

Noémie Hirtzig

Multi-Sensors Relocation

Master's thesis in Master MIR

Supervisor: Asgeir Sørensen, Vincent Hugel, Philippe Weingertner

Co-supervisor: Oscar Pizarro, Nicolas Lalanne

January 2023

Noémie Hirtzig

Multi-Sensors Relocation



Master's thesis in Master MIR

Supervisor: Asgeir Sørensen, Vincent Hugel, Philippe Weingertner

Co-supervisor: Oscar Pizarro, Nicolas Lalanne

January 2023

Norwegian University of Science and Technology

Faculty of Engineering

Department of Marine Technology



Norwegian University of
Science and Technology

Multi-Sensors Relocation

Noémie Hirtzig

July 21, 2023

Contents

Contents	iii
Abstract	v
Résumé	vii
Figures	ix
Tables	xi
Code Listings	xiii
1 Introduction	1
2 State-Of-The-Art	3
2.1 Geometry-Based Detection	3
2.2 Mosaicing	4
2.3 Features-Based Detection	4
2.4 AI based detection	5
2.5 Partial Conclusion	5
3 Navigation and FLS data processing	7
3.1 Data Types	7
3.2 Processing Approach	8
3.2.1 Extraction	8
3.2.2 Combination	10
3.3 FLS floor areas calculation	12
3.4 Partial Conclusion	16
4 FLS and SAS Association	17
4.1 Detection	17
4.1.1 Features Detection	18
4.1.2 Physics based Detection	26
4.1.3 Partial Conclusion	31
4.2 Association	32
4.2.1 KML images superposition	32
4.2.2 Association methods ideas	34
4.2.3 Partial Conclusion	35
5 Conclusion	37
Bibliography	39
A Particle Swarm Optimisation	41
B Authorisation letter for Data use in MSc Thesis	45

Abstract

Our research has focused on exploring the integration of FLS and SAS data for target relocation, contributing to the advancement of underwater exploration and survey missions for both industrial and research applications.

Through our background research, we identified a gap in the literature concerning the combination of two different sonar modalities for object relocation. Motivated by this observation, we embarked on investigating the possibilities of combining data from distinct sonar sources to enhance object relocation capabilities.

Our approach successfully extracted, synchronised, and combined various crucial data parameters, such as attitude, latitude, longitude, altitude, position in the Cartesian world frame, FLS range, aperture, gain, and frequency. This process provided valuable insights into the floor coverage area of the FLS during the mission and created functions that can be applied in future similar data processing tasks.

Our efforts to explore and test various feature detectors for object matching in SAS and FLS images encountered challenges in achieving robust and consistent detections. As a result, we utilised the detection framework provided by the company to extract objects of interest from the data. Nevertheless, obtaining similar results proved to be challenging, leaving the association of data from the detections an open problem.

To address this association issue in the future, obtaining specifically floor-oriented FLS images may yield more promising results, enabling effective data association and fusion across different sonar modalities. Our proposed approaches, involving the use of the Hungarian Algorithm and exploring the potential of Deep Learning methods, offer potential solutions for addressing the object relocation challenge. Ultimately, our objective is to assist in the relocation of ROVs, which may lack the high-performing navigation equipment of AUVs. By achieving reliable detections and establishing markers as "anchors", akin to SLAM, we can enhance the localisation capabilities of ROVs and correct potential drifts in the navigation systems of AUVs, leading to improved localisation schemes, especially through real-time processing of FLS data.

Résumé

Ce mémoire de Master porte sur l'intégration des données FLS et SAS pour la relocalisation de cibles, contribuant à l'avancement des missions sous-marines pour applications industrielles et recherche.

Nous avons identifié une lacune dans la littérature concernant la combinaison de deux modalités sonar pour la relocalisation d'objets. Motivés par cette observation, nous avons étudié les possibilités de combiner des données de sources sonar distinctes pour améliorer la localisation d'objets.

Notre approche a extrait, synchronisé et combiné divers paramètres de données essentiels, incluant attitude, latitude, longitude, altitude, position dans le repère cartésien, portée du FLS, ouverture, gain et fréquence. Cela a fourni des informations précieuses sur la zone de couverture du sol du FLS et créé des fonctions utiles pour le traitement futur de données similaires.

Nos essais avec différents détecteurs de "features" pour la correspondance d'objets dans les images SAS et FLS ont rencontré des difficultés pour obtenir des détections robustes et cohérentes. Nous avons alors utilisé les algorithmes de détection fournis par la société pour extraire les objets d'intérêt des données. Néanmoins, il était difficile d'obtenir des résultats similaires, rendant l'association des détections un problème ouvert.

Pour résoudre ce problème à l'avenir, il serait bénéfique d'obtenir des images FLS spécifiquement orientées vers le sol et de développer un algorithme de détection SAS approprié. Cela pourrait permettre une association et une fusion efficaces des données entre les sonars. Les approches que nous proposons, incluant l'utilisation de l'algorithme hongrois et le potentiel des méthodes de deep learning, offrent des solutions potentielles pour relever le défi de la relocalisation des objets.

En fin de compte, notre objectif est d'aider à la relocalisation des ROV, qui peuvent ne pas disposer du même équipement de navigation performant que les AUV. En réalisant des détections fiables et en établissant des points clés comme "ancres" de façon similaire au SLAM, nous pouvons améliorer les capacités de localisation des ROV et corriger les dérives potentielles dans les systèmes de navigation des AUV, ce qui permet d'améliorer les schémas de localisation grâce au traitement en temps réel des données FLS.

Figures

3.1	Data integrity check plots	9
3.2	Common data check plots	11
3.3	Shape of the intersection between the floor and the beam	12
3.4	Side view of FLS beam number 1	13
3.5	Side view of FLS beam number 2	14
3.6	Top view of FLS beam in vehicle frame	15
3.7	One every nine FLS floor tile	16
4.1	ORB between successive FLS frames	18
4.2	ORB between SAS tiles with same illuminations	19
4.3	AKAZE between successive FLS frames	20
4.4	AKAZE between SAS tiles with same insonification	21
4.5	SURF between successive FLS frames	22
4.6	SURF between SAS tiles with same insonification	23
4.7	SURF between SAS tiles with opposite insonification	23
4.8	Gaussian Pyramid images matching and warping	25
4.9	Tiles with common clutter	26
4.10	Example of FLS detector application on SAS tile	27
4.11	Detections with $\beta = 110$	28
4.12	Detections with $\beta = 120$	29
4.13	First tile example	30
4.14	Second tile example	30
4.15	Google Earth image superposition	33
A.1	Optimisation runs output examples	41

Tables

4.1	ORB Matching results performance on FLS	19
4.2	ORB Matching results performance on same insonification SAS . .	20
4.3	AKAZE Matching results performance on FLS	21
4.4	AKAZE Matching results performance on same insonification SAS .	21
4.5	SURF Matching results performance on FLS	22
4.6	SURF Matching results performance on same insonification SAS . .	23
4.7	SURF Matching results performance on opposite insonification SAS	24

Code Listings

3.1	FLS files extraction	10
3.2	Timestamp conversion	10
4.1	KML data	32
A.1	Particle Swarm Optimisation	41

Chapter 1

Introduction

In recent decades, sonars have become of paramount importance in the domain of underwater target localisation and relocation. The ability to accurately locate and relocate targets underwater is crucial for various exploration and survey missions conducted in both industrial and research settings. This MSc thesis explores the concept of target relocation by leveraging data from two distinct sonar systems: the Forward Looking Sonar (FLS) and the Synthetic Aperture Sonar (SAS).

The ultimate objective is to assist in the relocation of ROVs, which may lack the high-performing navigation equipment of AUVs. The goal is to achieve reliable detections and establish keypoints as "anchors", akin to SLAM, which could enhance the localisation capabilities of the ROVs. Additionally, these approaches could be employed to correct potential drifts in the navigation systems of AUVs, leading to improved localisation schemes, especially when the correction could be done by the real-time processing of FLS data information.

Context and Relevance

In underwater exploration and survey missions, reliable target relocation techniques are essential for identifying and revisiting targets of interest. This research is relevant to the field as it addresses the need for efficient and precise target relocation, more particularly in the context of mine countermeasure operations.

Research Gap and Motivation

Surprisingly, little prior work has been done on the combination of FLS and SAS data for target relocation. This thesis aims to explore the potential of using both sonar systems in tandem. By integrating data from the FLS and SAS, we seek to develop a framework capable of associating targets previously seen only by a SAS using an FLS.

Research Objectives

The primary objective of this thesis is to develop a framework for target relocation, utilising the data obtained from the FLS and SAS systems. We aim to achieve association between the detected objects from both sonars. Our focus is on post-operational data in the context of mine countermeasure operations.

Structure

To ensure clarity, the organisation of the thesis will deviate from the chronological order of the internship work. Instead, it will present how all the components fit together to form the framework. The following sections will serve as the core components of this work:

- State-of-the-Art** This section will present a review of the existing literature and research related to underwater target relocation techniques, with an emphasis on FLS and SAS technologies.
- Data Processing** Here, we will detail the process of reading, synchronising and combining data files obtained from the FLS and navigation systems, ensuring data readiness for subsequent analyses.
- Detection** This section will delve into the techniques used for target detection from the FLS data extracted in the previous chapter and the SAS data provided through different means, highlighting the challenges and solutions encountered during the detection process. It is the first step for the process of matching FLS and SAS data.
- Association** The focus of this part will be on methods to associate data to enable the matching of detected markers between the FLS and SAS data. This is the second step of the association of FLS and SAS data

Conclusion and Future Work

In the concluding section, we will summarise the research conducted so far and present the findings of our study. Additionally, we will discuss the potential applications, limitations, and future improvements that can be made to enhance the reliability and effectiveness of the proposed target relocation framework.

By exploring the integration of FLS and SAS data for target relocation, this thesis contributes to the advancement of underwater exploration and survey missions, useful to both industrial and research communities.

Chapter 2

State-Of-The-Art

Underwater target detection, localisation, and relocation using sonars have emerged as crucial research areas in recent times. This literature review aims to examine the state-of-the-art works related to these topics, showing the advancements and challenges in the field.

A deep understanding of the existing research in underwater target detection using sonar is important before delving into the subject. This review summarises some important works in the domain.

By exploring and analysing previous works, this review aims to identify gaps in the current research landscape. It will also provide a foundation for the research conducted in this thesis.

2.1 Geometry-Based Detection

In 2011, D.P. Williams and J. Groen [1] proposed a fast and adaptive algorithm for detecting man-made objects on the seafloor using Synthetic Aperture Sonar (SAS) data. Their approach involves several steps aimed at isolating potential target echoes in the SAS images while maintaining computational efficiency.

The authors first estimate the integral image representation to facilitate subsequent calculations. Next, they estimate the reverberation level in the background and generate a shadows map, providing an estimation of shadows that could belong to objects of interest. Additionally, they retain low-quality parts for later analysis. The shadows considered interesting are marked as Regions Of Interest (ROIs). For the uncertain parts set aside, they calculate the echo to determine if it qualifies as a target. The authors also ensure that the detected echoes are not range-dependent and sort them based on signal strength. By making the detection range-independent, they use a single threshold for sorting. Finally, the resulting detections are highlighted in the result map.

This work was an advancement at the time and inspired the next one we will talk about. Galceran *et al.* (2012) proposed a method for detecting man-

made underwater objects using Forward Looking Sonar (FLS) data. Their approach involves several steps aimed at isolating potential target echoes in the FLS images while maintaining computational efficiency. Initially, they define a region of interest (ROI) within the FLS image to exclude noisy borders and reduce processing time. Next, they extract the integral image of the ROI to facilitate subsequent calculations. The authors then estimate the reverberation level in the background, generating a *background map*. Subsequently, they construct an *echo map* containing high-intensity echoes. These maps are used to identify and highlight echoes with significant intensity, followed by a geometric verification step to ensure the highlighted objects align with the desired target characteristics. This work is highly relevant to our thesis, as we adopt a similar detection framework based on their approach.

