



ESCOLA NAVAL

talant de biefaire



Vicente José da Conceição Galvão Teles Viegas

Stereo Vision Simultaneous Localization and Mapping for Uncrewed Surface Vessels in Uncharted Ports

A dissertation submitted in partial fulfillment of
the requirements for the degree of Naval Military Sciences,
specialisation of Marinha



Alfeite

2025



ESCOLA NAVAL

talento e brio



Vicente José da Conceição Galvão Teles Viegas

*Stereo Vision Simultaneous Localization and Mapping for
Uncrewed Surface Vessels in Uncharted Ports*

A dissertation submitted in partial fulfillment of
the requirements for the degree of Naval Military Sciences,
specialisation of Marinha

Orientation of: Victor José de Almeida e Sousa Lobo

Co-orientation of: Nuno Alexandre Antunes Martins Pessanha Santos

The Master Student,

The Supervisor,

Assinado por: **Vicente José da Conceição Galvão
Teles Viegas**



Vicente Viegas

Assinado por: **VICTOR JOSÉ DE ALMEIDA E SOUSA
LOBO**
Num. de Identificação: 09302569
Data: 2025.09.03 13:56:19+01'00'



Victor Lobo

Alfeite
2025

“If one does not know to which port one is sailing, no wind is favorable.”
- Seneca

Dedicated to the memory of my grandfather, João Pedro Rodrigues da Conceição.

Acknowledgement

The conclusion of this thesis would not have been possible without the support and encouragement of many people who accompanied me during my academic journey at the Portuguese Naval Academy. To all of them, I extend my sincere appreciation.

I am especially grateful to my supervisors, Professor Victor Lobo, and Dr. Nuno Pessanha Santos, whose availability, guidance, expertise, and commitment have been invaluable. Their advice and trust provided both direction and confidence throughout this work.

To my family and friends, I owe profound gratitude for their unwavering support and understanding.

To my classmates of the class *Chefe de Divisão da Armada António Lopes da Costa e Almeida*, I wish to express my heartfelt thanks for the camaraderie, teamwork, and shared resilience over these five years.

Finally, I would like to acknowledge all those who, as teachers, instructors, or commanding officers, played a role in my preparation to become a navy officer. Their example and guidance have left an enduring mark on both my professional and personal growth.

Abstract

Uncrewed Surface Vessels (USVs) are increasingly regarded as force multipliers in naval operations, capable of extending operational reach while reducing risk to personnel. Their effectiveness, however, depends critically on robust autonomous navigation, particularly in Global Navigation Satellite System (GNSS)-denied or contested environments such as ports in conflict zones. This thesis investigates the application of stereo vision-based Simultaneous Localization and Mapping (SLAM) as a passive, autonomous alternative for USV navigation in uncharted maritime environments.

A methodological framework was developed integrating a high-resolution stereo camera with the Robot Operating System 2 (ROS 2), complemented by custom dataset acquisition at the Lisbon Naval Base using a Rigid Hull Inflatable Boat. Comparative experiments were conducted between two stereo vision approaches: Semi-Global Block Matching (SGBM) and Scale Invariant Feature Transform (SIFT) to evaluate their performance in generating depth maps and 3D point clouds under maritime conditions. Subsequently, SLAM pipelines were implemented using Real-Time Appearance-Based Mapping (RTAB-Map) and the ZED SDK, tested both on the custom dataset and the publicly available Pohang Canal dataset.

Results demonstrate that stereo vision is viable for USV navigation in complex port environments, with each algorithm exhibiting distinct trade-offs. SGBM produced dense reconstructions but was sensitive to reflections and low-texture regions, while SIFT offered robustness to environmental variability at the cost of sparser maps. SLAM evaluations revealed that RTAB-Map achieved competitive localization accuracy when benchmarked against GPS ground truth, while the ZED SDK provided a compact and efficient solution. Together, these findings highlight the operational potential of stereo visual SLAM as a navigation solution for USVs in GNSS-denied scenarios, while also identifying key limitations and directions for future research.

Keywords: Autonomous Navigation, GNSS-Denied, Stereo Vision, Uncrewed Surface Vessels (USV), Visual SLAM

Resumo

Os Veículos de Superfície Não Tripulados (USV) são cada vez mais reconhecidos como multiplicadores de força nas operações navais, capazes de ampliar o alcance operacional e reduzir o risco para o pessoal. A sua eficácia, contudo, depende criticamente de métodos de navegação autónoma robustos, sobretudo em ambientes contestados ou negados ao sistema global de navegação por satélite (*GNSS*), como portos em zonas de conflito. Esta dissertação investiga a aplicação da técnica de *Simultaneous Localization and Mapping (SLAM)* baseada em visão estéreo como alternativa passiva e autónoma para a navegação de *USV* em ambientes marítimos desconhecidos.

Foi desenvolvido um enquadramento metodológico que integrou uma câmara estéreo com o *Robot Operating System 2*, complementado pela aquisição de um conjunto de dados na Base Naval de Lisboa, utilizando uma embarcação semirígida. Realizaram-se experiências comparativas entre dois métodos de visão estéreo: *Semi-Global Block Matching (SGBM)* e *Scale-Invariant Feature Transform (SIFT)*, avaliando o seu desempenho na geração de mapas de disparidade e nuvens de pontos 3D em cenários marítimos. Subsequentemente, *pipelines* de *SLAM* foram implementadas com o *Real-Time Appearance-Based Mapping (RTAB-Map)* e com o *ZED SDK*, testados tanto no *dataset* adquirido na BNL como no *dataset* público *Pohang Canal*.

Os resultados demonstram a viabilidade da visão estéreo para a navegação de *USV* em portos, evidenciando diferentes compromissos de desempenho. O *SGBM* produziu reconstruções densas, mas mostrou-se sensível a reflexos e regiões de baixa textura, enquanto o *SIFT* ofereceu maior robustez a variações ambientais, embora com mapas mais esparsos. As avaliações de *SLAM* revelaram que o *RTAB-Map* alcançou precisão competitiva face ao *GPS* de referência, enquanto o *ZED SDK* constituiu uma solução eficiente e compacta. Os resultados evidenciam o potencial operacional do *SLAM* visual estéreo para navegação em cenários sem disponibilidade de *GNSS*, identificando limitações e futuras linhas de investigação.

Palavras-chave: Autonomous Navigation, GNSS-Denied, Stereo Vision, Uncrewed Surface Vessels (USV), Visual SLAM

Contents

Introduction	1
Problem Formulation	3
Objectives	4
Challenges	5
Published Work	6
Thesis Structure	6
1 Background	9
1.1 The Pinhole Camera Model	9
1.1.1 Coordinate Systems and Homogeneous Coordinates	10
1.1.2 Extrinsic Parameters	10
1.1.3 Intrinsic Parameters	11
1.1.4 Distortion	13
1.1.4.1 Radial distortion	13
1.1.4.2 Tangential distortion	14
1.2 Stereoscopic Vision	15
1.2.0.1 Epipolar Geometry	15
1.3 Simultaneous Localization and Mapping	17
1.4 Performance Evaluation	20
2 State of the Art	23
2.1 Datasets	23
2.2 Stereoscopic Vision	25
2.2.1 Stereo Matching Algorithms	25
2.3 Simultaneous Localization and Mapping	27
3 Methodology	29
3.1 Overview	29
3.1.1 System Setup	31
3.1.1.1 ZED 2i Stereo Camera Specifications	31
3.1.1.2 ZED SDK	32

3.1.1.3	ZED 2i calibration parameters	32
3.1.1.4	ROS 2 integration	33
3.1.1.5	Garmin GPS 18	34
3.2	Dataset Creation	35
3.3	Algorithms	36
3.3.1	Stereo Comparison	37
3.3.1.1	Image Rectification	37
3.3.1.2	SGBM	39
3.3.1.3	SIFT	41
3.3.2	SLAM	43
3.3.2.1	RTAB-Map Integration - BNL	44
3.3.2.2	RTAB-Map Integration - Pohang Canal	47
3.3.2.3	ZED SDK SLAM	49
4	Experimental Results	51
4.1	Stereo images of the Pohang Canal Dataset	51
4.2	Simultaneous Localization and Mapping	53
4.2.1	Pohang Canal Dataset	53
4.2.2	BNL Dataset	56
4.2.2.1	Processing the BNL dataset with RTAB-Map . . .	56
4.2.2.2	Processing the BNL dataset with ZED SDK . . .	61
4.2.2.3	Comparison	64
4.2.3	Discussion	67
	Conclusion	67
	Conclusions	69
	Future Work	70
	Bibliography	73

List of Figures

1	Ukrainian USV lines up an attack on the Russian Navy Ropucha Class landing ship Olenegorsky Gornyyak, night of August 3-4 (Sutton 2025).	2
2	Map of GPS interference hotspots across the Baltic Sea region on 29 of August 2025, illustrating the jamming zones affecting civilian and military navigation (Adapted from Wiseman 2025).	3
1.1	The frontal pinhole camera model illustrating the projection of a 3D point onto a 2D image plane. (Ma et al. 2004).	10
1.2	A rotation and translation from a world coordinate frame W to a camera frame C . (Hartley 2003)	11
1.3	Conversion from the image plane to the pixel array. (OpenCV 2021a)	12
1.4	Left: No Distortion Center: Barrel Distortion Right: Pincushion Distortion. (OpenCV 2021a)	14
1.5	Generic stereo vision setup. (Hariyama et al. 2008)	16
1.6	Example stereo scene (captured by the author in a classroom at the Portuguese Naval Academy).	16
1.7	Epipolar Geometry (adapted from OpenCV 2023).	17
1.8	Stereo disparity map of the example scene presented in Figure 1.6.	17
2.1	Map overview of the Pohang Canal Dataset, illustrating the data collection area and route (Chung et al. 2023).	25
3.1	Stereo comparison diagram.	30
3.2	SLAM in the Pohang Canal diagram.	30
3.3	SLAM in the BNL diagram.	31
3.4	ZED 2i Stereo Camera (StereoLabs 2024).	32
3.5	ZED SDK Functional Diagram. (Stereolabs 2021)	33
3.6	Garmin GPS 18 USB (Garmin 2024).	34
3.7	Cgps client showing GPS data.	34
3.8	Xgps client showing location of visible GPS satellites.	35
3.9	RHIB setup for dataset capture in the maneuvering basin of the Lisbon Naval Base.	35

3.10	Trajectory of the RHIB while recording the dataset. A and B correspond to the two different ROS 2 bags used in the trials.	36
3.11	Comparison between the original and rectified image.	39
3.12	Disparity map produced using the SGBM algorithm.	40
3.13	Filtered disparity map.	40
3.14	Dense scene reconstruction from the computed disparity map in a 3D point cloud.	41
3.15	SIFT keypoints represented with scale and orientation on the left rectified image.	41
3.16	Filtered SIFT keypoint matches.	42
3.17	Filtered SIFT keypoints with calculated distances.	42
3.18	Sparse scene reconstruction from the computed pointcloud.	43
3.19	RTAB-Map system diagram (Adapted from Labbé 2015).	44
3.20	<i>rtabmapnode</i> (Labbé and Michaud 2019).	45
3.21	<i>rgbd_odometry</i> node (Adapted from Labbé and Michaud 2019). . .	46
3.22	RTAB-Map GUI showing SLAM on BNL dataset.	47
3.23	<i>stereo_odometry</i> node (Adapted from Labbé and Michaud 2019). .	48
3.24	RTAB-Map GUI showing SLAM on Pohang Canal Dataset.	48
3.25	RViz2 visualization of ZED SDK SLAM on the BNL dataset.	50
4.1	Comparison of SGBM and SIFT: (a) mean depth, (b) average point cloud size, (c) valid points percentage, and (d) average execution time.	52
4.2	RTAB-Map results on the Pohang Canal dataset (images 1300-2300).	54
4.3	Two images taken from the Pohang Canal dataset compared to two screenshots of the generated pointcloud.	55
4.4	RTAB-Map results on BNL A dataset: GPS vs SLAM trajectory alignment and error metrics.	57
4.5	Overlay of RTAB-Map generated graph on satellite imagery of the mapped vessel.	58
4.6	An image captured in the BNL A dataset compared to a screenshot of the generated pointcloud.	59
4.7	RTAB-Map results on BNL B dataset: GPS vs SLAM trajectory alignment and error metrics.	60
4.8	An image captured in the BNL B dataset compared to a screenshot of the generated point cloud.	61
4.9	ZED SDK SLAM results on BNL A dataset: GPS vs ZED trajectory alignment and error metrics.	62
4.10	ZED SDK SLAM vs GPS 3D error over time.	63

4.11 ZED SDK SLAM results on BNL B dataset: GPS vs ZED trajectory alignment and error metrics.	64
4.12 Comparison of GPS, RTAB-Map, and ZED SDK SLAM trajectories on BNL A dataset.	65
4.13 Comparison of GPS, RTAB-Map, and ZED SDK SLAM trajectories on BNL B dataset.	66

List of Tables

1.1	SLAM Classification Variables and Typical Options.	19
3.1	Key Specifications of the ZED 2i Stereo Camera (Adapted from Stereolabs 2021).	32
3.2	Calibration parameters of the ZED 2i at 2K resolution (2208×1242), extracted from the camera’s factory calibration file.	33
3.3	Intrinsic Parameters of the Left and Right Pohang Stereo Cameras (Adapted from Chung et al. 2023).	37
3.4	Extrinsic Parameters of the Pohang Stereo Cameras (Adapted from Chung et al. 2023).	38
4.1	RTAB-Map vs. fused baseline on Pohang Canal.	54
4.2	Comparison of localization error metrics for RTAB-Map and ZED SDK on BNL A dataset.	65
4.3	Comparison of localization error metrics for RTAB-Map and ZED SDK on BNL B dataset.	66

List of Abbreviations

AHRS	Attitude and heading reference system
API	Application Programming Interface
ATE	Absolute Trajectory Error
BNL	Base Naval de Lisboa
C2	Command and Control
CMOS	Complementary Metal-Oxide-Semiconductor
CPU	Central Processing Unit
DGPS	Differential Global Positioning System
EDA	European Defense Agency
EKF	Extended Kalman filter
ENU	East North Up
EW	Electronic Warfare
FLANN	Fast Library for Approximate Nearest Neighbors
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
GT	Ground Truth
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurement Unit
IUM	Instituto Universitário Militar
KD	K-dimensional
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute
LiDAR	Light Detection and Ranging
LTM	Long-Term Memory
LTS	Long Term Support
OES	Oceanic Engineering Society
RAM	Random-Access Memory
RGB	Red Green Blue
RGB-D	Red Green Blue - Depth
RHIB	Rigid Hull Inflatable Boat

RMSE	Root Mean Squared Error
ROI	Region of Interest
ROS	Robot Operating System
RTAB-Map	Real-Time Appearance-Based Mapping
RTK	Real Time Kinematics
SAD	Sum of Absolute Differences
SAR	Search and Rescue
SBM	Stereo Block Matching
SDK	Software Development Kit
SGBM	Semi-Global Block Matching
SGM	Semi-global matching
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SSD	Sum of Squared Differences
SVD	Singular Value Decomposition
USB	Universal Serial Bus
USV	Uncrewed Surface Vessel
VAIS	Visual Aerial Imaging for Surveillance
VGA	Video Graphics Array
WGS84	World Geodetic System 1984
WLS	Weighted Least Squares
WM	Working Memory
WTA	Winner-Takes-All

Introduction

Autonomous systems are steering the evolution of both civilian and military maritime operations. Enabled by advances in robotics, artificial intelligence, and real-time sensing, Uncrewed Surface Vessels (USVs) now execute missions across the operational spectrum, from surveillance and reconnaissance to mine counter-measures and precision strikes. Within the naval domain, USVs emerge as force multipliers: they extend operational reach, minimize risks to personnel, and deliver cost-effective responses to asymmetric threats.

Yet, the maritime environment imposes unique obstacles. Unlike ground or aerial systems, USVs must navigate constantly shifting dynamics such as waves, currents, reflections, and occlusions, often in Global Navigation Satellite System (GNSS) degraded or denied domains such as contested coastal and port areas. Robust perception and localization thus become indispensable for dependable autonomous navigation.

Problem Motivation

Uncrewed Surface Vessels have become essential tools in modern naval operations, gaining increased attention due to their tactical use as "kamikaze drones" during the conflict in Ukraine as a means of sea denial against a stronger traditional navy (Pettyjohn 2024). USVs provide a cost-effective alternative to traditional weapons systems, enabling precise attacks while minimizing human risk. For example, in the Ukrainian conflict, USVs have been deployed to establish Sea Denial capabilities by targeting Russian warships in supposedly secure ports. The Russian Federation faced significant challenges in countering these inexpensive and hard-to-detect platforms (Sutton 2024). These vehicles are capable of striking targets theoretically shielded by the port environment, all while avoiding direct human casualties on the attacker's side.

In light of this evolving threat, enhancing autonomous navigation capabilities for USVs operating within hostile or entirely unknown port environments is crucial. Figure 1 illustrates a frame captured during an actual operation where a

Ukrainian USV attacked a Russian Federation warship. According to reports the attack inflicted major damage to the ship that was then reported as combat ineffective (Balmforth 2023).



FIGURE 1: Ukrainian USV lines up an attack on the Russian Navy Ropucha Class landing ship Olenegorsky Gornyyak, night of August 3-4 (Sutton 2025).

One of the primary constraints in such scenarios is the inability to maintain continuous communication with the USV during the mission. Remote Command and Control (C2) cannot be relied upon to adapt the mission in real time, due to two critical factors: (i) the risk of detection by the enemy, and (ii) the enemy's capacity to disrupt communications via Electronic Warfare (EW) techniques. This reality has been starkly illustrated in Ukraine, where widespread Russian jamming has rendered traditional radio frequency links unreliable, forcing both sides to experiment with alternatives such as fiber-optic tethered aerial drones that bypass the electromagnetic spectrum entirely (BBC News 2024). Additionally, Global Navigation Satellite System (GNSS) signals are highly vulnerable to jamming and spoofing, as demonstrated in the conflict (Chiu and Stewart 2022), which severely degrades the reliability of navigation systems dependent on satellite positioning.

In addition to the reported loss of the effectiveness of guided weapons using Global Positioning System (GPS) in the Ukraine conflict (Schmitt and Santora 2024), GNSS signals are often degraded in strategically important areas such as the Baltic Sea or over Kaliningrad (Turgeon 2025), as illustrated in Figure 2.

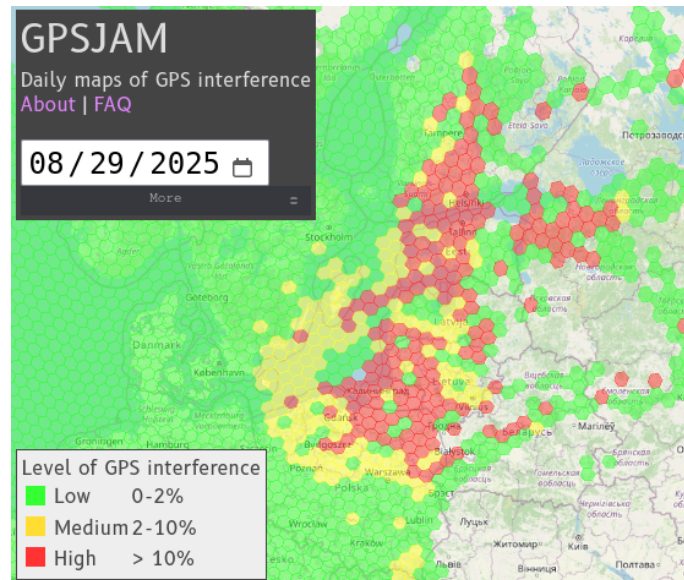


FIGURE 2: Map of GPS interference hotspots across the Baltic Sea region on 29 of August 2025, illustrating the jamming zones affecting civilian and military navigation (Adapted from Wiseman 2025).

Together, these developments expose a pressing need: USVs must be equipped with autonomous, GNSS-independent, passive navigation systems to operate effectively in contested or denied environments.

Problem Formulation

In order to conduct research in support of this type of naval operations, focused on the use of autonomous uncrewed surface vessels, it is necessary to define and limit the area of research.

