



# Master Thesis Report

---

## Combining Multibeam and Photogrammetry point clouds

Leonie Buchele

Master Thesis

Erasmus Mundus Master in  
Marine and Maritime Intelligent Robotics

Universitat Jaume I

September 30, 2024

Supervised by: Raul Marin Prades and Timmy Gambin



Co-funded by the  
Erasmus+ Programme  
of the European Union





## ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my thesis supervisor, Raul Marin Prades, for his invaluable time, guidance, and insightful contributions throughout the writing of this thesis.

I am also incredibly grateful to my co-supervisor, Timmy Gambin, for the countless opportunities he provided during my time in Malta. I am especially thankful for his trust in allowing me to work with the Autonomous Underwater Vehicle (AUV), which has been a truly meaningful experience for me. His support extended beyond the thesis itself, and I sincerely appreciate his encouragement in involving me in various dive projects.

I would like to extend my heartfelt thanks to my parents for their constant encouragement and belief in me. Their unwavering support throughout my journey, especially in the decisions that ultimately led me to Malta, has been invaluable.

A special thank-you goes to Alberto and Julia for their unwavering assistance with the AUV. I truly appreciate their patience in answering my many questions about the software and for always being there to help and shown me how to operate the AUV. I am also very grateful to Rafael for always being there to answer my programming-related questions. His support and reliability made a significant difference in the success of this thesis.

To all the wonderful people in Malta who helped me along the way, I am deeply appreciative. Special thanks to Nikos and Nick for proofreading and offering valuable corrections, and to Gabby for her incredible cooking, which provided much-needed nourishment during this process. Lastly, a heartfelt thank you to Nick's dog, Lucy, for the unexpected yet much-needed moral support and for being my loyal foot warmer during many long writing sessions.

# ABSTRACT

This thesis investigated the integration of Multibeam Echosounder (MBES) data and photogrammetry models to enhance the quality of underwater archaeological surveys, particularly focusing on plane wrecks in Malta. The objective was to combine two techniques—acoustic MBES for wide-area bathymetric mapping and photogrammetry for detailed 3D reconstruction—into a unified point cloud for improved accuracy and data usability.

This research employed the Autonomous Underwater Vehicle (AUV) Gavia, marking the first time the University of Malta has used this platform for MBES data collection on plane wrecks. Extensive mission planning was necessary to ensure the safe and accurate operation of the AUV, with special attention given to environmental factors such as sea-floor obstacles and wave conditions. The survey area was carefully selected and overlapping lines were programmed into the mission plan to ensure sufficient data coverage for post-processing.

Post-processing of the collected multibeam data involved converting raw files into compatible formats for use in SonarWiz. The data were filtered and cleaned using several built-in functions—such as Static Box Filters, Cutoff Angle Filters, and Sample Density Filters—to remove outliers and ensure the accuracy of the final output. Various point cloud registration methods, including Iterative Closest Point (ICP) and Random sample consensus (RANSAC), were tested for aligning and merging datasets from both sources. The project developed an automatic registration process and a novel method to color MBES point clouds using visual data, contributing to more robust and efficient underwater mapping workflows. Future work will focus on enhancing registration accuracy and investigating machine learning approaches to optimize the fusion of MBES and photogrammetry datasets under varying environmental conditions.

**Keywords:** Autonomous Underwater Vehicle (AUV), Sea trials, Multibeam Echosounder (MBES), Photogrammetry (PG), Point cloud registration (PCR), Underwater Archaeology, Iterative Closest Point (ICP), Random sample consensus (RANSAC)



# CONTENTS

<b>Contents</b>	<b>iv</b>
<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Work Motivation . . . . .	2
1.2 Objectives . . . . .	2
1.3 Environment and Initial State . . . . .	4
1.3.1 Multibeam and Photogrammetry in Malta . . . . .	4
1.3.2 Reconstruction of Wrecks with Multibeam . . . . .	5
1.3.3 Reconstruction of Wrecks using Multibeam and Photogrammetry . . . . .	5
1.3.4 Match and Compare point clouds . . . . .	7
1.4 Test Objects: Plane Wrecks . . . . .	11
1.4.1 JU88 South . . . . .	12
1.4.2 JU88 North . . . . .	12
1.4.3 Spitfire . . . . .	12
<b>2 Planning and resources evaluation</b>	<b>14</b>
2.1 Planning . . . . .	14
2.2 Resource Evaluation . . . . .	19
<b>3 System Analysis and Design</b>	<b>21</b>
3.1 AUV Gavia . . . . .	22
3.2 Multibeam . . . . .	23
3.2.1 Function of a Multibeam . . . . .	23
3.2.2 Blueview Multibeam . . . . .	24
3.2.3 R2Sonic V-2026 Multibeam . . . . .	25
3.3 Photogrammetry Model . . . . .	27
3.4 Libraries to compare point clouds . . . . .	29
3.4.1 Software CloudCompare . . . . .	29
3.4.2 Open 3D Python Library . . . . .	29
3.5 Methods to compare point clouds . . . . .	30

---

3.5.1	Iterative Closest Point (ICP) . . . . .	30
3.5.2	Random sample consensus (RANSAC) . . . . .	32
3.6	Values to compare the matching process . . . . .	33
3.7	Combining Photogrammetry and Multibeam . . . . .	35
3.7.1	Intrinsic Parameters . . . . .	36
3.7.2	Extrinsic Parameters . . . . .	37
3.7.3	Complete Projection Model . . . . .	37
<b>4</b>	<b>Work Development and Results</b>	<b>38</b>
4.1	Work Development to get the MBES point cloud . . . . .	38
4.1.1	Side Scan Sonar data in 3D . . . . .	39
4.1.2	Pre-processing AUV mission . . . . .	40
4.1.3	Post-Processing AUV Multibeam data . . . . .	42
4.2	Registration Process . . . . .	46
4.2.1	Registration in CloudCompare . . . . .	46
4.2.2	Registration in Open 3D Libraries . . . . .	48
4.2.3	Tuning parameter in Open 3D Libraries . . . . .	52
4.2.4	Tuning Results in Open 3D Libraries . . . . .	54
4.3	Results Coloring . . . . .	58
4.3.1	Pictures taken from existing point cloud . . . . .	58
4.3.2	Pictures taken by Divers . . . . .	62
4.3.3	Tests with Agisoft . . . . .	63
<b>5</b>	<b>Conclusions and Future Work</b>	<b>65</b>
5.1	Conclusions . . . . .	65
5.2	Discussion . . . . .	66
5.3	Future work . . . . .	67
	<b>Bibliography</b>	<b>69</b>
<b>A</b>	<b>Appendices</b>	<b>72</b>
A.1	Tuning Results . . . . .	72
A.2	Mission Protocol JU88 South . . . . .	73
A.3	Risk assessment . . . . .	74
A.4	Checklist . . . . .	75
<b>B</b>	<b>Source code</b>	<b>76</b>
<b>C</b>	<b>Additional Pictures and Videos</b>	<b>77</b>

## ABBREVIATIONS

<b>AUV</b>	Autonomous Underwater Vehicle
<b>DVL</b>	Doppler Velocity Log
<b>FLS</b>	forward-looking sonar
<b>FPFH</b>	Fast Point Feature Histograms
<b>GCC</b>	Geometric Constraint Cluster
<b>GCM</b>	Geometric Constraint-based Method
<b>GUI</b>	Graphical User Interface
<b>ICP</b>	Iterative Closest Point
<b>INS</b>	Inertial Navigation System
<b>LGV</b>	Local and Global Voting
<b>MBES</b>	Multibeam Echosounder
<b>NDT</b>	Normal Distribution Transform
<b>PCR</b>	Point cloud registration
<b>PG</b>	Photogrammetry
<b>PPS</b>	Pulse-Per-Second
<b>RAF</b>	Royal Air Force
<b>RANSAC</b>	Random sample consensus
<b>RMSE</b>	Final Root Mean Square Error
<b>ROV</b>	Remotely Operated Vehicle
<b>SSS</b>	Side Scan Sonar
<b>UHD</b>	Ultra High Density
<b>UHR</b>	Ultra High Resolution

**USBL** Ultra-short baseline underwater positioning system

**V-GTM** Variant of Game Theoretic Matching

## LIST OF FIGURES

1.1	Location of the plane wrecks . . . . .	11
2.1	Timeline for the Project (Part 1) . . . . .	15
2.2	Timeline for the Project (Part 2) . . . . .	16
2.3	Timeline for the Project (Part 3) . . . . .	17
2.4	Timeline for the Project (Part 4) . . . . .	18
2.5	Diving Boat used for missions . . . . .	19
2.6	Inside view of the boat, (picture during transit) . . . . .	20
3.1	Different Modules of the AUV [11] . . . . .	22
3.2	Function of a Multibeam Echosounder [27] . . . . .	23
3.3	Angles for the Multibeam [15] . . . . .	24
3.4	Teledyne BlueView MB2250 [10] . . . . .	25
3.5	R2Sonic V-2026 Multibeam [24] . . . . .	26
3.6	3D Photogrammetry Model JU88 South [8] . . . . .	28
3.7	3D Model JU88 North [8] . . . . .	28
3.8	3D Model Spitfire [8] . . . . .	28
4.1	Side Scan Sonar data of the Fairey Fulmar . . . . .	39
4.2	3D view of the Side Scan Sonar data . . . . .	39
4.3	Mission planning AUV JU88 South . . . . .	40
4.4	Steps during the cleaning process . . . . .	43
4.5	Position Offset Spitfire . . . . .	45
4.6	Registration in CloudCompare with ICP . . . . .	47
4.7	Registration process . . . . .	48
4.8	Insufficient result of Registration . . . . .	54
4.9	Visual comparing Point-to-Point and Point-to-Plane function . . . . .	55
4.10	Values for Tuning ICP parameters . . . . .	57
4.11	Coloring process . . . . .	58
4.12	Colored multibeam point cloud with camera position . . . . .	59
4.13	Results of coloring MBES point cloud with pictures from divers . . . . .	63
4.14	New Photogrammetry model in known environment . . . . .	64
4.15	Test in Python with new model . . . . .	64

---

A.1	Results tuning RANSAC algorithm . . . . .	72
A.2	Mission Protocol JU88 South . . . . .	73
A.3	Risk Assessment for the AUV . . . . .	74
A.4	Checklist for the AUV . . . . .	75

## LIST OF TABLES

1.1	Project Goals . . . . .	3
1.2	Comparing Sources . . . . .	10
3.1	Blueview’s multibeam parameters [10] . . . . .	25
3.2	R2- Sonic V-2026 multibeam parameters [24] . . . . .	27
4.1	Mission Parameters for JU88 South . . . . .	41
4.2	Mission Parameters for JU88 South: BlueView Bathymetric Sonar . . . . .	41
4.3	Mission Parameters for JU88 South: EdgeTech Side Scan Sonar . . . . .	41
4.4	Comparing point clouds in CloudCompare . . . . .	46
4.5	Numerical comparing Point-to-Point and Point-to-Plane function . . . . .	55
4.6	Tuning parameters for RANSAC . . . . .	56
4.7	Tuning parameters for ICP . . . . .	56

## INTRODUCTION

**Contents**


---

1.1	Work Motivation . . . . .	<b>2</b>
1.2	Objectives . . . . .	<b>2</b>
1.3	Environment and Initial State . . . . .	<b>4</b>
1.3.1	Multibeam and Photogrammetry in Malta . . . . .	4
1.3.2	Reconstruction of Wrecks with Multibeam . . . . .	5
1.3.3	Reconstruction of Wrecks using Multibeam and Photogrammetry . . . . .	5
1.3.4	Match and Compare point clouds . . . . .	7
1.4	Test Objects: Plane Wrecks . . . . .	<b>11</b>
1.4.1	JU88 South . . . . .	12
1.4.2	JU88 North . . . . .	12
1.4.3	Spitfire . . . . .	12

---

The exploration of underwater environments has become increasingly sophisticated with the advent of advanced technologies, particularly in the realm of bathymetric (acoustic) data measurements and photogrammetry models. Acoustic signals emerge as the preferred method for obtaining a large layout of the seafloor and extracting data related to depth. Bathymetry's significant advantage lies in the possibility to use it over a wide area, making it the preferred choice for comprehensive underwater surveys. In addition, photogrammetry can provide a more detailed overview on specific features of interest. Both techniques involve the deployment of advanced hardware configurations and the use of diverse software applications to analyze and visualize the acquired data. The integration of technologies and methodologies in this project is part of the ongoing efforts to enhance the understanding of archaeological underwater environments



generally, in underwater plane wrecks specifically, and contribute to the broader field of marine exploration.

## 1.1 Work Motivation

The motivation for starting this thesis is to find out, if and how Multibeam data can be combined with Photogrammetry. Therefore, different data sets needed to be collected, and the broader view of how to register and combine them were investigated. Finally, the best practice for fine tuning the code had to be investigated and developed in order to yield the best possible results

It is important to mention that the technologies are only compared within the realm of archaeology surveys. To further reduce the scope, only three plane wrecks chosen as representatives of a variety of different wreck sides. It is investigated how the technology worked for a first survey, to complement a full dataset with multibeam and photogrammetry and according to its quality. First surveys are mainly concerned with generating a good state-of-the-art dataset that can be used for later research or future surveys. The next step following a first survey is to monitor wrecks and detect how they change over time. This is, naturally, a long term project. Therefore, it is more important to compare different datasets or study more details and get a fast overview.

In reference to the two different sources for the point clouds; this paper investigated how the combination would help to improve the quality of the 3D model and what options for the combination and registration are possible.

## 1.2 Objectives

The project was initiated with the goal of understanding the limitations and potential approaches for combining Multibeam and Photogrammetry point clouds. The main objective was to explore how both technologies can be used to enhance the quality and speed of generating results in situations where Multibeam point clouds and visual inputs—such as images or videos—are collected using a single vehicle, specifically the Autonomous Underwater Vehicle (AUV) Gavia.

The focus of this work is primarily on Multibeam technology, as the photogrammetry models have already been created and the corresponding point cloud was available. This research marked the first time the University of Malta had used the AUV Gavia to collect Multibeam data for bathymetric purposes. Prior to this investigation, the University of Malta had been scanning the seafloor at low resolution until a target of interest was identified. Divers or a high-resolution Side Scan Sonar (SSS) would then be employed for a more detailed investigation.

This project encompasses a comprehensive understanding of the AUV from mission planning to data collection at sea using Multibeam Echosounder (MBES). Once the data was

collected, it underwent post-processing and cleaning. Afterward, the Multibeam point cloud was compared with the existing photogrammetry point cloud. In this step, it is important to understand the different registration methods and explore options for combining these datasets. A potential approach for integration is to develop an automatic coloring process. For the photogrammetry point cloud, it is essential to understand how the model is generated and gain an overview of the techniques and limitations involved.

The goals of the project are summarized in table 1.1. They are prioritized according to importance, starting with "Priority 1," which is critical for the project, to "Priority 3," which is a nice-to-have. Future work that is not included in this research is categorized under "not included." The project goals include operating the AUV, post-processing the data, and developing two sets of code. One code was designed to handle the automatic registration of point clouds, including parameter tuning, and the second was used to color the Multibeam Echosounder point cloud using visual data from images.

Table 1.1: Project Goals

<b>Goal</b>	<b>Description</b>	<b>Criteria</b>	<b>Priority</b>
Understand AUV	Understanding function and planning of missions	Mission is prepared	Priority 1
Gather data	MBES point cloud is gathered for all Planes	point cloud gathered	Priority 1
Hardware: combine camera and MBES	Use a camera and MBES in same vehicle	Both mounted	not included
Software: combine camera and MBES	Program to combine/register both point clouds	Program tested	Priority 1
Costs	all cost for the project	calculation of cost	Priority 3
Finishing date	Finish the thesis earlier	Thesis handed in	Priority 2

The project started at the beginning of February 2024, due to weather conditions, the initial phase focused on general research and gaining an understanding of the AUV and its software. Once weather conditions improved, fieldwork commenced and MBES data collection began. After data collection, the comparison and programming phase followed. A more detailed record of the planning stage can be seen later in Chapter 2.1.

The project budget included costs for AUV operation, renting a boat for two full days, and licensing various software programs. All expenses were covered by the University of Malta.

Several risks were identified for the project in the beginning. The most significant risk to the project was the weather, as conditions for collecting the Multibeam data had to be clear. There was no existing data available for initial tests, so it was crucial to prioritize early data collection. Additionally, the project could negatively be impacted by a bad data quality for the point clouds or technical issues with the vehicle.

## 1.3 Environment and Initial State

In this section, a literature review with various papers and books are presented to get an overview of the field. The focus is on papers that explore either multibeam data for reconstruction of wrecks or the combination of both multibeam and photogrammetry technologies. The sources are mainly from the field of underwater archaeology. These sections should help to classify the topic in the broader context and get information about advantages and disadvantages that other researchers found. In the table 1.2 on the final page of this section, the sources are listed and compared according to the different topics that are covered in this thesis. Those categories are: Archeology, Multibeam Echosounder (MBES), Photogrammetry (PG), Combining both, Registration, Coloring.

### 1.3.1 Multibeam and Photogrammetry in Malta

In 2021 the paper "*From discovery to public consumption: The process of mapping and evaluating underwater cultural heritage in Malta*" [14] gives an overview about underwater cultural heritage in Malta and the context of surveys with Side Scan Sonar (SSS). It was used in this Master thesis as a starting point. Therefore, the following section of this thesis will describe the main achievements of the paper.

The research noted that most identified shipwrecks around the world are currently found in waters shallower than 50 meters but the authors anticipated that this would change due to technological advances and increased deep-water exploration. The paper highlights the synergy between remote sensing technologies and underwater heritage management, using ongoing research projects like Malta's surveying approach and scoring system for historic wreck sites as examples. This approach categorized sites based on various parameters, aiding in the creation of management strategies. The goal of the paper was to show how large-scale remote sensing surveys can significantly contribute to site management by enhancing decision-making processes for divers and the public. Additionally, it highlighted the importance of promoting and preserving underwater cultural heritage sites.

Another notable source was the website *Underwater Malta* [8]. On the website, the various researched wrecks around Malta are displayed. Additional information about the wrecks are listed and the user is able to visualize them in a 3D Photogrammetry model. In the section 3.3 Photogrammetry Models, it is in more detail described how the models are generated and how they were used in the context of this thesis.

### 1.3.2 Reconstruction of Wrecks with Multibeam

The use of multibeam data is a well-established practice in the field of underwater archaeology. Acoustic methods are mostly used to find either wrecks or quickly survey large areas of the seafloor.

An example is listed in the book "*Jutland 1916*" [21] from 2018, where the author used Multibeam data to map a World War I battlefield. Innes McCartney used multibeam data to detect different shipwrecks - in combination with a Remotely Operated Vehicle (ROV) - to positively identify them. The book analyzed the data to understand both the order of the battle and the damage the wrecked ships had received. In the future, the author hoped the research would protect the ships from further destruction and illegal plundering. Therefore, it was important to map the current condition as exact as possible and preserve it in 3D models.

Other institutions have already used 3D modelling to display wreck sites online. A good example is the website of *3DVisLab* [19]. Contracted by the UK Government, the purpose of this project, is to investigate historic wreck sites and preserve them with 3D models. For this purpose, they use only acoustic data (multibeam) and exclude photogrammetry completely. This application's main research goal is to explore the potential of multibeam sonar for accurately and rapidly surveying shipwrecks; it was, therefore, unnecessary to gather pictures as the main structure of the wrecks was of interest. The different 3D point clouds models of the wrecks are displayed on their website and can be easily accessed.

Another notable example is the website *Infomar* [16], which adopts a broader perspective. The primary focus of Ireland's seabed mapping initiatives is to comprehensively survey the physical, chemical, and biological characteristics of the nation's seabed. The strategy aims to establish a marine baseline dataset supporting various national interests, including security and informed decision-making across economic, environmental, infrastructural, and policy realms. The existing dataset comprises a diverse range of geophysical measurements, including MBES bathymetry and backscatter, shallow seismic profiles, gravity, magnetics, SSS, and oceanographic water column profiles. Additionally, it incorporates valuable data concerning physical ground-true samples and documentation of over 420 discovered shipwrecks.

### 1.3.3 Reconstruction of Wrecks using Multibeam and Photogrammetry

Compared to acoustic surveys, photogrammetry is a newer technology that is still evolving. Therefore, fewer examples of the combination of both technologies can be found. The existing sources give either a theoretical background about the technologies, or if both techniques are used on a case like a shipwreck, it is to build the first model.

One book that delves into the theoretical background of state-of-the-art methods for 3D recording and mapping is "*State of the art and applications in archaeological underwater 3D recording and mapping*" [22] from 2018. It describes the advantages and disadvantages of using both multibeam and photogrammetry in the context of archaeological sites. The paper discusses the technical aspects, giving an overview about different sensors and the challenges on various locations and focuses more on the different technologies separately and not on the combination of both. In chapter 3.3 of the paper, techniques for integrating underwater data from acoustic and optical systems to overcome their individual limitations are discussed. The challenges of aligning such data are expounded upon, with optic-acoustic extrinsic calibration and feature matching identified as major issues that require further attention. They concluded that—despite the current lack of popularity in underwater archaeological mapping—the fusion of visual and acoustic data, coupled with advancements in acoustic system resolution held promise for improving underwater mapping, especially in turbid waters.

Another paper from 2016 that provides a theoretical overview for both technologies is: "*Underwater optical and acoustic imaging: A time for fusion? A brief overview of the state-of-the-art*" [12]. This paper explores the fusion of optical and acoustic imaging in underwater environments, addressing the inherent limitations and advantages of each modality. While optical imaging provides high resolution and color information, it suffers from light attenuation and water turbidity. On the other hand, sonar systems offer robustness to these issues but generally have lower resolution and lack color information. The Researchers propose, combining data from both modalities to improve underwater imaging, especially with recent advancements in higher resolution sonar systems. Applications such as autonomous navigation, mapping, and object recognition stand to benefit from this fusion approach. The paper reviews various approaches to the fusion of the technologies, highlighting examples ranging from simple data combination to more complex inter-sensor fusion and high-level data fusion. Examples include combining acoustic arrays with optic systems, pencil beam sonars with optical cameras, multibeam sonars with optics, and forward-looking sonars with optics. Finally, the paper identifies the same challenges as in "*State of the art and applications in archaeological underwater 3D recording and mapping*" [22], such as extrinsic calibration and feature matching between optical and acoustic data. Extrinsic calibration involves aligning the coordinate systems of both sensors, while feature matching aims to identify corresponding features across modalities. Also, they come to the same conclusion as in the paper before [22]. Despite the progress in optic-acoustic systems, more research is needed to address calibration and feature matching issues. The paper concludes by emphasizing the potential of fusion in underwater imaging and the importance of theoretical studies to further advance the field.