2.2 Mosaicing

In [2], Natalia Hurtos (2014) presents a comprehensive framework for creating mosaics from FLS images. The work begins with a thorough description of FLS usage and geometry. The author employs a Fourier-based pairwise registration method, known for its robustness against noise and acoustic artifacts, to align the images. To achieve global alignment and consistency among images, they focus on blending the images to reduce the signal-to-noise ratio and enhance the final mosaic's resolution.

Through extensive experimentation, Hurtos identifies photometric irregularities that may arise from the sonar imaging configuration and proposes effective strategies to minimise them. This work provides valuable insights into FLS image processing and pairwise registration, making it particularly pertinent to our thesis.

2.3 Features-Based Detection

In 2020, Tueller *et al.* [3] explore the use of feature detection on sonar images, inspired by their reliability in air environments. The authors compare various detectors, including Harris and Stephens, Shi *et al.*, STAR, FAST, SURF, SIFT, and ORB, to address the challenges of underwater feature detection, such as separating the target from the background and setting up feature parameters. They adopt a SLAM-like method that does not assume prior knowledge about the target. As the seabed can significantly influence detection and classification, they first identify the type of seabed. They compare various parameters with mean values from known seabeds, ensuring they fall within a specific threshold. Once the seabed type is determined, the appropriate detection algorithm, trained for that seabed, is used. To set the target detection parameters efficiently, the authors employ active learning. Instead of exhaustively exploring the entire parameter domain, which can be computationally expensive, it strategically samples only a subset of the parameter space, focusing on areas that are likely to yield the most

informative data. This allows the author to achieve balance between detection accuracy and computational resources.

In their work, the authors introduced a feature extraction-based method that does not need prior knowledge about the target and enables effective detection across diverse seabed types. As they also tested their results on simulated data, multibeam sonar data, and SAS data, it seemed highly relevant to our subject since we wanted to combine different types of sensors.

2.4 AI based detection

In 2017, Kim and Yu [4] addressed the challenge of real-time detection on Forward Looking Sonar (FLS) images. Their work proposes the use of Haar-like features, considering that sonar images typically exhibit both highlighted regions and shadows. To combine multiple weak detection algorithms into a robust one, they employ an AdaBoost algorithm.

To reduce the candidate region for object detection, the authors employ a series of strong detectors followed by a severe one in the back-end. They train and test their approach using both indoor tank data and real sea data. This work is relevant to our thesis as it presents an original proposal that utilises machine learning and the combination of multiple detection algorithms. It could serve as a reference for future work.

In 2021, Palomeras *et al.* [5] explore target detection and recognition in Forward Looking Sonar (FLS) data using Machine Learning techniques, particularly Convolutional Neural Networks (CNNs). Their work holds significant promise for the future of underwater target relocation.

To ensure the feasibility of their algorithm, the researchers meticulously filter out unsuitable data, excluding images that are too far from objects, blurry, or have inaccurate pitch and roll values. Their detection module effectively distinguishes between clutter and potential targets. In their experiments, the team tests and compares four CNN architectures, focusing on diversity to mitigate limitations despite the relatively small dataset. To address overfitting, they augment the dataset by applying rotations, crops, and translations to existing images. Target classification is performed using only the *target* labels in the subsequent classification step.

Palomeras *et al.* introduce a probabilistic map to enhance target classification and facilitate the identification and filtering of False Positives. As more data becomes available, this approach is expected to emerge as one of the most promising methods for underwater target relocation, underscoring its pertinence to our thesis.

2.5 Partial Conclusion

After conducting our background research, we identified various methods focused on detecting, localising, and relocating objects using single types of sonar data.

We noticed, however, a gap in the literature regarding the combination of two different types of sonar data for the purpose of object relocation.

To the best of our knowledge, there were no existing studies that explored the integration of multiple sonar modalities to enhance object relocation capabilities. Motivated by this observation, we started investigating the possibilities of combining data from two distinct sonar sources for object relocation.

Chapter 3

Navigation and FLS data processing

<p><i>Problematics:</i> How to reunite a wide variety of data into a standard type? How to assemble separated data?</p>

3.1 Data Types

To gather the necessary data for our research, access to missions possessing FLS and SAS data around the same location was required. Unfortunately, due to the limited availability of this specific data, we had to rely on files from a single mission conducted around the Dornier plane wreck in the bay of Hyères (France).

The mission was carried out using an A18D Autonomous Underwater Vehicle (AUV) equipped with a Kraken Synthetic Aperture Sonar (SAS) and a Blueview Forward Looking Sonar (FLS). These sonar systems provided the primary data for our analysis.

It is to be noted that the primary use of an FLS is to detect obstacles in front of the vehicle, not to detect objects on the seafloor. By repurposing its data for our research, we must be aware that the mission was not originally designed with this specific application in mind.

The data files provided by Exail, the organisation responsible for the mission, were in different formats. The navigation data was stored in DAT files, while the FLS data was provided in SON files. Additionally, KML and TIL files were used for the SAS data.

Despite the limitations of having data from a single mission, the availability of these files allowed us to conduct our analysis and develop our proposed framework for target relocation in the underwater environment.

The code framework to achieve this part of the work is separated into two C++ classes. The first is a generalist class called `FLSAUV` that can be used on any type of data as it handles values but is ignorant of their origin. It is this class that handles all calculations explained in 3.3. The second is called `Manipulation`, and is used

to extract and combine all the data from given files. It is not as generalist as it depends on the formatting of the files. However, given similarly formatted files or by making a few changes inside the functions, it will be able to handle them.

3.2 Processing Approach

3.2.1 Extraction

To process the data, we created various functions to extract and combine the necessary information from different file formats. For the **navigation data**, we employed the functions showed in the following snippet 3.2.1.

```
std::map<long long, double> get_lone_value(std::string filename,
    std::map<std::string, std::string> column_keywords, std::string target_column)

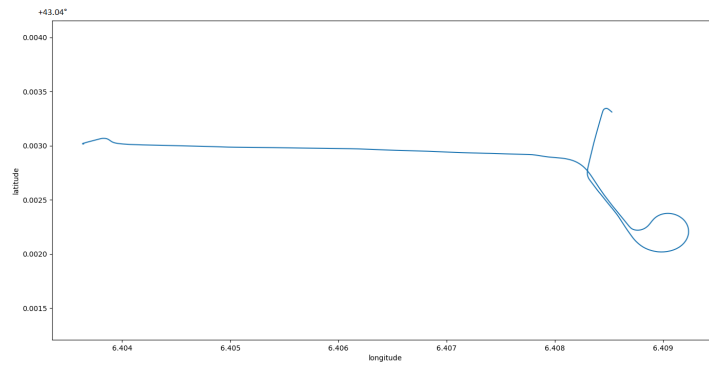
std::map<long long, std::pair<double, double>> get_pair_against_timestamp
    (std::string filename, std::map<std::string, std::string> column_keywords)

std::map<long long, double> get_rad_angle_value(std::string filename,
    std::map<std::string, std::string> column_keywords, std::string target_column)
```

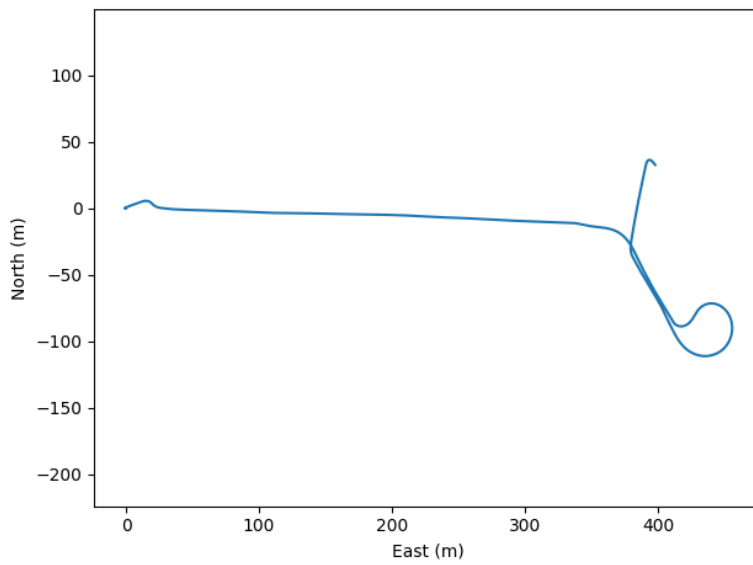
Those take the filename, a map of column keywords, and the target column name as inputs. These functions extracted specific values from the navigation file, such as the attitude of the vehicle (roll, pitch, and yaw), its geographic coordinates (latitude and longitude) or its altitude, and returned a map with timestamps as keys and the extracted values as values.

The navigation data was provided in International Scientific Units, and angles were given in degrees. Therefore, we integrated a conversion process within the extraction function, thus creating the `get_rad_angle_value` function. We also included error handling to address issues with the altitude sensor data, providing the previous value in case of a *NaN* value since the file possessed a lot of those.

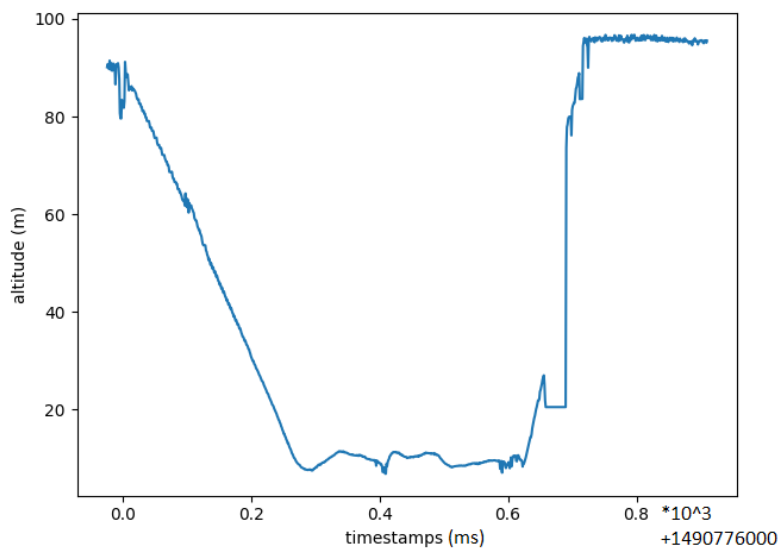
All the resulting structures are maps, holding the values extracted with respect to the timestamps so it is possible to combine with the FLS data. This step is explained further in subsection 3.2.2. A good way to verify the integrity of the data extracted from the file was to plot the navigation and the altitude. As the altitude sensor was often giving errors, we had to add a small error handling in the extraction that gives the previous value in case of a *NaN* value in the data. Those plots can be seen in the plots 3.1. In the first, we verified the integrity of the values in latitude and longitude, in the second, we verified that the conversion to the cartesian frame was giving corresponding values. In the last, we plotted the altitude, to check that the values had a meaning, even with the implemented error handling. We can see that even if there were *NaN*, they were not too many so the result is usable.



(a) Navigation in world coordinates



(b) Navigation in cartesian coordinates



(c) Altitude throughout the mission

Figure 3.1: Data integrity check plots

From the last plot, we can see that the vehicle is going downwards then upwards, giving us confirmation that part of the FLS images will have bigger floor contact and therefore possible detections of markers on the floor.

For the FLS data, we first extracted the SON files using the provided Blueview sonar reading functions. The extracted data, including polar and Cartesian sonar images, were saved in CSV and PNG formats, respectively, organised within corresponding folders named after the SON files read. This gave a complex arborescence to navigate when trying to extract all the data before combining it. Due to that the computation time at the end was quite higher.

We developed the function of the snippet 3.1 to process the CSV and PNG files and create a map structure with timestamps as keys. This facilitated the step of bringing together the FLS and navigation data.

Code listing 3.1: FLS files extraction

```
std::map<long long, std::map<std::string, std::string>>
  processFolder(const boost::filesystem::path &folderPath,
               const char &delimiter = ',')
```

Once the data became accessible from maps with timestamps as keys, we proceeded to combine them.

3.2.2 Combination

To address the absence of timestamps in the navigation file, we developed the function shown in 3.2. It converts the date and time information from the file into their corresponding timestamps in milliseconds. The expected format for the date string input is *DD/MM/YYYY*, while the time string input follows the convention *HH : MM : SS.MMMM*, both used in the DAT file.

