Summer 2020

A Multi-Sensor Fusion-Based Underwater Slam System

Sharmin Rahman

A MULTI-SENSOR FUSION-BASED UNDERWATER SLAM SYSTEM

by

Sharmin Rahman

Bachelor of Science
Military Institute of Science and Technology, 2012

Master of Science
University of South Carolina, 2020

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2020

Accepted by:

Ioannis Rekleitis, Major Professor

Jason O'Kane, Committee Member

Song Wang, Committee Member

John R. Rose, Committee Member

Nikolaos Vitzilaios, Committee Member

Cheryl Addy, Vice Provost and Dean of Graduate School

Dedication

To my two little sisters.

Acknowledgments

I would like to thank my supervisor Dr. Ioannis Rekleitis for his endless support and guidance through all these years. Yiannis, I certainly learnt a lot from you. Thank you for your kindness, listening to me hours after hours, and your sincere willingness to help me beyond your role as academic advisor. I would like to express my gratitude to Dr. Alberto Quattrini Li for providing me your valuable suggestions and criticisms to improve my work when you were a postdoc at University of South Carolina and even after that. Alberto, thank you so much for making me a better researcher and helping me to get out of my comfort zone and see the bigger picture.

I want to thank my lab mates in Autonomous Field Robotics Laboratory (AFRL) for your unconditional support and help during my hard times. Marios, Modasshir, Bharat, Jason, Nare, Mike, and Shervin I will be forever grateful to you for making my PhD journey a fun one. I will always cherish the memories of going out for lunch together and the troubles we went through to decide where to go. Finally, my deepest appreciation to Dr. Jason O'Kane, Dr. Song Wang, Dr. John Rose, and Dr. Nikolaos Vitzilaios for serving in my committee and providing your valuable inputs, comments, and suggestions.

Abstract

This dissertation addresses the problem of real-time Simultaneous Localization and Mapping (SLAM) in challenging environments. SLAM is one of the key enabling technologies for autonomous robots to navigate in unknown environments by processing information on their on-board computational units. In particular, we study the exploration of challenging GPS-denied underwater environments to enable a wide range of robotic applications, including historical studies, health monitoring of coral reefs, underwater infrastructure inspection e.g., bridges, hydroelectric dams, water supply systems, and oil rigs. Mapping underwater structures is important in several fields, such as marine archaeology, Search and Rescue (SaR), resource management, hydrogeology, and speleology. However, due to the highly unstructured nature of such environments, navigation by human divers could be extremely dangerous, tedious, and labor intensive. Hence, employing an underwater robot is an excellent fit to build the map of the environment while simultaneously localizing itself in the map.

The main contribution of this dissertation is the design and development of a real-time robust SLAM algorithm for small and large scale underwater environments. SVIn – a novel tightly-coupled keyframe-based non-linear optimization framework fusing Sonar, Visual, Inertial and water depth information with robust initialization, loop-closing, and relocalization capabilities has been presented. Introducing acoustic range information to aid the visual data, shows improved reconstruction and localization. The availability of depth information from water pressure enables a robust initialization and refines the scale factor, as well as assists to reduce the drift for the tightly-coupled integration. The complementary characteristics of these sensing

modalities provide accurate and robust localization in unstructured environments with low visibility and low visual features – as such make them the ideal choice for underwater navigation. The proposed system has been successfully tested and validated in both benchmark datasets and numerous real world scenarios. It has also been used for planning for underwater robot in the presence of obstacles. Experimental results on datasets collected with a custom-made underwater sensor suite and an autonomous underwater vehicle (AUV) Aqua2 in challenging underwater environments with poor visibility, demonstrate performance never achieved before in terms of accuracy and robustness.

To aid the sparse reconstruction, a contour-based reconstruction approach utilizing the well defined edges between the well lit area and darkness has been developed. In particular, low lighting conditions, or even complete absence of natural light inside caves, results in strong lighting variations, e.g., the cone of the artificial video light intersecting underwater structures and the shadow contours. The proposed method utilizes these contours to provide additional features, resulting into a denser 3D point cloud than the usual point clouds from a visual odometry system. Experimental results in an underwater cave demonstrate the performance of our system. This enables more robust navigation of autonomous underwater vehicles using the denser 3D point cloud to detect obstacles and achieve higher resolution reconstructions.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Simultaneous Localization and Mapping Problem

Exploration and mapping underwater environments such as caves, bridges, dams, and shipwreck, are extremely important tasks for the economy, conservation, and scientific discoveries. Currently, most of these efforts are performed by divers that need to take measurements manually using a grid and measuring tape, or using hand-held sensors [39], and the data is post-processed afterwards. Autonomous Underwater Vehicles (AUVs) present unique opportunities to automate this process; however, there are several open problems that still need to be addressed for reliable robotic exploration and navigation, including real-time robust Simultaneous Localization and Mapping (SLAM), the focus of this dissertation. Fig. 1.1 shows mapping of an underwater cave by an autonomous robot.

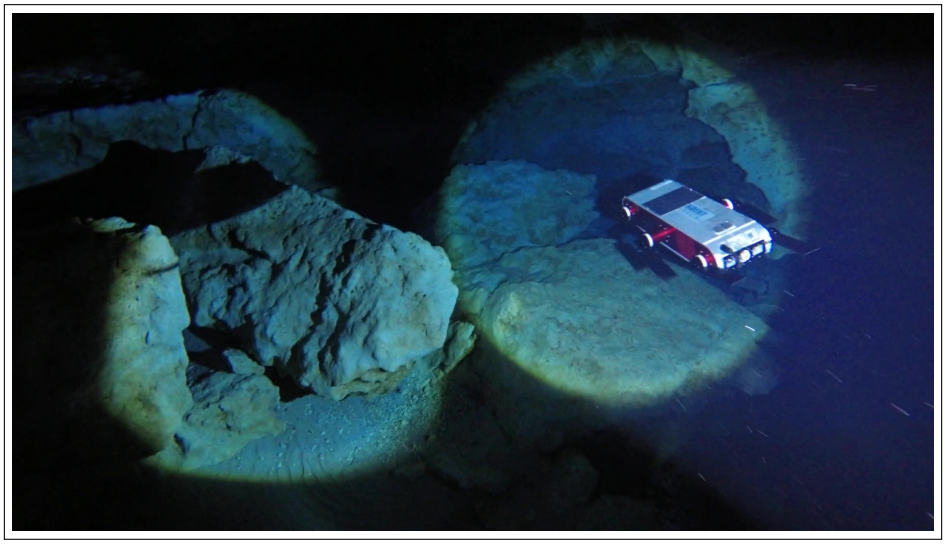


Figure 1.1: Underwater cave mapping using Aqua2 autonomous robot.

Mapping is the process of constructing a map of an unknown environment online given that the location of the robot is known and **localization** is the process of estimating the robot's poses given a map. **Simultaneous Localization and Mapping** (SLAM) aims to solve the localization problem by building a map of the environment in order to localize the robot within this map simultaneously by processing information from on-board sensors. SLAM is a much harder problem as both the locations and map are unknown and errors in map and pose estimates are correlated. In real world, the associations between observations and the landmarks are unknown, and wrong data associations could lead to divergence.

For a robot to localize itself in the environment, there might exist an external infrastructure, e.g., Global Positioning System (GPS), motion capture system, etc. However, these systems might not be always available depending on environments such as underwater and may not be as accurate as required. As such, the simplest method to localize a robot is to process the on-board sensory information to recover the path incrementally which is known as **odometry**. However, odometry only provides local consistency as computing incremental motion inevitably accumulates drifts in the trajectory. On the other hand, the goal of SLAM is to obtain a global, consistent, and drift-free estimate of the robot path by keeping track of a map of the environment.

1.2 Challenges and Importance of Underwater SLAM

Underwater state estimation by an autonomous robot has many open challenges, including visibility, light and color attenuation [87], floating particulates, blurriness, varying illumination, and lack of features [65]. Indeed, in some underwater environments, there is very low visibility, which prevents seeing objects that are only a few meters away – Fig. 1.3 represents few of these challenges. Such challenges make underwater localization very challenging, leaving an interesting gap to be investigated

in the current state of the art. In addition, light attenuates with depth, with some wavelengths of the ambient light being absorbed very quickly – e.g., the red wavelength is almost completely absorbed at 5 m. This results in a change in appearance of the image, which will affect feature tracking, even in grayscale.

One of the primary motivations of this work is the mapping of underwater caves where exploration by human divers is an extremely dangerous operation due to the harsh environment. Figure 1.2 shows a typical cave segment. In addition to underwater vision constraints, cave environments suffer from the absence of natural illumination. Currently, for surveying, divers manually measure distances between attachment points, using a cave-line with knots every 3 m. Simultaneously, the divers also measure the water depth at each attachment point, as well as the azimuth of the line leading to the next attachment point. This process is error-prone and time consuming, and at greater depths results in significant decompression times, where total dive time can reach between 15 to 28 hours per dive. Therefore, employing robotic technology to map the cave would not only reduce the cognitive load of the divers, but also save time and resources.

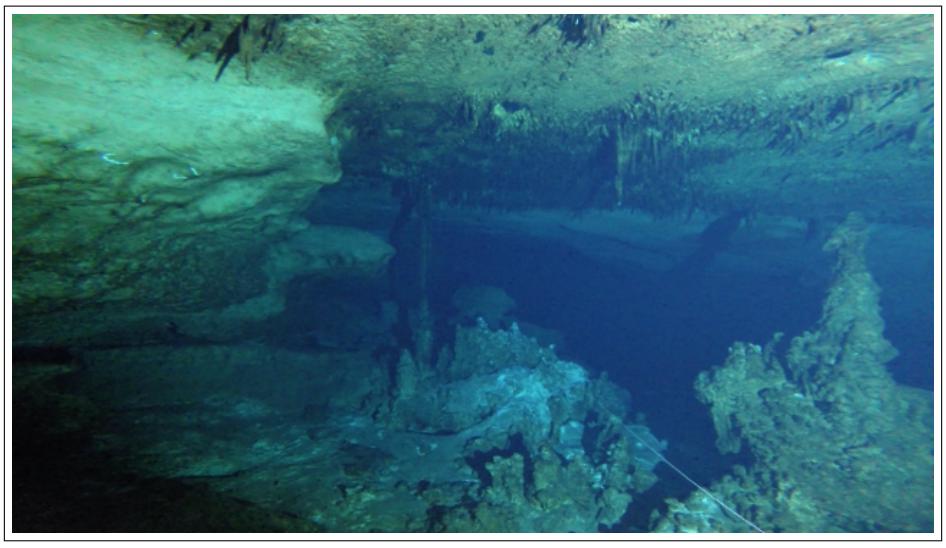


Figure 1.2: Typical scene from an underwater cave.

The importance of underwater cave mapping spans several fields. First, it is crucial in monitoring and tracking groundwater flows in karstic aquifers. According to Ford and Williams [28], 25% of the world's population relies on karst water resources. Our work is motivated by the Woodville Karst Plain (WKP) which is a geomorphic region that extends from Central Leon County around the "Big Bend" of Florida [52]. Due to the significance of WKP, the Woodville Karst Plain Project (WKPP) has explored more than 34 miles of cave systems in Florida since 1987 [11], proving the cave system to be the longest in USA [37]. This region is an important source of drinking water and is also a sensitive and vulnerable ecosystem. There is much to learn from studying the dynamics of the water flowing through these caves. Volumetric modeling of these caves will give researchers a better perspective about their size, structure, and connectivity. These models have even greater importance than simply enhancing the mapping. Understanding the volume of the conduits and how that volume increases and decreases over space is a critical component to characterizing the volume of flow through the conduit system. Current measurements are limited to point-flow velocities of the cave metering system and a cross-sectional volume at that particular point. The proposed approach results in 3-D reconstructions which will give researchers the above described capabilities. Furthermore, volumetric models will be incredibly helpful for those involved with environmental and agricultural studies throughout the area, and once perfected this technology could help map other subterranean water systems, as well as any 3-D environment that is difficult to map. The Woodville Karst Plain area is sensitive to seawater intrusions which threaten the agriculture and the availability of drinking water; for more details see the recent work by Zexuan et al. [104]. Second, detailed 3-D representations of underwater caves will provide insights to the hydrogeological processes that formed the caves. Finally, because several cave systems contain historical records dating to the prehistoric times, producing accurate maps will be valuable to underwater archaeologists.

1.3 Sensors for Underwater SLAM

SLAM depends on the robot's on-board exteroceptive sensing of the environment for observations of landmarks which can be used to estimate the current pose of the robot. LiDAR (Light Detection And Ranging), RADAR (RAdio Detection And Ranging), SONAR (SOund Navigation And Ranging), or camera are the commonly used sensors for solving the SLAM problem. LiDAR, RADAR, and SONAR directly measure the 3D structure to build the map of environment, whereas cameras cannot recover the range information from the measurements. Combining camera measurements with other sensors, e.g., Inertial Measurement Unit, RGB-D camera, Time-Of-Flight (ToF) etc, or stereo camera can recover the depth information. While SLAM can be addressed using many different sensors and their combinations depending on indoor, outdoor, and underwater for large scale or small scale environments, in this section we describe the main sensors for underwater SLAM.

1.3.1 Acoustic Sensor based Underwater Navigation

Sonar based underwater SLAM and navigation systems have been exploited for many years. Most of the underwater navigation algorithms [55, 53, 89, 44, 76] are based on acoustic sensors such as Doppler Velocity Log (DVL), Ultra-Short Baseline (USBL), and sonar. Nevertheless, collecting data using DVL, sonar, and USBL while diving is expensive and sometimes not suitable in challenging underwater environment, e.g., cave. Corke et al. [14] compared acoustic and visual methods for underwater localization, showing the viability of using visual methods underwater in some scenarios. Folkesson et al. [27] used a blazed array sonar for real-time feature tracking. A feature re-acquisition system with a low-cost sonar and navigation sensors were described [23]. Below we provide a list of acoustic sensors used for underwater SLAM.

- **Doppler Velocity Log**. A Doppler Velocity Log (DVL) is a sonar system that emits sound bursts along beams angled downward in various directions and measures the frequency shift of the echoes of these signals. As the DVL is aboard a moving vehicle, returning echoes carry a change in pitch known as the Doppler Effect. Combining these measurements provides the estimation of the velocity with respect to static objects, such as the sea-floor or the surface.

- **Ultra-Short Baseline** An Ultra-Short Baseline (USBL) is an underwater acoustic positioning system which consists of an array of acoustic transceivers mounted on static object such as ship, and a transponder placed on an Autonomous Underwater Vehicle (AUV) or a Remotely Operated Vehicle (ROV). These two units work together to communicate the vehicle position relative to the static object. The larger the spacing of acoustic array elements, the higher the positioning accuracy of the USBL. USBL is suitable to find objects on known GPS locations, examining the seabed, and record the exact location for these findings.

- **Sonar**. Sonar determines the distance and direction of underwater objects by acoustic means. Sound waves emitted by or reflected from the object are detected by it and analyzed for calculating range information. Some major categories of Sonars are imaging sonar, multibeam sonar, mechanical scanning profiling sonar, echo-sounder, and side-scanner sonar. Contrary to vision, Sonar measurements are not affected by turbidity or light and color attenuation, hence making it a suitable complement of camera.

Robotic exploration of underwater caves is in its infancy. Visual odometry (VO) is a challenging problem there due to the lack of natural light illumination and dynamic obstacles in addition to the underwater vision constraints i.e. light and color attenuation. There are not many works for mapping and localization in an under-

water cave. One of the first attempts was to explore a Cenote, a vertical shaft filled with water [34], by the vehicle DEPTHX (DEep Phreatic THermal eXplorer) [91] designed by Stone Aerospace [90], equipped with LiDAR and sonar. More recently, Mallios et al. demonstrated the first results of an Autonomous Underwater Vehicle (AUV) performing limited penetration inside a cave [57]. The main sensor used for SLAM was a horizontally mounted scanning sonar. A robotic fish was proposed for discovering underwater cave entrances based on vision and performing visual servoing, with experiments restricted to a swimming pool [12]. More recently, Sunfish [75] – an underwater SLAM system using a multibeam sonar, an underwater dead-reckoning system based on a fiber-optic gyroscope (FOG) IMU, acoustic DVL, and pressure-depth sensors – has been developed for autonomous cave exploration.

1.3.2 Vision-based Underwater Navigation

Exploiting SLAM techniques in underwater environment is a difficult task due to the highly unstructured nature of the environment. However, camera is one of the cheapest, small, light-weight, and energy-efficient sensors which provides rich and versatile information about the environment. Salvi et al. [79] implemented a real-time Extended Kalman Filter based SLAM incorporating a sparsely distributed robust feature selection and 6-DOF pose estimation using only calibrated stereo cameras. Johnson et al. [45] proposed an idea to generate 3D model of the seafloor from stereo images. Beall et al. [7] presented an accurate 3D reconstruction on a large-scale underwater dataset by performing bundle adjustment over all cameras and a subset of features rather than using a traditional filtering technique. A stereo SLAM framework named *selective SLAM* (SSLAM) for autonomous underwater vehicle localization was proposed in [8].

1.3.3 Underwater navigation fusing Vision with other sensors

Vision is often combined with IMU and other sensors for improved estimation of pose for the complementary characteristics of these sensors. Visual-inertial SLAM systems were proposed in [79, 7, 42, 41, 100] for underwater reconstruction and navigation. Sáez et al. [78] proposed a 6DOF Entropy Minimization SLAM to create dense 3D visual maps of underwater environments using a dense 3D stereo-vision system and IMU; it is an offline method. Shkurti et al. [86] proposed a state estimation algorithm for underwater robot by combining information from monocular camera, IMU, and pressure sensor based on the multi-state constrained Kalman filter [62]. In the following, we discuss sensors which can be fused together with vision to aid state estimation.

- **Pressure Sensor** A pressure sensor measures water depth applied by both the atmosphere and water column above it. Ferrera et al. [24] proposed a tightly-coupled Visual-Inertial-Pressure SLAM for underwater.

- **Inertial Measurement Unit**. Inertial Measurement Units (IMU) are proprioceptive sensors composed of a 3-axes gyroscope that measures the angular velocity, and a 3-axes accelerometer that measures the linear acceleration of the platform to which it is rigidly connected. As gyroscope provides velocity measurements, robot orientation can be estimated by integrating these measurements. Similarly, from accelerometer measurements it is possible to recover the velocity and position of the robot by integration of the acceleration measurements. However, unfortunately, in addition to sensor noises, both gyroscope and accelerometer measurements are affected by slowly varying time-evolving biases – as such simple integration of high-rate IMU measurements results into pose estimates unreliable for long-term navigation. Furthermore, the accelerometer is subject to gravity and needed to be subtracted to compute motion.

With the recent advancements of hardware design and manufacturing, low-cost light-weight Micro Electro-Mechanical Systems (MEMS) IMUs have become extremely popular which enables to put them into any electronic system including mobile phone, allowing high-accuracy localization. While vision observes the external world, an IMU provides information of self-motion, which makes both sensors complementary. In addition, combining vision with IMU can solve the *scale* issue in monocular vision-based SLAM, as it can be used estimate the motion between camera frames. Gravity can also be estimated which makes two rotational DoF i.e., absolute pitch and roll observable – another advantage of integrating vision with IMU. However, in order to fuse IMU and vision, both sensors should be synchronized perfectly such that the measurements from both sensors are timestamped with the same clock without any offset.

1.4 Motivation

The underwater environment presents unique challenges to vision-based state estimation. In particular, suspended particulates, blurriness, and light and color attenuation result in features that are not as clearly defined as above water. Consequently, results from different vision-based state estimation packages show a significant number of outliers resulting in inaccurate estimate or even complete tracking loss. To illustrate these challenges we present next a comprehensive study of the performances of state-of-the-art open-source Visual and Visual-Inertial state estimation algorithms in underwater domain and draw out the scope of improvements by introducing acoustic and pressure sensor [69, 46].

1.4.1 Underwater Datasets

Most of the standard benchmark datasets for SLAM represent only a single scenario, such as a lab space (e.g. [93, 10]), or an urban environment (e.g. Kitti [35]), and

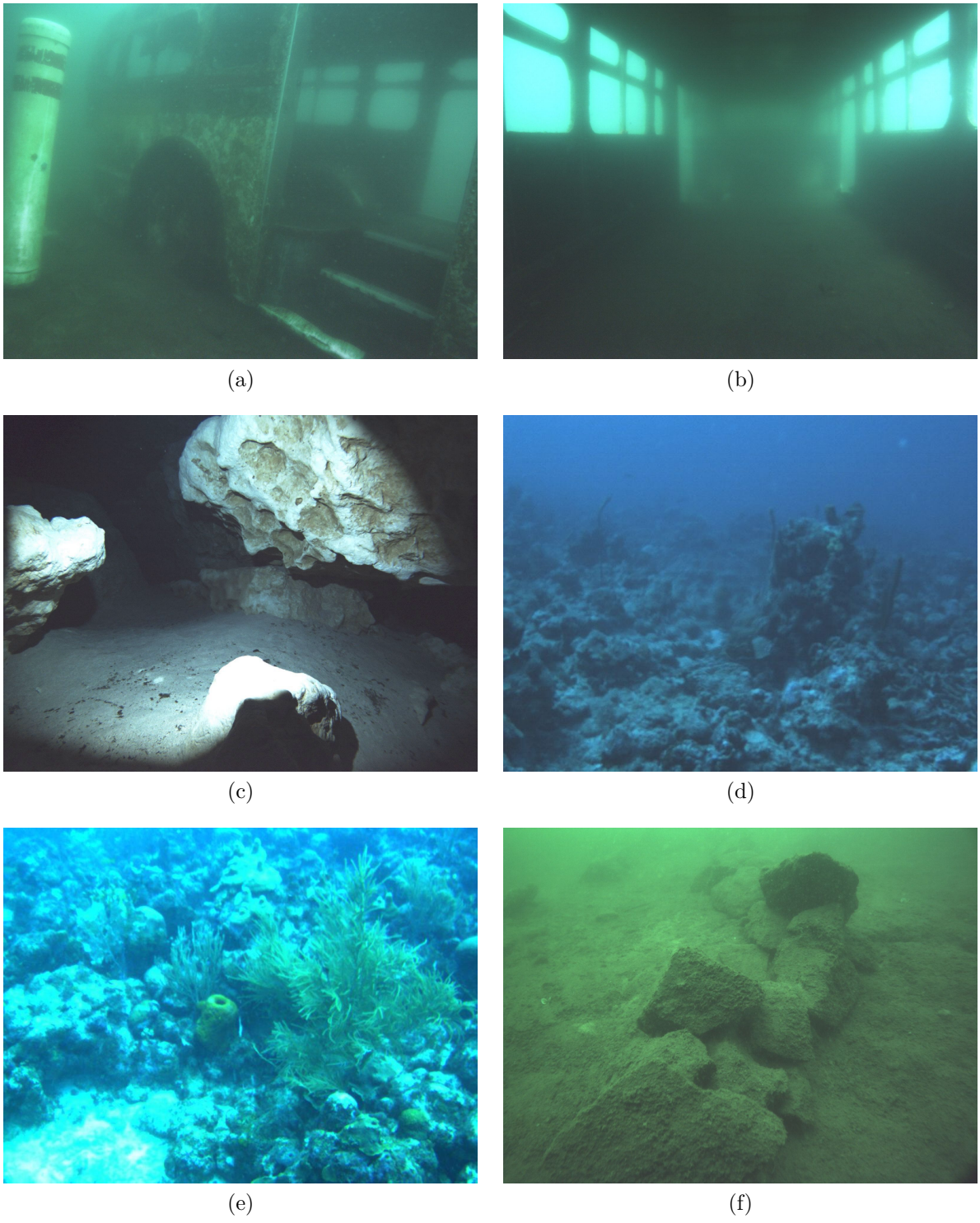


Figure 1.3: Sample images from the evaluated datasets. (a) UW sensor suite outside a sunken bus (NC); (b) UW sensor suite inside a sunken bus (NC); (c) UW sensor suite inside a cave (FL); (d) UW sensor suite mounted on a Diver Propulsion Vehicle (DPV) over a coral reef; (e) Aqua2 AUV over a coral reef; (f) AUV over a fake cemetery (SC).

with good visual quality. The limited nature of the public datasets is one of the primary motivations to evaluate these packages with datasets collected by our lab over the years in more challenging environments, such as underwater. We made these datasets publicly available (https://afrl.cse.sc.edu/afrl/resources/datasets/) for the research community and to present the challenges that an autonomous robot would need to face to operate safely and reliably in real-world conditions. We categorized the datasets according to the robotic platform used for collecting them:

- Underwater sensor suite operated by a diver around a sunken bus (Fantasy Lake, North Carolina) – see Fig. 1.3(a),(b) – and inside an underwater cave (Ginnie Springs, Florida); see Fig. 1.3(c). The custom-made underwater sensor suite is equipped with an IMU operating at 100 Hz (MicroStrain 3DM-GX15) and a stereo camera running at 15 fps, 1600×1200 (IDS UI-3251LE).
- Underwater sensor suite mounted on an Diver Propulsion Vehicle (DPV). Data collected over the coral reefs of Barbados; see Fig. 1.3(d).
- Aqua2 Autonomous Underwater Vehicle (AUV) over a coral reef (Fig. 1.3(e)) and an underwater structure (Lake Jocassee, South Carolina) (Fig. 1.3(f)), with the same setup as the underwater sensor suite.

1.4.2 Performances of state-of-the-art Visual and Visual-Inertial State Estimation Algorithms in Underwater

We have considered ten state estimation packages which are characterised by the following:

- *number of cameras*, e.g., monocular, stereo, or multiple cameras;
- the presence of an *IMU*;
- *direct* vs. *indirect* (feature-based) methods;

- *loosely* vs. *tightly-coupled* optimization when multiple sensors are used – e.g., camera and IMU;
- the presence of a *loop closing* mechanism.

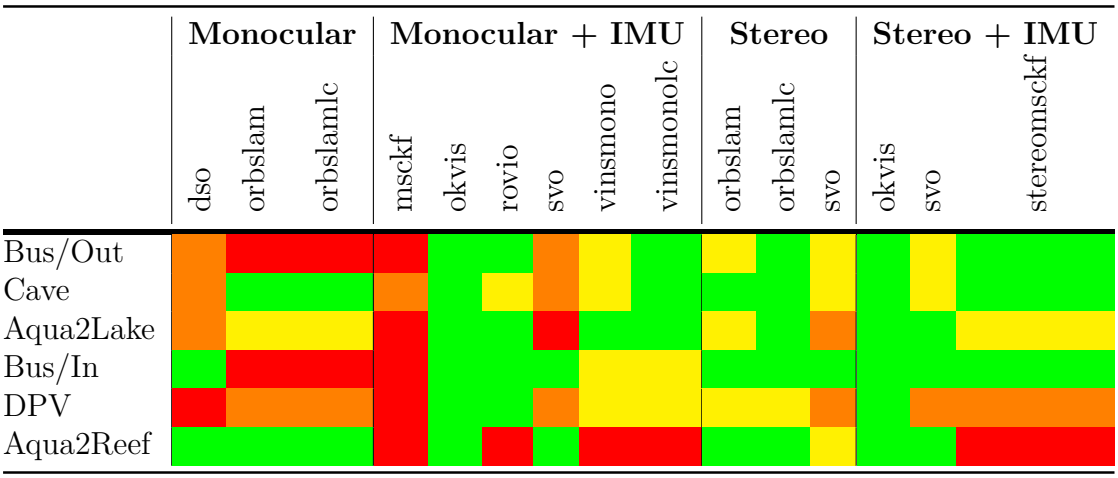
Table 1.1 lists the methods evaluated and their properties.

Table 1.1 Summary of characteristics for evaluated methods.

Method		**Camera**	**IMU**	**Indirect/ Direct**	**(L)oosely/ (T)ightly**	**Loop Closure**
LSD-SLAM	[21]	mono	no	direct	N/A	yes
DSO	[20]	mono	no	direct	N/A	no
SVO	[30]	multi	optional	semi-direct	N/A	no
ORB-SLAM2	[63]	mono, stereo	no	indirect	N/A	yes
REBiVO	[95]	mono	optional	indirect	L	no
Mono-MSCKF	[74]	mono	yes	indirect	T	no
Stereo-MSCKF	[94]	stereo	yes	indirect	T	no
ROVIO	[9]	multi	yes	direct	T	no
OKVIS	[56]	multi	yes	indirect	T	no
VINS-Mono	[67]	mono	yes	indirect	T	yes

The overall performance of the tested packages is shown in Table 1.2. LSD-SLAM [21], REBiVO [95], and Monocular SVO were unable to produce any consistent results, as such, they have been excluded. It is clear that *direct VO* approaches are not robust as there are often no discernible features and requires accurate photometric calibration. As such DSO and SVO, quite often fail to track the complete trajectory, however, they produced good dense 3D reconstruction for the tracked parts. Similar approaches that depend on the existence of a specific feature, such as edges, are not appropriate in underwater environments in general. Visual-inertial systems – e.g., OKVIS, VINS-Mono, and ROVIO – performed better in comparison to pure vision-based methods. Overall, as expected, stereo performed better than monocular, the presence of loop closure enabled the VO/VIO packages showed reduced drift in the trajectory, and the presence of inertial data aided to produce both the accurate the scale estimation along with poses.

Table 1.2 Performance of the different open source packages. Datasets: UW sensor suite outside a sunken bus (Bus/Out); UW sensor suite inside a cave (Cave); Aqua2 (AUV) over a fake cemetery (Aqua2Lake) at Lake Jocassee; UW sensor suite inside a sunken bus (Bus/In); UW sensor suite mounted on a Diver Propulsion Vehicle over a coral reef (DPV); Aqua2 AUV over a coral reef (Aqua2Reef). Qualitative analysis: the color chart legend is: red–failure; orange–partial failure; yellow–partial success; green-success.