Port layouts can be highly dynamic, changing with the positioning of vessels and other structures, or even be completely unknown, as is the case with submarine pens. Without timely and accurate updates, large portions of a port can be effectively considered uncharted territory. Lastly, the use of active sensors like navigation radars or Light Detection and Ranging (LiDAR) can increase the likelihood of detection and mission failure, as these emit signals that can be intercepted or jammed by the enemy’s EW capabilities.

These challenges underscore the necessity for USVs to operate autonomously with robust perception and navigation systems that do not rely on external communication or active sensing, especially within hostile, cluttered, and unknown port environments.

With no outside communication or access to GNSS, and no prior knowledge of the target port, a USV will be faced with two problems upon arrival:

- **Localization:** Knowing its position.
- **Mapping:** Knowing its surroundings.

This challenge is known as the Simultaneous Localization and Mapping (SLAM) problem, whose solution is to build a map of the environment while simultaneously estimating the vehicle's location within it (Durrant-Whyte and Bailey 2006).

SLAM techniques can be used with many different types of sensors, the most used are usually LiDAR or conventional cameras (Cadena et al. 2017). Among the available sensing modalities, stereo visual SLAM is particularly attractive for naval operations. Its passive nature avoids the detectability risks of active sensors, it requires relatively low power, and it preserves visual information that can further support recognition, identification, and targeting tasks.

For this thesis the focus will be on solving the SLAM problem once inside an uncharted port by using a stereo camera.

Objectives

This thesis aims to contribute to the development of robust autonomous navigation methods for USVs operating in unknown or hostile port environments. Specifically, it focuses on improving visual perception and localization using stereo vision techniques that do not rely on GNSS or active sensors.

The proposed goal is to generate a 3D map of an initially unknown port and the present ships, as well as determine the USVs position inside that port at all times, without external communication or GPS signals.

This thesis also contributes directly to achieve the goals of the SABUVIS II research project, financed by the European Defense Agency (EDA), and the results are part of the deliverables and final demonstration of the project.

Challenges

This thesis tackles the practical difficulties of deploying visual SLAM for uncrewed surface vessels in real port environments. Unlike ground robotics, maritime platforms introduce unique constraints that complicate both sensing and computation.

A first challenge lies in stereo vision itself. Water surfaces produce reflections, specular highlights, and regions of low texture that lead to poor feature extraction and mismatched disparity estimates. Stereo matching algorithms, which rely on consistent textures between image pairs, often fail under these conditions, resulting in noisy depth maps and sparse reconstructions. Furthermore, the relatively small baseline of compact stereo cameras limits depth accuracy at longer ranges, exacerbating the difficulty of reliably reconstructing distant structures in maritime environments.

Second, the motion of the vessel introduces its own complications. A USV is never stationary but constantly subject to pitch, roll, and heave induced by usual port dynamic elements such as waves and wakes. These movements cause rapid changes in viewpoint and motion blur, stressing visual odometry and increasing the risk of tracking failures.

Third, hardware constraints must be considered. Onboard computers available to small USVs have limited processing power and energy budgets compared to laboratory setups. SLAM must run in real time, balancing the need for accurate feature tracking, loop-closure detection, and map generation with the realities of embedded GPU/CPU performance and power consumption.

Finally, validation is hampered by the scarcity of relevant datasets. While many established SLAM benchmarks exist for ground and aerial robotics, few datasets capture the particular challenges of maritime navigation: moving water, reflections, and dynamic port environments. This lack of prior data requires both the adaptation of algorithms originally designed for land-based robots and the creation of new evaluation scenarios more representative of naval operations.

Another significant challenge encountered in this work was the limited availability of key experimental resources, namely the stereo camera and the USV platform. As the stereo camera only became accessible at a later stage of the project, the initial phase relied on publicly available datasets to investigate stereo matching and SLAM algorithms. This preliminary effort provided a methodological baseline that

later informed the application of SLAM to our own experimental setup. Similarly, the absence of a dedicated USV¹ was addressed by recording a dataset designed to emulate USV operation under the required conditions.

Published Work

During the development of this thesis, two related scientific papers were produced and accepted for presentation.

- “*Autonomous Mapping of Uncharted Ports with Stereo Vision*” - Accepted at the **VI Encontro de Investigação e Desenvolvimento em Ciências Militares** promoted by Instituto Universitário Militar (IUM) that took place at Escola Naval in November of 2024. This work introduced the operational relevance of SLAM for maritime USVs and an initial exploration of the possible solutions, leading to a stereo-based approach. The contribution was presented in poster format, since the author was deployed on a Search and Rescue (SAR) mission in the Azores at the time.
- “*Autonomous Mapping of Uncharted Ports: Comparison of SIFT and Semi-Global Block Matching Based Stereo Vision on Real Port Data*” - Accepted for presentation at **OCEANS 2025 - Great Lakes** (IEEE/OES, Chicago, September-October 2025). This paper extends the research with a comparative evaluation of two stereo vision approaches applied to a real maritime dataset, highlighting their respective trade-offs and making the first step towards implementing SLAM in uncharted ports.

Together, these publications illustrate the progressive development of the work, from preliminary results to peer-reviewed international dissemination.

Thesis Structure

The remainder of this document is organized as follows:

- **Chapter 1** introduces the theoretical background relevant to this work, including the principles of stereo vision, disparity computation, and Simultaneous Localization and Mapping (SLAM).

¹The USV platforms planned for the SABUVIS II project ended up not being built at the Naval Academy, and were available far too late to be used in the development of this thesis. However, for the final SABUVIS II trials, the system developed in this thesis was mounted on a SABUVIS II USV, although not integrated into its navigation and control system.

-
- **Chapter 2** reviews the state of the art, presenting related research in visual SLAM, stereo vision, and maritime robotics, as well as available datasets and identifying the main gaps that motivate this thesis.
 - **Chapter 3** describes the adopted methodology, detailing the stereo vision algorithms, SLAM frameworks, experimental setup, and datasets employed in this study.
 - **Chapter 4** presents the experimental results, first for stereo matching methods and then for SLAM performance on both public and custom datasets.
 - **Conclusion** - concludes the thesis by summarizing the main contributions, discussing the operational implications of the findings, and outlining promising directions for future work.

Chapter 1

Background

This chapter presents the theoretical foundations necessary to support the development of a stereo-based SLAM solution for autonomous navigation of Uncrewed Surface Vessels. Given the multidisciplinary nature of the problem, the chapter systematizes the main concepts from the fields of computer vision and robotics, with particular emphasis on camera models, stereo geometry, and mapping techniques. These foundations provide the basis for the methodologies and experimental work described in the following chapters.

1.1 The Pinhole Camera Model

A camera model provides a mathematical framework for projecting 3D points onto a 2D image plane (Poling 2015). In contrast, we can state that computer vision focuses on the inverse challenge: interpreting 2D image data to infer the structure and content of the original scene. To effectively approach this problem, it is essential to first understand the mechanisms behind image formation and what information the image conveys about the physical environment.

The pinhole camera model serves as a fundamental concept in computer vision, particularly in stereo vision applications such as those explored in this thesis. It simplifies imaging by assuming that light rays pass through a single point (the pinhole) before forming an image on a 2D plane. This simplifies the process of image formation to that of tracing rays originating from reflective surfaces in space to points in the image plane.

As illustrated in Figure 1.1, this idealized camera consists of a camera center o , which acts as the origin of a 3D coordinate system, through which all light rays pass, and an image plane where the image is formed. For the purposes of this analysis, we assume that the image plane is positioned parallel to the xy -plane of

the 3D coordinate system, located at $z = f$, where f is the focal length. In order to avoid having an inverted image, the image plane is placed in front of the camera center - this is the frontal pinhole model 1.1. We define a 2D coordinate system on this image plane with its origin at $(0, 0, f)^\top$ in 3D Euclidean coordinates. The x - and y -axes of this 2D frame are aligned parallel to the X - and Y -axes of the 3D frame, respectively. Hereafter, we refer to this 2D system as the “image coordinates”. A visual representation of this geometry is provided in Figure 1.1.

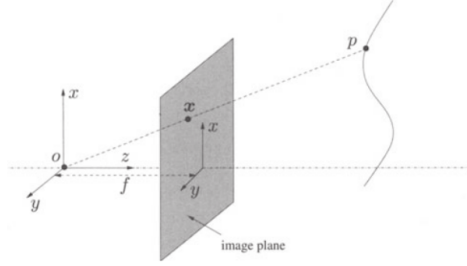


FIGURE 1.1: The frontal pinhole camera model illustrating the projection of a 3D point onto a 2D image plane. (Ma et al. 2004).

1.1.1 Coordinate Systems and Homogeneous Coordinates

In order to express 3D transformations using matrix multiplication in a consistent and compact way, it is common to represent points in *homogeneous coordinates*. A 3D point $P = [X, Y, Z]^\top$ in Euclidean space can be expressed in homogeneous form as a 4D vector:

$$\tilde{P} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1.1)$$

Homogeneous coordinates are also essential in camera projection models, as they facilitate the combination of intrinsic and extrinsic parameters into a single projection matrix. After transformation and projection, the resulting point is converted back to Euclidean coordinates by dividing by its last (homogeneous) component.

1.1.2 Extrinsic Parameters

In order to know what the camera "sees", it is necessary to know where it is "looking" at. This information is referred to as the extrinsic parameters or camera pose - the global orientation and position of the camera (Sturm et al. 2011).

In general, points in 3D space are represented in a coordinate system known as the world coordinate frame (W). The relationship between the world coordinate system and the camera's coordinate system (C) is established through a combination of a rotation and a translation, as illustrated in Figure 1.2 (Hartley 2003).

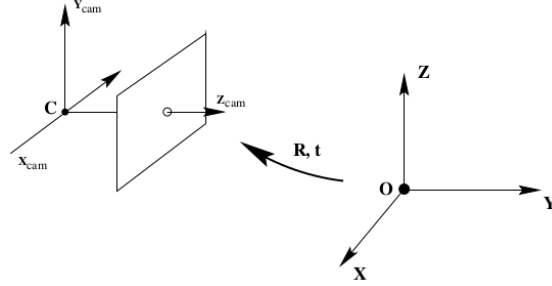


FIGURE 1.2: A rotation and translation from a world coordinate frame W to a camera frame C . (Hartley 2003)

These transformations are described by the extrinsic matrix, which consists of a rotation matrix R and a translation vector t (Zhengyou Zhang 2002), represented by the following rigid body transformation matrix:

$$[R | t] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{array} \right] \quad (1.2)$$

The transformation from a 3D point to a 2D point, as illustrated in Figure 1.2, can be expressed by 1.3.

$$P_C = [R | t]P_W \quad (1.3)$$

where P_C is the point in the camera frame, and P_W is the point in the world frame.

1.1.3 Intrinsic Parameters

The projection of a 3D point $P = [X, Y, Z]^T$ onto the 2D image plane coordinates $x = [x, y]^T$ is governed by the perspective projection equations:

$$x = f \frac{X}{Z} \quad y = f \frac{Y}{Z} \quad (1.4)$$

where f is the focal length.

In theory, image formation is described using a reference frame centered at the optical center of the camera, as described above. However, real-world digital cameras capture images in pixel coordinates, where the locations are expressed as (u, v) . The image coordinate system typically begins at the top left corner, with the horizontal and vertical axes aligned with the image rows and columns, respectively. To connect the image plane with the pixel array, we have to establish the relationship between the two (Figure 1.3).

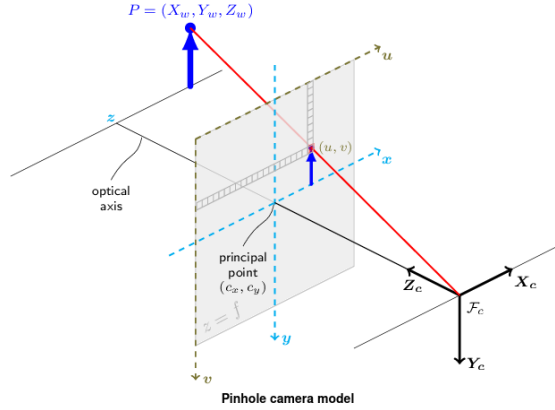


FIGURE 1.3: Conversion from the image plane to the pixel array.
(OpenCV 2021a)

The first step in modeling the projection process of a digital camera involves defining the measurement units along the axes of the image plane. If the coordinates (x, y) are defined in metric units (e.g., millimeters), and we need to convert them into pixel coordinates (x_s, y_s) . A scaling transformation is applied, which can be expressed using the following matrix, where s_x and s_y are the pixel scaling factors, depending on the pixel size and s_θ is called a *skew factor*, which will be different than 0 if the pixels are not rectangular:

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} s_x & s_\theta \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (1.5)$$

We still need to account for the fact that the pixel coordinate system used by digital cameras has its origin in the upper left corner of the image. Pixel indices increase to the right and downward from this corner. However, the coordinates derived from the camera model are typically defined relative to the principal point, the location where the camera's optical axis intersects the image plane. Since these two reference frames differ, we must apply a translation to shift the origin of the projected coordinates to align with the pixel-based system:

$$\begin{aligned} u &= x_s + o_x, \\ v &= y_s + o_y, \end{aligned} \tag{1.6}$$

where (o_x, o_y) are the pixel coordinates of the principal point relative to the image reference frame. This translation, following the scaling step, converts from the physical measurement system to the pixel-based representation used by digital image sensors. The complete process can be represented in the matrix shown in equation 1.7.

$$K = \begin{pmatrix} fs_x & s_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \tag{1.7}$$

This K matrix is called the calibration matrix, or the intrinsic matrix, as it contains five parameters intrinsic to the camera itself (Sturm et al. 2011) that dictate how the camera maps 3D rays onto 2D pixels.

1.1.4 Distortion

In a practical application, using only a pinhole to form an image would allow an extremely limited amount of light to reach the sensor, resulting in very dark and noisy images (Hecht 2017). To overcome this limitation, real cameras employ lenses. Lenses collect and focus significantly more light, enabling clearer and brighter images while maintaining a defined field of view and enabling control over focal length and depth of field.

Real-world cameras deviate from the ideal pinhole model due to imperfections in lens manufacturing and alignment. These deviations, known as distortion, can significantly affect the accuracy of computer vision tasks such as feature detection, stereo matching and triangulation Szeliski 2022.

1.1.4.1 Radial distortion

Radial distortion is a common type of optical distortion introduced by the curvature of lenses, particularly in wide-angle cameras. It causes straight lines in the world to appear curved in the image, especially near the edges of the frame. This deviation from the ideal pinhole model must be accounted for in camera calibration and image rectification processes. There are two main types of radial distortion (Szeliski 2022):

- **Barrel distortion:** straight lines appear to bulge outward from the center of the image.
- **Pincushion distortion:** straight lines curve inward toward the center.

Both types are illustrated in Figure 1.4.

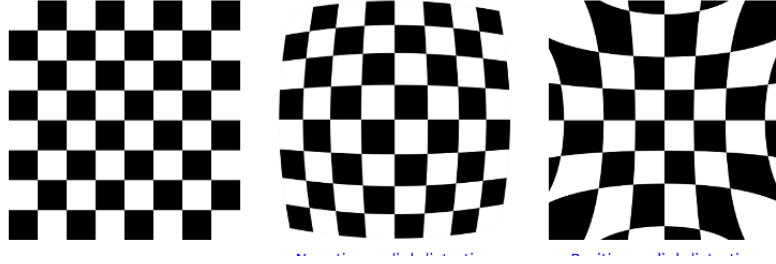


FIGURE 1.4: Left: No Distortion Center: Barrel Distortion Right: Pincushion Distortion. (OpenCV 2021a)

These effects are typically more pronounced at the periphery of the image and can be mathematically modeled using a set of distortion coefficients k_1 , k_2 , and k_3 :

$$\begin{aligned} x_{\text{distorted}} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{\text{distorted}} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{1.8}$$

where $r^2 = x^2 + y^2$ is the squared radial distance from the image center. Correcting radial distortion is essential for tasks such as feature detection, triangulation, and accurate 3D reconstruction (Tomasi 2015).

1.1.4.2 Tangential distortion

Tangential distortion arises when the lens is not perfectly parallel to the image sensor plane. In an ideal camera, the optical axis is perpendicular to the image plane, ensuring symmetric projection of points. However, slight misalignments during manufacturing or assembly can cause this assumption to fail.

As a result, image points appear to shift tangentially — that is, sideways — from their ideal locations. This displacement skews the image geometry, especially toward the periphery of the image, introducing asymmetric distortions that cannot be explained by radial distortion alone. Tangential distortion is modeled using two coefficients, p_1 and p_2 (OpenCV 2021a), and is typically expressed by the following equations:

$$\begin{aligned}x_{\text{distorted}} &= x + [2p_1xy + p_2(r^2 + 2x^2)] \\y_{\text{distorted}} &= y + [p_1(r^2 + 2y^2) + 2p_2xy]\end{aligned}\tag{1.9}$$

where $r^2 = x^2 + y^2$ is the squared distance from the optical center, and (x, y) are the normalized coordinates in the image plane. These terms correct the non-radial components of distortion due to lens decentering and tilt.

To compensate for imperfections introduced by the imaging system, modern camera models incorporate both radial and tangential distortion components, using a total of five parameters: k_1, k_2, k_3 for radial distortion, and p_1, p_2 for tangential distortion. The distorted coordinates $(x_{\text{distorted}}, y_{\text{distorted}})$ are computed as follows:

$$\begin{aligned}x_{\text{distorted}} &= x(1 + k_1r^2 + k_2r^4 + k_3r^6) + 2p_1xy + p_2(r^2 + 2x^2) \\y_{\text{distorted}} &= y(1 + k_1r^2 + k_2r^4 + k_3r^6) + p_1(r^2 + 2y^2) + 2p_2xy\end{aligned}\tag{1.10}$$

This model enables geometric corrections that are essential for precise calibration and robust 3D perception.

1.2 Stereoscopic Vision

Stereo vision is a technique used in computer vision to infer depth information from two or more images taken from slightly different viewpoints. Inspired by human binocular vision, stereo vision systems typically employ two cameras placed a known distance apart (called the baseline), capturing the same scene simultaneously. The goal is to recover 3D geometric structure by analyzing the differences between corresponding points in the left and right images.

Figures 1.5 and 1.6 show an example of a stereo camera setup and the images produced. The disparity map of the stereo scene is shown in Figure 1.8.

1.2.0.1 Epipolar Geometry

Epipolar geometry describes the intrinsic projective geometry between two views of a scene. It is fundamental to stereo vision because it governs the relationship between corresponding points in the left and right images, thereby constraining the search space for disparity computation. It is independent of scene structure, and only depends on the cameras' internal parameters and relative pose (Hartley 2003).

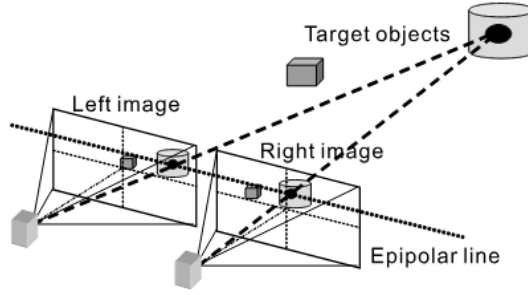


FIGURE 1.5: Generic stereo vision setup. (Hariyama et al. 2008)

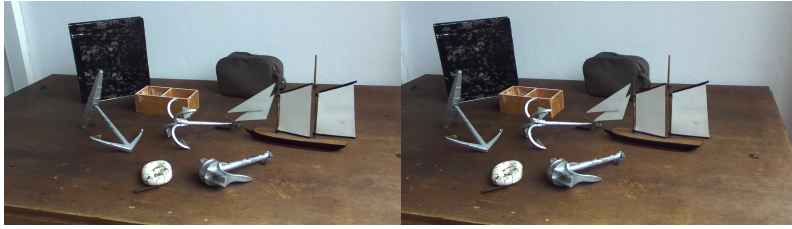


FIGURE 1.6: Example stereo scene (captured by the author in a classroom at the Portuguese Naval Academy).

When a 3D point X is projected onto two camera image planes, it yields two corresponding image points x in the left image and x' in the right image. The lines that connect the 3D point X to the centers of the two cameras defines a plane known as the epipolar plane. This plane intersects each image plane along lines known as epipolar lines (defined as l and l' , intersecting the baseline at the epipoles e and e' as illustrated in Figure 1.7 (Hartley 2003).

