FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Underwater mapping using a SONAR

João Pedro Bastos Fula

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof.Dr. Bruno Ferreira

Second Supervisor: Prof.Dr. Nuno Cruz

July 06, 2020

Resumo

Os *autonomous underwater vehicles* (AUVs) têm um papel crucial na investigação subaquática sendo que estes possibilitam a criação de mapas submarinos sem arriscar vidas humanas. A criação de mapas em três dimensões (3D) de ambientes estruturados subaquáticos é de grande importância para grande parte das áreas de investigação subaquática. Contudo, isto só é possível se for obtida uma estimação precisa da posição. Este estudo explora as capacidades do feixe de um *mechanical scanning imaging SONAR* (MSIS), de forma a construir um mapa dos arredores e encontrar a localização do veículo dentro do mapa. A informação do *sound navigation and ranging* (SONAR) alimenta uma técnica que extrai a distância ao primeiro ponto de interesse numa leitura do SONAR e um algoritmo de localização e mapeamento simultâneo (SLAM), composto por um modelo dinâmico inverso do SONAR que é usado para atribuir probabilidades aos vóxeis de um Octomaps e um filtro de partículas (PF). Este método é capaz de estimar a posição do veículo enquanto obtem o mapa do ambiente envolvente. A robustez deste método de construção de mapa autónomo foi testada em várias situações dentro de um ambiente estruturado com e sem movimento, que são apresentadas ao longo deste estudo.

Abstract

Autonomous underwater vehicles (AUVs) play a major role in underwater research since they enable the mapping of submarine regions without risk to human lives. The creation of three dimensional (3D) maps of underwater structured environments is of great importance for most underwater research fields. However, this is only possible if an accurate position estimation is obtained. This study explores the capacities of a mechanical scanning imaging sonar (MSIS) beam, in order to build a map of the surroundings and find the location of the vehicle within the map. The data from the sound navigation and ranging (SONAR) feeds a technique that extracts the range to the first feature within a SONAR scan and a simultaneous localization and mapping (SLAM) algorithm composed of a dynamic inverse SONAR model that is used to attribute probabilities to the voxels in an Octomaps and a particle filter (PF). This method is capable of estimating the pose of the vehicle while mapping the surrounding environment. The robustness of the autonomous map building method was tested through several situations within a structured environment with and without motion, which are presented throughout this study.

Acknowledgements

My acknowledgements go, in first place, to my supervisor, Professor Doctor Bruno Ferreira, for having accepted to supervise this dissertation and my work in the area in the last three years and to who I owe my enthusiasm for underwater robotics.

I would also like to thank my second supervisor, Professor Doctor Nuno Cruz, for all the knowledge and abstract thinking that brought a new dimension to the project.

To Professor Doctor Mingxi Zhou for readily and clearly aiding me in understanding a crucial part of the project.

To INESC TEC for supplying the equipment and allowing the use of their facilities to work and to test the proposal.

To my family for the unconditional love and for all the help, to my girlfriend for all the support and to my friends.

This work is financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within the project "POCI-01-0145-FEDER-006961", and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia as part of the project "UID/EEA/50014/2013".

João Fula

"Everything is theoretically impossible
until it is done."

Robert A. Heilein

Contents

1 Introduction **1**
- 1.1 Motivation . . . 2
- 1.2 Objective . . . 2
- 1.3 Dissertation structure . . . 2

2 State of the art **5**
- 2.1 Localization techniques and sensors . . . 6
  - 2.1.1 Inertial navigation . . . 6
  - 2.1.2 Trilateration and triangulation . . . 7
  - 2.1.3 Scan matching . . . 9
- 2.2 State estimation . . . 11
  - 2.2.1 Kalman filter . . . 12
  - 2.2.2 Particle filter . . . 13
- 2.3 Survey of mapping representations and techniques . . . 15
  - 2.3.1 Mapping representations . . . 15
  - 2.3.2 Probability attribution . . . 17
- 2.4 Simultaneous localization and mapping . . . 18
- 2.5 Related work . . . 20

3 Map building and position estimation **23**
- 3.1 Mapping . . . 25
  - 3.1.1 Range to first feature . . . 25
  - 3.1.2 Probability attribution . . . 26
  - 3.1.3 Probability update . . . 28
- 3.2 Localization . . . 29
- 3.3 Simultaneous localization and mapping . . . 34

4 Experimental setup and results **37**
- 4.1 Setup description . . . 37
- 4.2 Data collection experiments . . . 39
- 4.3 Performance analysis . . . 43
- 4.4 Synthesis . . . 49

5 Discussion **51**

6 Conclusions and future work **53**
- 6.1 Objective achievement . . . 53
- 6.2 Future work . . . 53

List of Figures

2.1 Octomaps (left) and Octrees (right). 16

2.2 Representation of the angles within a SONAR beam. The black and blue ellipse represents the vehicle, the SONAR is represented as a black circle, the SONAR beam in yellow, the center of the beam in a dashed grey line and a ray within the beam is represented in red. 19

3.1 A block diagram of the system overview with the main blocks that will take part in the project 24

3.2 The graphical representation of the distance between a particle and the outer boundaries of the map for a ray with angle α, within the SONAR beam with angle β. The map is represented as the grid, the particle as the black dot and the intersection with the outer map limits by a red cross. The length of the grey dotted line represents the observed distance for the angle β. 32

4.1 Water tank where the tests were carried out, with the location of every corner and a representation of the vehicle. The positions of the corners are relative to the static tests from the second to the fourth test. 37

4.2 The SHAD AUV used for the testing of the algorithm in this study. 38

4.3 Graphical representation of three vertical rays placed within the SONAR beam volume. The rays can be expanded vertically and horizontally. 39

4.4 The position of the AUV within the water tank and the ground truth positions of the corners of the tank in meters. 40

4.5 Representation of the second test case with the ground truth localization of the vehicle and the beacon (red). 41

4.6 Representation of the third test case with the ground truth localization of the vehicle and the initial and final position of the beacon (red). 41

4.7 Representation of the fourth test case with the ground truth localization of the vehicle, the localization of the small pole (grey with black outline) and the initial and final position of the beacon (red). 42

4.8 Representation of the fifth and last test case with the ground truth localization of the vehicle's initial and final position, the localization of the pole (grey with black outline) and the position of the beacon (red). 42

4.9 Images drawn from SONAR arrays for the first test case. without any processing (a) and after running the *range to first feature* algorithm (b). 43

4.10 Image of the occupied region in an octomaps representation with nine rays per SONAR beam after one full SONAR revolution. Axis are in millimeters. 44

4.11 The square of the errors in relation to the angle variations for the first test. 45

4.12 Image of the free voxels in an Octomaps representation (blue) with the observation of the third iteration of the algorithm (red), taken in the first test. Axis are in millimeters. 45
4.13 The square of the errors in relation to the angle variations for the second test. . . 46
4.14 Image of the free voxels in an Octomaps representation (blue) with the observation of the seventh iteration of the algorithm (red) (a) and (b); Image of the occupied region in an octomaps representation (c) and (d). Both refer to the second test. Axis are in millimeters. 46
4.15 The square of the errors in relation to the angle variations for the third test. 47
4.16 Image of the free voxels in an Octomaps representation (blue) with the observation of the twenty first iteration of the algorithm (red), after movement from the buoy, taken in the third test. Axis are in millimeters. 48
4.17 The square of the errors in relation to the angle variations for the fourth test. . . . 48
4.18 Image of the free voxels in an Octomaps representation (blue) with the observation of the seventeenth iteration of the algorithm (red), after movement from the buoy, taken in the fourth test. Axis are in millimeters. 49
4.19 The square of the errors in relation to the angle variations for the fifth test. 49
4.20 Image of the free voxels in an Octomaps representation (blue) with the observation of the seventeenth iteration of the algorithm (red), after movement of the vehicle, taken in the fifth test. Axis are in millimeters. 50

List of Tables

2.1 Types of underwater acoustic positioning systems. 8
2.2 Most common state estimation techniques. 11
2.3 Parameter definition and reference. 17
2.4 Common SLAM methods. 20

3.1 Set of steps that compose the particle filter algorithm. 30
3.2 Pseudocode for the *initialize particles* algorithm. 31
3.3 Pseudocode for the *predict movement* algorithm. 31
3.4 Pseudocode for the *update* step from the particle filter. 33
3.5 Pseudocode for the *normalization* step. 33
3.6 Pseudocode for the *end of the particle filter iteration* where the position estimation is produced. 34
3.7 Pseudocode for the *systematic resampling* algorithm. 35

4.1 Errors in the position and heading after some iterations for each test. Errors are in meters 43

Abbreviations and symbols

3D	Three Dimensions
AHRS	Attitude and Heading Reference System
AOA	Angle Of Arrival
ASC	Autonomous Surface Craft
AUV	Autonomous Underwater Vehicles
CRAS	Center of Robotics and Autonomous Systems
DGPS	Differential Global Positioning System
DR	Dead Reckoning
DT	Detection Threshold
DVL	Doppler Velocity Log
EKF	Extended Kalman Filter
GIB	Global Positioning System Intelligent Buoy
GPS	Global Positioning System
IAE	Innovation Adaptive Estimation
ICP	Iterative Closest Points
IMU	Inertial Measurement Unit
INESC TEC	Institute for Systems and Computer Engineering, Technology and Science
INS	Inertial Navigation System
KF	Kalman Filter
MCL	Monte Carlo Localization
MMAE	Multiple Model Adaptive Estimation
MSIS	Mechanical Scanning Imaging SOund NAvigation and Ranging
OG	Occupancy Grid
PDF	Probability Distribution Function
PF	Particle Filter
pIC	probabilistic Iterative Correspondence
PSM	Polar Scan Matching
RBPF	Rao-Blackwellized Particle Filter
SHAD	Small Hovering AUV with Differential actuation
SONAR	SOund NAvigation and Ranging
TOA	Time Of Arrival

Chapter 1

Introduction

Autonomous underwater vehicles are increasing in importance for their many underwater applications. They are nowadays being used in oceanographic surveys, bathymetric data collection in marine environments, underwater search and rescue missions, climate change assessments, etc. To perform these activities, the AUV has to be able to navigate. However, this is a challenging topic in the underwater robotics field of research. There are four main blocks that compose navigation: *perception*, the AUV must extract meaningful information from the sensor's data; *localization*, the vehicle must estimate its pose (position and orientation) in the environment; *cognition*, the robot must plan the next move in order to successfully complete its goals; *motion control*, the actuators must begin functioning to move towards the next objective.

Mapping allows the understanding of the surrounding environment and is a key component in the planning process. Furthermore, the *localization* block in the navigation process depends on the existence of a map in order to function. By analysing a map and the surroundings, it is possible to estimate the localization on the map and decide what course of action to take given the newly acquired knowledge. Be it to stand still and enhance the current map estimation, gather samples of wildlife in the vicinity or define a new target position. However, in some cases, it is necessary to map the region in real-time. A mapping algorithm that creates a probabilistic map of the environment may be crucial in order to navigate. But a mapping algorithm is only as effective as the vehicle's capability to locate itself. Since the localization depends on the features of the environment and the mapping depends on the knowledge of the vehicle's position, there is a circular dependency between mapping and localization.

This circular dependency is solved by SLAM. This is a series of algorithms that are able to produce a localization and a map estimation continuously. Because of this, it is not necessary to have a previous map of the region and the vehicle can move since the pose is being estimated. However, this is arguably the most difficult problem to solve in underwater robotics.

This masters dissertation is integrated in the master in electrical and computers engineering course and came from the cooperation of the main author with the center of robotics and autonomous systems (CRAS) of the institute for systems and computer engineering, technology and science (INESC TEC).

1.1 Motivation

When travelling below the ocean's surface, some technologies, like the conventional global positioning system (GPS) and radio-frequency signals are severely constricted. So in order to perform the tasks in hand, other methods should be used. In this environment, acoustic-based sensors and communications have an overall better performance. Even though these sensors work better, there are still several noise sources that affect the readings. Therefore, these sensors and communications must go through a filtering process that discards most of these noises. Common localization technologies include the necessity of installing beacons prior to the missions. This comes with costs both money-wise and time-wise. Therefore, there is a need to develop a method that is able to produce accurate maps and localization estimations without the need of equipment external to the vehicle. Common technologies that solve this problem implement heavy weight or computationally heavy equipment that prevents them from being implemented in every type of aquatic vehicle, for example, lightweight vehicles.

1.2 Objective

The objective of this dissertation is to develop an autonomous underwater map building algorithm through data collected from an imaging sensor to be used in a variety of aquatic vehicles.

From the already existing small hovering AUV with differential actuation (SHAD), emerged the challenge of developing an autonomous map building technique. This vehicle is the testing platform for which this study is developed, with the main goal of creating an autonomous map building method applicable to several vehicle types. Since the sensor options for small vehicles are more restricted than for other types, the SHAD is a great testing platform to develop a technique adaptable to several vehicle varieties. From the sensor options available in this vehicle, the SONAR sensor was selected to incorporate this technique. It is possible however, to change the used sensor, as long as it provides the information regarding the surrounding environments, similarly to the MSIS sensor. Furthermore, the algorithm developed integrates a SLAM method composed of an Octomaps and a PF implementation.

It is expected that the work here produced may help to enhance further optimizations of small AUV mapping systems underwater, which may impact the development of more accurate maps and further deepen scientific knowledge in this field of research.

1.3 Dissertation structure

Besides the introduction, this dissertation is composed of five more chapters. In chapter 2, the state of the art is described. Here is where the sum up of the latest and most common technologies in the area are detailed. Furthermore, some previous studies that approach the same topic are also presented to later be discussed with the outcome of this study. After having reviewed the state of the art, a choice on the approach to the problem was made and it is described in chapter 3. Here,

an overall description of the structure of the algorithm, its components and an in depth exposure of the objectives, can be found. Furthermore, this chapter presents a description of the several methods that compose the project and how they relate to one another in order to accomplish the objectives. The experimental setup and the results of the several tests is introduced in chapter 4. The outcome of the method developed in this study is presented in this chapter. Moreover, the resulting maps, created by the algorithm, are presented and analysed. Chapter 5 contains the discussion of the project and comparison with the related work. The final chapter contains the conclusions, objective satisfaction and a discussion about future work.

Chapter 2

State of the art

In order to understand which is the current state of the art in autonomous map building, a research focused on localization techniques, sensors, mapping and SLAM algorithms, was made. This research resulted in the choice of SONAR sensors, which is a crucial aspect that is highlighted in this state of the art review.

This section contains a revision on several techniques used for localization and a description of the sensors that best suit each technique. Furthermore, a focused analysis of Kalman filters (KFs) and PFs is also presented. The selection of these two algorithms came from being the basis for most algorithms found in robotics localization. Moreover, mapping representation and map update techniques, with a larger discussion about Octomaps is approached in this chapter. A description of techniques that integrate both localization and mapping in several SLAM algorithms are also presented in this chapter. Lastly, a revision over studies with similar objectives and implementations was made.

When traveling below the ocean's surface, some technologies, like the conventional global positioning system (GPS) and radio-frequency signals are severely constricted. So in order to perform the tasks in hand, other methods should be considered (Hidalgo and Bräunl, 2015a). On the other hand, in this environment, acoustic-based sensors and communications have an overall better performance. Some of the most common acoustic interferences, as suggested by Chitre et al. (2005), Kohlbrecher et al. (2011), Paull et al. (2014), are:

- Small bandwidth, which means certain techniques have to be used in order to share information;
- High ambient noise;
- Low data rate, which constrains the amount of data transmitted;
- High latency because the speed of sound in water is slow compared to the speed of light;
- Variable sound speed due to factors like temperature and salinity;
- Multipath because of the existence of an upper and lower boundary together with the variation in sound speed;

- Unreliability which means that the system has to deal with data loss.

2.1 Localization techniques and sensors

Localization techniques allows the pose of the vehicle to be estimated. This is a crucial step to autonomous map building as without having the localization, it is not possible to match map readings to one another.

Sensors provide the measurements for the localization techniques to process. Each localization technique requires a specific set of sensors to function properly. These sensors are described under the localization methods. The sensor types within the localization techniques can be divided accordingly (Paull et al., 2014):

- Inertial/Dead reckoning (DR): Measures retrieved by accelerometers, gyroscope and doppler velocity log (DVL) to increase the accuracy in state propagation.
- Acoustic transponders and modems: Based on measuring time of arrival (TOA) and the angle of arrival (AOA) of acoustic beacons or modems. These sensors usually run triangulation or trilateration algorithms in order to find the vehicle's localization.
- Geophysical: Localization based on external environmental information. This final sensor type is usually related with scan matching techniques.

SONAR sensors are distinct from other acoustic-based systems because they rely on external environment, rather than on external signals.

2.1.1 Inertial navigation

When in open sea, the AUV can find itself not getting any measurements from the surrounding environment. When this happens, the vehicle is DR. In this state, the AUV predicts its position based only on the knowledge of its orientation and known or estimated velocities.

DR is not used as a primary method of navigation but it is important in modern navigation systems. DR's main issue is that the errors keep adding up, so the AUV's position error grows with the distance.