1.5 Contributions

This dissertation focuses on real-time, robust, and accurate state estimation in challenging environments for autonomous mobile robots deployed in real world conditions. A keyframe-based tightly-coupled formulation of an underwater SLAM system combining multiple sensor information has been designed and developed. In particular, we made the following contributions:

A robust SLAM system combining Sonar, Visual, Inertial and water Depth information. We propose a tightly-coupled keyframe-based SLAM system fusing Sonar, Visual, Inertial and Depth information in a non-linear optimization-based framework for underwater domain. The underwater domain presents unique challenges in the quality of the visual data available; as such, augmenting the exteroceptive sensing with acoustic range data results in improved reconstructions of the underwater structures. Depth data from water pressure measurement bounds the localization error. The proposed tightly-coupled formulation considers all correlations

amongst these four sensors which is the key for high-precision localization. Moreover, to ensure real-time operation, a bounded sliding window of states are maintained for optimization through marginalization of past measurements and states while incorporating the prior in optimization. To address drift and loss of localization – one of the main problems affecting other packages in underwater domain – a robust initialization method to refine scale using depth measurements, a fast pre-processing step to enhance the image quality, and a real-time loop-closing and relocalization method using bag of words (BoW) have been provided. To validate the robustness and accuracy of our approach, we deployed an autonomous underwater vehicle (AUV) Aqua2 running our method on-board. Further datasets were collected with a custom-made underwater sensor suite both on hand-held mode while diving and deploying with a Diver Propulsion Vehicle (DPV). Experimental results from underwater wrecks, an underwater cave, a fake underwater cemetery, over coral reefs, and a submerged bus demonstrate the performance of our approach.

A contour-based reconstruction of underwater environment. Another contribution is a contour-based real-time reconstruction of an underwater environment using Sonar, Visual, Inertial, and Depth data. A central observation guiding this work is the fact that visual features, such as shadows, occlusion edges, and the boundaries illuminated by the artificial video light – are all located at the floor, ceiling, and walls of the underwater structures. As such, in addition to the tracked visual features and sonar features produced by the SLAM system during local mapping and local tracking, the proposed method utilizes visual features on these well defined edges on the boundaries for the 3D reconstruction of the surroundings. Experimental result in an underwater cave validates the method providing an improved reconstruction with a dense point cloud.

Chapter 2

Background and Related Work

Visual odometry (VO) is the problem of estimating the camera poses from a set of images sharing a common field-of-view (FOV). Please refer to [31, 32] for the fundamentals and practices in VO. While VO recovers frame-to-frame motion estimation, visual SLAM (V-SLAM) refers a more complete, accurate, and consistent system integrating a *loop-closure* mechanism and possibly a global optimization step to obtain a consistent map. There are two main approaches for solving visual SLAM: feature-based and direct methods. In this chapter, we revise the SLAM pipeline and describe the state-of-the-art visual SLAM systems based on feature-based and direct methods along with some intermediate categories.

2.1 Feature-based (Indirect) Visual SLAM Methods

Feature-based methods pre-process images to find keypoints which are distinctive interest points in a scene with high *intensity gradient* and establish correspondences, then optimize the *geometric error*. Ideally, these keypoints can be reliably and repeatedly detected in subsequent images of the same scene under variations of illumination conditions and different viewpoints. A descriptor is a vector of binary or real values, which describes the image patch around an interest point. Together an interest point and its descriptor is usually called a *feature*. Feature matching across images using binary descriptors – for example, BRISK (Binary Robust Invariant Scalable Keypoints), BRIEF (Binary Robust Independent Elementary Features), ORB (Oriented FAST and rotated BRIEF) – becomes very fast as it requires only descriptor

comparison for a similarity measure. A typical VO pipeline requires the following steps:

- feature detection
- feature matching
- motion estimation of calibrated camera via 2D-to-2D and 3D-to-2D (Perspective from N Points (PnP)) correspondences
- local optimization (windowed bundle adjustment)

Feature-based optimization computes the camera poses and the 3D landmarks by minimizing the *reprojection error*. Given a correspondence between a 3D point in world coordinates frame $\mathbf{P}_W$ and a 2D keypoint $\mathbf{p}_C$ in a monocular camera coordinate frame, the reprojection error, $\mathbf{e}_{proj}$ is defined as:

$$\mathbf{e}_{proj} = \mathbf{p}_C - \mathbf{h}_m({}_C\mathbf{R}_W\mathbf{P}_W + {}_C\mathbf{t}_W) \tag{2.1}$$

where $\mathbf{h}_m(\cdot)$ is the camera projection model, ${}_C\mathbf{R}_W \in SO(3)$ and ${}_C\mathbf{t}_W \in \mathbb{R}^3$ are the rotation and translation respectively from world to camera coordinates.

2.1.1 Monocular and Stereo Visual Odometry/SLAM systems

The literature presents many vision-based state estimation techniques, which use either *monocular* or *stereo* cameras including, MonoSLAM [17], PTAM [48], and ORB-SLAM [63]. To avoid scale ambiguity in monocular system, stereo camera pairs are used. Oskiper et al. [66] proposed a real-time VO using two pairs of backward and forward looking stereo cameras and an IMU in GPS denied environments. Howard [43] presented a real-time stereo VO for autonomous ground vehicles. This approach is based on *inlier detection*— i.e., using a rigidity constraint on the 3D location of features before computing the motion estimate between frames. Konolige et al. [49]

presented a real-time large scale VO on rough outdoor terrain integrating stereo images with IMU measurements. Kitt et al. [47] presented a visual odometry based only on stereo images using the trifocal geometry between image triples and a RANSAC based outlier rejection scheme. Their method requires only a known camera geometry where no rectification is needed for the images. Badino et al. [6] proposed a new technique for improved motion estimation by using the whole history of tracked features for real-time stereo VO.

PTAM (Parallel Tracking And Mapping) [48] is the first keyframe-based monocular SLAM algorithm able to handle Bundle Adjustment in real-time. The key idea is to split the tracking and mapping, thus offloading the accurate refinement using expensive batch optimization (bundle adjustment) to a background thread. This allows the tracking thread to obtain feature tracking and pose estimation in real-time, even if the bundle adjustment takes longer. The proposed system was specifically designed for a small AR workspace without any prior knowledge of the scene. MonoSLAM [17] is a monocular vision based real-time SLAM approach which includes an *active* approach to mapping and measurement, a general motion model for smooth camera movement, and solutions for monocular feature initialization and feature orientation estimation.

Loop closure – the capability of recognizing a place that was seen before – is an important component to mitigate the drift of the state estimate in a sliding window and marginalization-based framework. Currently ORB-SLAM [63] and its extension with IMU [64] is one of the most reliable and complete vision-based SLAM systems with loop-closing and relocalization capabilities. ORB-SLAM is build on the main ideas of PTAM, the place recognition work of Gálvez-López and Tardós based on bag-of-words (BoW) method, and the loop closing of Strasdat et al. [38].

2.1.2 Visual-Inertial Odometry/SLAM

To improve the pose estimate, vision is augmented with IMU for their complementary characteristics, commonly known as Visual-Inertial Navigation Systems (VINS). *Filtering* and *optimization* are the two approaches to optimally fuse IMU measurements and camera images to provide motion estimation.

Mourikis and Roumeliotis [62] developed one of the earliest successful real-time tightly coupled VINS algorithms based on Extended Kalman Filter (EKF), known as the Multi-State Constraint Kalman Filter (MSCKF). In particular, instead of adding features to the state vector, MSCKF performs nonlinear-triangulation of landmarks from a window of camera poses over time and obtains motion constraints that are used for EKF update. While this operation reduces the computational complexity by removing the need of co-estimating potentially hundreds and thousands of visual features, it also prevents the re-linearization of the non-linear feature measurements at later times, thus deteriorating accuracy and performance. Stereo-MSCKF [94], an extension of MSCKF, uses the Observability Constrained EKF (OC-EKF) [40], which does not heavily depend on an accurate initial estimation. Also, the camera poses in the state vector can be represented with respect to the inertial frame instead of the latest IMU frame so that the uncertainty of the existing camera states in the state vector is not affected by the uncertainty of the latest IMU state during the propagation step. As a result, Stereo-MSCKF can initialize well enough even without a perfect stand still period. It uses the first 200 IMU measurements for initialization and is recommended to not have fast motion during this period. REBiVO [95] (Realtime Edge Based Inertial Visual Odometry) is another filtering based approach which has been specifically designed for Micro Aerial Vehicles (MAV). This approach first processes the images to detect edges to track and map.

The other spectrum of methods optimizes the sensor states, possibly within a sliding window, formulating the problem as a *graph optimization* problem. In contrast to

filtering-based methods, batch optimization methods solve a non-linear least-squares (bundle adjustment [96]) problem over a set of visual and inertial measurements, allowing for the reduction of highly non-linear error through iterative re-linearization but have high computational cost. For feature-based visual-inertial systems, as in OKVIS [56] and Visual-Inertial ORB-SLAM [64], the optimization function includes the IMU error term and the reprojection error. The *frontend* tracking mechanism maintains a local map of features in a marginalization window which are never used again once out of the window. VINS-Mono [67] uses a similar approach and maintains a minimum number of features for each image and existing features are tracked by Kanade-Lucas-Tomasi (KLT) sparse optical flow algorithm in the local window. Delmerico and Scaramuzza [18] did a comprehensive comparison specifically monitoring resource usage by the different methods. While KLT sparse features allows VINS-Mono to run in real-time on low-cost embedded systems, it often results into tracking failure in challenging environments, e.g., underwater environments with low visibility. In addition, for loop detection additional features and their descriptors are needed to be computed for keyframes.

2.2 Direct Visual SLAM

Direct methods compare raw pixel intensities in the image and optimize the *photometric error*. Given the 2D coordinates of a pixel $\mathbf{p} = (u, v)^T$ and the intensity of that pixel in the *reference* image $\mathbf{I}_{ref}(\mathbf{p})$. Let $\mathbf{I}'(\mathbf{p}')$ be the *current* image after the camera motion. In direct method based tracking, an increment of the camera motion parameters $\Delta\boldsymbol{\theta} \in \mathbb{R}^d$ is calculated by minimizing the photometric error, defined as:

$$\Delta\boldsymbol{\theta}^* = \arg\min_{\Delta\boldsymbol{\theta}} \sum_{\mathbf{p}\in\Omega_{ref}} \left\| \mathbf{I}'(\mathbf{w}(\mathbf{p}; \boldsymbol{\theta} \boxplus \Delta\boldsymbol{\theta})) - \mathbf{I}_{ref}(\mathbf{p}) \right\|^2 \tag{2.2}$$

where Ω_{ref} is a subset of pixel coordinates of interest in the reference frame, $\mathbf{w}(\cdot)$ is a *warping* function that depends on the parameter vector we seek to estimate, and θ is an initial estimate of the motion parameters. At each iteration, the current estimate of parameters is updated (i.e. $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} \boxplus \Delta\boldsymbol{\theta}$), where $\boxplus$ generalizes the addition operator over the optimization manifold.

In direct methods, all information in the image can be exploited, however, they require a good initial guess and high frame rate with very limited frame-to-frame motion due to the photometric consistency assumption. Recently, direct methods – e.g., LSD-SLAM [21], DSO [20] – based SLAM systems show promising performance in 3-D reconstruction of large-scale map in real time while feature-based approach produces a very sparse map, as well as accurate pose estimation based on direct image alignment. In addition, in comparison to the indirect methods, direct approaches have potentials in textureless scenarios. However, theses methods are sensitive to the *brightness consistency* assumption which limits the baseline of the matches and in low visibility with small contrast environment like underwater, often result into tracking loss [46]. In addition, direct method suffers in presence of strong geometric noise, such as rolling shutter. For good reconstruction, they require perfect photometric calibration for modeling gain and exposure. DSO [20] shows an improvement in performance providing a full photometric calibration that accounts for lens attenuation, gamma correction, and known exposure times. In purely monocular vision based direct SLAM, like DSO, the initialization is slow and requires very small rotational change. As such, indirect methods are more widely used in practical applications due to its maturity and robustness. FAB-MAP [16, 15] is an appearance-based method to recognize places in a probabilistic framework.

LSD-SLAM is another well-known direct SLAM method to build large scale semi-dense maps for indoor and outdoor environments which operates in real time without any GPU acceleration. The photometric error is used to estimate motion between camera frames, while *inverse depth* estimation is calculated using small baseline stereo with fixed camera. ROVIO (Robust Visual Inertial Odometry) [9] employs an Iterated Extended Kalman Filter to tightly fuse IMU data with images from one or multiple cameras. The photometric error is derived from image patches that are used as landmark descriptors and is included as residual for the update step.

2.3 Semi-direct Method

Semi-Direct Visual Odometry is an in-between approach which utilizes both the direct and feature-based method for motion estimation, e.g., SVO [30] relies on direct method for tracking and triangulating pixels with high image gradients and a feature-based method for jointly optimizing structure and motion. It uses the IMU prior for image alignment and can be generalized to multi-camera systems. SVO [30] is able to track the camera pose over long trajectories, even in parts with few features by creating a depth map of a scene. SVO has been extended to support stereo cameras and inertial measurements, however, it is sensitive to large rotational change in the scene and does not support map reuse or loop-closure.

2.4 Structure-from-Motion (SfM) and Multiview Stereo (MVS)

Structure-from-Motion (SfM) is the problem of recovering relative camera poses as well as the 3-D structure from a set of calibrated or uncalibrated camera images. SfM from unstructured collections of photographs to build the 3-D model of the scene has been addressed in different solutions, *Bundler* [88] and *VisualSFM* [101]. They provided algorithmic analysis to improve computational complexity and per-

formance accuracy. *COLMAP* [82, 83] proposes a SfM algorithm to improve on the state-of-the-art incremental SfM methods for 3D reconstruction from unordered image collections. They provide scene graph augmentation, a next best view selection mechanism, and an efficient triangulation and Bundle Adjustment (BA) technique. COLMAP jointly estimates depth and surface normal, by leveraging photometric and geometric priors for pixel-wise view selection, and utilizes geometric consistency for simultaneous refinement. COLMAP outperforms state-of-the-art SfM system on benchmark datasets with a large number of photos from Internet with varying camera density and distributed over large area.

Multiview Stereo (MVS) is another well known method for reconstruction. Merrell et al. [59] presented a *viewpoint-based* approach to fuse multiple stereo depth maps for reconstructing 3-D shape from video. By decoupling the processing into two stages, they are able to run large-scale reconstruction in real-time using a GPU implementation for efficient computation. The computational power available on board of the robot is very limited, making the deployment of bundle adjustment based methods not feasible on the robot.

2.5 Non-linear Least-Squares Problems

Bundle adjustment (BA) is the problem of refining a visual reconstruction to produce *jointly optimal* 3D structure and viewing parameters, i.e., camera pose and/or calibration estimates. BA and graph optimization problems are modeled as non-linear least-squares optimization problems. Given a system described by a set of m observation functions $[f_1(\mathbf{x}), ...f_m((\mathbf{x})]^T$, the state vector $\mathbf{x} \in \mathbb{R}^n$ of n dimensional vector of variables, and $\mathbf{z}_i$ be a measurement of the state $\mathbf{x}$ with $\hat{\mathbf{z}}_i = f_i(\mathbf{x})$ be a function which maps $\mathbf{x}$ to a *predicted* measurement $\hat{\mathbf{z}}_i$, the error $\mathbf{e}_i$ is defined as the difference between the predicted and actual measurement:

$$\mathbf{e}_i(\mathbf{x}) = \mathbf{z}_i - \hat{\mathbf{z}}_i = \mathbf{z}_i - f_i(\mathbf{x}) \tag{2.3}$$

The error is assumed to be *normally* distributed and has a *zero* mean Gaussian error with information matrix $\mathbf{\Omega}_i$. The non-linear least-squares approach estimates the parameter values that minimizes the weighted Sum of Squared Error (SSE) cost or error function. The squared error depends only on the states and can be defined as:

$$e_i(\mathbf{x}) = \mathbf{e}_i(\mathbf{x})^T \mathbf{\Omega}_i \mathbf{e}_i(\mathbf{x}) \tag{2.4}$$

The goal of optimization problem is to find the state $\mathbf{x}^*$ which minimizes the error given all the measurements:

$$\begin{aligned} \mathbf{x}^* &= \arg\min_{\mathbf{x}} \sum_i e_i(\mathbf{x}) \\ &= \arg\min_{\mathbf{x}} \sum_i \mathbf{e}_i(\mathbf{x})^T \mathbf{\Omega}_i \mathbf{e}_i(\mathbf{x}) \end{aligned} \tag{2.5}$$

The two main stream algorithms to solve non-linear least-squares problems are *Line Search* (e.g., gradient descent, Newton's method, and Quasi-Newton method) and *Trust Region* (Levenberg-Marquardt, and Dogleg). The assumptions are that a "good" initial guess of the state is available and the error functions are "smooth" in the neighborhood of the minima. Both of them perform repeatedly *iterative local linearization* with the following steps until convergence:

- Linearize the error terms around the current solution/initial guess by Taylor expansion:

  $$\mathbf{e}_i(\mathbf{x} + \Delta\mathbf{x}) \approx \mathbf{e}_i(\mathbf{x}) + \mathbf{J}_i(\mathbf{x})\Delta\mathbf{x} \tag{2.6}$$

  where $\Delta\mathbf{x}$ is the correction to the state vector $\mathbf{x}$ and $\mathbf{J}_i$ is the Jacobian (first-order partial derivatives) computed at the current estimate of the state vector.

- Compute the first derivative of the squared error function and set it to zero. This step involves computing the global squared error in the neighborhood of the current solution $\mathbf{x}$ which takes the form:

$$\begin{aligned} F(\mathbf{x}+\Delta\mathbf{x}) \simeq \sum_i e_i(\mathbf{x}+\Delta\mathbf{x}) &= \mathbf{e}_i^T(\mathbf{x}+\Delta\mathbf{x})\boldsymbol{\Omega_i}\mathbf{e}_i(\mathbf{x}+\Delta\mathbf{x}) \\ &= c + 2\mathbf{b}^T\Delta\mathbf{x} + \Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x} \end{aligned} \tag{2.7}$$

where $\mathbf{H}$, $\mathbf{b}$, and c are computed as follows:

$$\begin{aligned} c &= \sum_i \mathbf{e}_i(\mathbf{x})^T\boldsymbol{\Omega}_i\mathbf{e}_i(\mathbf{x}) \\ \mathbf{b}^T &= \sum_i \mathbf{e}_i(\mathbf{x})^T\boldsymbol{\Omega}_i\mathbf{J}_i(\mathbf{x}) \\ \mathbf{H} &= \sum_i \mathbf{J}_i(\mathbf{x})^T\boldsymbol{\Omega}_i\mathbf{J}_i(\mathbf{x}) \end{aligned} \tag{2.8}$$

Taking the derivative of $F(\mathbf{x}+\Delta\mathbf{x})$ and setting it to zero gives the *normal* equation:

$$\begin{aligned} \frac{\partial F(\mathbf{x}+\Delta\mathbf{x})}{\partial\Delta\mathbf{x}} &\simeq 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x} \\ \mathbf{0} &= 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x} \\ \mathbf{H}\Delta\mathbf{x} &= -\mathbf{b} \end{aligned} \tag{2.9}$$

- Iteratively solve the normal equation to obtain the increment $\Delta\mathbf{x}^*$:

$$\Delta\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{b} \tag{2.10}$$

The linear system can be solved efficiently using Cholesky factorization, QR decomposition, or Conjugate Gradients (for large systems) without *matrix inversion*. These methods are much faster and numerically more accurate than explicit matrix inversion, particularly for sparse matrices.

- Update the new state which is hopefully closer to the minimum:

$$\mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}^* \tag{2.11}$$

Please refer to [96] which presents a comprehensive study on efficiently designing and solving a BA problem.

The state space for the BA system – containing a set of camera poses and point feature positions – can be parameterized in many different ways, which include the minimal representations for rotation, such as, Euler angles, unit quaternions, or modified Rodrigues parameters. However, all the minimal representations for the Special Orthogonal Group $SO(3)$ group have singularities or extra constraints on the parameters which present unnecessary challenges to the optimization process. For example, Euler angles suffer from the Gimbal lock singularity, unit quaternions must maintain the unit-length constraint, and rotation matrix must maintain orthogonality with positive unit determinant. The least-squares optimization methods either require the addition of extra equations or an explicit normalization step to enforce this constraint. The group $SO(3)$ also forms a smooth manifold. The tangent space of the manifold at the identity is denoted as $\mathfrak{so}(3)$ known as *Lie Algebra*. Similarly, Special Euclidean Group $SE(3)$ which describes the group of rigid motion in 3D, is defined as $SE(3) = \{(\mathbf{R}, \mathbf{t}) : \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3\}$, and its associated algebra is denoted as $\mathfrak{se}(3)$. These manifolds act as a real vector space *locally*, but can encode a more complex *global* topology, such as that of $SO(3)$ and $SE(3)$. Parameterizations of $SO(3)$ and $SE(3)$ manifolds for non–linear optimization is explained in details in [29].

Chapter 3

SVIN: An Underwater SLAM System using Sonar, Visual, Inertial, and Depth Sensor

3.1 Introduction

In recent years, many vision-based state estimation algorithms have been developed using monocular, stereo, or multi-camera system mostly for indoor and outdoor environments. Vision is often combined with Inertial Measurement Unit (IMU) for improved estimation of pose in challenging environments, termed as *Visual-Inertial Odometry* (VIO) or *Visual-Inertial SLAM* [64, 56, 67, 62, 94]. However, the underwater environment – e.g., see Fig. 3.1 – presents unique challenges to vision-based state estimation. As shown in previous studies [69, 46], it is not straightforward to deploy the available vision-based state estimation packages underwater. In particular, the low lighting conditions or complete absence of natural light, variation of illuminations, and lack of reliable visual features, make vision-based estimation very

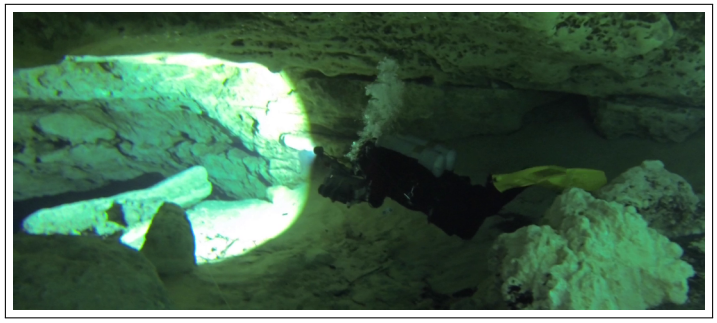


Figure 3.1: Underwater cave in Ginnie Springs, FL, where data have been collected using an underwater stereo rig.

difficult for underwater. As such, we propose a SLAM system combining acoustic and visual information by taking advantages of the complimentary sensing capabilities of these sensors in challenging and highly unstructured underwater environments.

In this chapter, we describe the formulation of a novel SLAM system, *SVIn*, targeted for underwater environments – e.g., wrecks and underwater caves – and easily adaptable for different sensor configuration: acoustic (mechanical scanning profiling sonar), visual (stereo camera), inertial (linear accelerations and angular velocities), and depth data. This makes our system versatile and applicable on-board of different sensor suites and underwater vehicles. The idea is that acoustic range data, though sparser, provide robust information about the presence of obstacles, where visual features reside; together with a more accurate estimate of scale. To fuse range data from sonar into the traditional VIO framework, we propose a new approach of taking a visual patch around each sonar point, and introduce extra constraints in the pose graph using the distance of the sonar point to the patch. The proposed method operates under the assumption that the visual-feature based patch is small enough and approximately coplanar with the sonar point. The resulting pose-graph consists of a combination of visual features and sonar features. In addition, we adopt the principle of *keyframe* based approaches to keep the graph sparse enough to enable real-time optimization. A particular challenge arises from the fact that the sonar features at an area are sensed some time after the visual features due to the different field of view of the two sensors.

In our recent work, [72], acoustic, visual, inertial, and water depth data is fused together to map different underwater structures by augmenting the visual-inertial state estimation package OKVIS [56]. This improves the trajectory estimate especially when there is varying visibility underwater, as sonar provides robust information about the presence of obstacles with accurate scale. However, in long trajectories, drifts could accumulate resulting in an erroneous trajectory. In [73], we extend

our work by including an image enhancement technique targeted to the underwater domain, introducing depth measurements in the optimization process, loop-closure capabilities, and a more robust initialization. These additions enable the proposed approach to robustly and accurately estimate the sensor's trajectory, where every other approach has shown incorrect trajectories or loss of localization.

To validate our proposed approach, first, we assess the performance of the proposed loop-closing method, by comparing it to other state-of-the-art systems on the EuRoC micro-aerial vehicle public dataset [10], disabling the fusion of sonar and depth measurements in our system. Second, we test the proposed full system on several different underwater datasets in a diverse set of conditions. More specifically, underwater data – consisting of visual, inertial, depth, and acoustic measurements – has been collected using a custom made sensor suite [71] from different locales; furthermore, data collected by an Aqua2 underwater vehicle [19] include visual, inertial, and depth measurements.

The results on the underwater datasets illustrate the loss of tracking and/or failure to maintain consistent scale for other state-of-the-art systems while our proposed method maintains correct scale without diverging. Experimental data were collected from the Ginnie ballroom cavern at High Springs, in Florida; a submerged bus in North Carolina; a fake underwater cemetery in Lake Jocassee in South Carolina; and an artificial shipwreck in Barbados. The custom-made sensor suite (described in Chapter 5) and Aqua2 robot employing stereo camera, mechanical scanning profiling sonar, IMU, and pressure sensor have been used for data collection.

The chapter is structured as follows. Section 3.2 - Section 3.3 presents the overview of proposed pipeline along with the approach developed for image preprocessing step and the notations, Section 3.4 describes the mathematical formulation and derivation of the tightly-coupled Sonar, stereo camera, inertial, and depth sensor integration. Section 3.5 and Section 3.6 present pose initialization, loop-closure, and relocalization

step respectively. We conclude this chapter with a discussion on lessons learned and directions of future work. Experimental results from a publicly available aerial dataset and a diverse set of challenging underwater environments will be presented in Chapter 6.

3.2 System Overview

This section describes the proposed system, SVIn2, depicted in Fig. 3.2. The full proposed state estimation system can operate on a robot that has stereo camera, IMU, sonar, and depth sensor – the last two can be also disabled to operate as a visual-inertial system.

Due to low visibility and dynamic obstacles, it is hard to find good features to track in underwater. In addition to the underwater vision constraints, e.g., light and color attenuation, vision-based systems also suffer from poor contrast. Hence, we augment the pipeline by adding an image pre-processing step to improve feature detection underwater. In particular, we use a *contrast limited adaptive histogram equalization* (CLAHE) filter in the *image pre-processing* step.

In the following, after defining the state, we describe the proposed initialization, sensor fusion optimization, loop closure and relocalization steps.

3.3 Notations and States

The full sensor suite is composed of the following coordinate frames: Camera (stereo), IMU, Sonar (acoustic), Depth (pressure), and World which are denoted as C, I, S, D, and W respectively. The transformation between two arbitrary coordinate frames X and Y is represented by a homogeneous transformation matrix ${}_X\mathbf{T}_Y = [{}_X\mathbf{R}_Y | {}_X\mathbf{p}_Y]$ where ${}_X\mathbf{R}_Y$ is rotation matrix with corresponding quaternion ${}_X\mathbf{q}_Y$ and ${}_X\mathbf{p}_Y$ is position vector.

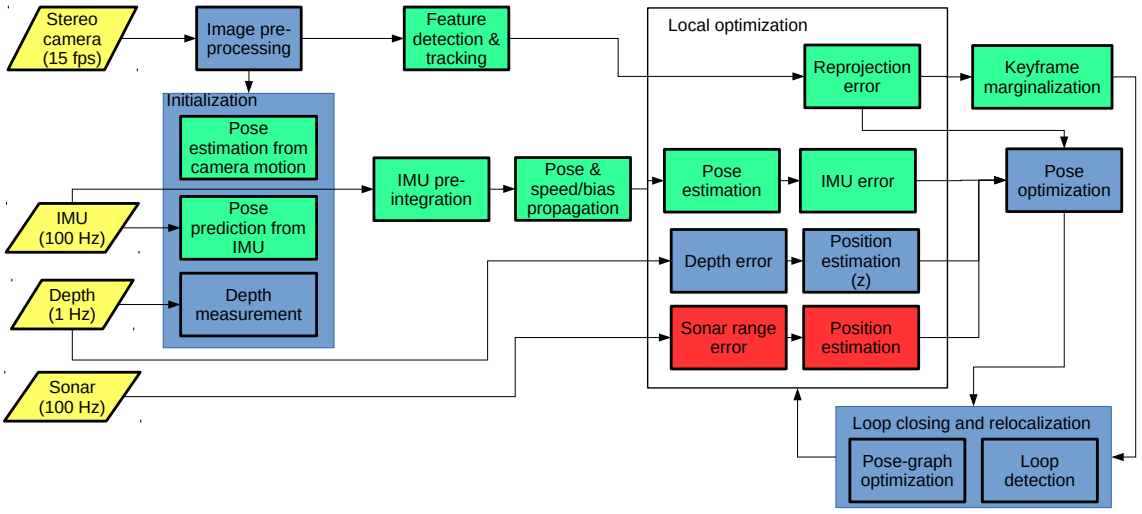


Figure 3.2: Block diagram of the proposed system, SVIn2; in yellow the sensor input, in green the components from OKVIS, in red the contribution from our work [72], and in blue the contributions in [73].

Let us now define the robot R state $\mathbf{x}_R$ that the system is estimating as:

$$\mathbf{x}_R \;=\; [{}_W\mathbf{p}_I^T, {}_W\mathbf{q}_I^T, {}_W\mathbf{v}_I^T, {\mathbf{b}_g}^T, {\mathbf{b}_a}^T]^T \in \mathbb{R}^3 \times SO(3) \times \mathbb{R}^9 \tag{3.1}$$

which contains the position ${}_W\mathbf{p}_I$, the attitude represented by the quaternion ${}_W\mathbf{q}_I$, the linear velocity ${}_W\mathbf{v}_I$, all expressed as the IMU reference frame I with respect to the world coordinate W; moreover, the state vector contains the gyroscopes and accelerometers bias $\mathbf{b}_g$ and $\mathbf{b}_a$.

