Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

Erlend Røilid Vollan

# Visual Simultaneous Localization and Mapping Applied on Work Class ROVs

Master's thesis in Marine Technology

Supervisor: Martin Ludvigsen

July 2020

NTNU
Norwegian University of
Science and Technology

Erlend Røilid Vollan

# Visual Simultaneous Localization and Mapping Applied on Work Class ROVs

Master's thesis in Marine Technology
Supervisor: Martin Ludvigsen
July 2020

Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

NTNU
Norwegian University of
Science and Technology

**NTNU Trondheim**
**Norwegian University of Science and Technology**
*Department of Marine Technology*

# MASTER THESIS IN MARINE CYBERNETICS

## SPRING 2020

## FOR

## STUD. TECHN. ERLEND RØILID VOLLAN

Visual Simultaneous Localization and Mapping Applied on Work Class ROVs

**Work description**

This Master's Thesis aims to investigate the possibilities of increasing the local situation awereness and local position accuracy of Work Class Remotely Operated Underwater Vehicles (WC-ROV) by utilizing the Visual Simultaneous Localization and Mapping (VSLAM) algorithm ORB-SLAM2. Increasing the situtational awereness and local position accuracy of WC-ROVs could contribute to the development of autonomous solution, proving safer and more efficient operation of the WCROV in its wide range of underwater application areas.
It was concluded in the Project Thesis of the autumn of 2019 that ORB-SLAM2 showed possibilities of increasing local position accuracy of Work class ROVs. The Project Thesis established requirements for sensors and test data. In this Master's Thesis the investigation will be based upon the continuation of the Project Thesis work.

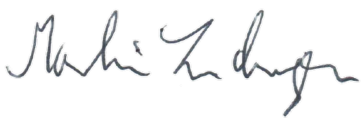
**Scope of work**

1. Review necessary literature within the fields of:
    a. Underwater VSLAM
    b. Underwater imaging process
    c. Camera calibration
2. Perform underwater calibration of the stereo camera rig of the WC-ROV Minerva in the Marine Cybernetics Laboratory (MCLab).
3. Investigate and adapt the VSLAM algorithm ORB-SLAM2 to account for underwater imaging effects.
4. Propose an real-time implementation in ROS (Robot Operating System) using the adapted ORB-SLAM2 algorithm.
5. Propose a obstacle detection algorithm that utilizes the created pointcloud and/or the estimated path produced by ORB-SLAM2.
6. Perform tests in MCLab on the real-time ROS implementation using the stereo camera rig. Both under ideal and simulated deepsea conditions.
7. Conduct field tests with the real-time ROS implementation using Minerva on an expedition with the research vessel Gunnerus
8. Analyse and compare the results from the field tests and the tests in MCLab.

The report shall be written in English and edited as a research report including literature survey, description of mathematical models, description of implementation, test results, discussion and a conclusion including a proposal for further work. Source code should be provided. It is supposed that Department of Marine Technology, NTNU, can use the results freely in its research work, unless otherwise agreed upon, by referring to the student’s work.

The thesis should be submitted within 10 June.

Professor Martin Ludvigsen, Supervisor



# Abstract

This thesis investigates the possibility of using Visual Simultaneous Localization and Mapping (VSLAM) to increase the local situational awareness and local positioning accuracy of Work Class Remotely Operated Vehicles (WC-ROV). The existing methods covering the local situational awareness and local positioning are mainly acoustic systems and inertial navigation, each having different trade-offs in terms of accuracy, cost and complexity. The motivation of this thesis, is to contribute in development of autonomous solutions on WC-ROVs by providing low-cost and accurate alternative to the existing methods.

The main contribution of this thesis is a proposed real-time WC-ROV VSLAM system based on using the stereo camera rig of NTNU's WC-ROV Minerva and the VSLAM method ORB-SLAM2. The system accounts for underwater imaging effects, provides estimates of the WC-ROV position, orientation and point cloud of local environment, detects the closest observed obstacle, and conveys the closest detected obstacle to the Autonomy Framework of Minerva. The real-time WC-ROV VSLAM system was implemented in the framework Robot Operating System (ROS), using the programming language C++. The baseline of the stereo camera rig was set to 0.2 m based on calculations on the stereo overlapping field of view and expected disparity values of matched features in the left and right stereo image pair. The system uses Contrast Limited Adaptive Histogram Equalization (CLAHE) to contrast enhance the unevenly illuminated underwater stereo image pairs received from the stereo camera rig, ORB-SLAM2 to estimate position, orientation and point cloud of the surrounding environment, plane fitting with Random Sample Consensus (RANSAC) and an Euclidean based clustering method to infer the closest detected obstacle, and communicates and provides the global coordinate of the closest detected obstacle to the Autonomy Framework of Minerva using TCP connection.

The system was tested in an underwater obstacle course in the Marine Cybernetics Lab (MC-lab) at NTNU, both under ideal light conditions and subsea simulated lighting conditions, using the full resolution capacities of the stereo rig cameras and in a halved resolution mode with increased light sensitivity. The optical measurement system Qualisys was used as ground truth for the position estimates, the measured dimensions of the obstacle course were used as ground truth for the estimated map and closest detected obstacle algorithm. Prior to the experiment, the stereo camera rig was camera calibrated under water at the distances 1, 3, 4 and 5 m establishing relevant camera parameters to be used in ORB-SLAM2. The integration of the system in the Autonomy Framework of Minerva was tested doing Hardware In the Loop-testing with an altered version of the system generating synthetically obstacles instead of true obstacles from visual inputs.

The results from the underwater obstacle course tests showed that the position estimates of the real-time WC-ROV VSLAM system provided good accuracy in local areas in short times intervals, but error accumulated in the estimated positions when the stereo camera rig explored larger areas of the environment. The estimated maps provided adequate spatial relations with some inconsistency of previously and newly mapped obstacles. The closest obstacle detection managed to detect and infer the closest obstacles, but the performance reduced in the subsea simulated test cases due to increased noise levels and misalignments in the estimated map. Additionally, showed the results that the benefits of using cameras in full resolution was inferior to the binned mode due to reduced estimation frequency of ORB-SLAM2.

The thesis concluded with that the use of the VSLAM method ORB-SLAM2 in the real-time WC-ROV VSLAM system showed that the local situational awareness could be increased by using the estimated position and map of ORB-SLAM2, and that they could be used in autonomous features of the WC-ROV such as for example the proposed closest obstacle detection. The use of the estimated position to increase the local positioning, was however more questionable due to the increased drift occurring and jumps in the estimated positions due to relocalization and loop-closures.

# Sammendrag

Denne oppgaven undersøker muligheten for å bruke Visuell Simultan Lokalisering og Kartlegging (VSLOK) til å øke den lokale situasjonsbevistheten og lokale posisjonsnøyaktighet til fjernstyrte undervannsarbeidsfarkoster (FUAF). Eksisterende metoder som dekker lokal situasjonsbevistheten og lokal posisjoneringen, er hovedsakelig akustiske systemer og treghetsnavigasjon som begge har ulike kompromisser i henhold til nøyaktighet, kostnad og kompleksitet. Motivasjonen for denne oppgaven er å bidra til utviklingen av autonome løsninger til FUAF ved å komme med et lavkostnads og nøyaktig alternativ til de eksisterende metodene.

Hoved bidraget i denne oppgaven er et foreslått sanntids FUAF-VSLOK system basert på stereokamerariggen til NTNUs FUAF Minerva og VSLOK metoden ORB-SLAM2. Systemet tar hensyn til virkningen som oppstår av bilder tatt under vann, gir ut estimater på posisjon, orientering og en punktsky av de lokale omgivelsene til Minerva, detekterer nærmeste observerte hindring, og videreformidler den nærmeste detektere hindringen til det Autonomi Rammeverket til Minerva. Sanntids FUAF-SLOK systemet ble implementert i rammeverket Robot Operativ System (ROS) ved å bruke programmeringsspråket C++. Avstanden mellom kameraene i stereokamerariggen ble satt til 0.2 m basert på beregninger gjort på det stereo-overlappende synsfeltet og forventet misforholdsverdier til korresponderende nøkkelpunkter i de venstre og høyre stereo bildene. Systemet bruker Kontrast Begrenset Adaptiv Histogram Utgjevning (KBAH) for å forbedre kontrasten i de ujevnt belyste undervannsstereobildeparene mottatt fra stereokamerariggen, ORB-SLAM2 for å estimere posisjon, orientering og punktsky av omgivelsene, plantilpasning av punkter med Tilfeldig Sampling Konsensus (TSK) og en Euklidsk basert klyngemetode for å avgjøre den nærmeste detekterte hindring, og kommuniserer og formidler det globale koordinatet av den nærmeste detekterte hindringen til Autonomi Rammeverket til Minerva ved å bruke TCP tilkobling.

Systemet ble testet in en undervannshinderløype i Marin Kybernetikk Laboratoriumet Lab på NTNU, både med ideell belysning og med dyphavs simulert belysning, ved å bruke den fulle oppløsningskapasiteten til stereokamerariggen og i en halvert oppløsningsmodus med forbedret lyssensitivitet. Det optiske målesystemet Qualisys ble brukt som referansemåling til posisjonsestimatene, de målte dimensjonene på undervannshinderløypen ble brukt som referansemåling til det estimerte kartet av omgivelsene og den nærmest detekterte hindringsalgoritmen. I forkant av eksperimentet, ble stereokamerariggen kamerakalibrert under vann på avstandene 1, 2, 4 og 5 m for å etablere de relevante kamera parameterne brukt i ORB-SLAM2. Integrasjonen av systemet i Autonomi Rammever-

ket til Minerva ble testet ved å foreta Fastvare I Løkken (FIL) med en endret versjon av sanntids FUAF VSLOK systemet som genererte syntetiske hindringer istedenfor ekte hindringer observert fra visuell inndata.

Resultatene fra undervannshinderløypen viste at posisjonsestimatet fra sanntids FUAF VSLOK systemet gav god nøyaktighet i lokale områder i korte tidsintervall, men feilen akkumulerte i estimatet etter hvert som stereokamerariggen utforsket større områder av omgivelsene. Det estimerte punktskykartet gav adekvat romlig sammenheng med noen uregelmessigheter mellom tidligere og nye kartlagte hindringer. Den nærmeste hindring detekteringsalgoritmen klarte å detektere og fastslå nærmeste hindring, men ytelsen ble redusert i de dyphavs simulerte test scenarioene på grunn av økt støynivå og forskyvninger i det estimerte kartet. I tillegg, viste resultatene at fordelene med å bruke kameraene i full oppløsning var mindre enn i den lyssensitive modusen på grunn av redusert estimeringsfrekvens i ORB-SLAM2.

Denne oppgaven konkluderte med at bruken av VSLOK metoden ORB-SLAM2 i sanntids FUAF VSLOK systemet viste at den lokale situasjonsbevistheten kunne bli økt ved å bruke den estimerte posisjonen og kartet fra ORB-SLAM2, og at de kan bli brukt i utviklingen av autonome egenskaper i FUAF'er slik som for eksempel den foreslåtte nærmeste hindrings detekteringsalgoritmen. Bruken av den estimerte posisjonen til å øke den lokale posisjoneringen, var ikke egnet på grunn av den økende feilen og hoppene i de estimerte posisjonene som følge av relokalisering og løkkelukking.

# Preface

This thesis is result of the work done in TMR4930 Marine Technology - Master's Thesis at NTNU, and represents the final delivery of the Master of Science in Marine Cybernetics. The work was based upon the the project thesis from the Autumn of 2019, and was conducted from January 2020 to June 2020.

# Acknowledgments

I would like to thank my supervisor Professor Martin Ludvigsen for his guidance and support during the work of this thesis. He has ensured to provide the necessary hardware and facilities for this thesis to be possible, and has given invaluable feedback during the progress of this thesis.

This thesis would also not have been possible without the help from Ole Erik Vinje and Torgeir Wahl for making necessary laboratory equipment, and the fellow master students Ambjørn Waldum and Øyvind Denvik for the help conducting the laboratory experiments. For this, I thank them a lot.

I would also thank the PhD candidates at NTNU AUR-lab for their, help, inspiration and the lending the desktop computer used in this thesis.

Last, but not the least, I would thank all my family and friends for providing unconditional support during my studies at NTNU.

**Erlend Røilid Vollan, July 1, 2020**

# Table of contents

# List of Figures

# List of Tables

# Abbreviations

| | | |
|---|---|---|
| WC-ROV | = | Work Class Remotely Operated Underwater Vehicle |
| SLAM | = | Simultaneous Localization and Mapping |
| VSLAM | = | Visual Simultaneous Localization and Mapping |
| MC-lab | = | Marine Cybernetics laboratory |
| ROV | = | Remotely Operated Underwater Vehicle |
| AUR-Lab | = | Applied Underwater Robotics Laboratory |
| GNNS | = | Global Navigation Satellite System |
| EKF | = | Extended Kalman Filter |
| PF | = | Particle Filter |
| MSIS | = | Mechanical Scanned Image Sonars |
| FLS | = | Forward-Looking Sonar |
| SSS | = | Side-Scan Sonars |
| IEKF | = | Iterative Extended Kalman Filter |
| ROS | = | Robot Operating System |
| RANSAC | = | Random Sample Consensus |
| FOV | = | Field of View |
| CLAHE | = | Contrast Limited Adaptive Histogram Equalization |
| ORB | = | Oriented FAST and Rotated BRIEF |
| FAST | = | Features from Accelerated and Segmented Test |
| BRIEF | = | Binary Robust Independent Elementary Feature |
| MAP | = | Maximum A Posteriori |
| RGB-D | = | Red Green Blue Depth |
| BA | = | Bundle Adjustment |
| SDK | = | Software Developing Kit |
| FPS | = | Frames Per Second |
| TCP | = | Transmission Control Protocol |
| NED | = | North East Down |
| OFOV | = | Overlapping Field of View |
| TOFOV | = | Total Overlapping Field of View |
| HIL | = | Hardware In the Loop |

# Chapter 1

# Introduction

## 1.1 Background

Harvesting resources from the ocean has always been vital for the human survival and development. Many industries have through the ages been established from ocean resources, ranging from fisheries to mineral extraction, by designing appropriate technologies for the ocean environment. The growing human population raises the demand for new technologies to meet the challenges of utilizing the ocean resources more efficiently. In the existing offshore hydrocarbon production industry, the lack of new easily accessible reservoirs has driven the operations to deeper waters. The growing aquaculture industry with its focus on fish welfare has a demand for underwater monitoring and maintenance. The emerging offshore wind industry requires appropriate technology to handle the difficulties in the accompanying subsea installations of the offshore windmills.

For many subsea applications, the Work Class Remotely Operated Vehicle (WC-ROV) is an extensively used technology. The WC-ROV acts as an efficient extension for human workers as it makes it possible to observe and intervene in the harsh and inhospitable subsea environments. However, the WC-ROV has its limitations. The intervention capabilities of the WC-ROV is heavily dependent on the operator's experience and relies on a surface vessel to operate from in order to execute its mission. Skilled WC-ROV operators are costly to hire, and the day rates contributes to the high costs of missions where an WC-ROV is used.

A possible solution to the operator requirements and the associated costs of operating an WC-ROV, is to introduce autonomous capabilities to the WC-ROV. Autonomous features such as obstacle avoidance, path planning and unsupervised mission execution, could make the mission success be less dependent on the operator skill and remove the WC-ROV dependability of a surface vessel.

Two important waypoints on the path to autonomous WC-ROVs are local situational awareness and precise local position estimates of WC-ROVs. The situational awareness and positioning of WC-ROVs are today mainly covered by underwater acoustic systems or inertial navigation (also called dead-reckoning). Each of these methods have different trade-offs in terms of accuracy, cost, and complexity. A promising new low-cost approach

for increasing the local situation awareness and local position accuracy of WC-ROVs, could be Simultaneous Localization and Mapping (SLAM). SLAM is the method of concurrently estimating the position of you agent, or robot, and the environment that the robot is sensing. SLAM methods uses a variety of sensors to make observations of the environment, and in the branch of Visual SLAM (VSLAM), the utilized sensors are cameras. The cameras are an attractive SLAM sensor due to their low cost and capability of obtaining high amounts of data. Since most WC-ROVs have existing camera setups, either in mono or stereo configuration, the benefits of using VSLAM for localization and mapping can be exploited directly without any hardware modifications.

## 1.2 Motivation

The motivation for this Master's thesis is to contribute in the development of the autonomous capabilities of WC-ROVs by utilizing VSLAM, in order to increase the local situational awareness and local positioning accuracy of WC-ROVs. The local map produced by VSLAM can be used to establish autonomous features such as obstacle avoidance, while the accurate local positional estimates could prove vital in autonomous mission executions in areas inherited by subsea infrastructure.

## 1.3 Objective and Scope

The goals in which this thesis was set to investigate was comprised into the following objectives:

- Review necessary literature within the fields of underwater SLAM and VSLAM, underwater imaging and camera calibration.
- Perform underwater camera calibration of the stereo camera rig of the WC-ROV Minerva in the Marine Cybernetics Laboratory (MC-Lab).
- Propose a real-time VSLAM system for the WC-ROV Minerva based on the VSLAM method ORB-SLAM2 and the stereo camera rig of Minerva. The real-time system should account for underwater imaging effects, utilize the produced point cloud of ORB-SLAM2 to conduct closest obstacle detection, and be capable of conveying the closest detected obstacle to the existing Autonomy Framework of Minerva.
- Test the performance of the real-time WC-ROV VSLAM system in underwater obstacle course experiments in MC-Lab, both under ideal and subsea simulated lighting conditions.
- Evaluate the performance of the real-time WC-ROV VSLAM system by comparing the position estimates with optical ground truth measurements, and the estimated map and closest obstacle detection capability with ground truth measured underwater obstacle course.
- Verify the integration of the real-time WC-ROV VSLAM system with the Autonomy Framework on Minerva.

**Limitations** In the master agreement presented in the beginning of this thesis, one of the work objectives was to carry out field tests of the proposed real-time WC-ROV

VSLAM system. Due to the Covid-19 situation occurring in the spring of 2020, the field tests were not possible to conduct. It is worth noting that the expectations of conducting field tests played a major part in the selected direction for the determined objectives.

## 1.4 Remotely Operated Underwater Vehicles

This section aims to give an introduction to ROVs, present the operational domain of WC-ROVs, and present the WC-ROV Minerva. The content of this section has served as the basis for many of the design choices of the WC-ROV VSLAM system.

Remotely Operated Undewater Vehicles (ROV) are a category of unmanned underwater vehicles, and are characterized by being directly controlled by an operator through its surface tethered umbilical. The umbilical allows for unlimited power supply and high bandwidth data transmission between ROVs and the surface, but limits their spatial range. Due to the ROVs being manually controlled, they standardly equipped with cameras, but can also be euqipped with other surveying sensor such as sonars. Larger ROVs are often equipped with manipulator arms making them capable of performing subsea intervention missions and collect samples.

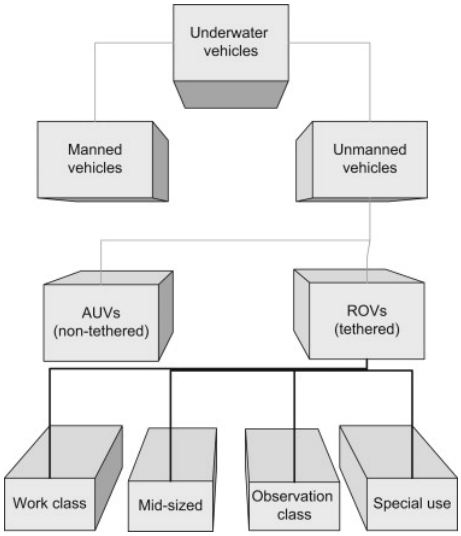



**Figure 1.1:** Categories of underwater vehicles. [1]

The common operation areas of ROVs are monitoring, intervention, mapping and sampling. These operations vary depending on the operational depths and market segments of the operators, and the classification of ROVs follows a similar structure. The different ROV classes are defined in [1] as observation class, mid-sized, work class and special-use vehicles.

- **Observation Class**: Ranges from the smallest ROVs up to ROV sizes of 100 kg. Limited to depths of 300 m, and are generally DC-powered. They are most often used for underwater inspections and typically hand launched with a hand tending tether.

- **Mid-sized**: Weighs from 100 kg up to a 100 kg, and are generally deeper rated version of the observation class with AC-power. They are mostly all electric, but with

some hydraulic power for the operation of manipulators and small tooling package options. Due to the weight, and launch and recovery system and a tether management system is often needed.

- **Work Class**: Generally heavy electromechanical vehicles running on high-voltage AC circuits. The delivered power is generally changed immediately to hydraulic power for the vehicle locomotion, manipulation and tooling functions.
- **Special-use Vehicles**: These are ROVs not falling under the main categories due to their non-swimming nature. In general crawling underwater vehicles, towed vehicles or structurally compliant vehicles.

### 1.4.1 ROV Minerva

The ROV Minerva is an WC-ROV owned and operated by the Applied Underwater Robotics Laboratory[1] (AUR-Lab) at the Institute of Marine Technology, NTNU. The application area of the Minerva is to access the seafloor for sampling and observation in marine science, and conduct experiments in engineering research such as control systems and autonomy [2].

Minerva weighs 2400 kg in air and has a depth rating down to 3000 m. It is rigged with seven thrusters capable of controlling the vehicle in six degrees of freedom using its control system developed at AUR-lab. The navigation sensors equipped on Minerva are an acoustic transponder for global positioning, an Inertial Measurement Unit (IMU) for inertial measurements, and a Doppler Velocity Log for velocity measurements. The surveying sensors are a stereo camera rig, a forward looking sonar and a video system. Minerva is also rigged with a manipulator arm capable of collecting samples and perform interventions tasks. Figure 1.2 shows Minerva being deployed from NTNU's research vessel Gunnerus.


**Figure 1.2:** The deployment of the WC-ROV Minerva.

[1] https://www.ntnu.edu/aur-lab

### 1.4.2 Operation Domain of WC-ROVs

The operation domain of WC-ROVs is characterized by performing interventions tasks, surveying or sample collection down to below 3000 m. The subsea environment at these depths introduces challenges to the visual data because of no ambient lighting and reduced visibility due to turbidity in the water. The quality of the visual data is additionally reduced due to scattering effects occurring due to the use of artificial illumination. The expected visibility is hence often assumed to be from three to six meters.

Many WC-ROV missions are conducted at, or in the close proximity to, subsea installations. This introduces challenges of the WC-ROV unintentionally colliding with obstacle damaging both the WC-ROV and the subsea installations. Due to installations being placed carefully in planned areas, it can be assumed that the seabed is flat in the operational area of the WC-ROV.

## 1.5 ROV Minerva Autonomy Framework

This thesis was a part of a larger work on developing increased situational awareness and autonomy for ROVs. The work comprised of at team of master student working on joining our thesis works into a combined solution in an proposed Autonomy Framework to be applied on the WC-ROV Minerva. The parts of the solution consisted of a mission planning and management architecture, a path planner, a VSLAM based motion estimation system and the obstacle detection of the proposed WC-ROV VSLAM system of this thesis. The work was formulated in to the manuscript in Appendix A, and applied to the Oceans 2020[2] conference.

The integration of the Autonomy Framework on Minerva was based upon the Minerva control system developed by AUR-Lab since 2010. The control system was initially developed for dynamical positioning and trajectory tracking [3], and is comprised of two modules: a graphical user interface allowing high-level control and mode selection of missions, and a low-level control system. The proposed Autonomy Framework was added to the graphical user interface module.

[2] `https://gulfcoast20.oceansconference.org/`

## 1.6 Literature Review of Underwater SLAM and VSLAM

This section presents the literature review on SLAM and VSLAM methods which was essential in the decision of selecting the VSLAM method ORB-SLAM2 for the real-time WC-ROV system.

Most of the literature describing the SLAM problem operates in land based environments. This often in relations to localization of robots in Global Navigation Satellite System (GNNS) suppressed buildings or for robot path planning and decision making. SLAM in the underwater setting offers the same problems as in land based SLAM, but introduce new limitation and challenges. These challenges are mostly related to the restrictions on the sensors that are available for vehicles operating in subsea environments and the reduced visibility when using VSLAM methods.

In Hidalgo and Bräunl [4], a review of different SLAM solutions applied in the underwater domain based upon the major SLAM paradigms of Extended Kalman Filters (EKF), Particle Filters (PF) and Graph-based SLAM is conducted. Most of the reviewed methods were based on the EKF, and only two of the methods used a camera, or cameras, as the primary sensory. The two camera based methods were both Graph-based. The sensor setup of the reviewed methods varied depending on the given underwater environment. For an structured environments the most common sensors were Mechanical Scanned Image Sonars (MSIS), Forward-Look Sonars (FLS) and cameras. In unstructured environments, e.g. seafloor application, usage of Side-Scan Sonars (SSS), FLS and cameras were the most common.

The EKF based SLAM methods struggles with computational complexity and linearization errors. This is especially a problem in the underwater domain where SLAM is applied in a large-scale manner. There are some EKF methods that avert this problem by reducing the global problem to a set of subproblems, and then joining the subproblems together to a global solution, e.g. Aulinas et al. [5] and Burguera Burguera and Bonin-Font [6]. In Aulinas et al. [5], feature bounded independent local maps are built from features extracted from SSS data. The global map is built from the constructed submaps by relating them through loop closure mechanisms. In Burguera Burguera and Bonin-Font [6], a similar approach is made, but instead of conjoining feature based maps, local trajectories of relative motion are used to construct a global trajectory. Both the local and global trajectories are estimated from matched visual image feature and altitude measurements using Iterative Extended Kalman Filters (IEKF). The loop closure detector is based upon finding loop candidates from image hashes.

Another method for reducing the computational complexity of the EKF is to utilize Information Filtering Techniques. These methods takes advantage of the sparsity of the inverse covariance matrix to reduce the complexity. The method described in Eustice et al. [7] is based upon the Information Filter, and uses visual inputs together with inertial measurements to estimate the location of an surveying ROV.

Due to the challenges of the EKF there has been in the recent years a great focus on graph-based SLAM methods. Especially on VSLAM solutions, much due to their low-cost and highly accurate and rich data acquisition capabilities. In Quattrini Li et al. [8], a experimental comparison of the most used open-source VSLAM methods is done. The comparisons are performed on a variety of different data sets including four data sets

collected in underwater environments. The overall best performing method was the graph-based monocular camera ORB-SLAM [9]. It, together with PTAM [10], performed best on the underwater datasets. Much to to their robust nature of being in-direct methods.

In Carrasco, Bonin-Font, and Codina [11], a VSLAM method specifically made for AUVs operating in shallow waters is proposed. The method is graph-based and takes visual inputs from a set of stereo cameras. The method shows many similarities to ORB-SLAM, and in its further extension in Negre, Bonin-Font, and Oliver [12], ORB-SLAM is compared with the extended version. The extension in Negre, Bonin-Font, and Oliver [12] further accompanies its shallow waters operational area by introducing a new loop detection method. Loop closing is performed by grouping visual features from multiple keyframes to create clusters of features, and matching is performed on the clusters instead of keyframe features. In the comparison it performs better than ORB-SLAM, much do to its increased capability to obtain loop-closures. The comparison is conducted in shallow waters in an environment colonized by sea grass.

In the graph-based VSLAM method proposed in Kim and Eustice [13], the challenge of loop detection is addressed by establishing a local and global metric of saliency for the image frames registered by the camera. The proposed method is designed for underwater ship inspection with an ROV where the obtained camera frames often are feature less. The local metric focuses on the image saliency of the frame, linking only salient frames to the graph. The global saliency metric focuses on the uniqueness of the current salient frame, and is used for guiding the ROV to areas of high uniqueness to obtain large-scale loop closure as well as detecting anomalies on the ship hull.

In Menna et al. [14], a preliminary accuracy assessment is performed on ORB-SLAM2 and a visual odometry approach proposed by the authors. The assessment was conducted on an small observation class ROV moving in a straight line in a small feature-sparse pool. The results showed promising, but further tests were recommended to be done. In Weidner et al. [15] ORB-SLAM2 was tested in an underwater cave environment. The test was motivated by the authors to compare their proposed underwater $3D$ mapping algorithm in an underwater cave environment with variable light sources. ORB-SLAM2 showed promising result regards to tracking and mapping, but did not conduct any loop closure due to the non looping image data set.

## 1.7 Contribution

The contribution of this thesis is the investigations of using VSLAM on WC-ROVs for increased local situational awareness and local position accuracy. The contribution revolves around utilizing the existing VSLAM method ORB-SLAM2 in a developed real-time WC-ROV VSLAM system for the ROV Minerva. The thesis contribution can be summarized as the following:

- A literature review mapping different SLAM and VSLAM approaches used in the underwater domain.
- A proposed real-time WC-ROV VSLAM system using the framework Robot Operating System (ROS) composed of the following parts:
  - The stereo camera rig of the WC-ROV Minerva.

  - An image processing part accounting for underwater imaging effects.
  - The VSLAM method ORB-SLAM2.
  - A closest obstacle detection algorithm based on point cloud clustering.
  - A communication part conveying the closest detected obstacle position to the Autonomy Framework of Minerva.

- The testing of the system in an underwater laboratory experiment, displaying promising closest obstacle detection capabilities, but struggling with accumulated drift in the estimated positions and map.
- The manuscript in Appendix A of the joint master student work on the Autonomy Framework of the WC-ROV Minerva applied to the Oceans 2020 conference.

## 1.8 Outline of Report

This thesis is divided into six chapters. The introduction in **Chapter 1** presents the thesis definition, the underwater SLAM and VSLAM literature review, and provides information about ROVs, the WC-ROV Minerva and the joint master student work on the Autonomy Framework of Minerva. In **Chapter 2** the theory serving as the foundation in the development of the WC-ROV VSLAM system is presented. The theory is within mono and stereo camera geometry, VSLAM and ORB-SLAM2, underwater imaging, image processing and point cloud processing. In the methods part of **Chapter 3**, the architecture and methodology for developing the WC-ROV VSLAM system is presented. **Chapter 4** explains the setup of the experiments testing the WC-ROV VSLAM system, while **Chapter 5** presents the results of the experiments. The discussion in **Chapter 6** discusses the results from the experiment and experiences made in the thesis work. Lastly, **Chapter 7** presents the thesis conclusion and recommendations for further work.

# Chapter 2

# Theory

This chapter presents the theory serving as the foundations of the methods used in the development of the real-time WC-ROV VSLAM system. Section 2.1 explains the geometry of mono and stereo cameras, presents the features of stereo vision and gives a short introduction to theory behind camera calibration. Section 2.2 introduces the concepts of contrast enhancement using image histograms, feature detection, and descriptors in image processing. Section 2.3 gives an introduction to SLAM, graph-based SLAM and VSLAM, and gives an detailed explanation of the VSLAM method ORB-SLAM2. Section 2.4 describes the characteristics of the underwater imaging process. Lastly, in Section 2.5 Random Sampling Consensus (RANSAC), and an Euclidean based clustering method for point cloud processing is presented.

## 2.1 Camera Geometry

This section introduces the pinhole camera model in digital imaging, the geometry of stereo vision systems, and gives an short introduction to the theory behind camera calibration.

### 2.1.1 Pinhole Camera Model

A camera projects $3D$ world points onto a $2D$ image plane. The mathematical description of this behavior can be achieved using the perspective camera model, or the *pinhole camera model*. In the pinhole camera model illustrated in Figure 2.1, the light from a scene passes through a tiny opening creating an inverted projection of the scene onto an image plane placed behind the pinhole. The distance between the pinhole and the image plane is the focal length $f$, and the inversion can be avoided by modeling the pinhole camera with a virutal image plane placed at the distance $f$ in front of the pinhole.

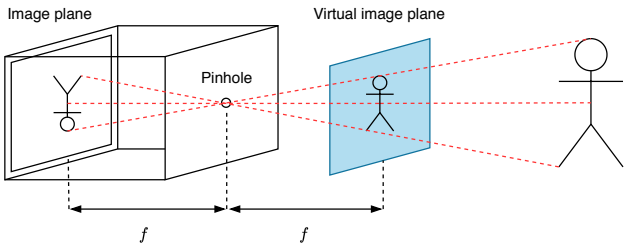



**Figure 2.1:** The pinhole camera model.

In digital imaging, the pinhole camera model describes the correspondence between observed $3D$ points in the world and the $2D$ pixels of the captured image. Figure 2.2 illustrates the three coordinate frames of the model: the world frame $\mathcal{F}_w$, the camera frame $\mathcal{F}_c$ and the image frame $\mathcal{F}_i$. The $\mathcal{F}_w$ is an arbitrary selected reference frame, while $\mathcal{F}_c$ has its origin at the camera projective center $C$ with the $Z$-axis pointing forward. The $\mathcal{F}_i$ spans the *normalized image plane* at $z = 1$ in $\mathcal{F}_c$, where the principal point $P$ is where the $Z$-axis intersects the $\mathcal{F}_i$.

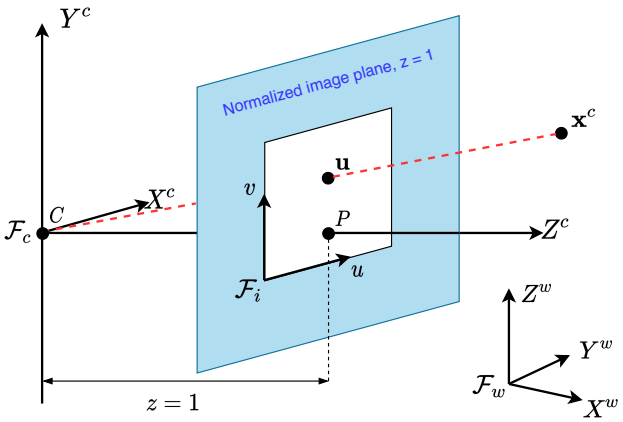



**Figure 2.2:** The world frame $\mathcal{F}_w$, camera frame $\mathcal{F}_c$ and image frame $\mathcal{F}_i$ of the pinhole camera mode. The red dotted line is the mapping of $\mathbf{x}^c$ to $\mathbf{u}$ onto $\mathcal{F}_i$.

The pinhole camera model in digital imaging consists of three parts each containing a linear transformation:

- An extrinsic part covering the transformation from $3D$ coordinates in the world frame to the camera frame.
- A perspective projection of $3D$ camera coordinates to $2D$ coordinates on the normalized image plane.
- A camera specific intrinsic part covering the affine transformation of $2D$ coordinates from the normalized image plane to the image plane.

Both the $3D$ and $2D$ points are represented by *homogeneous coordinates* using homogeneous matrices for the linear transformations.

**Homogeneous Coordinates:** A homogeneous coordinate vector $\tilde{\mathbf{x}}^c = \lambda[x, y, z, 1]^T$, where $\tilde{\mathbf{x}} = \lambda\tilde{\mathbf{x}}$ for all non-zero scalars $\lambda$, can be constructed from a Cartesian coordinate vector $\mathbf{x}^c = [x, y, z]^T$ with the mapping in Equation (2.1).

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \quad \mapsto \quad \tilde{\mathbf{x}} = \breve{\mathbf{x}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \in \mathbb{P}^3 \tag{2.1}$$

The $\breve{\mathbf{x}}$ denotes the normalized homogeneous coordinate according to Euclidean normalization, where $\tilde{\mathbf{x}}$ is scaled such that its extra dimension is 1. The extra dimension of the homogeneous coordinate allows for rigid-body transformations of coordinates to be represented as a linear matrix multiplication [16].

#### 2.1.1.1 Camera Extrinsic

The extrinsic part of the pinhole camera model describes the relative pose between $\mathcal{F}_w$ and $\mathcal{F}_c$. The relation is expressed by the homogeneous transformation matrix $\mathbf{T_{ab}}$ in Equation (2.2) called the *pose matrix*.

$$\mathbf{T}_{ab} = \begin{bmatrix} \mathbf{R}_{ab} & \mathbf{t}_{ab}^{a} \\ 0^{\top} & 1 \end{bmatrix} \in SE(3) \tag{2.2}$$

The pose matrix contains a rotation matrix $\mathbf{R}_{ab} \in SO(3)$ describing the orientation of a frame $\mathcal{F}_b$ relative to a frame $\mathcal{F}_a$, and a translation vector $\mathbf{t}_{ab}^{a} \in \mathbb{R}^3$ given in $\mathcal{F}_a$ giving the position of $\mathcal{F}_b$ relative to $\mathcal{F}_a$. The homogeneous coordinate position of $\tilde{\mathbf{x}}^w$ in $\mathcal{F}_w$ can hence be transformed into the homogenous coordinate position $\tilde{\mathbf{x}}^c$ in $\mathcal{F}_c$ using the pose matrix $\mathbf{T}_{cw}$

$$\tilde{\mathbf{x}}^c = \mathbf{T}_{cw}\tilde{\mathbf{x}}^w \tag{2.3}$$

#### 2.1.1.2 3D to 2D Projection

The $3D$ to $2D$ projection of the pinhole camera model describes the transformation from a coordinate $\mathbf{x}^c = (x^c, y^c, z^c)$ in $\mathcal{F}_c$ to a coordinate $\mathbf{x}_n = (x_n, y_n)$ in the normalized image plane. By using similar triangles in Figure 2.3, the transformation is described by Equation (2.4).

$$\mathbf{x_n} = \frac{1}{z^c}\begin{bmatrix} x^c \\ y^c \end{bmatrix} \tag{2.4}$$

In homogeneous coordinates, the transformation in Equation (2.4) can be represented by the *standard perspective projection matrix* $\Pi_0$ in Equation (2.5), where $\tilde{\mathbf{x}}_n$ is the homogeneous representation of the $2D$ point in the frame of the normalized plane, and $\tilde{\mathbf{x}}^c$ is the homogeneous representation of $\mathbf{x}^c$.

$$\tilde{\mathbf{x}}_n = \mathbf{\Pi}_0\tilde{\mathbf{x}}^c, \quad \Pi_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 4} \tag{2.5}$$

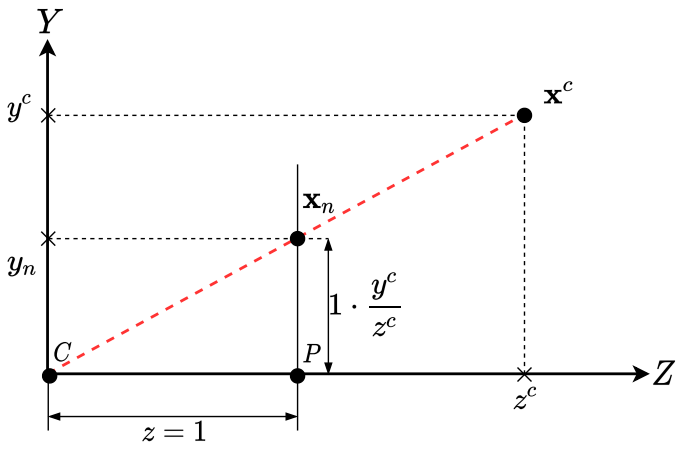



**Figure 2.3:** The relation $v_n = 1 \cdot \frac{y^c}{z^c}$ between the camera point $\mathbf{x}^c$ and the point $\mathbf{x}_n$ in the normalized image plane using similar triangles.

