig. 4.3.

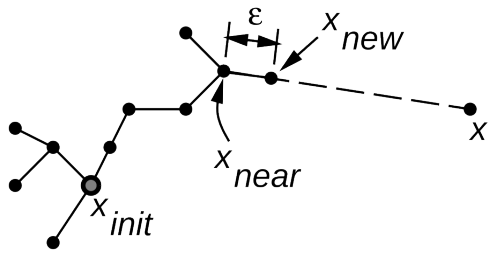


Figure 4.3: The RRT extend operation, as taken from [53].

Bircher et al. proposed combining the RRT planner with the OctoMap representation [11]. The planner follows a receding horizon *next-best-view* algorithm. It first grows an RRT tree, and evaluates the branch that gives the best coverage of unmapped space. The exploration step executes only the first edge of branch, and this heuristic is recomputed. This biases robot exploration towards unknown volumes in the environment.

4.2.2 SONAR Submaps

A submap SLAM framework considers the global map as a collation of local submaps described in their own coordinate frame. This method has found wide application in field robotics [27, 78, 95]. Each submap is associated with a base pose node, and we connect these to form the pose graph (Fig. 4.6). The base pose is the reference pose with respect to which the local sonar submap is constructed, as used by Teixeira et al. [86]. This fragments the map into manageable pieces, and operations on these fixed-size submaps are computationally feasible. A resulting global map can be seen in Fig. 4.4. How do we decide on the time period and size of the submap? Their size must be large enough for successful loop closure detection, and small enough that there is limited local odometry drift.

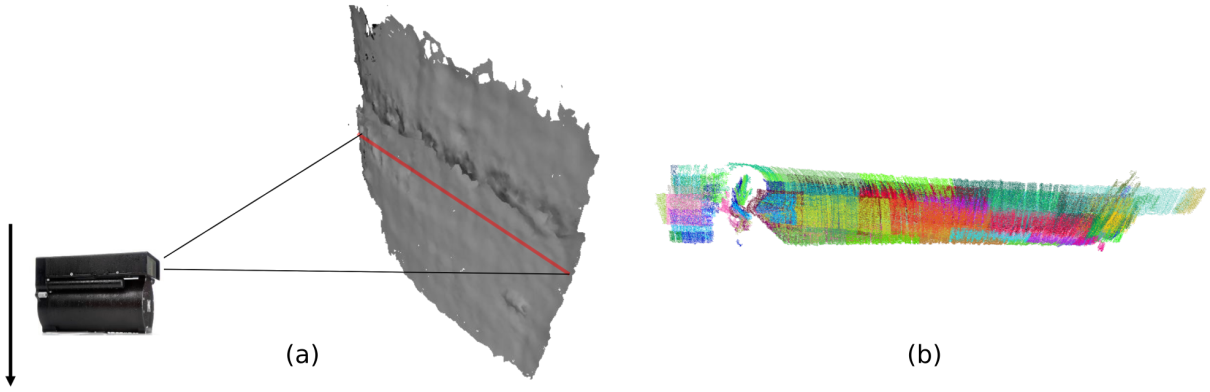


Figure 4.4: **(a)** Vehicle odometry creates a sonar sweep, image sourced from Kaess et al. [46]. **(b)** Teixeira et al. formulated a SLAM framework that performs ICP for submaps for pose-to-pose constraints.

4.2.3 The Virtual Occupancy Grid Map

For completeness, we briefly describe VOG-Map and direct the reader to the original manuscript for further details [37]. In short, VOG-Map incorporates a pose-graph SLAM framework while maintaining free-space information for motion planning and exploration. The components of the system are shown in Fig. 4.5.

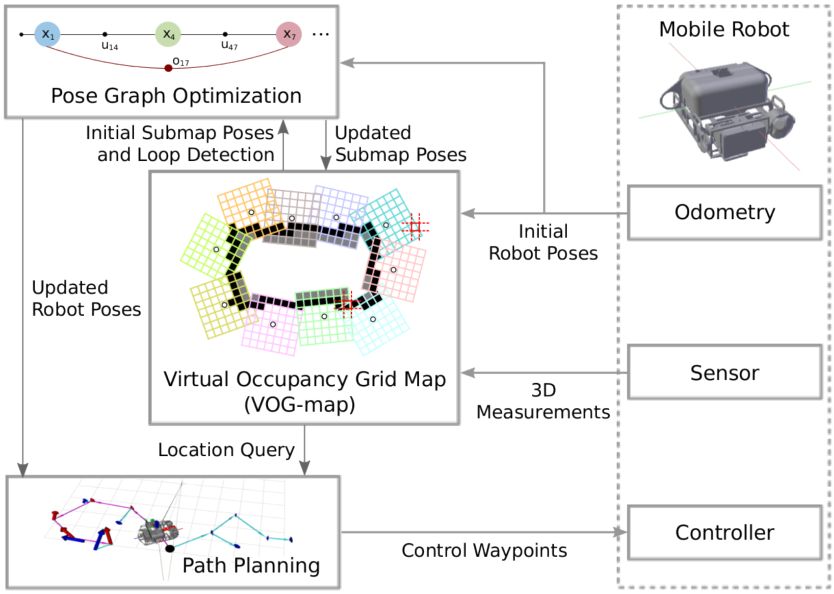


Figure 4.5: The system description of the VOG-Map framework by Ho et al. [37] for autonomous underwater exploration with the HAUV platform.

Free-space Representation

VOG-Map differs from the global occupancy grid maps [11, 70, 91] in that this representation can be deformed to correct accumulated drift. The base poses in the VOG-Map pose graph correspond

to the local occupancy grid maps (Fig. 4.6).

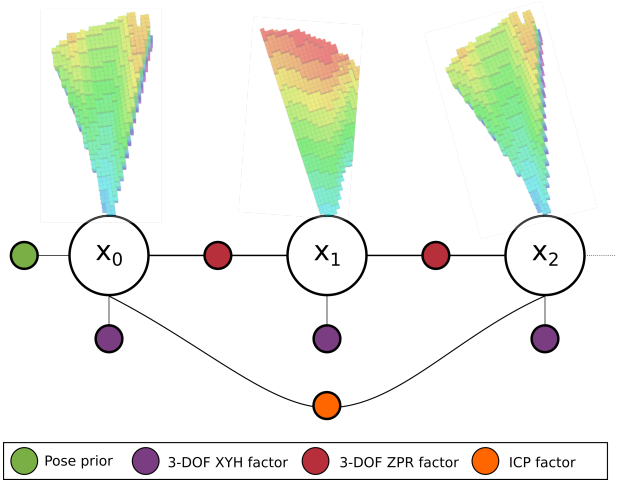


Figure 4.6: The pose graph as formulated in [Teixeira et al., 2016] upon which VOG-Map is built. When the optimization updates the pose estimates of the nodes, base poses of the local occupancy grid maps are also updated using VOG-Map deformation operation. This corrects the VOG-Map for drift or accumulated noise.

The local occupancy grid map is created by collating the sonar scans over a period of time. They are with respect to the base pose, which in our case is the pose of the first scan in the submap. The sonar scans that comprise the submap are integrated through raycasting operations and the occupancy probabilities of a voxel v is updated in accordance to [39, 61]. They use a log-odds update rule and assume a uniform prior probability $P(v) = 0.5$.

ICP Loop Closure

The loop closure constraint finds the relative pose transformation between submaps using the ICP algorithm [86]. The corresponding point clouds associated with each local occupancy grid is cached by the system. As described in Section 2.2 and 3.4.1, the pose estimation is restricted to 3DoF. 2D Point-to-Point ICP is carried out between the submap clouds to compute the X, Y translation and yaw that aligns them. The algorithm is described in detail by Sorkine-Hornung et al. [84]. VOG-Map runs through all the submaps while computing loop closure factors—the complexity rises linearly with number of submaps.

Standard Operations

There are two operations executed in the system—deformation and occupancy querying:

Deformation: Upon adding a loop closure to the graph, a batch optimization is carried out and the base poses are updated. This results in a map deformation, by modifying the arrangement of the local occupancy grid maps. The complexity of this operation rises linearly with the number

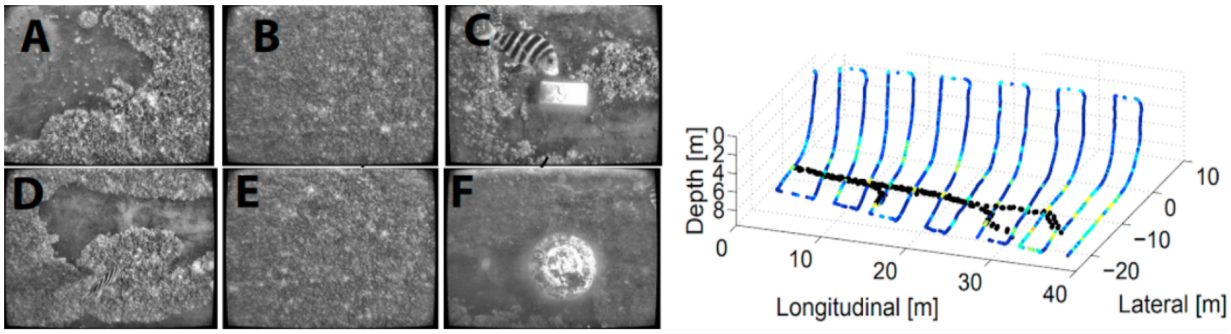
of submaps accumulated.

Occupancy Querying: This operation takes in an $\mathbb{R}^3$ point in space and outputs the occupancy probability with status (*free*, *occupied*, or *unknown*). This is done by querying all the local occupancy grid maps and combining the log-odds terms.

However, the method assumes perfect state estimate, so drift can result in poor quality maps such as Fig. 4.1. We require a method that enables the robot to plan, explore and map an apriori unknown environment while considering pose uncertainty.

4.2.4 Saliency for Active SLAM

The idea of saliency comes from the seminal work on the human perception model by Itti and Koch [41]. They highlight the presence of visually salient regions in an image that command the attention of the viewer. In computer vision, this has been explored in the context of a bag-of-words (BoW) representation [68]. The BoW was first used for textual data, and image features descriptors were found to be analogous to words. This was later used in appearance-based visual SLAM—FAB-MAP learnt commonly-occurring visual words offline for online loop-closure detection [20]. The *term frequency-inverse document frequency* (tf-idf) statistic captures the rarity of words in a dataset, and has been used for text classification [67, 82].