The associated error-state vector is defined in minimal coordinates, while the perturbation takes place in the tangent space of the state manifold. The transformation from minimal coordinates to tangent space can be done using a bijective mapping [56, 29]:

$$\delta\boldsymbol{\chi}_R \;=\; [\delta\mathbf{p}^T, \delta\boldsymbol{\theta}^T, \delta\mathbf{v}^T, \delta{\mathbf{b}_g}^T, \delta{\mathbf{b}_a}^T]^T \in \mathbb{R}^{15} \tag{3.2}$$

The above represents the error for each component of the state vector with $\delta\boldsymbol{\theta} \in \mathbb{R}^3$ being the minimal perturbation for rotation (can be converted to its quaternion equivalent via exponential mapping).

3.4 Tightly-coupled Non-Linear Optimization with Sonar-Visual-Inertial-Depth (SVIND) Measurements

In this section, we propose to tightly fuse stereo vision, acoustic, inertial, and pressure measurements within our SLAM system. For the tightly-coupled non-linear optimization, we use the following cost function $J(\mathbf{x})$, which includes the reprojection error $\mathbf{e}_r$ and the IMU error $\mathbf{e}_s$ with the addition of the sonar error $\mathbf{e}_t$, and the depth error e_u:

$$\begin{aligned} J(\mathbf{x}) &= \sum_{i=1}^{2}\sum_{k=1}^{K}\sum_{j\in\mathcal{J}(i,k)} \mathbf{e}_r^{i,j,k^T}\mathbf{P}_r^k\mathbf{e}_r^{i,j,k} + \sum_{k=1}^{K-1}\mathbf{e}_s^{k^T}\mathbf{P}_s^k\mathbf{e}_s^k \\ &+ \sum_{k=1}^{K-1}\mathbf{e}_t^{k^T}\mathbf{P}_t^k\mathbf{e}_t^k + \sum_{k=1}^{K-1} e_u^{k^T}P_u^k e_u^k \end{aligned} \tag{3.3}$$

where i denotes the camera index – i.e., left ($i = 1$) or right ($i = 2$) camera in a stereo camera system with landmark index j observed in the k^{th} camera frame. $\mathbf{P}_r^k$, $\mathbf{P}_s^k$, $\mathbf{P}_t^k$, and P_u^k represent the information matrix (weights) of visual landmarks, IMU, sonar range, and depth measurement for the k^{th} frame respectively.

For completeness, we discuss the formulation in details for each of the error terms. The reprojection error describes the difference between a keypoint measurement in camera coordinate frame C and the corresponding landmark projection according to the projection model. The IMU error term combines all accelerometer and gyroscope measurements by *IMU pre-integration* [29] between successive camera measurements and represents the *pose*, *speed and bias* error between the prediction based on previous and current states. Both reprojection error and IMU error term follow the formulation by Leutenegger et al. [56]. Sonar error limits error in the trajectory using the *range* constraint between acoustics measurement and visual feature patch as well as provides additional points for the 3D reconstruction. At last, the depth error bounds the error in the gravity direction. All the error terms are added in the Google's *Ceres Solver* – the non-linear optimization framework – [2] to estimate the robot state in real-time.

3.4.1 IMU Error Term Formulation

An Inertial Measurements Unit (IMU) provides accelerometer and gyroscope readings – integrating them leads to a dead-reckoning positioning system which drifts fast with time. By fusing the dead-reckoning with absolute positioning readings (for example, vision), drifts can be avoided. Below we present the detailed formulation of non-linear IMU kinematics and bias model, more specifically, formulation of IMU kinematic model, linearized error-state model for orientation and velocity, and IMU measurement error.

A. IMU Kinematic Model

We employ a simple model that relates the raw gyroscope measurements, $\boldsymbol{\omega}_m$ and raw accelerometer measurements, $\mathbf{a}_m$ from IMU to the real angular velocity $\boldsymbol{\omega}$ and real linear acceleration respectively as

$$\begin{aligned} \boldsymbol{\omega}_m(t) &= {}_I\boldsymbol{\omega}(t) + \mathbf{b}_g(t) + \mathbf{n}_g(t) \\ \mathbf{a}_m(t) &= {}_I\mathbf{a}(t) + \mathbf{b}_a(t) + {}_I\mathbf{R}_W(t){}_W\mathbf{g} + \mathbf{n}_a(t) \end{aligned} \tag{3.4}$$

In the above equation, the IMU measurements are taken in its local frame, i.e, I, which accounts for the gravity ${}_W\mathbf{g}$, gyroscope bias $\mathbf{b}_g$, acceleration bias $\mathbf{b}_a$, and additive noise. The additive noise both in acceleration and gyroscope are assumed to be Gaussian white noise with characteristics $\mathbf{n}_a \sim \mathcal{N}(\mathbf{0}_{3\times1}, \boldsymbol{\sigma}_a^2.\mathbf{I}_{3\times3})$, $\mathbf{n}_g \sim \mathcal{N}(\mathbf{0}_{3\times1}, \boldsymbol{\sigma}_g^2.\mathbf{I}_{3\times3})$ respectively where we assume that the noise is equal in all three spatial direction for simplification. The gyro and accelerometer bias are non-static and simulated as random walk process with characteristics $\mathbf{n}_{bg} \sim \mathcal{N}(\mathbf{0}_{3\times1}, \boldsymbol{\sigma}_{bg}^2.\mathbf{I}_{3\times3})$, $\mathbf{n}_{ba} \sim \mathcal{N}(\mathbf{0}_{3\times1}, \boldsymbol{\sigma}_{ba}^2.\mathbf{I}_{3\times3})$. Following the formulation from [56], the accelerometer bias is modeled as a bounded random walk with time constant $\tau > 0$ whereas gyro bias is modeled as random walk.

The bias driving noise, i.e., $\mathbf{n}_{bg}$ and $\mathbf{n}_{ba}$ corresponds to the process noise, whereas the rate noise, i.e., $\mathbf{n}_b$ and $\mathbf{n}_a$ corresponds to the measurement noise.

In the following, we define the differential equations that describe IMU kinematics combined with bias models as:

$$\begin{aligned}
{}_I\dot{\mathbf{q}}_W(t) &= \frac{1}{2}\mathbf{\Omega}(\boldsymbol{\omega}_m(t) - \mathbf{b}_g(t) - \mathbf{n}_g(t)){}_I\mathbf{q}_W(t) \\
\dot{\mathbf{b}}_g(t) &= \mathbf{n}_{bg}(t) \\
{}_W\dot{\mathbf{v}}_I(t) &= {}_W\mathbf{R}_I(t)(\boldsymbol{a}_m(t) - \mathbf{b}_a(t) - \mathbf{n}_a(t)) - {}_W\mathbf{g} \\
\dot{\mathbf{b}}_a(t) &= -\frac{1}{\tau}\mathbf{b}_a(t) + \mathbf{n}_{ba}(t) \\
{}_W\dot{\mathbf{p}}_I(t) &= {}_W\mathbf{v}_I(t)
\end{aligned} \tag{3.5}$$

where the matrix $\mathbf{\Omega}$ is defined as:

$$\mathbf{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}, \lfloor\boldsymbol{\omega}\rfloor_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \in \mathfrak{so}(3)$$

The discrete evolution of the IMU state at time $t + \Delta t$ is obtained by integrating Eq. (3.6) as follows:

$$\begin{aligned}
{}_W\mathbf{p}_I(t+\Delta t) &= {}_W\mathbf{p}_I(t) + {}_W\mathbf{v}_I(t)\Delta t - \frac{1}{2}{}_W\mathbf{g}\Delta t^2 + {}_W\mathbf{R}_I(t)(\boldsymbol{a}_m(t) - \mathbf{b}_a(t) - \mathbf{n}_a(t))\Delta t^2 \\
{}_W\mathbf{v}_I(t+\Delta t) &= {}_W\mathbf{v}_I(t) - {}_W\mathbf{g}\Delta t + {}_W\mathbf{R}_I(t)(\boldsymbol{a}_m(t) - \mathbf{b}_a(t) - \mathbf{n}_a(t))\Delta t \\
{}_W\mathbf{q}_I(t+\Delta t) &= {}_W\mathbf{q}_I(t) \otimes \frac{1}{2}\mathbf{\Omega}(\boldsymbol{\omega}_m(t) - \mathbf{b}_g(t) - \mathbf{n}_g(t)){}_I\mathbf{q}_W(t)\Delta t
\end{aligned} \tag{3.6}$$

The motion between two consecutive keyframes k and $k+1$ in time interval $\Delta t_{k,k+1} \in [t_k, t_{k+1}]$ can be defined in terms of the preintegration terms, $\boldsymbol{\alpha}_{I_k}^{k+1}$, $\boldsymbol{\beta}_{I_k}^{k+1}$, and $\boldsymbol{\gamma}_{I_k}^{k+1}$ as follows:

$$\begin{aligned}
{}_I\mathbf{R}_W^k \cdot {}_W\mathbf{p}_I^{k+1} &= {}_I\mathbf{R}_W^k({}_W\mathbf{p}_I^k + {}_W\mathbf{v}_I^k\Delta t_{k,k+1} - \frac{1}{2}{}_W\mathbf{g}\Delta t_{k,k+1}^2) + \boldsymbol{\alpha}_{I_k}^{k+1} \\
{}_I\mathbf{R}_W^k \cdot {}_W\mathbf{v}_I^{k+1} &= {}_I\mathbf{R}_W^k({}_W\mathbf{v}_I^k - {}_W\mathbf{g}\Delta t_{k,k+1}) + \boldsymbol{\beta}_{I_k}^{k+1} \\
{}_I\mathbf{q}_W^k \otimes {}_W\mathbf{q}_I^{k+1} &= \boldsymbol{\gamma}_{I_k}^{k+1}
\end{aligned} \tag{3.7}$$

where

$$\begin{aligned}
\boldsymbol{\alpha}_{I_k}^{k+1} &= \int\int_{t\in[t_k,t_{k+1}]} \mathbf{R}_{I_k}^t(\boldsymbol{a}_m^t - \mathbf{b}_a^t - \mathbf{n}_a^t)dt^2 \\
\boldsymbol{\beta}_{I_k}^{k+1} &= \int_{t\in[t_k,t_{k+1}]} \mathbf{R}_{I_k}^t(\boldsymbol{a}_m^t - \mathbf{b}_a^t - \mathbf{n}_a^t)dt \\
\boldsymbol{\gamma}_{I_k}^{k+1} &= \int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_m^t - \mathbf{b}_g^t - \mathbf{n}_g^t)\boldsymbol{\gamma}_{I_k}^t dt
\end{aligned} \tag{3.8}$$

Taking the expectation of the above equation (Eq. (3.6)) yields the prediction equations:

$$\begin{aligned}
{}_I\dot{\hat{\mathbf{q}}}_W(t) &= \frac{1}{2}\boldsymbol{\Omega}(\hat{\boldsymbol{\omega}}(t)){}_I\hat{\mathbf{q}}_W(t) \\
\dot{\hat{\mathbf{b}}}_g(t) &= \mathbf{0}_{3\times 1} \\
{}_W\dot{\hat{\mathbf{v}}}_I(t) &= {}_W\mathbf{R}_I(t)(\hat{\mathbf{a}}(t)) - {}_W\mathbf{g} \\
\dot{\hat{\mathbf{b}}}_a(t) &= \mathbf{0}_{3\times 1} \\
{}_W\dot{\hat{\mathbf{p}}}_I(t) &= {}_W\hat{\mathbf{v}}_I(t)
\end{aligned} \tag{3.9}$$

where

$$\begin{aligned}
\hat{\boldsymbol{\omega}}(t) &= \boldsymbol{\omega}_m(t) - \hat{\mathbf{b}}_g(t) \\
\hat{\mathbf{a}}(t) &= \mathbf{a}_m(t) - \hat{\mathbf{b}}_a(t)
\end{aligned} \tag{3.10}$$

Here the bias is constant over the integration interval, as such, the quaternion can be integrated using the zero-th or first order integrator, using $\hat{\boldsymbol{\omega}}$ and $\hat{\mathbf{a}}$ instead of $\boldsymbol{\omega}$ and $\mathbf{a}$.

B. Continuous Time Linearized Error-State Model

The goal is to determine the continuous time linearized error-state dynamics of the system, given as follows:

$$\delta\dot{\mathbf{p}} = \delta\mathbf{v} \tag{3.11}$$

$$\delta\dot{\boldsymbol{\theta}} = -\lfloor\boldsymbol{\omega}_m - \hat{\mathbf{b}}_g\rfloor_\times\delta\boldsymbol{\theta} - \delta\mathbf{b}_g - \mathbf{n}_g \tag{3.12}$$

$$\delta\dot{\mathbf{v}} = -\hat{\mathbf{R}}\lfloor\mathbf{a}_m - \hat{\mathbf{b}}_a\rfloor_\times\delta\boldsymbol{\theta} - \hat{\mathbf{R}}\delta\mathbf{b}_a - \hat{\mathbf{R}}\mathbf{n}_a \tag{3.13}$$

$$\delta\dot{\mathbf{b}}_g = \dot{\mathbf{b}}_g - \dot{\hat{\mathbf{b}}}_g = \mathbf{n}_{bg} \tag{3.14}$$

$$\delta\dot{\mathbf{b}}_a = \dot{\mathbf{b}}_a - \dot{\hat{\mathbf{b}}}_a = -\frac{1}{\tau}\delta\mathbf{b}_a + \mathbf{n}_{ba} \tag{3.15}$$

For this section, we dropped the coordinate frame subscripts for readability. The error-state dynamics for position (Eq. (3.11)), gyro bias (Eq. (3.14)), and accelerometer bias (Eq. (3.15)) are trivial and are derived from linear equations. For velocity (Eq. (3.12)), and orientation error (Eq. (3.13)) some non-trivial manipulations of the respective non-linear equations are required to obtain linearized error-state dynamics. In the following two sections, we provide the derivation for both of them.

B.1 Linear Orientation Error

In order to derive the continuous time linear orientation error $\delta\dot{\boldsymbol{\theta}}$, we will start with the definition of error quaternion $\delta\mathbf{q}$ which describes the small rotation that causes the true quaternion $\mathbf{q}$ and it's estimate $\hat{\mathbf{q}}$ to coincide.

$$\begin{aligned} \mathbf{q}(t) = \mathbf{q} &= \delta\mathbf{q}\otimes\hat{\mathbf{q}} \qquad |\frac{d}{dt} \\ \dot{\mathbf{q}} &= \delta\dot{\mathbf{q}}\otimes\hat{\mathbf{q}} + \delta\mathbf{q}\otimes\dot{\hat{\mathbf{q}}} \end{aligned} \tag{3.16}$$

Substituting the definition of quaternion time derivative $\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix}\boldsymbol{\omega}\\0\end{bmatrix}\otimes\mathbf{q} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\mathbf{q}$ in the above for $\dot{\mathbf{q}}$ and $\dot{\hat{\mathbf{q}}}$:

$$\begin{aligned}
\frac{1}{2}\begin{bmatrix}\boldsymbol{\omega}\\0\end{bmatrix}\otimes\mathbf{q} &= \delta\dot{\mathbf{q}}\otimes\hat{\mathbf{q}}+\delta\mathbf{q}\otimes(\frac{1}{1}\begin{bmatrix}\hat{\boldsymbol{\omega}}\\0\end{bmatrix}\otimes\hat{\mathbf{q}})\\
\delta\dot{\mathbf{q}}\otimes\hat{\mathbf{q}} &= \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\omega}\\0\end{bmatrix}\otimes\mathbf{q}-\delta\mathbf{q}\otimes\begin{bmatrix}\hat{\boldsymbol{\omega}}\\0\end{bmatrix}\otimes\hat{\mathbf{q}}\right) \qquad |\otimes\hat{\mathbf{q}}^{-1}\\
\delta\dot{\mathbf{q}} &= \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\omega}\\0\end{bmatrix}\otimes\delta\mathbf{q}-\delta\mathbf{q}\otimes\begin{bmatrix}\boldsymbol{\omega}\\0\end{bmatrix}\right) \qquad (3.17)
\end{aligned}$$

Combining the gyro model for $\boldsymbol{\omega}$ and the definition of $\hat{\boldsymbol{\omega}}$ yields.

$$\boldsymbol{\omega} = \hat{\boldsymbol{\omega}}-\delta\mathbf{b}_g-\mathbf{n}_g \qquad (3.18)$$

Substituting into the above leads to

$$\begin{aligned}
\delta\dot{\mathbf{q}} &= \frac{1}{2}\left(\begin{bmatrix}\hat{\boldsymbol{\omega}}\\0\end{bmatrix}\otimes\delta\mathbf{q}-\delta\mathbf{q}\otimes\begin{bmatrix}\hat{\boldsymbol{\omega}}\\0\end{bmatrix}\right)-\frac{1}{2}\begin{bmatrix}\delta\mathbf{b}_g+\mathbf{n}_g\\0\end{bmatrix}\otimes\delta\mathbf{q} \qquad | \quad \lfloor\delta\mathbf{q}\rfloor_\times\hat{\boldsymbol{\omega}}=-\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times\delta\mathbf{q}\\
&= \frac{1}{2}\left(\begin{bmatrix}-\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times & \hat{\boldsymbol{\omega}}\\-\hat{\boldsymbol{\omega}}^T & 0\end{bmatrix}.\,\delta\mathbf{q}-\begin{bmatrix}\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times & \hat{\boldsymbol{\omega}}\\-\hat{\boldsymbol{\omega}}^T & 0\end{bmatrix}.\,\delta\mathbf{q}\right)-\frac{1}{2}\begin{bmatrix}\delta\mathbf{b}_g+\mathbf{n}_g\\0\end{bmatrix}\otimes\delta\mathbf{q}\\
&= \frac{1}{2}\begin{bmatrix}-2\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times & \mathbf{0}_{3\times1}\\\mathbf{0}_{3\times1}{}^T & 0\end{bmatrix}.\,\delta\mathbf{q}-\frac{1}{2}\begin{bmatrix}\delta\mathbf{b}_g+\mathbf{n}_g\\0\end{bmatrix}\otimes\delta\mathbf{q}\\
&= \frac{1}{2}\begin{bmatrix}-2\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times & \mathbf{0}_{3\times1}\\-\mathbf{0}_{3\times1}{}^T & 0\end{bmatrix}.\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\theta}\\1\end{bmatrix}-\frac{1}{2}\begin{bmatrix}-\lfloor(\delta\mathbf{b}_g+\mathbf{n}_g)\rfloor\times & (\delta\mathbf{b}_g+\mathbf{n}_g)\\-(\delta\mathbf{b}_g+\mathbf{n}_g)^T & 0\end{bmatrix}.\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\theta}\\1\end{bmatrix}\\
&= \frac{1}{2}\begin{bmatrix}-2\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times & \mathbf{0}_{3\times1}\\\mathbf{0}_{3\times1}{}^T & 0\end{bmatrix}.\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\theta}\\1\end{bmatrix}-\frac{1}{2}\begin{bmatrix}\delta\mathbf{b}_g+\mathbf{n}_g\\0\end{bmatrix}-O(|\delta\mathbf{b}_g||\delta\boldsymbol{\theta}|,|\mathbf{n}_g||\delta\boldsymbol{\theta}|) \qquad (3.19)
\end{aligned}$$

Ignoring the second and higher order terms,

$$\delta\dot{\mathbf{q}}=\begin{bmatrix}\frac{1}{2}\delta\dot{\boldsymbol{\theta}}\\\dot{1}\end{bmatrix}=\begin{bmatrix}-\lfloor\hat{\boldsymbol{\omega}}\rfloor_\times\frac{1}{2}\delta\dot{\boldsymbol{\theta}}-\frac{1}{2}(\delta\mathbf{b}_g+\mathbf{n}_g)\\0\end{bmatrix} \qquad (3.20)$$

and finally recalling $\hat{\boldsymbol{\omega}}=\boldsymbol{\omega}_m-\hat{\mathbf{b}}_g$ (Eq. (3.8)) leads to the linearized dynamics of orientation error

$$\delta\dot{\boldsymbol{\theta}}=-\lfloor\boldsymbol{\omega}_m-\hat{\mathbf{b}}_g\rfloor_\times\delta\boldsymbol{\theta}-\delta\mathbf{b}_g-\mathbf{n}_g \qquad (3.21)$$

B.2 Linear Velocity Error

To determine the dynamics of continuous time linear velocity error, $\delta\dot{\mathbf{v}}$, we start with the definition of the error rotation for small-signal approximation of $\mathbf{R}(t)$:

$$\begin{aligned} \mathbf{R}(t) = \mathbf{R} = \hat{\mathbf{R}}\delta\mathbf{R} = &\quad \hat{\mathbf{R}}(\mathbf{I} + \lfloor\delta\boldsymbol{\theta}\rfloor_\times) + O(|\delta\boldsymbol{\theta}|) \\ \dot{\hat{\mathbf{v}}} = &\quad \hat{\mathbf{R}}\hat{\mathbf{a}} - {}_W\mathbf{g} \end{aligned} \tag{3.22}$$

where $\hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a$ (Eq. (3.8)) Combining the accelerometer model for $\mathbf{a}$ and the definition of $\hat{\mathbf{a}}$ yields:

$$\mathbf{a} = \mathbf{R}(\hat{\mathbf{a}} - \delta\mathbf{b}_a - \mathbf{n}_a) - {}_W\mathbf{g} \tag{3.23}$$

Now, we express $\dot{\mathbf{v}}$ in two different forms – left and right developments – as follows:

$$\begin{aligned} \dot{\mathbf{v}} = \dot{\hat{\mathbf{v}}} + \delta\dot{\mathbf{v}} &= \hat{\mathbf{R}}(\mathbf{I} + \lfloor\delta\boldsymbol{\theta}\rfloor_\times)(\hat{\mathbf{a}} - \delta\mathbf{b}_a - \mathbf{n}_a) - {}_W\mathbf{g} \\ \hat{\mathbf{R}}\hat{\mathbf{a}} - {}_W\mathbf{g} + \delta\dot{\mathbf{v}} &= \hat{\mathbf{R}}\hat{\mathbf{a}} - \hat{\mathbf{R}}\delta\mathbf{b}_a - \hat{\mathbf{R}}\mathbf{n}_a + \hat{\mathbf{R}}\lfloor\delta\boldsymbol{\theta}\rfloor_\times\hat{\mathbf{a}} - \hat{\mathbf{R}}\lfloor\delta\boldsymbol{\theta}\rfloor_\times(\delta\mathbf{b}_a + \mathbf{n}_a) - {}_W\mathbf{g} \\ &\quad | - \hat{\mathbf{R}}\hat{\mathbf{a}} + {}_W\mathbf{g} \\ \delta\dot{\mathbf{v}} &= \hat{\mathbf{R}}(-\delta\mathbf{b}_a - \mathbf{n}_a + \lfloor\delta\boldsymbol{\theta}\rfloor_\times\hat{\mathbf{a}}) - \hat{\mathbf{R}}\lfloor\delta\boldsymbol{\theta}\rfloor_\times(\delta\mathbf{b}_a + \mathbf{n}_a) + O(|\delta\mathbf{b}_a|, |\delta\boldsymbol{\theta}|) \end{aligned} \tag{3.24}$$

Ignoring the second-order and higher terms and applying $\lfloor\mathbf{a}\rfloor_\times\mathbf{b} = -\lfloor\mathbf{b}\rfloor_\times\mathbf{a}$

$$\delta\dot{\mathbf{v}} = \hat{\mathbf{R}}(-\lfloor\hat{\mathbf{a}}\rfloor_\times\delta\boldsymbol{\theta} - \delta\mathbf{b}_a - \mathbf{n}_a) \tag{3.25}$$

or finally recalling $\hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a$ (Eq. (3.8)),

$$\begin{aligned} \delta\dot{\mathbf{v}} &= \hat{\mathbf{R}}(-\lfloor\mathbf{a}_m - \hat{\mathbf{b}}_a\rfloor_\times\delta\boldsymbol{\theta} - \delta\mathbf{b}_a - \mathbf{n}_a) \\ &= -\hat{\mathbf{R}}\lfloor\mathbf{a}_m - \hat{\mathbf{b}}_a\rfloor_\times\delta\boldsymbol{\theta} - \hat{\mathbf{R}}\delta\mathbf{b}_a - \hat{\mathbf{R}}\mathbf{n}_a \end{aligned} \tag{3.26}$$

At last, the linearized model of the error state takes into the form:

$$\delta\dot{\boldsymbol{\chi}}_R = \begin{bmatrix} \delta\dot{\mathbf{p}} \\ \delta\dot{\boldsymbol{\theta}} \\ \delta\dot{\mathbf{v}} \\ \delta\dot{\mathbf{b}}_g \\ \delta\dot{\mathbf{b}}_a \end{bmatrix} \approx \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\lfloor\boldsymbol{\omega}_m - \hat{\mathbf{b}}_g\rfloor_\times & \mathbf{0} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\hat{\mathbf{R}}\lfloor\mathbf{a}_m - \hat{\mathbf{b}}_a\rfloor_\times & \mathbf{0} & -\hat{\mathbf{R}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\frac{1}{\tau}\mathbf{I} \end{bmatrix} \begin{bmatrix} \delta\mathbf{p} \\ \delta\boldsymbol{\theta} \\ \delta\mathbf{v} \\ \delta\mathbf{b}_g \\ \delta\mathbf{b}_a \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\hat{\mathbf{R}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{n}_g \\ \mathbf{n}_a \\ \mathbf{n}_{bg} \\ \mathbf{n}_{ba} \end{bmatrix}$$

$$= \mathbf{F}_c(\mathbf{x}_R)\delta\boldsymbol{\chi}_R + \mathbf{G}_c(\mathbf{x}_R)\mathbf{n} \tag{3.27}$$

C. Discrete Time Error State

Since the continuous-time system matrix $\mathbf{F}_c$ is constant over the integration period, discrete-time linearized error state transition matrix can be obtained by:

$$\begin{aligned} \mathbf{F}_d(\mathbf{x}_R, \Delta t) = &\quad exp(\mathbf{F}_c(\mathbf{x}_R)\Delta t) \\ \approx &\quad \mathbf{I}_{15} + \mathbf{F}_c(\mathbf{x}_R)\Delta t \end{aligned} \tag{3.28}$$

where Δt is the time difference. The covariance propagation equation can be computed recursively by first-order discrete-time covariance update:

$$\mathbf{W}_R^{p+1} = \mathbf{F}_d(\hat{\mathbf{x}}_R, \Delta t)\mathbf{W}_R^p\mathbf{F}_d(\hat{\mathbf{x}}_R, \Delta t)^T + \mathbf{G}_d(\hat{\mathbf{x}}_R)\mathbf{Q}_d\mathbf{G}_d(\hat{\mathbf{x}}_R)^T\Delta t \tag{3.29}$$

where $\mathbf{Q}_d = diag(\boldsymbol{\sigma}_g^2.\mathbf{I}_3, \boldsymbol{\sigma}_a^2.\mathbf{I}_3, \boldsymbol{\sigma}_{bg}^2.\mathbf{I}_3, \boldsymbol{\sigma}_{ba}^2.\mathbf{I}_3)$ is the diagonal covariance matrix containing all the noise densities of the respective processes.

D. IMU Measurement Error Formulation

We express the IMU error term $\mathbf{e}_s^k(\mathbf{x}_R^k, \mathbf{x}_R^{k+1}, \mathbf{z}_s^k)$ as a function of robot states at time steps k and $k+1$ (when the images are taken) and all the IMU measurements $\mathbf{z}_s^k$, containing gyro and accelerometer data in-between these time instances. We assume an approximate normal conditional probability density function f with zero mean and

variance $\mathbf{W}_s^k$, and the associated conditional covariance $\mathbf{Q}(\delta\hat{\mathbf{x}}_R^{k+1}|\mathbf{x}_R^k, \mathbf{z}_s^k)$ for given robot states at camera measurements k and $k+1$:

$$f(\mathbf{e}_s^k|\mathbf{x}_R^k, \mathbf{x}_R^{k+1}) \approx \mathcal{N}(\mathbf{0}, \mathbf{W}_s^k) \tag{3.30}$$

Using the prediction equations, we can now formulate the IMU error term as the following which is simply the difference between the *prediction* based on the previous state and the *actual* state:

$$\mathbf{e}_s^k(\mathbf{x}_R^k, \mathbf{x}_R^{k+1}, \mathbf{z}_s^k) = \begin{bmatrix} {}_I\mathbf{R}_W^k({}_W\hat{\mathbf{p}}_I^{k+1} - {}_W\mathbf{p}_I^{k+1}) \\ 2[{}_I\mathbf{q}_W^k \otimes {}_W\hat{\mathbf{q}}_I^{k+1} \otimes {}_W\mathbf{q}_I^{k+1^{-1}}]_{1:3} \\ {}_I\mathbf{R}_W^k({}_W\hat{\mathbf{v}}_I^{k+1} - {}_W\mathbf{v}_I^{k+1}) \\ \hat{\mathbf{b}}_g^{k+1} - \mathbf{b}_g^{k+1} \\ \hat{\mathbf{b}}_a^{k+1} - \mathbf{b}_a^{k+1} \end{bmatrix} \in \mathbb{R}^{15} \tag{3.31}$$

By applying the error propagation law, the associated information matrix is obtained by:

$$\mathbf{P}_s^k = \mathbf{W}_s^{k^{-1}} = \left(\frac{\partial \mathbf{e}_s^k}{\partial \delta\hat{\boldsymbol{\chi}}_R^{k+1}} \mathbf{Q}(\delta\hat{\mathbf{x}}_R^{k+1}|\mathbf{x}_R^k, \mathbf{z}_s^k) \frac{\partial \mathbf{e}_s^k}{\partial \delta\hat{\boldsymbol{\chi}}_R^{k+1}}^T \right)^{-1} \tag{3.32}$$

3.4.2 Reprojection Error Formulation

The camera observes the visual corners, which are used to update the motion estimate of the robot. The reprojection error is formulated as the difference between the feature observation $\mathbf{z}^{i,j,k}$ in image coordinate and the projection of the corresponding 3D point ${}_{C_i}\mathbf{p}^j$ on to the image plane where i ($i = 1$ *or* 2) is the camera index in stereo system, j is the 3D landmark index which is visible in k^{th} image frame:

$$\mathbf{e}_r^{i,j,k} = \mathbf{z}^{i,j,k} - \mathbf{h}_i({}_{C_i}\mathbf{p}^j) \tag{3.33}$$

here $\mathbf{h}_i(\cdot)$ denotes the camera projection model.