The key property of epipolar geometry is that the corresponding point x' of a point x must lie on the epipolar line associated with X . In a rectified stereo pair, where both image planes are aligned so that the epipolar lines are horizontal and collinear, this constraint simplifies the correspondence problem to a one-dimensional search along the image rows.

Due to the different viewpoints of the cameras, a 3D point projects to two different positions in the two image planes. The displacement between these projections, known as disparity, is inversely proportional to the depth of the point:

$$Z = \frac{f \cdot b}{d} \quad (1.11)$$

where Z is the depth of the point, f is the focal length of the cameras, b is the baseline distance between the cameras and d is the disparity.

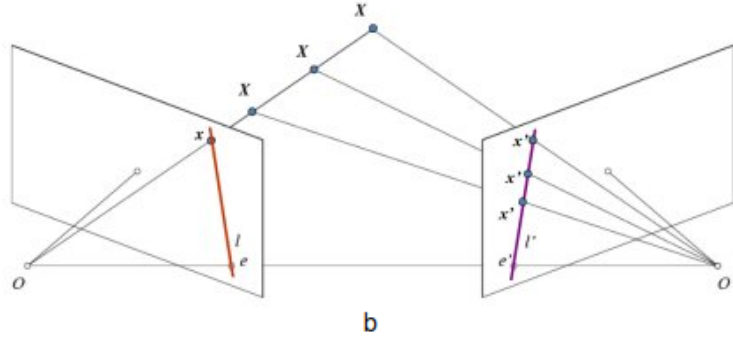


FIGURE 1.7: Epipolar Geometry (adapted from OpenCV 2023).

By computing this disparity, which is the horizontal pixel difference between corresponding points, across the image, stereo vision systems are able to reconstruct a dense disparity map, which encodes depth information for the entire scene. An example of such a map is shown in Figure 1.8. Once the depth Z is known, the 3D X and Y coordinates can be derived by back-projecting (Hartley 2003) the pixel coordinates (u, v) back into 3D space, relative to the camera's principal point (o_x, o_y) :

$$\begin{aligned} X &= \frac{(u - o_x) \cdot Z}{f} \\ Y &= \frac{(v - o_y) \cdot Z}{f} \end{aligned} \tag{1.12}$$

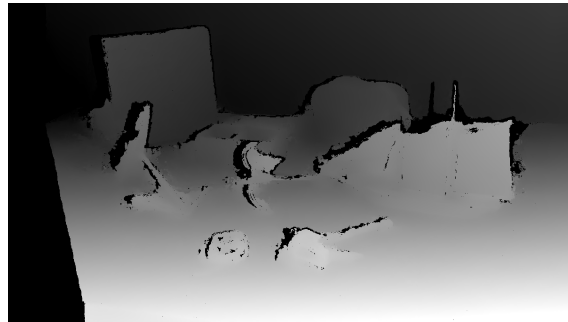


FIGURE 1.8: Stereo disparity map of the example scene presented in Figure 1.6.

1.3 Simultaneous Localization and Mapping

Simultaneous Localization and Mapping refers to the process by which a robot or autonomous system builds a map of an unknown environment while simultaneously determining its location within that map (Macario Barros et al. 2022).

This is a fundamental capability for autonomous navigation, especially in GNSS-denied environments and uncharted areas where there are no available prior maps or up-to-date tactical information. In essence, SLAM solves two interdependent problems at once:

- **Localization:** Estimating the current pose (position and orientation) of the robot;
- **Mapping:** Incrementally constructing a map of the surrounding environment.

Historically, localization and mapping were treated as distinct problems, but it became evident that they are strongly interdependent (Al-Tawil et al. 2024), thus referred to as the "Chicken and Egg" problem (Taheri and Xia 2021).

SLAM algorithms rely on sensor inputs such as cameras, LiDAR, Inertial Measurement Units (IMUs), and GNSS. Due to the inherent noise in the sensor measurements, a SLAM problem is usually described by using a probabilistic mathematical formulation (Grisetti et al. 2011). In this thesis, we focus on stereo visual SLAM, which utilizes the depth information derived from stereo camera systems (as described in Section 1.2) to generate 3D environmental representations.

Using SLAM, a USV can autonomously explore, map, and navigate such environments safely, avoiding obstacles and planning efficient routes in real-time.

According to (Thrun et al. 2006), there are two main forms of the SLAM problem: Online SLAM and Full SLAM. Online SLAM involves estimating the posterior over the momentary pose and the map:

$$p(\mathbf{x}_t, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \quad (1.13)$$

where $\mathbf{x}_t$ denotes the robot's pose at time t , $\mathbf{m}$ is the map, $\mathbf{z}_{1:t}$ are the measurements (e.g., landmarks or features), and $\mathbf{u}_{1:t}$ are the control inputs up to time t . Full SLAM, in contrast calculates the posterior over the entire trajectory of the robot along with the map:

$$p(\mathbf{x}_{1:t}, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \quad (1.14)$$

where $\mathbf{x}_{1:t}$ denotes the full sequence of robot poses from time 1 to t , $\mathbf{m}$ represents the map, $\mathbf{z}_{1:t}$ are the sensor measurements, and $\mathbf{u}_{1:t}$ are the control inputs. This formulation allows for improved global consistency in both the estimated trajectory and the reconstructed environment.

Depending on the formulation they address, SLAM implementations can be broadly classified into filtering-based for Online SLAM and optimization-based approaches for Full SLAM (Grisetti et al. 2011).

Beyond the problem formulation, SLAM systems can also be systematically classified according to several criteria. These include the algorithmic approach (e.g., filtering versus optimization), the sensing modalities employed, the data processing strategy, and the type of map representation. Additional distinctions concern whether implementations are open-source, the method of loop closure, and the degree of IMU integration (Tourani et al. 2022). Table 1.1 provides a concise overview of these classification variables and their most common options.

TABLE 1.1: SLAM Classification Variables and Typical Options.

Variable	Typical Options
Algorithmic Approach	Filter-based, Optimization-based
Sensor Modality	Monocular Camera, Stereo Camera, RGB-D ² Camera, LiDAR, IMU, Visual-Inertial, Multimodal
Data Processing Strategy	Feature-based, Direct (Dense/Semi-dense), Hybrid, Semantic SLAM, Dynamic-aware, Dynamic-inclusive
Map Representation	Sparse, Semi-dense, Dense, Topological, Object-level
Loop Closure Method	Visual Bag-of-Words, Scan Matching, Pose Graph Optimization
IMU Fusion	No IMU, Loose Coupling, Tight Coupling

A typical stereo visual SLAM pipeline consists of the following (Kazerouni et al. 2022):

1. **Sensor Data Acquisition:** Acquisition of raw data from onboard sensors, normally stereo cameras and IMUs;
2. **Feature Extraction and Matching:** Detection of keypoints and their correspondence across consecutive frames;
3. **Pose Estimation:** Estimation of the sensor’s motion based on visual correspondences;
4. **Loop Closure and Optimization:** Recognition of previously visited locations to reduce accumulated drift and refine the trajectory;

²RGB-D stands for Red, Green, Blue–Depth. An RGB-D camera captures both the standard color information for each pixel and an additional depth channel.

5. **Mapping:** Triangulation of matched points to reconstruct the 3D structure of the environment.

In this context, a *keypoint* is a distinctive and repeatable image location that can be detected reliably under varying conditions such as changes in scale, rotation, or illumination (Szeliski 2022). Typical examples include corners, blobs, and other texture-rich regions (Tareen and Saleem 2018). Each keypoint is associated with a numerical descriptor, enabling robust matching across frames.

This modular pipeline provides the basis for the SLAM framework further explored in this thesis.

1.4 Performance Evaluation

Evaluating the performance of SLAM systems is essential for assessing their accuracy, robustness, and feasibility in real-world applications. In the context of stereo visual SLAM for USVs, performance metrics must capture both localization precision and mapping fidelity under challenging maritime conditions, such as dynamic water surfaces, reflections, low-texture regions, and moving obstacles.

For the stereo vision stage, disparity estimation was assessed through metrics such as mean depth, point cloud size, percentage of valid points, and average execution time across image pairs. These indicators provide a direct comparison between algorithms in terms of density, robustness to texture variation, and computational cost, all of which directly impact SLAM performance.

For localization, quantitative evaluation is typically based on the comparison between estimated trajectories and ground truth. In this thesis, two types of ground truth references were employed: (i) the known baseline trajectory provided in the Pohang Canal dataset (Chung et al. 2023) and (ii) GPS trajectories recorded during the acquisition of the Base Naval de Lisboa (BNL) dataset.

Localization accuracy was then quantified using standard metrics: the mean error, median error, root-mean-square error (RMSE), and maximum deviation between SLAM and ground truth poses. Both 3D and planar (XY) errors were reported, with an emphasis on XY as surface navigation is predominantly planar.

Mapping performance was evaluated qualitatively by comparing the generated maps and point clouds with known structures in the environment. For example,

in the BNL dataset, one of the SLAM reconstructions was overlaid on satellite imagery to verify alignment with a known docked vessel. Such analysis highlights not only the metric accuracy but also the structural fidelity of the produced maps.

Together, these quantitative and qualitative measures aim to provide a comprehensive framework for assessing the stereo visual SLAM implementations presented in this work.

Chapter 2

State of the Art

This chapter reviews the current state of the art relevant to the topic of this thesis. It covers datasets commonly used for visual SLAM, main stereoscopic vision applications, an overview of SLAM techniques with an emphasis on maritime applications, and concludes with identified gaps in the literature that motivate this research.

2.1 Datasets

Robust evaluation of visual SLAM algorithms requires representative datasets that closely reflect the target operational environment. While numerous stereo SLAM datasets exist, the majority focus on land-based or urban scenarios. Some of the more popular datasets used for SLAM are:

- **KITTI Dataset** (Geiger et al. 2013): One of the most widely used benchmarks for autonomous driving, providing stereo camera data with high-quality ground truth from LiDAR, GNSS or INS sensors in urban and rural road environments.
- **Cityscapes Dataset** (Cordts et al. 2015): Primarily designed for semantic segmentation and object detection, this dataset includes stereo images captured in diverse urban street scenes.
- **Oxford RobotCar Dataset** (Maddern et al. 2017): A large-scale autonomous driving dataset collected over a year, featuring stereo imagery, GPS, and LiDAR data across changing weather and lighting conditions in Oxford, United Kingdom.

In contrast, maritime datasets remain relatively scarce, and those available tend to emphasize object detection or scene understanding rather than continuous

sensor movement and SLAM applications. Moreover, many lack comprehensive ground truth localization data or stereo visual inputs. The following datasets were analyzed as candidates during this theses:

- **Maritime Obstacle Detection Dataset 2 (MODD2)** (Bovcon et al. 2018): Contains annotated video sequences for obstacle detection in maritime environments, captured from a static or slowly moving camera platform.
- **Visual Aerial Imaging for Surveillance (VAIS)** (M. M. Zhang et al. 2015): Focuses on aerial maritime surveillance imagery, with limited applicability for SLAM due to the absence of continuous trajectory data.
- **Large-scale Autonomous Robotic Sailing Dataset (LaRS)** (Žust et al. 2023): A diverse maritime dataset with per-pixel labeled frames for panoptic obstacle detection across lakes, rivers, and seas. Focuses on obstacle segmentation rather than continuous stereo visual SLAM data.
- **Pohang Canal Dataset** (Chung et al. 2023): A multi-modal maritime dataset acquired in restricted waters in Pohang, South Korea. The sensor suite includes three LiDARs, a marine radar, a stereo camera, an infrared camera, a 6-direction omnidirectional camera, an Attitude and heading reference system (AHRS), and a GPS with Real Time Kinematics (RTK).

Among maritime datasets, the **Pohang Canal Dataset** (Chung et al. 2023) stands out as the most complete for maritime stereo visual SLAM research. The dataset provides sensor calibration parameters and a SLAM-based baseline trajectory along a 7.5 km route covering narrow canals and port areas. The dataset captures challenging maritime conditions such as water reflections, dynamic obstacles, and environmental variability, making it well-suited for developing and benchmarking SLAM algorithms tailored to naval and coastal USVs. For this work only the July 1st daytime portion was used, it contains 22,183 stereo image pairs in 2048×1080 resolution totaling 112.2 GB. Figure 2.1 highlights the different areas captured in the dataset.



FIGURE 2.1: Map overview of the Pohang Canal Dataset, illustrating the data collection area and route (Chung et al. 2023).

2.2 Stereoscopic Vision

Stereo vision relies on solving the correspondence problem, namely determining which pixels in a stereo pair correspond to the same 3D point in the scene (Lazaros et al. 2008). This section reviews the principal stereo matching methods described in the literature.

2.2.1 Stereo Matching Algorithms

Stereo matching is a core component of stereo vision systems that aims to determine corresponding points from a conjugate image pair. To do this, it is necessary to measure the similarity between points. Over the years, stereo matching algorithms have evolved significantly, and can broadly be classified into two categories: sparse and dense methods. This classification corresponds with the distinction between feature-based and area-based techniques (Lazaros et al. 2008).

Sparse methods, also known as feature-based approaches, extract distinctive keypoints (e.g., corners, edges) and attempt to match them across the stereo pair. These methods are computationally efficient and robust in low-texture environments but provide depth only at selected points. Among these, some of the most widely used to identify keypoints are:

- **Scale Invariant Feature Transform (SIFT)** (Lowe 2004): A robust keypoint detector and descriptor, SIFT provides high invariance to scale, rotation, illumination, and viewpoint changes (Leutenegger et al. 2011). Keypoints are detected by constructing a *Difference of Gaussians* pyramid, obtained by subtracting images blurred with Gaussians at successive scales. This operation efficiently approximates the Laplacian of Gaussian and emphasizes blob-like

structures that are stable across space and scale (Yousif et al. 2015). After detection, each keypoint is assigned one or more orientations based on local gradient directions, and its neighborhood is encoded into a high-dimensional descriptor vector, ensuring robust matching under varying imaging conditions (Lowe 2004).

- **Features from Accelerated Segment Test (FAST)** (Rosten and Drummond 2006): A corner detection algorithm that identifies keypoints by comparing pixel intensities in a circular pattern around a candidate point. Optimized for speed using a decision tree, FAST is ideal for real-time applications (Rosten, Porter, et al. 2008). It lacks inherent scale or rotation invariance, often requiring complementary descriptors (Rublee et al. 2011) and edge filtering for robust matching.
- **Oriented FAST and Rotated BRIEF (ORB)** (Rublee et al. 2011): ORB combines FAST keypoint detection with a rotation-aware BRIEF descriptor (Yousif et al. 2015). It adds an orientation component and scale pyramids, making it more robust than FAST alone, its features are invariant to scale, rotation, and limited affine changes (Tareen and Saleem 2018). ORB is fast, open-source, and suitable for real-time visual SLAM applications, as demonstrated by its use in ORB-SLAM and its successors.

Dense methods, or area-based approaches, estimate depth for every pixel by evaluating similarity across fixed-size windows or by optimizing global cost functions. These techniques yield high-resolution depth maps but are more computationally demanding and sensitive to lighting, occlusions, and textureless regions. Global methods often involve complex energy minimization frameworks that are computationally intensive and difficult to scale for real-time applications (Hamzah and Ibrahim 2016), especially on embedded or resource-constrained platforms.

- **Stereo Block Matching (SBM)**: One of the earliest and simplest dense stereo methods, SBM computes disparity by comparing fixed-size blocks (windows) between the left and right images using a similarity measure such as Sum of Absolute Differences (SAD) or Sum of Squared Differences (SSD) (Lazaros et al. 2008). Despite its simplicity and efficiency, SBM often suffers from poor accuracy in textureless regions and along depth discontinuities due to its fixed window size and local matching approach;
- **Semi-Global Matching (SGM)** (Hirschmuller 2007): SGM improves over SBM by aggregating matching costs along multiple one-dimensional paths

through the image, effectively balancing local and global constraints without the high computational cost of full global optimization. It achieves smoother and more accurate disparity maps, preserving sharp edges while handling occlusions better.;

- **Semi-Global Block Matching (SGBM)** (OpenCV 2021g): SGBM is a practical implementation of SGM retaining the core principles of SGM, such as cost aggregation along multiple scanlines, but adds enhancements such as block-based matching using small windows, subpixel refinement, and built-in post-processing filters. SGBM computes initial matching costs over blocks using SAD and performs semi-global cost aggregation with penalties P_1 and P_2 to enforce smoothness while preserving depth discontinuities. Finally, a Winner-Takes-All (WTA) approach selects the optimal disparity per pixel. Internal filters like uniqueness ratio checks and speckle filtering help improve robustness in occluded or textureless regions, making it a widely adopted dense stereo method for real-time and embedded applications.

2.3 Simultaneous Localization and Mapping

SLAM has been an active research area for decades. Historically the first SLAM algorithm is based on the Extended Kalman Filter (EKF). It solves the online SLAM problem, producing feature-based maps. It's limited by its quadratic update complexity and unsuitability for large maps, as it is generally limited to maps with under 1,000 features (Thrun et al. 2006).

Over the years, SLAM approaches have evolved significantly, and optimization-based SLAM methods have gained prominence. These approaches model SLAM as a graph optimization problem where nodes represent poses or landmarks and edges represent spatial constraints between them (Grisetti et al. 2011). By optimizing the entire pose graph or factor graph, these methods improve accuracy and consistency.

The advent of low-cost cameras has popularized Visual SLAM (V-SLAM), which relies primarily on images rather than range sensors. Visual SLAM can be further categorized into monocular, stereo, and RGB-D approaches, each with its trade-offs in accuracy and robustness. Among visual SLAM methods, ORB-SLAM (Mur-Artal, Montiel, et al. 2015) and its successors ORB-SLAM2 (Mur-Artal and Tardós 2017) and ORB-SLAM3 (Campos et al. 2021) stand out for their robustness,

real-time performance, and rich features such as loop closure and multi-map support. ORB-SLAM uses a graph-based approach, combining sparse feature extraction (ORB features) with pose graph optimization.

Real-Time Appearance-Based Mapping (RTAB-Map) Labbé and Michaud 2019 is a graph-based SLAM framework specifically designed for real-time performance with visual sensors, including stereo and RGB-D cameras. It integrates appearance-based loop closure detection and incremental graph optimization, making it suitable for large-scale environments and long-term autonomy. Its modular design make it a preferred choice for robotic applications, including those in challenging outdoor and marine settings.

Both RTAB-Map and ORB-SLAM are regarded as mature and feature-rich visual SLAM approaches, but RTAB-Map has better performance in outdoor scenarios, or even in degraded low-feature conditions (Merzlyakov and Macenski 2021).

Chapter 3

Methodology

This chapter outlines the methodological framework adopted in this thesis to implement and evaluate stereo visual SLAM algorithms for autonomous navigation in uncharted maritime environments. The goal is to establish a reproducible methodology that addresses the operational constraints and challenges of real-world deployments. This chapter will present the equipment and software used and developed, the process of creating a realistic dataset reflecting a real-world naval scenario, as well as the algorithms implemented for stereoscopic vision and SLAM. To ensure reproducibility is possible, this chapter demands detailed technical descriptions.

3.1 Overview

This work adopts a practice-driven methodology that progresses from system setup and dataset collection to algorithmic implementation and evaluation. Concretely, we execute three complementary experimental tracks that together answer the research questions posed in Chapter 1 and set up the analyses in Chapter 4.

1. **Stereo comparison on Pohang Canal.** We evaluate two open-source stereo pipelines on rectified image pairs from the Pohang Canal dataset: a dense method based on Semi-Global Block Matching and a feature-based method using Scale-Invariant Feature Transform. as summarized in Figure 3.1

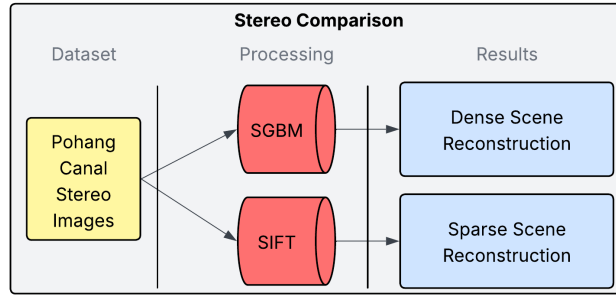


FIGURE 3.1: Stereo comparison diagram.

