FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Underwater localization in complex environments

Maria Sara Delgadinho Noronha

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Bruno M. Ferreira

Co-supervisor: Nuno A. Cruz

July 26, 2021

© Sara Delgadinho Noronha, 2021

Resumo

Com o aumento do poder da computação e os avanços em robótica, os veículos subaquáticos realizam as suas tarefas com cada vez mais exatidão. Os veículos subaquáticos autónomos (AUVs) são de grande interesse, especialmente para tarefas que são perigosas para os humanos, como inspeção, vigilância e segurança do meio ambiente. Estas operações estão a ser executadas em diversos ambientes, sendo estes cada vez mais complexos. Ambiente complexo é definido aqui como um cenário com diversas estruturas e com objetos destacados das paredes, podendo o objeto ter uma certa variabilidade de orientação, com posição nem sempre conhecida. Por exemplo, portos, um tanque, ou mesmo uma barragem.

A capacidade de localização de um veículo subaquático autónomo é de extrema importância, e torna-se crucial quando se encontra em ambientes complexos, para conhecer a sua posição, mas especialmente para conhecer a localização dos obstáculos. A posição dos obstáculos é importante pois podem representar um perigo para o AUV ou para este navegar face aos mesmos.

Este trabalho endereça a localização subaquática nestes ambientes. A motivação principal é um caso de aplicação num observatório subaquático onde é necessária a localização de uma amarração. Embora a posição do cabo no mundo seja conhecida, a dinâmica do cabo introduz incerteza quanto à sua posição exata. Assim, o veículo deve localizar-se relativamente ao cabo.

Os AUVs normalmente empregam uma variedade de sensores para localização e percepção. Para o problema descrito, esta dissertação baseia-se em sensores óticos, uma câmera, e acústicos, sonar de varrimento, a fim de extrair características ambientais relevantes que permitam estimar a localização do veículo face ao cabo. Os recursos extraídos de cada um dos sensores, usando a transformada de Hough para a imagem óptica e o template matching para a imagem acústica, são usados para alimentar um estimador de posição, implementando um Filtro de Kalman Estendido (EKF), a fim de estimar a posição do cabo.

Palavras-Chave: Análise de Imagem, AUV, EKF, Imagem Acústica, Localização, Sonar, Template Matching, Transformada de Hough

Abstract

With increasing computing power and advances in robotics, underwater vehicles carry out their tasks with increasing accuracy. Autonomous underwater vehicles (AUVs) are of great interest, especially for tasks that are dangerous to humans, such as inspection, surveillance, and environmental safety. These operations are being performed in different environments, which are more complex. Complex environment is defined here as a scenario with different structures and objects detached from the walls, the object may have a certain variability of orientation, with a position not always known. For example, harbors, a tank, or even a dam.

The localization ability of an Autonomous Underwater Vehicle (AUV) is extremely important, and it becomes crucial when in complex environments, to know its position, but especially to know the obstacles' position. The position of obstacles is important as they can pose a danger to the AUV or for the vehicle to navigate according to it.

This work addresses the underwater localization in these environments. The main motivation is an application case in an underwater observatory where the localization of a mooring is necessary. Although the position of the cable in the world is known, the cable's dynamics introduces uncertainty as to its exact position. Thus, the vehicle must locate itself according to the cable.

AUVs typically employ a variety of sensors for localization and perception. For the problem described, this dissertation is based on optical sensors, a camera, and acoustic sensors, a scanning sonar, in order to extract relevant environmental characteristics that allow estimating the position of the vehicle relative to the cable. The resources extracted from each of the sensors are processed, using the Hough transform for the optical image and the template matching for the acoustic image, and then used to feed a position estimator, implementing an Extended Kalman Filter (EKF), in order to estimate the cable's position.

Keywords: Acoustic Images, AUV, EKF, Image Analysis, Hough Transform, Localization, Sensor Fusion, Sonar, Template Matching

Acknowledgments

This work is financed by the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project QuALTOS (PT-IL/0002/2019).

First, I would like to express my sincere gratitude to both my dissertation supervisors, Bruno Ferreira and Nuno Cruz, for all the patience, guidance and learning moments.

The conclusion of this thesis marks the end of a great chapter in my life. It culminates 5 years of change and a lot of learning.

I can't thank my family enough. To my parents, my sister and my uncles who supported me, unconditionally, during this journey. To my cousins, Tiago and Pepas, housemates over the years, who brought many stories to tell. To my grandmother and grandfather who always accompanied me with great joy and patience.

Finally, I could not fail to thank my friends of a lifetime, but especially the friends that college brought me and without them these years would not be the same. With special affection, I want to thank Joana, Vasco and Capi, as well as Miguel and the Lousada family who, during all the hustle and bustle of the development of this dissertation, managed to bring me a lot of joy.

Sara Delgadinho Noronha

“Life is like a box of chocolates.
You never know what you’re gonna get.”

Forrest Gump

Contents

List of Figures

List of Tables

Acronyms

AHRS	Attitude and Heading Reference System
ANNs	Artificial Neural Networks
AUV	Autonomous Underwater Vehicle
CNNs	Convolutional Neural Networks
DL	Deep Learning
DR	Deduced Reckoning or Dead Deckoning
DVL	Doppler Velocity Log
DoG	Difference-of-Gaussians
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
GIBs	GPS Intelligent Buoys
GN	Geophysical Navigation
GPU	Graphical-Processing Units
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ISO	International Standards Organization
KF	Kalman Filter
LBL	Long Baseline System
LoG	Laplacian of Gaussian
MSIS	Mechanically Scanned Imaging Sonars
SAS	Synthetic Aperture Sonar
SBL	Short Baseline System
SLAM	Simultaneous Localization And Mapping
SONAR	Sound Navigation And Ranging
SSBL	Super-Short Baseline System
TOF	Time Of Flight
USBL	Ultra-Short Baseline System
YOLO	You Only Look Once

Chapter 1

Introduction

1.1 Context

Over the years, a large number of Autonomous Underwater Vehicles (AUVs) are being designed to accomplish a wide range of applications in the scientific, commercial, and military areas due to the increase in the complexity and risk of the tasks to be performed. For oceanographic studies, AUVs have become very popular to explore, collect data, and create 3D reconstructions or maps. AUVs are also used for port and harbour security tasks such as environmental inspection and surveillance.

Most of the tasks employed by AUVs are given through spatial localization, which makes localization, absolute or relative, essential for AUVs. Absolute localization is the position in the world, usually obtained through acoustic signals or satellite based signals. Relative localization is the position relative to a given object or feature, usually acquired through optical and acoustic sensors.

The ability of an AUV to move accurately is dependent on the performance of its localization. In complex environments robots can, similarly to humans, take advantage of the characteristics of the environment to locate themselves. Complex environment is defined here as a scenario with objects detached from walls, for example the object can have a certain dynamic therefore its position is not known at all times. Examples of scenarios can be a harbour, a tank or even a dam with obstacles inside. In such environments, the presence of points of interest (features) can be dense but their relevance to the localization is variable.

In order to take advantage of these features there are various potential AUV sensors, and state estimators to aid navigation. Some sensors provide various information such as its position, velocity, and acceleration, information about the robot itself, while others such as cameras, acoustic sensors and force sensors gather the information of the surrounding environment. These determine the position relative to the properties or features of the environment. As there are several sensors that contribute to the same objective (the localization), an algorithm is necessary, for instance the state estimators, which merge the data of these sensors.

1.2 Motivation

Interest in underwater robotics research has increased over the last couple of decades, and yet our knowledge of this vast environment is still limited.

When an AUV is operating beneath the sea surface and without an external positioning system, the ability of an autonomous underwater vehicle (AUV) to locate itself in a complex environment is of crucial importance for successful navigation. This is usually done through the use of cameras and forward-looking sonars. However, this is particularly challenging in underwater environments due to dark environment, for the optical sensors, and the rapid attenuation of acoustic signals.

As a motivation there is a concrete problem to solve, which is to locate a suspended cable in a water column in a known region in the sea and navigate according to it. Although the cable's position in the world is well known, the cable dynamics does not allow knowing where it is exactly. So, in order to inspect the cable, the vehicle must localize itself according to it. The proposed solution for this problem uses optical and acoustic sensors, since cameras and sonars are usually available in the vehicle for environment monitoring (cameras), or for obstacle avoidance (sonars).

1.3 Objectives

The purpose of this dissertation is to develop an algorithm for feature detection to work in underwater environments which are lacking in significant features or variations, and combine the advantages of both types of sensors, optical and acoustic, allowing a better performance in different ranges.

This thesis aims to:

- Develop an algorithm for feature detection in complex underwater environments
- Use the aforementioned algorithm to locate a suspended cable and navigate according to it.

1.4 Document Structure

This document is composed by six more chapters, apart from the present one.

Chapter 2 is focused on a review of the current state of AUV localization and navigation, from common methodologies to sensor technology employed. Furthermore, an overview of common image processing methods and algorithms is also introduced, as well as sensor fusion techniques.

Chapter 3 introduces the structure of the developed solution in further detail and the logical reasoning behind it.

The work carried out for this purpose is more thoroughly detailed throughout chapters 4, 5 and 6. In chapter 4, Hough transform is applied to optical images and an assessment of contour tracing algorithm is made, to evaluate its contribution to Hough line transform results. In chapter 5, feature extractor algorithms and template matching are applied to acoustic images and their performance is compared. In chapter 6, an Extended Kalman Filter is developed in order to fuse the outputs from both optical and acoustic processing. A careful analysis of the results obtained is presented in each of the chapters and the solutions performance evaluated.

Lastly, chapter 7 highlights the main conclusions and contributions of this work, as well as some considerations regarding future work.

Chapter 2

State of The Art

To tackle the problem of estimating the cable's position, this chapter presents an analysis of what the scientific community has to offer to this work. The diagram 2.1 formulates the structure of this chapter, as well as the sequence of the solution presented in this thesis.

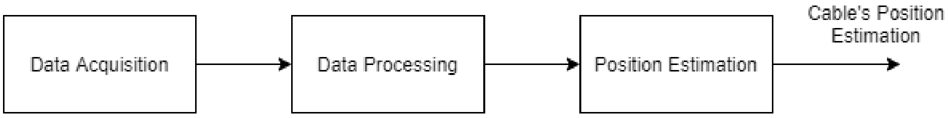


Figure 2.1: Chapter and solution's structure

The information, acquired in Data Acquisition block needs to be processed, by the Data Processing block, in order to provide the required information to feed a position estimator, the next step, Position Estimation block.

Starting with the Data Acquisition block, which includes all the tools and technologies used for the data acquisition, the sensors and the different approaches, all with the same purpose - to provide the necessary information for the localization of the vehicle or of a specific feature and the posterior navigation according to this information.

For the second block, this chapter will cover not only the methods that allow extracting the features but also some pre-processing techniques.

In the third and last section, which receives parameters that define the features as inputs, is a description of the filter used for this purpose along with an overview of other possible filters with the same aim.

Localization and navigation in underwater environments continue to be great challenges due to the rapid attenuation of high frequency signals making the implementation of general and broad used strategies applied in the ground and aerial systems, like GPS for global localization, ineffective in an underwater scenario. However, these vehicles have great application potential, since working underwater can be difficult and dangerous for humans. Localization is the process of estimating the pose of a rigid body within a given reference coordinate system. Sensors that provide this pose directly, with the robustness, noise and frequency characteristics required by the robot's actions, are usually not available. This is usually solved by fusing the information of multiple sensors into an estimate of the vehicle's pose [3]. Navigation consists not only in determining the vehicle's position in the frame of reference, localization, but also planning its route to the desired location and then executing the plan.

2.1 Data Acquisition - Tools, Technologies and Approaches

In robot localization and navigation, there are a variety of sensors that are necessary for their missions. In this section, some of the most relevant sensors and methodologies for this purpose will be explored, and are organized according to the authors in [4] by categories as Figure 2.2.

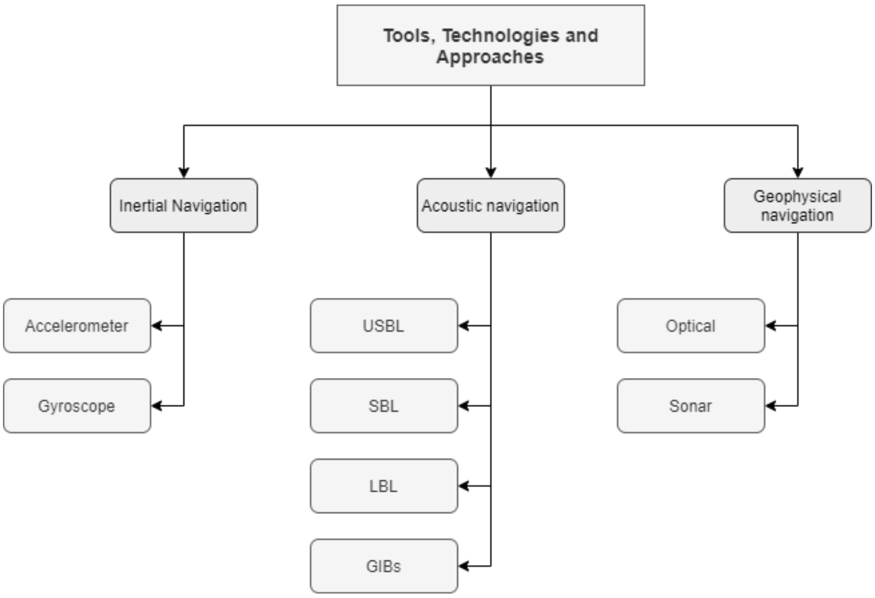


Figure 2.2: AUV tools and approaches for localization and navigation

2.1.1 Inertial Navigation

Inertial sensors exploit the property of inertia, the resistance to a change in momentum, for instance accelerometers and gyroscopes. An Inertial Navigation System (INS) is a navigation system in which measurements provided by inertial sensors are used to calculate the position, orientation, and velocity of a moving object, without the need for outside references, by dead reckoning. Dead reckoning is the process of calculating the current position of some moving object using a previously known position through estimation of speed, and attitude over time.

These systems suffer from integration drift, i.e., biases in the measurement of acceleration and angular velocity become progressively greater errors when integrating to obtain velocities and position [4].

Although this type of navigation system is not within the scope of this dissertation, it could be used in complement to the proposed solution, working in the absence of or between consecutive position fixes.

2.1.2 Acoustic Navigation

Acoustic navigation consists of techniques which use acoustic transponders and modems to estimate the AUV's position measuring the time of flight (TOF) of an acoustic signal from beacons or modems to perform navigation. These signals propagate better in water and can reach hundreds of miles [5], compared with other signals such as radio and electromagnetic [6]. These systems are frequently used in a wide variety of applications, including ocean exploration.

As exposed in Figure 2.2, these positioning systems are generally categorized into three classes: Long Baseline (LBL), Ultra-short-baseline (USBL), and Short-Baseline (SBL) systems differing on the disposition of the transducers and, consequently on the area size to explore [7]. With GPS intelligent buoys (GIBs) systems, the beacons are placed at the surface over a wide area [8]. Besides these, when trying to minimize the cost, single transponders are used instead, where the baseline is simulated by creating small artificial baselines through the vehicle's movement. Also, acoustic modems allow to transmit information along with the signal used for range estimation using TOF [4].

All of these solutions have a limited range and accuracy since they suffer from multipath, caused by reflection and scattering, and Doppler effects, when the source of the waves is moving towards the observer [3].

2.1.3 Geophysical Navigation

Geophysical navigation uses unique features of the surrounding as a reference to estimate AUV's relative position and navigate. For this, different sensors capable of detecting and identifying these features are used.

There are different types of sensors inserted in this category according to [9], such as magnetic sensors, which use magnetic field maps for localization, and chemical sensors, through the identification of specific substances in the water, optical sensors and acoustic sensors or sonars.

2.1.4 Optical Sensors

Monocular or stereo cameras are considered cheap solutions, offering superior update rates when compared to other commonly used sensors in robotics [10]. These can be used to capture colored and high-resolution images, providing information about the environment through the features extracted. The dark underwater environment scattering of light, low visibility range are some of the problems in underwater vision-based systems [11]. Thus, different algorithms are being studied for the features identification in the environment or targets for the AUV's mission.

Many previous algorithms have been proposed and applied in ground and air robotics, such as in [12], where the detection of railway tracks is done using a monocular camera and processing the images with Hough Transform. Although it is also a line detection problem, the environment challenges in ground robotics are different. To overcome these challenges in underwater scenarios, Alcocer et al. [13] adapts a color space to identify features of interest in underwater images, while

Ferrera et al. [14] tackles the problem using the feature extractor algorithms covered later in section 2.2.2.

2.1.5 Sonar

Sonar is a popular sensor in several areas, including robotics. This sensor employs acoustic pulses and their echoes to measure the distance to an object, ranging sonars, or to create images, imaging sonars, which provides more information. Since the sound speed is usually known, the object distance is proportional to the echo's travel time. At ultrasonic frequencies, the sonar energy is focused in a beam, delivering directional information in addition to the distance [15]. In robotics, it can have different purposes such as obstacle avoidance, mapping, and object recognition.

While ranging sonars provide only one measurement of range per ping, calculated by TOF, imaging sonars return several intensity measurements, corresponding to the multiple echoes reflected in the area. This information allows to create an image of the surroundings present on the emitted signal direction of travel. The fundamental principle of imaging sonar is to estimate the reflectivity for all calculated ranges and in all selected directions. In this case, according to Hansen in [16], the field of view is given by the angular width of each element. The angle and range resolution is conveyed by the array length measured in wavelengths and by the bandwidth of the system, respectively.

An important imaging sonar, for this study, is a Mechanical Scanning Imaging Sonar (MSIS). The creation of the acoustic image is made by the mechanic rotation of the transducer's head inside the sonar. In other words, each element of the output array corresponds to a specific distance from the transducer head. By varying the length of the listening period, the sensor can achieve a higher resolution along the beam axis or a greater range, as the total number of discretized spaces tends to be fixed on the sonar device [17].

Multibeam sonars are also a relevant type of sonar, they explore the concept of wave interference. Combining multiple receiving elements and precisely controlling the timing between the emission and receiving times, multibeam sonar allows to distinguish the direction of the returning echoes. The key advantage of a multibeam sonar is that this kind of device can gather information of a larger area much faster if compared to the mechanical scanning sonar [3].

The side-scan sonar, another imaging sonar, requires the AUV to move to acquire imagery on a given area [18], which is not always the case. When the vehicle approaches the feature, it will significantly slow down or even stop, this type of sonar is not a good solution for this problem. However, both multibeam sonar and imaging sonar do not have this imposition, becoming interesting for this mission.

2.2 Data Processing

This section explains how the data coming from the optical and acoustic sensors can be processed. Data processing block aims to extract relevant information from the optical and acoustic images for the AUV's relative position to the cable, the feature in the proposed problem.

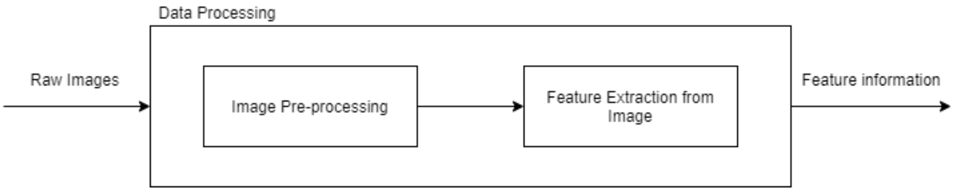


Figure 2.3: Data Processing Procedure

2.2.1 Image Pre-processing

Image pre-processing is the name for operations on images whose aim is to improve the image data, suppressing undesired distortions or enhancing some essential image features for further processing [19]. There are many operators for image preparation, the following methods were the ones used in this dissertation.

Canny Edge Operator

Canny operator was developed to be an optimal edge detector. Edges correspond to areas where image brightness or color change rapidly. This operator produces as output an image showing the positions of tracked intensity discontinuities [20]. The effect of the Canny operator is determined by three parameters — the width of the Gaussian kernel used in the smoothing phase, and the upper and lower thresholds used by the tracker. Increasing the width of the Gaussian kernel reduces the detector's sensitivity to noise, at the expense of losing some of the finer detail in the image.The position error of the edge recognition also increases slightly with the increase of the Gaussian width. If the lower threshold is set too high, the noisy edges will be destroyed. Setting the upper threshold too low increases the number of false edge fragments appearing in the output.

Edge detection filters out irrelevant details while retaining information about the edges of objects in the image. The Canny edge detector is the best known and most widely used among many edge operators. Canny's edge detector consist of four steps:

1. Find the intensity gradient of the image.

2. Apply non-maximum suppression to minimize erroneous edge detection.

3. Apply double threshold to determine potential edges.

4. Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are not connected to strong edges.

Usually images where this operator is applied are blured, the reason being that calculating the derivative in the spatial domain corresponds to accentuating high frequencies and hence magnifying noise.

Contour

Contours are represented as linked edges. The contour can be open or closed, as Figure 2.4 shows. Closed contours correspond to region boundaries, and the pixels in the region may be found by a filling algorithm. An open contour may be part of a region boundary, or simply a contour of a curve.