Assuming a perspective camera model, the feature measurement with zero-mean, and Gaussian white noise, $\mathbf{n}^{i,j,k}$ is defined as:

$$\mathbf{z}^{i,j,k} = \frac{1}{z_{i,j,k}} \begin{bmatrix} x^{i,j,k} \\ y^{i,j,k} \end{bmatrix} + \mathbf{n}^{i,j,k} \tag{3.34}$$

$$\begin{bmatrix} x^{i,j,k} \\ y^{i,j,k} \\ z^{i,j,k} \end{bmatrix} = {}_{C_i}\mathbf{p}^j \quad = \quad {}_{C_i}\mathbf{C}_I(\mathbf{q}) {}_I\mathbf{C}_W(\mathbf{q}^k)({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) + {}_{C_i}\mathbf{p}_I \tag{3.35}$$

The measurement Jacobian $\mathbf{H}_i$ is calculated as:

$$\mathbf{H}_i = \mathbf{H}_{proj} \; {}_{C_i}\mathbf{C}_I(\mathbf{q}) \; \begin{bmatrix} \mathbf{H}_\theta^k & \mathbf{0}_{3\times 9} & \mathbf{H}_{\mathbf{p}}^k \end{bmatrix} \tag{3.36}$$

where $\mathbf{H}_{proj}$, $\mathbf{H}_\theta$, and $\mathbf{H}_{\mathbf{p}}$ are the Jacobian of the projection $\mathbf{h}_i(.)$ into the i^{th} camera with respect to the landmark in the homogeneous coordinates, orientation, and translation respectively. Below we show the derivation of each of the them. Calculation of $\mathbf{H}_{proj}$ and $\mathbf{H}_{\mathbf{p}}$ are straight-forward and given by:

$$\mathbf{H}_{proj} = \frac{1}{\hat{z}^{i,j,k}} \begin{bmatrix} 1 & 0 & -\frac{\hat{x}^{i,j,k}}{\hat{z}^{i,j,k}} \\ 0 & 1 & -\frac{\hat{y}^{i,j,k}}{\hat{z}^{i,j,k}} \end{bmatrix} \tag{3.37}$$

$$\mathbf{H}_{\mathbf{p}}^k = -{}_I\mathbf{C}_W(\hat{\mathbf{q}}) \tag{3.38}$$

For deriving $\mathbf{H}_\theta$, recall the definition of error quaternion from Eq. (3.16), and expressing in corresponding rotation matrix

$$\begin{aligned} \mathbf{q} &= \delta\mathbf{q} \otimes \hat{\mathbf{q}} \\ {}_I\mathbf{C}_W(\mathbf{q}) &= {}_I\mathbf{C}_W(\delta\mathbf{q} \otimes \hat{\mathbf{q}}) \\ &= {}_I\mathbf{C}_I(\delta\mathbf{q}) \; . \; {}_I\mathbf{C}_W(\hat{\mathbf{q}}) \end{aligned} \tag{3.39}$$

Using the *additive* error model where $\mathbf{q} = \delta\mathbf{q} + \hat{\mathbf{q}}$:

$$\begin{aligned}
\mathbf{H}_\theta^k &= \left({}_I\mathbf{C}_W(\mathbf{q}^k) - {}_I\mathbf{C}_W(\hat{\mathbf{q}}^k)\right) \cdot ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) \\
&= \left({}_I\mathbf{C}_I(\delta\mathbf{q}^k) \cdot {}_I\mathbf{C}_W(\hat{\mathbf{q}}^k) - {}_I\mathbf{C}_W(\hat{\mathbf{q}}^k)\right) \cdot ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) \\
&= \left(({}_I\mathbf{C}_I(\delta\mathbf{q}^k) - \mathbf{I}_3)\ {}_I\mathbf{C}_W(\hat{\mathbf{q}}^k)\right) \cdot ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) \cdot ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) \\
&\quad \lfloor {}_I\mathbf{C}_W(\delta\mathbf{q}) \approx \mathbf{I}_3 - \lfloor\delta\boldsymbol{\theta}\rfloor_\times \\
&\approx -\lfloor\delta\boldsymbol{\theta}\rfloor_\times {}_I\mathbf{C}_W(\hat{\mathbf{q}}^k)\ ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I) \\
&= \lfloor\left({}_I\mathbf{C}_W(\hat{\mathbf{q}}^k)\ ({}_W\mathbf{p}^j - {}_W\mathbf{p}_I)\right)\rfloor_\times \delta\boldsymbol{\theta}
\end{aligned} \tag{3.40}$$

3.4.3 Sonar Error Term Formulation

The concept behind calculating the sonar range error is that, if the sonar detects any obstacle at some distance, it is more likely that the visual features would be located on the surface of that obstacle, and thus will be approximately at the same distance. The step involves computing a visual patch detected in close proximity of each sonar point to introduce an extra constraint, using the distance of the sonar point to the patch. Here, we assume that the visual-feature based patch is small enough and approximately coplanar with the sonar point.

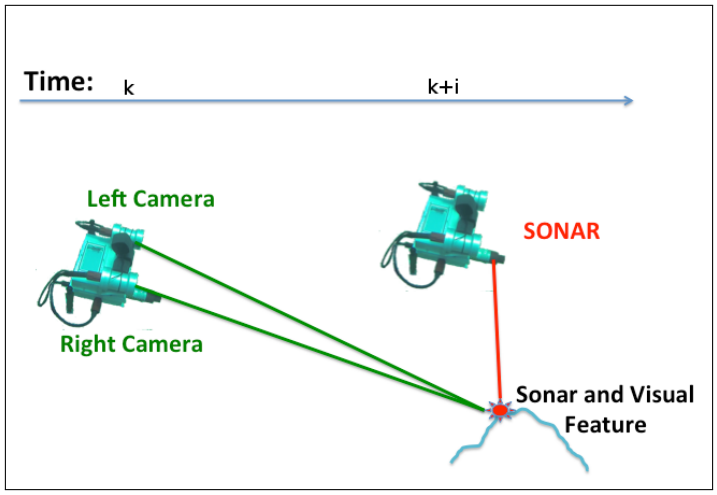


Figure 3.3: The relationship between sonar measurement and stereo camera features. A visual feature detected at time k is only detected by the sonar with a delay, at time $k+i$, where i depends on the speed the sensor is moving.

In the presented system, the sonar measurements are used to correct the robot *pose* estimate as well as to optimize the use of landmarks coming from both vision and sonar. Due to the low visibility of underwater environments when it is hard to find visual features, sonar provides features with accurate scale. A particular challenge is the temporal displacement between the two sensors, vision and sonar. Figure 3.3 illustrates the structure of the problem, at time k some features are detected by the stereo camera, it takes some time (until $k+i$) for the sonar to pass by these visual features and thus obtain a related measurement. To address the above challenge, visual features detected in close proximity to the sonar return are grouped together and used to construct a patch. The distance between the sonar and the visual patch is used as an additional constraint.

For computational efficiency, the sonar range correction only takes place when a new camera frame is added to the pose graph. As sonar has a faster measurement rate than the camera, only the nearest *range* to the robot *pose* in terms of timestamp is used to calculate a small patch from *visual landmarks* around the sonar landmark for that given *range* and *head_position*. Algorithm 1 shows how to calculate the *range error* $\mathbf{e}_t^k$ given the robot position ${}_W\mathbf{p}_I^k$ and the sonar measurement $\mathbf{z}_t^k$ at time k.

The sonar returns *range* r and *head_position* θ measurements which are used to obtain each sonar landmark ${}_W\boldsymbol{l} = [l_x, l_y, l_z, 1]$ in homogeneous coordinate by a simple geometric transformation in world coordinates:

$$ {}_W\boldsymbol{l} = ({}_W\mathbf{T}_I {}_I\mathbf{T}_S[\mathbf{I}_3 | r\cos(\theta), r\sin(\theta), 0]_S^T) \tag{3.41} $$

where ${}_W\mathbf{T}_I$ and ${}_I\mathbf{T}_S$ are the respective transformation matrices used to transform the sonar measurement from the sonar coordinates to the world coordinates. ${}_I\mathbf{T}_S$ represents the transformation from the sonar frame of reference to the IMU reference frame, and ${}_W\mathbf{T}_I$ represents the transformation from the inertial (IMU) frame to the world coordinates. Consequently, the sonar range prediction is calculated using the lines 2-9 of Algorithm 1:

Algorithm 1 SONAR Range Error Algorithm

Input: Estimation of robot position ${}_W\mathbf{p}_I^k$ at time k
Sonar measurement $\mathbf{z}_t^k = [r, \theta]$ at time k
List of current visual landmarks, $\mathcal{L}_v$
Distance threshold, T_d
Output: Range error e_t^k at time k

/*Compute sonar landmark in world coordinates*/
${}_W\boldsymbol{l} = ({}_W\mathbf{T}_{II}\mathbf{T}_S[\mathbf{I}_3|r\cos(\theta), r\sin(\theta), 0]_S^T)$
/*Create list of visual landmarks around sonar landmark*/
$\mathcal{L}_S = \emptyset$
for (every $\boldsymbol{l}_i$ in $\mathcal{L}_v$) **do**
/*Compute Euclidean distance from visual landmark to sonar landmark*/
  $d_S = \|{}_W l - l_i\|$
  if ($d_S < T_d$) **then**
    $\mathcal{L}_S = \mathcal{L}_S \cup \boldsymbol{l}_i$
  end if
end for
$\hat{r} = \left\|{}_W\hat{\mathbf{p}}_I^k - \text{mean}(\mathcal{L}_S)\right\|$
return $r - \hat{r}$

$$\hat{r} = \|{}_W\hat{\mathbf{p}}_I - \text{mean}(\mathcal{L}_S)\|\,, \tag{3.42}$$

where $\mathcal{L}_S$ is the subset of visual landmarks around the sonar landmark. As mentioned above, the concept behind calculating the *range error* is that, if the sonar detects any obstacles at some distance, it is more likely that the visual features would be located on the surface of that obstacle, and thus will be at approximately the same distance. Thus, the error term is the difference between the two distances. Note that we approximate the visual patch with the centroid (mean($\mathcal{L}_S$)), to filter out noise on the visual landmarks.

As such, given the sonar measurement $\mathbf{z}_t^k$, the error term $\mathbf{e}_t^k({}_W\mathbf{p}_I^k, \mathbf{z}_t^k)$ is based on the difference between those two distances which is used to correct the position ${}_W\mathbf{p}_I^k$. We assume an approximate normal conditional probability density function f with zero *mean* and $\mathbf{W}_t^k$ *variance*, and the conditional covariance $\mathbf{Q}(\delta\hat{\mathbf{p}^k}|\mathbf{z}_t^k)$, updated iteratively as new sensor measurements are integrated:

$$f(\mathbf{e}_t^k|_W\mathbf{p}_I^k) \approx \mathcal{N}(\mathbf{0}, \mathbf{W}_t^k) \tag{3.43}$$

The information matrix is:

$$\mathbf{P}_t^k = \mathbf{W}_t^{k-1} = \left(\frac{\partial \mathbf{e}_t^k}{\partial \delta \hat{\mathbf{p}^k}} \mathbf{Q}(\delta \hat{\mathbf{p}^k} | \mathbf{z}_t^k) \frac{\partial \mathbf{e}_t^k}{\partial \delta \hat{\mathbf{p}^k}}^T \right)^{-1} \tag{3.44}$$

The Jacobian can be derived by differentiating the expected *range* r measurement with respect to the robot position:

$$\frac{\partial \mathbf{e}_t^k}{\partial \delta \hat{\mathbf{p}^k}} = \left[\frac{-l_x + {}_W p_x}{r}, \frac{-l_y + {}_W p_y}{r}, \frac{-l_z + {}_W p_z}{r} \right] \tag{3.45}$$

3.4.4 Depth Error Term Formulation

The pressure sensor provides accurate depth measurements based on the water pressure. Depth values are extracted along the *gravity* direction which is aligned with the z of the world W – observable due to the tightly coupled IMU integration. The depth data at time k is given by:

$${}_W p_{z_D}{}^k = d^k - d^0 \tag{3.46}$$

More precisely, ${}_W p_{z_D}{}^k = (d^k - d^0) + init\_disp\_from\_IMU$ to account for the initial displacement along z axis from IMU, which is the main reference frame used by visual SLAM to track the sensor suite/robot.

With depth measurement z_u^k, the depth error term $e_u^k({}_W p_{z_I}{}^k, z_u^k)$ can be calculated as the difference between the robot position along the z direction and the depth data to correct the position of the robot. The error term can be defined as:

$$e_u^k({}_W p_{z_I}{}^k, z_u^k) = |{}_W p_{z_I}^k - {}_W\, p_{z_D}^k| \tag{3.47}$$

The information matrix calculation follows a similar approach as the sonar and the Jacobian is straight-forward to derive.

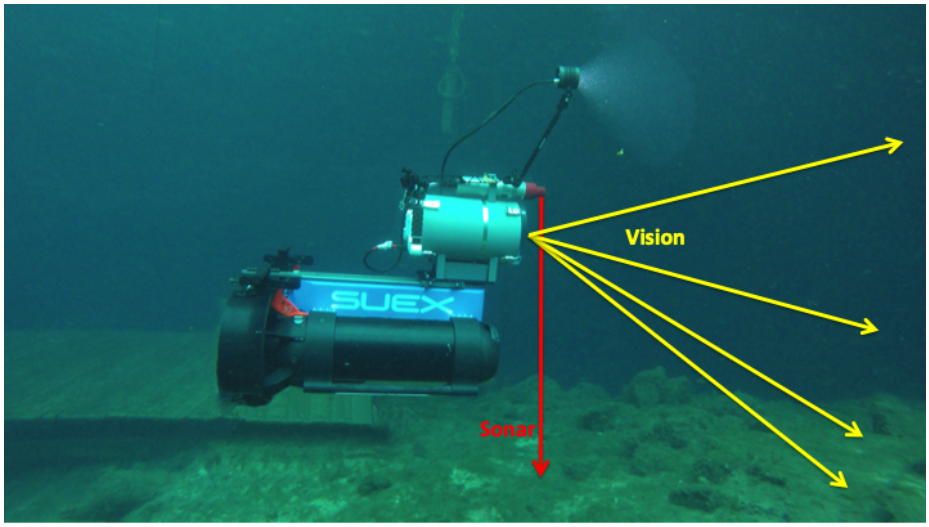


Figure 3.4: Custom made sensor suite mounted on a dual DPV. Sonar scans around the sensor while the cameras see in front.

3.5 Initialization: Two-step Scale Refinement

A robust and accurate initialization is required for the success of tightly-coupled non-linear systems, as described in [64, 67]. For underwater deployments, this becomes even more important as vision is often occluded as well as is negatively affected by the lack of features for tracking. Indeed, from our comparative study of visual-inertial based state estimation systems in [46], in underwater datasets, most of the state-of-the-art systems either fail to initialize or make wrong initialization resulting into divergence. Hence, we propose a robust initialization method using the sensory information from stereo camera, IMU, and depth for underwater state estimation. The reason behind using all these three sensors is to introduce constraints on *scale* to have a more accurate estimation on initialization. Note that no acoustic measurements have been used because the sonar range and visual features contain a temporal difference, which would not allow to have any match between acoustic and visual features, if the robot is not moving. This is due to the fact that the sonar scans on a plane over 360° *around the robot* and camera detects features *in front of the robot* [72]; see Fig. 3.4.

In particular, the proposed initialization works as follows. First, we make sure that the system only initializes when a minimum number of visual features are present to track (in our experiments 15 worked well). Second, the two-step refinement of the initial scale from the stereo vision takes place.

The depth sensor provides accurate depth measurements which are used to refine the initial scale factor from stereo camera. Including a scale factor s_1, the transformation between camera C and depth sensor D can be expressed as

$$_{W}p_{zD} = s_1 * {}_{W}p_{zC} + {}_{W}\mathbf{R}_{zCC}\mathbf{p}_D \tag{3.48}$$

For *keyframe* k, solving the above equation for s_1, provides the first refinement r_1 of the initial stereo scale ${}_{W}\mathbf{p}_{r1C}$, i.e.,

$$_{W}\mathbf{p}_{r1C} = s_1 * {}_{W}\mathbf{p}_C \tag{3.49}$$

In the second step, the refined measurement from stereo camera in Eq. (3.49) is aligned with the IMU pre-integral values. Similarly, the transformation between camera C and IMU I with scale factor s_2 can be expressed as:

$$_{W}\mathbf{p}_I = s_2 * {}_{W}\mathbf{p}_{r1C} + {}_{W}\mathbf{R}_{CC}\mathbf{p}_I \tag{3.50}$$

In addition to refining the scale, we also approximate initial *velocity* and *gravity* vector similar to the method described in [67]. Recalling Eq. (3.7) and Eq. (3.7), the state prediction from IMU measurements between two consecutive keyframes k and *k+1* in time interval $\Delta t_{k,k+1} \in [t_k, t_{k+1}]$ can be written as:

$$\begin{aligned}
{}_{W}\hat{\mathbf{p}}_I^{k+1} &= {}_{W}\mathbf{p}_I^k + {}_{W}\mathbf{v}_I^k\Delta t_{k,k+1} - \frac{1}{2}{}_{W}\mathbf{g}\Delta t_{k,k+1}^2 + {}_{W}\mathbf{R}_I^k\boldsymbol{\alpha}_{I_k}^{k+1} \\
{}_{W}\hat{\mathbf{v}}_I^{k+1} &= {}_{W}\mathbf{v}_I^k - {}_{W}\mathbf{g}\Delta t_{k,k+1} + {}_{W}\mathbf{R}_I^k\boldsymbol{\beta}_{I_k}^{k+1} \\
{}_{W}\hat{\mathbf{q}}_I^{k+1} &= \boldsymbol{\gamma}_{I_k}^{k+1}
\end{aligned} \tag{3.51}$$

Eq. (3.51) can be re-arranged with respect to $\boldsymbol{\alpha}_{I_i}^{i+1}$, $\boldsymbol{\beta}_{I_i}^{i+1}$ as follows:

$$\begin{aligned}\boldsymbol{\alpha}_{I_k}^{k+1} &= {}_I\mathbf{R}_W^k({}_W\hat{\mathbf{p}}_I^{k+1} - {}_W\mathbf{p}_I^k - {}_W\mathbf{v}_I^k\Delta t_{k,k+1} + \frac{1}{2}{}_W\mathbf{g}\Delta t_{k,k+1}{}^2) \\ \boldsymbol{\beta}_{I_k}^{k+1} &= {}_I\mathbf{R}_W^k({}_W\hat{\mathbf{v}}_I^{k+1} - {}_W\mathbf{v}_I^k + {}_W\mathbf{g}\Delta t_{k,k+1}) \end{aligned} \tag{3.52}$$

Substituting Eq. (3.50) into Eq. (3.52), we can estimate $\boldsymbol{\chi}_S = [\mathbf{v}_I^k, \mathbf{v}_I^{k+1}{}_{,W}\mathbf{g}, s_2]^T$ by solving the linear least square problem in the following form:

$$\min_{\boldsymbol{\chi}_S} \sum_{k\in K} \left\| \hat{\boldsymbol{z}}_{S_k}^{k+1} - \mathbf{H}_{S_k}^{k+1}\boldsymbol{\chi}_S \right\|^2 \tag{3.53}$$

where $\hat{\boldsymbol{z}}_{S_k}^{k+1} =$

$$\begin{bmatrix} \hat{\boldsymbol{\alpha}}_{I_k}^{k+1} - {}_I\mathbf{R}_W^k{}_W\mathbf{R}_C^{k+1}{}_C\mathbf{p}_I^{k+1} + {}_I\mathbf{R}_C^k{}_C\mathbf{p}_I^k \\ \hat{\boldsymbol{\beta}}_{I_k}^{k+1} \end{bmatrix}$$

and $\mathbf{H}_{S_k}^{k+1} =$

$$\begin{bmatrix} -\mathbf{I}\Delta t_{k,k+1} & \mathbf{0} & \frac{1}{2}{}_I\mathbf{R}_W^k\Delta t_{k,k+1}{}^2 & {}_I\mathbf{R}_W^k({}_W\mathbf{p}_{r1C}^{k+1} - {}_W\mathbf{p}_{r1C}^k) \\ -\mathbf{I} & {}_I\mathbf{R}_W^k{}_W\mathbf{R}_I^{k+1} & {}_I\mathbf{R}_W^k\Delta t_{k,k+1} & \mathbf{0} \end{bmatrix}$$

3.6 Loop-closing and Relocalization

In a sliding window and marginalization based optimization method, drift accumulates over time on the pose estimate. A global optimization and relocalization scheme is necessary to eliminate this drift and to achieve global consistency. We adapt DBoW2 [33], a bag of binary words (BoW) place recognition module, and augment OKVIS for loop detection and relocalization. For each keyframe, the descriptors for only the *keypoints* detected during the local tracking are used to build BoW database. No new features will be detected in the loop closure step.

A pose-graph is maintained to represent the connection between keyframes. In particular, a node represents a keyframe and an edge between two keyframes exists if

the matched keypoints ratio between them is more than 0.75. In practice, this results into a very sparse graph. With each new keyframe in the pose-graph, the loop-closing module searches for candidates in the bag of words database. A query for detecting loops to the BoW database only returns the candidates outside the current marginalization window and having greater than or equal to score than the neighbor keyframes of that node in the pose-graph. If loop is detected, the candidate with the highest score is retained and feature correspondences between the current keyframe in the local window and the loop candidate keyframe are obtained to establish connection between them. The pose-graph is consequently updated with loop information. A 2D-2D descriptor matching and a 3D-2D matching between the known landmark in the current window keyframe and loop candidate with outlier rejection by PnP RANSAC is performed to obtain the geometric validation.

When a loop is detected, the global relocalization module aligns the current keyframe pose in the local window with the pose of the loop keyframe in the pose-graph by sending back the drift in pose to the windowed sonar-visual-inertial-depth optimization thread. Also, an additional optimization step, similar to Eq. (3.3), is taken only with the matched landmarks with loop candidate for calculating the sonar error term and reprojection error; see Eq. (3.54).

$$J(\mathbf{x}) = \sum_{i=1}^{2}\sum_{k=1}^{K}\sum_{j\in Loop(i,k)} \mathbf{e}_r^{i,j,k^T}\mathbf{P}_r^k\mathbf{e}_r^{i,j,k} + \sum_{k=1}^{K-1}\mathbf{e}_t^{k^T}\mathbf{P}_t^k\mathbf{e}_t^k \tag{3.54}$$

After loop detection, a 6-DoF (position, $\mathbf{x_p}$ and rotation, $\mathbf{x_q}$) pose-graph optimization takes place to optimize over relative constraints between poses to correct drift. The relative transformation between two poses $\mathbf{T}_i$ and $\mathbf{T}_j$ for current keyframe in the current window i and keyframe j (either loop candidate keyframe or connected

keyframe) can be calculated from $\Delta\mathbf{T}_{ij} = \mathbf{T}_j{\mathbf{T}_i}^{-1}$. The error term, $\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}$ between keyframes i and j is formulated minimally in the tangent space:

$$\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}} = \Delta\mathbf{T}_{ij}\hat{\mathbf{T}}_i{\hat{\mathbf{T}}_j}^{-1} \tag{3.55}$$

where $\hat{(\cdot)}$ denotes the estimated values obtained from local sonar-visual-inertial-depth optimization. and the cost function to minimize is given by

$$J(\mathbf{x_p},\mathbf{x_q}) = \sum_{i,j}{\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}}^T\mathbf{P}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}} + \sum_{(i,j)\in Loop}\rho({\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}}^T\mathbf{P}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}\mathbf{e}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}) \tag{3.56}$$

where $\mathbf{P}^{i,j}_{\mathbf{x_p},\mathbf{x_q}}$ is the information matrix set to identity, as in [92], and ρ is the Huber loss function to potentially down-weight any incorrect loops.

3.7 Conclusions

In this chapter, we presented a state estimation system with robust initialization, sensor fusion of depth, sonar, visual, and inertial data, and loop closure capabilities. While the proposed system can also work out of the water, by disabling the sensors that are not applicable, our system is specifically targeted for underwater environments. Experimental results in a standard benchmark dataset and different underwater datasets demonstrate excellent performance (see Chapter 6).

The appearance of color underwater is different than above, including the color loss with depth. When most color shifts to blue, there is a loss of sharpness, which further degrades performance. This will be a venue for further research in the future, in order to investigate the effect of any color restoration to the state estimation process. It is worth noting that maintaining the proper attitude of the traversed trajectory and providing an estimate of the distance traveled will greatly enhance the autonomous capabilities of the vehicle [81]. Furthermore, accurately modeling the surrounding structures would enable Aqua2, as well as other vision based underwater vehicles to operate near, and through, a variety of underwater structures, such as caves, shipwrecks, and canyons.

Chapter 4

Contour based Reconstruction of Underwater Structures Using Sonar, Visual, Inertial, and Depth Sensors

4.1 Introduction

Underwater cave exploration is one of the most extreme adventures pursued by humans [22]. It is a dangerous activity with more than 600 fatalities, since the beginning of underwater cave exploration, that currently attracts many divers. Generating models of the connectivity between different underwater cave systems together with data on the depth, distribution, and size of the underwater chambers is extremely important for fresh water managements [13], environmental protection, and resource utilization [104]. In addition, caves provide valuable historical evidence as they present an undisturbed time capsule [1], and information about geological processes [50].

Before venturing beyond the light zone with autonomous robots, it is crucial to ensure that localization and mapping abilities have been developed and are adequately robust. Constructing a map of an underwater cave presents many challenges. First of all, vision underwater is plagued by limited visibility, color absorption, hazing, and lighting variations. Furthermore, the absence of natural light inside underwater caves makes localization and mapping more difficult; however, the use of an artificial light can be used to infer the structure [98]. The most common underwater mapping sensor is based on sonar, which, when mounted on a moving platform, requires a secondary

sensor to provide a common frame of reference for the range measurements collected over time. Furthermore, the majority of sonar sensors generate multiple returns in enclosed spaces making mapping caves extremely difficult.

In our earlier work, the cone of light perceived by a stereo camera was used to reconstruct offline the boundaries of a cave in Mexico [98]. No other sensor was available and the stereo-baseline of 0.03 m limited the accuracy of the reconstruction for objects further than a couple of meters. More recently, augmenting the visual-inertial state estimation package OKVIS [56], we fused visual and inertial data together with acoustic range measurements from a pencil beam sonar, which provide more reliable distance estimate of features. This allows a more robust and reliable state estimation [72]. One of the limitations is the granularity of the resulting 3D point cloud: only few keypoints are typically tracked, resulting in very sparse 3D point cloud, which cannot be directly used, for example, by an Autonomous Underwater Vehicle (AUV) to navigate and avoid obstacles. Applying a direct-based method, such as LSD-SLAM [21], is not straightforward, given the sharp changes in illumination in the underwater scene. A fundamental difference with most vision based estimation approaches is that in a cave environment, the light source is constantly moving thus generating shadows that also move. Consequently the majority of the strong features cannot be used for estimating the pose of the camera.

In this chapter, we propose a novel system that is able to track the state estimate and at the same time improve the 3-D reconstruction from visual edge based information in the cave boundaries.

In particular, the proposed approach for real-time reconstruction of the cave environment with medium density is based on an underwater SLAM system that combines acoustic (sonar range), visual (stereo camera), inertial (linear accelerations and angular velocities), and depth data to estimate the trajectory of the employed sensor suite. The inspiration for a denser point cloud comes from the following observation: visual

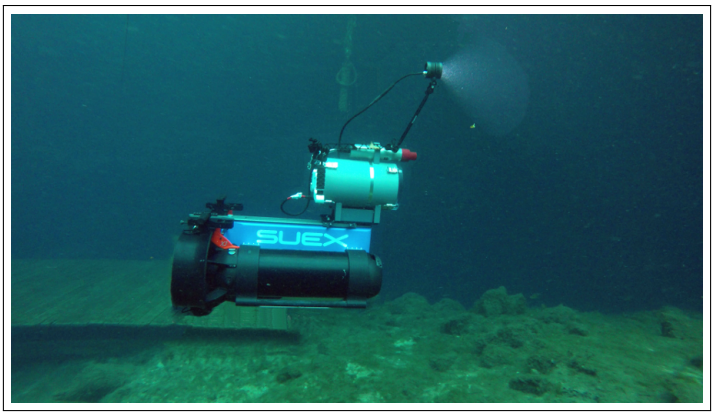


Figure 4.1: The stereo, inertial, depth, and acoustic sensor suite mounted on a dual diver propulsion vehicle (DPV) equipped with a flashlight, in front of the Blue Grotto cavern.

features on the boundaries created by shadows, occlusion edges, and the boundaries of the artificial illumination (video light) – are all located at the floor, ceiling, and walls of the cave. The point cloud resulting from such edges is then optimized in a local bundle adjustment, and can be used for providing a denser reconstruction, enabling the deployment of AUVs like Aqua2 [19] with advanced swimming gaits [58], navigating around obstacles without disturbing the sediment at the bottom. Experiments in caverns and caves validate the proposed approach.