#### 2.1.1.3 Camera Intrinsics

The intrinsic part of the pinhole camera model describes the affine transformation from $2D$ coordinates in the normalized image plane to $2D$ pixel coordinates in the image frame $\mathcal{F}_i$. The transformation is described by the homogeneous transformation given in Equation (2.6) where $\tilde{\mathbf{u}}$ is the $2D$ pixel coordinates in $\mathcal{F}_i$, $\mathbf{K}$ is the *intrinsic camera matrix* and $\tilde{\mathbf{x}}_n$ is the $2D$ coordinates in the normalized image plane

$$\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}_n, \quad \mathbf{K} = \begin{bmatrix} f_u & s_\theta & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{2.6}$$

The intrinsic matrix $\mathbf{K}$, often called the calibration matrix, is camera specific. The elements of $\mathbf{K}$ gives the relationship between the pixels in $\mathcal{F}_i$ and the coordinate positions in the normalized plane. The elements have the following meaning:

- $f_u$ and $f_v$ is the size of unit length in horizontal and vertical pixels. They can be expressed as $f_u = fs_u$ and $f_v = fs_v$ where $f$ is the camera focal length in metric unit and $s_u$ and $s_v$ are is the scaling factor giving the pixel density.
- $c_u$ and $c_v$ is the $u$- and $v$-coordinate of the principle point $P$ in pixels.
- $s_\theta$ is the skew of the pixel, most often close to zero.

The complete homogeneous projection of a world point $\tilde{\mathbf{x}}^w$ given in $\mathcal{F}_w$ to the pixel coordinate $\tilde{\mathbf{u}}$ in $\mathcal{F}_i$ using the intrinsic matrix $\mathbf{K}$, the standard projection matrix $\pi_0$ and the pose matrix $\mathbf{T}_{cw}$ is given by

$$\tilde{\mathbf{u}} = \mathbf{K}\boldsymbol{\Pi}_\mathbf{0}\mathbf{T}_\mathbf{cw}\tilde{\mathbf{x}}^w \tag{2.7}$$

#### 2.1.1.4 Projection Function

The projection function $\pi_p$ in Equation (2.8) gives the Euclidean point projection of a world point $\mathbf{x}^w$ to the pixel coordinate $\mathbf{u}$ in the image frame $\mathcal{F}_i$. It is equivalent to the homogeneous projection of Equation (2.7), using $\mathbf{R}_{cw}$ and $\mathbf{t}^c_{cw}$ of the pose matrix in Section 2.1.1.1 to transform $\mathbf{x}^\mathbf{w}$ to $\mathbf{x}^\mathbf{c}$.

$$\begin{aligned} \mathbf{u} = \pi_p\big(\mathbf{T}_{cw},\ \mathbf{x}^w\big) &= \begin{bmatrix} f_u \frac{x^c}{z^c} + c_u \\ f_v \frac{y^c}{z^c} + c_v \end{bmatrix} \\ \begin{bmatrix} x^c & y^c & z^c \end{bmatrix}^T &= \mathbf{R}_{cw}\mathbf{x}^w + \mathbf{t}^c_{cw} \end{aligned} \tag{2.8}$$

#### 2.1.1.5 Field of View

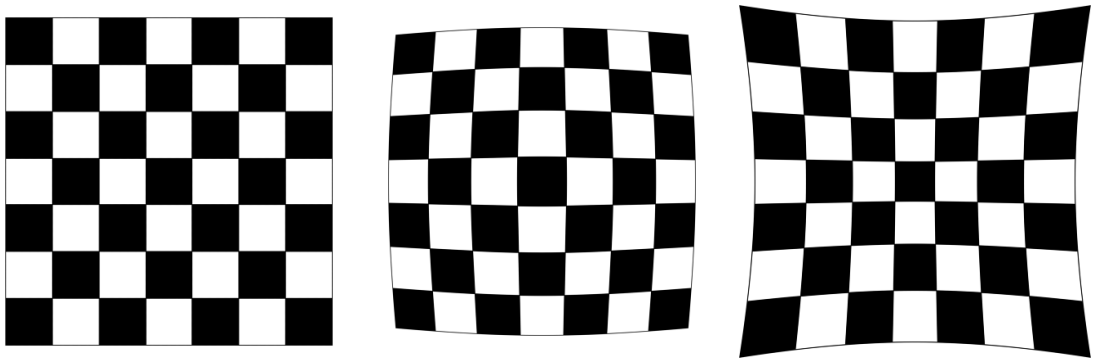
The field of view (FOV) of a camera express how wide the viewing angle of a camera is, i.e how much of a scene a camera can observe. For square image sensors, it is expressed by the FOV angle $\theta$ and can be calculated using the camera focal length $f$ and the sensor length $l$. Depending on definition, the sensor length $l$ could be the image sensor diagonal, its horizontal length, or vertical length. Equation (2.9) solves the trigonometric problem giving the camera field of view $\theta$.

$$\theta = \arctan \frac{l/2}{f} \tag{2.9}$$

#### 2.1.1.6 Lens Distortion

Distortion in the pinhole camera model is the deviation of the straight $3D$ line assumption from the camera point $\mathbf{x}^c$ in $\mathcal{F}_c$ to the pixel points $\mathbf{u}$ in $\mathcal{F}_i$. Distortion is caused by the presence of the camera lens and is in the *descentering model* of Brown [17] separated into *radial-* and *tangential* distortion.

**Radial Distortion**: Radial distortion cause pixel points in the image frame to appear closer or further away in the radial direction of the principle point $P$. Positive radial distortion cause a barreling effect on the image, while negative radial distortion causes a pincushioning effect, see Figure 2.4. The radial distortion is caused by the radial change in thickness of the camera lens causing different bending angles of the light rays in the radial direction from $P$ due to refraction. See Section 2.4.2 for refraction.



**(a)** No distortion **(b)** Positive radial distortion **(c)** Negative radial distortion

**Figure 2.4:** The radial distortion effect.

**Tangential Distortion**: Tangential distortion is caused by misalignment of the image sensor and camera lens, illustrated in Figure 2.5. Correction for tangential distortion will manifest itself as a tilting of the image plane.

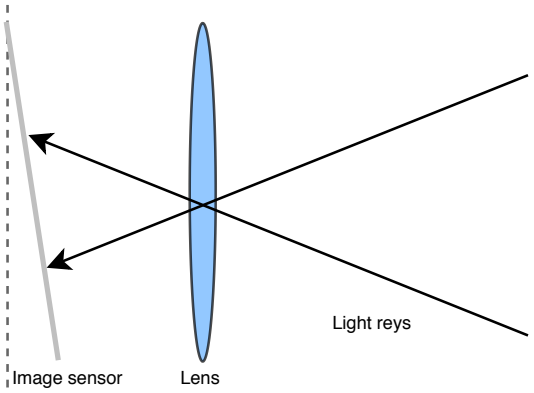



**Figure 2.5:** Unaligned image sensor and lens.

Both radial and tangential distortion are non-linear effects and can be modeled by a polynomial correcting the position of the image pixel points. In the descentering model, Equation (2.10) is used for the correction where $\mathbf{u}_d = (u_d, v_d)$ is the distorted pixel coordinates, $\mathbf{u}_\mathbf{u} = (u_u, v_u)$ is the undistorted pixel coordinates, $k_1$, $k_2$ and $k_3$ are the radial

distortion coefficients, and $p_1$ and $p_2$ are the tangential distortion coefficients.

$$\begin{aligned} u_d =& u_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 u_d v_d + p_2(r^2 + 2u_d^2) \\ v_d =& v_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2v_d^2) + 2p_2 u_d v_d \end{aligned} \tag{2.10}$$

The coefficients of Equation (2.10) can be obtained for a given camera by *camera calibration*. See Section 2.1.4. The distortion coefficients when obtained can then be used to remove the lens distortion of an image by applying the inverse mapping of Equation (2.10).

### 2.1.2 Stereo Vision

In stereo vision, $3D$ information is extracted from a scene by comparing pairwise images from two different viewpoints. The depth of the scene is inferred by comparing the relative positions of objects in the pairwise images. For point features this procedure is referred to as *point triangulation* and is best described using *epipolar geometry*.

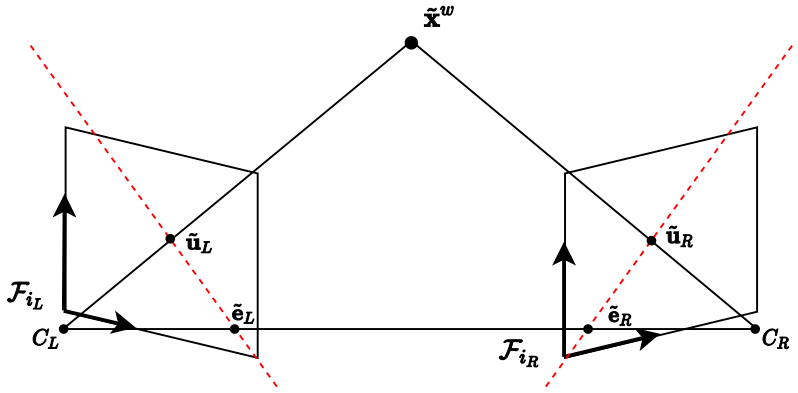



**Figure 2.6:** The epipolar geometry of two cameras observing the world point $\mathbf{x}^w$.The projected points $\mathbf{u}_L$ and $\mathbf{u}_R$ are contained in the left and right red dotted epipolar lines.

#### 2.1.2.1 Epipolar Geometry and Point Triangulation

Epipolar geometry describes the constraints image views of the same scenes poses on to themselves. In Figure 2.6 two cameras with both known $\mathbf{K}$ and $\mathbf{T}$ are observing a world point $\tilde{\mathbf{x}}^w$. The projections of $\tilde{\mathbf{x}}^w$ in the image views are denoted $\tilde{\mathbf{u}}_L$ and $\tilde{\mathbf{u}}_R$. The projections of each optical centre in the other camera's image plane are called the *epipoles* and are denoted $\tilde{\mathbf{e}}_L t$ and $\tilde{\mathbf{e}}_R$. The epipolar lines are the red dotted lines intersecting $(\tilde{\mathbf{e}}_R, \tilde{\mathbf{u}}_R)$ and $(\tilde{\mathbf{e}}_L, \tilde{\mathbf{u}}_L)$. These are the $2D$ projections of the virtual backprojected $3D$ lines of $\tilde{\mathbf{u}}_L$ and $\tilde{\mathbf{u}}_R$ in the opposite image planes.

**Epipolar constraint:** The *epipolar constraint* states that projection of $\tilde{\mathbf{x}}^w$ on the left and right camera image view, $\tilde{\mathbf{u}}_L$ and $\tilde{\mathbf{u}}_R$, must be contained on the epipolar lines intersecting $(\tilde{\mathbf{e}}_L, \tilde{\mathbf{u}}_L)$ and $(\tilde{\mathbf{e}}_R, \tilde{\mathbf{u}}_R)$ respectively.

Hence, if both image points $\tilde{\mathbf{u}}_L$ and $\tilde{\mathbf{u}}_R$ are known, the world point $\tilde{\mathbf{x}}^w$ can be triangulated using both the camera's $\mathbf{K}$ and $\mathbf{T}$. The reader is referred to Hartley and Zisserman [18] for a detailed description of point triangulation in multiple view geometry.

#### 2.1.2.2 Image Rectification

Image rectification is the procedure of applying projective transformations on a set of images such that the images appear on a common image plane. It can be applied to pairs of stereo images in order to obtain the assumption of a *fronto parallel* stereo camera. The assumption gives the two properties: the epipolar lines are parallel to he horizontal axis, the corresponding image points have the identical vertical coordinate [19].The two properties are often utilized in stereo vision as searches for descriptor matches is only necessary along the horizontal pixel lines of the image. See Section 2.2 for descriptors.

Figure 2.7 depicts the stereo camera of Figure 2.6 where image rectification has been applied with the two mappings

$$\begin{aligned} \tilde{\mathbf{u}}_L^{rec} &= \mathbf{H}_L \tilde{\mathbf{u}}_L, \quad \mathbf{H_L} \in \mathbb{R}^{3\times 3} \\ \tilde{\mathbf{u}}_R^{rec} &= \mathbf{H}_R \tilde{\mathbf{u}}_R, \quad \mathbf{H}_R \in \mathbb{R}^{3\times 3} \end{aligned} \tag{2.11}$$

on every homogeneous pixel coordinate of the left an right image view using the projection transformation matrices $\mathbf{H}_L$ and $\mathbf{H}_R$. The frames $\mathcal{F}_{i_L}$ and $\mathcal{F}_{i_R}$ of the left and right image views in Figure 2.6, are transformed into the rectified frames $\mathcal{F}_{i_L}^{rec}$ and $\mathcal{F}_{i_R}^{rec}$ of Figure 2.7 residing in a common image plane such that the fronto parallel assumption holds. The projection transformation matrices $\mathbf{H_L}$ and $\mathbf{H_R}$ can be determined from the known relative orientation $\mathbf{R}_{LR}$ and translation $\mathbf{t}_{RL}^R$ of the cameras. The reader is advised to Fusiello, Trucco, and Verri [20] for detailed description about this methodology.

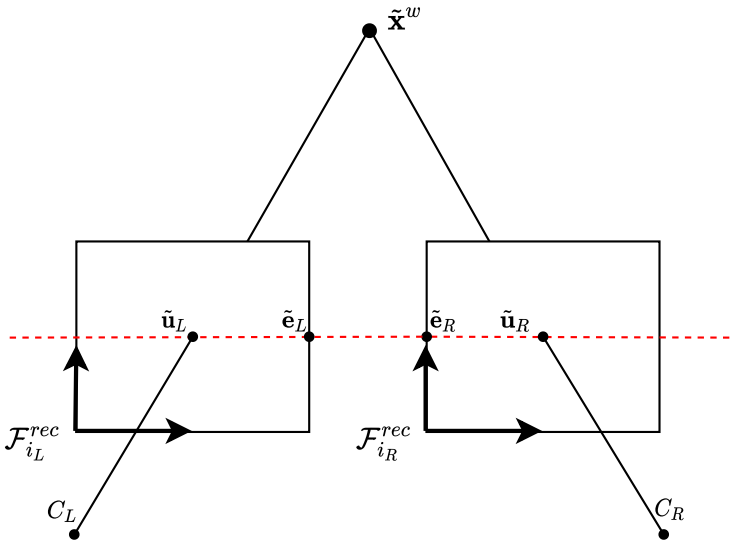



**Figure 2.7:** The rectified image frames $\mathcal{F}_{i_L}^{rec}$ and $\mathcal{F}_{i_R}^{rec}$ transformed from the image frames $\mathcal{F}_{i_L}$ and $\mathcal{F}_{i_R}$ of Figure 1.4. The red dotted epipolar line of $\tilde{\mathbf{u}}_L^{rec}$ and $\tilde{\mathbf{u}}_R^{rec}$ is parallel to the horizontal axises of the rectified image frames.

### 2.1.3 Range of Depth and Disparity

The *range of depth* of a fronto-parallel stereo vision system is the minimum and maximum depth that can be perceived for given stereo vision system's range of *disparity* values. The disparity is the horizontal pixel difference $d = u_L - u_R$ of two left and right image points $\mathbf{u_L}$ and $\mathbf{u_R}$ matched to the world point $\mathbf{x}^w$. The minimum disparity value is usually a single pixel, while the maximum is the width of the stereo image resolution. The equation used

to determine the range of depth can be obtained from the geometry of a stereo vision setup observing a scene with a world point $\mathbf{x^w}$. Figure 2.8 depicts two fronto-parallel cameras represented by the image frames $\mathcal{F}_{i_L}$ and $\mathcal{F}_{i_R}$ where the baseline $b$ is the horizontal distance between them. From the two pairs of similar triangles $T_1/T_3$, and $T_2/T_4$, the world point depth $z^w$ can be expressed by Equation (2.12)

$$z^w = f\frac{b}{d} \tag{2.12}$$

where $f$ is the focal length, $d$ is the disparity and $b$ is the baseline. Hence the ability of a stereo system differentiating depth is affected by $f$ and $b$. The focal length is usually fixed, thus the stereo visions range of depth can be altered by adjusting $b$.

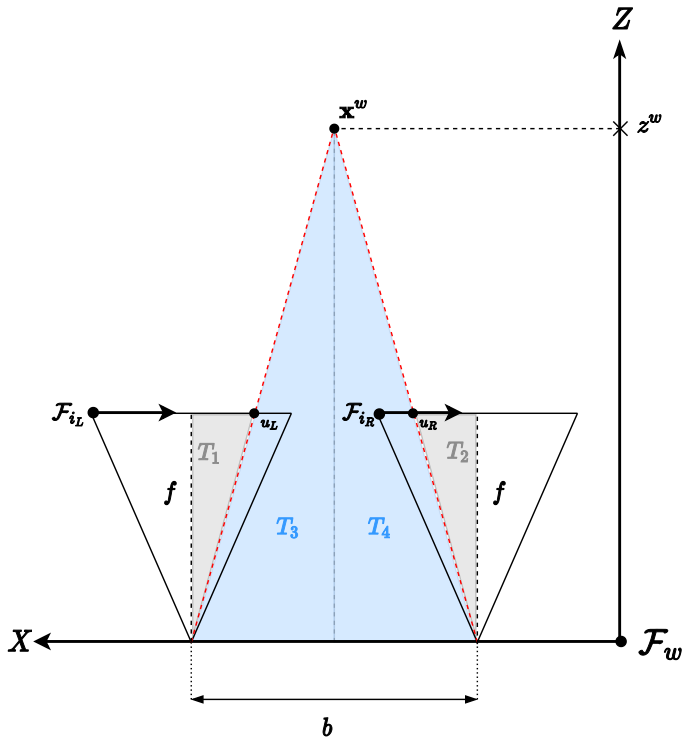



**Figure 2.8:** The image frames $\mathcal{F}_{i_L}$ and $\mathcal{F}_{i_R}$ of two fronto-parallel cameras and their geometry described by similar triangles $T_1/T_3$, and $T_2/T_4$.

#### 2.1.3.1 Stereo Projection Function

The stereo projection function is a function that describes the Euclidean point projection of world point $\mathbf{x^w}$ to a pixel coordinate $\mathbf{u}$ in a fronto-parallel stereo vision setup. By assuming the stereo cameras being fronto-parallel, Equation (2.12) can be utilized to gain an additional constraint in the the single camera projection function of Equation (2.8). The additional constraint gives an extra equation to the projection, thus making the scale of the projected point observable. The equation containing the extra constraint is obtained by inserting $d = u_L - u_R$ in Equation (2.12) and solving for $u_R$, resulting in the stereo

projection function $\pi_s$ in Equation (2.13).

$$\mathbf{u}_s = \begin{bmatrix} u_L^c \\ v_L^c \\ u_R^c \end{bmatrix} = \pi_{\mathrm{s}}\big(\mathbf{T}_{cw},\ \mathbf{x}^w\big) = \begin{bmatrix} f_u \frac{x^c}{z^c} + c_u \\ f_v \frac{y^c}{z^c} + c_v \\ f_u \frac{x^c - b}{z^c} + c_u \end{bmatrix} \\ \begin{bmatrix} x^c & y^c & z^c \end{bmatrix}^T = \mathbf{R}_{cw}\mathbf{x}^w + \mathbf{t}_{cw}^c \tag{2.13}$$

### 2.1.4 Camera calibration

Camera calibration is the procedure of estimating the intrinsic parameters and the distortion coefficients of an camera. It can be extended to stereo camera calibration, where the extrinsic parameters of two cameras are additionally determined. There are a variety of techniques used in camera calibration ranging from using $3D$ calibration objects Wei and Ma [21], to self-calibration methods based on taking images by moving the camera in static scene Maybank and Faugeras [22].

A common and renowned method for camera calibration is *Zhang's methods* given in Zhang [19]. The method is based upon using projective transformations to establish constraints used in a non-linear minimization problem for estimating the camera parameters. The projective transformations are described by Equation (2.14), where $\tilde{\mathbf{u}}$ is an image point, $\mathbf{x}^w$ is a world point, $\mathbf{r}_1$ and $\mathbf{r_2}$ are the first and second columns of the rotation matrix $\mathbf{R}$, $\mathbf{t}$ is the translation vector, and $\mathbf{K}$ is the camera intrinsic matrix.

$$\tilde{\mathbf{u}} = \mathbf{H}\mathbf{x}^w, \quad \mathbf{H} = \mathbf{K}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \tag{2.14}$$

The established projective transformations describes the transformation of an known plane with different orientations in a given set of images. The plane is typically a rigid board printed with checker board pattern of known dimensions, and the image set is collected by capturing images of the plane in different orientations. The resulting non-linear minimization problem of Zhang's method is presented in Equation (2.15),

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \left\| \mathbf{u}_{ij} - \breve{\mathbf{u}}\left(\mathbf{K}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, \mathbf{x}_j^w\right)\right\|^2 \tag{2.15}$$

where $\breve{\mathbf{u}}(\mathbf{K}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, \mathbf{x}_j^w)$ is the projection of a point $\mathbf{x}_j^w$ in an image $i$ according to the established projective transformation for image $i$ in Equation (2.14) followed by a distortion correction using $k_1$ and $k_2$. To obtain the resulting non-linear minimization problem, a series of steps is performed including the distortion correction. The reader is referred to Zhang [19] for details about these steps.

Due to the use of projective transformations, Zhang's method has a pitfall. The establishment of the projective transformation constraints is based upon properties of the rotation matrix. If two consecutive images has the same rotation matrix, they will not provide any additional constraint. Hence, if the plane undergoes pure translations, the second of the two consecutive images will not contribute to the calibration problem [19]. Pure translations is thus unwanted while conduction camera calibration according to Zhang's method.

## 2.2 Image Processing

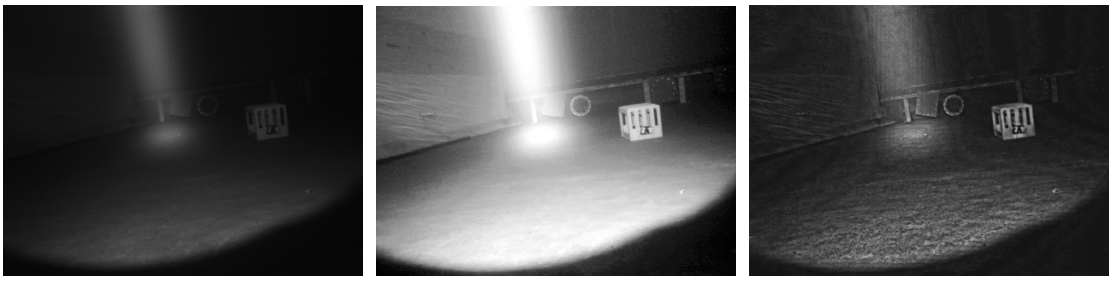
This section presents the image contrast enhancing techniques of histogram equalization and contrast limited adaptive histogram equalization, and explains the concepts of feature detectors and descriptors.

#### 2.2.0.1 Histogram Equalization

Histogram equalization is a method of increasing the contrast of an image by adjusting the image's grayscale values based on its image histogram. In an image histogram, the grayscale pixel intensities are along the x-axis, while the y-axis is the number of pixels. The pixel intensities of an image is often centered around the intensity mean, and the concept of histogram equalization is to evenly distribute these pixel intensities over the whole intensity range of the image. The intensity range of an image depends on the pixel format, and for grayscale images where each pixel consists of a byte, it consists of 256 graylevels. By having the intensities spread in the histogram, the contrast of the image is enhanced thus highlighting the information contained in the image. The histogram equalization of an image is shown in Figure 2.9b, with corresponding histogram in Figure 2.10b.

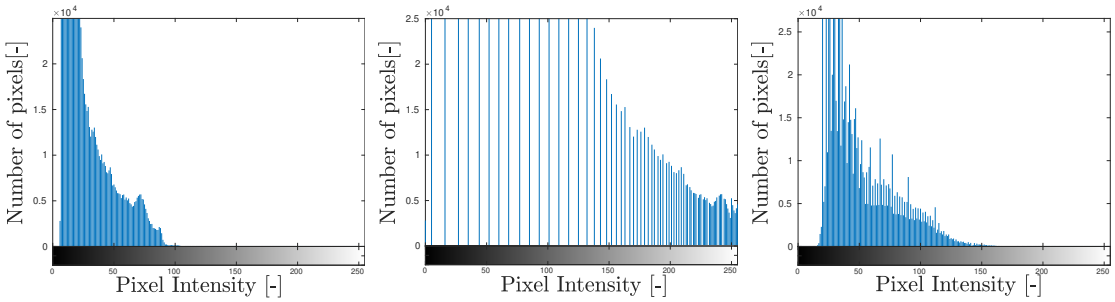
#### 2.2.0.2 Contrast Limited Adaptive Histogram Equalization

On images where the pixel intensities are not centered around a mean, i.e images where the different areas of the images are either dark or bright, ordinary histogram equalization would not prove the necessary local contrast enhancing. The *Contrast Limited Adaptive Histogram Equalization* (CLAHE) [23] is an extended version of histogram equalization coping with the problem of local contrast enhancing by dividing the images into local regions and perform local histogram equalization in the divided regions. An issue with local contrast enhancing is that noise in homogeneous regions of the image will be overamplified. The CLAHE solves this by limiting the amplification by redistributing the intensity values of the histogram from above a specific given intensities value. This intensities value limit is called the clip limit, as it clips the height of the histogram, and uniformly allocates the clipped intensities values along the whole histogram [23]. Homogeneous regions of the image is shown as peak in the histogram, and thus by clipping these peaks, the effect of intensity increasing these peaks is reduced. Figure 2.9c shows the contrast enhancing of an image with containing both dark and bright regions using histogram equalization and CLAHE, Figure 2.10c shows the corresponding histograms.



**(a)** Unprocessed **(b)** Histogram equalization **(c)** CLAHE

**Figure 2.9:** The contrast enhancing of an image with uneven brightness.




**(a)** Unprocessed **(b)** Histogram equalization **(c)** CLAHE

**Figure 2.10:** The histograms of the images in Figure 2.9

### 2.2.1 Feature Detection and Descriptors

In image processing, feature detection is the method of locating the image pixel coordinates related areas of interest in an image. Descriptors are encodings applied to the detected image features in order to differentiate the detected features. Feature detection could be based on larger regions such as edges, or a specific interest points given by a single pixel coordinate such as corner points. Figure 2.11 show an image with it's detected corners points using *harris corner detection* [24]. Descriptors applied to corner points often selects a surrounding patch of a pixel width and height around the detected corner, and performs calculations on this patch in order to obtain an encoding of the corner point. The calculated encodings can be used to compare the corner point descriptors of different images observing the same scene, in order to match features across images.

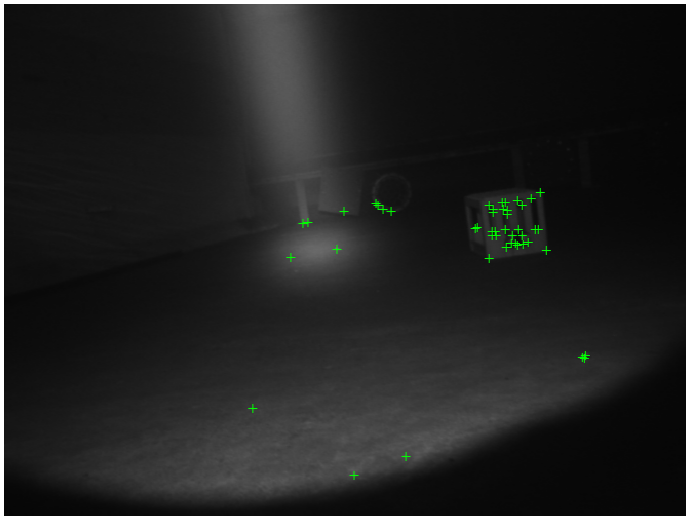


**Figure 2.11:** Harris corner detection applied to an image.

#### ORB Features

Oriented FAST and Rotated BRIEF (ORB) [25], is an efficient combined feature detector and descriptor. It is built upon the keypoint detector Features from Accelerated and Segmented Test (FAST) [26], and the descriptor (Binary Robust Independent Elementary Feature) [27]. In ORB, modifications are added to the FAST and BRIEF in order to make the features scale and rotation invariant. ORB is made partly scale invariant by using

a multiscale image pyramid. The multiscale image pyramid is an image representation consisting of sequence of image scaled down in resolution from the original image. By detecting features on each level, ORB is detecting features on different scales. The rotation invarians is made by calculating the feature orientation based upon the change in intensities within the feature.

## 2.3 Simultaneous Localization and Mapping

This section presents the problem definition of Simultaneous Localization and Mapping (SLAM), explains the branch of graph-based SLAM, introduces Visual SLAM (VSLAM), and gives an extensive description of the VSLAM method ORB-SLAM2.

### 2.3.1 The SLAM Problem Definition

SLAM consists of the simultaneous estimation of a robot's state and the construction of a model of the environment, the map, that the sensors of the the robot are perceiving [28]. The SLAM problem can be formulated in two ways; the *full* SLAM problem or the *online* SLAM problem [29]. The mathematical formulation of the full SLAM problem can be expressed as the joint posterior probability in Equation (2.16).

$$p(\mathbf{X}_T, m|\mathbf{Z}_T, \mathbf{U}_T) \tag{2.16}$$

In Equation (2.16), the objective is to estimate the sequence of robot locations $\mathbf{X}_T$ and a model of the world $m$ from all of the available data. The available data is the sequence of odometry measurements $\mathbf{U}_T$ and sensor measurements $\mathbf{Z}_T$ perceived by the robot. The online SLAM formulation expressed in Equation (2.17) differs from full SLAM as it only seeks to recover the current robot location $\mathbf{x}_t$ instead of the entire path.

$$p(\mathbf{x}_t, m|\mathbf{Z}_T, \mathbf{U}_T) \tag{2.17}$$

There is a variety of different methods proposed for solving the SLAM problem. Most of these methods are derived from the three major paradigms of SLAM: *Extended Kalman Filter* (EKF), *Particle Filter* (PF) and graph-based [29]. The two first, EKF SLAM and PF SLAM, are regarded as filtering methods as they seek to solve the online SLAM formulation in Equation (2.17), while graph-based methods in Equation (2.16) solve the full SLAM problem.

This thesis will consider the graph-based VSLAM method ORB-SLAM2. The following subsections will thus present the basics of graph-based SLAM and VSLAM.

### 2.3.2 Graph-Based SLAM

Graph-based SLAM uses graphs to describe the sensory constraints of a robot exploring the world. The observed landmarks $\mathbf{L}$ and robot locations $\mathbf{X}_T$ can be thought of as the nodes in a graph, while the constraints between the robot locations and landmarks are the graph edges. These are connecting every consecutive pair of locations $\mathbf{x_{t-1}}, \mathbf{x}_t$, and between landmarks $l_i$ and locations $x_t$ given that the robot sensed the landmark $i$ at time $t$.

A simple graph from a robot exploring is illustrated in Figure 2.12. The graph was constructed by adding nodes for each new robot location and measured landmark. The edges were connected between the current and previous locations, and the current location and new sensor measurements.

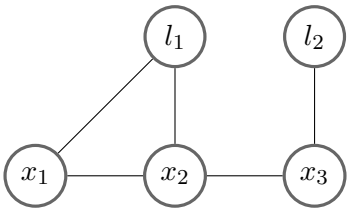



**Figure 2.12:** A simple graph-based SLAM formulation. A robot at the three separate locations $x_1$, $x_2$ and $x_3$ are observing the two landmarks $l_1$ and $l_2$.

Due to the inherent uncertainty of the odometry and sensory measurements, the graph edges act as soft constraints between the nodes of the graph. It allows for the graph-based SLAM problem to be formulated as an non-linear optimization problem. The key to graph based SLAM is that while the graph increases in size during exploration, it will remain sparse due to the limited number of constraints between each node. The number of constraints is at worst *linear* in the time elapsed and in the number of nodes in the graph [29]. The sparsity makes graph-based SLAM applicable for efficiently solving the full SLAM formulation through non-linear optimization.

### Maximum A Posteriori Estimate

The full SLAM formulation defined in Equation (2.16) can for graph-based SLAM be solved using the *maximum a posteriori* (MAP) estimate in Equation (2.18) [30].

$$\mathbf{X_T}^*, \mathbf{L}^* = \arg\max_{\mathbf{X_T}, \mathbf{L}} p(\mathbf{X}_T, \mathbf{L} | \mathbf{Z}_T, \mathbf{U}_T) \tag{2.18}$$

In Equation (2.18), $\mathbf{x_i} \in \mathbf{X}_T$ are the robot poses, $\mathbf{l}_k \in \mathbf{L}$ are the landmark locations, $\mathbf{u}_i \in \mathbf{U_T}$ are the odometry measurements, and $\mathbf{z}_j \in \mathbf{Z}$ are the measurements of the landmark features. The MAP estimate seeks to find the set of robot poses and landmark locations that maximizes the posterior density for the given odometry and landmark measurements.

Assuming measurement models with additive Gaussian noise [30], the optimization of Equation (2.18) leads to the following nonlinear least-squares problem:

$$\begin{aligned}
\mathbf{X_T}^*, \mathbf{L}^* &= \arg\max_{\mathbf{X_T}, \mathbf{L}} p(\mathbf{X}_T, \mathbf{L} | \mathbf{Z}_T, \mathbf{U}_T) \\
&= \arg\min_{\mathbf{X_T}, \mathbf{L}} -logp(\mathbf{X}_T, \mathbf{L} | \mathbf{Z}_T, \mathbf{U}_T) \\
&= \underset{\mathbf{X_T}, \mathbf{L}}{\operatorname{argmin}} \left[ \sum_i \| \mathbf{x}_i - f_i \left( \mathbf{x}_{i-1}, \mathbf{u}_{i-1} \right) \|^2_{\Sigma^i_w} + \sum_j \left\| \mathbf{z}_j - h_j \left( \mathbf{x}_{i_j}, \mathbf{l}_{k_j} \right) \right\|^2_{\Sigma^j_i} \right]
\end{aligned} \tag{2.19}$$

where $f_i$ and $h_j$ are the odometry and sensory measurement models with zero-mean additive Gaussian noise with covariances $\Sigma^i_w$ and $\Sigma^i_v$. By defining $\|\mathbf{e}\|^2_\Sigma = \mathbf{e}^T \Sigma^{-1} \mathbf{e}$ and linearizing about the current estimate, Equation (2.19) converts into the linear least-squares form for the state update vector solved with the normal equations in Equations (2.20) and (2.21)

$$\arg\min_{\Delta\boldsymbol{\Theta}} \|\mathbf{A}\Delta\boldsymbol{\Theta} - \mathbf{b}\|^2 \tag{2.20}$$

$$\Delta\boldsymbol{\theta} = \left( \mathbf{A}^\top \mathbf{A} \right)^{-1} \mathbf{A}^\top \mathbf{b} \tag{2.21}$$

The vector $\mathbf{\Theta}$ consists of the robot location and landmarks, $\mathbf{A}$ is the stacked whitened measurement Jacobian, and $\mathbf{b}$ is the corresponding residual vector. The non-linear problem can solved by iteratively re-linearizing and solving until the solution converges. Common methods for this are the Gauss-Newton or Levenberg Marquadt algorithms.

### 2.3.3 Visual SLAM

In Visual SLAM (VSLAM) the sequence of odometry and landmark measurements, $\mathbf{U_T}$ and $\mathbf{Z_T}$ are obtained from the visual input from either monocular, stereo or RGB-D (Red, Blue, Green, Depth) camera. The state of the robot is represented by its pose $\mathbf{T}_{cw}$, as shown in Equation (2.2) which is the position and orientation of the given robot relative to a reference frame. The model constructed from the environment, or the landmarks $\mathbf{L}$, depends on the selected VSLAM approach, where *dense methods* reconstructs the environments through the estimated depth while *sparse methods* uses point locations of observed landmarks.

When solving the full SLAM problem in VSLAM, it is common to separate the VSLAM system into two main components: the *front end* and the *back end* [28].

- **Front end**: The front end extracts relevant data from the raw measurements provided by the cameras, performs data association between measurements and the map, building the optimization problem.
- **Back end**: The back end performs inference on the data provided by the front end, solving the optimization problem over the robot poses and measurement.

The data association in the front end consists mainly of the task of detecting and match features from the observed camera frame, and perform data recognition to invoke loop-closure when places are being revisited.

The back end depends on the formulation of the nonlinear optimization problem. In VSLAM are it is either formulated as a *direct* or *in-direct*. The direct formulation seeks to minimize the photometric error, while the in-direct method seeks to minimize the geometric error formulated through the either the projection function defined Equation (2.8), or the stereo projection function defined in Equation (2.13). Minimizing the geometric error is often referred to as *Bundle Adjustment*.

#### Bundle Adjustment

Bundle Adjustment (BA) is the problem of refining a visual reconstruction in order to obtain a jointly optimal $3D$ structure and viewing parameter estimates [31]. The viewing parameters could either be the camera poses, calibration parameters or both.

In SLAM, the interest is to optimize the $3D$ world points $\mathbf{x}_j^w \in \mathbb{R}^3$ and camera poses $\mathbf{T}_{iw} \in SE(3)$ (see Section 2.1.1.1) with regards to the world frame $\mathcal{F}_w$. BA does this by minimizing the reprojection error with respect to matched image points $\mathbf{u}_{i,j} \in \mathbb{R}^2$ of separate images of the visual data set. The reprojection error of an observation of a map point $j$ in an image frame $i$ is the difference in its measured and predicted the pixel position given in Equation (2.22).

$$\mathbf{e_{i,j}} = \mathbf{u}_{i,j} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_j^w) \tag{2.22}$$

The projection function $\pi_i$ projecting map points to a given image frame $i$, could either be defined for a monocular or stereo camera using the mono or stereo projection function in Equation (2.8) or Equation (2.13). The cost function minimized in bundle adjustment is given in Equation (2.23), where $\rho_h$ is the Huber robust cost function and $\Omega_{i,j} = \sigma_{i,j}^2 \mathbf{I_{2\times 2}}$.

$$C = \sum_{i,j} \rho_h \left( \mathbf{e}_{i,j}^T \Omega_{i,j}^{-1} \mathbf{e}_{i,j} \right) \tag{2.23}$$

BA can be performed in different variations depending on the image frames and map points involved. Some of these variations are the following:

- **Motion Only Bundle Adjustment**: The camera pose, or poses, are optimized based upon the observed map points.
- **Local Bundle Adjustment**: A local window of image frames and map points are optimized
- **Full Bundle Adjustment**: All image frames and map points are optimized.

### 2.3.4 ORB-SLAM2

ORB-SLAM2 is a VSLAM system for monocular, stereo or RGB-D cameras [32]. It is a real-time VSLAM method capable of map reuse, loop closing, and relocalization. It is an sparse in-direct based methods with a BA based back end optimizing the camera poses and map points. The design is based upon its monocular predecessor, ORB-SLAM, where most of the architecture and theory is still used. The following theory section is based upon the theory described in the ORB-SLAM [9] and ORB-SLAM2 [32] papers.

#### 2.3.4.1 System Overview

In this section, the key concepts of ORB-SLAM2 according to Figure 2.13 are explained.

**Parallel Threads**: ORB-SLAM2 runs in three parallel threads with each of its specific objective:

1. **Tracking Thread**: The tracking thread ensures to localize the camera on every incoming frame, and decides when to expand the map by adding keyframes.
2. **Local Mapping Thread**: The local mapping thread manages the local map of keyframes by optimizing the map points using local BA.
3. **Loop Closing Thread**: The loop closing thread corrects for accumulated drift by detecting loops and perform pose-graph optimization. This thread also launches a fourth thread executing full BA to obtain the structure and motion solution.