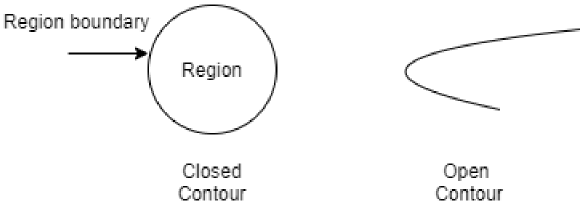


Figure 2.4: Difference between closed contours and open contours

Gaps may appear at the boundaries of regions, because the contrast between regions may not be enough for the boundary detector to find edges along the boundaries. Contours provide very important information for object representation and image recognition.For example, they are used to separate an object from its background and calculate the size of the object classify shapes and to find the feature points of objects using the length and shape of their contour pixels. Therefore, there have been many studies on contour-tracing algorithms for extracting and tracing the contour of an object. [21]

Contour Tracing Algorithm

In digital image processing, image segmentation is the process of partitioning a digital image into multiple segments with the goal to to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. The problems of contour detection and segmentation are related. In general, contour detectors offer no guarantee that they will produce closed contours and hence do not necessarily provide a partition of the image into regions. Closed contours can be recovered from regions in the form of their boundaries.

In conventional contour tracing algorithms, the tracer starts tracing at the contour of an object after it saves the starting point along with its initial direction. Then, it determines the next contour point using its specific rule of following paths according to the adjacent pixel, then move to the boundary point and change its absolute direction.When the viewfinder reaches the starting point, the scanning process ends.

Contour Following can be helpful, enhancing the line contour, to the Hough Transform to detect lines in our problem. However, as in [22], where objects are detected by contour tracing algorithms, the execution time can be a problem in our situation due to the real time demand.

2.2.2 Feature Extraction

Feature extraction is done after the pre-processing phase.The main purpose of feature extraction is to obtain the most relevant information from the original data. In the following sections some of the methods that allow the extraction of information from the image are presented.

Image Transforms

These operators transform the images, in terms of the geometry of the information in the image or the nature of the information itself. The usual purpose of applying a transformation is to help make more obvious or explicit some desired information.

In the case of the operators in this group, the main effect is to make explicit:

- **Distance Transform**, the thickness of image features
- **Fourier Transform**, the spatial frequency composition of the image
- **Hough Transform**, the parameters of geometric shapes.

Hough Transform

The Hough transform, named after Paul Hough, who patented the method in 1962, is a technique that can be used to isolate features of a particular shape within an image. Since this requires parameter adjustments to the required attributes, the classic Hough transform is used to find curves, such as straight lines, circles, ellipses, etc.

The advantage of the Hough transform is that not all pixels in a row need to be continuous. This is useful when you are trying to find lines that appear to be interrupted due to noise, or when an object is partially occluded. However, if the objects are accidentally aligned, this may produce incorrect results. Also the output lines from Hough Line Transform are infinite lines, rather than finite lines with defined end points.

Feature Extractor Algorithms

A feature is a piece of relevant information about the content of an image. Feature extractor algorithms aim to determine interest points on images that remain locally invariant, allowing its detection even with rotation or scale changes [9]. These algorithms represent a solution for the identification of relevant characteristics from the environment, and for this reason are of great importance in many applications such as object recognition and tracking. According to the authors in [18] and [9], the detectors are the algorithms that find the interest points and the descriptors are those who describe the features.

Depending on the complexity of the image, a feature can be an edge, a point where there is a boundary between two image regions, a corner, rapid changes in direction, or it can represent a blob, that is, a set of pixels/pixel values that are related in some way and different from the background.

A brief explanation of the feature detectors and descriptors mainly used in similar works is presented hereafter. The following feature extractors are the compared in Table 2.1 in terms of feature type they detect.

The Harris Corner Detector

This operator was introduced in 1988 by Chris Harris Mike Stephens [23] and takes the differential of the corner score into account with reference to direction directly, instead of using shifting patches for every 45° angle as the previous corner detector and has been proved to be more accurate in distinguishing between edges and corners.

FAST - Features from Accelerated Segment Test

FAST algorithm is essentially a corner detector, created in 2010 by Rosten et al. [24], which computes the difference in brightness based off a circular neighbourhood. The most promising advantage is its computational efficiency. The FAST corner detector is suitable for real-time video processing application because of this low computational complexity.

SIFT - Scale Invariant Feature Transform

This detector, put forward by D.G.Lowe in 1999 [25] is based on Difference-of-Gaussians (DoG) operator which is an approximation of Laplacian-of-Gaussian (LoG). Feature points are detected by searching local maxima using DoG at various scales of the subject images. The description method extracts a neighbourhood around each detected feature and further segments the region into sub-blocks.

SURF - Speeded up robust features

This algorithm is inspired on the SIFT algorithm and it was created in 2008 by Herbert Bay et al. [26]. This detector is based on the computation of the Hessian Matrix and uses integral images to improve computation time and speed. Its feature descriptor is based on the sum of the Haar wavelet [27] response around the point of interest.

ORB - Oriented FAST and Rotated BRIEF

The ORB algorithm is a fast and reliable FAST-based feature detector, and a modified version of the BRIEF algorithm [28] , not only the speed has greatly improved over BRIEF, but also it has the rotational invariance compared with the single BRIEF algorithm. This detector was created in 2011 by Ethan Rublee et al. [29].

KAZE

KAZE features was put forward in 2012 by P.F.Alcantarilla et al. [30] and operates completely in a nonlinear scale space. Through non-linear diffusion filtering makes blurring in images locally adaptive to feature-points, reducing noise and retaining the region's boundaries.

BRISK Binary Robust Invariant Scalable Keypoints

This corner detector put forward by S.Leutenegger et al. in 2011 [31] uses FAST for detecting while searching for maxima in the scale space pyramid. Its description is based on identifying the characteristic direction of each feature achieving rotation invariance.

Algorithm	Type of feature
Harris	Corners
FAST	Corners
SIFT	Corners
SURF	Blobs
ORB	Corners
KAZE	Blobs
BRISK	Corners

Table 2.1: Feature type extractors

In terms of applications, Harris, FAST, ORB and BRISK are usually applied for point tracking, image registration with little or no scale change, corner detection in scenes of human origin, such as streets and indoor scenes. SURF algorithm is typically applied to object detection and image registration with scale and rotation changes. Being able to detect different types of features, SIFT has a lot of applications such as object or landmarks recognition [32], robotic mapping and navigation, 3D modelling, gesture recognition and video tracking, however it is not open-source. These feature extractors are being widely used in acoustic image analysis. For instance, Bandara et al. [18] apply and compare these detectors and descriptors in side-scan sonar, on images for purposes of localization and mapping. With the same purpose, Oliveira et al. in [9] uses these extractors in acoustic images from an imaging sonar.

Template Matching

Template matching is a technique that identifies parts on an image that match a predefined template. Many of the applications of computer vision simply need to know whether an image contains some previously defined object or whether a pre-defined sub-image or pattern is contained within a test image. This auxiliary image is called a template and should be a perfect representation of the template or object being searched in the image.

There are two forms of template matching:

- Global Template Matching - Template used is a global representation of the object.

- Local Template Matching - Several templates are used which represent local features of the object, e.g., corners, edges, characteristic marks etc. present in the object.

One of the problems with pattern matching is that each pattern represents an object or part of it, as we would expect. If the expected orientation can vary, then it requires a separate template for each orientation and each one must be matching with the image. In this cases, template matching may become computationally expensive, especially with large templates. One way of counter this computational overhead is to use smaller local templates to detect salient features in the image which characterize the object.

A measure of matching between the template and every location of the image, called measure of similarity, is calculated and a decision is made based on its value.

Several similarity measures are possible, some based on the summation of differences between the image template, others based on the cross-correlation techniques. In terms of similarity measures, there exist many. Hereafter,there are the methods that are available in the library used, OpenCV 4.5.1, in [33], where T is the template image, I the original image and R the result image from the respective method.

Square Difference

$$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2 \tag{2.1}$$

Normalized Square Difference

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \tag{2.2}$$

Correlation

$$R(x,y) = \sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))^2 \tag{2.3}$$

Normalized Cross-Correlation

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \tag{2.4}$$

Correlation Coefficient

$$R(x,y) = \sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))^2 \tag{2.5}$$

where

$$T'(x',y') = T(x',y') - 1/(w \cdot h) \cdot \sum_{x',y'} T(x'',y'') \tag{2.6}$$

$$I'(x+x',y+y') = I(x+x',y+y') - 1/(w \cdot h) \cdot \sum_{x',y'} I(x+x'',y+y'') \tag{2.7}$$

Normalized Correlation Coefficient

$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y') \cdot I'(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}} \tag{2.8}$$

2.2.3 Deep Learning

At a very basic level, Deep Learning (DL) is a field of machine learning based on deep neural networks that teaches a computer to filter inputs through layers to learn how to predict and classify information. The data can be images, text files, or sound, for example.

Deep learning is being increasingly used for image analysis, among other applications. For instance, in [34] an adaptive neural network image-based visual servo controller is proposed; this control scheme allows placing the underwater vehicle in the desired position concerning a fixed target. Also, [35] presents an image matching algorithm based on a convolutional neural network (CNN) to aid in the navigating of an AUV and to improve the matching accuracy of sonar images in the deep ocean with dynamic backgrounds, low-intensity, and high-noise scenes. Another interesting article in terms of DL, is [36] which uses You Only Look Once (YOLO) which is a clever CNN for doing object detection in real-time, in this case for motion Behaviour Recognition.

These methods can be very useful for feature extraction, however it requires a large data set of images, high-performance GPUs for real-time processing. In the context of this dissertation, creating an image bank, in various conditions, is truly difficult when faced with a deadline of weeks and costs related to access to the sea.

2.3 Position Estimation

When sensors that provide position information directly, such as GPS, are not available, it is necessary to resort to sensory fusion or SLAM to estimate the position.

SLAM (Simultaneous Localization and Mapping) is known as the chicken and egg problem, i.e., a map is needed to localize the robot and its pose estimate is needed to build a map, so SLAM provides the solution - building a map while estimating the pose of the robot relative to this map. In SLAM the robot moves through a partially known environment, uses odometry and observations of nearby features as inputs of the algorithm, and estimates a map of features and the robot pose. Since the purpose of this solution is to provide information about a cable in the sea, and the sea is a unstructured environment SLAM was not considered.

Sensor fusion processes and merges the available sensors data and allows the improvement of the AUV state estimation. For the purpose of this dissertation, the fusion of optical and acoustic data is needed for the position estimation, however this idea is not new, as referred in [37]. Although SLAM and sensor fusion are different methods, they use the same tools - filters, which are described below. The need to merge acoustic and optical data comes from the fact that each sensor has problems that can be counterbalanced by the fusion of data produced by the two. Fusing optical and acoustic data can be done at different levels and with different sensors. There are many examples already explored, for example using an acoustic array with optic systems, as in [38], or pencil beam sonar with optic systems in [39], or using a multibeam and optical system in [40].

It is known that sensors are imperfect and pick up noise along with their measurements, which leads to uncertainties in not only the measurements but also in the consequent estimation. The level of uncertainty that an underwater system is exposed to is much higher if compared to "classic" robotic applications which perform the exact same task exhaustively. Examples of uncertainties in this environment are unmodelled underwater currents, exact weight distribution, and the noisy readings coming from a sonar sensor. Filtering has been widely adopted for dealing with uncertainties and data fusion, Kalman Filters (KFs), Particle Filters (PFs) are some of the popular state estimators used for AUV localization with available sensor data, which will further be analysed individually in the sections below as well as compared in section 2.3.3 and in Table 2.2 some notable features are summarized. In this specific case, the filters will combine the information from the sonar and the optical sensor to estimate the location of the cable.

Filter	**System/Update Model**	**Noise Model**
KF	Linear	Gaussian distribution
EKF	Non-linear	Gaussian distribution
UKF	Non-linear	Gaussian distribution
Particle	No particular	No particular

Table 2.2: Comparison between filters. Adapted from [2]

2.3.1 Kalman Filter

Kalman filter is a set of mathematical equations, providing efficient calculation tools for recursive state estimation, given the measurements observed over time in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects such as supporting estimations of past, present, and even future states. Also, has the major advantage to provide the state and the uncertainty, which may be crucial for decision making, in real-time.[41]

Kalman Filters are normally applied to approximately linear models, assume sensor measurements distribution to be gaussian. Its drawback lies in the fact of narrowing its application within the normal distribution uncertainty, what make this approach limited to more simple models. Applications with multiple hypothesis, according to [8], such as global estimation, are not well suited to be implemented with a KF.

This recursive algorithm consists of two stages – motion prediction or propagation and measurement update or correction, depending on the literature. The first one is based on the kinematic model which will increase the uncertainty. Then the estimator incorporates measurements to update or correct the estimation, which will reduce the uncertainty of the filter.[41]

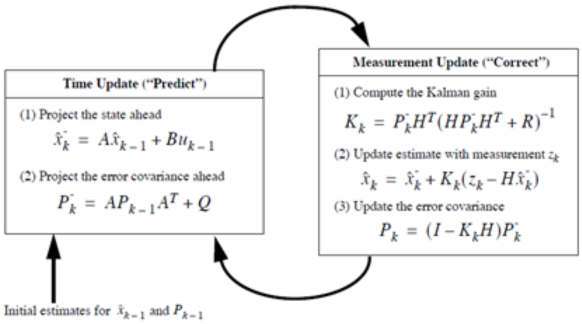

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 2.5: Kalman Filter Algorithm from [1]

Extended Kalman Filter - EKF

Since the KF assumes all equations, motion and measurement, to be linear, in the case of non-linear systems, a linearization of the state transition and measurement models can be performed as in the Extended Kalman Filter (EKF), where the linearization is obtained via a first-order Taylor series expansion.[4]

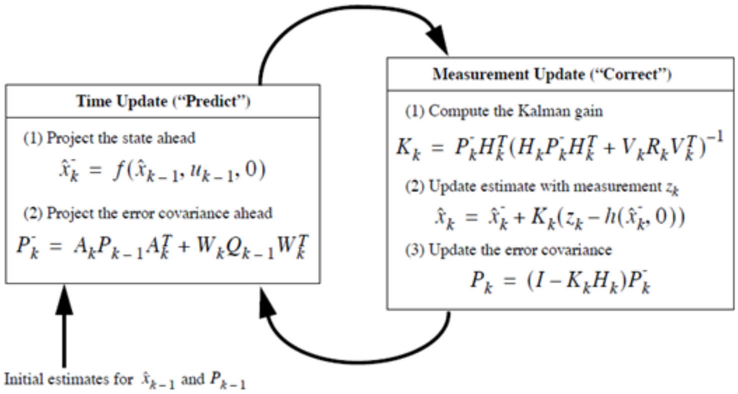

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0)$
(2) Project the error covariance ahead
$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0))$
(3) Update the error covariance
$P_k = (I - K_k H_k)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 2.6: Extended Kalman Filter Algorithm from [1]

The EKF is very similar to the KF in terms of equations, except some minor difference. In this case the A matrix is the Jacobian matrix of partial derivations of process update function with respect to the state vector. And the H matrix represents the Jacobian matrix of partial derivative of the measurement function with respect to the state vector.

Since it is computationally cheaper than other nonlinear filtering methods such as particle filters [41], the Extended Kalman Filter has been used in various real-time applications like navigation systems.

Unscented Kalman Filter

The main drawback of EKF is the approximation by means of the first order truncation. It can create errors in the posterior estimation and may lead to suboptimal performance or even divergence. This becomes significant when the non-linear models are approximated and the higher order terms in the Taylor series are more important than the first order term. To deal with this weakness, a new improvement of the extended Kalman filter was proposed, the Unscented Kalman

Filter, [42]. The UKF uses a deterministic sampling technique known as the unscented transformation (UT) to pick a minimal set of sample points (called sigma points) around the mean. The sigma points are then propagated through the nonlinear functions, from which a new mean and covariance estimate are then formed. The resulting filter depends on how the transformed statistics of the UT are calculated and which set of sigma points are used.

2.3.2 Particle Filter

All the filters introduced assume the associated noises follow Gaussian distributions. One solution for all the other noise models that do not follow this distribution is the particle filter, a popular optimal estimation solution for non-linear and non-Gaussian applications. Particle Filter is called a nonparametric filter, which means that the probability distributions involved in the state estimation does not depend on a fixed set of parameters to be represented. [8]

The particle filter works by simulating the system's operation multiple times, each time drawing a new random number from the process noise's distribution. This way, as the number of particles gets large, the empirical distribution of the particles will approach the true probability distribution of the posterior. In addition, each particle is assigned a weight that determines how likely it is to be the true state, given the received measurements. The weights are updated every time a new measurement is received – the weights are increased for particles closer to the measurement.

The main gain of particle solutions over standard approximation methods is that it is not dependent on any functional linearization or approximation but depends on random particles with associated weights to characterize the posterior distribution. Thus, particle filter can be used in more circumstances and with much flexibility.[3] This flexibility comes with a cost, it usually is computationally expensive.

2.3.3 Filter Comparison

First, if the system in consideration is linear, then a Kalman filter is most likely the best approach. Even if the noise is not Gaussian (or even if it is hard to identify any probability distribution), the Kalman filter still has strong theoretical guarantees [43]. In addition, its numerous successful applications over the years have confirmed its place as the most popular filtering algorithm.

If the system's model is non-linear, then the choice of filter is made more difficult. The extended Kalman filter (EKF) is also a popular choice in this case, it is especially powerful in systems that are non-linear. [44] However, depending on the type of system, the EKF may diverge due to linearizing around a wrong estimate. Furthermore, the EKF is computationally expensive in high-dimensional spaces and may be too slow on resource limited devices, such as microprocessors.

Particle filters accuracy can always be enhanced given more particles. Thus, the advancement of modern computers is yet another argument in favor of the particle filter. [3] On the other hand, particle filters perform poorly in high-dimensional spaces.

Since localization is the main topic of this thesis, mapping and localization related techniques were reviewed. Filtering is essential for a better estimation performance due to the constant presence of noise in the state space and measurements.

Chapter 3

System Overview

In this chapter, a more comprehensive overview of the problem, proposed solution and base system is presented.

3.1 Problem Statement

Autonomous underwater vehicles (AUVs) are noteworthy, particularly for missions that are unsafe to people, such as observation, and natural security. These operations are being performed in numerous situations, which are more complex. Complex environment is characterized here as a situation with diverse structures and objects confined from the walls, the object may have a position not continuously known. For instance, harbors, a tank, or a dam.

The localization ability of an Autonomous Underwater Vehicle (AUV) is vital, and even more when in complex situations, in order to know the vehicle's position, but particularly to know the obstacles' position.

The problem of this dissertation arises from an ongoing project, in which there is a real challenge - to locate a cable in an underwater observatory. Although the position of the cable in the world is known, the cable's dynamics introduces uncertainty as to its exact position. Thus, the vehicle must locate itself according to the cable.

Even though this problem is a unique situation, it can also be adapted to other scenarios where the localization is necessary in relation to certain structures present in the environment.

3.2 Assumptions and Base System

For the purpose of this dissertation, and to address the stated problem a Tritech Micron sonar and a Go Pro camera were made available. Just like the sensors, their position in the vehicle was taken as an assumption since the beginning of this study.

Camera

For this study, a GoPro Hero 3 was positioned facing forward, allowing the visualization of the mooring from the stated problem. This small and light camera provides a cheap solution for the development of a vision-based system.

Sonar

The forward-looking sonar mentioned is an imaging sonar, having a wide vertical beamwidth and a small horizontal opening, is well suited to perform obstacle avoidance and navigation. Objects or targets presented above or below the transducer axis will be detected while maintaining a good horizontal angular resolution. In the following Table are the acoustic specifications of the Micron sonar that support its choice for this problem.

Beamwidth	35° vertical, 3° horizontal
Maximum range	75m
Minimum range	0.3m
Mechanical resolution	0.45°, 0.9°, 1.8°, 3.6°
Scanned sector	Variable up to 360°

Table 3.1: Micron Acoustic Specifications

An imaging sonar can emit a sound beam and measure the intensity of subsequent echoes from a specific location. This information can be subsequently used to generate acoustic images of the surrounding environment. This sonar has an acquisition rate of 8 seconds, i.e., a full revolution that corresponds to an acoustic image of the complete surroundings (360º) takes this period of time to be available.

IMU

An inertial measurement unit was also made available in order to complement acoustic and optical data. This electronic device provides force, angular rates, and usually the orientation of the body through a combination of accelerometers, gyroscopes and occasionally magnetometers.

3.3 Proposed solution

After reviewing all the tools and methodologies to address underwater localization problems, it is possible to design a system that solves this problem. Figure 3.1 presents the structure of this solution in the form of a block diagram.