The chapter is structured as follows. Section 4.2 describes the proposed method. Section 4.5 concludes the chapter. Experimental results are presented in Chapter 6 Section 6.5.

4.2 System Overview

The proposed system augments the approach described in [72, 73] in Chapter 3 to generate real-time a denser reconstruction of underwater structures exploiting the boundaries of the structure and the cone-of-light. The proposed system is depicted in Fig. 4.2. The target hardware system is composed of a stereo camera, a mechanical scanning profiling Sonar, an IMU, a pressure sensor, and an on-board computer – as

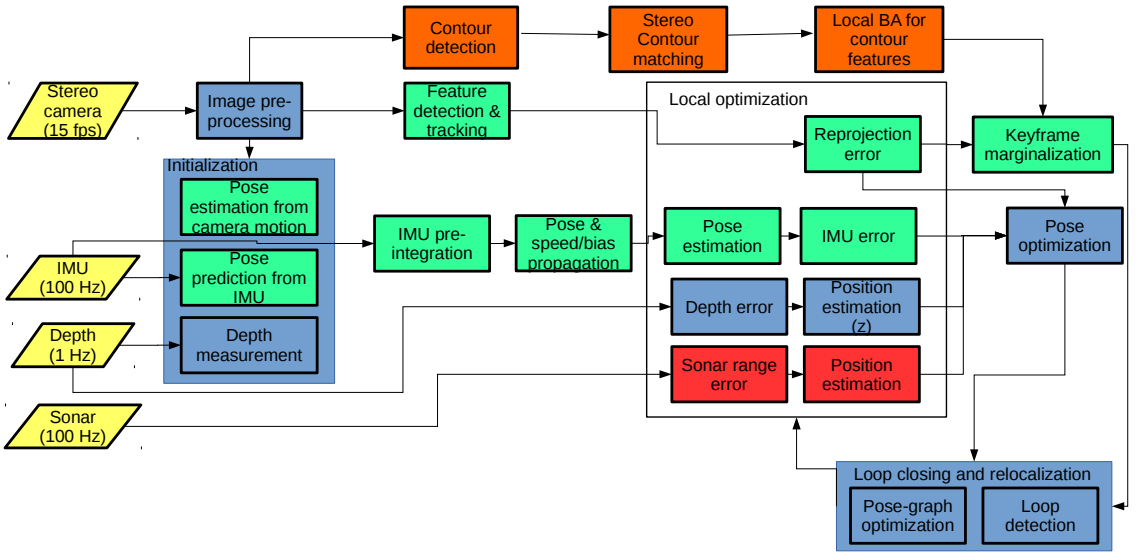


Figure 4.2: Block diagram of the proposed system; in yellow the sensor input with frequency from the custom-made sensor suite, in green the components from OKVIS, in red and blue the contribution from our previous works [72] and [73], and in orange the new contributions in this paper.

described in Chapter 5. Next, we describe the proposed 3D reconstruction based on contour matching and the local optimization of such point cloud.

4.3 Feature Selection and 3D Reconstruction from Stereo Contour Matching

To ensure that the VIO system and the 3D reconstruction can be run in real-time in parallel, we replaced the OKVIS feature detection method with the one described [84], which provides a short list of the most prominent features based on the *corner response* function in the images. This reduces the computation in the *frontend* tracking and, as shown in the results, retains the same accuracy with less computational requirements. Some studies [68, 85] show comparison between different feature descriptors for a variety of domains.

We included a real-time stereo contour matching algorithm followed by an outlier rejection mechanism to produce the point-cloud on the contour created by the light; see Fig. 6.17c for an example of all the edge-features detected. The approach of

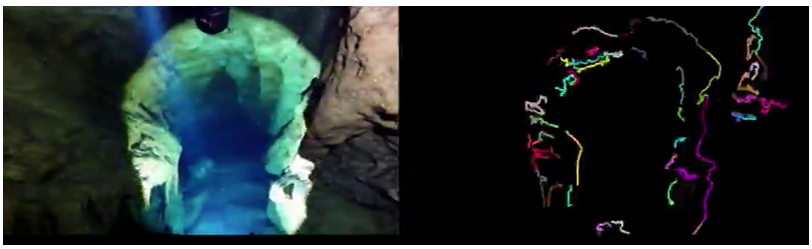


Figure 4.3: Image in a cave and the detected contours.

Weidner et. al [98] has been adapted for the contours from the intersection of the cone of light with the cave wall; see Fig. 4.3 for the extracted contours from an underwater cave. In particular, adaptive thresholding of the images based on the light and dark areas ensures that the illuminated areas are clearly defined. In our current work, we also found that sampling from pixels which have rich gradient, e.g., edges, provides better and denser point-cloud reconstructions. As such, both types of edges – one marking the boundaries between the light and dark areas and the other from visible cave walls – are used to reconstruct the 3-D map of the cave. The overview of the augmenting Stereo Contour Matching method in our tightly-coupled Sonar-Visual-Inertial-Depth optimization framework is as follows.

For every *frame* in the local optimization window, a noisy edge map is created from the edges described above, followed by a filtering process to discard short contours by calculating their corresponding bounding boxes and only keeping the largest third percentile. This method retains the highly defined continuous contours of the surroundings while eliminating spurious false edges, thus allowing to use the pixels on them as good features to be used in the reconstruction. In a stereo frame, for every image point on the contour of the left image a BRISK feature descriptor is calculated and matched against the right image searching along the epipolar line. Then a sub-pixel accurate localization of the matching disparity is performed. Another layer of filtering is done based on the grouping of the edge detector, i.e., keeping only the consecutive points belonging to the same contour in a stereo pair. These stereo contour matched features along with the depth estimation are projected into

3-D which are projected back for checking the reprojection error consistency resulting into a point-cloud with very low reprojection error.

The reason behind choosing stereo matched contour features rather than tracking them using a semi-direct method is to avoid any spurious edge detection due to lighting variation in consecutive images, which could lead to erroneous estimation or even tracking failure. The performance of SVO [30], an open-source state-of-the-art semi-direct method, in underwater datasets [69, 46] validates this statement. In addition, though indirect feature extractors and descriptors are invariant to photometric variations to some extent, using a large number of features for tracking and thus using them for reconstruction is unrealistic due to the computational complexity of maintaining them.

4.4 Local Bundle Adjustment (BA) for Contour Features

In the current optimization window, a local BA is performed for all newly detected stereo contour matched features and the keyframes they are observed in, to achieve an optimal reconstruction. A joint non-linear optimization is performed for refining k^{th} keyframe pose ${}_W\mathbf{T}_{C_i}{}^k$ and homogeneous *landmark* j in world coordinate W, ${}_W\boldsymbol{l}^j = [{l_x}^j, {l_y}^j, {l_z}^j, {l_w}^j]$ minimizing the cost function:

$$J(\mathbf{x}) \quad = \quad \sum_{j,k} \rho({\mathbf{e}^{j,k}}^T \mathbf{P}^{j,k} \mathbf{e}^{j,k}) \tag{4.1}$$

Hereby $\mathbf{P}^{j,k}$ denotes the information matrix of associated landmark measurement, ρ is the Huber loss function to *down-weight* outliers. The reprojection error, $\mathbf{e}^{j,k}$ for landmark j with matched keypoint measurement $\mathbf{z}_{j,k}$ in image coordinate in the respective camera i is defined as:

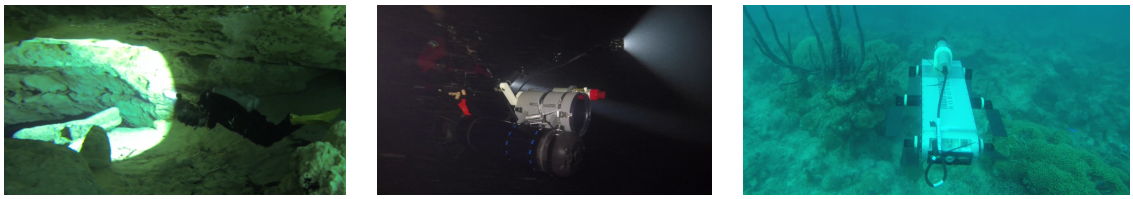
$$\mathbf{e}^{j,k} \quad = \quad \mathbf{z}^{j,k} - \mathbf{h}_i({}_W\mathbf{T}_{C_i}{}^k, {}_W\boldsymbol{l}^j) \tag{4.2}$$

with camera projection model $\mathbf{h}_i$. We used Levenberg-Marquardt to solve local BA problem which obtains a good estimation for the non-linear optimization system.

4.5 Discussion and Conclusion

The proposed system improves the point cloud reconstruction and is able to perform in real time even with additional capabilities. One of the lessons learned during experimental activities is that the position of the light affects also the quality of the reconstruction. In the next version of the sensor suite, we plan to mount the dive light in a fixed position so that the cone of light can be predicted according to the characteristics of the dive light. Furthermore, setting the maximum distance of the Sonar according to the specific environment improves the range measurements.

For this work, we deployed the sensor suite either hand-held by a diver – see Fig. 4.4a – or mounted on a DPV – see 4.4b – in a variety of locations. Future plans are to deploy the sensor suite on a dual DPV which will provide greater stability – see Fig. 4.1 for preliminary tests. Furthermore, the sonar can be deployed on an AUV, such as an Aqua2 [19] vehicle – see 4.4c, for autonomous operations.



(a) (b) (c)

Figure 4.4: Data collection approaches: (a) Diver holds the sensor swimming through the cave. (b) Sensor suite mounted on a DPV. (c) an Aqua 2 vehicle [19] with similar hardware carrying the scanning sonar collects data over a coral reef.

While this work presents the first initiative towards real-time semi-dense reconstruction of challenging environments with lighting variations, there are several scopes for improvements. One future work of interest is to combine a *direct* method and an *indirect* method, similar to [30], but instead of relying on the direct method for tracking, we would rely on the robust Sonar-Visual-Inertial-Depth estimate. Thus we will achieve a denser 3-D reconstruction by jointly minimizing the *reprojection* and *pho-*

tometric error followed by a robust tracking method. We also plan to acquire ground truth trajectories [99] by placing *AprilTags* along each trajectory for quantitative analysis. By deploying the sensor suite on a dual DPV, more accurate results due to the greater stability are expected, as shown in Fig. 4.1 for preliminary tests.

Chapter 5

A Modular Sensor Suite for Underwater Reconstruction

5.1 Introduction

This chapter presents the design, development, and deployment of an underwater sensor suite to be operated by human divers. The literature mainly focuses on AUVs and Autonomous Surface Vehicles (ASVs), and a body of work studies the Simultaneous Mapping and Localization (SLAM) problem and oceanographic reconstruction. Leedekerken et al. [54] presented an Autonomous Surface Craft (ASC) for concurrent mapping both above and below the water surface in large scale marine environments using a surface craft equipped with imaging sonar for subsurface perception and LIDAR, camera, and radar for perception above the surface.

Folaga [3], a low cost AUV, can navigate on the sea surface and dive only at selected geographical points when measurements are needed. Roman et al. [77] proposed an AUV equipped with camera and pencil beam sonar for applications including underwater photo-mosaicking, 3D image reconstruction, mapping, and navigation. Aqua2 [19], a visually guided legged swimming robot uses vision to navigate underwater and the target application areas are environmental assessment [42] and longitudinal analysis of coral reef environments [36]. Our aim is to accelerate state estimation research in the underwater domain that can be eventually deployed robustly in autonomous underwater vehicles (AUV) by enabling easy data collection by human divers. In particular, a specific target application is cave mapping, where the

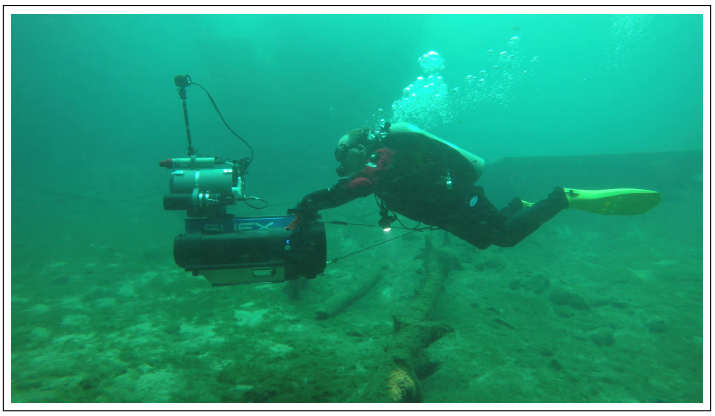


Figure 5.1: Our proposed underwater sensor suite mounted on a dual Diver Propulsion Vehicle (DPV), where a stability check was performed at Blue Grotto, FL.

diving community has protocols in place for exploring and mapping such dangerous environments. The primary design goal of the proposed underwater sensor suite is to reduce the cognitive load of human divers by employing robotic technologies to map underwater structures. A second design goal is to enable software interoperability between different platforms, including AUVs. In particular, the sensor suite presented in this chapter contains identical sensors with an Aqua2 AUV [19], and can be deployed in different modes, hand-held by a diver, mounted on a single Diver Propulsion Vehicle (DPV), or on a dual DPV for better stability; see Fig. 5.1. The selected sensors include a mechanical scanning sonar, which provides robust range information about the presence of obstacles. Such a design choice improves the scale estimation by fusing acoustic range data into the visual-inertial framework [72].

The chapter is structured as follows. The next section outlines the design layout of hardware and software, deployment strategies, and the two versions of the sensor suite. The chapter concludes with a discussion on lessons learned and directions of future work.

5.2 Sensor Suite Design Requirements

The sensor suite hardware has been designed with underwater cave mapping [98] as the target application to be used by divers during cave exploration operations. In general, it can be used for mapping a variety of underwater structures and objects. In the following, the main requirements, hardware, and software components, are presented. Note that the full documentation for building and maintaining the hardware, as well as the necessary software can be found on our lab wiki page [5].

Given that the sensor suite will be primarily used by divers who are not necessarily engineers or computer scientists, the following requirements drive the hardware and software design of the proposed sensor suite:

- Portable.
- Neutrally buoyant.
- Hand-held or DPV deployment.
- Simple to operate.
- Waterproof to technical-diver operational depths.

Furthermore, the following desiderata are considered to make research in state estimation applied to the proposed sensor suite easily portable to other platforms, such as AUVs and ASVs:

- Standardization of hardware and software.
- Easy data storing.
- Low cost.

5.3 Hardware Design

In this section, the electronics selected and the designed enclosure are discussed, together with lessons learned during the construction of the proposed sensor suite.

5.3.1 Electronics

To assist vision-based state estimation, we employ an Inertial Measurement Unit (IMU), a depth, and an acoustic sensor for accurate state estimation in underwater environments. The specific sensors and electronics of the sensor suite were selected for compatibility with the Aqua2 Autonomous Underwater Vehicles (AUVs) [19]. Figure 5.2 shows the computer and internal sensors on a Plexiglas plate, where the different electronic boards were placed optimizing the space to reduce the size of the sensor suite. In particular, the electronics consists of:

- two IDS UI-3251LE cameras in a stereo configuration,
- Microstrain 3DM-GX4-15 IMU,
- Bluerobotics Bar30 pressure sensor,
- Intel NUC as the computing unit,
- IMAGENEX 831L Sonar.

The two cameras are synchronized via a TinyLily, an Arduino-compatible board, and are capable of capturing images of 1600×1200 resolution at 20 Hz. The sonar provides range measurement with maximum range of 6 m distance, scanning in a plane over 360°, with angular resolution of 0.9°. A complete scan at 6 m takes 4 s. Note that the sonar provides for each measurement (point) 255 intensity values, that is $6/255$ m is the distance between each returned intensity value. Clearly, higher response means a more likely presence of an obstacle. Sediment on the floor, porous material, and

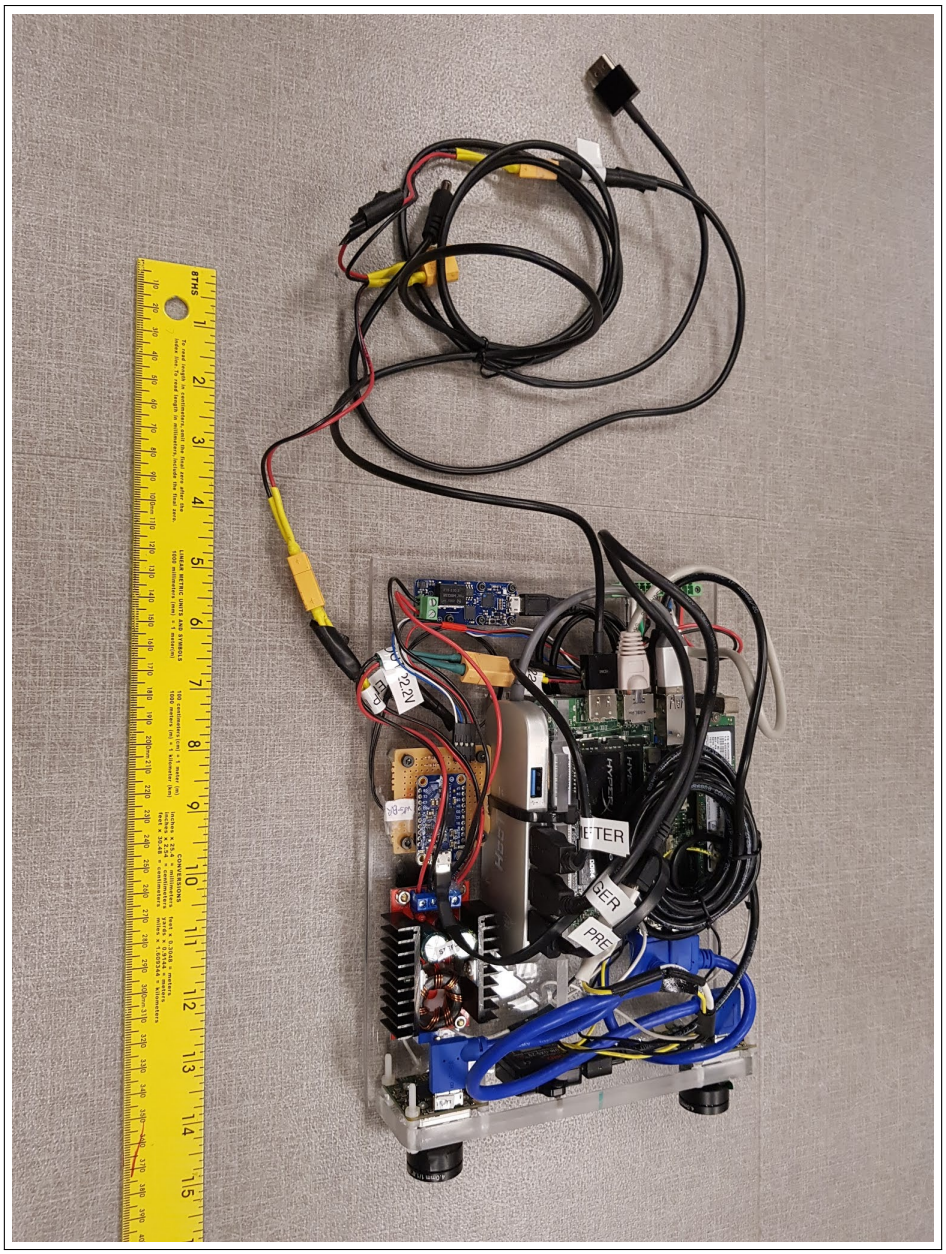
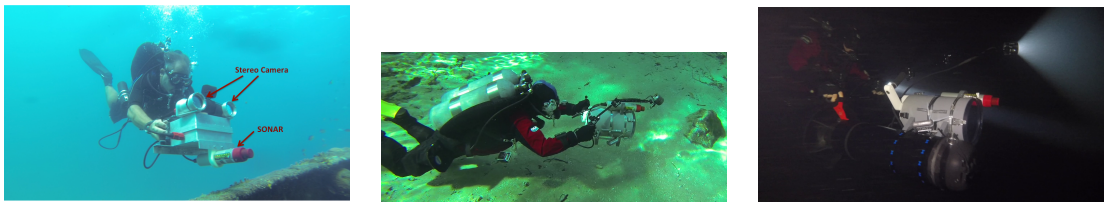


Figure 5.2: The Main Unit containing stereo camera, IMU, Intel NUC, and Pressure sensor.

multiple reflections result in a multi-modal distribution of intensities. The IMU produces linear accelerations and angular velocities in three axis at a frequency of 100 Hz. Finally, the depth sensor produces depth measurements at 1 Hz. To enable the easy processing of data, the Robot Operating System (ROS) framework in [70] has been utilized for the sensor drivers and for recording timestamped data.



(a) (b) (c)

Figure 5.3: (a) First version of the stereo vision setup, where the two cameras are mounted externally to the main unit. (b) Second version of the sensor suite, where the stereo camera is inside the main unit. (c) Second version where the sensor suite is mounted on a DPV.

A 5 inch LED display has been added to provide visual feedback to the diver together with a system based on AR tags is used for changing parameters and to start/stop the recording underwater in [102] (see Section 5.4).

5.3.2 Enclosure

The enclosure for the electronics has been designed to ensure ease of operations by divers and waterproofness up to 100 m. In particular, two different designs were tested. Both of them are characterized by the presence of handles for hand-held operations. The handles have been chosen so that a dive light can be easily added using a set of articulated arms. Note that all enclosures are sealed with proper o-rings/gaskets (details are reported in the linked documentation).

In the first design (see Fig. 5.3(a)) the main unit, a square shaped aluminum box – composed of two parts tighten together by screws – contained the computer, sensors, and other related electronics. The two cameras were sealed in aluminum tubes with tempered glass in front of the camera lenses. The stereo camera and display were mounted on the top of the main unit whereas the sonar was on the bottom of it. Both the cameras and sonar were connected to the main unit by underwater cables. The rationale behind such a design was to allow for an adjustable stereo baseline. Unfortunately, the USB 3.0 interfacing standard used by the cameras is not

compatible with the underwater cables available in the market, resulting in highly degraded performance for the cameras with multiple dropped frames. In addition, the aluminum body made the sensor suite relatively heavy and negative buoyant. Furthermore, the position of the screen was not optimal for seeing it during regular diver deployment.

In the second design (see Fig. 5.3b), we took into account the lessons learned from the first design. In particular, a PVC tube was used instead of the aluminum box. This made the enclosure lighter and positive buoyant. Some rails at the bottom allows for adding extra weights for ballasting. Furthermore, the main enclosure hosted the two cameras as well. In this way, the cameras can be directly connected to the computer with standard USB 3.0 cables, to avoid unnecessary transmission of data over underwater cables as it was in the first design. The front panel is made of transparent Plexiglas, 33 mm thickness, while the back panel is made of aluminum, where a waterproof switch, a display, pressure sensor, and underwater connector for the sonar are mounted. Stainless steel Latches are used to close the panels with the PVC tube, so that it can be easily open and maintained. The sonar was mounted on the top with the scanning plane parallel to the image plane and connected to the main unit by a standard SubConn underwater cable. Such a design and choice of material reduced the size and weight, and made it easier to carry and maintain. In addition, the second design of the sensor suite allows for modularity in terms of electronics used: a Plexiglas plate inside the enclosure was used to mount all the electronics and can be easily removed for troubleshooting or changed with a different computer, cameras, and IMU.

The second version of the sensor suite has been designed considering two different deployment strategies: hand-held and on different diver propulsion vehicles (DPV). Such deployment strategies depend on the structure of the environment and the distance to cover. The hand-held approach is more appropriate for covering a smaller

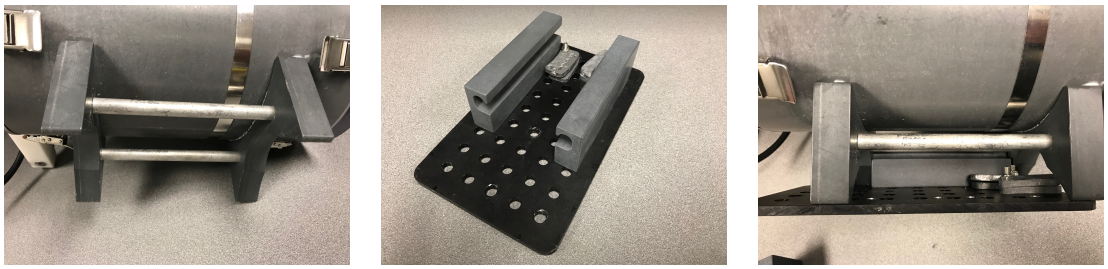
Figure 5.4: Front top view of the assembled sensor suite.

area for a short period of time, whereas the sensor suite can be mounted on a single or double DPV in order to collect data over longer distances while being under water. Mounting the rig on a DPV is specifically useful in cave diving, at larger depths, to make better use of limited underwater time. Hand-held operations are possible through the handles on the side of the PVC tube, as shown in Fig. 5.3(b). DPV operations can be performed in two ways. First, mounted on a single DPV unit; see Fig. 5.3(c). Second, mounted on a dual DPV unit; see Fig. 5.1.

Fig. 5.4 shows a front view of the sensor suite fully assembled. The two side-ring holders are used to mount a canister battery for the video light; usually, a 13.5Ah NiMH standard battery.

5.3.3 Mounting Options

Mounting the sensor suite on single or dual DPVs uses different attachment methods. For single DPV attachment hose-clamps are used through the two metal bars to secure the sensor; see Fig. 5.5a. Please note, the bottom of the sensor suite has a



(a) (b) (c)

Figure 5.5: (a) The mounting system for single DPV deployment. (b) Mounting attachment for use with a dual DPV. (c) The dual DPV attachment partially mount on the bottom of the sensor suite.

round hollow that fits on a SUEX[1] DPV; either XJ37 or XK1 models. For mounting on a dual DPV, an attachment system is used; see Fig. 5.5b. The PVC components are hooked through the supporting metal poles at the bottom; see Fig. 5.5c where the plate is half mounted. When the plate is attached to the bottom of the sensor suite, then it locks on the railing system of the dual DPV unit. The mounting on the DPV can be carried out while in water, allowing divers to easily carry modular parts to the entry point for the dive. It is worth noting that the cheese-board and rail design allow for changing the location of the sensor on the dual DPV.

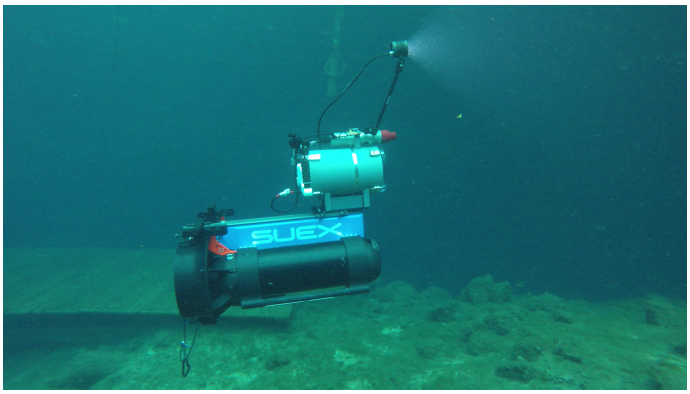


Figure 5.6: Sensor suite on a dual DPV free floating, neutrally buoyant.

[1]https://www.suex.it/

Fig. 5.6 demonstrates the stability of the sensor suite on a dual DPV. The unit floats in the water neutrally buoyant, with the video light on top illuminating forward.

5.4 Software Design

The main software components of the sensor suite consist of:

- drivers for each hardware unit,
- a ROS interface for communication between sensors and data processing,
- an interface for user and sensor suite interaction.

5.4.1 Drivers

The aim for the software design is to have a modular system that ensures re-usability for both the system as a whole and also for each component. Each driver provides consistent interface for communication with the Robot Operating System (ROS). The main ROS drivers are:

- UEye driver for each camera, together with the Arduino code for the trigger to synchronize the cameras – available open-source [4].
- IMU driver – available open-source [51].
- Sonar driver – developed in our lab, released open-source [5].
- Depth sensor driver - developed in our lab, released open-source [5].

5.4.2 ROS platform

For easy data collection, each sensor node publishes the related data. All the operations are performed on the computer that runs a Linux-based operating system. In

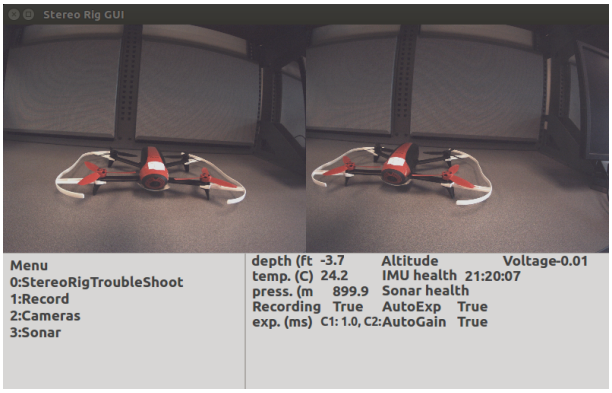


Figure 5.7: The default view of the menu.

particular, the Software was tested both on Ubuntu 14.04 and 16.04. After the operating system boot, a startup script runs all sensor nodes and at the same time starts the recording of sensor data through ROS bag file[2] that allows for easy play-back.

5.4.3 Interface

The interface consists of two components: Graphical User Interface (GUI) for online data monitoring; and AR tags [26] that supports user and sensor suite interaction, similarly to the proposed system by Sattar et al. [80]. The GUI – based on Qt[3] for modularity – shows the current video stream of each camera and outputs the overall health of the system. Fig. 5.7 shows the sensor data from the GUI. Depth in feet and altitude represent the distance from the surface and from the bottom respectively; measured by the depth and the Sonar sensors. The temperature of the CPU is also reported in case there is overheating, especially if operations are started above water. In addition, the GUI shows a menu with a list of options that a user can select; left side of the screen. Each option has a corresponding AR tag associated with its number. Through the menu a user can perform basic operations on the computer – such as reboot or shutdown – start or stop recording data, get access to both camera

[2]http://wiki.ros.org/rosbag

[3]https://www.qt.io/

or sonar settings. When a camera is selected, a user can change its gain and exposure and perform camera calibration. In addition, sonar data can be visualized through rviz[4] by selecting the corresponding option. Note that such a menu is modular and straightforward to add, remove, or modify the menu entries. Fig. 5.7 shows how the GUI looks like.