There are more examples of published papers that tested both technologies on a specific project to reconstruct wrecks or archaeological sites. One notable example can be found in the book "*3D Recording and Interpretation for Maritime Archaeology*" [20],

chapter 12, published in 2019. The paper discusses an innovative approach to create a 3D visualization of HMS *Falmouth*, a British light cruiser sunk during World War I off the Yorkshire coast. The visualization combined Multibeam Echosounder survey data of the wreck with photogrammetry and laser scanning of the original builder's model of the ship. The main difference between this paper and the ones before is that there are no technical challenges discussed but the impact and benefits to the community. The purpose of the paper was to, both generate public and media interest as well as to serve as a platform for engaging the public with underwater archaeology. Moreover, the author discusses the potential for future development of visualization techniques, including incorporating still photographs and videos taken by divers and creating a "virtual dive route" for non-diving audiences. The paper underscores the significance of HMS *Falmouth* as a heritage asset, particularly in the broader context of preserving the United Kingdom's maritime history, which often remains submerged and overlooked. In sum, it underscores the value of 3D visualization in animating underwater cultural heritage, fostering public engagement, and safeguarding historical narratives for future generations.

Another relatively new paper from 2021 that combines different 3D mapping methods is: "*Exploration and reconstruction of a medieval harbor using hydroacoustic, 3D shallow seismic and underwater photogrammetry: A case study from puck, southern baltic sea*" [17]. This study focused on exploring the submerged archaeological heritage harbor in Puck, Poland, which was one of the largest medieval harbors in the Baltic Sea. Multiple surveying methods—including MBES, parametric sub-bottom profiler, and aerial photography—were employed to explore the site. High-resolution bathymetry revealed seabed features and archaeological artifacts, while 3D shallow seismic datasets identified buried structures, shipwrecks, excavation trenches, and the harbor boundary. The fusion of the different datasets enables a comprehensive visualization of the heritage site. The researchers combined these technologies to increase the resolution of their data, but did not dedicate space in their paper comparing and contrasting the different competing methods.

For the context of this thesis, it is interesting to note the fact that the paper compares the new generated models with archival documentation. As a result, the paper documented significant erosion rate of biogenic layers containing archaeological objects over a period of 26 years. The authors concluded by highlighting the effectiveness of underwater remote-sensing methods for exploring and preserving underwater heritage sites.

### 1.3.4 Match and Compare point clouds

Point cloud registration (PCR) methods play a crucial role in aligning different point clouds to create a unified representation of a scene or object in 3D. This registration methods plays a vital role in various applications, including 3D reconstruction, object recognition, and autonomous navigation. Despite its significance, selecting the most suitable PCR method remains challenging due to the diversity of available techniques and the lack of comprehensive comparative studies. Therefore, in the following pages,

different papers that focused on the topic of PCR using various methods will be compared.

The paper "*Comparison of point cloud Registration Techniques on Scanned Physical Objects*" [9] from 2024 presents a comparison of six PCR methods, including classical techniques like Iterative Closest Point (ICP), Random sample consensus (RANSAC) and modern deep-learning-based approaches such as PointNetLK (Point Network, Lucas and Kanade algorithm), RPMNet (Robust Point Matching Network), and ROPNet (Retinopathy of Prematurity Network). The author used as a baseline for comparing of synthetic and real-world datasets, key metrics such as: precision, variance, speed, generalizability, and required pre-processing.

The validation results on the test dataset demonstrated the effectiveness of the PCR methods in achieving accurate point cloud alignment. The comparison highlights the influence of various parameters—such as voxel size and Final Root Mean Square Error (RMSE) threshold—on registration accuracy and computational efficiency. Their analysis of registration parameters revealed significant impact on the performance of PCR methods. Varying voxel sizes affected the recall metric differently across methods, emphasizing the need for parameter optimization based on the specific application and dataset characteristics. The comparative evaluation underscores the strengths and limitations of classical and deep-learning-based PCR methods. While classical techniques like RANSAC and ICP offer real-time performance and robustness to environmental clutter, learning-based approaches like PointNetLK and RPMNet demonstrate superior accuracy but require extensive training and higher computational resources.

In 2018 the authors of the paper: "*Registration of Laser Scanning Point Clouds: A Review*" [5] compared the different disadvantages and advantages of classical registration methods. The paper concludes that each registration method has its advantages and disadvantages. The ICP algorithm provides high precision alignment but requires dense point clouds and accurate initial alignment. RANSAC is robust to outliers, while it struggles with poor initial alignment. Feature-based methods offer efficiency in coarse registration, yet may face challenges in feature extraction and uneven feature distribution. The Normal Distribution Transform (NDT) algorithm offers computational efficiency, however suffers from limitations in handling complex surface geometries. Understanding the strengths and limitations of each method is essential for selecting the most appropriate approach based on the characteristics of the point cloud data and the specific registration requirements.

Published in 2019, the paper: "*A Comprehensive Performance Evaluation for 3D Transformation Estimation Techniques*" [3] evaluated eleven transformation estimation techniques across four benchmark datasets and revealed significant insights for selecting appropriate methods in 3D space applications. The methods were assessed based on their performance on both descriptor-based and synthetic correspondences. The researchers did this by considering factors like dataset quality, overlap ratios, combining

different transformation techniques, local descriptors, and Local Reference Frame/Axis techniques. Results indicated that Variant of Game Theoretic Matching (V-GTM) consistently achieved 100% correct registration and demonstrated superior performance across all datasets due to its iterative evolution with a game theory approach, which allowed it to effectively exclude false correspondences. The paper states that methods like Geometric Constraint Cluster (GCC) and Geometric Constraint-based Method (GCM) performed poorly—despite their efficiency—as they struggle with low ratios of correct registration across datasets. This is contributed to their reliance solely on geometric constraints without iterative evolution. Moreover, the efficiency of methods varied, with GCC and GCM being the most efficient but least accurate, while RANSAC and Local and Global Voting (LGV) struck a balance between efficiency and performance. Guidance for selecting appropriate methods in specific applications was provided based on dataset quality and application requirements. For high-quality datasets, V-GTM emerged as the top choice due to its excellent performance and efficiency. On the other hand, for low-quality datasets, 1-Point-RANSAC combined with ICP proved to be the best option. Additionally, for applications utilizing Local Reference Frame-based descriptors, 1-Point-RANSAC was recommended for its superior performance. In time-critical applications like robotics and mobile platforms, RANSAC was preferred for its efficiency and acceptable performance. V-GTM, however, remained a viable option due to its superior overall performance and execution speed.

The last paper that will be discussed in this section was: "*A Review of Point Cloud Registration Algorithms for Laser Scanners: Applications in Large-Scale Aircraft Measurement*" [25] from 2022. The paper compared PCR methods, each with its own set of advantages and disadvantages. Hierarchical optimization methods, such as the ICP algorithm, offered iterative refinement of point cloud alignment, ensuring convergence to a local minimum. These methods, however, require a good initial guess for convergence and may struggle with complex geometries.

On the other hand, stochastic and probability distribution-based approaches—like the RANSAC algorithm, exhibited robustness against outliers and noise, making them suitable for real-world scenarios. Nonetheless, these methods may not always converge to the optimal solution and necessitate parameter tuning for different datasets. Feature-based methods that focus on specific geometric features provide robustness and efficiency in registration. Despite their advantages, they can be computationally intensive and struggle with smooth surfaces lacking in distinctive features. In the context of large-scale aircraft measurement, specialized algorithms address challenges such as efficient boundary feature extraction and multi-view registration. Misalignment issues persist due to the smooth surfaces of aircraft, requiring further advancements in registration precision. Future research might include improving the fusion of traditional methods with deep learning techniques, developing specialized algorithms tailored to specific application scenarios, and overcoming challenges with partial overlap in point clouds. Overall, while significant progress has been made in PCR, there remain ample opportunities for innovation and refinement in this field.



Table 1.2: Comparing Sources

<b>Name</b>	<b>Source</b>	<b>Year</b>	<b>Archeology</b>	<b>MBES</b>	<b>PG</b>	<b>Combining both</b>	<b>Registration</b>	<b>Coloring</b>
<i>From discovery to public consumption: The process of mapping and evaluating underwater cultural heritage in Malta.</i> [14]	Paper	2021	yes, Malta	yes	no	no	no	no
<i>Underwater Malta</i> [8]	Website	-	yes, in Malta	no	yes	no	no	no
<i>Jutland 1916</i> [21]	Book	2018	yes, Jutland	yes	no	no	no	no
<i>3DVisLab</i> [19]	Website	-	yes, UK	yes	no	no	no	no
<i>Infomar</i> [16]	Website	-	yes, Ireland	yes	no	no	no	no
<i>State of the Art and Applications in Archaeological Underwater 3D Recording and Mapping</i> [22]	Paper	2018	yes	yes	yes	a little bit	no	no
<i>Underwater Optical and Acoustic Imaging: A Time for Fusion? A Brief Overview of the State-of-the-Art</i> [12]	Paper	2019	no	yes	yes	yes	no	no
<i>3D Recording and Interpretation for Maritime Archaeology</i> [20]	Book	2019	yes	yes	yes	yes	no	no
<i>Exploration and Reconstruction of a Medieval Harbor...</i> [17]	Paper	2021	yes	yes	yes	yes	no	no
<i>Comparison of Point Cloud Registration Techniques on Scanned Physical Objects</i> [9]	Paper	2024	no	no	no	no	yes	no
<i>Registration of Laser Scanning Point Clouds: A Review</i> [5]	Paper	2018	no	no	no	no	yes	no
<i>A Comprehensive Performance Evaluation for 3D Transformation Estimation Techniques</i> [3]	Paper	2019	no	no	no	no	yes	no
<i>A Review of Point Cloud Registration Algorithms for Laser Scanners: Applications in Large-Scale Aircraft Measurement</i> [25]	Paper	2022	no	no	no	no	yes	no

## 1.4 Test Objects: Plane Wrecks

To efficiently compare the technologies, three similar plane wrecks located off the coast of Malta were selected. Planes are relatively small compared to shipwrecks, making it feasible to obtain bathymetric data in a single dive using an AUV. The locations of all three plane wrecks are depicted in the following map (Figure 1.1).



Figure 1.1: Location of the plane wrecks

The next sections will provide further insights into the submerged planes and the specific location of the wrecks.

During World War II, Malta emerged as a vital Allied stronghold, positioned strategically between Gibraltar and Alexandria. With the German Luftwaffe establishing a presence in Sicily, Malta became a prime target for Axis bombings. The island's proximity to Sicily facilitated frequent air raids, often involving upwards of 200 JU88 bombers daily. Allied forces, however, managed to bolster their aerial defenses, leading to their eventual reclamation of air superiority over Malta by early May 1942 [8].

### 1.4.1 JU88 South

Uncovered during another remote sensing survey in 2019, the JU88 South revealed its true identity as a Junkers JU88 following a visual inspection by divers in 2021. Positioned up- right on a sandy seabed at a depth of 106 meters, the aircraft is in excellent condition overall, although signs of wear and tear are evident, particularly in the nose and tail sections. The damage observed on the port tail wing of the JU88 suggests potential scenarios contributing to its demise, such as engagement in aerial combat, anti-aircraft fire, or controlled ditching. One theory as to the cause of the wreck notes that the damage pattern on the plane suggests the possibility of the aircraft being struck from behind, potentially during a dogfight. The precise cause of the damage remains undetermined, with multiple factors potentially contributing to the plane's current state. Therefore, a more detailed examination is required to determine the exact cause [8].

The Junkers JU88, a tactical medium-range bomber, stands out as one of the most adaptable combat aircraft of the Germans in the World War II. First taking flight in December 1936, the JU88 quickly garnered military interest, leading to its deployment with the *Luftwaffe* by late 1939. Manufactured by *Junkers Flugzeug und Motorwerke AG*, production soared, with approximately 15,000 units rolling off the assembly lines by war's end in 1945. The Junkers JU88 was operated by a four-member crew comprising, all situated within the glazed cockpit [8].

### 1.4.2 JU88 North

The JU88 North was found in 2009, during an offshore remote sensing survey. The wreck lies approximately 3km outside Salina Bay at a depth of 57 meters on a sandy seabed. Remarkably well-preserved, the aircraft exhibits a broken tail, situated a short distance from the main wreckage site. Notably, the forward-facing machine gun remains intact, mounted within the cockpit. Same as the JU88 South, it is not known why the plane crashed, but hints at potential encounters with Allied forces during the period of renewed aerial contention [8].

### 1.4.3 Spitfire

The Supermarine Spitfire was an iconic aircraft for the Royal Air Force (RAF) throughout the World War II, noted for its continuous production among British aircraft. Designed by R. J. Mitchell of Supermarine Aviation Works in 1934, the prototype first flew in March 1936, and the aircraft entered RAF service by August 1938. By the end of the war, over 20,000 Spitfires had been built, with more than 20 Marks and 50 sub-variants [8].

The Spitfire was a single-seat, high-performance interceptor featuring advanced attributes such as a variable pitch propeller, all-metal construction, retractable undercarriage, and an elliptical wing. This design allowed it to house eight machine guns and

endure high-speed maneuvers. The Spitfire gained legendary status during the Battle of Britain and was a key fighter in the air battle of Malta in 1942 [8].

The Spitfire wreck located off the coast of the island of Gozo is linked to Operation Husky, the Allied invasion of Sicily from July to August 1943. Following the Allied victory in North Africa, the focus shifted to Italy, with Malta serving as a crucial base. During the invasion, Spitfire Squadrons based in Gozo, including the 307th, 308th, and 309th, provided air coverage. On 30 June 1943, the first Spitfires landed at the newly constructed Xewkija airfield in Gozo. A missing aircrew report from 30 June 1943 details a Spitfire from the 308th Fighter Squadron that disappeared while en route from Korba, Tunisia, to Gozo. The aircraft, a Spitfire Mk Vc with a Merlin 45 engine, was lost after a missed approach to Xewkija. Despite a search by the Malta Air/Sea Rescue, the aircraft was not found, and the pilot remains missing [8].

In April 2021, a side-scan sonar survey discovered the wreck site at a depth of 70 meters. The Spitfire is upright on a sandy seabed, with the rear fuselage folded and the tail upside down. Violent ditching is indicated by the collapse of the rear fuselage and the disintegration of the wooden Dowty Rotol propeller blades. This evidence suggests that the wreck is indeed the missing aircraft from the 308th Squadron [8].

## PLANNING AND RESOURCES EVALUATION

### Contents

---

2.1	Planning . . . . .	14
2.2	Resource Evaluation . . . . .	19

---

This chapter first provides an overview of the timeline established for the master’s thesis and the necessary dependencies to achieve the final goal. In the second subsection, the planning of a mission with the Autonomous Underwater Vehicle (AUV) was examined in greater detail. The section evaluated the costs, logistics, and safety factors that were considered for the AUV mission.

### 2.1 Planning

At the start of the project, a timeline was created, which is shown in figure 2.1 till 2.4. This timeline was more general and covered the entire project. Specific tasks were divided into subtasks as needed. For instance, the task "*00-3-A: Plan AUV missions*" also included the application process and permit verification.

The project followed a classic project management approach, divided into six milestones over six months. The critical path was marked in red on the timeline. While the overall time plan was adhered to, updates were necessary after the data collection phase. This was due to the unpredictable nature of the data collection dates, so a significant time buffer was added in the initial planning. Additionally, as much work as possible was completed before fieldwork began.

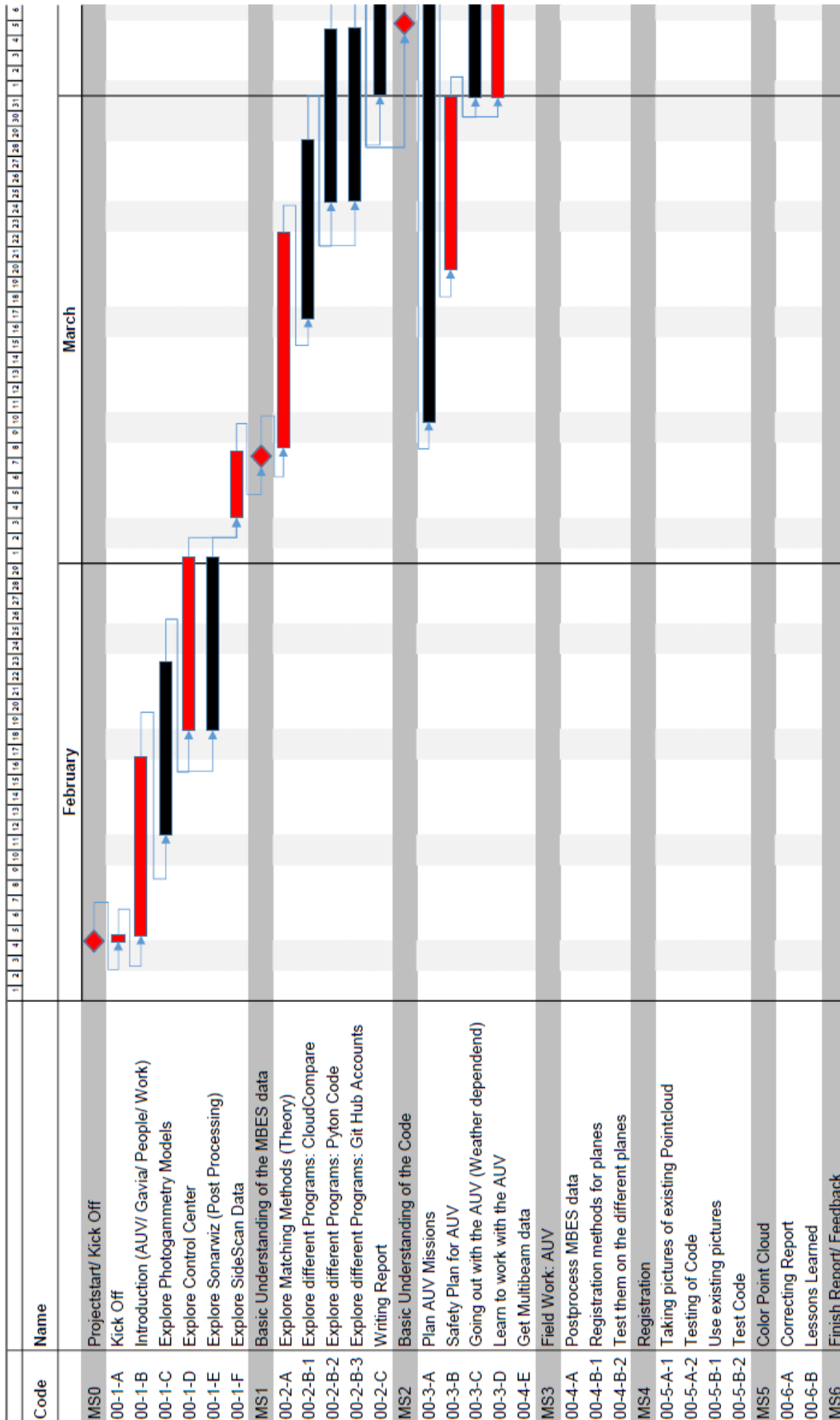


Figure 2.1: Timeline for the Project (Part 1)

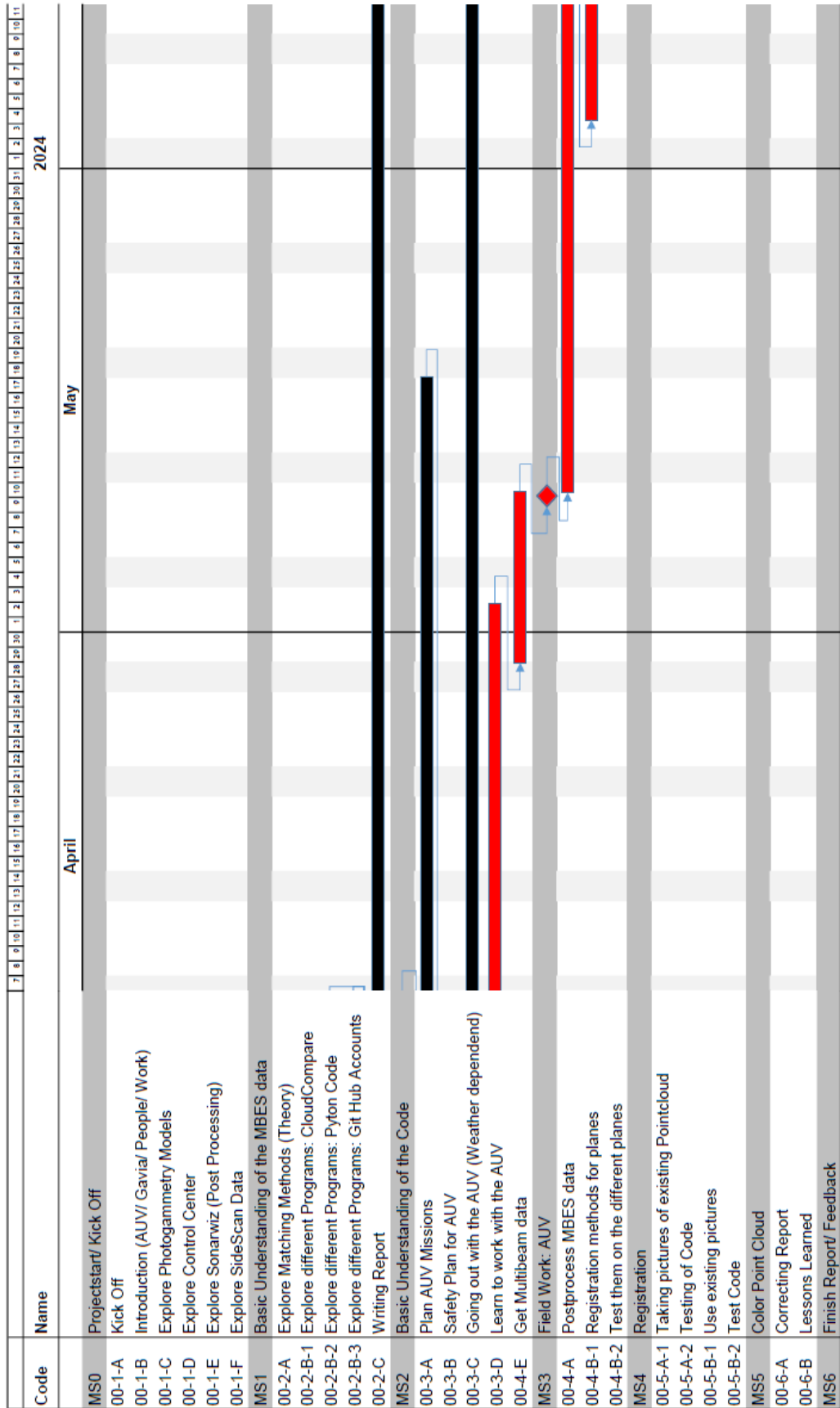


Figure 2.2: Timeline for the Project (Part 2)