(a) (b)

Figure 4.7: (a) Examples of visual candidates for loop-closure camera registrations in work by Kim et al. [51] (b) Good revisit candidates are shown in brighter colors.

Local feature descriptors and keypoint detectors have also been introduced for 3-D point cloud data. They have found application in 3-D object recognition, classification, shape analysis, and model retrieval. A survey and comparison of the different feature descriptors can be found in [36]. Rendondo et al. [75] quantized 3-D SURF descriptors to generate 3-D visual words. This 3-D BoW representation is then used to recognize point cloud object categories. In this thesis, we use a BoW representation for 3-D feature descriptors that we learn offline. This is used in conjunction with a tf-idf global saliency metric to quantify the uniqueness of a submap base pose.

Kim et al. established that there is a strong correlation between visual saliency of a scene and the probability of making a successful visual loop closure. Some examples of salient scenes are shown in Fig. 4.7.a, and these revisit poses appear as brighter shades in Fig. 4.7.b. They introduce a global saliency score for keyframes by measuring the idf of visual word occurrences in candidate images. The score represents inter-image rarity, and these poses are viable revisit candidates in the active SLAM framework. In the coming sections, we explore an analogous idea for sonar submaps.

4.3 Submap Saliency

4.3.1 Vocabulary Generation

Figure 4.8: Building a 3-D scene dictionary offline from a large collection of sonar submaps from an underwater tank environment.

We illustrate the process of building a dictionary of 3-D visual words in Fig. 4.8. Given a submap point cloud, we first subsample *keypoints* from it. 3-D keypoints are unique elements identified in a points cloud that can be used in place of the entire cloud. We use the Harris3D keypoint detector, which ports the classical Harris operator to 3-D data [81]. Following this, we extract a set of descriptors using the signature of histograms of orientations (SHOT) [88]. SHOT divides the local support into a spherical grid. Within a local radius, $\cos\theta_i = n_u \cdot n_{v_i}$ is computed, where n_u is the keypoint normal, and n_{v_i} is the normal of a point in the radial neighbourhood. Each angle i is binned into the section of the sphere that corresponds to $\cos\theta_i$, which results in a 32-dimension SHOT descriptor. This histogram is L-1 normalized for robustness to point density. This representation is invariant to rotation, translation, and noise. The SHOT descriptors are clustered into $N = 50$ 3-D visual words that make up our BoW submap dictionary.

To build this dictionary, we use 3 datasets collected from teleoperating the vehicle in our underwater tank. We place a rectangular piling in the center and accumulate sonar submaps of the environment. Fig. 4.9 shows camera stills from operating the vehicle while Fig. 4.8 shows a global map from one dataset.

4.3.2 The GloSSy Metric

Global saliency was introduced for images by Kim et al. [51] as a measure of feature uniqueness across images. It was applied to locate revisit locations in a trajectory that maps a ship hull. In place of 128-dimensional SURF descriptors, we use SHOT descriptors to compute a global submap saliency (GloSSy) metric. The tf-idf [67, 82] measures the inverse-document frequency

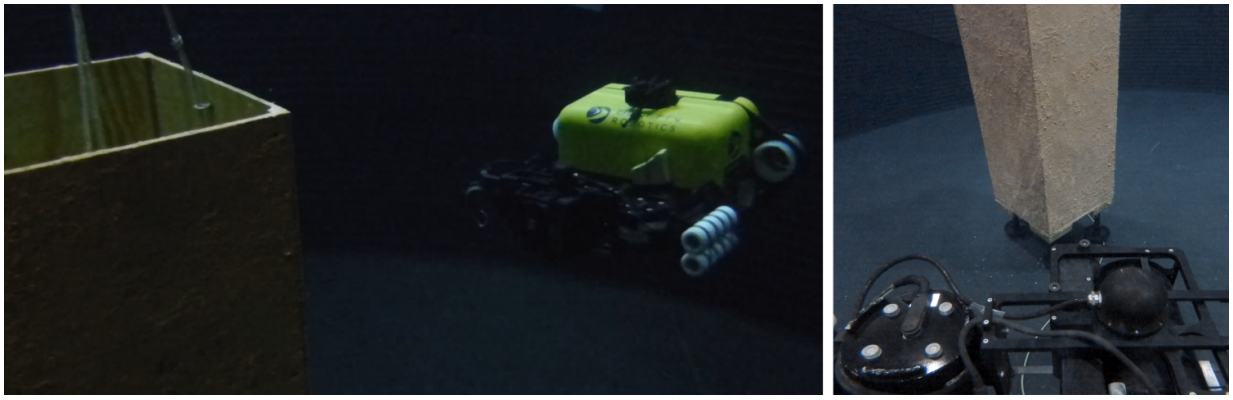


Figure 4.9: Camera stills from our vehicle recording sonar submaps for training our BoW dictionary. *(left)* view from underwater camera *(right)* view from GoPro mounted on vehicle. We image a central rectangular piling and the tank walls, the resulting global map can be seen in Fig. 4.8.

of a word, or how often it is encountered in the environment. Given word w, the tf-idf is computed as:

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \tag{4.1}$$

where n_{id} is the number of occurrences of word i in document d, n_d is the total number of words in document d, N is the total number of documents, and n_i is the number of occurrences of word i across all documents. Intuitively, the score given to a word is low if we encounter it often in the entire database. A simpler metric often used is simply the idf, the latter half of Equation 4.1. Similar to [51] we define an inter-submap rarity term as a summation of idf:

$$\mathcal{G}_s(t) = \sum_{j \in \mathcal{W}_s} \log_2 \frac{N(t)}{n_{w_j}(t)} \tag{4.2}$$

where $\mathcal{W}_s \subset \mathcal{W}(t)$ is the subset of words found in submap s, $n_{w_j}(t)$ is the number of submaps encountered that contain word w_j, and $N(t)$ is the current total number of submaps accumulated. This metric must be recomputed for all $N(t)$ submaps every time a new submap is received. While [51] uses an inverted index update scheme, it is not required in our case due to the small number of submaps. Thus, the idf update has linear complexity with $N(t)$. The summed idf is normalized to a $[0, 1]$ score, with $\mathcal{G}_{\text{max}}$ being the maximum summed idf encountered:

$$S_{\mathcal{G}_s}(t) = \frac{\mathcal{G}_s(t)}{\mathcal{G}_{\text{max}}} \tag{4.3}$$

4.3.3 Revisit Candidates

Fig. 4.10 shows the top/bottom revisit poses from a real-world dataset. In the dataset, the vehicle is imaging the rectangular piling, and the resulting global point cloud is shown in gray. The revisit poses are the base poses associated with the respective submaps. Qualitatively, we observe the best revisit poses are those looking head-on at the piling's edge. In an ICP loop closure framework these submaps can be easily aligned and avoid degeneracy. The bottom set of poses comprise

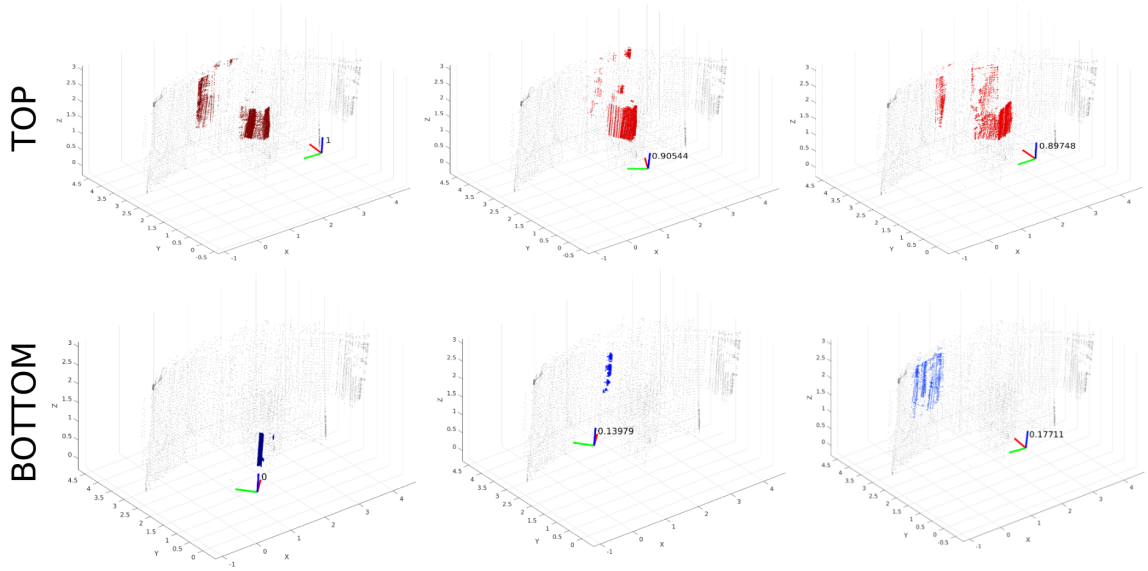


Figure 4.10: Top/bottom 3 revisit poses according to GloSSy scores in a real-world dataset. The gray point cloud represents the complete global map, and the colored sections show the submap that belongs to the revisit pose. The 6DoF pose of the vehicle is visualized, along with the GloSSy score in small print.

of the vehicle imaging the tank wall's curvature, or the flat side of the piling. These submaps are not as distinct, and can lead to degeneracy or incorrect loop-closures. Therefore, we note an empirical correlation between the GloSSy metric and *good poses to revisit*. In our system, we sort the revisit poses by their $S_{\mathcal{G}_s}(t)$ values and consider the top $N = 4$ candidates in our active SLAM formulation.