The main idea to solve this problem is to use optical and acoustic sensors in order to obtain position estimates in complex environments. For this purpose, the solution involves the development of two parallel processes, one for each sensor. Both optical and acoustic sensors provide useful

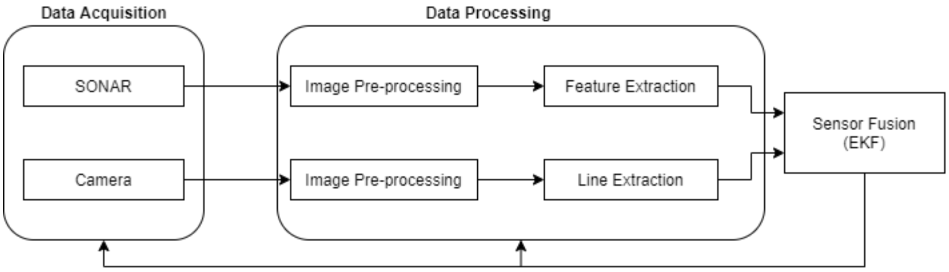


Figure 3.1: System Block Diagram

information for the localization, however each sensor provides different information from the environment. The processes will run in parallel as they have different acquisition rates. Whenever the sensors acquire an image, the raw data acquired will be processed and the result will be fed to a state estimator in order to estimate the cable's position.

3.3.1 Data Acquisition

For this study, a GoPro Hero 3 was positioned facing forward. The camera provides frames, such as the one presented in Figure 3.2, and each pixel can be used to infer information regarding two angles, horizontal and vertical.

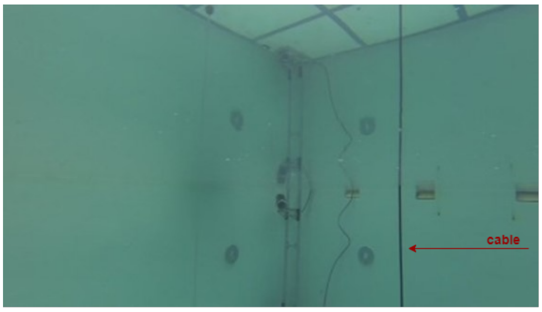


Figure 3.2: Video frame taken in tank with test cable

For the optical images, the aim is to extract information about the cable, where it is in relation to the AUV, angle α , and its orientation, θ, as can be seen in Figures 3.3 and 3.4, through line extraction algorithms.

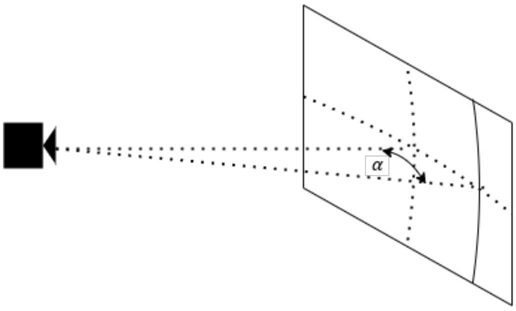


Figure 3.3: Horizontal angle relative to the line

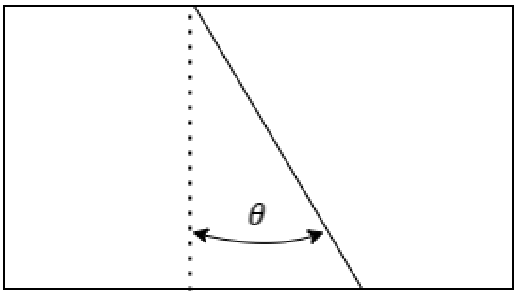


Figure 3.4: Line orientation in camera's frame

The imaging sonar provides an array of intensities corresponding to one beam, and consequently to a line in the acoustic image, when using polar coordinates, left image in 3.5. The arches observable in the left image or the square-shape in the right image (image in Cartesian coordinates), represent the water tank walls and the intersection of these (its corners). These contrasting shapes are due to the nature of the sonar data, since every detected reflection is represented by a detection angle and a detection distance.

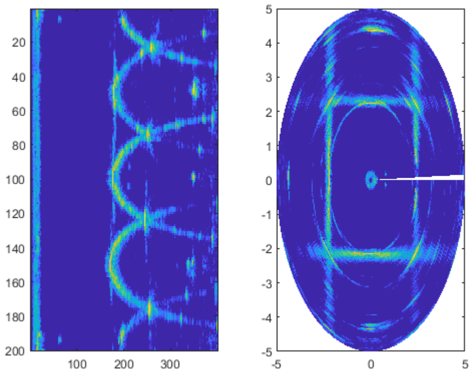


Figure 3.5: Imaging sonar revolution in test tank

In the acoustic image in polar coordinates, it is possible to obtain the feature's position in the image, i.e., the range, ρ, and angle, ϕ, in the horizontal sonar plane, as Figure 3.6 shows, through feature extractor algorithms or taking advantage of knowing the feature's appearance and apply template matching.

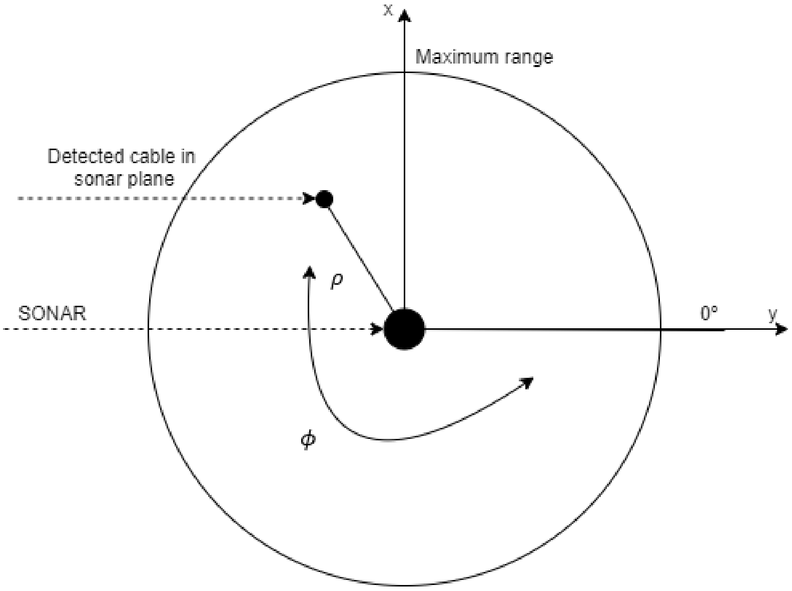


Figure 3.6: Sonar plane

3.3.2 Data processing

After the images have been extracted, they are processed in order to extract the relevant information present in each image. Relevant information is usually referred to as features, which can be defined as a prominent or distinctive part or characteristic in the image, which can be an isolated point, a zone with a big variation in comparison to the rest, among others. Through the algorithms studied in chapter 2, hereafter are those that will be used during the development of this solution.

The application scenario mentions the need to localize a cable in a water column. So, the vision process will deal with the problem through line detection algorithms. Given the good performance of Hough line transform, mentioned in chapter 2, in extracting lines from images, this algorithm is used in the feature detection of the vision process. In complex environments, with various structures involving the mooring, it is expected to have many lines detected by the Hough algorithm. Therefore, as the cable, showed in Figure 3.2, stands out for its color in the surroundings, the Contour Tracing algorithm will also be explored due to its capability to follow the pixels that are different from its neighborhood. It is worth mentioning, that this algorithm is explored in the attempt to improve the Hough transform results.

To process the acoustic images, the feature extraction algorithms performance will be assessed since they detect abstract features, such as corners and blobs. As the cable is a specific feature unlike the features from the previous algorithms, template matching will also be explored.

3.3.3 Sensor Fusion

After processing the data coming from each of the sensors, the result will be processed asynchronously, as they have different acquisition and processing rates, through a filter and the cable's position, as well as the orientation, will be estimated.

The filter's estimation will allow the removal of outliers within the processed data. For instance, if the line extraction returns several results, with the cable's position estimation it will be possible to exclude within all the returned lines the ones that are far from the estimation.

Through the sensor fusion's feedback, it would also be possible to change sensor parameters such as the sector scan, restricting it to the interest zone. This automatic adjustment would be innovative and would spare the microprocessor.

3.4 Requirements

After structuring the proposed solution, it is necessary to establish some requirements according to the problem and according to the imposed sensors.

Being a real-time solution, to operate, for instance, 10 Hz control system, it would be favorable to have, at least, 1 or 2 processing results in each control cycle. This means that, in the case of the optical sensor, the processing algorithm must execute and return a result, at most, in 100 ms or 50ms, respectively. The sonar acquisition rate is much lower, as a revolution of a beam takes approximately 40ms. Thus, it would be desirable that during each acquisition, the feature extraction algorithm could finish processing the data present in an acoustic image line.

In addition to the defined time requirement, an operational requirement is defined. It was assumed that this solution will be used in a scenario with relative visibility, that is, where the vision sensor has visibility at least up to 3m, an estimated value.

3.5 Tools

In order to implement the proposed solution, MATLAB R2021a software was initially used, along with functions and tools available in the Image Processing Toolbox, for image preparation and feature detection, in order to apply feature extraction algorithms. After this phase, all methods were implemented in C ++, for being a high-performance language, along with the OpenCV library, and all developed methods and algorithms were tested using a Raspberry Pi 3 Model B+.

For the given problem, there is a predefined vehicle for testing , Figure 3.7. In specific, an AUV with 0.12 m of diameter and 0.90 m of length combined with 6.0 kg of weight. The vehicle has a torpedo-like geometry. Its geometry and dimensions make it a vehicle easy to deploy and operate. The thrusters' configuration grant it four degrees of motion freedom. [45]

In the development of this thesis, the experimental part was carried out in in a rectangle-shaped tank with dimensions 4.6 m (length) x 4.4 m (width) x 1.8 m (depth).

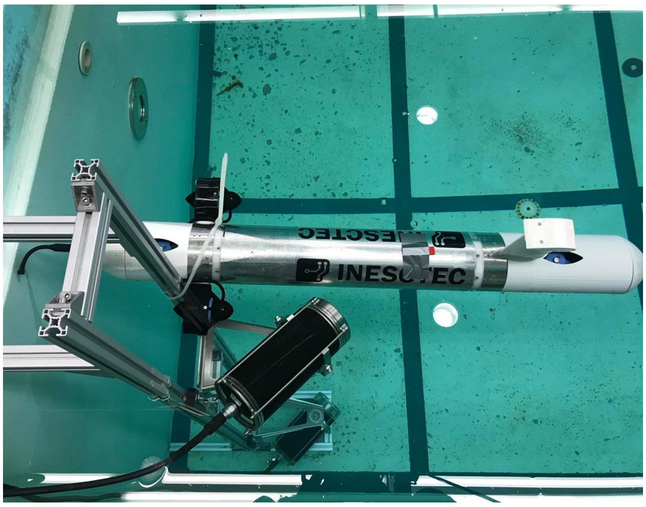


Figure 3.7: Vehicle in test tank

Chapter 4

Optical Image Processing

4.1 Tools and Methodology

In Figure 4.1, the methodology used for extracting the necessary information from the optical sensor is presented and detailed here and in the next sections. This procedure will provide two angles, the line orientation angle, θ and the horizontal angle relative to the cable, α. This information will then feed the position estimator and improve the cable position estimation.

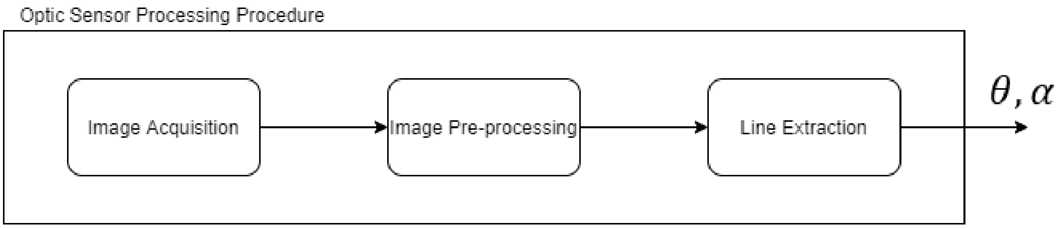


Figure 4.1: Optic Image Processing Procedure

4.2 Image Acquisition

The image resolution is an important configuration in image processing projects being the amount of detail an image holds. Therefore, it will have a great impact in processing time of the algorithm itself. Being that this thesis focuses on localization for navigation, time becomes a challenge.

With the purpose of achieving better results in terms of time, a study of the compromise between resolution reduction and loss of crucial information was made.

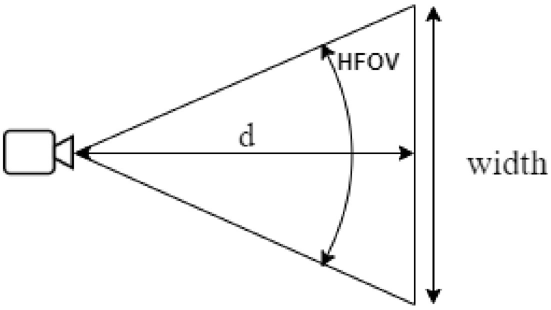


Figure 4.2: Horizontal Field of View Representation

Knowing the distance *d* and the horizontal field of view angle *HFOV*, the following expression, 4.1 can be obtained from trigonometry.

$$width = 2 \cdot d \cdot \tan \frac{HFOV}{2} \tag{4.1}$$

As mentioned, for this part a GoPro Hero 3 was used in video mode. In this mode, some settings are configurable, for instance video resolution and the field of view. So, for this camera and the stated settings, the horizontal field of view angle, represented in the Figure 4.2 as *HFOV*, is 64.4°.

It was defined that in order to use the proposed feature extraction methods, the cable must represent at least 1 pixel at any distance within the established maximum distance, represented in the Figure 4.2 by *d*, which in the requirements was defined to be 3 meters. So, for this specific case, the frame's width represents 3.78 meters. Therefore, for the 1 cm wide cable to correspond to 1 pixel, the frame's width in pixels has to be at least 378px. For 360p (640x360) resolution, the cable represents 1,69 pixels. This way it is possible to lower the processing time while still being able to identify the cable.

4.3 Image Pre-processing

Prior to the analysis of each frame, represented in Figure 4.3, a pre-processing step is required. This step is crucial to the effectiveness of this algorithm and will greatly impact the overall performance of subsequent operations.

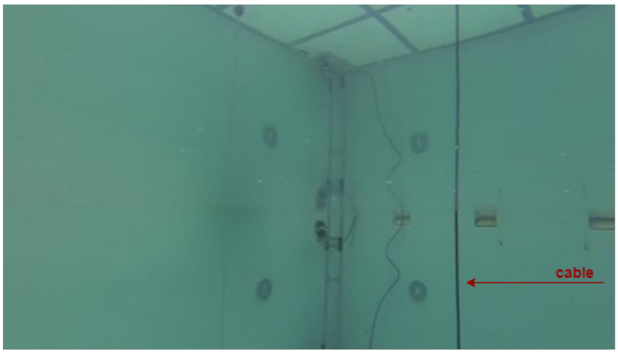


Figure 4.3: Frame example from Go Pro video

Due to the computational weight of processing RGB images, the image was converted to grayscale. Then, in order to prepare the image to the edge detection algorithm, the image is blurred due to a general problem for edge detection which is its sensitivity to noise.

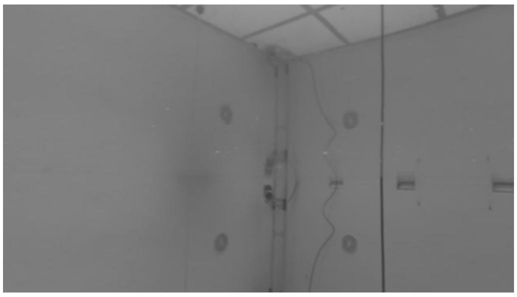


Figure 4.4: Blurring Result

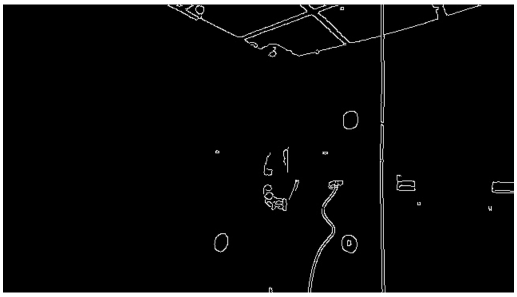


Figure 4.5: Canny Edge Detector Result

4.4 Line Extraction Algorithms

After the removal of irrelevant details in the image, it is possible to focus on the important information present in the image. From the methods explained hereafter it is expected to obtain the relative position in angle of the cable and its orientation.

Hough transform is theoretically capable of providing this information. However, for the purposes of this thesis, Contour Tracking is considered as an intermediate step for the line detection, meaning that it was explored with the goal of complementing the other algorithms.

4.4.1 Hough Transform

Since the final goal is to detect lines, more specifically a cable suspended in a water column, Hough Line Transform should be the most suitable.

The basic theory of the Hough line transform, explained in [46], is that any point in a binary image could be part of some set of possible lines. If we parameterize each line by, for example, a slope **a** and an intercept **b**, then a point in the original image is transformed to a locus of points in the (a, b) plane corresponding to all of the lines passing through that point.

If every nonzero pixel is converted in the input image into such a set of points in the output image and such contributions are summed, then lines that appear in the input (i.e., (x, y) plane) image will appear as local maxima in the output (i.e., (a, b) plane) image. Because we are summing the contributions from each point, the (a, b) plane is commonly called the accumulator plane.

Note that the slope-intercept form is not really the best way to represent all the lines passing through a point, because of the considerably different density of lines as a function of the slope, and the related fact that the interval of possible slopes goes from –∞ to +∞. It is for this reason that the actual parameterization of the transform image used in numerical computation is somewhat different. The preferred parameterization represents each line as a point in polar coordinates (ρ, θ), with the implied line being the line passing through the indicated point but perpendicular to the radial from the origin to that point. The equation for such a line is:

$$\rho = x\cos\theta + y\sin\theta \tag{4.2}$$

The OpenCV Hough transform algorithm does not make this computation explicit to the user. Instead, it simply returns the local maxima in the (ρ, θ) plane. However, it is necessary to understand this process to understand the arguments to the OpenCV Hough line transform function.

HoughLines(InputArray image, OutputArray lines, double rho, double theta, int threshold, double srn = 0, double stn = 0, double mintheta = 0, double maxtheta = π)

The first argument is the input image. It must be an 8-bit image, but the input is treated as binary information. The second argument is a pointer to a place where the results can be stored, which can be either a memory storage or a plain N-by-1 matrix array. The next two arguments, rho and theta, set the resolution desired for the lines (i.e., the resolution of the accumulator plane). The units of rho are pixels, and the units of theta are radians; thus, the accumulator plane can be thought of as a two-dimensional histogram with cells of dimension rho pixels by theta radians. The threshold value is the value in the accumulator plane that must be reached for the routine to report a line.

In order to improve the processing time and avoid overlapping lines, the parameters of rho and theta were configured after a study of their influence on the final result. Rho, as mentioned before, is the distance resolution of the accumulator, i.e., is the size of the accumulator array for all incoming rho. For rho, the default value was set and proved to have the best outcome. Theta defined as the angle resolution of the accumulator in radians, or in other words, the size of the accumulator array for all incoming theta. For theta, a brief analysis was made in order to decrease the total number of detected lines but always making sure that the cable was detected, and, given that it was detected, increase the similarity between the detected orientation and the actual cable orientation. Table 4.1 displays, for different values of theta, the number of frames in which the algorithm detected the cable and it was present (true positives), and the number of frames where the orientation of the Hough line corresponds to the cable orientation. The decision whether or not the cable was detected and whether the orientation corresponds to the orientation of the cable was made by visual inspection.

Accumulator's Angle Resolution	Number of frames (with the cable) in which the cable is detected	Number of frames in which the cable is detected, and the line orientation is correct	Average of total number of lines detected in a frame
1º	10/10	10/10	272
2º	10/10	10/10	143
4º	10/10	10/10	76
6º	10/10	10/10	52
9º	10/10	9/10	35
10º	10/10	10/10	35
12º	9/10	8/9	28
15º	9/10	5/9	19
18º	9/10	3/9	19
20º	10/10	2/10	19
30º	0/10	-	19

Table 4.1: Resolution angle for Hough Transform

As the aim was to reduce the total number of lines in each frame, but at the same time still detect the cable whenever it is present, the resolution angle was set to 4º.

Moving to the fifth parameter, threshold, all the data was analysed, and the threshold was configured to reduce the total number of detections but always detect the cable.

Since we are looking for vertical lines, the search for it using the *HoughLines* function was restricted through the min_theta and max_theta angles. As we are working in the aquatic environment, the cable may have a certain inclination, that is, not be completely vertical. Thus, a angle interval in the orientation of the line detected was considered for the configuration of these parameters, from -18º to 18º.

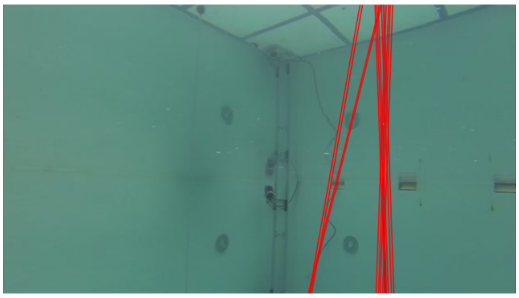


Figure 4.6: Detected lines with Hough Line Transform

The result of the Hough algorithm for this frame in particular is thirteen lines, each one of these lines will return two angles – the orientation of the line itself and the horizontal angle from the camera relative to the center point of the line, as showed in Table 4.2.