The parallel threads will in more detail be separately explained in Sections 2.3.4.2 to 2.3.4.4

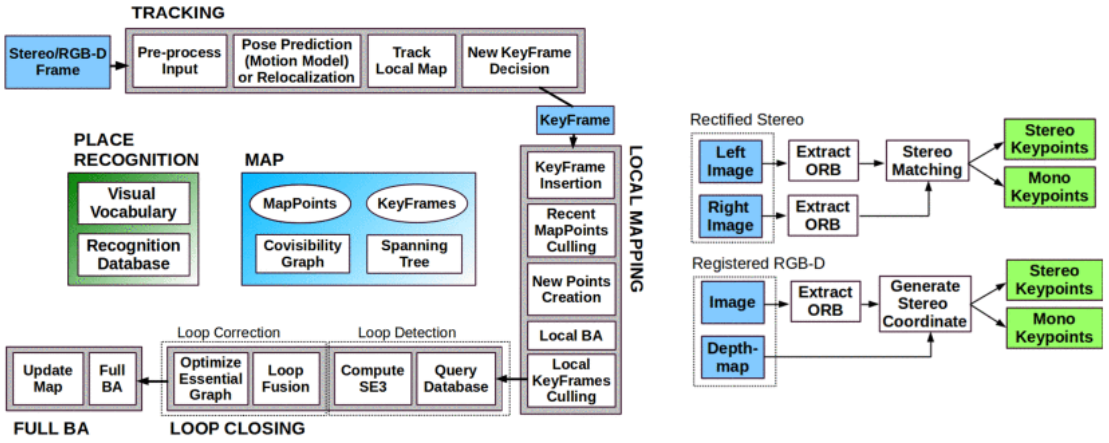



**Figure 2.13:** The system structure of ORB-SLAM2. Left: Three parallel threads, tracking, local mapping and loop-closing. Right: the input preprocessing. Courtesy of [32]

**Orb Features**: ORB-SLAM uses ORB features for tracking, mapping and place recognition tasks, see Section 2.2.1. They are fast to extract and match allowing for real time operation and have good precision in bag of word place recognition

**Keypoints**: Keypoints are the detected features of an image where feature detection has been applied. In the mono camera configuration of ORB-SLAM2, keypoints are extracted as ORB features from every single incoming frame. In the stereo configuration however, the keypoints are extracted from two incoming frames and the keypoints are distinguished by being either monocular or stereo keypoints. See Figure 2.13.

- **Stereo keypoints**: Stereo keypoints are defined by three coordinates $\mathbf{x}_s = (u_L, v_L, u_R)$, where $(u_L, v_L)$ are the coordinates of the left image and $u_R$ are the horizontal coordinate of the right image. ORB features are extracted from each image, and every left ORB is searched for match in the right image. The searching is made efficient by the epipolar constraint assuming fronto-parallel cameras. A stereo keypoints is classified as *close* if its associated depth is less than 40 times the stereo baseline. It is classified as *far* if above. Close keypoints can safely be triangulated from one frame as depth is accurately estimated. The triangulation provides scale, translation and rotation information. Far keypoints provide strong rotational information, but weaker scale and translation. These are only triangulated if supported by multiple views.

- **Monocular keypoints**: Monocular keypoints are defined by two coordinates $\mathbf{x}_m = (u_L, v_L)$ on the left image, and corresponds to all the ORB features for which a stereo match could not be found. These keypoints are triangulated from multiple views, and do not contain any information about scale.

**Map points and Keyframes**: The map points and keyframes are the building blocks of the optimization problems of the parallel threads and consists of the following:

**Map Points**, $p_i$ store the following:

1. Its $3D$ position $\mathbf{x}_i^w$ in the world coordinate system.

2. The viewing direction $\mathbf{n}_i$, which is the mean unit vector of all its viewing direction.

3. A representative ORB descriptor $\mathbf{D}_i$.

4. The maximum and $d_{max}$ and minimum $d_{min}$ distances at which the point can be observed according to the scale invariance limits of the ORB features.

**Keyframes**, $K_i$ stores the following:

1. The camera pose $\mathbf{T}_{iw}$ relative the world coordinate system.

2. The camera intrinsic matrix $\mathbf{K}$.

3. All the ORB features extracted in the frame, either associated with a map point or not.

In ORB-SLAM2, the map points and keyframes are generated generously with a focus on rather removing redundant keyframes, wrongly matched or non-traceable map points afterwards. It allows for flexible map expansion by bounding the map size during revisiting of previously explored areas.

**Covisibilty Graph**: The covisibility graph links the camera keyframes together by their common observed map points. It is an undirected weighted graph, where the nodes are keyframes and the edges are set between keyframes sharing observations of the same map points. The weight, $\theta$, of the edge is the number of common map points. Which for the covisibilty graph is minimum 15.

**Essential Graph**: The essential graph is a reduced version of covisibility graph. It retains all the keyframes of the covisibility graph, but has fewer edges. The edges of the essential graph are the subset of edges from the covisibility graph with high covisibility, $\theta_{min} = 100$, the loop closure edges and the edges of the spanning tree. The spanning tree is a subgraph of covisibility graph having minimal edges. It is built incrementally from the initial keyframe by whenever a new keyframe is inserted, it is connected in the spanning tree with the keyframe that it shares the most observed map points.

**Bag of Words Place Recognition**: The embedded bag of words place recognition module of ORB-SLAM2 is based upon DBoW2 [33]. The bag of words is a technique that uses a visual vocabulary to convert an image into a sparse numerical vector. It allows management of big sets of images, and are in ORB-SLAM2 used to perform loop detection and relocalization. The visual vocabulary is created offline by discretizing the ORB descriptors from a large data set into visual words. The vocabulary is then used online by incrementally building an database that contains an invert index, which stores for each visual word in the vocabulary, in which keyframe it has been observed. The database also accounts for overlapping keyframes and keyframes being deleted. Additionally, the vocabulary is used to speed up the brute force matching in the triangulation of new points by requiring the ORB features to be matched to belong on the same node in the vocabulary tree at a certain level. (2 out of 6).

**Map Initalization**: The map initalization depends whether ORB-SLAM2 is configured as monocular, stereo or RGB-D.

- **Monocular**: The depth for a monocular configuration can not be determined from a

single frame. The map initialization for the monocular case therefor uses a structure from motion approach, computing two parallel geometrical model accounting for initialization either in a planar or general scene.

- **Stereo and RGB-D**: With stereo and RGB-D configuration, the depth is known for every frame. The map is then initialized in the first frame.

#### 2.3.4.2 Tracking Thread

This section gives an outline of the steps of the tracking thread in Figure 2.13. The tracking steps are performed on every received frame.

**ORB Extraction**: On the each incoming frame, or stereo image pair, a pre-selected number of FAST corners are extracted. The number of extracted FAST corners should vary with the image resolution, where images of higher resolution requires a higher number. The FAST corners are extracted on a multiscale pyramid representation of the images, where the default number of levels are eight with a default scaling factor of 1.2. In order to ensure a homogeneous distribution of features, on each scale level the image is divided into a grid where at least a pre-set number of corners is tried extracted from each cell. The amount of required corners extracted from each cell is adapted if no corners are retained. The orientation and ORB descriptor are computed on the retained FAST corners.

**Initial Pose Estimation**: The initial pose estimation depends whether the tracking was successful for the last frame or not.

- **Successful tracking**: If the tracking was successful, a constant velocity model is used to predict the camera pose such that a guided search for map points observed in the last frame can be done. If the number of matches found is not sufficient, a wider search is then conducted. With the found correspondences, the pose is optimized using motion-only BA.

- **Lost Tracking**: If the tracking is lost, a search for candidate keyframes in the bag of words database will be conducted to try obtain global relocalization. Correspondences with ORB associated with map points are in each keyframe computed. If a camera pose with enough inliers is found, the pose is optimized and the tracking continues.

**Track Local Map**: With the initial estimated pose and the initial set of feature matches, the map is projected into the frame, and more map point correspondences are searched for. The complexity is bounded by not projecting the whole map, but only a local map. The local map contains the set of keyframes $\mathcal{K}_1$, which share map points of the current frame, and a set $\mathcal{K}_2$ with neighbors to the keyframes $\mathcal{K}_1$ in the covisibility graph. A reference keyframe, $K_{ref} \in \mathcal{K}_1$, that shares most map points with the current frame is also defined. Each map point seen in $\mathcal{K}_1$ and $\mathcal{K}_2$ are then searched in the current frame according to Algorithm 1. When the searching procedure is complete, the camera pose is optimized

again using the map points found in the frame.

**Algorithm 1:** Searching procedure for finding common map points in the current frame, $\mathcal{K}_1$ and $\mathcal{K}_2$

1. Project the map points $\mathbf{x}$ in the current frame. Discard if outside of image bounds.
2. Calculate the angle between the current viewing ray $\mathbf{v}$ and the map point mean viewing direction $\mathbf{n}$. Discard if $\mathbf{v} \cdot \mathbf{n} < \cos(60°)$.
3. Calculate the distance $d$ from map point to camera center. Discard if it is out of the scale invariance region of the map point $d \notin [d_{min}, d_{max}]$.
4. Calculated the scale in the frame by the ratio $d/d_{min}$.
5. Compare the representative descriptor $\mathbf{D}$ of the map point with the still unmatched ORB features in the frame, near $\mathbf{x}$, at the predicted scale, and associate the map point with the best match.

**New Keyframe Decision**: The last step of the tracking thread is to decided whether the current frame should be delivered as a new keyframe in the local map. The general idea is that keyframes will be inserted as fast as possible to provide robustness, and the culling mechanism in the local mapping thread will remove redundant keyframes later on. The following conditions must be met for a keyframe to be inserted:

1. More than 20 frames must have passed since the last global relocalization.
2. Local mapping is idle, or more than 20 frames have passed from last 20 keyframe insertion.
3. Current keyframe tracks at least 50 points.
4. Current frame tracks less than 90% points than $K_{ref}$

If a keyframe is inserted while the local mapping is busy, the local BA is stopped to further process with the new keyframe.

**Stero Camera Keyframe Decision**: Due to the distinction between close and far points in stereo cameras, an additional constraint is made for keyframe insertion for stereo cameras. It is important to have sufficient amount of close points to accurately estimate translation. If the number of tracked close points drops below $\tau_t$ and the frame can create at least $\tau_c$ new close stereo points, the system will insert a new keyframe.

#### 2.3.4.3 Local Mapping Thread

The following subsection describes the steps done for every newly inserted keyframe $K_i$ performed in the local mapping thread of Figure 2.13.

**Keyframe Insertion**: A new node is added to the covisibility graph for the keyframe $K_i$, the edges of the graph are updated corresponding to the shared map points of $K_i$ with other keyframes. The spanning tree are then updated by linking $K_i$ with the keyframe with most common map points, and the bag of words representation of $K_i$ are computed.

**Recent Map Points Culling**: Map points, after the first three keyframes of being created, must pass a restrictive test that ensures that they are trackable. A point must fulfill the conditions:

1. The tracking must find the point in more than 25% of the frames in which it is predicted to be visible.

2. If more than one keyframe have passed from map point creation, it must be observed from at least three keyframes.

After a point passes this test, it can only be removed if it is observed from less then three keyframes. This can happen when keyframes are culled or when local BA discards outlier observations.

**New Map Point Creation**: New map points are created by triangulating ORB features from connected keyframes $\mathcal{K}_c$ in the covisibility graph. For each unmatched ORB feature in $K_i$, matches are searched for in other unmatched points from other keyframes. This matching is sped up by using bag of words, as explained in Section 2.3.4.1, and points not fulfilling the epipolar constraint are discarded. ORB feature pairs are triangulated, and new points are accepted after checking positive depth in both cameras, parallax, reprojection error, and scale consistency. The new points could also possibly be observed from more keyframes, and are therefore projected into connected keyframes and correspondences are searched for according to Section 2.3.4.2.

**Local Bundle Adjustment**: In the local BA the optimization is conducted on the currently processed keyframe $K_i$, all its connected keyframes in the covisibility graph $\mathcal{K}_c$, and all the map points seen be these keyframes. All other unconnected keyframes that observe these map points are included in the optimization, but remain fixed. The observations that are marked as outliers are discarded at the middle and at the end of the optimization.

**Local Keyframe Culling**: In order to reduce the complexity of the problem, the local mapping tries to detect and remove redundant keyframes. This is beneficial for the bundle adjustment, but also for bounding the problem growth in small environments. Keyframes in $\mathcal{K}_c$ whose 90% of the map points have been observed in a least three other keyframes in same or finer scale, are discarded.

#### 2.3.4.4 Loop Closing Thread

In this section the steps of the loop closing thread of Figure 2.13 is explained. The loop closing thread takes the last frame $K_i$ processed by the local mapping and tries to detect and close loops.

**Loop Candidates Detection**: To detect loop candidates the similarity between the bag of words vector of $K_i$ and all its neighbors in the covisibility graph is computed. The lowest score $s_{min}$ is retained, and every keyframe that scores lower than this in the recognition database is discarded. All the keyframes connected to $K_i$ are discarded from the results. To accept a loop candidate, three consecutive loop candidates that are connected in the covisibility graph must be detected. There can be several loop candidates if there are more places with similar appearance to $K_i$.

**Loop Candidate Validation**: The loop candidate validation depends whether ORB-SLAM is configured as a monocular RGB-D or stereo camera setup:

- **Monocular**: In monocular SLAM the scale is unobservable, and therefore to close

a loop, a *similarity transform* is computed from the current keyframe $K_i$ to the loop keyframe $K_l$ that gives information about the error accumulated in the loop. The computed similarity serves also as geometrical validation of the loop.

- **Stereo and RGB-D**: Since the scale is observable, the geometric validation are based on rigid body transformations instead of similarities.

**Loop Fusion**: If a loop has been validated and confirmed, duplicated map points are fused and a new edges representing the loop is inserted in the covisibilty graph. The method for doing this depends on the monocular, RGB-D or stereo configuration:

- **Monocular**: The current keyframe $K_i$ is corrected with the similarity transform from the "Loop Candidate Validation", and the correction is propagated to all its neighbors aligning both sides of the loop. Map points observed by the loop key frame and its neighbors are projected into $K_i$ and it neighbors. A narrow match search, according to Section 2.3.4.2 is done around the projection. The matched map points and inliers in the computation of the similarity transforms are fused. The edges of the covisibility graph is updated accordingly, creating edges attaching the loop closure.

- **Stereo and RGB-D**: Due to the observable scale, the similarity transform consideration is unnecessary. The loop fusion is based upon rigid body transformations.

**Essential Graph Optimization**: The effects of the loop fusion are put to work by performing pose graph optimization over the Essential Graph. The optimization distributes the loop closing error along the graph. After the optimization each map point is transformed according to the one of the keyframes that observes it.

**Full Bundle Adjustment**: After the pose graph optimization, full BA is initiated in a separate thread to achieve optimal solution. If a new loop is detected while the optimization is running, the full BA is aborted, and a new full BA is launched.

## 2.4 Underwater Imaging

This section presents the challenges in underwater imaging addressed through the underwater imaging process and refraction of light.

### 2.4.1 Underwater Imaging Process

Imaging processes in water are affected by a variety of traits causing a deteriorated performance compared to land based situations. In deep waters, the light reflections recorded by the camera are caused by an artificial light source often close, or following the camera. Before the transmitted photons returns to the camera, they experience a variety of disturbances causing loss of light intensity. These losses are caused by light attenuation, backscatter and small-angle forward scattering [34], and reduces the image quality significantly.

Figure 2.14 illustrates the light intensity loss in the underwater imaging process caused by the light attenuation, backscatter and small-angle forward scattering. The light intensity losses can be categorized into the six sub problems of Figure 2.14: projected light outward scattered (A), projected light attenuated (B), projected light backscattered (C), reflected light attenuated (D), reflected light small angle forward scattered (E), reflected light outward scattered (F).

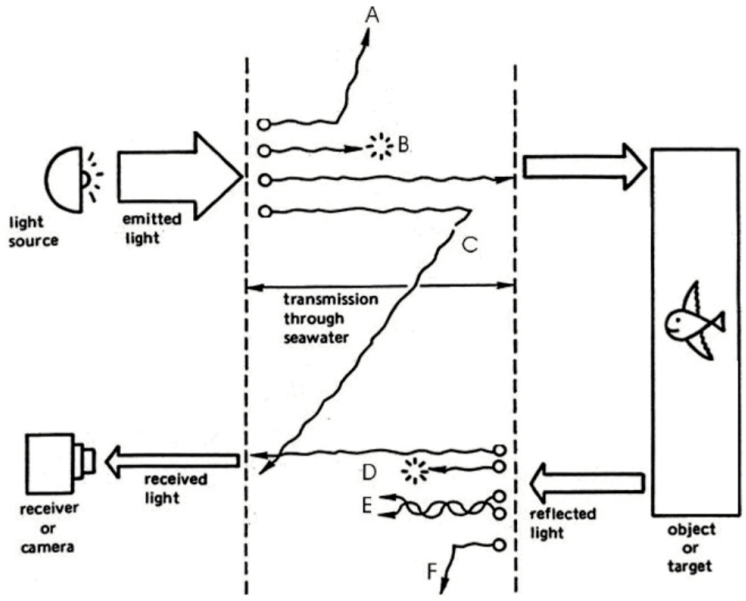



**Figure 2.14:** The light losses in the underwater imaging process [34].

#### 2.4.1.1 Light attenuation

Water selectively attenuates light as a function of wavelength or color [35]. The function depends in the amount of dissolved organic material in the water, but in general is it the longer wavelengths that attenuates the most. In pure water the colors around blue and green are well transmitted while the reds are strongly attenuated. Two of the implications of this are the perceived color of an object will depend on the range of the camera from the target and that reds are difficult to detect at significant range.

#### 2.4.1.2 Backscatter

Backscatter is the amount of reflected light the camera records caused by particles and inhomogenities in the water [34]. The backscatter reduces the image contrasts giving a lower quality image.

#### 2.4.1.3 Small-Angle Forward Scattering

Small-angle forward scattering introduces resolution losses [34]. Scattering is the process by which the direction of individual photons is changed without any other alternation. If the scattered angle is small, the photons reaches the camera and gets recorded. The trajectory of the scattered photon from the seabed to camera is not straight and will cause blur to the image reducing its resolution. The scattering is nearly independent of wavelength in seawater, because of the scattering being mostly caused by the different sizes of particles in the seawater.

### 2.4.2 Refraction of Light - Snell's Law

Refraction of light is the change in direction of light waves passing through an interface of two media of different refraction indexes [36]. The refraction of light follows Snell's Law in Equation (2.24) where $\theta_2$ is the angle of refraction, $\theta_1$ is the angle of incidence, $n_1$ and $n_2$ are the corresponding tabulated refractive indices.

$$\frac{\sin\theta_2}{\sin\theta_1} = \frac{n_1}{n_2} \tag{2.24}$$

Figure 2.15 illustrates a light ray $P$ with $\theta_1$ passing through an interface between two medias of $n_1$ and $n_2$ being refracted into the light ray $Q$. Due to $\theta_2 < \theta_1$ the second media having a higher refractive index ($n_2 > n_1$). It is the equivalent of what is occurring in in underwater imaging, where the larger refraction index of water causes reduced field of view of the cameras compared to in air. The approximate refraction indices of air and water are presented in [37]. Note that these values depend on temperature, pressure and other physical conditions.

**Table 2.1:** Approximate refraction indexes of water and air.

| Refraction Indexes | | |
|---|---|---|
| $n_{air}$ | 1.00029 | [–] |
| $n_{water}$ | 1.33 | [–] |

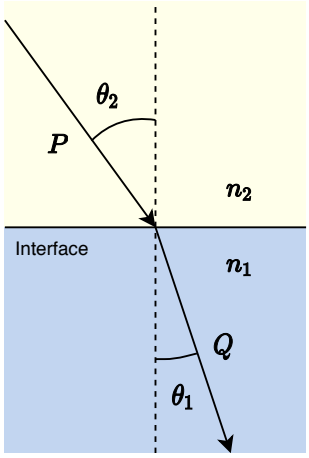



**Figure 2.15:** The refraction of a light ray passing through the interface two media of different refractive indexes $n_1$ and $n_2$.

## 2.5 Point Cloud Processing

This section presents two methods for point cloud processing; the random sample consensus, and clustering.

### 2.5.1 Random Sample Consensus

Random Sample Consensus (RANSAC) [38] is an algorithm that in its essence fits a given model to a set of data points. The assumptions is that the set of data points both contain *inliers* and *outliers*, where the inliers corresponds to the underlying true model and outliers are contaminated data. The fitting is done by repeatedly drawing random samples of data points from the set, fit models to the drawn sample data, and calculated the consensus of the different models with the complete data set. The model with the highest consensus becomes the final model. The procedure can be formalized with five steps presented in Algorithm 2.

**Algorithm 2:** The RANSAC algorithm for fitting a model to a set of data points **S**. Adapted from [18]

1. Select a random sample of **s** data points from the data set **S** and fit a model to this subset.
2. Determine the subset $\mathbf{S}_i$ of **S** that are within a distance threshold $t$ of the model. The set $\mathbf{S}_i$ are the inliers of the sample and is the consensus set.
3. If the number of inlier in $\mathbf{S}_i$ is above than some threshold **T**, re-fit the model using all the points in $\mathbf{S}_i$ and terminate.
4. If the size of $\mathbf{S}i$ is less than **T**, select a new sample subset and repeat 1 ,2 and 3.
5. After **N** trials the largest set $\mathbf{S}_i$ is selected, and the model is re-fitted using all the data points of $\mathbf{S}_i$

#### 2.5.1.1 Threshold Values and Number of Sample Draws

**Distance threshold:** The distance threshold **t** can be selected based upon the the probability distribution of a point being an inlier of the fitted model. Since this distribution is often unknown, **t** is usually chosen empirically. The reader is referred to [18] on the methodology for selecting **t** based upon probability distributions.

**Number of Sample Draws:** The number of samples **N** can be decided based upon the probability $p$ of a random sample of $s$ points only containing inliers. In Equation (2.25), **N** is determined using the probability that a point is an outlier $\epsilon$ and $s$.

$$N = \frac{log(1-p)}{log(1-(1-\epsilon)^s)} \tag{2.25}$$

The probability $p$ is usually set to 0.99 [18], while $\epsilon$ needs to be known. In the case of unknown $\epsilon$, **N** can be determined adaptively by counting the number of outliers for each sample in Algorithm 2 to determine $\epsilon$ and update Equation (2.25) accordingly. The algorithm is initialized using a worst case guess of $\epsilon$.

**Consensus Threshold:** The size of the acceptable consensus set **T** is usually selected based upon a rule of thumb [18], where its size is assumed to be the around the same

number of inliers in **S**. For a known $\epsilon$ and $n$ data points **T** is given in Equation (2.26).

$$T = (1 - \epsilon)n \tag{2.26}$$

In case of unknown $\epsilon$, the termination of Algorithm 2 will be determined by the adaptive **N** in Equation (2.25) removing the dependability of **T**.

#### 2.5.1.2 Estimating a Plane with RANSAC

The RANSAC algorithm can be used to fit a plane to a set of $3D$ points. The plane is defined by its model coefficients $a$, $b$, $c$ and $d$ satisfying Equation (2.27).

$$ax + by + cz + d = 0 \tag{2.27}$$

### 2.5.2 Point Cloud Clustering

In terms of point clouds, clustering is the procedure of segmenting a set of data points into smaller subsets where the data points in each subsets, or clusters, are closer to each other in terms of a given distance metric than the rest of the data set.

In [39] a clustering algorithm is proposed on the basis using the *euclidean distance* as the distance metric and kd-trees for efficient nearest-neighbors search.

For two distinct point clusters $O_i = \{\mathbf{p_i} \in \mathcal{P}\}$ and $O_j = \{\mathbf{p_j} \in \mathcal{P}\}$ Equation (2.28) must hold where $d_t$ is a given maximum imposed distance threshold.

$$\min \|\mathbf{p}_i - \mathbf{p}_j\|_2 \geq d_t \tag{2.28}$$

**Algorithm 3:** An Euclidean based point cloud clustering algorithm as presented in [39].

**Result:** A list of clusters $C$ containing all the points of $\mathcal{P}$

1. Make a kd-tree representation of the input point cloud data set $\mathcal{P}$;
2. Establish an empty list of clusters $C$ and a queue of the points to be checked $Q$;
3. **for** *every* $\mathbf{p_i} \in \mathcal{P}$ **do**
   - add $\mathbf{p_i}$ to the queue $Q$;
   - **for** *every point* $\mathbf{p_i} \in \mathcal{Q}$ **do**
     - Search for the set $\mathcal{P}_i^k$ of point neighbors of $\mathbf{p_i}$ in a sphere with radius $r < d_t$;
     - **for** *every neighbor* $\mathbf{p}_i^k \in \mathcal{P}_i^k$ **do**
       - **if** *point has not been processed* **then**
         - add point to $Q$;
   - When all points of $Q$ has been processed, add $Q$ to the list of clusters $C$ and reset $Q$.
4. The algorithm terminates when all points $\mathbf{p_i} \in \mathcal{P}$ have been processed and are a part of the list of point clusters $C$

**Kd-tree:** A kd-tree [40] is in computer science a data structure that organize points in a $k$-dimensional space. Its use case area are mainly in multidimensional searches, e.g nearest neighbor searches and range searches.

# Chapter 3

# Method

This chapter presents the methodology of establishing the real-time WC-ROV VSLAM system. Section 3.1 presents the overall architecture of the system. Section 3.2 gives an introduction to (ROS), which serves as the framework of the implementation. Section 3.3 presents the hardware of the system, consisting of the stereo camera rig and the processing computers. Sections 3.4 to 3.8 gives individual explanations of the different parts of the system architecture. Section 3.9 explains the calculations performed in order to determine the baseline of the stereo camera rig. Lastly, Section 3.10 presents the conducted camera calibration together with the obtained parameters.

## 3.1 System Architecture of the WC-ROV VSLAM System

The real-time WC-ROV VSLAM system is a VSLAM system designed to be used in conjunction with the stereo cameras of the ROV Minerva. The system produces estimates of the ROV position and orientation, a map of the ROV surroundings and provides the position of the immediate closest obstacle to the ROV. The architecture of the system is presented in Figure 3.1 comprising of: the stereo camera rig of Minerva consisting of two fronto-parallel cameras, a camera driver that administers the stereo camera settings and interprets the camera signals producing stereo image pairs, an image processing part that synchronizes, undistorts and rectifies the stereo image pairs, the renowned VSLAM method ORB-SLAM2 [32] which is the core component of the system, a point cloud processing part inferring the surrounding obstacles and provides the closest detected obstacle, and lastly a LabView communication part distributing the closest detected obstacle to the Autonomy Framework of the ROV Minerva. The different parts were implemented in to separate ROS *nodes*. ROS and ROS nodes are explained in section Section 3.2.

The WC-ROV system was implemented in C++ using the framework ROS on the distribution Kinetic Kame. The system consists of both existing implementation. e.g ORB-SLAM2, and implementation written by the author. In this thesis, the system runs on a desktop computer using Ubuntu 16.04.

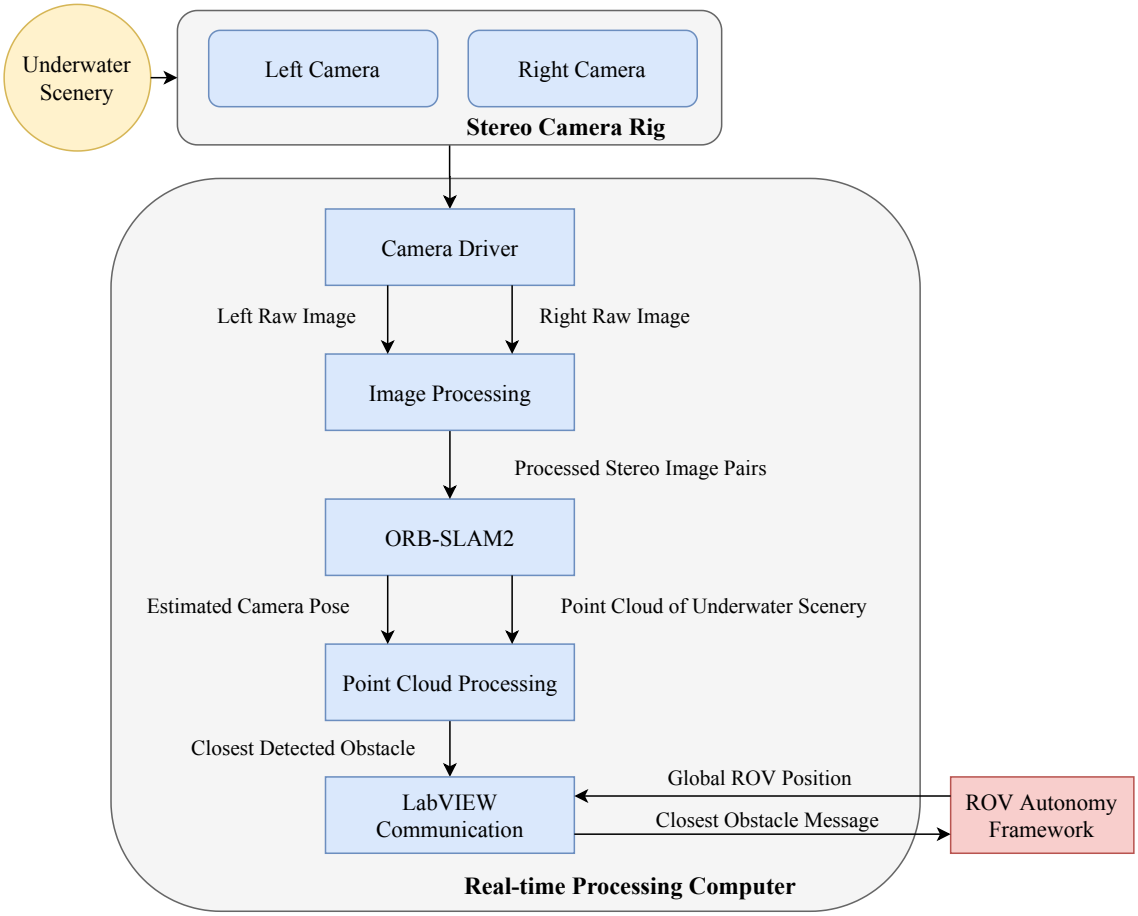



**Figure 3.1:** System architecture.

## 3.2 Robot Operating System

Robot Operating System (ROS)[1] is an open-source framework for robot software development. It provides a wide range of functionalities to efficiently create robot applications, including package management, message-passing between processes, hardware abstraction, and low-level device controls. ROS based systems are usually built consisting of multiple processes communicating using the message-passing capabilities of ROS. The processes are small programs performing system specific tasks e.g. camera driver, data filtering. The message-passing is independent of programming languages, and supports both synchronous and asynchronous message exchange.

The main reason for selecting ROS as the framework of the WC-ROV VSLAM system was due to its use case area of real-time applications and existing implementations relevant to the objectives of this thesis. Additionally, does a standardized framework help transfer implementations and experience for other than the author to use on a later basis. The logging capabilities of ROS also makes it easy for others to utilize the obtained data of this thesis. The following subsections explains in detail about the process communication, logging capabilities and other relevant tools for the system integration within ROS, and how to use it.

[1] `https://www.ros.org`

### 3.2.1 ROS Communication

ROS uses a graph structure to organize the communication between processes. Processes are represented as nodes in a graph, while the connecting edges, termed topics, handles information exchange through message-passing between the nodes. If a node produces an output to be distributed to the system, it *advertises* a topic containing the output. If a node receives input, it *subscribes* to a topic. ROS also contains a database of shared static or semi-static parameters called the *parameter server*. The information exchange is done in the form of ROS Messages. ROS messages are a predefined data packet containing specific data fields for the given message type. Listing 3.1 is an example of ROS image message type

**Listing 3.1:** ROS Image message

```
1   # This message contains an uncompresses image
2
3   Header header           # Header containing timestamp, etc
4
5   # Image parameters
6   uint32 height            # Image height, number of rows
7   uint32 width             # Image width, number of columns
8   string encoding          # Encoding of pixels
9
10  uint8 is_bigendian       # is data bigendian
11  uint32 step              # Full row length in bytes
12  uint8[]                  # Actual matrix data
```

#### 3.2.1.1 ROS Master

Every ROS system is initialized using the process called the *ROS Master*. The ROS master sets up peer-to-peer communication between nodes using its register of published and subscribed topics, and keeps track of the global parameters of the system using the parameter server. Each time a node has been initialized, it registers itself to the ROS Master informing about its current subscribed and/or published topics. The decentralized architecture is illustrated in Figure 3.2 where the ROS Master is providing the connection information to the two nodes *talker* and *listener*.

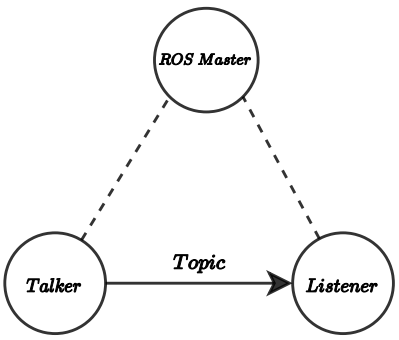



**Figure 3.2:** The node talker publishing to the subscribing node listener. The communication is registered by ROS Master

### 3.2.2 ROS Launch

ROS nodes are separate processes and needs to be run in separate terminals. The package *roslaunch* provides the possibility if launching multiple nodes using roslaunch configuration files, or lauch file. Additionally can the parameters, or static variables, in the parameters server of the ROS system be set using the launch file. Launch files for a ROS system startup can be initiated using the following command

**Listing 3.2:** Launch a ROS system using roslaunch

```
1   $ user@hostname roslaunch ros_configuration.launch
```

### 3.2.3 Recording of Data in ROS

ROS message data can be recorded and played back using the ROS package *rosbag*. Rosbag logs subscribes topics into a file format called bags, which allows for messages to be played back in a manner that replicates the original publishing of the recorded nodes. Rosbag is command line based, and to initate a recording of a specific set of topics the following command can be used

**Listing 3.3:** Record subscribed topics using rosbag

```
1   $ user@hostname rosbag record topic1 topic2 topic3
```

To play previous recorded messages

**Listing 3.4:** Play previous recorded topics using rosbag

```
1   $ user@hostname rosbag play name_recording.bag
```

## 3.3 Hardware

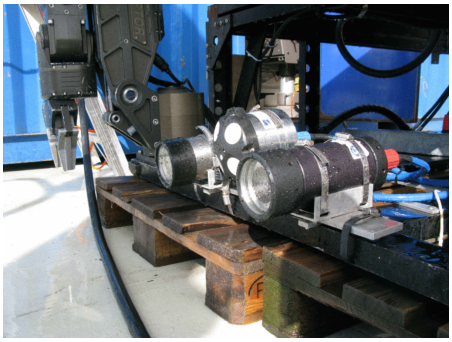
The hardware of the WC-ROV VSLAM system consists of the stereo camera rig of Minerva, a desktop computer for real-time processing of the system, and a laptop computer for camera calibration and experiment data collection.

### 3.3.1 Stereo Camera Rig

The stereo camera rig of Minerva consists of two Allied Vision Prosilica GC 1380C mounted horizontally displaced in a fronto-parallel manner. The GC 1380C cameras by Allied Vision [41] are suitable for underwater operation due to their high sensitivity to lightning and low signal to noise ratio. They are capable of recording at a resolution of $1360 \times 1024$ with a frame rate of 20 frames per second. The cameras have a global shutter and allows for increased framerates by reducing the resolution height. Additional camera specifications i.e. exposure time, gain level and pixel format can be set by accessing the in-camera settings. The data is transmitted from the cameras by a GigE Vision compliant Gigabit Ethernet interfaces (IEEE 802.3 1000 baseT) allowing a bandwidth of 125 MB/s.

The cameras of the stereo camera rig are installed in waterproof casings with interfaces from Ethernet ($RJ45$) to SubConn 13-pin bulkhead connector ($DBH13F$) to allow con-

nection with the ROV standardized framework. The waterproof casings are mounted to a rail allowing the horizontal displacement of the cameras to be adjusted, hence changing the baseline. See Figure 3.3a for the mounting rail.



**(a)** The stereo camera rig mounted on the SUB-fighter 30K ROV during a mission in Trondheimsfjorden, courtesy of [42]

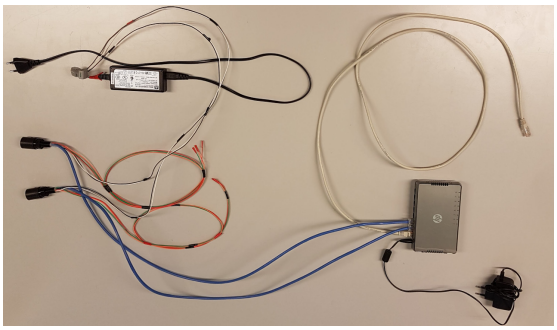

**(b)** The Prosilica GC1380C of the stereo camera rig, courtesy of Allied Vision [41]

**Figure 3.3:** The stereo camera rig.

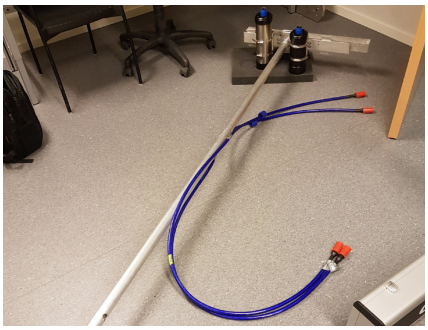
A lab kit and a rod mount for the stereo camera rig was established to accommodate the laboratory camera calibration and testing of the WC-ROV VSLAM system.

**Lab Kit:** The lab kit of the stereo camera rig allows for the cameras to be accessed by a computer while residing in the waterproof casing. The kit consists of an $DBH13F$ to $RJ45$ (Ethernet) cable with an attached a 12 V, and a 2.5 [m] DIL13F to DIL13M cable extender. Both of the cameras are accessed simultaneously by using an Gigabit Ethernet switch. The kit is presented in Figure 3.4a.

**Stereo Camera Rig Rod Mount:** The rod mount for the stereo camera rig makes it possible to perform underwater translation and orientation of the stereo camera rig during laboratory experiments. The mount consists of a 2 m long aluminum rod with a bracket at the end for mounting the stereo camera rig. The rod mount is displayed in Figure 3.4b.



**(a)** Lab kit



**(b)** Rod mount

**Figure 3.4:** The lab kit and rod mount of the stereo camera rig.

### 3.3.2 Desktop Computer for Real-time Processing

The WC-ROV VSLAM system is run real-time on a desktop computer provided by NTNU AUR-Lab. The desktop computer uses an Intel Core i7-7820X CPU running at 3.60GHz with 16 cores and has 62.3-GB RAM. The computer is intended to be used on bard Gunnerus during the field experiments.

### 3.3.3 Laptop Computer for Data Collection

The data collection in the laboratories experiments are conducted using a laptop computer. The use of a laptop is due to the limited practicality of using a desktop computer while maneuvering in confined spaces. The laptop computer is a Lenovo X230 using an Intel Core i5-3210M CPU running at 2.50Ghz with 4 cores and 7.5-GB RAM.

## 3.4 Camera Driver

The camera driver of the WC-ROV VSLAM system is the first ROS node of the pipeline. It converts the signals received from the stereo cameras to ROS image messages, and administers the settings of the cameras. The camera driver is integrated in system using an existing ROS camera driver implementation[2] based on the Software Developing Kit (SDK) provided by Allied Vision.