2. **SLAM in the Pohang Canal using RTAB-Map.** Using the same dataset, we build a SLAM pipeline in which SGBM supplies depth and the left camera provides the color images to RTAB-Map, with visual odometry estimated from the image stream, the output is a map and trajectory.

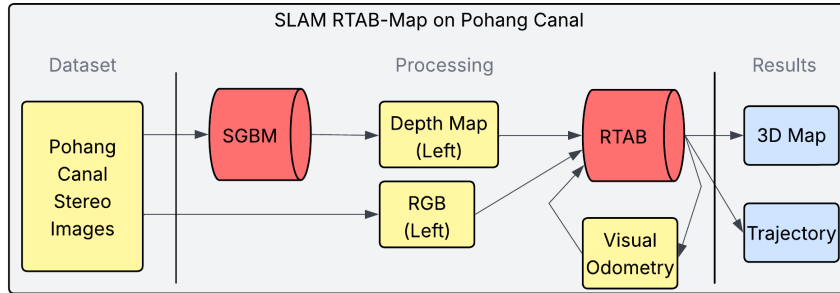


FIGURE 3.2: SLAM in the Pohang Canal diagram.

3. **SLAM in the BNL with RTAB-Map compared to ZED SDK.** For the novel dataset captured in the BNL, two practical pipelines are compared (Figure 3.3). The first is RTAB-Map, running in RGB-D mode, receiving the color images and IMU measurements from the ZED camera, and a depth image from the ZED SDK, outputting a map and trajectory. The other is the ZED SDK pipeline, which implements Stereolabs proprietary ZED SDK, using the Positional Tracking and Spatial Mapping models in order to output a map and trajectory.

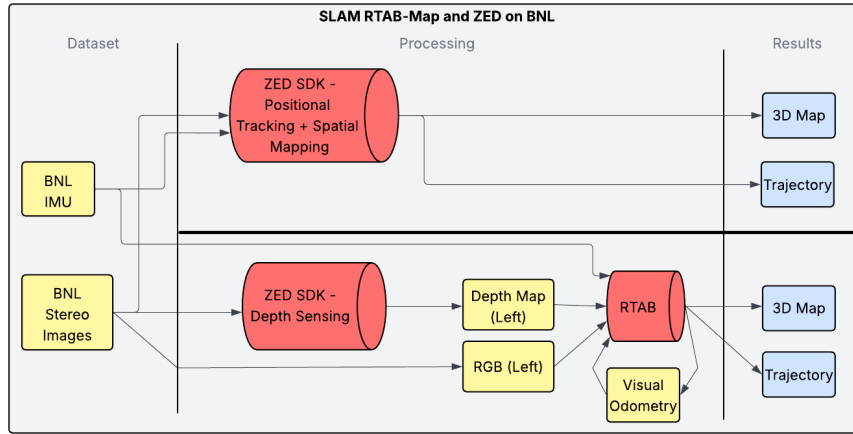


FIGURE 3.3: SLAM in the BNL diagram.

3.1.1 System Setup

The experimental framework for this thesis was developed using a combination of hardware and software components carefully selected to support stereo visual SLAM in maritime scenarios. Ubuntu 22.04 Long Term Support (LTS) Linux Operating System (Canonical Ltd. 2024) was chosen for its stability and wide support for robotics software. Robot operating system (ROS) 2 Humble Hawksbill (Open Robotics 2024) was employed as the primary middleware, providing a modular and real-time capable communication framework with the valuable capability of recording and reproducing sensor data. A Garmin GPS 18 USB receiver was used to provide ground truth positional data during data acquisition, and the ZED 2i stereo camera by Stereolabs was used for its high resolution depth sensing, integrated sensors, and robust Software Development Kit (SDK) support (Stereolabs 2024b).

3.1.1.1 ZED 2i Stereo Camera Specifications

The ZED 2i is a high-performance stereo camera designed for real-time depth sensing and tridimensional perception in challenging environments. Its integrated inertial and environmental sensors, along with the provided SDK make it well-suited for mobile robotic applications such as SLAM. Figure 3.4 shows the ZED 2i stereo camera and Table 3.1 shows the camera specifications as per the Datasheet provided by the manufacturer.

³CMOS stands for *Complementary Metal-Oxide-Semiconductor*, a widely used type of image sensor in digital cameras.



FIGURE 3.4: ZED 2i Stereo Camera (StereoLabs 2024).

TABLE 3.1: Key Specifications of the ZED 2i Stereo Camera (Adapted from Stereolabs 2021).

Parameter	Specification
Output Resolution	$2 \times (2208 \times 1242) @ 15 \text{ fps}$, $2 \times (1920 \times 1080) @ 30 \text{ fps}$, $2 \times (1280 \times 720) @ 60 \text{ fps}$, $2 \times (672 \times 376) @ 100 \text{ fps}$
Baseline	12 cm
Focal Length	4 mm
Field of View (4 mm)	$72^\circ \text{ (H)} \times 44^\circ \text{ (V)} \times 81^\circ \text{ (D)}$
Depth Range (4 mm)	1.5 m to 35 m (ideal: 1.5 m to 20 m)
Depth Accuracy (4 mm)	$< 0.4\%$ at 2 m, $< 7\%$ at 20 m
RGB Sensor	Dual $1/3''$ 4MP CMOS ³ , 2688×1520 pixels
Motion Sensors	Accelerometer ($\pm 8 \text{ g}$, 0.244 mg resolution), Gyroscope ($\pm 1000^\circ/\text{s}$, $0.03^\circ/\text{s}$ resolution), Magnetometer ($\pm 2500 \mu\text{T}$, $0.3 \mu\text{T}$ resolution)
Environmental Sensors	Barometer (300 hPa–1100 hPa, 0.18 Pa resolution), Temperature (-40°C to 125°C , $\pm 0.5^\circ\text{C}$ accuracy)
Dimensions	$175.3 \text{ mm} \times 30.3 \text{ mm} \times 43.1 \text{ mm}$
Weight	229 g
Operating Temperature	-10°C to 45°C

3.1.1.2 ZED SDK

The ZED SDK provides comprehensive tools and Application Programming Interface (API) for interfacing with the ZED stereo cameras, enabling real-time depth sensing, spatial mapping, object detection and other advanced computer vision functionalities. Developed and maintained by Stereolabs, it serves as the foundational software layer for accessing camera data and integrating it into robotics and vision applications, as represented in Figure 3.5. Although the ZED SDK is proprietary, it allows access not only to processed depth images and point clouds but also to raw sensor data, offering low-level control of the camera.

3.1.1.3 ZED 2i calibration parameters

The ZED 2i comes factory calibrated, and a calibration file was generated during the manufacturing process. This file contains intrinsic parameters for each

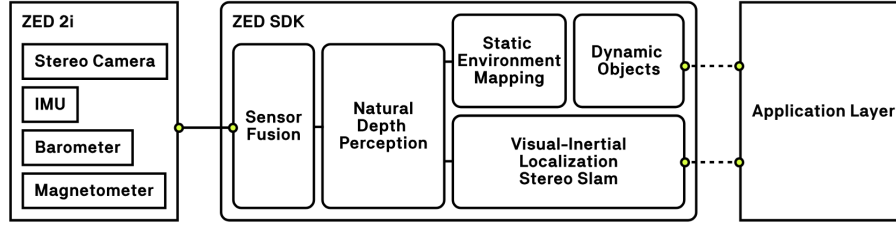


FIGURE 3.5: ZED SDK Functional Diagram. (Stereolabs 2021)

camera adapted per resolution and extrinsic parameters between them. Consequently, there is no need to perform further camera calibration. Table 3.2 lists the calibration parameters for the ZED2i used in this work at 2K resolution.

TABLE 3.2: Calibration parameters of the ZED 2i at 2K resolution (2208×1242), extracted from the camera’s factory calibration file.

Parameter	Left Camera	Right Camera
Focal Length (fx, fy)	(1905.39, 1905.22)	(1908.99, 1909.59)
Principal Point (cx, cy)	(1077.21, 615.691)	(1099.17, 652.848)
Distortion Coefficients (k1, k2, k3)	-0.0627781, -0.0522924, -0.0132741	-0.0641699, -0.046248, -0.0221382
Tangential Distortion (p1, p2)	-0.000431355, -0.000143438	0.000423426, -0.00039704
Extrinsic Parameters		
Baseline (mm)	119.679	
Rotation (Rodrigues notation, radians)	RX = 0.00382799, RZ = 0.00257864	
Translation (mm)	TY = -0.217477, TZ = 1.06505	

3.1.1.4 ROS 2 integration

ROS 2 provides a flexible and modular communication framework that enables fine-grained control over sensors and processing pipelines. It supports synchronized recording of datasets in ROS bags, capturing GPS and camera data for later analysis and reproducibility. Furthermore, ROS 2’s architecture allows seamless integration with SLAM applications such as RTAB-Map, enabling robust real-time navigation and mapping.

The ROS 2 wrapper for the ZED SDK is an open-source software package developed by Stereolabs that acts as a bridge between the proprietary ZED SDK and the ROS 2 ecosystem. It enables seamless integration of the ZED stereo camera within ROS 2, exposing the camera’s functionalities through standardized ROS 2 topics and services. The wrapper can publish multiple topics, some with just the raw sensor data, and others with processed information. Some examples are:

- `/zed2/zed_node/left/image_raw` and `/zed2/zed_node/right/image_raw`: Raw stereo images from the left and right cameras.

- `/zed2/zed_node/imu/data`: IMU measurements (accelerometer and gyroscope) from the camera's onboard sensors.
- `/zed2/zed_node/depth/depth_registered`: Registered depth map aligned to the RGB image frame.
- `/zed2/zed_node/point_cloud/cloud_registered`: 3D point cloud data representing the environment.

3.1.1.5 Garmin GPS 18

The Garmin GPS 18 USB, illustrated in Figure 3.6, is a reliable GPS receiver featuring a 1 Hz update rate and outputting data in the Garmin Binary format, with a Differential Global Positioning System (DGPS) error of less than 3 meters (Garmin 2024).



FIGURE 3.6: Garmin GPS 18 USB (Garmin 2024).

Its data stream can be accessed using the `gpsd` daemon, a widely-used GPS service. In Figures 3.7 and 3.8 we can see the GPS information in the `cgps` and `xgps` client applications, a terminal-based interface and a graphical interface that display the GPS data in real time.

		Seen 12/Used 71				
Time:	2025-04-10T14:15:12.000Z (18)	GNSS	PRN	Elev	Azim	SNR Use
Latitude:	38.66238203 N	GP 1	1	28.0	247.0	45.0 Y
Longitude:	9.14803050 W	GP 2	2	56.0	272.0	44.0 Y
Alt (HAE, MSL):	106.844, 56.798 m	GP 8	8	66.0	331.0	32.0 Y
Speed:	0.04 km/h	GP 10	10	48.0	66.0	48.0 Y
Track (true, var):	194.8, -2.1 deg	GP 16	16	34.0	163.0	47.0 Y
Climb:	-0.36 m/min	GP 26	26	9.0	160.0	40.0 Y
Status:	3D DGPS FIX (112 secs)	GP 27	27	61.0	80.0	43.0 Y
Long Err (XDOP, EPX):	2.15, +/- 8.1 m	GP 7	7	13.0	276.0	n/a N
Lat Err (VDOP, EPY):	1.45, +/- 5.5 m	GP 23	23	13.0	37.0	n/a N
Alt Err (VDOP, EPV):	3.60, +/- 27.2 m	GP 32	32	6.0	111.0	42.0 N
2D Err (HDOP, CEP):	2.60, +/- 13.3 m	SB123	36	34.0	134.0	44.0 N
3D Err (PDOP, SEP):	4.43, +/- 30.4 m	SB136	49	43.0	158.0	51.0 N
Time Err (TDOP):	3.00					
Geo Err (GDOP):	5.35					
Speed Err (EPS):	+/- 58.0 km/h					
Track Err (EPD):	n/a					
Time offset:	-0.029115032 s					
Grid Square:	IM58kp28					

FIGURE 3.7: Cgps client showing GPS data.

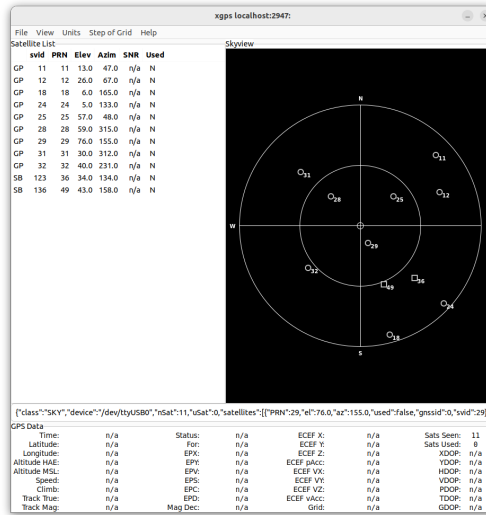


FIGURE 3.8: Xgps client showing location of visible GPS satellites.

3.2 Dataset Creation

Dataset recording was performed using ROS 2's native bag system to capture synchronized sensor data streams. This included stereo camera images, depth data, IMU measurements, and GPS information. The synchronized dataset enables consistent replay and analysis for SLAM evaluation. As illustrated in Figure 3.9 The ZED 2i camera was mounted on the bow of a Rigid Hull Inflatable Boat (RHIB), 1 meter above water level. The Garmin GPS was placed 1m aft, on the RHIB's console.



FIGURE 3.9: RHIB setup for dataset capture in the maneuvering basin of the Lisbon Naval Base.

While recording both the camera and GPS output, the RHIB followed the path shown in Figure 3.10, aiming to the pier or docked ships in the captured image, within the practical working range of the camera.

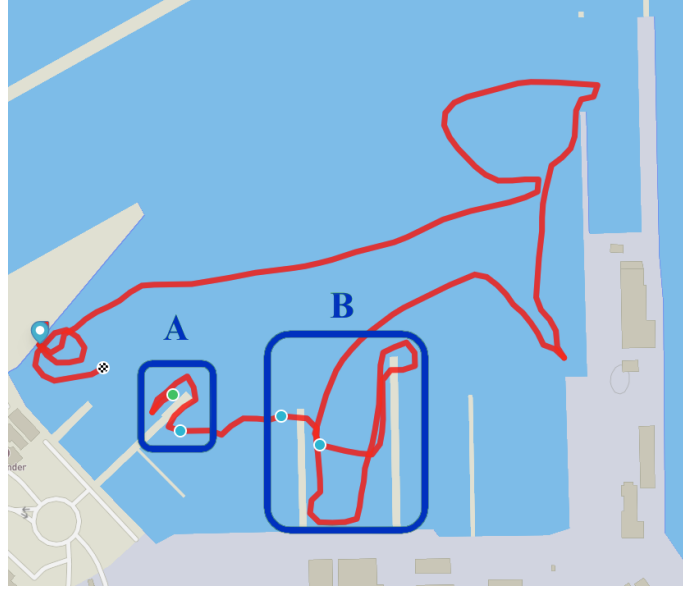


FIGURE 3.10: Trajectory of the RHIB while recording the dataset. A and B correspond to the two different ROS 2 bags used in the trials.

In total 12 ROS 2 bags were recorded. These bags collectively contain 21,512 stereo RGB image pairs, disparity maps, and depth images, alongside 291,866 IMU data points, 21,512 camera poses, 21,513 odometry points, and 2,937 GPS positions. The dataset totals 50.46 GB and encompasses 1,670,736 messages across all topics, providing a comprehensive record of sensor data for analysis. Of all the bags, two stand out as the best for SLAM implementation, due to their proximity to the pier and ships. These two will be considered as two datasets for the application of the SLAM pipelines, and referred to as BNL A and BNL B. The areas they were recorded in are identified in Figure 3.10.

3.3 Algorithms

This section is divided into two main parts. The first subsection explores stereo vision methods, comparing dense disparity-based approaches with feature-based techniques for depth estimation. The second subsection focuses on SLAM algorithms, evaluating three different implementations on three datasets.

3.3.1 Stereo Comparison

The problem formulation and thesis objectives lead us to adopt stereo visual SLAM as a tactically sound approach using passive sensing. However, as discussed in Chapter 2, this choice still leaves open the decision between generating dense or sparse maps. Although dense maps provide richer information, they may be too resource-intensive for practical applications. Therefore, we compare two stereo vision approaches for depth estimation: a direct method, Semi-Global Block Matching, and a feature-based method employing Scale-Invariant Feature Transform. These two methods were chosen for their open-source availability. Both methods were applied for deriving distance information from real stereo image pairs from the Pohang Canal Dataset.

3.3.1.1 Image Rectification

Stereo rectification is a fundamental preprocessing step in the computer vision pipeline for 3D reconstruction from stereo images. The process involves applying a projective transformation to each image plane to make them coplanar.

To perform stereo rectification, a complete set of intrinsic and extrinsic camera parameters must be known. These parameters are provided in the Pohang Canal Dataset in two files: `extrinsics.json` and `intrinsics.json` and contain the information shown in tables 3.3 and 3.4.

TABLE 3.3: Intrinsic Parameters of the Left and Right Pohang Stereo Cameras (Adapted from Chung et al. 2023).

Parameter	Left Camera	Right Camera
Principal point o_x (pixels)	1026.20	1041.30
Principal point o_y (pixels)	524.80	519.07
Distortion coefficients	$-0.0823, 0.1037, -0.0035$ $+0.0008, 0.0937$	$-0.0802, 0.0931, -0.0029$ $+0.0011, 0.1168$
Focal length f (pixels)	1814.51	1812.26
Image height (pixels)	1080	1080
Image width (pixels)	2048	2048

Because the extrinsic parameters are using the vehicle platform as a reference coordinate system, as opposed to the common left camera origin, we must first calculate the relative transform between the two cameras.

Using the SciPy library (Virtanen et al. 2020), we convert the rotation information from quaternions to rotation matrices: L_r and R_r for the left and right cameras, respectively.

TABLE 3.4: Extrinsic Parameters of the Pohang Stereo Cameras
(Adapted from Chung et al. 2023).

Parameter	Left Camera	Right Camera
Quaternion (w, x, y, z)	0.505	0.500
	0.493	0.493
	0.503	0.503
	0.499	0.504
Translation (m)	5.036	5.010
	-0.441	0.402
	-1.771	-1.762

With these rotation matrices and the translation vectors L_t and R_t from the dataset, the relative rotation R and translation $\mathbf{t}$ from the right to the left camera are computed according to equation 3.1.

$$R = R_r^\top L_r \quad \mathbf{t} = R_r^\top (t_{\text{left}} - t_{\text{right}}) \quad (3.1)$$

Next, the OpenCV library function `cv2.stereoRectify()` can be used to compute the rectification transforms for each stereo camera (OpenCV 2021f). With the output, we can use `cv2.initUndistortRectifyMap()` to generate undistortion and rectification transformation maps for each camera (OpenCV 2021c), making sure the epipolar lines on both images become horizontal and have the same y coordinate. The function computes, for every pixel (u, v) in the rectified image, its corresponding location in the original distorted image. Its output can be directly fed to `cv2.remap()` to apply a geometrical transformation to the original images, resulting in a rectified pair (OpenCV 2021e).

In Figure 3.11 it is possible to observe an original image and its rectified counterpart.

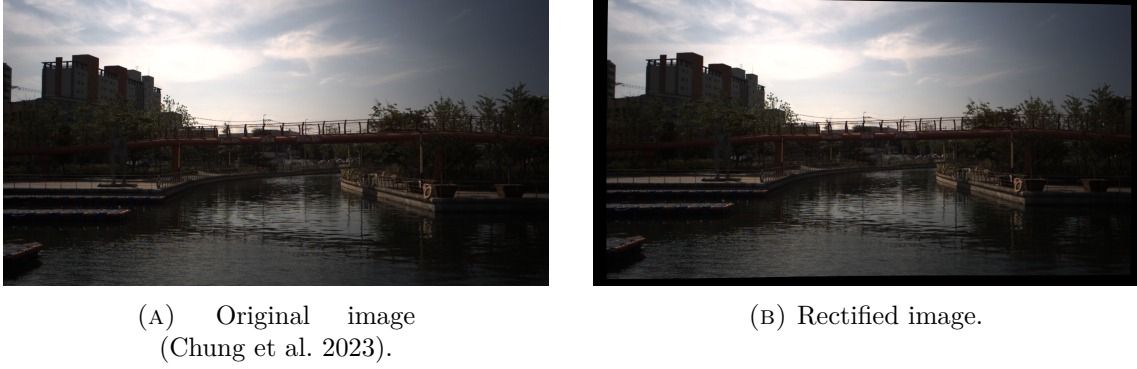


FIGURE 3.11: Comparison between the original and rectified image.

