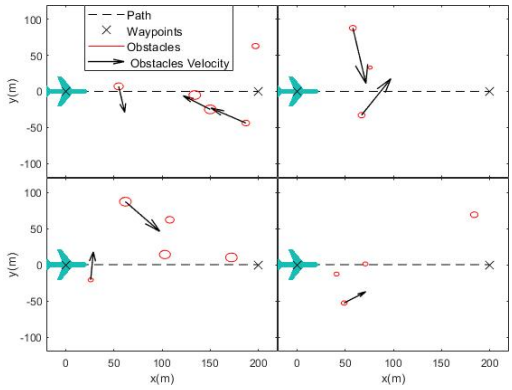
TÉCNICO LISBOA



Optimization of Obstacle Detection for Small UAVs

Pedro Miguel Silva Serrano

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisor: Prof. André Calado Marta

Examination Committee

Chairperson: Prof. José Fernando Alves da Silva
Supervisor: Prof. André Calado Marta
Member of the Committee: Prof. Alexandra Bento Moutinho

June 2022

Dedicated to Gabriela Ferradosa

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

Firstly I would like to thank my parents for their unconditional support during my academics journey. I would also like to thank my girlfriend for her help and emotional support during all stages of this thesis, and my sister for her good advices as a former IST student.

A special thanks to all my colleagues (Aníbal, António, Damião, Filipe, João, Luís, Meireles, Rui and Tiago) that made this journey a lot more pleasant and fun.

Finally, I would like thank Prof. André Marta for its patience and guidance since day one of this thesis.

Resumo

O mercado dos UAVs cresce rapidamente, sendo necessário um grande investimento no desenvolvimento de sistemas de Sense and Avoid. Posto isto, esta tese centra-se neste tipo de sistemas com especial foco na fase da deteção em UAVs pequenos de asa fixa.

Em primeiro lugar, é feita uma apresentação dos vários tipos de sensores e de algoritmos de evasão culminando com uma lista dos sensores adequados a este trabalho e uma explicação de fundo sobre o método de campos potenciais utilizado como algoritmo de evasão. De seguida, são definidos parâmetros para caracterizar sistemas de deteção, testando-os depois em missões simuladas de evasão de obstáculos. No final destas simulações, foram tecidas várias sonsiderações sobre a influência dos parâmetros na performance geral do sistema.

O trabalho continua com uma apresentação do hardware e software necessário para um sistema de Deteção e Evasão. Após a escolha dos sensores e do controlador, um esquema é elaborado documentando todos os dispositivos necessários e as ligações existentes entre eles. Além disto, o firmware do controlador e o software para a estação de controlo são escolhidos tendo em conta as suas especificações.

De seguida, o trabalho avança para a elaboração de experiências com os sensores escolhidos, de forma a obter informações reais sobre as suas capacidades tais como alcance máximo, erro médio das medições etc.

Por último um sistema muito simples de deteção e evasão é implementado num pequeno rover com o objetivo de realizar experiências que demonstrem as capacidades do sistema de deteção e evasão.

Palavras-chave: UAVs, Deteção e evasão, Campos Potenciais, Parâmetros, Experiências, Rover

Abstract

The UAV market grows every year and in order to keep this, it is required a large investment in Sense and Avoid systems. With this in mind, the thesis is centred on this type of systems with special focus on the detection phase in small fixed-wing UAVs.

Firstly, a presentation of the various types of available sensors and avoidance algorithms is made culminating with a list of the most suitable sensors for this type of work and a background explanation on the Potential fields method that will be used as the avoidance algorithm.

Next, parameters are defined to characterize the sensing systems and then tested in simulated obstacle avoidance missions. At the end of these simulations, several conclusions were drawn on the influence of the parameters in the overall performance of the system.

The work continues with a presentation of the hardware and software required to implement a Sense and Avoid system. After the sensors and controller have been chosen, a schematic is drawn up with all the necessary devices and the connections between them. Furthermore, the flight controller firmware and the ground control station software are chosen taking into account their specifications.

Next, the work proceeds to the elaboration of experiments with the sensors, obtaining real information about their capabilities such as maximum range, average error of the measurements, etc.

Finally, a simple sense and avoid system is implemented in a small rover in order to perform experiences that can evaluate the capabilities of the sense and avoid system.

Keywords: UAVs, Sense and Avoid, Potential Fields, Parameters, Experiments, Rover

Contents

Acknowledgments vii
Resumo ix
Abstract xi
List of Tables xv
List of Figures xvii
Nomenclature xix
Glossary xxi

1 Introduction **1**
1.1 UAV Market Overview 1
1.2 Sensing Systems Applications 3
1.3 Motivation 4
1.4 Objectives and Deliverables 5
1.5 Thesis Outline 6

2 Sense and Avoid Systems **7**
2.1 Architecture of a Sense and Avoid System 7
2.2 Sensing Stage 8
2.2.1 RADAR 9
2.2.2 Laser Rangefinder/LIDAR 9
2.2.3 Ultrasonic Sensors 10
2.3 Avoiding Stage 11
2.3.1 Geometric Approach 11
2.3.2 Graph Search Algorithms 11
2.3.3 Potential Fields 11
2.4 Final Remarks 14

3 Sensor Parametric Studies **17**
3.1 UAV Model 17
3.2 Numerical Model 19
3.2.1 Scenarios Generation 19
3.2.2 Simulation and Objective Function 21

3.3 Laser Rangefinder . . . 22
3.3.1 Model . . . 22
3.3.2 Parameters . . . 23
3.3.3 Optimal setup . . . 27
3.4 Ultrasonic Sensor . . . 28
3.4.1 Model . . . 28
3.4.2 Parameters . . . 30
3.4.3 Optimal Setup . . . 33
3.5 Final Remarks . . . 33

4 Hardware and Software Implementation **35**
4.1 Sensor Hardware . . . 35
4.1.1 Ultrasonic Sensors . . . 35
4.1.2 Laser Rangefinder . . . 36
4.2 Flight Controller . . . 37
4.3 Electrical Layout . . . 38
4.4 Flight Controller Software . . . 39
4.5 Ground Control Software . . . 40

5 Sensor Experiments **43**
5.1 Bench Tests . . . 43
5.1.1 Ultrasonic Sensor . . . 43
5.1.2 Laser Rangefinder . . . 48
5.2 Rover Tests . . . 50
5.3 Summary . . . 57

6 Conclusions **59**
6.1 Achievements . . . 59
6.2 Future Work . . . 60

Bibliography **61**

List of Tables

1.1 Classification of UAVs 2

2.1 Sensors qualitative comparison 15

3.1 $\mathbf{R_d}$ and β discrete domains for the laser rangefinder simulations 24
3.2 Scenarios parameters 25
3.3 Laser rangefinder optimal results 27
3.4 $\mathbf{R_d}$ and β discrete domains for the ultrasonic sensor simulations 31
3.5 Ultrasonic sensor optimal results 33

4.1 Qualitative analysis of I2CXL-MaxSonar-EZ series 36
4.2 Comparison of different laser rangefinders 37
4.3 Comparison of different laser rangefinders [7] 37

5.1 LW20/C connections 48
5.2 Firmware setup parameters 53

List of Figures

1.1 Civil UAV production 2
1.2 AR4 by Tekever 3
1.3 Examples of sensing systems applications 4
1.4 UAS challenges 5

2.1 Architecture of a S&A system 7
2.2 Sensing methods 8
2.3 RADAR being used as altimeter 9
2.4 Lidar 3-D point cloud 10
2.5 Ultrasonic sensors working principle 10
2.6 Potential fields concept 12
2.7 Safety zones 12
2.8 Attractive field with two components 13
2.9 Repulsive field 14

3.1 Head-on collision 18
3.2 Turning rate required to avoid obstacle safely for different velocities 19
3.3 Sensor studies steps 19
3.4 Scenario generation algorithm 20
3.5 Set of 4 random scenarios 21
3.6 Distribution of the penalties weights 22
3.7 Sensing system model using laser rangefinders 24
3.8 Pre-determined path in collision free condition 25
3.9 Sensor parameters influence in the objective function 26
3.10 linear Regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$(laser rangefinder case) 27
3.11 Types of ultrasonic sensor beam patterns. 28
3.12 Sonar model steps 30
3.13 Ultrasonic sensor parameters 31
3.14 Ultrasonic sensor parameters influence in the objective function. 31
3.15 Linear regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$ (narrow beam pattern case) 32
3.16 Linear regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$ (wide beam pattern case) 33

4.1 MB1242 ultrasonic sensor 36
4.2 Lightware LW20/C 37
4.3 Flight Controller 38
4.4 Electrical scheme 39
4.5 QGroundcontrol environment 42

5.1 QGroundcontrol setup for MB1242 44
5.2 Ultrasonic sensor bench test 45
5.3 MB1242 detection rate from several distances and orientations 46
5.4 MB1242 experimental beam pattern 47
5.5 MB1242 average absolute error from several distances and orientations 47
5.6 Lightware LW20/C connection with USB adaptor 48
5.7 Lightware Upgrader environment 49
5.8 LW20/C detection rate from several distances 50
5.9 LW20/C average absolute error from several distances 50
5.10 Electrical connections for the rover experiments 51
5.11 PPM encoder 52
5.12 Mission Planner environment 53
5.13 Logical scheme of the ArduRover object avoidance algorithm 55
5.14 Rover tests scheme 55
5.15 Final rover configuration 56
5.16 Simple avoidance results 56

Nomenclature

Greek symbols

α_{PF} Weighting term.

β Sensor orientation.

θ Angle between $\mathbf{d_0}$ and $\mathbf{m}$.

$\theta_{cut-off}$ Angle of θ that resets repulsive field to zero.

ϕ UAV banking angle/Objective function weighting term.

$\dot{\Omega}$ UAV turning rate.

Roman symbols

$\mathbf{R_a}$ Action Radius.

$\mathbf{R_d}$ Detection Radius/Sensor maximum range.

$\mathbf{R_C}$ Collision Radius.

$\mathbf{R_S}$ Safety Radius.

$\mathbf{f}_{at}$ Attractive Force.

P_{close} Closest point on the path .

P_{next} Next waypoint on the path .

P_{UAV} UAV position .

$\mathbf{s}$ Swirling unit vector .

$\hat{\mathbf{k}}$ z axis unit vector .

S_{max} Maximum potential in repulsive fields.

$\mathbf{d_0}$ Vector pointing from object centre to the UAV .

$\mathbf{m}$ Direction of motion .

f Objective Function.

$\mathbf{y}$	Parameters vector .
d_{min}	Minimum distance to an obstacle .
r^2	Coefficient of determination .
v	UAV speed .

Subscripts

max	Maximum.

Glossary

UAV	Unmanned Aerial Vehicles
FAA	Federal Aviation Admnistration
IoT	Internet of Things
UAS	Unmanned Aircraft System
MTOW	Maximum Takeoff Weight
UA	Unmanned Aircraft
NASA	National Aeronautics and Space Agency
LIDAR	Light Detection and Ranging
RADAR	Radio Detection and Ranging
ADS-B	Automatic Dependent Surveillance - Broadcast
GPS	Global Positioning System
FOV	Field of View
I2C	Inter-Integrated Circuit
FMU	Flight Management Unit
IMU	Inertial Measurement Unit
I/O	Input/Output
PWM	Pulse Width Modulation
VTOL	Vertical Takeoff and Landing
CAN	Controller Area Network
ESC	Electronic Speed Controller
PPM	Pulse Position Modulation
LIPO	Lithium Polymer

Chapter 1

Introduction

1.1 UAV Market Overview

UAVs started to be developed during World War I. Since then, several subsequent wars led to the continuous development of these systems. It is no surprise the fact that UAVs have a military background. These type of systems provide real-time intelligence, surveillance, and reconnaissance information from hostile areas [1]. As years passed, these characteristics started to be adapted to non-military applications, transforming these vehicles to one of the most promising markets in the world. These applications include:

- **Fire Combat**: UAVs are now being experimented on fire detection and combat. This idea is being explored by a multidisciplinary team at Instituto Superior Técnico in a project called FireFront. This team is trying to integrate a system composed with positioning sensors and video cameras that can detect a fire and predict its future location [2];
- **Transportation**: This gathers several sectors where drones can be used to autonomously transport load. Parcel delivery and automation of intra logistics are two examples where this technology is starting to find its space. Although there are still legal barriers regarding air space use, some progress is being made as companies like Google, Amazon and UPS have already granted FAA air carrier certificates in 2019 [3];
- **Precision Agriculture**: UAV-based IoT(Internet of Things) technology is widely spread in precision agriculture. These systems offer an opportunity to acquire data in an highly effective way for a reduced cost. Low altitude flights provide ultra-high spacial resolution images that can significantly, improve the crops monitoring system [4];
- **Cinema and Television**: These sectors of activities can benefit a lot from UAV's. Whether the goal is to capture an artistic photo/video or an important moment for a journalistic piece, drones can make it possible at reduced cost.

Due to this variety of applications, which include way more sectors than the ones showed before, the UAVs market is projected to keep growing despite the Covid 19 pandemic. According to Teal Group's

most recent report, civil UAV production will increase from current five thousand million dollars annually in 2020 to over eighteen thousand millions in 2029, more than one hundred thousand million dollars over the decade [5], as shown in Figure 1.1.

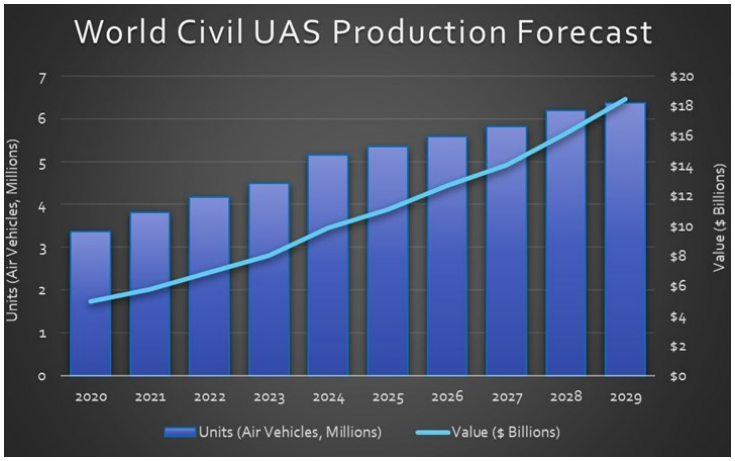


Figure 1.1: 2020 Civil UAV production and projections until 2029 (values in thousand million US$)) [5]

These type of autonomous vehicles gather a wide spectrum of characteristics that can be dissected in order to catalogue each UAV. Firstly, all models are divided between 2 configurations: fixed and rotary wing. Then, both categories can be distinguished following other criteria like it is shown in Table 1.1.

Table 1.1: Classification of UAVs [6]

Category	MTOW (kg)	Range (km)	Flight alt. (m)	Endurance (h)
Micro	<5	<10	250	1
Mini	<20/25/30/150	<10	150/250/300	<2
Close Range (CR)	25-150	10-30	3,000	2-4
Short Range (SR)	50-250	30-70	3,000	3-6
Medium Range (MR)	150-500	70-200	5,000	6-10
MR endurance(MRE)	500-1,500	>500	8,000	10-18
Low altitude deep penetration (LADP)	250-2,500	>250	50-9,000	0.5-1
Low altitude long endurance (LALE)	15-25	>500	3,000	>24
Medim altitude long endurance (MALE)	1,000-1,500	>500	3,000	24-48
High Altitude long endurance (HALE)	2,500-5,000	>2,000	20,000	24-48
Stratospheric (Strato)	>2,500	>2,000	>20,000	>48
Exo-stratospheric (EXO)	TBD	TBD	>30,500	TBD
Unmanned combat AV (UCAV)	>1,000	1,500	12,000	2
Lethal (LET)	TBD	300	4,000	3-4
Decoys (DEC)	150-250	0-500	50-5,000	<4

Although not explored in Table 1.1, autonomy level is another variable that can be used to perform a precise classification. Each UAV must fall into one of the following categories [6]:

- **Remotely piloted**: A certified pilot remotely controls the system;
- **Remotely operated**: The UA is given high-level commands (waypoints, objects to track, etc.) and its performance is monitored by a trained operator;
- **Fully autonomous**: The UA is given general tasks and is capable of determining how to accomplish them, even at the face of unforeseen events.

After presenting these classification systems, it is possible to precisely describe the type of UAV that will be used as a reference for all simulations performed hereupon. Maintaining the same approach present in [7], the focus will be on small fixed wing UAVs. A good example of a UAV that falls within this category is the AR4 model produced by the portuguese company Tekever. This model, that can be visualized in Figure 1.2, has a $2.1\,\mathrm{m}$ wingspan and a length of $1.35\,\mathrm{m}$ with an MTOW of $4\,\mathrm{kg}$ that includes a payload capacity of $1\,\mathrm{kg}$ [8].

Figure 1.2: AR4 by Tekever[8]

1.2 Sensing Systems Applications

Sensors are a key element to pass from a piloted vehicle to an autonomous one. They translate information regarding its surroundings and send it to the rest of the system. In UAVs, sensors typically provide information related to position, velocity, acceleration and obstacle positions. With proper data fusion techniques, a Sense and Avoid system may be developed.

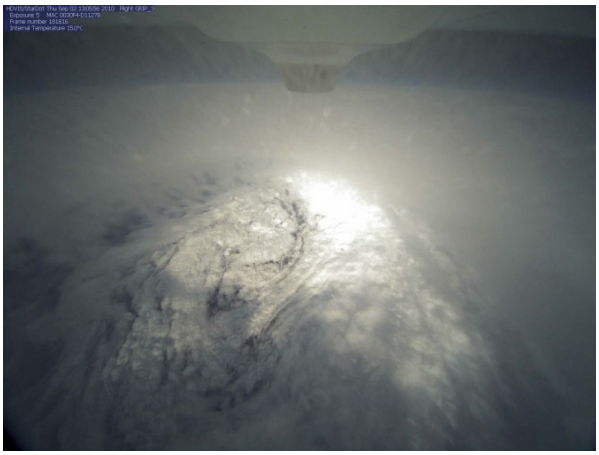
A good product example of a UAV sensing system would be Guidance, developed by DJI, that offers an elegant way to improve a simple flight system with small dimensions (Mini UAV) as seen in Figure 1.3(a). Alongside with a DJI flight controller, it can automatically avoid collisions even at high speeds. In order to do his job, Guidance is equipped with a processing core, integrated visual cameras, ultrasonic sensors and computer vision algorithms.