Line	Line orientation angle	Relative horizontal angle
1	2º	16º
2	-2º	15º
3	-2º	15º
4	2º	16º
5	-2º	15º
6	2º	16º
7	-2º	14º
8	2º	16º
9	10º	11º
10	14º	13º
11	-2º	15º
12	14º	12º
13	10º	11º

Table 4.2: Result from Optic Image Process

One of these values, filtered by a method explained afterwards, will feed the position estimator, and through the feedback of the position estimator it is possible to remove the outliers between the different values that this algorithm returns.

To assess if this algorithm meets the time requirement of a real-time solution, the run time was measured for each frame, and is presented in Figure 4.7.

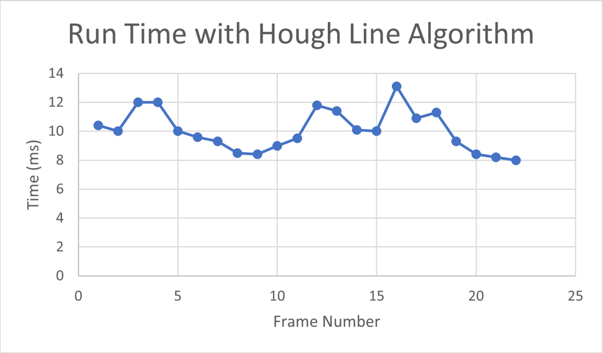


Figure 4.7: Run time of Hough Line Transform algorithm for a set of video frames

Through this graphic, it is easy to perceive that all frames are processed in less than 14 ms, which is bellow the time interval it was set as our goal in chapter 3.

4.4.2 Contour Tracking

The possibility of using contour following to find the cable was analysed. For this, the image was subjected to the pre-processing already described and then the function *findContours()*, from the OpenCV library, was used.

Contours can be explained simply as a curve joining all the continuous points along a boundary, having same colour or intensity. The contours are a useful tool for shape analysis and object detection and recognition, as mentioned in chapter 2.

This function computes contours from binary images. It can take images created by the Canny edge detection, which have edge pixels in them, or images created by functions in which the edges are implicit as boundaries between positive and negative regions, like threshold or adaptive threshold.

After the preparation of the image the contours were found using the OpenCV function, *findContours()*, which returns the detected contours. For each contour found, they are drawn in the Canny result image, Figure 4.8.

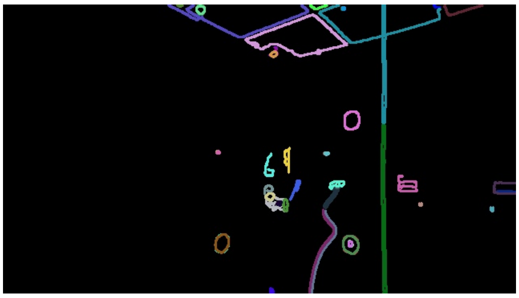


Figure 4.8: Contour tracking algorithm result

In Figure 4.8 is observable that this algorithm does not define the cable with only one contour but two. This may be caused by a potential gap in the definition of the line. This discontinuity may be related to the suspended particles in the water. The next step is to apply an algorithm that finds the cable. In order to assess its viability to complement the other algorithms, the run time was measured in a Raspberry Pi 3 Model B+, for a set of video frames and is displayed graphically in Figure 4.9.

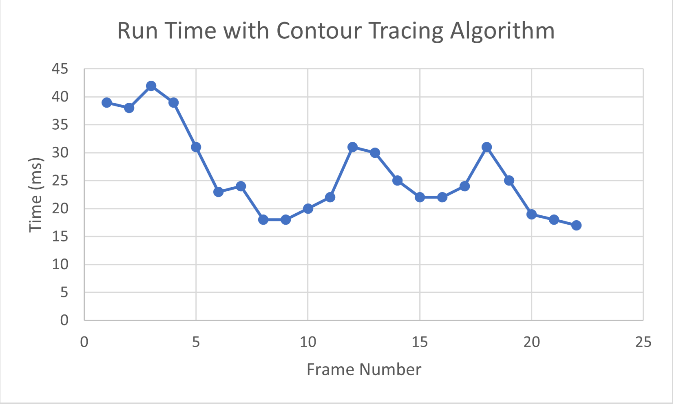


Figure 4.9: Run time of a contour tracking algorithm for a set of video frames

For the Contour Tracking method, through Figure 4.9 it is possible to conclude that it returns the contours present in an image and it meets the time requirements of this thesis.

To decide on this topic, further information is necessary. On one hand, whether it will improve the results when joined with an algorithm that can detect the feature itself. On the other hand, whether this merge will cause the run time to be higher than the required for a real-time solution.

In order to understand if Contour Tracking should be used prior to Hough Line Transform, toward a more robust optical algorithm, the two were applied to the frames. As illustrated in Figure 4.10, this was not the case, concluding that Hough Transform works better after a Canny operator than following the enhancement of the contours.

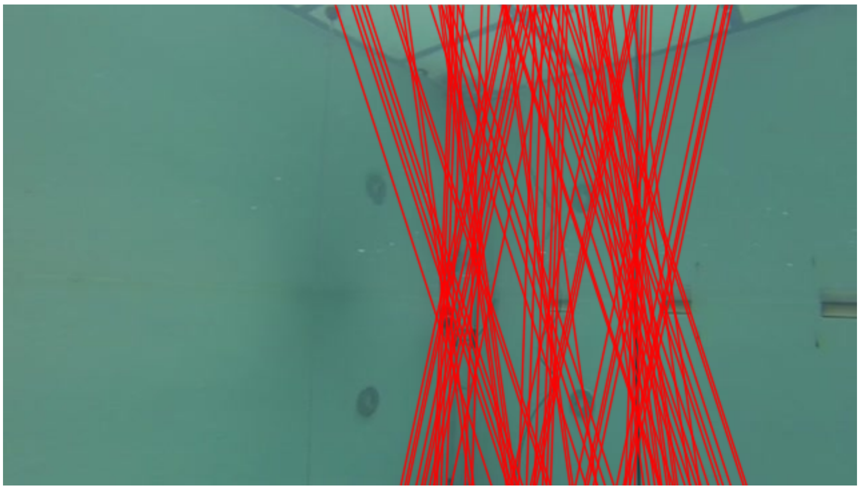


Figure 4.10: Result of Hough Line Transform after Contour Tracking algorithm

4.5 Conclusions

In this chapter, in order to obtain the relevant information present in an optic image, the behavior of each of the algorithms described was assessed.

With Hough Line Transform it was expected that it would not have any problem to detect the cable, but with a big cost in terms of time. However, this was not what was seen. Concluding that if the feature we are seeking is well defined, such as in this case, where the feature consists in a vertical line with certain limited orientation, this function's parameters can be properly defined and so it can process the image in short periods of time, and therefore can be used in real-time problems, such as localization and navigation.

Chapter 5

Acoustic Data Processing

5.1 Tools and Methodology

In Figure 5.1, the methodology used for extracting the necessary information from the acoustic sensor is presented and detailed hereafter, in the next sections. This procedure will provide a range, ρ, the distance to the cable's point in the sonar horizontal plane, and an angle, ϕ, the angle in the horizontal plane which the cable's point is detected. This information will then feed the position estimator and improve the state estimation.

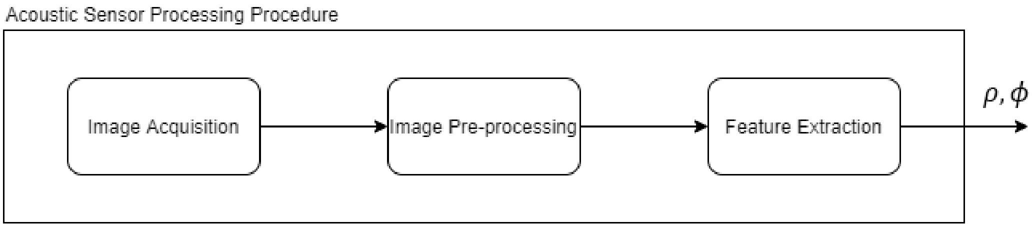


Figure 5.1: Acoustic image processing procedure

From the methods explained hereafter it is expected to obtain the relative position of the cable in polar coordinates. Both methods, the Feature Detectors Algorithms and Template Matching, are in theory capable of providing this information, however with different feature type assumptions. While the feature extractor algorithms search for an abstract feature, template matching looks for a specific pattern in the image. In the following subsections, an evaluation of various methods is made in order to select the best one.

5.2 Image Acquisition

The Tritech Micron sonar is connected to an on-board computer, providing an array of echo-intensities, which are usually represented in 8 bits and can be displayed as images. The Mechanical Scaning Sonar's (MSIS) head rotates and forms an overall acoustic image of the vehicle's surroundings. Figure 5.2 shows the result of a sonar scan provided by an MSIS in the middle of the test tank described in chapter 3. As it can be seen the way a MSIS maps the surroundings is by an array of bins characterized by a distance (range), an angle and the echo intensity. It can be represented directly in the polar coordinate system, on the left, or in cartesian coordinate system, on the right. Some of the readings are affected by multipath interference, multiple echoes, or reflections on multiple walls. For instance, in the image example, in polar coordinates the horizontal parabola-like formats correspond to the walls of the tank, which are easier to see in cartesian coordinates.

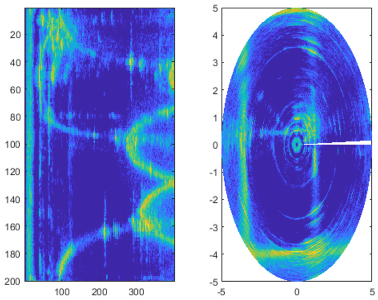


Figure 5.2: Sonar scan in the middle of the test tank

The intrinsic parameters of every sensor should be assessed carefully, specifically in this sonar there are three important configurable parameters – gain, range and step. The **gain** setting controls the sensitivity of the sonar receiver to compensate for water depth and water clarity. Increasing the gain shows more sensitivity, and decreasing the gain reduces screen clutter. The **range** setting will control the maximum radial distance between the sonar and the reflector. Increasing the distance allows to cover a larger area, however decreases the acquisition rate. The sonar's **step**, also called mechanical resolution step is the movement of the motor in terms of angle. In Table 5.1, the different parameters influence is showed in the respective acoustic image. After this assessment, the parameters used for all tests presented in this thesis were 0.5, 5m, and 1.8º for gain, range and step angle, respectively.

The cable has an associated pattern in the acoustic image, usually called signature. The cable pattern will be always the same independently of the point of view. This pattern has a comet-like form, a nucleus with high intensity and a tail of lower intensity pixels, demonstrated in Figure 5.4 which was cut out of a sonar scan in the test tank with the cable, Figure 5.3 is the complete scan without any processing.

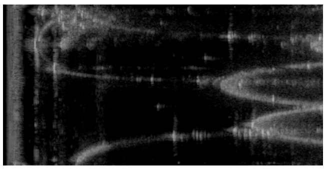
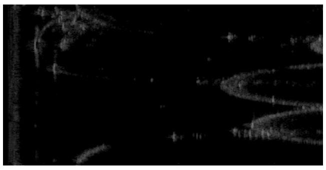
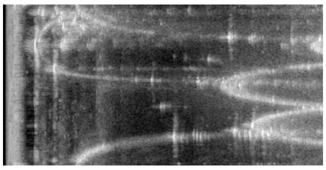
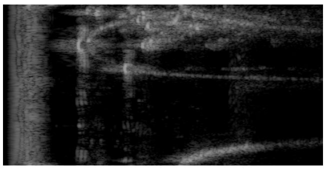
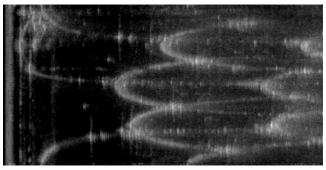
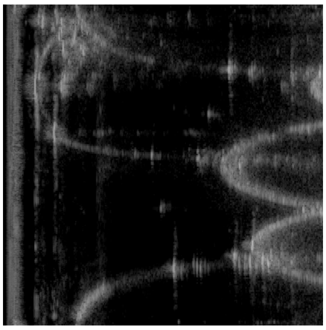
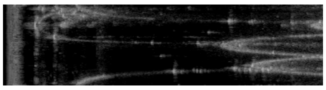
Gain	Range (m)	Step Angle	Result
0.5	5	1.8º	
0.1	5	1.8º	
1	5	1.8º	
0.5	**2**	1.8º	
0.5	**10**	1.8º	
0.5	5	**0.9º**	
0.5	5	**3.6º**	

Table 5.1: Study results of the different sonar parameters

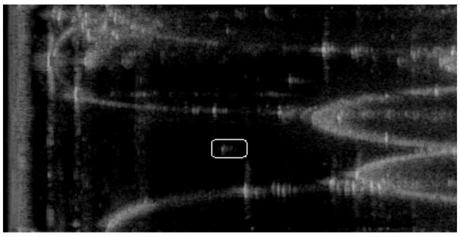


Figure 5.3: Sonar scan in test tank with different cables

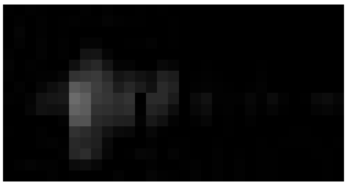


Figure 5.4: Cable pattern in acoustic image

This pattern directly depends on the position of the cable in the tank, this is because of the distance influence on the acoustic image echoes. Therefore, the distance will directly affect the echo in the pattern, which corresponds to the comet's tail. The higher the distance to the cable, the bigger the echo or the tail, as the "footprint" of the sonar beam increases with distance.

5.3 Feature Detectors Algorithms

Depending on the algorithm used, the type and overall shape of the feature varies.

Table 5.2 contains the algorithms application to the pre-processed image results, it shows that all corner and blob detectors detect the feature, except for SURF Algorithm, a blob detector. The cable's pattern is similar to a blob, i.e., a cluster of pixels that are distinguishable from its background. Therefore, it would be expected that the SURF algorithm, being a blob detector, would detect the cable in the acoustic image, just like the KAZE algorithm. The bibliography, referred in chapter 2, states that this algorithm applies filters of at least 9x9 to locate the points of interest. Since the cable's pattern is inferior than the filter that this method applies to the image, the cable is not detected.

For filtering purposes, the problem is not only the amount of correct and wrong detections, but also the distance at which the wrong ones are from the right ones. Therefore, to evaluate this measurement, a square centered on the feature with 1m of width , which in the image corresponds to approximately 93 pixels, was cut from the image and the algorithms that detected the cable were applied to it. The results of this analysis are shown in Table 5.3.

In order to understand which of these algorithms to use, a study was conducted on how many features each algorithm detects in a set of sonar revolutions, as well as its processing time.

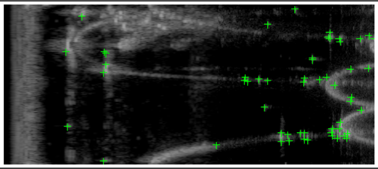
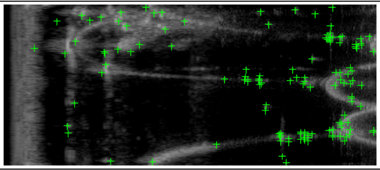
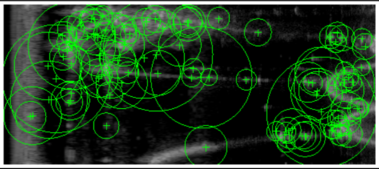
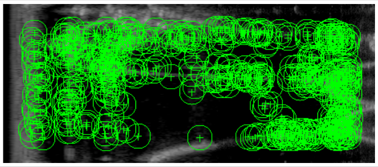
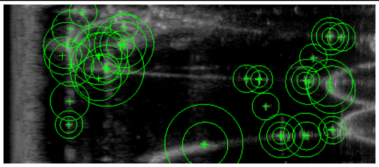
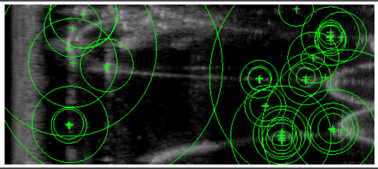
Algorithm	**Type of feature**	**Image Result**	**Number of Features Detected**	**Detected?**
Harris	Corners		54	Yes
FAST	Corners		126	Yes
SURF	Blobs		92	No
ORB	Corners		460	Yes
KAZE	Blobs		41	Yes
BRISK	Corners		40	Yes

Table 5.2: Feature extractors results in MATLAB R2021a software

Harris, in addition to detecting many features, has a very high variance which is an indicator that it is not reliable. ORB and KAZE are the best and relatively similar in terms of number of detected features, but they are also the slowest.

FAST and BRISK algorithms have the lowest mean of processing time of all, and it meets the imposed time requirement, 40 ms. However, BRISK's standard deviation of 6.6ms means it will take more than 40 ms, in some cases. Among these, the BRISK algorithm detects, on average, fewer features and tends to consistently detect a number of features around the average (51.4). As shown in Table 5.4, it would be easier to filter them since the number of detections reduces considerably in the area of interest. Therefore, with this dataset of acoustic revolutions, the BRISK algorithm would be the most suitable.

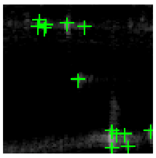
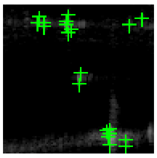
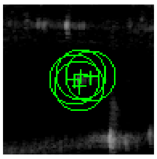
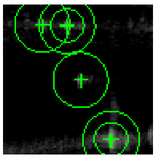
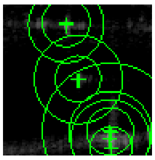
Algorithm	**Number of Features Detected in the original image**	**Image Result**	**Number of Features Detected**
Harris	54		13
FAST	126		20
ORB	460		4
KAZE	41		6
BRISK	40		9

Table 5.3: Feature extractors results in region of interest in MATLAB R2021a software

	Number of detected features		**Processing Time** (ms)	
Feature Detector Algorithm	Mean	Standard deviation	Mean	Standard deviation
Harris	62.7	15.4	56	8.7
FAST	87,7	8,9	22	1.6
ORB	39	3.4	339,1	45.2
KAZE	39.7	3.5	134	2
BRISK	51.4	3.8	33.1	6.6

Table 5.4: Mean and standard deviation of the number of detected features and processing time comparison in a data set of 20 acoustic images

5.4 Image Pre-Processing

Due to the distance influence in the feature to search for, and with the aim to always look for the same pattern, the image must be prepared for the extraction algorithms. It is known that the horizontal axis in the acoustic image in polar coordinates is the distance, so for different columns of the image the feature will differ. Pre-processing prevents searching for a different form in each column or set of columns. Searching for the same form in the entire image compared to dividing it in sets of columns and applying different methods to each one is computationally less expensive, and therefore was the path taken.

Not only the influence of the range is known, but also it is proportional to the size of the echo. As the feature pattern stretches with distance, the image needs to be resized in the opposite sence,so that the feature retains its geometrical characterstics (especially dimensions) over the entire image, stretched in the lowest distances and schrunk for the higher distances. With this purpose, the original image is divided into equal parts, resized, and then merged. For the used maximum range, 5 meters, it was considered that 3 parts was sufficient, but when increasing this value, the number of parts should increase as well.

For comparison purposes, the two images are presented hereafter, the original sonar scan, Figure 5.5, and the resized image, Figure 5.6.

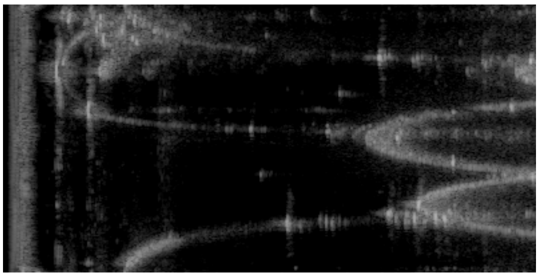


Figure 5.5: Original acoustic image

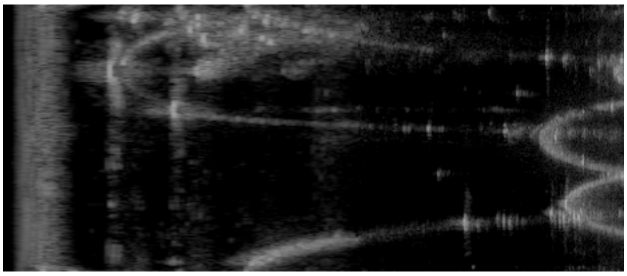


Figure 5.6: Acoustic image after pre-processing

5.5 Template Matching

Pattern matching in computer vision refers to a set of computational techniques which enable the localization of a template pattern in a sample image or signal. Such template pattern can be a specific facial feature, an object of known characteristics or a speech pattern such as a word.

The advantage is knowing what is expected to see in the acoustic image when the cable is present. This knowledge allows the construction of a template. For this step, there are two options, cutting out the pattern directly off the image, Figure 5.7, or building the expected pattern.