3.3.1.2 SGBM

As referred in Section 2.2.1, SGBM is a dense stereo correspondence algorithm that builds upon the original Semi-Global Matching approach by incorporating block-based matching for improved robustness and additional filtering capabilities. In Figure 3.12, we present the disparity map generated by applying the SGBM algorithm to the pair of previously rectified stereo images from the Pohang Canal Dataset. The disparity map illustrates depth variations across the scene, where brighter values represent pixels closer to the camera and darker values indicate farther objects. The algorithm was initialized using OpenCV’s `cv2.StereoSGBM_create()` function with carefully selected parameters to balance quality and computational performance. The `minDisparity` was set to 0, and `numDisparities` was set to 128, defining the search range of disparity values. The `blockSize` controls the size of the matching window and must be an odd number. Larger values improve noise resistance but reduce detail around fine structures, a size of 11 pixels was used resulting in a 11×11 pixel window. The smoothness penalties `P1` and `P2` were set proportionally to the square of the block size, following common heuristics: $P1 = 8 * 3 \text{ (number of image channels)} * blockSize^2$ and $P2 = 32 * 3 \text{ (number of image channels)} * blockSize^2$ (OpenCV 2021g). These parameters discourage abrupt disparity changes, encouraging spatial consistency. The parameter `disp12MaxDiff` was set to 1 to enforce left-right consistency. To improve matching reliability, the `uniquenessRatio` was increased to 15, which ensures that the best match is significantly better than the second-best. Finally, speckle filtering parameters `speckleWindowSize` and `speckleRange` were set to 100 and 32, respectively, to remove small, isolated disparity regions likely caused by noise. The algorithm ran in `STEREO_SGBM_MODE_HH4` mode, enabling a more exhaustive cost aggregation along 16 paths for better accuracy. The cost aggregation stores the matching costs of each pixel at each disparity, and then a Winner Takes All approach

is used to select each pixels disparity that minimizes the total cost, producing the disparity map seen in Figure 3.12.

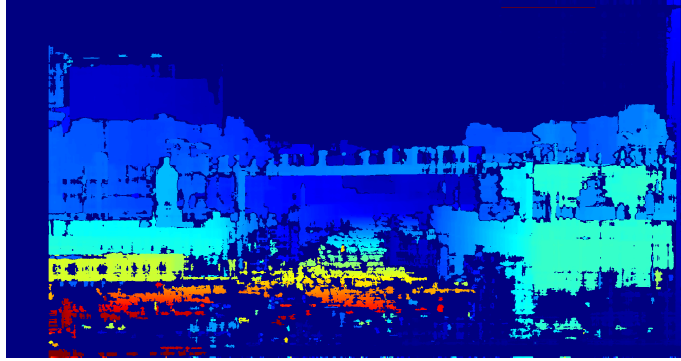


FIGURE 3.12: Disparity map produced using the SGBM algorithm.

This raw output still contains some artifacts, particularly in low-texture areas and around occlusion boundaries, which are addressed in the subsequent post-processing stage. First a second disparity map is generated from the perspective of the right image, so that we can identify any pixels that do not match as being unreliable, often due to occlusion. Then a Weighted Least Squares (WLS) Filter is applied. This filter takes the noisy left disparity map and the original left image as a guide. It smooths the disparity map while preserving sharp discontinuities that align with edges in the original image. It uses the information from the left-right consistency check to effectively fill in occluded regions. The filter's behavior is controlled by the `Lambda` (amount of smoothing) and `SigmaColor` (sensitivity to image color/intensity changes) variables. Figure 3.13 shows the filtered disparity map produced using $\textit{Lambda} = 8000.0$ and $\textit{SigmaColor} = 0.8$.

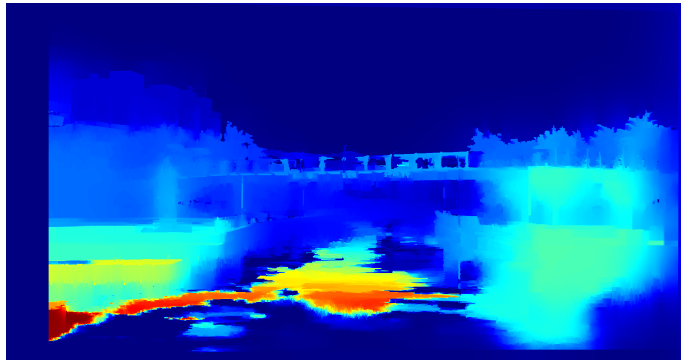


FIGURE 3.13: Filtered disparity map.

Using the focal length provided in table 3.3 and the baseline computed in equation 3.1, we calculate the depth for each pixel, according to its disparity, as

shown in equation 1.11. We can then apply equations 1.12 to obtain the full set of 3D coordinates, then passed to the `open3d.geometry.PointCloud()` function from Open3D (Zhou et al. 2018) to create a point cloud with the reconstructed 3D scene, as shown in Figure 3.14.



FIGURE 3.14: Dense scene reconstruction from the computed disparity map in a 3D point cloud.

3.3.1.3 SIFT

Conversely, the SIFT-based method detects distinctive, scale and rotation invariant keypoints in stereo pairs and matches them to compute sparse 3D points through triangulation. This yields a sparse but robust 3D reconstruction well-suited for feature tracking under varying visual conditions.

Starting from a rectified stereo image pair, we must construct a SIFT object with `cv2.SIFT_create()`, defining its variables in such a way to set no limit for the number of features, using 4 `OctaveLayers`, a `contrastThreshold` of 0.08, an `edgeThreshold` of 20, and a `sigma` of 1.6. Originally the values used were the ones recommended in OpenCV’s documentation (OpenCV 2021d), which were in turn a summary of the empirical data of the original SIFT paper (Lowe 2004). They were then adapted to get better results on the Pohang Canal Dataset. Image 3.15 shows the rectified left image with the detected keypoints, drawn with OpenCV’s `cv2.drawKeypoints()` function.



FIGURE 3.15: SIFT keypoints represented with scale and orientation on the left rectified image.

Next, we use a Fast Library for Approximate Nearest Neighbors (FLANN) based descriptor matcher. FLANN is a library composed of a collection of algorithms optimized for fast nearest neighbor search (Muja and Lowe 2009). This approach efficiently matches SIFT descriptors between images, enabling robust feature correspondence even in large datasets (OpenCV 2021b). Given the relatively low dimensionality of SIFT descriptors (128 dimensions), the KD-tree (Bentley 1975) algorithm with 5 trees and 150 checks aims to balance speed and accuracy (Muja and Lowe 2009). Since raw feature matches often contain a significant number of incorrect correspondences, two filtering techniques are applied to improve match quality. The first is Lowe’s Ratio Test (Lowe 2004), which compares the distance of the best match to the distance of the second-best match. A match is retained only if the best match distance is less than 0.75 times the second-best distance. Then, we verify that matching points lie on the same epipolar line by checking the vertical pixel difference.

Figure 3.16 shows the matched keypoints between a stereo pair after filtering.

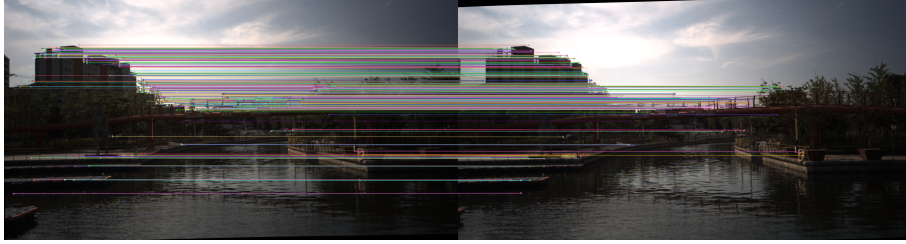


FIGURE 3.16: Filtered SIFT keypoint matches.

At this stage we have a high-confidence set of 2D point pairs that correspond to the same 3D points in the scene. Unlike with SGBM, we cannot create a dense disparity map, but we can calculate the disparities of each pair, and therefore the distances of those points. Figure 3.17 shows the calculated distances for each point in the scene.



FIGURE 3.17: Filtered SIFT keypoints with calculated distances.

Using `cv2.triangulatePoints()` a 3D point cloud can be reconstructed from matched stereo keypoints. The function reconstructs 3-dimensional points from the matched SIFT keypoints, left camera's intrinsic matrix, the right camera's projection matrix P_r calculated as seen in Equation 3.2.

$$\mathbf{P}_r = \mathbf{K}_r \cdot [\mathbf{R} \mid \mathbf{T}] \quad (3.2)$$

$\mathbf{K}_{\text{right}}$ is the intrinsic matrix of the right camera. $\mathbf{R}$ is the rotation matrix from the left camera coordinate system to the right camera coordinate system, and $\mathbf{T}$ is the translation vector from the left camera to the right camera.

The combination of the projection matrices with the matched SIFT keypoints provides the input required for triangulation, from which the 3D scene coordinates are recovered (OpenCV 2021a). This results in a sparse point cloud, as seen in Figure 3.18.



FIGURE 3.18: Sparse scene reconstruction from the computed point-cloud.

The evaluation of the obtained distances and point clouds for each method can be found in chapter 4, along with a comparison of both methods.

3.3.2 SLAM

To evaluate stereo-based Simultaneous Localization and Mapping in uncharted port environments, two different SLAM frameworks were tested: Real-Time Appearance-Based Mapping - RTAB-Map (Labbé and Michaud 2019) and the ZED SDK SLAM. This dual approach allows a comparison between an open-source graph-based system widely adopted in robotics research, and a proprietary implementation tightly integrated with the ZED 2i camera hardware. Experiments were conducted on the Pohang Canal dataset using only RTAB-Map, as the ZED SDK cannot process images from another sensor. Furthermore, two scenarios of our own port environment dataset with both approaches, to demonstrate the applicability of these algorithms in GNSS-denied maritime scenarios.

3.3.2.1 RTAB-Map Integration - BNL

Real-Time Appearance-Based Mapping (RTAB-Map) is a graph-based SLAM framework designed for real-time performance with visual sensors. It was deployed for the BNLA A and BNL B datasets in RGB-D mode with IMU fusion within ROS 2 Humble, making use of the official ZED 2i ROS 2 wrapper (Stereolabs 2024a).

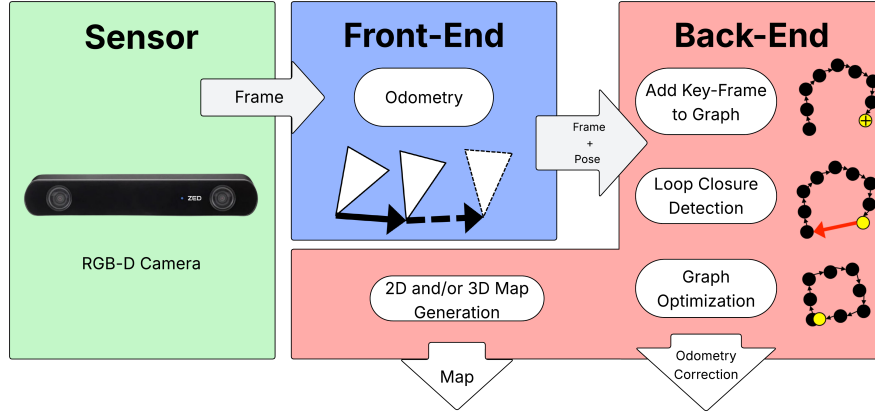


FIGURE 3.19: RTAB-Map system diagram (Adapted from Labbé 2015).

Figure 3.19 summarizes the architecture adopted in this work, following the author’s presentation (Labbé and Michaud 2019) and node diagrams (Labbé 2015). The system is split into a *Sensor* stage, a *Front-End* for odometry, and a *Back-End* for graph operations and map generation.

1. **Sensing and Pre-processing:** A ZED 2i provides rectified RGB frames, a *registered* depth image, and IMU measurements (RGB-D + IMU).
2. **Front-End (Odometry):** The RGB-D odometry node (`rgbd_odometry`) estimates the camera motion at frame rate. Using the depth image as a mask, binary features are detected and tracked between frames (descriptor matching); valid depth at matched pixels yields metric 3-D correspondences. We use the *frame-to-map* (*F2M*) strategy, aligning the current frame to a local feature map aggregated from recent keyframes; IMU samples supply gravity and short-term rotation priors and aid motion initialization/de-skew. The output is an incremental pose and the current frame.
3. **Back-End (Graph SLAM):** The `rtabmap` node receives the frame and pose:
 - (a) **Add keyframe to graph:** selected frames are inserted as nodes; odometry provides the temporal edges.

- (b) **Loop-closure detection:** appearance-based bag-of-words approach to identify past images that are visually similar to the current one. These candidate matches are then geometrically verified using depth information to ensure consistency. If the verification succeeds, the loop-closure is accepted and introduced as a constraint in the SLAM pose graph.
 - (c) **Graph optimization:** a nonlinear optimizer (g2o) refines all poses using odometry and loop-closure constraints, producing a globally consistent trajectory.
4. **Map Generation:** Depth images are fused along the optimized trajectory to produce dense occupancy maps (2-D and/or 3-D), aligned with the estimated poses.

The operation of RTAB-Map can be further understood by examining its main ROS node, `rtabmap`, illustrated in Figure 3.20. In the BNL portion of this work, only RGB-D images and the odometry estimated by RTAB-Map’s `rgbd_odometry` node were provided as input.

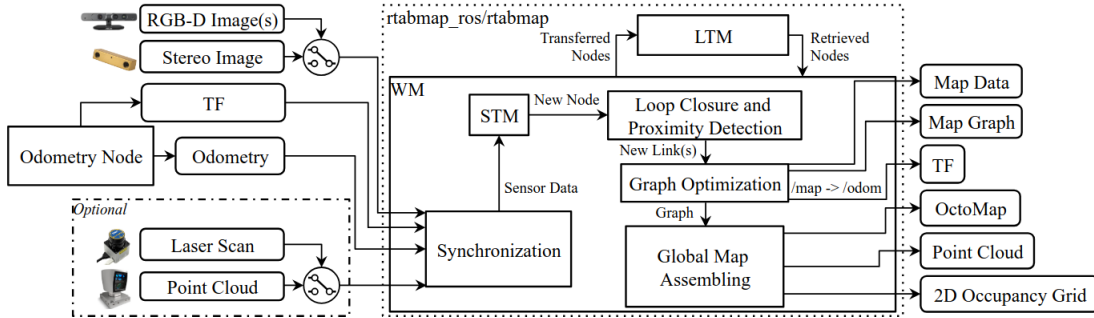


FIGURE 3.20: `rtabmapnode` (Labbé and Michaud 2019).

Beyond odometry and graph optimization, a central component of RTAB-Map is its memory management system, which governs how information is retained for long-term online SLAM. As we can see, `rtabmap` stores the input data in the Working Memory (WM), where it performs Loop Closure, Graph Optimization and Global Map assembly, and stores less relevant features in its Long-Term Memory (LTM), this is done with heuristics that prioritize locations by observation time: sites observed for longer durations are deemed more important and are therefore kept in WM, allowing for long-term online SLAM in large environments (Labbé and Michaud 2019).

The previously discussed RTAB-Map odometry is the output of the *rgbd_odometry* node, whose block diagram is shown in Figure 3.21.

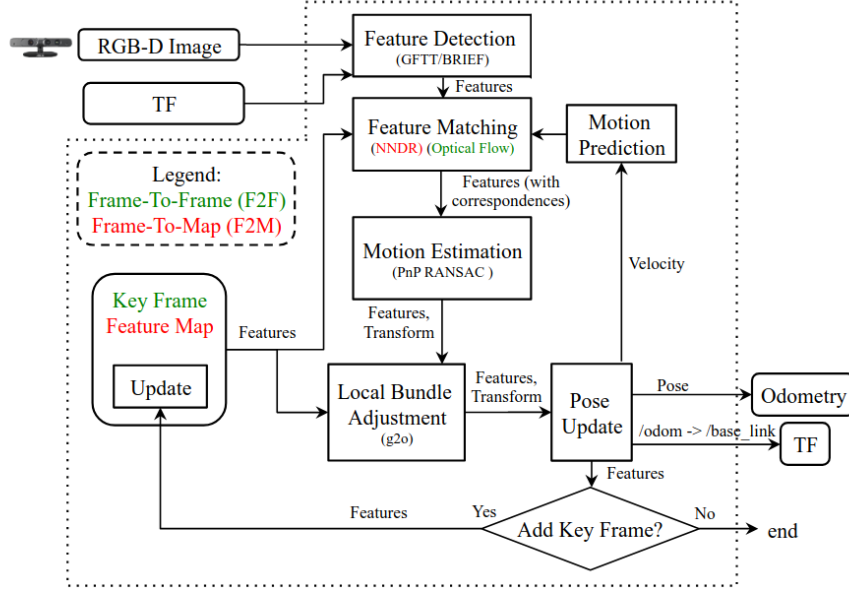


FIGURE 3.21: *rgbd_odometry* node (Adapted from Labbé and Michaud 2019).

For visual odometry, two standard strategies (Scaramuzza and Fraundorfer 2011) are widely used: *frame-to-frame* (F2F) and *frame-to-map* (F2M). The key distinction is the registration target: F2F aligns the current frame to the most recent keyframe, whereas F2M aligns the current frame to a locally maintained feature map aggregated from several past keyframes (Labbé and Michaud 2019). In this work we adopt F2M because it offers greater robustness to brief occlusions, rapid rotations, and viewpoint changes, and it more readily recovers from short tracking losses. The additional computational cost is acceptable given our available resources.

Runtime configuration For runtime execution, the ZED 2i camera operated at VGA (640x480) resolution with a frame rate of 15Hz, producing synchronized RGB and depth images as well as IMU measurements. Visual odometry was configured in *frame-to-map* (F2M) mode, using ORB features (up to 1000 per frame) with a minimum of 20 inliers required for motion estimation. Bundle adjustment was enabled, and IMU deskewing was applied during initialization to improve robustness. The main rtabmap node employed ORB features as well (500 maximum), with loop closure detection enabled using a bag-of-words dictionary and a geometric verification threshold of 0.11. Graph optimization was performed online using the g2o backend.

Memory management used a short-term buffer of 10 nodes, with older data transferred to long-term memory. To mitigate spurious features from water reflections near the RHIB, a region-of-interest (ROI) mask was applied using the `Kp/RoiRatios` parameter. This parameter is defined in normalized image coordinates (0-1) for *top*, *bottom*, *left*, *right*. In our configuration (0.0 0.15 0.0 0.0), the bottom 15% of each frame was excluded from feature extraction, while the remainder of the image was retained. This aims to remove some of the dynamic water band from the input without losing too much pier/ship information, improving odometry robustness by reducing erroneous water feature matches. A screenshot of RTAB-Map’s Graphical User Interface (GUI), in Figure 3.22 shows the program running, with a Loop Closure Detection (top left), Visual Odometry (bottom left) and 3D Map (right) views. In the visual odometry view, the colored points represent identified features, where green are inliers, meaning they can be used to calculate the visual odometry, yellow are features without corresponding matches in previous frames, and red are outliers. In the 3D Map view, the generated point cloud is visible with the original object colors, as well as the feature points in yellow and the camera path in the form of a blue line.