### 3.4.1 Camera Settings

The different cameras setting of the Prosilica GC1380C were addressed in the manners presented in the following paragraphs. Most notably is the two different resolution modes, which has contributed to the foundation for testing the WC-ROV VSLAM system.

**Resolution Modes:** The cameras were arranged into operating in two different pixel resolutions modes; *full resolution mode* and *binned mode*. The full resolution mode utilized the cameras full pixel capacity giving maximum information per image. At this resolution, the framerate of the cameras were capped at 20 Hz. The binned camera mode used pixel binning. Pixel binning is the process of combining the electric charge from adjacent sensor pixels of the camera. The process enhances the signal-to-noise ratio, and is especially advantageous in areas width low-light conditions [43]. The binned mode of the cameras were set to combine adjacent sensor pixel both in the vertical and horizontal direction resulting in halved the pixel resolution. Due to the lower resolution, the maximum framerate of the binned mode was 45 Hz. It is worth mentioning that the exposure time of the binned mode needed to be considerably less as the pixel binning provides double the light sensitivity. The resulting camera resolutions corresponding maximum framerate are presented in Table 3.1.

**Table 3.1:** The resolutions of the stereo camera resolution modes.

| | **Full Resolution** | **Binned Mode** | |
|---|---|---|---|
| $w$ | 1360 | 680 | $[px]$ |
| $h$ | 1024 | 512 | $[px]$ |
| $FPS_{max}$ | 20 | 45 | $[Hz]$ |

[2] `http://wiki.ros.org/avt_vimba_camera`

**Pixel Format:** The cameras were configured into sending images on the monochrome image format *MONO8*. The image format MONO8 gives grayscale images where each pixel contains a single byte giving the pixel intensity. The image format MONO8 was selected instead of other camera capable color pixel formats, i.e RGB, due to the ORB features of ORB-SLAM2 being designed to work on grayscale images.

**Exposure Time:** The exposure times of the stereo cameras were selected prior to each time they were operated due to the varying illumination conditions of the operation locations. Prior to the camera operations, the necessary exposure times was helped being determined by using the auto focus feature of the camera software *Vimba Viewer*[3] provided by Allied Vision. The determined exposure times will be presented in this thesis when relevant.

**Gain:** The gain setting of the cameras was consequently set to 0 to reduced the camera noise.

**Lens Focusing Distance:** The focusing distances of the lenses of the Prosilica GC1380 cameras were adjustable. To accompany the operating distance of the ROV, the lenses were manually adjusted to a focusing distance of 3 m. The adjustment was conducted by the use of a high contrast optical target placed in a measured distance of 3 m away from the cameras.

### 3.4.2 Bandwidth Calculations

The joint bandwidth usage of the stereo cameras were calculated with the determined camera settings in order to ensure that there was no loss in the transmission of the stereo image data. The bandwidth usage was calculated using Equation (3.1), where $n_{cam}$ is the number of cameras, $FPS$ (Frames Per Second) is the camera framerate, $h_r$ and $w_r$ are the pixel height and width, and $^{bytes}/_{pixel}$ is the number of bytes per pixel.

$$Bandwidth\ usage = n_{cameras} \times FPS \times h_r \times w_r \times {}^{bytes}/_{pixel} \tag{3.1}$$

By inserting the height, width and maximum framerate for both the full resolution and binned mode in Equation (3.1), the maximum bandwidth usages were correspondingly 55.7 MB/s for the full resolution mode, and 31.3 MB/s for the binned mode. It confirmed that the bandwidth capacity of the Gigabit interface was not violated, and hence the framerate could later be selected freely without considering the bandwidth usage.

## 3.5 Image Processing

The image processing of the WC-ROV VSLAM system consists of synchronization, contrast enhancing, stereo rectification and correction of lens distortion. The processing was implemented in the image processing node of the ROS architecture, subscribing to the raw stereo images of the camera drive and publishing processed stereo image pairs. The image processing pipeline is displayed in Figure 3.5. For each received raw image pair, a callback is triggered processing the incoming pair and publishing the processed version. The open source library OpenCV[4] was used to implement the processing pipeline.

[3] `https://www.alliedvision.com/en/products/software.html`
[4] `https://opencv.org/`

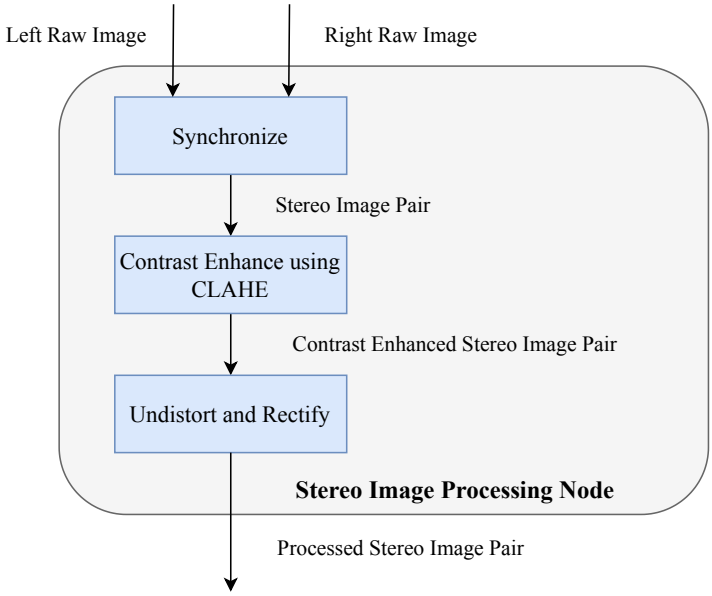



**Figure 3.5:** The image processing pipeline.

### 3.5.1 Synchronization

The synchronization of the incoming raw left and right image was performed using the existing ROS package *message filters*[5]. The message filters works by subscribing to two or more topics and initiating a common callback function when the timestamps of the subscribed topics matches.

### 3.5.2 Contrast Enhancing

The contrast enhancing of the stereo image pairs was conducted using CLAHE of Section 2.2.0.2. The method CLAHE was selected due to the uneven lighting conditions of ROVs working subsea caused by their artificial illumination. The CLAHE was integrated in the image processing node using the class `CLAHE` of OpenCV. Contrast enhancing is applied on both images using a predetermined clip limit and tile size of the CLAHE. The selected clip limit and tile size is stored in the parameter server of the WC-ROV VSLAM system and can be changed by altering the ROS launch file. The value of the clip limit depended on the current conditions, where dark scenes required a higher clip limit. The tile sizes were selected based upon the aspect ratio of the image resolution in order to reduce the occurrence of image anomalies caused by successive overlapping local histograms. The size of the tile was selected based upon how affected the current scene was of combined dark and bright regions, where larger til sizes reduces overbrightening effect, and smaller tile sizes gave a contrast enhancing similar to normal histogram equalization.

Figure 3.6 displays CLAHE of the image processing node applied on an image with severe uneven lighting. The clip limit is set to 5, while the tile is a non-aspect ratio defined small square of size 5. The final selected CLAHE parameter values is presented in Chapter 5 with their accompanying data sets.

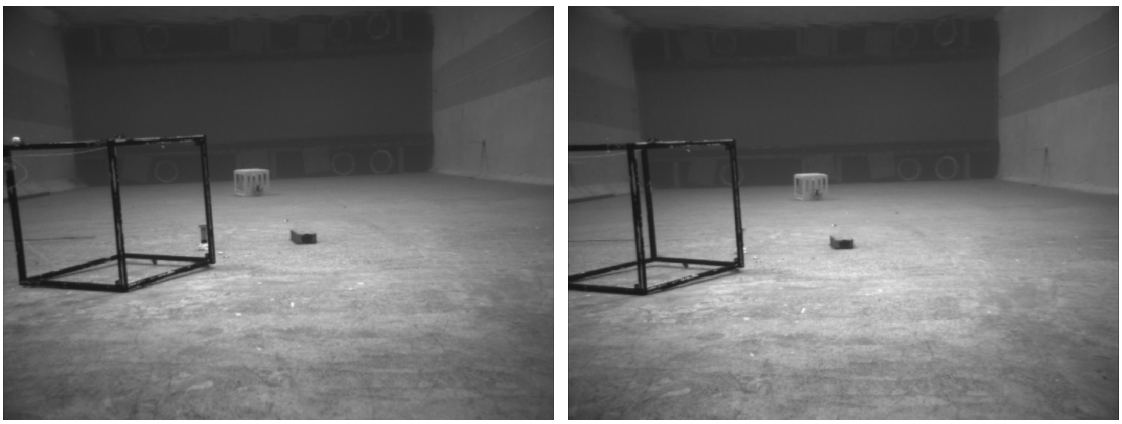
[5] `http://wiki.ros.org/message_filters`

**(a)** Unprocessed. **(b)** Processed with CLAHE.

**Figure 3.6:** CLAHE applied on an underwater image with uneven lighting. Clip limit = 5. tile size $5 \times 5$

### 3.5.3 Distortion and Rectification

The rectification and undistortion of the stereo image pairs was implemented by creating a mapping from the raw left and right images to rectified and undistorted stereo image pairs using a set of predetermined projection transform matrices, the camera intrinsic parameters and the camera distortion coefficients. The mappings were created using the function `initUndistortRectifyMap` of OpenCV, where the projection transform matrices are established using the intrinsic and extrinsic parameters, and the distortion coefficients of the stereo camera using the OpenCV function `stereoRectify`. The projection transform matrices, intrinsic matrices and distortion coefficients used in the map establishement are stored in a `yaml` file which is loaded on the initialization of the system. The mappings applied to a stereo image pair is displayed in Figures 3.7 and 3.8, where Figure 3.7 is the unprocessed stereo image pair, and Figure 3.8 is the contrast enhanced, undistorted and rectified stereo image pair.



**(a)** Left camera **(b)** Right camera

**Figure 3.7:** An unprocessed stereo image pair.

**(a)** Left camera **(b)** Right camera

**Figure 3.8:** The stereo image pair of Figure. FIGURE contrast enhanced, undistorted and rectified.

## 3.6 ORB-SLAM2

The VSLAM algorithm ORB-SLAM2 [32] is the core of the WC-ROV VSLAM system by performing the estimation of the stereo camera rig position, orientation and point cloud of the surrounding environment. The reader is referred to Section 2.3.4 for an extensive explanation of the theory and details about the algorithm. ORB-SLAM2 was integrated in the WC-ROV VSLAM system using the existing ROS implementation of ORB-SLAM2 provided by *Applied AI Initiative*[6]. The ROS implementation was heavily based upon the original open source ORB-SLAM2 implementation[7] provided by the author of ORB-SLAM2. The ORB-SLAM2 node of the WC-ROV VSLAM system subscribes to the synchronized, contrast enhanced, and rectified stereo image pairs provided by the image processing node, and publishes itself the estimated stereo camera rig pose and the estimated point cloud of its surrounding environment. The ORB-SLAM2 node publishes at the frequency of either the currently set framerate of the camera driver node, or at its own rate of computation. The rate of computation is strongly dependent on the resolution of the received processed stereo image pairs. The published position, orientation and point cloud are defined in a local frame with an origin at position of the WC-ROV VSLAM system initiation. The local frame is in this thesis referred to as the local VSLAM frame $\mathcal{F}_l$.

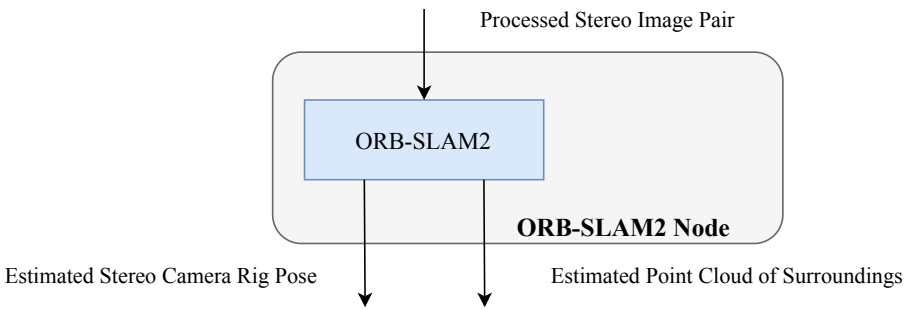



**Figure 3.9:** ORB-SLAM2 pipeline.

[6] https://github.com/appliedAI-Initiative/orb_slam_2_ros
[7] https://github.com/raulmur/ORB_SLAM2

**ORB-SLAM2 Parameters:** The ORB-SLAM2 algorithm has a variety of static parameters that can be set in the system ROS launch file. These parameters are *nFeatures*, *scaleFactor*, *nLevels*, *iniThFAST*, *minThFAST*, and *ThDepth*. The first five are parameters are related to the extraction of ORB features: nFeatures sets the number of ORBs to be extracted on each of the two incoming stereo image pairs, nLevels sets the number of levels in the ORB image scaling pyramid, iniThFAST and minThFAST sets the maximum and minimum number of FAST corners detected in the grid divided scale level. See Section 2.3.4.2 for more details. The ThDepth sets the threshold of stereo keypoints being defined as close or far points, see Section 2.3.4.1.

The ORB-SLAM2 does additionally publish a debug stream showing the tracked map points at each incoming frame. An example from the debug stream is shown in Figure 3.10.

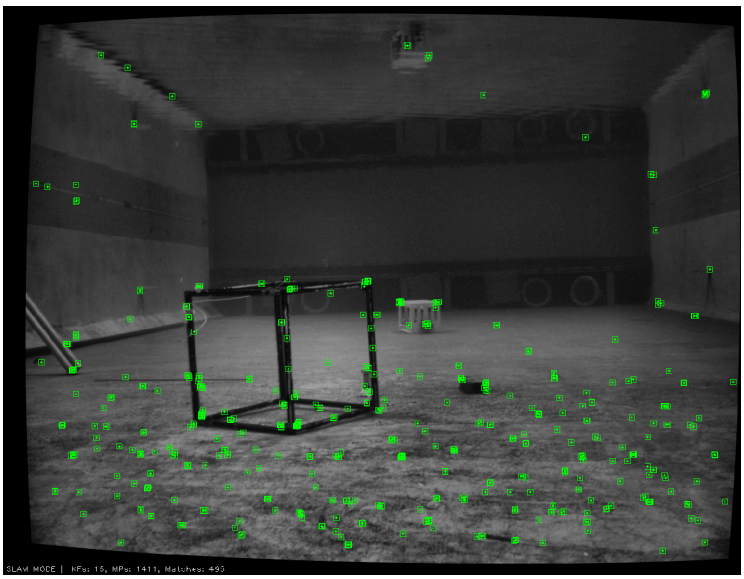


**Figure 3.10:** The debug screen published by ORB-SLAM2.

## 3.7 Point Cloud Processing

The objective of the point cloud processing of the WC-ROV VSLAM system is to identify the closest detected obstacle and determine its relative position and size. The point cloud processing was implemented in the point cloud processing node using the C++ library *PCL*[8]. The node subscribes to the estimated stereo camera rig pose and point cloud published by the ORB-SLAM2 node. The point cloud processing node publishes itself a custom ROS message containing the closest detected obstacle position and dimension, the stereo camera rig position, and the distance between the closest detected obstacle and stereo camera rig. For each incoming point cloud message, the processing is conducted. The point cloud processing pipeline is presented in Figure 3.11 and consists of estimation and removal of seabed points, clustering of the remaining points to identify obstacles, and determining and publishing the currently closest obstacle to the stereo camera rig.

[8] `https://pointclouds.org/`

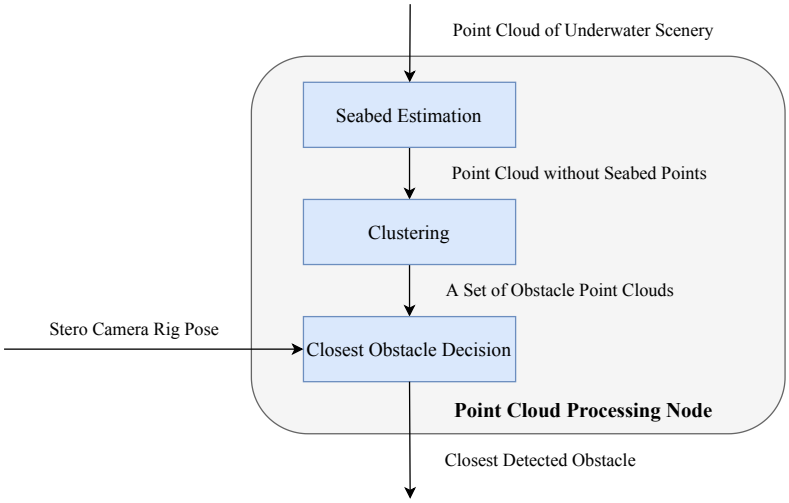
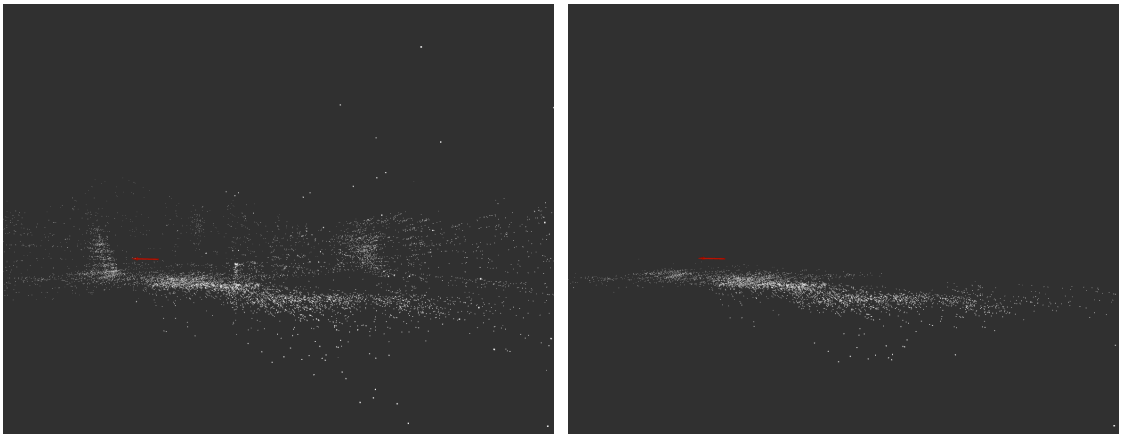



**Figure 3.11:** The point cloud processing pipeline.

### 3.7.1 Seabed Estimation

The first part of the point cloud processing pipeline is the removal of the points associated with the seabed. The seabed points are identified by fitting points to a plane perpendicular to the z-axis. The model of the plane are estimated using RANSAC of Section 2.5.1 for a given distance threshold value $\mathbf{t}$, and given slack angle of the plane being perpendicular to the z-axis set in the launch file. All the points within the distance threshold value of the estimated plane are then removed from the point cloud before further processing. The RANSAC is conducted using the class `EuclideanClusterExtraction` of PCL. Figure 3.12b shows the points the point cloud associated with an estimated seabed plane of an original point cloud provided by the ORB-SLAM2 node in Figure 3.12a. The plotting tool used to make figures is the ROS tool *Rviz*[9]-



**(a)** Point Cloud produced by ORB-SLAM2. **(b)** Estimated seabed points of Figure fig. 3.12a

**Figure 3.12**

[9] `http://wiki.ros.org/rviz`

### 3.7.2 Clustering

The second part of the point cloud processing pipeline is obstacle identification by conducting clustering. The point cloud with removed seabed points is segmented into clusters of points using the Euclidean based clustering algorithm of Section 2.5.2. The algorithm is implemented in the class `EuclideanClusterExtraction` of PCL and was used in the integration of the WC-ROV VSLAM system. The cluster point distance tolerance, the minimum and maximum cluster size are set in the launch file. The resulting clusters of the point cloud are assumed to be obstacles of the mapped surroundings. The obstacles are modeled as cuboids where the dimensions of the cuboids $\mathbf{o_d} = (l, w, h)$ are determined using the maximum and minimum values of the points contained in the cluster according to

$$\begin{aligned} l &= x_{max} - x_{min} \\ w &= y_{max} - y_{min} \\ h &= z_{max} - z_{min} \end{aligned} \tag{3.2}$$

The obstacle positions $\mathbf{o_p} = (x_o, y_o, z_o)$ are determined using the mean of the maximum and minimum points of the cluster

$$\begin{aligned} x_p &= \frac{x_{max} + x_{min}}{2} \\ y_p &= \frac{y_{max} + y_{min}}{2} \\ z_p &= \frac{z_{max} + z_{min}}{2} \end{aligned} \tag{3.3}$$

Figure 3.13 shows a sample of the clustering node estimating cuboid clusters, plotted as green cubes, from point clouds received from ORB-SLAM2. The cluster point tolerance is set to 0.15 m and the minimum and maximum cluster size is respectively set to 20 and 10000. The plot was made using Rviz.

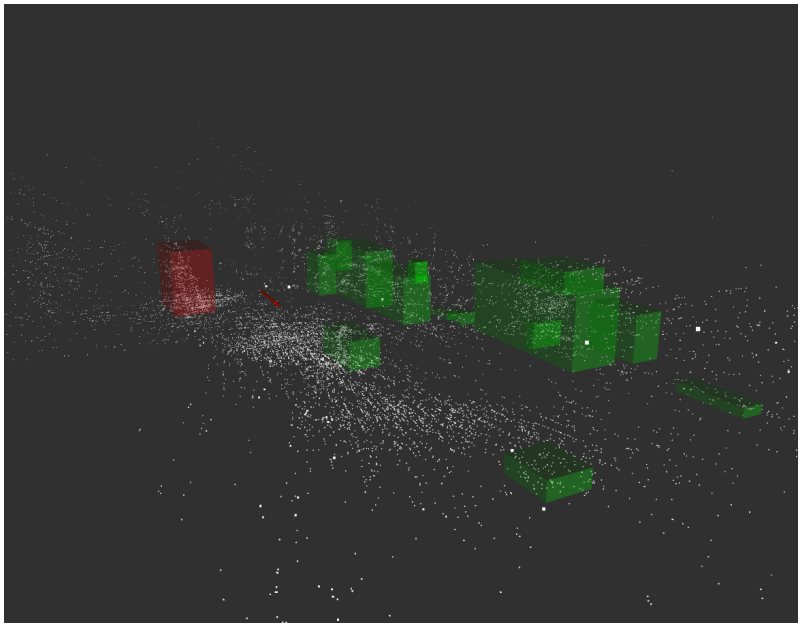


**Figure 3.13:** The identified cuboid obstacles (green), closest detected obstacle (red), and current estimated stereo camera pose (red arrow).

### 3.7.3 Closest Obstacle Decision

The last part of the point cloud processing pipeline is determining the closest obstacle in the set of detected obstacles. For each cuboid defined obstacle, the distance from the current estimate stereo camera rig position $\mathbf{c}_p = (x_c, y_c, z_c)$ is calculated to each cuboid face and corner position $\mathbf{o}_{f/c} = (x_{f/c}, y_{f/c}, z_{f/c})$ using the Euclidean distance in Equation (3.4).

$$d(\mathbf{c}_p, \mathbf{o}_{f/c}) = \sqrt{(x_c - x_{f/c})^2 + (y_c - y_{f/c})^2 + (z_c - z_{f/c})^2} \tag{3.4}$$

The smallest of the calculated face and corner distances $d(\mathbf{c}_p, \mathbf{o}_{f/c})$ becomes the distance between the obstacle and camera. The obstacles in the obstacle set with the closest distance to the camera is the resulting closest detected obstacle. The custom ROS message in Listings. 3.5 is then generated and published containing the current stereo camera rig position, the closest detected obstacle position and dimensions, and the distance between the camera and closest detected obstacle.

**Listing 3.5:** ROS closest obstacle message

```
1  # The position and distance is given in the VSLAM frame
2
3  Header header           # Header containing timestamp, etc
4
5  geometry_msgs/Point positionVSLAM     # Position of stereo camera
6  geometry_msgs/Point positionObstacle  # Position of obstacle
7
8  float64 distance                  # Distance between camera and obstacle
9  float64 dimensionX                # Obstacle cuboid length
10 float64 dimensionY                # Obstacle cuboid width
11 float64 dimensionZ                # Obstacle cuboid height
```

## 3.8 LabView Communication

The LabView communication of the WC-ROV VSLAM system handles the information exchange with the autonomy framework of Minerva. The WC-ROV VSLAM system receives the global ROV position, while sending back information about the closest detected obstacle to the ROV autonomy framework. Obstacle information is only sent given that a distance threshold of 5 m is satisfied. The LabView communication was implemented in the LabView communication node establishing TCP (Transmission Control Protocol) connections using sockets in C++[10]. The LabView communication node subscribes to the closest detected obstacle ROS message published by the point cloud processing node, and consists of the two threads *Obstacle Callback* and *TCP Protocol*. Each thread operates asynchronously on separate frequencies, where the frequency of the Obstacle Callback thread matches the frequency of the WC-ROV VSLAM system (either the framerate or the ORB-SLAM2 computational time), and the TCP Protocol thread matches the decided LabView communication frequency. The pipeline, and the interlinking of the two threads are displayed in Figure 3.14.

[10] `http://www.linuxhowtos.org/C_C++/socket.htm`

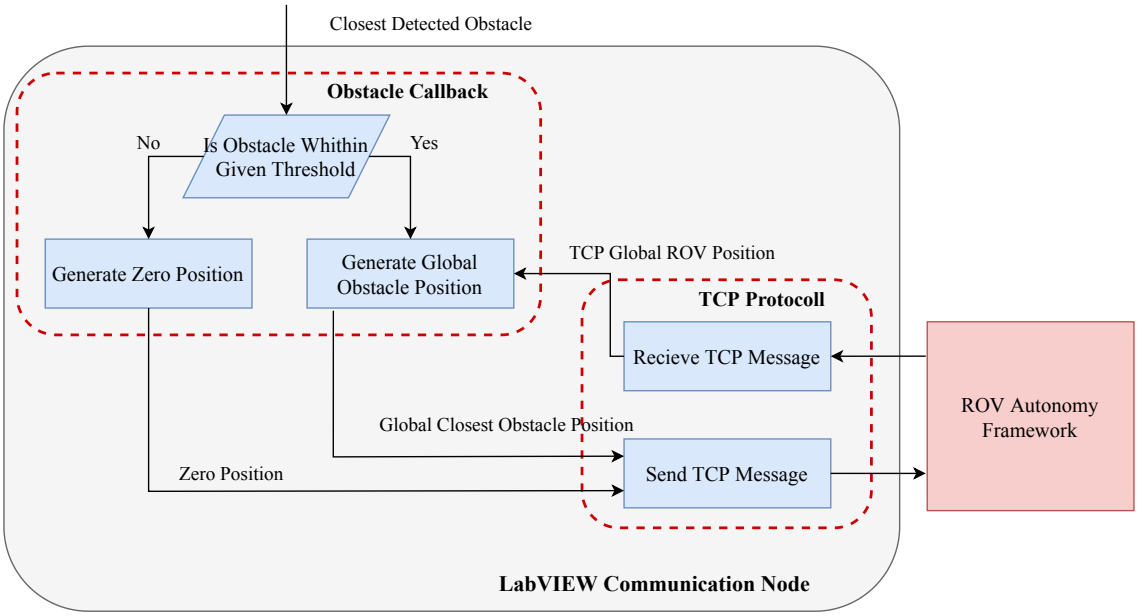



**Figure 3.14:** The LabView communication pipeline.

### 3.8.1 TCP Protocol

The TCP Protocol thread conducts the TCP communication according to the communication protocol decided with the team of master students working on ROV Minerva Autonomy Framework, see Section 1.5. The autonomy framework in LabView act as the TCP server, while the WC-ROV VSLAM system is a TCP client connecting to the server IP-address and port number. The protocol starts by LabView sending a string of the global ROV position to the WC-ROV VSLAM system on the format in Equation (3.5) where $x_{ROV}$, $y_{ROV}$ and $z_{ROV}$ are the ROV position in the North-East-Down (NED) global frame $\mathcal{F}_g$. The string is converted to the translation vector $\mathbf{t}_{gr}^{g}$ of the ROV body frame $\mathcal{F}_r$ relative to $\mathcal{F}_g$.

$$\$x_{ROV}^{g}, y_{ROV}^{g}, z_{ROV}^{g}; \tag{3.5}$$

The protocol continues by the WC-ROV VSLAM system sending information about the closest detected obstacle to LabVeiw if the distance threshold is violated. The obstacle avoidance of LabView expects spherical obstacles, and the message sent contains the global obstacle positions $x_o^g$, $y_o^g$ and $z_o^g$, and the sphere diameter $d_o$. The message is a four element array of doubles on the format given in Equation (3.6).

$$\left[\ x_o^g,\quad y_o^g,\quad z_o^g,\quad d_o\ \right] \tag{3.6}$$

If there are no detected obstacles within the distance threshold, the obstacle message sent to LabView only contains zeros. The TCP Protocol thread operates at the frequency of 2 Hz.

### 3.8.2 Obstacle Callback

The Obstacle Callback thread converts the closest obstacle ROS messages received from the point cloud processing node to TCP messages containing spherical obstacles. For every incoming closest obstacle ROS message, the distance threshold is checked. If the obstacle distance is larger than the threshold, a TCP message containing only zeros is

generated. If the obstacle distance is lower than the threshold, a TCP message on the format of Equation (3.6) containing the global obstacle position and its spherical diameter is generated.

**Global Obstacle Position:** The global position of the obstacle $\mathbf{x}_o^g = (x_o^g, y_o^g, z_o^g)$ is determined by conducting coordinate translation transformation from the local VSLAM frame $\mathcal{F}_l$ to the global NED frame $\mathcal{F}_g$ on the local VSLAM obstacle position $\mathbf{x}_o^l$. First, the obstacle position in the stereo camera rig frame $\mathcal{F}_c$ is determined by Equation (3.7), where $\mathbf{x}_c^l$ and $\mathbf{x}_o^l$ are the stereo camera rig position and obstacle position in the local VSLAM frame $\mathcal{F}_l$. Both $\mathbf{x}_c^l$ and $\mathbf{x}_o^l$ are contained in the closest obstacle ROS message.

$$\mathbf{x}_o^c = \mathbf{x}_o^l - \mathbf{x}_c^l \tag{3.7}$$

The global position of the obstacle $\mathbf{x}_o^g$ is then found by adding the translation vector $\mathbf{t}_{gr}^g$ of the ROV body frame $\mathcal{F}_b$ relative to the global NED frame $\mathcal{F}_g$, the translation vector $\mathbf{t}_{rc}^r$ of the stereo camera rig frame $\mathcal{F}_c$ relative to the ROV body frame $\mathcal{F}_r$, and the obstacle position $\mathbf{x}_o^c$ given in $\mathcal{F}_c$.

$$\mathbf{x}_o^g = \mathbf{t}_{gr}^g + \mathbf{t}_{rc}^r + \mathbf{x}_o^c \tag{3.8}$$

No rotation transform is needed in the coordinate transformation from $\mathcal{F}_c$ to $\mathcal{F}_r$ due to the frames being aligned. The coordinate frames along with their translation transformation vectors are displayed in Figure 3.15.

**Diameter:** The sphere diameter $d_o$ of the closest detected obstacle is determined by selecting the largest of the cuboid dimensions $\mathbf{o_d} = (l, w, h)$ contained in the received closest obstacle ROS message.

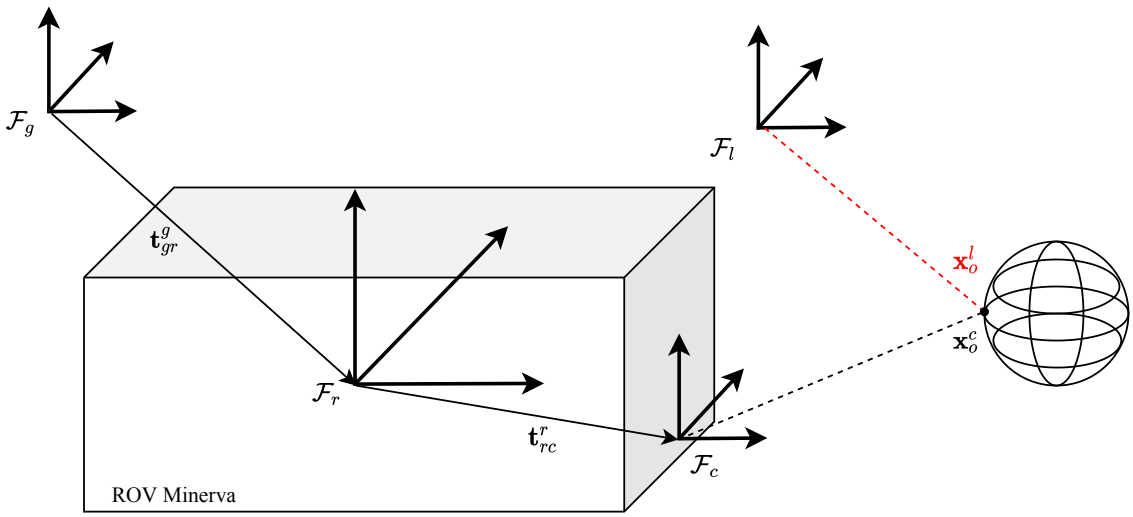



**Figure 3.15:** The coordinate frames of the WC-ROV VSLAM system.

## 3.9 Baseline Selection

In this section the calculations on determining the stereo camera rig baseline and the selected baseline is presented. The baseline of the stereo camera rig influences a variety traits concerning the performance of the stereo imaging algorithms of ORB-SLAM2. A wider baseline gives higher range of depth, while a narrower baseline grants higher overlapping field of view (OFOV), and a lower minimum stereo observable distance between the stereo camera and scene. The baseline selection can be considered as a trade-off, and was in this thesis selected based upon the philosophy of having the widest possible baseline as long as the OFOV and minimum distance are not too critically affected. The baseline was decided by performing calculations on the OFOV and the range of depth of the stereo camera in both full resolution and binned mode.

### 3.9.1 Overlapping Field of View

The OFOV depends on the current baseline $b$ and the distance between the observed scene and the stereo camera $d_{OFOV}$. The OFOV in water is affected by the reduction of the single camera field of view $\theta$ due to refraction in water. By assuming stereo geometry, fronto-parallel cameras and symmetrical $\theta$ along the principle axis on both cameras, the refraction corrected overlapping field of view $OFOV_{corr}$ can be modeled as illustrated in Figure 3.16. The `MATLAB` script `FieldOfViewCalculations.m` in Appendix B estimates the percentage $OFOV_{corr}$ of the total refraction corrected overlapping field of view $TOFOV_{corr}$ at a given distance $d_{OFOV}$ and baseline $b$. The calculations uses the in air calibration focal lengths $f_{air_l}$ and $f_{air_r}$, the camera resolution width $w$ and the refractive indices of air and water $n_{air}$ and $n_{water}$ from Table 2.1. Additionally does the script output the minimum stereo observable distance $d_{min}$ for the current given baseline $b$. The script performs the estimations by following the steps of Algorithm. 4.

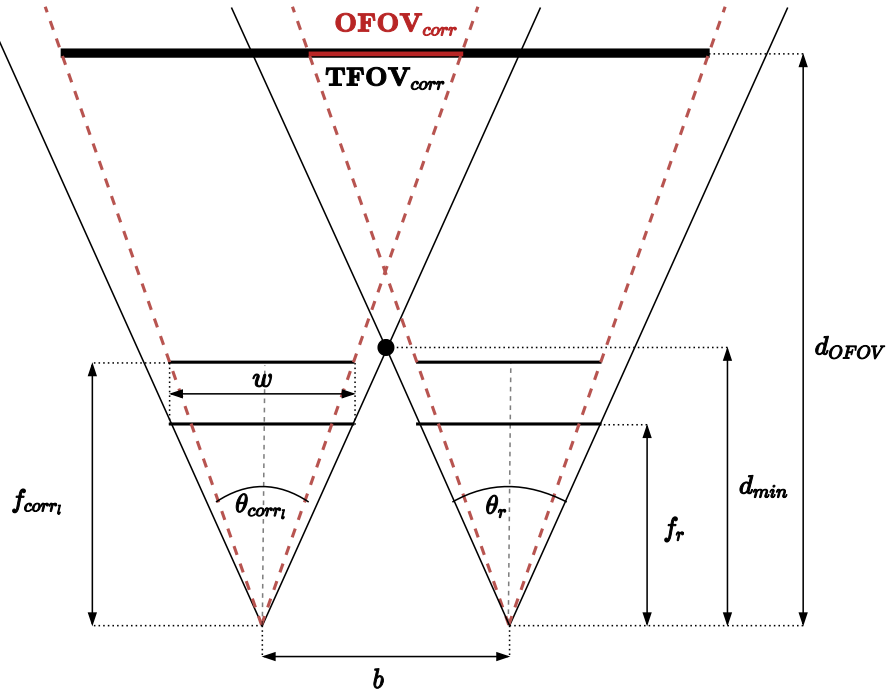



**Figure 3.16:** The refraction corrected overlapping field of view ($OFOV_{corr}$) of a stereo camera with baseline $b$.

**Algorithm 4:** Refraction corrected overlapping field of view calculations

**Result:** The percentage $OFOV_{corr}$ of $TFOV_{corr}$ for given $b$ and $d_{OFOV}$, and $d_{min}$ for given $b$.

1. Calculate $\theta_l$ and $\theta_r$ of each camera using $f_{air_l}$ and $f_{air_r}$ according to Equation (2.9);
2. Calculate $\theta_{corr_l}$ and $\theta_{corr_r}$ by correcting for refraction in water using $n_{air}$ and $n_{water}$ according to Snell's Law in Equation (2.24). The red stippled lines in Figure 3.16 sketches the corrected FOVs;
3. Calculate the minimum distance $d_{min}$ for the given $b$, $\theta_{corr_l}$ and $\theta_{corr_r}$ using trigonometry;
4. Calculate the $OFOV_{corr}$ for a given distance $l_{OFOV}$ using $d_{min}$, $\theta_{corr_l}$ and $\theta_{corr_r}$;
5. Calculate the refraction corrected left and right camera field of view $FOV_{corr_l}$ and $FOV_{corr_r}$ using $\theta_{corr_l}$ and $\theta_{corr_r}$;
6. Calculate the percentage of $OFOV_{corr}$ that covers the total corrected field of view $TFOV_{corr}$ by utilizing $TFOV_{corr} = FOV_{corr_l} + FOV_{corr_r} - OFOV_{corr}$;

Using the camera parameters of Table 3.2 the percentage $OFOV_{corr}$ of $TOFOV_{corr}$ and $d_{min}$ were calculated the for three baseline widths $b_1 = 0.1\,\text{m}$, $b_2 = 0.2\,\text{m}$ and $b_3 = 0.3\,\text{m}$ at the three observation distances $d_1 = 1\,\text{m}$, $d_2 = 3\,\text{m}$ and $d_3 = 5\,\text{m}$. The observation distances were selected based upon the expected visibility and operation distance of the WC-ROV Minerva. The results are presented in Tables 3.3 to 3.6.