4.4 Active SLAM

4.4.1 Exploration Policy

The vehicle exploration is defined by the motion policy $\pi(t)$ being executed. It is formulated as a dual-behavior policy:

$$\pi(t) = \begin{cases} \pi_{\text{nbv}}, & \text{if } \mathcal{U}_{\mathbf{r}}(t) \leq 1 \\ \pi_{k^*}, & \text{otherwise} \end{cases} \tag{4.4}$$

where π_{nbv} is the next-best-view exploration behavior existing in [37], π_{k^*} is the revisit behavior we introduce in this work (Section 4.4.4), and $\mathcal{U}_{\text{robot}}(t)$ is the uncertainty ratio (Section 4.4.2). Intuitively, this strategy performs information-theoretic exploration when the robot uncertainty is low, and toggles to revisit previous poses upon exceeding the threshold. We choose from the revisit candidates provided by the saliency thread (Section 4.3). We employ a heuristic penalty function that considers **(i)** the propagated pose covariance at the revisit pose **(ii)** the view-utility gain from executing the revisit policy. The cached RRT path is reused—along with a shortcutting

operation—removing the need for motion planning. Algorithm 2 summarizes the policy selection, and the steps are explained in the following sections.

Algorithm 2 Exploration policy thread. Submapping, voxel updates, saliency computation and pose graph optimization happens in parallel. $\mathcal{U}_r(t)$ is the *D-opt* uncertainty ratio of the robot at the current timestep t. Policy π_{k^*} guides the robot to the selected revisit pose and policy π_{nbv} executes the VOG-Map *next-best-view* strategy.

Require:
VOG-Map representation of the current world state $\mathcal{M}_{\text{vog}}(t)$
Ensure:
Trajectory policy $\pi(t)$ for planner
if $\mathcal{U}_r(t) > 1$ **then**
  Select top N salient revisit poses.
  for $k \leftarrow 1$ to N **do**
    Get revisit policy π_k with revisit trajectory P_r^k ▷ ***Policy*** $\pi(t) \leftarrow \pi_{k^*}$.
    Propagate virtual odometry to obtained Σ_{π_k} and compute $\mathcal{U}_{\pi_k}$.
    Compute view-utility gain $\text{Gain}(\pi_k)$ given map $\mathcal{M}_{\text{vog}}(t)$.
    Compute revisitation penalty $\mathcal{P}_{\pi_{\mathbf{k}}}$
  Find $\mathbf{k}^* = \arg\max R_r^k$
  $\pi(t) \leftarrow \pi_{k^*}$
else
  Compute exploration policy π_{nbv} with trajectory P_r^{nbv} ▷ ***Policy*** $\pi(t) \leftarrow \pi_{nbv}$
  $\pi(t) \leftarrow \pi_{nbv}$
Execute policy $\pi(t)$

4.4.2 Uncertainty Criteria

Carillo et al. [14] compares the choice of uncertainty criteria for the active SLAM problem. According to the Theory of Optimal Experimental Design (TOED) [72, 73] one can compare two policy classes π_1 and π_2 if:

$$\text{Cov}(\pi_1) - \text{Cov}(\pi_2) \in \text{NND}(l) \tag{4.5}$$

Where $\text{Cov}(\pi_i)$ represents the resulting $l \times l$ covariance matrix from carrying out the policy π_i. This criteria dictates that the covariance difference must belong to the group of positive semi-definite matrices, $\text{NND}(l)$. While this is helpful in telling us if π_1 is better than π_2, it fails to quantify the difference between them. For this, we require a mapping between $\text{NND}(l)$ and a scalar quantity: $\text{NND}(l) \rightarrow \mathcal{R}$.

This mapping must satisfy certain properties, as highlighted in [14]—**(i)** positive homogenous **(ii)** isotonic, and **(iii)** concave. Further, it must faithfully capture the extent of uncertainty of $\text{Cov}(\pi_i)$. TOED prescribes certain functions that satisfy these requirements—A-optimality

criterion (*A-opt*), D-optimality criterion (*D-opt*) and E-optimality criterion (*E-opt*) [72, 73].

A-opt describes the mean uncertainty of the covariance matrix, and has been successfully used in active SLAM [52, 55]. Kiefer et al. [50] provides evidence that only *D-opt* is proportional to the uncertainty ellipse of the state parameters. Carillo et al. [14] establishes through experimental comparison that *D-opt* gives meaning information in an active SLAM context, and has certain desired properties. Thus, we use the aforementioned *D-opt* for our purposes.

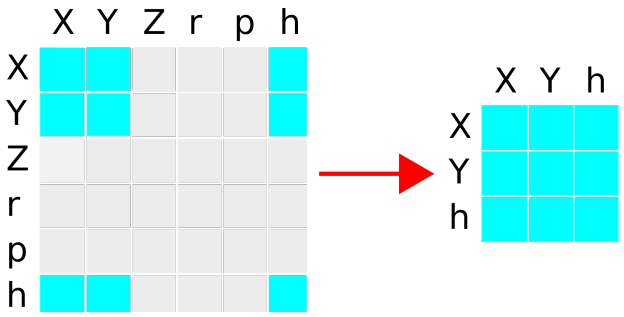


Figure 4.11: The marginal pose covariance for a 6-DoF robot, with our condensed form on the right. This 3×3 matrix encodes the required information for the *D-opt* criterion.

Kiefer et al. [50] shows that the normalized *D-opt* criterion is:

$$\phi(\pi_i) = \det\left(\Sigma_{\pi_i}\right)^{1/l} \tag{4.6}$$

We consider the marginal covariance of the robot pose, which would make our covariance matrix 6×6. We use an efficient method described by Kaess et al. to recover a part of the full covariance matrix [47]. However, as described in Section 2.2, only the $[X, Y, \text{yaw}]$ quantities drift. We can therefore consider only the 3×3 covariance matrix for the *D-opt* criterion, as shown in Figure 4.11. The ratio of the scalar with the maximum allowable value is used, which we call the *uncertainty ratio*:

$$\mathcal{U}_{\pi_i}(t) = \frac{\det\left(\Sigma_{\pi_i}(t)\right)^{1/3}}{\det\left(\Sigma_{\text{allow}}\right)^{1/3}} \qquad \mathcal{U}_r(t) = \frac{\det\left(\Sigma_r(t)\right)^{1/3}}{\det\left(\Sigma_{\text{allow}}\right)^{1/3}} \tag{4.7}$$

where $\mathcal{U}_{\pi_i}$ is the uncertainty ratio of the n-step propagated covariance from executing policy π_i, and $\mathcal{U}_r$ is the uncertainty ratio of the current robot pose. Σ_{allow} is the maximum allowable uncertainty for the vehicle, which we empirically decide.

In Algorithm 2, $\mathcal{U}_r(t)$ acts as a gating function to choose between π_{nbv} and π_{k^*}, while $\mathcal{U}_{\pi_i}(t)$ is used to compute revisit penalties (Section 4.4.4). A similar strategy is employed by Kim et al. to trigger and choose revisit poses [51].

4.4.3 Revisit Trajectories

Given a set of candidate revisit poses $k \leftarrow 1$ to N, we wish to compute a trajectory P_r^k for each policy π_k. The trajectories we compute must have **(i)** similar submap coverage for reliable

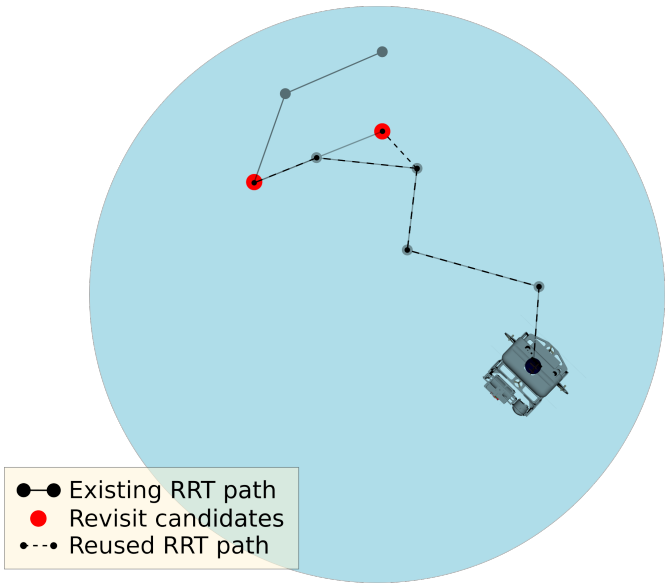


Figure 4.12: Computing a revisit trajectory for the robot to two candidates (red). The cached RRT is displayed with transparency, while the revisit trajectories are in dotted lines. Note the shortcutting operation for one of the candidates.

loop-closures **(ii)** minimal planning overhead. Running a new RRT or A* planner for all N revisit poses is expensive and may not give similar submap coverage. Instead, we cache the existing RRT state and run our *retrace with shortcutting* operation. This is similar to the work by Stenning et al. that uses the existing RRT path for revisits, when the robot must re-localize itself. Example trajectories to revisit candidates are shown in Fig. 4.12.

Algorithm 3 Retrace with shortcutting operations for revisit candidates.

Require:

Cached RRT nodes $\mathcal{R}(t)$

Revisit candidates $\{x_1 \ \dots \ x_N\}$

Ensure:

Revisit trajectories $\{P_r^1 \ \dots \ P_r^N\}$

1: **for** $k \leftarrow 1$ to N **do**

2: Find $i^* = \arg\min(\text{dist}(n_i, x_k))$, the node in $\mathcal{R}(t)$ closest to x_k

3: Compute trajectory $P_r^{i^*}$ via $n_r \ \dots \ n_{i^*}$ in $\mathcal{R}(t)$ ▷ *Retracing*

4: Interpolate trajectory $P_{i^*}^k$ and append to $P_r^{i^*} \rightarrow P_r^k$ ▷ *Shortcutting*