On the opposite side of the complexity spectrum there is the Global Hawk sensing system. This UAS developed by Northrop Grumman is the largest UAS ever produced and is stacked with the best sensing technology in the world. A good example of its capabilities would be NASA's missions where a Global Hawk is being used to collect weather data over storms. Alongside all the sensing technology

which allows Global Hawk to fly safely over a storm there are other instruments responsible for collecting weather data such as temperature, humidity, pressure, wind speed and direction [9], as seen in Figure 1.3(b)

To sum up, Sensing systems are key for UAVs as they replace human eyes. Improvements in this field are essential to increase safety and can lead to an integrated airspace between manned and unmanned aircrafts in the future. Moreover, the UAVs ability to reach places where manned aircraft would not be capable to operate, makes sensing systems crucial to capitalize it.

(a) Guidance by DJI



(b) NASA's Global Hawk in the Eye of Hurricane Earl[10]

Figure 1.3: Examples of sensing systems applications

1.3 Motivation

In Section 1.1, it was shown the huge margin of growth in the UAV industry. However, there are still major issues which can jeopardize its future.

These challenges can be divided into two branches: technical and legal issues. This can be seen in Figure 1.4.

Technical Issues

Energy Management is currently one of the UAS (Unmanned Aerial System) technical issues. Reducing batteries weight, wireless charging or solar powered UAV's are examples of approaches to achieve a maximum energy efficiency.

Then, there is the collision avoidance issue. There are already various algorithms capable of this task, however optimization is still in progress. Limited power and payload raises questions regarding typically heavy sensors such as LIDARs and RADARs. Moreover, increasing UAVs speed will be a challenge for these algorithms.

The next technical issue is related to communication protocols. Frequent connections interruptions, fluid network topology and limited energy resources makes this a hard problem to solve.

Finally, these type of vehicles will be subject to a constant danger of cyber attacks, which needs to be dealt with instantaneously. Events like this could happen on UAV systems, ground control stations or

throw communication links and may cause loss of data or even crashes [11].

Legal Issues

The great majority of UAVs in the future will be equipped with ultra high definition cameras which brings serious privacy issues. Additionally, there are still legal barriers concerning air space usage. Currently, nothing points to a common air space between manned and unmanned aircrafts, and BVLOS (Beyond Visual Line of Sight) drones are still a fantasy [12].

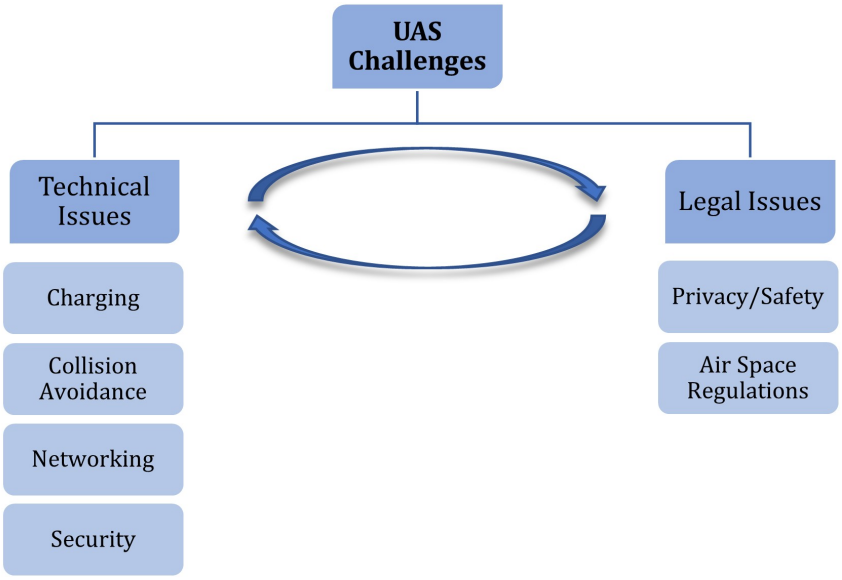


Figure 1.4: UAS challenges

In addition, there is also an interconnection between technical and legal issues. An example can be the quick integration of high definition cameras on board of these systems which brings privacy issues. On the opposite side, developments in Sense and Avoid technology may help companies to be granted permits to operate more freely. Additionally, the appearance of new and more rigid legislation may challenge companies to improve the final product.

This thesis focus on Collision Avoidance as it is a key feature for future UAV operation in urban areas. Within this Collision Avoidance topic, this thesis will mainly focus on the detection phase, using pre-developed avoidance algorithms.

1.4 Objectives and Deliverables

This thesis is preceded by two other master thesis. In [13], a study has been made regarding collision avoidance algorithms, with special focus on the Potential Fields algorithm. In [7], the author simulated various detection systems with different types of sensors and configurations. Then, it was used the potential fields code developed in [13], together with an optimization algorithm, to reach optimal configurations regarding UAV detection system.

In this thesis, the main goal is to use these two studies as background information and start implementing an efficient Sense and Avoid system, firstly on a simple rover, and ideally on a micro fixed wing

UAV. It is important to mention that the focus will be almost exclusively on the detection phase, using only pre-developed work for the avoidance stage like the simulation code developed in [13]. The most important phases required to present a final product are:

1. Redo simulations performed in [7] for the available sensors. This might require the development of the ultrasonic sensor model as it was not included in [7];
2. Select the most appropriate software and hardware;
3. Implement everything on hardware and conduct experiments;
4. Readjust simulations based on experimental results and perform new optimization;
5. Validate the optimal configuration.

Taking into account these objectives, there is one final product that will be the deliverable. This is a rover and/or a micro UAV with a proper Sense and Avoid system installed.

1.5 Thesis Outline

The structure of this thesis is as follows:

- **Chapter 2: Sense and Avoid Systems** makes a brief introduction to Sense and Avoid systems. It starts by explaining the system as a whole and then presents various alternatives for sensing and avoiding stages;
- **Chapter 3: Sensor Parametric studies** presents simulations regarding the performance of the available sensors;
- **Chapter 4: Hardware/Software Implementation** explains how the selected hardware and software works;
- **Chapter 5: Sensor Experiments** explains how all the tests will work and presents all the results with respect to bench and ground tests.

Chapter 2

Sense and Avoid Systems

The main goal of this chapter is to expose a general idea about Sense and Avoid (S&A) Systems. Initially, an analysis of the typical architecture of a S&A system will be done. Then, the focus will be on the Detection part of the system, enumerating the various types of sensors on the market and their advantages and drawbacks. Finally, several theoretical approaches for the avoidance algorithm will be discussed.

2.1 Architecture of a Sense and Avoid System

With the huge rise of the UAV's market came the urgency of developing an autonomous system capable of controlling the vehicle as a human would do (or better). This is the task performed by a S&A system. In Figure 2.1 it is shown the different phases followed by these types of systems in order to achieve the final goal.

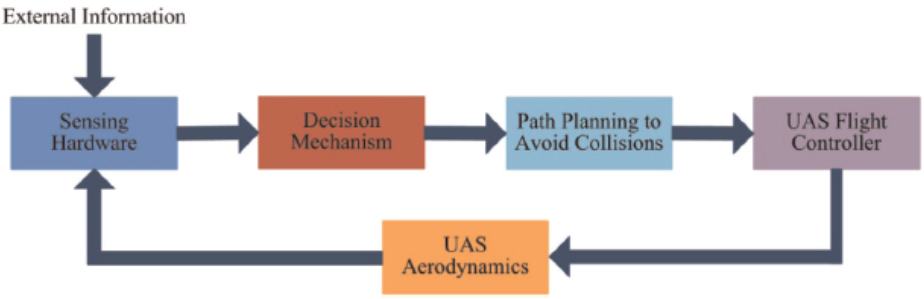


Figure 2.1: Architecture of a S&A system [14]

The process begins with the sensing stage. The sensors should retrieve data regarding the surrounding environment. This data is going to be analysed by the decision mechanism that will decide whether there are any conflicts with the predefined path. If there is any conflict, the system will redefine its path to avoid the detected object and return to its original trajectory as soon as possible. Then, the controller will use the actuators at its disposal to follow the new path. Finally, it is important to mention the time constraint adjacent to this task. If these stages are not followed quickly enough, collisions may

be inevitable [14].

2.2 Sensing Stage

The sensing stage is the first major task of a S&A System. In order to optimize the process, it will be necessary to select the sensors according to the mission at hand. The best way to do that is by exploring all the possible solutions in the market (see Figure 2.2).

Currently, there are two types of approach to this problem: cooperative and non cooperative models. The first one implies the existence of a similar system on the intruder while the latter is independent. Non-cooperative sensing can also be divided between active and passive. The difference resides in the source of the detected signal: passive sensors rely on external signals while the active ones send their own and receive its reflection [15]. This sensor taxonomy is described in Figure 2.2.

As it was exposed in Chapter 1, this thesis aims to develop a S&A system for a small UAV. In order to achieve this goal, it was decided that the best solution would be necessarily non cooperative sensors. This decision is based on [7] as it concludes that cooperative detection is inadequate due to their high cost and weight. Moreover, these models are only able to detect other aircraft equipped with the same technology which leaves no protection to static obstacles like the ground or trees and unequipped aircrafts. Based on this decision, only active sensors will be dissected with more detail.

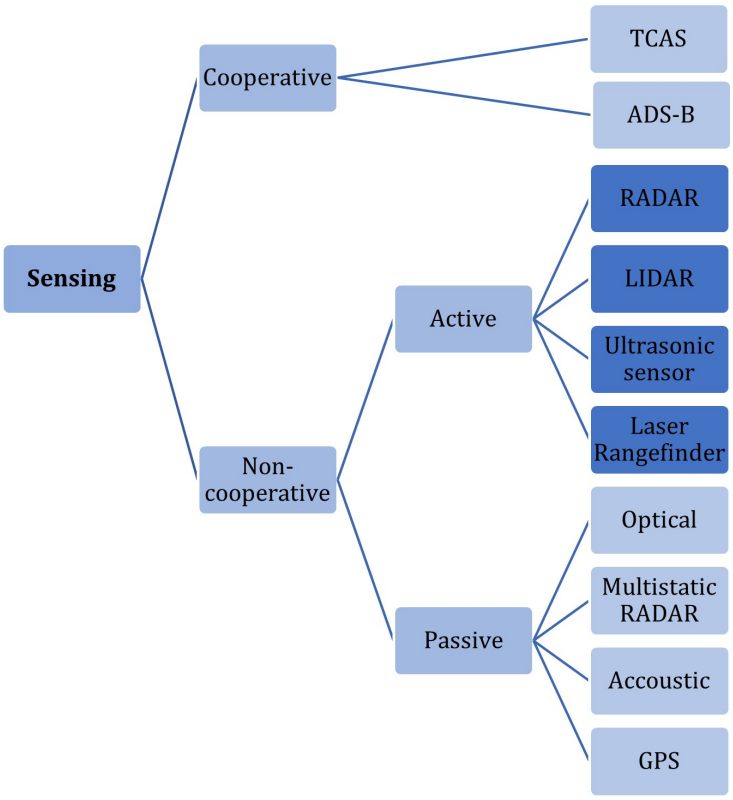


Figure 2.2: Sensing methods and examples

2.2.1 RADAR

RADAR, acronym for Radio Detection and Ranging, is a system which uses radio electromagnetic waves to determine information about surrounding targets. The objective is to transmit a particular type of wave form, for example a pulse modulated sine wave, and then receive an echo which is the reflection of the transmitted signal on the target. By measuring the time lapse between the transmitted and received signal its possible to determine the distance between RADAR and target. The angular position of the target should be within the receiver antenna beam, though narrow beams can give precise angular positions. Additionally, if relative motion exists between target and radar, the shift in the carrier frequency of the reflected wave (doppler effect) is a measure of the target's relative (radial) velocity and may be used to distinguish moving targets from stationary objects [16].

There are several advantages for RADAR users. Firstly, it is reliable even with the presence of clouds, fogs or snow. Secondly, RADARs may reach outstanding ranges (up to 3700km for over-the horizon RADARs). This kind of distances is, obviously, not applicable for the purpose of this thesis (due to power and size required), however it gives an idea of the optimal capabilities of this system.

Figure 2.3: RADAR being used as altimeter [17]

2.2.2 Laser Rangefinder/LIDAR

Laser Rangefinders use the same principle as RADARs. The difference resides on the nature of the electromagnetic waves used. While RADAR's use radio waves, this type of sensors uses higher frequency electromagnetic waves such as infrared and visible light waves. This radiation has significantly shorter wavelengths, which can detect smaller objects. Additionally, range is also a positive aspect of this sensor as it can detect objects up to $20\,\mathrm{km}$. On the downside, this type of radiation may be affected by the presence of fog, rain or snow due to its shorter wavelength.

LIDAR, acronym for Light Detection and Ranging is a method based on laser technology. When a laser is attached to a scanning surface such as a servo or if an oscillating mirror deflects the laser beams, a large number of distance measurements can be acquired, thus creating a 3-D point cloud, as represented in Figure 2.4 [7].

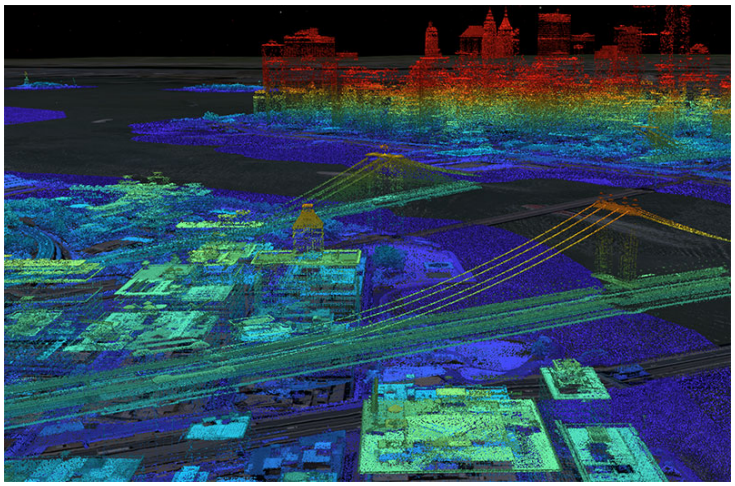


Figure 2.4: LiDAR topobathymetric point cloud of New York City [18]

2.2.3 Ultrasonic Sensors

Ultrasonic sensors follow a working principle similar to bat echolocation. An ultrasonic pulse is sent and, if there is any object in the surroundings, a reflection will be received (see Figure 2.5). By calculating the travel time and the speed of sound in that particular location, a distance between the sensor and the object can be obtained.

This system may be very useful as it is capable to detect transparent objects and to operate under dark environments. Additionally, it is a low cost option.

On the negative side, there are several limitations which are worth considering. Firstly, this type of sensors cannot exceed ranges from around $10\,\mathrm{m}$ which may not be enough for various applications. Secondly, there are issues regarding the speed of sound. Changes in temperature or air density affect this variable and consequently may cause errors in distance measurements if the system is not recalibrated. Finally, there are issues regarding sound absorption from soft materials which may lead to undetected objects. This is specially problematic in applications where there is the need to detect humans since clothes are mostly made off sound absorbing materials. [19]

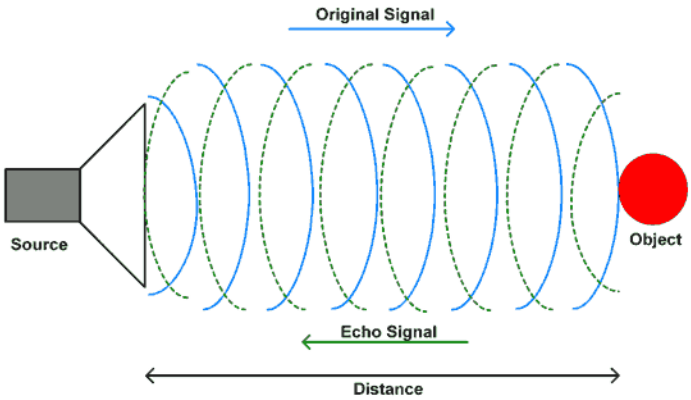


Figure 2.5: Ultrasonic sensors working principle [20]

2.3 Avoiding Stage

The Avoiding stage is responsible to generate a new path capable of deviating the aircraft from the intruder and to return to the previous defined trajectory. To do that, it is necessary the adoption of the most suitable algorithm.

According to [13], Potential Fields algorithm is fast and computationally inexpensive, being a feasible solution for real-time implementation. Based on this conclusion, potential fields algorithm will be used on this thesis and, for this reason, it will be described with more detail than other alternative methods.

2.3.1 Geometric Approach

The simplest way to deal with this problem is probably using a geometric approach. This method uses the intruders relative positions and velocities to the UAV in order to generate a geometry guidance based, for example on the orthogonal rule [15] [21].

The major issue regarding this approach is related to the assumption that each intruder is moving with constant velocity at each sampling time. This can obviously be improved with the implementation of extended Kalman filtering [21], however there is no guarantee to find an optimal solution, specially with multiple intruders. On the bright side, this approach is easily implemented on a real-time system due to its simplicity.

2.3.2 Graph Search Algorithms

This method consists in transforming a continuous space into a grid of unit squares (2D) or unit cubes (3D). Each one of this units will be considered a vertex of the graph while their borders are the graph's edges.

Using algorithms such as A*[22], an optimal solution can be obtained with the right heuristics. This function evaluates a future solution cost for each vertex. By minimizing it, the algorithm is able to prioritize vertexes that have higher probability to be part of the optimal solution.