**Table 3.2:** In air camera calibration parameters of the stereo camera rig.

| | **Full Resolution** | **Binned Mode** | |
|---|---|---|---|
| $f_{air_l}$ | 1109.1 | 613.3 | [px] |
| $f_{air_r}$ | 1112.3 | 613.4 | [px] |
| $w$ | 1360 | 680 | [px] |

**Table 3.3:** Percentage $OFOV_{corr}$ of $TOFOV_{corr}$ at given $b$ [m] and distance from camera $d_{OFOV}$ [m] at full resolution.

| | $d_1 = 1$ | $d_2 = 3$ | $d_3 = 5$ |
|---|---|---|---|
| $b_1 = 0.1$ | 79.9% | 92.5% | 94.5% |
| $b_2 = 0.2$ | 62.0% | 85.6% | 91.0% |
| $b_3 = 0.3$ | 48.0% | 79.0% | 86.9% |

**Table 3.4:** Percentage $OFOV_{corr}$ of $TOFOV_{corr}$ at given $b$ [m] and distance from camera $d_{OFOV}$ [m] at binned mode.

| | $d_1 = 1$ | $d_2 = 3$ | $d_3 = 5$ |
|---|---|---|---|
| $b_1 = 0.1$ | 77.3% | 91.8% | 95.0% |
| $b_2 = 0.2$ | 59.3% | 84.4% | 90.3% |
| $b_3 = 0.3$ | 44.5% | 77.4% | 85.8% |

**Table 3.5:** Minimum observable distance $d_{min}$ for given baseline $b$ at full resolution.

| | $d_{min}$ | |
|---|---|---|
| $b_1 = 0.1$ | 0.117 | [m] |
| $b_2 = 0.2$ | 0.234 | [m] |
| $b_3 = 0.3$ | 0.351 | [m] |

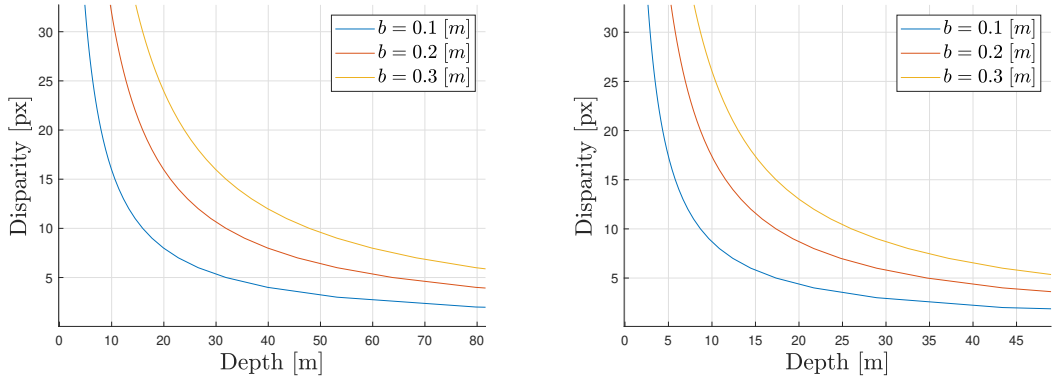
**Table 3.6:** Minimum observable distance $d_{min}$ for given baseline $b$ at binned mode.

| | $d_{min}$ | |
|---|---|---|
| $b_1 = 0.1$ | 0.128 | [m] |
| $b_2 = 0.2$ | 0.255 | [m] |
| $b_3 = 0.3$ | 0.380 | [m] |

### 3.9.2 Range of Depth

The range of depth was examined by plotting the disparity $d$ against the observation distance, or depth $z$, using the focal length $f$ and a given baseline $b$ according to Equation (2.12). The produced plots shows the expected disparity values at given depths. Ideally large differences in disparity values for the different distances is wanted such that the stereo imaging algorithm more easily can distinguish the depths.

The disparity plots of the stereo camera in full resolution and binned mode are presented in Figure 3.17. The plots are calculated using refraction corrected focal lengths obtained from $f_{air_l}$ and $f_{air_r}$ of Table 3.2 and the baselines $b_1 = 0.1\,\text{m}$, $b_2 = 0.2\,\text{m}$ and $b_3 = 0.3\,\text{m}$.




**(a)** Full resolution mode

**(b)** Binned mode

**Figure 3.17:** The disparity versus depth.

### 3.9.3 Selected Baseline

According to Tables 3.3 and 3.4 the percentage $OFOV_{corr}$ for $b_3 = 0.3\,\text{m}$ at $d_1 = 1\,\text{m}$ are for both binned mode and full resolution too low to be accepted. The depth versus disparity plot of binned mode in Figure 3.17a shows that for $b_1 = 0.3\,\text{m}$ produces small changes in disparity at depths close to 5 m compared to $b_1 = 0.3\,\text{m}$ and $b_1 = 0.3\,\text{m}$. From these two arguments, and considering the calculations being mere estimates, it was decided to set the stereo camera baseline to $b = 0.2\,\text{m}$.

## 3.10 Underwater Camera Calibration

Underwater camera calibration was performed in order to obtain the intrinsic parameters $\mathbf{K}_l$ and $\mathbf{K}_r$, the distortion coefficients $\mathbf{d}_l$ and $\mathbf{d}_r$, and the relative orientation of the cameras in the stereo camera rig $\mathbf{T}_{lr}^{l}$. The calibration was conducted underwater due to refraction in water affecting the calibration parameters.

The calibration was performed in both the full resolution and binned mode, at four different distances in between the stereo camera rig and the calibration object. The selected distances were 1, 3, 4 and 5 m reflecting the expected visibility of a subsea operating WC-ROV. The determined baseline of $b = 0.2\,\text{m}$ of Section 3.9 was used in all calibrations. The calibration data was acquired in the basin of MC-lab, while the calibration parameters were estimated using MATLAB's camera calibration software. The results of the underwater camera calibration are eight sets of intrinsic calibration parameters, four for

each camera of the stereo camera rig, and four corresponding sets of extrinsic parameters giving the relative orientation of the cameras.

### 3.10.1 Calibration Data Acquisition

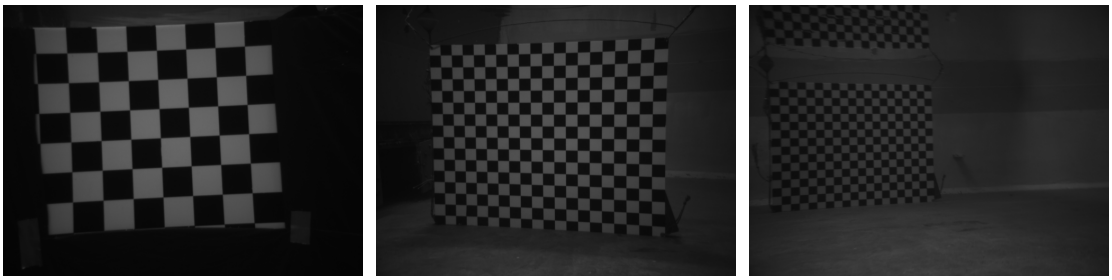
The calibration object used in the calibration was an 1.5 m × 1.2 m checkerboard with 20 × 15 complete 80 mm squares provided by AUR-Lab. The checkerboard was kept fixed while the stereo camera rig was oriented and translated such that the calibration samples together covered the total FOV of both the left and right camera. The calibration data was obtained by recording the synchronized stereo image stream of the stereo camera rig in rosbags at 2Hz. The exposure time was in-field adjusted to obtain non-blurred images with adequate brightness.

**(a)** The checkerboard fixed at 3 m **(b)** The recording of a calibration data set

**Figure 3.18:** The underwater camera calibration in MC-lab.

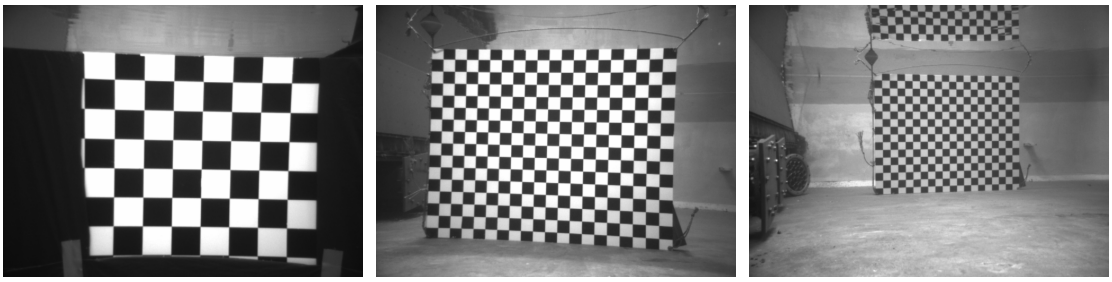
**Calibration Image Sets:** The calibration image sets were established by extracting stereo image pairs from each calibration data rosbag. The original size of the extracted stereo image sets ranged from 200 to 600 stereo image pairs, and was reduced in size by selecting stereo images of high quality with unique orientation and translation. The resulting calibration image set sizes ranged from 20 to 50 stereo images depending on the quality of the recorded calibration rosbag. Sample images from the calibration image sets are shown in Figures 3.19 and 3.20.

Note that due to the limited FOV at the calibration distance of 1 m, parts of the checkerboard was covered proving a reduced checkerboard of 8 × 7 squares.



**(a)** 1 m **(b)** 3 m **(c)** 5 m

**Figure 3.19:** Sample images from the full resolution mode left camera calibration data sets. Exposure time is set to 80 ms.



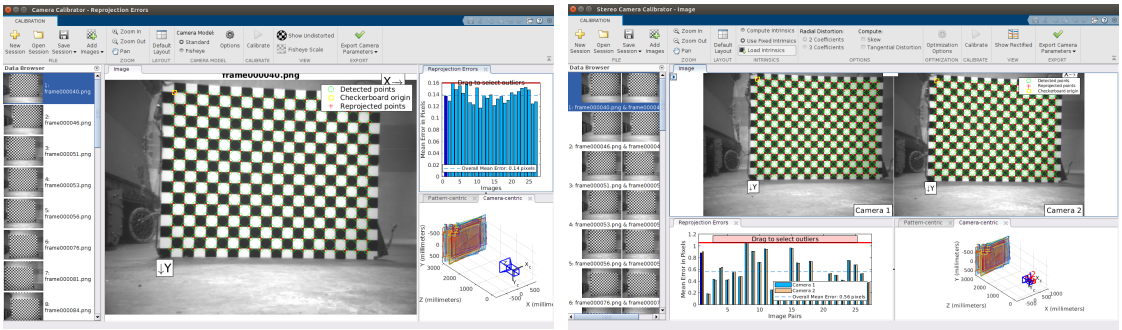
**(a)** 1 m **(b)** 3 m **(c)** 5 m

**Figure 3.20:** Sample images from the binned mode left camera calibration data sets. Exposure time is set to 50 ms.

### 3.10.2 Camera Calibration with MATLAB

The single camera and stereo camera calibration of the obtained calibration image sets were performed using MATLAB's *Camera Calibrator App* and *Stereo Camera Calibrator App*. The applications are based upon Zheng's method described in Section 2.1.4 and provides tools for efficiently analyzing the calibration image data sets.

The calibrator apps outputs after each estimation the mean pixel reprojection error, and the pixel reprojection error corresponding to each calibration image or stereo calibration image pair. It contributes to determining the necessary distortion parameters, and allows for efficient supervision of the calibration images data set by removing sample outliers.



**(a)** The Camera Calibrator App of MATLAB **(b)** The Stereo Camera Calibrator App of MATLAB

**Figure 3.21**

The cameras were first calibrated separately on each corresponding left and right calibration image set, in order to obtain the individual intrinsic parameters and distortion coefficients for the given image set. Each of the obtained intrinsics and distortion coefficients, were then used to do stereo camera calibration in order to obtain the corresponding relative orientation of the given calibration image set.

**Distortion Coefficients** It was selected to use the first and second radial distortion coefficients, $k_1$ and $k_2$ and the tangential distortion coefficients $p_1$ and $p_2$. The choice was made due to estimations with $p_1$ and $p_2$ obtained lower a mean pixel reprojection error than without. It was tested to perform estimates with additional radial distortion coefficients, but the mean reprojection error remained the unaffected. Every individual left and right camera calibration were hence performed with two the radial distortion coefficients, $k_1$ and $k_2$, and the tangential distortion coefficients, $p_1$ and $p_2$.

### 3.10.3 Underwater Calibration Results

The resulting intrinsic parameters $\mathbf{K}_l$ and $\mathbf{K}_r$, distortion coefficients $\mathbf{d}_l$ and $\mathbf{d}_r$, and relative camera orientation $\mathbf{T}_{lr}^{l}$ for the calibration distances 1, 2, 3, 4 and 5 m are presented in the following tables. Tables 3.7 and 3.8 gives the $\mathbf{K}_l$ and $\mathbf{K}_r$ of the left and right camera, Tables 3.9 and 3.10 gives the $\mathbf{d}_l$ and $\mathbf{d}_r$ of the left and right camera, while Table 3.11 gives the $\mathbf{T}_{lr}^{l}$ of the cameras.

**Calibration Parameters Set for Further Use:** The preferred camera calibration parameters for further use was determined by considering the resulting relative translation vector of the cameras as the quality metric. The camera calibration data sets which had the relative translation vector that reflected the true measured translation the most, were the parameter sets that was recommended for further use. Hence, according to Table 3.11 were the camera calibration set obtained at 1 m recommended for the full resolution mode, and the camera calibration set obtained at 3 m recommended for the binned mode.

**Table 3.7:** The intrinsic parameter at **full resolution mode**. The parameters are given in pixels.

| | **Left** | | | | **Rigth** | | | |
|---|---|---|---|---|---|---|---|---|
| | $f_u$ | $f_v$ | $c_u$ | $c_v$ | $f_u$ | $f_v$ | $c_u$ | $c_v$ |
| **1 m** | 1703.3 | 1700.7 | 704.4 | 530.2 | 1697.0 | 1696.2 | 669.5 | 534.2 |
| **3 m** | 1693.8 | 1689.9 | 705.0 | 532.4 | 1695.7 | 1693.1 | 673.0 | 528.1 |
| **4 m** | 1684.0 | 1681.1 | 713.7 | 520.0 | 1689.7 | 1688.1 | 682.3 | 517.9 |
| **5 m** | 1665.2 | 1662.1 | 702.9 | 525.1 | 1672.4 | 1670.2 | 680.0 | 521.5 |

**Table 3.8:** The intrinsic parameter at **binned mode**. The parameters are given in pixels.

| | **Left** | | | | **Rigth** | | | |
|---|---|---|---|---|---|---|---|---|
| | $f_u$ | $f_v$ | $c_u$ | $c_v$ | $f_u$ | $f_v$ | $c_u$ | $c_v$ |
| **1 m** | 847.1 | 845.8 | 355.6 | 264.7 | 845.7 | 845.0 | 337.2 | 266.4 |
| **3 m** | 841.5 | 839.7 | 355.2 | 264.9 | 841.0 | 839.8 | 340.2 | 263.9 |
| **4 m** | 840.3 | 838.7 | 355.7 | 261.0 | 846.2 | 845.1 | 341.2 | 259.5 |
| **5 m** | 804.8 | 804.1 | 353.3 | 254.6 | 812.9 | 812.5 | 340.3 | 255.9 |

**Table 3.9:** The distortion coefficients at **full resolution mode**.

| | **Left** | | | | **Right** | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_1$ | $k_2$ | $p_1$ | $p_2$ | $k_1$ | $k_2$ | $p_1$ | $p_2$ |
| **1 m** | 0.1347 | 0.6513 | 0.0024 | 0.0058 | 0.1162 | 0.7675 | 0.0042 | -0.0047 |
| **3 m** | 0.1130 | 0.6656 | 0.0015 | 0.0054 | 0.1180 | 0.6307 | 0.0020 | -0.0046 |
| **4 m** | 0.1131 | 0.5601 | -0.0028 | 0.0070 | 0.1111 | 0.5895 | -0.0022 | -0.0023 |
| **5 m** | 0.1004 | 0.5728 | 0.0023 | 0.0079 | 0.1114 | 0.6370 | -0.0004 | -0.0046 |

**Table 3.10:** The distortion coefficients at **binned mode**

| | **Left** | | | | **Right** | | | |
|---|---|---|---|---|---|---|---|---|
| | $k_1$ | $k_2$ | $p_1$ | $p_2$ | $k_1$ | $k_2$ | $p_1$ | $p_2$ |
| **1 m** | 0.1295 | 0.6593 | 0.0013 | 0.0084 | 0.1252 | 0.6190 | 0.0031 | -0.0029 |
| **3 m** | 0.1080 | 0.6842 | 0.0012 | 0.0075 | 0.1173 | 0.5990 | 0.0021 | -0.0019 |
| **4 m** | 0.1059 | 0.6149 | -0.0020 | 0.0071 | 0.1088 | 0.6296 | -0.0013 | -0.0015 |
| **5 m** | 0.1001 | 0.4388 | -0.0040 | 0.0057 | 0.1018 | 0.4901 | -0.0028 | -0.0021 |

**Table 3.11:** The relative translation of the stereo cameras at **full resolution mode** and **binned mode**. The translation is given in millimeters.

| | **Full Resolution** | | | **Binned Mode** | | |
|---|---|---|---|---|---|---|
| | $t_x$ | $t_y$ | $t_z$ | $t_x$ | $t_y$ | $t_z$ |
| **1 m** | -198.1 | -0.6 | 0.1 | -193.5 | -0.6 | 1.4 |
| **3 m** | -193.1 | -7.0 | 7.0 | -195.7 | -1.9 | 1.1 |
| **4 m** | -191.1 | 5.3 | 12.7 | -197.1 | -8.5 | 27.6 |
| **5 m** | -204.9 | -2.1 | 19.5 | -198.8 | -6.3 | 50.5 |

# Chapter 4

# Experiments

In this chapter, the experiments testing the real-time WC-ROV VSLAM system are presented. The conducted experiments consisted of an laboratory experiment of the system in an underwater obstacle course, and a Hardware In the Loop-testing (HIL) with the system operating together with the Autonomy Framework of Minerva. Section 4.1 describes the underwater obstacle course experiment where the system's capability of estimating its position, orientation, surrounding environment and detect its closest obstacle was tested. The lab tests was conducted in both full resolution and binned mode, both under ideal lighting conditions and in subsea simulated conditions. Section 4.2 describes the HIL-testing in which the communication with the Autonomy Framework was verified using an altered version of the LabView Communication node in Section 3.8 generating synthetically detected obstacles.

## 4.1 Obstacle Course Experiment

The obstacle course experiment was conducted in order to test the capabilities of the WC-ROV VSLAM system estimating its position and orientation, surroundings, and detecting its immediate closest obstacle. The testing was performed using both the full resolution and binned mode. Both modes were tested in order to address the increased computational complexity occurring at higher resolutions, and the increased light sensitivity using the binned mode. Each of the modes were tested in ideal illumination condition with full ambient lighting, and in subsea simulated conditions with no ambient lighting and artificial illumination. The test data from the obstacle course was collected in rosbags on the laptop computer, while the real-time runs of the WC-ROV VSLAM system was conducted using a desktop computer by playing back the rosbags. The computers are described in Section 3.3. Measurements obtained from the Qualisys motion tracking system acts as the ground truth for the position and orientation estimates, while the measured dimensions of the obstacle course acts as the ground truth for the estimated map. The setup of the experiment and the procedure for data collection is explained in the following two subsections.

### 4.1.1 Underwater Obstacle Course Experiment Setup

The underwater obstacle course experiment was conducted in Marine Cybernetics lab (MC-lab). The ground truth for the WC-ROV VSLAM motion and map estimates were

respectively the Qualisys motion tracking system and the dimensions of the constructed underwater obstacle course. The subsea simulated artificial illumination was generated by using an underwater flashlight. The system used the selected camera calibration parameters of Section 3.10, and some specifically determined exposure times and framerates for both the resolution modes. The parts of the experiment setup is thoroughly explained in the following paragraphs.

**Marine Cybernetics lab:** The MC-lab at Marinteknisk senter in Trondheim consists of a $15 \times 20[m]$ basin with a depth ranging from $0.5[m]$ to $1.5[m]$ installed with a wave generator, towing carriage and Qualisys for motion tracking.

**Qualisys Motion Tracking System** The Qualisys motion tracking system is an optical tracking system capable of tracking 6 degrees of freedom. Qualisys tracks the motion of a desired object by optically measuring the position of optical targets installed on the desired object. The Qualisys of MC-lab has three cameras installed above water and four underwater cameras installed in the basin. The rod mounted camera rig was installed with three Qualisys motion tracking target spheres for motion tracking. The spheres are located 1.4 m in the $y$-direction of the stereo camera frame, see Figure 4.3b. The bone length tolerance of the 6DOF tracking was set to 80 mm. The Qualisys measurements were collected from the three cameras above the water due to these providing a larger calibration volume.

**Obstacle Course:** The obstacle course was constructed by placing objects of known dimensions to a selected area of $11.36\,\text{m} \times 6.45\,\text{m}$ in the basin. The area was selected based upon the calibrated volume of Qualisys. The relative positions of the placed obstacles were measured and presented in Figure 4.1. The obstacles were a soda case, a aluminum bar, a hollow aluminum cube and a stepladder. Their dimensions are presented Table 4.2.

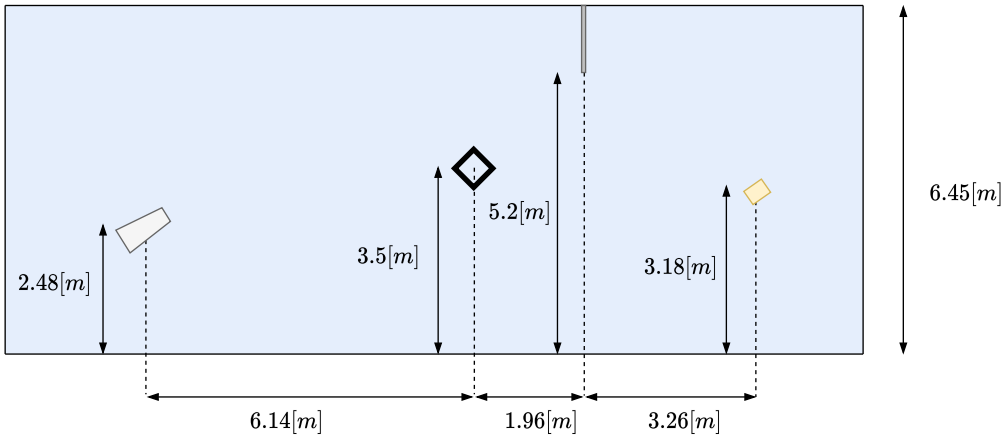


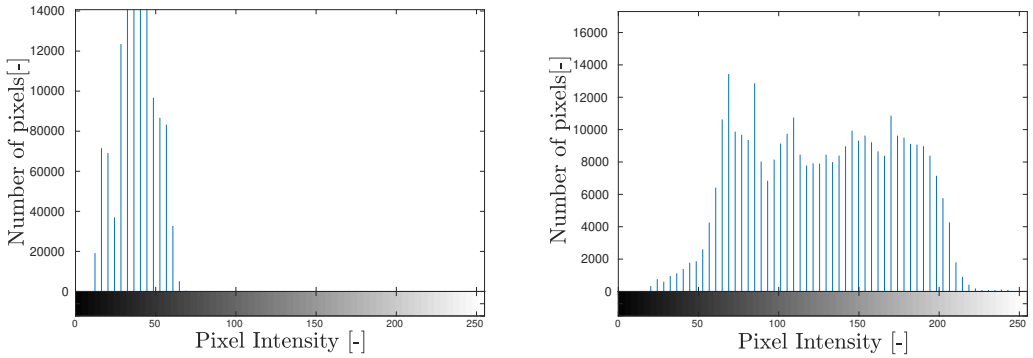

**Figure 4.1:** Obstacle course. The positions is given from the volumetric center of the obstacles.

**Table 4.1:** The dimensions of the obstacles of the obstacle course, where $l$ is the length, $w$ is the width, $h$ is the height and $d$ is the diameter.

| | $l$ | $w$ | $h$ | $d$ | |
|---|---|---|---|---|---|
| **Soda case** | 0.4 | 0.3 | 0.3 | - | [m] |
| **Cube** | 0.54 | 0.54 | 0.54 | - | [m] |
| **Stepladder** | 0.93 | 0.48 | 1.72 | - | [m] |
| **Bar** | 2 | - | - | 0.1 | [m] |

**Artificial Illumination:** The subsea conditions was simulated by keeping the ceiling lights of MC-lab turned off and mounting a flashlight to the stereo camera rig for artificial illumination. The flashlight was a *Magicshine MJ-810E* diving torch producing 740 lm at its maximum light intensity setting. The light intensity setting is set to max during the experiment.

**Exposure Time and Framerate:** The exposure time and framerate of the full resolution mode and the binned mode were set individually. According to the bandwidth calculations in Section 3.4.2, the upper restriction of the frame rate was bounded by the camera shutter speed and could thus be set freely. Regarding the exposure time, the histograms of the previously collected calibration data sets of Figures 3.19 and 3.20 were analyzed. The histograms showed that the pixel intensities in full resolution mode was shifted to the lower band implying that a reduced exposure time would reduce the information provided in the images. The histogram of the binned mode showed that the pixel intensities were evenly distributed giving a lower requirement of the exposure time. To maintain an acceptable framrate, the exposure time for the full resolution mode was marginally reduced to 71 ms. The exposure time of the binned mode was set to 40 ms to increase the framerate capacity. The framerate settings were matched to the selected exposure time at 14 Hz for the full resolution mode and 24 Hz for the binned mode.




**(a)** Full resolution sample image in FIG REFF. Exposure time is 80 ms.

**(b)** Binned mode sample image in FIG REFF. Exposure time is 50 ms.

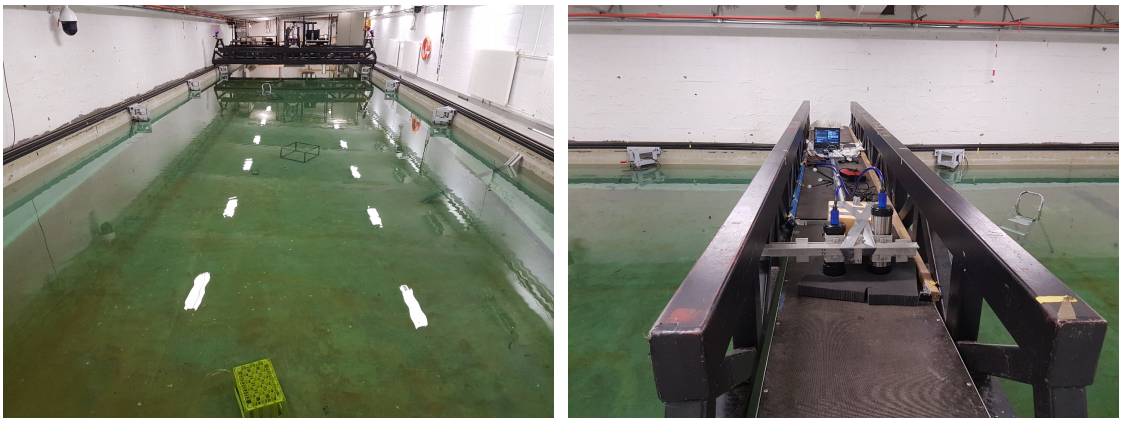
**Figure 4.2:** Image histogram of the full resolution and binned mode camera calibration sample images.

### 4.1.2 Procedure of Data Collection

The underwater obstacle course test data was collected by moving the stereo camera rig through the underwater obstacle course and recording the produced stereo camera outputs in rosbags.

The movement of the stereo camera rig was conducted by maneuvering the stereo camera rig from the small towing carriage of MC-lab, see Figure 4.3b. The rig was moved in a circular pattern starting from the middle left of the obstacle course in Figure 4.1, moving to the top right and then returning to the starting point. The circular pattern was selected in order to induce loop closings of the ORB-SLAM2 algorithm. Performing the motion of the stereo camera rig requires two persons. The first person operated the rig on the towing cart, orienting and translating in the widthwise direction, while the second person pushed the towing cart translating the rig in the lengthwise direction.

The underwater obstacle course test data was recorded on the laptop computer using rosbags. The rosbag recordings contained the raw image streams of the left and right camera of the stereo camera rig. The recordings were initiated simultaneously with the Qualisys motion tracking system by having a Qualisys operator in the control room communicating with the WC-ROV VSLAM system operator using wireless communication. The real-time results of the underwater obstacle course experiment was produced by playing back the rosbag data sets on the WC-ROV VSLAM system using the desktop computer. See section Section 3.2 regarding ROS and rosbags.



**(a)** The underwater obstacle course with the four placed obstacles: stepladder, cube, rod and soda case

**(b)** The stereo camera rig with installed motion tracking targets prior to obstacle course data collection

**Figure 4.3:** The underwater obstacle course setup in MC-Lab.

### 4.1.3 Summary of Underwater Obstacle Course Experiment

To summarize the underwater obstacle course experiment description, data sets of four different test cases were collected. The test cases are presented in Table 4.2.

**Table 4.2:** The different cases for obstacle course data collection

| | Resolution mode | FPS [Hz] | Exposure time [ms] | Artificial Ilumiation |
|---|---|---|---|---|
| **Case 1** | Full Resolution | 14 | 71 | Yes |
| **Case 2** | Full Resolution | 14 | 71 | No |
| **Case 3** | Binned | 24 | 40 | Yes |
| **Case 4** | Binned | 24 | 40 | No |

## 4.2 HIL-testing

The HIL-testing was conducted in order to verify the integration of the WC-ROV VSLAM system in to the ROV Autonomy Framework. The experiment was performed by running the ROV Autonomy Framework together with the control system of the ROV Minerva connected to a simulator replacing the sensor signals of Minerva. Due to the difficulty of arranging an image data set to match the simulator environment, it was decided to test the integration of the systems by generating synthetically detected obstacles instead of obstacles truly detected by the WC-ROV VSLAM system inputted by an image data set. An altered version of the ROS LabView communication node of Section 3.8 was

established sending TCP messages of detected obstacles from a predefined set of obstacles instead of obstacles received from the point cloud processing node of Section 3.7. The main objective of the HIL-testing was thus to verify the TCP communication protocol of the WC-ROV system and the ROV autonomy framework.

The setup of the HIL-testing consisted of two computers connected by TCP connection. The first computer ran the ROV Autonomy Framework and the Minerva control system and simulator, while the second computer was the described desktop computer of this thesis running the alternated Labview Communication Node. The messages conveying the closest detected obstacle and ROV position between the two computers was set to be transmitted at 2 Hz. The communication flow is illustrated in Figure 4.4, and details about the Minerva simulator and Autonomy Framework is described in the manuscript of the joint work by the master students working on ROV autonomy in Appendix A.

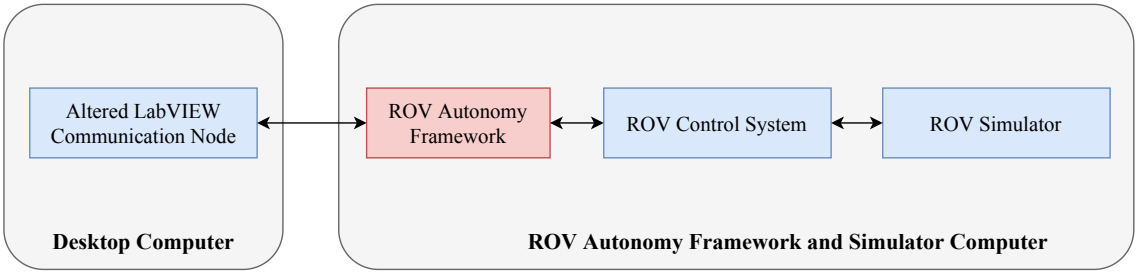



**Figure 4.4:** The communication between the two computers of the virtual experiment.

# Chapter 5

# Results

This chapter presents the results of the conducted experiments described in Chapter 4. Section 5.1 presents the results of the four cases of the underwater obstacle course experiment of Table 4.2, the test cases are: full resolution and binned mode in both ideal and subsea simulated lighting condition. Section 5.2 presents the results of the HIL-testing.

## 5.1 Underwater Obstacle Course Experiment Results

The results of the underwater obstacle course experiments are presented using comparative plots of the WC-ROV VSLAM system estimated and the measured Qualisys position and orientation. The comparative plots are accompanied by the calculated error plots. The notation of the orientations are $\phi$, $\theta$, and $\psi$ for the roll, pitch and yaw. The results of the map estimation is presented by plotting the estimated map points against the measured dimensions of the underwater obstacle course in a three dimensional plot. The closest obstacle detection results are presented using a three dimensional plot of the estimated map points, the obstacle course dimensions and vector denoting the detection of a newly detected closest obstacle. A new obstacle is defined as an obstacle that differs at a distance of 0.15 m from the previously detected closest obstacle.

For each test case, a sample image from the data set in both unprocessed and CLAHE applied version is given, and the used WC-ROV VSLAM static parameters are presented in tables. In the tables, the $clip_l$, $tile_w$ and $tile_h$ are the clip length, width and height of CLAHE. The $n_f$ is the number of extracted ORB features extracted from a single image in ORB-SLAM2, $\mathbf{t}_r$ is the distance threshold of the RANSAC, and $\mathbf{t}_c$, $c_{min}$ and $c_{max}$ are the respectively point distance threshold, minimum and maximum cluster size of the Euclidean clustering. In the full resolution mode, the underwater obstacle course data sets were run with halved the frequency giving the data sets doubled duration compared to collected data set. More on this in the discussion of Section 6.1.1. In the parameter table, it is presented as the framerate factor $f_{fps}$ which is multiplied by the original frame rate.

The results were obtained using the recommended camera calibration parameters determined in the underwater camera calibration of Section 3.10. The recommended camera calibration parameter sets obtained at 1 m was used in the full resolution mode, and the

recommended camera calibration parameter set obtained at 3 m was used in the binned mode.

**Data set Alignment:** Prior to the plotting of the WC-ROV VSLAM results and Qualisys measurements, the data sets needed to be temporally and spatially aligned. The temporal difference occurred due to the minor difference in the logging initiation of the obstacle course data sets and the Qualisys measurements. The spatial difference was due to the coordinate reference frames of the WC-ROV VSLAM system and Qualisys measurements having different origin and orientation. The data sets were first temporally aligned by comparing the shift in the estimated and measured attitudes, the data sets were then spatially aligned by first rotating the estimated positions of the WC-ROV VSLAM system by examining the error drift in the positions caused by misaligned reference frames, and then translating both data sets starting points to the origin. To generate the error plots, the Qualisys data set were interpolated to match the same amount of data points as the ones generated by the WC-ROV VSLAM system. The map estimate and closest detected obstacle plots were aligned by manually rotation and translating the point cloud at trajectory to fit the measured underwater obstacle course.

### 5.1.1 Full Resolution Mode Ideal Light Conditions

**Table 5.1:** The static parameters of the WC-ROV VSLAM system and data set attributes of the test case full resolution mode ideal light conditions.

| | | | |
|---|---|---|---|
| **CLAHE** | $clip_l$ | 4 | [-] |
| | $tile_w$ | 5 | $[px]$ |
| | $tile_h$ | 5 | $[px]$ |
| **ORB-SLAM2** | $n_f$ | 2400 | [-] |
| **RANSAC** | $\mathbf{t_r}$ | 0.13 | $[m]$ |
| **Clustering** | $\mathbf{t_c}$ | 0.15 | $[m]$ |
| | $c_{min}$ | 20 | [-] |
| | $c_{max}$ | 10000 | [-] |
| **Data Set** | FPS | 14 | [-] |
| | $f_{fps}$ | 1/2 | [-] |

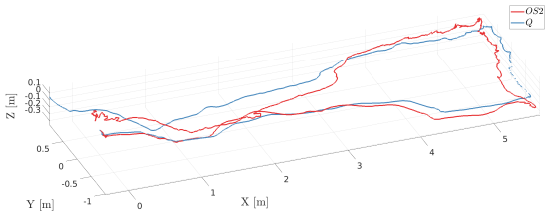
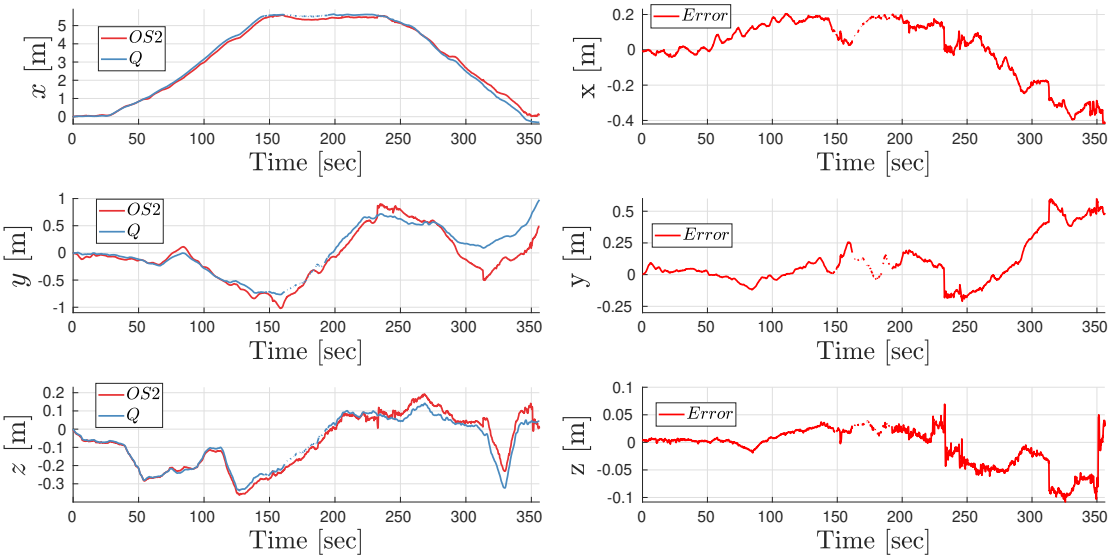



**Figure 5.1:** Trajectory of ORB-SLAM2 (orange) and Qualisys (blue) in the test case full resolution mode ideal light conditions.