Code listing 3.2: Timestamp conversion

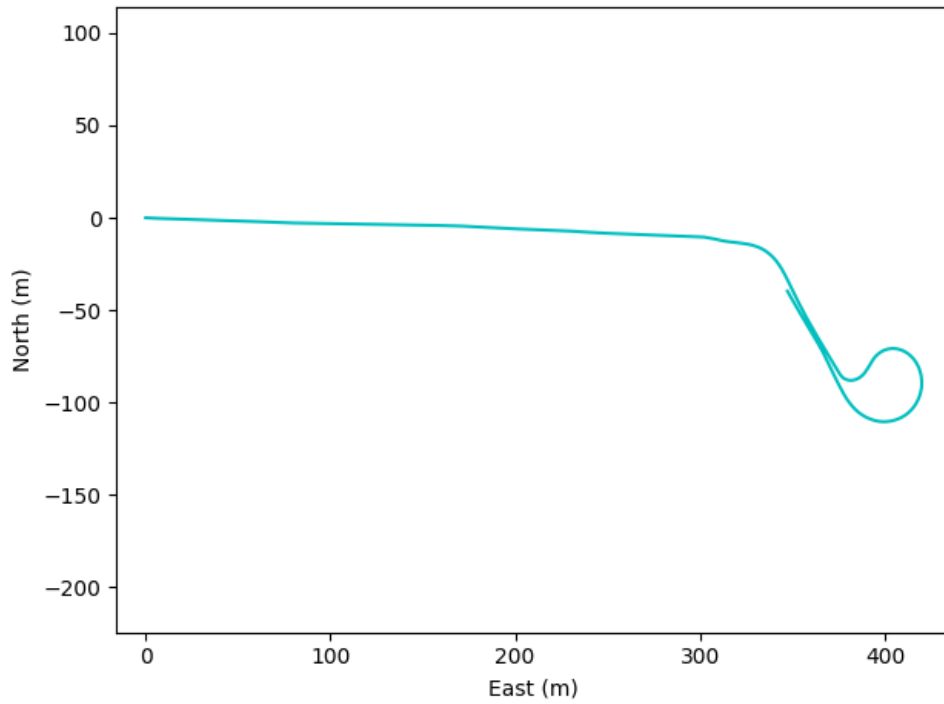
```
long long convert_datetime_to_timestamp( std::string date, std::string time)
```

We made sure the data in the FLS data map was consistent by converting all the relevant values to their appropriate data type.

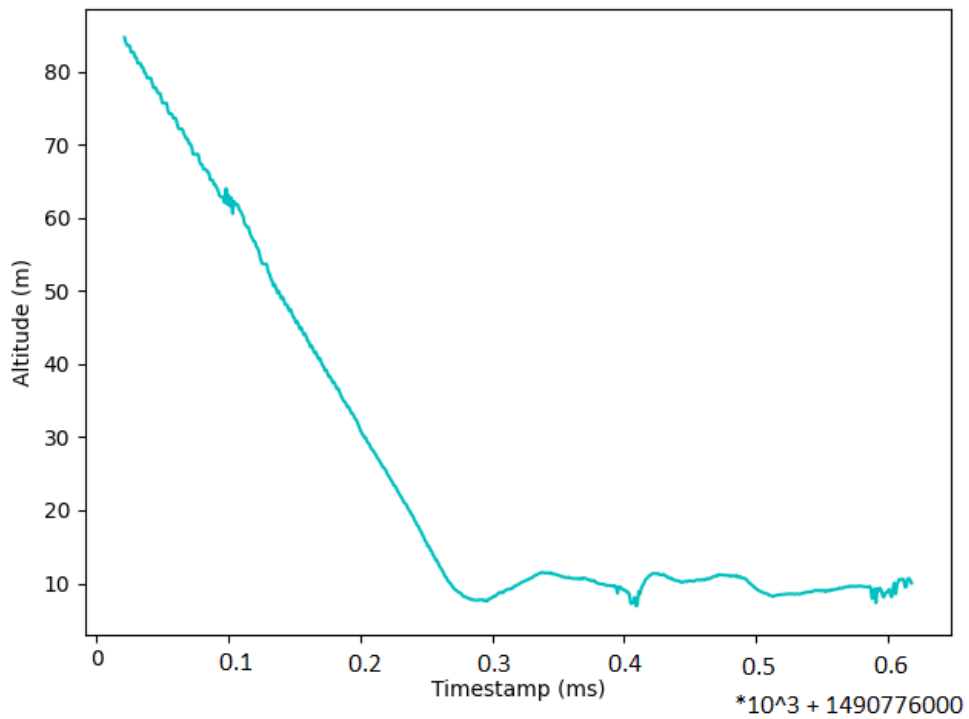
Initially, we attempted to combine the FLS and navigation data directly using the common timestamps. However, we encountered an asynchronous sampling issue, as the FLS data was sampled at a rate of 0.4 seconds, while the navigation data was sampled every 0.9 seconds. To address this challenge, we employed interpolation techniques on the navigation data.

By applying interpolation, we estimated the navigation values at the FLS timestamps, creating a synchronised data set that aligned the FLS and navigation data.

To ensure the accuracy and integrity of our interpolation function, we plotted the navigation data corresponding to the common timestamps between the FLS and navigation datasets. As the lengths of these datasets were different, we anticipated a slight variation in the plot compared to the navigation data alone. The resulting plot is depicted in figure 3.2 and shows the navigation data in Cartesian coordinates and the altitude values at the common timestamps.



(a) Navigation in cartesian coordinates for common timestamps



(b) Altitude in common timestamps

Figure 3.2: Common data check plots

As expected, the common data is slightly smaller, but this allows us to work as the data had good integrity.

3.3 FLS floor areas calculation

With the combined and synchronised data, we proceeded to the next step: calculating the areas of the seafloor observed in the FLS images to find the ones relevant to apply detection to. It is important to note that not all images have direct contact with the seafloor due to factors such as the maximum range of the FLS and the altitude of the vehicle. We wanted to focus on the images that provided valuable information for applying the detection framework later.

Calculating the observed floor areas provided us with a better understanding of the seafloor coverage of the FLS during the mission and allowed us to identify the areas where applying the detection algorithm would be relevant.

We visualised the shape of the intersection between the sonar beam and the seafloor that we should find through the mathematical calculations thanks to the geometrical representation depicted in red in figure 3.3.

This helped us for a more accurate plotting of the areas later on.

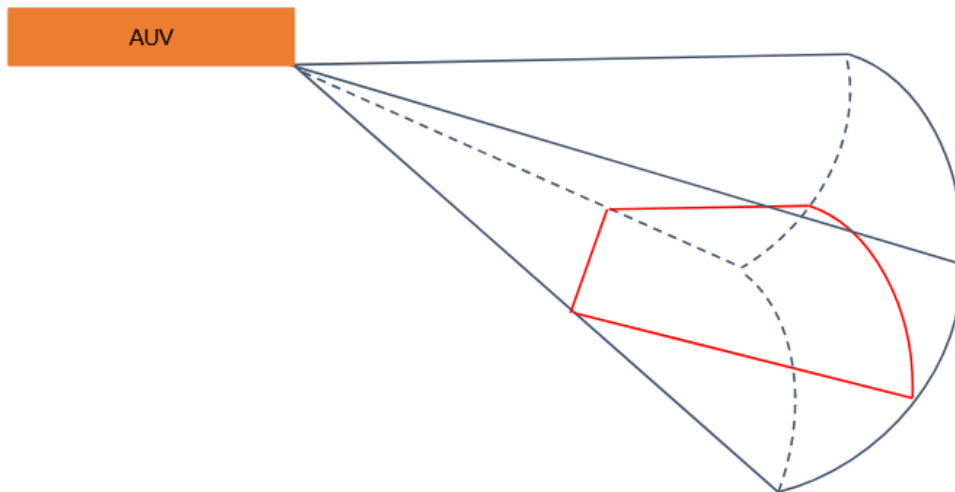


Figure 3.3: Shape of the intersection between the floor and the beam

We are considering that our vehicle is at all times parallel to the floor for the following calculations and representations.

We are using the following notation for the calculations:

- r_m is the minimal range where the beam is crossing the floor.
- r_{m_f} is the projection on the floor of r_m .
- r_M is the maximal range.
- r_{M_f} is the projection on the floor of r_M .
- r is the difference between r_{M_f} and r_{m_f} , used in the calculation of r_{M_f}

- a is the altitude of the vehicle.
- ϵ is the elevation, meaning the vertical aperture of the beam.
- o is the observation angle, corresponding to the addition of the aperture and the pitch of the FLS.

The latitude and longitude coordinates provided in the data represent locations in the *NED* (North-East-Down) world frame. However, for our calculations and to simplify the analysis, we are using a local Cartesian frame with its origin set at the starting point of the mission. All other positions are then expressed relative to this origin. Additionally, we define a vehicle frame with its origin at the nose of the vehicle.

To maintain consistency in our calculations, we translate all distances and positions measured in the vehicle frame into the Cartesian world frame. This allows us to work with a unified coordinate system throughout the analysis, making it easier to compare and combine data.

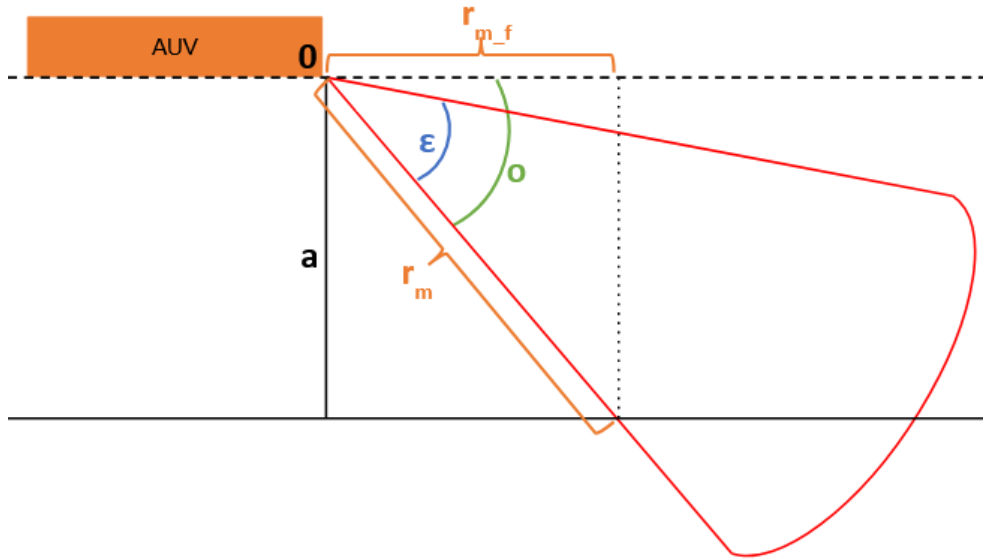


Figure 3.4: Side view of FLS beam number 1

Let us start by defining the formula to calculate r_{m_f} . From figure 3.4, we can use trigonometry to say:

$$r_m = \frac{a}{\sin o}$$

$$r_{m_f} = r_m \cos o$$

Now, we can determine the calculation of the length of r_{M_f} .