5.5 Conclusion

In this chapter, we presented the design and development of a sensor suite for underwater reconstruction, together with some lessons learned during its construction. Our proposed sensor suite has been used by divers in coral reefs, shipwrecks, and cave systems to collect visual, inertial, and sonar data, and different algorithms have been studied to improve state estimation in caves.

Immediate future work on the proposed sensor suite includes a comprehensive study on the quality of cameras for underwater operations, as well as a more user-friendly electronics placement and wiring. More broadly, such a sensor suite will be mounted on a platform that can operate autonomously, to allow for easy swap of sensors on a robot.

[4]http://wiki.ros.org/rviz

Chapter 6

Experimental Results and Applications

6.1 Evaluation of SVIn2 on Visual-Inertial Benchmark

The proposed state estimation system, termed as SVIn2, is quantitatively validated first on a standard dataset, to ensure that loop closure and the initialization work also above water. Moreover, it is compared to other state-of-the-art methods, i.e., VINS-Mono [67], the basic OKVIS [56], and the MSCKF [62] implementation from the GRASP lab [74]. Second, we qualitatively test the proposed approach on several different datasets collected utilizing a custom made sensor suite [71] and an Aqua2 AUV [19].

Here, we present results on the EuRoC dataset [10], one of the benchmark datasets used by many visual-inertial state estimation systems, including OKVIS (Stereo), VINS-Mono, and MSCKF. To compare the performance, we disable depth and sonar integration in our method and only assess the loop-closure scheme.

Following the current benchmarking practices, an alignment is performed between ground truth and estimated trajectory, by minimizing the least mean square errors between estimate/ground-truth locations, which are temporally close, varying rotation and translation, according to the method from [97]. The resulting metric is the Root Mean Square Error (RMSE) for the translation, shown in Table 6.1 for several Machine Hall sequences in the EuRoC dataset. For each package, every sequence has been run 5 times and the best run (according to RMSE) has been shown. Our method shows reduced RMSE in every sequence from OKVIS, validating the improve-

Table 6.1 The best absolute trajectory error (RMSE) in meters for each Machine Hall EuRoC sequence.

	SVIn2	OKVIS(stereo)	VINS-Mono	MSCKF
MH 01	0.13	0.15	0.07	0.21
MH 02	0.08	0.14	0.08	0.24
MH 03	0.07	0.12	0.05	0.24
MH 04	0.13	0.18	0.15	0.46
MH 05	0.15	0.24	0.11	0.54

ment of pose-estimation after loop-closing. SVIn2 has also less RMSE than MSCKF and slightly higher in some sequences, but comparable, to results from VINS-Mono. Fig. 6.1 shows the trajectories for each method together with the ground truth for the *Machine Hall* sequence.

6.2 Experimental Results of SVIn2 on Our Underwater Datasets

Our proposed state estimation system – SVIn2 – is targeted for the underwater environment, where sonar and depth can be fused together with the visual-inertial data. Here, we show results from four different datasets in three different underwater environments. First, a sunken bus in Fantasy Lake (NC), where data was collected by a diver with a custom-made underwater sensor suite [71]. The diver started from outside the bus, performed a loop around and entered in it from the back door, exited across and finished at the front-top of the bus. The images are affected by haze and low visibility. Second and third, data from an underwater cavern in Ginnie Springs (FL) is collected again by a diver with the same sensor suite as for the sunken bus. The diver performed several loops, around one spot in the second dataset – Cavern1 – and two spots in the third dataset – Cavern2 – inside the cavern. The environment

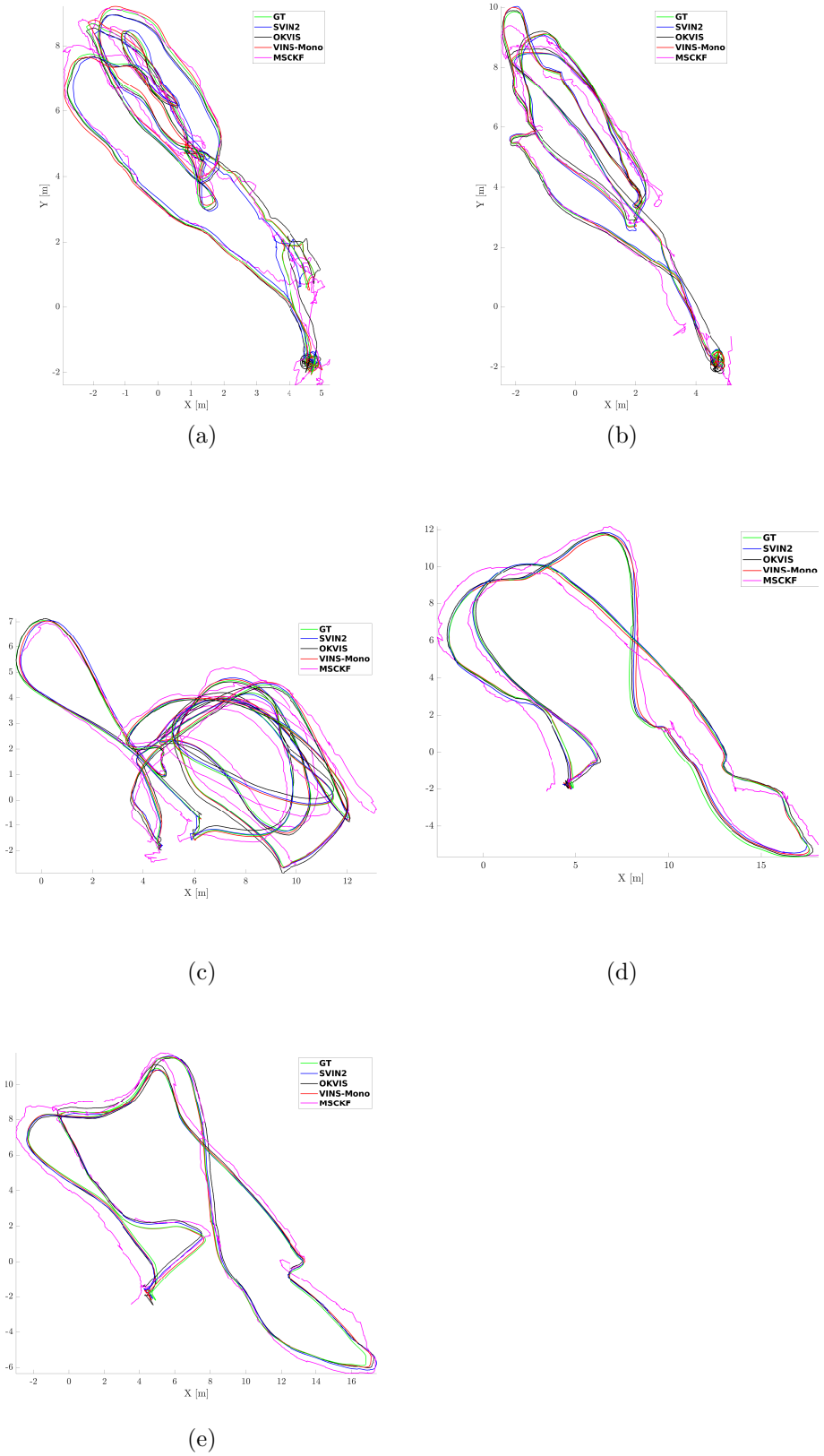


Figure 6.1: Trajectories on the MH sequence of the EuRoC dataset.

is affected by complete absence of natural light. Fourth, an AUV – Aqua2 robot – collected data over a fake underwater cemetery in Lake Jocassee (SC) and performed several loops around the tombstones in a square pattern. The visibility, as well as brightness and contrast, was very low. In the underwater datasets, it is a challenge to get any ground truth, because it is a GPS-denied unstructured environment. As such, the evaluation is qualitative, with a rough estimate on the size of the environment measured beforehand by the divers collecting the data.

6.2.1 Experimental Setup

The experimental data were collected using a custom made sensor suite [71] and Aqua2 robot Fig. 6.2, consisting of a stereo camera, an IMU, a depth sensor and a mechanical scanning Sonar, as described in Chapter 5. More specifically, two USB-3 uEye cameras in a stereo configuration provide data at 15 Hz, an IMAGENEX 831L mechanical scanning Sonar sensor acquires a full 360° scan every four seconds; the Bluerobotics Bar30 pressure sensor provides depth data at 1 Hz; a MicroStrain 3DM-GX4-15 IMU generates inertial data at 100 Hz; and an Intel NUC running Linux and ROS consolidates all the data. A video light is attached to the sensor suite unit to provide artificial illumination of the scene. The Sonar is mounted on top of the main unit which contains the remaining electronics. The unit can be seen deployed mounted on a dual Diver Propulsion Vehicle (DPV); please note, the system is neutrally buoyant and stable. The experiments were run on a computer with an Intel i7-7700 CPU @ 3.60GHz, 32 GB RAM, running Ubuntu 16.04 and ROS Kinetic and on an Intel NUC with the same configuration.

6.2.2 Trajectory Evaluation

Figs. 6.3-6.6 show the trajectories from SVIn2, OKVIS, and VINS-Mono in the datasets just described. MSCKF was able to keep track only for some small segments

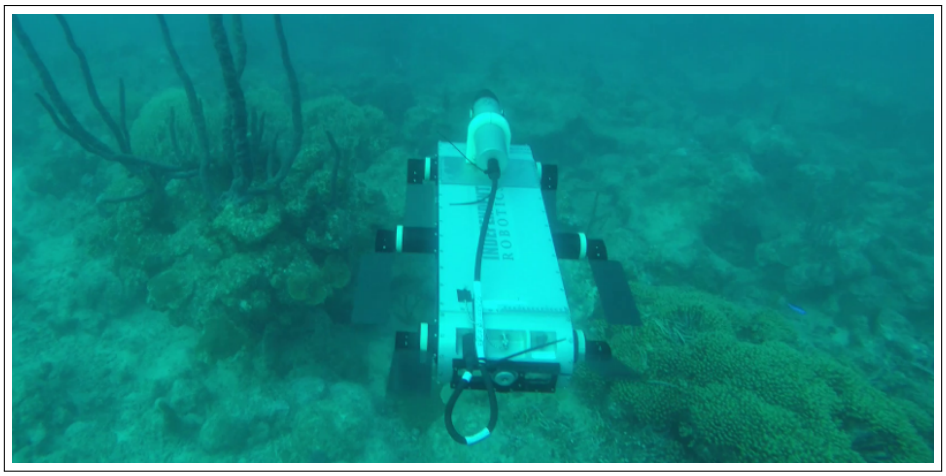


Figure 6.2: The Aqua2 AUV [19] equipped with the scanning sonar collecting data over the coral reef.

in all the datasets, hence excluded from the plots. For a fair comparison, when the trajectories were compared against each other, sonar and depth were disabled in SVIn2. All trajectories are plotted keeping the original scale produced by each package.

Fig. 6.3 shows the results for the submerged bus dataset. In particular, VINS-Mono lost track when the exposure increased for quite some time. It tried to re-initialize, but it was not able to track successfully. Even using *histogram equalization* or a *contrast adjusted histogram equalization* filter, VINS-Mono was not able to track. Even if the scale drifted, OKVIS was able to track using a *contrast adjusted histogram equalization* filter in the image pre-processing step. Without the filter, it lost track at the high exposure location. The proposed method was able to track, detect, and correct the loop, successfully.

In Cavern1 – see Fig. 6.4 – VINS-Mono tracked successfully the whole time. However, as can be noticed in Fig. 6.4c, the scale was incorrect based on empirical observations during data collection. OKVIS instead produced a good trajectory, and SVIn2 was also able to detect and close the loops.

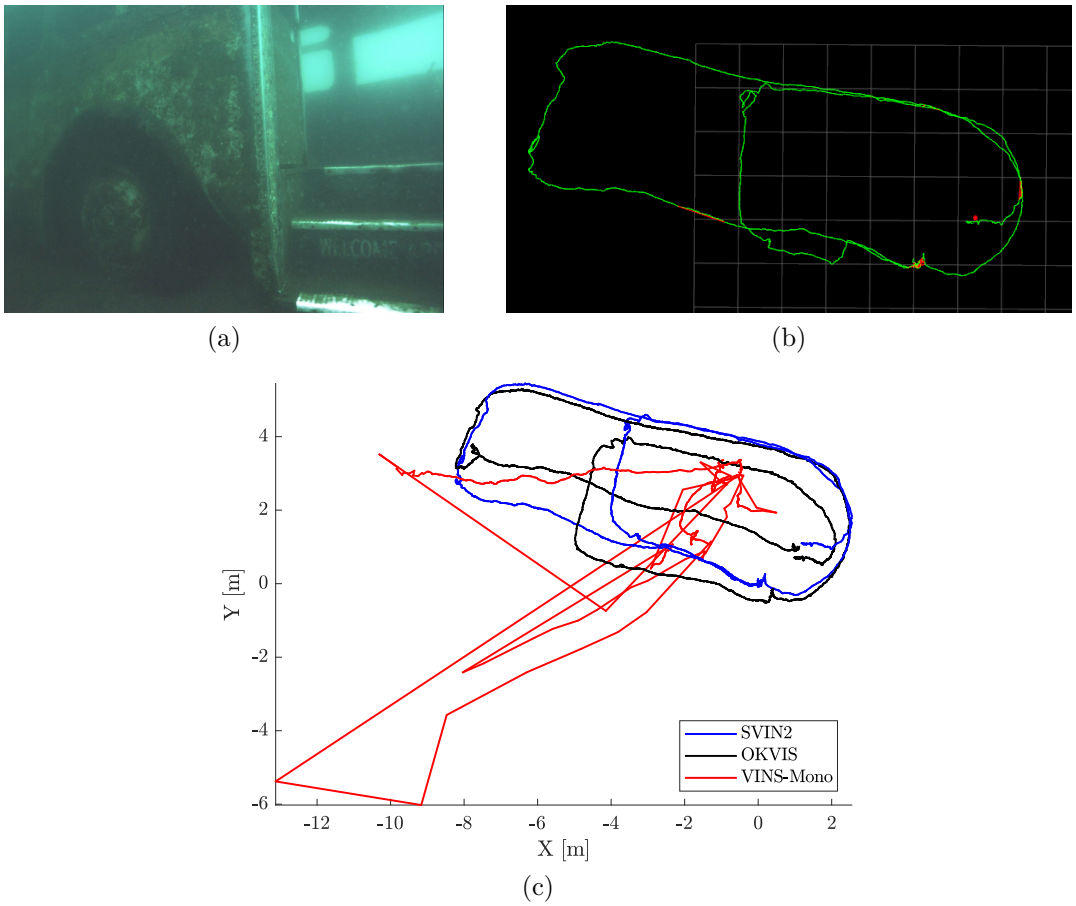


Figure 6.3: (a) Submerged bus, Fantasy Lake, NC, USA with a 53 m trajectory; trajectories from SVIn2 with all sensors enabled shown in rviz (b) and aligned trajectories from SVIn2 with Sonar and depth disabled, OKVIS, and VINS-Mono (c) are displayed.

In Cavern2 (Fig. 6.5), VINS-Mono lost track at the beginning, reinitialized, was able to track for some time, and detected a loop, before losing track again. VINS-Mono had similar behavior even if the images were pre-processed with different filters. OKVIS tracked well, but as drifts accumulated over time, it was not able to join the current pose with a previous pose where a loop was expected. SVIn2 was able to track and reduce the drift in the trajectory with successful loop closure.

In the cemetery dataset – Fig. 6.6 – both VINS-Mono and OKVIS were able to track, but VINS-Mono was not able to reduce the drift in trajectory, while SVIn2 was able to fuse and correct the loops.

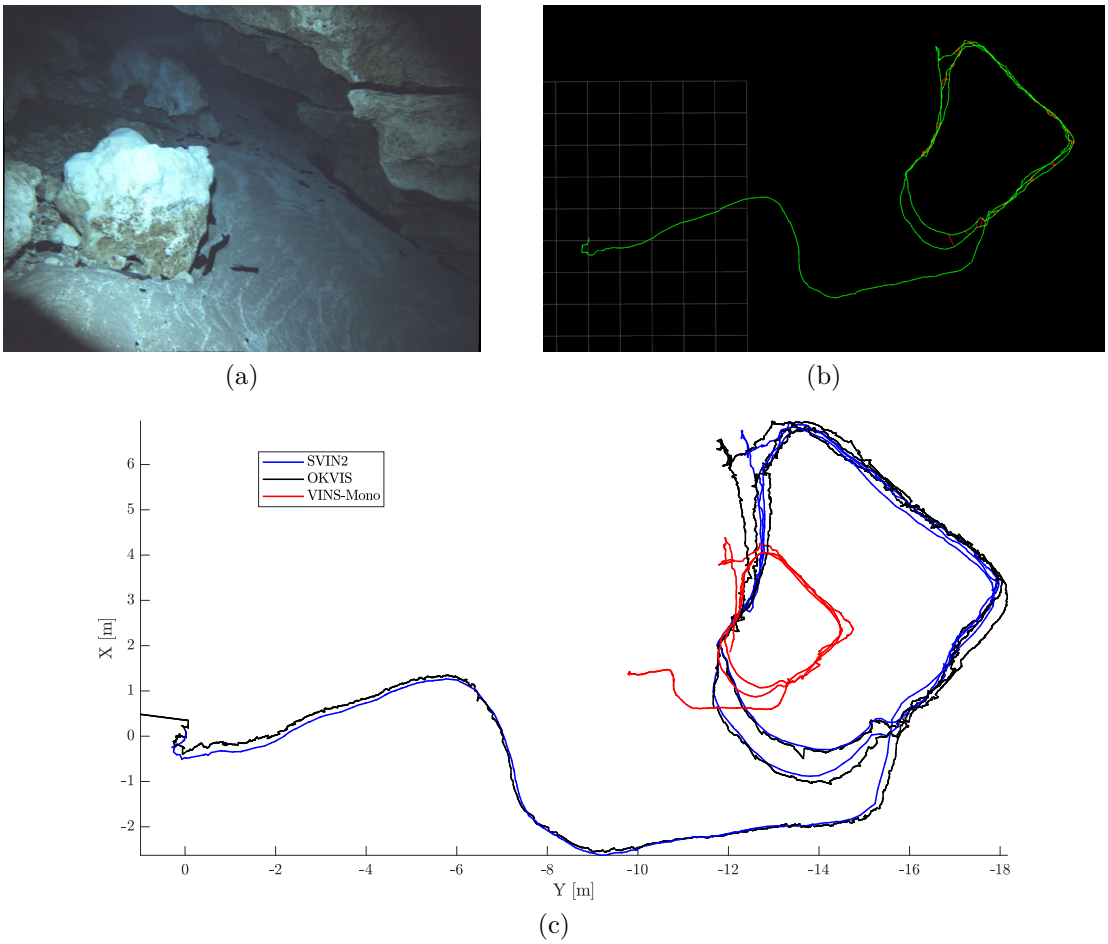
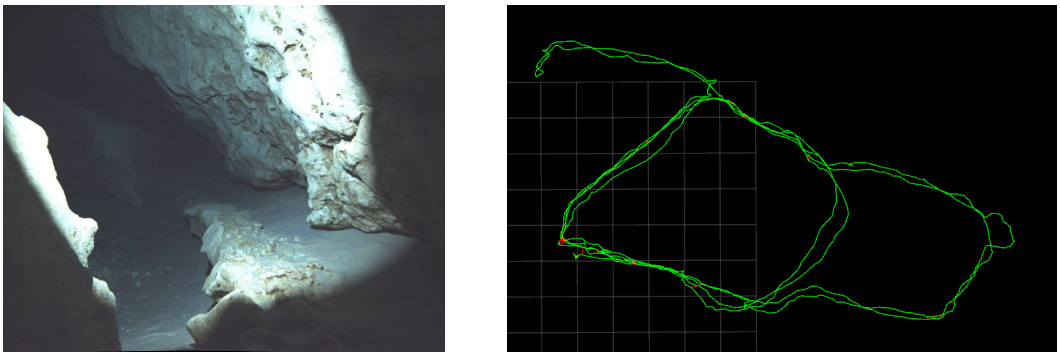


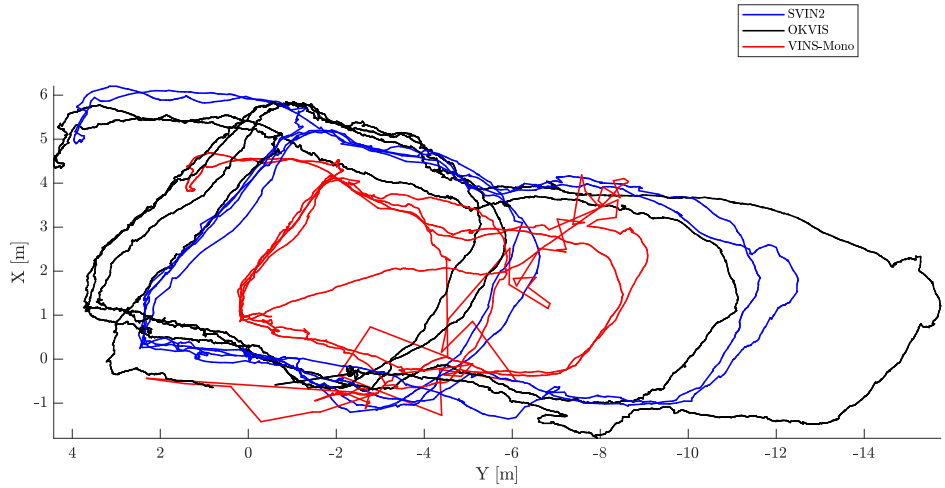
Figure 6.4: (a) Cave environment, Ballroom, Ginnie Springs, FL, USA, with a unique loop covering a 87 m trajectory; trajectories from SVIn2 with all sensors enabled shown in rviz (b) and aligned trajectories from SVIn2 with Sonar and depth disabled, OKVIS, and VINS-Mono (c) are displayed.

6.2.3 COLMAP as Comparative Baseline

COLMAP [83], an opensource SfM library, performs best among the conventional state-of-the-art MVS algorithms as it uses tight integration of multiple techniques, e.g., robust neighbor view selection and incorporation of visibility constraints. We used it to generate trajectories for each of our underwater datasets with loop detection enabled via vocabulary tree search and use them for quantitative performance evaluation. While COLMAP provides an estimation on the shape of the trajectories, they cannot be considered as absolute ground truth as the *scale* information



(a) (b)



(c)

Figure 6.5: (a) Cave environment, Ballroom, Ginnie Springs, FL, USA, with two loops in different areas covering a 155 m trajectory; trajectories from SVIn2 with all sensors enabled shown in rviz (b) and aligned trajectories from SVIn2 with Sonar and depth disabled, OKVIS, and VINS-Mono (c) are displayed.

cannot be recovered. Indeed, for submerged bus, Cavern2, and cemetery datasets COLMAP only produced partial trajectories due to the water turbidity, low visibility, and lack of good features to track for a longer period in the scene – an indication that vision-only state estimation for underwater is not reliable. As such, we aligned the estimated trajectories with scale from our system as well as the other opensource visual-inertial packages with respect to COLMAP and calculated RMSE for each of them, shown in Table 6.2. SVIn2 has lower RMSE in each of the datsets. MSCKF has been excluded from the table as it failed to track in all of them. Fig. 6.7 shows the

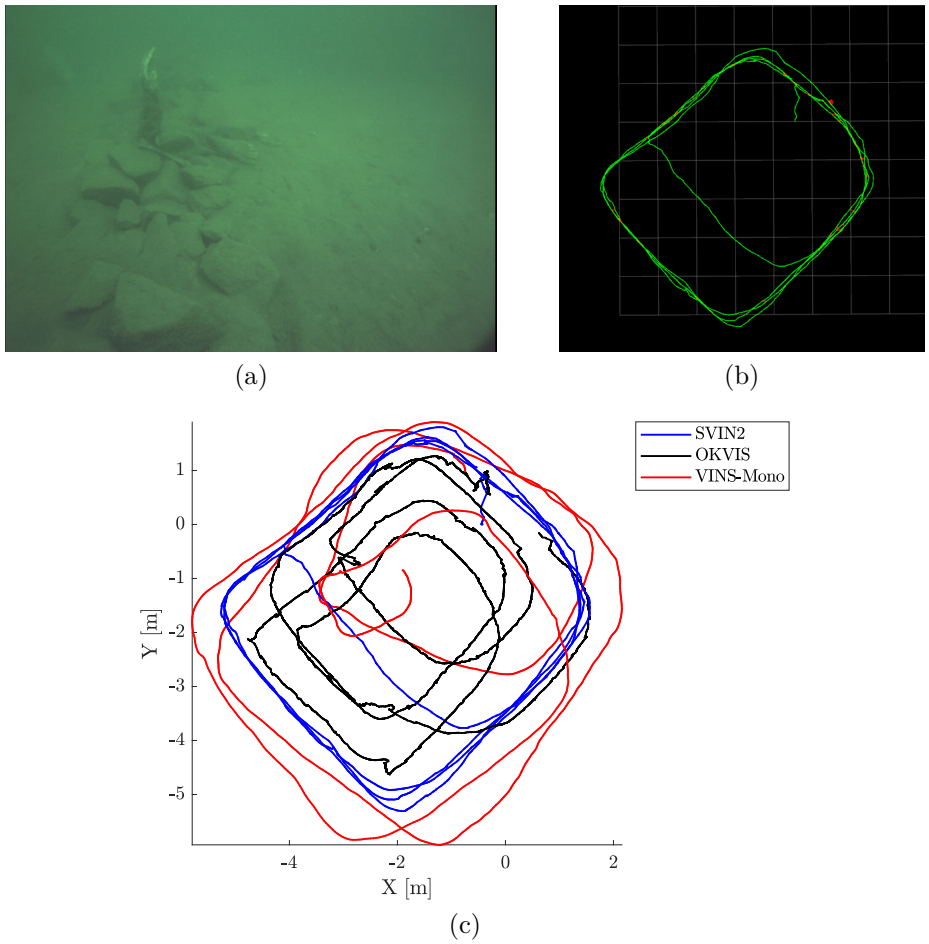


Figure 6.6: (a) Aqua2 in a fake cemetery, Lake Jocassee, SC, USA with a 80 m trajectory; trajectories from SVIn2 with visual, inertial, and depth sensor (no sonar data has been used) shown in rviz (b) and aligned trajectories from SVIn2 with Sonar and depth disabled, OKVIS, and VINS-Mono (c) are displayed.

trajectories for each method together with the ground truth generated by COLMAP for our underwater datasets.

6.3 Reconstruction using Sonar Data

The proposed approach in Chapter 3 has been tested in numerous challenging environments. In this section, descriptions of each dataset together with the state estimate of the sensor suite and challenges during the field trials are presented.

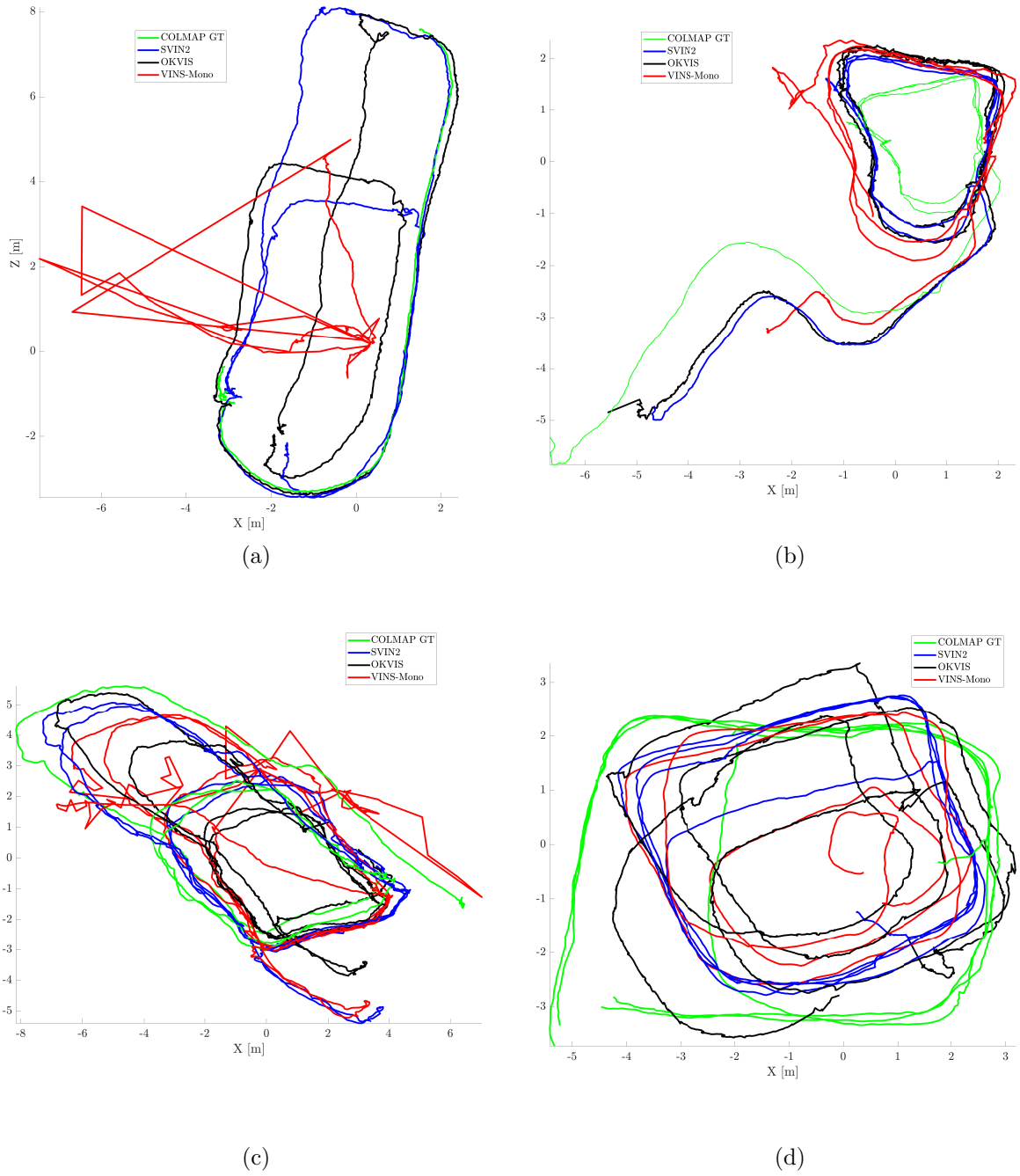


Figure 6.7: COLMAP GT trajectories and estimated trajectories from SVIn2, OKVIS (stereo), and VINS-Mono aligned with scale. (a) Submerged bus, (b) Cave environment with a unique loop, (c) Cave environment with two loops in two different area, and (d) Fake cemetery are shown.