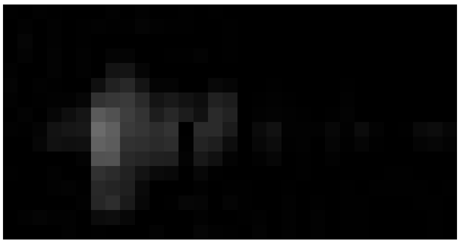


Figure 5.7: Template cropped of the original acoustic image

As described earlier, the pattern has a high intensity pixel which corresponds to the first reflection, that is, the direct or shortest path. Because of the horizontal beam opening there is vertical dispersion, lower intensity pixels, in the image. On the horizontal axis, there is also dispersion caused by the reflections of the spherical section formed by the beams. Knowing this, the image was built with OpenCV, and it is presented in Figure 5.8.

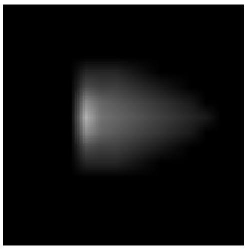


Figure 5.8: Expected pattern

The expected pattern was created with 16 lines to have a black background, as the cable will always be detached from the walls and consequently, the pattern is detached from other reflections corresponding to the walls.

The OpenCV template matching function, *matchTemplate()*, simply slides the template image and compares the template and patch of input image under the template image. Several comparison methods are implemented in OpenCV, as described in chapter 2.2.2. The most used methods are the square difference and the normalized square difference or the normalized correlation coefficient. One of them is not inherently better than the others, it really depends on the imagery and the template. In general, if the implementation requires exact or very close to exact matches, the square difference method will be a good choice since it is fast, and it directly maps the difference between the template and image patch. In that case, there is no need to normalize, it is just added overhead. If the requirements are similar but there will be used multiple templates to be comparable, then the normalized square difference is a good choice. Lastly, when working with real-world data that may have exposure or contrast differences, the mean shifting and variance equalization from the normalized correlation coefficient will likely be the best choice. Therefore, for this implementation the normalized correlation coefficient comparison method was used.

The template matching algorithm detects the pattern if the matched image has a similiraty value superior to 0.8. After implementing the algorithm for both templates, the performance of each in a set of acoustic revolutions was assessed, Table 5.5.

	Template cropped from the original acoustic image			Template constructed from expected pattern		
Sonar Revolution	**Detected?**	**Normalized Correlation Coefficient [0-1]**	**Processing Time (ms)**	**Detected?**	**Normalized Correlation Coefficient [0-1]**	**Processing Time (ms)**
1	Yes	0.99	34	Yes	0.89	34
2	Yes	0.99	33	Yes	0.89	33
3	Yes	0.84	34	No	-	-
4	Yes	0.86	33	No	-	-
5	Yes	0.85	33	No	-	-
6	Yes	0.87	34	Yes	0.87	34
7	Yes	0.87	32	Yes	0.86	32

Table 5.5: Comparison between templates

Since the template developed using the expected pattern was not able to correctly detect the feature in some cases, the template cropped of the acoustic image was the one used for the following steps.

	Complete Image Analysis			Iterative Analysis		
Sonar Revolution	**Detected?**	**Similarity Coefficient [0-1]**	**Processing Time (ms)**	**Detected?**	**Similarity Coefficient [0-1]**	**Processing Time (ms)**
1	Yes	0.99	34	Yes	0.99	1.9
2	Yes	0.99	33	Yes	0.99	1.8
3	Yes	0.84	34	Yes	0.84	2.1
4	Yes	0.86	33	Yes	0.86	1.7
5	Yes	0.85	33	Yes	0.85	1.9
6	Yes	0.87	34	Yes	0.87	2.6
7	Yes	0.87	32	Yes	0.87	1.8

Table 5.6: Comparison between the complete image's analysis and the analysis every 16 lines

In order to improve the processing time of the template matching algorithm, and not wait for the full 360º sonar revolution, which takes 8s, the image was analyzed every 16 lines, as soon as a new array of intensities (one line of the image) arrives, the oldest line is discarded. This approach is done through a stack of lines, and when the new line is inserted in the stack the algorithm is applied. These two analysis methods are compared in Table 5.6. As the intensity peaking point, the pixel that represents the position of the cable in the image, is in the center of the template, whenever it exists in the image the template is only detected when the next 8 lines are returned. This method imposes a delay. Another option was to define the pattern as just a single line, the line that contains the maximum point. This option would have a faster processing; however, it would not be able to determine if this peak would have vertical continuity, which would cause an increase in false positive detections. Yet another option would be to set the template as the previous 8 lines to the peak intensity line, i.e., the first 8 lines from the default template, but it would have the same trade-off.

Figure 5.9: Example of 16 line segment image for iterative analysis

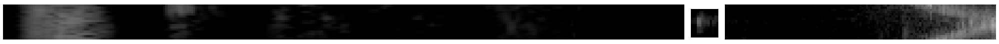


Figure 5.10: Feature detection in iterative analysis

After the pattern has been found in the image, it is possible through the strongest pixel location to return the requested information, the polar coordinates of the cable.

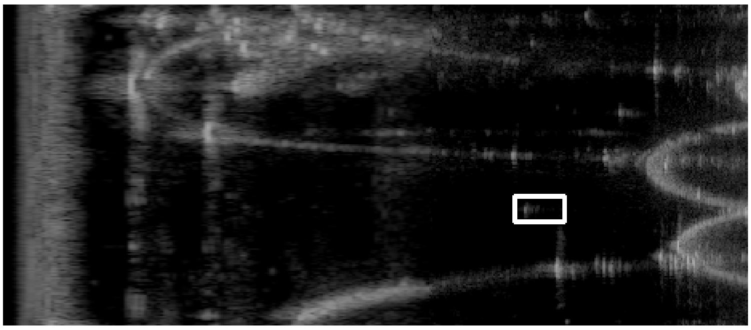


Figure 5.11: Template matching result

In the sonar image in Figure 5.11, and according to the template matching algorithm, the cable's coordinates in the acoustic image are (193,128). Which means that the cable, in the horizontal sonar plane, is at 2.41 m of distance and at 230º angle, as showed in the illustrative image presented in Figure 5.12. This conversion is made since the image has 200 lines, 399 columns and the maximum range was set to 5m.

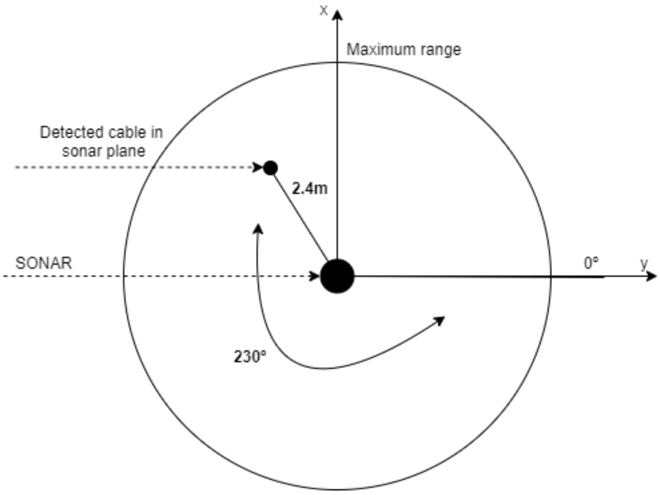


Figure 5.12: Illustrative image of sonar plane with detected cable

This measurement allows to feed and update the position estimator and improve its estimation.

5.6 Conclusions

In the previous sections, the behaviour of both methods was assessed, feature detectors and template matching, in order to obtain the relevant information present in an acoustic image.

While the template matching algorithm only returns a result with a high degree of certainty, feature detectors return dozens of results, which makes it difficult to filter out false positives. To use feature detectors, a filtering process would then have to exist to feed the state estimator with a single value, the image segment with the highest similarity value.

Not forgetting that a temporal requirement was imposed, that the acoustic processing algorithm returns a result, at least, during the acquisition time of an acoustic image line, 40 ms. For this reason, the temporal analysis of each of the algorithms was carried out, Figure 5.13. It is worth noting that, in this case, the full image was used and not the iterative analysis, because while template matching is able to return no features in cases where the feature is not present, the feature detectors always return a large number of features, and even if the outlier removal could be done in the entire image, repeating the process for every segmentation of the image would potentially be computational expensive.

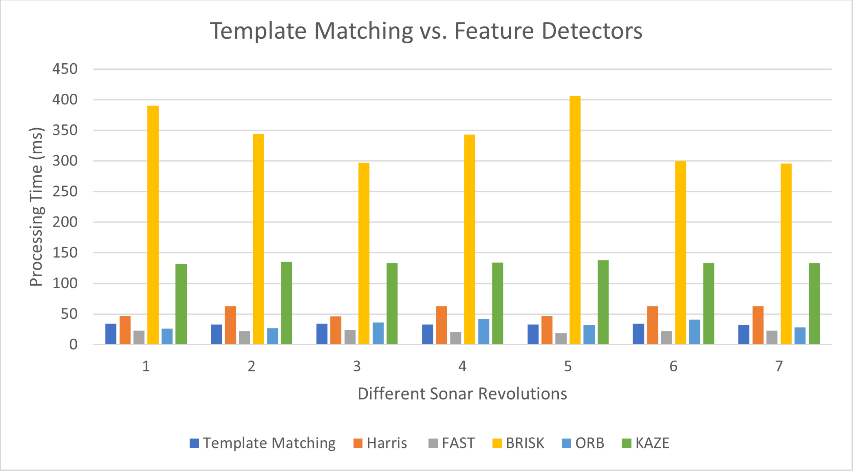


Figure 5.13: Template matching and feature detectors algorithm comparison

For the stated problem, the template matching algorithm is the ideal approach, since it greatly reduces the number of detections to a single result, and fulfills the time requirements, however it is a very specific method. To extend the solution's application area, it would have to know all the existing moorings and their corresponding patterns or adopt more generic algorithms, such as feature detectors.

Chapter 6

Sensor Fusion

As mentioned throughout this document, the aim of the analysis of the acquired optical and acoustic images is to detect the cable and feed this information to the developed state estimator, as represented by the block diagram in Figure 6.1.

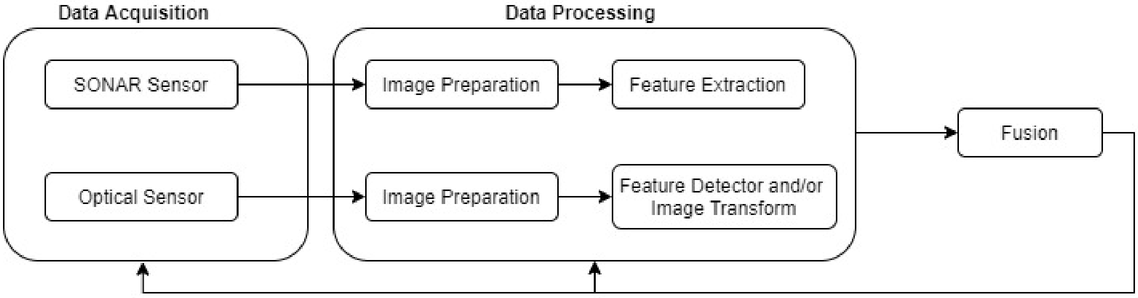


Figure 6.1: Developed solution in block diagram

6.1 System Model

To analyze the system model, the vision and acoustic model have to be studied independently.

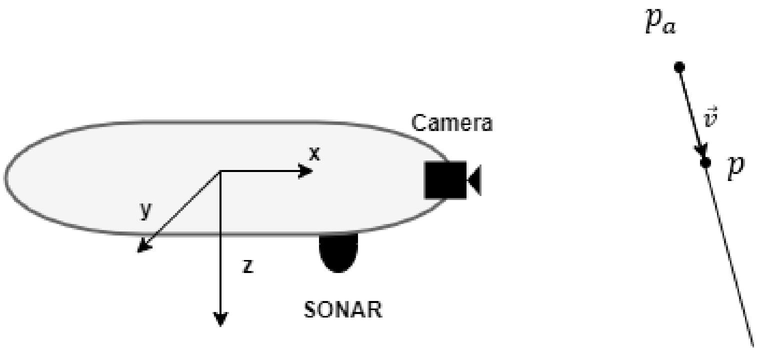


Figure 6.2: Cable seen by AUV's sensors

Figure 6.2 shows the cable seen by the two sensors, optical and acoustic. The cable is defined by the point pa, the point of the cable at the water's surface, and a vector $\overrightarrow{v}$ that defines the cable's orientation. The point p is a random point belonging to the line that represents the sum of the point pa with the vector $\overrightarrow{v}$.

$$p_I = p_a + \overrightarrow{v} \tag{6.1}$$

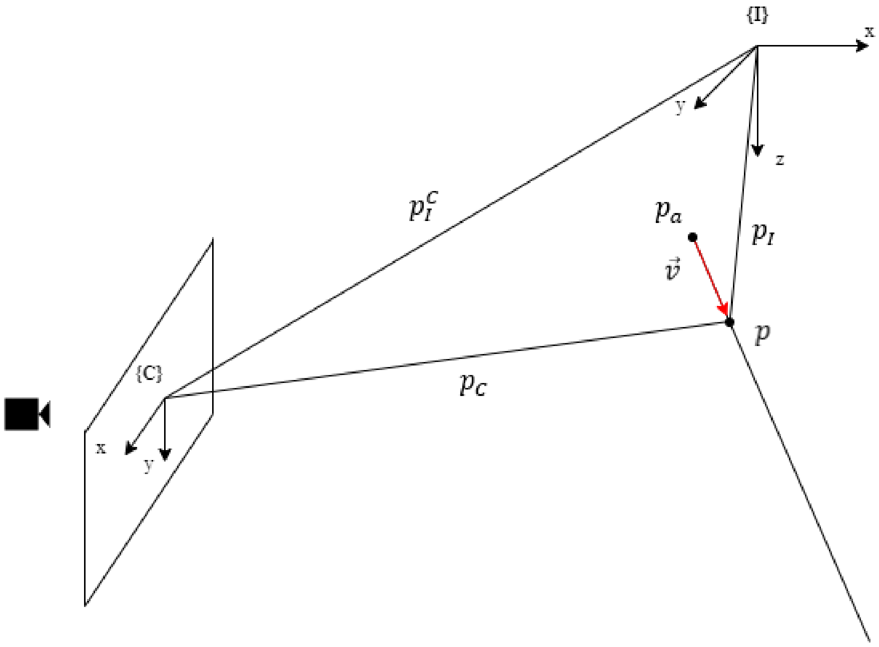


Figure 6.3: Camera's perspective of the cable

The vector $\overrightarrow{v}$ consists of 3 components, v_x, v_y and v_z, where v_x and v_y represent the vector's components in the respective axis and v_z is the position of the vehicle on the z axis in relation to the inertial frame.

To analyze the cable in the camera frame, C, Figure 6.3, it is necessary to calculate the point p in the camera frame. The cable will be projected in the plane of the camera image, this representation is made through the projection matrix P_C.

$$P_C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.2}$$

Let $R_{I \to C}$ be the rotation matrix from the inertial frame to the camera's frame. Through the previous figure, becomes clear that the point p in the camera's coordinate system will be:

$$p_c = P_C \cdot R_{I \to C} \cdot (p_C^I + p_I) \tag{6.3}$$

$$= P_C \cdot R_{I \to C} \cdot (p_C^I + p_a + \overrightarrow{v}) \tag{6.4}$$

$$= P_C \cdot R_{I \to C} \cdot p_C^I + P_C \cdot R_{I \to C} \cdot p_a + P_C \cdot R_{I \to C} \cdot \overrightarrow{v} \tag{6.5}$$

Vector $\overrightarrow{v}$ seen by the camera, can be defined in the camera's frame, $\overrightarrow{v}_c$ through:

$$\overrightarrow{v}_c = P_C \cdot R_{I \to C} \cdot \overrightarrow{v} \tag{6.6}$$

Through the Equation 6.3, that defines p in the camera frame, it is possible to define the angles provided by the optical image processing, α and θ, through p_c.

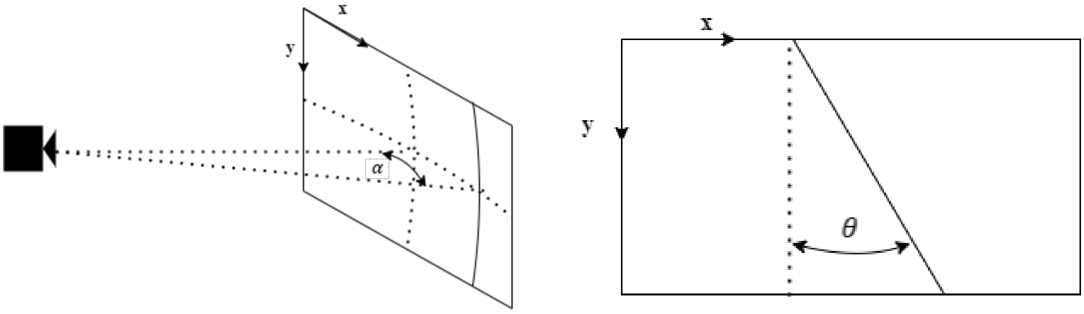


Figure 6.4: Output from optical image processing - α and θ

$$\begin{bmatrix} \alpha \\ \theta \end{bmatrix} = \begin{bmatrix} \frac{HFOV}{2} \cdot (p_{c_x} - \frac{width}{2}) \\ atan(\frac{\overrightarrow{v}_c(1)}{\overrightarrow{v}_c(0)}) \end{bmatrix} \tag{6.7}$$

Where **HFOV** represents the horizontal field of view configured by the camera, in this case 64.4º, and **width** represents the frame width, in pixels, 640px.

Similarly for the acoustic sensor, Figure 6.5, to analyze the cable in the sonar frame, S, it is necessary to calculate the point p in the corresponding frame. The cable will be projected in the horizontal plane of the sonar, this representation is made through the projection matrix P_S, equation 6.8.

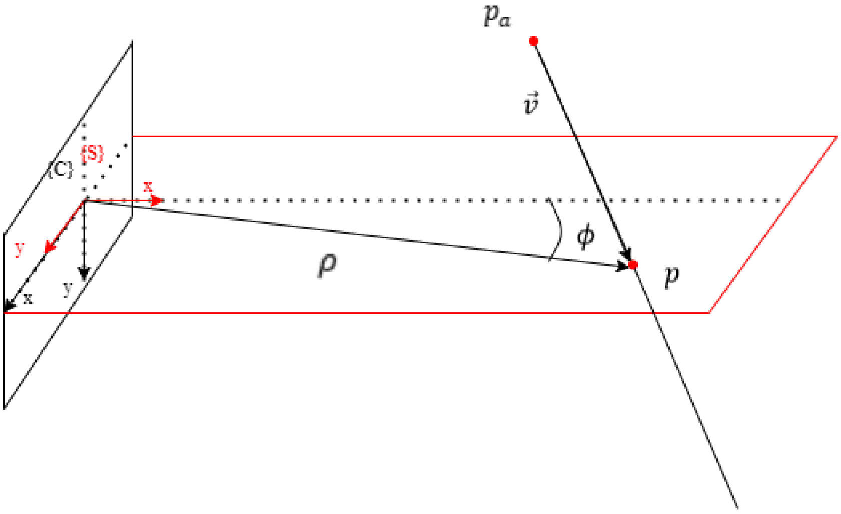


Figure 6.5: Sonar's perspective of the cable

$$P_S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{6.8}$$

Calculating p in the sonar frame as before:

$$\begin{aligned} p_s &= P_S \cdot R_{I \to S} \cdot (p_S^I + p_I) && (6.9) \\ &= P_S \cdot R_{I \to S} \cdot (p_S^I + p_a + \overrightarrow{v}) && (6.10) \\ &= P_S \cdot R_{I \to S} \cdot p_S^I + P_S \cdot R_{I \to S} \cdot p_a + P_S \cdot R_{I \to S} \cdot \overrightarrow{v} && (6.11) \end{aligned}$$

Where $R_{I \to S}$ represents the rotation matrix from the inertial frame to the sonar's frame. Through the expression that defines p in the sonar frame, it is possible to define the data provided by the acoustic image processing, ρ and ϕ, through p_s, Equation 6.9.

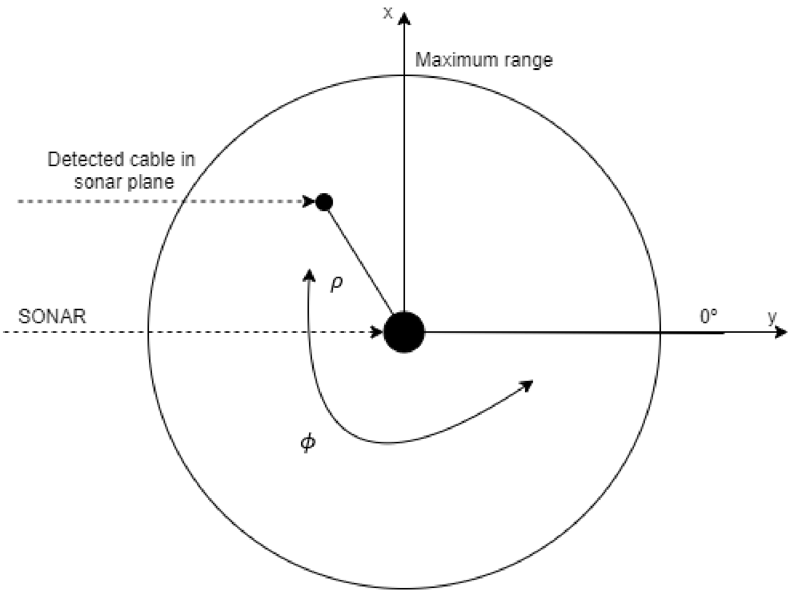


Figure 6.6: Output from acoustic image processing - ρ and ϕ

$$\begin{bmatrix} \rho \\ \phi \end{bmatrix} = \begin{bmatrix} ||p_s|| \\ atan(\frac{p_{s_x}}{p_{s_y}}) \end{bmatrix} \tag{6.12}$$

Considering that the origin of the inertial frame is the initial position of the vehicle, and that the camera and sonar frames are centered on the center of mass of the vehicle itself, the rotation matrix from the inertial to the camera and to the sonar will be the yaw, pitch and roll rotation.