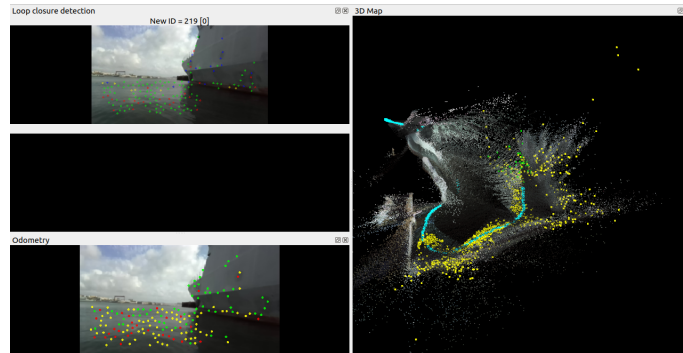


FIGURE 3.22: RTAB-Map GUI showing SLAM on BNL dataset.

3.3.2.2 RTAB-Map Integration - Pohang Canal

For the Pohang Canal dataset, RTAB-Map was deployed in stereo mode rather than RGB-D mode, as only rectified left-right image pairs were available. The system configuration therefore differed from the BNL experiments.

The `stereo_odometry` node, pictured in Figure 3.23, replaced the `rgbd_odometry` front-end, estimating camera motion directly from stereo correspondences without relying on a registered depth image or IMU priors.

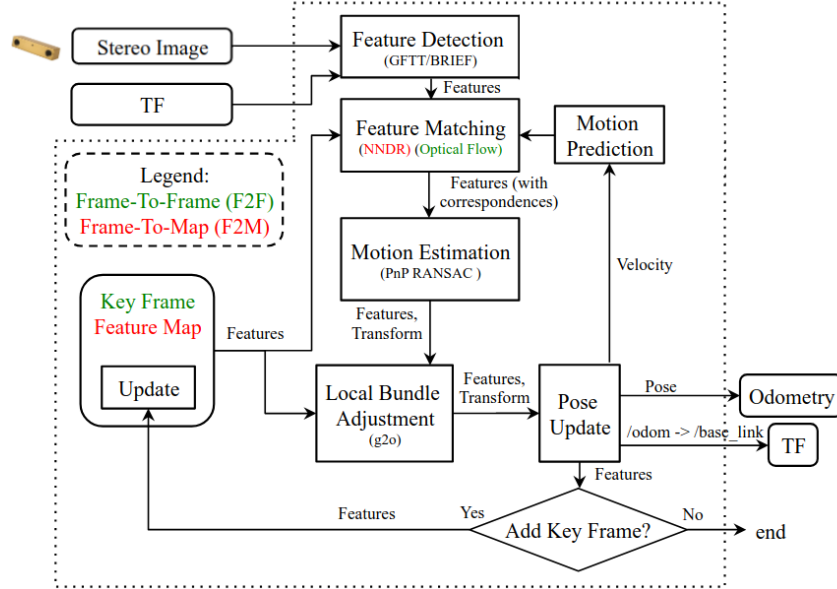


FIGURE 3.23: `stereo_odometry` node (Adapted from Labbé and Michaud 2019).

This reduced the available sensor modalities but preserved the overall SLAM pipeline: depth was calculated from image pairs using SGBM, features were extracted and matched frame-to-map, incremental poses were forwarded to the main `rtabmap` node, and loop closure detection and graph optimization were performed identically to the RGB-D case. As shown in Figure 3.24, RTAB’s visualization tool’s layout is similar to before keeping the same Loop Closure, Visual Odometry and 3d Map views.

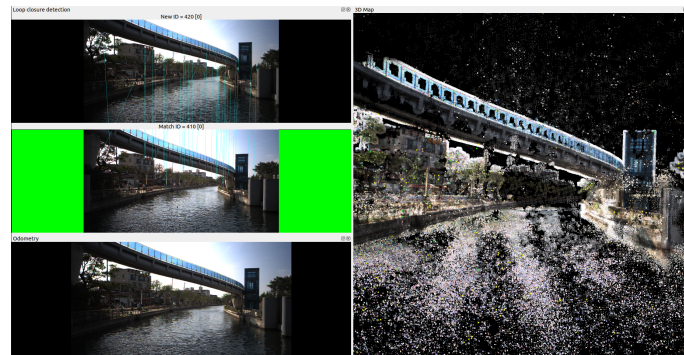


FIGURE 3.24: RTAB-Map GUI showing SLAM on Pohang Canal Dataset.

3.3.2.3 ZED SDK SLAM

The proprietary SLAM implementation provided by the ZED SDK was also evaluated, as to have a comparison for the RTAB-Map results. The ZED SDK SLAM relies on the proprietary Positional Tracking module to implement Visual-Inertial Odometry, and the Spatial Mapping module to create a 3D map as well as perform its version of loop closure, through what is referred to as Area Memory (Stereolabs 2024b). According to Stereolabs support, the underlying algorithms are the result of internal R&D and are not publicly disclosed (Myzhar 2024). Consequently, this work treats the ZED SLAM system as a black box, using the stereo images and onboard IMU data as inputs, and recording the estimated trajectory and spatial map as outputs.

1. **Sensing and Pre-processing:** The camera provides RGB frames, and IMU measurements. Pre-processing generates a *registered* depth image from the stereo pair.
2. **Visual-inertial fusion:** Positional tracking integrates visual odometry with filtered IMU measurements to improve robustness.
3. **Positional tracking and spatial mapping:** The SDK provides real-time six-degree-of-freedom (6-DoF) pose estimation together with 3D spatial mapping capabilities.

Runtime Configuration The ZED 2i was set to acquire stereo pairs at VGA resolution (640×480) with an internal grab rate of 15 Hz. Depth estimation used the `NEURAL_LIGHT` mode with a valid range of 0.3-40.0 m. Visual-inertial positional tracking was enabled with IMU fusion active; loop-closure support was turned on via Area Memory with drift resets on loop closures. Gravity alignment was enabled. Spatial mapping resolution and range were configured to 0.1 m and 40 m, respectively.

Displayed in Figure 3.25 is the RViz2 GUI showing the selected ZED SDK data output through the ZED ROS2 wrapper. In this Figure we can identify the rectified stereo pair (top), the depth image (bottom left), the 3D view (bottom right), where we see the point cloud generated for that frame, the camera path in the green line, and the current camera pose represented by the red arrow.

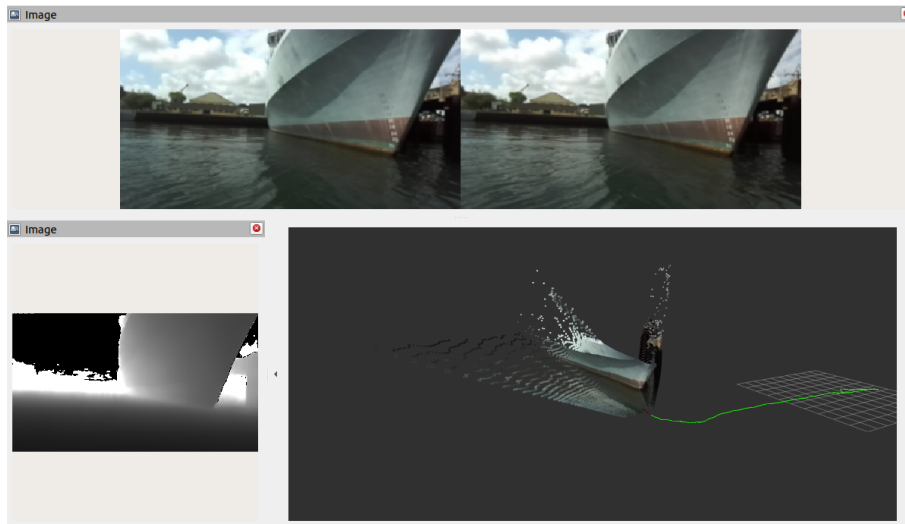


FIGURE 3.25: RViz2 visualization of ZED SDK SLAM on the BNL dataset.

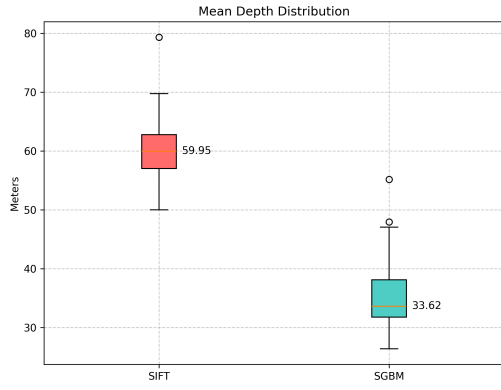
Chapter 4

Experimental Results

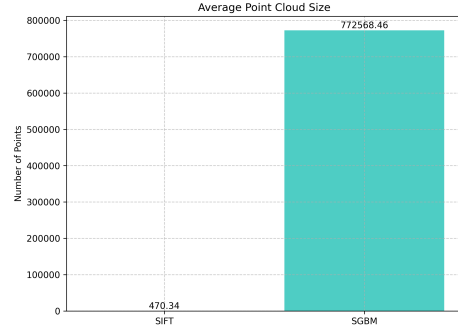
This chapter presents the experimental results obtained from collecting the BNL dataset and applying the stereo vision and SLAM algorithms described previously to the Pohang Canal and BNL datasets. The results are analyzed to assess their effectiveness in real-world maritime environments, focusing on navigation accuracy and map quality.

4.1 Stereo images of the Pohang Canal Dataset

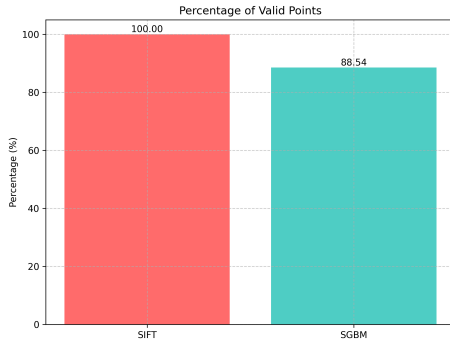
After running both algorithms on a subset of 50 stereo image pairs from the Pohang Canal Dataset, the key metrics of mean depth, point cloud size, percentage of valid points, and average execution time to process each image pair were computed. Figure 4.1 summarizes the averages across the dataset, providing a direct comparison between SGBM and SIFT.



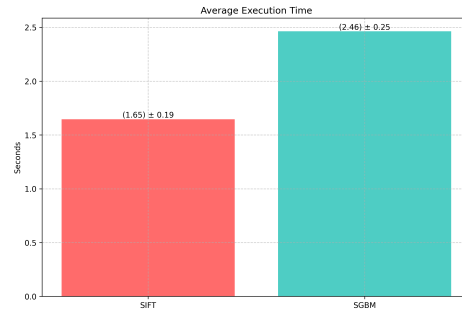
(A) Mean depth distribution (The main box is centered on the average, it's height is twice the standard deviation. The thin horizontal lines represent the minimum and maximum values, with the exception of the circles that represent outliers not used for computation.



(B) Average point cloud size.



(C) Average valid points percentage.



(D) Average execution time.

FIGURE 4.1: Comparison of SGBM and SIFT: (a) mean depth, (b) average point cloud size, (c) valid points percentage, and (d) average execution time.

As expected, given the nature of both algorithms, SGBM produced much denser point clouds, but at the expense of increased execution time and noisier estimates in low-texture regions such as water surfaces, resulting in a lower percentage of valid depth points. In contrast, SIFT reconstructions were sparser but more robust to texture variations, with faster runtimes and consistent feature geometry. The execution time difference (2.46 s for SGBM vs. 1.65 s for SIFT, both with small

standard deviations) confirms that SGBM requires more computation. Nevertheless, even the more resource-intensive SGBM remains computationally light enough to be feasible for online SLAM, which motivates its use for generating dense maps of the Pohang Canal dataset in subsequent experiments.

4.2 Simultaneous Localization and Mapping

This section presents the outcomes of the SLAM algorithms tested on the selected datasets, extending the stereo vision results into full localization and mapping pipelines. The analysis focuses on localization accuracy, map consistency, and overall robustness under varying maritime conditions, first using the Pohang Canal dataset and subsequently the BNL dataset.

Before evaluation, it is necessary to account for the fact that SLAM and groundtruth trajectories are expressed in different coordinate frames and thus need to be adjusted before comparison (Zichao Zhang and Scaramuzza 2018). A direct comparison of trajectories without prior alignment is meaningless, since global rotations and translations would dominate the error. To address this, we apply a Singular Value Decomposition (SVD) rigid-body transformation to all SLAM trajectories to align them with their respective ground truth (Arun et al. 1987). This ensures that evaluation metrics reflect only the relative discrepancies in trajectory shape and scale, rather than arbitrary coordinate frame differences.

4.2.1 Pohang Canal Dataset

RTAB-Map was run on part of the canal sequence (from image pair 1300 to 2300) and its poses were time-aligned to the dataset epoch. For evaluation of SLAM on the Pohang Canal dataset, we used a fused baseline trajectory produced by the original authors of the dataset by fusing GPS, AHRS, and LiDAR data into a graph SLAM framework using the GTSAM library (Chung et al. 2023). The fused baseline was then interpolated at the SLAM timestamps. We compute the trajectory on the resulting fitted trajectory. Figure 4.2 illustrates both the ground truth and SLAM trajectories and the XY planar error over time. Table 4.1 summarizes the evaluation metrics.

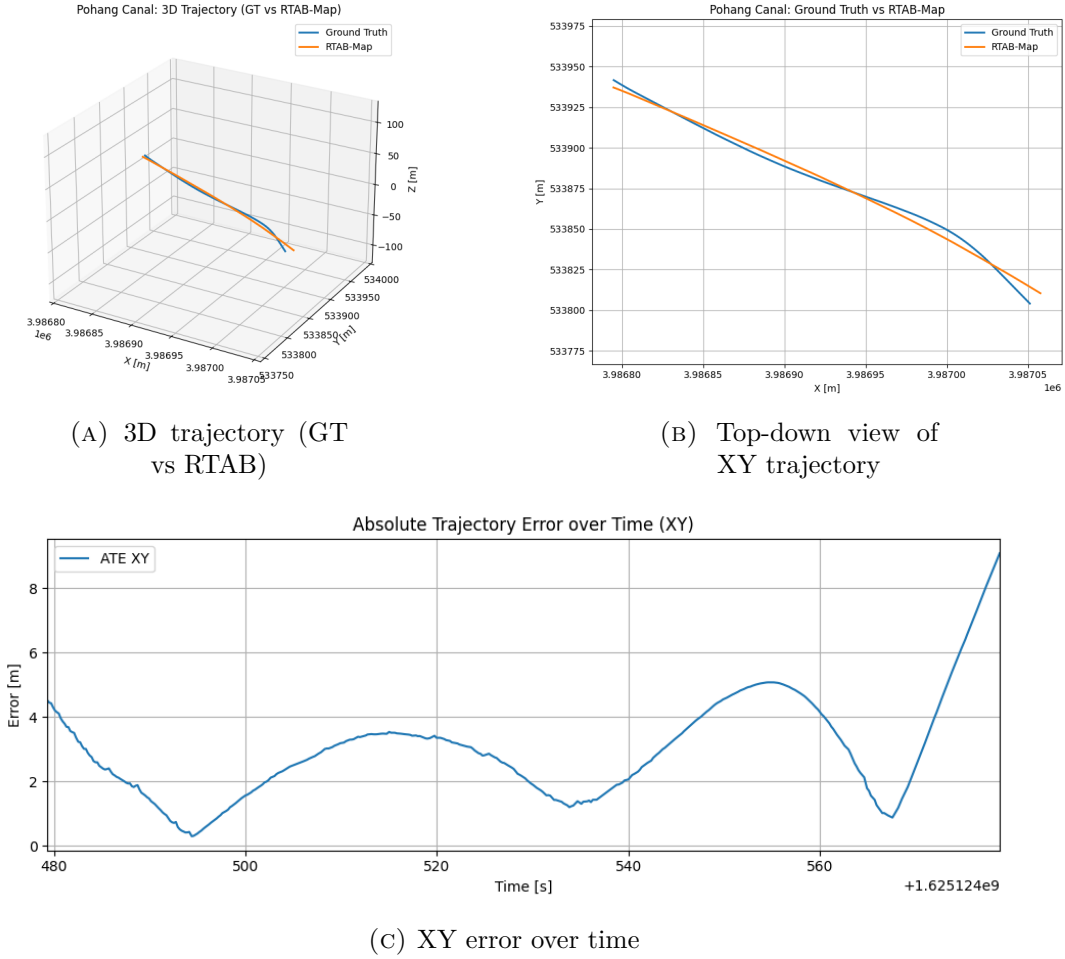


FIGURE 4.2: RTAB-Map results on the Pohang Canal dataset (images 1300-2300).

TABLE 4.1: RTAB-Map vs. fused baseline on Pohang Canal.

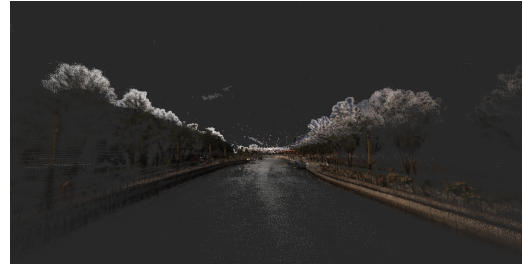
Segment	Mean [m]	Median [m]	RMSE [m]	Max [m]
1300-2300	2.93	2.86	3.30	9.08

Over the 371 evaluated pose pairs, we obtain a mean planar XY error of 2.93 m, RMSE of 3.30 m, and a maximum error of 9.08 m. The 3D values match the planar XY, as there is negligible residual vertical discrepancy, as expected in a surface vehicle, especially inside the canal. Qualitatively, RTAB-Maps trajectory has smoother curves visible in Figure 4.2, failing to accompany the rotation of the vessel to its full extent.

The generated 3D map accurately resembles the original scene, allowing for a clear representation of the canal limits, as illustrated in Figure 4.3.



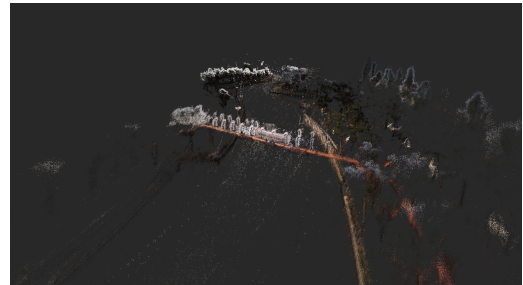
(A) Left image 1400
from the dataset.



(B) Canal view in
the generated point-
cloud.



(c) Left image 1925
from the dataset.



(D) Floating 3D view
capturing the bridge
and canal in the gen-
erated pointcloud (a
simple rotation and
translation would
align the generated
map with the view
in image (c).

FIGURE 4.3: Two images taken from the Pohang Canal dataset compared to two screenshots of the generated pointcloud.

4.2.2 BNL Dataset

To quantitatively assess SLAM performance on the BNL dataset, both RTAB-Map and ZED SDK trajectories were aligned to GPS ground truth and evaluated frame by frame. The RTAB-Map trajectory was obtained from RTAB-Map’s `databaseViewer` tool after refining links and detecting additional loop closures, ensuring that the exported poses reflected a globally optimized graph. The ZED SDK trajectory was extracted from the ROS 2 bag using a custom parser for the `/zed/zed_node/pose` topic. GPS data was converted from latitude, longitude, and altitude (WGS84) to a local East-North-Up (ENU) frame using `pyproj`, and interpolated to the SLAM timestamps to ensure temporal alignment.

Both the SLAM and GPS ENU trajectories are relative only to themselves, with no common origin or reference point. In order to compare them, all SLAM trajectories were spatially aligned to GPS by applying a SVD based rigid transformation, in other terms, a geometrical operation that changes a shape’s position or orientation but preserves its size and shape. This operation consisted of calculating the singular vectors and singular values. The singular vectors were aligned to ensure consistent orientation between the calculated and ground truth trajectories. The singular values were not used directly, as the overall scale was object of evaluation. The distance between the camera and the GPS antenna was also accounted for and compensated. Localization accuracy was then quantified as the Euclidean distance between SLAM and GPS positions.

4.2.2.1 Processing the BNL dataset with RTAB-Map

The RTAB-Map SLAM results for the BNL A and B trajectories were aligned to the GPS ground truth and evaluated both quantitatively and qualitatively.

BNL A Figure 4.4 shows the RTAB-Map trajectory compared with GPS for BNL A. The system maintained relatively close agreement with ground truth, with mean planar error of 5.36 m, RMSE of 6.05 m, and a maximum deviation of 12.27 m. Errors accumulated gradually but remained bounded, showing the few loop-closure detections that occurred were not sufficient to correct drift over time. Qualitatively, the top-down trajectory demonstrates that RTAB-Map was able to reproduce the overall navigation path with good fidelity, while error-over-time plots confirm bounded drift throughout the run.