Table 6.2 The RMSE in meters for each underwater datasets.

	SVIn2	OKVIS(stereo)	VINS-Mono
Bus (partial)	**0.1031**	0.3896	3.1336
Cavern1	**0.7052**	0.7110	1.1387
Cavern2 (partial)	**0.9247**	1.4229	1.2796
Cemetery (partial)	**1.6778**	1.7917	2.2006

One of the first datasets was collected at an artificial shipwreck in Barbados; see Fig. 6.9a. The initial deployment of the sonar sensor suffered from a misconfiguration where data was collected at a very slow rate and at a maximum range of one meter resulting on only collecting sonar data from the floor. Note that Fig. 6.9c shows the trajectory of the camera going slightly upwards, although the frame shows the floor parallel to the motion. The shipwreck was sunken on the sea floor with some inclination, that the IMU was able to capture.

We collected also a short segment from inside a cavern in Ginnie Springs, in Florida (USA). Such footage provided preliminary data from an underwater cave environment; see Fig. 6.10a. The video light utilized was providing illumination on only part of the scene. The reconstruction shows both visual landmarks and sonar points giving a sense of the cavern as the diver was swimming around. In such a case, the sonar was correctly configured; however, because the light was not uniformly illuminating the scene, the visual features were not optimal.

Finally, the inside of a sunken bus was mapped at Fantasy Lake Scuba Park, NC, USA; see Fig. 6.11a. The image quality was quite poor due to the many particulates in the water. In all environments, the images contain a significant amount of blur (softness) which clearly increases with distance. In addition, dynamic obstacles, such as fish, but more importantly floating particles that reflect back with high intensity;

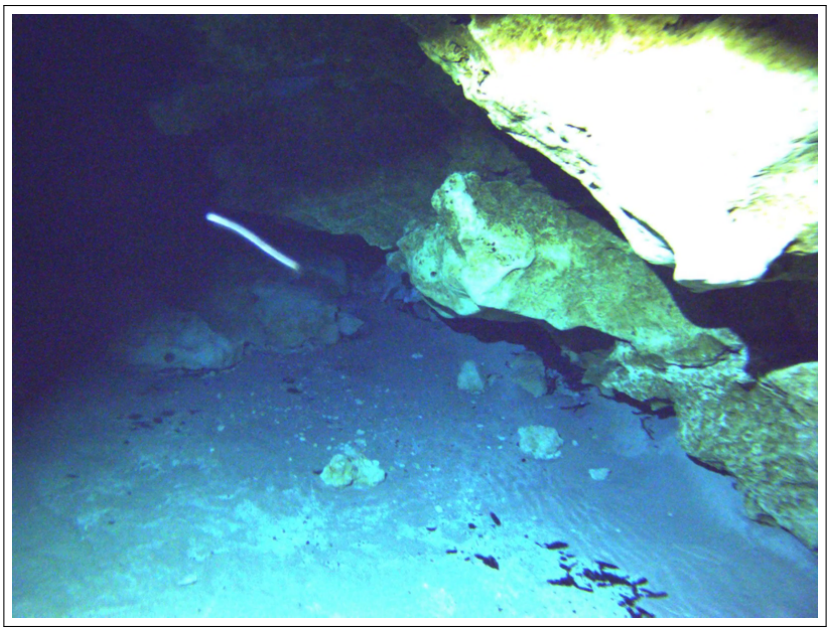
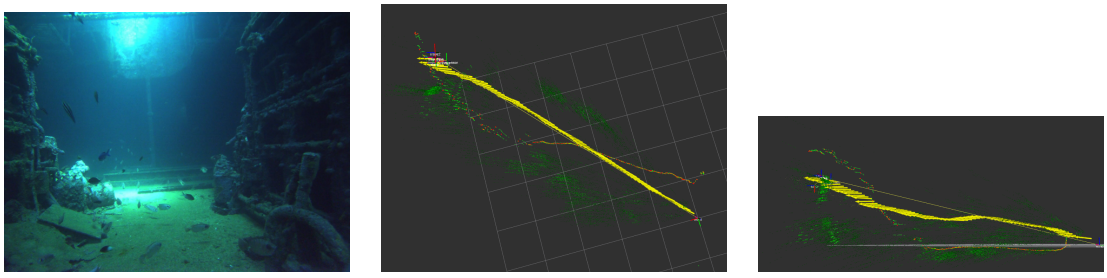


Figure 6.8: A small particle reflecting back at high speed generating a blurry streak. In addition light reflecting back from a nearby surface completely saturates the camera.

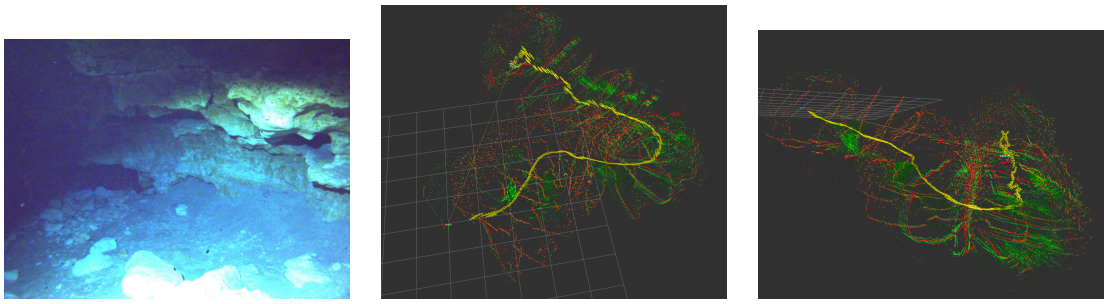
see Fig. 6.8, where floating particles were present in all datasets.

In such challenging environments, it is very hard to get ground truth. However, the trajectory and the distance covered resembled the one followed by the diver. Further, the sonar landmarks were indeed used to correct the pose estimate. All the results in the datasets, except for the shipwreck, show several rings, indicating the mapping of the structure.



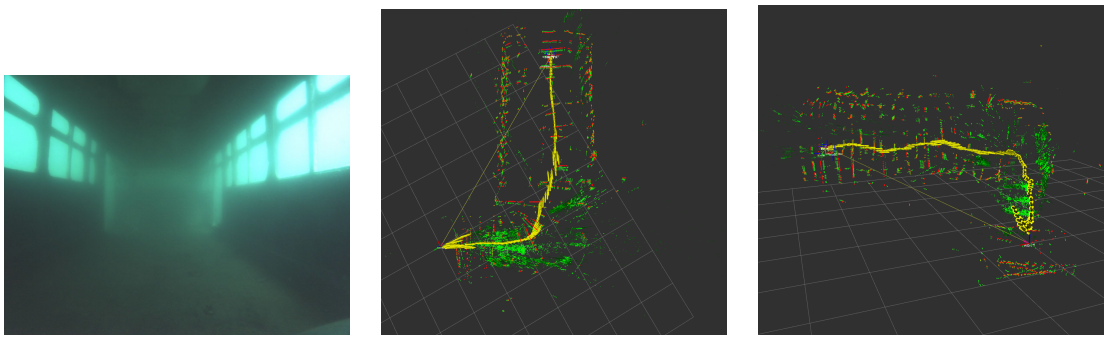
(a) (b) (c)

Figure 6.9: Bajan Queen artificial reef (shipwreck) in Carlisle Bay, Barbados. (a) Sample image of the data collected inside the wreck (beginning of trajectory). (b) Top view of the reconstruction.



(a) (b) (c)

Figure 6.10: Underwater cave, Ballroom Ginnie cavern at High Springs, FL, USA. (a) Sample image of the data collected inside the cavern. (b) Top view of the reconstruction. (c) Side view of the reconstruction.



(a) (b) (c)

Figure 6.11: Sunken bus, Fantasy Lake Scuba Park, NC, USA. (a) Sample image of the data collected from inside the bus. (b) Top view of the reconstruction. (c) Side view of the reconstruction, note the stairs detected by visual features at the right side of the image.

6.4 Validation on Public Underwater Datasets

Ferrera et al. [25] provided a sequence of underwater datasets close to seabed, named AQUALOC, collected by a Remotely Operated Vehicle (ROV) equipped with a monocular monochromatic camera, a MEMS-IMU, and a pressure sensor. The datasets are characterized by turbidity, backscattering effect, and sandy clouds – Fig. 6.12 shows some representative pictures of the sites. They provided COLMAP trajectories as ground truth to compare and evaluate performance for SLAM systems. In few

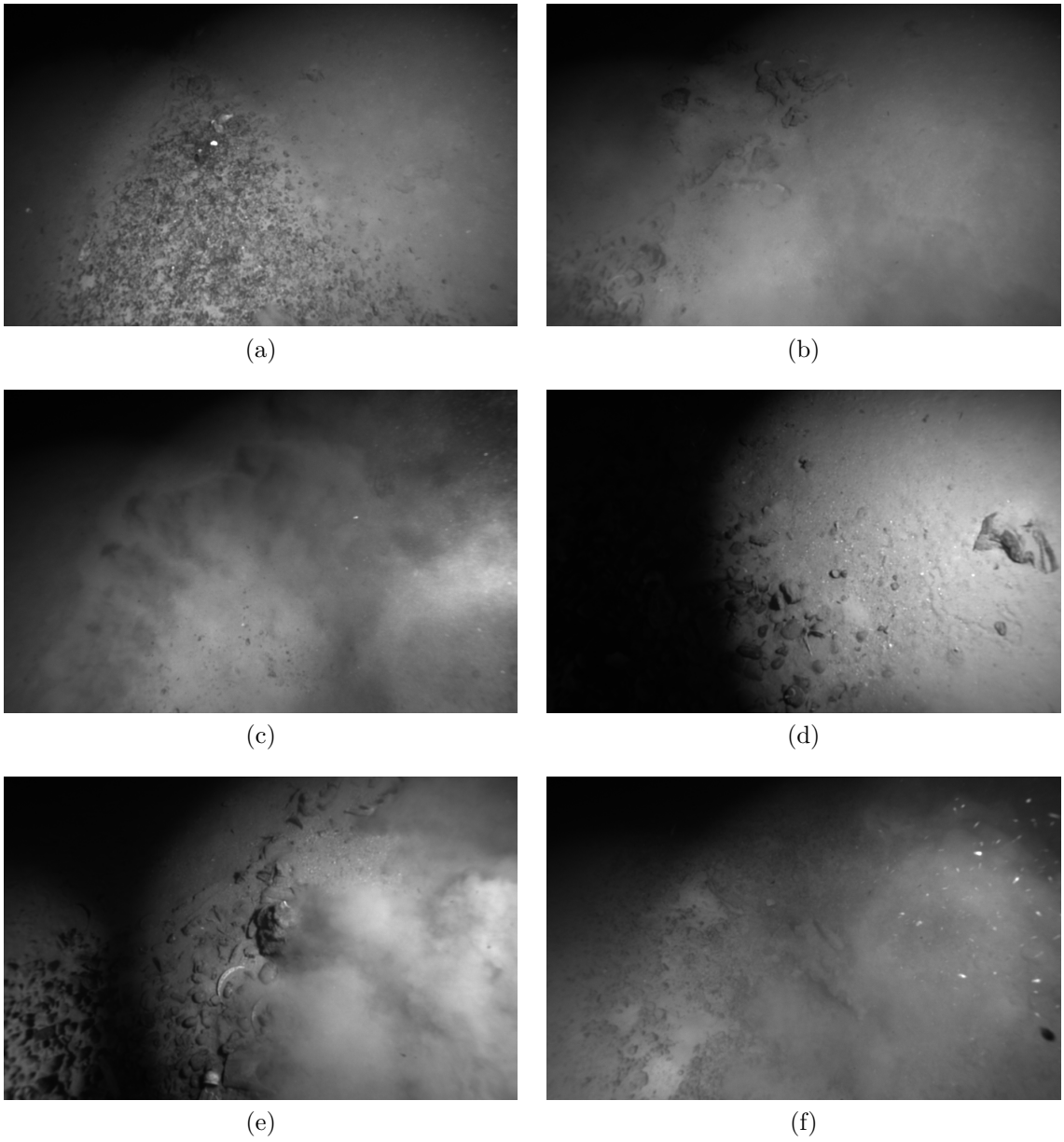


Figure 6.12: Sample images from AQUALOC Archaeological sites sequences, affected by sandy cloud, low and repetitive texture, and lack of light and features.

sequences, e.g., sequence 4 and 7, the ground truth trajectories were not continuous, hence only provide partial information on the camera poses. We ran SVIn2 on the two archaeological sites at a depth of approximately 270 meters and 380 meters respectively. SVIn2 was able to generate complete trajectories in all of the sequences without losing track, the RMSE error typically around 2% of its length, with the

Table 6.3 The RMSE in meters for each AQUALOC Archeological sites sequences.

Sequence #	1	2	3	4	5
SVIn RMSE(m)	1.0814	2.4403	0.2801	0.1983	2.7213
Error %	3.33	3.79	2.617	1.09	6.48
Sequence #	6	7	8	9	10
SVIn RMSE(m)	0.6085	1.0526	0.2465	1.5092	2.3710
Error %	1.91	0.86	0.59	2.30	2.83

lowest as in sequence 8 with an error of the 0.5% or highest as in sequence 5 with 6.4%, shown in Table 6.3.

In Fig. 6.13, Fig. 6.14, and Fig. 6.15, we scale align the estimated trajectories from SVIn and provided ground truth for all the sequences in Archaeological sites datasets except for sequence 4, as the provided ground truth for this dataset is highly discontinuous and hence has been plotted with their own scale.

6.5 Experimental Results of the Contour Based Reconstruction

In the following, we present, first, preliminary experiments with DSO [20], a direct method based visual odometry system showing the problem with photometric consistency, and, second, a qualitative result of the proposed approach in underwater environment.

6.5.1 Comparison with DSO

Fig. 6.16 shows the result of DSO in the underwater cave dataset in two different runs, Fig. 6.16a and Fig. 6.16b. DSO did not track for the full length of cave, instead it was able to keep track just for a small segment due to the variation of the light and hence violating the *photometric consistency* assumption of a direct method. Also, the *initialization* method is critical as it requires mainly translational movement and a very small rotational change due to the fact that it is a pure monocular visual

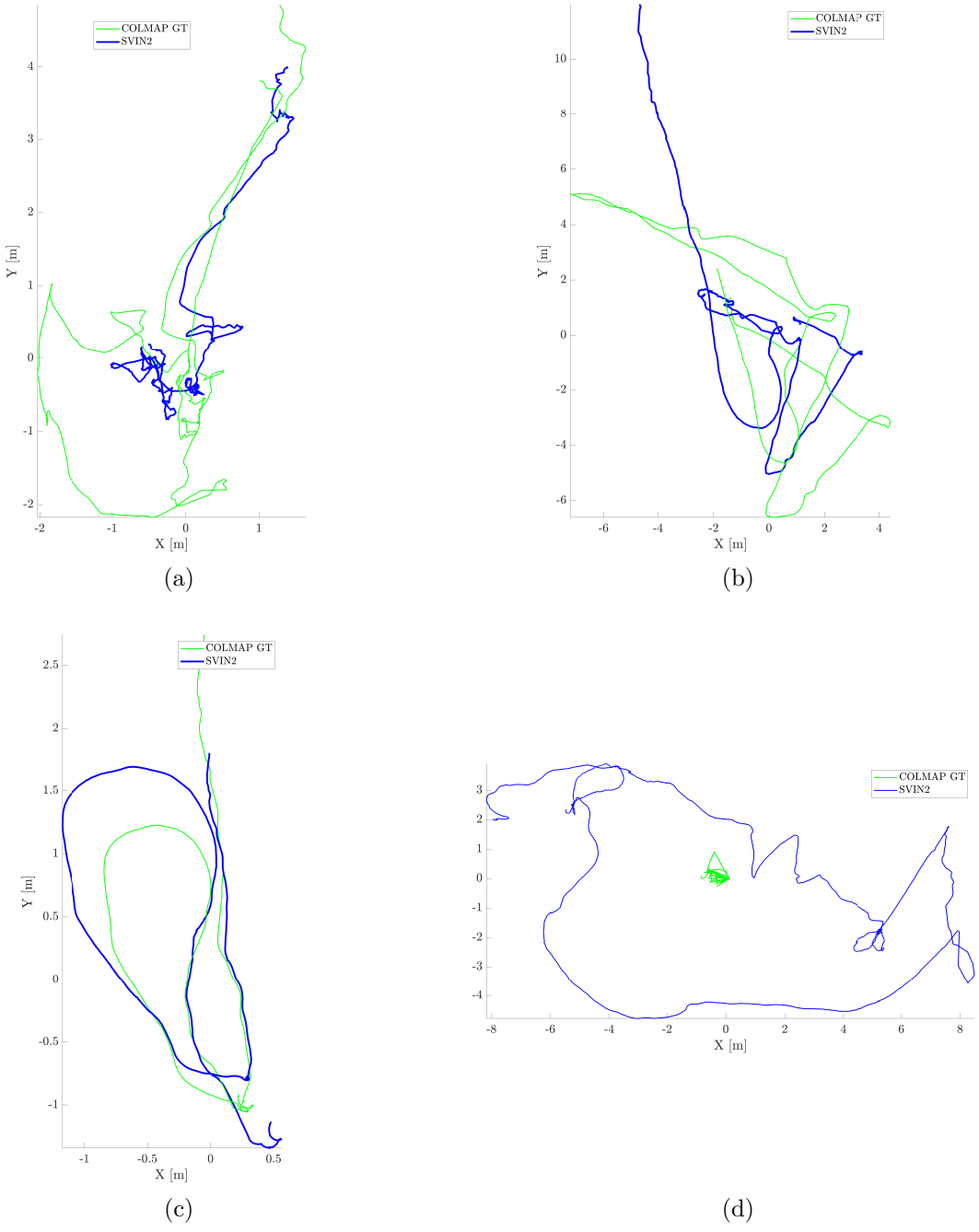


Figure 6.13: SVIn2 trajectories and ground truth alignment for Archaeological sequences 1 - 4 in (a) - (d) respectively.

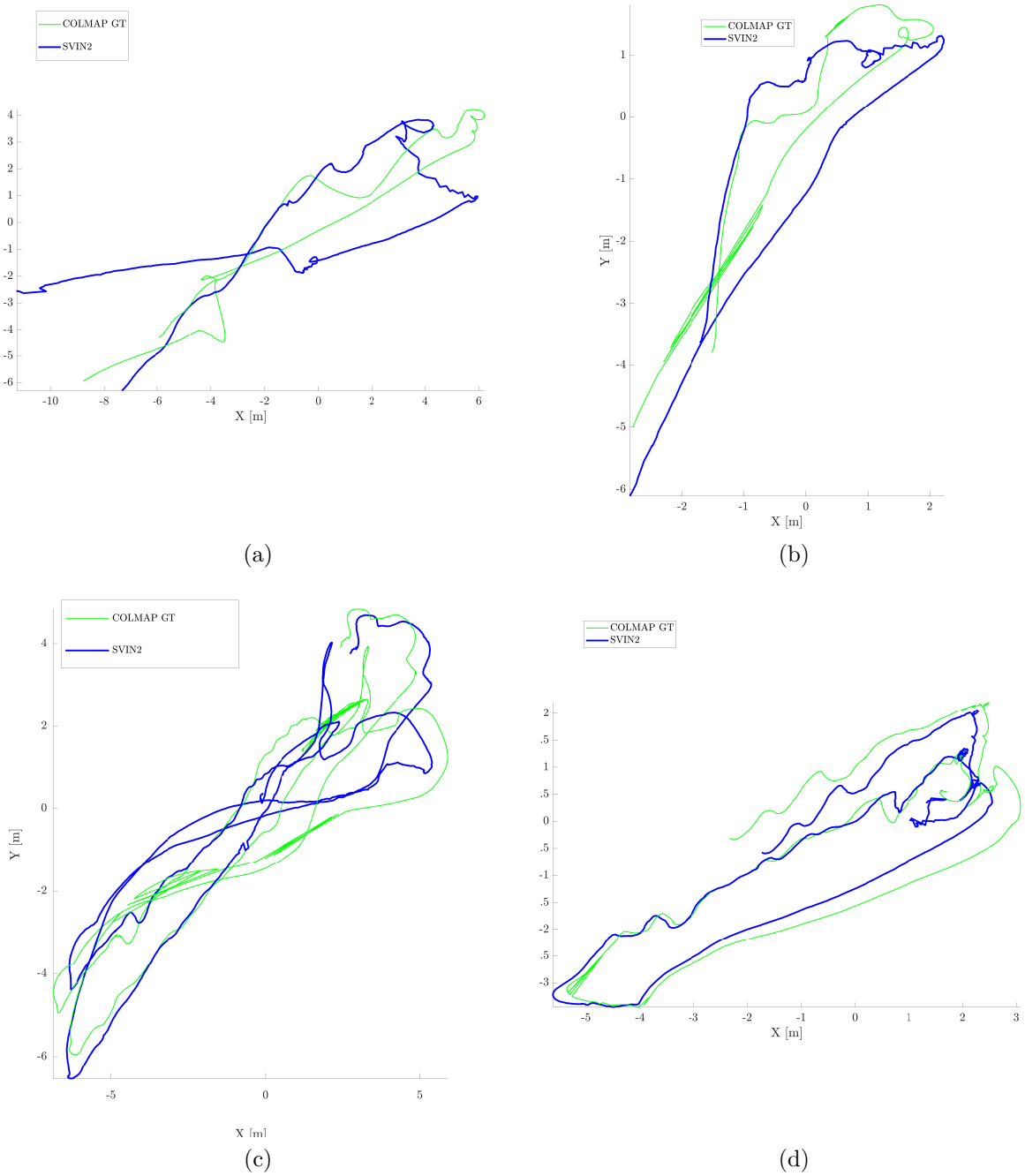


Figure 6.14: SVIn2 trajectories and ground truth alignment for Archaeological sequences 5 - 8 in (a) - (d) respectively.

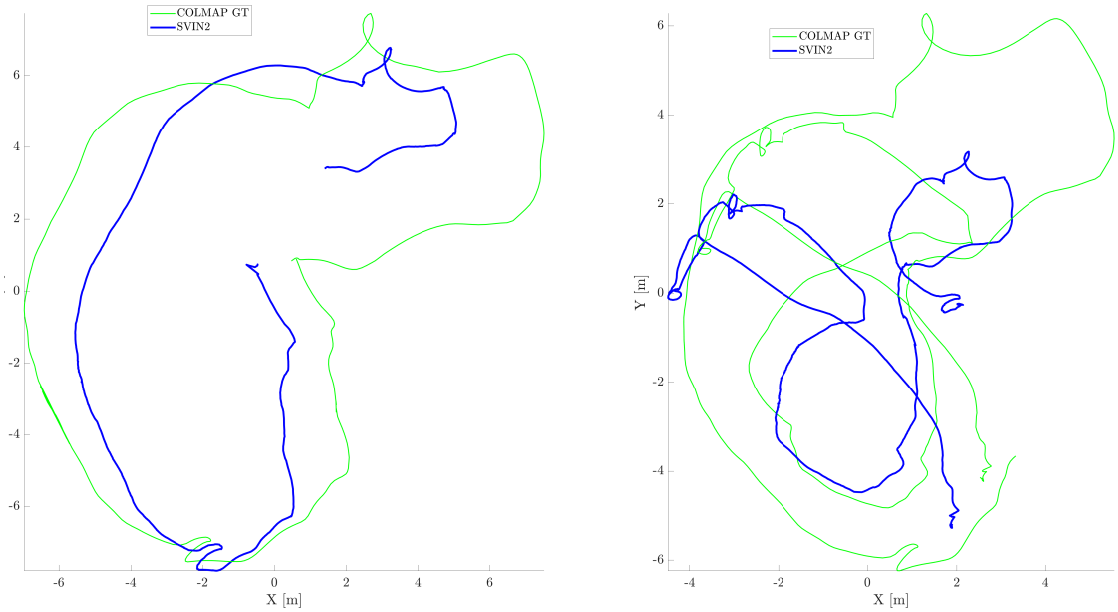


Figure 6.15: SVIn2 trajectories and ground truth alignment for Archaeological sequences 9 and 10 in (a) and (b) respectively.

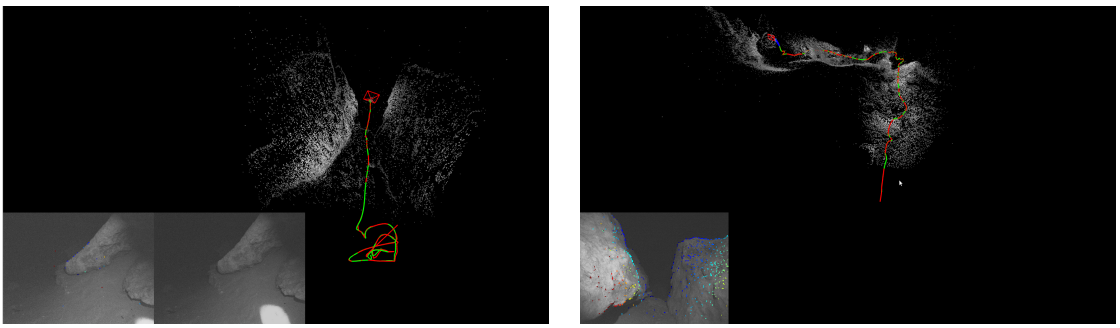
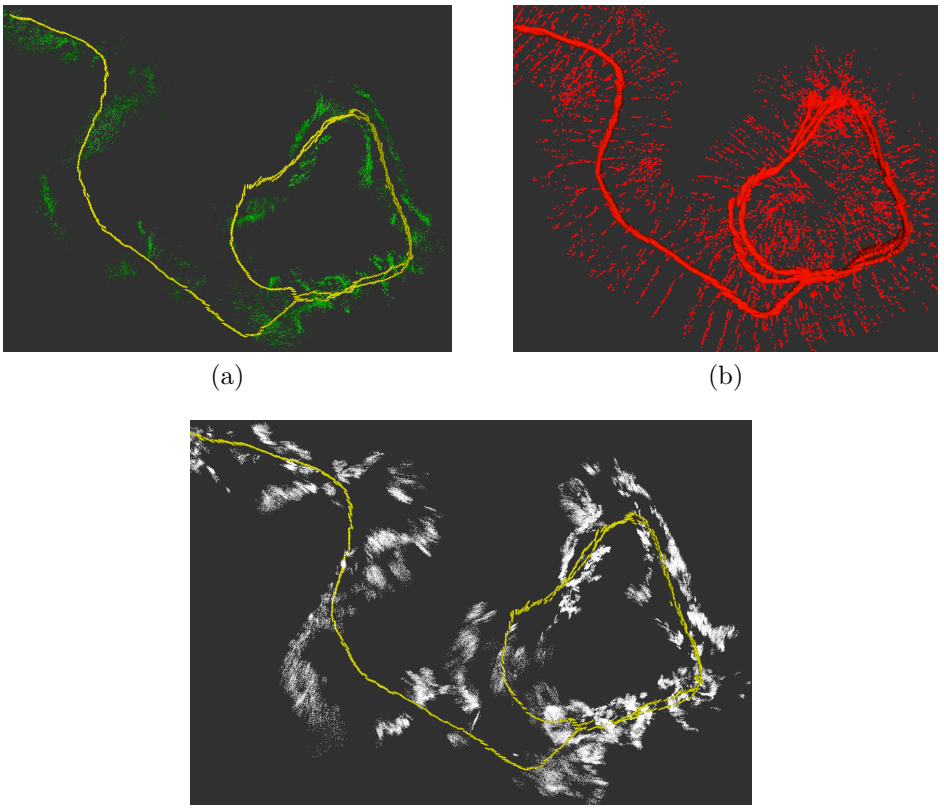


Figure 6.16: Partial trajectories generated by DSO. (a) Incorrect odometry and failing to track just after a few seconds and (b) longer trajectory after starting at a place with better illumination which also fails later on.

SLAM. We ran DSO with different starting points of the dataset to have a better initialization; the best one can be seen Fig. 6.16b – eventually failed too because of the poor lighting.