2.3.3 Potential Fields

This method is based on Coulomb's law. The idea is to consider every intruder as a repulsive charge and the next waypoint on the path as an attractive charge[22] as can be seen in Figure 2.6. The summation of these charges should return the safest direction to follow at each instance of time. The elegance and simplicity of this approach is remarkable. However, there are some issues that need to be discussed.

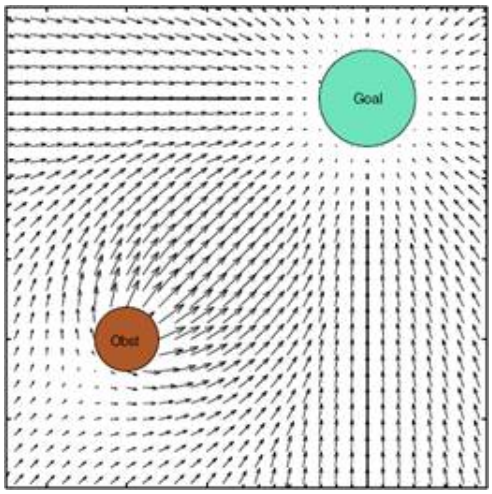


Figure 2.6: Potential fields concept [23]

Safety zones

This algorithm was previously implemented with safety zones (see Figure 2.7) associated to each obstacle [13][7]. In these implementations, all objects were defined as spheres and each one has a collision radius associated ($\boldsymbol{R_C}$), which is equal to the obstacle radius. A **collision** is said to occur when an UAV passes through this radius.

The safety radius ($\boldsymbol{R_S}$) defines the distance that should be maintained between the obstacle and the UAV, accounting for possible deviations during flight and uncertainties on the detection phase. In order to account for these uncertainties, $\boldsymbol{R_S}$ was computed as follows: $\boldsymbol{R_S}$=$\boldsymbol{R_C}$+$\boldsymbol{R_{UAV}}$, where $\boldsymbol{R_{UAV}}$ is the UAV half wingspan. Basically, there is no guarantee that trespassing this frontier will cause a collision but it is definitely not desirable. For this reason, it was defined as a **close call** every situation where $\boldsymbol{R_S}$ is trespassed but not $\boldsymbol{R_C}$.

The action radius ($\boldsymbol{R_a}$) is used as a trigger for the avoidance system and sets the distance from which the replanned path starts to depart from the original path given by the global planner. Finally, there is a detection radius ($\boldsymbol{R_d}$), which defines the maximum distance from where the sensing system is capable to detect an obstacle. This radius can have significant differences when using different sensors on board.

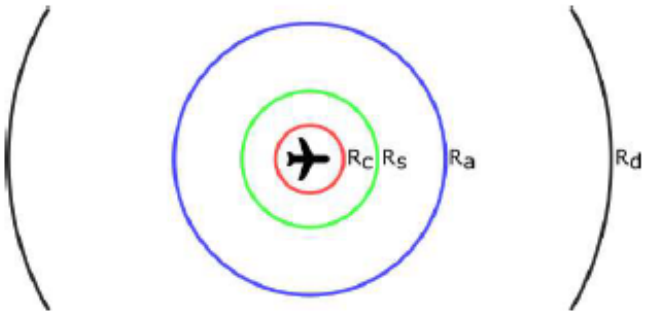


Figure 2.7: Safety zones [7]

Attractive Field

In most cases, the UAV needs to follow a well determined path. The idea is to sum up two components for the attractive field as seen in equation (2.1). The first one pointing from the UAV position, P_{UAV} to the closest point on the path, P_{close} and the last one pointing from P_{close} to the next waypoint on the path, P_{next}. There is also a parameter, α_{PF} that adjusts the relevance of each term. When $\alpha_{PF}$is close to 1 the algorithm is valuing more the approximation to the path. On the other hand, when α_{PF} is close to 0, the algorithm is preferring to follow the path direction [13]. Figure 2.8 shows an example of an attractive field defined by equation 2.1, where these two components can be identified.

$$\mathbf{f}_{at} = \alpha_{PF}\frac{P_{close} - P_{UAV}}{\|P_{close} - P_{UAV}\|} + \alpha_{PF}\frac{P_{next} - P_{close}}{\|P_{next} - P_{close}\|} \tag{2.1}$$

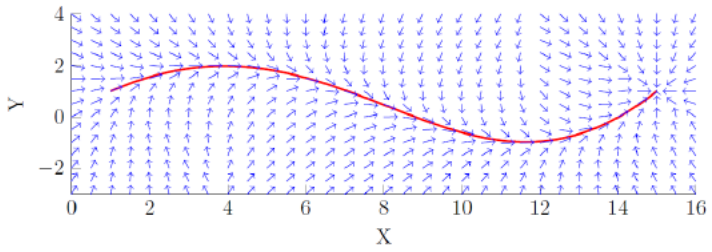


Figure 2.8: Attractive field with two components (α_{PF}=0.5)[13]

Repulsive Field

The repulsive field is similar to the attractive one. However, there are new variables to be dealt with, specially distance. For this reason, $\boldsymbol{d_0}$ is used as a vector pointing from the object centre to the UAV. In addiction to this variable there is also a swirling unit vector,

$$\boldsymbol{s} = \frac{\hat{\boldsymbol{k}} \times \boldsymbol{d_0}}{\|\boldsymbol{d_0}\|}, \tag{2.2}$$

where $\hat{\boldsymbol{k}}$ is the z axis unit vector. The idea is to define this potential function according to the barriers exposed in Figure 2.7 resulting in a repulsive field like the one represented in Figure 2.9. Inside the collision area, the potential function will be infinite with $\boldsymbol{d_0}$ orientation . Within the close call region, the potential will be at its maximum value, S_{max} but the direction will be given by $\boldsymbol{s}$ or its inverse. For the action area the potential is almost the same, but it decreases linearly from S_{max} to zero. Finally, outside the action area the repulsive field will be null, as expected. Each one of this cases is described in this specific order in (2.3).

$$\begin{cases} \infty \dfrac{\boldsymbol{d_0}}{\|\boldsymbol{d_0}\|} & \text{, if } \|\boldsymbol{d_0}\| \leq R_C \\ S_{max}\boldsymbol{s} & \text{, if } R_C < \|\boldsymbol{d_0}\| \leq R_S \\ S_{max}\dfrac{R_a - \|\boldsymbol{d_0}\|}{R_a - R_S}\boldsymbol{s} & \text{, if } R_S < \|\boldsymbol{d_0}\| \leq R_a \\ 0 & \text{, if } \|\boldsymbol{d_0}\| \geq R_a \vee \theta \leq \theta_{cut-off} \end{cases} \tag{2.3}$$

This approach with a swirling potential in safety and action areas prevents the UAV from an irregular motion around the object but at the same time can trap it in a continuously circular motion. This problem is solved considering an angle, θ, between the desired direction of motion ($\boldsymbol{m}$) and $\boldsymbol{d_0}$ given by

$$\theta = \arccos \frac{\boldsymbol{m} \cdot \boldsymbol{d_0}}{\|\boldsymbol{d_0}\|\|\boldsymbol{m}\|}. \tag{2.4}$$

When θ reaches a fixed value, $\theta_{cut-off}$, the potential function returns to 0.

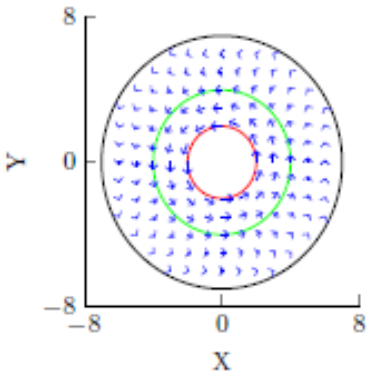


Figure 2.9: Repulsive field [7]

2.4 Final Remarks

In order to develop a Sense and Avoidance system, as proposed in Chapter 1, it is important to decide how the system will work regarding sensing and avoiding stages. For the sensing stage, it was shown the importance of having non-cooperative active models in order to make the system rely solely on its sensors capabilities. Within these models, it was evident that RADARs and Laser Rangefinder/LIDARs may be better suited for larger distances, however ultrasonic sensors are more budget friendly options (see Table 2.1). When it comes to the avoidance stage, potential fields algorithm was selected as the optimal method, since it has low computational requirements.

Table 2.1: Sensors qualitative comparison [7]

Sensor	Weight	Electric Power	Signal Processing	Cost	Range	Directionality	FOV
Laser Rangefinder	low	low	simple	low	high	directional	very narrow
LIDAR	medium	medium	simple	medium	high	multidirectional	very narrow
RADAR	medium	low	simple	medium	high	(multi)directional	broad
Ultrasonic Sensors	low	low	simple	low	low	directional	medium

Chapter 3

Sensor Parametric Studies

The main objective of this chapter is to explain how sensors were evaluated in MATLAB simulations. This includes the code responsible for replicating a real situation where sensors would be used in UAVs and a function which can evaluate the general performance of the system. Additionally, results regarding laser rangefinders and ultrasonic sensors will be presented. Both RADARs and LIDARs were not considered because it was not possible to acquire them in due time.

3.1 UAV Model

This segment of the work includes studies about the influence of UAV properties in the global performance of the system. The idea is to consider a simple head-on collision scenario where an UAV is flying with an arbitrary sensor, which is perfectly detecting a $2\,\mathrm{m}$ radius object. This obstacle is in front of it and travels with the same speed as the UAV (see figure 3.1). What would be the turning rate, $\dot{\Omega}$, required to make sure that this UAV could successfully avoid its obstacle without any collision, close call or danger for the UAV structure? This is the question that will hopefully be answered for the range of speeds at which a small UAV operates ($8 \sim 15\,\mathrm{m/s}$).

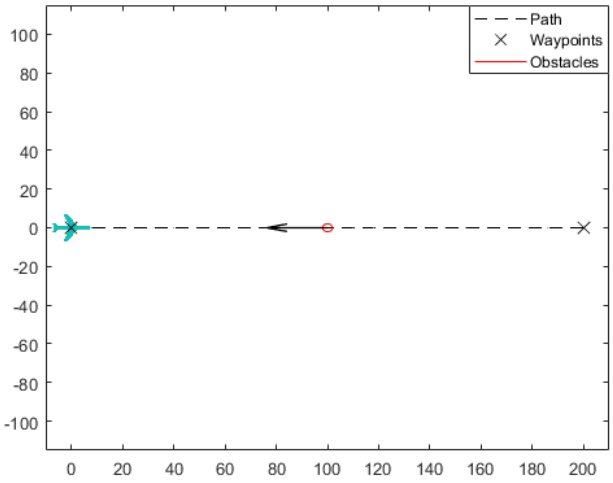


Figure 3.1: Head-on collision

As it was expected, the results can vary a lot depending on when the UAV starts departing from its original path, so studies were conducted for several action radius, $\boldsymbol{R_a}$, as exhibited in Figure 3.2. The red line represents the maximum turning rate function, $\dot{\Omega}_{max}(v)$ so all points above it are a potential danger for the aircraft structure. In order to obtain this line, the maximum load factor that the structure of the UAV can handle was set to $4g$. Considering this limit and that the UAV is maintaining a level flight, the UAV maximum banking angle is

$$\phi_{max} = \arccos\left(\frac{1}{n_{max}}\right) \approx 75.5\,°, \tag{3.1}$$

where n_{max} is the maximum load factor. After gathering this information, the $\dot{\Omega}_{max}(v)$ function was formulated as

$$\dot{\Omega}_{max}(v) = \frac{\tan(\phi_{max})g}{v}, \tag{3.2}$$

where v is the UAV speed and g the standard acceleration due to gravity [24].

Now looking at Figure 3.2, it can be concluded that the UAV is capable of avoiding any obstacle without exceding the maximum load factor when $\mathbf{R_a}$ is fixed at $10\,\mathrm{m}$ or more. When the action radius is reduced, this ideal situation vanishes, as it can be seen by the green and black dots positioned above the red line. In the case of $\mathbf{R_a} = 8\,\mathrm{m}$, the UAV is capable of safely avoiding an obstacle until $12\,\mathrm{m/s}$ while in the case of $\mathbf{R_a} = 6\,\mathrm{m}$ that limit velocity changes to $10\,\mathrm{m/s}$.

It is now pretty clear that the UAV has some limitations when faced with critical collisions routes. This does not mean that the UAV cannot operate with speeds above those limits mentioned, but it is important to keep in mind that any sensor that does not guarantee an action radius of at least $10\,\mathrm{m}$ may not be enough to avoid a collision/close call in more critical situations as the head-on collision studied. This will happen because, from now on, every simulation will limit the UAV turning rate following Equation 3.2.

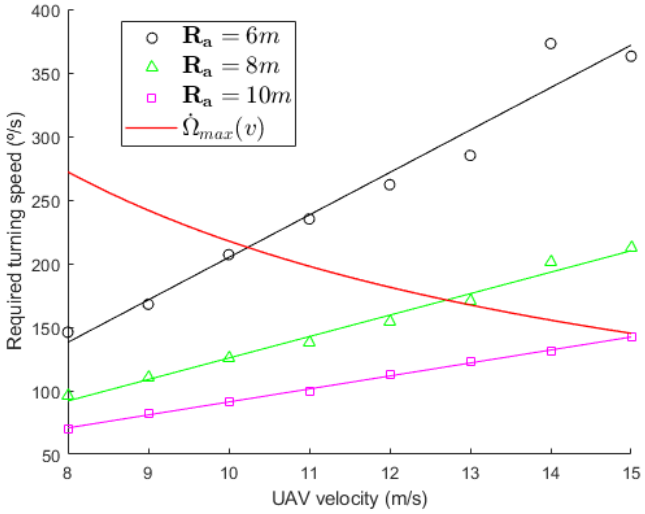


Figure 3.2: Turning rate required to avoid obstacle safely for different velocities

3.2 Numerical Model

The idea behind these studies is to evaluate how sensors perform under various conditions. In order to do that, the following procedure was followed: Firstly, 50 appropriate scenarios will be generated; Then each scenario is tested in a MATLAB simulation and an objective function will be calculated for each case; Finally these values may be summed to obtain a final evaluation of the systems performance. To test any sensor parameter, everything since step 2 must be iterated and then a global image of the influence of that parameter in the system performance is obtained. This procedure is schematically shown in Figure 3.3

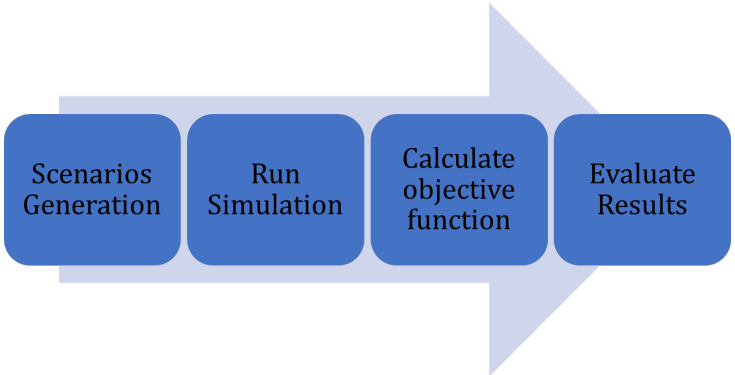


Figure 3.3: Sensor studies steps

3.2.1 Scenarios Generation

This phase is performed by an algorithm which generates appropriate scenarios for the study under consideration. Each scenario needs to include information regarding obstacles initial position, constant velocity and its corresponding radius, R_C, firstly introduced in Chapter 2. Additionally, it also has UAV's

constant speed, pre-planned path and various waypoints necessary for the collision avoidance algorithm. A graphical representation of this algorithm is exposed in Figure 3.4, which will be better explained in the following paragraphs.

The main idea with this code is to generate values for all variables described previously which will fully characterize a scenario. To do this, various bounds regarding cinematic/dimension properties for obstacles and UAV speed were defined, as it is shown in Figure 3.4. From these intervals, various random and partially random processes were taken. Partially random processes were used for obstacles velocity orientation and static obstacles initial (and constant) position. In the first case, the goal is to ensure that, for initial conditions, the obstacle's distance to the graphical window centre is not increasing. This criteria was chosen because it increases collision chances, which is important for this study. In the second case, static obstacle's initial positions are not allowed to be inside safety circles around waypoints. This is because waypoints are defined as path points which the UAV needs to pass mandatorily.

UAV path and set of waypoints are just an input for this function which adds it to a set of moving/static obstacles to create a scenario. This final product will be tested without any sensor and discarded if the UAV does not surpass any obstacle's R_S during the entire simulation. This procedure is repeated until there are 50 scenarios with an imminent collision outcome. In Figure 3.5, it can be seen 4 out of 50 scenarios that were obtained with this algorithm.