Another option for navigation is the use of an algorithm based on particle filtering that uses bathymetric maps in order to navigate. Some errors may occur with the change of the tide level, especially in low bathymetric variation, but this can be avoided if the tide level is monitored. This is called "terrain-aided navigation". It is a good option for DVL and multibeam SONARs. The most common filtering scheme in this type of navigation is the EKF.

In this localization method, the most commonly used sensors are:

- Inertial measurement unit: The inertial measurement unit (IMU) is a sensor that works well in a DR navigation system. It consists of a combination of accelerometers, gyroscopes and sometimes magnetometers to estimate the vehicle's orientation, velocity and acceleration.

- Doppler Velocity Log: The DVL is a device that tracks the ocean's bottom through the use of three to four downward looking beam transducers. The sensor measures the apparent bottom velocity along the beams and processes the responses to compute the velocities according to the Doppler effect (L. Chen et al., 2013; Paull et al., 2014; Rigby et al., 2006). When the DVL cannot detect the bottom, it estimates the velocity based on the surrounding water. Since water is not a fixed frame, this option is less desirable than the seafloor tracking. This sensor produces good velocity estimations without requiring pre-deployed stations. However, it is characterized for being big, a few dozen centimeters both in diameter and in height, and heavy, generally around 6 kilograms, and, therefore, it may not be suitable for small or light AUVs.

2.1.2 Trilateration and triangulation

Given several distances to beacons with a well known position, it is possible to determine the location of the vehicle, with high precision, through a simple algorithm. For trilateration, the minimum beacons for a 2D and 3D localization is 3 and 4, respectively. The position of the vehicle is estimated through the solution of a non-linear optimization that minimizes the error between the expected and actual distances from the beacons to the vehicle. The triangulation approach, unlike the trilateration, involves angles. There have been several proposed triangulation algorithms, such as the ones described by L. Chen et al. (2013). The main setbacks of these algorithms are when there are not enough visible beacons to extract a good position or if the beacons and the vehicle are all positioned inside the same circumference.

Within this localization method, for surface implementations, the most common sensor is the GPS. This device provides absolute location and time information anywhere on or near the earth. The position is estimated using the time of flight of the signals from synchronized satellites. These signals are affected by the technique used, atmospheric conditions, the number of satellites and others (L. Chen et al., 2013; Grewal et al., 2007). These errors and the overall accuracy of the sensor can be improved if integrated with an inertial navigation system (INS). This sensor can be applied on AUVs working on shallow waters without a long period of submergence and it is widely used in surface vehicles (L. Chen et al., 2013; Paull et al., 2014). The GPS is a must in vehicles that are expected to navigate in shallow waters since it provides a good position estimation, it generally has a low cost and it is a light component. However, GPS signals are severely constricted in higher depths, and therefore, AUVs that navigate outside the surface should not rely on GPS (Paull et al., 2014).

There are, however, underwater solutions that utilize triangulation or trilateration in order to solve localization. These are the underwater acoustic positioning systems. These systems estimate the location of the vehicle relative to the position of known baseline stations. The location of these stations is well known and the vehicle's position is acquired through triangulation or trilateration. Table 2.1 contains a description of the most common underwater acoustic positioning systems UAPS types. These techniques provide accurate position estimations and are useful in many cases.

However, UAPS require pre-deployment of one or more stations and that comes with costs, both time and money wise.

Table 2.1: Types of underwater acoustic positioning systems based on L. Chen et al. (2013), Paull et al. (2014).

Acoustic method	Description
SBL	In this method, the beacons are placed on opposite ends of an aiding surface vehicle. The localization is obtained by the difference between the TOA of a transponder attached to the vehicle and the TOA of the transmitters placed in the surface vessel. The baseline length is usually from 20 to 50 meters (L. Chen et al., 2013). This method is not optimal because the accuracy of the position is based on the length of the vehicle.
LBL	This method consists of placing a beacon network on the seafloor and derive the position based on the network. Generally, the transponders are installed in the corners of the operation area. The position is extracted by triangulating the TOA coming from or to the stations. The baseline length is from 100 to 6000 meters (L. Chen et al., 2013). LBL however, faces limited ranges and the reliance on the speed of sound of the water column.
USBL	Unlike SBL and LBL, this method does not use triangulation. The target distance from a transducer pole is measured by signal run time. The direction is determined through the phase shift of the reply signal. The baseline length is generally below 10cm (L. Chen et al., 2013). USBL is usually integrated with DVL and INS for localization improvement.
Single Fixed Beacon	Similarly to LBL, this technique is based on the installation of seafloor beacon installation but reduces the network of beacons to a single one. The network is simulated by propagating the same signal range until a new update is received. This technique struggles when the vehicle is moving directly toward of away from the beacon for it will cause unbounded errors (Paull et al., 2014).
GPS intelligent buoys (GIBs)	This system is based on the placement of intelligent buoys equipped with differential GPS (DGPS) and submerged hydrophones and the localization is based on the triangulation of the signals. This technique has reduced costs in relation to LBL, since it does not need prior installation of beacons on the seafloor or posterior recovery of these.
Acoustic modem	The full autonomy of the AUV can be accomplished with the help of Autonomous Surface Crafts (ASCs) equipped with acoustic modems. However, the acoustic modem is large, heavy and expensive so not all AUVs are able to implement it. This method negates the necessity to georeference the beacons and allows the beacons to move throughout the mission.

2.1.3 Scan matching

The process of scan matching consists of comparing laser scans with one another, called motion estimation, or with an existing map, called position estimation. Martínez et al. (2006) propose an organization of these techniques under the following categories:

- Feature-based techniques use features of the surrounding structure to use for localization, such as corners (Shaffer et al., 1993), line segment (Reina and González-Jiménez, 2000) or edges (Weber et al., 2002).

- Compact data methods use histograms, eigenvectors or motion fields.

- Point matching techniques. The most common in these category are the iterative closest points (ICP) (Besl and McKay, 1992), the probabilistic iterative correspondence (pIC) (Montesano et al., 2005) and the polar scan matching (PSM) (Diosi and Kleeman, 2007). The first consists of computing the correspondence between two scans and then calculating the transformation $T = [\phi_0, T_X, T_Y]$ that minimizes the Euclidean distance between two corresponding points (Besl and McKay, 1992). The pIC is a variant of the IPC that improves the performance in terms of efficiency and accuracy. The PSM introduces a preprocessing to eliminate outliers of a reference and the current scan. Then it projects the currents scan into the reference and extracts the translation estimation. Hernández et al. (2009), present a scan grabbing algorithm using data from a MSIS to use within a semi-artificial environment, which was given the name of MSISpIC.

In this method to obtain the localization, the most used sensors are:

- Sound navigation and ranging: A SONAR device uses acoustics to detect and locate objects. SONARs can be divided into two main categories, active and passive. Active SONARs emit a sound wave and await its return. The distance to the object is given by measuring the time between the emission and the return of the wave and multiplying by the underwater sound speed. Passive SONARs, unlike active ones, do not emit any wave and instead only listen to outside sounds. Sounds made by vessels and sea-life are detected by this type of SONAR. SONARs work better underwater than on the surface since sound propagates better in water (Hidalgo and Bräunl, 2015b; Paull et al., 2014). Active SONAR sensors can be divided into three main types:

  - MSIS:
    A MSIS is a device that rotates the SONAR beam through a series of small steps. In every step, it emits a sound and awaits its reflection. An echo intensity profile is created for each beam. After performing a full revolution, a 2D acoustic image of the environment is retrieved. Generally, the SONAR beam has a pyramidal shape, since it has vertical and horizontal widths (L. Chen et al., 2013; International, 2019).

  - Multi-beam SONAR:
    A multi-beam SONAR can map several locations of the ocean floor with a single ping. The resolution of this mapping is higher than the conventional SONARs. In most cases, the multi-beam SONAR is mounted on the lower part of the hull of the AUV looking down and is used for mapping.
  - Side-scan SONAR:
    This device collects higher resolution data when it is positioned closer to the sea bottom. These sensors are often used to collect data of the seafloor. It can also be used to compare readings with *a priori* map in order to extract the localization (L. Chen et al., 2013).

- Optical sensors: Video cameras and laser-based vision fall into this category. The first are limited to short range because they are easily influenced by factors like visibility and lighting. Video cameras have their uses in object recognition but are usually only used in controlled environments or in near surface applications (Abukawa et al., 2016; Abukawa et al., 2015). Laser-based systems, on the other hand, are able to produce consistent results in any environment.

  Laser-based systems consist of a laser working together with a camera in order to extract the 3D shape of objects. It is not subject to the factors that limit the use of video cameras due to the intensity of the laser beam that is barely affected by water (Gomaa et al., 2015). The beam allows the measurement of distances to each object and the camera allows for shape recognition (Himri et al., 2018). This allows the sensors to produce highly accurate results in any environment (Karras et al., 2006; Rodrigues et al., 2018) and even at hundreds of meters from the target (Massot-Campos and Codina, 2015). However, these sensors are quite expensive, falling into the thousands of euros price range and are also characterized for being computationally heavy.

There are some sensors that do not fall exactly into any of the above categories but some are even crucial in underwater navigation. These are, for example (Paull et al., 2014):

- Pressure sensor: Pressure is a force that increases as you pull away from the ocean's surface. Therefore, it is possible to measure the depth of the vehicle by implementing a pressure sensor. This sensor is a must in most AUVs since it produces an accurate estimation of the vehicle's depth at a low cost.

- Compass: Compasses, based on the earthly magnetic field, shows the direction angle relative to the north direction in angles. For open seas missions, this type of sensor produces a good heading estimation and therefore is quite useful. However, within structured environments or next to any object that produces a magnetic field, this sensor tends to have a poor behaviour.

2.2 State estimation

State estimation algorithms save the pose of the vehicle in each time t in order to weight the effect that each observation should have on the current state of the vehicle's pose. If there are inconsistencies between one measurement and the remainder in the state space, the effect of that reading in the state is significantly reduced due to these types of algorithms. A short description of some of the state estimation techniques is present on table 2.2. It contains a description of the algorithm implemented by each technique and a small conclusion regarding operation time and computational needs.

Table 2.2: Most common state estimation techniques based on L. Chen et al. (2013), Paull et al. (2014), Welch and Bishop (1995).

State estimator	Description
Kalman Filter	Filter parameterized by mean, μ and covariance, Σ. It requires the state propagation model and the measurement model. The first describes the state transition, while the second relates to the relation between the system and the environment. Works well where both models are linear.
Extended Kalman Filter	It is a variation of the Kalman Filter that allows the filter to work on non-linear processes and measurement models. It uses the Jacobians to linearize the estimation around the present state. Prediction is fast but measurement update takes a long time.
Unscented Kalman Filter (UKF)	Another variant of the Kalman Filter that estimates the mean and covariance through a sampling approach. It is slower than the EKF but more accurate and often requires less computation (Khazraj et al., 2016).
Extended Information Filter	Can process multiple measurements simultaneously easily through addition. Prediction is slow but measurement update is fast.
Particle Filter	Distribution is represented by particles with associated weights. Computation increases with the number of particles. Non-Gaussian distributions and non-linear models can be added.
Least Squares Regression	State estimate is formulated posterior to the testing as a least square optimization which can be formulated analytically. Past states are maintained which can be useful for trajectory optimization or SLAM.
Adaptive Kalman Filter	Follows the principle that the process covariance matrix (Q) and the measurement noise covariance (R) change through continuous iteration. There are two main approaches: Multiple model adaptive estimation (MMAE) and innovation adaptive estimation (IAE). The innovation is the difference between the estimation and the measurement. In the first approach, a set of Kalman filters run in parallel with different models for Q and R. Each of these has a weight based on the innovation parameter. Then the optimal state is determined as the weighted sum of each estimate (L. Chen et al., 2013). For the IAE, the matrixes Q and R, are updated discretely based on the innovation sequence.

Generally, state estimation is divided into two parts: The prediction and the update. All the sensors that give information regarding movement or velocities are inputs on the first part while

sensors that contain information regarding the observation, are inputs on the latter. The most commonly used state estimators are the Kalman filter and its variants. These algorithms work mostly as a way to fuse the data from various sensors and get this information to work together to create an estimation of the location of the vehicle. INS sensors are key in sensor fusion. Generally these are implemented together with GPS, DVL's, SONARs or visual sensors.

In Fula et al. (2018), a map was given to the AUV so that it could self-localize within the tank, by comparing the MSIS observations with the map. It uses an extended kalman filter (EKF) in order to estimate the state of the vehicle at each SONAR scan revolution. H. Zhang et al. (2018) combined Monte Carlo localization (MCL) with an EKF. The filter is used to update the current state and covariance to exclude useless particles that were created by the MCL.

2.2.1 Kalman filter

The Kalman filter is a technique for implementing Bayes filters. This state estimator computes a belief for continuous states. At time t, the belief is represented as μ_t and the covariance Σ_t. The posterior beliefs are Gaussians, if the following conditions are verified:

1. The next state probability, $p(x_t|u_t,x_{t-1})$ must be linear and with Gaussian noise:

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{2.1}$$

   where, x_t and x_{t-1} are the state vectors at time t and $t-1$, respectively and u_t is the control vector at time t. A_t and B_t are both matrixes. The linearity in this function is acquired from the multiplication between A_t and B_t and the state and control vectors, respectively. The noise variable ε_t is a random Gaussian vector that adds randomness to the state transition.

2. The measurement probability, $p(z_t|x_t)$ must also be linear with added Gaussian noise.

$$z_t = C_t x_t + \delta_t \tag{2.2}$$

   The matrix C_t is the Gaussian that assures linearity in the arguments and δ_t is the measurement noise with covariance Q_t.

3. The initial belief $bel(x_0)$ must be normal distributed.

From this assumptions, it is possible to ensure that the posterior belief $bel(x_t)$ is always Gaussian (Thrun et al., 2005).

The algorithm takes as inputs the belief at time $t-1$, represented as μ_{t-1} and Σ_{t-1}. These variables are updated according to the control u_t and the measurement z_t. The output of the filter is μ_t and Σ_t (Fula et al., 2018; Thrun et al., 2005).

This algorithm for this filter can be divided into two stages:

1. Belief propagation:

$$\bar{\mu}_t = A_t\,\mu_{t-1} + B_t\,u_t$$

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$$

2. Posterior belief:

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$$

$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

where R_t and Q_t are the covariances of the random noise vector ε_t present in eq. (2.1) and the covariance of the measurement noise vector δ_t present in eq. (2.2). The kalman gain is represented as K_t and it is the variable that specifies the degree of effect the measurement has on the new state estimate.

2.2.2 Particle filter

The PF is an algorithm that, unlike KF, EKF or UKF, is not restricted to its posterior Gaussian distribution nor to an unimodal solution (Yozevitch and Ben-Moshe, 2016). It is part of the Bayesian filters family. The idea of the PF is to represent the belief $bel(x_t)$ by a set of random state samples. The particles are represented as:

$$x_t := x_t^1, x_t^2, \ldots, x_t^N \tag{2.3}$$

where N represents the number of particles. Each particle has an associated weight w_t^K which is proportional to the likelihood of that particle (Thrun et al., 2005; Yozevitch and Ben-Moshe, 2016).

$$x_t^{(K)} \sim p(x_t | z_{1:t}, u_{1:t}) * w_t^{(K)} \tag{2.4}$$

where $p(x_t | z_{1:t}, u_{1:t})$ represents the likelihood of the particle with a known observation and motion models at time t. The motion part is usually predicted using inertial sensors such as IMUs ou DVLs. The observation model is taken from sensors that give information regarding the surrounding environment. The weight of a particle can be evaluated by measuring the distances to M features and comparing them with the observation model (Thrun et al., 2005; Yozevitch and Ben-Moshe, 2016):

$$p(x_t) \sim \sum_{i=1}^{M} f(x_i | \mu_i, \sigma^2) \tag{2.5}$$

where $f(x_i | \mu_i, \sigma^2)$ is given by:

$$f(x_i | \mu_i, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{\frac{-(x_i - \mu_i)}{2\sigma^2}} \tag{2.6}$$

where x_i is the distance to the ith feature and μ_i is the predicted distance between the particle and the feature.

Resampling is a major step in the PF algorithm. It consists of creating and distributing a new set of particles based on the probability distribution function (PDF) that is determined by the previous steps of the algorithm. Simultaneously, it attempts to avoid degeneracy through the exploration of the space state and modification of the random measure. This means that measures with higher weights are more likely to be included several times in the modified state space. It is important, however, that the random measure approximates the original distribution (Kozierski et al., 2013; T. Li et al., 2015; T. Li et al., 2012). Doucet and Johansen (2009), state that sample degeneracy is a consequence of resampling but it comes from a deeper problem that resampling mitigates. It is impossible to represent a distribution on a space of high dimension with fixed sample size. Sample impoverishment happens in the lack of resampling and has a similar effect to that of sample degeneracy. Increasing the number of particles in every iteration also doesn't solve any of the problems as this would lead to exponential growth in sample size. Resampling acts like a reset in the system in a way that diminishes the quality of the path samples, in order to maintain a good representation of the final time marginals. By focusing on this, it avoids the problem of increasing dimensionality. There are several resampling algorithms:

- Single-distribution sampling group

  Multinomial resampling: The complexity of this algorithm is $O(NM)$ where M comes from the search of the mth particle that satisfies the multinomial condition. This method is not optimal (Carpenter et al., 1999; T. Li et al., 2015).