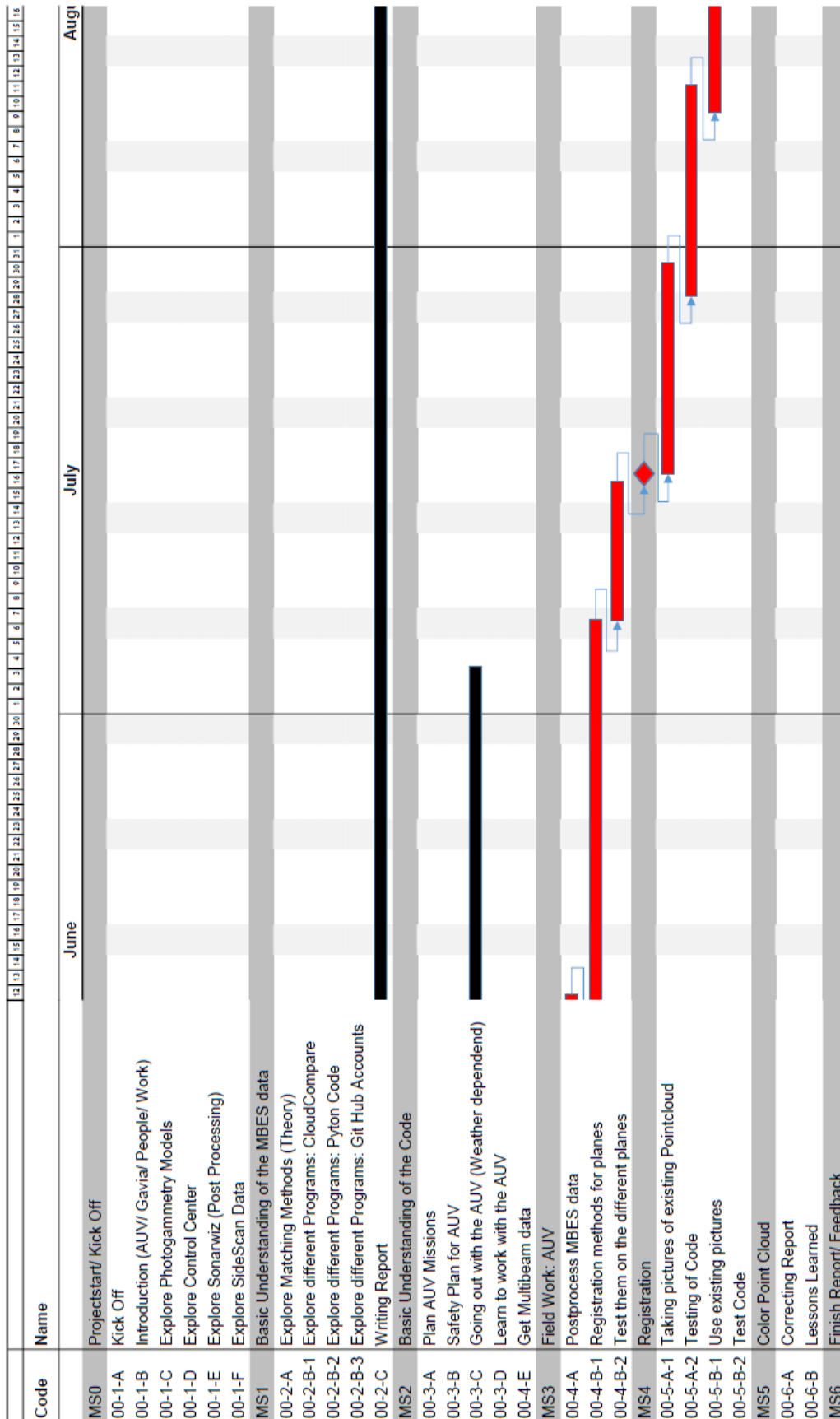


Figure 2.3: Timeline for the Project (Part 3)



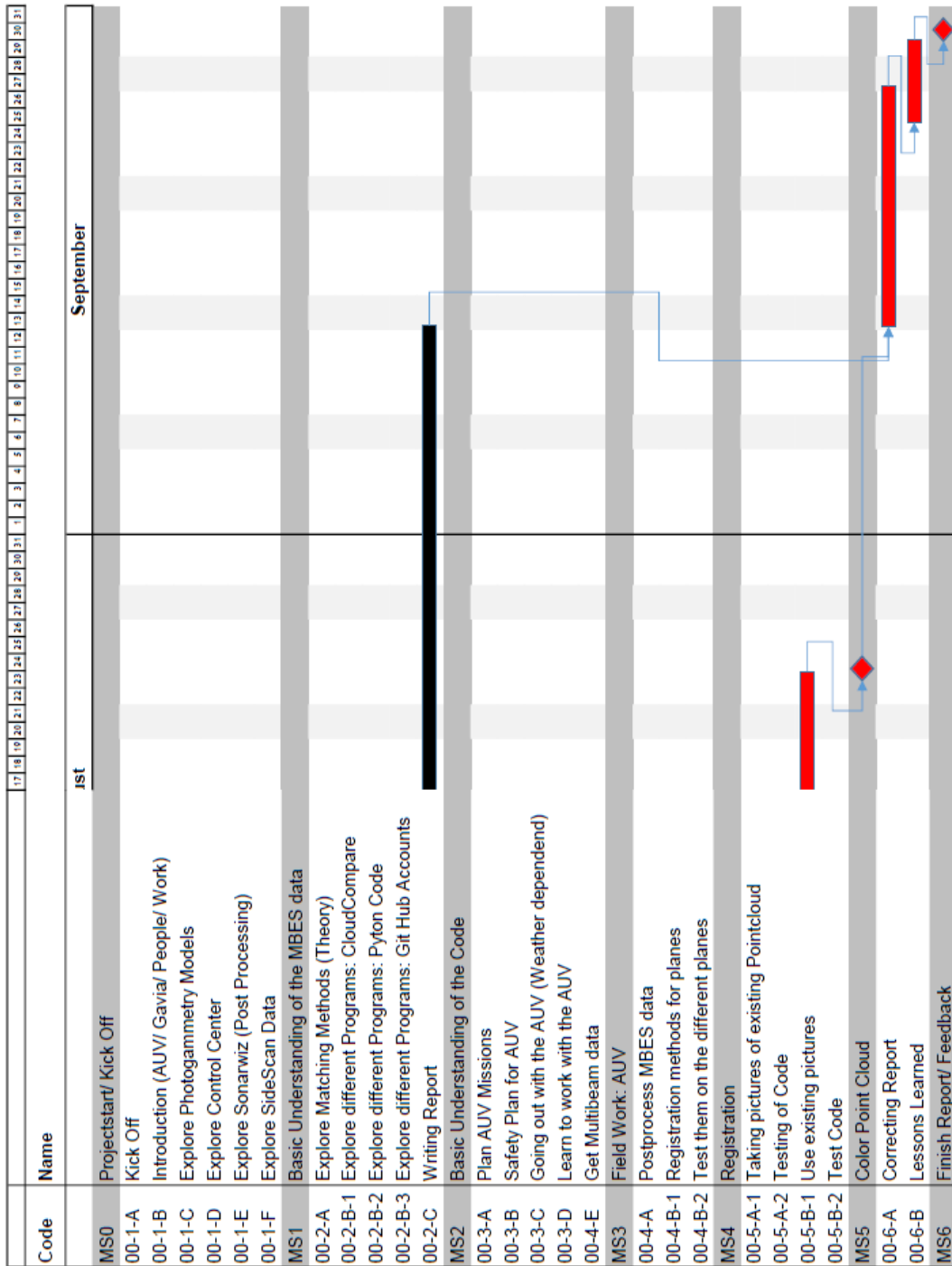


Figure 2.4: Timeline for the Project (Part 4)

## 2.2 Resource Evaluation

The project required a variety of resources, and this section delves into the specific resources, costs, and risks associated with the project. Each item is detailed below, along with how they were managed throughout the project.

- **Permit:**

It is important to remember that the sites are managed by Heritage Malta, meaning any activity surrounding that area needs an official permit for the time and day. This needs to be prebooked on the website. For normal dives there is an admission to pay, in case of the AUV the sites need to be closed to divers during the time of the survey.

- **Vehicle/ Boat:**

To transport the AUV and all additional equipment to the harbour were the boat leaves and back to the University, a big VAN (Peugeot Ducato) was provided by the University. For each day out on the sea, a Diving boat including a captain was rented for the full day. The boat can be seen in figure 2.5, the figure 2.6 displays the preparation inside the boat on the transit to the mission side.



Figure 2.5: Diving Boat used for missions

- **People:**

A minimum of four people was required to handle the AUV. The vehicle weighed around 130 kg and needed to be loaded into the van, transferred to the boat, and later deployed into and retrieved from the water after the mission.

- **Software:**

The project also required a high-performance computer equipped with the Sonarwiz software to read and post-process the collected data, the license for that is provided by the University.

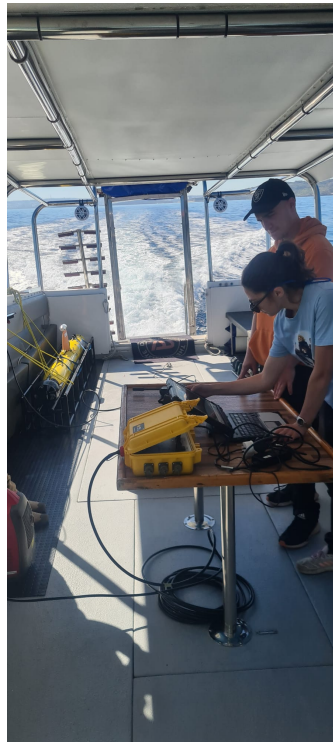


Figure 2.6: Inside view of the boat, (picture during transit)

**Costs:**

As for the costs, these were all covered by the University of Malta. They have ownership of the AUV with all additional equipment and own additionally the license for all the Software. The Dive boat was rented for missions with the AUV and the team members that helped to operate the AUV were students of the Underwater Archaeological course.

**Risks:**

The primary risks for both personnel and the AUV occurred during data collection at sea. To mitigate these risks, special care was taken for the sea days, a detailed risk assessment plan was prepared and is included in the appendix A.3. To guarantee the AUV's safety, the mission planning included checks for seafloor obstacles or significant height variations. Backup plans were developed in case circumstances changed, such as challenges in locating or retrieving the AUV, or if the weather forecast worsened on the scheduled day. Furthermore, attention was given to boat traffic in the area and the potential presence of fishing gear in the water. The most dangerous phases for the AUV were during its descent and ascent and on the surface [13].

## SYSTEM ANALYSIS AND DESIGN

**Contents**


---

3.1	AUV Gavia . . . . .	<b>22</b>
3.2	Multibeam . . . . .	<b>23</b>
3.2.1	Function of a Multibeam . . . . .	23
3.2.2	Blueview Multibeam . . . . .	24
3.2.3	R2Sonic V-2026 Multibeam . . . . .	25
3.3	Photogrammetry Model . . . . .	<b>27</b>
3.4	Libraries to compare point clouds . . . . .	<b>29</b>
3.4.1	Software CloudCompare . . . . .	29
3.4.2	Open 3D Python Library . . . . .	29
3.5	Methods to compare point clouds . . . . .	<b>30</b>
3.5.1	Iterative Closest Point (ICP) . . . . .	30
3.5.2	Random sample consensus (RANSAC) . . . . .	32
3.6	Values to compare the matching process . . . . .	<b>33</b>
3.7	Combining Photogrammetry and Multibeam . . . . .	<b>35</b>
3.7.1	Intrinsic Parameters . . . . .	36
3.7.2	Extrinsic Parameters . . . . .	37
3.7.3	Complete Projection Model . . . . .	37

---

The following chapter gives some background information needed to understand and compare the results. It is intended to help the reader understand the used equipment, software and technologies behind. Therefore, it focuses on the AUV, the software, the Multibeam Echosounder (MBES), a brief introduction to Photogrammetry and how the results were compared.

### 3.1 AUV Gavia

To gather the acoustic model, an Autonomous Underwater Vehicle (AUV) was used, which is a type of robot that operates underwater without direct human control. These vehicles are equipped with various sensors, navigational systems, and propulsion mechanisms, enabling them to perform tasks such as oceanographic research, underwater mapping, environmental monitoring, and military surveillance autonomously. AUVs were designed to withstand the harsh conditions of ocean depths and can operate at significant depths for extended periods, gathering valuable data that would have been difficult or impossible to obtain using traditional manned methods. Their autonomy allows them to navigate through complex underwater environments, collect data, and execute pre-programmed missions with minimal human intervention, making them essential tools for a wide range of marine applications [11].

For this master's thesis in particular, the AUV *Gavia*, from the company *Teledyne Marine*, was used. This AUV is owned by the University of Malta. It is a self-contained, low-logistics, modular survey platform capable of delivering high-quality data while operating from vessels of opportunity or from the shore. The *Gavia* AUV could dive to depths of up to 1000 meters, surpassing many others in its category. Its modular construction ensured maximum flexibility, allowing for easy customization and adaptation to different mission requirements, even between dives [11].

In figure 3.1, the AUV with all the modules is displayed. The AUV used was equipped with two battery modules, recognizable by the handle on top, and the propeller unit at the back. In the front, the noise cone with an obstacle avoidance sensor was installed. In the middle, the control module was located, identified by the orange fin. It included an acoustic modem transducer, an antenna tower, and a sound velocity meter. On both the left and right sides, a Side Scan Sonar (SSS) was mounted. Additionally, the AUV was equipped with a high-accuracy, survey-grade Inertial Navigation System (INS) navigation system, along with a Doppler Velocity Log (DVL), ensuring precise positioning and navigation during missions. For the data collection in this thesis, a module for Multibeam Echosounder (MBES) was mounted. This was the third module from the back, positioned between *Battery 2* and the *Control Module*.

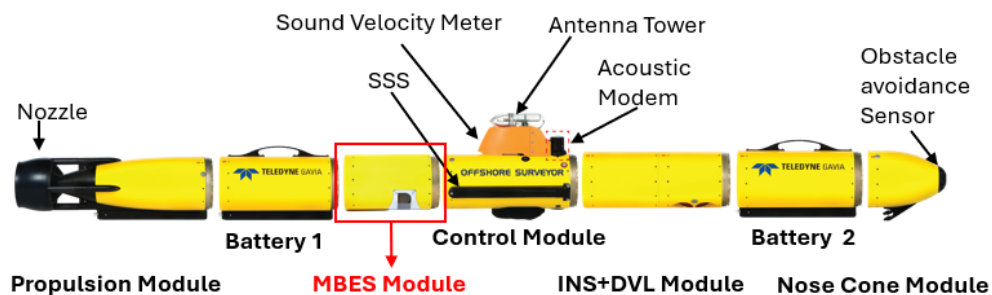


Figure 3.1: Different Modules of the AUV [11]

To control the AUV, a chart-based Graphical User Interface (GUI) called *Control Center* was provided. This interface simplified mission planning and monitoring for operators during missions. To display and process the recorded data, *Teledyne* used the software *SonarWiz*. It served as an all-in-one solution for seafloor mapping, offering a comprehensive suite of features designed to streamline the mapping process from start to finish. The software included intuitive management tools, reliable data acquisition capabilities, powerful post-processing functions, and adaptable reporting options [11].

## 3.2 Multibeam

To obtain bathymetric data, a MBES or SSS system was used. For this specific project, a Multibeam device from *BlueRobotics* has been mounted. The subsequent section provides general information about Multibeam systems, followed by specific details about the particular device used in this project.

### 3.2.1 Function of a Multibeam

MBES are acoustic devices that simultaneously emit and receive multiple beams of sound to measure the depth and shape of the seafloor. They are widely used for hydrographic surveying, seabed mapping, and underwater exploration. The systems provide high-resolution and accurate bathymetric data over a large area in a short time and can be mounted on a boat or underwater vehicle.

In an acoustic multibeam sonar, a transducer array emits multiple beams of sound waves in different directions. By analyzing the returning echoes, the system can create a detailed and accurate three-dimensional map of the seafloor. The use of multiple beams allows for a broader coverage area and faster data acquisition compared to traditional single-beam sonar systems [27].

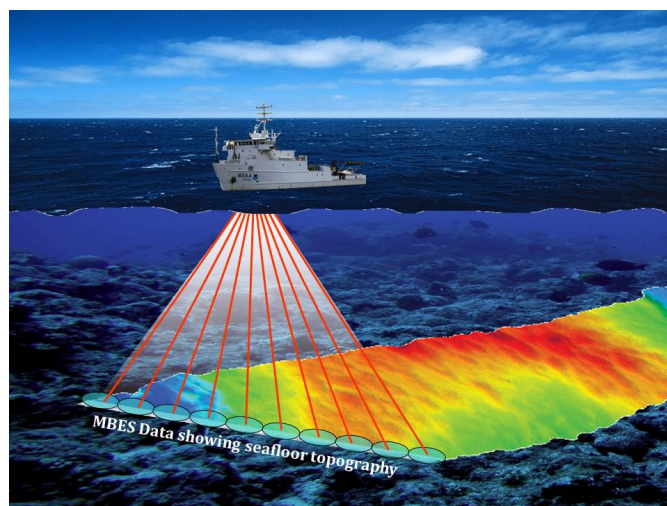


Figure 3.2: Function of a Multibeam Echosounder [27]

The functional principle can be seen in figure 3.2. It works by emitting multiple pulses of sound, called pings, through a transducer at a specific frequency. The same pulse is then received through a receiver placed very close to the transducer. The time it takes for the sound pulse to reflect off the bottom and return to the receiver is used to calculate the water depth [27].

Unlike other sonars and echo sounders, MBES use beam forming to extract directional information from the returning sound waves, producing a swath of depth soundings from a single ping (Figure 3.3). The swath angle varies per system, but is generally somewhere between  $120^\circ$  and  $170^\circ$ , giving swath widths on the bottom in the order of 3.5 to 25 times the water depth. Most MBES have a single transmit beam and a number of receive beams. The received beams are formed on reception [15].

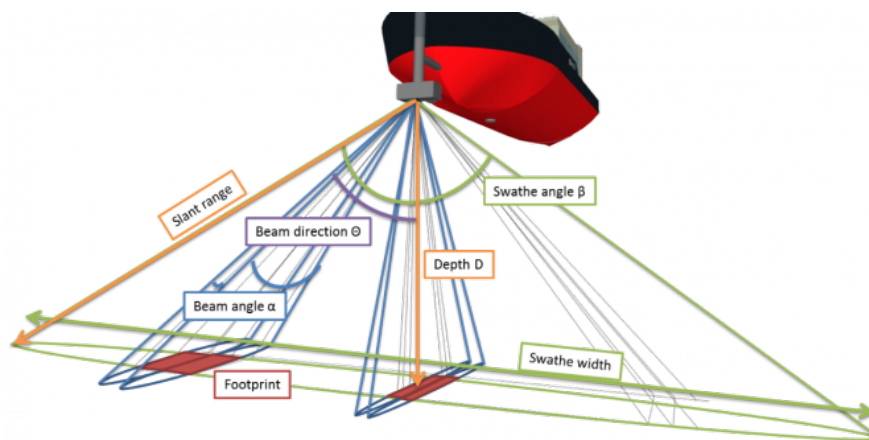


Figure 3.3: Angles for the Multibeam [15]

The final data density is defined by the number of beams (depths) and the ping rate, or the number of swaths that the MBES can measure per second. The ping rate depends on the water depth, but can be as high as 60 pings per second in shallow water. The accuracy of the depth measurements depends on several factors, such as the sound velocity, the pulse length, the bandwidth, and the motion compensation of the system. The advantage of MBES lies in its ability to provide high-resolution and accurate bathymetric data over a large area in a short time. They can also detect objects in the water column or along the seafloor, such as underwater features, fish, floating nets, wrecks, or pipelines. This technology is widely used for different applications like hydrographic surveying, seabed mapping, and underwater exploration [15].

### 3.2.2 Blueview Multibeam

The *Teledyne BlueView MB2250* is a high-frequency (2.25 MHz) micro-bathymetry echo sounder, it delivers high-resolution data in a proprietary ".son" format. Equipped with built-in data logging capabilities, this system has a local storage capacity up to 240 GB. One of the key features is the use of a Pulse-Per-Second (PPS) signal on the *Gavia*



*BUS*, facilitating clock synchronization. This synchronization ensures precise alignment between the sonar ping and navigation data, enabling accurate association of each ping with its corresponding geographical location [10]. The module is displayed in the following figure 3.4.



Figure 3.4: Teledyne BlueView MB2250 [10]

Table 3.1 lists important parameters for the BlueView Multibeam system.

Table 3.1: Blueview's multibeam parameters [10]

Parameter	Value
Number of beams	512
Frequency	2250 kHz
Resolution	6 mm
Ping rate	up to 30 Hz
Ping period	50 ms
Maximum range	10 m
Swath Range	15° (3 m) to 120° (8.5 m)
Beam width	1° x 1° (nominal)

### 3.2.3 R2Sonic V-2026 Multibeam

The R2Sonic V-2026 Multibeam Echosounder is an advanced component of the Sonic™-V Series, designed to provide high-resolution bathymetric data with a focus on user-friendliness and operational flexibility. This boat mounted MBES offers a range of features that enhance its performance for marine surveying applications. The R2Sonic incorporates the Compact VOX-IM system, which features serial connectors for integrating motion sensors, GPS time messages, and sound velocity probes, ensuring comprehensive and accurate data collection. The system supports a wide range of voltage inputs, both AC and DC, making it compatible with various vessel power systems. Its VOX Control User Interface has a modern and intuitive design, which simplifies operation and



configuration through improved logic, configurable hotkeys, language translations, and a detailed help feature [24].