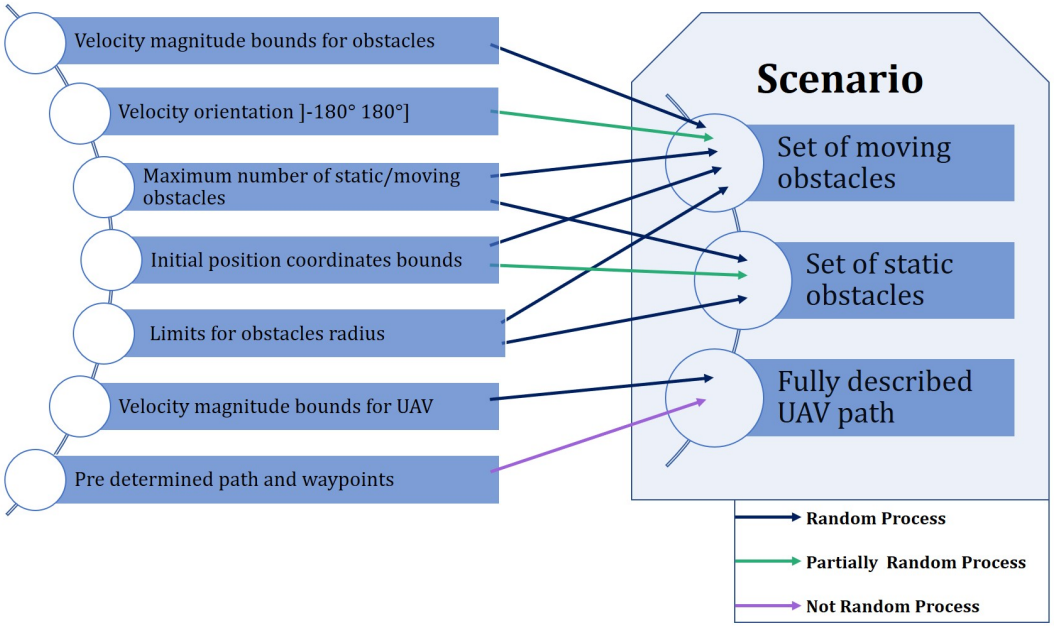


Figure 3.4: Scenario generation algorithm

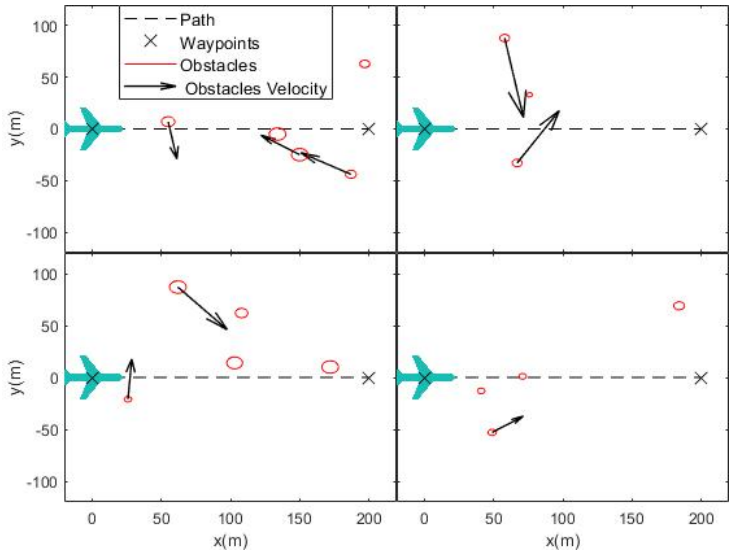


Figure 3.5: Set of 4 random scenarios

3.2.2 Simulation and Objective Function

After generating 50 suitable scenarios, there are conditions to finally test the sensing system. To do this, it was used a collision avoidance algorithm developed in [13] together with sensor models and estimators developed in [7]. At this point, it is important to find the best metric to evaluate how the system performed. The most obvious metrics to consider are the number of collisions or the number of collisions and close calls, however both metrics raise some concerns. The first one only considers collisions, completely disregarding the fact that close calls are potential collisions. The latter, does not differentiate them, putting close calls and collisions on the same level.

In order to evaluate the sensing system in a more adequate way, it was used an objective function, f, just like in [7], which is incorporated in the simulation code. By the end of all simulations, this objective function will assign a value to a specific set of parameters represented by $\boldsymbol{y}$, which include the following parameters:

1. Sensor orientation, β
2. Sensor maximum range, $\boldsymbol{R_d}$
3. Beam format - Binary Variable (Narrow or Wide) only used for ultrasonic sensors

The major problem when evaluating the sensing system performance with the metric explained previously is the fact that a close call severity is not objective. The only thing that can be assumed is that the collision risk increases when the UAV gets closer to the collision frontier and, for this reason, each close call must be evaluated independently. Additionally, there is a clear goal in a Sense and Avoidance mission, which is maximizing the minimum distance to all obstacles.

The objective function, f, is modified to incorporate these guidelines, leading to,

$$\begin{cases} f(\mathbf{y}) = g(\mathbf{y}) + p(\mathbf{y}) \\ g(\mathbf{y}) = -\sum_{i=1}^{n} d_{min}(i) \\ p(\mathbf{y}) = \phi \sum_{i=1}^{n} min(max(0, \mathbf{R_S}(i) - d_{min}(i)), \mathbf{R_S}(i) - \mathbf{R_C}(i)) \end{cases} \tag{3.3}$$

where two distinct functions are summed: goal and penalty function ($g(\mathbf{y})$ and $p(\mathbf{y})$). The first one uses d_{min}, which is an $n \times 1$ matrix with n being the total number of obstacles within the simulation and each entry the minimum distance to that specific obstacle during the entire simulation. The goal function $g(\mathbf{y})$ is symmetric to the sum of all d_{min} entries. The penalty function, $p((\mathbf{y})$ adds penalties if the UAV has passed the Close Call or Collision frontiers. These penalties weights follow a linear function just like it is shown in Figure 3.6 augmenting from zero at the close call frontier to its maximum penalty, $\phi(\mathbf{R_S} - \mathbf{R_C})$, inside all the Collision region. Finally, the parameter ϕ can be adjusted to augment or decrease the penalty function impact on $f(\mathbf{y})$. At the end, the most successful combination of sensor parameters will have the lowest values of $f(\mathbf{y})$.

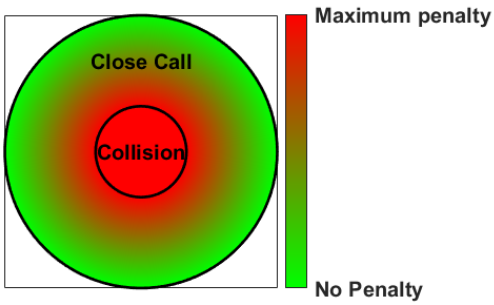


Figure 3.6: Distribution of the penalties weights

3.3 Laser Rangefinder

This section will be dedicated to the presentation of the simulations performed following the logic presented above where two symmetric laser rangefinders were used as the simulated sensor hardware. Firstly, a brief explanation of how the laser behaviour was simulated in the MATLAB environment is presented. Secondly, a series of simulations are performed to evaluate the influence of the parameters that regulate the configuration of the sensing system. The idea is to vary two of the three sensor parameters defined before ($\mathbf{R_d}$ and β) to identify direct relations between the variation of these parameter values and the objective function. Finally, the last subsection presents the optimal configuration found during these studies.

3.3.1 Model

The Matlab model used for the upcoming simulations was developed in [7]. As all the objects used in the simulations are spheres, this problem is nothing more than a simple interception between a line

and a spherical surface. Considering a spherical surface and a line given by

$$\begin{cases} \|\mathbf{x}-\mathbf{c}\|^2 = r^2 \\ \mathbf{x} = \mathbf{o} + d\hat{\mathbf{u}}, \end{cases} \tag{3.4}$$

where $\mathbf{x}$ is a generic point on the line and/or sphere, $\mathbf{c}$ is the centre point of the sphere, r is its radius, $\hat{\mathbf{u}}$ is the unit vector that defines the line direction in 3D space and d is the distance from the origin of the line. Combining both equations we get

$$d^2(\hat{\mathbf{u}}\cdot\hat{\mathbf{u}}) + 2d[\hat{\mathbf{u}}\cdot(\mathbf{o}-\mathbf{c})] + (\mathbf{o}-\mathbf{c})\cdot(\mathbf{o}-\mathbf{c}) - r^2 = 0, \tag{3.5}$$

that is a quadratic equation that can be easily solved. After solving this equation, the model returns a solution if $0 < d_{sol} < \mathbf{R_d}$. If there are two solutions in this interval, the smallest one must prevail. In real world the laser would reflect in the closest point, never reaching the other one.

After solving Equation 3.5, the reflection point with the spherical surface can be easily obtained going back to the line equation,

$$\mathbf{x_{sol}} = \mathbf{o} + d_{sol}\hat{\mathbf{u}} \tag{3.6}$$

3.3.2 Parameters

As it was mentioned in the begging of this section, a series of studies were conducted to find direct relations between the sensor parameters and the sensing system performance. Using the numerical model explained in Section 3.2, a simulated UAV with a variable sensing system configuration defined by $\mathbf{R_d}$ and β was tested in several scenarios to obtain the objective function output for each pair of parameters. Figure 3.7 illustrates how these parameters define the sensing system model and how changing their values affects the configuration of this system. Both β and $\mathbf{R_d}$ discrete domains are also described in Table 3.1.

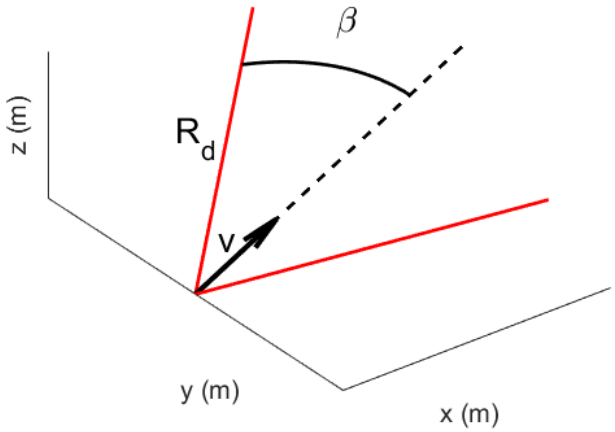


Figure 3.7: Sensing system model using laser rangefinders

Table 3.1: $\mathbf{R_d}$ and β discrete domains for the laser rangefinder simulations

	Interval	**Increment**
$\mathbf{R_d}$ (m)	[0 100]	2
β (°)	[0 90]	5

For this task, 50 scenarios were randomly created and, for each pair of β and $\boldsymbol{R_d}$, simulated to return collision percentages ($\boldsymbol{R_C}$ violation), close call + collision percentages ($\boldsymbol{R_S}$ violation) and the total value of the objective function (with $\phi = 100$). UAV speed was randomly distributed throughout the scenarios and UAV maximum turning rate, $\dot{\Omega}_{max}$, was fixed according to Equation (3.2). The action radius, $\boldsymbol{R_a}$ was set to be equal to the laser range, $\boldsymbol{R_d}$, ensuring the most cautious approach possible. Before discussing results, it is important to show all inputs used to generate the scenarios. This is shown in two parts. Figure 3.8 shows the pre-determined path, waypoints and objects initial position bounds. In addition to this, Table 3.2 contains all the other variables mentioned in Figure 3.4.

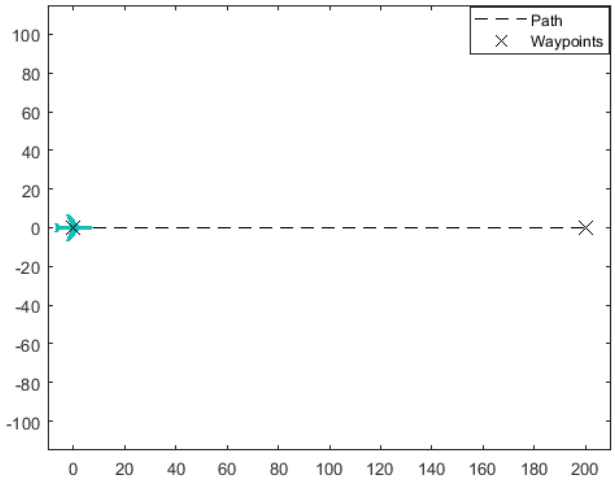


Figure 3.8: Pre-determined path in collision free condition

Table 3.2: Scenarios parameters

UAV properties	Obstacles properties			
Speed (m/s)	Collision radius (m)	Maximum number	Speed (m/s)	Velocity orientation(°)
$[8 \quad 15]$	$[0.5 \quad 2]$	3 static/3 moving	$[5 \quad 15]$	$]-180 \quad 180]$

The results obtained by running the simulations for all combinations of R_d and β are represented in figure 3.9. The optimal point that minimizes $f(\mathbf{y})$ is represented by a blue dot. Although this point is important to understand where the system is behaving better, it is not acceptable to disregard the random component present within these simulations. Taking that into account, the most correct approach is trying to find direct relations between the sensor parameters and the objective function value and then check if the optimal point is coherent with the conclusions extracted from this analysis.

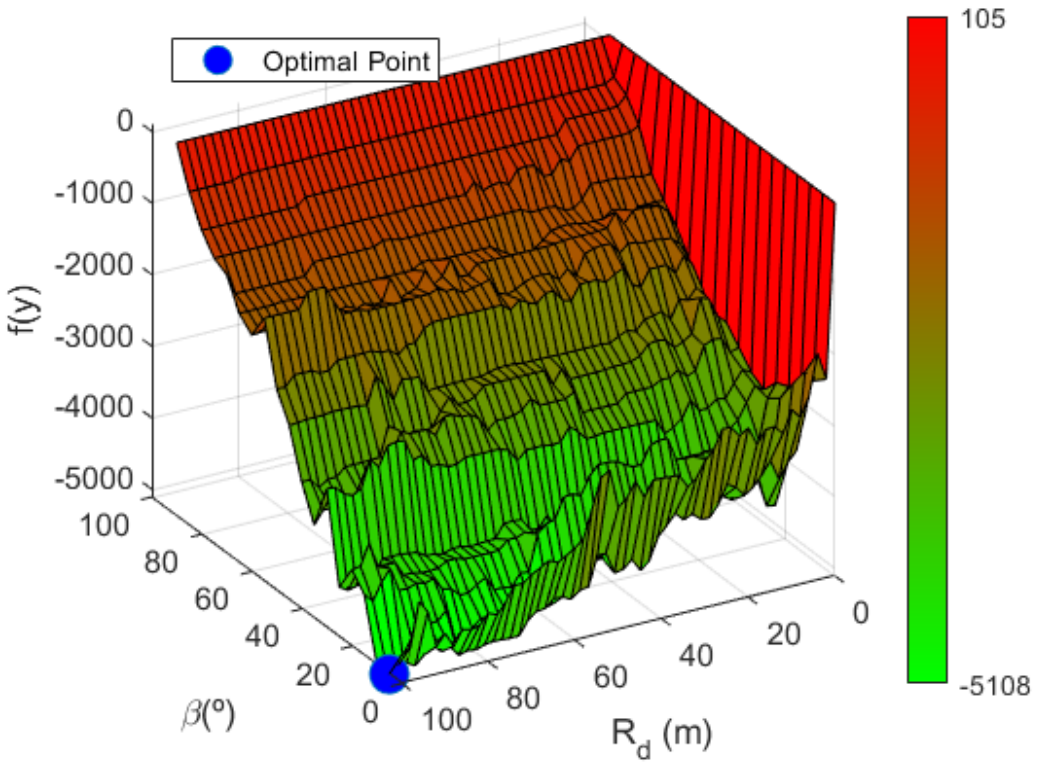
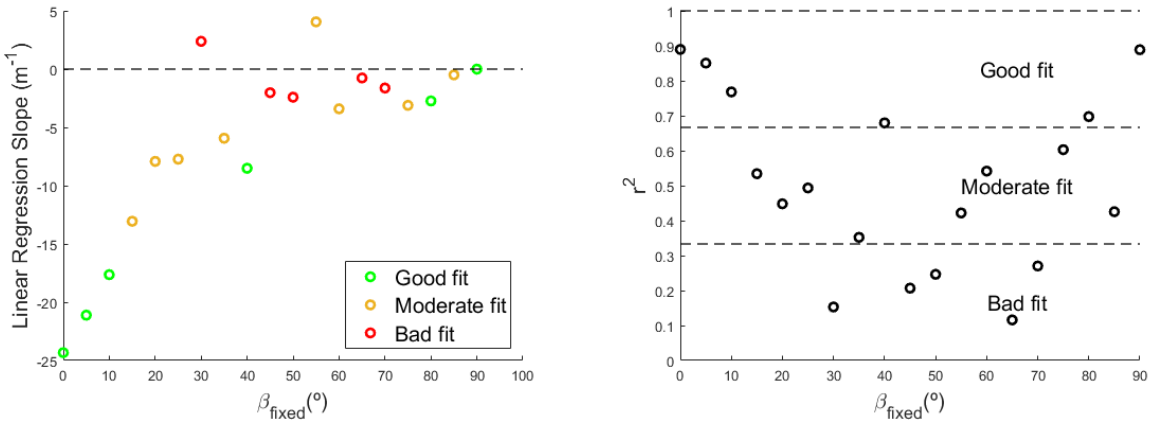


Figure 3.9: Sensor parameters influence in the objective function

Regarding the sensor orientation β, it is pretty clear looking at Figure 3.9 that the performance of the system suffers a considerable degradation for sensor angular orientations above $20°$. This behaviour is consistent independently of whatever $\mathbf{R_d}$ value is being fixed. For $\beta \leq 20°$, it is also clear that the system performance is maximized for $\beta = 5°$, where the optimal point is located. To sum up, the most clear conclusion to take from these results is that the system behaves better when the laser is more aligned with the UAV velocity. This does not mean that $\beta = 0°$ is the best possible orientation because small values of β can also cover all the obstacles present directly in front of the UAV and with the advantage of having 2 lasers working at the same time. This could obviously suffer a dramatic change if, for example, the dimensions of the obstacles were reduced.

Regarding the maximum range $\mathbf{R_d}$, it is not so easy to look at Figure 3.9 and immediately draw conclusions about some kind of relation between $f(\mathbf{y})$ and $\mathbf{R_d}$, except for the abrupt reduction of the objective function value from $\mathbf{R_d} = 0\,\text{m}$ to the remaining values. Due to this fact, a statistical analysis was performed to understand if there is, in fact, a relation between augmenting the laser range and the improvement of the sensing system performance. This analysis was then materialized isolating every function of the type $f(\mathbf{R_d}, \beta = \beta_{fixed})$, and performing a linear regression to each one to understand if the resulting slope of this approximation is negative. Figure 3.10a), represents all the slopes obtained for each β_{fixed} value whereas the r^2 parameter for each linear regression is presented in Figure 3.10b). It is important to mention that all the points with $\mathbf{R_d} = 0\,\text{m}$ were disregarded for this study.



(a) Linear Regression Slope

(b) r^2

Figure 3.10: linear Regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$(laser rangefinder case)

It can be observed in Figure 3.10a) that all the linear regressions made for $\beta_{fixed} \leq 25°$ present a negative slope, suggesting that the sensing system performance tends to augment when the laser range is increased. For $\beta_{fixed} \geq 25°$, this trend almost disappears, suggesting that the bad performance of these configurations is almost determined by their lasers angular orientation with almost no room for improvement.