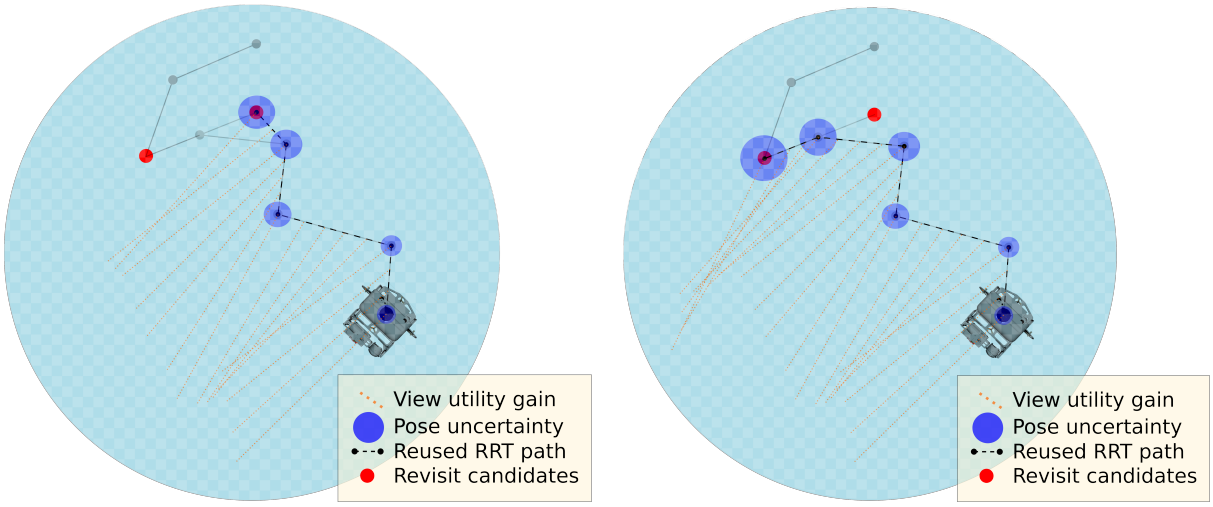
The process is described in Algorithm 3—for each revisit candidate we compute the node in the tree closest to it. We then retrace our path down the tree and perform local interpolation from the closest point to the revisit candidate. This ensures the vehicle follows a similar trajectory, while also cutting short circuitous routes. Note that the trajectory is generated only with respect to $[X, Y, \text{yaw}]$. The dotted lines in Fig. 4.12 show the revisit trajectories for two example candidates. We only generate an inbound trajectory, and the vehicle continues its exploration policy from that point forward.

4.4.4 Penalty Term

To choose the policy P_r^k, we must evaluate a value function for each candidate and choose the action that maximizes this. Kim et al. formulates active SLAM as optimizing a value function that depends on propagated uncertainty and redundant area coverage [51]. The relative weighting of both these terms can be adjusted, which leads to a spectrum of different vehicle behaviors. In general, this value function can depend on uncertainty, trajectory length, time, and energy consumed [14].

Our value function is a penalty defined below in Equation 4.8, and the individual terms are defined in the following sections. α parameterizes the relative weights between the terms, we select the policy π_{k^*} that minimizes the penalty term.

$$\begin{aligned} \mathcal{P}_{\pi_\mathbf{k}}(t) = & \underbrace{\alpha \cdot \mathcal{U}_{\pi_k}(t)}_{\text{uncertainty penalty}} - \underbrace{(1-\alpha) \cdot \text{Gain}(\pi_k)(t)}_{\text{gain bias}} \\ \mathbf{k}^* = & \arg\min \ \mathcal{P}_{\pi_\mathbf{k}}(t) \end{aligned} \tag{4.8}$$



(a) Candidate waypoint I

(b) Candidate waypoint II

Figure 4.13: We reuse the RRT path for revisits, along with short-cutting to the revisit poses. We create virtual pose graph nodes at the tree vertices, and compute the propagated vehicle uncertainty at the candidate waypoint. This uncertainty magnitude is represented as the blue ellipses. We interpolate our revisit path and accrue the gain from these intermediate waypoints for the total revisit gain.

Uncertainty Term

Given a candidate trajectory k, we are interested in the terminating covariance matrix Σ_{π_k} as a measure of uncertainty. This uncertainty ratio term $\mathcal{U}_{\pi_k}(t)$ (Equation 4.7) penalizes revisits to far-off candidates as the drift incurred from that action can make an imperfect state estimate worse. Σ_{π_k} is computed by propagating the current covariance n-steps forward by adding virtual

odometry factors to the existing factor graph (Fig. 4.14). Here n is the number of nodes in the revisit path (Section 4.4.3).

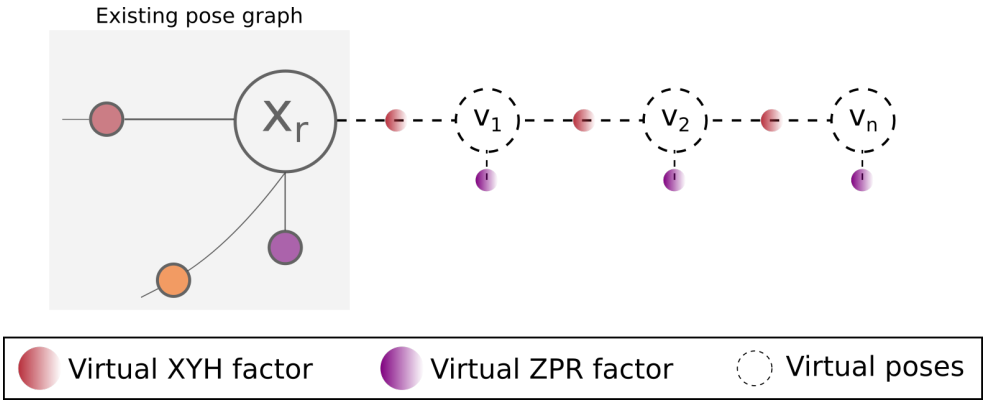


Figure 4.14: Addition of virtual nodes to the existing pose graph. Here x_r is the current robot pose, v_i is a virtual pose node with the connected odometry factors. The graph terminates at the candidate pose v_n.

We consider the worst-case (i.e., no loop-closures) along the virtual trajectory P_r^k and add only odometry factors. The noise on the 3-DoF odometry constraint is scaled proportional to the travel distance between the consecutive nodes. Note that we must assume a constant velocity model for the vehicle, which is an approximation. Once we add the factors, we run a batch optimization on the graph, and recover the marginal pose covariance Σ_{π_k} as carried out by Kaess et al. [47]. We obtain the uncertainty ratio $\mathcal{U}_{\pi_k}(t)$ according to Equation 4.7. Finally, these factors are removed from the graph.

Gain Term

While $\mathcal{U}_{\pi_k}(t)$ penalizes candidates that are far-off, we add a bias term $\mathbf{Gain}(\pi_k)$ that rewards view-utility gain along the trajectory P_r^k. This incentivizes trajectories that further the task of exploration along our revisit path. This is computed by discretizing P_r^k and summing up the visibility gain according to Bircher et al. [11]:

$$\mathbf{Gain}(\pi_k) = \sum_{i=1}^{n} \mathbf{Visible}(\mathcal{M}_{vog}, v_i) e^{-\lambda c(\sigma_{v_{i-1}}^{v_i})} + \mathbf{Gain}(x_r) \tag{4.9}$$

where $\mathcal{M}_{vog}$ is the VOG-Map representation of the world, v_i is the i^{th} virtual node pose, x_r is the current robot pose, $\sigma_{v_{i-1}}^{v_i}$ is the path between nodes and $c(\sigma_{v_{i-1}}^{v_i})$ is the traversal cost. To compute $\mathbf{Visible}(\mathcal{M}_{vog}, v_i)$ we count the total number of visible and unmapped voxels along the sensor ray direction of the v_i.

Weighting term

Fig. 4.13 depicts the uncertainty ellipses and sensor ray directions for the trajectories computed in Section 4.4.3. In Equation 4.8, α weighs the effect of the uncertainty and gain terms in policy

selection. With observe that k^* switches between candidate 2 and 3 by varying α. Thus one can prioritize between drift and view-utility gain in the revisit trajectories.

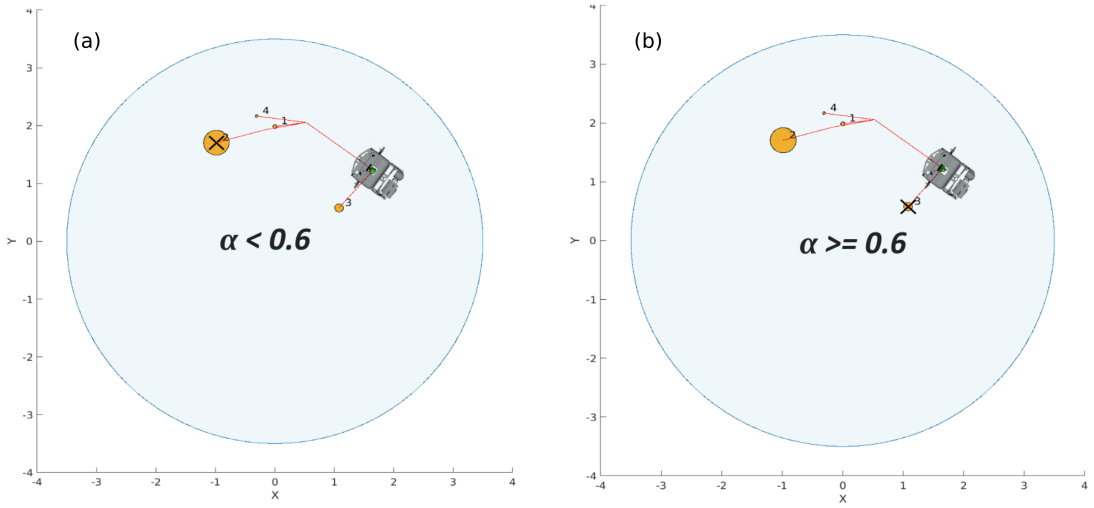


Figure 4.15: Revisit candidates with their corresponding paths. The size of the circles represent the magnitude of $\mathbf{Gain}(\pi_k)$. In (a) the vehicle prefers to go to the candidate with maximum gain but accumulates more drift, while (b) depicts the opposite behavior.

4.5 Simulated Experiments

We evaluate our active SLAM system in simulation—the HAUV exploring a simulated underwater tank environment. The system is benchmarked against **(i)** VOG-Map [37] **(ii)** VOG-Map with random revisits.

4.5.1 Setup

For successful experimentation, our simulator must have capabilities for mapping, planning, low-level vehicle control and state estimation. We use the *UUV Simulator* [59] based on gazebo and modify the vehicle to emulate the HAUV. It has a profiling sonar with 96 beams at a 29° horizontal FoV and a 1° vertical FoV. This is approximated as a 1-D line scanner in the simulator. A visualization of the vehicle imaging a structure up close is shown in Fig. 2.6.