One of the standout features of the R2Sonic V-2026 is its Ultra High Density (UHD) data acquisition capability, which allows for up to 1024 true soundings per ping, providing high data density across the swath and enabling detailed seabed mapping. The system also supports remote upgrades to technical modes, allowing for ongoing enhancement of its capabilities. It offers Ultra High Resolution (UHR) at 700 kHz for certain models and extended sounding depth with 90/100 kHz for the 2026-V model. Additionally, it includes TruePix® / Multispectral backscatter with compressed water column and a switchable forward-looking sonar (FLS). The R2Sonic V-2026 is available with depth ratings of 100 meters, 4000 meters, and 6000 meters, making it suitable for a variety of surveying depths.

How the Multibeam looks and how it can be mounted on a Boat can be seen in the figure 3.5. The used boat was a special hybrid that could also drive on land and in the sea, what made surveying different locations and unmounting the Multibeam easy. [24].



Figure 3.5: R2Sonic V-2026 Multibeam [24]

In the following table 3.2 are important parameters listed for the R2 Sonic Multibeam:

Table 3.2: R2- Sonic V-2026 multibeam parameters [24]

Parameter	Value
Number of beams	1024
Frequency	170kHz - 450kHz, Low: 90kHz - 100kHz
Resolution	3 mm
Ping rate	up to 60 Hz
Ping period	15 $\mu$ s -2ms
Maximum range	800 m
Swath Range	10° to 160°
Beam width	0.45° x 0.45° at 450kHz

When compared to AUV-mounted Multibeam systems, such as the Gavia from *Teledyne BlueView*, the R2Sonic V-2026 offers several advantages. Boat-mounted systems benefit from a stable power supply provided by the vessel, avoiding the limitations of battery life associated with AUVs. They also allow for real-time data processing and immediate access to results, which can enhance decision-making during surveys. Additionally, boat-mounted systems are easier to maintain and upgrade due to their accessibility from the vessel [24].

However, boat-mounted systems have some downsides compared to AUV-mounted systems. They may have limited access to deeper or more challenging environments where AUVs excel. AUVs offer greater mobility and can operate autonomously in a wider range of conditions and locations, making them suitable for surveys in deeper waters and confined spaces. Furthermore, the performance of boat-mounted systems is subject to the vessel's stability and environmental conditions, while AUVs can mitigate some of these external factors due to their autonomous nature [15].

In summary, the R2Sonic V-2026 MBES is a robust and versatile solution for high-resolution bathymetric surveys from a boat. It provides significant benefits in terms of power supply stability, real-time data management, and ease of maintenance. However, for surveys requiring greater mobility and access to challenging underwater environments, AUV-mounted systems like the Gavia may be more suitable. The choice between these systems should be guided by specific survey requirements, including operational depth, accessibility, and environmental conditions.

### 3.3 Photogrammetry Model

The 3D photogrammetry models of the three plane wrecks already existed. Figure 3.6 till 3.8 show the plane wrecks that are of interest for this thesis. They were created with the assistance of divers who used cameras to capture images of the wrecks underwater. The divers, needed special equipment to access the wrecks because some of them are in a depth of over 100 m. Part of their equipment was for diving, rebreather technology

and for the high-resolution images, powerful cameras and special underwater lights to systematically record the sites [8].



Figure 3.6: 3D Photogrammetry Model JU88 South [8]



Figure 3.7: 3D Model JU88 North [8]



Figure 3.8: 3D Model Spitfire [8]

Afterward, the 3D models were generated using a software called *Agisoft Metashape*, which employs photogrammetry techniques to reconstruct three-dimensional objects from two-dimensional images. Photogrammetry is a method used to measure and create accurate 3D models from photographs. It involves analyzing the position, orientation, and scale of objects in multiple images to recreate their spatial properties. Underwater photogrammetry, in particular, is widely used for documenting submerged structures, archaeological sites, and marine life.

*Agisoft Metashape* is one of the leading software solutions for photogrammetry and 3D modeling. It allows users to process large sets of images, align them, and generate detailed 3D models with textures and colors. Additionally, the software calculates the position of the camera for each picture. In the post-processing of the model, it is also possible to adapt the scale of the model and give it specific positions like the world coordinates instead of a local coordinate system [2].

The models of the plane wrecks can be viewed on the website of the Underwater Museum Malta [8], where they serve as educational resources and tools for underwater exploration and research.

## 3.4 Libraries to compare point clouds

To reach the final goal of comparing the point clouds, different tools were explored. The main points of interest included their accuracy, along with the time and technical resources required. Therefore, the existing program *CloudCompare* was used, in addition to the Library *Open3D* to generate an own customized code.

### 3.4.1 Software CloudCompare

To get a first and fast overview of the models and compare the results generated by different technologies (Multibeam and Photogrammetry), the tool *CloudCompare* was employed. *CloudCompare* is a powerful open-source software designed for processing and visualizing 3D point cloud data. The software offers a comprehensive suite of tools for analyzing, comparing, and manipulating point cloud datasets.

It is programmed in C++ with a user-friendly interface and robust feature set. It empowers users to efficiently perform tasks such as registration, alignment, segmentation, and mesh generation on large-scale point clouds. The program has already the Iterative Closest Point (ICP) registration as a function included for point clouds. [7]

### 3.4.2 Open 3D Python Library

In the next step, for a more detailed insight and testing of further registration methods, different Python Libraries that are joined under *Open3D* [1] were introduced. It was important to be careful with the different versions of this Library, ensuring that they were not mixed and that the fitting modules were properly installed.

*Open3D* was designed with a modular architecture, comprising distinct modules for different functionalities. These modules can be combined flexibly to address specific requirements, ensuring scalability and extensibility. The core components include data structures for representing 3D geometry, algorithms for processing and analyzing 3D data, and utilities for visualization and interaction.

The key features are the following ones:

- 3D data I/O: Support for loading and saving various 3D data formats, such as point clouds and meshes.
- Geometry processing: Operations for mesh simplification, smoothing, and alignment, as well as point cloud registration and downsampling.
- 3D visualization: Interactive visualization tools for exploring and analyzing 3D data in real-time.
- Deep learning integration: Seamless integration with popular deep learning frameworks like TensorFlow and PyTorch for tasks such as 3D object detection and segmentation [1].

For the project also the Visualizer with multiple drawing options was installed, as well as the different methods for pre-processing, like for example transformations or point removal. Another function to work with multiple clouds was the registration tool, which was employed to align them. For alignment, there were different options available, including the Iterative Closest Point (ICP) and the Random sample consensus (RANSAC) methods [1].

## 3.5 Methods to compare point clouds

Python Libraries and *CloudCompare* used different methods for registration. In the following part, more details are provided to each one, it is explained what are the differences and the advantages. The focus lays here in the already existing libraries, without introducing additional algorithms or providing more details here.

### 3.5.1 Iterative Closest Point (ICP)

The Iterative Closest Point (ICP) algorithm serves as a fundamental technique in the field of 3D point cloud processing, aimed at finding the optimal transformation between two point clouds by minimizing the distance between corresponding points. Originally proposed in the early 1990s, ICP evolved into several variants and adaptations, each tailored to specific scenarios and requirements.

ICP operates on the principle of iteratively refining the transformation (translation and rotation) between two point clouds to minimize the distance metric, typically the sum of squared distances or Point-to-Point distance. The basic workflow of ICP involves the following steps:

- Correspondence estimation: Establishing point correspondences between the source and target point clouds.
- Transformation estimation: Computing the optimal transformation that aligns the source points with the target points.
- Transformation update: Updating the transformation and repeating the process until convergence criteria are met [5].

In general, the ICP algorithm served as a cornerstone in the field of 3D point cloud processing, providing a robust and efficient framework for point cloud alignment and registration. The advantage of the ICP algorithm lies in its ability to iteratively minimize the distance between corresponding points in two point clouds, thereby achieving accurate registration. However, the ICP algorithm requires high-density point clouds and an accurate initial alignment for optimal performance. Additionally, the computational complexity of the ICP algorithm can be high, especially for large-scale point cloud datasets. Despite these challenges, the ICP algorithm remains a popular choice for fine registration due to its effectiveness in achieving precise alignment [5].

Over the years, numerous variants and extensions of the original ICP algorithm have been proposed to address specific challenges and enhance its performance. Some notable variants include:

- Point-to-Plane ICP: Minimizing the distance between points and the tangent plane of the target surface.
- Point-to-Point ICP: Minimizing the distance between points in the two surfaces.
- Robust ICP: Incorporating robust estimation techniques to mitigate the influence of outliers and noise.
- Multi-resolution ICP: Employing multi-scale representations to improve convergence and efficiency [28].

In this thesis, the first two extensions (Point-to-Point and Point-to-Plane) will be researched in more detail. A mathematical description of both of these can be seen below. The function contains always two point clouds as follows:

Source point cloud  $\mathbf{P}$  with  $N$  points:

$$\mathbf{P} = \{\mathbf{p}_i \mid i = 1, \dots, N\}$$

Target point cloud  $\mathbf{Q}$  with  $M$  points:

$$\mathbf{Q} = \{\mathbf{q}_j \mid j = 1, \dots, M\}$$

#### **Point-to-Point:**

The goal of the algorithm is to find the rigid transformation (rotation and translation) that minimizes the distance between corresponding points in the source point cloud  $\mathbf{P}$  and the target point cloud  $\mathbf{Q}$ . Therefore, the rotation matrix  $\mathbf{R} \in \mathbf{R}^{3 \times 3}$  and the translation vector  $\mathbf{t} \in \mathbf{R}^3$  for the source point cloud that minimize the following cost function needs to be found:

$$E_{\text{point}}(\mathbf{R}, \mathbf{t}) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_{i'}\|^2$$

where  $\mathbf{q}_{i'}$  is the closest point in  $\mathbf{Q}$  to the transformed point  $\mathbf{R}\mathbf{p}_i + \mathbf{t}$ . The Cost Function  $E_{\text{point}}(\mathbf{R}, \mathbf{t})$  measures the sum of squared distances between each point  $p_i$  in the source point cloud after transformation ( $\mathbf{R}$  and  $\mathbf{t}$ ) and its closest point  $\mathbf{q}_{i'}$  in the target point cloud [1].

#### **Point-to-Plane:**

The goal of the Point-to-Plane ICP algorithm is similar to that of the Point-to-Point algorithm, but instead of minimizing the distance between corresponding points, it minimizes the distance between each point in the source point cloud and the plane defined by its corresponding point in the target point cloud. This approach often leads to faster convergence, especially when the initial alignment is already close.

The cost function to be minimized in the Point-to-Plane ICP is defined as follows:

$$E_{\text{plane}}(\mathbf{R}, \mathbf{t}) = \frac{1}{N} \sum_{i=1}^N ((\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_{i'}) \cdot \mathbf{n}_{i'})^2$$

$\mathbf{n}_{i'}$  is the normal vector at the point  $\mathbf{q}_{i'}$  in the target point cloud  $\mathbf{Q}$  and therefore  $(\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_{i'}) \cdot \mathbf{n}_{i'}$  represents the scalar projection of the vector from  $\mathbf{q}_{i'}$  the transformed point  $\mathbf{R}\mathbf{p}_i + \mathbf{t}$  onto the normal vector  $\mathbf{n}_{i'}$ .

The cost function  $E_{\text{plane}}(\mathbf{R}, \mathbf{t})$  measures the sum of squared distances between each transformed point  $\mathbf{p}_i$  in the source point cloud and its corresponding plane in the target point cloud, defined by  $\mathbf{q}_{i'}$  and its normal vector  $\mathbf{n}_{i'}$ . This method minimizes the orthogonal distance to the planes rather than the Euclidean distance to the points, which can provide a more robust and stable alignment in cases where the surface normals are well-defined and reliable [1].

### 3.5.2 Random sample consensus (RANSAC)

Random sample consensus (RANSAC) is a powerful iterative algorithm designed to robustly estimate model parameters from a set of observed data points containing outliers and noise. Proposed in the early 1980s, RANSAC has since become a cornerstone in many computer vision and geometric modeling applications, owing to its ability to handle data with significant levels of corruption. [4]

The RANSAC is a random feature based algorithm, because of the random factor of each runtime, the results differ from the one before. Feature-based registration methods leverage distinctive features extracted from point clouds, such as points, lines, or surfaces, that help in identifying, matching, or analyzing objects and achieve coarse registration. Features refer to specific descriptors or characteristics calculated from the point cloud data, like the Fast Point Feature Histograms (FPFH). These features capture local geometric properties of the points, such as their spatial arrangement and relative distances. They simplify and enhance the process of aligning or classifying point clouds by focusing on key attributes rather than raw data. These methods are efficient in reducing computational demands by providing a good initial alignment for subsequent fine registration steps. However, feature-based methods may encounter challenges in scenarios with sparse or unevenly distributed features, leading to registration errors. Furthermore, the selection of appropriate features is crucial for the success of feature-based methods, and extracting relevant features can be challenging in certain environments. [4]

The RANSAC algorithm operates on the principle of iteratively sampling subsets of data points, fitting models to each subset, and evaluating the quality of the fit through a predefined criterion. The basic workflow of RANSAC consists of the following steps:

- Sample selection: Randomly select a minimal subset of data points to form a potential model hypothesis.
- Model fitting: Estimate model parameters using the selected subset of data points.

- Inlier detection: Determine the subset of data points that are consistent with the model within a predefined tolerance threshold.
- Model evaluation: Assess the quality of the model based on the number of inliers.
- Iteration: Repeat the process for a specified number of iterations or until convergence criteria are met.
- Model refinement: Optionally, refine the model using all inliers found during the iterations [4].

RANSAC stands as a cornerstone algorithm in computer vision, image processing, and geometric modeling, providing a robust and efficient framework for fitting models to noisy data contaminated with outliers. Despite its widespread success, RANSAC exhibits certain limitations, such as sensitivity to parameter settings and computational complexity, which necessitate careful consideration and optimization in practical applications. Nonetheless, RANSAC remains a versatile and indispensable tool for robust estimation and model fitting in a wide range of real-world scenarios [5].

### 3.6 Values to compare the matching process

In the context of point cloud registration, where the objective is to align multiple sets of 3D points into a single coordinate system, it is essential to have quantitative metrics that can evaluate the quality of the registration. Among the commonly used metrics are Final Root Mean Square Error (RMSE), Fitness, Mean Distance, and Standard Deviation. These metrics provide a comprehensive understanding of how well the point clouds have been aligned and help in assessing the accuracy and precision of the registration process.

#### **Final Root Mean Square Error (RMSE):**

The Final RMSE is a measure of the average distance between corresponding points in the registered point clouds. It is calculated as the square root of the mean of the squared differences between the points. Mathematically, for two sets of corresponding points  $\mathbf{P}$  and  $\mathbf{Q}$ , the RMSE error is given by:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N \|\mathbf{P}_i - \mathbf{Q}_i\|^2}$$

where  $N$  is the total number of points, and  $\|\mathbf{P}_i - \mathbf{Q}_i\|$  represents the Euclidean distance between the corresponding points  $\mathbf{P}_i$  and  $\mathbf{Q}_i$ . The Final RMSE is crucial because it gives a single scalar value that summarizes the overall alignment error. A lower RMSE value indicates a better fit between the point clouds, implying that the registration has been performed more accurately. This metric is widely used due to its simplicity and effectiveness in providing a quick assessment of the registration quality [23].



**Mean Distance:**

Mean Distance measures the average Euclidean distance between each pair of corresponding points in the registered point clouds. It is calculated as:

$$\text{Mean Distance} = \frac{1}{N} \sum_{i=1}^N \|\mathbf{P}_i - \mathbf{Q}_i\|$$

Unlike RMSE, which squares the distances, the Mean Distance provides a more direct measurement of the average error. It is particularly useful for understanding the overall displacement or shift between point clouds. A lower mean distance suggests a closer alignment, making it a valuable metric for evaluating the quality of the registration [23].

**Standard Deviation:**

Standard Deviation in the context of point cloud registration refers to the variability or spread of the distances between corresponding points after registration. It is defined as:

$$\text{Standard Deviation} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\|\mathbf{P}_i - \mathbf{Q}_i\| - \text{Mean Distance})^2}$$

This metric is important because it provides insights into the consistency of the registration. A low standard deviation indicates that the distances between corresponding points are similar across the entire dataset, suggesting uniform alignment. Conversely, a high standard deviation reveals variations in alignment quality across different regions of the point clouds, which could indicate potential issues with the registration process [23].

**Fitness:**

Fitness refers to a metric that quantifies how well two point clouds are aligned based on the number of good matches or correspondences between the points in the source and target point clouds. Fitness measures the quality of this alignment by looking at how many points from the source point cloud have close correspondences in the target point cloud after applying the transformation (rotation and translation). Fitness is typically defined as the fraction of points in the source point cloud that finds a corresponding point in the target point cloud within a certain distance threshold. Mathematically, it can be expressed as:

$$\text{Fitness} = \frac{\text{Number of inlier correspondences}}{\text{Total number of points in the source point cloud}}$$

Where *Number of inlier correspondences* are pairs of points (one from the source and one from the target point cloud) that are close enough to each other after applying the current transformation. The "closeness" is determined by a predefined distance threshold, and *Total number of points in the source point cloud* is the number of points in the source point cloud.

**High Fitness:** A high fitness value (close to 1) indicates that a large proportion of the points in the source point cloud have found good matches in the target point cloud. This

suggests a good alignment between the two point clouds.

**Low Fitness:** A low fitness value indicates that fewer points in the source point cloud have close correspondences in the target point cloud, suggesting poor alignment [23].

These four metrics—Final RMSE, Fitness, Mean Distance, and Standard Deviation—are vital for several reasons:

- **Comprehensive Assessment:** Together, they offer a complete picture of the registration quality. While RMSE provides a general error estimate, Fitness measures the proportion of points in the source point cloud that have corresponding points in the target point cloud within a specified distance threshold, indicating how well the point clouds are aligned. Mean Distance gives an intuitive average alignment error, and Standard Deviation indicates consistency across the point cloud.
- **Objective Comparison:** They allow for objective comparison between different registration algorithms or configurations, helping in identifying the most effective method for a given dataset. Fitness, in particular, helps in evaluating how thoroughly two point clouds are aligned by comparing the proportion of successfully matched points.
- **Optimization Guidance:** By analyzing these metrics, one can fine-tune the registration parameters to achieve the best possible alignment. Lower RMSE and Mean Distance values generally indicate better registration performance, while higher Fitness values suggest a more robust alignment with more point correspondences.
- **Error Diagnosis:** High values in RMSE or Mean Distance, low Fitness, or large Standard Deviation can help diagnose problems with the registration, such as noise, outliers, or incorrect correspondences, which may require further refinement or preprocessing [18].

In summary, the use of Final RMSE, Fitness, Mean Distance, and Standard Deviation as evaluation metrics provides a robust framework for assessing and comparing the quality of point cloud registrations. These metrics help ensure that the alignment is both accurate and consistent, which is crucial for applications that rely on precise 3D modeling and reconstruction [1].

### 3.7 Combining Photogrammetry and Multibeam

A combination of two techniques is known from the terrestrial and aerial background. There it is explored in what way Lidar and Photogrammetry can be combined to get better and faster results for a final point cloud. A similar idea came up for the combination with Multibeam and Photogrammetry. The idea was to take the pictures from the Divers that were used to generate the 3D model and color the Multibeam point cloud with them. To achieve this, different methods were explored. The main issue is that the position of the camera in respect to the Multibeam point cloud needs to be

known. Therefore, during the registration process the transformation matrix needs to be calculated. With the transformation matrix known, the Multibeam point cloud can be changed, so that the location of both planes are at the same position and the point clouds match. The position of the camera in x, y and z can be plotted directly in the Multibeam point cloud. As a second step, it is important to know how the camera is angled. Finding the values for the angles is much more difficult. However, if both the location and angles are known, it should be possible to place the picture in the correct position and then rotate it accordingly [26].

With the camera position known in the room, the next steps are the camera intrinsic and extrinsic parameters that need to be added to the code. In computer vision, understanding the geometry of image formation is crucial for tasks such as 3D reconstruction, camera calibration, and object recognition. The process of capturing an image involves projecting a three-dimensional scene onto a two-dimensional image plane, and this transformation is governed by both intrinsic and extrinsic camera parameters [26].

### 3.7.1 Intrinsic Parameters

The intrinsic parameters of a camera define the internal characteristics of the camera's lens and sensor, which influence how the camera captures an image. These parameters are specific to each camera and include:

- Focal Length (f): This parameter determines the camera's field of view and the scale at which objects are projected onto the image plane. It is often represented as two values,  $f_x$  and  $f_y$ , corresponding to the focal lengths in the x and y directions of the image plane.
- Principal point (c): The principal point ( $c_x, c_y$ ) is the point on the image plane where the optical axis intersects it. Ideally, this point is at the center of the image, but in practice, it may deviate slightly due to lens imperfections.
- Skew Coefficient (s): This parameter accounts for the angle between the x and y axes of the image plane. In most cases, the skew coefficient is zero, meaning that the axes are perpendicular.
- Distortion Coefficients: These parameters model the radial and tangential distortions caused by the lens. Radial distortion occurs because light rays bending near the edges of the lens cause straight lines to appear curved. Tangential distortion happens when the lens and image plane are not perfectly parallel [26].

Mathematically, the relationship between the 3D coordinates of a point  $(X, Y, Z)$  and its 2D image coordinates  $(u, v)$  is given by the intrinsic camera matrix  $K$ , which is defined as:

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

### 3.7.2 Extrinsic Parameters

The extrinsic parameters define the position and orientation of the camera in the world coordinate system. They describe how the camera is situated in the scene, which is crucial for understanding the perspective from which the image is captured. The extrinsic parameters include:

- **Rotation Matrix (R):** This  $3 \times 3$  matrix describes the orientation of the camera relative to the world coordinate system. It rotates the world coordinates into the camera's coordinate frame.
- **Translation Vector (T):** This  $3 \times 1$  vector describes the position of the camera in the world coordinate system. It translates the origin of the world coordinate system to the camera's position [26].