Looking now at Figure 3.10 as a whole, it is clear that the linear regression was not accurate for a considerable number of cases. Nevertheless, it is also evident that the smallest values of β_{fixed} put together a well determined linear relation and the most negative slopes.

To sum up, and joining both analyses, it can be concluded that the system behaves better when the laser orientation is closer to the UAV velocity direction. Augmenting the laser range improves the performance of the system, but the relation deteriorates when β augments. These results were expected as they reflect the necessity of prioritising the direction where the UAV is moving with more speed just like a driver looks up front and not to the lateral windows. However, this methodology can always backfire if there is an obstacle hitting from the side.

3.3.3 Optimal setup

Now that there is a general understanding of how varying the laser rangefinder parameters affects the outcome of sense and avoidance missions, it is time to present the optimal configuration that is marked by a blue dot in Figure 3.9. Table 3.3 shows the optimal set of parameters, its objective function output and the concrete number of incidents that happened during those simulations.

Table 3.3: Laser rangefinder optimal results

Number of Incidents			$\mathbf{y_{optimal}}$		$\mathbf{f(y_{optimal})}$
Collisions	Close Calls	Total	$\mathbf{R_d}$ (m)	β (°)	
6	19	25	100	5	-5108

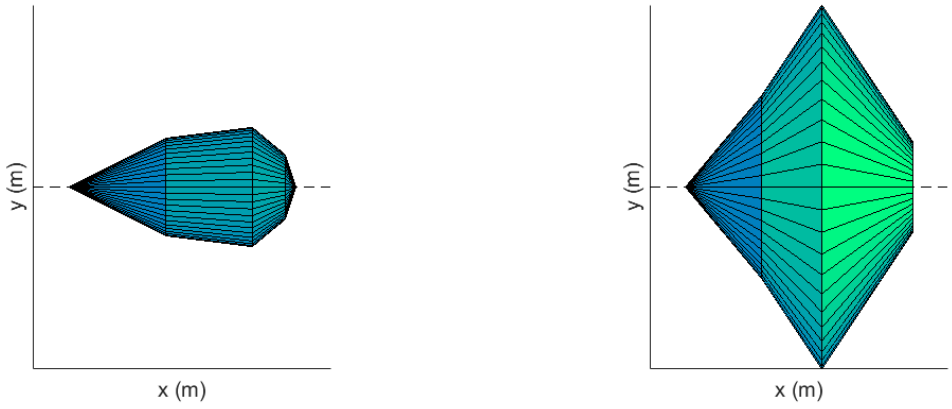
This optimal setup matches the conclusions conveyed in the last subsection as its range is exactly the maximum $\mathbf{R_d}$ value studied and its angular orientation is closer to $0°$ but not exactly zero. The total number of incidents occurred for this optimal setup is exactly 50% of the total number of scenarios tested, combining six collisions and 19 cases where the outcome of the mission is not clear. Assuming the worst outcome for all 19 close calls, it may be argued that this final optimal system is far from satisfactory. Nevertheless, considering the limitations regarding its field of view it was always going to be difficult to be successful with only a pair of lasers sensors in such a diverse universe of scenarios.

3.4 Ultrasonic Sensor

In contrast to laser sensor, whose model was previously developed in [7] in MATLAB environment, this model was not built until now. Due to this fact, this section will begin with a more in depth analysis of the ultrasonic sensor model implemented. Nevertheless, the structure of this section will be the same, starting with the ultrasonic sensor model, presenting the parameter studies afterwards and finalizing with the optimal setup obtained.

3.4.1 Model

While the laser sensor can be modelled as a simple line segment with its origin located on the UAV position, the ultrasonic sensor has a wider field of vision, making its modelling slightly more complex. With this in mind, it was selected as the best approach analysing two different types of beam patterns: narrow and wide. Both beam patterns types have axial symmetry (see Figure 3.11) and are based on MaxBotix sonar models available on the market.



(a) Narrow Beam Pattern (b) Wide Beam Pattern

Figure 3.11: Types of ultrasonic sensor beam patterns.

During the laser rangefinder studies presented previously, the laser performance was evaluated taking in consideration its range and angular orientation. In this case, the ultrasonic sensor will have an additional variable, the beam pattern format. This variable will only have two possible values: narrow or

wide, corresponding to the beam formats in Figure 3.11. Adding this extra complexity to the study will, hopefully, give a good indication on which beam pattern to choose in the implementation of our solution.

Heading now towards the target detection, there are two important considerations to address. Firstly, it is easy to predict considerable errors regarding the target position detection. This will happen because the ultrasonic sensor only outputs a distance, leaving as potential object positions all the interior beam points located at that specific distance from the UAV.

Secondly, there are some issues regarding sound reflection. According to the sound reflection law, its direction after hitting a surface will conserve the angle with the normal of that surface. Hereupon, it is clear that the ultrasonic sensor needs a perpendicular surface to have a successful detection, making the targets format important for the outcome of its mission. Due to the fact that these simulations will only use spherical shaped targets, it is important to acknowledge that the final results can only be used as a reference and other targets formats could improve or deteriorate the outcome.

Hereupon, there are conditions to start explaining how this ultrasonic sensor model will accurately recreate the real sensor behaviour. The idea consists in always verifying two conditions:

1. The presence of any spherical surface point within the sonar beam pattern;
2. The perpendicularity of the sound wave direction with its reflection surface.

In order to verify these conditions, the algorithm needs a considerable amount of computing time. To deal with this issue, it was implemented a progressively complex approach that avoids unnecessary blocks of code.

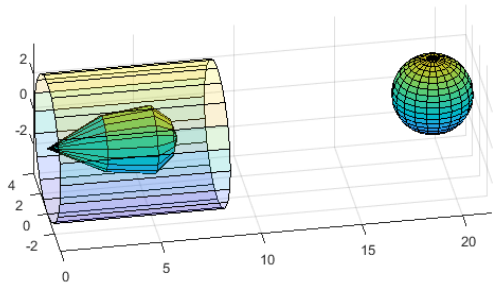
Firstly, the beam pattern is simplified to a cylinder with specific dimensions, such that the obstacle position (centre of a sphere) absence from its interior or surface, guarantees that none of the obstacle's surface points are detectable by the ultrasonic sensor (see Figure 3.12a). By doing this in the first step of the algorithm, all the complex blocks of code required to ensure those two conditions mentioned before will not be executed unless the obstacle is already close to the beam pattern.

Whenever the obstacle centre is located within the cylinder, a more precise and time consuming analysis is made to determine which segment (if any) of the spherical surface is actually inside the beam pattern. This procedure begins with the second step of this algorithm, which defines each obstacle as a list of 200 points equally spaced in latitude and longitude coordinates. This list of points is then evaluated to determine which points are inside the beam pattern (see Figure 3.12b).

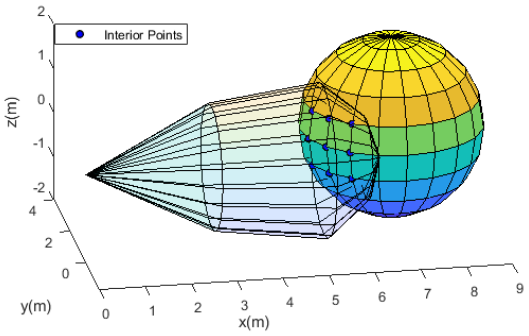
Hereupon, if any of those 200 points are actually inside the beam pattern, the algorithm passes to its third step, where a final surface is defined as the portion of the spherical surface that includes all the interior points and their adjacent ones, just like in Figure 3.12c).

Finally, the last step is responsible to address the perpendicularity issue. This is done by defining the final surface determined in the previous step as a list of points with a 25 times larger sampling ratio in each spherical coordinate (in relation to the first list of points). The idea is to calculate the angle between the sound wave direction and the sphere tangent surface in that specific point. If this angle is close to $180°$ (tolerance of $5°$), the point is assumed to be detected by the sonar. Although the vast majority of this final surface points are already inside the beam pattern, a second evaluation must be made for all

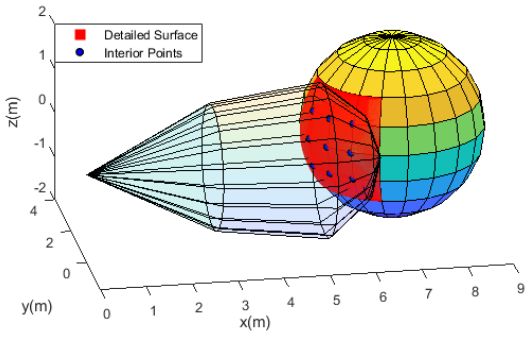
points that were classified as detectable. From all the points that surpassed these filters, the closest one to the UAV position is selected as the Detected Point. The UAV reference point will be located in the beam pattern axis with the same distance to the UAV position as the Detected Point and will be seen as the reflection point for the Replanning Algorithm (see Figure 3.12d).



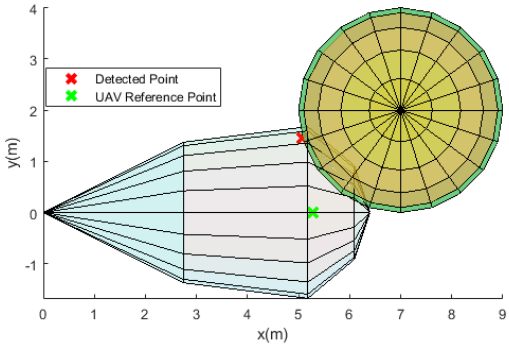
(a) First step



(b) Second step



(c) Third step

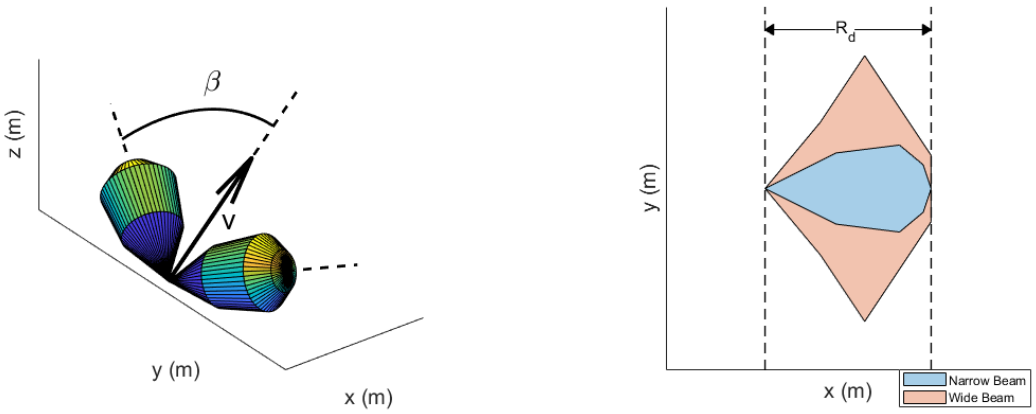


(d) Fourth step

Figure 3.12: Sonar model steps

3.4.2 Parameters

Following the same principle used with the laser rangefinder simulations, 50 random scenarios were also created with the same input variables as those exposed in Table 3.2. The objective function parameter, ϕ, was tuned with the same value as in Section 3.3 and the UAV maximum turning rate was defined following Equation 3.2. The action radius, $\boldsymbol{R_a}$ was set to be equal to the ultrasonic sensor range, assuring an immediate response by the replanning algorithm. As it was explained in the section above, the sensor characteristics that will be evaluated are: the sensor orientation (β), its maximum range ($\mathbf{R_d}$) and its beam format (wide or narrow) (see Figure 3.13). $\mathbf{R_d}$ and β discrete domains are described in Table 3.4.



(a) Ultrasonic Sensor orientation (β)

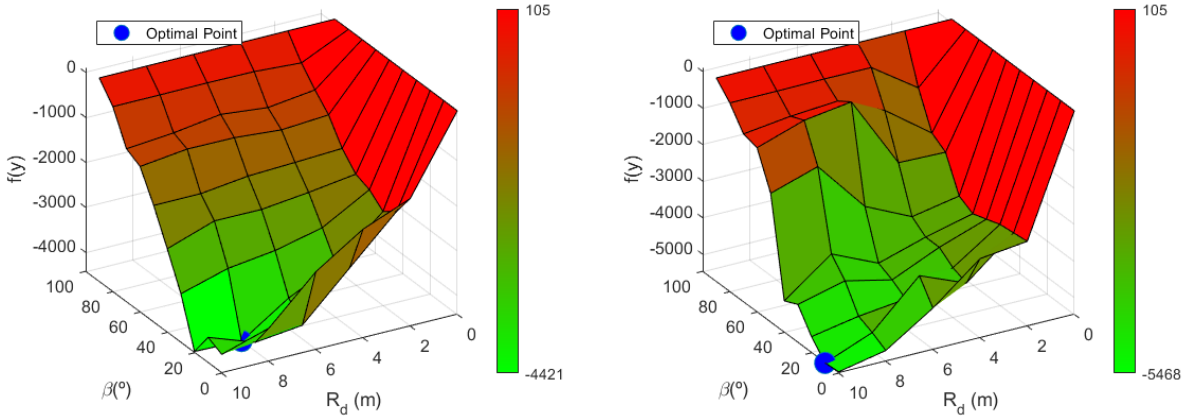
(b) Ultrasonic Sensor Range (R_d)/Beam Format

Figure 3.13: Ultrasonic sensor parameters

Table 3.4: $\mathbf{R_d}$ and β discrete domains for the ultrasonic sensor simulations

	Interval	**Increment**
$\mathbf{R_d}$ (m)	[0 10]	2
β (°)	[0 90]	10

The results of these simulations are expressed in Figure 3.14 where the blue dots represent the optimal parameters. With regard to β, it is obvious looking at Figure 3.14, that there is a clear minimum at $\beta = 20°$ and $\beta = 10°$ for the narrow and wide beam patterns respectively and that is valid for any range value. When augmenting these angles, the results deteriorate in both cases, although the wide beam pattern can sustain a reasonable performance until $\beta = 40°$.



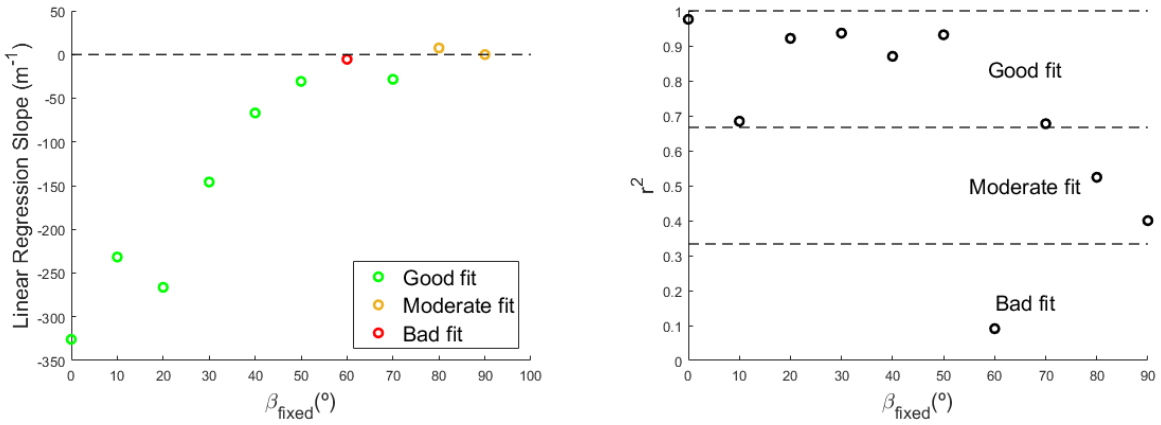
(a) Narrow Beam Pattern

(b) Wide Beam Pattern

Figure 3.14: Ultrasonic sensor parameters influence in the objective function.

Similarly to what was done with the laser rangefinder studies, a linear regression approach was used to find any possible relations between $\mathbf{R_d}$ and the sensing system performance.

After isolating several functions of the type $f(\mathbf{R_d}, \beta = \beta_{fixed})$ and removing all points with $\mathbf{R_d} = 0\,\text{m}$, the graphics illustrated in Figures 3.15 and 3.16 were obtained. In Figure 3.15a), it can be observed that the great majority of the plotted points correspond to a negative slope, meaning that augmenting $\mathbf{R_d}$ is causing a decrease of the objective function value or, in other words, improving the performance of the sensing system. Additionally, the magnitude of these slopes tends to decrease for larger β_{fixed} values, which leads to the same conclusion explained in Section 3.3. The bad performance of some configurations are almost guaranteed with larger values of β and augmenting $\mathbf{R_d}$ does not produce any relevant effect on the global performance of the system.

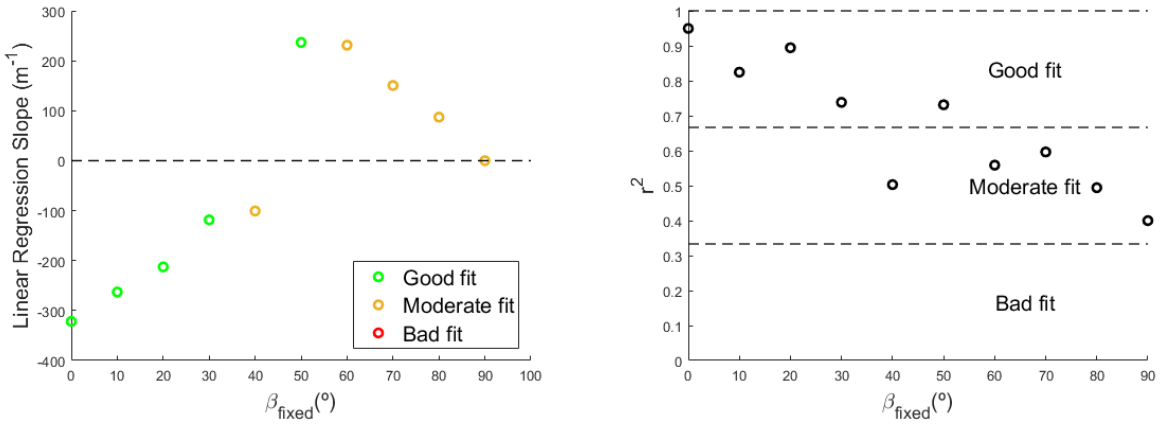


(a) Linear Regression Slope

(b) r^2

Figure 3.15: Linear regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$ (narrow beam pattern case)

Looking now at Figure 3.16 it may be concluded that, for $\beta_{fixed} \leq 40°$, augmenting $\mathbf{R_d}$ is beneficial for the systems performance. On the opposite side, for $\beta_{fixed} \geq 50°$, this relation reverses completely which can be seen by all positive slopes presented within this interval. This may be a result of the high probability of errors when trying to exactly locate a detected object within a wide beam pattern. These errors may cause the UAV to ignore obstacles present in their pre-planned path or route it to a unnecessary avoidance path that will cause even more risks to its mission. When $\mathbf{R_d}$ is augmented, this type of errors may be even more evident, which leads to weaker performance. Lastly, it is important to remember that, when β_{fixed} is relatively small ($\leq 20°$), the system behaves exactly like the narrow beam pattern and the laser rangefinder. Augmenting the sensor range is correlated with an overall improvement of the systems performance.