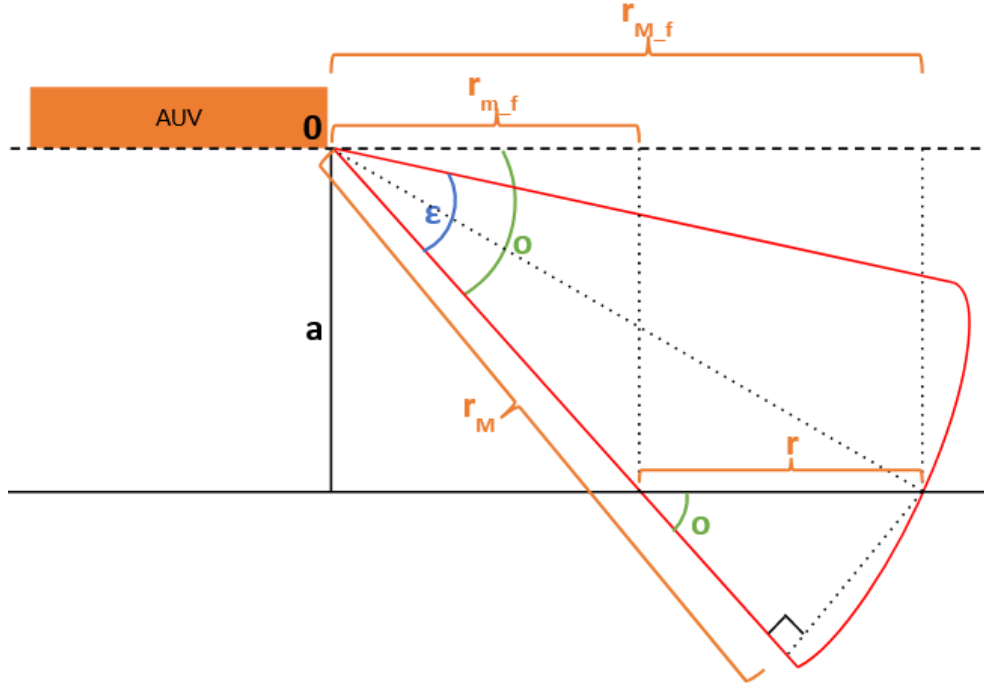


Figure 3.5: Side view of FLS beam number 2

Using figure 3.5 as a reference, we can use the property of the corresponding angles since we consider the vehicle on a parallel line to the floor. By approximating the curvature of the beam as a straight line forming a 90 angle, we can approximate the distance r using trigonometry. Let us also consider that $d = r_M - r_m$.

$$r = \frac{d}{\cos o}$$

From there, we can tell that :

$$r_{M_f} = r_{m_f} + r$$

Now that we have the formulas to get the distances from the FLS in the vehicle frame, we need to project them into the Cartesian world frame. For this, we use the following matrix :

$$T = \begin{bmatrix} \cos \theta & -\sin \theta & x \\ \sin \theta & \cos \theta & y \\ 0 & 0 & 1 \end{bmatrix}$$

To this matrix, we multiply the vector in the vehicle frame that we want to translate into the local world frame. The figure 3.6 will help to understand the reasoning.

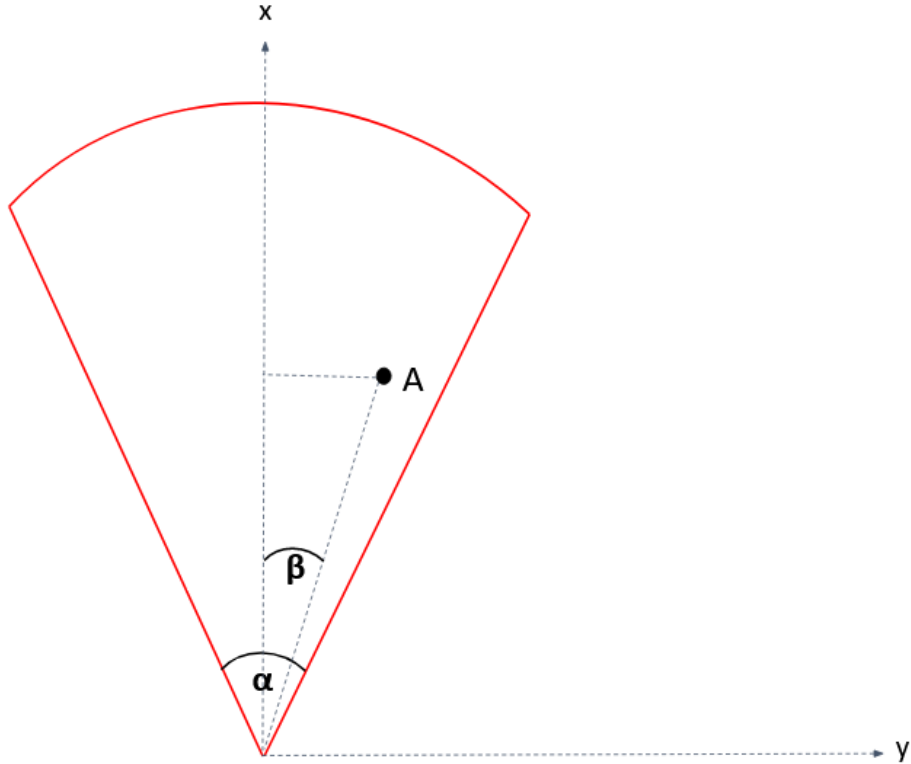


Figure 3.6: Top view of FLS beam in vehicle frame

The point **A** is defined by the following vector:

$$\mathbf{A} = \begin{bmatrix} p \cos \beta \\ p \sin \beta \\ 1 \end{bmatrix}$$

In our calculation, we are considering $\beta = ct \times \alpha$ with $ct \in \mathbb{Z}$. We are using this to calculate the different points to create the polygon corresponding to the floor area.

Since the minimum range is a straight line due to geometry, we only had to apply $ct = 0.5$ and $ct = -0.5$. However, for better representation, we chose to increment the constant value between -0.5 and 0.5 to create the circular arc for the floor upper boundary. In our code, we are separating the different points for the polygon by their x axis value. Therefore, with $p' = p \cos \theta$, our vector looks like :

$$\mathbf{A}' = \begin{bmatrix} p' \\ p' \tan \beta \\ 1 \end{bmatrix}$$

The plot in figure 3.7 depicts one every nine other geometric floor tiles, also referred to as polygons. There is an error handling verifying that a polygon is

not empty before trying to plot it. A geometric floor tile will only be created if $r_{m_f} < r_{M_f}$ and $r_{m_f} < 0.0$. This prevents aberrant values.

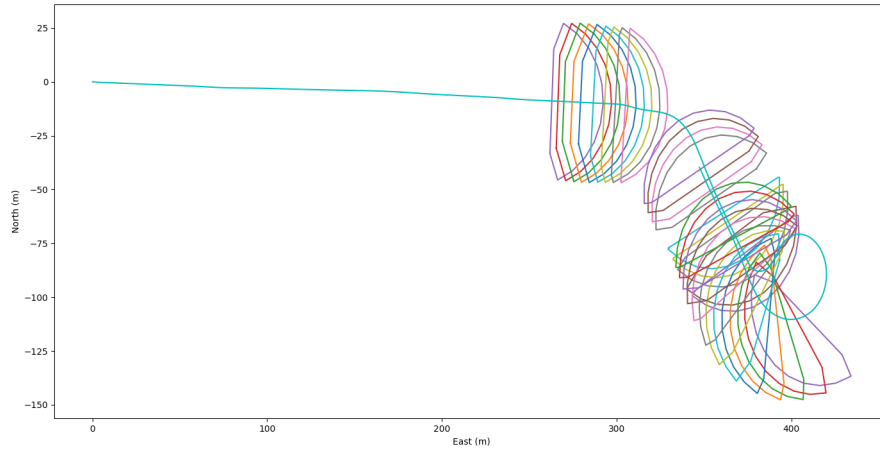


Figure 3.7: One every nine FLS floor tile

3.4 Partial Conclusion

In summary, our approach successfully extracted, synchronised, and combined various crucial data parameters, including attitude, latitude, longitude, altitude, position in the Cartesian world frame, or FLS range, aperture, gain, and frequency. Our standardised format of data is a map with the data and the timestamp in key. This allowed us to successfully combine desynchronised data from the navigation and the FLS.

It is important to note that having a common navigation on all the data will help for a more accurate detection description, whatever the method chosen.

This process provided valuable information on the floor coverage area of the FLS throughout the mission. Moreover, the functions developed during this phase can be utilised for future similar data processing tasks. In the next chapter, we will tackle the detection problem and discuss the different approaches undertaken to address it.

Chapter 4

FLS and SAS Association

This phase is divided into two parts: the detection of markers in images, and their association.

The SAS images already have their own attached location in KML files.

We are looking to associate the FLS images and the SAS images, therefore we are looking for common markers between the data.

4.1 Detection

<p>Problematic: What markers can we extract from detection, in order to give it to the association part?</p>

The detection phase focuses on identifying and highlighting potential alarms or objects of interest in the images extracted from the data. In our initial trials, we employed feature detectors as in the paper by Tueller *et al.* [3]. This approach was inspired by the successful application of feature detectors in waterborne environments.

Indeed, we wanted to know if the feature detectors were robust enough to match detections from a SAS and to an FLS. We first tried them on same type data pairs (SAS to SAS and FLS to FLS). The idea was that if it is robust enough between the same data it might be applicable between different datas.

We knew that Exail had already working detectors based on object morphology, which give the location of the detected object, leading to a more classical matching using location and incertitudes.

It is important to note that this chapter presents the initial work conducted in the thesis, where we did not have access to the data used in the data processing phase (3) which arrived later on. The focus of this chapter was to explore and evaluate different detection methods, including feature detectors, to identify potential alarms or objects within the available images.

We first try a traditional approach, the feature detectors, then the morphology-based approach.

4.1.1 Features Detection

Prior to the thesis project, during the first semester, preliminary work was conducted in Python, focusing on image data analysis. Although distinct from the thesis itself, this pre-thesis work served as a foundation for understanding image processing techniques and methodologies. Although it was carried out on image data rather than sonar data, it provided some insights in working with visual data and extracting meaningful information. The work with Python-based image processing libraries and techniques proved useful to the transition to the C++ based ones.

In the initial stage of our research, we conducted experiments using various feature detectors, such as SURF, ORB, SIFT, and AKAZE, to compare their performance and determine the most suitable detectors for our application. Unfortunately, we encountered difficulties with the SIFT detector [6], which did not compile properly and therefore could not be utilised in our analysis. However, the remaining detectors were implemented and evaluated on the initial images data.

The images for the initial work comprised FLS data of a mission and SAS data of the Ferrando shipwreck.

ORB

The ORB (Oriented FAST and Rotated BRIEF) detector [7] was one of the feature detectors we applied to the images. ORB is a fast and efficient algorithm in airborne environments. It combines the FAST corner detector [8] and the BRIEF descriptor [9]. It is known for its robustness to scale and rotation changes.

In our implementation, we applied the ORB algorithm to two successive FLS images. We then attempted to warp the second image back to the position of the first image using the homography matrix between them. Figure 4.1 shows the results of applying ORB to the FLS images, with the matching between successive images shown in subfigure 4.1a and the warped image displayed in subfigure 4.1b. The performance results are provided in Table 4.1.

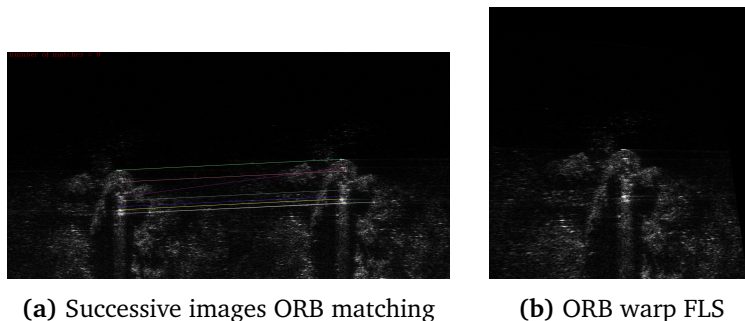


Figure 4.1: ORB between successive FLS frames

Despite the availability of enough homography points to perform image warping, the results obtained from the ORB algorithm on the FLS data indicate some

Keypoints in image 1	500
Keypoints in image 2	500
Number of matches	34
Inliers	0
Inliers ratio	0

Table 4.1: ORB Matching results performance on FLS

limitations. In the table 4.1, which provides an overview of the ORB algorithm's performance, we observe that out of the 500 keypoints in each image, only 34 keypoints were successfully matched. The matched points are the keypoints that the algorithm determined as the same from their place in image and feature data. The table also presents the number of inliers, which refers to the matched keypoints that accurately align with their corresponding keypoints locations after the warping process.

Despite achieving keypoint matches, the warping fails to correctly position the keypoints, resulting in a lack of inliers.

These results suggest that we will have troubles using the ORB algorithm in later tasks like tracking in the FLS images.