The extrinsic parameters form the transformation matrix that converts 3D world coordinates  $(X_w, Y_w, Z_w)$  into 3D camera coordinates  $(X_c, Y_c, Z_c)$

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = R \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix}$$

### 3.7.3 Complete Projection Model

The complete process of mapping a 3D world point to a 2D image point involves both intrinsic and extrinsic parameters. The projection can be expressed as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \cdot [R \quad T] \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

Where  $[R \quad T]$  is the concatenation of the rotation matrix and translation vector, forming a  $3 \times 4$  extrinsic matrix. This equation summarizes the entire imaging process, converting world coordinates into image coordinates [26].

## WORK DEVELOPMENT AND RESULTS

### Contents

4.1	Work Development to get the MBES point cloud . . . . .	<b>38</b>
4.1.1	Side Scan Sonar data in 3D . . . . .	39
4.1.2	Pre-processing AUV mission . . . . .	40
4.1.3	Post-Processing AUV Multibeam data . . . . .	42
4.2	Registration Process . . . . .	<b>46</b>
4.2.1	Registration in CloudCompare . . . . .	46
4.2.2	Registration in Open 3D Libraries . . . . .	48
4.2.3	Tuning parameter in Open 3D Libraries . . . . .	52
4.2.4	Tuning Results in Open 3D Libraries . . . . .	54
4.3	Results Coloring . . . . .	<b>58</b>
4.3.1	Pictures taken from existing point cloud . . . . .	58
4.3.2	Pictures taken by Divers . . . . .	62
4.3.3	Tests with Agisoft . . . . .	63

In the following section, a detailed description of the work, the achieved results, and the obstacles encountered are presented. These are listed in chronological order, from the Autonomous Underwater Vehicle (AUV) preparation through the post-processing to the data analysis and comparing.

### 4.1 Work Development to get the MBES point cloud

The section explains how the Multibeam Echosounder (MBES) and Side Scan Sonar (SSS) data is collected and the various steps leading to the final result. In chapter 2, the project timeline is displayed, that this chapter follows.

### 4.1.1 Side Scan Sonar data in 3D

As an initial test to delve deeper into the post-processing program *SonarWiz*, an attempt was made to generate a point cloud from SSS data. Once again, a plane was used, the data collection process was described in the paper by [14]. The final dataset was acquired during a dive in 2020 for high-resolution documentation at 1600 KHz while maintaining a distance of 6 m from the seabed. The aircraft identified is a Fairey Fulmar, which played a crucial role in convoy protection to and from Malta. Four lines of SSS data were collected from this aircraft during the mission.

The data is illustrated in figure 4.1. It is evident that there are four distinct files along the same axis, overlapping each other.

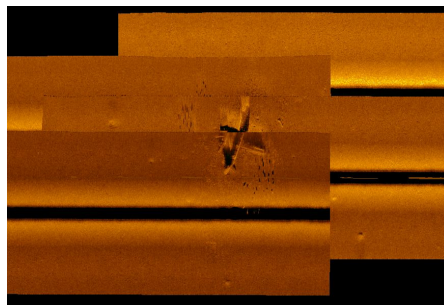


Figure 4.1: Side Scan Sonar data of the Fairey Fulmar

The subsequent step involved integrating the SSS image with the actual depth data obtained from the AUV. The actual depth of the seafloor can be determined by combining the sensor depth from a pressure sensor with the altitude. When both datasets are merged within a 3D viewer, the resulting visualization is depicted in figure 4.2.

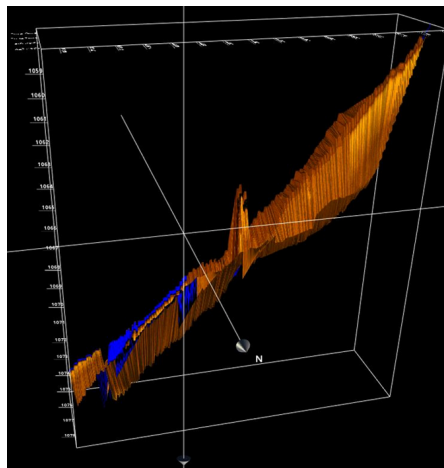


Figure 4.2: 3D view of the Side Scan Sonar data

It is evident that the results are unsatisfactory. One reason for this discrepancy is that SSS is not intended for use in three dimensions. Rather, it is a technology employed

to capture two-dimensional images of the seafloor, utilizing shadows and varying colors to roughly indicate the height of objects on the seabed. To achieve better results with SSS data, additional lines are required along different axes.

From this observation, it can be inferred that the existing SSS data is not suitable for producing satisfactory outcomes in a three-dimensional environment.

#### 4.1.2 Pre-processing AUV mission

As a first step, the different missions for the AUV were planned in the Software *Control center*, and the parameters were adjusted accordingly. The mission was always prepared in advance, with only minor changes anticipated, primarily depending on local conditions, such as nearby fishing gear. Additionally, the state of the seafloor is checked beforehand, if any data is available.

As an example, the mission for the JU88 South is used in the following section. For other planes, nearly the same parameters are applied, with only the width and location being changed. In the figure 4.3, the GUI for the *Gavia Control Center* is shown. It is evident that the mission requires planning both a descent and ascent, as well as defining the area to be surveyed. To achieve optimal results, the AUV passed over the plane from two different angles in a crosshatch pattern. This pattern was chosen because, tests on another plane demonstrated that only parallel lines can be post-processed together, and from a time management perspective, two directions provide a sufficiently good result within an acceptable time frame.

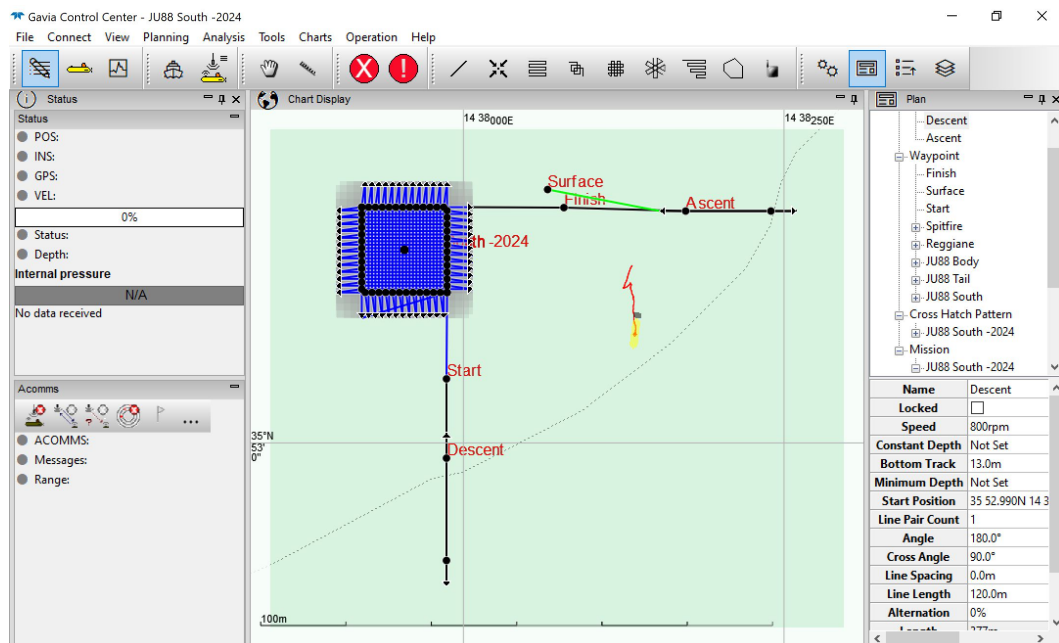


Figure 4.3: Mission planning AUV JU88 South

Table 4.1 shows all the important parameters for the mission. One critical aspect is ensuring that there were no obstacles higher than 6 meters on the seafloor. The AUV has an obstacle avoidance sensor in the nose cone, but it needs time to adjust and can only handle smaller elevations or those with a gradual slope, depending on its speed. Additionally, it was important to cover a sufficiently large area to ensure the plane is within it. The data cannot be checked on-site, therefore it was better to plan for a larger area to avoid needing to return to the location.

Table 4.1: Mission Parameters for JU88 South

<b>Parameter</b>	<b>Value</b>
Name	JU88 South-2024
Location	35 53.122N 14 37.954E
Angle	0.0°
Width	100 m
Spacing	4.0 m
Bottom Track	6.0 m
Speed	700 rpm
Dive Time	01:55:34 seconds
Dive Length	12.2 km
Power consumption	20.4 Ah

The next two tables, 4.2 and 4.3, show the specific values used for the SSS and MBES sonars. These values needed to be adapted to the height above the seafloor. Additionally, the survey lines required sufficient overlap so that in the post-processing, the edges with outliers could be cut out while still maintaining enough overlap for a good display of the data.

Table 4.2: Mission Parameters for JU88 South: BlueView Bathymetric Sonar

<b>Parameter</b>	<b>Value</b>
Start range	3.0 m
End range	8.5 m
Ping period	50 ms

Table 4.3: Mission Parameters for JU88 South: EdgeTech Side Scan Sonar

<b>Parameter</b>	<b>Value</b>
Frequency	Low (600 kHz)
Range	30.0 m



Primarily due to unstable weather conditions and large waves, the data collection phase was started later than expected, when conditions were more favorable. For data collection at sea, a risk assessment was generated, and a mission protocol was completed. The risk assessment was a general document for surveying days from a dive boat. The mission protocol contained important information and observations for the mission. Both can be found in the appendix under A.3 and A.2

During the data collection at sea, there were no issues. One observation was that the AUV required a wave to submerge, as it was a calm day without waves and not enough ballast was added to the vehicle. Additionally, there was a significant depth variation when passing over the plane, which could also be related to the ballast. Otherwise, it was a quiet day with no other issues, and the connection via Ultra-short baseline underwater positioning system (USBL) to the AUV was stable throughout the mission. This means there was a regular update of the AUV's status approximately every 30 seconds in the monitoring software. The status update included internal pressure, calculated position, depth, and height above the seabed.

#### 4.1.3 Post-Processing AUV Multibeam data

As the next step, after the data were successfully collected at sea, it needed to be cleaned and post-processed. The post-processing was done using the software *SonarWiz*. To import the files into this software, their format needed to be changed from .son to .s7k. This conversion was performed using the *Teledyne PDS Control Center* software. The reason for this was that .son files can only be opened in the internal Teledyne software, and there was no license for additional post-processing options.

After changing the format, the data were imported into *SonarWiz*. It was possible to import both SSS data and MBES files.

*SonarWiz* already has various functions for post-processing implemented. A more detailed description of these functions and the program in general can be found in the **User Manual** [6].

There was no clear guide on how to post-process the data step by step. Therefore, it involved trying different filters to see how they affected the data. It was important to note that the same filters do not work the same way for all the planes, which made this process take longer than planned.

In this chapter a description of the process is made, additionally in figure 4.4 the process is visually presented.

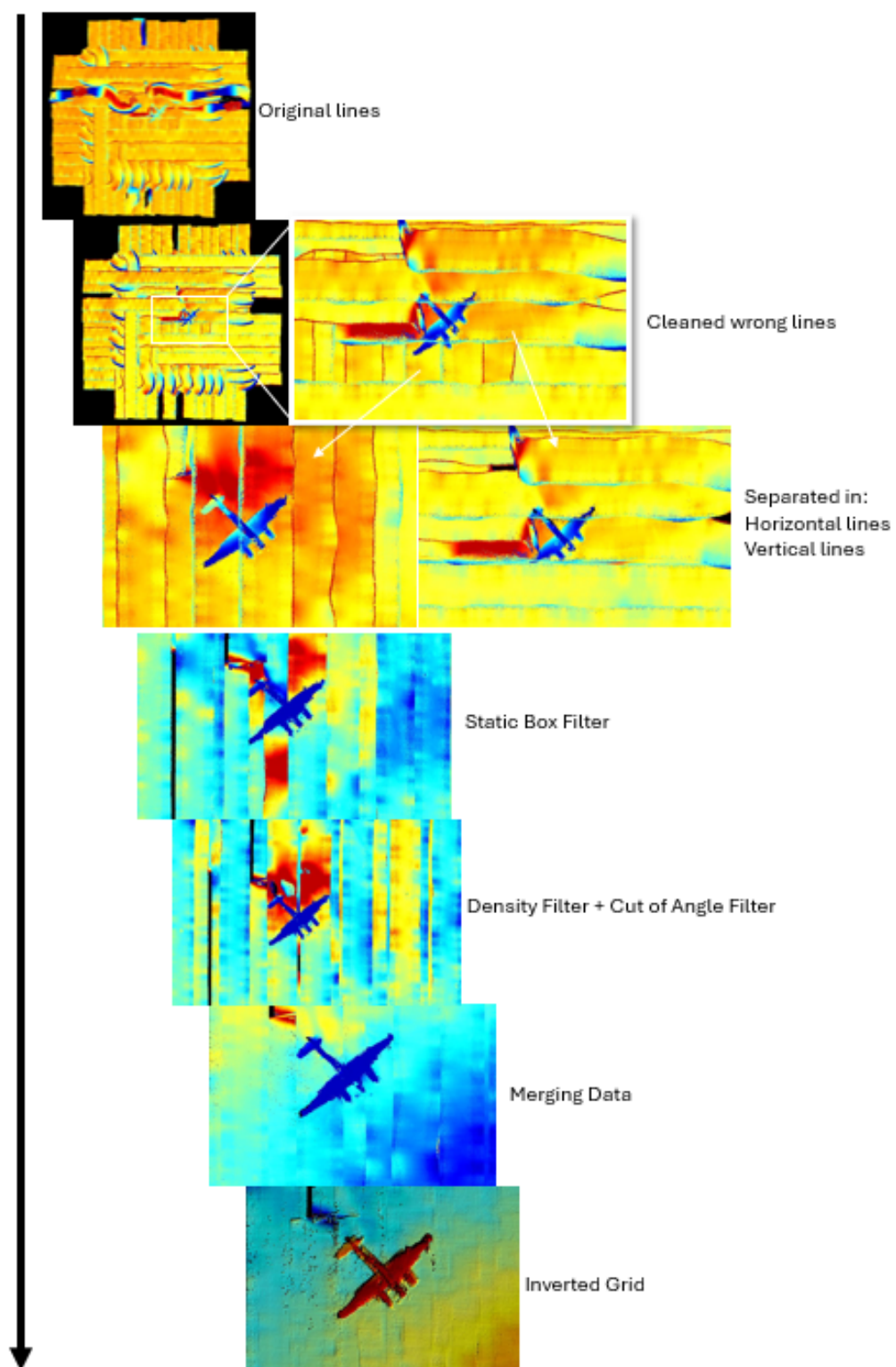


Figure 4.4: Steps during the cleaning process

When importing the data, the correct coordinate system (WGS84 UTM33N) and the appropriate import configuration must first be selected. In the next step, the vessel with the sensors was configured in the project. The tide and sound velocity can be neglected because the sensors were mounted on an underwater vehicle that flies over the seafloor at a fixed height [6].

After setting up the project parameters, all the files were imported into the project. In the first step, lines that were completely off and or don't hold any important detail were deleted. In the case of the JU88 South the lines were offset, possibly due to the suboptimal ballast and the presence of strong currents. In the next steps, they needed to be filtered and merged. Merging was necessary to compensate for time, position, and attitude offsets between the lines [6].

Several filtering options were available, but only the ones applied are described here. One filter that helped remove outliers and peaks at the edges was the *Static Box Filter*. It flags soundings that were shallower than the minimum depth or deeper than the maximum depth. The filter can also flag soundings that were to the port of the minimum horizontal range or to the starboard of the maximum horizontal range. Horizontal ranges were negative to port and positive to starboard [6]. For example, for the JU88 South plane, the horizontal swath range was reduced from -4 till 4 m. Therefore, the swath width was cut off to a maximum of 8 m instead and all outliers that were bigger deleted [6].

Additionally, a height filter was implemented for the depth measurement so that outliers below the seafloor or far above were removed. Mainly because the AUV did adjust his height over the plane strongly, that lead to uneven MBES lines [6].

Moreover, the *Cutoff Angle Filter* was applied, which flags all soundings between the inner minimum cutoff angle and the inner maximum cutoff angle (the nadir region). It also flags all soundings beyond the outer minimum cutoff angle and beyond the outer maximum cutoff angle (the outside beams) [6].

To remove as many outliers as possible also the *Sample Density Filter* was used. It divides the swath cross-section into bins. The filter flags all soundings in bins that do not have the minimum number of samples in them. There were two types of bins equidistant and equiangular: Equidistant bins are defined by horizontal and vertical bin size and Equiangular bins are defined by angle and range [6].

An observation made during the merging process was that the positioning was not accurate, like you can see in figure 4.5. Therefore, it was only possible to merge parallel lines, but not lines with different angles. It is clearly visible that the final position of the plane is in different locations than in the merged patches. There was no way to determine the true coordinates versus measurement errors. It was assumed that the first

pass is more accurate, so this data batch was used for further work if it had sufficient quality.

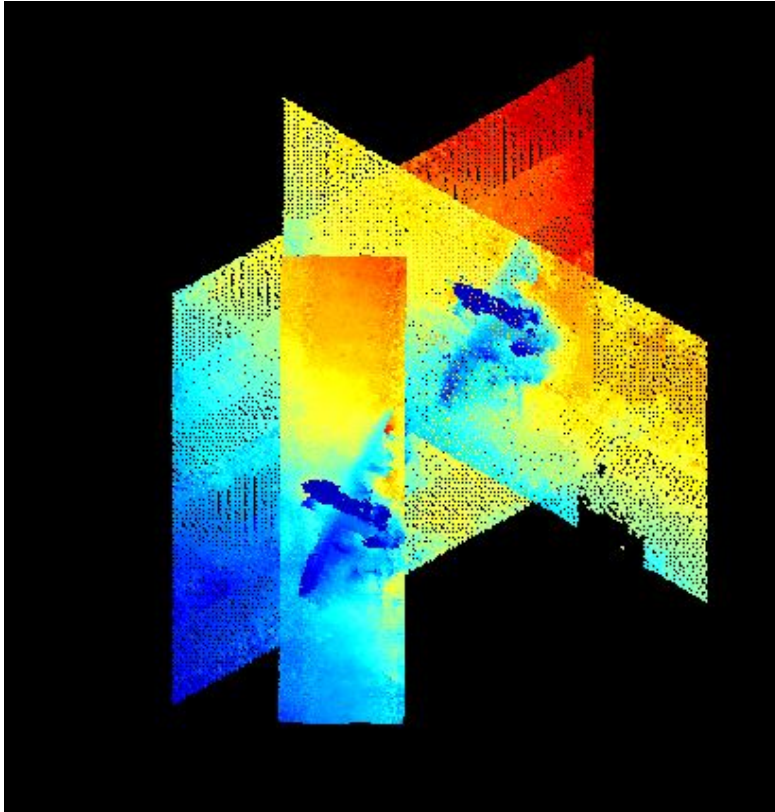


Figure 4.5: Position Offset Spitfire

Another observation made during the post-processing was that the model was displayed 180° rotated and needed to be inverted.

The final step to clean the data as thoroughly as possible was to create a depth grid with only a small area around the plane. This removed most outliers from the file, leaving a merged and filtered small bathymetric point cloud that can be exported as a .xyz file. The document contained three columns: WGS84 UTM33 coordinates for x and y, and the depth in negative values for z. The .xyz file can then be used in the next step for further comparison with other point clouds.

## 4.2 Registration Process

The following section describes the steps to compare point clouds and register them. Starting with the different programs and then different methods. For the registration methods, the first test was done with the same point cloud but one manually changed. The python code can be found in the appendix.

### 4.2.1 Registration in CloudCompare

The initial tests were conducted using the existing Software *CloudCompare*. Chapter 3.4.1 the fundamentals of registration and explained the values for comparison. The software included a basic Iterative Closest Point (ICP) registration function. To achieve optimal results, first the center of the bounding boxes for the point clouds was aligned. In the next step, the pre-programmed ICP algorithm was applied. This function needed to be reapplied until both point clouds were fully aligned. For comparison purpose, the values: Final Final Root Mean Square Error (RMSE), Mean Distance, and Standard Deviation were used. The RMSE represented the square root of the mean square errors. Therefore, it was nearly equivalent to the standard deviation of the relative distances between the two clouds. The primary difference was that it was computed on fewer points—in this case, 50000 points.

In the first test phase, the same point cloud was used. The reason was the weather dependency, therefore the MBES data collection on the planes started later. For the first tests, the photogrammetry point cloud was compared. One point cloud was transformed and snipped by hand, the other one was not changed. The other tests were comparing the point cloud from different sources with each other. The center of the bounding boxes paired and afterward the ICP algorithm run till they equal. The amount of overlap was tested exclusively on the plane by removing the seafloor, as the size of the seafloor varied depending on the collection method.

In table 4.4 the parameters for the registration of different point cloud combinations are displayed, allowing for numerical comparison. Subsequently, figure 4.6 illustrates the differences between the point clouds using colors. Blue shows the non-overlapping areas, while green means differences.