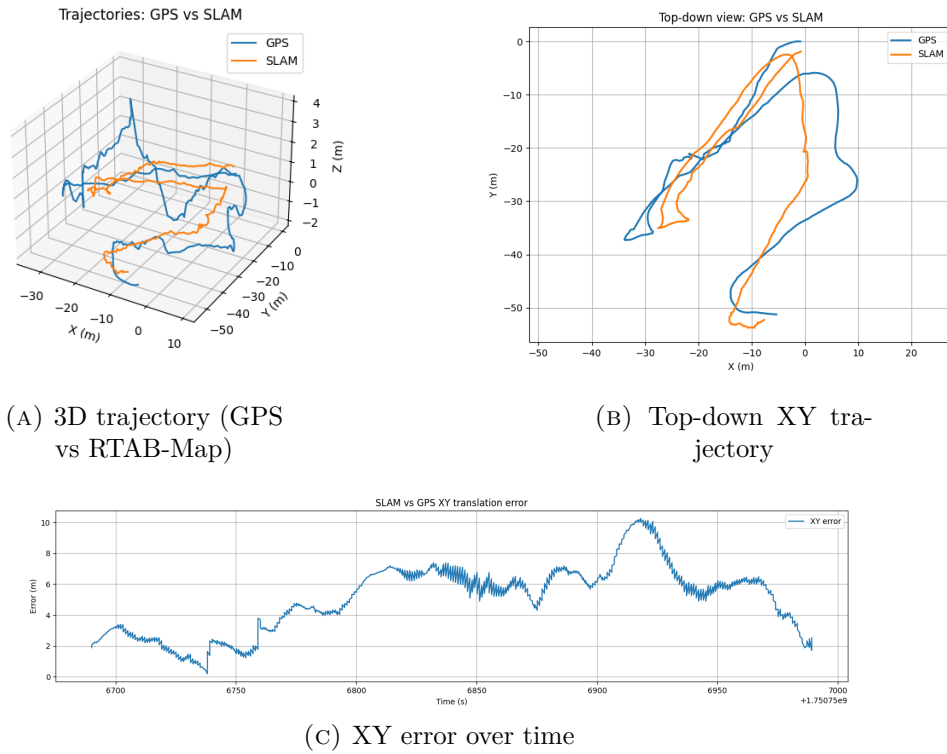
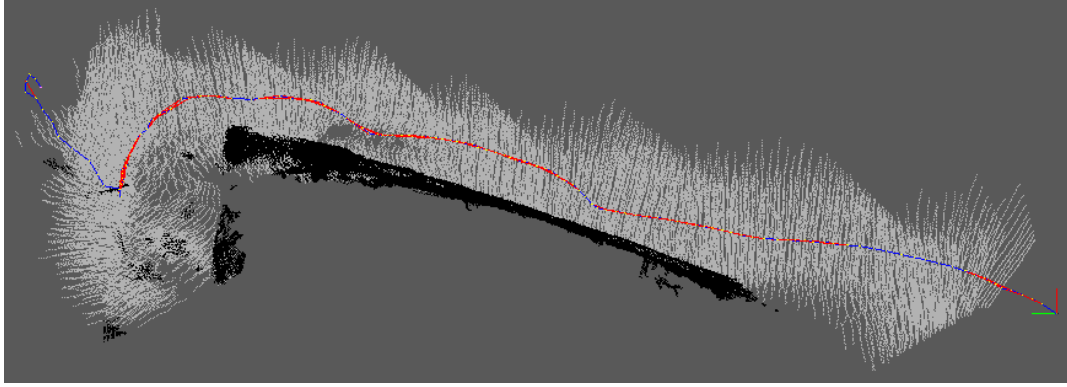


FIGURE 4.4: RTAB-Map results on BNL A dataset: GPS vs SLAM trajectory alignment and error metrics.

For this particular trajectory, we can also compare the generated graph to the shape of the vessel docked in the pier, as seen in Figure 4.5. This was done by rotating and scaling the graph to match the satellite image, and then the detected object (vessel) was highlighted in red.



(a) Partial RTAB generated graph with camera path (red line)



(A) Satellite image of the vessel



(B) Graph overlaid on satellite image

FIGURE 4.5: Overlay of RTAB-Map generated graph on satellite imagery of the mapped vessel.

Overall, we can observe that the 3D map is able to represent the original scene, although the point cloud has more noise than in the Pohang case, especially around object edges, as illustrated in Figure 4.6.



(A) Image from the BNL A dataset.



(B) View of the captured vessel in the generated pointcloud.

FIGURE 4.6: An image captured in the BNL A dataset compared to a screenshot of the generated pointcloud.

BNL B Results for BNL B are presented in Figure 4.7. In this case, performance degraded relative to BNL A: the mean planar error increased to 11.20 m with an RMSE of 12.00 m, and a maximum error of 19.69 m. A pronounced spike in 3D error was observed in BNL B. Inspection of the GPS record shows a short burst of faulty altitude readings (altitude jumping from +7 m to +49 m and then to -11.9 m within ≈ 5 s), while the planar XY track remained smooth. We know this vertical outlier that produced a large 3D discrepancy was a GPS error, as the RHIB stayed on the surface for the duration of the recording. Despite this, the planar XY alignment preserved the general navigation pattern. The reduced SLAM accuracy is attributed to adverse visual conditions, including water reflections and limited shoreline features, which reduced loop-closure opportunities and constrained graph optimization effectiveness.

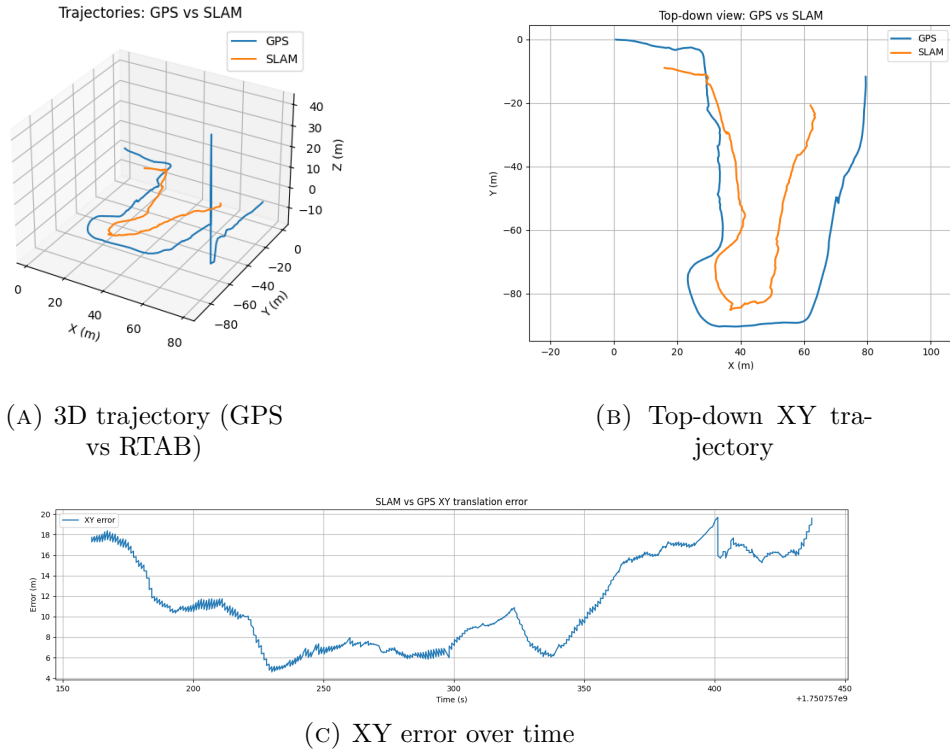


FIGURE 4.7: RTAB-Map results on BNL B dataset: GPS vs SLAM trajectory alignment and error metrics.

Across both BNL trajectories, RTAB-Map demonstrated the ability to generate globally optimized trajectories in challenging maritime conditions. In the more favorable BNL A sequence, the mean error remained in the 5-6 m range with good loop-closure corrections, yielding a trajectory closely matching GPS. In contrast, BNL B highlighted the sensitivity of the system to degraded visual conditions, where water reflections and lack of stable features as well as a longer traveled path led to larger deviations and occasional failures, with errors rising above 10 m. These results indicate that while RTAB-Map can achieve meter-level accuracy in structured environments, its performance in maritime settings is strongly dependent on the availability of consistent visual cues for loop closure and graph optimization.

An example comparison between a BNL B dataset image and the generated point cloud is illustrated in Figure 4.8



(A) Image from the BNL B dataset.



(B) Floating 3D view of the captured vessel and pier in the generated point cloud.

FIGURE 4.8: An image captured in the BNL B dataset compared to a screenshot of the generated point cloud.

uod

4.2.2.2 Processing the BNL dataset with ZED SDK

The ZED SDK SLAM results for the BNL A and B trajectories were aligned to the GPS ground truth and evaluated both quantitatively and qualitatively.

BNL A Figure 4.9 presents the ZED SDK trajectory compared to GPS ground truth for the BNL A dataset. Quantitative evaluation showed a mean planar error of 8.33 m, with RMSE of 8.65 m, and a maximum deviation of 16.36 m. Full 3D errors were similar, with a mean of 8.88 m and a maximum of 16.76 m.

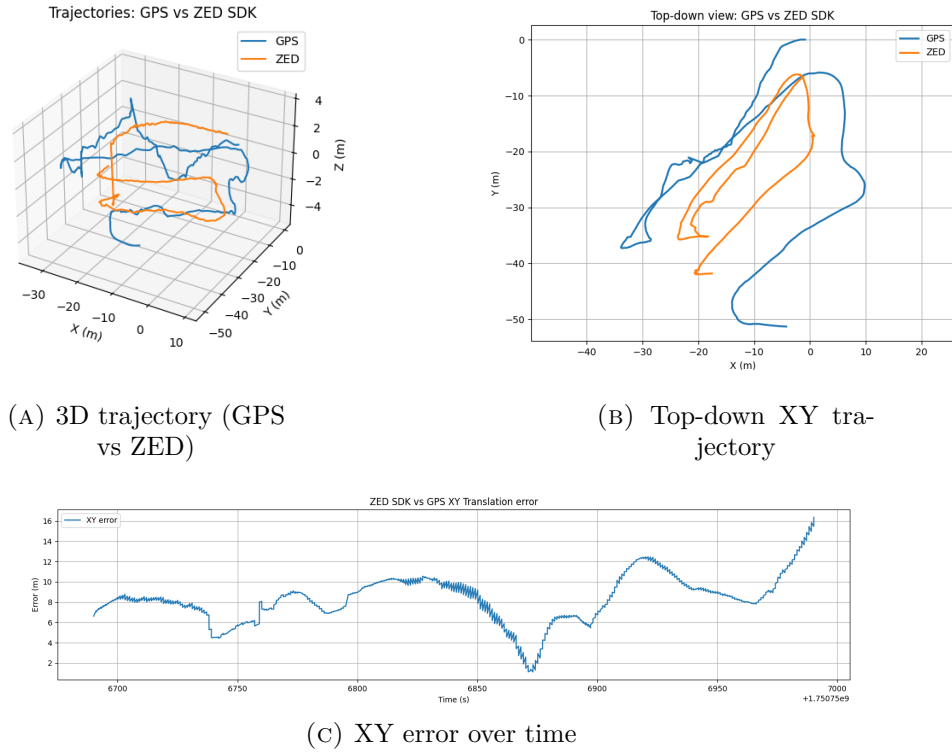


FIGURE 4.9: ZED SDK SLAM results on BNL A dataset: GPS vs ZED trajectory alignment and error metrics.

The error-over-time plots show relatively smoother behavior without significant divergence, indicating that the onboard visual-inertial fusion provided stable pose tracking. A notable result, however, is an apparent altitude drop in the ZED trajectory, which does not correspond to actual vessel motion. This vertical bias amplifies the 3D error in one segment of the run, as seen in Figure 4.10. Qualitatively, the ZED trajectory preserved the general shape of the route but failed to align closely with GPS, especially in long straight segments. This suggests that while the SDK’s closed-form odometry is robust to noise and drift in XY, it struggles to maintain global consistency and reliable altitude estimates in low-feature maritime scenes.

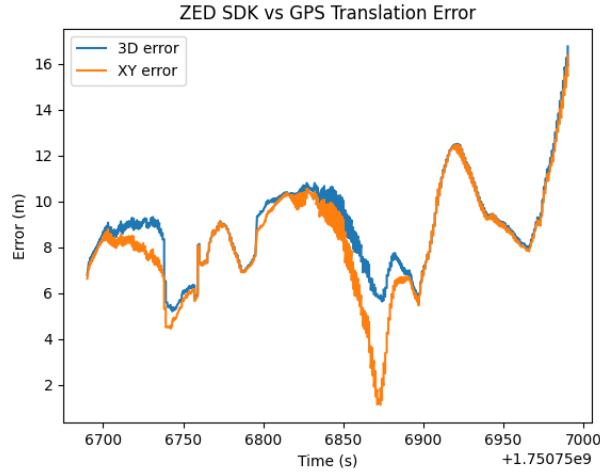


FIGURE 4.10: ZED SDK SLAM vs GPS 3D error over time.

The gradual downward drift illustrated in Figure 4.9 (A) suggests that the Visual-Inertial Odometry is overweighting short-term vertical accelerations from the IMU and, in the absence of strong visual constraints along z , does not fully return to the original vertical plane after each heave.

BNL B Figure 4.11 shows the ZED SDK trajectory compared with GPS for the BNL B dataset. Quantitative evaluation indicated degraded performance compared to BNL A: the mean planar error rose to 13.02 m, with RMSE of 13.63 m and a maximum deviation of 19.82 m.

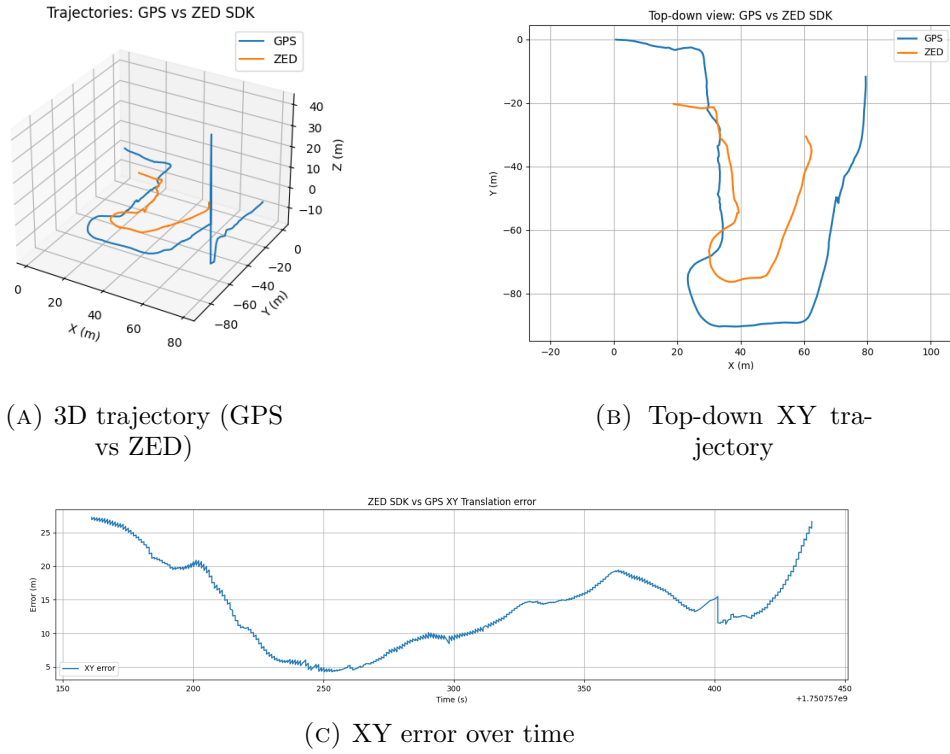
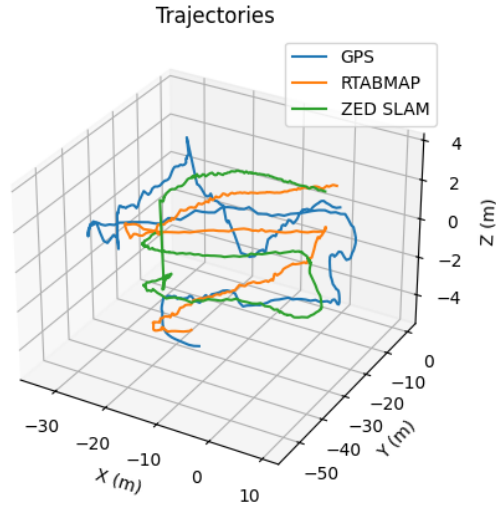


FIGURE 4.11: ZED SDK SLAM results on BNL B dataset: GPS vs ZED trajectory alignment and error metrics.

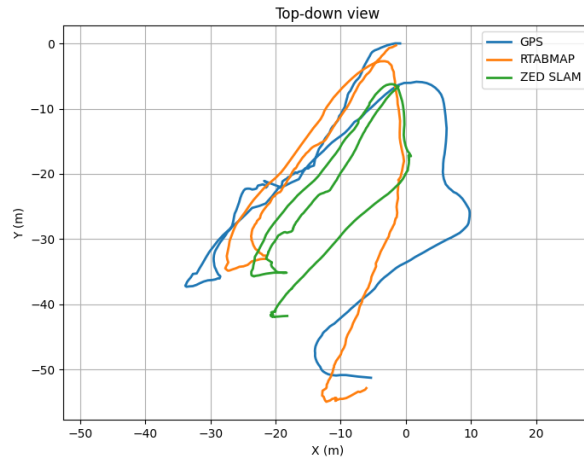
As in BNL A, the ZED trajectory remained smooth without intense jumps, but suffered from systematic drift relative to GPS. The same spurious vertical spike is again present in the GPS trajectory, causing brief amplification of 3D error, while the planar XY error profile remained more representative of true navigation accuracy. Qualitatively, the SDK preserved the overall shape of the route but underestimated path curvature and straight-line segments, leading to compression of the estimated trajectory relative to ground truth. This behavior indicates that the closed-source visual-inertial odometry is stable for local motion but less effective at correcting long-term drift in feature-poor maritime environments.

4.2.2.3 Comparison

Both RTAB-Map and ZED SDK SLAM were quantitatively compared against GPS ground truth. For BNL A, Figure 4.12 overlays the three trajectories in 3D and in top-down XY view, while Table 4.2 summarizes the localization error metrics. The same is done for BNL B in Figure 4.13 and Table 4.3.



(A) 3D trajectories (GPS vs RTAB-Map vs ZED SDK)

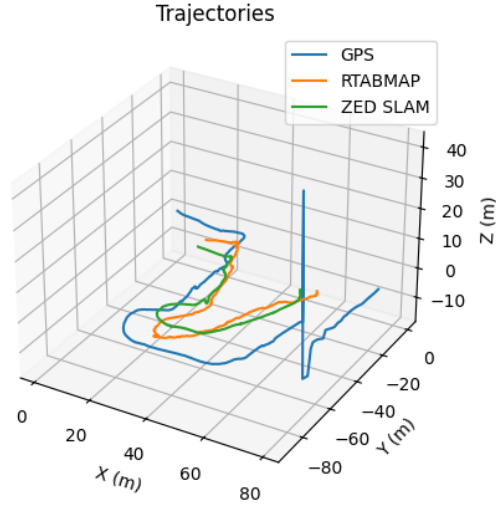


(B) Top-down XY trajectories

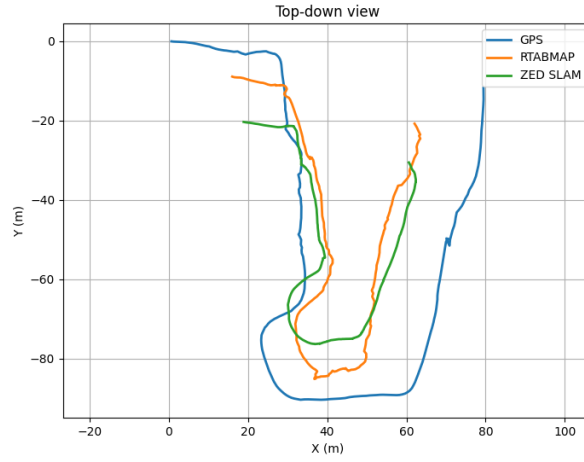
FIGURE 4.12: Comparison of GPS, RTAB-Map, and ZED SDK SLAM trajectories on BNL A dataset.