(a) Linear Regression Slope

(b) r^2

Figure 3.16: Linear regression for $f(\mathbf{R_d}, \beta = \beta_{fixed})$ (wide beam pattern case)

3.4.3 Optimal Setup

All the information relative to the optimal points identified previously in Figure 3.14 can be found in Table 3.5. The most important aspects to take from these optimal points are the fact that both are located at the maximum value of $\mathbf{R_d}$ or close to it, which demonstrates the advantage of increasing this parameter. Additionally, both beam patterns worked better for relatively low values of β (10° to 20°) and the wide beam had slightly better results than its narrow counterpart. As expected, after seeing the objective function outputs of both optimal setups, the wide beam pattern had considerable less incidents, although the number of collisions is not that different.

Table 3.5: Ultrasonic sensor optimal results

	Number of Incidents			$\mathbf{f(y_{optimal})}$	$\mathbf{y_{optimal}}$	
	Collisions	Close Calls	Total		$\mathbf{R_d}$(m)	β(°)
Narrow Beam Pattern	8	17	25	-4421	8	20
Wide Beam Pattern	7	9	16	-5468	10	10

3.5 Final Remarks

This chapter began with an evaluation regarding the UAV model and its capability to avoid an obstacle in perfect conditions of detection and localization of the intruder which is moving in a head-on collision trajectory with the UAV, and with the same velocity magnitude but in the opposite direction. It was proven that the UAV needs at least 10 m of anticipation to ensure that the obstacle can be avoided with no collisions/close calls and without exceeding the maximum acceptable g force during the avoidance manoeuvre. This does not mean that a sense and avoidance system which does not guarantees an action radius of 10 m must be discarded, as most of collision routes that it will face are not as critical as the ones analysed here. Nevertheless, it is now clear that there is a great advantage for all systems with

$\mathbf{R_d} \geq 10\,\text{m}$ as this guarantees that the action radius, $\mathbf{R_a}$, can be set to $10\,\text{m}$ or even more.

The tests performed after this are related to the real performance of these sensors when faced with several scenarios that would lead to potential collisions/close calls. With this in mind, the system was simulated for all different configurations of sensor range ($\mathbf{R_d}$), angular orientation (β), and beam format (specific for the ultrasonic sensor). It was proven that there is a clear advantage for all sensors when selecting relatively small values of β ($0° \leq \beta \leq 30°$). With regard to the sensors range ($\mathbf{R_d}$), it was evident that augmenting the value of this parameter is directly related with an improvement on the systems performance. Additionally, it was observed that this relation starts to fade when the angular orientation approximates to $90°$. The exception from these trends was identified in the simulations with an ultrasonic sensor with a wide beam pattern. The results from this case revealed that for a specific interval of β ($50°\beta \leq 90°$), augmenting the sensor is actually prejudicial to the global performance of the system. This may be related with a considerable amount of errors regarding the localization of a detected object when using sensors with low directionality as this one.

Comparing now the optimal results for the laser rangefinder and both types of ultrasonic sensors, it is clear that the wide beam pattern ultrasonic sensor achieved the best performance. This cannot be seen as a surprise due to the fact that its field of view is considerable larger than its counterparts. Nevertheless, it is almost impossible to find an ultrasonic sensor model that manages to gather such a wide field of view and a $10\,\text{m}$ range. When limiting this sensor to more realistic ranges (up to $4\,\text{m}$), both the laser rangefinder and the narrow beam ultrasonic sensor managed to surpass it in terms of performance. After imposing this limitation, the laser rangefinder appears as the best option and the narrow beam pattern is now the best option for ultrasonic sensors.

To sum up, a good sensor configuration must have the greatest possible range mixed with an angular orientation relatively close to the UAV velocity. It must be said that an even better detection system could have been achieved by considering a combination of multiple sensor, that was found to be outside the scope of this work.

Chapter 4

Hardware and Software Implementation

This chapter will be fully dedicated to all the hardware and software needed to execute the proposed tasks. The goal is to present available options on the market and dissect with more detail the chosen ones.

4.1 Sensor Hardware

In Section 2.2, several sensors suitable for this type of project were firstly introduced. This was a short list of four types of sensors (RADAR, LIDAR, Ultrasonic Sensor, Laser Rangefinder). From these four options, only the Ultrasonic Sensors and the Laser Rangefinders were considered due to their availability for the project.

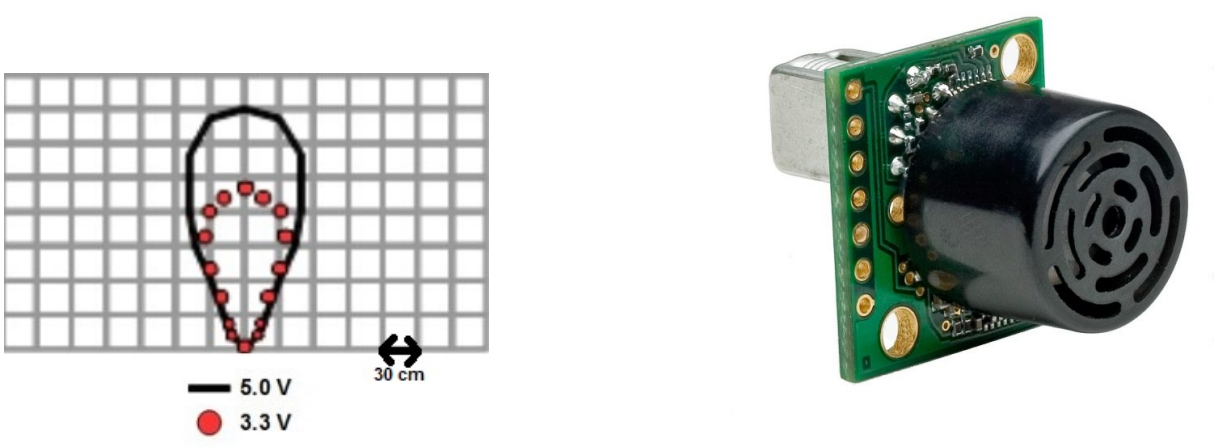
4.1.1 Ultrasonic Sensors

MaxBotix [25] is a company known for producing a series of ultrasonic sensors (I2CXL-MaxSonar-EZ), which offers an easy to use I2C interface. They provide high acoustic power output, along with real-time auto-calibration for changing conditions, that ensure users receive the most reliable (in air) ranging data for every reading taken. The low power $3 - 5.5\,\mathrm{V}$ operation provides very short to long-range detection and ranging, in a compact form factor. The I2CXL-MaxSonar-EZ detects objects from $0\,\mathrm{cm}$ to $765\,\mathrm{cm}$ and provides sonar range information from $20\,\mathrm{cm}$ or $25\,\mathrm{cm}$ out to $765\,\mathrm{cm}$ with $1\,\mathrm{cm}$ resolution [26].

Table 4.1: Qualitative analysis of I2CXL-MaxSonar-EZ series[26]

People Detection Wide Beam High Sensitivity	Best Balance	Large Targets Narrow Beam Noise Tolerance	Very Large Targets Narrow Beam Extreme Noise Tolerance
MB1202	MB1212	MB1222	MB1232	MB1242

From these five options presented in Table 4.1 one sensor must be chosen. MB1242 (see Figure 4.1) was selected due to its better performance for larger targets and its extreme noise tolerance. Figure 4.1 also illustrates a beam pattern on a 30 cm grid. It must be highlighted that this beam pattern was obtained for a 8.9 cm dowel, and for this reason, it is only illustrating the sensors sensitivity and beam wideness but not its maximum range.



(a) Beam pattern 8.9 cm dowel

(b) Physical model

Figure 4.1: MB1242 ultrasonic sensor [26]

In addition to what has already been unveiled, the MB1242 ultrasonic sensor includes:

- Readings rate up to 40 Hz for up-close objects. 15 Hz rate for full range;
- Operational temperature from 0 ℃ to 65 ℃;
- Ultrasonic signal frequency of 42 kHz;
- Relatively small dimensions and weight. It fits inside an 11.05 cm^3 cube and weights around 5.9 g;
- Maximum range around 640 cm.

4.1.2 Laser Rangefinder

Lightware [27] offers a wide range of laser rangefinder models, which are worth analysing individually. In [7], 3 Lightware models were presented and its specifications can be found in Table 4.3. Considering that, at this stage, these devices are meant to be used only as a simple laser rangefinder (not a LiDAR) it is not preponderant to have such high measurement frequencies as SF30/C and SF30/D models. Now,

focusing on the other specifications, the best two are the LW20/C and the SF30/D with a range of 100 m and 200 m, respectively. Although the LW20/C range is 100 m shorter than the SF30/D, its lower cost and weight allied with an included protective casing (see Figure 4.2), can level both options. Finally, it is important to mention that this sensor can communicate either with I2C or Serial interfaces, which facilitates the task of integrating it with the Flight Controller that will be selected in the next section.

Table 4.2: Comparison of different laser rangefinders [7]

	Lightware LW20/C	Lightware SF30/C	Lightware SF30/D
Protective casing	yes	no	no
Mass (g)	20	35	35
Range (m)	100	50	200
Maximum measurement frequency (Hz)	388	20010	20000
Cost(€)	255	275	365

Table 4.3: Comparison of different laser rangefinders [7]

Figure 4.2: Lightware LW20/C [28]

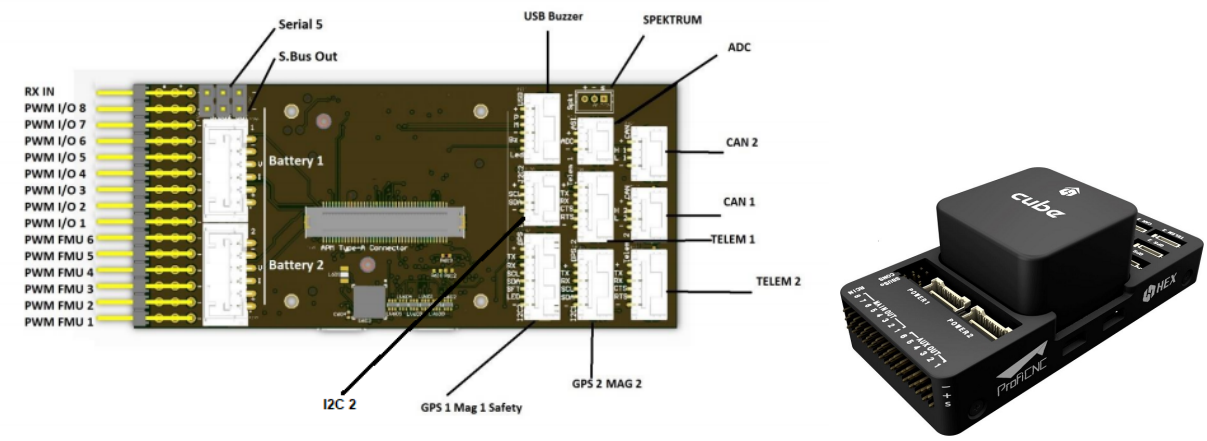
4.2 Flight Controller

In order to perform experiments with distance sensors, it is required to integrate them in a flight controller. In this subject, there are several options, all products of the Pixhawk project sponsored by Dronecode Foundation [29] members. Although, there are more recent releases of the Pixhawk series, the one chosen was the Pixhawk 2.1, now better known as the Hex Cube Black, that is exposed in Figure 4.3 This flight controller may be divided into three key components:

- **Pixhawk FMU (Flight Management Unit) Main Board**: The main features of this component are its 32 bit microcontroller, 256kB of RAM, 2MB of flash, an integrated accelerometer/gyro and an altimeter;
- **Vibration Damped IMU board**: Extra sensors such as accelerometer, magnetometer, gyroscope

and altimeter which will reject vibrations due to its location in a vibration damped board. This extra information free of vibrations will generate redundancy on the measurements, augmenting the system overall reliability;

- **I/O ports** Includes 14 PWM servo outputs which can be used to power the vehicle motors and propellers, 2 CAN Bus interface, 2 I2C ports etc [30].



(a) Main board and several I/O ports

(b) External perspective

Figure 4.3: Flight Controller [30]

4.3 Electrical Layout

After the flight controller and sensors are selected, the electrical layout can be designed. Using all the connections and extra devices (GPS and Power module) which come with the Cube Black package, it is possible to connect all parts following the diagram in Figure 4.4. The GPS is connected to the GPS 1 port while the Laser Rangefinder and Sonar are connected to the I2C 2 port, individually or together using an I2C bus splitter module. For the power supply, it is required to use a power module to provide the flight controller a regulated power source while it powers the electric speed controller (ESC) simultaneously.

On one hand, the flight controller supply is made with a 6-wire cable which includes two 5V connections, two ground connections and a battery voltage and current connector (V and I respectively). These last two connections are important to provide the flight controller information regarding the battery status.

On the other hand, the ESC also receives a battery power supply and, using a PWM signal from one of the PWM I/O entries, it controls the motor. It is important to add that, in Figure 4.4, there is only one motor connected to the ESC, however, if more than one motor is needed, the ESC must receive one PWM signal for each one.

Additionally, there is a PPM Sum Receiver, that needs to be connected to an RX IN entry. This device translates PWM signals received from the radio receiver into a single PPM signal, which can then be interpreted by the Fligh Controller.

Finally, the telemetry module sends and receives radio signals to another telemetry module connected to a ground station, sharing real-time data and allowing the user to send commands to the vehicle. These communications are performed using the MAVLINK protocol [31] which is already used for all other links between on board components. This protocol is another Dronecode Foundation project and it may be defined as a binary telemetry protocol designed for resource-constrained systems and bandwidth-constrained links [31].

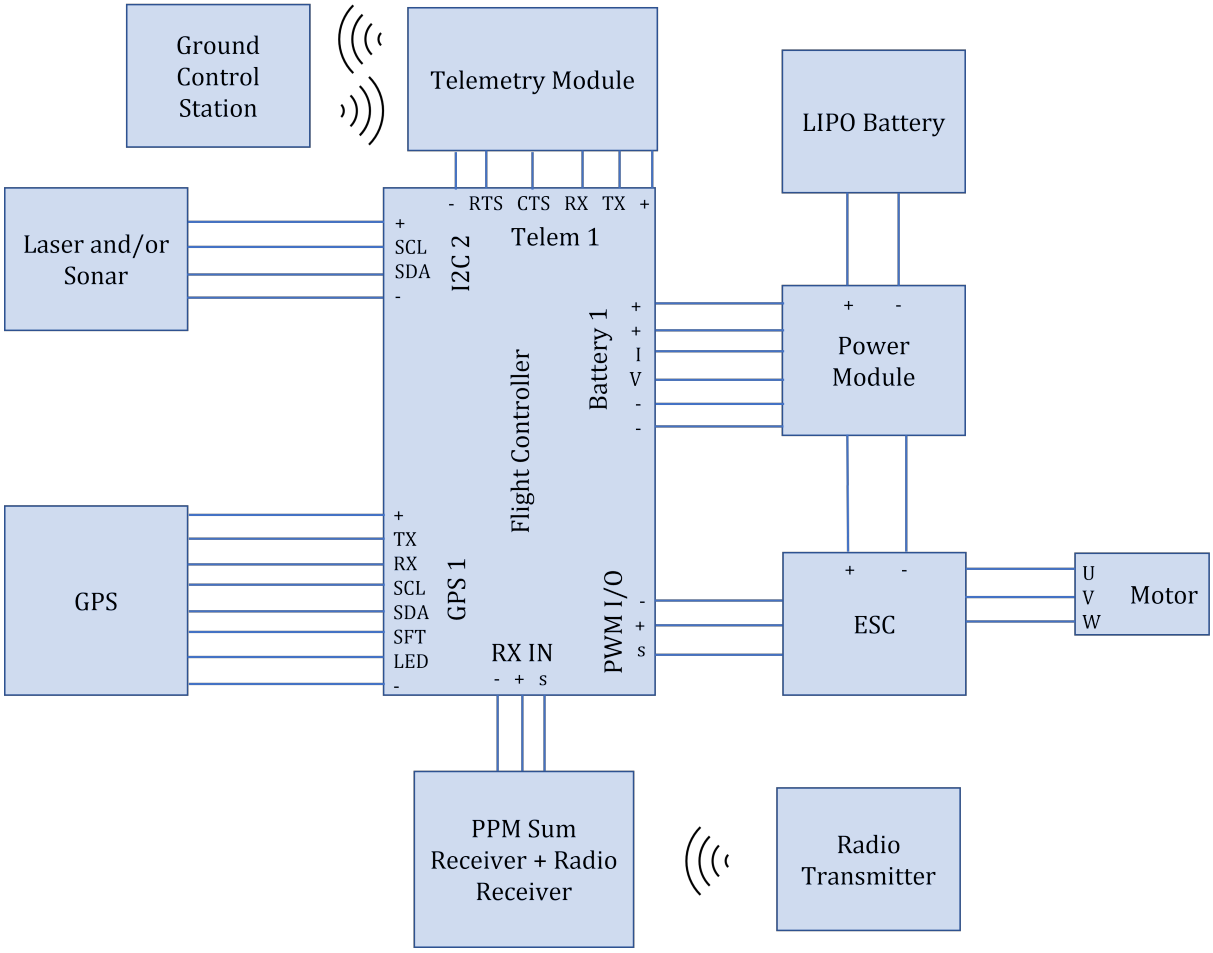


Figure 4.4: Electrical scheme

4.4 Flight Controller Software

There are two major open source options to upload in the Pixhawk 2.1 flight controller: Ardupilot and PX4. The ArduPilot Project [32] provides an advanced, full-featured and reliable open source autopilot software system. The first ArduPilot open code repository was created in 2009 and since then, it has been developed by a team of diverse professional engineers, academics, computer scientists, and other members of our global community. It is capable of controlling almost any vehicle system: conventional and VTOL airplanes, gliders, multirotors, helicopters, sailboats, powered boats, submarines, ground

vehicles and even balance-bots. The supported vehicle types frequently expand as use cases emerge for new and novel platforms [33].