Table 4.4: Comparing point clouds in CloudCompare

Point cloud Combination	Final RMSE	Mean Distance	Std Derivation
Photogrammetry- Photogrammetry	0.12945	0.119071	0.112123
Photogrammetry- Gavia MBES	0.21479	0.179959	0.169478
Photogrammetry- R3Vox MBES	0.51913	0.41805	0.407169
R3Vox MBES - Gavia MBES	0.151015	0.202256	0.182296

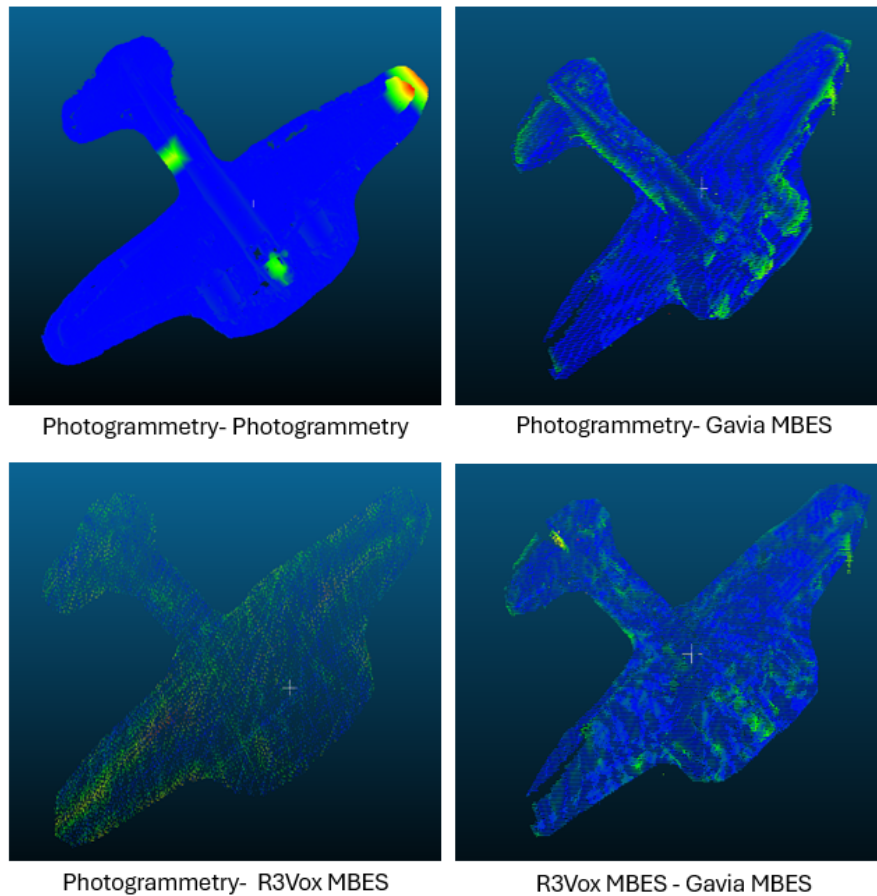


Figure 4.6: Registration in CloudCompare with ICP

The registration values for the two point clouds- Photogrammetry and Photogrammetry- with the lowest numbers were logical, as most of the points aligned in the same positions for the majority of the point cloud, given that they originated from the same source. In contrast, the other three cases exhibited different numbers of points and varying positions from cloud to cloud, which naturally increased the values for all three metrics. The largest discrepancy was observed between the photogrammetry and R3Vox MBES point clouds. One possible explanation for this discrepancy was the quality of the point cloud data: the R2Sonic Multibeam, being a surface-mounted system on a boat, was affected by approximately 0.5 meters of swell on the day of data collection, with the wreck located at a depth of 100 meters. As a result, the data quality was suboptimal. When the Gavia MBES point clouds were compared to the other two, the values fell within a similar range. This suggested that the number of points in the point cloud was intermediate compared to the others, providing a plausible explanation for the observed results.

## 4.2.2 Registration in Open 3D Libraries

Like described in 3.4.2 the library includes different functions that can be used for working with point clouds. In the following figure 4.7 the process is displayed for the final registration.

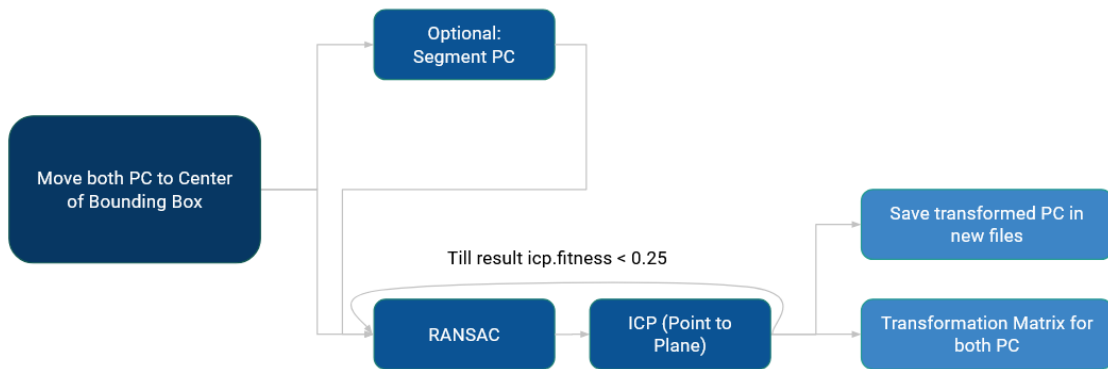


Figure 4.7: Registration process

Each box is described in the following section in a chronological order and there are given additional information to each one:

### Move both PC to Center of Bounding Box:

For the subsequent steps, the first box was modified so that only one point cloud was transformed while the other retained its original parameters. This adjustment eliminated the need to transform both point clouds each time, simplifying the process for coloring. In cases where the point cloud was already saved in a georeferenced x and y coordinate system, this step was not necessary.

### Optional Segment PC:

Some tests were conducted with segmenting the point cloud and saving it before registration. However, this approach did not significantly improve the registration process. Additionally, it was difficult to establish a general guideline for when this method would be beneficial or when it would fail to offer improvements. For the test cases, a visual inspection was used to assess its effectiveness.

### RANSAC:

The RANSAC algorithm was explained in more detail in 3.5.2. It is a random algorithm that produced different results with each execution. To standardize the process and use the values for comparison, a variable *icp.fitness* was used, which was set to be smaller than 0.25 for the registration to stop. Additionally, a manual solution was tested and repeatedly run until the fitness value was also smaller than 0.25. Once this condition was met, the values were saved in a pickle variable, and these values could then be loaded and used for the ICP registration.

The `execute_global_registration` function, as described in Listing 4.1, aligned two downsampled point clouds by performing global registration using the RANSAC algorithm with feature matching. It computed a distance threshold based on the voxel size to filter valid correspondences. The function utilized the Open3D library's `registration_ransac_based_on_feature_matching` to estimate the best transformation between the source and target point clouds. Mutual filtering was applied to refine the correspondences, and Point-to-Point transformation estimation was used, incorporating edge length and distance checkers for validation. The Point-to-Point function was used because it relied on matching feature descriptors, such as FPFH, and did not inherently require surface normals, which are necessary for the Point-to-Plane transformation. The RANSAC convergence criteria were set with high iteration and validation limits to thoroughly search for an optimal alignment. The function returned the registration result, which included the transformation matrix and inlier information.

```
1 def execute_global_registration(source_down, target_down, source_fpfh,
2   target_fpfh, voxel_size):
3     """
4     Perform global registration between source and target point clouds using
5     RANSAC based on feature matching.
6
7     Parameters:
8     - source_down: Downsampled source point cloud.
9     - target_down: Downsampled target point cloud.
10    - source_fpfh: Fast Point Feature Histograms (FPFH) features of the source
11    point cloud.
12    - target_fpfh: Fast Point Feature Histograms (FPFH) features of the target
13    point cloud.
14    - voxel_size: Voxel size used for downsampling. This parameter influences
15    the distance threshold.
16
17    Returns:
18    - result: Registration result containing the transformation matrix and
19    inlier information.
20    """
21
22    # Set a distance threshold for RANSAC. This threshold determines the
23    # maximum distance between corresponding points in the source and target
24    # point clouds to be considered as inliers.
25    distance_threshold = voxel_size * 1.5
26
27    # Perform global registration using RANSAC based on feature matching
28    result = o3d.pipelines.registration.registration_ransac_based_on_feature_matching(
29        source_down, # Source downsampled point cloud
```



```

30     target_down, # Target downsampled point cloud
31     source_fpfh, # FPFH features of the source point cloud
32     target_fpfh, # FPFH features of the target point cloud
33     True, # Use mutual filter, which helps filter out bad correspondences
34     distance_threshold, # Maximum allowable distance between correspondences
35     o3d.pipelines.registration.TransformationEstimationPointToPoint(False),
36     # Estimation method (Point-to-Point)
37     3, # Number of RANSAC iterations
38     [
39         # Check if the relative lengths of edges between corresponding
40         # points are similar
41         o3d.pipelines.registration.CorrespondenceCheckerBasedOnEdgeLength(
42             0.9),
43         # Check if the distance between corresponding points is within the
44         # threshold
45         o3d.pipelines.registration.CorrespondenceCheckerBasedOnDistance(
46             distance_threshold)
47     ],
48     # Convergence criteria for RANSAC
49     o3d.pipelines.registration.RANSACConvergenceCriteria(
50         max_iterations=8000000, # maximum number of iterations of algorithm
51         max_validation_step=400) # maximum number of validation steps
52                                 # performed for each model
53 )
54
55 # Return the registration result which contains the transformation matrix
56 # and inliers
57 return result

```

Listing 4.1: Python code for global registration with RANSAC

### ICP:

The ICP algorithm was explained in more detail in 3.5.1. For the different algorithms, both the ICP Point-to-Point and Point-to-Plane methods were tested, as both were already implemented in the library and accessible via different commands. The method that worked best and produced the most favorable results was the ICP Point-to-Plane algorithm, as determined through test runs, that will be described later.

The code: Listing 4.2 refined the alignment of two point clouds using the ICP algorithm by invoking the *registration\_icp* function from Open3D's pipeline. The ICP algorithm aligned the source point cloud with the target point cloud through an iterative process that minimized the distance between corresponding points. In this function, the *TransformationEstimationPointToPlane* method was utilized, which computes transformations based on minimizing Point-to-Plane distances. Additionally, the *ICPConver-*

*genceCriteria* was set with a *max\_iteration* parameter of 2000, limiting the number of iterations the ICP algorithm could perform. This ensured that the algorithm stopped either when a satisfactory alignment was achieved or when the maximum number of iterations was reached, preventing excessive computation time.

```
1 def refine_registration(source, target, voxel_size, initial_transformation):
2     """
3     Refines the registration of two point clouds using the Iterative Closest
4     Point (ICP) algorithm.
5
6     Parameters:
7     - source: The source point cloud to be registered (an
8     'open3d.geometry.PointCloud' object).
9     - target: The target point cloud to which the source will be aligned (an
10    'open3d.geometry.PointCloud' object).
11    - voxel_size: The size of the voxel grid used to downsample the point
12    clouds (a float).
13    - initial_transformation: The initial transformation matrix (4x4 numpy
14    array) to align the source to the target before refining.
15
16    Returns:
17    - result: The result of the ICP registration as an
18    'open3d.pipelines.registration.RegistrationResult'
19    object.
20    """
21    # Set the radius for normal estimation based on the voxel size.
22    radius_normal = voxel_size * 2
23
24    # Estimate normals for the downsampled source point cloud.
25    # Normals are needed for point-to-plane ICP.
26    source.estimate_normals(
27        o3d.geometry.KDTreeSearchParamHybrid(radius=radius_normal, max_nn=30))
28    # Estimate normals for the downsampled target point cloud.
29    target.estimate_normals(
30        o3d.geometry.KDTreeSearchParamHybrid(radius=radius_normal, max_nn=30))
31
32    # Set the distance threshold for the ICP algorithm, based on the voxel size.
33    # The threshold determines the maximum distance between corresponding points
34    distance_threshold = voxel_size * 0.4
35
36    # Perform the ICP (Iterative Closest Point) registration to refine the
37    # alignment
38    # between the source and target point clouds.
```

```

39     # - Uses Point-to-Plane ICP, which minimizes the Euclidean distance between
40     # corresponding points and a plane.
41     # - The 'max_iteration' parameter specifies the maximum number of ICP
42     # iterations (2000 in this case).
43     result = o3d.pipelines.registration.registration_icp(
44         source,                # Source point cloud
45         target,                # Target point cloud
46         distance_threshold,    # Max. distance threshold
47         for_point_pairs
48         initial_transformation, # Initial transformation matrix
49         o3d.pipelines.registration.TransformationEstimationPointToPlane(),
50         # Point-to-Plane distance minimization
51         o3d.pipelines.registration.ICPConvergenceCriteria(
52             relative_fitness= 1e-5, # Smaller value for more precise alignment
53             relative_rmse= 1e-5 ,  # Smaller value for more precise alignment
54             max_iteration=2000     # Increase for more iterations
55         )
56         # Convergence criteria for ICP
57     )
58
59     # Return the result of the ICP registration.
60     return result

```

Listing 4.2: Python code for refine registration with ICP

#### Save transformed PC in new file and Transformation Matrix for PC:

As a final result, once the point cloud was registered, it could either be saved as a file or the transformation matrix could be stored and used by another program for further modifications.

This process was initially tested on a single point cloud. The primary reason was to verify the functionality of the code and then determine the optimal parameters while testing the RANSAC and ICP algorithms. The goal was to fine-tune the parameters so that the registration process would be both reliable and reproducible. The second objective was to accelerate the process, making it as efficient as possible. To optimize the code, various parameters were tested. For instance, downsampling the point clouds reduced the number of points and improved algorithm performance. Feature computation was implemented to assist in finding correspondences between point clouds. Adjustments were made to the voxel size and thresholds to suit the density and quality of the point clouds.

#### 4.2.3 Tuning parameter in Open 3D Libraries

The following section describes which parameters can be changed in the code to speed up the process of registration or to achieve more accuracy.

**Adjusting Voxel Size:**

The voxel size was critical for down sampling and feature computation. It should be chosen based on the density and scale of the point clouds. Starting with a moderate value (e.g., 0.05) and adjust based on the point cloud size and density.

- Larger Voxel Size: Coarser downsampling, faster computation, but may lose fine details.
- Smaller Voxel Size: Finer downsampling, slower computation, preserves more details.

**Adjusting Distance Thresholds:**

Distance thresholds are used in various parts of the algorithm to determine if points are close enough to be considered corresponding. Started with 1.5 times the voxel size for RANSAC and 0.4 times the voxel size for ICP, then both were adjust as necessary.

- RANSAC Threshold: This should be set based on the scale of the point clouds. Too large can result in incorrect correspondences, too small might miss valid correspondences.
- ICP Threshold: Typically smaller than the RANSAC threshold for fine tuning.

**RANSAC Convergence Criteria:**

These criteria determine how long RANSAC will run and when it should stop. Adjusting these can help balance between speed and accuracy. it was started with the provided values and adjusted based on the complexity and quality of the initial alignment.

- Max Iterations: More iterations can improve accuracy but take longer.
- Max Validation: The number of times the algorithm validates a hypothesis.

**ICP Convergence Criteria:**

These criteria determine how long the ICP algorithm runs and when it should stop. Adjust these values based on the required precision and computational resources. The optimal settings for these thresholds depend heavily on the specific dataset and the nature of the point clouds. Experiment with different values to find what works best for the application were done.

- Max Iterations: More iterations allow the algorithm to converge more finely, but take longer.
- Relative Fitness: Set this parameter to a small value to allow the algorithm to continue refining even when the improvement in fitness (alignment quality) becomes minimal. Increasing this value will stop the algorithm earlier, potentially reducing runtime but possibly at the cost of less precise alignment.

- **Relative RMSE:** Similar to Relative Fitness, this parameter controls how small the change in RMSE needs to be for the algorithm to stop. A smaller value means the algorithm will run longer in search of an even smaller error, whereas a larger value will stop sooner.

#### **Adjusting Feature Radius:**

The radius used for normal estimation and features computation affects the feature quality.

- **Normal Radius:** Affects how normals are estimated, influencing the feature quality.
- **Feature Radius:** Affects the robustness of feature matching.

#### **4.2.4 Tuning Results in Open 3D Libraries**

The following section lists the results of the registration process and explains why different values were chosen.

##### **Comparing Point-to-Point and Point-to-Plane registration**

All tests were conducted using the JU88 South plane wreck. The two point clouds used are from photogrammetry and multibeam data, both in lower resolution. The number of points in the photogrammetry point cloud is 522,703, while the multibeam point cloud contains 556,494 points. These two point clouds were selected because, in previous tests using *CloudCompare*, they produced the best results for two different source point clouds.

The code used to obtain the results follows the algorithm described in figure 4.7, excluding the optional segmentation block. To ensure the results were as consistent as possible, RANSAC and ICP were repeatedly run until the final fitness value in ICP was smaller than 0.25. The main reason for that can be seen in figure 4.8. It shows the final registration result for fitness values that are larger than 0.25. The results are not satisfactory. Also, running only the ICP algorithm repeatedly is insufficient, as it requires a well-established initial alignment to be effective. As demonstrated in the images, this initial alignment was not adequately achieved, resulting in suboptimal registration outcomes.

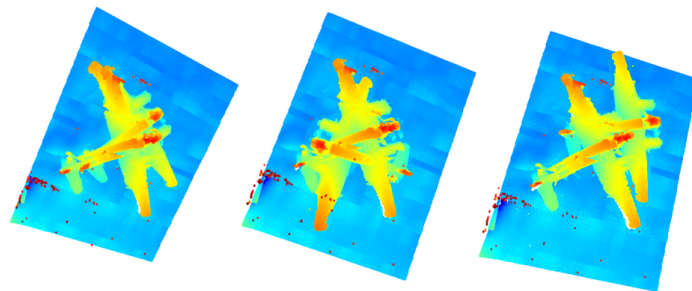


Figure 4.8: Insufficient result of Registration

The code was run multiple times for each algorithm. In the ICP block, the function alternated between the function:

*o3d.pipelines.registration.TransformationEstimationPointToPoint()* (Point in the table) and *o3d.pipelines.registration.TransformationEstimationPointToPlane()* (Plane in the table). For comparison purposes, the RMSE and correspondence set values are provided in table 4.5 for the ICP registration.

Table 4.5: Numerical comparing Point-to-Point and Point-to-Plane function

ICP registration	Fitness	Inlier RMSE	Correspondence Set
Point	0.2547	0.0274	133124
Plane	0.2572	0.0725	134440

The results in the figure 4.9 show that the Point-to-Plane algorithm performed slightly better. But it is from the visual picture hard to tell, the result look nearly the same when comparing point clouds from different sources. Only in the last picture to the right, it is possible to see the difference, when comparing the same point cloud.

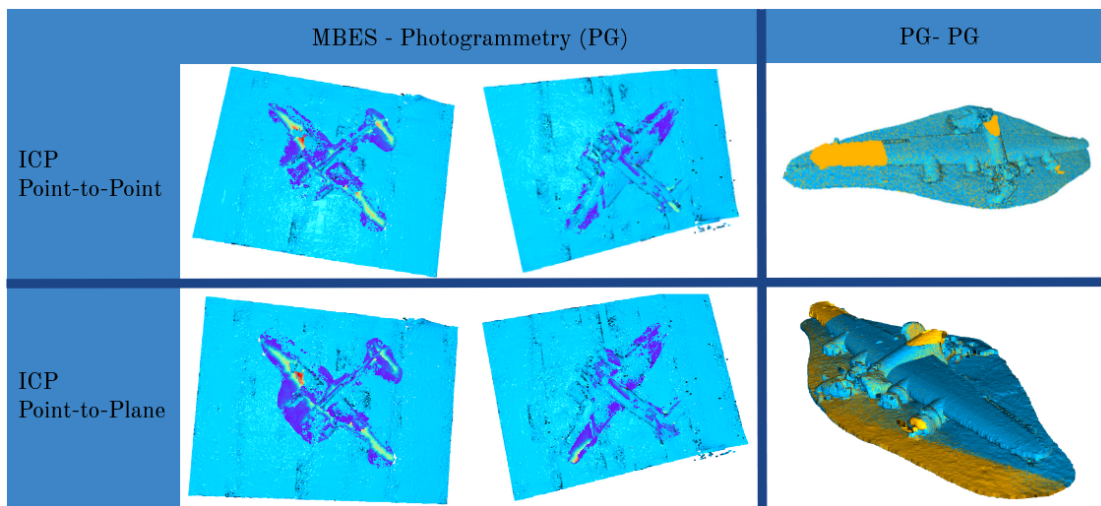


Figure 4.9: Visual comparing Point-to-Point and Point-to-Plane function

The Point-to-Plane method requires accurate surface normals and was more suited for fine-tuning alignments once a good initial alignment had been established. It minimized the distance between points and planes, which proved to be more effective when the points were already somewhat aligned.

This was confirmed through multiple test runs, with the best result displayed in the table 4.5. It is best to see when looking at the correspondence set, because again the values are nearly the same and there is not a big difference. Still, the function Point-to-Plane performed better consequently, this function will be used for future tests.

### Tuning the code

The parameters that were tuned are described in the previous chapter. In the first step, adjustments were made to the parameters for the RANSAC registration. The main challenge was that RANSAC is random, making it difficult to determine the exact influence of each change. Therefore, it was run multiple times with the recommended parameters, and the results were compared using both lower and higher values. For comparison, both visual results and numerical values (fitness, RMSE, and correspondence set) were used. To select the best value, the correspondence set needed to be large, the RMSE low, and the fitness high.

It was not always possible to find a value where all three parameters were optimal. In such cases, the correspondence set was primarily used to determine the best result. A trend was identified, and the parameters were tuned accordingly (Table: 4.6).

Table 4.6: Tuning parameters for RANSAC

<b>Tuning Parameters</b>	<b>Values</b>
Voxel size	0.05
Distance Threshold	1.5
Max. Iterations	8000000
Max. Validation	400
Normal Radius	2
Feature Radius	13

The complete table with the results (values for the correspondence set, fitness, and RMSE) for all tests with different parameter values can be found in Appendix A.1. The optimal values are also highlighted in figure A.1.

The second step involved saving the optimal RANSAC results in a pickle file, which was reloaded each time. This ensures the same starting point was used, eliminating the randomness, making it possible to numerically compare the ICP results and obtain consistent values for each run.

The following table 4.7 list the final values for the registration:

Table 4.7: Tuning parameters for ICP

<b>Tuning Parameters</b>	<b>Values</b>
Voxel size	0.2
Distance Threshold	3
Max. Iterations	5000
Normal Radius	1
Fitness	0.06
RMSE	0.08

The full table with all the results can be seen in figure 4.10.