We also applied the ORB algorithm to the SAS images, using two tiles capturing the same area of the Ferrando wreck.

Initially, we tested the algorithm on images with the same insonification (illumination conditions) then on images with opposite insonification. Similarly to the FLS images method, we attempted to warp the second image back to the position of the first image using the homography.

Figure 4.2 illustrates the results of the ORB algorithm applied to SAS tiles with the same insonification, including the matching in figure 4.2a and the warped image in Figure 4.2b. The performance results are provided in Table 4.2.

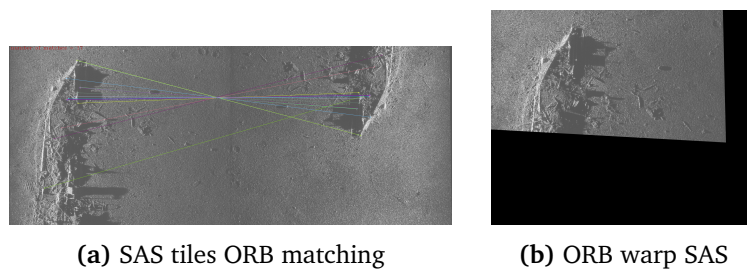


Figure 4.2: ORB between SAS tiles with **same** illuminations

From the results presented in table 4.2, it is evident that ORB performs better on the SAS images compared to the FLS images. Indeed, 28 out of the 40 matched points were placed to their matched keypoints location after warping.

This could be attributed to the presence of more detailed information in the SAS images, providing better features, meaning a higher chance of successful match-

Keypoints in image 1	500
Keypoints in image 2	500
Number of matches	40
Inliers	28
Inliers ratio	0.7

Table 4.2: ORB Matching results performance on **same** insonification SAS

ing.

Despite the good results on the same insonification images when applying ORB to the SAS images with opposite insonification, the algorithm was unable to find any matches. This indicates that the different illumination conditions pose challenges for the feature detection and matching process of the ORB algorithm.

We continued our investigation with the AKAZE algorithm.

AKAZE

The AKAZE (Accelerated-KAZE) detector is another feature detector we employed in our analysis. AKAZE is an extension of the KAZE algorithm [10] that enhances its speed while maintaining its robustness to scale and affine transformations. It is well-suited for images with significant viewpoint changes.

Following the same procedure as the ORB algorithm test, we applied the AKAZE algorithm to two successive FLS images and attempted to warp the second image back to the position of the first image using their homography matrix.

Figure 4.3 illustrates the results of applying AKAZE to the Forward Looking Sonar (FLS) images, with the matching between successive images shown in Figure 4.3a and the warped image displayed in Figure 4.3b. The performance results are provided in Table 4.3.

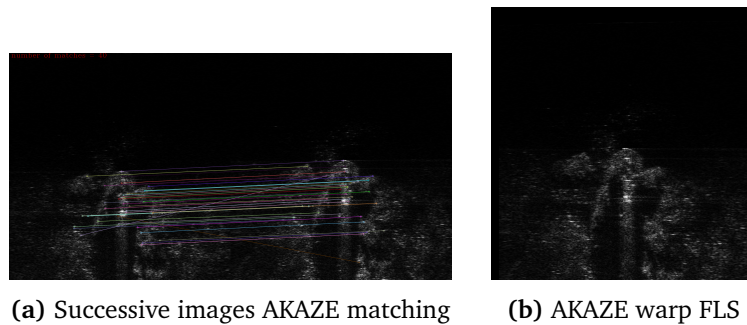


Figure 4.3: AKAZE between successive FLS frames

The results shown in the table 4.3 have a striking similarity to the ones in table 4.1, notably the lack of inliers in the warped back image. We believe this result might be for the same reason as for the ORB algorithm. Therefore, the same conclusion is drawn: the AKAZE algorithm will be hard to use in the later stages

Keypoints in image 1	565
Keypoints in image 2	664
Number of matches	69
Inliers	0
Inliers ratio	0

Table 4.3: AKAZE Matching results performance on FLS

for tracking for example.

Similarly, we applied the AKAZE algorithm to the SAS images. Figure 4.4 shows cases the results of AKAZE applied to SAS tiles with the same insonification, including the matching between successive images in Figure 4.4a and the warped image in Figure 4.4b. The performance results are provided in Table 4.4.

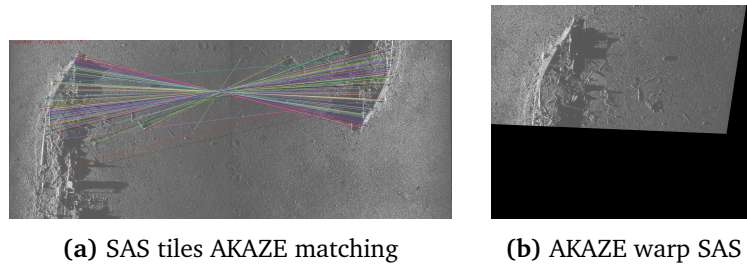


Figure 4.4: AKAZE between SAS tiles with **same** insonification

Keypoints in image 1	2774
Keypoints in image 2	1939
Number of matches	242
Inliers	163
Inliers ratio	0.673554

Table 4.4: AKAZE Matching results performance on **same** insonification SAS

Table 4.4 shows that the AKAZE algorithm finds four times more keypoints in the images than the ORB algorithm. However, when it comes to the inliers ratio, there is a 67% correspondence of the warped image keypoints, against a 70% one for the ORB algorithm (table 4.2). The real difference is that the number of matches and inliers is bigger for AKAZE, making it perform better since it achieves almost the same results but with a higher number of points.

Again, in spite of the good results on the same insonification images, the algorithm was unable to find any matches on the opposite insonification SAS images. This indicates that AKAZE is also challenged by the change of illumination in images.

SURF

The SURF (Speeded-Up Robust Features) detector [11] was also included in our evaluation. SURF is known for its robustness to scale changes and affine transformations, making it suitable for a variety of applications. It utilises the integral image to compute the image features efficiently, offering robustness to image noise and geometric transformation.

We applied SURF on successive FLS images using the same methodology as the previous detectors. Figure 4.5 showcases the results of applying SURF to the FLS images. The matching between successive images is presented in Figure 4.5a, while the warped image is displayed in figure 4.5b and the warping in figure 4.5b. The performance results are provided in table 4.5.

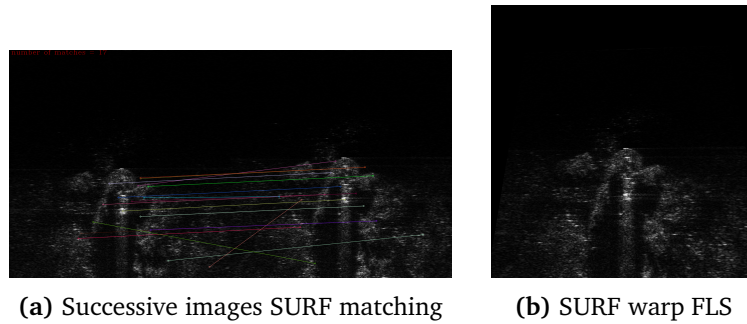


Figure 4.5: SURF between successive FLS frames

Keypoints in image 1	1126
Keypoints in image 2	1272
Number of matches	87
Inliers	1
Inliers ratio	0.0115

Table 4.5: SURF Matching results performance on FLS

The performance analysis reveals that the SURF algorithm demonstrates higher matching capabilities compared to ORB and AKAZE.

In contrast to the other detectors, SURF gets one inlier point, whereas the other detectors had none. This suggests that the SURF algorithm might be better suited for FLS detection and tracking applications.

Additionally, we applied the SURF algorithm to the SAS images. Figure 4.6 presents the results of SURF applied to SAS tiles with the same insonification, including the matching in figure 4.6a and the warped image in figure 4.6b. The performance results of the SURF algorithm on the SAS images are provided in Table 4.6.

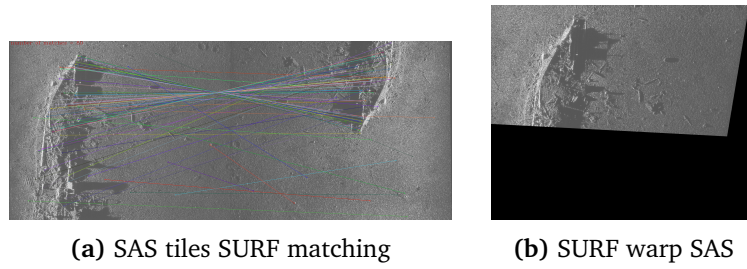


Figure 4.6: SURF between SAS tiles with **same** insonification

Keypoints in image 1	6271
Keypoints in image 2	6796
Number of matches	327
Inliers	75
Inliers ratio	0.2294

Table 4.6: SURF Matching results performance on **same** insonification SAS

Contrary to the ORB and AKAZE algorithms, the results obtained from the SURF algorithm on the same insonification tiles exhibit lower performance, with an inlier ratio of only 22%. Despite this lower ratio, the algorithm still produces a considerable number of matches, nearly three times the number obtained by the other detectors. This larger number of matches provides a substantial amount of reliable data points for further analysis. While Figure 4.6a displays inaccurate matches, these outliers do not appear to significantly impact the overall result. Notably, the SURF algorithm was the only one capable of handling opposite insonification SAS images. It is important to remember that the images used are from the same wreck but seen from two different point of view. Figure 4.7a demonstrates the matching process, while Figure 4.7b presents the resulting warped image. The performance metrics for this particular application of the SURF algorithm are provided in Table 4.7.

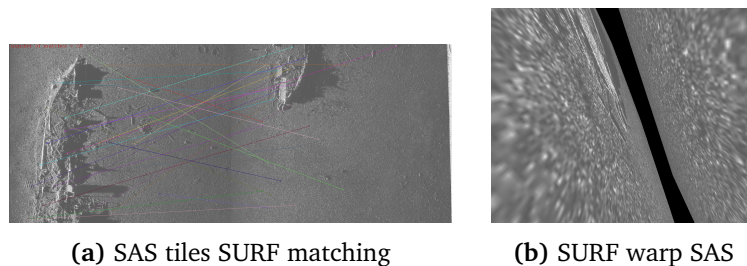


Figure 4.7: SURF between SAS tiles with **opposite** insonification

Despite the relatively high number of matches indicated by the performance metrics, figure 4.7a reveals a significant number of outliers compared to the same

Keypoints in image 1	6271
Keypoints in image 2	2202
Number of matches	241
Inliers	2
Inliers ratio	0.0083

Table 4.7: SURF Matching results performance on **opposite** insonification SAS

insonification matches. These outliers contribute to inaccuracies in the warped image, making it unreliable for further analysis. More caution should be exercised when applying the SURF algorithm to SAS images with opposite insonification, for example additional care when matching keypoints.

Robustness tests

Gaussian Pyramid

To address the difference in resolution between the SAS objects and the close-up of the FLS images, we applied a Gaussian Pyramid scheme. The objective was to replicate the resolution difference between the two types of images.

As the initial data did not include a common object observed in both FLS and SAS, we reduced the size of the FLS images and padded it with noise to keep the same size of image before applying the detection algorithms.

Figure 4.8 illustrates the matching results on the left-hand side and the corresponding warped back small image on the right-hand side. It is evident that as the image size decreases, there is a noticeable loss of resolution. The ORB and AKAZE algorithms were not able to pass this test, therefore we are showing the SURF algorithm results. So far, the SURF algorithm was the most consistent in detections. In our attempts to further reduce the image size by a factor of 16, we encountered challenges with the SURF algorithm. Although we were able to obtain a reduced image and perform the matching process, the presence of numerous outliers prevented the calculation of a valid homography between the initial and reduced image.

This limitation highlights the boundaries of the SURF algorithm in handling significant scale variations. It is important to note that this method of robustness verification assumes an object viewed from the same perspective but at a different scale.