**(a)** Unprocessed **(b)** With CLAHE

**Figure 5.2:** Sample image from the full resolution mode ideal light conditions data set.

#### 5.1.1.1 Position Estimate

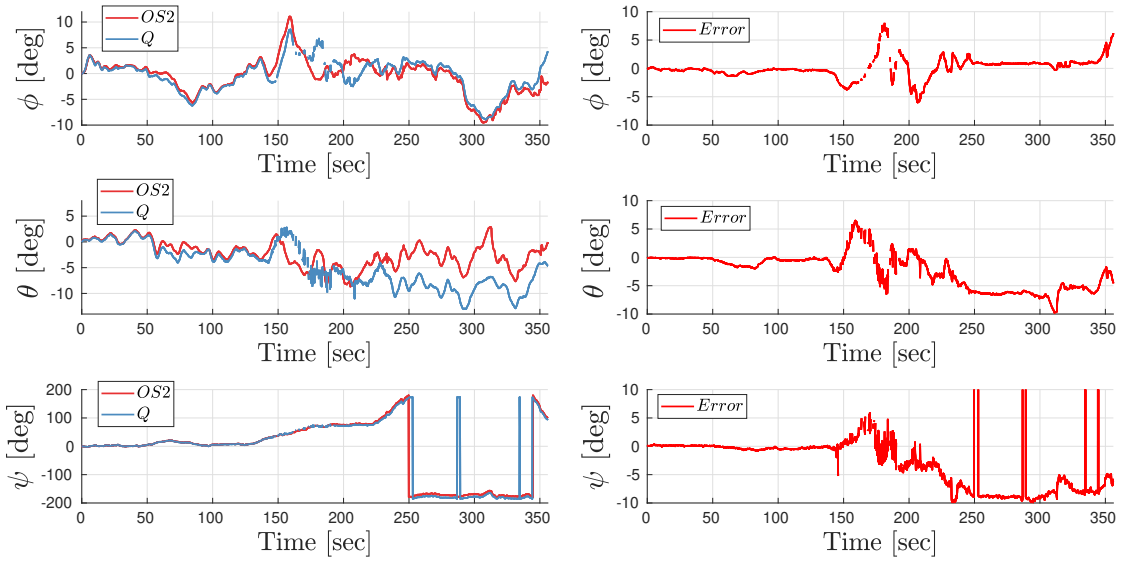



**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) position.

**(b)** The calculated error of the comparative plots of OS2 and Q position.

**Figure 5.3:** The position estimates of the test case full resolution mode in ideal light conditions.

#### 5.1.1.2 Orientation Estimate




**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) orientation.

**(b)** The calculated error of the comparative plots of OS2 and Q orientation.

**Figure 5.4:** The orientation estimates of the test case full resolution mode in ideal light conditions.

#### 5.1.1.3 Map Estimation

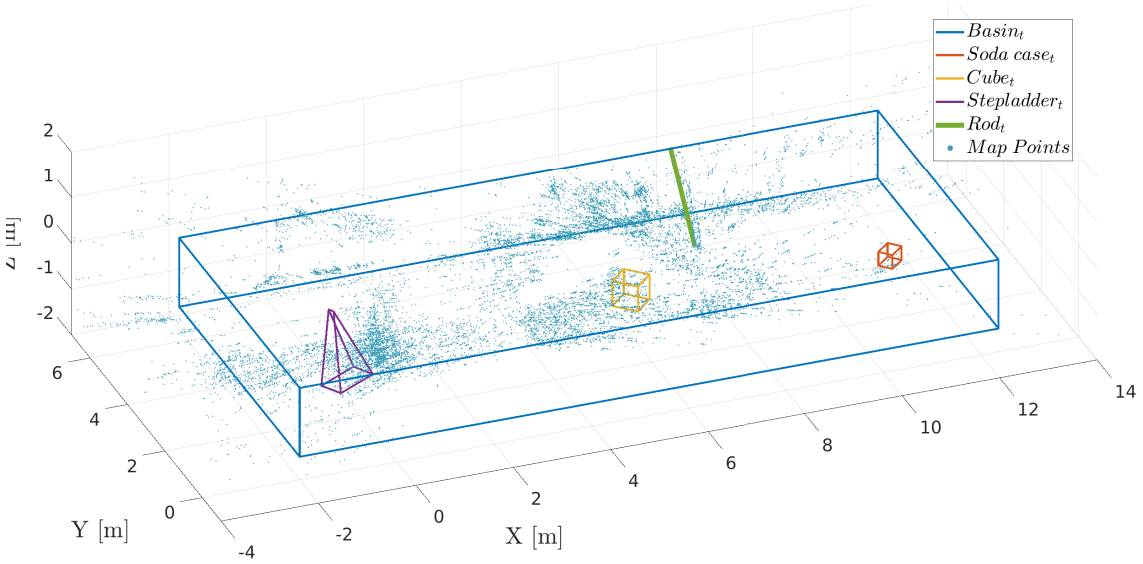



**Figure 5.5:** The estimated map points and measured underwater obstacle course of the test case full resolution mode ideal light conditions.

#### 5.1.1.4 Closest Obstacle Detection

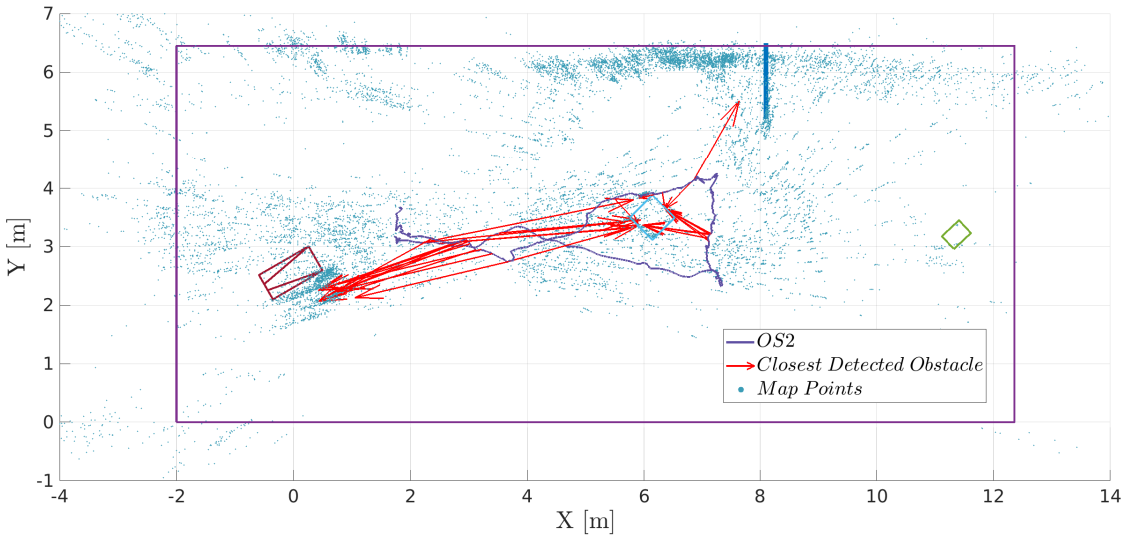
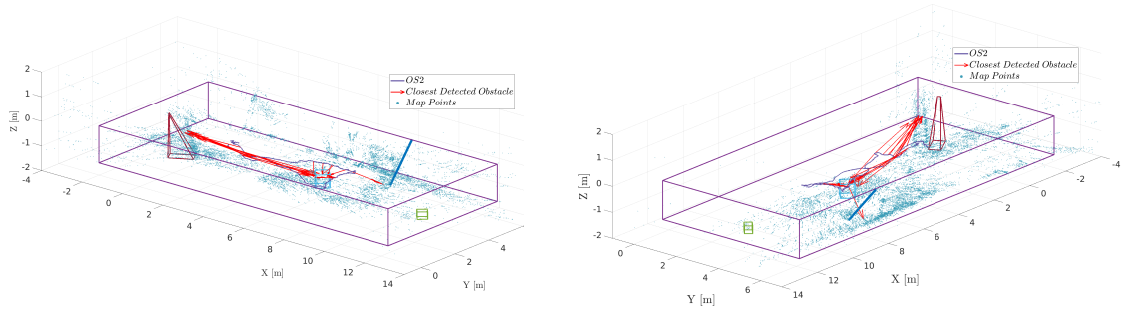



**Figure 5.6:** Top view of the closest detected obstacle of the test case full resolution mode ideal light conditions, visualized as the red vectors.




**(a)** **(b)**

**Figure 5.7:** Bird view **(a)** and **(b)** of the closest detected obstacle plot, of the test case full resolution mode ideal light conditions.

### 5.1.2 Binned Mode Ideal Light Conditions

**Table 5.2:** The static parameters of the WC-ROV VSLAM system and data set attributes of the test case binned mode ideal light conditions.

| | | | |
|---|---|---|---|
| **CLAHE** | $clip_l$ | 1.5 | [-] |
| | $tile_w$ | 5 | $[px]$ |
| | $tile_h$ | 5 | $[px]$ |
| **ORB-SLAM2** | $n_f$ | 1500 | [-] |
| **RANSAC** | $\mathbf{t_r}$ | 0.13 | $[m]$ |
| **Clustering** | $\mathbf{t_c}$ | 0.15 | $[m]$ |
| | $c_{min}$ | 20 | [-] |
| | $c_{max}$ | 10000 | [-] |
| **Data Set** | FPS | 24 | [-] |
| | $f_{fps}$ | 1 | [-] |

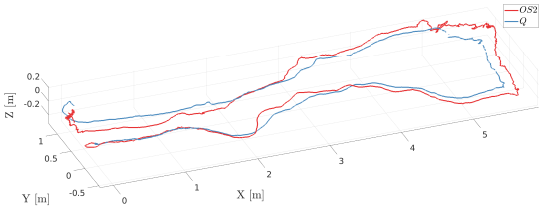
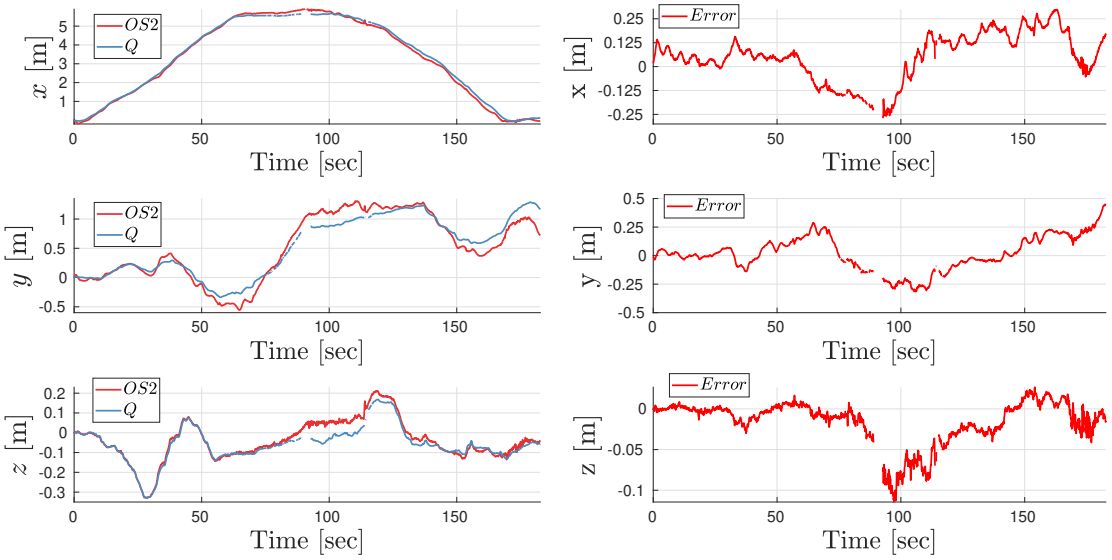



**Figure 5.8:** Trajectory of ORB-SLAM2 (orange) and Qualisys (blue) in the test case binned mode ideal light conditions.

**(a)** Unprocessed **(b)** With CLAHE

**Figure 5.9:** Sample image from the binned mode ideal light conditions data set.

#### 5.1.2.1 Position Estimate

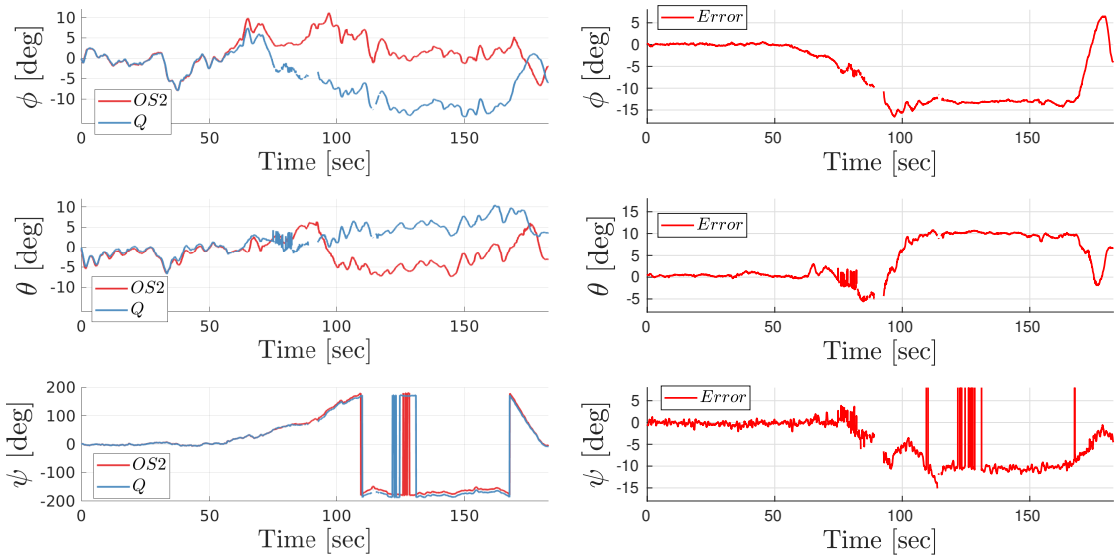



**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) position.

**(b)** The calculated error of the comparative plots of OS2 and Q position.

**Figure 5.10:** The position estimates of the test case binned mode in ideal light conditions.

#### 5.1.2.2 Orientation Estimate




**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) orientation.

**(b)** The calculated error of the comparative plots of OS2 and Q orientation.

**Figure 5.11:** The orientation estimates of the test case binned mode in ideal light conditions.

#### 5.1.2.3 Map Estimation

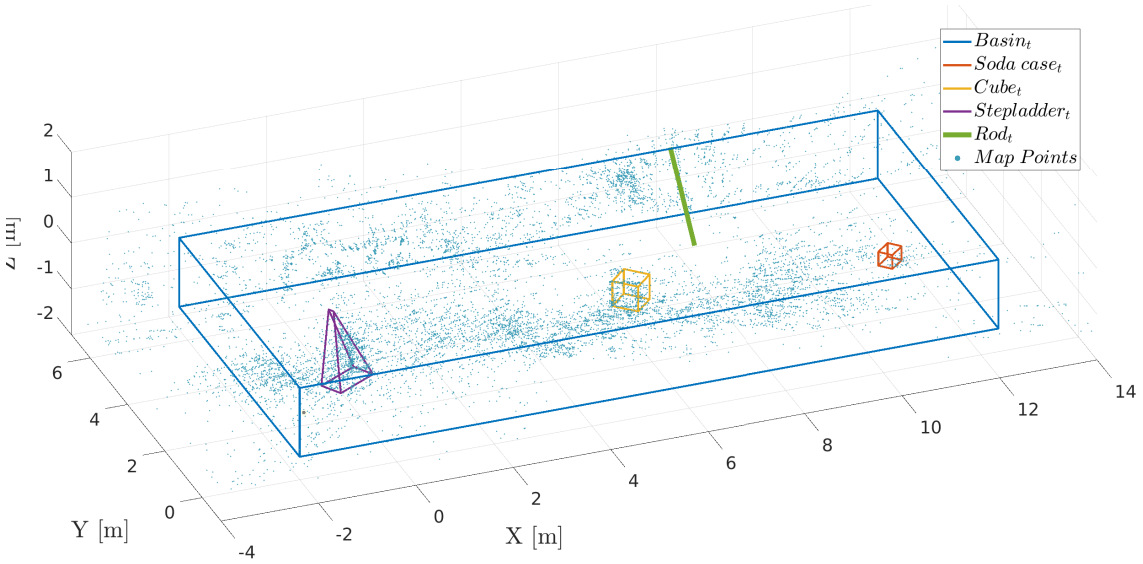



**Figure 5.12:** The estimated map points and measured underwater obstacle course of the test case binned mode ideal light conditions.

#### 5.1.2.4 Closest Obstacle Detection

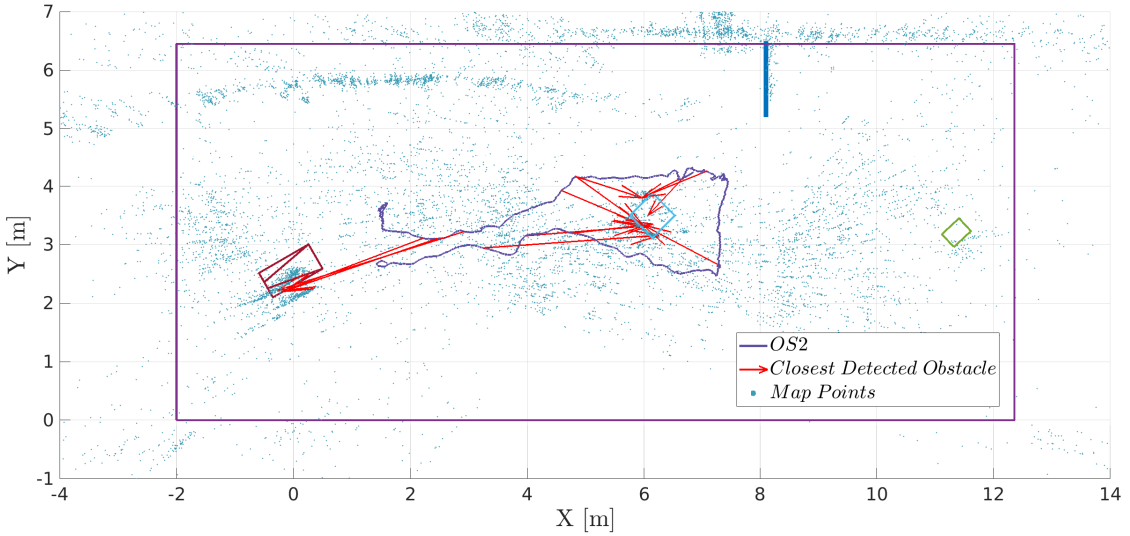
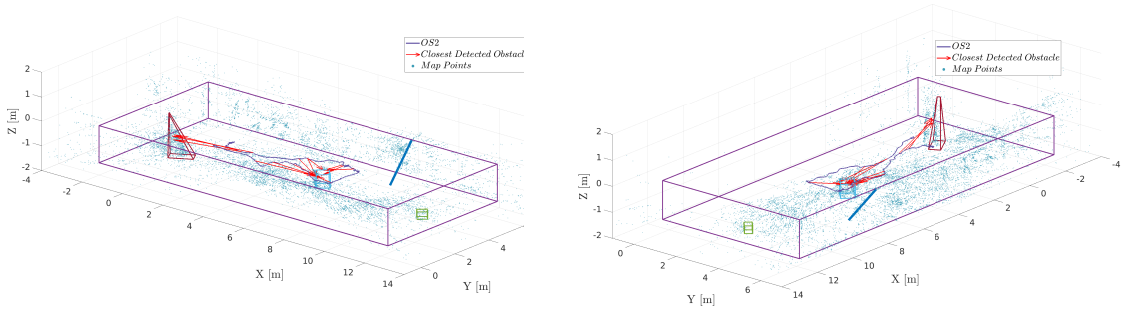



**Figure 5.13:** Top view of the closest detected obstacle of the test case binned mode ideal light conditions, visualized as the red vectors.




**(a)** **(b)**

**Figure 5.14:** Bird view **(a)** and **(b)** of the closest detected obstacle plot, of the test case binned mode ideal light conditions.

### 5.1.3 Full Resolution Mode Subsea Simulated Lighting Condition

**Table 5.3:** The static parameters of the WC-ROV VSLAM system and data set attributes of the test case full resolution subsea simulated lighting conditions.

| | | | |
|---|---|---|---|
| **CLAHE** | $clip_l$ | 16 | [-] |
| | $tile_w$ | 9 | $[px]$ |
| | $tile_h$ | 6 | $[px]$ |
| **ORB-SLAM2** | $n_f$ | 2400 | [-] |
| **RANSAC** | $\mathbf{t_r}$ | 0.13 | $[m]$ |
| **Clustering** | $\mathbf{t_c}$ | 0.15 | $[m]$ |
| | $c_{min}$ | 20 | [-] |
| | $c_{max}$ | 10000 | [-] |
| **Data Set** | FPS | 14 | [-] |
| | $f_{fps}$ | $1/2$ | [-] |

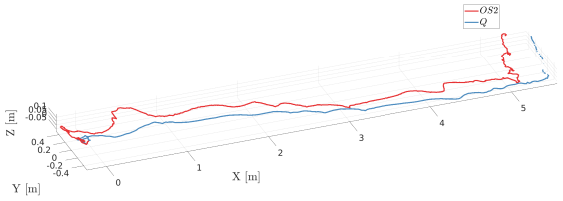
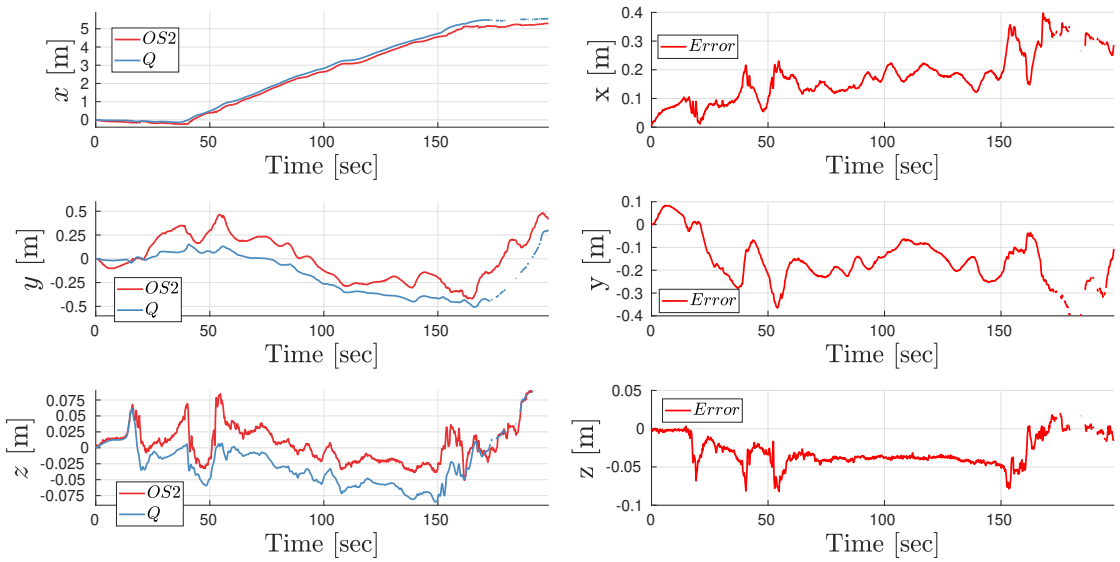



**Figure 5.15:** Trajectory of ORB-SLAM2 (orange) and Qualisys (blue) in the test case full resolution mode subsea simulated lighting conditions.

**(a)** Unprocessed **(b)** With CLAHE

**Figure 5.16:** Sample image from full resolution mode subsea simulated light conditions data set.

#### 5.1.3.1 Position Estimate

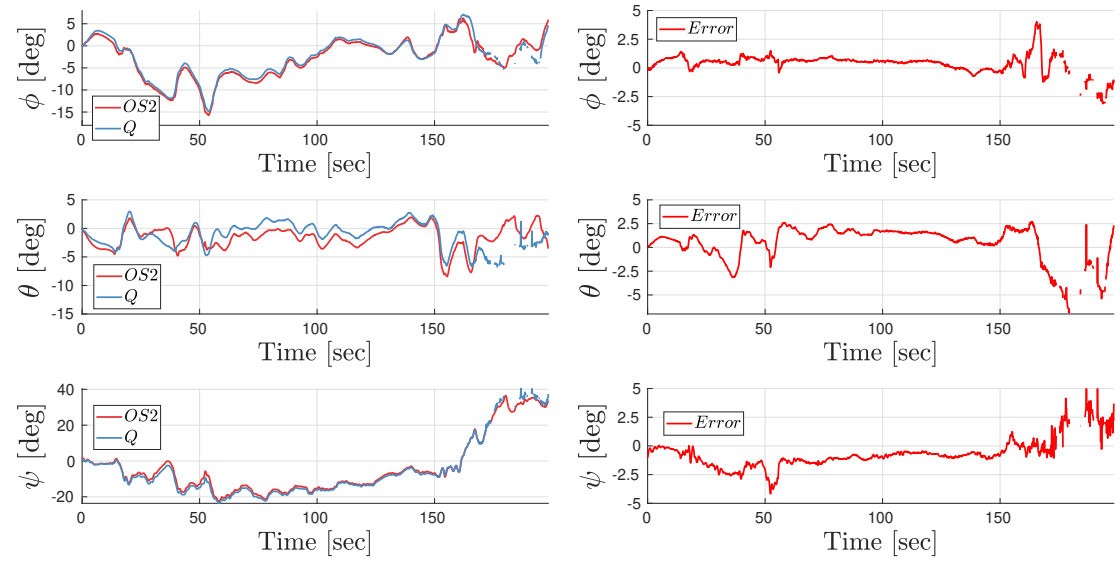



**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) position.

**(b)** The calculated error of the comparative plots of OS2 and Q position.

**Figure 5.17:** The position estimates of the test case full resolution mode subsea simulated lighting conditions.

#### 5.1.3.2 Orientation Estimate




**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) orientation.

**(b)** The calculated error of the comparative plots of OS2 and Q orientation.

**Figure 5.18:** The orientation estimates of the test case full resolution mode subsea simulated lighting conditions.

#### 5.1.3.3 Map Estimation

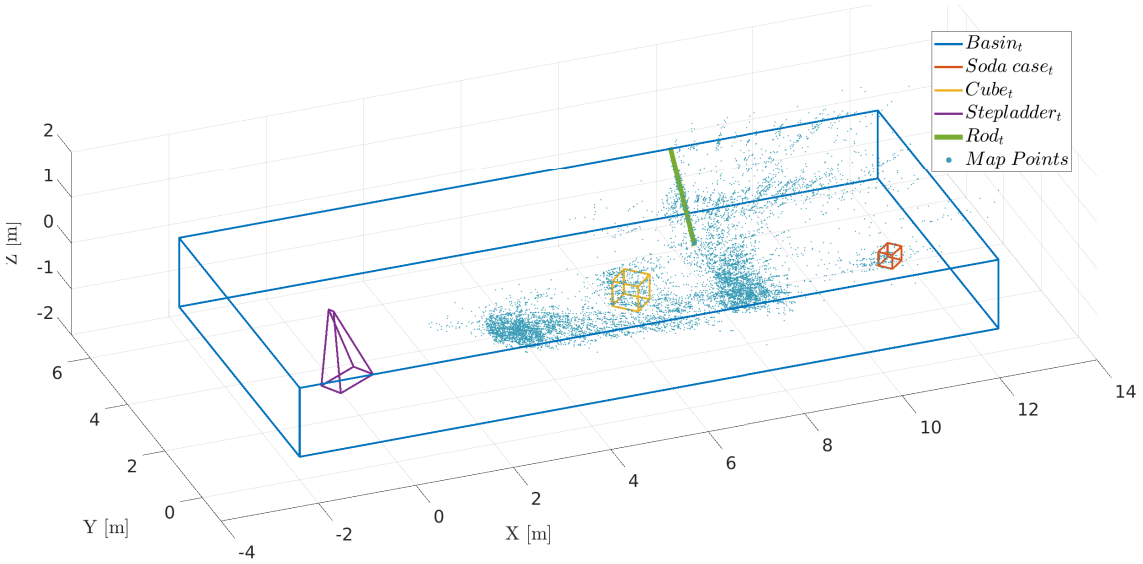



**Figure 5.19:** The estimated map points and measured underwater obstacle course of the test case full resolution mode subsea simulated lighting conditions.

#### 5.1.3.4 Closest Obstacle Detection

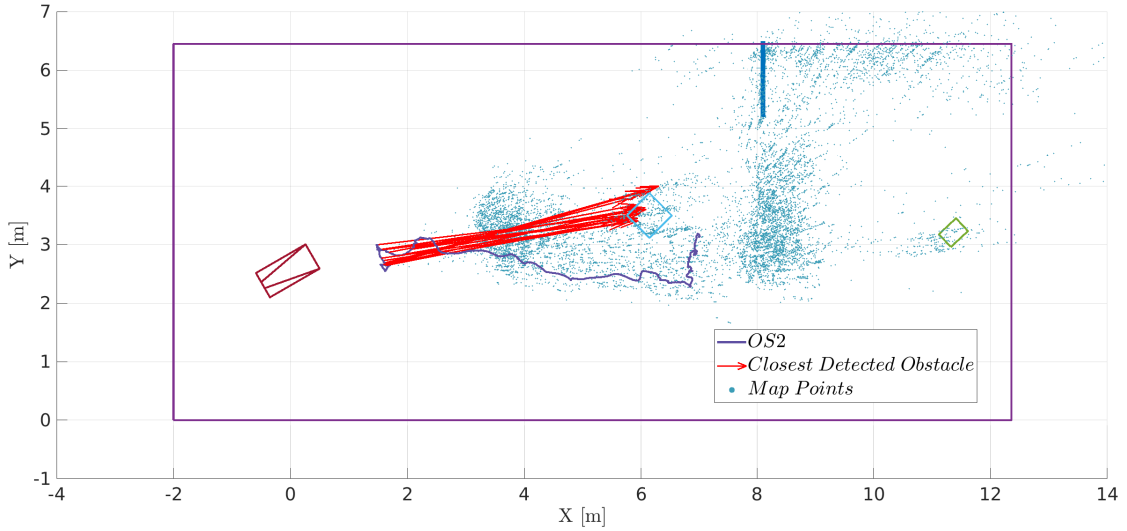
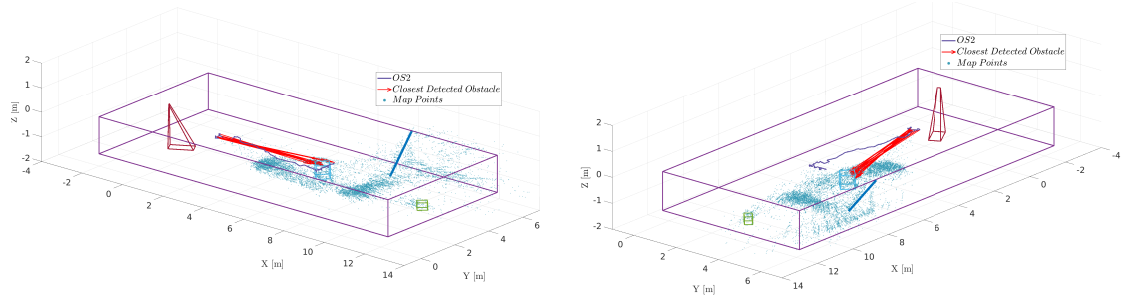



**Figure 5.20:** Top view of the closest detected obstacle of the test case full resolution mode subsea simulated lighting conditions, visualized as the red vectors.




**(a)** **(b)**

**Figure 5.21:** Bird view **(a)** and **(b)** of the closest detected obstacle plot, of the test case full resolution mode subsea simulated lighting conditions.

### 5.1.4 Binned Mode Subsea Simulated Lighting Conditions

**Table 5.4:** The static parameters of the WC-ROV VSLAM system and data set attributes of the test case binned mode subsea simulated lighting conditions.

| | | | |
|---|---|---|---|
| **CLAHE** | $clip_l$ | 8 | [-] |
| | $tile_w$ | 85 | $[px]$ |
| | $tile_h$ | 64 | $[px]$ |
| **ORB-SLAM2** | $n_f$ | 1500 | [-] |
| **RANSAC** | $\mathbf{t_r}$ | 0.13 | $[m]$ |
| **Clustering** | $\mathbf{t_c}$ | 0.09 | $[m]$ |
| | $c_{min}$ | 20 | [-] |
| | $c_{max}$ | 10000 | [-] |
| **Data Set** | FPS | 24 | [-] |
| | $f_{fps}$ | 1 | [-] |

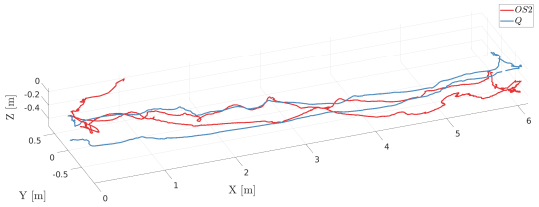
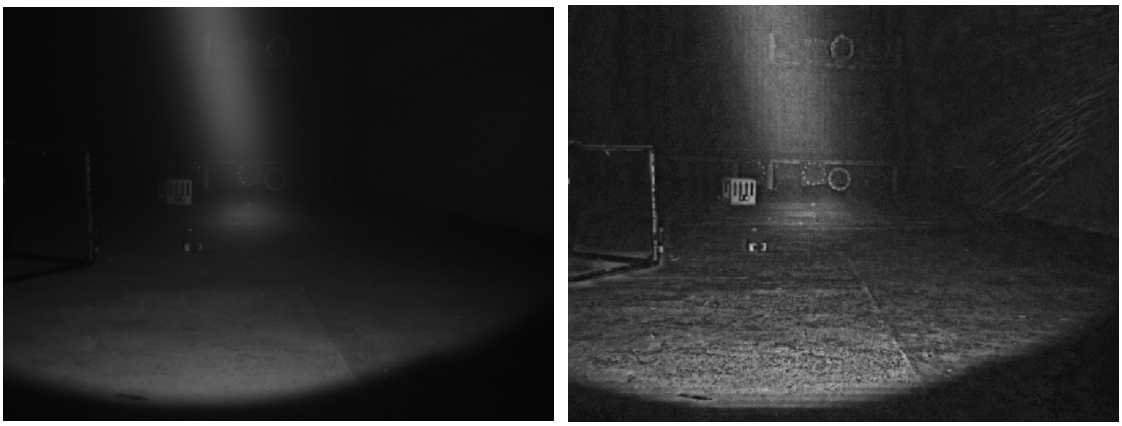


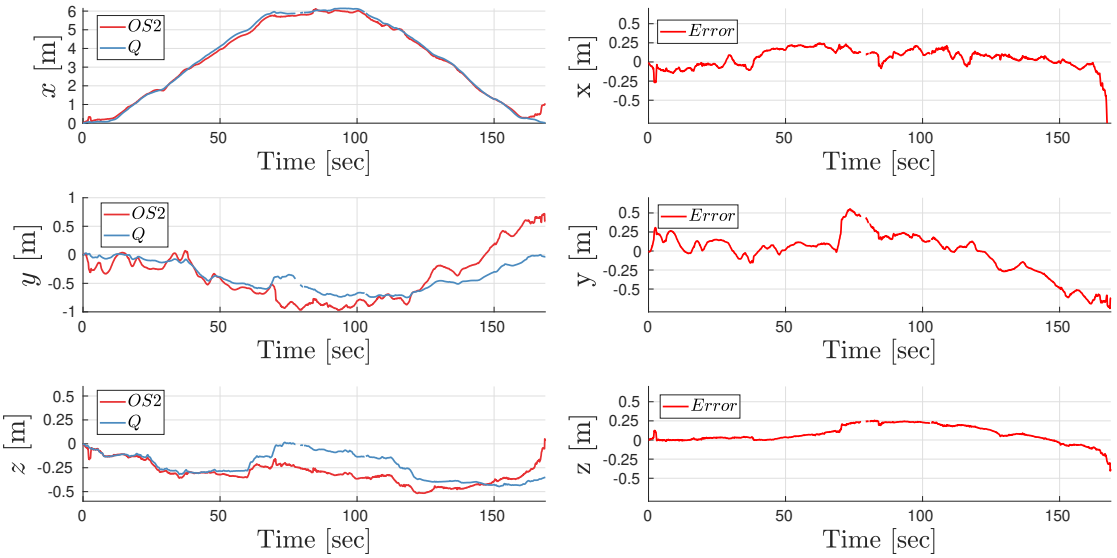

**Figure 5.22:** Trajectory of ORB-SLAM2 (orange) and Qualisys (blue) in the test case binned mode subsea simulated lighting conditions.



**(a)** Unprocessed **(b)** With CLAHE

**Figure 5.23:** Sample image from binned mode subsea simulated light conditions data set.

#### 5.1.4.1 Position Estimate

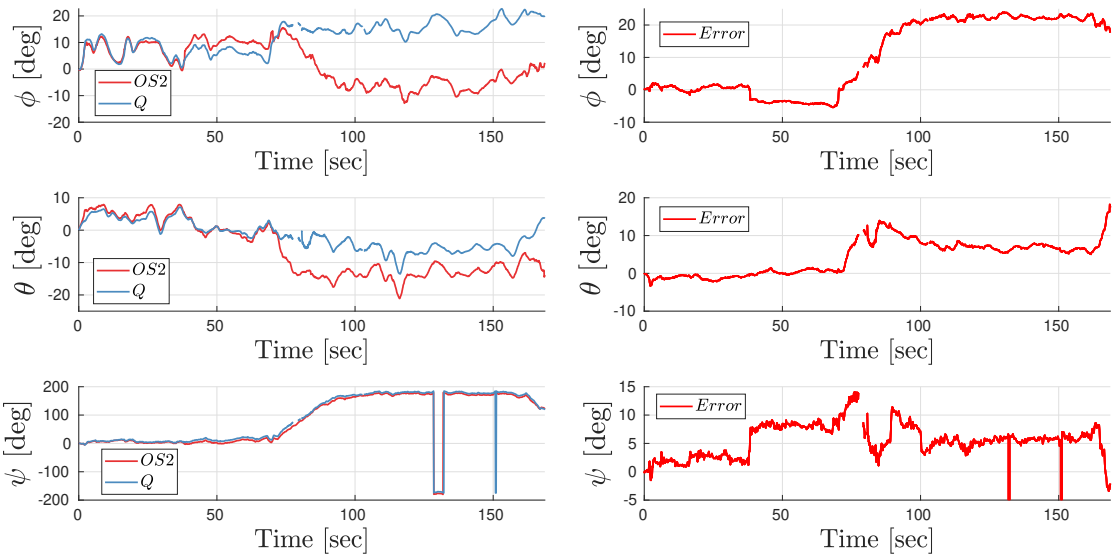



**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) position.

**(b)** The calculated error of the comparative plots of OS2 and Q position.

**Figure 5.24:** The position estimates of the test case binned mode subsea simulated lighting conditions.

#### 5.1.4.2 Orientation Estimate




**(a)** Comparative plot of the ORB-SLAM2 (OS2) estimated and Qualisys measured (Q) orientation.

**(b)** The calculated error of the comparative plots of OS2 and Q orientation.

**Figure 5.25:** The orientation estimates of the test case binned mode in ideal subsea simulated lighting conditions.

#### 5.1.4.3 Map Estimation

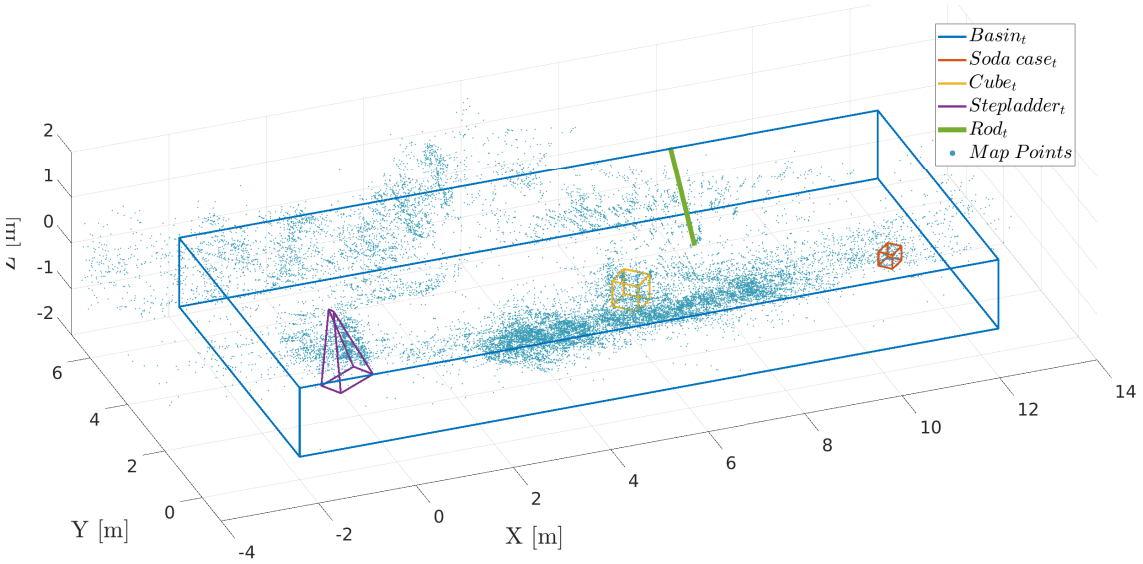



**Figure 5.26:** The estimated map points and measured underwater obstacle course of the test case binned mode subsea simulated lighting conditions.