Registration Result ICP										
<b>Normal Radius: Affects how normals are estimated, influencing the feature quality.</b>	3	2	1	0.5						
fitness	2.55E-01	2.55E-01	2.57E-01	1.13E-01						
inlier rmse	2.73E-02	2.73E-02	2.72E-02	2.81E-02						
correspondence_set size	133502	133481	134148	59299						
<b>Iterations</b>	50000000	5000000	50000	5000	500					
fitness	2.57E-01	2.57E-01	2.57E-01	2.57E-01	2.56E-01					
inlier rmse	2.72E-02	2.72E-02	2.72E-02	2.72E-02	2.72E-02					
correspondence_set size	134148	134148	134148	134148	134003					
<b>Relative fitness</b>	1E-07	1E-06	1E-05	1E-08						
fitness	2.57E-01	2.57E-01	2.56E-01	2.57E-01						
inlier rmse	2.72E-02	2.72E-02	2.72E-02	2.72E-02						
correspondence_set size	134148	134148	133998	134148						
<b>Distance threshold</b>	0.4	0.3	0.5	1	1.5	2	3	5	10	15
fitness	2.57E-01	1.55E-01	3.56E-01	5.85E-01	6.98E-01	7.76E-01	8.78E-01	9.71E-01	1.00E+00	1.00E+00
inlier rmse	2.72E-02	2.08E-02	3.31E-02	5.17E-02	6.91E-02	8.55E-02	1.17E-01	1.62E-01	1.92E-01	1.93E-01
correspondence_set size	134148	80985	186028	305996	365042	405567	458827	507436	522526	522703
<b>RMSE</b>	1E-07	1E-06	1E-08							
fitness	8.78E-01	8.78E-01	8.78E-01							
inlier rmse	1.17E-01	1.17E-01	1.17E-01							
correspondence_set size	458827	458833	458841							
<b>Voxel size</b>	0.1	0.09	0.11	0.15	0.2	0.3				
fitness	8.78E-01	8.62E-01	9.02E-01	9.57E-01	9.87E-01	9.99E-01				
inlier rmse	1.17E-01	1.09E-01	1.26E-01	1.54E-01	1.76E-01	1.90E-01				
correspondence_set size	458841	450510	471570	500014	515737	522335				

Figure 4.10: Values for Tuning ICP parameters

The best results for each test parameter are highlighted in yellow. Similar to RANSAC, the results were compared both numerically and visually. Additionally, the time required to obtain a result was evaluated. For the correspondence set, the values should be as large as possible, indicating more accurate registration and a more robust alignment. A lower RMSE suggests that inliers are closer together, which reflects precise alignment. A higher fitness value indicates that more points are correctly aligned. For the parameters "max iterations" and "relative fitness," the values remain constant until a small threshold was reached, after which they decline. Therefore, smaller values were preferable, as they result in faster outcomes. For the voxel size and distance threshold, the goal was to find a balance between quick results and good performance. These parameters are also interrelated through the following formula:

$$\text{distance threshold} = \text{voxel size} * 3$$

For the "normal radius" and the "RMSE", the correspondence set was a good initial indicator to assess the general behavior of the registration. After that, the fitness and RMSE were evaluated.



### 4.3 Results Coloring

The main concept behind the coloring process was similar to the method used in drones, where Lidar and Photogrammetry were combined to achieve better results. This approach aimed to improve the quality of the point cloud and obtain faster results.

The process, illustrated in figure 4.11, involved various tests to verify the results and evaluate how the process worked. In the first step, the existing photogrammetry point cloud was utilized, and images were captured from it. Subsequently, images taken by divers were tested to determine their applicability for coloring the point cloud. The final step for rectifying errors involved generating an entirely new photogrammetry model from the surface and using the same model and images for testing.

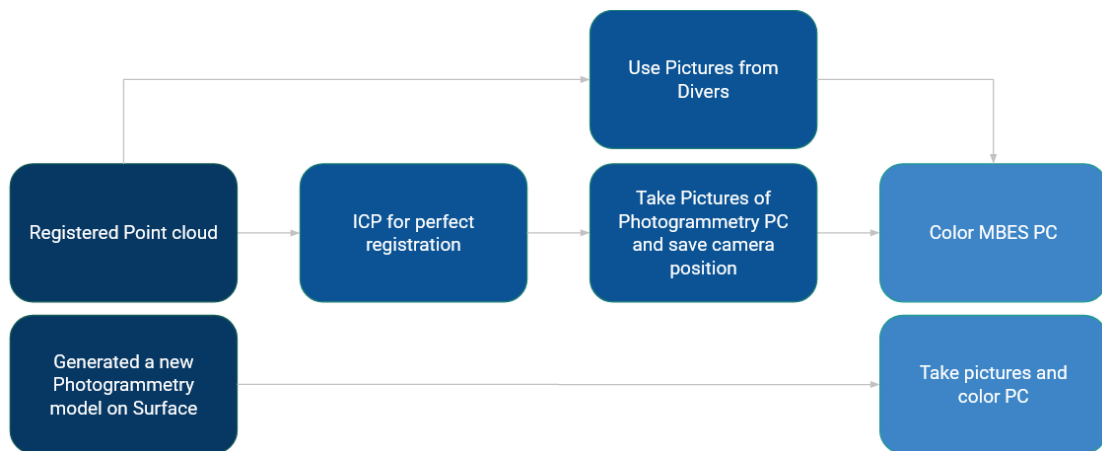


Figure 4.11: Coloring process

#### 4.3.1 Pictures taken from existing point cloud

In the initial tests, the existing photogrammetry point cloud was used, and images were captured from it. This involved loading the point cloud in Python and generating a virtual camera to take various pictures with known positions, rotations, and intrinsic and extrinsic parameters. Starting with the two registered point clouds, the position was verified by running ICP registration again on the multibeam point cloud. After aligning the multibeam point cloud, images were captured from the photogrammetry point cloud. These images were then projected onto the multibeam point cloud, with each point being colored based on the calculated color. To achieve this, the camera position, rotation, and parameters needed to be known.

The final result was illustrated in figure 4.12, with the camera positions marked with red dots in the bottom line. In the three resulting models, it is clearly visible that the plane can be nearly perfectly reproduced and the colors are looking close to the original model. Therefore, with a known camera position, the MBES point cloud can be colored.

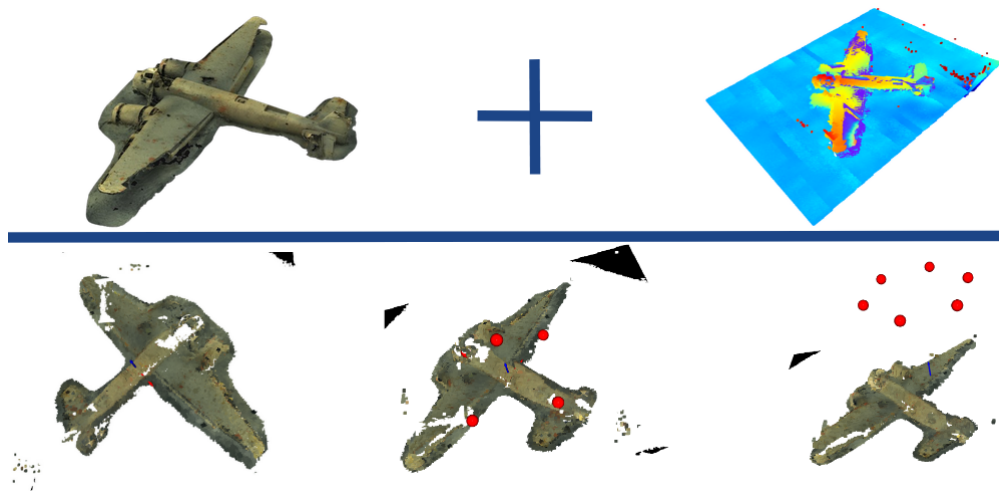


Figure 4.12: Colored multibeam point cloud with camera position

To verify that the model was recreated with, different camera position and numbers of cameras and pictures taken of the original model.

The following section, Listing: 4.3 describes the code for capturing images from the 3D point cloud. The input consists of the point cloud and the camera trajectory, and the output is the set of images that will be used in the next step.

```

1 def capture_images_from_trajectory(pcd, trajectory, dimension=640):
2     """
3     Captures RGB images from a 3D point cloud at different viewpoints along a
4     trajectory.
5
6     Parameters:
7     - pcd (open3d.geometry.PointCloud): 3D point cloud from which images are
8     captured.
9     - trajectory (list of open3d.camera.PinholeCameraParameters): A list of
10    camera viewpoints defining the trajectory.
11    - dimension (int, optional): The resolution of the captured images
12    (default is 640x640).
13
14    Returns:
15    - images (list of np.ndarray): List of captured RGB images as float buffers.
16    """
17    vis = o3d.visualization.Visualizer()
18    print(f"Creating_visualizer_window_with_dimension:_{dimension}x{dimension}")
19    vis.create_window(visible=False, width=dimension, height=dimension)

```

```

20 vis.add_geometry(pcd)
21 images = []
22
23 for idx, viewpoint in enumerate(trajjectory):
24     # Set camera parameters
25     ctr = vis.get_view_control()
26     ctr.convert_from_pinhole_camera_parameters(
27         viewpoint, allow_arbitrary=True)
28     vis.poll_events()
29     vis.update_renderer()
30
31     # Capture image
32     image = vis.capture_screen_float_buffer(do_render=True)
33     images.append(np.asarray(image))
34
35     # Save the images for review
36     plt.imsave(f'image_{idx}.png', np.asarray(image))
37
38 vis.destroy_window()
39 return images

```

Listing 4.3: Python code for capturing images

This function Listing: 4.4 is used to color the multibeam point cloud using the existing images. The inputs include the images, the camera trajectory, and the point cloud. The output will be the colored point cloud. It is important to note that accurate extrinsic and intrinsic camera parameters are necessary for a good result. Additionally, the point cloud must be adjusted according to each new camera position.

```

1 def project_colors(pc2, images, trajectory, dimension=640):
2     """
3     Projects the colors from one 3D point cloud (pc1) onto another 3D point
4     cloud (pc2) using the captured images, depth maps, and viewpoints. Rotates
5     the points before projection.
6
7     Parameters:
8     - pc2 (open3d.geometry.PointCloud): Target point cloud onto which colors
9     are projected.
10    - images (list of np.ndarray): List of RGB images captured along trajectory.
11    - trajectory (list of open3d.camera.PinholeCameraParameters): A list of
12    camera viewpoints used for projection.
13    - dimension (int, optional): The resolution of the images and depth maps
14    (default is 640x640).
15

```

```

16 Returns:
17 -pc2 (open3d.geometry.PointCloud): The second point cloud with colors
18 projected from the first.
19 """
20 pc2_colors = np.zeros((len(pc2.points), 3))
21 pc2_points = np.asarray(pc2.points)
22
23 for image, viewpoint in zip(images, trajectory):
24     intrinsic = viewpoint.intrinsic
25     extrinsic = viewpoint.extrinsic
26     fx, fy = intrinsic.get_focal_length()
27     cx, cy = intrinsic.get_principal_point()
28
29     # Project colors onto the pc2 points
30     for i, point in enumerate(pc2_points_rotated):
31         # Transform the 3D point into the camera frame
32         p = extrinsic @ np.append(point, 1.0)
33
34         # Skip points that are behind the camera or at zero depth
35         if p[2] <= 0:
36             continue
37
38         # Project the 3D point to the 2D image plane
39         u = int(fx * p[0] / p[2] + cx)
40         v = int(fy * p[1] / p[2] + cy)
41
42         if 0 <= u < dimension and 0 <= v < dimension:
43             color = image[v, u, :3]
44             pc2_colors[i] = color
45
46 pc2.colors = o3d.utility.Vector3dVector(pc2_colors)
47 return pc2

```

Listing 4.4: Python code for coloring point cloud

This code Listing 4.5 can be found in the main file of the script. It generates the camera trajectory and then calls the two functions described above.

```

1 # Generate a trajectory of viewpoints
2 for i in range(6):
3     # Create a new camera parameter object
4     view = o3d.camera.PinholeCameraParameters()
5
6     # Define the intrinsic parameters for a pinhole camera model (image
7     # dimensions, focal lengths, principal point)

```

```

8     intrinsic = o3d.camera.PinholeCameraIntrinsic(640,640,210,210,320,320)
9     view.intrinsic = intrinsic # Set the intrinsic parameters for the camera
10
11    # Define the extrinsic parameters (4x4 transformation matrix) for camera
12    # position and orientation
13    extrinsic = np.eye(4) # Initialize as an identity matrix
14    # Set camera position in 3D space (move along a circular path with varying
15    # height)
16    extrinsic[:3, 3]=[5*np.sin(np.pi/3*i),5*np.cos(np.pi/3*i),10]
17
18    # Set camera orientation using x, y, z axes (assuming x_axis, y_axis,
19    # z_axis are predefined)
20    extrinsic[:3, :3] = np.column_stack((x_axis, y_axis, z_axis))
21    view.extrinsic = extrinsic # Set the extrinsic parameters for the camera
22
23    # Append the camera viewpoint to the trajectory list
24    trajectory.append(view)
25
26    # Capture images from the generated trajectory using the point cloud (pc1)
27    images, depth_maps = capture_images_from_trajectory(pc1, trajectory)
28
29    # Project the captured colors onto another point cloud (pc2_aligned)
30    pc2_colored = project_colors(pc1, pc2_aligned, images, trajectory)

```

Listing 4.5: Python code for generating trajectory for camera position

### 4.3.2 Pictures taken by Divers

In the next step, the goal was to use the images taken by the divers instead of capturing new images from the photogrammetry point cloud. This meant that the images used to generate the photogrammetry model, along with the camera position and rotation in x, y, and z, were uploaded. The camera position and rotation were estimated using *Agisoft* while generating the 3D model. Therefore, only one point cloud needed to be loaded into the code: the MBES point cloud. However, this point cloud had to be translated and rotated according to the matrix from the registration process. A function was added to load the images from a specific folder. Additionally, the trajectory and intrinsic and extrinsic matrices were loaded directly from a file generated by *Agisoft*. The function for translating and rotating the point cloud for coloring remained the same, but the input variables were adjusted.

Initially, the algorithm was tested with only one image, and then the number of images was increased to 30 to shorten the testing time.

## Result

The result of this test can be seen in figure 4.13. There was a significant difference between the results obtained from the images captured with Python and those taken by the divers. It is clearly visible that the colors are completely off, and it is not possible at all to see the structure of the plane. Also, the size of the images and the rotating is not fitting with the model. To explore this discrepancy in more detail, different tests were conducted.

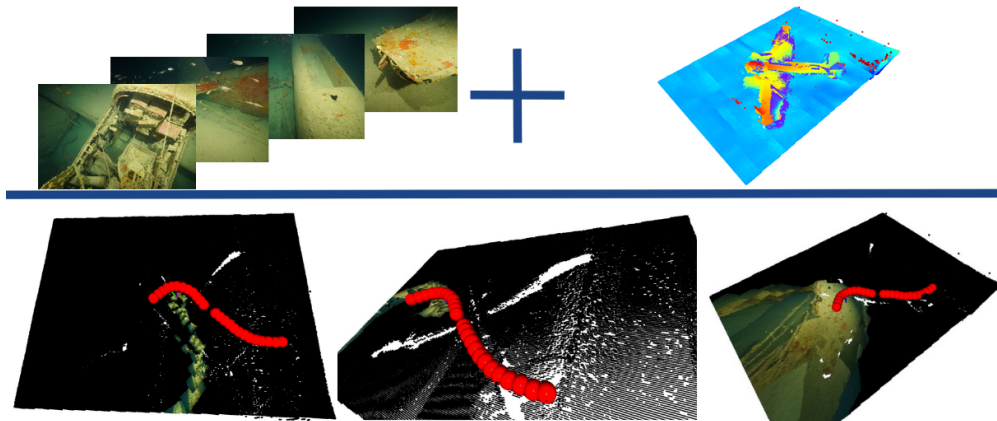


Figure 4.13: Results of coloring MBES point cloud with pictures from divers

First, it was verified that the camera position in  $x$ ,  $y$ , and  $z$  is correct, as confirmed by the red dots. In the second step, efforts were made to determine if the rotation was accurate and if the extrinsic and intrinsic parameters could be trusted. All these values were taken directly from *Agisoft*, which means they were calculated by the software. One approach was to use the known coordinates of the plane in the model and remove the translation, but this did not improve the results.

After testing with different values and not achieving better results, it was decided to start from scratch and generate a new model in *Agisoft*. This process will be described in the next chapter.

### 4.3.3 Tests with Agisoft

To determine if there is a way to improve the results and obtain more accurate values for the intrinsic and extrinsic parameters, a new model was created in a controlled environment. A table in a classroom with known scales on top was chosen for this purpose. Since the dimensions of the table and the scales were known, they could be incorporated into the software later. The final result of the table as a 3D model can be seen in figure 4.14. The idea was to use the generated point cloud as a base model and then color it with the original images taken. Some of the original pictures can be seen in the bottom of the figure 4.14.

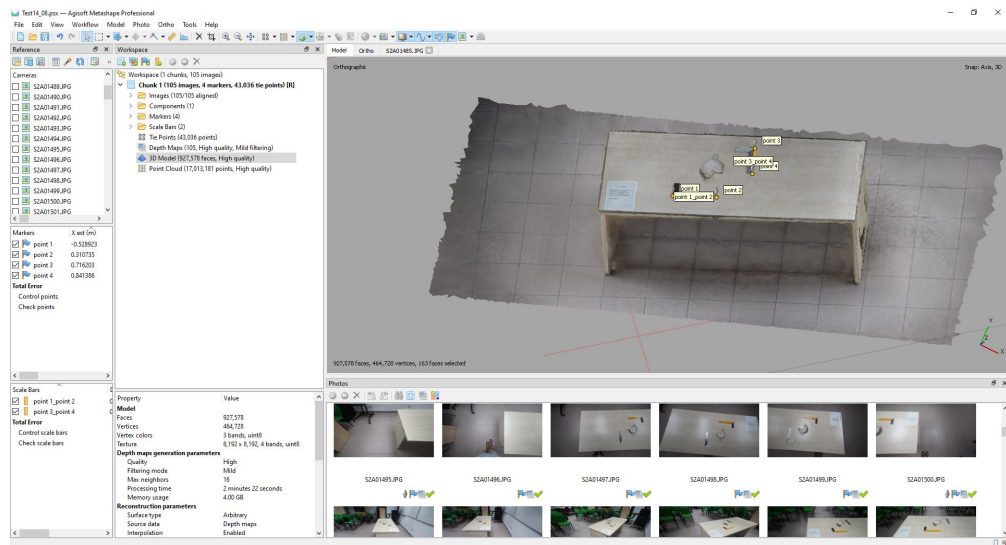


Figure 4.14: New Photogrammetry model in known environment

The model was generated with fixed scales and setting up the model with a local coordinate system, this process worked without any issues. In the next step, the calculated camera position (translation and rotation), as well as the intrinsic and extrinsic parameters, are exported to the code in python. After running, the code as usual the result is printed, shown in figure 4.15. With this additional test, it was clearly indicated that the exported parameters from *Agisoft* are incorrect. The reason was that the new model showed the same pattern already visible as a result before.

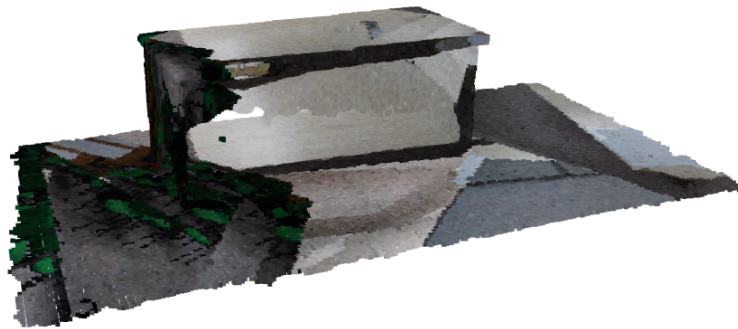


Figure 4.15: Test in Python with new model

To verify the model, different test were made on top of generating only the model. One was to move the zero point of the coordinate system to the first camera position, and export all parameters and the model again. This leads to the same result, as shown in figure 4.15. Additionally, an attempt was made using just three pictures to find an offset that could be added. However, the rotation error varies for each picture, indicating that it is not possible to obtain the exact rotation from *Agisoft*.

## CONCLUSIONS AND FUTURE WORK

### Contents

---

5.1	Conclusions . . . . .	<b>65</b>
5.2	Discussion . . . . .	<b>66</b>
5.3	Future work . . . . .	<b>67</b>

---

In this chapter, the conclusions of the work done, as well as its future work that could be done are shown.

### 5.1 Conclusions

This project aimed to explore the integration of Multibeam Echosounder (MBES) data with photogrammetry models to enhance underwater mapping, particularly focusing on submerged archaeological sites. The study demonstrated that automatic registration between the two types of point clouds is possible, though with limitations in accuracy depending on the dataset and environment. Additionally, the coloring of MBES point clouds using images proved viable, although further refinements in the process are needed to improve precision.

The results indicate that while MBES and photogrammetry serve different purposes—MBES provides accurate bathymetric data over large areas, and photogrammetry offers detailed surface imagery—the combination of these techniques has the potential to enhance both accuracy and data richness in underwater mapping. This study highlights the importance of precise registration techniques when combining different data sources. Achieving accurate alignment between MBES and photogrammetry point clouds is critical for producing reliable, integrated models. Moreover, the study emphasizes the role



of automated processes in improving efficiency and reducing the time required for data post-processing in underwater surveys.

Challenges such as aligning the two types of data, tuning the registration algorithms, getting the camera positions and being dependent on favorable weather conditions for fieldwork were faced during the project. In general, the biggest obstacle for the project, was to get an exact camera location for combining different source point clouds.

Given the objectives outlined at the beginning, this report has successfully achieved its goals of investigating the limitations and options for combining MBES and photogrammetry data. It also developed two key outputs: an automatic point cloud registration process and a method for coloring MBES point clouds with visual data, both of which contribute to improving the quality and usability of underwater mapping results.

## 5.2 Discussion

The integration of MBES data with Photogrammetry, as explored in this thesis, builds upon the growing body of research into underwater archaeological mapping. The sources reviewed in Chapter 1.3 provide crucial theoretical and practical foundations for the techniques employed in this work. Specifically, the literature demonstrates the evolving role of MBES and Photogrammetry in underwater exploration, especially in mapping submerged archaeological sites, such as shipwrecks and plane wrecks.

The research conducted in Malta (Gambin et al., 2021) [14] and the Virtual Underwater Museum [8] outlines how underwater cultural heritage sites are increasingly surveyed using Side Scan Sonar (SSS) and MBES technologies. This aligns with the goals of this thesis, which sought to leverage MBES for large-area mapping and photogrammetry for high-resolution surface details. However, as noted in studies like "*Underwater Optical and Acoustic Imaging: A Time for Fusion*" (2016) [12], the fusion of acoustic and optical methods remains technically challenging due to differences in resolution, data alignment, and environmental limitations.

This thesis confirmed many of the challenges identified in the literature. For example, while MBES is well-suited for gathering bathymetric data across wide areas, it struggles with the fine detail provided by photogrammetry. The photogrammetry models, while detailed, suffer from issues related to water turbidity and light attenuation, as previously documented. We encountered similar challenges, especially in achieving an accurate alignment between MBES point clouds and photogrammetry data. Despite using Iterative Closest Point (ICP) and Random sample consensus (RANSAC) methods for registration, the accuracy of the final models was still limited by environmental factors and the difficulty in obtaining precise camera positions.