The PX4 [34] is a more recent product and one of several projects directed by the Dronecode Foundation. The project provides a flexible set of tools for drone developers to share technologies to create tailored solutions for drone applications. It provides a standard to deliver drone hardware support and software stack, allowing an ecosystem to build and maintain hardware and software in a scalable way. Its most important characteristics include a modular architecture, autonomy stack, configurability, real world validation and powerful safety features [35].

After evaluating both softwares two arguments were elaborated. Firstly, it might be safer to choose the oldest software, as it is probably more robust due to its larger testing and validation. In contrast, the PX4 is more recent and it is being developed in a scalable way. For this reason it is probably the most promising software.

These two arguments were taken into account, the PX4 was selected as it offers the most active user development community. Although today, it is probably safer to choose Ardupilot, it is believed that this advantage will most likely disappear in the near future making the PX4 the most logic option.

4.5 Ground Control Software

A ground control station can be seen as an interface between a flight controller and a human operator, working almost as a virtual cockpit. It is typically a software application installed on a ground based computer that communicates with the flight controller via wireless telemetry. This interface is an highly relevant tool, which allows the human operator to send commands to the vehicle during flight and receive useful information like position, velocity, acceleration or any other sensor data.

Just like flight controller firmwares, there are two main options for ground control softwares: QGroundControl [36] and Mission Planner [37]. They both use the MAVLink communication protocol to interact with the flight controller and are compatible with both PX4 and Ardupilot. Despite this fact, it is well known that QGroundControl is a project of the DroneCode Foundation just like PX4 and Mission Planner is an Ardupilot product. Due to this relation, it was decided that QGroundControl is probably the best option, given that PX4 was the selected firmware for the flight controller so a seamless integration is expected between flight controller and ground control software.

QGroundControl Windows version is easily installed opening the executable file that can be downloaded from the QGroundControl user guide [38]. There are also versions for Mac OS X, Ubuntu Linux, Android and iOS.

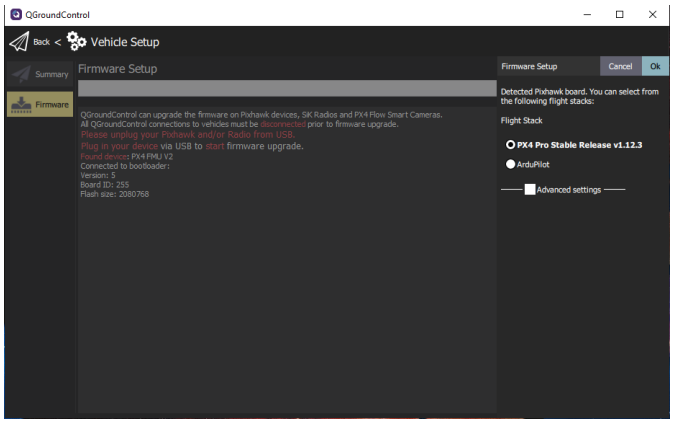
Within the QGroundControl environment, it is possible to upload the PX4 firmware to the flight controller as it can be seen in Figure 4.5(a). To do that, the flight controller must be connected to the computer via USB. In the QGroundControl application the vehicle setup tool must be selected and then in the Firmware Setup option the user can simply select which firmware he wants to upload to his controller.

After this, QGroundControl asks the user to select which airframe corresponds to his vehicle (there

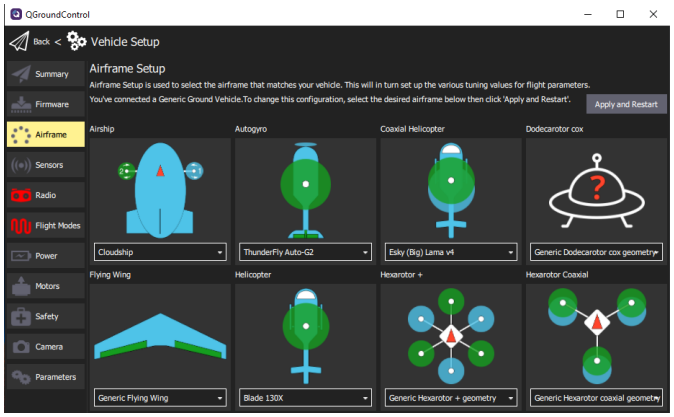
are more than 30 options supported by PX4 including the ones seen in Figure 4.5(b)).

Finally, the user must perform several calibration processes regarding the controller integrated sensors, radio receiver, flight modes, power, motors, among others. Figure 4.5(c) shows the calibration of the gyroscope that is installed inside the flight controller.

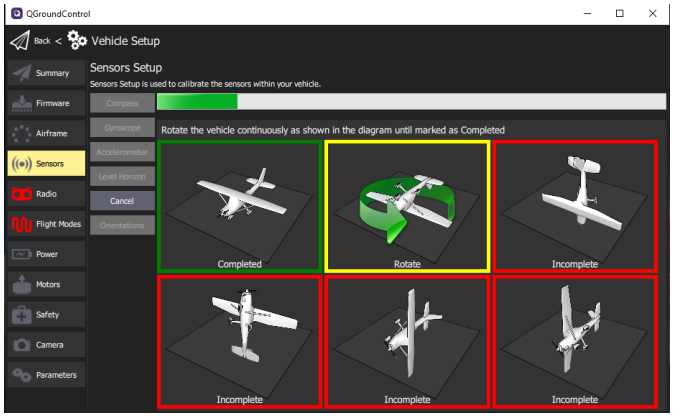
In addition, it is important to mention that QGroundControl offers a simple way to interact with the PX4 firmware changing various parameter values, which can be done in the parameters tab inside the vehicle setup option as it is shown in Figure 4.5. Some key parameters for this study are SENS_EN_MB12XX and SENS_EN_SF1XX which enable MB1242 and LW20/C respectively. There is also a family of parameters started with EKF2_RNG_ which define variables such as the rangefinder (Sonar or Laser) offset position from the vehicle centre of gravity, noise properties of its measurements and approximate data delay.



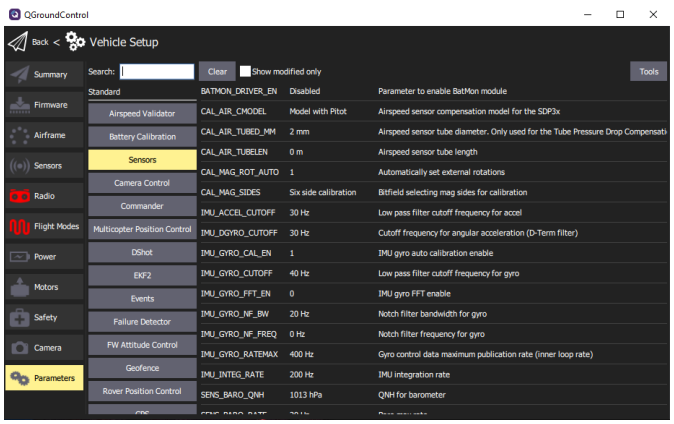
(a) Firmware upload



(b) Airframe selection



(c) Sensors calibration



(d) Changing parameters

Figure 4.5: QGroundcontrol environment

Chapter 5

Sensor Experiments

In this chapter, the idea is to perform a large number of experiments in order to validate the real capabilities of a Sense and Avoidance system with the hardware and software presented in the previous chapter. Firstly, the sensors will be tested individually to understand their real limitations. After this, the complete system can be tested. Due to the risk associated with a flight test, the experiments will be based on ground tests using a simple rover.

5.1 Bench Tests

Even though, there are proper data sheets [28] [26] for each one of the sensors used, it is prudent to perform various experiments to obtain real limitations in terms of ranging and field of view for the selected devices. Hereupon, the idea with these experiments is to determine the sensors detection rates and the precision/accuracy of their measurements. These data will be collected for various distances and angles.

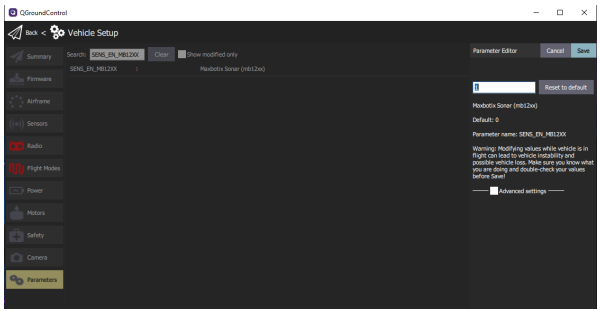
5.1.1 Ultrasonic Sensor

Before anything else, it is important to describe all the steps needed to integrate this kind of sensor with the flight controller.

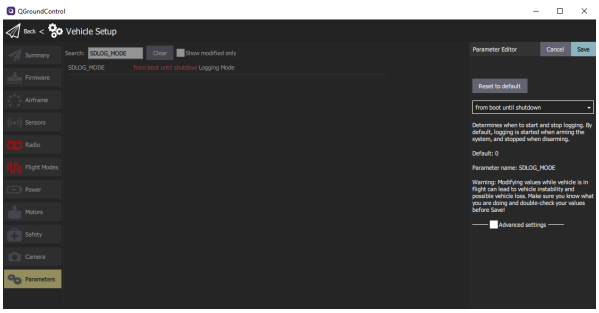
Firstly, the MB1242 ultrasonic sensor must be connected to the Pixhawk 2.1 I2C 2 port using a GH cable just like those that come with the Pixhawk 2.1 kit. For this specific task, it is highly recommended to confirm several times if all individual wires are coherent between Sonar and flight controller. This is specially important due to the fact that most of GH cables for I2C connections which come with the pixhawk kit have the red cable on the left side of its plug while the pixhawk I2C 2 port requires a ground connection on its left side (see Figure 4.3).

Secondly, the ultrasonic sensor must be enabled within the QGroundControl environment. To do this, the user must access the vehicle setup section and, within the parameters tab, set the SENS_EN_MB12XX to 1, as illustrated in Figure 5.1(a). Additionally, it is required to activate the SD logging mode by changing the SDLOG_MODE parameter following the same path as explained above (see Figure 5.1(b)).

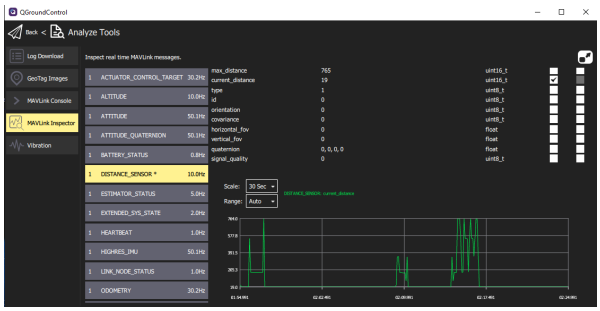
During the experiments, it is useful to monitor what the sensor is measuring in real-time and QGroundcontrol offers a great tool to do that. The user only needs to access the MAVLINK inspector tool which is inside the Analyze Tools section and select its desired variable (DISTANCE_SENSOR in this case). Most of these variables have several sub variables inside them, which can be selected to follow their live status (see Figure 5.1(c)). Finally, these files stored by the flight controller onto the SD card must be downloaded to the computer using the Log Download tool inside the Analyze Tools section as illustrated in Figure 4.5(d).



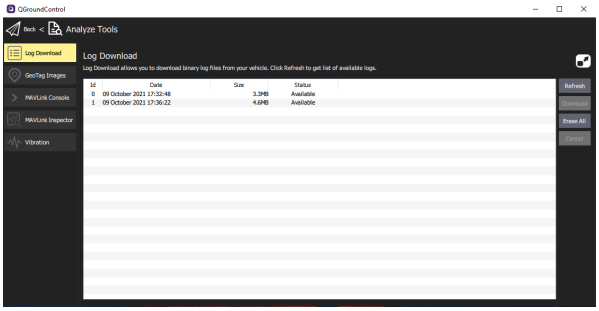
(a) MB1242 activation parameter



(b) SD logging activation parameter



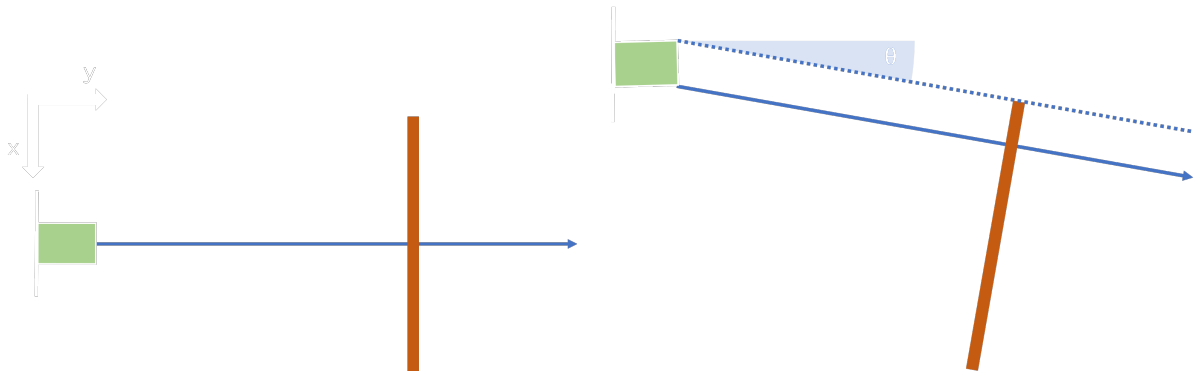
(c) MAVLINK inspector



(d) Log Download tool

Figure 5.1: QGroundcontrol setup for MB1242

After explaining all particularities related to this sensor's connections and its activation in Qgroundcontrol, it is possible to proceed with the experiments explanation. Figure 5.2(a) demonstrates an experiment where the object to detect is in front of the sensor. In Figure 5.2, the idea is to determine the sonar capability of detecting an object which has an angular deflection (θ) in relation to the sensor. In both cases, the target object is a square wooden board with a $46\,\mathrm{cm}$ side and $2\,\mathrm{cm}$ thick as seen in Figure 5.2(c). This board is always perpendicular to the blue line, which represents the minimal distance between the sonar and the object. During these experiments, the board was positioned at several distances from the Sonar ($20\,\mathrm{cm}$ to $760\,\mathrm{cm}$) and kept at each position for 30 seconds. The idea with this approach was to obtain, for each position, the fraction of time where the sensor was actually detecting its target and how much these measurements were deviated from the correct distance. All the sensor data collected during this period of time was stored in a ulog file format using the Pixhawk SD card. Then, these files were transferred to a computer, using QGroundcontrol's Log Download tool. Finally, using MATLAB these files were analysed and information such as detection rate and average absolute error was withdrawn from the original ulog file.



(a) Straight on obstacle

(b) Off-set obstacle

(c) Experimental setup

Figure 5.2: Ultrasonic sensor bench test

Before proceeding to the results presentation, there are some considerations to take into account first. Due to the fact that these results were obtained with a statistical approach, there is the need to set a minimum detection rate which is considered satisfactory for a specific task. For this particular project, it was defined as a precautionary measure that only a perfect detection rate (100%) would be enough to guarantee that the object will in fact be detected. Furthermore, it is important to consider that the target material could affect the performance of this sensor. In [39] it is mentioned that this sensor's ideal surface to detect is hard, smooth and non-porous. Although wood is not a perfect example of an ideal surface, its properties are not far from that category. For other types of materials, this case might not be the same and their physical properties should be evaluated to understand if these results can be extrapolated. The last important aspect to refer is the fact that a targets rotation within its inertial referential affects this sensor performance. The sonar can only detect a target if the emitted sound is reflected back. Following the sound reflection laws this can only be possible if the normal vector of the surface in question is aligned with the emitted sound trajectory until it reaches the target.

Following the logic explained previously, the sonar was tested and the key results can be observed in the next figures.

Figure 5.3 shows the sensor's detection rate for various distances and orientations. As expected, the sensor performed better when the obstacle was completely in front of it, achieving a maximum range of 660 cm (slightly better than the 630 cm stated in the data sheet). Additionally, the maximum range decreased when augmenting θ, which was also an expected behaviour.