  Stratified resampling: The complexity of the stratified resampling is $O(N)$ and therefore an improvement regarding the previously suggested algorithm.

  Systematic resampling: The complexity of this method is the same as the stratified resampling. However, this algorithm is computationally more efficient than the previous due to it only having to generate one random number.

  Residual resampling: Residual resampling runs through two stages. The first is a replication of each particle whose weight is bigger than $1/N$. The second stage randomly samples using the remaining weights. The complexity is the sum of complexity of the two stages.

- Compound sampling group

  Optimal resampling: The optimal resampling (Fearnhead and Clifford, 2003; T. Li et al., 2015) uses a dynamic threshold in order to separate the groups. All the particles with weights above the threshold are preserved instead of replicated. The remaining particles are discarded. This means that the sample size reduces at each iteration and it may not be advised in every situation. Furthermore, this algorithm has high computational requirements.

  Reallocation resampling: Unlike the optimal resampling, reallocation resampling (J. Liu and Logvinenko, 2001; T. Li et al., 2015) uses a fixed threshold. This value is arbitrary but

generally defined as $1/N$. The sample size, similarly to optimal resampling also decreases and therefore, not advised for every implementation.

The main difference between the two resampling groups is that the single-distribution sampling methods resample the same mth particle, $N_t^{(m)}$ times. The number $N_t^{(m)}$ is proportional to the weight of that particle $\omega_t^{(m)}$. This means that all particles are sampled in the same way.

The compound sampling methods group particles by predefined criteria prior to resampling and resample each group differently. Usually, these criteria are defined by the weight threshold, therefore, similarly weighted particles fall within the same group. Each group is resampled in a different way. These methods decrease operating time and preserve diversity. Every particle produced by these methods, satisfy the condition of unbiasedness and are all equally weighted (T. Li et al., 2015).

2.3 Survey of mapping representations and techniques

Maps are crucial for navigation since they are the reference point to extract a position from a certain observation. This section analyses several mapping representations and probability attribution methods.

2.3.1 Mapping representations

Mapping representations are the strategies used to illustrate the map. This affects the way the points that compose a map are saved which influences the probabilistic methods used, the memory consumption and the operation time. There are several map representations that vary from keeping a structure that save the position of a feature to saving the three dimensional position of cubes, their edge size and the occupancy probability.

2.3.1.1 Occupancy grid

One of the map representations that is most commonly used is the occupancy grid (OG). It consists of mapping an unknown environment into a binary matrix. The zeros represent unknown or occupied locations and the ones represent free cells. This representation has some limitations, for example, the map size does not change, which makes this representation unreliable when the dimensions of the environment are unknown (Kundu et al., 2016). Furthermore, OGs are used for two dimensional representations and for three dimensional maps, other representation forms should be considered.

Ferri et al., 2019 present a novel OG mapping framework that builds the grid iteratively as acoustic measurements are collected. This algorithm discards moving targets as they are not a part of the map. Lee and Lee (2016) propose a method of representation of the seafloor using OGs with 3D point cloud. In this study, a multi-beam SONAR was used and the depth map was built by means of a Bayesian-updating model.

2.3.1.2 Octomaps

Octomaps are based in Octrees, this is, the map is divided into cells, each one represented by a node, that is recursively subdivided into 8 smaller nodes, until a defined size or a maximum cell number (De Silva et al., 2018; Floriani et al., 2017; Vasquez-Gomez et al., 2013). This division only happens when there is a probability of the voxel containing an object. Due to this feature, Octomaps is considered to be an efficient algorithm in terms of memory. A voxel is the Octomaps representation of a cube with a defined center point and a defined edge size. Fig. 2.1, shows a graphical illustration of the cubes that represent the voxels in the Octomaps and the nodes that compose the Octrees. In this image, the subdivision into smaller voxels or nodes is also visible.

Octomaps are recursive and always adapting, which means that the size of the map increases as the vehicle travels, so it deals with one of the major limitations of the occupancy grids.

Hornung et al. (2013) proposes an algorithm that divides the environment in free, occupied, or unknown cells. It makes an easy to understand map and allows us to use some of the capabilities of the MSIS, such as the ability to focus the SONAR beam on a specific area, such as the unknown parts of the map, to enhance the map.

2.3.1.3 Feature based maps

This representation is characterized by mapping an area through points of interest found within the retrieved data from the surrounding environments. Features are points that aid in localizing the vehicle. The strength of a feature is given by the information regarding localization that the feature provides. For example, a wall would not be a strong feature as it is generally not possible to know exactly where the observation was taken, while corners, edges and any object that stands out, have known positions and it is possible to estimate the vehicle's position based on those.

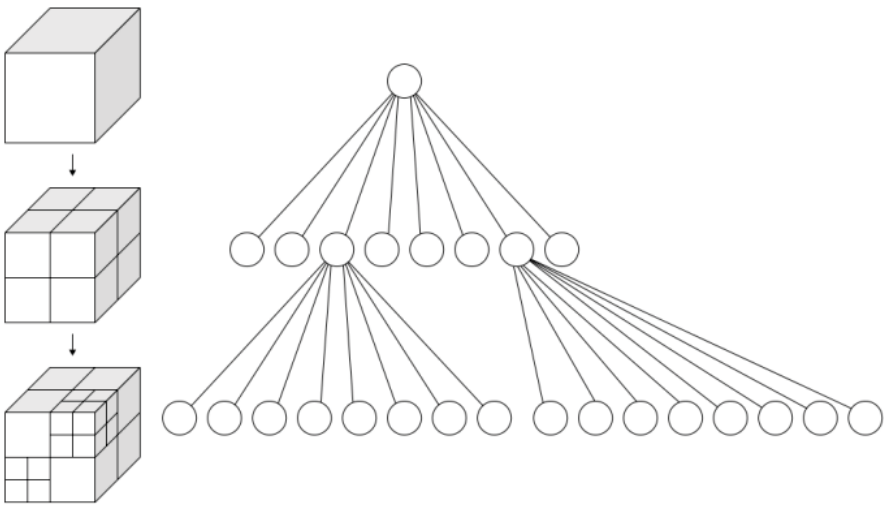


Figure 2.1: Octomaps (left) and Octrees (right) (De Silva et al., 2018).

2.3.2 Probability attribution

The data returned from SONARs is not certain and similarly to the state estimation in the localization section, probabilistic methods play a role in mapping.

The SONAR beam is characterized by its shape. Unlike the pin point of a laser based system, this beam has a certain coverage area. The wideness of the area depends on the specifications of the SONAR model used. On the other hand, the data recovered by the SONAR is an array of intensities in relation with the distance. Due to this, it is impossible to know the exact shape of the detection based on a single SONAR beam. Therefore, a SONAR model must be used to attempt to create a closer approximation to reality. The complexity of these models depends on the operation they are meant to be used in. In missions where a higher accuracy is necessary, the transmission loss must be calculated along multiple paths. Noise sources must be taken into account. Scattering and reverberation produced by bathymetric features must have efficient and realistic models. The calculations for these factors become even more complicated in shallow water, where the acoustic interactions with the unpredictable and variable sea-floor topographies add further compounds to the models.

SONAR equations are used to quantify aspects of the sensor's performance (Etter, 1995). Table 2.3 contains a definition of the parameters used in these equations.

Active SONAR models vary with noise or reverberation background. A simplified version of the equations is, as proposed by Etter (1995), Urick (1995), Zhou et al. (2016):

Active SONARs: Noise Background

$$SL - 2TL + TS = NL - DI + RD \tag{2.7}$$

Table 2.3: Parameter definition and reference based in Etter (1995), Urick (1995), Zhou et al. (2016).

Symbol	Parameter	Origin
SL	Source level	Based on the transducer power.
TL	Transmission loss	Sound covered area.
TS	Target strength	Energy reflected towards the SONAR after hitting a target.
NL	Noise level	Amount of noise at the hydrophone location.
DI	Receiving directivity index	Varies with ray angle.
RL	Reverberation level	Level of complexity of the reverberation model.
RD	Recognition differential	Depends on the background.
AL	Attenuation loss	Energy loss in the propagation.
EI	Gain relation	Obtained through eq. (2.9).

Active SONARs: Reverberation Background

$$SL - 2TL + TS = RL + RD \tag{2.8}$$

SONAR equation

$$EI = SL + DI - 2TL - 2AL + TS \tag{2.9}$$

SONAR equation in terms of power

$$P_{EI}(\alpha) = \frac{P_{SL}.P_{DI}(\alpha).P_{TS}(\alpha)}{P_{TL}^2.P_{AL}^2} \tag{2.10}$$

In Eqs. (2.7 to 2.9), all terms are in dB. Eq. (2.9) is the basis for the ray-trace model. In eqs. (2.7) and (2.8), RD works as a detection threshold (DT).

The parameters defined in table 2.3, relate to basic acoustic models such as (Etter, 1995):

- TL can be estimated through propagation models;
- NL may be determined through noise models;
- RL is estimated through reverberation models.

The dynamic inverse SONAR model presented in Zhou et al. (2016) is a derivation of the SONAR equation, eq. (2.10) in order of the internal beam angle α. The probabilistic values associated with each angle and range, $P(\alpha, R)$, are obtained from a normalization of P_{EI}. This model can be presented as:

$$P_{EI}(\alpha) = P_{DI}(\alpha).P_{TS}(\alpha) \tag{2.11}$$

The internal beam angle, α, corresponds to the horizontal angle between the center of the beam and any ray considered within the SONAR beam horizontal area that travels from the position of the sensor to the outer circle surface where the first reflection was detected, as it is visible in fig. 2.2 (Zhou et al., 2016).

2.4 Simultaneous localization and mapping

SLAM is the process of having the AUV build the map of an environment while localizing itself within that environment. These algorithms are divided into two main categories: The online algorithms, where the current position is estimated alongside the map and the full where the whole trajectory is calculated afterwards.

Additionally, SLAM implementations can be either feature based, where the features are extracted and maintained in the state space together with the pose, or view based, also called pose-based, where the state space is consists of historical vehicle states composed of the vehicle poses at the times the observations were taken (Eustice et al., 2006).

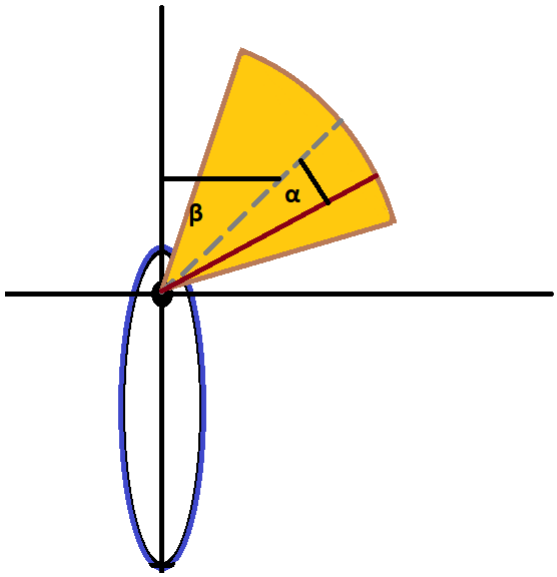


Figure 2.2: Representation of the angles within a SONAR beam. The black and blue ellipse represents the vehicle, the SONAR is represented as a black circle, the SONAR beam in yellow, the center of the beam in a dashed grey line and a ray within the beam is represented in red.

The most common algorithms for SLAM are the EKF, the Fast SLAM and the Graph SLAM. Most underwater SLAM techniques are based on these three algorithms. Table 2.4 contains a description of these algorithms and some others that are worth mentioning.

In Ribas et al. (2006), the data gathered by a MSIS and a DVL is used to correct the motion distortion of the SONAR data and estimate the trajectory using an EKF. Afterwards, the filtered data together with the motion estimation are run through an EKF SLAM algorithm.

He et al. (2009) used a modified Rao-Blackwellized particle filter (RBPF) based SLAM. The RBPF, also called mixture kalman filter, is built upon the idea that some filtering equations are better evaluated analytically than by sampling. The marginalization replaces the finite Monte Carlo sampling with an infinite closed particle set. This allows for a more accurate estimation. To compute the distribution it is necessary to run a KF. The modified version implemented by He et al. (2009), uses the UKF to incorporate the current observations with the ones from the previous states, instead of using the usual KF. This allowed for a more accurate estimation of the location of the vehicle and environment features.

In He et al. (2012) a modified Fast SLAM algorithm is used. Two groups of sensors were implemented in this case: The internal that include digital compass, gyro, attitude and heading reference system (AHRS), and pressure sensor. The external which include MSIS, DVL, altimeter, CCD camera and GPS. The MSIS was used for mapping while the other sensors were used for estimating the position. The difference of this SLAM technique to the traditional Fast SLAM it that it introduces a method that combines single particle maximum likelihood with negative evidence and ranks the particles during the re-sampling phase to overcome the problem of particle depletion (He et al., 2012).

Table 2.4: Common SLAM methods based on Hidalgo and Bräunl (2015a), Paull et al. (2014).

SLAM method	Description
EKF SLAM	Extended Kalman Filter SLAM, applies a recursive predict-update cycle to estimate the position and the map. It is applicable in both view based and feature based SLAM. Works well with well defined features. However, new features greatly increase processing time.
SEIF SLAM	Sparse Extended Information Filter maintains the information matrix which maintains the Gaussian shape. Good for multiple-robot SLAM. Information matrix has to be constantly increasing.
Fast SLAM	It is based on the particle filter. The features and positions appear as particles in the state space. It estimates both the current position and the trajectory of the vehicle. Increasing the number of features does not increase the running time by a lot and does not depend on the parameters of the motion models. The ability to close loops depends on the set of particles.
Graph SLAM	It works through the Taylor expansion. This technique builds a simplified estimation that replaces raw sensor measurements with edges that each represents a nonlinear constrain. Requires more computation than most of the other methods.
AI SLAM	This method is based on neural networks. It is an efficient method but requires training or parameter tuning.

The MSISpIC previously mentioned in section 2.1.4 is extended in Mallios et al. (2010) to a pose based SLAM framework. In this study, the SLAM algorithm uses a Augmented State EKF to estimate the vehicle's pose.

For the most part, SLAM algorithm approaches are based on the use of MSIS, Forward or side looking SONARs and video cameras. For the pose estimation, the common implementations are using DVLs and IMUs.

2.5 Related work

This work aims to be a contribution to this area of research. Therefore it is necessary to identify related studies with similar objectives. In the results section of this study, a comparison between this proposition and the studies highlighted in this section will be carried out. This section will consider studies that had as objective, a proposition for solving localization and mapping.

The most common proposition considers the EKF SLAM as a viable option (Cao et al., 2016; W. Chen and Sun, 2018; Choi et al., 2016; Soylu et al., 2018). However, most of these options

use the DVL which, as mentioned previously, is a big and heavy device, not suitable for small and lightweight vehicles. Q. Zhang et al. (2018) proposes the use of an UKF based SLAM to integrate several sensors, that showed a better performance than the EKF SLAM. However, besides also using a DVL, it also relies on GPS, which means that it can't stay underwater for long periods of time and in order to withdraw the orientation, it uses a compass that may struggle within structured environments.

Some studies, such as Vidal et al. (2018), Zhou et al. (2016), aim to improve maps through means of techniques generally used in mobile robotics. In Zhou et al. (2016) a dynamic inverse-SONAR model is utilized to compensate the uncertainty in the direction of the echo received by a MSIS. This study uses a GPS and DR to estimate the vehicle's pose. The correction for the position was made by back propagating the error between the dead-reckoned surfacing position and the GPS position. This limited the duration of the mission. Furthermore, the data from the SONAR was not consistent and would sometimes return noisy, making it unusable. The implementations suggested by Zhou et al. (2016), L. Li et al. (2014) and Kwon et al. (2017) propose mapping techniques using SONARS where the wide beam produced by the sensor is explored.

Chapter 3

Map building and position estimation

The goal of this study is to develop an autonomous map building method to be implemented in small hovering AUVs. In order to test the efficacy of this method, it will be implemented on the SHAD, presented by Gonçalves et al., 2016. The size and weight of this type of vehicle limit the sensor options because big, heavy or high power consumption sensors are not a viable option. Therefore, a MSIS was selected to integrate the system. A localization solution has been addressed by Fula et al., 2018 and integrated in this vehicle. However, this required a prior map which is not the goal of this study. Instead, in this study the solution is based on the system overview presented in fig. 3.1.