#### 5.1.4.4 Closest Obstacle Detection

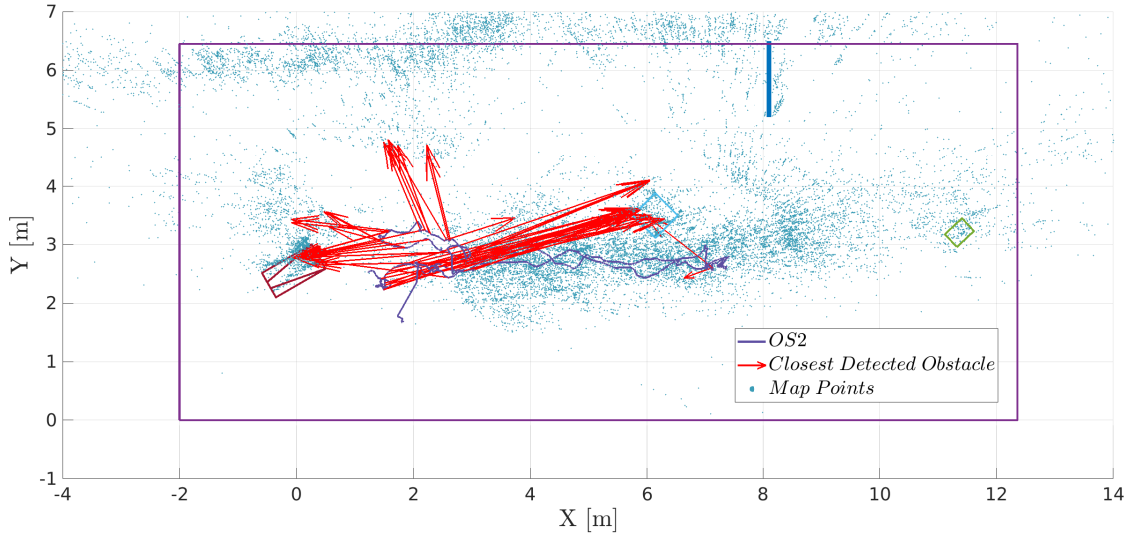
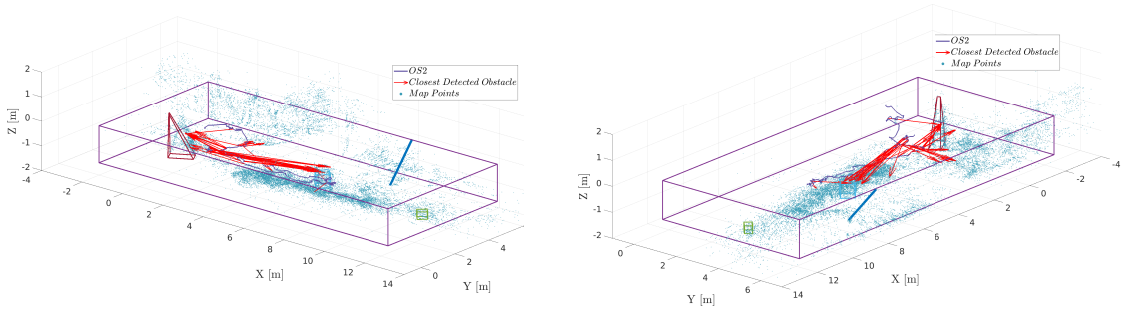



**Figure 5.27:** Top view of the closest detected obstacle of the test case binned mode subsea simulated lighting conditions, visualized as the red vectors.




**(a)** **(b)**

**Figure 5.28:** Bird view **(a)** and **(b)** of the closest detected obstacle plot, of the test case binned mode subsea simulated lighting conditions.

### 5.1.5 Induced Loop Closures

In order to explore the effect of loop closures, the data set for the binned mode ideal lighting condition was ran on the WC-ROV system in loops to forcibly induce loop closures. After three loops, the resulting estimated map of the underwater obstacle course is presented in Figure 5.29.

#### 5.1.5.1 Closest Obstacle Detection

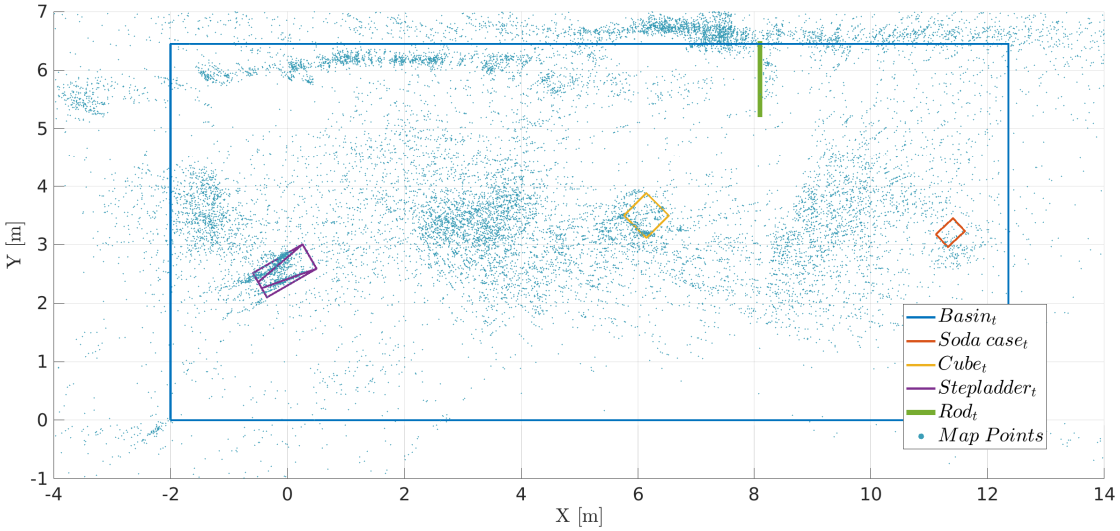



**Figure 5.29:** The estimated map of the binned mode ideal lighting condition test case run in three loops.

## 5.2 HIL-testing Results

The results of the HIL-testing is presented using a three dimensional plot of the logged ROV position messages received by the Autonomy Framework, and the logged obstacle messages sent by the altered LabView communication node.

Figure 5.30 shows the data of the messages sent and received in an arbitrary simulation of the ROV Minerva encountering an obstacle during a mapping mission. The blue line is

the received global ROV position in the NED frame, while the red line denotes the ROV position when detected obstacle messages were sent. In the simulation setup, the altered LabView communication node sent the closest detected obstacle messages when the ROV was within a distance of 5 m from the detected obstacle.

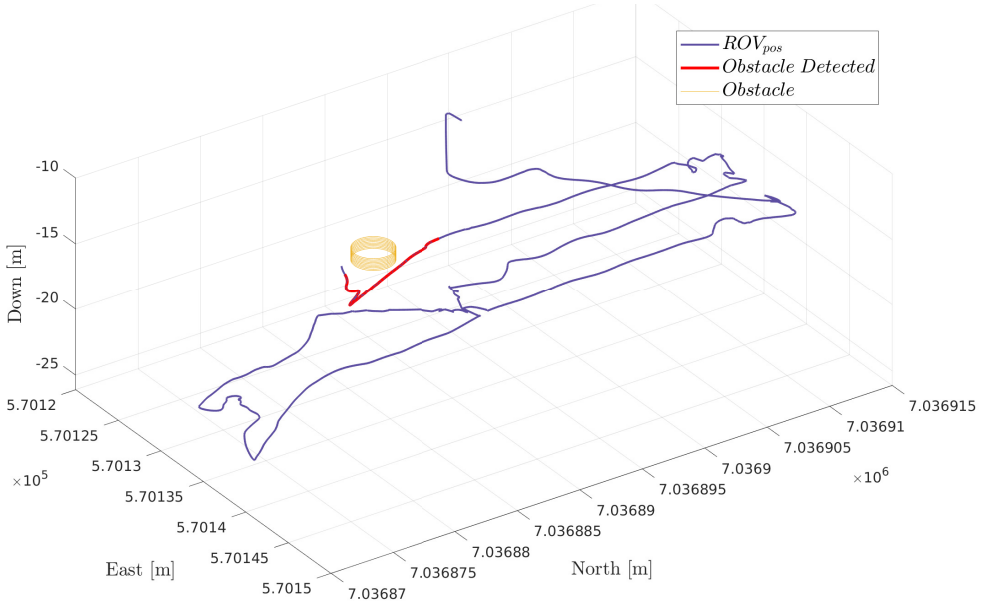



**Figure 5.30:** The data of the messages sent and received during an arbitrary simulation mission consisting of the received ROV position (blue line), and the interval of when detected closest obstacle messages is sent (red line).

# Chapter 6

# Discussion

## 6.1 Underwater Obstacle Course Experiment Discussion

This section presents the discussion of the results from the underwater obstacle course experiment: the position estimates, map estimates and the closest obstacle detection.

### 6.1.1 Initial Remarks

#### Qualisys Problems

Before addressing the experiment results, the problems of the ground truth measurement made by Qualisys needs to be mentioned. Due to Qualisys loosing the measurements of the optical tracking targets, the stereo camera rig body frame of Qualisys was reinitialized during both the binned mode test cases, and for the full resolution in ideal lighting condition. The reinitialization caused the direction of the x- and y-axis of the Qualisys body frame to change, and thus switching the signs of the measured roll and pitch. The switching of signs was manually corrected in the orientation plots in the Figures 5.4, 5.11 and 5.25, but the difficulty of finding the exact moment of axis direction change, may have caused the magnitude of the errors after the direction change to be somewhat misleading. The loss of measurement occurred approximately halfway through in the mentioned test cases, and is visible as missing measurements.

The reinitialization was not a problem in the full resolution subsea simulated lightning test case due to ORB-SLAM2 failing to estimate before reaching halfway through the test. Note that the Qualisys loss of tracking occured at the very end of the underwater obstacle course, suggesting that the obstacle course was place poorly within the calibration volume of Qualisys.

#### Halved Stereo Image Frequency in Full Resolution Mode

Initially, when running WC-ROV VSLAM system in full resolution mode, ORB-SLAM2 lost frequently tracking during the test runs. The tracking loss was caused by the increased computational complexity of ORB-SLAM2 conducting feature detection and matching on the full resolution images. The frequency of published estimations reduced from 14 Hz to 7 Hz, resulting in the scenery change in between each received image pair processed

by ORB-SLAM2 being to large for ORB-SLAM2 to track successfully. In order to still examine the effects of using the full resolution capabilities of the cameras, the publishing frequency of the raw stereo image pairs was halved from 14 Hz to 7 Hz, slowing down the experienced motion of the stereo camera rig and doubling the running time of the tests.

#### Loop Closure

In all of the test cases, no loop-closures were induced. No previously visited keyframes of ORB-SLAM2 were connected to new emerging keyframes while exploring. It meant that no important spacial constraints where introduced to the BA, and thus eliminating the possibility to correct for error drift. In the loop induced test of the binned mode ideal light condition in Figure 5.29, it is shown that the top basin wall has been better aligned with the introduction of loop closures. An important prerequisite for loop closures, is to have partly the same viewing angle of the scene such that the keyframe signatures in the bag of word place recognition module of ORB-SLAM2 are sufficiently similar enough to be identified as connected. The path of the stereo camera rig in the underwater obstacle course did barely show any revisited places from the same viewing angle, thus resulting in no detected loop closures. If loop closures had occurred in the data sets, less drift in mapping and positioning would have occur.

### 6.1.2 Position Estimates

The position estimates of ORB-SLAM2 in the WC-ROV VSLAM system in the Figures 5.3, 5.10, 5.17 and 5.24, shows in general to have small errors with low standard deviation in the first 25 s to 40 s of the tests. However, as the system continuous on exploring the environment the magnitudes of the errors increases. The increasing error is mainly caused by ORB-SLAM2 failing to map the environment consistently during excessive rotations, thus estimating the corresponding positions and orientations wrongly. It is clearly seen during the 180° yaw rotation halfway through the tests, where the orientation errors increase drastically. The orientation error manifests itself to the position estimates when the camera translates, and the positions of $x$, $y$ and $z$ can be seen to drift accordingly. The reader is referred to the Figures 5.10 and 5.11 of the binned mode ideal light conditions for example of the position error drift, where the error magnitudes of the orientations increases between 10° to 15° during the 180°. In the full resolution mode subsea simulated lighting in Figures 5.17 and 5.18, the estimation failed before reaching the 180° turn, but an small increase in the pitch error can be seen occurring by the abrupt yaw rotation at around 30 s.

The accuracy difference in the position estimates running in ideal lighting conditions or subsea simulated conditions are minimal. The biggest difference is the two times occurrence of tracking loss and relocalization in the binned mode subsea simulated lighting, on contrary to the ideal lighting where non occurred. The tracking loss and relocalization is seen in Figure 5.24 around 40 s and 75 s as a sudden change in the position. Tracking loss and relocalization did also occur for the full resolution ideal light in Figure 5.3, but this was not related to the lighting conditions, but rather because of computational time. In the full resolution mode subsea simulated lighting, a complete tracking loss with no relocalization resulting in the estimation ending. On closer inspection of the data set, it was discovered that the tracking loss occurred during a fast and sudden rotation proving to much view difference for features of stereo image pairs and estimated map to be matched.

To summarize the position estimates, the estimated positions have good accuracy in local areas in short time intervals, but the position error accumulates when the stereo camera rig continues on exploring the environment. Abrupt and excessive rotations are a source of increased errors during the exploration. Loop closures would correct the error drift of the current position, but will introduce, as with relocalizations, a sudden jump in the position estimate and hence limiting their application area e.g. could introduce gain jumps if used in the WC-ROV controller causing unstable control and damages to the WC-ROV actuators.

### 6.1.3 Map Estimates

The estimated map of each test case provided in general adequate spatial relation between the obstacles of the underwater obstacle course, with some inconsistency of previously mapped and newly mapped obstacles. The maps estimated in full resolution mode provided slightly higher local detailing compared to the binned mode, while the binned mode ideal light condition test case, was the only test case where the last yaw rotation of 180° was performed without loss of tracking. In its estimated map in in Figure 5.12, it is shown that the basin wall mapped in the last rotation misaligned with the existing mapped basin wall. This illustrates how the error drift affects the consistency of previously mapped obstacles with newly mapped obstacle. The same test case was run doing three loops to induce loops closures, and in its corresponding map in Figure 5.29, it can be seen how the basin wall aligned after the map update due to the linking of loop keyframes in the BA.

The difference between the mapping in the ideal and subsea simulated lighting conditions are related to the ORB features being detected in a smaller area of the camera FOV, and the increased use of CLAHE introducing more noise in the visual measurements. The smaller detection area of ORB features, were caused by the CLAHE not capable of compensating for the complete uneven light distribution of the artificial light source. The detected features were accordingly centered around the illuminated areas of the images, giving an undesired low spread of tracked map points. This is undesired both for the mapping purpose, and for the tracking in ORB-SLAM2. The increased noise of CLAHE, did also cause phantom points to occur by feature points in the light cone of the artificial illumination being matched and triangulated. Some of these points were caused by the backscattering of larger particles in the water, which is something that could be expected to happen more of if the WC-ROV VSLAM system is tested in the field on Minerva.

To summarize the map estimation, the estimated maps provided adequate spatial relations, but with some inconsistency of previously and newly mapped obstacles. The full resolution mode provides a slightly more locally detailed map, and the maps of the subsea simulated environment have more noise than the ideal light conditions maps. Note that the maps were manually aligned, making the true mapping results of the different test cases deviate slightly.

### 6.1.4 Closest Obstacle Detection

The closest obstacle detection algorithm performs well in the underwater obstacle course. In all test cases, it manages to detect the closest obstacle, and provided an accurate local spatial relation between the stereo camera rig and immediate closest obstacles. In the subsea simulated lighting conditions however, the algorithm was affected by the increased noise of the point cloud, and the reduced accuracy of the basin floor estimation. The effect

of the noise is visible as more arrows being drawn due to the fluctuation of the inferred obstacle position, while the inaccurate estimation of the basin floor is shown as closest detected obstacles being detected at the basin floor.

It was attempted to cope with the increased noise by reducing the clustering distance threshold in the subsea simulated lighting test cases giving some improvements, but a considerable amount of the fluctuations remained. The fluctuations of the inferred obstacle position did also increase with the duration of the test runs. It occurred due to the plane fitting of RANSAC being confused by new misaligned estimated basin floors being introduced in the point cloud. The RANSAC plane is fitted at each received point cloud, causing different set of points to be defined as seabed floor and removed. This causes continuous changes in basis of the clustering, resulting in increased fluctuation in the estimated position of the obstacles.

The obstacles detected at the basin floor, is caused by the RANSAC algorithm failing to remove every point in the point cloud associated with the assumed flat seabed. This is again due to the misalignment of the estimated basin floor, which results in the estimation consisting of more than just one plane. The effect of noise and inaccurate seabed estimations are especially seen in the binned mode subsea simulated lighting condition of Figure 5.28, where the basin floor was inferred as the closest obstacle seven times.

To summarize the closest obstacle detection, the closest obstacle detection manages to detect and infer the closest obstacle, but the performance reduces in the subsea simulated test cases due to the increased noise levels and map estimation misalignment. The robustness of the algorithm is thus somewhat questionable, and the performance in an real subsea environment, where the seabed could be naturally unaligned and contain curves, is hard to address without real life tests. It is expected that the performance of the algorithm would be negatively affected in real life tests.

### 6.1.5 Closing Remarks

#### Resolution Mode

The marginal quality increase using full resolution, and the lowered ORB-SLAM2 estimation frequency makes it inferior to the binned resolution mode. The binned resolution mode has increased light sensitivity, lowered signal to noise ration, and still manages to provide visual measurements of the environment containing enough information for ORB-SLAM2 to estimate results similar to the full resolution mode. Additionally, does the increased estimation frequency of ORB-SLAM2 provide more frequent estimations proving better usability in autonomous applications.

#### Comparability to Real Life WC-ROV Operation

The differences between the underwater obstacle course experiment compared to real life operation of an WC-ROV, is the size of the spatial and temporal interval covered during real life missions, the deviation from a flat seabed assumption in true subsea environments, the practically non existent turbidity in the basin of MC-lab limiting the scattering effects, and the quick and excessive rotations happening during the data collection in MC-lab. Due to these differences, it is hard conclude that the testes WC-ROV VSLAM system in the underwater obstacle course reflects the real life performance of the system on Minerva.

## 6.2 HIL-testing

The sole purpose of the HIL-testing in terms of the WC-ROV VSLAM system, was to verify the integration of the system in to the ROV Minerva Autonomy Framework. The provided plot in Figure 5.30 confirms that the communication was successful, thus proving that field tests on the ROV Minerva using the WC-ROV VLSAM system together with the ROV Autonomy Framework is possible.

## 6.3 Camera Calibration

It was initially expected to uncover a relation between the calibration parameters of the stereo camera rig and the given calibration distance. However, no such relation was uncovered due to the consistency of the calibration results largely depending on the quality of the data set. The variation of the calibration data set quality originates in the images of the calibration object having too similar planar orientation, thus producing weak constraints for non-linear minimization problem performed to estimate the camera parameters. The estimated camera parameters of poor calibration data sets, showed itself by estimating way different focal lengths of the two cameras of the stereo camera rig, and giving wrong relative orientations and translation.

# Chapter 7

# Conclusion

In this thesis, a real-time WC-ROV VSLAM system based on the stereo camera rig of the WC-ROV Minerva and the VSLAM method ORB-SLAM2 has been developed and implemented. In the system, ORB-SLAM2 estimates the position, orientation and a map of the environment based upon stereo image pairs provided by the stereo camera rig. The stereo images are before received in ORB-SLAM2, compensated for the uneven lighting caused by artificial illumination using the contrast enhancing method CLAHE, and are undistorted and rectified using the camera calibration parameters obtained from underwater camera calibration. The estimated stereo camera rig position and point cloud of ORB-SLAM2, is used in an clustering based closest obstacle detection algorithm in order to infer the closest obstacle to the stereo camera rig. The algorithm is based upon fitting a plane with RANSAC to filter out the points associated with the seabed, and then perform Euclidean based clustering on the remaining points to identify the surrounding obstacles. The real-time WC-ROV VSLAM system is capable of conveying the closest detected obstacle to the Autonomy Framework of Minerva using TCP connection.

The performance of the real-time WC-ROV VSLAM system was tested in an underwater obstacle course established in the basin of MC-lab. The system was both tested in ideal lighting condition and in subsea simulated lighting conditions, using both the full resolution capacity of the cameras, and the binned resolution mode halving the resolution and increasing the light sensitivity. In the underwater obstacle course experiment, the optical motion measurement from Qualisys was used as ground truth for the estimated position and orientation, while the measured obstacle dimensions were used as ground truth for the estimated map and the closes obstacle detection. The integration of the WC-ROV VSLAM system with the Autonomy Framework of Minerva, was verified with HIL-tests conducted together with the master students working on the Autonomy Framework of Minerva. The WC-ROV VSLAM system was altered to generate synthetic obstacles instead of true obstacles detected by visual input.

The results from the underwater obstacle course shows that the position estimates have good accuracy in local areas in short times intervals, but error accumulates when the stereo camera rig explores the environment. The estimated maps provides adequate spatial relations with some inconsistency of previously and newly mapped obstacles. The closest obstacle detection manages to detect and infer the closest obstacles, but the performance

reduces in the subsea simulated test cases due to increased noise levels and estimated map misalignment. Additionally, shows the results that the benefits of using cameras in full resolution is inferior to the binned mode due to reduced estimation frequency of ORB-SLAM2.

To conclude this thesis, the use of the VSLAM method ORB-SLAM2 in WC-ROV VSLAM system shows that the local situational awareness of WC-ROVs could be increased by using the estimated position and map of ORB-SLAM2, and that they can be used in autonomous features for the WC-ROV such as the proposed closest obstacle detection. The use of the estimated position to increase the local positioning, is however more questionable due to the increasing drift occurring and jumps in the estimates positions due to relocalization and loop-closures.

## Further Work

The following is the proposed further work in order to improve the WC-ROV VSLAM system.

The first proposed further work, is to the test the WC-ROV VSLAM system in true subsea conditions. Only by doing this, the real challenges of VSLAM applied on WC-ROV will be revealed. The second, is to enhance the synchronization of the stereo images by installing physical triggers between the stereo cameras. The Prosilica CG1380C support this feature, and can be done by wiring the specific trigger pin-out of the cameras together. The third, is to deal with lack of robustness in the closest detected obstacle algorithm of the WC-ROV VSLAM system. It can be dealt with by investigating alternative point cloud filtering and clustering techniques, focus on successfully removing the seabed associated points, or introduce logic that eliminates closest obstacles detected at seabed. The last proposed further work, is to investigate alternative underwater image enhancing methods. The Gray-World transformations of Buchsbaum [44], or the Distance Adjusted Gray-World approach of Bryson et al. [45] could be promising methods.

# References


[1] Robert D. Christ Robert L. Wernli Sr. *The ROV Manual*. Second Edition. Elsevier, 2014.

[2] *ROV MINERVA - NTNU*. URL: https://www.ntnu.edu/aur-lab/rov-minerva (visited on 06/25/2020).

[3] Asgeir Sørensen et al. "Development of dynamic positioning and tracking system for the ROV Minerva". In: Jan. 2012, pp. 113–128. DOI: 10.1049/PBCE077E_ch6.

[4] Franco Hidalgo and Thomas Bräunl. "Review of underwater SLAM techniques". In: *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*. 2015 6th International Conference on Automation, Robotics and Applications (ICARA). ISSN: null. Feb. 2015, pp. 306–311. DOI: 10.1109/ICARA.2015.7081165.

[5] Josep Aulinas et al. "Selective Submap Joining for underwater large scale 6-DOF SLAM". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. ISSN: 2153-0858. Oct. 2010, pp. 2552–2557. DOI: 10.1109/IROS.2010.5650438.

[6] Antoni Burguera Burguera and Francisco Bonin-Font. "A Trajectory-Based Approach to Multi-Session Underwater Visual SLAM Using Global Image Signatures". In: *Journal of Marine Science and Engineering* 7.8 (Aug. 2019), p. 278. DOI: 10.3390/jmse7080278. URL: https://www.mdpi.com/2077-1312/7/8/278 (visited on 11/29/2019).

[7] Ryan Eustice et al. "Visually Navigating the RMS Titanic with SLAM Information Filters". In: *Robotics: Science and Systems I*. Robotics: Science and Systems 2005. Robotics: Science and Systems Foundation, June 8, 2005. ISBN: 978-0-262-70114-3. DOI: 10.15607/RSS.2005.I.008. URL: http://www.roboticsproceedings.org/rss01/p08.pdf (visited on 12/01/2019).

[8] Alberto Quattrini Li et al. "Experimental Comparison of Open Source Vision-Based State Estimation Algorithms". In: *2016 International Symposium on Experimental Robotics*. Ed. by Dana Kulić et al. Springer Proceedings in Advanced Robotics. Cham: Springer International Publishing, 2017, pp. 775–786. ISBN: 978-3-319-50115-4. DOI: 10.1007/978-3-319-50115-4_67.

[9] 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 (Oct. 2015), pp. 1147–1163. ISSN: 1941-0468. DOI: 10.1109/TRO.2015.2463671.

[10] Georg Klein and David Murray. "Parallel Tracking and Mapping for Small AR Workspaces". In: *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. ISMAR '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 1–10. ISBN: 978-1-4244-1749-0. DOI: 10.1109/ISMAR.2007.4538852. URL: https://doi.org/10.1109/ISMAR.2007.4538852 (visited on 12/14/2019).

[11] Pep Lluis Negre Carrasco, Francisco Bonin-Font, and Gabriel Oliver Codina. "Stereo Graph-SLAM for Autonomous Underwater Vehicles". In: *Intelligent Autonomous Systems 13*. Ed. by Emanuele Menegatti et al. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2016, pp. 351–360. ISBN: 978-3-319-08338-4. DOI: 10.1007/978-3-319-08338-4_26.

[12] Pep Lluis Negre, Francisco Bonin-Font, and Gabriel Oliver. "Cluster-based loop closing detection for underwater slam in feature-poor regions". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016 IEEE International Conference on Robotics and Automation (ICRA). ISSN: null. May 2016, pp. 2589–2595. DOI: 10.1109/ICRA.2016.7487416.

[13] Ayoung Kim and Ryan M. Eustice. "Real-Time Visual SLAM for Autonomous Underwater Hull Inspection Using Visual Saliency". In: *IEEE Transactions on Robotics* 29.3 (June 2013), pp. 719–733. ISSN: 1941-0468. DOI: 10.1109/TRO.2012.2235699.

[14] F. Menna et al. "EVALUATION OF VISION-BASED LOCALIZATION AND MAPPING TECHNIQUES IN A SUBSEA METROLOGY SCENARIO". In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-2/W10 (Apr. 17, 2019), pp. 127–134. ISSN: 2194-9034. DOI: 10.5194/isprs-archives-XLII-2-W10-127-2019. URL: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2-W10/127/2019/ (visited on 12/01/2019).

[15] Nick Weidner et al. "Underwater cave mapping using stereo vision". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017 IEEE International Conference on Robotics and Automation (ICRA). ISSN: null. May 2017, pp. 5709–5715. DOI: 10.1109/ICRA.2017.7989672.

[16] Yi Ma et al. "Image Formation". In: *An Invitation to 3-D Vision: From Images to Geometric Models*. Ed. by Yi Ma et al. Interdisciplinary Applied Mathematics. New York, NY: Springer, 2004, pp. 44–74. ISBN: 978-0-387-21779-6. DOI: 10.1007/978-0-387-21779-6_3. URL: https://doi.org/10.1007/978-0-387-21779-6_3 (visited on 11/30/2019).

[17] D. C. Brown. "Decentering Distortion of Lenses". In: *Photogrammetric Engineering and Remote Sensing* (1966). URL: https://ci.nii.ac.jp/naid/10022411406/ (visited on 06/07/2020).

[18] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Google-Books-ID: si3R3Pfa98QC. Cambridge University Press, 2003. 676 pp. ISBN: 978-0-521-54051-3.

[19] Z. Zhang. "A flexible new technique for camera calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (Nov. 2000), pp. 1330–1334. ISSN: 01628828. DOI: 10.1109/34.888718. URL: http://ieeexplore.ieee.org/document/888718/ (visited on 12/13/2019).

[20] Andrea Fusiello, Emanuele Trucco, and Alessandro Verri. "A compact algorithm for rectification of stereo pairs". In: *Machine Vision and Applications* 12.1 (July 1, 2000), pp. 16–22. ISSN: 1432-1769. DOI: 10.1007/s001380050120. URL: https://doi.org/10.1007/s001380050120 (visited on 06/07/2020).

[21] G.-Q. Wei and S.D. Ma. "A complete two-plane camera calibration method and experimental comparisons". In: *1993 (4th) International Conference on Computer Vision*. 1993 (4th) International Conference on Computer Vision. May 1993, pp. 439–446. DOI: 10.1109/ICCV.1993.378183.

[22] Stephen J. Maybank and Olivier D. Faugeras. "A theory of self-calibration of a moving camera". In: *International Journal of Computer Vision* 8.2 (Aug. 1992), pp. 123–151. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/BF00127171. URL: http://link.springer.com/10.1007/BF00127171 (visited on 06/25/2020).

[23] Stephen M Pizer et al. "Adaptive histogram equalization and its variations". In: *Computer vision, graphics, and image processing* 39.3 (1987), pp. 355–368.

[24] Chris Harris and Mike Stephens. "A combined corner and edge detector". In: *In Proc. Fourth Alvey Vision Conference*. 1988, pp. 147–152.

[25] Ethan Rublee et al. "ORB: an efficient alternative to SIFT or SURF". In: *In ICCV*.

[26] Edward Rosten and Tom Drummond. "Machine Learning for High-Speed Corner Detection". In: *Computer Vision – ECCV 2006*. Ed. by Aleš Leonardis, Horst Bischof, and Axel Pinz. Vol. 3951. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 430–443. ISBN: 978-3-540-33832-1 978-3-540-33833-8. DOI: 10.1007/11744023_34. URL: http://link.springer.com/10.1007/11744023_34 (visited on 06/30/2020).

[27] Michael Calonder et al. "BRIEF: Binary Robust Independent Elementary Features". In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 778–792. ISBN: 978-3-642-15561-1. DOI: 10.1007/978-3-642-15561-1_56.

[28] Cesar Cadena et al. "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". In: *IEEE Transactions on Robotics* 32.6 (Dec. 2016), pp. 1309–1332. ISSN: 1941-0468. DOI: 10.1109/TRO.2016.2624754.

[29] Cyrill Stachniss, John J. Leonard, and Sebastian Thrun. "Simultaneous Localization and Mapping". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer Handbooks. Cham: Springer International Publishing, 2016, pp. 1153–1176. ISBN: 978-3-319-32552-1. DOI: 10.1007/978-3-319-32552-1_46. URL: https://doi.org/10.1007/978-3-319-32552-1_46 (visited on 11/28/2019).

[30] Stephen M. Chaves et al. "Pose-Graph SLAM for Underwater Navigation". In: *Sensing and Control for Autonomous Vehicles: Applications to Land, Water and Air Vehicles*. Ed. by Thor I. Fossen, Kristin Y. Pettersen, and Henk Nijmeijer. Lecture Notes in Control and Information Sciences. Cham: Springer International Publishing,

2017, pp. 143–160. ISBN: 978-3-319-55372-6. DOI: 10.1007/978-3-319-55372-6_7. URL: https://doi.org/10.1007/978-3-319-55372-6_7 (visited on 12/02/2019).

[31] Bill Triggs et al. "Bundle Adjustment — A Modern Synthesis". In: *Vision Algorithms: Theory and Practice*. Ed. by Bill Triggs, Andrew Zisserman, and Richard Szeliski. Red. by Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen. Vol. 1883. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 298–372. ISBN: 978-3-540-67973-8 978-3-540-44480-0. DOI: 10.1007/3-540-44480-7_21. URL: http://link.springer.com/10.1007/3-540-44480-7_21 (visited on 12/09/2019).

[32] Raúl Mur-Artal and Juan D. Tardós. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras". In: *IEEE Transactions on Robotics* 33.5 (Oct. 2017), pp. 1255–1262. ISSN: 1941-0468. DOI: 10.1109/TRO.2017.2705103.

[33] Dorian Galvez-López and Juan D. Tardos. "Bags of Binary Words for Fast Place Recognition in Image Sequences". In: *IEEE Transactions on Robotics* 28.5 (Oct. 2012), pp. 1188–1197. ISSN: 1941-0468. DOI: 10.1109/TRO.2012.2197158.

[34] Tarindu Hewage, Kristine Klingan, and Erik Holven. *Lecture Notes TMR4120 Underwater Engineering*. Unpublished.

[35] C. J. Funk, S. B. Bryant, and Jr Heckman. *Handbook of Underwater Imaging System Design*. NUC-TP-303. NAVAL UNDERSEA CENTER SAN DIEGO CA, July 1972. URL: https://apps.dtic.mil/docs/citations/AD0904472 (visited on 12/11/2019).

[36] Torger Holtsmark and Johannes Skaar. *brytning – optikk*. In: *Store norske leksikon*. June 12, 2018. URL: http://snl.no/brytning_-_optikk (visited on 06/07/2020).

[37] Eugene Hecht. *Optics: Pearson New International Edition*. Pearson Education UK, 2013. ISBN: 978-1-292-03480-5. URL: http://ebookcentral.proquest.com/lib/ntnu/detail.action?docID=5137983 (visited on 05/25/2020).

[38] Martin A. Fischler and Robert C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (June 1, 1981), pp. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: https://doi.org/10.1145/358669.358692 (visited on 06/07/2020).

[39] Radu Bogdan Rusu. "Clustering and Segmentation". In: *Semantic 3D Object Maps for Everyday Robot Manipulation*. Vol. 85. Series Title: Springer Tracts in Advanced Robotics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 75–85. ISBN: 978-3-642-35478-6 978-3-642-35479-3. DOI: 10.1007/978-3-642-35479-3_6. URL: http://link.springer.com/10.1007/978-3-642-35479-3_6 (visited on 06/07/2020).

[40] Jon Louis Bentley. "Multidimensional binary search trees used for associative searching". In: *Communications of the ACM* 18.9 (Sept. 1, 1975), pp. 509–517. ISSN: 0001-0782. DOI: 10.1145/361002.361007. URL: https://doi.org/10.1145/361002.361007 (visited on 06/25/2020).

[41] *Prosilica GC 1380 Data Sheet*.

[42] Stein M. Nornes, Martin Ludvigsen, and Asgeir J. Sørensen. "Automatic relative motion control and photogrammetry mapping on steep underwater walls using ROV".

In: *OCEANS 2016 MTS/IEEE Monterey*. OCEANS 2016 MTS/IEEE Monterey. Sept. 2016, pp. 1–6. DOI: 10.1109/OCEANS.2016.7761252.

[43] *Camera calibration With OpenCV — OpenCV 2.4.13.7 documentation*. URL: https://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html (visited on 12/14/2019).

[44] G. Buchsbaum. "A spatial processor model for object colour perception". In: *Journal of the Franklin Institute* 310.1 (July 1, 1980), pp. 1–26. ISSN: 0016-0032. DOI: 10.1016/0016-0032(80)90058-7. URL: http://www.sciencedirect.com/science/article/pii/0016003280900587 (visited on 06/30/2020).

[45] Mitch Bryson et al. *Colour-Consistent Structure-from-Motion Models using Underwater Imagery*.

# Appendices

# Appendix A

## Manuscript Applied to Oceans 2020

# Increased Autonomy and Situation Awareness for ROV Operations

Fan Gao
*Department of Marine Technology*
*Norwegian University of Science and technology, NTNU*
Trondheim, Norway

Signe B. Moltu
*Department of Marine Technology*
*Norwegian University of Science and technology, NTNU*
Trondheim, Norway

Erlend R. Vollan
*Department of Marine Technology*
*Norwegian University of Science and technology, NTNU*
Trondheim, Norway

Shuyuan Shen
*Department of Marine Technology*
*Norwegian University of Science and technology, NTNU*
Trondheim, Norway

Martin Ludvigsen
*Department of Marine Technology*
*Norwegian University of Science and technology, NTNU*
Trondheim, Norway
martin.ludvigsen@ntnu.no

***Abstract*—This paper proposes a semi-autonomous mission planning and management architecture for a Remotely Operated Vehicle (ROV). The work has focused on three main aspects to increase the autonomy of ROV inspections: developing a plan-based mission management architecture for ROV global navigation and local intervention, integrating path planning as a refinement of actions, and real-time communication with visual VSLAM (VSLAM) systems for obstacle detection and motion estimation. The architecture is inspired by the hybrid agent architecture [1] using a deliberative and a reactive layer to perform planned tasks and handle contingencies. Hardware-in-the-loop (HIL) simulations are carried out to test and verify the capability and feasibility of the proposed layered mission management architecture. The results demonstrate that the mission planning and management system can automatically generate the correct plan and guide the ROV to achieve its mission goals.**

***Index Terms*—autonomy, obstacle avoidance (OA), mission planning, visual-SLAM, path planning**

## I. Introduction

Enabling autonomy in ROV operations will increase efficiency and save costs [2]. The general purpose of the work in this paper is to develop a mission planning and management system for underwater navigation and operation, and integrate vision-based situation awareness in ROV localization and obstacle detection. Under manual operation, human operators face risks of wrongly analyzing the situation, performing unnecessary or even fatal operations, increasing operation risks and costs. A plan-based mission management system can significantly avoid the above problems and simplify the execution procedure.

By placing a docking station on the seabed, long-term resident ROVs are feasible. With a docking station placed on the seabed, ROVs can charge their batteries and receive and transfer data to the operators onshore. Creating functionality for this is a big step in the direction of fully autonomous long-term missions. Furthermore, in order to increase ROV autonomy, situation awareness and accurate local positioning are required. Existing acoustic positioning systems covers the global position of ROVs well, but tends to lack adequate local positioning precision. Stereo cameras are cheap sensors and can be used for ROV local navigation when paired with computer vision techniques. Visual simultaneous localization and mapping (VSLAM) is the problem of using visual inputs to concurrently construct a map of an unknown environment while estimating its location within it. VSLAM has been extensively researched on land-based vehicles proving good results. Developing subsea-based VSLAM systems could increase the local positioning accurately and simultaneously grant local situational awareness by providing a local map.

An overview of existing ROV standards considering underwater vehicle autonomy is presented in [2]. Four levels of autonomy were suggested in [3], being Manual Operation, Management by consent, Management by exception, and Fully autonomous. The level of situational awareness, decision making and control are increased with increasing levels of autonomy. This paper aims to increase the level of autonomy towards level three (Management by exception) for ROV subsea intervention. A new definition of symbolic actions for ROV mission planning and management is proposed, enabling automated planning to guide and integrate with mission management systems for task execution.

### A. Related Work

This paper is an extension of the work done in [4]. The extension is concerned with implementing a governing mission

planner, deciding what the sequence of actions to carry out to reach the goal states. In addition, a path planner is added to ensure safe transit around known obstacles. VSLAM is further included to detect and warn about unknown obstacles.