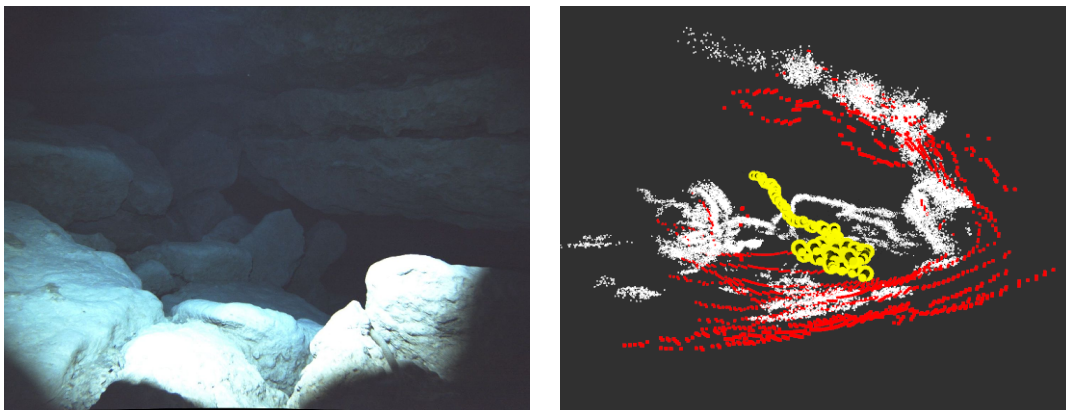
(a) (b)

(c)

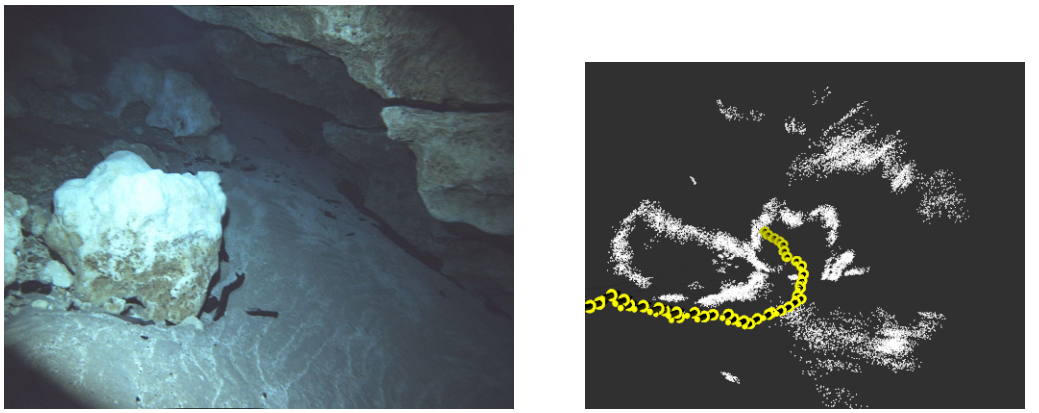
Figure 6.17: (a) Odometry using only a few strong features (green) for tracking. (b) Scanning Sonar measurements (red) aligned along the trajectory. (c) Reconstruction of the cave using the edges detected in the stereo contour points (gray).

6.5.2 Odometry and 3D Cave-Wall Reconstruction

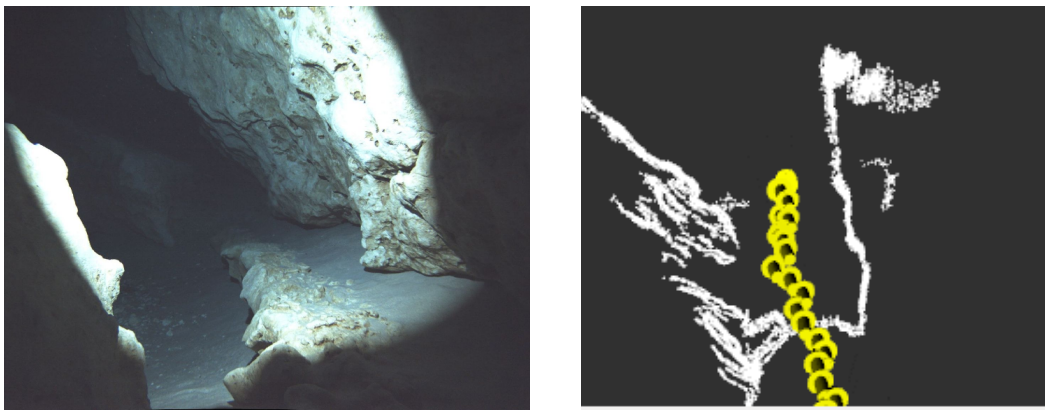
The ballroom at Ginnie Springs, FL, is a cavern open to divers with no cave-diving training. It provides a safe locale to collect data in an underwater cave environment. From entering the cavern at a depth of seven meters, the sensor was taken down to fifteen meters, and then a closed loop trajectory was traversed three times. As there is no ground truth available underwater, such as a motion capture system, we validate our approach from the information collected by the divers during the data collection procedure. The length of the trajectory produced by our method is 87 meters, consistent with the measure from the divers.



(a) (b)



(c) (d)

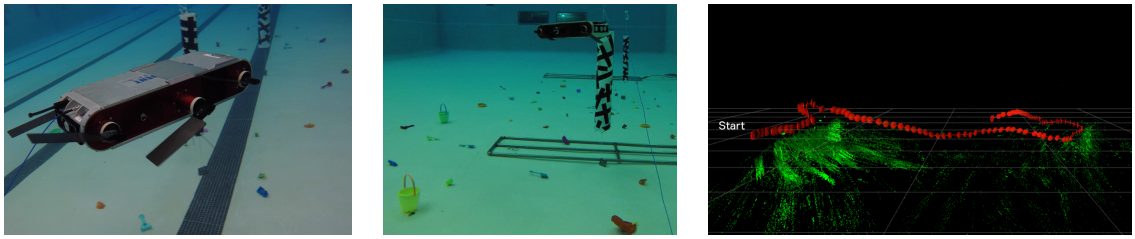


(e) (f)

Figure 6.18: Stereo contour reconstruction results in (b), (d), (f) and the corresponding images in (a), (c), (e) respectively.

Fig. 6.17 shows the whole trajectory with the different point clouds generated by the features used for tracking, Sonar data, and stereo contour matching. Keeping a small set of features for only tracking helps to run the whole system in real-time. As shown in the figure, Sonar provides a set of sparse but robust points using *range* and *head_position* information. Finally, the stereo contour matched point generates a denser point-cloud to represent the cave environment. Fig. 6.18 highlights some specific sections of the cavern, with the image and the corresponding reconstruction – in gray, the points from the contours; in red the points from the Sonar. As it can be observed, our proposed method enhances the reconstruction with a dense point cloud; for example rocks and valleys are clearly visible in Fig. 6.18.

6.6 Applications



(a) (b) (c)

Figure 6.19: (a),(b) show representative photos from the deployments in the pool in an unknown environment. (a) Avoiding two obstacles in the shallow swimming pool; (b) Avoiding two obstacles in the deep diving pool. (c) presents the online map produced by SVIn as a screenshot of RViz for the environments of (b), the robot avoid the first cylinder, moves forward and then avoids the second, while using the features from the bottom of the pool to localize.

SVIn enabled other robotics applications including 3D planning for an autonomous underwater robot in the presence of obstacles [103] to navigate safely in challenging environments of substantial complexity, with respect to other studies in the past, even without utilizing a known hydro-dynamics model. The proposed framework introduces online planning for the Aqua2 hexapod underwater robot with real-time

re-planning of up to 2 Hz, with using limited on-board computing resources, working in parallel with SVIn. During operations in an unknown environment, more accurate localization is useful to detect nearby obstacles, as such the resulting point-cloud produced by SVIn enables the AUV to navigate safely around obstacles. Fig. 6.19 shows pool experiments with additional sand-toys weighted and placed on the floor to improve the odometry estimation together with obstacles to test obstacle avoidance.

Chapter 7

Conclusions

In this dissertation, the problem of Simultaneous Localization and Mapping in underwater environments, combining visual, inertial, acoustic, and pressure information has been investigated. We focused on the design and development of a robust and accurate system that exploits the complementarity of different sensors so that robots can operate autonomously in very harsh environments with robustness, safety, and reliability to accomplish a task in real-time with limited computational resources. As vision based stated estimation achieves a certain degree of maturity, more sensors are being integrated for higher performance accuracy. Extending the well studied problem of Visual Inertial integration, we introduced a new sensor, a mechanical scanning sonar, which returns range measurements based on acoustic information. While the primary motivation of our work has been the mapping of underwater caves, the technique was tested in different environments, including the a shipwreck at the clear waters of Barbados, to artificial wrecks in the lakes of the Carolinas.

In Chapter 3 we have presented SVIn, a tightly coupled keyframe-based SLAM system by integrating sonar, visual, inertial, and pressure measurements in a non-linear optimization framework. The system includes a method for robust initialization by two-step scale refinement, loop-closure, and relocalization capabilities as a failure recovery mechanism. During the different experiments, it became clear that a minimum visibility and clarity in the visual data is required for basic performance; however, the visibility in underwater data degraded to a degree not often seen in VO or VIO approaches. Moreover, the use of a strong video light while necessary in the

cave environment, it requires careful calibration of its position in order to not saturate the camera. Furthermore, different surfaces resulted in different reflectance properties of the acoustic signal; we are currently analyzing the sonar data to improve the quality. Experimental results in indoor, outdoor, and underwater environments with publicly available datasets together with collected datasets, proves the effectiveness of our system.

In Chapter 4 we extended SVIn by utilizing the cone of artificial light and well defined contours to build a denser 3D model of the environment in real-time. The integration of multiple sensors improves the quality of the estimation in addition to the density of the reconstruction. A variety of domains will be affected with underwater archaeology and speleology being the primary areas. The resulting technology has been integrated to existing AUVs and ROVs for improving their autonomous capabilities, including for the purpose of health monitoring of coral reefs combining a Convolutional Neural Network (CNN) for coral detection and the SLAM system for mapping and counting [61, 60]; and 3D planning for an autonomous underwater robot in the presence of obstacles.

Bibliography

[1] Alison Abbott. "Mexican skeleton gives clue to American ancestry". In: *Nature News* (May 2014). Springer Nature.

[2] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org. 2015.

[3] A Alvarez, A Caffaz, Andrea Caiti, G Casalino, E Clerici, F Giorgi, L Gualdesi, A Turetta, and R Viviani. "Folaga: a very low cost autonomous underwater vehicle for coastal oceanography". In: *International Federation of Automatic Control Congress (IFAC)*. 2005, pp. 31–36.

[4] Anqi Xu and contributors. *ueye_cam package*. https://github.com/anqixu/ueye_cam. Accessed: 2018-08-11. 2018.

[5] Autonomous Field Robotics Lab. *Stereo Rig Sensor Documentation*. https://afrl.cse.sc.edu/afrl/resources/StereoRigWiki/. Accessed: 2018-08-14. 2018.

[6] Hernán Badino, Akihiro Yamamoto, and Takeo Kanade. "Visual odometry by multi-frame feature integration". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2013, pp. 222–229.

[7] Chris Beall, Frank Dellaert, Ian Mahon, and Stefan B Williams. "Bundle adjustment in large-scale 3D reconstructions based on underwater robotic surveys". In: *MTS/IEEE OCEANS, Spain*. 2011, pp. 1–6.

[8] Fabio Bellavia, Marco Fanfani, and Carlo Colombo. "Selective visual odometry for accurate AUV localization". In: *Autonomous Robots* (2015), pp. 1–11.

[9] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback". In: *IEEE The International Journal of Robotics Research (IJRR)* 36 (2017), pp. 1053–1072. DOI: 10.1177/0278364917728574.

[10] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. "The EuRoC micro

aerial vehicle datasets". In: *IEEE The International Journal of Robotics Research (IJRR)* 35.10 (2016), pp. 1157–1163. DOI: 10.1177/0278364915620033.

[11] C. McKinlay. *Woodville Karst Plain Project (WKPP)*. URL:http://www.wkpp.org. Apr. 2015.

[12] Shi-Feng Chen and Jun-Zhi Yu. "Underwater cave search and entry using a robotic fish with embedded vision". In: *Chinese Control Conference (CCC)*. 2014, pp. 8335–8340.

[13] *Climate Change and Sea-Level Rise in Florida: An Update of "The Effects of Climate Change on Florida's Ocean and Coastal Resources."* Tech. rep. Tallahasee, FL: Florida Ocean and Coastal Council, 2010.

[14] Peter Corke, Carrick Detweiler, Matthew Dunbabin, Michael Hamilton, Daniela Rus, and Iuliu Vasilescu. "Experiments with underwater robot localization and tracking". In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2007, pp. 4556–4561.

[15] Mark Cummins and Paul Newman. "Appearance-only SLAM at large scale with FAB-MAP 2.0". In: *IEEE The International Journal of Robotics Research (IJRR)* 30.9 (2011), pp. 1100–1123.

[16] Mark Cummins and Paul Newman. "FAB-MAP: Probabilistic localization and mapping in the space of appearance". In: *IEEE The International Journal of Robotics Research (IJRR)* 27.6 (2008), pp. 647–665.

[17] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. "MonoSLAM: Real-time single camera SLAM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6 (2007), pp. 1052–1067.

[18] J. Delmerico and D. Scaramuzza. "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots". In: *International Conference on Robotics and Automation (ICRA)*. 2018.

[19] Gregory Dudek et al. "A Visually Guided Swimming Robot". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2005, pp. 1749–1754.

[20] Jakob Engel, Vladlen Koltun, and Daniel Cremers. "Direct sparse odometry". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3 (2018), pp. 611–625.

[21] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European Conference on Computer Vision*. Springer. 2014, pp. 834–849.

[22] Sheck Exley. *Basic Cave Diving: A Blueprint for Survival*. ISBN 99946-633-7-2. National Speleological Society Cave Diving Section, 1977.

[23] Maurice F Fallon, John Folkesson, Hunter McClelland, and John J Leonard. "Relocating underwater features autonomously using sonar-based SLAM". In: *IEEE Journal of Oceanic Engineering* 38.3 (2013), pp. 500–513.

[24] Maxime Ferrera. "Monocular Visual-Inertial-Pressure Fusion for Underwater Localization and 3D Mapping." PhD thesis. Université de montpellier, 2019.

[25] Maxime Ferrera, Vincent Creuze, Julien Moras, and Pauline Trouvé-Peloux. "AQUALOC: An underwater dataset for visual–inertial–pressure localization". In: *The International Journal of Robotics Research* 38.14 (2019), pp. 1549–1559.

[26] Mark Fiala. "Artag revision 1, a fiducial marker system using digital techniques". In: *National Research Council Publication* 47419 (2004), pp. 1–47.

[27] John Folkesson, John Leonard, Jacques Leederkerken, and Rob Williams. "Feature tracking for underwater navigation using sonar". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2007, pp. 3678–3684.

[28] D. C. Ford and P. W. Williams. *Karst Geomorphology and Hydrology*. Chapman & Hall, 1994.

[29] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Transactions on Robotics* 33.1 (2017), pp. 1–21.

[30] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems". In: *IEEE Transactions on Robotics* 33.2 (2017). DOI: 10.1109/TRO.2016.2623335.

[31] Friedrich Fraundorfer and Davide Scaramuzza. "Visual odometry: Part i: The first 30 years and fundamentals". In: *IEEE Robotics and Automation Magazine* 18.4 (2011), pp. 80–92.

[32] Friedrich Fraundorfer and Davide Scaramuzza. “Visual odometry: Part ii: Matching, robustness, optimization, and applications”. In: *IEEE Robotics & Automation Magazine* 19.2 (2012), pp. 78–90.

[33] Dorian Gálvez-López and Juan D Tardós. “Bags of binary words for fast place recognition in image sequences”. In: *IEEE Transactions on Robotics* 28.5 (2012), pp. 1188–1197.

[34] Marcus Gary, Nathaniel Fairfield, William C Stone, David Wettergreen, George Kantor, and John M Sharp Jr. “3D mapping and characterization of Sistema Zacatón from DEPTHX (DE ep P hreatic TH ermal e X plorer)”. In: *Sinkholes and the Engineering and Environmental Impacts of Karst*. 2008, pp. 202–212.

[35] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. “Vision meets Robotics: The KITTI Dataset”. In: *IEEE The International Journal of Robotics Research (IJRR)* 32.11 (2013), pp. 1231–1237.

[36] Philippe Giguere, Gregory Dudek, Christopher Prahacs, Nicolas Plamondon, and Katrine Turgeon. “Unsupervised learning of terrain appearance for automated coral reef exploration”. In: *Conference on Computer and Robot Vision (CRV)*. IEEE. 2009, pp. 268–275.

[37] B. Gulden. *WORLD LONGEST UNDERWATER CAVES*. URL:http://www.caverbob.com/uwcaves.htm. Apr. 2015.

[38] Strasdat Hauke, J. M. M. Montiel, and Andrew J. Davison. “Scale drift-aware large scale monocular SLAM”. In: *Proc. RSS* (2010).

[39] Jon Henderson, Oscar Pizarro, Matthew Johnson-Roberson, and Ian Mahon. “Mapping submerged archaeological sites using stereo-vision photogrammetry”. In: *International Journal of Nautical Archaeology* 42.2 (2013), pp. 243–256.

[40] Joel A Hesch, Dimitrios G Kottas, Sean L Bowman, and Stergios I Roumeliotis. “Observability-constrained vision-aided inertial navigation”. In: *University of Minnesota, Dept. of Comp. Sci. & Eng., MARS Lab, Tech. Rep* 1 (2012), p. 6.

[41] Marc Hildebrandt and Frank Kirchner. “Imu-aided stereo visual odometry for ground-tracking auv applications”. In: *MTS/IEEE Oceans, Sydney*. 2010, pp. 1–8.

[42] Andrew Hogue, Andrew German, and Michael Jenkin. “Underwater environment reconstruction using stereo and inertial data”. In: *IEEE International Conference on Systems, Man and Cybernetics*. IEEE. 2007, pp. 2372–2377.

[43] Andrew Howard. “Real-time stereo visual odometry for autonomous ground vehicles”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2008, pp. 3946–3952.

[44] Hordur Johannsson, Michael Kaess, Brendan Englot, Franz Hover, and John Leonard. “Imaging sonar-aided navigation for autonomous underwater harbor surveillance”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2010, pp. 4396–4403.

[45] Matthew Johnson-Roberson, Oscar Pizarro, Stefan B Williams, and Ian Mahon. “Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys”. In: *Journal Field Robotics* 27.1 (2010), pp. 21–51.

[46] Bharat Joshi, Brennan Cain, James Johnson, Michail Kalitazkis, Sharmin Rahman, Marios Xanthidis, Alan Hernandez, Alberto Quattrini Li, Nikolaos Vitzilaios, and Ioannis Rekleitis. “Experimental Comparison of open source Vision-Inertial-Based State Estimation Algorithms”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019).

[47] Bernd Kitt, Andreas Geiger, and Henning Lategahn. “Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme.” In: *Intelligent Vehicles Symposium*. 2010, pp. 486–492.

[48] G. Klein and D. Murray. “Parallel Tracking and Mapping for Small AR Workspaces”. In: *IEEE and ACM Int. Symp. on Mixed and Augmented Reality*. 2007, pp. 225–234. DOI: 10.1109/ISMAR.2007.4538852.

[49] Kurt Konolige, Motilal Agrawal, and Joan Sola. “Large-scale visual odometry for rough terrain”. In: *Robotics research*. Springer, 2010, pp. 201–212.

[50] N. Kresic and A. Mikszewski. *Hydrogeological Conceptual Site Models: Data Analysis and Visualization*. Boca Raton, LA: CRC Press, 2013.

[51] Kumar Robotics. *imu_3dm_gx4 package*. https://github.com/KumarRobotics/imu_3dm_gx4. Accessed: 2018-08-11. 2018.

[52] Ed Lane. *The Spring Creek Submarine Springs Group, Wakulla County, Florida*. Tech. rep. Special Publication 47. Tallahasee, Fl: Florida Geological Survey, 2001.

[53] Chong-Moo Lee et al. “Underwater navigation system based on inertial sensor and doppler velocity log using indirect feedback Kalman filter”. In: *International Journal of Offshore and Polar Engineering* 15.02 (2005).

[54] Jacques C Leedekerken, Maurice F Fallon, and John J Leonard. "Mapping complex marine environments with autonomous surface craft". In: *International Symposium on Experimental Robotics (ISER)*. 2014, pp. 525–539.

[55] John J Leonard and Hugh F Durrant-Whyte. *Directed sonar sensing for mobile robot navigation*. Vol. 175. Springer Science & Business Media, 2012.

[56] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. "Keyframe-based visual-inertial odometry using nonlinear optimization". In: *IEEE The International Journal of Robotics Research (IJRR)* 34.3 (2015), pp. 314–334.

[57] Angelos Mallios et al. "Toward autonomous exploration in confined underwater environments". In: *Journal Field Robotics* 33.7 (2016), pp. 994–1012.

[58] David Meger, Juan Camilo Gamboa Higuera, Anqi Xu, Philippe Giguere, and Gregory Dudek. "Learning legged swimming gaits from experience". In: *International Conference on Robotics and Automation (ICRA)*. 2015, pp. 2332–2338.

[59] Paul Merrell, Amir Akbarzadeh, Liang Wang, Philippos Mordohai, Jan-Michael Frahm, Ruigang Yang, David Nistér, and Marc Pollefeys. "Real-time visibility-based fusion of depth maps". In: *IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2007, pp. 1–8.

[60] Md Modasshir, Sharmin Rahman, and Ioannis Rekleitis. "Autonomous 3D Semantic Mapping of Coral Reefs". In: *12th Conference on Field and Service Robotics (FSR), Tokyo, Japan*. 2019.

[61] Md Modasshir, Sharmin Rahman, Oscar Youngquist, and Ioannis Rekleitis. "Coral Identification and Counting with an Autonomous Underwater Vehicle". In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2018, pp. 524–529.

[62] Anastasios I Mourikis and Stergios I Roumeliotis. "A multi-state constraint Kalman filter for vision-aided inertial navigation". In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2007, pp. 3565–3572.

[63] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. "ORB-SLAM: A Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.

[64] Raúl Mur-Artal and Juan D Tardós. "Visual-inertial monocular SLAM with map reuse". In: 2.2 (2017), pp. 796–803.

[65] Kenton Oliver, Weilin Hou, and Song Wang. “Image feature detection and matching in underwater conditions”. In: *Ocean Sensing and Monitoring II*. Vol. 7678. International Society for Optics and Photonics. 2010, 76780N.

[66] Taragay Oskiper, Zhiwei Zhu, Supun Samarasekera, and Rakesh Kumar. “Visual odometry system using multiple stereo cameras and inertial measurement unit”. In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2007, pp. 1–8.

[67] Tong Qin, Peiliang Li, and Shaojie Shen. “VINS-Mono: A robust and versatile monocular visual-inertial state estimator”. In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.

[68] A. Quattrini Li, A. Coskun, S. M. Doherty, S. Ghasemlou, A. S. Jagtap, M. Modasshir, S. Rahman, A. Singh, M. Xanthidis, J. M. O′Kane, and I. Rekleitis. “Vision-Based Shipwreck Mapping: on Evaluating Features Quality and Open Source State Estimation Packages”. In: *MTS/IEEE OCEANS - Monterrey*. Sept. 2016, pp. 1–10.

[69] Alberto Quattrini Li, Adem Coskun, Sean M. Doherty, Shervin Ghasemlou, Apoorv S. Jagtap, MD Modasshir, Sharmin Rahman, Akanksha Singh, Marios Xanthidis, Jason M. O’Kane, and Ioannis Rekleitis. “Experimental Comparison of open source Vision based State Estimation Algorithms”. In: *International Symposium on Experimental Robotics (ISER)*. 2016.

[70] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. “ROS: an open-source Robot Operating System”. In: *International Conference on Robotics and Automation (ICRA) workshop on open source software*. Vol. 3. 3.2. Kobe, Japan. 2009, p. 5.

[71] Sharmin Rahman, Nare Karapetyan, Alberto Quattrini Li, and Ioannis Rekleitis. “A Modular Sensor Suite for Underwater Reconstruction”. In: *MTS/IEEE Oceans Charleston*. 2018, pp. 1–6.

[72] Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. “Sonar Visual Inertial SLAM of Underwater Structures”. In: *International Conference on Robotics and Automation (ICRA)*. 2018.

[73] Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. “SVIn2: An Underwater SLAM System using Sonar, Visual, Inertial, and Depth Sensor”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019.

[74] Research group of Prof. Kostas Daniilidis. *Monocular MSCKF ROS node*. https://github.com/daniilidis-group/msckf_mono. 2018.

[75] Kristof Richmond, Chris Flesher, Laura Lindzey, Neal Tanner, and William C Stone. "SUNFISH®: A human-portable exploration AUV for complex 3D environments". In: *MTS/IEEE OCEANS Charleston*. 2018, pp. 1–9.

[76] Paul Rigby, Oscar Pizarro, and Stefan B Williams. "Towards geo-referenced AUV navigation through fusion of USBL and DVL measurements". In: *MTS/IEEE Oceans*. 2006, pp. 1–6.

[77] Chris Roman, Oscar Pizarro, Ryan Eustice, and Hanumant Singh. "A new autonomous underwater vehicle for imaging research". In: *MTS/IEEE OCEANS Conference and Exhibition*. Vol. 1. 2000, pp. 153–156.

[78] Juan Manuel Sáez, Andrew Hogue, Francisco Escolano, and Michael Jenkin. "Underwater 3D SLAM through entropy minimization". In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2006, pp. 3562–3567.

[79] Joaquim Salvi, Yvan Petillo, Stephen Thomas, and Josep Aulinas. "Visual SLAM for underwater vehicles using video velocity log and natural landmarks". In: *MTS/IEEE OCEANS*. 2008, pp. 1–6.

[80] Junaed Sattar, Eric Bourque, Philippe Giguere, and Gregory Dudek. "Fourier tags: Smoothly degradable fiducial markers for use in human-robot interaction". In: *Conference on Computer and Robot Vision (CRV)*. 2007, pp. 165–174.

[81] Junaed Sattar, Gregory Dudek, Olivia Chiu, Ioannis Rekleitis, Philippe Giguere, Alec Mills, Nicolas Plamondon, Chris Prahacs, Yogesh Girdhar, Meyer Nahon, and John-Paul Lobos. "Enabling Autonomous Capabilities in Underwater Robotics". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2008, pp. 3628–3634.

[82] Johannes Lutz Schönberger and Jan-Michael Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[83] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *European Conference on Computer Vision (ECCV)*. 2016.

[84] Jianbo Shi et al. "Good features to track". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 1994, pp. 593–600.

[85] Florian Shkurti, Ioannis Rekleitis, and Gregory Dudek. “Feature Tracking Evaluation for Pose Estimation in Underwater Environments”. In: *Canadian Conference on Computer and Robot Vision (CRV)*. St. John, NF Canada, 2011, pp. 160–167.

[86] Florian Shkurti, Ioannis Rekleitis, Milena Scaccia, and Gregory Dudek. “State estimation of an underwater robot using visual and inertial information”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011, pp. 5054–5060.

[87] S. Skaff, J.J. Clark, and Ioannis Rekleitis. “Estimating Surface Reflectance Spectra for Underwater Color Vision”. In: *British Machine Vision Conference (BMVC)*. Leeds, U.K., Sept. 2008, pp. 1015–1024.

[88] Noah Snavely, Steven M Seitz, and Richard Szeliski. “Photo tourism: exploring photo collections in 3D”. In: *ACM transactions on graphics (TOG)*. Vol. 25. 3. 2006, pp. 835–846.

[89] Jeff Snyder. “Doppler Velocity Log (DVL) navigation for observation-class ROVs”. In: *MTS/IEEE Oceans, Seattle*. 2010, pp. 1–9.

[90] Stone Aerospace. *Digital Wall Mapper*. URL:http://stoneaerospace.com/digital-wall-mapper/. Apr. 2015.

[91] W. C. Stone. “Design and Deployment of a 3-D Autonomous Subterranean Submarine Exploration Vehicle”. In: *International Symposium on Unmanned Untethered Submersible Technologies (UUST)*. 512. 2007.

[92] Hauke Strasdat. “Local accuracy and global consistency for efficient visual SLAM”. PhD thesis. Citeseer, 2012.

[93] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. “A Benchmark for the Evaluation of RGB-D SLAM Systems”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2012, pp. 573–580.

[94] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar. “Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 965–972. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2793349.

[95] Juan José Tarrio and Sol Pedre. “Realtime Edge Based Visual Inertial Odometry for MAV Teleoperation in Indoor Environments”. In: *Journal of Intelligent & Robotic Systems* (2017), pp. 235–252. DOI: 10.1007/s10846-017-0670-y.

[96] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. "Bundle adjustment – a modern synthesis". In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372.

[97] Shinji Umeyama. "Least-Squares Estimation of Transformation Parameters Between Two Point Patterns". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.4 (1991), pp. 376–380. ISSN: 0162-8828. DOI: 10.1109/34.88573.

[98] Nicholas Weidner, Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. "Underwater Cave Mapping using Stereo Vision". In: *International Conference on Robotics and Automation (ICRA)*. Singapore, May 2017, pp. 5709–5715.

[99] Eric Westman and Michael Kaess. *Underwater AprilTag SLAM and calibration for high precision robot localization*. Tech. rep. CMU-RI-TR-18-43. Carnegie Mellon University, 2018.

[100] Stephan Wirth, Pep Lluis Negre Carrasco, and Gabriel Oliver Codina. "Visual odometry for autonomous underwater vehicles". In: *MTS/IEEE Oceans, Bergen*. 2013, pp. 1–6.

[101] Changchang Wu. "Towards linear-time incremental structure from motion". In: *2013 International Conference on 3D Vision-3DV 2013*. IEEE. 2013, pp. 127–134.

[102] X. Wu, R. E. Stuck, I. Rekleitis, and J. M. Beer. "Towards a Framework for Human Factors in Underwater Robotics". In: *Human Factors and Ergonomics Society International Annual Meeting*. 2015, pp. 1115–1119.

[103] Marios Xanthidis, Nare Karapetyan, Hunter Damron, Sharmin Rahman, James Johnson, Allison O'Connell, Jason O'Kane, and Ioannis Rekleitis. "Navigation in the Presence of Obstacles for an Agile Autonomous Underwater Vehicle". In: *International Conference on Robotics and Automation (ICRA)*. 2020.

[104] Zexuan Xu, Seth Willis Bassett, Bill Hu, and Scott Barrett Dyer. "Long distance seawater intrusion through a karst conduit network in the Woodville Karst Plain, Florida". In: *Scientific Reports* 6 (Aug. 2016), pp. 1–10.