This block diagram consists of the main blocks that compose this method. Each time the MSIS concludes a full revolution, the data is processed by a filter, composed by the preprocessing and range extraction blocks, that reads the arrays of intensities passed by the sensor and takes the range to the first feature for each scanned angle. The output of this filter is fed into a particle filter. The particles that were previously create are moved based on the predicted vehicle motion. The weight of each particle is updated based on the difference between the distance to the outer map boundaries and the sensor observation, for each scanning angle. Then, based on these weights, the position and velocities are estimated and finally the particles are redistribute based on their weights. When the position is estimated, the difference between the new estimation and the previous state are calculated. If this difference is bigger than the defined movement thresholds for the mapping algorithm, the question "*Mapping Necessary?*" is answered affirmatively and the map is updated through the mapping portion of the algorithm. On the other hand, if the vehicle movement was less than the thresholds, there is no mapping and the map remains the same.

The vehicle considered in this study is the SHAD AUV, described by Gonçalves et al., 2016. It is a torpedo shaped, micro-sized AUV with 120mm of diameter and 1,10m long. It weighs 9kg and it has a customizable payload. The vehicle has four degrees of freedom with its current thrusters configuration. This vehicle is equipped with a Tritech Micron SONAR, mounted parallel to the floor in order to obtain information regarding the vehicle's surroundings rather than the seafloor, since it will be used for localization and mapping of the involving environment.

It is also important to understand that the SONAR beam is described as a pyramidal shaped

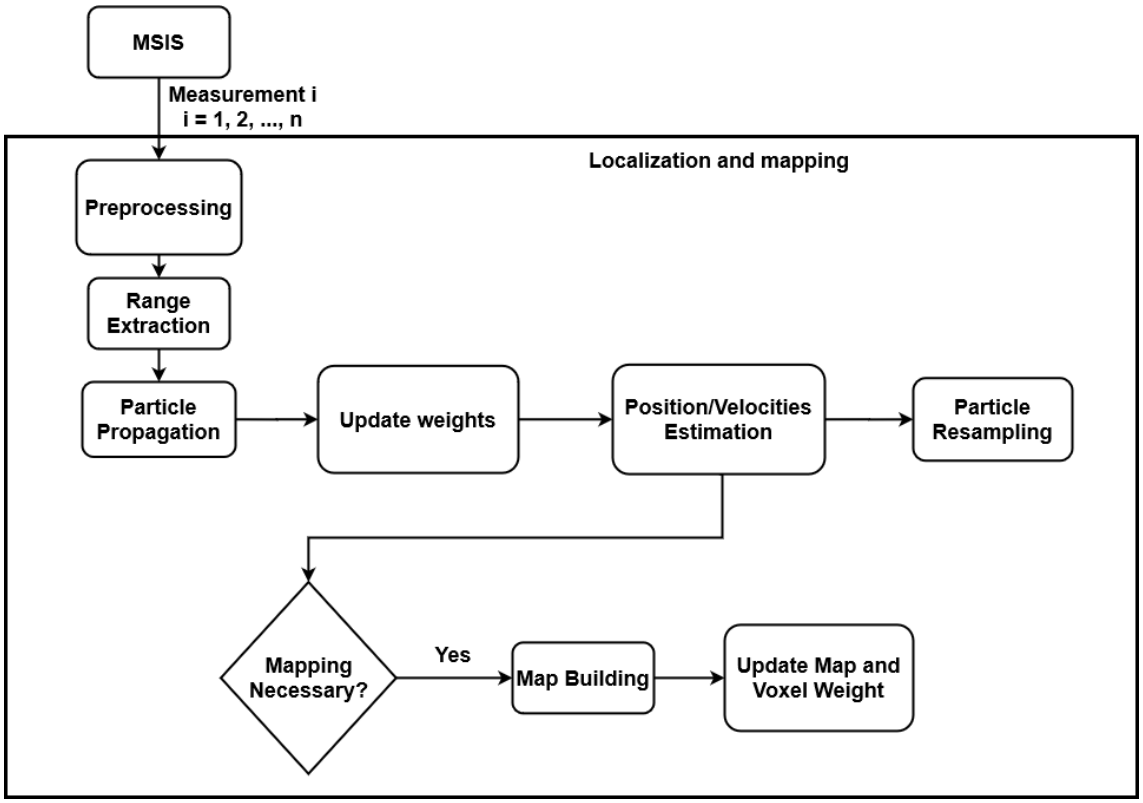


Figure 3.1: A block diagram of the system overview with the main blocks that will take part in the project

beam instead of a laser pointer. In the horizontal configuration, for each beam, the SONAR can detect 3 degrees horizontally and 35 degrees vertically. Even though the SONAR beam covers this area, the returned array does not specify exactly where the beam reflection happened but only that it happened at a certain distance. Therefore, in this study, a probabilistic model is created in order to map the surrounding environment. Furthermore, as these readings are affected by multiple interferences, a simple filter is used to make the information usable to the system, (Fula et al., 2018).

The several parts of this method are described in each section. In this study a component of the probabilistic maps is explored and defined as follows:

- Certainty is a variable that defines how sure the system is of the state of a cell. A low value reflects the lack of confidence from the system on the state of the cell. A high value shows the system trusts the defined state on the cell. The variance relates to the probability and informs on how much the probability of a cell changes. It is not possible to be sure of the state of a cell if the probability is changing quickly between empty and full. Therefore, the variance ends up reflecting in the certainty of a cell. Furthermore, the number of times a cell was detected with a low variance is also saved and it reflects on the certainty.

The project development of this study separated the final section, SLAM, into two subsections, mapping and localization. These two were developed and tested almost independently. After both

subsections had been developed and tested, they were joined together into the SLAM algorithm which allowed both algorithms to be working simultaneously and limit how often the map weights are updated. For this subsections to be developed independently and for the SLAM algorithm to function properly, a few assumptions are necessary:

- The initial position is considered to be the Origin.
- The Vehicle doesn't move for a few seconds.
- The speed of sound in water is 1500 m/s.
- The velocities of the vehicle are low enough that the images taken from the SONAR are not deformed and the information is usable.

3.1 Mapping

The mapping representation used to visualize the produced maps in this study is the Octomaps. The BuildOctree Matlab function created by David, 2020 was used to build the Octree that was later reproduced in Octomaps. Furthermore, in order to know the state of occupation of a certain voxel, a probability of occupancy was created. The probability of a voxel is the chance of that voxel containing an object. This is:

- If the probability is zero, the voxel is empty.
- If the probability is one, the voxel is occupied.
- If the probability is 0.5, the occupation of the voxel is unknown.

Taking into account the shape of the SONAR beam, after acoustic array pre-processing, it is assumed that the first reflection found corresponds to the closes object in the insonified volume. In other words, the space between the vehicle and the first detection is assumed to be free. However, the SONAR may sense some noise before hitting an object, therefore the first reflection may come from something other than an object. Such noise is usually not detected in consecutive measures and therefore, with a dynamic algorithm, this type of noise may be suppressed.

3.1.1 Range to first feature

The feature extractor used in this study, follows the one used in Fula et al., 2018. The data returned by the SONAR consists on arrays of intensity values represented by 8-bit integers. Each of these arrays corresponds to a certain scanning angle.

$$I_i = \{I_{i1}, I_{i2}, ..., I_{im}\} \tag{3.1}$$

where m is the total number of bins in the array. This number is configured in the SONAR. Furthermore, the maximum range of each SONAR beam and the distance between each consecutive value are known. The latter remains constant for each array.

Underwater acoustic signals are affected by a number of interferences, as stated earlier in this chapter. This algorithm aims to extract the distance between the SONAR and the first object contained within the volume covered by the SONAR beam. The algorithm is applied to each array like so:

1) Ignore the first few measurements with a corresponding distance smaller than d_0:

$$I_i' = \{I_{i1}, I_{i2}, ..., I_{ij}..., I_{im}\}, j.d_i > d_0 \tag{3.2}$$

This step is intended to discard the reflections on the vehicle's own body and avoid late readings from reflections from the previous echo signal.

2) Apply an intensity threshold T_i to I_i':

$$I_i'' = \{I_{i1}, I_{i2}, ..., I_{ij}, ..., I_{im}\}, \tag{3.3}$$

with $I_{ij} = 0$ if $I_{ij} < T$

3) Apply an edge detector

$$E = \{e_{i1}, e_{i2}, ..., e_{ij}, ..., e_{im}\}, \tag{3.4}$$

with $e_{ij} = I_{i(j+1)} - I_{ij}$

4) Apply a differential threshold T_d and select the smallest j that satisfied the condition, $e_{ij} > T_d$. The output of this algorithm is a structure that contains:

- The angle associated with each array of bins, this is, $\beta = i * step\_angle$, where i is the array number;
- The distance to the first reflection for each array, d;
- The intensity of that reflection, I_i;
- All the other intensities set to zero.

This algorithm discards most noise sources in the SONAR data and extracts unoccupied distances between the vehicle and the surrounding environment. Nevertheless, the post processed data is not fully accurate and feeding this information directly into the spatial representation would generate errors. In this sense, the output of this algorithm goes through a SONAR model and a probabilistic update of the voxels of the Octree in order to reduce the effect of noise that is not discarded by this algorithm.

3.1.2 Probability attribution

The general SONAR model considers only the central line of the SONAR beam and that all reflections must occur on that line Zhou et al., 2016. In reality, the reflections may occur anywhere within the beam and considering only the central line may result in a map that barely resembles the real map.

In this study, a three dimensional adaptation of the dynamic inverse SONAR model described in Zhou et al., 2016 is suggested. This representation is based on the ray-tracing SONAR model Etter, 1995. Instead of considering solely the central line, these studies consider that the reflections may happen anywhere in the central horizontal plane of the beam. Therefore, several rays starting from the transducer until the detection range are simulated. In this project, this consideration is expanded to the third dimension and it is considered that the reflections may occur at any point within the volume created by the SONAR beam. The number of rays simulated to cover the area is arbitrary for the horizontal and vertical planes. A single horizontal and vertical ray would only cover the central beam. If the number of horizontal rays would be three, this would cover the edges of the horizontal plane of the beam and the central ray. This can be expanded to cover most of the volume but that would drastically increase the operating time.

When differentiating P_{EI}, in eq. (2.10) in terms of the internal beam angle, α, the only terms remaining are P_{DI} and P_{TS}. P_{DI} can be obtained through eq. (3.5) and P_{TS} through eq. (3.9) Zhou et al., 2016.

$$P_{DI}(\alpha) = \frac{sin(\frac{kh}{2}.sin(\alpha))}{\frac{kh}{2}.sin(\alpha)} \tag{3.5}$$

where k is the wave number obtained by solving eq. (3.6) and h is the length of the transducer obtained from the analysis of the SONAR beam pattern and solving eq. (3.7) in order of h of Naval Personnel, 1953.

$$k = \frac{2\pi}{\lambda} \tag{3.6}$$

$$\gamma = sin^{-1}0.258\frac{\lambda}{h} \tag{3.7}$$

where λ is obtained from:

$$\lambda = \frac{v}{f} \tag{3.8}$$

in which v represents the speed of sound underwater and f the frequency of the SONAR. The horizontal length of the transducer is different from the vertical length. Furthermore, the eq. (3.5) remains the same for the vertical expansion, but instead of depending of α, it is a function of the elevation θ. This is one of the innovations introduced by this project.

The circular angle, γ, is found by subtracting the angles of the intersections between the beam pattern and the -3dB curve. At different power values in the beam pattern, the above mentioned formula is changed, as stated in of Naval Personnel, 1953. For this power value, the vertical and horizontal circular angles are 35º and 3º respectively.

$$PTS(\alpha) = \mu.sin^2\theta_i(\alpha) \tag{3.9}$$

where μ is the scattering coefficient obtained through experiments and θ_i is the incident angle, this is, the angle between the SONAR beam angle and the incident wall, which is obtained by solving:

$$\theta_i(\alpha) = 90 - (\beta + \alpha + \theta_s^v) \tag{3.10}$$

θ_s^v is the angle between a wall and the vehicle's axis. β and α are visible in fig. 2.2. When considering the full SONAR beam volume, introduced in this study, eq. (3.9) stops being a function of α and instead becomes a function of ϕ which is defined as the 3D angle between the line that travels across the center of the beam and any ray contained within the boundaries of the beam, given by:

$$\phi = cos^{-1}\left(\frac{v1.v2}{\|v1\|\|v2\|}\right) \tag{3.11}$$

where $v1$ and $v2$ correspond to the central and any other ray within the SONAR beam.

In practical terms, for this project, P_{TS} is withdrawn from the intensity values of the bins returned from the MSIS, since these values depend on the power of the TS.

In order to be used in a probabilistic manner, the value for the power $P_{EI}(\alpha,\theta,\phi)$ described in Eq. 2.11 has to run through a normalization which results in the probability $P(n|z_t)$:

$$P(n|z_t) = \frac{P_{EI}(\alpha,\theta,\phi) - min\{P_{EI}(\alpha,\theta,\phi)\}}{2(max\{P_{EI}(\alpha,\theta,\phi)\} - min\{P_{EI}(\alpha,\theta,\phi)\})} + 0.5 \tag{3.12}$$

where n represents the number of the iteration and is a product of β, α and θ obtained through:

$$n = i + \frac{\alpha}{step_h} + \frac{\theta}{step_v} \tag{3.13}$$

where $step_h$ represents the step angle between two consecutive angles within the horizontal part of the beam, $step_v$ the same in the vertical part of the SONAR volume and i the number associated with the scanning angle.

3.1.3 Probability update

The probability attribution follows the approach of Hornung et al., 2013 where the probability of a cell is given by:

$$P(n|z_{1:t}) = \left[1 + \frac{1-P(n|z_t)}{P(n|z_t)}\frac{1-P(n|z_{1:t-1})}{P(n|z_{1:t-1})}\frac{P(n)}{1-P(n)}\right]^{-1} \tag{3.14}$$

which can be rewritten as:

$$L(n|z_{1:t}) = L(n|z_{1:t-1}) + L(n|z_t) \tag{3.15}$$

where

$$L(n) = log(\frac{P(n)}{1-P(n)}) \tag{3.16}$$

and $P(n)$ is the probability of a cell being occupied. Furthermore, using the logarithm of the probability speeds the calculation of the probabilities for each step.

When mapping environments using Octrees, it is usual to apply a threshold to define when a voxel can be defined as occupied. The voxel is considered free if it has not reached that value. In order to change the state of a voxel, due to eq. 3.15, there have to be as many observations in a different state as there were to first define it. This has advantages, such as, easily discarding false positives and ignore moving objects that should not be included in the map. However, this would not allow for a very dynamic environment, this is, if a certain voxel is defined as free for a long time but as the vehicle approaches it, it starts observing it as occupied, it would take a long time to correct this error. This can be controlled by applying a maximum and minimum values that $L(n)$ can take. By doing this, the same voxel no longer has to be observed as occupied, the same amount of times it was seen as free. Therefore, the update cycle can instead be defined as Hornung et al., 2013:

$$L(n|z_{1:t}) = max(min(L(n|z_{1:t-1}) + L(n|z_t), l_{max}), l_{min}) \tag{3.17}$$

where *lmin* and *lmax* define the lower and the upper limits that the logarithm of the probability can take. Because of this limitation, the amount of time it takes to change the state of a voxel is restrained by the values chosen for the lower and upper limits Hornung et al., 2013. The main problem of this feature is that passing object may end up being included in the map instead of being discarded.

The final maps produced by this algorithm, have the objective of being processed by the system to be used for navigation. Therefore this study considers if two or more readings are positioned within a voxel, that voxel takes the probability of the highest probability value of all those bins. In this way, if all points have a low occupational probability, the state of the voxel may be considered as free. Furthermore, if all but one reading, within a voxel, have low probability but one bin has a high occupancy probability, the voxel must be signaled as occupied so that the vehicle will not travel to that position.

3.2 Localization

Once a good map estimate is obtained, it is necessary to know where the vehicle is positioned within that map at all times in order to navigate and to further enhance the current map. The knowledge on the vehicle's position allows the AUV to position itself in different locations that have other perspectives of the environment which can be used to identify zones that may not have been visible from previous positions. Combining this and the previous map algorithm allows the vehicle to navigate freely and build an accurate map without human intervention. On the other hand, if the system is not running any localization algorithm, as soon as the vehicle moves from its starting position, the map would become extensively filled with uncertainty. The localization algorithm impedes this from happening as the map remains, for the most part, consistent.

During the development of this section, an initial map taken from one iteration of the mapping algorithm was necessary in order to develop the localization in parallel with the mapping.

The localization algorithm that is considered in this study is the PF. This filter has the advantage of being easily applied alongside an Octomaps implementation as well as providing a good estimate of the position of the vehicle without the need of several iterations. This algorithm has some limitations however. The processing time increases with the number of particles, which forces the number of particles to be limited if reduced operation time is essential.

The PF is an iterative algorithm that continuously runs through a set of 6 steps to produce an estimation of the position and orientation. The steps for the particle filter are present in table 3.1.

The first step consists on the creation and uniform distribution of all the initial particles the system will consider. This is done by spreading random uniformly distributed particles in a sphere around the initial position. These particles are composed of the position and orientation, $x_t = x_t, y_t, z_t$ and θ_t, respectively, and the weight, ω_t. The weight of each particle is initially given by $\frac{1}{N}$ where N is the total number of particles. This is the only step that runs only on the first iteration of the filter. The pseudocode for this function can be seen in table 3.2.