A similar approach to enhance path planning was tested in [5]. An A* algorithm was proposed for the path planning of an autonomous underwater vehicle (AUV) in a partially-known environment with promising results. VSLAM was suggested for localization, where only a simple VSLAM method was used with a forward-facing sonar.

The benefits of choosing the A* algorithm as the path planning method for underwater vehicles were described in [6]. The algorithm has high maturity, is easy to implement and store, and has a low computational cost. The downsides, however, were that the algorithm has low efficiency and is not suitable for large scale space searches.

Filtering-based approaches and graph-based formulations are two typical kinds of method appiled to solve a SLAM problem. With the improvement of computer computing power, graph-based SLAM has been mainstream in recent decades. A graph is constructed whose vertices represent vehicle positions or landmarks and the edge between two vertices represents a sensor observation that constrains the vertices proposed by Lu and Milios [7]. The sparse linear algebra makes it efficient to solve the optimization of the error minimization problem.

The outline of the paper is as follows: Section II presents the ROV mission management architecture, the mission planning methods, path planning for homing and docking, and vision-based motion estimation and obstacle detection strategy. The simulation results are presented and discussed in Section III. Section IV concludes on the performance of the proposed semi-autonomous mission planner and suggests modifications for further work.

## II. Method

### A. ROV Control System and HIL Simulator

The ROV control system used for simulations has been developed by the Applied Underwater Robotics Laboratory (AUR Lab) since 2010. The control system was firstly developed for DP and trajectory tracking [8]. As shown in Fig. 1, the control system is built on two basic modules: Frigg Graphical User Interface (GUI) and Njord Control system. The Frigg GUI enables high-level control and mode selection of missions, while the Njord control system performs low-level control, including a guidance system and a Nonlinear PID-controller. The proposed Autonomy Framework is added in Frigg GUI, providing autonomous mission execution command for lower-level control.

The Verdandi Simulator is a HIL simulator which incorporates hardware components into the numerical simulation environment. The HIL simulations are necessary to test the performance of the system and debug the system before fieldwork. The simulator constructs a mathematical model of the ROV and can take disturbance and environmental forces into consideration during virtual experiment. HIL simulations

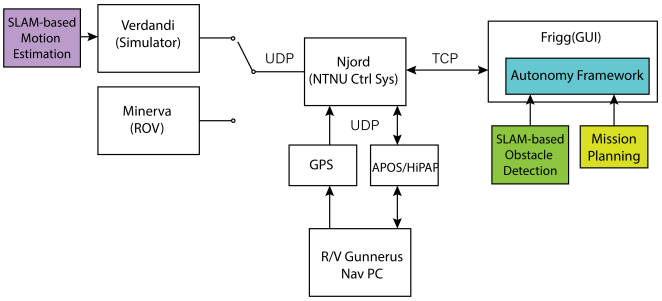



Fig. 1. Structural chart of the ROV control system

are implemented by connecting the Verdandi simulator and the ROV control system using TCP connection.

### B. Mission Planning and Management Architecture

A design of hybrid mission planning and management system is proposed, consisting of a deliberative layer performing planned actions to achieve mission goals, a reactive layer responding fast to non-predictable events and a control execution layer acting as a coordinate mechanism that determines if actions from either the deliberative or the reactive layer should be executed by the lower-level controllers. Fig. 2 is a diagram of the hybrid mission planning and management system.

The deliberative layer performs autonomous behaviors based on known situations and events. The mission planner in this layer enables the ROV to plan tasks automatically and performs necessary re-planning. For the given user commands and known environment derived from sensors and a world model, the mission planner can recognize and categorize the tasks, generating a plan that fulfills the mission requests, and deliver a set of actions or commands to the mission controller. The mission will be re-planned automatically due to events such as failures in control, environment changes (such as the encounter with unexpected obstacles), and changes of mission target. In the deliberative layer, a layered design is implemented for global navigation and local operations as sub-missions. With actions of *Station Keeping*, *Launch*, *Descent*, *Transit* and *Operation*, global navigation enables the vehicle to approach a target location where local operation is to be performed. After the performed operation, the ROV is asked to go back to the predefined ending location. The sub-planner acts as the refinement of *Local Operation*. Possible actions for *Local Operation* are *Mapping*, *Sampling* and *Charging*. A fast-forward search [9] method for mission planning is implemented to generate a near-optimal plan that fulfills the goals. An A* path planning algorithm for homing and general path planning is implemented in operation as refining of sampling and charging. A docking option is also implemented to simulate full homing and docking behavior.

The reactive layer responds to contingencies by analyzing sensor data, reasoning unexpected or unknown situations, modifying and interrupting missions. Exceedance of cable tension and obstacle avoidance are the two reactive behaviors implemented in the reactive layer. For actions that require

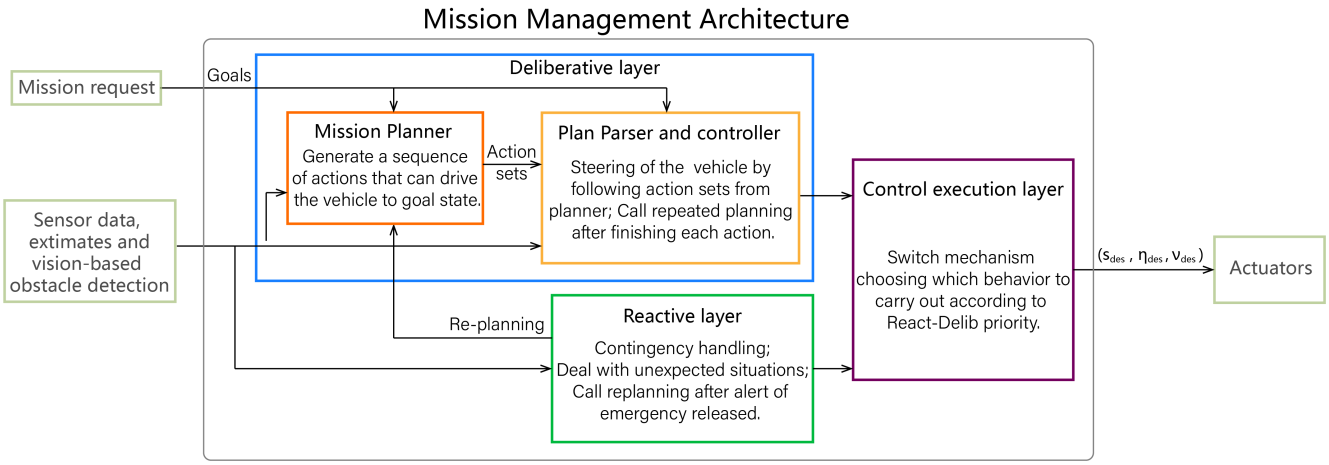



Fig. 2. Mission Planning and Management Architecture

path planning, such as sampling and charging, new detected obstacles are updated to the map, and re-planning is called to generate a new path that avoids both the known and the detected obstacles. For the other actions, a new waypoint is created based on the distance to the obstacle, the size of the obstacle and safety parameters for avoiding obstacles, as seen in Fig. 3.

A method of heading change is used to deal with this situation. When encountering a new obstacle, the direction of heading change is determined by $\beta$ and $\psi$. $\beta$ demonstrates the relative direction of the obstacle to the vehicle, and $\psi$ is the heading angle of the vehicle. If the heading angle $\psi$ is smaller than $\beta$, the new heading angle is $\psi_{new} = \beta - \alpha$. Otherwise, the new heading angle is $\psi_{new} = \beta + \alpha$. The angle $\alpha$ is calculated as $sin(\alpha) = \frac{R}{S}$, based on the diameter of the dangerous region and the distance between the vehicle and the obstacle. $d$ is the diameter of the obstacle, and $e$ is a safety parameter. Thus, $R = \frac{D}{2} + e$ is the radius of the dangerous region. $L$ is the calculated length from the vehicle to the new waypoint. The position of the new waypoint is calculated based on the new heading angle and length $L$.

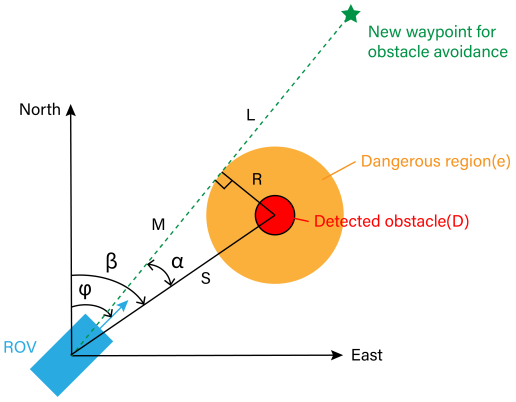



Fig. 3. Obstacle avoidance behavior

The global coordination function evaluates the output behaviors from the two architectures and will transfer inputs from the reactive architecture to the executor once it is activated. Since only one deliberative action is activated at a time in the deliberative layer and a coordinate mechanism is applied in the functional reactive layer to select the behavior with the highest priority among all activated behaviors under this layer. The control execution layer is thus organized as a selection of outputs between the deliberative layer and the reactive layer. It is responsible for deciding which layer and which action is activated and summing up the necessary parameters required by the control system.

### C. Integration of Path Planning

The A* algorithm, created by [10], was selected for path generation during missions that require path planning because it is easily adaptable, simplistic and performs well in known environments with low obstacle density [11]. Euclidean distance, presented in (1), was used for the heuristic function since it allows for movement in all eight neighboring tiles in a 2D grid space.

$$\sqrt{(x_{current} - x_{goal})^2 + (y_{current} - y_{goal})^2} \qquad (1)$$

As the entire mission uses depth-control, a 2D path is justified. However, as there will be fluctuations and uncertainties in depth, known obstacles above and below two meters of the desired depth will be seen as relevant obstacles by the algorithm. This is also reasonable considering the specifications of the simulated ROV.

The path created by the A* algorithm can often include unnecessary turns for an underwater vehicle (UUV), which can move in any direction, not just in a straight path from the center of one tile to the center of the next. To make the path more smooth and avoid unnecessary turns, the smoothing method *Moving Average Filter* was used, implemented with the equation shown in (2).

$$ys(i) = (y(i-2)+y(i-1)+y(i)+y(i+1)+y(i+2))/5 \quad (2)$$

Because of the Constant Jerk Guidance [12] used in the ROV control system, a high number of waypoints would mean a high number of stop's and go's. For this reason, collinear waypoints were firstly removed altogether from the generated path from the A* algorithm. However, as suggested in [13], the distance between waypoints should be seen in relation to the requirement of position computation. With longer path segments, the possibility of drift is higher. For this reason, a max waypoint distance is set. A comparison between the original path and the modified path can be seen in Fig. 4.

*1) Docking Behavior:* As the intended docking station for the mission is a platform docking station, a simple docking behavior is created. A pre-defined path, which can be seen in Fig. 5, is calculated from the size and orientation of the docking station. The path is intended to guide the UUV from the endpoint of the homing path to a waypoint with a fixed distance from the entrance of the dock, then to the next waypoint above the intended dock position, before descending down to land on it. The exact docking mechanism has not been addressed in this paper.

### *D. Integration of VSLAM-based motion estimation*

The flow diagram of the VSLAM system is presented in Fig. 6. The VSLAM system for motion estimations is programmed with C++ using open-source libraries OpenCV and g2o. Images from underwater environments often have different contrast in different image parts due to poor lighting conditions. It is hard to detect features in the low contrast part, so image enhancement is necessary. Every time a new frame is sent to the system, contrast limited adaptive histogram equalization (CLAHE) [14] is implemented to enhance the contrast of stereo images. It works by mapping the histogram of the image to another histogram with a wider distribution of intensity values, so the intensity values are spread over the whole range.

ORB algorithm [15] is implemented to detect and describe features. Then feature matching is performed by using

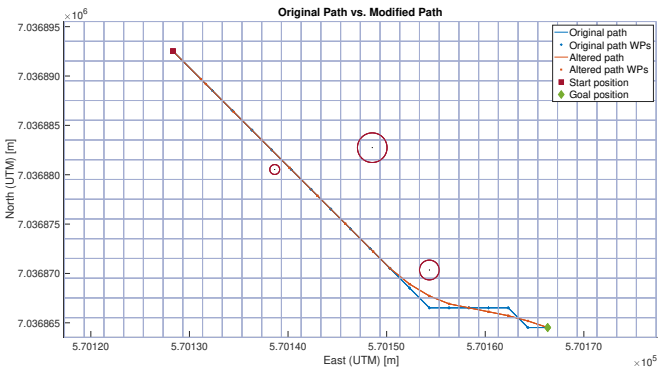



Fig. 4. Comparison between original path and improved path

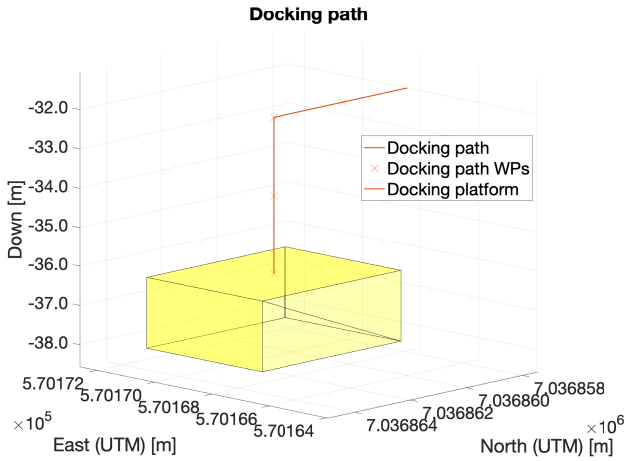



Fig. 5. Docking path

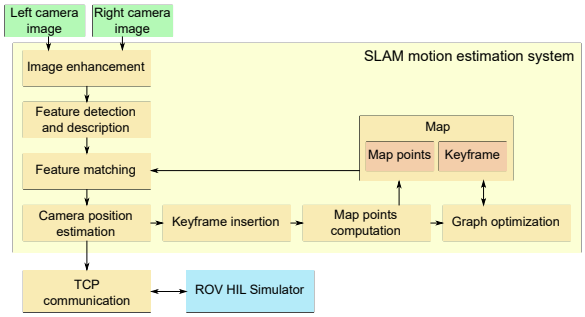



Fig. 6. Flow chart for VSLAM-based motion estimation

Hamming distance for ORB descriptors to find projection relationships between 2D pixel features and 3D map points from the VSLAM map. Generally, the matching results are not good enough to directly estimate the camera position because mismatching is hard to avoid. Distance filter and the RANSAC algorithm [16] are used together to remove mismatches. Next, the camera position is estimated by using the EPnP algorithm [17] based on the matched 3D-2D points in feature matching.

The map of the VSLAM system stores all map points and keyframes. It is initialized when the first stereo image frame is processed. Then the map is updated when new keyframes are added. A keyframe is chosen if the number of matched features in the current frame is small, and new map points are required. Features on the left and the right camera images are matched to compute new map points. Graph optimization [18] is implemented to minimize the projection error between observed and predicted image pixel locations to find the best map point positions and keyframe camera positions.

The local positions of the camera are converted to the local positions of the ROV based on the camera coordinates in the ROV body frame. The ROV positions are transferred to the ROV mission planner using TCP communication. The

system obtains the initial position of the ROV from the mission planner when it starts running so that the local position can be converted to a global position.

### E. Integration of VSLAM-based Obstacle Detection

The real-time VSLAM-based Obstacle Detection revolves around utilizing the outputs of the renowned Visual VSLAM method ORB-SLAM2 [19]. The outputs of ORB-SLAM2 are the estimations of the ROV pose and point clouds of the surrounding environment of the ROV. From these two outputs, the closest detected obstacle is inferred. The obstacle detection system is implemented in C++ using the framework Robot Operating System (ROS). It is comprised of: a camera driver for running the stereo camera rig of the ROV Minerva, an image processing part using the open-source library OpenCV, an existing ROS implementation of ORB-SLAM2[1], a point cloud processing part using the open-source library PCL and a communication part communicating with the ROV Autonomy Framework providing the closest detected obstacle. The communication is conducted using TCP connection. The system architecture is displayed in Fig. 7.

The stereo camera rig of Minerva consists of two Allied Vision Prosilica GC1380C mounted horizontally displaced. By considering the overlapping field of view and expected disparity values, the horizontal displacement, or baseline, is set to 0.2 meters. The cameras are configured in binned mode, reducing the resolution, but increasing the light sensitivity and signal to noise ratio. The image processing undistorts and rectifies the stereo image pairs based upon obtained underwater calibration parameters, and additionally, contrast enhances the images using CLAHE. The point cloud processing first removes point cloud points associated with the seabed by fitting a plane using RANSAC; the remaining points are clustered using the Euclidean based clustering method of [20]. The obstacle sizes $\mathbf{o}_d$ and position $\mathbf{o}_p$ are inferred using

$$\begin{aligned} \mathbf{o}_d &= \mathbf{c}_{max} - \mathbf{c}_{min} \\ \mathbf{o}_p &= \frac{\mathbf{c}_{max} + \mathbf{c}_{min}}{2} \end{aligned} \tag{3}$$

where $\mathbf{c}_{max}$ and $\mathbf{c}_{min}$ are the maximum and minimum point position of the cluster. The closest detected obstacle is determined by finding the obstacle with the shortest Euclidean distance to the current estimated ROV pose. The LabView communication handles the TCP connection with the Autonomy Framework and generates messages containing information about the closest detected obstacle on the format in (4) if the distance threshold of five meters is violated.

$$\left[ \; x_o^g, \quad y_o^g, \quad z_o^g, \quad d_o \; \right] \tag{4}$$

The message is an array of doubles containing the obstacle position in the NED frame $x_o^g$, $y_o^g$ and $z_o^g$, and the obstacle spherical diameter $d_o$. The spherical diameter is determined by selecting the largest estimated dimension of $\mathbf{o}_d$.

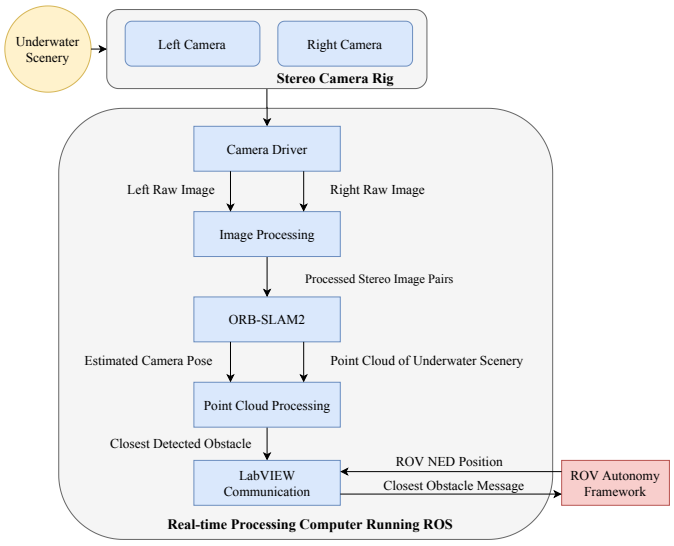



Fig. 7. System architecture of the vision based obstacle detection system

[1]http://wiki.ros.org/orb slam2 ros

## III. Simulation Results

### A. VSLAM-based motion estimation

The VSLAM system performance is tested on a seabed image set from a mission at Stokkbergneset in February 2017 by NTNU ROV SF-30k. For the simulation, 60 paired left and right camera images with a time interval of 2 seconds are used. The result presented in Fig. 8 is simulated on this image set and the corresponding navigation data from the ROV control system.

The result shows that the VSLAM motion estimation successfully estimates the position of ROV in this simulation. The estimated orientation of ROV movement is slightly different from the true orientation, which causes errors in both north and east positions.

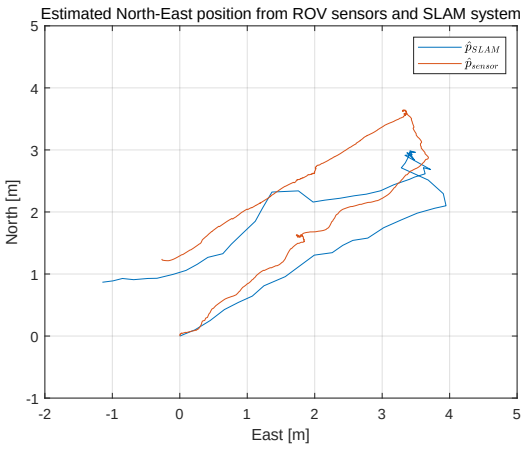



Fig. 8. Simulation of VSLAM-based motion estimation

### B. VSLAM-based obstacle avoidance

The real-time VSLAM-based obstacle detection system is tested at the Marine Cybernetics Lab (MC-Lab) at Marin teknisk senter, NTNU. The stereo camera rig is mounted to a rod and moved through a measured underwater obstacle course installed in the basin of MC-Lab. Subsea light conditions are simulated by having no ambient light and an artificial light source installed on the rod. The results are presented in Fig. 9 where the measured obstacle course is plotted together with the estimated trajectory (OS2), estimated point cloud and the currently estimated closest obstacle outputted if the distance to the previously detected closest obstacle is larger than 0.15 meters. The obstacle course consisted of two boxes, a stepladder and rod.

Fig. 10 presents position estimates and desired positions of the vehicle in the North-East (NE) plane, giving satisfactory simulation results of reactive obstacle avoidance. A VSLAM-based obstacle detection code is programmed to test the distance between the vehicle's current position and obstacles. When the distance becomes shorter than five meters, the vehicle will perform reactive obstacle avoidance. Table III-B shows the location for three detected obstacles. Under some circumstances, more than one obstacle might be encountered at a time. The system chooses the nearest one as the current obstacle and executes the obstacle avoidance correspondingly.

The vehicle is performing the *Transit* action when the first obstacle is detected. The vehicle firstly moves to the desired position and then detects the *Obstacle 01*, performing obstacle avoidance correspondingly. Before reaching the waypoint for avoiding *Obstacle 01*, the vehicle detects a new *Obstacle 02* and generates a new waypoint to avoid it. The third obstacle is detected after reaching the waypoint for avoiding *Obstacle 02*. The test result shows that the obstacle avoidance algorithm successfully guides the vehicle to avoid collisions during mission execution. The mission planning and management system can encounter more than one obstacle at a time and update its waypoints based on the position of the closest obstacle. The system is also able to tackle obstacle avoidance when performing other actions. Reactive obstacle avoidance is also tested in Section III-C.

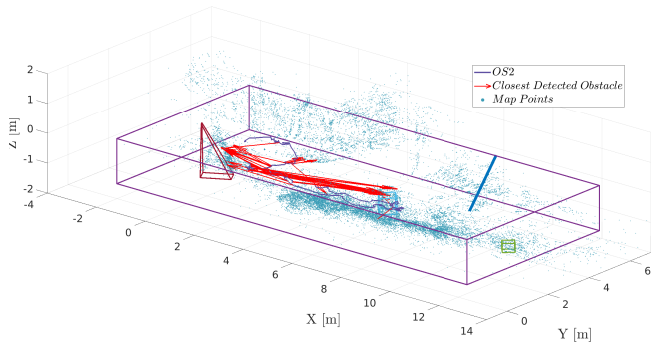



Fig. 9. Laboratory experiment of VSLAM-based obstacle detection.

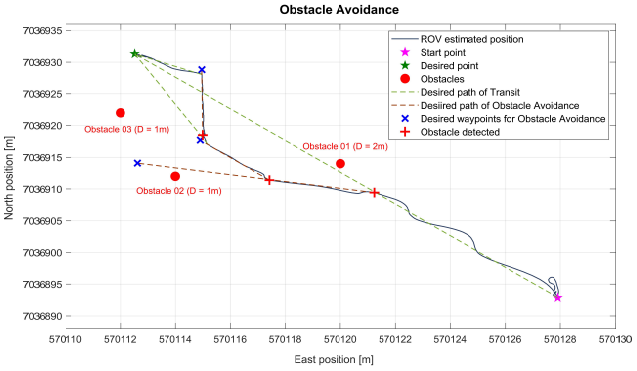



Fig. 10. Simulation of obstacle avoidance

TABLE I
OBSTACLES IN SIMULATION OF OBSTACLE AVOIDANCE

| Obstacle | North | East | Depth | Diameter | [-] |
|---|---|---|---|---|---|
| 01 | 7036914 | 570120 | 15 | 2 | m |
| 02 | 7036912 | 570114 | 15 | 1 | m |
| 03 | 7036922 | 570112 | 15 | 1 | m |

### C. Autonomous mission planning and management

*1) Mapping, charging and OA: Obstacle Avoidance* and *Charging* are tested during *Mapping* as reactive actions. The positions of the two manually set obstacles are listed in Table III-C1. The testing is carried out with hardware-in-the-loop (HIL) simulation. Fig. 11 shows that there are a lot of fluctuations when the vehicle changes its heading.

The performance of waypoint tracking is mainly determined by the low-level control system.

In the mission, the vehicle starts at (7036892, 570128, 10) in UTM coordinates and then moves towards waypoints generated from the mission planner sequentially. The vehicle firstly performs *Launch* and *Descent* to the desired depth of operation. Then, the vehicle transits to the starting position of mapping and performs mapping, tracking six waypoints sequentially, found in Table III-C1. The desired trajectory of mapping is defined before starting the mission, while the path planner plans the waypoints of charging for homing once the *Charge* action is activated. During mapping, 'low battery' is manually set, indicating that the vehicle has to abort mission and move to the docking station to charge. The charging station is set at (7036882, 570140, 27).

TABLE II
OBSTACLES IN SIMULATION OF MAPPING, CHARGING AND OA

| Obstacle | North | East | Depth | Diameter | [-] |
|---|---|---|---|---|---|
| 01 | 7036900 | 570144 | 15 | 2 | m |
| 02 | 7036886 | 570125 | 15 | 2 | m |

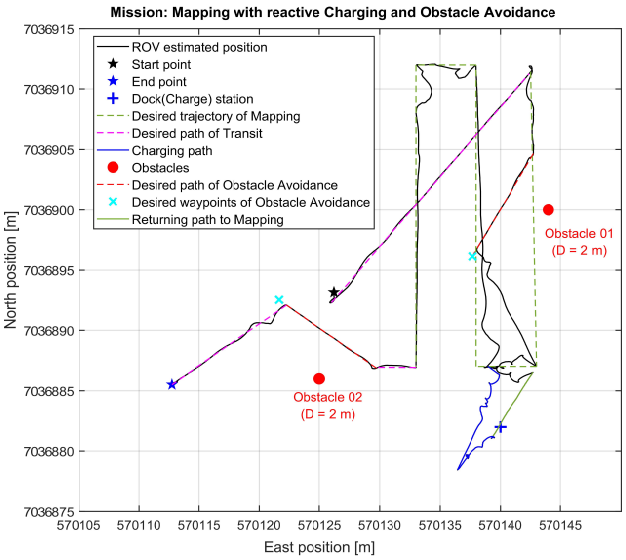



Fig. 11. Simulation of mapping, charging and OA

TABLE III
WAYPOINTS OF MAPPING

| Waypoint | North Position | East Position | Depth | [-] |
|---|---|---|---|---|
| 01 | 7036912 | 570143 | 15 | m |
| 02 | 7036887 | 570143 | 15 | m |
| 03 | 7036887 | 570138 | 15 | m |
| 04 | 7036912 | 570138 | 15 | m |
| 05 | 7036912 | 570133 | 15 | m |
| 06 | 7036887 | 570133 | 15 | m |

*2) Full mission with OA:* A similar mission as above, now with all four global states and three local states executed, was simulated. The resulting plots can be seen in 2D in Fig. 12 and in 3D in Fig. 13. Four "unknown" obstacles were detected by the VSLAM obstacle avoidance. The positions of theses obstacles are presented in Table III-C2. The action sequence of the full mission is *Launch-Descent-Transit-Mapping (OA and Charging)-Sampling (OA)-Transit (OA)-Descent*.

This simulation includes testing of obstacle avoidance for actions *Transit*, *Mapping* and *Sampling*. For *Transit* and

TABLE IV
OBSTACLES IN FULL MISSION

| Obstacle | North Position [m] | East Position [m] | Depth [m] | Diameter [m] |
|---|---|---|---|---|
| 01 | 7036900 | 570144 | 15 | 2 |
| 02 | 7036907 | 570122 | 16 | 1 |
| 03 | 7036900 | 570112 | 15 | 1 |
| 04 | 7036893 | 570105 | 15 | 2 |

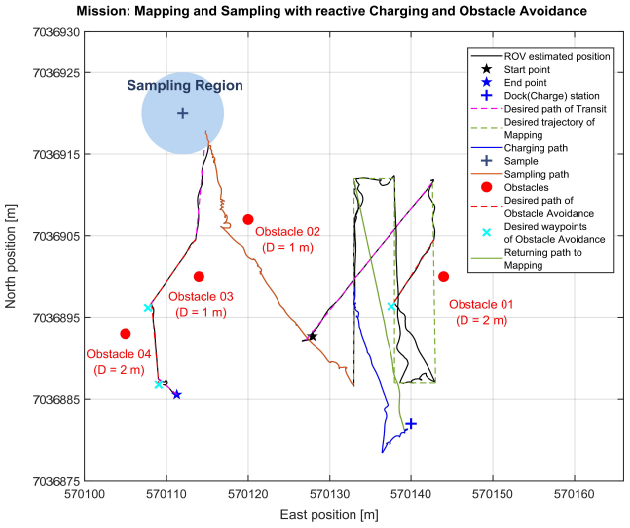



Fig. 12. 2D simulation of mission employing all states

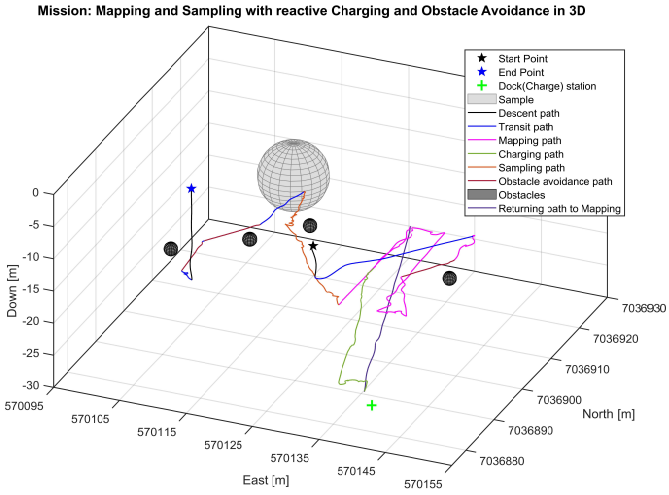



Fig. 13. 3D simulation of mission employing all states

*Mapping*, the reactive behavior is turning its heading and move to a temporary waypoint for obstacle avoidance. During *Sampling*, obstacle avoidance is implemented by adding the newly detected obstacle in the obstacle list and replan to generate a new path with the A* algorithm which avoids the new obstacle.

## IV. CONCLUSION

We have successfully implemented the autonomous mission planning and management system in the ROV control system, integrating vision-based situation awareness for motion estimation and obstacle detection. The system uses mission planning, guiding the vehicle to achieve the mission requests, and path planning is applied for local operations.

The use of the A* algorithm in an autonomous mission planning and management proved to be a good choice, as

it was convenient to implement it in the already existing framework. Adapting it to react to newly detected obstacles was also done in a successful manner, creating a robust path planner for the intended short range missions.

The VSLAM-based obstacle detection system performs well being capable of providing the currently closest obstacle in its locally estimated surroundings. However, as the laboratory experiments results imply, the removal of every seabed associated map point is not successful as some of the closest detected obstacles are located at the basin floor, and the positional consistency of the newly and previously mapped obstacles degrades over the running time due to accumulated drift.

The VSLAM-based motion estimation presents good simulation results, using images derived from previous sea-trials. The testing result is valid in short periods. The deviation increases with the testing process. However, real-time VSLAM-based motion estimation has not be tested yet.

Since conducting the joint missions for this paper, both the docking behavior and A* path planning behaviors have been altered. A safety distance around obstacles have been added in the A* algorithm, and an ultra-short-baseline transducer has been simulated to situate at the dock to ensure more reliable position measurements when docking.

### A. Further work

To verify the capability and performance of the autonomous mission planning and management system, sea-trails should be conducted. During simulations, sensor noise and environmental forces (such as current and waves) are not simulated. Also, the simulations use depth control during the execution of the mission. For further improvement, a combination of depth and altitude control should be designed such that the ROV could execute actions more accurately also relying on measurements from a doppler velocity log.

Up to now, the mission planning and management system is only capable of performing observations. Sampling and docking behaviors are simplified as trajectory tracking generated by path planning. More refinement should be designed for the actual realization of these actions.

The path planning algorithm is developed in a 2D plane while the ROV moves in a 3D underwater environment. The path planning in 3D can be further optimised including a full 3D model to the path generation procedure.

The VSLAM system for motion estimation is tested in a short time simulation. Loop closure techniques are expected to be added in the VSLAM system to reduce estimation error for long time position estimation.

## REFERENCES


[1] I. Rist-Christensen, "Autonomous robotic intervention using rov," Master's thesis, NTNU, 2016.

[2] J. Hegde, I. B. Utne, and I. Schjølberg, "Applicability of current remotely operated vehicle standards and guidelines to autonomous subsea imr operations," in *ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering*. American Society of Mechanical Engineers Digital Collection, 2015.

[3] C. Operations, N. Board, D. Sciences, and N. Council, *Autonomous vehicles in support of naval operations*, 09 2005.

[4] T. O. Fossum, M. Ludvigsen, S. M. Nornes, I. Rist-Christensen, and L. Brusletto, "Autonomous robotic intervention using rov: An experimental approach," in *OCEANS 2016 MTS/IEEE Monterey*. IEEE, 2016, pp. 1–6.

[5] J.-H. Li, M.-J. Lee, S.-H. Park, and J.-G. Kim, "Real time path planning for a class of torpedo-type AUVs in unknown environment," in *2012 IEEE/OES Autonomous Underwater Vehicles (AUV)*, Sep. 2012, pp. 1–6, iSSN: 2377-6536.

[6] D. Li, P. Wang, and L. Du, "Path Planning Technologies for Autonomous Underwater Vehicles-A Review," *IEEE Access*, vol. 7, pp. 9745–9768, 2019, conference Name: IEEE Access.

[7] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous robots*, vol. 4, no. 4, pp. 333–349, 1997.

[8] A. Sørensen, F. Dukan, M. Ludvigsen, D. Fernandes, and M. Candeloro, *Development of dynamic positioning and tracking system for the ROV Minerva*, 01 2012, pp. 113–128.

[9] J. Hoffmann and B. Nebel, "The ff planning system: Fast plan generation through heuristic search," *Journal of Artificial Intelligence Research*, vol. 14, pp. 253–302, 2001.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, pp. 100–107, 1968.

[11] A. Koubâa, H. Bennaceur, I. Chaari, S. Trigui, A. Ammar, M.-F. Sriti, M. Alajlan, O. Cheikhrouhou, and Y. Javed, *Robot Path Planning and Cooperation*. Springer, 2018, vol. 772.

[12] F. Dukan, "ROV motion control systems," Ph.D. dissertation, Norwegian University of Science and Technology, Faculty of Engineering Science and Technology, Department of Marine Technology, Trondheim, 2014.

[13] J. Yuh, T. Ura, and G. A. Bekey, *Underwater Robots*. Springer Science & Business Media, Dec. 2012, google-Books-ID: YwfaBwAAQBAJ.

[14] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, "Adaptive histogram equalization and its variations," *Computer vision, graphics, and image processing*, vol. 39, no. 3, pp. 355–368, 1987.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[16] K. G. Derpanis, "Overview of the ransac algorithm," *Image Rochester NY*, vol. 4, no. 1, pp. 2–3, 2010.

[17] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[18] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.

[19] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-d cameras," vol. 33, no. 5, pp. 1255–1262.

[20] R. B. Rusu, *Clustering and Segmentation*. Springer Berlin Heidelberg, vol. 85, pp. 75–85, series Title: Springer Tracts in Advanced Robotics.

# Appendix B

## *FieldOfViewCalculations.m*

```
%% OVERLAPPING FIELD OF VIEW IN WATER CALCULATIONS:
% Author: Erlend R ilid Vollan
% Last editet 25.04.2020

% Assumptions:
% * Stereo geometry is assumed. The image planes are aligned
% * The field of view is symmetric along the principle axis

% b = baseline [m]
% l_tot = distance to calculate field of view overlap [m]

% rw_l = resolution width left camera [px]
% rw_r = resolution widt right camera [px]
% f_l = focal length left camera in air [px]
% f_r = focal length right camera in air [px]
% theta_l = field of view left camera in water [rad]
% theta_r = field of view right camera in water [rad]

b = 0.2;
l_tot = 1;

rw_l = 1360;
rw_r = 1360;
f_l = 1109.1;
f_r = 1112.3;

%% Determine field of view each camera in air
% theta_l_a = field of view left camera in air [rad]
% theta_r_a = field of view right camera in air [rad]

theta_l_a = 2*atan((rw_l/2)/f_l);
theta_r_a = 2*atan((rw_r/2)/f_r);

%% Correct the field of view due to refraction between air and water
% n_a = refractive index in air [-]
% n_w = refractive index in water [-]
n_a = 1;
n_w = 1.33;
theta_l = 2*asin((n_a/n_w)*sin(theta_l_a/2));
theta_r = 2*asin((n_a/n_w)*sin(theta_r_a/2));
f_l_corr = (rw_l/2)/(tan(theta_l/2));
```

```
42 f_r_corr = (rw_r/2)/(tan(theta_r/2));
43
44 %% Calculate minimum distance for given baseline
45 % l_min = minimum where field of view overlap happens [m]
46 syms b_l_s b_r_s l_min_s
47
48 E1 = b_l_s + b_r_s == b;
49 E2 = b_l_s/(tan(theta_l/2)) == l_min_s;
50 E3 = b_r_s/(tan(theta_r/2)) == l_min_s;
51
52 solx = solve(E1,E2,E3,b_l_s,b_r_s,l_min_s);
53
54 l_min = solx.l_min_s;
55
56 %% Calculate overlapping field of view for given distance
57 % l_ofov = length where the field of views are overlapping
58 % ofov_l = overlapping field of view from left camera
59 % ofoc_r = overlapping field of view from right camera
60
61 l_ofov = l_tot - l_min;
62 ofov_l = tan(theta_l/2)*l_ofov;
63 ofov_r = tan(theta_r/2)*l_ofov;
64
65 OFOV = ofov_l + ofov_r;
66
67 %% Calculate field of view for left and right camera
68 %FOV_l, FOV_r = Field of view for left and right camera
69
70 FOV_l = 2*tan(theta_l/2)*l_tot;
71 FOV_r = 2*tan(theta_r/2)*l_tot;
72
73 %% Calculate percentage of overlapping field view of total
74 %TOFO = total field of view
75 TOFOV = FOV_l + FOV_r - OFOV;
76 percentage_covered = double(OFOV/TOFOV);
77
78 fprintf('Percentage of overlapping FOV of total FOV at %.1f [m] is %.1f ...
       percent \n', l_tot, percentage_covered*100);
79 fprintf('Minimum distance from cameras to scene is %.3f [m] \n', ...
       double(l_min));
```

Erlend Røilid Vollan

# Visual Simultaneous Localization and Mapping Applied on Work Class ROVs

NTNU
Norwegian University of
Science and Technology