TABLE 4.2: Comparison of localization error metrics for RTAB-Map and ZED SDK on BNL A dataset.

	Mean [m]	Median [m]	RMSE [m]	Max [m]
RTAB-Map (3D)	5.66	5.84	6.17	12.27
RTAB-Map (XY)	5.36	5.72	6.05	12.27
ZED SDK (3D)	8.85	8.84	9.05	16.39
ZED SDK (XY)	8.29	8.35	8.61	15.98



(A) 3D trajectories (GPS vs RTAB-Map vs ZED SDK)



(B) Top-down XY trajectories

FIGURE 4.13: Comparison of GPS, RTAB-Map, and ZED SDK SLAM trajectories on BNL B dataset.

TABLE 4.3: Comparison of localization error metrics for RTAB-Map and ZED SDK on BNL B dataset.

	Mean [m]	Median [m]	RMSE [m]	Max [m]
RTAB-Map (3D)	11.692	10.535	12.963	47.569
RTAB-Map (XY)	11.198	10.462	11.998	19.694
ZED SDK (3D)	14.509	14.553	15.992	45.435
ZED SDK (XY)	13.983	14.251	15.242	27.285

4.2.3 Discussion

All methods enabled the successful three-dimensional reconstruction of the scenario, with no substantial differences observed between the point cloud quality generated by RTAB-Map and the ZED system. Both approaches provided a clear representation of the captured environment; however, some noise was present, particularly along object boundaries and over the water surface. Quantitatively, while RTAB-Map consistently outperformed the ZED SDK in the BNL trials, the margin of improvement was moderate, suggesting that the maritime setting characterized by dynamic water surfaces, weak texture, and scarce structural cues was the dominant factor limiting performance. By contrast, the Pohang Canal results were much better. Beyond using a high-quality fused groundtruth for comparison, the canal sequence provided more favorable conditions for visual SLAM: a stable imaging platform with a wider stereo baseline positioned higher above the water improved image quality and depth estimation; calmer waters reduced specular highlights and texture loss while minimizing unpredictable platform motion due to waves. The built environment observed on both sides of the vessel offered repeated, rigid structures that supported loop closures and strengthened graph constraints. In the BNL dataset, the small-baseline sensor, platform heave/roll, and visually poor shorelines, mostly captured on one side of the vessel at a time, captured from relatively low height, reduced match quality and loop-closure opportunities, amplifying drift. Taken together, these findings indicate that, in surface navigation, both graph-based SLAM and visual-inertial odometry are primarily constrained by maritime scene characteristics; therefore, gains are more likely to come from environment-aware design (sensor placement, larger baseline, motion damping, region of interest masking, and loop-closure priors) than from algorithm choice alone.

Conclusion

Conclusions

This thesis addressed the problem of enabling autonomous navigation for Uncrewed Surface Vessels operating in uncharted or GNSS-denied port environments. The central objective was to evaluate the feasibility of stereo visual SLAM as a passive and low-power alternative to active sensing modalities such as LiDAR or radar, which are less suitable for covert or constrained naval operations.

To achieve this, two stereo matching approaches were implemented and tested - dense SGBM and feature-based SIFT triangulation. Stereo SLAM with RTAB-Map was evaluated on both the Pohang Canal and the BNL datasets. By contrast, the ZED SDK's proprietary visual-inertial SLAM was assessed only on the BNL dataset, using the ZED 2i's onboard stereo and IMU.

The main results can be summarized as follows:

- **Stereo vision:** SGBM produced dense point clouds but suffered from noise in low-texture regions and incurred a higher computational cost, whereas SIFT yielded sparser yet geometrically stable reconstructions with faster runtimes. However, the runtime advantage that the SIFT method gains by losing scene information was not large enough to preclude using the dense method within the SLAM pipeline.
- **SLAM:** RTAB-Map consistently outperformed the ZED SDK, particularly in structured environments. In BNL A, it achieved 5-6 m mean planar error, roughly twice as accurate as the ZED SDK under the same conditions. In BNL B, accuracy degraded due to poor visual conditions, with errors above 10 m. By contrast, Pohang Canal results were markedly better, with errors consistently below 3-4 m, reflecting more favorable conditions (wider baseline, stable platform, calmer waters, densely populated scene).
- **Operational implications:** These findings demonstrate that stereo visual SLAM can deliver meter-level localization accuracy, which may be sufficient

for navigation in constrained port environments. However, performance is strongly dependent on environmental factors such as platform stability, sensor baseline, and scene texture, often more so than on the choice of algorithm itself.

Despite achieving the stated objectives, a minor practical constraint was the late availability of the stereo camera hardware. Initial experiments therefore used public datasets, which provided a solid baseline and ensured continuity; subsequent BNL trials validated an adapted pipeline on our own data. Moreover, the lack of a true USV platform was mitigated by simulating one with a RHIB. Environmental challenges such as water reflections and feature scarce shorelines significantly affected reconstruction quality. These constraints influenced both the design and results of the experiments, but also provided valuable insights into the specific difficulties of maritime SLAM.

Future Work

The research carried out in this thesis opens several avenues for further investigation:

- **Extended environments:** Testing in larger ports, different basins, or open-sea conditions, including at night or in adverse weather, would broaden operational validation and stress-test system robustness.
- **Real-time implementation:** Profiling and optimizing runtime performance on embedded platforms is necessary to transition from offline evaluation to closed-loop navigation in deployed USVs.
- **Swarm capability:** Cooperative operation of multiple USVs can accelerate coverage and mapping, enable cross-platform loop closures and multi-view geometry for improved accuracy, and provide redundancy against individual sensor failures. From an operational perspective, coordinated swarms also offer a force-multiplication effect for surveillance or collective firepower.
- **Semantic mapping:** Incorporating object-level information (e.g., recognition and identification of ships, docks, and buoys) could improve loop closure opportunities and enable mission-specific tasks such as collision avoidance or target identification and striking.

By addressing these directions, future work can build upon the contributions of this thesis to develop more robust, operationally relevant stereo SLAM

solutions for uncrewed maritime platforms. Together, these efforts would bring the deployment of fully autonomous USVs in GNSS-denied and tactically constrained naval environments closer to reality.

Bibliography

- Arun, K. Somani, T. S. Huang, and S. D. Blostein (1987). “Least-squares fitting of two 3-D point sets”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9.5, pp. 698–700. DOI: 10.1109/TPAMI.1987.4767965.
- Balmforth, Tom (Aug. 2023). “Russian navy vessel damaged in drone attack - Ukrainian source”. In: Accessed: 2025-08-29. URL: <https://www.reuters.com/world/europe/russian-navy-vessel-damaged-drone-attack-ukrainian-source-2023-08-04/>.
- BBC News (May 2024). “Russia accused of widespread GPS disruption in Europe”. In: *BBC News*. URL: <https://www.bbc.com/news/articles/ckgn47e5qyno> (visited on 08/31/2025).
- Bentley, Jon Louis (1975). “Multidimensional binary search trees used for associative searching”. In: *Communications of the ACM* 18.9, pp. 509–517.
- Bovcon, Borja, Janez Perš, Matej Kristan, et al. (2018). “Stereo obstacle detection for unmanned surface vehicles by IMU-assisted semantic segmentation”. In: *Robotics and Autonomous Systems* 104, pp. 1–13.
- Cadena, Cesar, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard (2017). “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age”. In: *IEEE Transactions on robotics* 32.6, pp. 1309–1332.
- Campos, Carlos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós (2021). “Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam”. In: *IEEE transactions on robotics* 37.6, pp. 1874–1890.
- Canonical Ltd. (Sept. 2024). *Ubuntu 22.04.5 LTS (Jammy Jellyfish)*. Accessed: 2025-08-30. URL: <https://releases.ubuntu.com/jammy/>.
- Chiu, Leo and Christopher Stewart (2022). *Analysis: Off Target – Are Kyiv’s GPS-Aided Weapons Susceptible to Russian Electronic Warfare?* Accessed: 2025-06-22. URL: <https://www.kyivpost.com/analysis/33448>.
- Chung, D., J. Kim, C. Lee, and J. Kim (2023). “Pohang canal dataset: A multimodal maritime dataset for autonomous navigation in restricted waters”. In: *The*

- International Journal of Robotics Research* 42.12, pp. 1104–1114. DOI: 10.1177/02783649231191145.
- Cordts, Marius, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele (2015). “The cityscapes dataset”. In: *CVPR Workshop on the Future of Datasets in Vision*. Vol. 2, p. 1.
- Durrant-Whyte, Hugh and Tim Bailey (2006). “Simultaneous localization and mapping: part I”. In: *IEEE robotics & automation magazine* 13.2, pp. 99–110.
- Garmin (Jan. 2024). *GPS 18x Technical Specifications*. Document ID: 190-00879-08, Revision F, Accessed: 2025-06-21. URL: https://static.garmin.com/pumac/GPS_18x_Tech_Specs.pdf.
- Geiger, Andreas, Philip Lenz, Christoph Stiller, and Raquel Urtasun (2013). “Vision meets robotics: The kitti dataset”. In: *The international journal of robotics research* 32.11, pp. 1231–1237.
- Grisetti, Giorgio, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard (2011). “A tutorial on graph-based SLAM”. In: *IEEE Intelligent Transportation Systems Magazine* 2.4, pp. 31–43.
- Hamzah, Rostam Affendi and Haidi Ibrahim (2016). “Literature survey on stereo vision disparity map algorithms”. In: *Journal of Sensors* 2016.1, p. 8742920.
- Hariyama, Masanori, Naoto Yokoyama, and Michitaka Kameyama (2008). “Design of a trinocular-stereo-vision VLSI processor based on optimal scheduling”. In: *IEICE transactions on electronics* 91.4, pp. 479–486.
- Hartley, Richard (2003). *Multiple view geometry in computer vision*. Vol. 665. Cambridge university press.
- Hecht, Eugene (2017). *Optics*. 5th. Pearson. ISBN: 978-0134042882.
- Hirschmuller, Heiko (2007). “Stereo processing by semiglobal matching and mutual information”. In: *IEEE Transactions on pattern analysis and machine intelligence* 30.2, pp. 328–341.
- Kazerouni, Iman Abaspor, Luke Fitzgerald, Gerard Dooly, and Daniel Toal (2022). “A survey of state-of-the-art on visual SLAM”. In: *Expert Systems with Applications* 205, p. 117734.
- Labbé, Mathieu (2015). *Simultaneous Localization and Mapping (SLAM) with RTAB-Map*. Presentation slides. IntRoLab, 3IT, Département de Génie Électrique et Génie Informatique, Université de Sherbrooke. URL: <https://introlab.3it.usherbrooke.ca/images/3/31/Labbe2015ULaval.pdf>.

- Labbé, Mathieu and François Michaud (2019). “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation”. In: *Journal of field robotics* 36.2, pp. 416–446.
- Lazaros, Nalpantidis, Georgios Christou Sirakoulis, and Antonios Gasteratos (2008). “Review of stereo vision algorithms: from software to hardware”. In: *International Journal of Optomechatronics* 2.4, pp. 435–462.
- Leutenegger, Stefan, Margarita Chli, and Roland Y Siegwart (2011). “BRISK: Binary robust invariant scalable keypoints”. In: *2011 International conference on computer vision*. Ieee, pp. 2548–2555.
- Lowe, David G (2004). “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60, pp. 91–110.
- Ma, Yi, Stefano Soatto, Jana Košecká, and Shankar Sastry (2004). *An invitation to 3-d vision: from images to geometric models*. Vol. 26. Springer.
- Macario Barros, Andréa, Maugan Michel, Yoann Moline, Gwenolé Corre, and Frédérick Carrel (2022). “A comprehensive survey of visual slam algorithms”. In: *Robotics* 11.1, p. 24.
- Maddern, Will, Geoffrey Pascoe, Chris Linegar, and Paul Newman (2017). “1 year, 1000 km: The oxford robotcar dataset”. In: *The International Journal of Robotics Research* 36.1, pp. 3–15.
- Merzlyakov, Alexey and Steve Macenski (2021). “A comparison of modern general-purpose visual SLAM approaches”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 9190–9197.
- Muja, Marius and David G Lowe (2009). “Fast approximate nearest neighbors with automatic algorithm configuration.” In: *VISAPP (1)* 2.331–340, p. 2.
- Mur-Artal, Raul, Jose Maria Martinez Montiel, and Juan D Tardos (2015). “ORB-SLAM: A versatile and accurate monocular SLAM system”. In: *IEEE transactions on robotics* 31.5, pp. 1147–1163.
- Mur-Artal, Raul and Juan D Tardós (2017). “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras”. In: *IEEE transactions on robotics* 33.5, pp. 1255–1262.
- Myzhar (Feb. 2024). *Stereolabs Support Response on SLAM Algorithm Used by ZED (Forum Post)*. <https://community.stereolabs.com/t/slam-algorithm-used-by-zed/4498>. Stereolabs Community Forum, Support Response. Stereolabs.
- Open Robotics (2024). *Robot Operating System (ROS)*. Accessed: 2025-08-30. URL: <https://www.ros.org/>.

- OpenCV (2021a). *Camera Calibration and 3D Reconstruction*. Accessed: 2025-04-09. URL: https://docs.opencv.org/4.5.4/d9/d0c/group__calib3d.html.
- (2021b). *cv::FlannBasedMatcher Class Reference — OpenCV 4.5.4*. URL: https://docs.opencv.org/4.5.4/dc/de2/classcv_1_1FlannBasedMatcher.html.
- (2021c). *initUndistortRectifyMap Function Reference*. URL: https://docs.opencv.org/4.5.4/d9/d0c/group__calib3d.html.
- (2021d). *Introduction to SIFT (Scale-Invariant Feature Transform) Tutorial*. URL: https://docs.opencv.org/4.5.4/da/df5/tutorial_py_sift_intro.html.
- (2021e). *remap Function Reference*. URL: https://docs.opencv.org/4.5.4/da/d54/group__imgproc__transform.html.
- (2021f). *stereoRectify Function Reference*. URL: https://docs.opencv.org/4.5.4/d9/d0c/group__calib3d.html.
- (2021g). *StereoSGBM Class Reference*. URL: https://docs.opencv.org/4.5.4/d2/d85/classcv_1_1StereoSGBM.html.
- (2023). *Epipolar Geometry — OpenCV-Python Tutorials 1 documentation*. Accessed: 2025-06-16. URL: https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html.
- Pettyjohn, Stacie (2024). *Evolution Not Revolution: Drone Warfare in Russia’s 2022 Invasion of Ukraine*. Center for a New American Security.
- Poling, Bryan (2015). “A tutorial on camera models”. In: *University of Minnesota*, pp. 1–10.
- Rosten, Edward and Tom Drummond (2006). “Machine learning for high-speed corner detection”. In: *Computer Vision—ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer, pp. 430–443.
- Rosten, Edward, Reid Porter, and Tom Drummond (2008). “Faster and better: A machine learning approach to corner detection”. In: *IEEE transactions on pattern analysis and machine intelligence* 32.1, pp. 105–119.
- Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski (2011). “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. Ieee, pp. 2564–2571.
- Scaramuzza, Davide and Friedrich Fraundorfer (2011). “Visual odometry [tutorial]”. In: *IEEE robotics & automation magazine* 18.4, pp. 80–92.
- Schmitt, Eric and Marc Santora (May 2024). “Russia’s Jamming of U.S. Weapons in Ukraine Prompts Shift in War Tactics”. In: *The New York Times*. URL:

- <https://www.nytimes.com/2024/05/25/world/europe/us-weapons-russia-jamming-ukraine.html> (visited on 08/31/2025).
- StereoLabs (2024). *ZED 2i Stereo Camera*. Accessed: 2025-06-21. URL: <https://www.stereolabs.com/en-pt/store/products/zed-2i>.
- Stereolabs (2021). *ZED 2i Stereo Camera Datasheet*. Accessed: 2025-06-20.
- (2024a). *ZED ROS2 Wrapper*. <https://github.com/stereolabs/zed-ros2-wrapper>. Accessed: 2025-06-21.
- (2024b). *ZED SDK*. <https://github.com/stereolabs/zed-sdk>. Accessed: 2025-06-21.
- Sturm, Peter, Srikumar Ramalingam, Jean-Philippe Tardif, Simone Gasparini, Joao Barreto, et al. (2011). “Camera models and fundamental concepts used in geometric computer vision”. In: *Foundations and Trends® in Computer Graphics and Vision* 6.1–2, pp. 1–183.
- Sutton, H. I. (Sept. 2024). *Overview Of Maritime Drones (USVs) Of The Russo-Ukrainian War, 2022-24*. Covert Shores. Accessed: 2025-06-22. URL: <http://www.hisutton.com/Russia-Ukraine-USVs-2024.html>.
- (2025). *Timeline of Ukraine Invasion: War In The Black Sea*. Accessed: 2025-06-22. URL: <https://www.hisutton.com/Timeline-2022-Ukraine-Invasion-At-Sea.html>.
- Szeliski, Richard (2022). *Computer Vision: Algorithms and Applications*. Springer Nature.
- Taheri, Hamid and Zhao Chun Xia (2021). “SLAM; definition and evolution”. In: *Engineering Applications of Artificial Intelligence* 97, p. 104032.
- Tareen, Shaharyar Ahmed Khan and Zahra Saleem (2018). “A comparative analysis of sift, surf, kaze, akaze, orb, and brisk”. In: *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*. IEEE, pp. 1–10.
- Al-Tawil, Basheer, Thorsten Hempel, Ahmed Abdelrahman, and Ayoub Al-Hamadi (2024). “A review of visual SLAM for robotics: Evolution, properties, and future applications”. In: *Frontiers in Robotics and AI* 11, p. 1347985.
- Thrun, Sebastian, Wolfram Burgard, and Dieter Fox (2006). *Probabilistic Robotics*. Cambridge, MA: MIT Press. ISBN: 9780262201629.
- Tomasi, Carlo (2015). “A simple camera model”. In: *Notes from computer science* 527.
- Tourani, Ali, Hriday Bavle, Jose Luis Sanchez-Lopez, and Holger Voos (2022). “Visual slam: What are the current trends and what to expect?” In: *Sensors* 22.23, p. 9297.

- Turgeon, Travis (June 2025). *GNSS interference report: Russia 2024/2025 – Part 1 of 4: Kaliningrad & the Baltic Sea*. <https://spire.com/blog/space-reconnaissance/gnss-interference-report-russia/>. Accessed: 2025-08-31.
- Virtanen, Pauli et al. (2020). “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.
- Wiseman, John (2025). *GPSJAM: Daily Map of GPS/GNSS Interference (Map for 2025-08-29)*. <https://gpsjam.org/?lat=57.13765&lon=23.59151&z=3.5&date=2025-08-29>. Accessed: 2025-08-31.
- Yousif, Khalid, Alireza Bab-Hadiashar, and Reza Hoseinnezhad (2015). “An overview to visual odometry and visual SLAM: Applications to mobile robotics”. In: *Intelligent Industrial Systems* 1.4, pp. 289–311.
- Zhang, Mabel M, Jean Choi, Kostas Daniilidis, Michael T Wolf, and Christopher Kanan (2015). “VAIS: A dataset for recognizing maritime imagery in the visible and infrared spectrums”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 10–16.
- Zhang, Zhengyou (2002). “A flexible new technique for camera calibration”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.11, pp. 1330–1334.
- Zhang, Zichao and Davide Scaramuzza (2018). “A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry”. In: *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, pp. 7244–7251.
- Zhou, Qian-Yi, Jaesik Park, and Vladlen Koltun (2018). “Open3D: A Modern Library for 3D Data Processing”. In: *arXiv:1801.09847*.
- Žust, Lojze, Janez Perš, and Matej Kristan (2023). “Lars: A diverse panoptic maritime obstacle detection dataset and benchmark”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20304–20314.