The state estimation has the properties described in Section 2.2. We add Gaussian noise to the absolute quantities—Z, pitch and roll directions. In the X, Y and yaw directions, we corrupt the relative odometry with additive white Gaussian noise. This causes the sonar base poses to drift in the plane, but we assume no drift between scans in a submap. The measurement covariance matrices used are listed in Table 4.1.

Table 4.1: Covariance matrices (defined in Section 4.5.1) used in simulation experiments. They are diagonal square matrices of the form $\mathrm{diag}({M_0}^2, {M_1}^2, \ldots)$. The units for translation and rotation are meters and radians respectively.

Covariances	**Type**	**Square roots of matrix diagonal elements** (M)
$\mathbf{\Sigma_0}$	6-DoF pose prior	0.000138 m, 0.000138 m, 10^{-5} m, 9×10^{-6} m, 10^{-8} m, 10^{-8} m
$\mathbf{\Psi_i}$	3-DoF XYH odometry	0.00414 m, 0.00414 m, 9×10^{-5} rad
$\mathbf{\Phi_i}$	3-DoF ZPR measurement	10^{-5} m, 10^{-8} rad, 10^{-8} rad
$\mathbf{\Delta_k}$	3-DoF loop-closure	0.000138 m, 0.000138 m, 9×10^{-6} rad

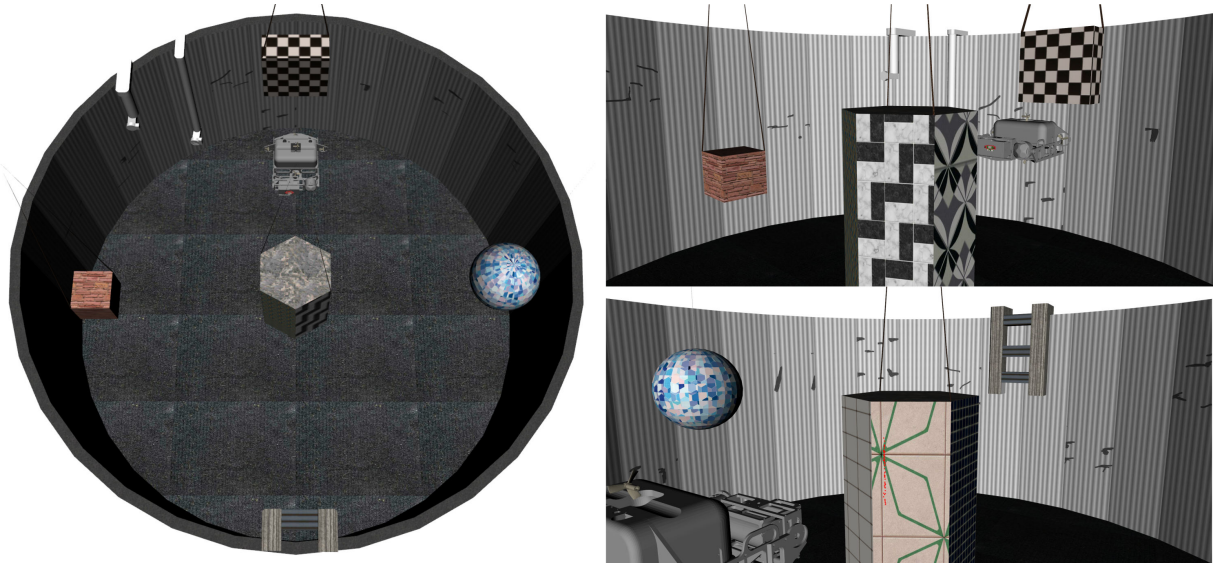


Figure 4.16: Simulation environment with HAUV model pictured. It is a metrically accurate rendering of the real-world tank environment, with targets of different geometries suspended. The central object is hexagonal piling-like structure.

The simulated HAUV is operated in a 3-D environment designed based on the real-world tank environment (Section 3.5.4). The tank and vehicle are metrically accurate, and objects suspended have different geometries. The objective is for the HAUV to explore and map the environment safely, and bound pose uncertainty.

4.5.2 Results

We run 3 different methods to evaluate our active SLAM method. These are (i) active SLAM, (ii) VOG-Map with random revisits, and (iii) VOG-Map. As each run is stochastic, we execute them 5 times and average the results. Each run records 20 submaps before termination, and we ensure that the entire tank is covered. We use the simulation parameter values described in Table 4.2.

Table 4.2: Simulation parameters for active mapping.

Parameter	Value	Comments
Octamap volume	7m x 7m x 1.5m	*Approximate dimensions of the environment*
Collision volume	0.6m x 0.6m x 0.5m	*Approximate dimensions of robot*
Free-space volume	2m x 2m x 2m	*Initial free-space assumption for the robot*
Octamap resolution	0.1m	*Size of each voxel*
RRT max range	2.25m	*Maximum distance of RRT extend operation*
Maximum # submaps	20	*Submap limit before termination*
Dictionary size	50	*Number of 3-D visual words*
Minimum ICP points	100	*Minimum number of points in a cloud for registration*
Sonar FoV	$1°, 29°$	*Vertical and horizontal field-of-view of sonar*
Sonar rate	5 scans/sec	*Rate of scan messages*
Submap size	100 scans	*Number of scans accumulated for one submap*
# Revisit Candidates	3	*Top N candidates to consider for revisit*
Gap between revisits	3 submaps	*Buffer between consecutive revisit actions*
$\mathcal{U}_r$ threshold	10^{-2}	*Uncertainty threshold for revisit action*
α	0.6	*Relative weighting in penalty function* $\mathcal{P}_{\pi_i}$
Virtual velocity	0.5 m/s	*Constant velocity assumption for uncertainty propagation*

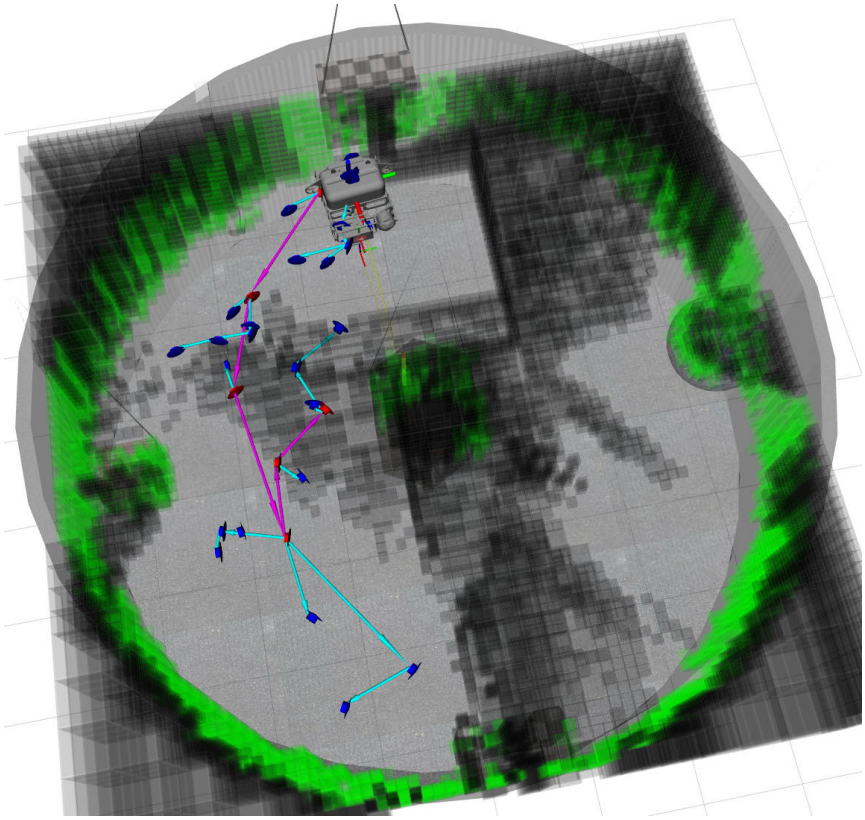


Figure 4.17: Our HAUV exploring the simulation environment. We grow an RRT tree based on information gain and choose the best edge to execute. The Octamap indicates free, unknown and occupied space.

In Fig. 4.18, We assess the quality of the global map we generate from active SLAM, and compare it against the dead-reckoning map. We can see structures are better aligned when incorporating the loop closures. This is backed up by our quantitative metrics in Table 4.3.

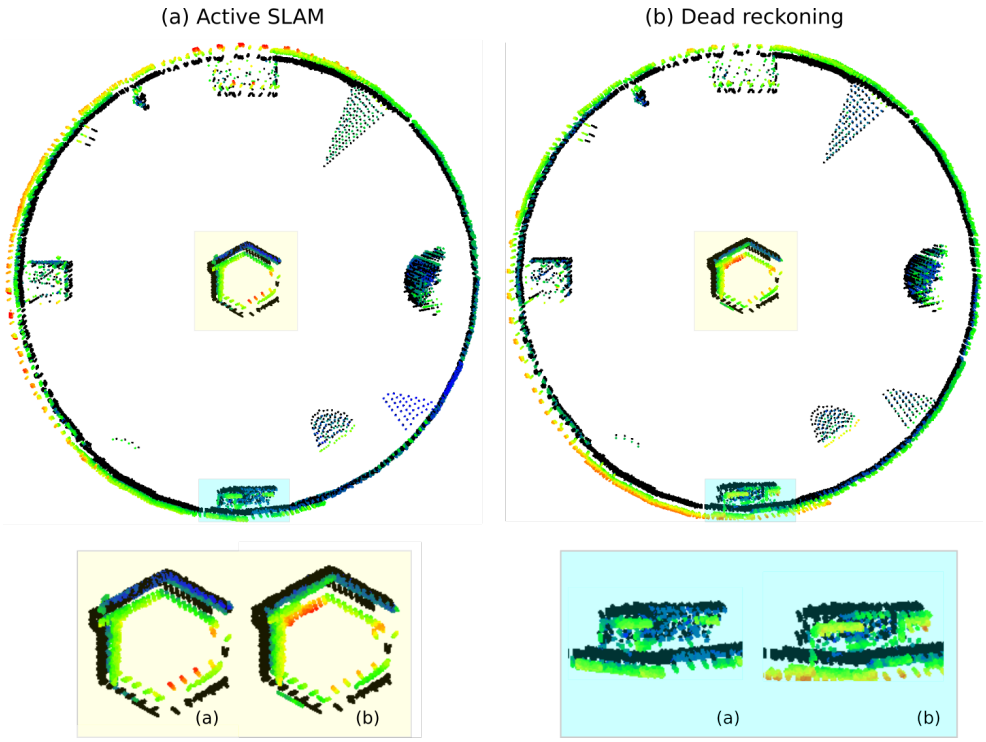


Figure 4.18: Ground truth point cloud with resultant map, where heatmap indicates the cloud to cloud error. This global map is a collation of 20 submaps in the simulation environment. We see that qualitatively, there is better alignment in structures such as the central piling and the ladder at the bottom.