The theoretical framework provided by sources like the book "*State of the Art in 3D Recording and Mapping*" (2018) [22] emphasizes the potential for combining MBES and

photogrammetry, but also highlights the current limitations in extrinsic calibration and feature matching. This thesis extended this research by developing an automatic registration process for aligning MBES and photogrammetry datasets, which represents a step forward in resolving some of the identified technical issues. Nonetheless, as identified by earlier researchers, the need for better optic-acoustic calibration was evident in our work. The variability in camera positioning and the inherent resolution differences between the two technologies made it difficult to achieve precise alignment without further post-processing.

The broader application of these technologies in underwater archaeology, as described in sources like "*3D Recording and Interpretation for Maritime Archaeology*" (2019) [20], highlights their value in improving the accuracy of underwater site documentation. This thesis demonstrated that, while combining MBES and Photogrammetry can enhance data richness, the practicality of deploying both technologies simultaneously—especially using an Autonomous Underwater Vehicle (AUV) requires further refinement in both hardware integration and software processing. As found in prior research, the fusion of optical and acoustic systems is still in its infancy but holds promise for improving underwater mapping, especially in less ideal conditions such as turbid waters.

The findings of this thesis both affirm and extend the current body of knowledge surrounding the combination of MBES and Photogrammetry for underwater mapping. By developing a semi-automated process for aligning point clouds and testing it under real-world conditions, this research has contributed to the ongoing effort to integrate these complementary technologies more effectively. However, consistent with the literature reviewed, more research is needed to overcome the intrinsic and extrinsic calibration issues that still limit the precision of fused datasets.

### 5.3 Future work

Future work should focus on improving the accuracy of point cloud registration, particularly in varying underwater conditions. One of the most promising avenues is enhancing the automatic alignment algorithms. In this thesis the method of tuning parameters was already explored and there is not a lot of improvement seen in this area. Therefore this could be achieved through integrating machine learning models to predict optimal registration techniques based on the specific environment and the quality of the data. By using machine learning, adaptive models could be developed to handle different levels of noise, water turbidity, and lighting conditions, which would significantly improve the robustness of the registration process.

Another area for exploration is the application of segmentation techniques to improve point cloud analysis. Segmentation could play a critical role in distinguishing different features within both MBES and photogrammetry data, helping to separate objects of

interest—such as shipwrecks or archaeological artifacts—from background seafloor noise. This project briefly explored the potential of segmentation, but it was halted due to the lack of sufficient data to effectively train a model. Future research could aim to gather more diverse datasets to address this limitation. With enough labeled data, advanced segmentation models could be trained to automatically classify various structures within point clouds, which would lead to faster and more accurate analysis, especially in complex underwater environments. These models could be used to segment the point clouds before registration, improving alignment accuracy by focusing on relevant features.

Long-term monitoring of wrecks and other underwater structures also offers promising research opportunities. By developing a dynamic model for periodic data collection, researchers could track changes in these sites over time. This would enable not only the study of environmental impacts on underwater objects but also the detection of structural decay or erosion. Such a system would require the development of more robust algorithms capable of integrating new data into existing models, allowing for seamless updates. This approach would significantly contribute to both conservation efforts and archaeological studies by offering continuous and high-precision monitoring.

A critical aspect of future work could also be exploring efficient methods for merging photogrammetry and multibeam datasets, especially in cases where data gaps exist. These gaps can occur due to environmental factors like poor visibility or technical limitations during data collection. Developing robust techniques to interpolate and merge these datasets could help create more complete and accurate 3D models of complex underwater structures, such as shipwrecks, which often feature intricate details and varying scales. Filling data gaps would ensure more reliable models and allow for better analysis and interpretation in underwater archaeology and related fields.

Finally, further refinement of the coloring process for MBES data should be pursued. Utilizing more advanced image-processing techniques, such as deep learning-based image segmentation or object recognition, could enhance the accuracy of assigning colors to MBES point clouds. Additionally, exploring more sophisticated 3D reconstruction technologies would improve the visual representation of the data, making the integration between MBES and photogrammetry models more seamless. These refinements could lead to a more user-friendly interface for exploring underwater environments, benefiting both scientific research and commercial applications like underwater inspections or virtual tourism.

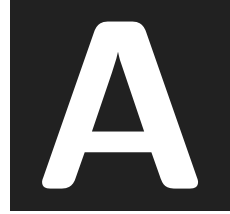
By addressing these areas, future work will not only refine the methodologies developed in this project but also broaden the scope of applications, making the combined use of MBES and photogrammetry more practical and versatile in real-world scenarios.

## BIBLIOGRAPHY

- [1] [www.open3d.org](https://www.open3d.org) 2018-2023. Open3d 0.18.0 documentation. [https://www.open3d.org/docs/release/getting\\_started.html](https://www.open3d.org/docs/release/getting_started.html). Accessed: 2024-04-16.
- [2] Company: Agisoft. Discover intelligent photogrammetry with metashape. <https://www.agisoft.com/>. Accessed: 2024-02-28.
- [3] Zhao Bao, Xiaobo Chen, Xinyi Le, and Juntong Xi. A comprehensive performance evaluation for 3d transformation estimation techniques. 01 2019.
- [4] Chandra Prakash Bathula. Machine learning concept 69: Random sample consensus (ransac). <https://medium.com/@chandu.bathula16/machine-learning-concept-69-random-sample-consensus-ransac-e1ae76e4102a>. Accessed: 2024-04-16.
- [5] Liang Cheng, Song Chen, Xiaoqiang Liu, Hao Xu, Yang Wu, Manchun Li, and Yanming Chen. Registration of laser scanning point clouds: A review. *Sensors*, 18:1641, 05 2018.
- [6] Inc. Chesapeake Technology. *SONARWIZ 7.12 User Guide*. Chesapeake Technology.
- [7] CloudCompare Community. Cloudcomparewiki. [https://www.cloudcompare.org/doc/wiki/index.php/Main\\_Page](https://www.cloudcompare.org/doc/wiki/index.php/Main_Page). Accessed: 2024-03-03.
- [8] Underwater cultural heritage. The virtual museum underwater malta. <https://underwatermalta.org/>. Accessed: 2024-02-26.
- [9] Menthy Denayer, Joris De Winter, Evandro Bernardes, Bram Vanderborght, and Tom Verstraten. Comparison of point cloud registration techniques on scanned physical objects. *Sensors*, 24(7), 2024.
- [10] Teledyne GAVIA ehf. *User Manual Gavia AUV*. Teledyne Gavia.
- [11] Teledyne GAVIA ehf. Gavia AUV, Posted on January 17, 2020. Accessed: 06.02.2024.
- [12] Fausto Ferreira, Diogo Machado, Gabriele Ferri, Samantha Dugelay, and John Potter. Underwater optical and acoustic imaging: A time for fusion? a brief overview of the state-of-the-art. 09 2016.

- 
- [13] Ben Ford, Donny L. Hamilton, and Alexis Catsambis. *The Oxford Handbook of Maritime Archaeology*. Oxford University Press, 12 2013.
- [14] Timmy Gambin, Alberto Bravo-Morata Rodríguez, and Maja Sausmekat. From discovery to public consumption: The process of mapping and evaluating underwater cultural heritage in malta. *Heritage*, 4(4):2732–2745, 2021.
- [15] Huibert-Jan Lekkerkerk Hydro International. State of the art in multibeam echosounders. <https://www.hydro-international.com/content/article/state-of-the-art-in-multibeam-echosounders>. Accessed: 2024-02-28.
- [16] Infomar. Mapping the irish seabed. <https://www.infomar.ie/>. Accessed: 2024-02-29.
- [17] Łukasz Janowski, Maria Kubacka, Mateusz Popek, and Andrzej Pydyn. Exploration and reconstruction of a medieval harbour using hydroacoustics, 3-d shallow seismic and underwater photogrammetry: A case study from puck, southern baltic sea. *Archaeological Prospection*, 28:1–16, 05 2021.
- [18] David G. Kleinbaum and Mitchel Klein. *Applied Regression Analysis*. Duxbury Press, Belmont, CA, 3rd edition, 2010.
- [19] 3D Vis Lab. 3d visualisation of marine environments. <http://www.serious-animation.com/marine/>. Accessed: 2024-02-29.
- [20] John McCarthy, Jonathan Benjamin, Trevor Winton, and Wendy van Duivenvoorde. *3D Recording and Interpretation for Maritime Archaeology*. 04 2019.
- [21] Innes McCartney. *Jutland 1916: The Archaeology of a Naval Battlefield*. 04 2018.
- [22] Fabio Menna, Panagiotis Agrafiotis, and Andreas Georgopoulos. State of the art and applications in archaeological underwater 3d recording and mapping. *Journal of Cultural Heritage*, 33, 04 2018.
- [23] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, Cambridge, MA, 1st edition, 2012.
- [24] R2Sonic. Sonic v-series 2020 /2022 / 2024 / 2026. <https://r2sonic.com/products/sonic-2026/>. Accessed: 2024-08-29.
- [25] Haiqing Si, Jingxuan Qiu, and Yao Li. A review of point cloud registration algorithms for laser scanners: Applications in large-scale aircraft measurement. *Applied Sciences*, 12(20), 2022.
- [26] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.
- [27] Wikipedia. Multibeam echosounder. [https://en.wikipedia.org/wiki/Multibeam\\_echosounders](https://en.wikipedia.org/wiki/Multibeam_echosounders). Accessed: 2024-02-28.

- 
- [28] Zhengyou Zhang. *Iterative Closest Point (ICP)*, pages 433–434. Springer US, Boston, MA, 2014.



## APPENDICES

This appendix is included to comment some aspects not considered in the rest of the work

## A.1 Tuning Results

Registration Result RANSAC											
<b>Voxel size RANSAC</b>	0.04	0.05	0.06	0.07							
fitness	8.12E-02	2.66E-01	2.73E-01	3.08E-01							
inlier rmse	4.06E-02	4.46E-02	5.39E-02	5.93E-02							
correspondence_set size	13147	28650	20869	17592							
<b>Distance Threshold RANSAC</b>	1	1.5	1.6	2							
fitness	8.25E-02	2.66E-01	2.40E-01	2.26E-01							
inlier rmse	3.51E-02	4.46E-02	4.52E-02	6.12E-02							
correspondence_set size	8867	28650	25779	24307							
<b>ransac_max_iter</b>	1000000	4000000	4500000	5000000	6000000	8000000	10000000				
fitness	8.15E-02	2.66E-01	1.90E-01	1.74E-01	1.57E-01	2.64E-01	2.22E-01				
inlier rmse	4.98E-02	4.46E-02	4.88E-02	4.53E-02	4.72E-02	4.38E-02	4.88E-02				
correspondence_set size	8768	28650	20482	18677	16881	28360	23921				
<b>ransac_max_validation</b>	350	370	380	390	400	450	500	520	550	570	
fitness	1.87E-01	2.64E-01	2.25E-01	2.54E-01	2.78E-01	2.46E-01	2.64E-01	2.04E-01	1.94E-01	2.02E-01	
inlier rmse	4.34E-02	4.54E-02	4.82E-02	4.65E-02	4.46E-02	4.92E-02	4.38E-02	4.32E-02	4.72E-02	4.68E-02	
correspondence_set size	20080	28351	24230	27261	29922	26424	28360	21962	20834	21720	
<b>Normal Radius: Affects how normals are estimated, influence feature quality</b>	0.2	0.4	0.5	1	1.5	1.9	2	2.1	2.5	3	3.5
fitness	2.31E-01	1.92E-01	2.60E-01	2.24E-01	2.39E-01	2.20E-01	2.78E-01	2.61E-01	2.57E-01	2.46E-01	2.41E-01
inlier rmse	4.64E-02	4.22E-02	4.51E-02	4.20E-02	4.63E-02	4.76E-02	4.46E-02	4.38E-02	4.64E-02	4.52E-02	4.67E-02
correspondence_set size	24874	20599	27938	24082	25693	23667	29922	28090	27635	26429	25946
<b>Feature Radius: Affects the robustness of feature matching</b>	4.5	5.5	6	6.5	7	9	11	13	15		
fitness	1.51E-01	1.62E-01	1.89E-01	1.91E-01	1.88E-01	2.64E-01	2.19E-01	2.70E-01	2.01E-01		
inlier rmse	4.63E-02	4.74E-02	4.04E-02	4.68E-02	4.38E-02	4.60E-02	4.77E-02	4.43E-02	4.55E-02		
correspondence_set size	16196	17373	20364	20554	20238	28430	23504	29066	21659		

Figure A.1: Results tuning RANSAC algorithm

## A.2 Mission Protocol JU88 South


		AUV Log. _____	
Date	23.05.2024	Location	JU88 South (Morsakala)
Area Name	JU88 South -2024		
Weather	Sunny, Flat (strong current)		
Start time on Quay	8.00	Time AUV surfaces	11.35
Time AUV in water	8.50	Time AUV recovered	14.45
Start point coord.	35 53.040N 14 37.984E	Other point coord.	35 53.167N 14 37.901E
End point coord.	35 53.160N 14 38.068E	Other point coord.	35 53.078N 14 37.897E
<b>Sidescan</b>			
Frequency	Low	Height from Seabed	6.0m
Swath Range	30m	Speed [rpm]	700
<b>Multibeam</b>			
Frequency	50ms	Height from Seabed	6.0m
Swath Range	30-8.5m	Speed [rpm]	700
<b>Camera</b>			
Observations/ Problems	<ul style="list-style-type: none"> <li>• Took 2 trys till AUV went down ↳ only with wave ↳ add more Ballast</li> <li>• AUV changed height around <math>\pm 3m</math> at the plane</li> </ul>		
AUV Operator/s	Leonie		
Crew	Timmy, Pablo, Matthias		

Figure A.2: Mission Protocol JU88 South



### A.3 Risk assessment

SJA title: Marine Operations AUV from Diving Boat Seahell		SJA no.:	Department / Discipline: UM - Maritime Archaeology	SJA responsible: Tag / Equipment no.:
Description of the work: AUV Mission around Malta and Gozo AUV Field Work		Area / Module / Deck:	Area / Module / Deck:	Area / Module / Deck:
Requirements / Preconditions:		WP / WO / Job package no.:	WP / WO / Job package no.:	WP / WO / Job package no.:
No.	Basic steps	Potential consequences	Measures	Person responsible for measures
1	Mission Planning	Loss of vehicle, loss of time/ data or wrong data	Measurements, check weather forecast, bring sufficient clothing, snacks, water, make sure robots are charged, know the ship routes/areas	Alberto, Julia, Leonie
3	Check Vehicle	Personal injury, damage AUV	wear appropriate clothing, personal protective equipment (PPE), use right tools	Alberto/Julia/ Leonie
2	Morning brief	Loss of vehicle, loss of time and data	Go over checklist and plan for the day	Julia/Leonie
3	Pre-check	Personal injury, damage AUV, loss of working day	PPE, wear appropriate clothing, use right tools	Alberto/Julia/ Leonie
4	Transport to Ship in Car	Personal injury, loss of vehicle, loss of working day	Communication between responsible persons/ Checklist/ Check traffic	Julia/Leonie
5	Load AUV and other equipment onto boat	Personal injury, damage of vehicles, dropping AUV between boat and dock	PPE, appropriate clothing, have enough people to carry everything, make several trips, Checklist	Boat crew/Students/ Leonie/Julia
7	Short travel distance: Start AUV/ Connect GPS and phone via wireless connection in harbour	AUV (not working) GPS connection takes time	Charge phone, have backup phone or battery pack, check weather	Julia/ Leonie
8	Transport to Dive site on ship	Damage/loss AUV, lost working day, personal injury, Environmental impact, Man over board, equipment overheats	PPE, know boat safety procedures, check weather forecast, secure AUV good, ensure shadow, take water	Boat skipper
7	Long travel distance: Start AUV/ Connect GPS and phone via wireless connection	AUV not working, loss of working day, damage vehicle, personal injury	Charge phone, have backup phone or battery pack	Julia/ Leonie
8	Take AUV out of frame and put it in the water	Damage AUV, Boat, personal injury, man over board	Be careful, have enough space, have enough people	Boat crew/Students/ Leonie/Julia
9	Calibrate hydrostatic parameters (buoyancy)	Loss of AUV/ data, Personal injury	PPE, check security of AUV	Julia/ Leonie
10	Remotely start AUV	Loss of AUV, loss of contact	Check weather, double check comms	Julia/ Leonie
10	Remotely start AUV	Loss of AUV/ USBL, Personal injury	Keep deck tidy, use PPE, check safety, Supervise operation and surrounding area with a buddy, file Log sheet	Boat skipper/ Leonie/ Julia
11	Monitoring mission	Loss/ tangling of AUV, loss of data, personal injury, Fault of mission.	PPE, take water, ensure shadow, Sea sickness pills, check speed of AUV, check for fishing vessels/ nets	Students/ Leonie/ Julia
12	Recovery of vehicles	Loss of AUV/data, Personal injury	Recover in coordinated teams, supervise operations, PPE	Boat crew/Students/ Leonie/Julia
13	Return to harbour	Personal injury, damage/loss AUV/data, Environmental impact, Man over board.	Contribute on navigation, PPE, Safety protocols, ensure shadow	Boat skipper
14	De-load AUV and other equipment from boat	Personal injury, damage of vehicles, dropping parts between boat and dock	PPE, appropriate clothing, have enough people to carry everything, make several trips, checklist	Boat crew/Students/ Leonie/Julia
15	Rinse AUV	Damage AUV/ data, personal injury	Work in coordinated teams when lifting equipment.	Julia/ Leonie
16	Return to UCHU	Personal injury, Damage AUV	Take breaks before and while driving, check status of the road and driver.	Julia/ Leonie
17	Download data - charge vehicle	Damage AUV parts/sensors, slip through hands	Work in clean and dry conditions, keep spare cables, monitor temperature	Julia/ Leonie
18	Debriefing	Loss of data - if not fully charged, problems with next mission	checklist, speak about problems failure	Julia/ Leonie
Is the total risk level acceptable: (Yes / No) ?				
Approval: (Signature, Date, Position, Responsible for execution)				
Conclusion / comments:				
Date / Signature				
Room				
(Appro)				
Area / Operations supervisor				
(Appro)				
Other position				
(Appro)				
Other position				
(Appro)				
Checklist for SJA applied: Tick off				
Summary of experience after completion of the work.				

Figure A.3: Risk Assessment for the AUV

## A.4 Checklist

Checklist for SJA no.:					
SJA Title:					
No.	Description	Taken care of ?			Comments
		Yes	No	N/A	
<b>A Documentation and experience</b>					
1	Is this a familiar work operation for personnel involved?	x			
2	Is there an adequate procedure/instruction/work package?	x			
3	Is the group aware of experiences/ incidents from similar activities/SJA?	x			
<b>B Competence</b>					
1	Have personnel the necessary skills for the job?	x			
2	Are there other parties that should participate in the SJA meeting?	x			
<b>C Communication and coordination</b>					
1	Is this a job where several disciplines /personnel must be coordinated?	x			
2	Is good communications and suitable means of communication in place?	x			
3	Are there potential conflicts with simultaneous activities (system/area/installation)?	x			
4	Has it been made clear who is in charge for the work?	x			
5	Has sufficient time been allowed for the planning of the activities?	x			
6	Has the team considered handling of alarm/emergency situations and informed emergency functions about possible measures/actions?	x			
<b>D Key physical safety systems</b>					
1	Are barriers, to reduce the likelihood of unwanted release/leakage maintained intact (safety valve, pipe, vessel, control system etc.)?	x			
2	Are barriers, to reduce the likelihood of the ignition of a HC leakage maintained intact (detection, overpressure protection, isolation of ignition sources etc.)?	x			
3	Are barriers to isolate leakage sources/lead hydrocarbons to safe location maintained intact (process/emergency shutdown system, blowdown system, x-mastree, drains etc.)?	x			
4	Are barriers to extinguish or limit extent/spread of fire/explosion maintained intact (detection/ alarm, fire pump, extinguishing system/equipment etc.)?	x			
5	Are barriers to provide safe evacuation of personnel maintained intact (emergency power/lightning, alarm/PA, escape-ways, lifeboats etc.)?	x			
6	Are barriers that provide stability to floating installations maintained intact (bulkheads/doors, open tanks, ballast pumps etc.)?	x			
<b>E Equipment worked on/involved in the job</b>					
1	Is the necessary isolation from energy provided (rotation, pressure, electrical voltage etc.)?	x			
2	May high temperature represent a danger?			x	
3	Is there sufficient machinery protection/shields?	x			
<b>F Equipment for the execution of the job</b>					
1	Is lifting equipment, special tools, equipment/material for the job available, familiar to the users, checked and found in order?			x	
2	Do the involved personnel have proper and adequate PPE?	x			
3	Is there danger of uncontrolled movement/rotation of equipment/tools?	x			
<b>G The area</b>					
1	Is it necessary to make a worksite inspection to verify access, knowledge about the working area working conditions etc.?	x			
2	Has work at heights/at several levels above each other/falling objects been considered?		x		
3	Has flammable gas/liquid/material in the area been considered?		x		
4	Has possible exposure to noise, vibration, poisonous gas/liquid, smoke, dust, vapour, chemicals, solvents or radioactive substances been considered?	x			
<b>H The work place</b>					
1	Is the work place clean and tidy?				
2	Has the need for tags/signs/barriers been considered?	x			
3	Has the need for transportation to/from the workplace been considered?	x			
4	Has the need for additional guards/watches been considered?			x	
5	Has weather, wind, waves, visibility and light been considered?	x			
6	Has access/escape been considered?	x			
7	Has difficult working positions, potential for work related diseases been considered?	x			
<b>I Additional local questions</b>					
1					
2					
3					

Figure A.4: Checklist for the AUV



## SOURCE CODE

Under the following link all the code and the used data can be found. It includes the Point cloud files as well as used pictures.

GitHub Link for Source Code: <https://github.com/Leo12Buch/PointCloud>



## ADDITIONAL PICTURES AND VIDEOS

Under the following link additional videos and pictures can be found. It includes a folder for getting the Multibeam data with the AUV, one to get the data with the R2Sonic Multibeam and another one for the Photogrammetry data.

Drive Link for Pictures and Videos: [https://drive.google.com/drive/folders/1RiQyww\\_eL18H1scq2Nmmyb0bLoF0oVP?usp=drive\\_link](https://drive.google.com/drive/folders/1RiQyww_eL18H1scq2Nmmyb0bLoF0oVP?usp=drive_link)