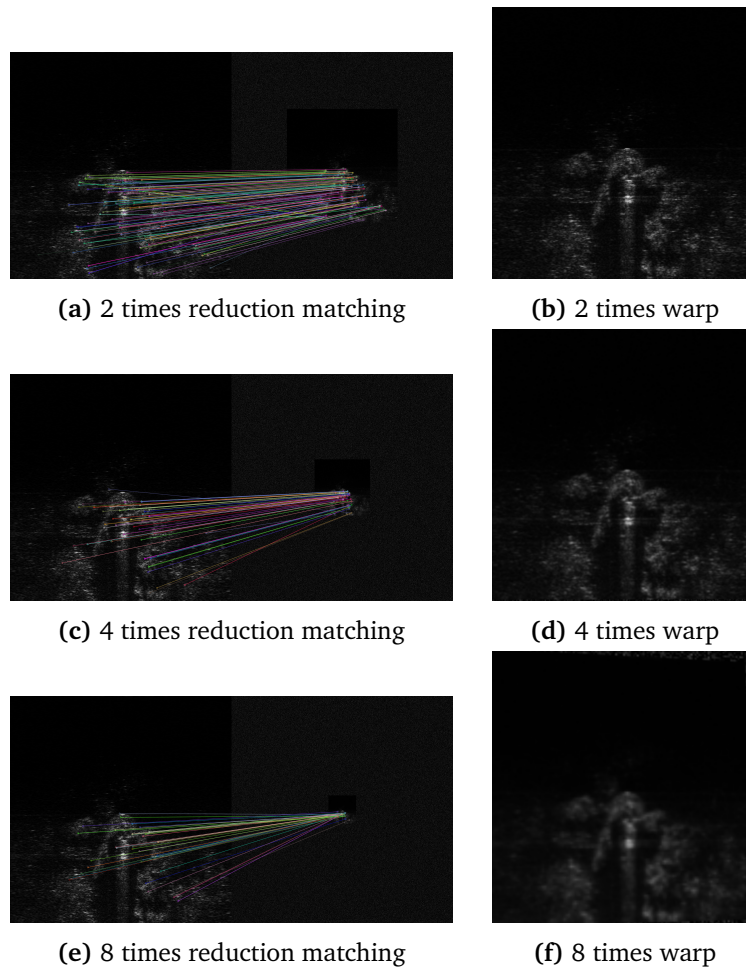


Figure 4.8: Gaussian Pyramid images matching and warping

Opposite insonifications

As mentioned in the section 4.1.1, we conducted an experiment to assess the performance of the feature detectors on SAS images with opposite insonification. We selected two pieces of SAS tiles that contained common scattered objects and had only a small translation between them.

These objects, highlighted by hand in figure 4.9, were expected to be matched by the feature detectors. However, none of them were able to produce any matches in this scenario.

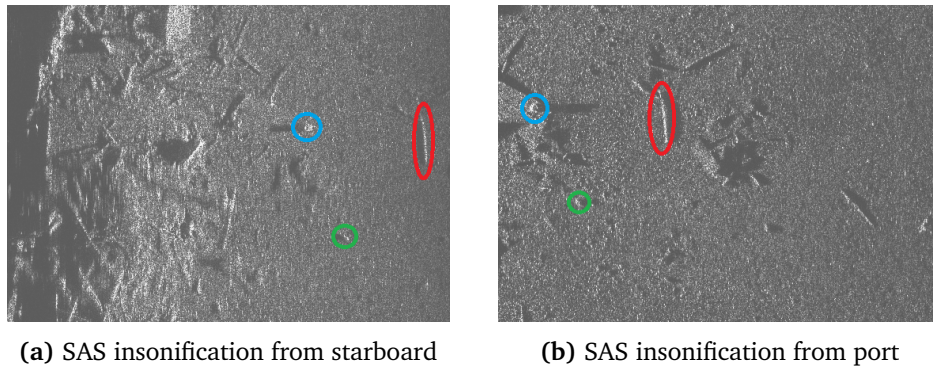


Figure 4.9: Tiles with common clutter

This experiment provided valuable insights and confirmed that the feature detection methods as described by Tueller *et al.* [3] (but without the sorting of background), were not suitable for our thesis application. It became evident that we needed to integrate the different detection frameworks developed by Exail, as they were proven to work on the type of data provided.

4.1.2 Physics based Detection

During this phase, we initially applied the detection framework provided by Exail to the same images used for the feature detectors.

As we obtained our final data, we included this detector and tracking algorithm into our reasoning.

In the early stages, we attempted to implement an optimisation algorithm to fine-tune the object search parameters. Due to the large number of parameters involved, we experimented with the Particle Swarm Optimisation (PSO) algorithm. The PSO algorithm showed promise by providing the best parameters for a single run. To account for potential variations, we ran the algorithm multiple times. However, due to time constraints, we made the decision to temporarily set aside this project and refocus on the primary objectives of the thesis. This content is provided in the appendix A.

FLS images detector

The detector used on the FLS images is based on the publication by Galceran *et al.* [12]. The use of this method allows for a fast detection and tracking in both polar and cartesian images. It is used usually to detect obstacles, but will work as well to detect objects even if they are not obstacles but on the seabed.

Initial tests

During the initial phase of the thesis, we attempted to apply the FLS detector framework to the SAS images shown in figure 4.9. The FLS detector successfully

detected objects in the SAS image, as demonstrated in figure 4.10. However, it primarily identifies areas with high intensity contrast, making it less reliable for images with opposite insonification.

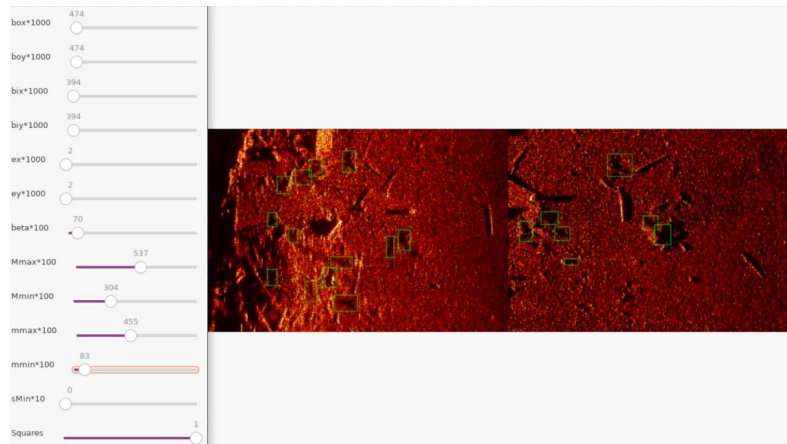


Figure 4.10: Example of FLS detector application on SAS tile

This experience led us to the decision of using a tailored detector specifically designed for the SAS data. While we abandoned the idea of using the same detector for both types of data, we integrated the FLS detector to process the FLS data in our final dataset.

The objective of this approach was to use the strengths of both detectors and optimise the object detection and tracking process for each respective sonar system.

Integration to the framework

In the penultimate step of our work, we applied the detector to the extracted data, which combined the FLS data from all missions with the corresponding navigation information. By using this common structure navigation, we successfully localised various objects of interest along the FLS mission path.

Notably, part of the Dornier plane wreck, the key point of interest in the mission, was highlighted with multiple detections around it. Since the path passed over the wreck twice, it was interesting to accurately identify and track it.

Figure 4.11 illustrates two different views of the Dornier plane wreck detected during the mission. The boxes represent the contacts found by the detector, with each number indicating a unique index for a detection run. The tracking algorithm effectively retrieved previously detected contacts, resulting in a higher number of detections upon returning to the site due to accumulation of contacts.

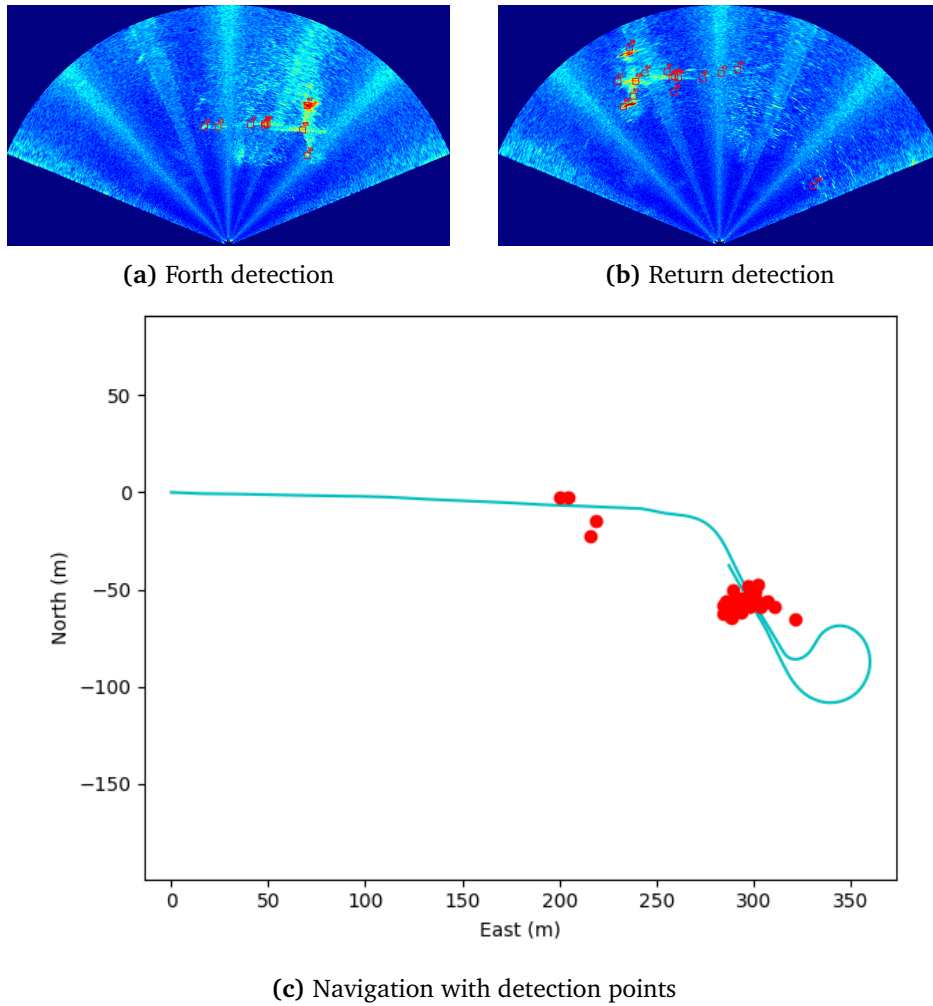


Figure 4.11: Detections with $\beta = 110$

The detections presented in figure 4.11 have a range of 0.5 to 1.5 meters, with an intensity threshold of $\beta = 1.10$, resulting in many detections. Specifically, the constellation of detection is dense around the Dornier plane wreck. However, due to the low intensity of the beam, giving a low resolution image, it was almost certain that these detections were mostly noise.

We then tried to focus on capturing the overall shape of the Dornier plane, or at least its major components, so we raised the intensity threshold.

The figure 4.12 display the detections for a $\beta = 1.20$. It is accompanied by a figure showing their location in the global plot.