Table 4.3: Map quality of the active SLAM solution as compared to the dead-reckoning estimates. While the former incorporates the optimized poses to deform local submaps, the latter considers drifting odometry as the base poses of the submaps. Mean error is the cloud to cloud error metric.

Dead-reckoning solution		**Active SLAM solution**	
Mean error (m)	std. deviation	**Mean error (m)**	**std. deviation**
0.066273	0.041402	**0.055343**	**0.037989**

Table 4.4: Average cloud to cloud error over 5 runs of the exploration policies. We see that the active method gives the best quality map, with the maximum number of loop closures. Interestingly, random revisits performs better than pure exploration. This implies that going back to places helps bound vehicle pose uncertainty.

	No revisits	Random revisits	**Active SLAM**
Mean error (m)	0.1001	0.0915	**0.0756**
Mean num. loop closures	12.6	15.8	**17**

We then compare our method with the two benchmarks, shown in Table 4.4. The active method performs the best, with the largest number of loop closures. We also note that random revisits performs better than pure exploration, proving that going back to places helps re-localize. We track the pose uncertainty ratio of the vehicle over runtime and plot them in Fig. 4.19 and 4.20. We see that in the our method the mean uncertainty ratio is close to the allowable threshold (Fig. 4.19), while it is higher in VOG-Map's case (Fig. 4.20). This is because we are able to bound vehicle drift through loop closures, while VOG-Map performs uncertainty-agnostic exploration.

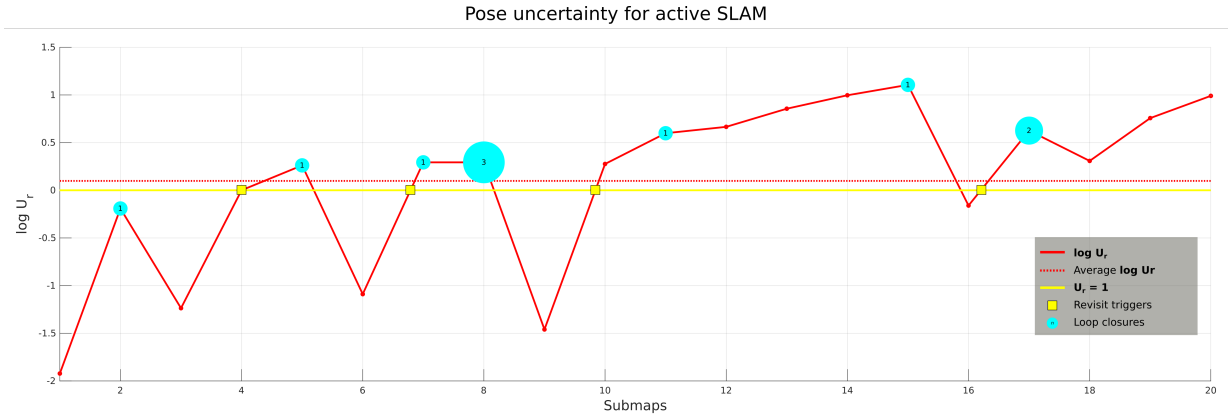


Figure 4.19: Plot showing the uncertainty ratio vs. submaps for the active SLAM method. The cyan circles denote loop closure occurrences and yellow line is the allowable uncertainty threshold. The mean uncertainty ratio lies close to this threshold as a result of informative loop closures.

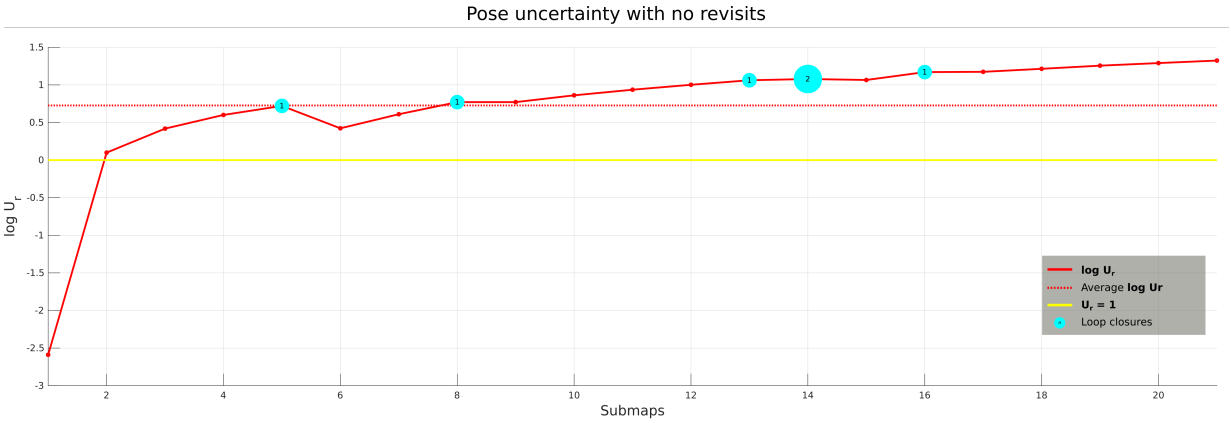


Figure 4.20: Plot showing the uncertainty ratio vs. submaps for VOG-Map. The cyan circles denote loop closure occurrences and yellow line is the allowable uncertainty threshold. Here, the mean uncertainty ratio is away from the threshold due to the lack of informative loop closures.

We analyze the revisits in Table 4.5, comparing random revisits and the penalty-informed ones. While the distances travelled are similar, the active policy gives better loop closures. We can tune the α value in Equation 4.8 to incur lesser distance travelled. Finally, we visualize the top revisit poses based on GloSSy scores in Fig. 4.21. The best revisit poses are looking at objects in the tank (ladder, piling), while bad ones look at the tank wall or fail to accumulate sufficient submaps.

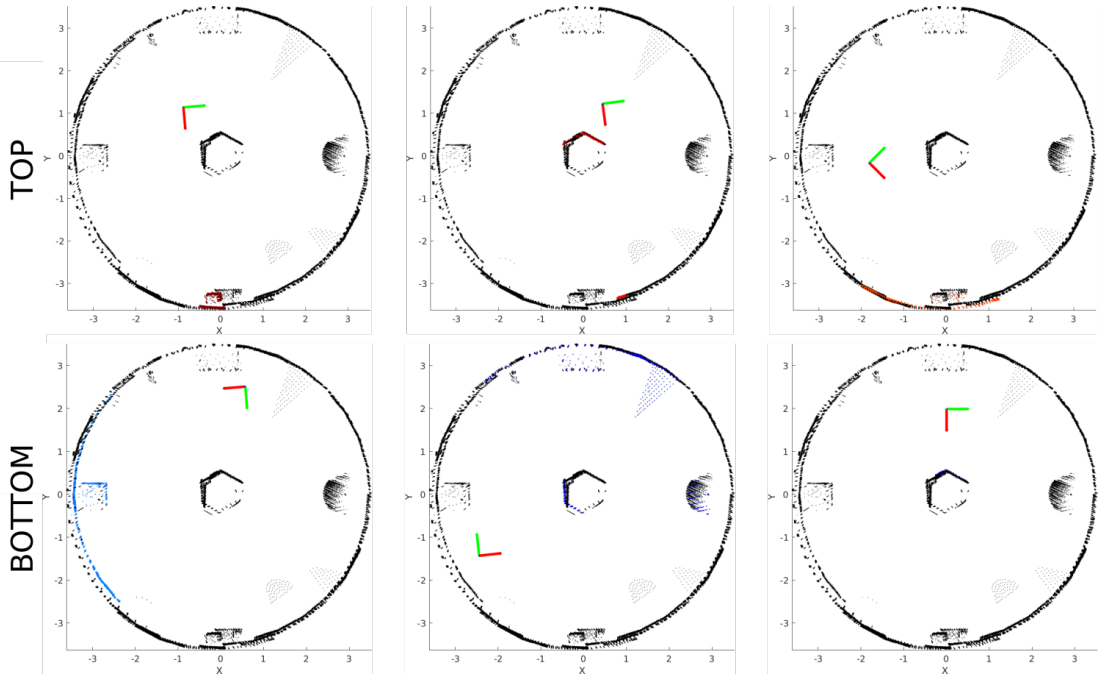


Figure 4.21: Top/bottom 3 revisit poses according to GloSSy scores in a simulation run. The gray point cloud represents the complete global map, and the colored sections show the submap that belongs to the revisit pose. The 6DoF base pose of the vehicle is visualized.

Table 4.5: Average number of revisits executed and distance travelled over 5 runs. We see that the distance travelled is similar, but Table 4.4 tells us that the active policy gives better loop closures. Modifying the value of α can give lesser revisit distance.

	Random revisits	**Active SLAM**
Mean num. of revisits	3.6	**3.4**
Mean total revisit distance	5.44956	**5.35588**

The simulation experiments show the merit of revisiting, and the need for an informed method to select revisit actions. While exploration over long dives causes dead-reckoning to drift, we can bound this drift with these loop-closures. Adding the visual localization method from Section 3.3 could further benefit accuracy and provide sensing redundancy.

Chapter 5

Conclusion

5.1 Contributions

In this thesis, we address the challenges in localization and exploration in indoor underwater environments. We present visual through-water localization to address the state estimation problem, and active mapping to obtain dense, accurate maps of the environment. We propose these methods towards the goal of an integrated SLAM framework for underwater inspection of SNF pools and ship ballast tanks. While the algorithms are tested on the Bluefin HAUV, they can be applied on hovering AUVs with similar sensing payloads.