The yaw angle represents a counterclockwise rotation of ψ about the z-axis. The rotation matrix is given by:

$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.13}$$

The pitch angle is a counterclockwise rotation of ω about the y-axis. The rotation matrix is given by:

$$R_y(\omega) = \begin{bmatrix} \cos\omega & 0 & \sin\omega \\ 0 & 1 & 0 \\ \sin\omega & 0 & \cos\omega \end{bmatrix} \tag{6.14}$$

The roll angle is a counterclockwise rotation of γ about the x-axis. The rotation matrix is given by:

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{6.15}$$

So, a single rotation matrix can be formed by multiplying the yaw, pitch, and roll rotation matrices to obtain the rotation matrix, $R = R_{I \to C} = R_{I \to S}$.

$$R = R(\psi, \omega, \gamma) = R_z(\psi) R_y(\omega) R_x(\gamma) \tag{6.16}$$

$$= \begin{bmatrix} \cos\psi \cdot \cos\omega & \cos\psi \cdot \sin\omega \cdot \sin\gamma - \sin\psi \cdot \cos\gamma & \cos\psi \cdot \sin\omega \cdot \cos\gamma + \sin\psi \cdot \sin\gamma \\ \sin\psi \cdot \cos\omega & \sin\psi \cdot \sin\omega \cdot \sin\gamma + \cos\psi \cdot \cos\gamma & \sin\psi \cdot \sin\omega \cdot \cos\gamma - \cos\psi \cdot \sin\gamma \\ -\sin\omega & \cos\omega \cdot \sin\gamma & \cos\omega \cdot \cos\gamma \end{bmatrix} \tag{6.17}$$

The Euler angles (roll, pitch and yaw), and consequently the rotation matrix R, will be calculated by integrating the angular velocities provided by the vehicle's gyroscope.

6.2 State Estimator

The implemented state estimator follows the EKF algorithm presented initially in chapter 2, which makes use of new measurements provided by the acquired information from each sensor, and the previous state vector to estimate the current state vector $\widehat{X}$. This state vector, Equation 6.51 , besides the vehicle's pose, includes the cable's surface position and its orientation, through the components of the vector $\overrightarrow{v}$.

$$\widehat{X} = \begin{bmatrix} x \\ y \\ z \\ pa_x \\ pa_y \\ v_x \\ v_y \end{bmatrix} \tag{6.18}$$

EKF algorithm encompass two major procedures during state estimation, the prediction cycle, and the measurement update cycle. This section is focused in presenting further detail regarding the state estimation implementation and its computation.

6.2.1 Prediction Cycle

The prediction cycle takes place between the acquisition of two consecutive measurements. The predicted state is defined as:

$$X_t = f(\hat{X}_{t-1}, u_{t-1}, w_{t-1}) \tag{6.19}$$

During it, the motion model presented hereafter is used to predict the vehicle's pose.

$$x = x_0 + \int \dot{x} \cdot dt \tag{6.20}$$

$$y = y_0 + \int \dot{y} \cdot dt \tag{6.21}$$

$$z = z_0 + \int \dot{z} \cdot dt \tag{6.22}$$

Where x0, y0, and z0 represent the vehicle's initial x,y,z position, t the time reference and $\dot{x}$, $\dot{y}$ and $\dot{z}$ are the velocities in each axis, and can be expressed by the linear velocities u, v and w and the Euler angles, roll (γ), pitch (ω) and yaw (ψ).

$$\dot{x} = u \cdot (\cos\omega \cdot \cos\psi) + v \cdot (\sin\gamma \cdot \sin\omega \cdot \cos\psi - \cos\gamma \cdot \sin\psi) + w \cdot (\cos\gamma \cdot \sin\omega \cdot \cos\psi + \sin\gamma \cdot \sin\psi) \tag{6.23}$$

$$\dot{y} = u \cdot (\cos\omega \cdot \sin\psi) + v \cdot (\sin\gamma \cdot \sin\omega \cdot \sin\psi + \cos\gamma \cdot \cos\psi) + w \cdot (\cos\gamma \cdot \sin\omega \cdot \sin\psi - \sin\gamma \cdot \cos\psi) \tag{6.24}$$

$$\dot{z} = u \cdot (-\sin\omega) + v \cdot (\sin\gamma \cdot \cos\omega) + w \cdot (\cos\gamma \cdot \cos\omega) \tag{6.25}$$

As we do not have access to linear velocities, we consider these as null, that is, with error that follows a Gaussian distribution with null mean. In order to perform the proposed state prediction, the model introduced is further discretized as follows:

$$X_t = \begin{bmatrix} x_t \\ y_t \\ z_t \\ pa_{x_t} \\ pa_{y_t} \\ v_{x_t} \\ v_{y_t} \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ z_{t-1} \\ pa_{x_{t-1}} \\ pa_{y_{t-1}} \\ v_{x_{t-1}} \\ v_{y_{t-1}} \end{bmatrix} \tag{6.26}$$

Where X_t represent the state at the current iteration and x_{t-1}, y_{t-1} and z_{t-1} represent the vehicle's x and y position and z at the previous iteration, $p_{x_{t-1}}$, $p_{y_{t-1}}$, $v_{x_{t-1}}$and $v_{y_{t-1}}$ the position and the orientation of the cable at the previous iteration.

Taking this discretized model into consideration [41], the state prediction is performed accordingly. To conclude the prediction cycle, after projecting the state ahead, the error covariance matrix, P, needs to be computed, Equation 6.27.

$$P_t = A_t P_{t-1} A_t^T + W_t Q_{t-1} W_t^T \tag{6.27}$$

Being A_t and W_t the Jacobian matrix of partial derivatives of f, Equation 6.19 with respect to X, Equation 6.40, and with respect to w, equation 6.29, respectively, and Q_{t-1} the covariance matrix of the process noise.

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial X_{[j]}}(\hat{X}_{t-1}, u_{t-1}, w_{t-1}) \tag{6.28}$$

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{X}_{t-1}, u_{t-1}, w_{t-1}) \tag{6.29}$$

Since the prediction state only depends on the previous state, the Jacobian matrix of the f function results in the identity matrix. It is presumed that all state variables are influenced by the uncertainty and process modelling errors, which is expressed in the matrix W, the Jacobian of partial derivatives relative to the process noise vector. Thus, these matrices are defined by:

$$A = I_{7x7} \tag{6.30}$$

$$W = I_{7x7} \tag{6.31}$$

Q is the covariance matrix associated with the noise in the states. The covariance matrices must be acquired by considering the stochastic properties of the corresponding noises; however, these are usually not known, the covariance matrices are employed as tuning parameters.

6.2.2 Update Cycle

The purpose of the update cycle is to fuse the measurements $z_t = h(X_t, v_t)$, where h is the non-linear function that relates the state X_t to the measurement z_t, and where v_t is the measurement noise.

From the optical sensor and the acoustic sensor different measurements are provided. Each of these measures results in different information, as described throughout the document, and consequently in different functions of h.

The optical process returns two angles, the horizontal angle relative to the center point of the line found, α, and the angle that defines the orientation of that line, θ. Therefore, the z_{v_t} view measure is expressed by:

$$z_{v_t} = \begin{bmatrix} \alpha(X_t) \\ \theta(X_t) \end{bmatrix} \tag{6.32}$$

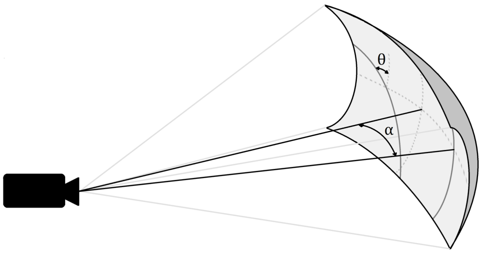


Figure 6.7: Illustration of horizontal angle relative to the frame's center axis and line orientation measurement

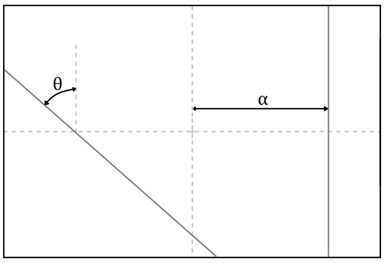


Figure 6.8: Frame as captured by the camera

And the the relation between these measurements and the state vector are defined hereafter.

$$\alpha(X_t) = \frac{HFOV}{2} \cdot ((\cos\gamma \cdot \cos\psi + \sin\gamma \cdot \sin\psi) \cdot (pa_y + v_y - y) + \cos\omega \cdot \sin\psi \cdot (pa_x + v_x - x) - 1) \tag{6.33}$$

$$\theta(X_t) = z \cdot \cos\omega \cdot \cos\gamma - v_x \cdot \sin\omega + v_y \cdot \cos\omega \cdot \sin\gamma \tag{6.34}$$

The acoustic process returns a point in polar coordinates, the distance, ρ, and the angle, ϕ. Therefore, the z_{s_t} view measure is expressed by:

$$z_{s_t} = \begin{bmatrix} \rho(X_t) \\ \phi(X_t) \end{bmatrix} \tag{6.35}$$

Where ρ and ϕ, were defined in the previous section, and the the relation between these measurements and the state vector are defined hereafter.

$$\rho(X_t) = (((\cos\gamma \cdot \cos\psi + \sin\omega \cdot \sin\gamma \cdot \sin\psi) \cdot (pa_y + v_y - y) + \cos\omega \cdot \sin\psi \cdot (pa_x + v_x - x))^2 \tag{6.36}$$

$$+((\cos\gamma \cdot \sin\psi - \cos\psi \cdot \sin\omega \cdot \sin\gamma) \cdot (pa_y + v_y - y) - \cos\omega \cdot \cos\psi \cdot (pa_x + v_x - x))^2)^{1/2} \tag{6.37}$$

$$\phi(X_t) = -atan(\frac{(\cos\gamma \cdot \cos\psi + \sin\omega \cdot \sin\gamma \cdot \sin\psi) \cdot (pa_y + v_y - y) + \cos\omega \cdot \sin\psi \cdot (pa_x + v_x - x)}{(\cos\gamma \cdot \sin\psi - \cos\psi \cdot \sin\omega \cdot \sin\gamma) \cdot (pa_y + v_y - y) - \cos\omega \cdot \cos\psi \cdot (pa_x + v_x - x)}) \tag{6.38}$$

In the same way that there is an expression of h for each of the sensors, there is the Jacobian matrix of partial derivatives of h with respect to the state vector for each sensor, H_v and H_s, as well as the Jacobian matrix of partial derivatives of h with respect to the respective measurement noise vector for each sensor, V_v and V_s.

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial X_{[j]}}(X_t, v_t) \tag{6.39}$$

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(X_t, v_t) \tag{6.40}$$

For the optical sensor, the Jacobian matrix of h with respect to the respective measurement noise vector is defined as:

$$V_v = I_{2x2} \tag{6.41}$$

For the acoustic sensor, the Jacobian matrix of h with respect to the respective measurement noise vector is defined as:

$$V_s = I_{2x2} \tag{6.42}$$

To update the prediction with the measurements of each sensor, the following equations are computed:

$$K_t = P_t H_t^T (H_t P_t H_t^T + V_t R_t V_t^T)^{-1} \tag{6.43}$$

$$\hat{X}_t = \hat{X}_t + K_t(z_t - h(X_t, v_t)) \tag{6.44}$$

$$P_t = (I - K_t H_t) P_t \tag{6.45}$$

Where K_t represents the Kalman gain, z_t the measurement returned by the sensor, $h(X_t, v_t))$ the measurement infered by the filter and R the covariance matrix of the measurement noise.

6.3 Results

In this section, a comprehensive analysis of the developed state estimator performance is presented, in light of the solution structure presented in Figure 6.1. Therefore, the information extracted from the data processing is fed into the develop state estimator.

For the filter's computation, the initial estimation of the state vector and the initial covariance, P, were set, as follows.

$$\widehat{X}_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ pa_{x_0} \\ pa_{y_0} \\ v_{x_0} \\ v_{y_0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.3 \\ 2.2 \\ 0 \\ 0 \end{bmatrix} \tag{6.46}$$

Assuming that the vehicle is at the origin of the inertial frame of reference and that the cable is located roughly in the center of the test tank with null orientation angle, i.e. vertically.

$$P_0 = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 10 \end{bmatrix} \tag{6.47}$$

The covariance matrices, Q and R, were defined as a function of the model's noise and the measurement's noise, and then tuned manually by trial and error methods.

$$Q = \begin{bmatrix} (0.001)^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (0.001)^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & (0.001)^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.001 \end{bmatrix} \tag{6.48}$$

The noise covariance matrices of the measurements of each one of the sensors were defined through their measurement error. By measuring the position of the cable in the test tank, it was possible to determine that it was at [2.2, 1.9] m. Thus, the sonar's measurement was ρ of 2.3 m and ϕ of 0.38 radians, and the camera's measurement was of α of 0.87 radians and θ of 0.1 radians. Through these measurements, the respective errors were calculated, and the matrices were defined as:

$$Rc = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.07 \end{bmatrix} \tag{6.49}$$

$$Rs = \begin{bmatrix} 0.65 & 0 \\ 0 & 0.48 \end{bmatrix} \tag{6.50}$$

Thus, to assess the performance of this last module, estimates of the vehicle position and cable location as a function of time were collected. The vehicle and cable's position estimations were collected.

First, the behavior of the state estimator with each sensor, individually, was evaluated, that is, only with optical sensor measurements and then only with acoustic sensor measurements. Secondly, the fusion of both systems was carried out, in a first scenario with the vehicle immobilized. All these tests were carried out in a square tank, 4.4m long along the x axis of the inertial frame and 4.6 wide along the y axis. Figure 6.9 shows the mooring, it was placed vertically at [2.2, 1.9] m and the vehicle was kept still throughout all data acquisition.

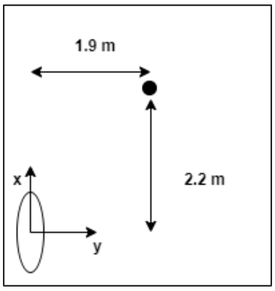


Figure 6.9: Cable's position in test tank

6.3.1 State Estimator Using Optical Data

The data acquired from the optical sensor is processed through the Hough transform, and results in multiple lines detected in the frame provided by the sensor. In order to feed the estimator, it is necessary to restrict these to a single result. Thus, an outlier removal algorithm was developed using the filter's previous estimate.

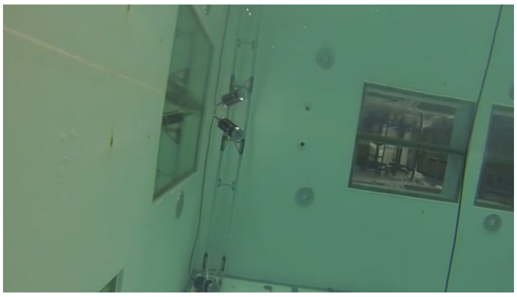


Figure 6.10: Frame example

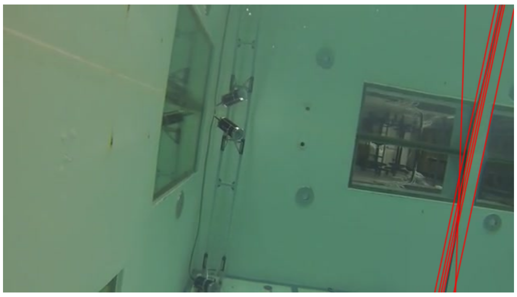


Figure 6.11: Hough transform result

Through the state estimation and resultant optical sensor measurement estimation calculated by the system's model, the algorithm analyzes which of the results provided by the Hough transform is closest to the estimate. If the estimate is far from any of the results, then the outlier removal algorithm provides the line parameters that are closest to the result of the previous iteration of the algorithm itself.

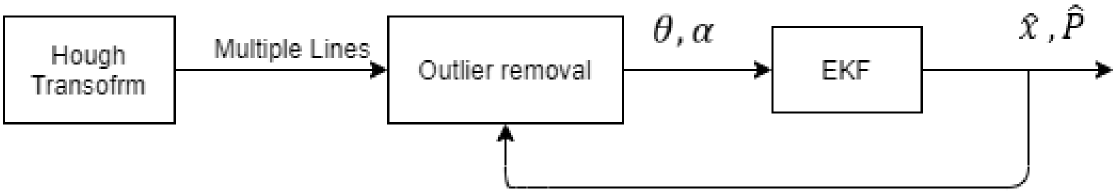
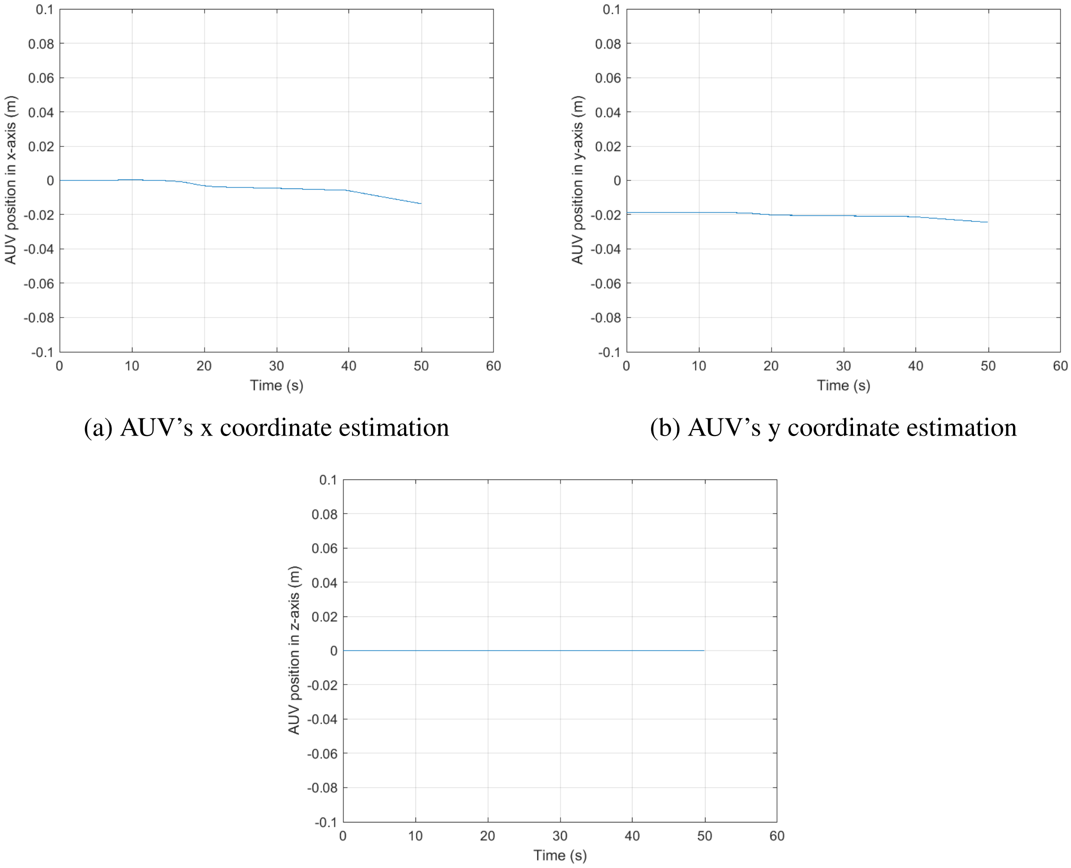


Figure 6.12: Block diagram of outlier removal after Hough transform

For the optical sensor, the estimate is made in cycles of 100 ms, since through the algorithm's processing time analysis in section 4.4.1, is concluded that this sensor's data processing is done in less than 50 ms, and so in each state estimation's cycle there is at least one measurement.