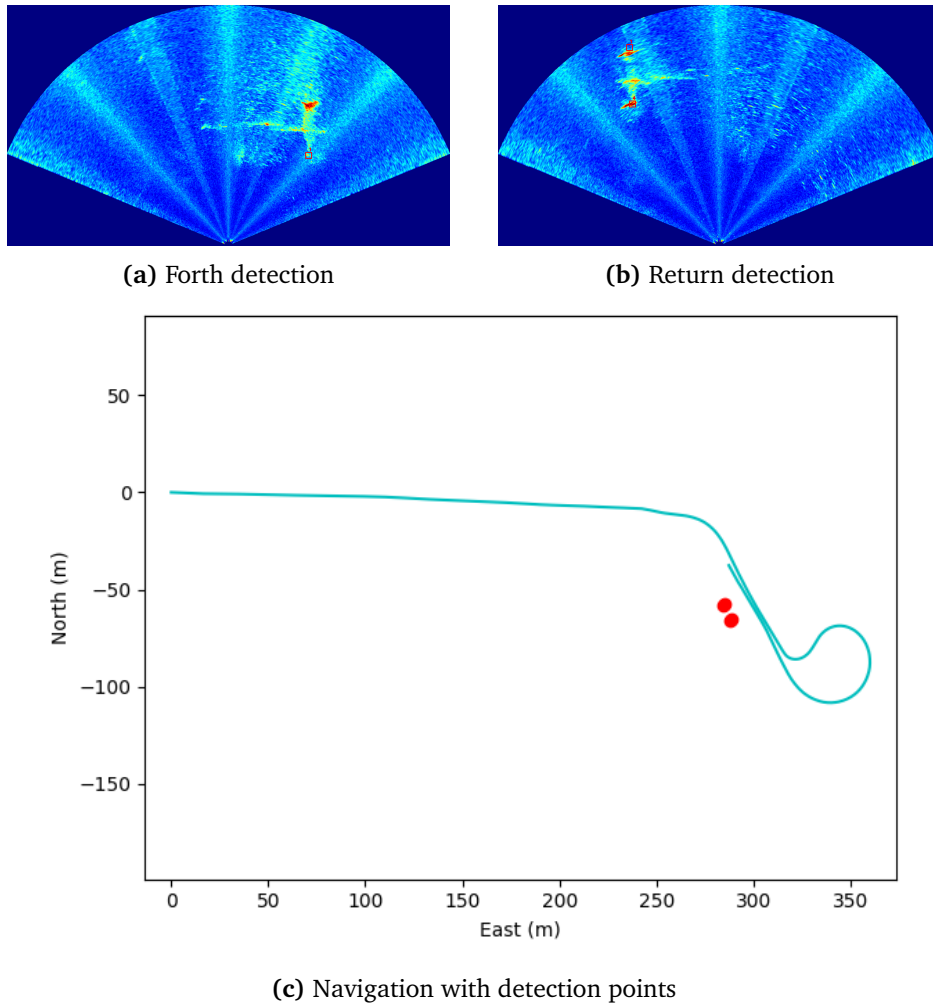


Figure 4.12: Detections with $\beta = 120$

The result of raising the intensity threshold value is clearly seen when comparing the results in figures 4.11 and 4.12. In conclusion to this, we can confirm that we managed to divert the initial use of the detector to fit our purpose: detecting objects from the floor scene and not front obstacles.

However, it is essential to note that despite making contact with the seafloor, the FLS images do not provide high-resolution results due to their limited intensity. Figures 4.11a, 4.11b, 4.12a and 4.12b only show a portion of the aircraft tail, which is elevated compared to the seafloor, making it better reflect the sonar beams. In contrast, the SAS tiles provide much more detailed information. In this situation, we cannot expect to detect the objects of the seafloor from the FLS data, therefore making it impossible to go forward with association. Given better resolution images, we could proceed to test our pair of sonar relocation theory.

In parallel, to complement the application of the FLS detector on the FLS images, we also applied the SAS detector framework to the SAS tiles to show the objects

of interest we should be comparing.

SAS images detector

The SAS detector is based on the idea presented by He *et al.* in their publication [13]. The algorithm uses Mask R-CNN (Region-based Convolutional Neural Networks), an advanced deep learning technique used for segmentation tasks. It is an extension of the Faster R-CNN object detection algorithm, which identifies objects in an image and draws bounding boxes around them. Mask R-CNN also accurately segments each object at the pixel level, generating precise masks that outline their boundaries. This method comprises two components: a region proposal network and a mask prediction branch. The region proposal network suggests potential regions for the object locations in the image, and the mask prediction branch refines them to pixel-level masks for each object.

The SAS detector uses this method to detect the characteristic echo-shadow pair of a target on the seafloor. To train the algorithm, they used a database of around 5000 samples with the echo-shadow pairs segmented by hand.

Thanks to this technique, they manage to get a precise understanding of the detected objects. Two example tiles are presented in figure 4.13 and figure 4.14.

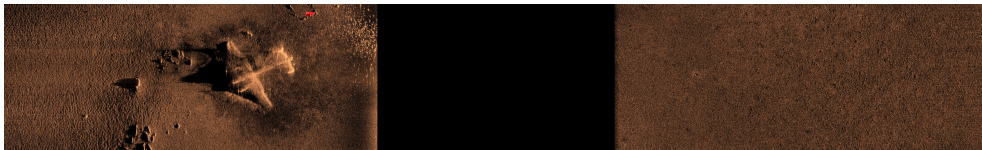


Figure 4.13: First tile example

It is to be noted that the black area in the middle of the image is below the vehicle, where the SAS cannot detect. One SAS image will have a starboard side and a port side, separated by a non detected dark area. It is the reason why a lot of surveys are done in a lawnmower path.

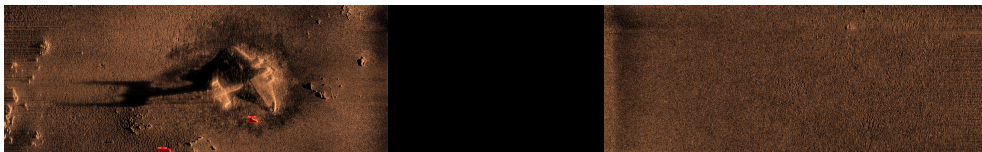


Figure 4.14: Second tile example

The differences between the results from the SAS and FLS images detections can be attributed to several factors.

First, the SAS images contain more details compared to the FLS images, which provide more potential targets for detection.

Second, the images of the FLS had a bad resolution, making it impossible to detect floor level objects with the FLS detector. The detector has the capacity to find the

markers if given good resolution data.

An other option we could have tried would be to parameterise the SAS detector to look for the complete tail echo-shadow couple. However, this could lead to the challenge of associating this one detection to multiple FLS detections, requiring further processing and data fusion techniques to establish a coherent and comprehensive understanding of the target object from both SAS and FLS.

4.1.3 Partial Conclusion

In this phase of the research, we explored and tested various feature detectors for object matching in SAS and FLS images, in the objective to match the two modalities. However, the results demonstrated the challenges in achieving robust and consistent detections. As a result, we used the detection frameworks provided by the company, which allowed us to extract objects of interest from the data.

We proved that it was possible to detect and project into the world frame the objects from the FLS images. We also determined that we needed to obtain specifically floor-oriented FLS images with a better resolution.

The fact that our data was the bottleneck of our research does not allow us to validate or invalidate our association method hypothesis. With enough detection on the FLS data, it would be possible to apply the next step, the Association, and validate our matching method.

In the next chapter, we are discussing the Association problematic, presenting our temporary solution to still visualise FLS and SAS data superimposed and a potential research plan for the detection association.

4.2 Association

Problematic: What method can we use to associate the markers provided by the detection step?

In this chapter, we explored the concept of association in the context of underwater object detection, localisation and relocation. Association is crucial for understanding the environment and determining if an object has been previously detected by a different sensor (or by the same sensor, but this is outside the scope of this thesis). We initially thought about achieving association through feature-based methods but encountered limitations with the currently existing tools as presented in the Detection phase. The second approach involved using the location of detected objects, but the disparities observed between the SAS and FLS detections did not allow us to proceed with our tests.

4.2.1 KML images superposition

As a solution, we utilised Google Earth and its KML files handling to geographically place the images. While the SAS images already had their KML files, we needed to create KML files for the FLS images. To achieve this, we developed a function that extracted the necessary data from the common structure established during the Data Processing phase. The function's formulation is presented in snippet 4.1. This allowed us to visualise and integrate the FLS and SAS images into the *NED* frame, providing a better understanding of their spatial relationships.

Code listing 4.1: KML data

```
static void calculate_kml_values
( const std::pair<double,double> & first_position_vehicle_world,
  const std::pair<double,double> & position_vehicle_cart,
  const double & ap_rad, const double & r_M_floor,
  double & latN, double & latS, double & lonE, double & lonO )
```

After extracting the latitudes and longitudes from the FLS images using the previously developed function, we populated its KML file with this geospatial data. By using Google Earth to incorporate the KML file into the *NED* frame, we were able to visualise the results, as shown in Figure 4.15. This approach enabled us to superimpose the images geographically, providing insights into the spatial relationships between the FLS and SAS data.

Upon examining the superimposed images, we noticed a consistent offset between the SAS and FLS data. Despite this offset, the images appear to align well with each other. Additionally, we observe a small shift between the two FLS images, likely caused by a slight drift in the vehicle's navigation data, used as the FLS navigation.

Although this superposition technique may not directly perform object association, it could serve as a useful verification step for cross-referencing the FLS and SAS data and validating potential associations between objects detected by both sensors.

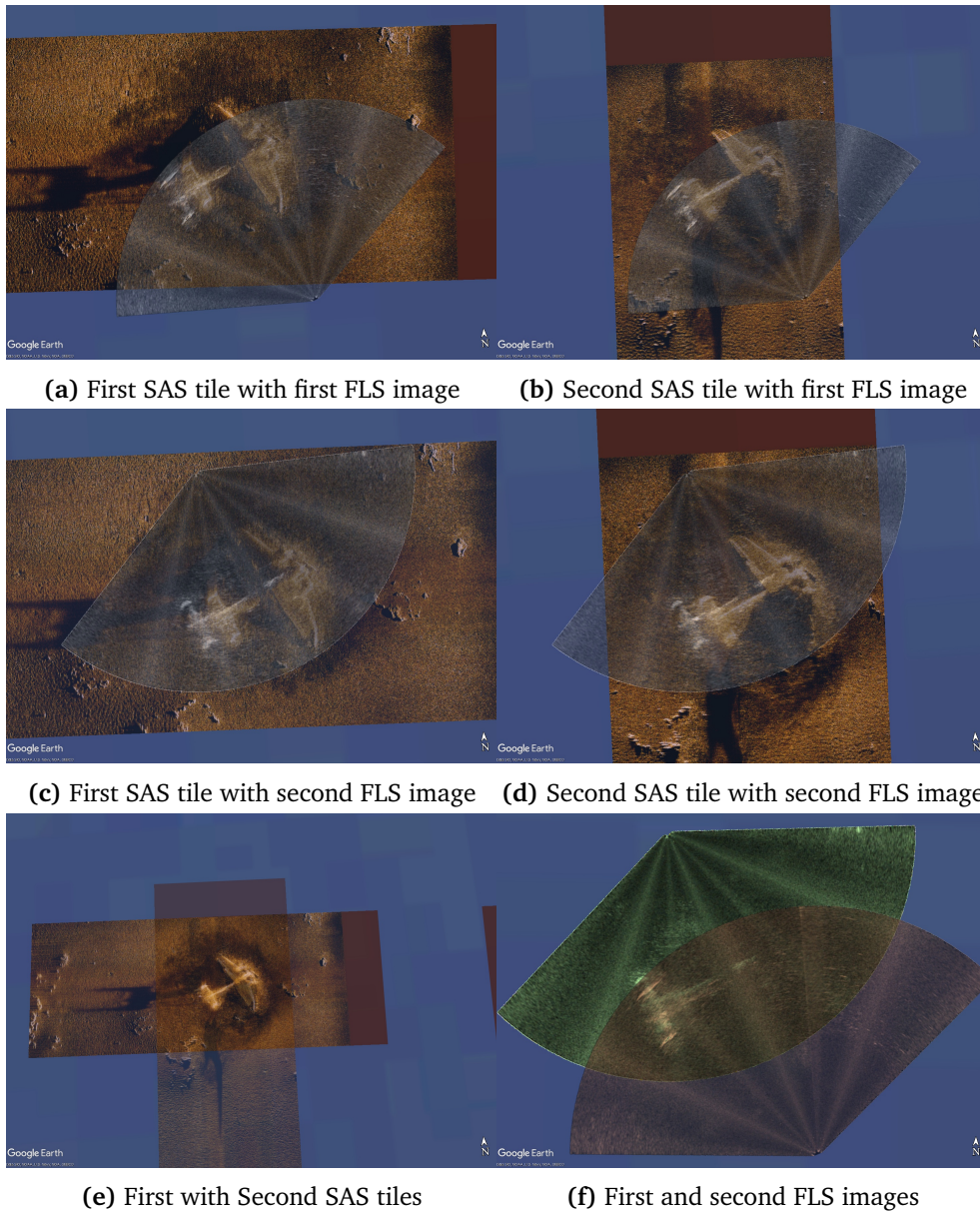


Figure 4.15: Google Earth image superposition

4.2.2 Association methods ideas

When exploring potential methods for associating the FLS and SAS data for object relocation, two main approaches emerged:

Current State-of-the-Art

One approach involved using the location information of the detected objects in both the FLS and SAS data. This method would require generating lists of detections for both sensors and then employing an assignment algorithm to find the best possible association between the two lists.