In Chapter 3, we present a novel visual localization framework for underwater vehicles. There exists no prior work that takes cues from above the water surface for underwater visual SLAM. By utilizing an onboard upward-facing stereo camera, our method is less prone to failure in cluttered environments as compared to traditional line-of-sight methods. We detail the challenges that refraction presents and develop a correction module. Previously, refraction correction had only been addressed for aerial photogrammetry and lens housing compensation. We formulate the landmark-based stereo SLAM problem and address the challenges faced by the frontend. We evaluate the method through simulation and a dozen real-world underwater experiments. While it assumes good visibility and illumination, we confirm it to perform well even in difficult conditions.

In Chapter 4, we extend the VOG-Map system to perform active SLAM with submaps. We develop a safe exploration policy for mapping in cluttered underwater environments with bounded pose uncertainty. The resulting dense maps can provide valuable information for inspection and monitoring of these facilities. We introduce the GloSSy metric for submap saliency, and use it to identify ideal revisit poses for loop-closures. We select a revisit policy based on robot uncertainty and information gain, and previous planning iterations to execute the best policy. We compare our method in simulation and show improvements over an uncertainty-agnostic SLAM framework.

5.2 Observations and Future Work

The work analyzes the potential of vision and sonar modalities individually, and results show potential for their integrated use. Multiple modalities enable redundancy, and algorithms that fuse these modalities have recently found success in the underwater domain [74].

In Chapter 3, our approximation of water surface planarity can be improved by modeling for waves and ripples [32]. The generic point feature frontend can be improved by taking ideas from the state-of-the-art in visual SLAM. It can be replaced by a dense or semi-dense method for mapping applications, or combined with lines for robust detection [35]. In larger environments, we can also integrate loop closure detection. For computational efficiency, an over-compensation factor can be used in the refraction module, or it can be completely replaced by a lookup-table [58]. A large baseline stereo pair will guarantee better results for distant stereo points. Further, we can also rectify stereo images to exploit epipolar constraints for faster matching. Point correspondences are restricted to epipolar curves due to the refractive interface, as detailed by [34]. The SLAM framework may also be extended to support the use of monocular cameras.

In Chapter 4, future work will be towards evaluating the system in real-world scenarios such as the tank environment (Fig. 3.8). The existing planner queries voxels over multiple submaps, leading to query times that rise linearly with the number of submaps encountered. With this computational bottleneck, the planner may in fact be operating on an outdated map. This compromises safe autonomy, and leads to ill-informed exploration policies. A system that maintains a global map without the need to merge from scratch at every loop closure iteration would be ideal. Propagating covariance through virtual nodes assumes a simplistic constant velocity motion model, and a more nuanced formulation is necessary. Submap-based SLAM can draw inspiration from LiDAR methods for map representation and place recognition. Dubé et al. [24] segment the point-cloud to obtain discriminative features and a compact representation.

Bibliography

[1] *Inspection of the Ballast of a Container Ship*. URL https://www.flyability.com/. (document), 1.1

[2] *HFIR reactor pool, Flickr photo by oakridgelabnews shared under a Creative Commons (BY) license*. . URL https://flickr.com/photos/oakridgelab/4092138391. (document), 1.1

[3] *Spent Fuel Pool, Flickr photo by NRCgov shared under a Creative Commons (BY-NC-ND) license*. . URL https://flickr.com/photos/nrcgov/6800268096. (document), 1.1

[4] *Spent Nuclear Fuel Pool, Flickr photo by by U.S. GAO shared as a United States Government Work (PD)*. . URL https://flickr.com/photos/usgao/8009918545. (document), 1.1

[5] *Spent nuclear fuel rods stored underwater at the Bruce Power site near Tiverton, Ontario, Canada by Norm Betts/Landov*. . URL https://kids.britannica.com/students/assembly/view/177382. (document), 4.1

[6] Amit Agrawal, Srikumar Ramalingam, Yuichi Taguchi, and Visesh Chari. A theory of multi-layer flat refractive geometry. *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 3346–3353, 2012. 3.3

[7] Motilal Agrawal. A Lie algebraic approach for consistent pose registration for general Euclidean motion. *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1891–1897, 2006. 3.4.1

[8] Oleksandr Bailo, Francois Rameau, Kyungdon Joo, Jinsun Park, Oleksandr Bogdan, and In So Kweon. Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution. *Pattern Recognition Letters*, 106:53–60, 2018. 3.4.2

[9] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988. 4.1

[10] Chris Beall, Frank Dellaert, Ian Mahon, and Stefan B Williams. Bundle adjustment in large-scale 3d reconstructions based on underwater robotic surveys. In *OCEANS 2011 IEEE-Spain*, pages 1–6. IEEE, 2011. 3.3

[11] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon" next-best-view" planner for 3d exploration. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1462–1468. IEEE, 2016. 1.2, 4.2.1, 4.2.3, 4.4.4

[12] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016. 2.1

[13] Marc Carreras, Pere Ridao, Rafael García, and Tudor Nicosevici. Vision-based localization of an underwater robot in a structured environment. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 1, pages 971–976. IEEE, 2003. 3.1

[14] Henry Carrillo, Ian Reid, and José A Castellanos. On the comparison of uncertainty criteria for active slam. In *2012 IEEE International Conference on Robotics and Automation*, pages 2080–2087. IEEE, 2012. 4.4.2, 4.4.2, 4.4.4

[15] Autonomous Undersea Vehicle Applications Center. Mares - auvac, . URL `https://auvac.org/platforms/view/208`. (document), 2.2

[16] DFKI GmbH Robotics Innovation Center. Dagon - robot systems - robotics innovation center - dfki gmbh, . URL `https://robotik.dfki-bremen.de/en/research/robot-systems/dagon.html`. (document), 2.2

[17] Stephen M Chaves, Ayoung Kim, Enric Galceran, and Ryan M Eustice. Opportunistic sampling-based active visual slam for underwater inspection. *Autonomous Robots*, 40(7): 1245–1265, 2016. 4.1

[18] Byung-Hak Cho, Seung-Hyun Byun, Chang-Hoon Shin, Jang-Bum Yang, Sung-Il Song, and Jung-Mook Oh. KeproVt: Underwater robotic system for visual inspection of nuclear reactor internals. *Nuclear engineering and design*, 231(3):327–335, 2004. (document), 3.2

[19] Felipe Codevilla, Joel De O Gaya, N Duarte, and S Botelho. Achieving turbidity robustness on underwater images local feature detection. *International Journal of Computer Vision*, 60 (2):91–110, 2004. 3.4.2, 3.4.2

[20] Mark Cummins and Paul Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008. 4.2.4

[21] Science Daily. Robots take over inspection of ballast tanks on ships, 2014. URL `https://www.sciencedaily.com/releases/2014/11/141121102646.htm`. 1.1

[22] Frank Dellaert. Factor graphs and GTSAM: A hands-on introduction. Technical report, Georgia Institute of Technology, 2012. 3.4.4

[23] Frank Dellaert and Michael Kaess. Factor graphs for robot perception. *Foundations and Trends in Robotics*, 6(1-2):1–139, 2017. (document), 2.1, 3.4.1

[24] Renaud Dubé, Andrei Cramariuc, Daniel Dugas, Juan Nieto, Roland Siegwart, and Cesar Cadena. Segmap: 3d segment mapping using data-driven descriptors. *arXiv preprint arXiv:1804.09557*, 2018. 5.2

[25] Nathaniel Fairfield and David Wettergreen. Active localization on the ocean floor with multibeam sonar. In *OCEANS 2008*, pages 1–10. IEEE, 2008. 4.1

[26] Nathaniel Fairfield and David Wettergreen. Active slam and loop prediction with the

segmented map using simplified models. In *Field and service robotics*, pages 173–182. Springer, 2010. 4.1

[27] Nathaniel Fairfield, David Wettergreen, and George Kantor. Segmented slam in three-dimensional environments. *Journal of Field Robotics*, 27(1):85–103, 2010. 4.2.2

[28] Maxime Ferrera, Julien Moras, Pauline Trouvé-Peloux, and Vincent Creuze. Real-time monocular visual odometry for turbid and dynamic underwater environments. *Sensors*, 19 (3):687, 2019. 3.4.2

[29] CMU Field Robotics Center. Deep phreatic thermal explorer (depthx) project. URL `https://frc.ri.cmu.edu/depthx/`. (document), 2.2

[30] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. *AAAI Conf. on Artificial Intelligence*, 1999 (343-349):2–2, 1999. 3.2

[31] John G. Fryer. Photogrammetry through shallow water. *Australian J. of Geodesy, Photogrammetry and Surveying*, 38:25–38, 1983. 3.2, 3.3.2

[32] John G. Fryer and H. T. Kniest. Errors in depth determination caused by waves in through-water photogrammetry. *The Photogrammetric Record*, 11(66):745–753, 1985. 5.2

[33] Rafael Garcia and Nuno Gracias. Detection of interest points in turbid underwater images. *OCEANS 2011*, pages 1–9, 2011. 3.4.2, 3.4.2

[34] Jason Gedge, Minglun Gong, and Yee-Hong Yang. Refractive epipolar geometry for underwater stereo matching. *Computer and Robot Vision (CRV), 2011 Canadian Conference on*, pages 146–152, 2011. 5.2

[35] Ruben Gomez-Ojeda, David Zuñiga-Noël, Francisco-Angel Moreno, Davide Scaramuzza, and Javier Gonzalez-Jimenez. PL-SLAM: a stereo SLAM system through the combination of points and line segments. *arXiv:1705.09479*, 2017. 5.2

[36] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, Jianwei Wan, and Ngai Ming Kwok. A comprehensive performance evaluation of 3d local feature descriptors. *International Journal of Computer Vision*, 116(1):66–89, 2016. 4.2.4

[37] Bing-Jui Ho, Paloma Sodhi, Pedro Teixeira, Ming Hsiao, Tushar Kusnur, and Michael Kaess. Virtual occupancy grid map for submap-based pose graph slam and planning in 3d environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2175–2182. IEEE, 2018. (document), 1.2, 1.3, 2.2, 2.2, 4.1, 4.2, 4.2.3, 4.5, 4.4.1, 4.5