The following point is the prediction. It consists on moving the particles according to the predicted motion of the vehicle. In this study the predicted movement is taken by dividing the position variation by the time difference since the last iteration of the filter:

$$\widetilde{V}_{t-1} = \frac{\Delta X_{t-1}}{\Delta t_{t-1}} \tag{3.18}$$

where $\widetilde{V}_{t-1}$ is the estimated velocity from the previous step. This velocity is then multiplied by the time it has passed since the last iteration to obtain $\Delta\widetilde{X}_t$

$$\Delta\widetilde{X}_t = \widetilde{V}_{t-1}\Delta t_t \tag{3.19}$$

To finish, this value is then added to the position of each particle. The pseudocode for the prediction step is visible in table 3.3. The arguments for this function are the particle positions

Table 3.1: Set of steps that compose the particle filter algorithm L. Chen et al., 2013; Kozierski et al., 2013; T. Li et al., 2015; Paull et al., 2014; Welch and Bishop, 1995.

$Algorithm[x_{mean}] = Particle\_Filter(x_{t-1}, \theta_{t-1}, \theta_{max}, R, z_t, u_t, N)$
1. $[x_t, \theta_t, q_t] = Initialize\_particles(x_{t-1}, \theta_{t-1}, \theta_{max}, R, N)$;
2. $while(True)$
3. $\quad[x_t] = Predict\_movement(x_t, u_t)$;
4. $\quad[q_t] = Update\_weights(x_t, z_t)$;
5. $\quad[q_t] = Normalize\_weights(q_t)$;
6. $\quad[x_{mean}] = Weighted\_mean(x_t, q_t)$;
7. $\quad[x_{t+1}, q_{t+1}] = Systematic\_resampling(x_t, q_t, N)$;
8. $\quad t = t + 1$;
9. *end*

Table 3.2: Pseudocode for the *initialize particles* algorithm.

$Algorithm[x_t, \theta_t, q_t] = Initialize\_particles(x_{t-1}, \theta_{t-1}, \theta_{max}, R, N)$;
1. $rvals = random[-1,1]_{Nx1}$;
2. $elevation = sin^{-1}(rvals)$;
3. $azimuth = random[0, 2\pi]_{Nx1}$;
4. $radius = random[0, R]_{Nx1}$;
5. $\phi = random[0, \theta_{max}]_{Nx1}$;
6. $\theta_t = \theta_{t-1} + \phi$;
7. $[x_t] = sph2cart(azimuth, elevation, radius)$;
8. $\omega_t = \frac{1}{N}$

and the vehicle's velocities are surge, sway, pitch and yaw, $u_t = [\dot{x}, \dot{y}, \dot{z}, \dot{\psi}]$. The function produces the translated position of the particle set.

The next step is the update. Each particle has an associated weight that quantifies how well that particle fits with the measurement model. As stated previously, in the first step of this algorithm, the weights are set to $\frac{1}{N}$. In this state, they are updated based on the PDF centered on the sensor measurements.

The pseudocode for the update phase is presented in table 3.4. For each particle, the scan matching between the predicted scans and the observed scans is done by measuring the distance from the particle, for each angle β, to the outer boundaries of the map and comparing that space with the interval to the first feature, observed for the same angle. By comparing these two distances, the particles that are closer to the real pose of the vehicle will have higher probability.

The observation may happen anywhere within the SONAR beam volume, therefore, for each angle, β, the SONAR beam is separated into several rays, each associated with a certain α and θ. To measure the predicted distance from the particles to the outer boundaries of the map, an algorithm that tests every intersection of a ray with the faces of the cubes that represent the voxels of the Octomaps is ran. This algorithm returns the distance between the ray and the face of each cube that it intersects. The largest distance is then selected for each ray, since it is associated with the furthest intersection of the ray with the voxel cube faces, which relates to the outer map boundaries.

The observed distance is equivalent to the smallest distance to an object that is placed inside the SONAR beam. Therefore, the smallest distance of all the rays within a certain angle, β is selected and that distance is equivalent to the predicted distance from the particle to the outer boundaries of the map.

The comparison between the predicted and the observed distances is made through a PDF centered on the observed distance. The standard deviation works as an acceptance threshold. The

Table 3.3: Pseudocode for the *predict movement* algorithm.

$Algorithm[x_t] = Predict\_movement(x_t, u_t)$;
1. $x_t = x_t + u_t \Delta t$

value taken from the PDF is then multiplied by the previous particle weight to obtain the new particle weigh as it is shown in eq. 3.20. This is done for every particle until all weights have been updated. Fig. 3.2, contains a graphical representation of this comparison being done for a particle. The limits of the colored area represent the spread and observed range for the angle β. It is visible that for that angle, the interval between the particle and the intersection with the outer limits is smaller than the observed distance and therefore, the weight of this particle would be affected negatively by this observation.

$$\omega_t = \frac{1}{N}\frac{1}{\sigma\sqrt{2\pi}}\exp^{\frac{-(x-\mu_s)}{2\sigma^2}} \tag{3.20}$$

where x is the predicted distance from the particle to the edge of the map for one scanning angle. μ_s is the distance associated with the observation model and σ is an acceptance threshold.

However, the weights in the end of the update step are quite low and the sum of all particle weights can be anything. In order to again understand the meaning of each weight, a normalization is necessary to bring the sum back to one. The normalization step consists of a simple equation to

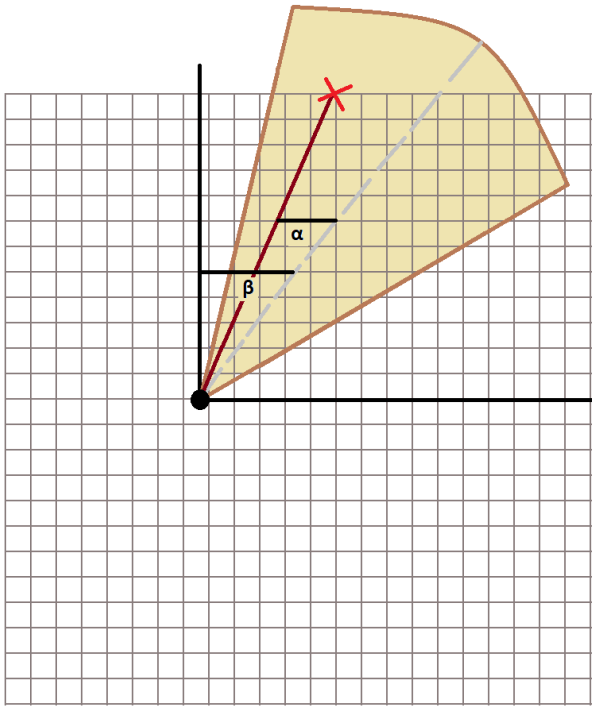


Figure 3.2: The graphical representation of the distance between a particle and the outer boundaries of the map for a ray with angle α, within the SONAR beam with angle β. The map is represented as the grid, the particle as the black dot and the intersection with the outer map limits by a red cross. The length of the grey dotted line represents the observed distance for the angle β.

Table 3.4: Pseudocode for the *update* step from the particle filter.

$Algorithm[q_t] = Update\_weights(x_t, z_t)$;
1. $angle = 0, \ldots, 2\pi$
2. $for\ i = 1, \ldots, angle\_size$
3. $\quad [y_i] = seperate\_in\_rays(x_t, angle_i)$
4. $\quad [d_i] = distance\_to\_map\_limits(x_t, y_i)$
5. $\quad y = min(y_i)$
6. $\quad q = q * Normpdf(z_t, d_t, \sigma)$
7. *end*

normalize the values of the weights.

$$\omega_i = \frac{\omega_i}{\sum_{a=1}^{N} \omega_a} \tag{3.21}$$

After having updated the weights for each particle, the position can be estimated. The end of the iteration step is where this is calculated. The pose is estimated by doing the weighted mean of all particles.

$$\widetilde{X}_t = \frac{\sum_{i=1}^{N} (X_i . \omega_i)}{\sum_{i=1}^{N} (\omega_i)} \tag{3.22}$$

where $\sum_{i=1}^{N} (\omega_i)$ is equal to one.

The last step is the resampling process. To choose which algorithm to use, the complexity and computation necessary to run each method was analysed. The systematic resampling method was selected for its linear complexity, for being computationally light and for keeping a constant sample size. It was also selected over stratified resampling for only needing one random value instead of N random values.

The code used for this study is based on those proposed by Kozierski et al., 2013; T. Li et al., 2015. The algorithm is presented in table 3.7. The systematic resampling function takes as arguments, the number of particles n, the pre-resampling particle set x_t and its weights q_t and produces a new particle set x_{t+1} with its associated weights q_{t+1}.

Systematic resampling Carpenter et al., 1999; Kitagawa, 1996; T. Li et al., 2015 divides the particle's sets in smaller subsets. The first random number $u_t^{(1)}$ is placed in the interval $[0, \frac{1}{N}]$ and the other numbers are obtained by:

$$u_t^{(n)} = u_t^{(1)} + \frac{n-1}{N}, n = 2, 3, \cdots, N. \tag{3.23}$$

Table 3.5: Pseudocode for the *normalization* step.

$Algorithm[q_t] = Normalize\_weights(q_t)$;
1. $q_t = q_t / sum(q)$

Table 3.6: Pseudocode for the *end of the particle filter iteration* where the position estimation is produced.

$Algorithm[x_{mean}] = Weighted\_mean(x_t, q_t)$;
1. $sum(q * x)/sum(q)$

The upper and lower limits for the number of times a particle is resampled is given by $|N\omega_t^{(m)}|+1$ and $|N\omega_t^{(m)}|$ respectively.

As the newly generated particle set is composed of replications of some of the particles from the older set, that had the higher weights, a particle scattering step is necessary in order not to lose information but continue to track the position of the vehicle. Therefore, the system returns to the movement step where it will begin the new iteration of the filter. This cycle will continue and the position will be known for as long as the vehicle is online.

3.3 Simultaneous localization and mapping

After having defined the mapping and the localization algorithms, both are integrated in the system together. In this section, the logic behind the function of the SLAM algorithm that integrates the two previous methods is explained. The SLAM method introduced is a Fast SLAM variation. The number of arrays within a SONAR revolution, is defined by the step angle. This number ranges from $[1, 2\pi/\nu]$ where ν is the step angle between two consecutive scans. Reducing the step angle means an increase in i which reflects a higher but also an increase in operating time.

The SLAM part of the algorithm is described by having the localization constantly running to always have an estimation of the vehicle's position. The mapping part of the algorithm runs whenever the distance traveled or angle rotated has reached a certain threshold. This is done for several reasons. Firstly, because if the pose estimation has not yet reached a correct estimation, constant mapping may results in a deformity in the map. Secondly, to avoid mapping a moving object. Mapping from the same position multiple times, generally, does not increase the current knowledge about the map but affects the operation time.

Table 3.7: Pseudocode for the *systematic resampling* algorithm based in Kozierski et al., 2013; T. Li et al., 2015.

$Algorithm[x_{t+1}, q_{t+1}] = Systematic\_resampling(x_t, q_t, N)$;
1. $j = 1; sumQ = q_j$;
2. $u = random[0, \frac{1}{N}]$;
3. $i = 1, \ldots, N$
4. $\quad while(sumQ < u)$
5. $\quad\quad j = j + 1$;
6. $\quad\quad sumQ = sumQ + q_j$;
7. $\quad end$
8. $\quad xx_i = x_j$;
9. $\quad u = u + \frac{1}{N}$;
10. *end*
11. $q_{t+1} = ones(N, 1) * \frac{1}{N}$;

Chapter 4

Experimental setup and results

The testing phase was performed in a closed water tank present in the facilities of INESC TEC, represented in fig. 4.1. The tank is the shape of a rectangular prism, with dimensions 4.40 by 4.58 meters with 1.8 meters depth. It is mostly made of the same material so there are not many variations in the intensity of the reflections due to material differences. However, as it is visible on the right side of fig. 4.9a, there are two windows that don't reflect the signal with the same strength as the rest of the tank that may generate some noise in those points, as it is detected in 4.9b.

4.1 Setup description

Several tests were performed by deploying the SHAD AUV, visible in fig. 4.2 in the testing site. The vehicle was equipped with a Tritech Micron SONAR mounted in the front of it, in a horizontal position, that allowed the SONAR to scan the surroundings of the vehicle. Furthermore, the SONAR was set to the following configuration:

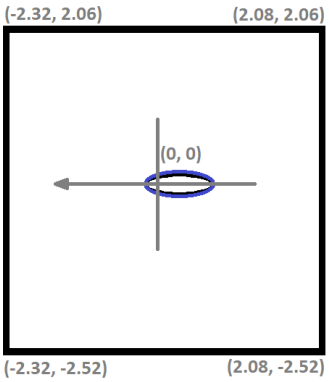


Figure 4.1: Water tank where the tests were carried out, with the location of every corner and a representation of the vehicle. The positions of the corners are relative to the static tests from the second to the fourth test.

- Number of bins - The amount of intensity measures per angle, set to 400;
- Maximum signal range - Set to 10 meters;
- Underwater sound speed - Was considered constant and assumed to be 1500 m/s;
- Scanning limits - The SONAR was set to continuous scanning rotation.
- Step angle size - This is the angle difference between two consecutive angles and was set to 1.8º;
- Gain - The gain of the transducer is the value that affects the intensity readings and was set to 0.5 after experimentation with several values;

This setup results in a full scan revolution taking 8.51 seconds to complete. Furthermore, in order to attribute the probabilities to the voxels, it is necessary to obtain the values for the wavenumber, k and the length of the transducer h. By using eq. 3.8, with $v = 1500$m/s and $f = 700k$Hz, the value for the wavelength is obtained, $\lambda = 2.14 * 10^{-}3$. Subsequently, using eq. 3.6, the value for the wavenumber is obtained, $k = 3.38 * 10^{3}$. By analysing the intersection of the SONAR beam horizontal and vertical patterns with the -3db curve, the values for the circular angles are obtained. Moreover, these are the values for the beam spread mentioned in the product datasheet of the Tritech Micron SONAR International, 2019: $\gamma_{horizontal}$ = 3º and $\gamma_{vertical}$ = 35º. The length of the transducer is obtained by solving eq. 3.7 in order of h, which becomes:

$$h = 0.258 \frac{\lambda}{sin(\gamma)} \tag{4.1}$$

From this equation, the vertical and horizontal values for the transducer length are obtained: $h_{vertical} = 9.2 * 10^{-3}$ and $h_{horizontal} = 8.35 * 10^{-4}$ meters.

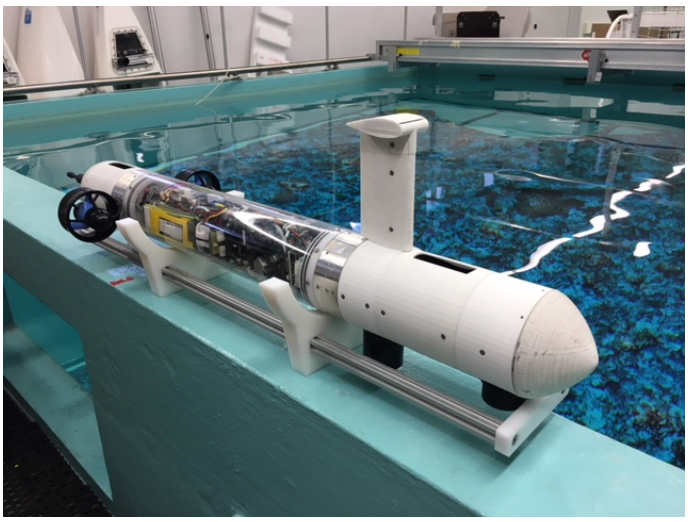


Figure 4.2: The SHAD AUV used for the testing of the algorithm in this study Fula et al., 2018.

The number of considered rays within a SONAR beam can be defined, as stated in section 3.1.2. Fig. 4.3 contains a graphical representation of the separation of the SONAR beam into three vertical rays. The number of horizontal rays for representation purposes was set to one. During the testing phase, in order to maintain a relatively low operation time, the number of rays is set to three horizontal by three vertical rays, which results in a total of nine rays within the SONAR beam volume being considered. These are placed in the outer edges of the pyramidal volume and in midpoints between edges.

Furthermore, when creating the Octree, it is necessary to define a size for the edge of the cubes in the Octomaps. After some experiments with several values, the value of 0.2 meters was selected. Moreover, in order to decrease operation time, the amount of scanning angles used to extract the localization of the vehicle was reduced in ten times. The resolution dropped from 1.8º to 18º which is enough to extract positions with an error below 10 centimeters.

The final algorithm constantly produces a map in the first three iterations of any testing data it runs. After this, it starts using the data from the SONAR to place itself in the tank. When the vehicle has moved more than 15 centimeters in either x, y or z or has rotated more than 30º, the algorithm remaps the surroundings. These values were chosen due to 15 centimeters being the acceptable position error, which means that if the position has moved more than that, it has either moved or there was an error in the map that caused an error in position and the map needs to be remapped. The reason is the same for the angle threshold.