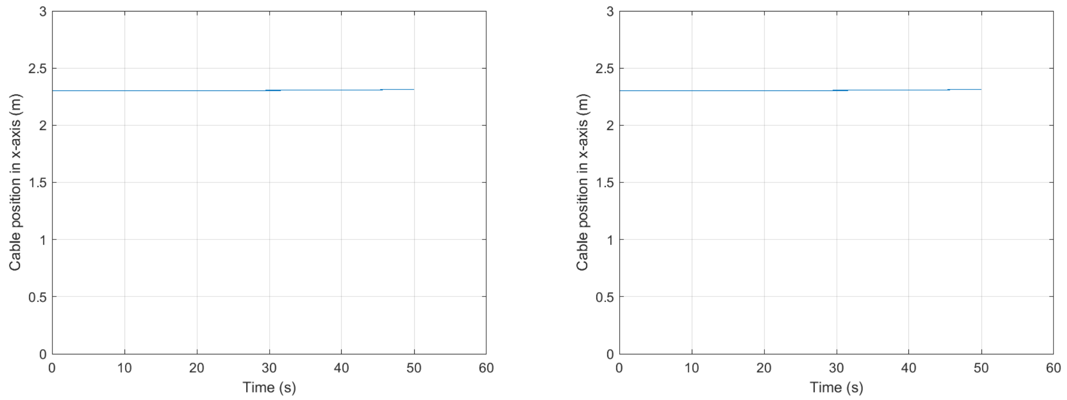
(a) AUV's x coordinate estimation

(b) AUV's y coordinate estimation

(c) AUV's z coordinate estimation

Figure 6.13: AUV's position estimation using optical sensor throughout time

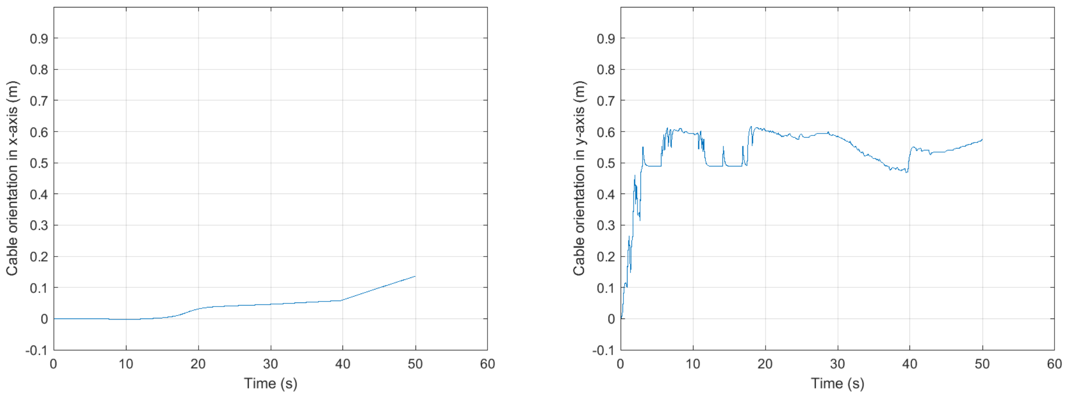
As observable in Figure 6.13, the vehicle's position does not diverge from the initial state, having only a small oscillation, possibly caused by uncertainty growth due to time evolution. This variation is more noticeable in x and in y, however still less than 1 cm. Therefore, it is possible to conclude that the state estimator converges to a final estimate.



(a) Cable's x coordinate estimation

(b) Cable's y coordinate estimation

Figure 6.14: Cable's position estimation using optical sensor throughout time



(a) Vector $\overrightarrow{v}$ x component estimation

(b) Vector $\overrightarrow{v}$ y component estimation

Figure 6.15: Cable's orientation estimation using optical sensor throughout time

The cable position's estimate, Figure 6.14, does not diverge from the initial state imposed on each of the components, concluding that the state estimator converges to a final estimate.

Observing the inertial frame of reference in Figure 6.9, it is noticeable that the cable's projection in the image, provided by the camera, only gives the cable orientation on the y axis of the inertial frame. Therefore, causing the difference between the temporal evolution of each of the components estimations of $\overrightarrow{v}$. While x component estimate diverges from the defined initial state, as expected, the y component presents a small oscillation with a maximum amplitude of 0.6.

6.3.2 State Estimator Using Acoustic Data

For the acoustic sensor, the prediction cycle is performed at time intervals of 0.4 seconds, while the update cycle is executed at time intervals of 8s, since a full sonar revolution is only completed after this period.

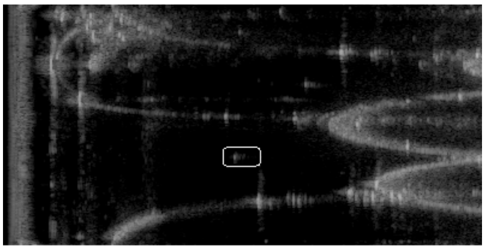


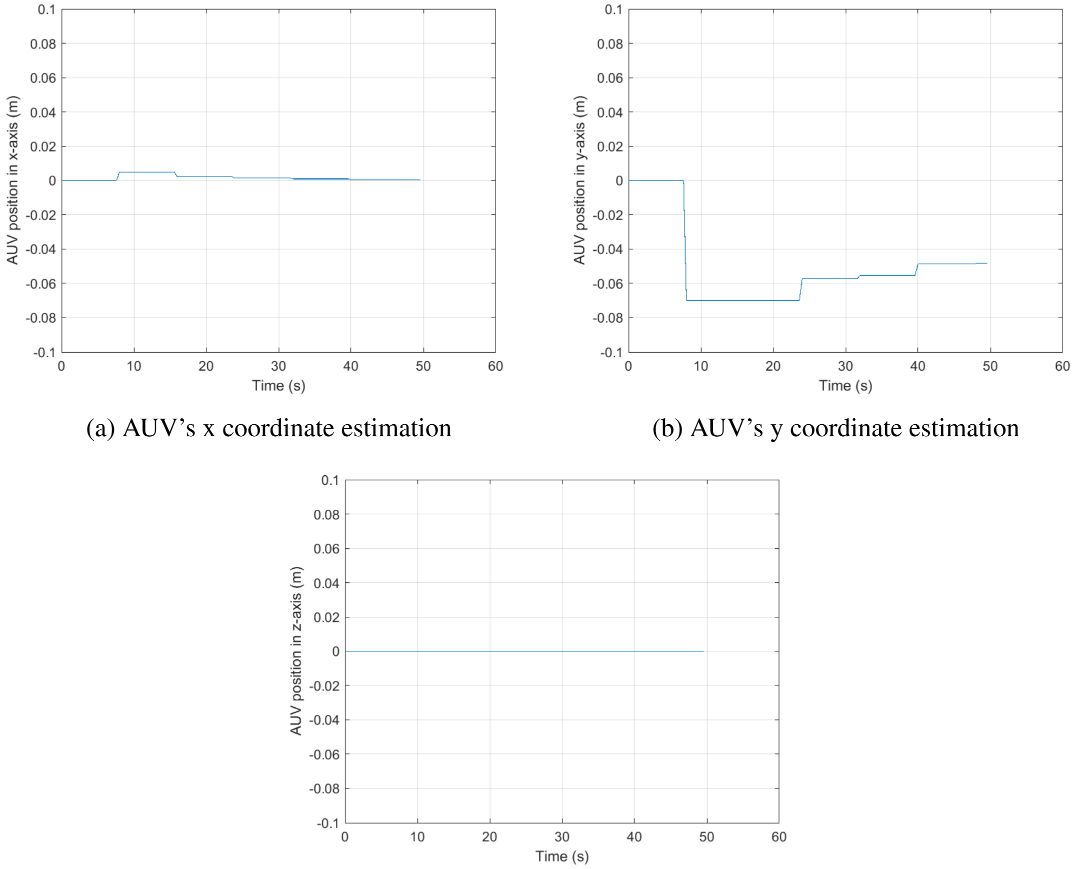
Figure 6.16: Template matching result with match signalized

In this case, the processing algorithm used for acoustic images, template matching, Figure 6.16, only returns one result, with an associated level of similarity. This similarity value allows that if it is low, considered as less than 0.8, it is discarded, and since the data set was acquired in a static scenario, the last match with an adequate similarity value will be passed to the state estimator. The results of this processing were then used to feed the state estimator, with the same initial conditions.

As in state estimation using optical measurements, the estimates of the vehicle position in x and in z, Figure 6.17, do not diverge from the initial state, and it can be concluded that it converges to a final estimate. However, the vehicle's position estimate in y presents sudden oscillations in the time instants that an acoustic measurement is available, and the update cycle is executed. The first update cycle executed results in a fall from the initial value to 7 cm, in each update cycle this value rises, finally converging at -5cm.

While using only optical measurements, which only provided information on the y axis, acoustic measurements deliver information on both axis, x and y, resulting in a great variation, in both coordinates of the cable's position estimates, whenever the correction cycle is executed. It is also visible that one component is mirrored to the other on the time axis, since the norm of these coordinates is the distance measured by the sonar. The last present estimate of pa_x is 2.2m, and of pa_y is 2.6m.

Information about the cable's orientation is only given by the optical sensor, since in the sonar plane the cable is a point. Therefore, the estimates of the vector $\overrightarrow{v}$ have oscillations through time, potentially caused by the increase of uncertainty since there is no information about these state variables through the sonar measurements, and so v_x oscillates between -48 and 0, while v_y varies between -18 and 36.

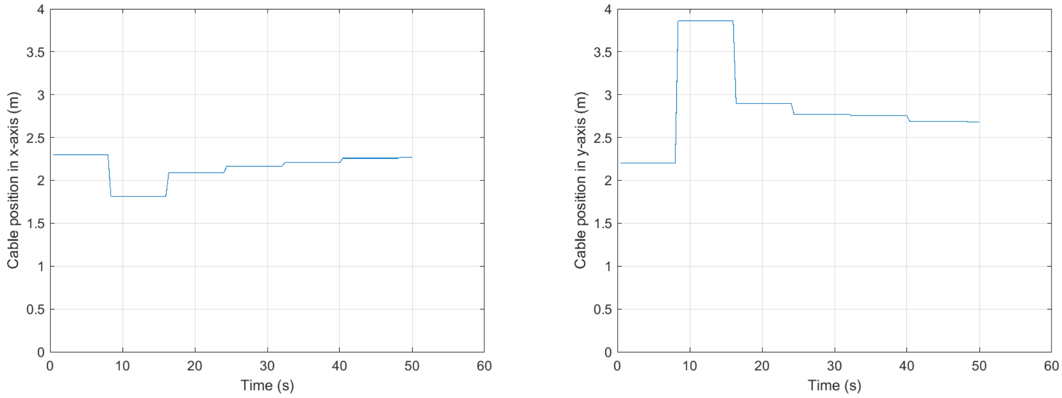


(a) AUV's x coordinate estimation

(b) AUV's y coordinate estimation

(c) AUV's z coordinate estimation

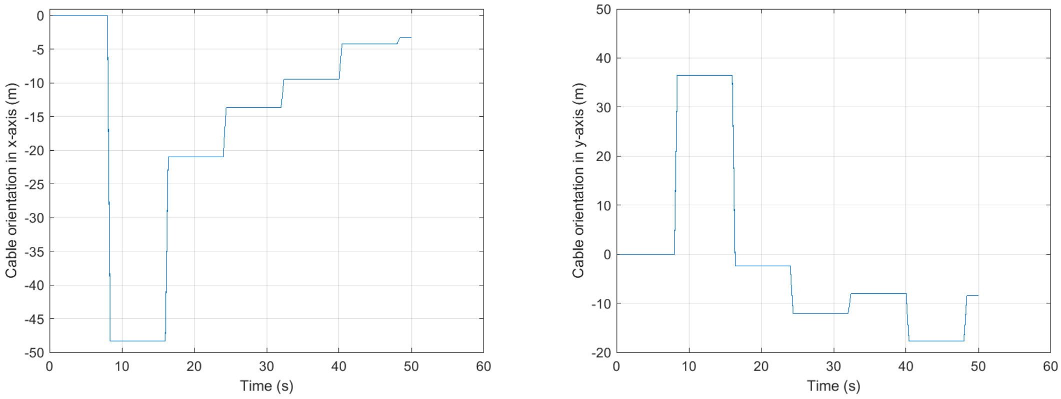
Figure 6.17: AUV's position estimation using acoustic sensor throughout time



(a) Cable's x coordinate estimation

(b) Cable's y coordinate estimation

Figure 6.18: Cable's position estimation using acoustic sensor throughout time



(a) Vector $\overrightarrow{v}$ x component estimation

(b) Vector $\overrightarrow{v}$ y component estimation

Figure 6.19: Cable's orientation estimation using acoustic sensor throughout time

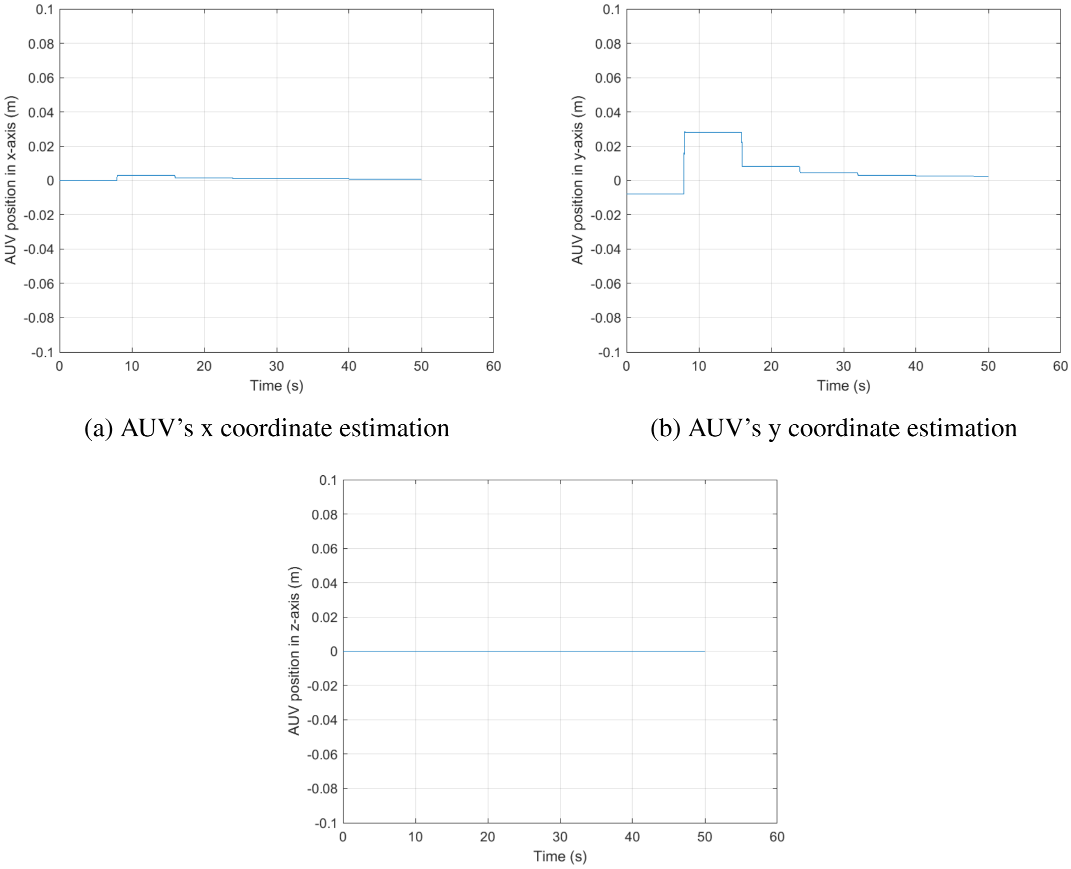
Using the measurements of each of the sensors individually presents errors in some state variables. In state estimation with optical measurements, not having information on the x-axis leads to errors in all estimates on this axis, as they are calculated through measurements on the y-axis, never having a correction for the state variables on this axis. In state estimation with acoustic measures, there is information about the position of the vehicle and the cable, yet not about the cable's orientation. Merging data from both sensors allows balancing these variations.

6.3.3 Optical and Acoustic Data Fusion

With the same initial conditions, the estimated vehicle position, Figure 6.20, again, does not diverge from the imposed initial state.

The cable's position estimate, Figure 6.21, in the two axes, x and y, presents a sharp variation in the time of execution of the first update cycle with acoustic measurements, but both converge to a final state. The estimate on the x-axis converges at 2.2 m, while on the y-axis it converges at 2.1 m. This estimate is the closest to reality, [2.2 , 1.9] m, from all tests of state estimators here presented.

The estimate of vector $\overrightarrow{v}$, cable orientation, in the x-axis there is variation due to the sonar correction cycle, and it finishes at -9.5. As this estimate is only being fed by the acoustic measurements, and these do not contain information about the orientation of the cable, the final value does not inform about the actual orientation of the cable on the x-axis as it is only being calculated with the uncertainty. To improve this estimate, the vehicle would have to be in motion and the optical measurements would provide information about the x-axis through movement. The estimate of v_y presents a constant variation due to correction cycles with acoustic measurements, the sudden variations corrected by the optical measurements. To evaluate the filter performance, the uncertainty of the filter of each of the components was also assessed. It was observed that the uncertainty over the iterations decreases until it stagnates at 6.5.

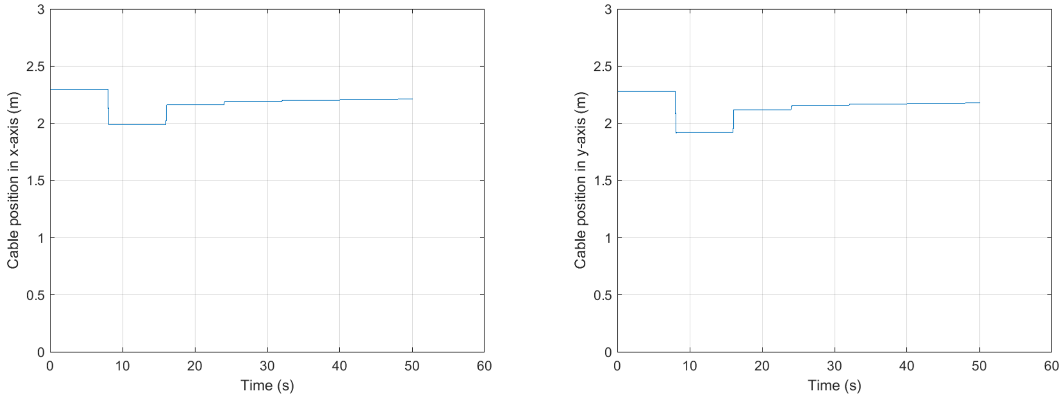


(a) AUV's x coordinate estimation

(b) AUV's y coordinate estimation

(c) AUV's z coordinate estimation

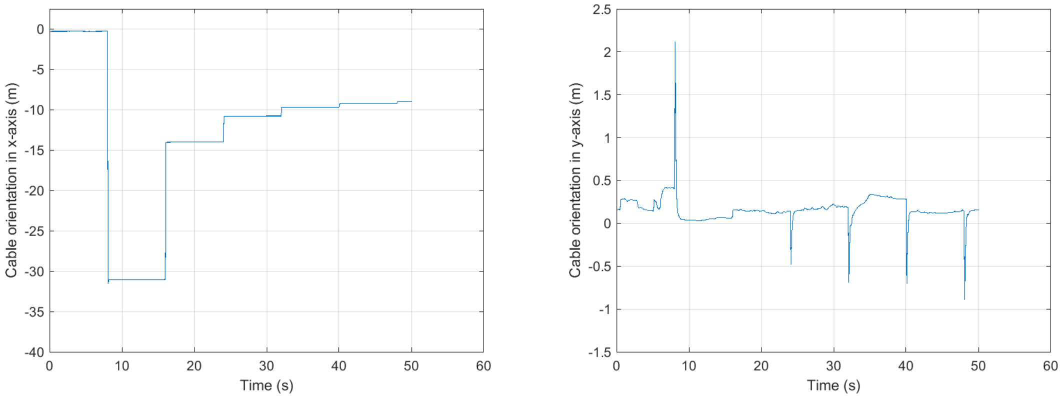
Figure 6.20: AUV's position estimation using sensor fusion



(a) Cable's x coordinate estimation

(b) Cable's y coordinate estimation

Figure 6.21: Cable's position estimation using sensor fusion



(a) Vector $\overrightarrow{v}$ x component estimation

(b) Vector $\overrightarrow{v}$ y component estimation

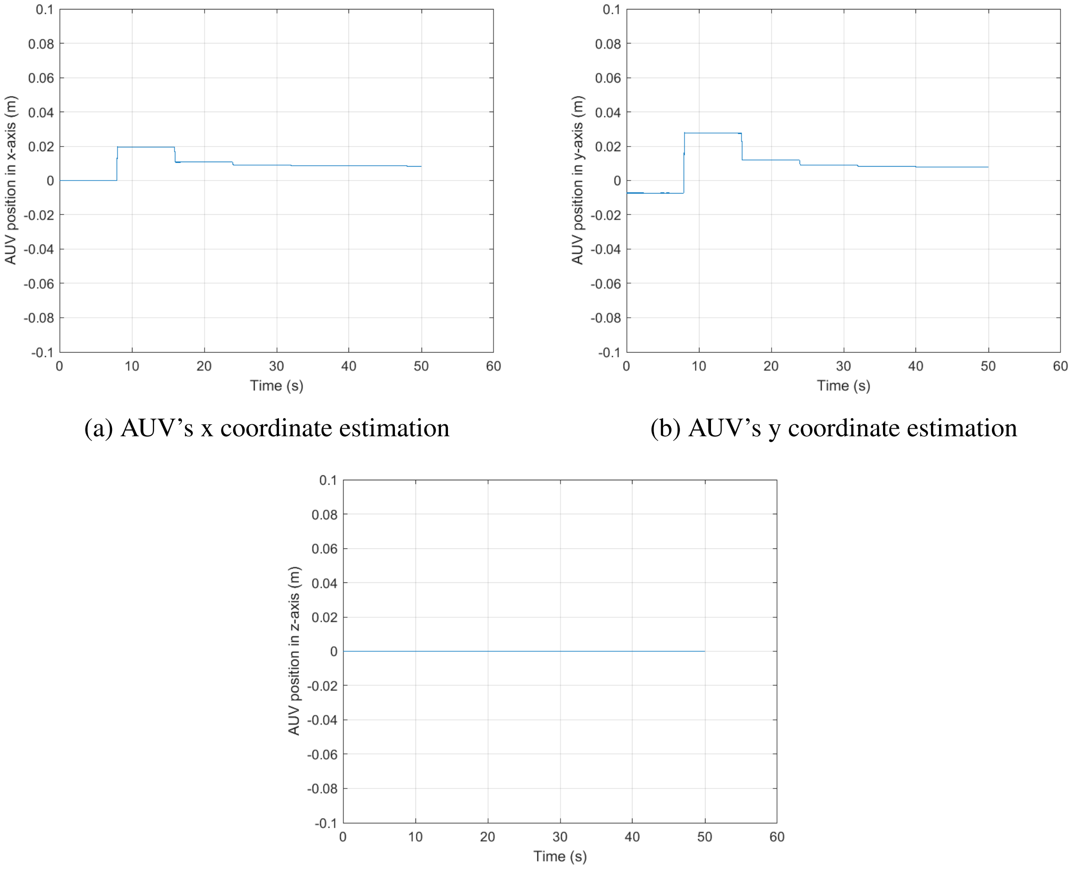
Figure 6.22: Cable's orientation estimation using sensor fusion

Another test was performed to assess the filter's performance. In this second test, the initial values of state and uncertainty were changed. The initial state of the cable's position was increased, in order to see if the filter really converges to values close to the desired one. For this, the initial uncertainty was also increased.

$$\widehat{X}_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ pa_{x_0} \\ pa_{y_0} \\ v_{x_0} \\ v_{y_0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.7 \\ 2.4 \\ 0 \\ 0 \end{bmatrix} \tag{6.51}$$

$$P_0 = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix} \tag{6.52}$$

The results of this last test are presented hereafter. The AUV's position estimate, Figure 6.23, shows no difference from the previous fusion. However, the cable's position estimate, Figure 6.24, shows more variations in y-axis due to the initial uncertainty increase. Comparing with the previous test with initial values of cable position closer to reality, the final cable's position estimates are [2.3 , 1.8] m, which represent values closer to reality, 10 cm error of the true value, but also an improvement on the y-axis since the last test. As for the estimate of the cable's orientation, Figure 6.25 this test does not show significant changes, having similar final estimates.