[38] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987. 3.5.1

[39] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34(3):189–206, 2013. 4.2.1, 4.2.3

[40] Franz S Hover, Ryan M Eustice, Ayoung Kim, Brendan Englot, Hordur Johannsson, Michael Kaess, and John J Leonard. Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *The International Journal of Robotics Research*, 31(12):

1445–1464, 2012. 1.1, 2.2

[41] Laurent Itti and Christof Koch. Computational modelling of visual attention. *Nature reviews neuroscience*, 2(3):194, 2001. 4.2.4

[42] WooYeon Jeong and Kyoung Mu Lee. CV-SLAM: A new ceiling vision-based SLAM technique. *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3195–3200, 2005. 3.2

[43] Matthew Johnson-Roberson, Oscar Pizarro, Stefan B Williams, and Ian Mahon. Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. *Journal of Field Robotics*, 27(1):21–51, 2010. 3.3

[44] Jongdae Jung, Yeongjun Lee, Donghoon Kim, Donghwa Lee, Hyun Myung, and Hyun-Taek Choi. AUV SLAM using forward/downward looking cameras and artificial landmarks. *Underwater Technology, 2017 IEEE*, pages 1–3, 2017. 3.2

[45] Jongdae Jung, Ji-Hong Li, Hyun-Taek Choi, and Hyun Myung. Localization of AUVs using visual information of underwater structures and artificial landmarks. *Intelligent Service Robotics*, 10(1):67–76, 2017. 3.2

[46] Michael Kaess. Localization and mapping with imaging sonar. In *International Conference on Robotics and Automation, Underwater Robotics Perception Workshop*, 2019. (document), 4.4

[47] Michael Kaess and Frank Dellaert. Covariance recovery from a square root information matrix for data association. *Robotics and autonomous systems*, 57(12):1198–1210, 2009. 4.4.2, 4.4.4

[48] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Trans. Robotics*, 24(6):1365–1378, 2008. 2.1, 3.4.1

[49] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John Leonard, and Frank Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Intl. J. of Robotics Research*, 31(2):216–235, 2012. 2.1, 3.4.1, 3.4.4

[50] Jack Kiefer. General equivalence theory for optimum designs (approximate theory). *The annals of Statistics*, pages 849–879, 1974. 4.4.2, 4.4.2

[51] Ayoung Kim and Ryan M Eustice. Real-time visual slam for autonomous underwater hull inspection using visual saliency. *IEEE Transactions on Robotics*, 29(3):719–733, 2013. (document), 4.1, 4.7, 4.3.2, 4.3.2, 4.3.2, 4.4.2, 4.4.4

[52] Thomas Kollar and Nicholas Roy. Trajectory optimization using reinforcement learning for map exploration. *The International Journal of Robotics Research*, 27(2):175–196, 2008. 4.4.2

[53] Steven M LaValle and James J Kuffner Jr. Rapidly-exploring random trees: Progress and prospects. 2000. (document), 4.2.1, 4.3

[54] Timothy E Lee and Nathan Michael. State estimation and localization for ROV-based reactor pressure vessel inspection. *Field and Service Robotics*, pages 699–715, 2018. (document), 3.2

[55] Cindy Leung, Shoudong Huang, and Gamini Dissanayake. Active slam using model predictive control and attractor based exploration. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5031. IEEE, 2006. 4.4.2

[56] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 3.4.2

[57] Tomasz Łuczyński, Max Pfingsthorn, and Andreas Birk. The pinax-model for accurate and efficient refraction correction of underwater cameras in flat-pane housings. *Ocean Engineering*, 133:9–22, 2017. 3.3

[58] Hans-Gerd Maas. On the accuracy potential in underwater/multimedia photogrammetry. *Sensors*, 15(8):18140–18152, 2015. 3.3.3, 3.3.3, 5.2

[59] Musa Morena Marcusso Manhães, Sebastian A Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach. Uuv simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–8. IEEE, 2016. 4.5.1

[60] S. E. Masry. Measurement of water depth by the analytical plotter. *The International Hydrographic Review*, 52(1), 2015. 3.2

[61] Hans Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 116–121. IEEE, 1985. 4.2.3

[62] Raul Mur-Artal and Juan D Tardós. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. 3.4.3

[63] Toshimi Murase, Miho Tanaka, Tomomi Tani, Yuko Miyashita, Naoto Ohkawa, Satoshi Ishiguro, Yasuhiro Suzuki, Hajime Kayanne, and Hiroya Yamano. A photogrammetric correction procedure for light refraction effects at a two-medium boundary. *Photogr. Eng. & Remote Sensing*, 74(9):1129–1136, 2008. 3.2, (i)

[64] Robin R Murphy, Eric Steimle, Michael Hall, Michael Lindemuth, David Trejo, Stefan Hurlebaus, Zenon Medina-Cetina, and Daryl Slocum. Robot-assisted bridge inspection. *Journal of Intelligent & Robotic Systems*, 64(1):77–95, 2011. 1.1

[65] Sarfraz Nawaz, Muzammil Hussain, Simon Watson, Niki Trigoni, and Peter N Green. An underwater robotic network for monitoring nuclear waste storage pools. *International Conference on Sensor Systems and Software*, pages 236–255, 2009. (document), 3.2

[66] Shahriar Negahdaripour, Hicham Sekkati, and Hamed Pirsiavash. Opti-acoustic stereo imaging: On system calibration and 3-d target reconstruction. *IEEE Transactions on image processing*, 18(6):1203–1214, 2009. 3.3

[67] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168. Ieee, 2006. 4.2.4, 4.3.2

[68] Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In *European conference on computer vision*, pages 490–503. Springer, 2006.

4.2.4

[69] Paul Ozog, Nicholas Carlevaris-Bianco, Ayoung Kim, and Ryan M Eustice. Long-term mapping techniques for ship hull inspection and surveillance using an autonomous underwater vehicle. *Journal of Field Robotics*, 33(3):265–289, 2016. 1.1

[70] Christos Papachristos, Shehryar Khattak, and Kostas Alexis. Uncertainty-aware receding horizon exploration and mapping using aerial robots. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 4568–4575. IEEE, 2017. 4.2.3

[71] Liam Paull, Sajad Saeedi, Mae Seto, and Howard Li. AUV navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, 39(1):131–149, 2014. 3.2

[72] Andrej Pázman. *Foundations of optimum experimental design*, volume 14. Springer, 1986. 4.4.2, 4.4.2

[73] Friedrich Pukelsheim. *Optimal design of experiments*. SIAM, 2006. 4.4.2, 4.4.2

[74] Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. Sonar visual inertial slam of underwater structures. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2018. 5.2

[75] Carolina Redondo-Cabrera, Roberto J López-Sastre, Javier Acevedo-Rodriguez, and Saturnino Maldonado-Bascón. Surfing the point clouds: Selective 3d spatial pyramids for category-level object recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3458–3465. IEEE, 2012. 4.2.4

[76] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. *Intl. Conf. on Computer Vision (ICCV)*, pages 2564–2571, 2011. 3.4.2, 3.4.2

[77] Ian Rust and Harry Asada. A dual-use visible light approach to integrated communication and localization of underwater robots with application to non-destructive nuclear reactor inspection. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2445–2450, 2012. 3.2

[78] Alan C Schultz and William Adams. Continuous localization using evidence grids. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, volume 4, pages 2833–2839. IEEE, 1998. 4.2.2

[79] Saab Seaeye. Sabertooth single hull saab seaeye. URL `https://www.saabseaeye.com/solutions/underwater-vehicles/sabertooth-single-hull`. (document), 2.2

[80] Mark R Shortis and Euan S Harvey. Design and calibration of an underwater stereo-video system for the monitoring of marine fauna populations. *International Archives of Photogrammetry and Remote Sensing*, 32:792–799, 1998. 3.3

[81] Ivan Sipiran and Benjamin Bustos. Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes. *The Visual Computer*, 27(11):963, 2011. 4.3.1

[82] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *null*, page 1470. IEEE, 2003. 4.2.4, 4.3.2

[83] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles*, pages 167–193. Springer, 1990. 2.1

[84] Olga Sorkine-Hornung and Michael Rabinovich. Least-squares rigid motion using svd. *Computing*, 1(1), 2017. 4.2.3

[85] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 573–580, 2012. 3.5.1

[86] Pedro V Teixeira, Michael Kaess, Franz S Hover, and John J Leonard. Underwater inspection using sonar-based volumetric submaps. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4288–4295. IEEE, 2016. 1.1, 2.2, 2.2, 4.2.2, 4.2.3

[87] G. C. Tewinkel. Water depths from aerial photographs. *Photogrammetric Engineering*, 29 (6):1037–1042, 1963. 3.2, (i)

[88] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010. 4.3.1

[89] Tali Treibitz, Yoav Schechner, Clayton Kunz, and Hanumant Singh. Flat refractive geometry. *IEEE Trans. Pattern Anal. Machine Intell.*, 34(1):51–65, 2012. 3.3

[90] Jerome Vaganay, Mike Elkins, Dave Esposito, Will O'Halloran, Franz Hover, and Mike Kokko. Ship hull inspection with the HAUV: US Navy and NATO demonstrations results. *OCEANS 2006*, pages 1–6, 2006. 1.2, 2.2

[91] Eduard Vidal, Juan David Hernández, Klemen Istenič, and Marc Carreras. Online view planning for inspecting unexplored underwater structures. *IEEE Robotics and Automation Letters*, 2(3):1436–1443, 2017. 4.2.3

[92] Nick Weidner, Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. Underwater cave mapping using stereo vision. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5709–5715, 2017. 3.1

[93] R. M. Westaway, S. N. Lane, and D. M. Hicks. The development of an automated correction procedure for digital photogrammetry for the study of wide, shallow, gravel-bed rivers. *Earth Surface Processes and Landforms*, 25(2):209–226, 2000. 3.2

[94] Eric Westman and Michael Kaess. Underwater AprilTag SLAM and calibration for high precision robot localization. Technical Report CMU-RI-TR-18-43, Carnegie Mellon University, October 2018. 2.2, 3.4.1

[95] Brian Yamauchi and Pat Langley. Place learning in dynamic real-world environments. *Proceedings of RoboLearn*, 96:123–129, 1996. 4.2.2