4.2 Data collection experiments

This section describes in detail the several tests performed for the algorithm considered in this study, the conditions of these tests, what the algorithm is expected to generate and the reason for having done each test.

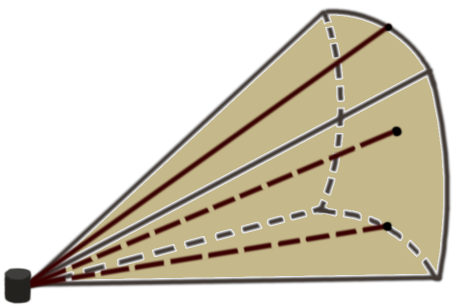


Figure 4.3: Graphical representation of three vertical rays placed within the SONAR beam volume. The rays can be expanded vertically and horizontally.

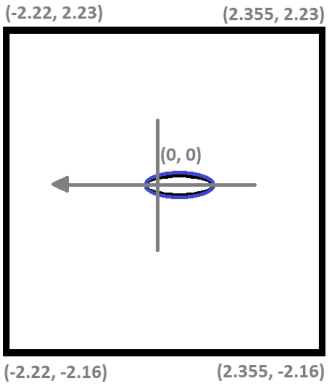


Figure 4.4: The position of the AUV within the water tank and the ground truth positions of the corners of the tank in meters.

- First test:

  The first test of the experimental phase consisted of placing the vehicle centered in the tank, as presented in fig. 4.4. The corners were localized, in meters, in the positions: $(2.355, 2.23)$, $(-2.22, 2.23)$, $(-2.22, -2.16)$ and $(2.355, -2.16)$. During this test, the tank was clear of any objects so that there were no interferences in the recovered data. The purpose of this test was to understand how the algorithm would behave in the simplest conditions. This is, what were the strong and the weak points of the algorithm in a situation where the vehicle was static and there were no sources of noise, outside the common acoustic noises mentioned in chapter 2.

- Second test:

  In the second set of tests, the AUV was positioned in the water tank as shown by fig. 4.1. The corner locations, in meters, were as follows: $(2.08, 2.06)$, $(-2.32, 2.06)$, $(-2.32, -2.52)$ and $(2.08, -2.52)$. In this phase, a surface buoy was placed in the tank in the position $(0.14, -1.78)$ meters. Similarly to test number one, no motion was introduced in the vehicle, nor in the buoy. A graphical representation of this test is seen in fig. 4.5. The goal of this test was to comprehend if the algorithm would still function properly if an object was placed in the tank and what changes would this bring. The size of the buoy allowed the sensor to easily detect it while still being of relatively small size, making it a strong feature in the map.

- Third test:

  The third test introduced movement in the buoy but the vehicle remained static. Since the vehicle did not initiate movement, the position of the origin point, relative to the four corners of the tank remained the same. The initial position of the object was $(1.48, -1.68)$ meters and the final location was $(-0.96, -1.48)$ meters. The vehicle remained static throughout this test. The objective of having movement in the buoy was to acknowledge the behaviour of the algorithm when the map changes. Fig. 4.6 contains an image representing this test

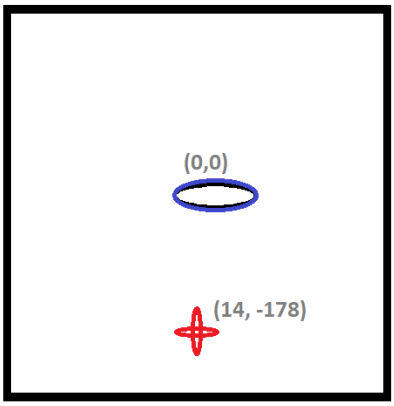


Figure 4.5: Representation of the second test case with the ground truth localization of the vehicle and the beacon (red).

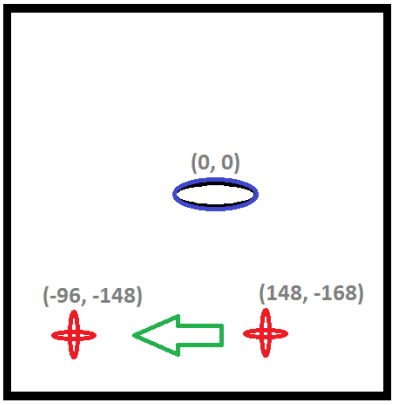


Figure 4.6: Representation of the third test case with the ground truth localization of the vehicle and the initial and final position of the beacon (red).

phase. It was expected that the mapping portion of the algorithm would not be able to add the moving object to the map and that it would struggle to find the correct position of the vehicle since the strong feature present in the map started being a source of noise.

- Fourth test:

  This testing phase added a pole to the body of water. Additionally, the buoy was put in movement, likewise to the previous test. The position of each corner is the same as in test number two. The pole was placed in the position $(-2.01, 0.24)$ meters. The initial position of the buoy was $(1.54, -1.68)$ meters and the final location was $(-0.17, -1.54)$ meters. A visual representation of this test is visible in fig. 4.7. This test adds a new strong feature to the map which should aid the localization portion of the algorithm, therefore, it is likely that this situation brings an improvement in localization relative to test number three. The pole is expected to be visible in the map and the buoy, similarly to test number three, is anticipated to not be part of the map.

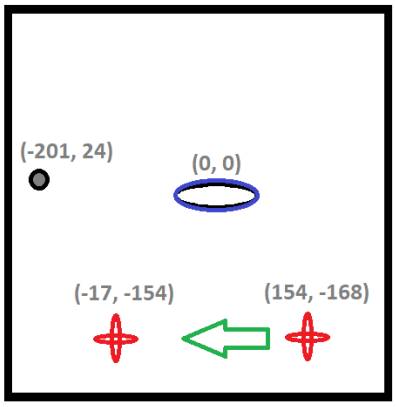


Figure 4.7: Representation of the fourth test case with the ground truth localization of the vehicle, the localization of the small pole (grey with black outline) and the initial and final position of the beacon (red).

- Fifth test:

  The fifth and last test introduced motion in the AUV but kept both objects static to not create further inteference. The location of the corners relative to the origin point remained the same of test number two. The pole was planted in the position $(-2.01, 0.24)$ meters and the buoy in $(-1.17, -1.54)$ meters. A representation of this situation can be visualized in fig. 4.8. The initial localization of the vehicle was the origin point and the final position was $(-1.92, 0.47)$. The results expected to be generated in this test were of a localization estimation with errors of fifteen centimeters and a heading error of twenty degrees.

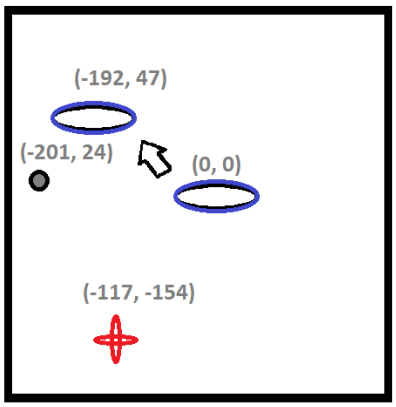
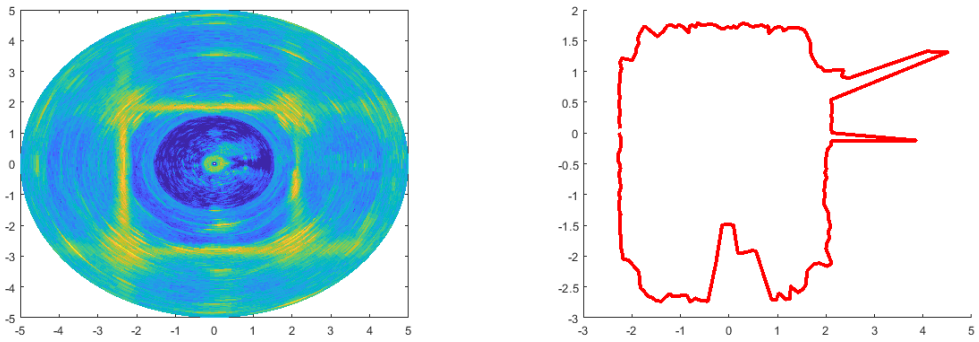


Figure 4.8: Representation of the fifth and last test case with the ground truth localization of the vehicle's initial and final position, the localization of the pole (grey with black outline) and the position of the beacon (red).



(a) Without processing.

(b) After processing.

Figure 4.9: Images drawn from SONAR arrays for the first test case. without any processing (a) and after running the *range to first feature* algorithm (b).

4.3 Performance analysis

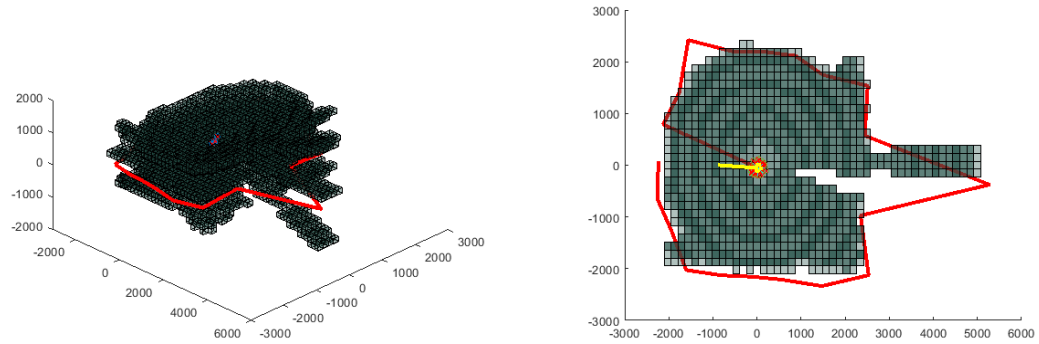
Initially the data received from the SONAR is composed of intensity arrays. The image shown in fig. 4.9a is composed of these arrays.

Even though the shapes in this figure are recognizable to the human eye, it contains a lot of noise and therefore, need a filter layer in order to extract valuable information of it. Multiple reflections, echos, self reflections and multipath are some of the most common error sources for this type of sensor. This is done by the *range to first feature* algorithm, presented in section 3.1.1. The output is an array that contains the intensities of the first feature detected by the beams at each scanning angle, β. The visual representation of this output is seen in fig. 4.9b.

The results of the position errors are obtained from the comparison, for each test, with the origin point. This is done because in all but the last test, there was no movement from the AUV. Therefore, the correct value for every test is the origin. The values for these errors are presented in table 4.1. The minimum error value, $6.24 * 10^{-5}$ meters^2, obtained by multiplying the x and y errors in position, happens in the third test, while the whole system was static, this is, while the buoy had not initiated any movement. It was expected that the fourth test or fifth test would

Table 4.1: Errors in the position and heading after some iterations for each test. Errors are in meters

Test	**Iterations**	$\|Errors\|$ **[x,y,z]**
1	3	[0.0205, 0.0532, 0.2085]
2	7	[0.0407, 0.0457, 0.0532]
3	4	[0.0006, 0.1040, 0.2525]
	21	[0.0179, 0.0511, 0.3448]
4	4	[0.003, 0.0902, 0.1810]
	17	[0.0405, 0.0512, 0.1226]
5	4	[0.0977, 0.0835, 0.2285]
	12	[0.0133, 0.0058, 0.1166]



(a) 3D representation.

(b) 2D representation, seen from above.

Figure 4.10: Image of the occupied region in an octomaps representation with nine rays per SONAR beam after one full SONAR revolution. Axis are in millimeters.

produce better results in the first few iterations since no body in the system had started to move. The z position was not part of this multiplication because the algorithm has bigger redundancy in the z axis than in the other two. This is due to the way the localization is obtained. The error in the z axis is the edge size of an Octomaps cube. Which has been defined in this study as 0.2. Since the vehicle, in any test, had the heave velocity, different from zero, the value for the z measure is zero in every test.

In fig. 4.10, a visual representation of the environment is presented in an Octomaps depiction, together with the sensor observation of the environment. The observation corresponds to the one used for localization, this is, with an 18º step between two consecutive angles. In the first test, the position and heading estimated were $(x, y, z, \psi) = (0.0205, -0.0532, -0.2085, -3.5)$. The sensor observation in fig. 4.10 is centered in the estimated localization and rotated with the heading. In order to understand how the heading affected the map matching, several headings were taken and the distances between the predicted and the observed measures were squared and added together. A graphical representation of this error is visible in fig. 4.11. It is visible in this figure that the error variation between -5º and 3º is quite low. In fact the variation of the square of the error in this region is of 0.0263 meters^2. This creates some redundancy in the heading estimation. The minimum value is set by the heading estimation and is $error^2 = 0.3025$ meters. However, by doing a visual analysis to the same image but with the original amount of scanning angles, it is clear that there's a small error in the heading. This is visible in fig. 4.12.

The pose estimated in the second test, after seven iterations of the algorithm, which corresponded to one minute, was $(x, y, z, \psi) = (0.0407, 0.0457, 0.0532, 6.90)$. The map sum of the square of the errors for this test are presented in fig. 4.13. The minimum error in this test is $error^2 = 2.6616$ meters^2. The low variation area in this test is much lower which reduces the redundancy in the heading estimation. This area goes from 7º to 10º and its variation is of 0.2345 meters^2. By comparing the minimum value of the square of the map error of the second test with the minimum from the first, there is an error ten times bigger. The Octomap representation and the last observation of this test are visible in fig. 4.14a and fig. 4.14b. The other two images in

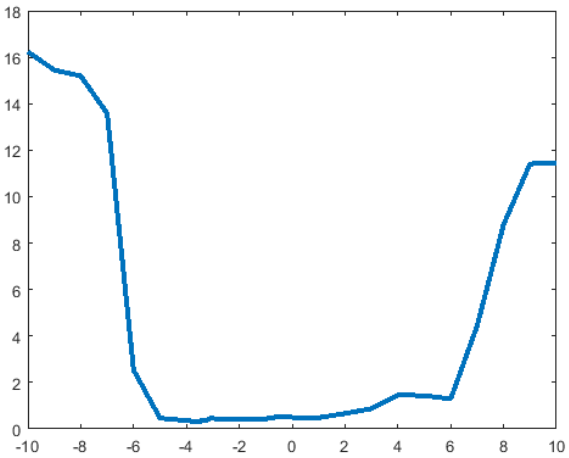
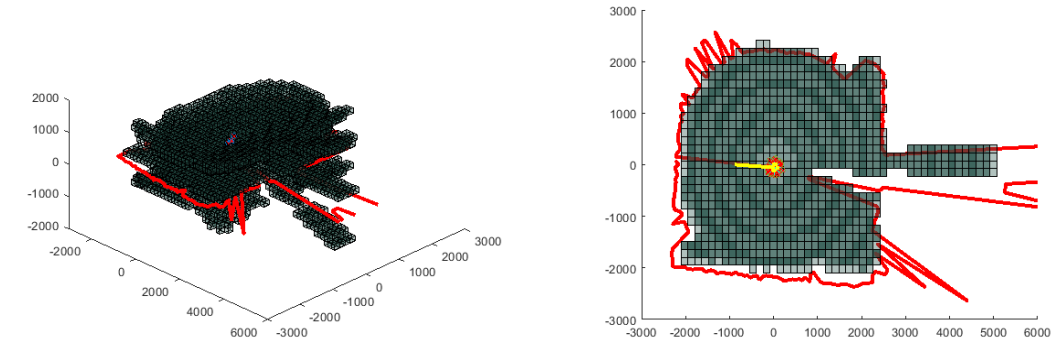


Figure 4.11: The square of the errors in relation to the angle variations for the first test.

that same figure is of a different representation of the Octomaps. Throughout this results chapter, the free voxels are presented. On the other hand, fig. 4.14c and fig. 4.14d represent the occupied regions of the map extracted in the end of the second testing phase.

The third test introduced the movement of the buoy. Initially all objects were static, the position for the AUV was $(x, y, z, \psi) = (0.0006, -0.1040, 0.2525, -5.99)$. When the object finished moving, the error in position was below 10 centimeters in the horizontal plan. However, the z value of the vehicle's position estimation had an error of 34 centimeters. Furthermore, the heading estimation was of -30º. This was the value that reduced the square of the error of the map, as seen in fig. 4.15. This image also shows a comparison of the values from the initial data taken with the static buoy and the ones from the final iteration when the buoy had finished moving. The square of the error raised from 0.5227 meters^2 to 3.2543 meters^2. This is a result of the map deformation caused by the movement of the buoy. The image exposed in fig. 4.16a shows the graphical repre-



(a) 3D representation.

(b) 2D representation, seen from above.

Figure 4.12: Image of the free voxels in an Octomaps representation (blue) with the observation of the third iteration of the algorithm (red), taken in the first test. Axis are in millimeters.