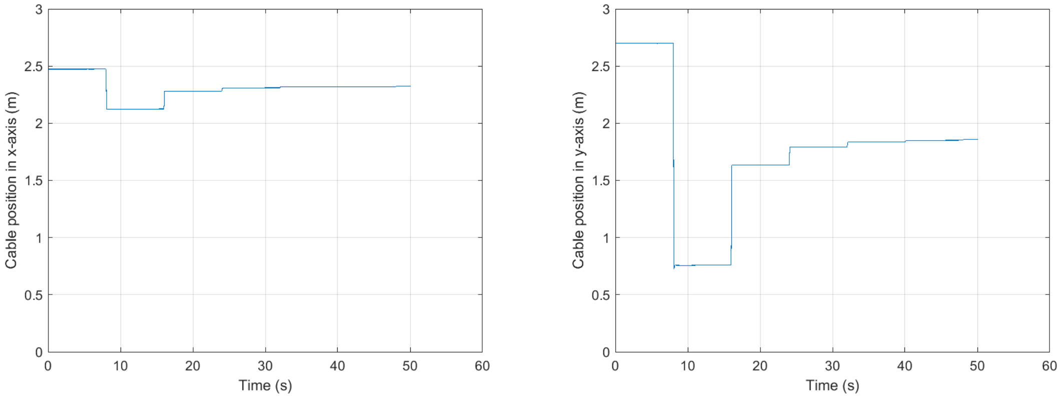


(a) AUV’s x coordinate estimation

(b) AUV’s y coordinate estimation

(c) AUV’s z coordinate estimation

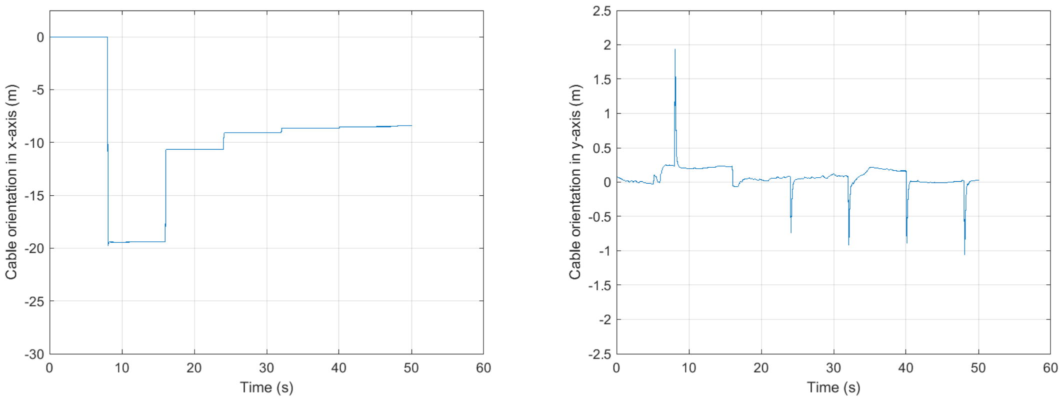
Figure 6.23: AUV’s position estimation using sensor fusion



(a) Cable’s x coordinate estimation

(b) Cable’s y coordinate estimation

Figure 6.24: Cable’s position estimation using sensor fusion



(a) Vector $\overrightarrow{v}$ x component estimation

(b) Vector $\overrightarrow{v}$ y component estimation

Figure 6.25: Cable's orientation estimation using sensor fusion

6.4 Conclusions

In this chapter, the development of the position estimator and sensor fusion, based on the implementation of the EKF algorithm, was presented. Further detail was given regarding the employed system model, measurement model and its incorporation into the state estimator.

A brief analysis of the develop state estimator performance was conducted with, in a first scenario with only optical measurements, then only using acoustic measurements and finally merging the sensor information of both. Through sensor fusion was possible to observe that the time evolution of the AUV's position and cable's position meets the real position. However, the cable's orientation estimate needs improvement, what could be potentially achieved through vehicle's movement and optical measurements from different angles.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

In retrospective, the main proposed objectives were accomplished. A working solution for underwater vehicles localization has been developed, based on optical and acoustic sensors.

The analysis of optical data was achieved through image processing techniques, with especial focus on line extraction algorithms. A comprehensive analysis of some of the most common algorithms for image pre-processing and line detection in literature has been conducted, in order to better understand the most suitable solution for the proposed problem: the suspended cable's detection. Afterwards, Hough line transform was applied, and its performance was assessed on the cable's detection in the camera's frames. Concluding that this algorithm is able to detect the cable for a real-time localization solution.

The study and interpretation of sonar data was achieved through the application of image processing techniques. A comprehensive analysis of some of the most common algorithms presented in literature for abstract features has been conducted, in order to better understand the most suitable solution for the proposed objectives. Furthermore, the comparison between feature extraction algorithms and template matching algorithm, provides valuable information regarding the application of these algorithms to acoustic images. In conclusion, both algorithms are able to detect the cable, however for a real-time problem and knowing the cable's pattern in an acoustic image, template matching proved to be the best approach.

Finally, a state estimator based on the common EKF algorithm has been developed and implemented in order to enable the estimation of the vehicle's pose as well as the cable's position through the fusion of the processed data from optical and acoustic sensors.

It is also worth mentioning that although this project was carried out with a small sized vehicle and a certain type of sensors, it can be adapted to the tools available and even to bigger scale problems.

Contributions:

- Developed an algorithm for line detection with both optical and acoustic sensors.
- Comparison between feature extraction algorithms and template matching for acoustic imaging.

7.2 Future Work

The next step would be to validate the developed solution in unstructured underwater environments, such as the sea, and eventually adapt it. For this case, it would be necessary to integrate other sensors to allow control in navigation.

It would also be interesting to change the sonar settings through filter estimation, in order to restrict the scanned sector. For this work, a real-time solution, it was not feasible since the sonar that was used takes a long time to reconfigure. Yet, using another sonar with lower reconfiguration period would significantly reduce the acquisition time of acoustic images.

In image processing, further improvement on the acoustic image feature detection to include less false positives and exploring machine learning methods would bring significant improvements to feature-based localization procedures.

References

[1] G. Welch and G. Bishop. An introduction to the kalman filter. *Proc. Siggraph Course*, 8, 01 2006.

[2] X. Guo. *Feature-Based Localization Methods for Autonomous Vehicles*. PhD thesis, University of Berlin, 2017.

[3] G. Alcantara. Underwater localization using imaging sonars in 3d environments. Master's thesis, Federal University of Rio de Janeiro, 2017.

[4] A. Sahoo, S. K. Dwivedy, and P.S. Robi. Advancements in the field of autonomous underwater vehicle. *Ocean Engineering*, 181:145–160, 2019. doi:https://doi.org/10.1016/j.oceaneng.2019.04.011.

[5] W. Kuperman and P. Roux. Underwater acoustics. *Springer Handbook of Acoustics, ISBN 978-0-387-30446-5. Springer-Verlag New York, 2007, p. 149*, -1:149, 01 2007. doi:10.1007/978-0-387-30425-0_5.

[6] R. Gonzalez and Z. Faisal. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., 2 edition, 2019.

[7] R. S. Lewis. *The autonomous underwater vehicle emergency localization system: an under ice AUV tracking technology for over-the-horizon operations*. PhD thesis, University of Tasmania, 2015.

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

[9] A.J.V. Oliveira. Feature-based underwater localization using an imaging sonar. Master's thesis, University of Porto, 2020.

[10] A. B. Figueiredo, B. M. Ferreira, and A. C. Matos. Vision-based localization and positioning of an AUV. In *OCEANS 2016 - Shanghai*, pages 1–7, 2016. doi:10.1109/OCEANSAP.2016.7485384.

[11] S. Song, B. Kim, J. Kim, and S. Yu. Real-time reliability evaluation of optical and acoustic images for feature-based localization of AUV. In *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*, pages 1–6, 2018. doi:10.1109/OCEANSKOBE.2018.8559126.

[12] G. Kano, T. Andrade, and A. Moutinho. Automatic detection of obstacles in railway tracks using monocular camera. In Dimitrios Tzovaras, Dimitrios Giakoumis, Markus Vincze, and Antonis Argyros, editors, *Computer Vision Systems*, pages 284–294, Cham, 2019. Springer International Publishing.

[13] R. Pérez-Alcocer, L. Torres-Méndez, E. Olguín-Díaz, and A. Maldonado-Ramírez. Vision-based autonomous underwater vehicle navigation in poor visibility conditions using a model-free robust control. *Journal of Sensors*, 2016, 2016. doi:10.1155/2016/8594096.

[14] M. Ferrera, J. Moras, P. Trouvé-Peloux, and V. Creuze. Real-time monocular visual odometry for turbid and dynamic underwater environments. *Journal of Sensors*, 19(3), 2019. doi:10.3390/s19030687.

[15] L. Kleeman and R. Kuc. *Sonar Sensing*, pages 491–519. Springer Berlin Heidelberg, 2008.

[16] R. Hansen. Introduction to sonar. 2009.

[17] E. Westman and M. Kaess. Degeneracy-aware imaging sonar simultaneous localization and mapping. *IEEE Journal of Oceanic Engineering*, 45(4):1280–1294, 2020. doi:10.1109/JOE.2019.2937946.

[18] H. M. D. D. Bandara. Sonar image based advanced feature extraction and feature mapping algorithm for under-ice AUV navigation. Master's thesis, University of Tasmania, 2017.

[19] O. Miljković. Image pre-processing tool. *Kragujevac Journal of Mathematics*, 32(32):97–107, 2009. URL: http://eudml.org/doc/252610.

[20] K. Yang, L. Yu, M. Xia, T. Xu, and W. Li. Nonlinear ransac with crossline correction: An algorithm for vision-based curved cable detection system. *Optics and Lasers in Engineering*, 141:106417, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0143816619318792, doi:https://doi.org/10.1016/j.optlaseng.2020.106417.

[21] J. Seo, S. Chae, J. Shim, D. Kim, C. Cheong, and T. Han. Fast contour-tracing algorithm based on a pixel-following method for image sensors. *Sensors*, 16(3), 2016. URL: https://www.mdpi.com/1424-8220/16/3/353, doi:10.3390/s16030353.

[22] A. Kazlouski and R.K. Sadykhov. Plain objects detection in image based on a contour tracing algorithm in a binary image. In *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*, pages 242–248, 2014. doi:10.1109/INISTA.2014.6873624.

[23] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[24] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):105–119, 2010. doi:10.1109/TPAMI.2008.275.

[25] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. doi:10.1109/ICCV.1999.790410.

[26] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. Similarity Matching in Computer Vision and Multimedia. URL: https://www.sciencedirect.com/science/article/pii/S1077314207001555, doi:https://doi.org/10.1016/j.cviu.2007.09.014.

[27] D. Zhang. *Wavelet Transform*. 05 2019. doi:10.1007/978-3-030-17989-2_3.

[28] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010*, pages 778–792, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[29] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. pages 2564–2571, 11 2011. doi:10.1109/ICCV.2011.6126544.

[30] P. Alcantarilla, A. Bartoli, and A. J. Davison. Kaze features. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision – ECCV 2012*, pages 214–227, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[31] S. Leutenegger, M. Chli, and R. Siegwart. Brisk: Binary robust invariant scalable keypoints. pages 2548–2555, 11 2011. doi:10.1109/ICCV.2011.6126542.

[32] A. Spears, A. M. Howard, M. West, and T. Collins. Acoustic sonar and video sensor fusion for landmark detection in an under-ice environment. In *2014 Oceans - St. John's*, pages 1–8, 2014. doi:10.1109/OCEANS.2014.7002992.

[33] A. Kaehler and G. Bradski. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., 1st edition, 2016.

[34] J. Gao, A. Proctor, and C. Bradley. Adaptive neural network visual servo control for dynamic positioning of underwater vehicles. *Neurocomputing Journal*, 167, 2015.

[35] W. Yang, S. Fan, S. Xu, P. King, B. Kang, and E. Kim. Autonomous underwater vehicle navigation using sonar image matching based on convolutional neural network. *IFAC-PapersOnLine*, 52(21):156–162, 2019. doi:10.1016/j.ifacol.2019.12.300.

[36] L. Tang, J. Hou, H. Xu, M. Wu, X. Xia, and Y. Cui. Motion behavior recognition of underwater vehicle based on YOLOv3. *Journal of Physics: Conference Series*, 1744(2):022144, feb 2021. URL: https://doi.org/10.1088/1742-6596/1744/2/022144, doi:10.1088/1742-6596/1744/2/022144.

[37] F. Ferreira, D. Machado, G. Ferri, S. Dugelay, and J. Potter. Underwater optical and acoustic imaging: A time for fusion? a brief overview of the state-of-the-art. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–6, 2016. doi:10.1109/OCEANS.2016.7761354.

[38] J. Potter, A. Pack, M. Hoffmann-Kuhnt, T. Koay, P. Seekings, and M. Chitre. A synchronised acoustic array, rangefinder video system with examples from 'singing' humpback whales (megaptera noveangliae). 01 2007.

[39] M.J. Chantler, D.B. Lindsay, C.S. Reid, and V.J.C. Wright. Optical and acoustic range sensing for underwater robotics. In *Proceedings of OCEANS'94*, volume 1, pages I/205–I/210 vol.1, 1994. doi:10.1109/OCEANS.1994.363924.

[40] G. Inglis. *Hybrid optical acoustic seafloor mapping*. PhD thesis, University of Rhode Island, January 2013.

[41] Y. Kim and H. Bang. *Introduction to Kalman Filter and Its Applications*. IntechOpen, 2019. doi:10.5772/intechopen.80600.

[42] S. J. Julier and J. K. Uhlmann. New extension of the Kalman filter to nonlinear systems. In Ivan Kadar, editor, *Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068, pages 182 – 193. International Society for Optics and Photonics, SPIE, 1997. URL: https://doi.org/10.1117/12.280797, doi:10.1117/12.280797.

[43] Y. Laamari, K. Chafaa, and B. Athamena. Particle swarm optimization of an extended kalman filter for speed and rotor flux estimation of an induction motor drive. *Electrical Engineering*, 97:129–138, 06 2015. doi:10.1007/s00202-014-0322-1.

[44] X. Yuan, J. Martínez-Ortega, J. Fernández, and M. Eckert. Aekf-slam: A new algorithm for robotic underwater navigation. *Sensors (Basel, Switzerland)*, 17, 05 2017. doi:10.3390/s17051174.

[45] C. S. Gonçalves, B. M. Ferreira, and A.C. Matos. Design and development of shad - a small hovering AUV with differential actuation. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–4, 2016. doi:10.1109/OCEANS.2016.7761457.

[46] Ad. Kaehler and G. Bradski. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., 1st edition, 2016.

[47] A. Gómez-Espinosa, E. Cuan-Urquizo, and J. González-García. Autonomous underwater vehicles: Localization, navigation, and communication for collaborative missions. *Applied Sciences*, 10:1256, 02 2020. doi:10.3390/app10041256.

[48] Ramesh Jain, Rangachar Kasturi, and Brian Schunck. *Machine Vision*. 01 1995.

[49] F. Mekelleche and H. Haffaf. Classification and comparison of range-based localization techniques in wireless sensor networks. *Journal of Communications*, 12(4), 2017. doi:10.12720/jcm.12.4.221-227.

[50] R. Luo and H. Guo. *Application of the Several Common Algorithms for Corner Detection to Sonar Image Registration*, pages 93–99. Springer, Singapore, 2021.

[51] G. V. Massaguer. *Online acoustic localization methods for autonomous underwater vehicles*. PhD thesis, University of Girona, 2018.

[52] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[53] M.H.A. Mangshor, R. Ambar, H.A. Kadir, K. Isa, I.Y. Amran, A.A.A. Kadir, N. S. Ibrahim, C. C. Choon, and S. Sagara. *Development of autonomous underwater vehicle equipped with object recognition and tracking system*, pages 37–56. Springer, 2021. doi:10.1007/978-981-15-5281-6_4.

[54] S. Zhao, A. Anvar, and T. Lu. Automatic object detection and diagnosis for AUV operation using underwater imaging sonar. In *2008 3rd IEEE Conference on Industrial Electronics and Applications*, pages 1906–1911, 2008. doi:10.1109/ICIEA.2008.4582851.

[55] S. Krupinski, R. Desouche, N. Palomeras, G. Allibert, and M. Hua. Pool testing of AUV visual servoing for autonomous inspection. *IFAC-PapersOnLine*, 48(2):274–280, 2015. doi:10.1016/j.ifacol.2015.06.045.

[56] A. Burguera, F. Bonin-Font, and G. Oliver. Trajectory-based visual localization in underwater surveying missions. *Journal of Sensors*, 2015.

[57] T. Zhang, S. Huang, D. Liu, L. Shi, C. Zhou, and R. Xiong. A method of state estimation for underwater vehicle navigation around a cylindrical structure. In *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, pages 101–106, 2016. doi:10.1109/ICIEA.2016.7603559.

[58] J. P. Fula, B. M. Ferreira, and A. J. Oliveira. AUV self-localization in structured environments using a scanning sonar and an extended kalman filter. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–6, 2018. doi:10.1109/OCEANS.2018.8604798.

[59] S. Suman, C. L. Kaiser, M. V. Jakuba, and J. C. Kinsey. Integration and algorithm development for forward looking imaging sonars on hybrid and autonomous underwater robots. In *OCEANS 2015 - MTS/IEEE Washington*, pages 1–6, 2015. doi:10.23919/OCEANS.2015.7404417.

[60] Y. Liu, S. Guo, L. Shi, H. Xing, X. Hou, H. Liu, Y. Hu, D. Xia, and Z. Li. Multi-sensor fusion based localization system for an amphibious spherical robot. In *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 2523–2528, 2019. doi:10.1109/ICMA.2019.8816345.

[61] F. Hidalgo and T. Bräunl. Evaluation of several feature detectors/extractors on underwater images towards vslam. *Journal of Sensors*, 20(15), 2020. doi:10.3390/s20154343.

[62] Z. Zhang, Y. Cao, M. Ding, L. Zhuang, and J. Tao. Monocular vision based obstacle avoidance trajectory planning for unmanned aerial vehicle. *Aerospace Science and Technology*, 106:106–199, 2020. doi:10.1016/j.ast.2020.106199.

[63] P. Tueller, R. Kastner, and R. Diamant. A comparison of feature detectors for underwater sonar imagery. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–6, 2018. doi:10.1109/OCEANS.2018.8604786.

[64] M. A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2018.