The assignment algorithm is a combinatorial optimisation problem that seeks to match detections based on their spatial proximity. By considering the locations of detections in both datasets and calculating their distances, we can establish associations between corresponding objects. To account for potential drifts and uncertainties, incorporating uncertainty values in the association process could be beneficial.

The Hungarian Algorithm, known for its guaranteed optimality and reasonable runtime complexity, could be a suitable choice for this task. After the assignment, projecting the associated detections in the *NED* frame visualising them using Google Earth would provide visual validation of the object associations.

As a relevant example, Hamuda *et al.* [14] proposed a tracking algorithm that combines a Kalman filter with the Hungarian algorithm. Although their application focuses on cauliflowers detection and tracking, it illustrates the principle of using the Hungarian algorithm for association purposes.

By adapting and applying this algorithm to our FLS and SAS markers detections, we could achieve reliable object association, enabling us to relocate objects between the two sensor modalities.

Future Solutions

Indeed, while the Hungarian Algorithm has been widely used for association tasks with localisation data, we are also interested in methods that would be able to associate data without localisation information for example. There is a real challenge in using feature detectors to associate detections from SAS and FLS with the description of the detected keypoints with a common descriptor between the two sensor modalities.

Considering this challenge, exploring Deep Learning methods, such as the SuperGlue association algorithm from Sarlin *et al.* [15], could be interesting to pursue. In particular, we could try using the SuperPoint detection and description algorithm on both SAS and FLS images. The work by Wu *et al.* [16] compares various SLAM algorithm performances and concludes that the SuperPoint detector is well-suited for underwater datasets.

On the other hand, Sarlin *et al.* [15] demonstrate the efficiency of SuperGlue in both indoor and outdoor datasets, making it an interesting choice in the case of

feature association works.

Currently, only a few recent studies have explored these deep learning methods, mostly in the context of SLAM applications. However, it is expected that the use of Deep Learning solutions will continue to emerge in the future, providing new insights and potential solutions for markers or keypoint association in multi-sensor datasets.

4.2.3 Partial Conclusion

These two proposed approaches offer potential solutions for associating the FLS and SAS data and addressing the object relocation challenge. Each approach comes with its own challenges and requirements. Further exploration and experimentation would be needed to determine their feasibility and effectiveness in achieving the desired associations.

Association remains an open problem due to the disparities between SAS and FLS detections. The outlined approaches provide potential solutions that merit further exploration in future research works. By addressing the association challenge, we can make significant strides in enhancing the capabilities of underwater robotics and improving object detection and relocation in diverse environments.

Chapter 5

Conclusion

In this work, we tried to tackle the issue of target relocation using two types of sonar data: an FLS and a SAS. The ultimate objective, as evoked in the introduction, is to assist the relocation of ROVs because of their lower quality sensors and help correct the drift of the AUVs navigation systems.

In our study, we successfully managed to create a generalised framework for reading, synchronising and combining data, with the underlying idea that we needed the navigation to be common for all data to perform the detection phase, either by using it to enhance the feature matching algorithm if it worked, or by adding a localisation to the morphological factors otherwise.

In the continuation of the research work, we showed that the feature detectors were not adapted to find common markers on FLS and SAS due to the changes of insonification and scale between the two. If the features had worked we could have used a target relocation based on a SLAM technique using the localisation.

This showed that the currently pertinent method for detection is using morphology-based detectors and represent the markers by their localisation as it is invariant.

On paper, the data provided for our thesis was interesting as it was crossing the same area multiple times. However, the images of the FLS had a low resolution due to the low intensity of the end of the FLS beam. This resulted in a less precise markers detection, meaning an impossibility to go forward with the procedure.

We can validate the procedure until the detection, but the association work is still an open door for future research. It requires good resolution FLS data, therefore it would be beneficial to open new campaigns and trials with an ROV to get better datasets. This underlines the importance of having created a generalist framework in the first step.

Bibliography

- [1] D. P. Williams and J. Groen, 'A fast physics-based, environmentally adaptive underwater object detection algorithm,' in *OCEANS 2011 IEEE-Spain*, IEEE, 2011, pp. 1–7.
- [2] N. Hurtós Vilarnau *et al.*, 'Forward-looking sonar mosaicing for underwater environments,' 2014.
- [3] P. Tueller, R. Kastner and R. Diamant, 'Target detection using features for sonar images,' *IET Radar, Sonar & Navigation*, vol. 14, no. 12, pp. 1940–1949, 2020.
- [4] B. Kim and S.-C. Yu, 'Imaging sonar based real-time underwater object detection utilizing adaboost method,' in *2017 IEEE Underwater Technology (UT)*, IEEE, 2017, pp. 1–5.
- [5] N. Palomeras, T. Furfaro, D. P. Williams, M. Carreras and S. Dugelay, 'Automatic target recognition for mine countermeasure missions using forward-looking sonar data,' *IEEE Journal of Oceanic Engineering*, vol. 47, no. 1, pp. 141–161, 2021.
- [6] D. G. Lowe, 'Object recognition from local scale-invariant features,' in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.
- [7] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, 'Orb: An efficient alternative to sift or surf,' in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.
- [8] E. Rosten, 'Fast corner detection,' <http://www.edardrosten.com/work/fast.html>, 2006.
- [9] M. Calonder, V. Lepetit, C. Strecha and P. Fua, 'Brief: Binary robust independent elementary features,' vol. 6314, Sep. 2010, pp. 778–792, ISBN: 978-3-642-15560-4. DOI: 10.1007/978-3-642-15561-1_56.
- [10] P. F. Alcantarilla, A. Bartoli and A. J. Davison, 'Kaze features,' in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*, Springer, 2012, pp. 214–227.

- [11] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, 'Speeded-up robust features (surf),' *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
- [12] E. Galceran, V. Djapic, M. Carreras and D. P. Williams, 'A real-time underwater object detection algorithm for multi-beam forward looking sonar,' *IFAC Proceedings Volumes*, vol. 45, no. 5, pp. 306–311, 2012.
- [13] K. He, G. Gkioxari, P. Dollár and R. Girshick, 'Mask r-cnn,' in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [14] E. Hamuda, B. Mc Ginley, M. Glavin and E. Jones, 'Improved image processing-based crop detection using kalman filtering and the hungarian algorithm,' *Computers and electronics in agriculture*, vol. 148, pp. 37–44, 2018.
- [15] P.-E. Sarlin, D. DeTone, T. Malisiewicz and A. Rabinovich, 'Superglue: Learning feature matching with graph neural networks,' in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.
- [16] D. Wu, Y. Wang, D. Hou, L. Hao, L. Zhang and D. Wang, 'Performance analysis of feature extraction methods towards underwater vslam,' in *2022 IEEE 17th International Conference on Control & Automation (ICCA)*, IEEE, 2022, pp. 605–610.

Appendix A

Particle Swarm Optimisation

We conducted the optimization part of the detection by determining the optimal parameters for our detector based on the work of Galceran *et al.* [12]. Although the results provided us with the best parameters for a single run of the optimization, they did not yield the overall optimal parameters. To approach the optimal values more effectively, we should have run the algorithm multiple times and selected the best parameters from all the runs. However, due to time constraints, we had to refocus on the main objective of the thesis. The code presented in snippet A.1 illustrates the principles of a particle swarm optimization algorithm, example that we modified to apply to Exail's detector algorithm. Figure A.1 demonstrates the type of outputs produced by the optimization, which involved checking the size of the bounding boxes, the echo size, and the intensity threshold.

```
Optimal parameters:
Parameter 0: 27.2795
Parameter 1: 83.2891
Parameter 2: 0.95004
Parameter 3: 30.6927
Optimal fitness: 1
[ OK ] PSO_TEST.optim_config_params (431143 ms)
[-----] 1 test from PSO_TEST (431143 ms total)

[-----] Global test environment tear-down
[=====] 1 test from 1 test case ran. (431143 ms total)
[ PASSED ] 1 test.
```

(a) Parameters set 1

```
Optimal parameters:
Parameter 0: 212.461
Parameter 1: 217.903
Parameter 2: 1.17797
Parameter 3: 198.365
Optimal fitness: 1
[ OK ] PSO_TEST.optim_config_params (435037 ms)
[-----] 1 test from PSO_TEST (435037 ms total)

[-----] Global test environment tear-down
[=====] 1 test from 1 test case ran. (435037 ms total)
[ PASSED ] 1 test.
```

(b) Parameters set 2

Figure A.1: Optimisation runs output examples

Code listing A.1: Particle Swarm Optimisation

```
// Define the structure for a particle
struct Particle
{
    vector<double> position;
    vector<double> velocity;
    vector<double> best_position;
    double best_value;
```

```

Particle(int n)
{
    position = vector<double>(n);
    velocity = vector<double>(n);
    best_position = vector<double>(n);
    best_value = numeric_limits<double>::max();
}
};

// Define the PSO algorithm
void pso(int n, int num_particles, int max_iterations,
         double c1, double c2, double w)
{
    // Initialize the particles
    vector<Particle> particles;
    random_device rd;
    mt19937 gen(rd());
    uniform_real_distribution<double> uniform_dist(-10.0, 10.0);
    for (int i = 0; i < num_particles; i++)
    {
        Particle p(n);
        for (int j = 0; j < n; j++)
        {
            p.position[j] = uniform_dist(gen);
            p.velocity[j] = uniform_dist(gen) / 10.0;
        }
        particles.push_back(p);
    }

    // Perform the iterations
    for (int iter = 0; iter < max_iterations; iter++)
    {
        // Update the particle velocities and positions
        for (auto &p : particles)
        {
            for (int j = 0; j < n; j++)
            {
                double r1 = uniform_dist(gen);
                double r2 = uniform_dist(gen);
                p.velocity[j] = w * p.velocity[j]
                    + c1 * r1 * (p.best_position[j] - p.position[j])
                    + c2 * r2 * (particles[0].best_position[j]
                        - p.position[j]);

                p.position[j] += p.velocity[j];
            }
            double value = function(p.position);
            if (value < p.best_value)
            {
                p.best_value = value;
                p.best_position = p.position;
            }
            if (value < particles[0].best_value)
            {
                particles[0].best_value = value;
                particles[0].best_position = p.position;
            }
        }
        cout << "Iteration_" << iter
            << ",_Best_value=" << particles[0].best_value << endl;
    }
}

```

} _____

Appendix B

Authorisation letter for Data use in MSc Thesis

Guillaume MARTIN
Responsable service Ingénierie Scientifique
EXAIL Robotics
220 Avenue des Frères Lumière
83200 La Garde

La Garde, le 21/07/2023

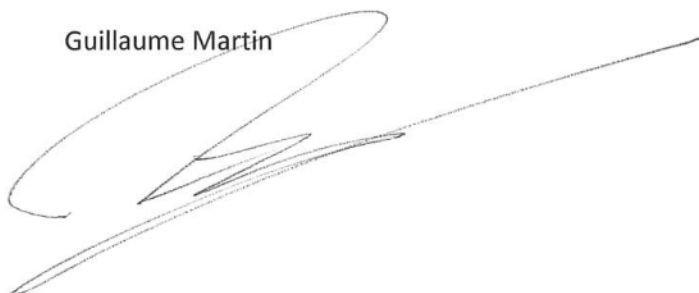
Objet : Autorisation d'utilisation d'images propriété EXAIL

En qualité de responsable du service ingénierie scientifique au sein du site Exail Robotics La Garde, j'autorise Noémie Hirtzig à inclure des images issues de ses travaux chez Exail au sein de son rapport de stage.

Toute utilisation ultérieure ou autre usage ne sont pas autorisés.

Cordialement,

Guillaume Martin

A handwritten signature in black ink, appearing to be 'Guillaume Martin', written over a horizontal line.



 **NTNU**

Norwegian University of
Science and Technology