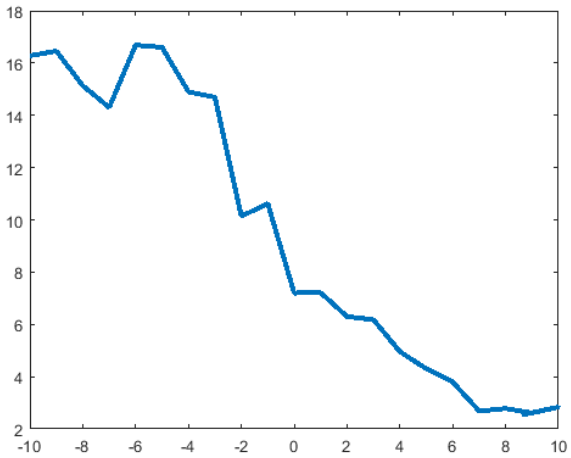
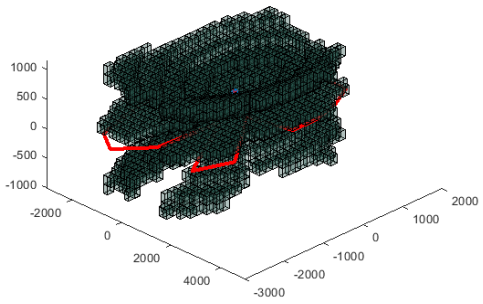
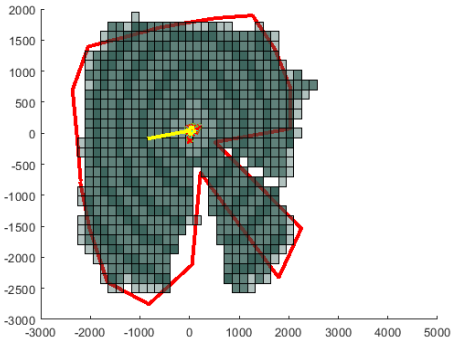


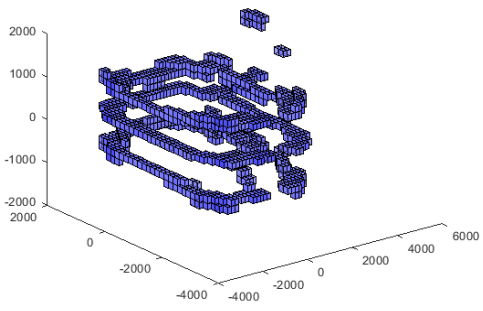
Figure 4.13: The square of the errors in relation to the angle variations for the second test.



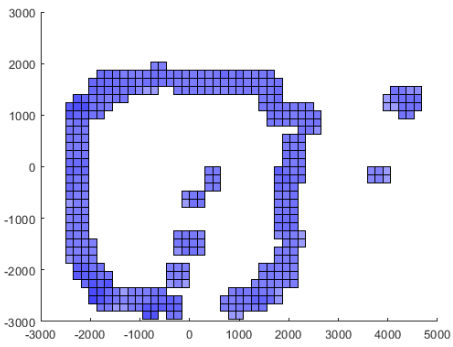
(a) 3D representation of the free cells.



(b) 2D representation of the free cells, seen from above.



(c) 3D representation of the occupied cells.



(d) 2D representation of the occupied cells, seen from above.

Figure 4.14: Image of the free voxels in an Octomaps representation (blue) with the observation of the seventh iteration of the algorithm (red) (a) and (b); Image of the occupied region in an octomaps representation (c) and (d). Both refer to the second test. Axis are in millimeters.

sentation of the free voxels from the Octomaps in blue and the sensor measurements, centered in the vehicle position and heading, in red. Visually, the observation fits well into the map. However, there is an error of around 30º ±5º in heading. The algorithm tried to reduce the error between the observations and the map by rotating the heading which resulted in this error.

The initial position and heading estimation from the fourth test is $(x,\ y,\ z,\ \psi) = (0.003,\ -0.0902,\ 0.1810,\ 0.6990)$. In the final full SONAR revolution, the position of the vehicle was $(x,\ y,\ z,\ \psi) = (-0.0405,\ -0.0512,\ 0.1226,\ 0.6990)$. As it is visible from these values, the addition of a second object that remained static during the operation resulted in a considerable reduction in the heading angle as well as an improvement in the z axis. This is reflected in the graphical representation of the square of the error between the map and the observation, as seen in fig. 4.17. This image contains the error variation for the different angles for the initial and the final situations. The minimum value prior to the movement of the buoy was of 0.2729 meters^2 and after movement, 1.7630 meters^2. The map and the last observation from this test are visible in fig. 4.18. Here, the displacement of the buoy relative to the initially mapped position is visible by comparing the white area in the southeast corner of the map with the southwest corner of the observation. Furthermore, the white area located in the direction of the heading estimation, are related to the pole added in this test. This area appears in white meaning that it is occupied.

The final test consisted on the movement of the vehicle. The algorithm miscalculated, in an early iteration, the position estimation. This resulted in a mapping error which propagated from there until a blur was formed and therefore, the algorithm failed to generate a good result. However, limiting the mapping part of the algorithm and running the last three observations of the SONAR through the localization algorithm alone, resulted in an accurate estimation of the final position. The choice of the use of the last three iterations came from visualization of the readings and comprehending that the amount of noise in the last three iterations was reduced when compared to the remaining. The values for the error of the twelfth iteration in table 4.1 are the

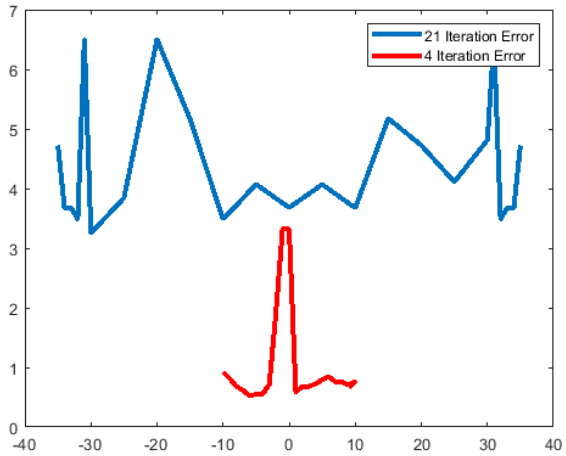
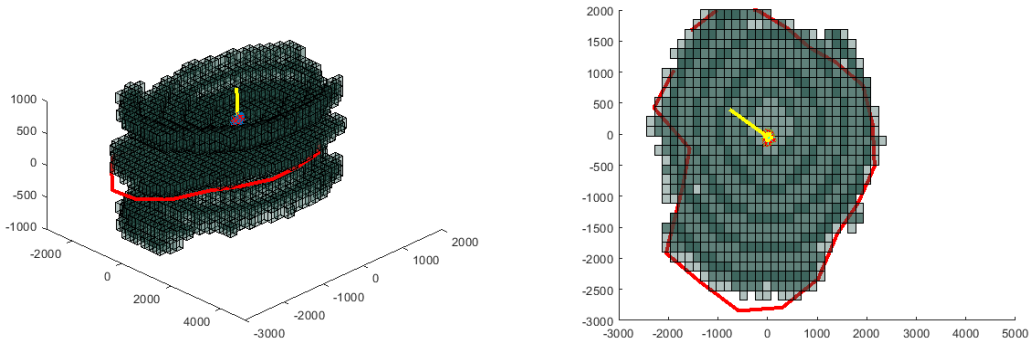


Figure 4.15: The square of the errors in relation to the angle variations for the third test.



(a) 3D representation.

(b) 2D representation, seen from above.

Figure 4.16: Image of the free voxels in an Octomaps representation (blue) with the observation of the twenty first iteration of the algorithm (red), after movement from the buoy, taken in the third test. Axis are in millimeters.

result of this change. The initial pose estimation was $(x, y, z, \psi) = (-0.0977, -0.0835, 0.2285, -7)$ and the final $(x, y, z, \psi) = (-1.9333, 0.4758, -0.1166, 1)$. The intermediate position given to the localization algorithm in order to converge to the final position was $(x, y, z, \psi) = (0.9, 0.2, 0.2285, -7)$. The graph containing the comparison between the initial and final square of the map and observation error is explicit in fig. 4.19. In this graph, it is possible to see that due to the lack of mapping, the error between the position estimation and the final map is significant.

The final observation, the probabilistic map taken from the first few iterations of the sensor and the graphical representation of the pose of the vehicle are visible in fig. 4.20. Here, it is visible that the map and observations are mostly coherent. However, some lack of detail in some of the zones, specially in the ones associated with the objects, is clear. This map could be improved by running the mapping algorithm in this position.

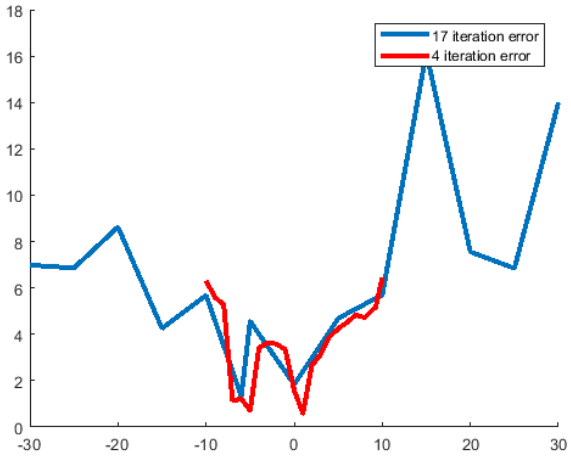
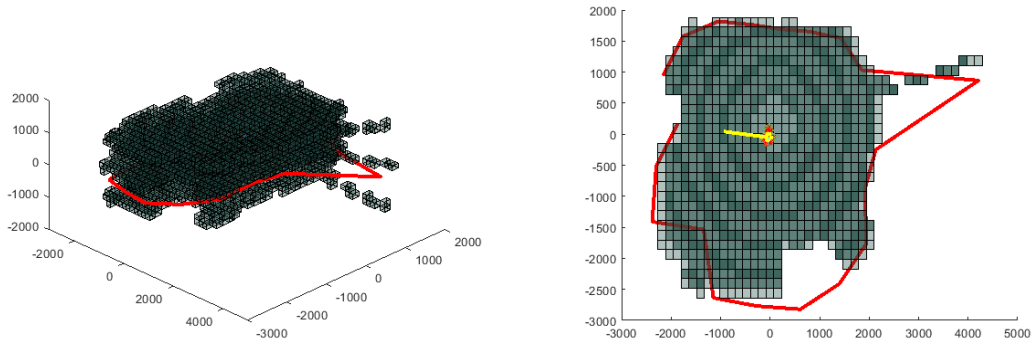


Figure 4.17: The square of the errors in relation to the angle variations for the fourth test.



(a) 3D representation.

(b) 2D representation, seen from above.

Figure 4.18: Image of the free voxels in an Octomaps representation (blue) with the observation of the seventeenth iteration of the algorithm (red), after movement from the buoy, taken in the fourth test. Axis are in millimeters.

4.4 Synthesis

From this chapter, it is possible to conclude that the algorithm produces good position estimations when all bodies present in the structured environment remain static. However, when the map is distorted by outside noise, such as the movement of a foreign object within the water tank, the map may be corrupted and the position estimation may be lost. In cases where more static objects are present and the algorithm may use these stronger features to estimate the localization, the error is much smaller. The complete efficacy of this algorithm when movement is introduced into the vehicle, needs to be validated in future studies.

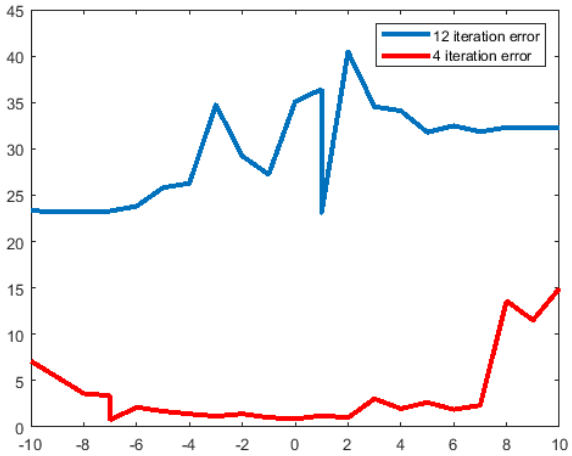
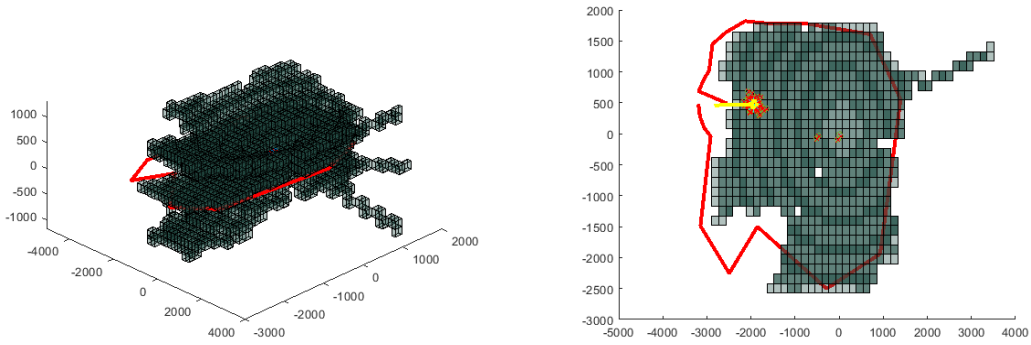


Figure 4.19: The square of the errors in relation to the angle variations for the fifth test.



(a) 3D representation.

(b) 2D representation, seen from above.

Figure 4.20: Image of the free voxels in an Octomaps representation (blue) with the observation of the seventeenth iteration of the algorithm (red), after movement of the vehicle, taken in the fifth test. Axis are in millimeters.

Chapter 5

Discussion

In the literature revision, a few references relating to the exploration of the third dimensional capacities of the SONAR beam were found and therefore, this study is a contribution to the investigation and to the future development in this area. Zhou et al., 2016 approached the topic of considering several rays within a SONAR beam but their proposal was limited to the horizontal plane. L. Li et al., 2014 proposed a map building technique based on fuzzy Dempster-Shafer information fusion that processed the data from a SONAR and considered the 3D capabilities of the SONAR beam to attribute the probability to the map. The approach was simulated and the accuracy of the results increased as time passed. proposes a 3D reconstruction of underwater objects where the 3D capabilities of the MSIS are also explored and turned into a point cloud through an inverse SONAR model. This information is then placed in an Octomaps representation to reconstruct the shapes of underwater objects.

Q. Zhang et al., 2018 proposed a map building approach through an UKF-SLAM algorithm. The feature extractor in their study combined Progressive Probabilistic Hough transform and Random Sampling Consensus. However, a direct comparison with the results of Q. Zhang et al., 2018 cannot be made since the tests done in this article have different conditions.

The algorithm proposed by Choi et al., 2016 consists on using EKF SLAM to process the data from two hydrophones equipped on an AUV. The heading of the vehicle was not estimated. The position error in the article would rise to 1 meter. However, it is worth noticing that the test environment for this master dissertation is, at maximum, below 6 meters and the test environment of Choi et al., 2016 reached 40 meters range in some points.

W. Chen and Sun, 2018 proposed a SLAM implementation to obtain the AUV localization through the use of three beacons with uncertain positions and used this information to further improve their knowledge on the beacons positions. This study compared the effectiveness of the algorithm with DR and a combination of DR and LBL and obtained better results. The position error was of about 20 meters in a simulated test where the vehicle would have travelled 1.6 kilometers. This master dissertation moves slightly away from this study.

Cao et al., 2016 proposed a SLAM algorithm that uses virtual noise compensation to reduce the modelling errors created by EKF algorithms. It showed an improvement regarding EKF SLAM

implementations but it is not possible to compare numerically with this master dissertation since the results shown only compare their approach with the common EKF implementation and do not generate numerical results for further comparison

Soylu et al., 2018 created a SLAM algorithm based on an EKF that processes data from a SONAR. The algorithm proposed managed a ten centimeter and 12" resolution within a similar testing environment. Moreover, a DVL was implemented to estimate the velocities of the vehicle. Numerical test results were not presented.

The algorithm proposed by Zhou et al., 2016 also considers the existence of different rays within the SONAR beam. In the study, the SONAR was set to look forward within a ± 45 degree angle. The mapping representation used for this study were the OGs. To obtain the position, the localization error was corrected by back propagating the error between DR surfacing location and the surface GPS position. The dynamic inverse SONAR model was necessary to compensate for the uncertainty of the direction of the sensor reflections. This study was able to produce a map of the region but it is not possible to compare the mapping effectiveness of both approaches.

Vidal et al., 2018, proposed a view planning exploration algorithm that had as goal the construction of 2D maps of the environment by quadtree representation. In this map representation, the states for the cells of the map, can take on six different states. These include a knowledge component, similarly to the definition of certainty proposed in this master dissertation. The algorithm was able to map irregularly shaped environments. The accuracy or resolution of these maps was not quantified and therefore, a comparison in that level did not take place.

The knowledge on autonomous map building, in recent years, has been advancing. Its practical applicability goes from bathymetric surveys and research on marine wildlife to rescue missions. Several studies have been contributing towards this topic, with this project being yet another improvement of the knowledge in this area. In the literature, different experiments have different conditions which increases difficulty in method comparison. Nevertheless, the information sharing between the research and investigation community is a contribution to the innovation and development of new studies.

Chapter 6

Conclusions and future work

This thesis proposed a solution for autonomous map building through means of a SLAM technique that incorporated a PF and a probabilistic method of attribution of occupancy states to the voxels of an Octree. The final representation of the map was done through the use of Octomaps. The information about the surrounding environment was obtained through the use of a MSIS sensor. The data from the sensor was cleared through a feature extractor to make the data usable. Furthermore, the beam from the SONAR was expanded to its third dimensional form instead of considering the beam as a single line.

6.1 Objective achievement

The results obtained in the different scenarios exposed the positive and the negative sides of the algorithm. When the map is static, the SLAM implementation functions quite well and produces a good estimation of the position and of the heading with a small error. This algorithm was able to produce maps autonomously even in situations where it was expected that the product would fail. Such as in the situation of an object having moved within the pool and a second object remained static, the algorithm was able to keep a small error. In this perspective, the results exceeded what was initially predicted as an objective. In a similar situation, where a second object did not exist, the algorithm produced a large heading error. However, this was expected to happen. Furthermore, the robustness of the algorithm with motion introduced into the vehicle needs further testing and configuration.

6.2 Future work

This algorithm has space for improvement and this section contains some of the possible enhancements for this algorithm and some things that were not possible to implement.

A possible enhancement to this algorithm is improving the intersection detection algorithm for each ray within the SONAR beam volume, with the faces of the Octomaps cubes in order to

enable the option of a real time implementation. In future work, this algorithm is meant to be implemented in real time and used for navigation.

Propagating the SONAR rays to cover the whole volume occupied by the SONAR beam instead of considering a finite number, such as the one showed in the present implementation that considered nine rays per beam, should also prove to show better and also better looking results.

In the beginning of chapter 3, the variable certainty was defined with the purpose of saving a variable that contains data relative to the reliability of each point of the map. This variable is changed according to the times that a voxel is seen in the same state. If a voxel's state keeps alternating between free and occupied, the information about the state of that voxel is not reliable. This usually happens due to the vehicle being far away from the uncertain voxels or an object was positioned in between the two bodies. Therefore, if these uncertain areas are identified, it is possible to command the vehicle to approach them for better readings, thus enhancing the information on those areas. However, not all uncertain voxels can be read. This is, if an uncertain voxel is situated behind a wall, it can never be read, even if the vehicle comes closes in on this position. Therefore, only uncertain points within the outer limits of the map can be identified as belonging to an uncertain area and, therefore, a place of interest. However, the algorithm of identification of these areas was not finalized and, therefore, was not presented in this study but it is an area of interest for future work. Developing a control algorithm for the vehicle that takes the uncertain areas and moves to a certainly free voxel in the vicinity of that zone, would allow the vehicle to visualize certain voxels from different angles and, therefore, enhance the map. Furthermore, the variation of certainty throughout the map could also be incorporated in the estimation of the pose which is expected to create an improvement in results.

Bibliography

Abukawa, K., Matsumoto, S., Hirabayashi, T., Sato, T., Iida, H., Nanri, M., Yoshie, M., Katakura, K., & Takahashi, K. (2016). Reconstructed real-time 3d image of chimneys using underwater acoustic video camera in the experimental field, In *Oceans 2016 mts/ieee monterey*. https://doi.org/10.1109/OCEANS.2016.7761385

Abukawa, K., Matsumoto, S., Hirabayashi, T., Shirai, K., Sato, T., Iida, H., Nanri, M., Yoshie, M., Katakura, K., & Takahashi, K. (2015). Experimentation for development of underwater acoustic video camera: In experiment dock, In *Oceans 2015 - mts/ieee washington*. https://doi.org/10.23919/OCEANS.2015.7401944

Besl, P. J., & McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(2), 239–256. https://doi.org/10.1109/34.121791

Cao, M., Li, Z., & Li, F. (2016). On improved simultaneous localization and mapping algorithm for underwater navigation, In *2016 35th chinese control conference (ccc)*. https://doi.org/10.1109/ChiCC.2016.7554177

Carpenter, J., Clifford, P., & Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, *146*(1), 2–7. https://doi.org/10.1049/ip-rsn:19990255

Chen, L., Wang, S., McDonald-Maier, K., & Hu, H. (2013). Towards autonomous localization and mapping of auvs: A survey. *International Journal of Intelligent Unmanned Systems. 1*, 97–120. https://doi.org/10.1108/20496421311330047

Chen, W., & Sun, R. (2018). Range-only slam for underwater navigation system with uncertain beacons, In *2018 10th international conference on modelling, identification and control (icmic)*. https://doi.org/10.1109/ICMIC.2018.8529843

Chitre, M., Shahabudeen, S., & Stojanovic, M. (2005). *Underwater acoustic communications and networking: Recent advances and future challenges* (Vol. 42). Marine Technology Society Journal.

Choi, J., Lee, Y., Kim, T., Jung, J., & Choi, H. (2016). Ekf slam using acoustic sources for autonomous underwater vehicle equipped with two hydrophones, In *Oceans 2016 mts/ieee monterey*. https://doi.org/10.1109/OCEANS.2016.7761439

David. (2020). Buildoctree [Last accessed in 2020/06/22]. MATLAB Central. https://www.mathworks.com/matlabcentral/fileexchange/36782-buildoctree

De Silva, K. T. D. S., Cooray, B. P. A., Chinthaka, J. I., Kumara, P. P., & Sooriyaarachchi, S. J. (2018). Comparative analysis of octomap and rtabmap for multi-robot disaster site mapping, In *2018 18th international conference on advances in ict for emerging regions (icter)*. https://doi.org/10.1109/ICTER.2018.8615469

Diosi, A., & Kleeman, L. (2007). Fast laser scan matching using polar coordinates. *I. J. Robotic Res.*, *26*, 1125–1153. https://doi.org/10.1177/0278364907082042

Doucet, A., & Johansen, A. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, *12*.

Etter, P. C. (1995). Underwater acoustic modeling: Principles, techniques and applications (Second).

Eustice, R. M., Singh, H., & Leonard, J. J. (2006). Exactly sparse delayed-state filters for view-based slam. *IEEE Transactions on Robotics*, *22*(6), 1100–1114. https://doi.org/10.1109/TRO.2006.886264

Fearnhead, P., & Clifford, P. (2003). On-line inference for hidden markov models via particle filters. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, *65*(4), 887–899. http://www.jstor.org/stable/3647589

Ferri, G., Tesei, A., Stinco, P., & LePage, K. D. (2019). A bayesian occupancy grid mapping method for the control of passive sonar robotics surveillance networks, In *Oceans 2019 - marseille*. https://doi.org/10.1109/OCEANSE.2019.8867152

Floriani, B. L., Palomeras, N., Weihmann, L., Simas, H., & Ridão, P. (2017). Model-based underwater inspection via viewpoint planning using octomap, In *Oceans 2017 - anchorage*.

Fula, J. P., Ferreira, B. M., & Oliveira, A. J. (2018). Auv self-localization in structured environments using a scanning sonar and an extended kalman filter, In *Oceans 2018 mts/ieee charleston*. https://doi.org/10.1109/OCEANS.2018.8604798

Gomaa, W., El-Sherif, A., & El-Sharkawy, Y. (2015). Underwater laser detection system. *Proceedings of SPIE - The International Society for Optical Engineering*, *9342*. https://doi.org/10.1117/12.2080181

Gonçalves, C. S., Ferreira, B. M., & Matos, A. C. (2016). Design and development of shad - a small hovering auv with differential actuation, In *Oceans 2016 mts/ieee monterey*. https://doi.org/10.1109/OCEANS.2016.7761457

Grewal, M., Weill, L., & Andrews, A. (2007). *Global positioning systems, inertial navigation, and integration* (Second). Wiley-Interscience, Singapore, https://doi.org/10.1002/9780470099728.ch3

He, B., Liang, Y., Feng, X., Nian, R., Yan, T., Li, M., & Zhang, S. (2012). Auv slam and experiments using a mechanical scanning forward-looking sonar. *Sensors (Basel, Switzerland)*, *12*, 9386–410. https://doi.org/10.3390/s120709386

He, B., Yang, L., Yang, K., Wang, Y., Yu, N., & Lu, C. (2009). Localization and map building based on particle filter and unscented kalman filter for an auv, In *2009 4th ieee conference on industrial electronics and applications*. https://doi.org/10.1109/ICIEA.2009.5138943

Hernández, E., Ridao, P., Romagós, D. R., & Joan, B. (2009). Msispic: A probabilistic scan matching algorithm using a mechanical scanned imaging sonar. https://doi.org/10.14198/JoPha.2009.3.1.02

Hidalgo, F., & Bräunl, T. (2015a). Review of underwater slam techniques, In *2015 6th international conference on automation, robotics and applications (icara)*. https://doi.org/10.1109/ICARA.2015.7081165

Hidalgo, F., & Bräunl, T. (2015b). Review of underwater slam techniques, In *2015 6th international conference on automation, robotics and applications (icara)*. https://doi.org/10.4031/002533208786861263

Himri, K., Pi, R., Ridao, P., Gracias, N., Palomer, A., & Palomeras, N. (2018). Object recognition and pose estimation using laser scans for advanced underwater manipulation. https://doi.org/10.1109/AUV.2018.8729742

Hornung, A., Wurm, K., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, *34*. https://doi.org/10.1007/s10514-012-9321-0

International, T. (2019). Micron sonar product datasheet. Tritech International (n.d.) https://www.tritech.co.uk/product/small-rov-mechanical-sector-scanning-sonar-tritech-micron

J. Liu, R. C., & Logvinenko, T. (2001). A theoretical framework for sequential importance sampling and resampling. *Sequential Monte Carlo Methods in Practice*, 1–24.

Karras, G. C., Panagou, D. J., & Kyriakopoulos, K. J. (2006). Target-referenced localization of an underwater vehicle using a laser-based vision system, In *Oceans 2006*. https://doi.org/10.1109/OCEANS.2006.307112

Khazraj, H., Faria da Silva, F., & Bak, C. L. (2016). A performance comparison between extended kalman filter and unscented kalman filter in power system dynamic state estimation, In *2016 51st international universities power engineering conference (upec)*. https://doi.org/10.1109/UPEC.2016.8114125

Kitagawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, *5*(1), 1–25. http://www.jstor.org/stable/1390750

Kohlbrecher, S., von Stryk, O., Meyer, J., & Klingauf, U. (2011). A flexible and scalable slam system with full 3d motion estimation, In *2011 ieee international symposium on safety, security, and rescue robotics*. https://doi.org/10.1109/SSRR.2011.6106777

Kozierski, P., Lis, M., & Zietkiewicz, J. (2013). Resampling in particle filtering - comparison. *studia z automatiky i informatiky*, *38*.

Kundu, A. S., Mazumder, O., Dhar, A., & Bhaumik, S. (2016). Occupancy grid map generation using 360° scanning xtion pro live for indoor mobile robot navigation, In *2016 ieee first international conference on control, measurement and instrumentation (cmi)*. https://doi.org/10.1109/CMI.2016.7413791

Kwon, S., Park, J., & Kim, J. (2017). 3d reconstruction of underwater objects using a wide-beam imaging sonar, In *2017 ieee underwater technology (ut)*. https://doi.org/10.1109/UT.2017.7890306

Lee, E., & Lee, S. (2016). Development of underwater terrain's depth map representation method based on occupancy grids with 3d point cloud from polar sonar sensor system, In *2016 13th international conference on ubiquitous robots and ambient intelligence (urai)*. https://doi.org/10.1109/URAI.2016.7625763

Li, L., Zhu, D., Sun, B., & Deng, Z. (2014). The 3-d map building of auv based on d-s information fusion, In *Proceedings of the 33rd chinese control conference*. https://doi.org/10.1109/ChiCC.2014.6896451

Li, T., Bolic, M., & Djuric, P. M. (2015). Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, *32*(3), 70–86. https://doi.org/10.1109/MSP.2014.2330626

Li, T., Sattar, T. P., & Sun, S. (2012). Deterministic resampling: Unbiased sampling to avoid sample impoverishment in particle filters. *Signal Processing*, *92*(7), 1637–1645. https://doi.org/https://doi.org/10.1016/j.sigpro.2011.12.019

Mallios, A., Ridao, P., Ribas, D., & Hernández, E. (2010). Probabilistic sonar scan matching slam for underwater environment, In *Oceans'10 ieee sydney*. https://doi.org/10.1109/OCEANSSYD.2010.5603650

Martínez, J., González-Jiménez, J., Morales, J., Mandow, A., & Garcia, A. (2006). Mobile robot motion estimation by 2d scan matching with genetic and iterative closest point algorithms. *Journal of Field Robotics*, *23*, 21–34. https://doi.org/10.1002/rob.20104

Massot-Campos, M., & Codina, G. O. (2015). Optical sensors and methods for underwater 3d reconstruction. *Sensors (Basel)*, 1–33. https://doi.org/doi:10.3390/s151229864

Montesano, L., Minguez, J., & Montano, L. (2005). Probabilistic scan matching for motion estimation in unstructured environments, In *2005 ieee/rsj international conference on intelligent robots and systems*. https://doi.org/10.1109/IROS.2005.1545182

of Naval Personnel, B. (1953). Essentials of echo-ranging equipment. NAVPERS 10884.

Paull, L., Saeedi, S., Seto, M., & Li, H. (2014). Auv navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, *39*(1), 131–149. https://doi.org/10.1109/JOE.2013.2278891

Reina, A., & González-Jiménez, J. (2000). A two-stage mobile robot localization method by overlapping segment-based maps. *Robotics and Autonomous Systems*, *31*, 213–225. https://doi.org/10.1016/S0921-8890(99)00098-6

Ribas, D., Ridao, P., Neira, J., & Tardos, J. D. (2006). Slam using an imaging sonar for partially structured underwater environments, In *2006 ieee/rsj international conference on intelligent robots and systems*. https://doi.org/10.1109/IROS.2006.282532

Rigby, P., Pizarro, O., & Williams, S. B. (2006). Towards geo-referenced auv navigation through fusion of usbl and dvl measurements, In *Oceans 2006*. https://doi.org/10.1109/OCEANS.2006.306898

Rodrigues, P. M., Cruz, N. A., & Pinto, A. M. (2018). Altitude control of an underwater vehicle based on computer vision, In *Oceans 2018 mts/ieee charleston*. https://doi.org/10.1109/OCEANS.2018.8604525

Shaffer, G., González-Jiménez, J., & Stentz, A. (1993). Comparison of two range-based estimators for a mobile robot. *Mobile Robots VII*. https://doi.org/10.1117/12.143791

Soylu, S., Hampton, P., Crees, T., Woodroffe, A., & Jackson, E. (2018). Sonar-based slam navigation in flooded confined spaces with the imotus-1 hovering auv, In *2018 ieee/oes autonomous underwater vehicle workshop (auv)*. https://doi.org/10.1109/AUV.2018.8729738

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics (intelligent robotics and autonomous agents)*. The MIT Press.

Urick, R. J. (1995). Principles of underwater sound (Third). McGraw-Hill Book Company.

Vasquez-Gomez, J. I., Sucar, L. E., & Murrieta-Cid, R. (2013). Hierarchical ray tracing for fast volumetric next-best-view planning, In *2013 international conference on computer and robot vision*. https://doi.org/10.1109/CRV.2013.42

Vidal, E., Hernández, J. D., Istenic, K., & Carreras, M. (2018). Optimized environment exploration for autonomous underwater vehicles, In *2018 ieee international conference on robotics and automation (icra)*. https://doi.org/10.1109/ICRA.2018.8460919

Weber, J., Franken, L., Jörg, K., & Puttkamer, E. (2002). Reference scan matching for global self-localization. *Robotics and Autonomous Systems*, *40*, 99–110. https://doi.org/10.1016/S0921-8890(02)00235-X

Welch, G., & Bishop, G. (1995). *An introduction to the kalman filter* (tech. rep.). University of North Carolina at Chapel Hill. USA.

Yozevitch, R., & Ben-Moshe, B. (2016). Advanced particle filter methods. Heuristics; Hyper-Heuristics - Principles; Applications, Javier Del Ser Lorente, IntechOpen. https://doi.org/10.5772/intechopen.69236

Zhang, H., Sun, C., Zeng, Y., & Li, P. (2018). A fusion localization algorithm combining mcl with ekf, In *2018 17th international symposium on distributed computing and applications for business engineering and science (dcabes)*. https://doi.org/10.1109/DCABES.2018.00063

Zhang, Q., Niu, B., Zhang, W., & Li, Y. (2018). Feature-based ukf-slam using imaging sonar in underwater structured environment, In *2018 ieee 8th international conference on underwater system technology: Theory and applications (usys)*. https://doi.org/10.1109/USYS.2018.8778989

Zhou, M., Bachmayer, R., & deYoung, B. (2016). Mapping for control in an underwater environment using a dynamic inverse-sonar model, In *Oceans 2016 mts/ieee monterey*. https://doi.org/10.1109/OCEANS.2016.7761190