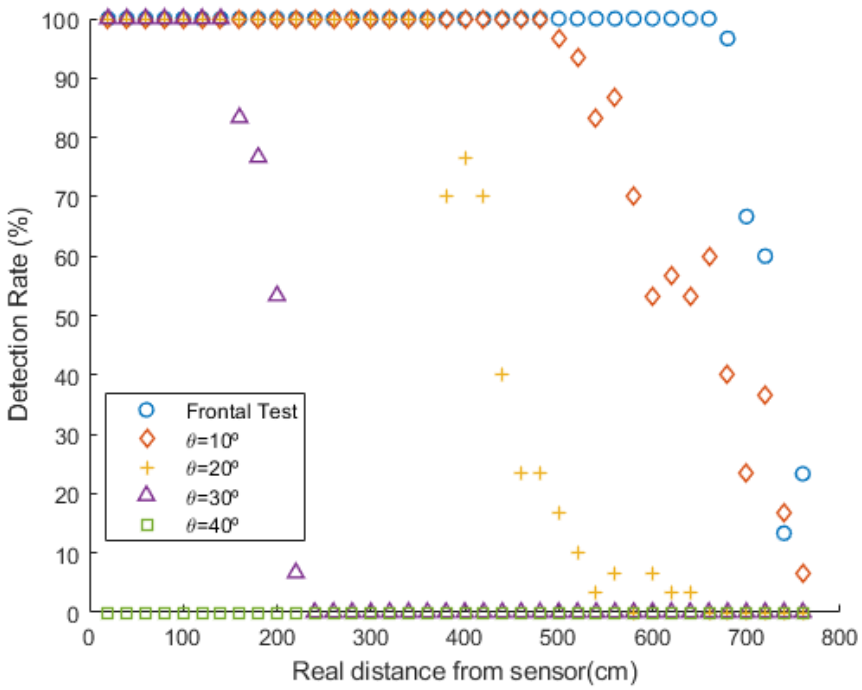


Figure 5.3: MB1242 detection rate from several distances and orientations

Moreover, this sensor proved to be very directional as it stopped detecting any targets at $\theta = 40°$. Using each maximum range obtained for its corresponding orientation of the obstacle, an experimental beam pattern was drawn as seen in Figure 5.4. Beam patterns like this, can be extremely useful for sense and avoidance missions, as they limit their detection volume to a well determined direction, so the device that is controlling it, knows with a considerable precision in which direction the target is located.

Finally, Figure 5.5 shows the average absolute error at each distance from the sensor. It can be said that it was not detected any relation between the real distance from the sensor and its error as all points seem to be almost randomly dispersed within the graphics window. Moreover, the average absolute error never surpassed 4 cm, which is a really good margin of error.

To sum up, this sensor capabilities were found to match the expectations presented in Chapter 4, mixing a narrow beam pattern with a maximum range above 6 m.

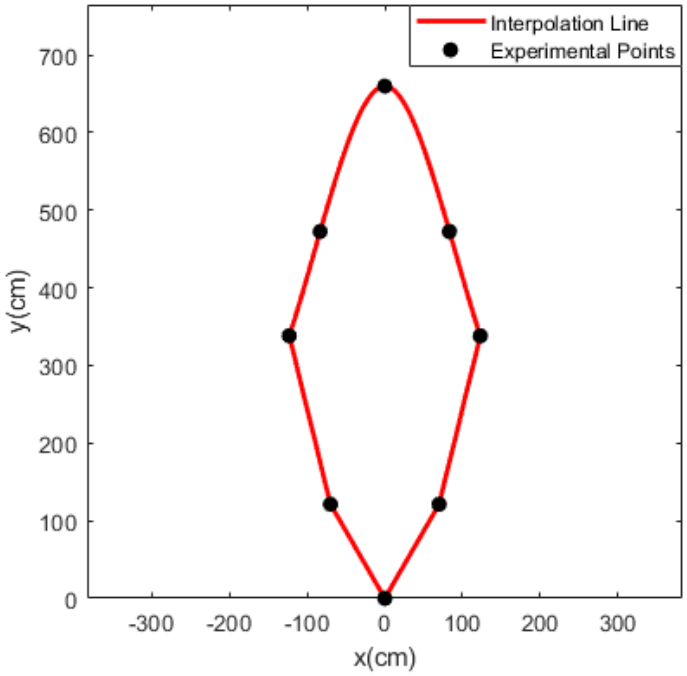


Figure 5.4: MB1242 experimental beam pattern

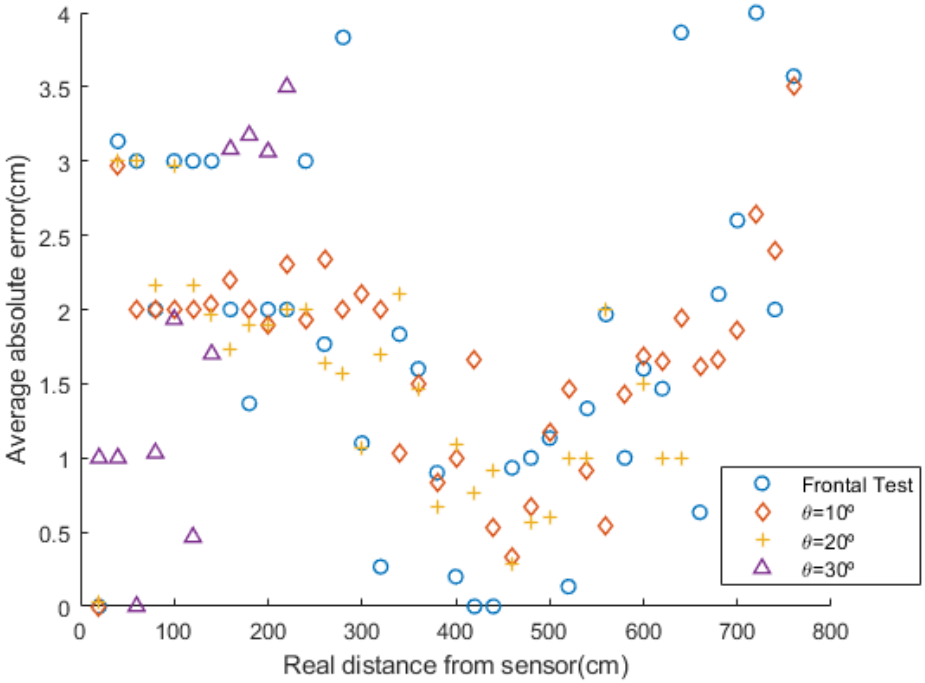


Figure 5.5: MB1242 average absolute error from several distances and orientations

5.1.2 Laser Rangefinder

Just like it was explained in Chapter 4, the chosen laser model was the LW20/C produced by Lightware. This device could have been used as a LiDAR, however this possibility will not be explored due to the fact that MATLAB simulations focused solely on a laser pointing to a fixed direction.

Even though this device was designed to be used with a Serial or I2C interface, only I2C can be used to communicate with the pixhawk 2.1 due to the fact that PX4 firmware only supports this interface. This limitation allied with the fact that LW20/C does not come with the pixhawk I2C compatibility mode active as default, forces a first connection to a computer using a Serial to USB adapter (see Figure 5.6) to activate this mode. Once this task is completed, the laser can be connected to the flight controller's I2C 2 port using an equivalent cable to the one used with the ultrasonic sensor connection.I2C and Serial connections are described in Table 5.1.

Table 5.1: LW20/C connections

Cable colours	Serial Connection	I2C Connection
Red	VCC	VCC
Black	GND	GND
Yellow	TXD	SDA
White	RXD	SCL

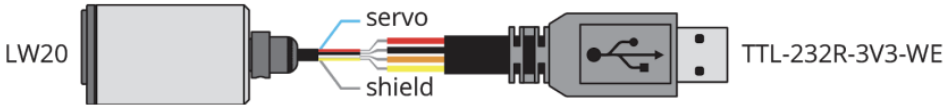


Figure 5.6: Lightware LW20/C connection with USB adaptor. [28]

As it was explained before, this laser must be connected to a computer using a serial to USB adapter to activate the I2C communication compatible with pixhawk. To solve this issue, Lightware offers an application called Lightware Upgrader, which has as a main objective allowing the user to change default settings. The main variable to change is Pixhawk I2C compatibility mode (see Figure 5.7) which must be updated to "on". Despite this, there are much more important settings which can be edited such as Baud Rate or I2C address. Lastly, it is important to refer the existence of an application called Lightware Studio which allows the user to update this device's firmware.

Once these tasks are executed, it is time to activate this sensor within the QGroundControl environment. To do this the SENS_EN_SF1XX parameter must be changed to the only option which includes LW20/C.

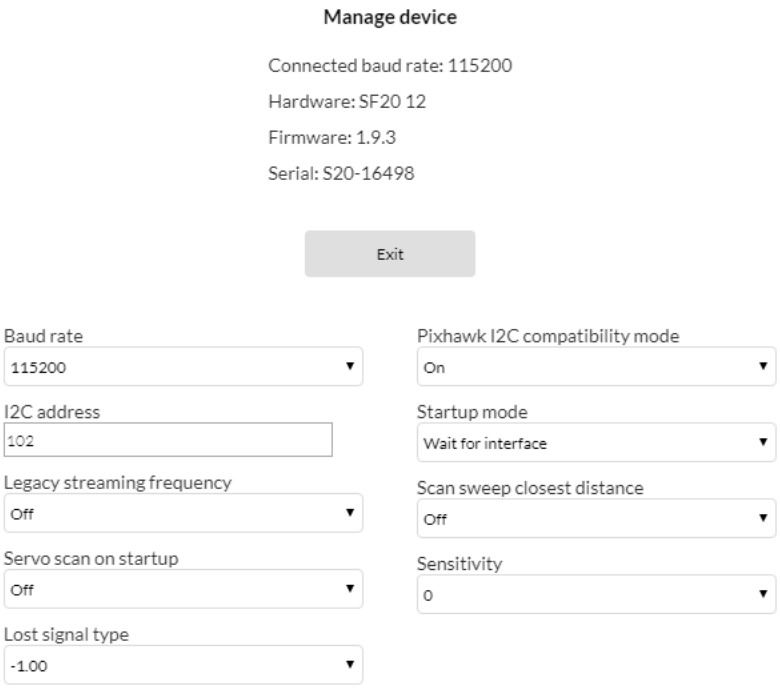


Figure 5.7: Lightware Upgrader environment

After all these technical procedures were finalized, the performance of the laser rangefinder was finally measured. For this purpose, an identical experiment to the one used for sonar was implemented but only with a frontal test as the laser rangefinder is completely directional.

Figure 5.8 shows that the laser kept a perfect detection rate until $85\,\mathrm{m}$. Beyond this distance, it started to fail. The $100\,\mathrm{m}$ mark stated in the data sheet, and represented as a dashed red line was the last distance that the laser managed to detect, even though it was already with a poor detection rate (less than 30%). Although the full range promised in the data sheet was not obtained with a perfect detection rate, this $85\,\mathrm{m}$ mark obtained during this experiment is more than enough to optimize a Sense and Avoidance system that uses a Laser Rangefinder, as proved in Chapter 3.

Regarding accuracy, Figure 5.9 shows that LW20/C managed to keep an average absolute error between $0\,\mathrm{cm}$ and $25\,\mathrm{cm}$ which is quite satisfactory taking into account the considerable distances that this type of sensor can measure.

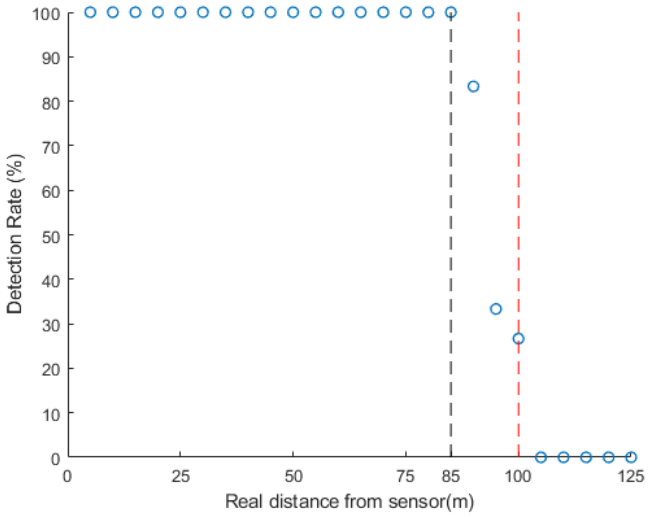


Figure 5.8: LW20/C detection rate from several distances

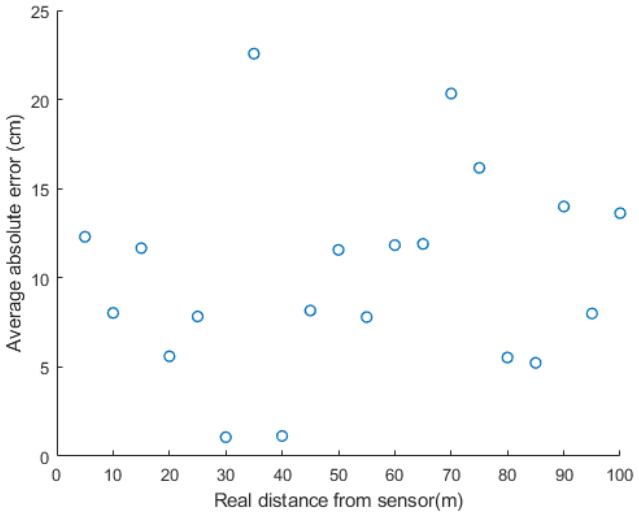


Figure 5.9: LW20/C average absolute error from several distances

5.2 Rover Tests

In order to test everything as a unique system there is the need to complete each one of the following tasks.

1. Integration between rover, controller(pixhawk), and a distance sensor;
2. Software setup;
3. Materialize experiments that can evaluate the system as a whole.

For the first task, the diagram proposed in figure 4.4 was followed with some tiny differences as it is visible in Figure 5.10. The radio receiver and PPM encoder blocks are now separated to better

understand the frontier between the rover and its peripherals. The power module that used to power both the vehicle motors and the flight controller is now only powering the first, leaving the rover's supply to a second battery. This last modification proved to be very useful as it substantially increased the autonomy of the entire system.

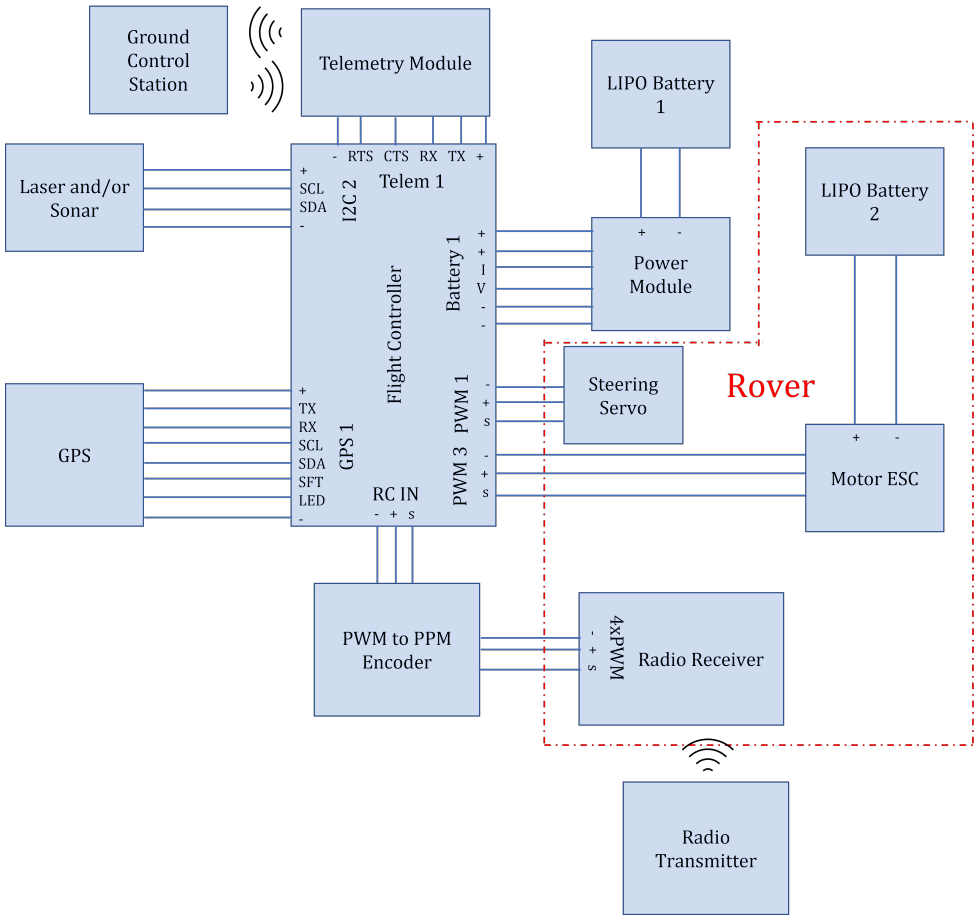


Figure 5.10: Electrical connections for the Rover experiments

The radio transmitter sends all radio signals that are detected by the radio receiver installed in the rover. In order to allow the pixhawk to receive all radio signals, there is the need to use a PPM encoder just like the one shown in Figure 5.11, that receives and translates all 4 radio channels to a single PPM channel that is sent to the pixhawk in the RCIN port. Using 2 of the 4 channels, mentioned before, the user can control the rover manually (1 channel for steering and the other for throttle). The other 2 channels can be used for complementary functions, such as drive mode selection (changing manual to auto mode for example) or to activate object avoidance methods. The pixhawk controls the vehicle sending throttle and steering PWM signals to the Motor ESC and Steering servo respectively. Regarding, the power supply, the steering servo only requires the voltage provided by the pixhawk while the motor is fed by a second second battery installed in the rover.

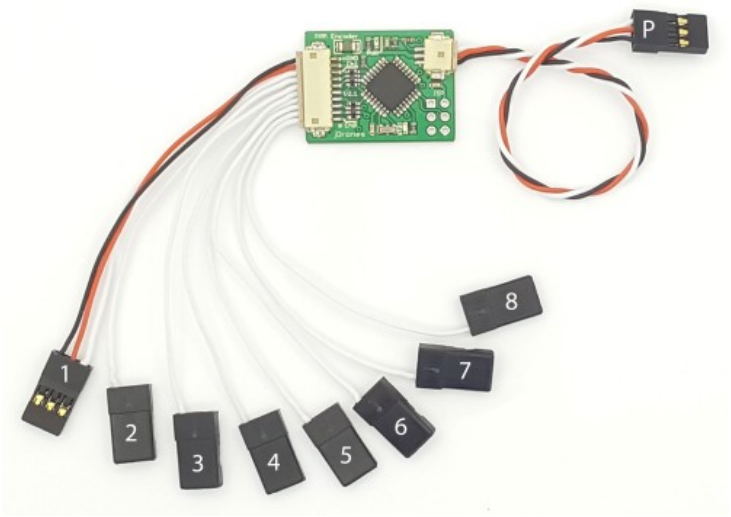
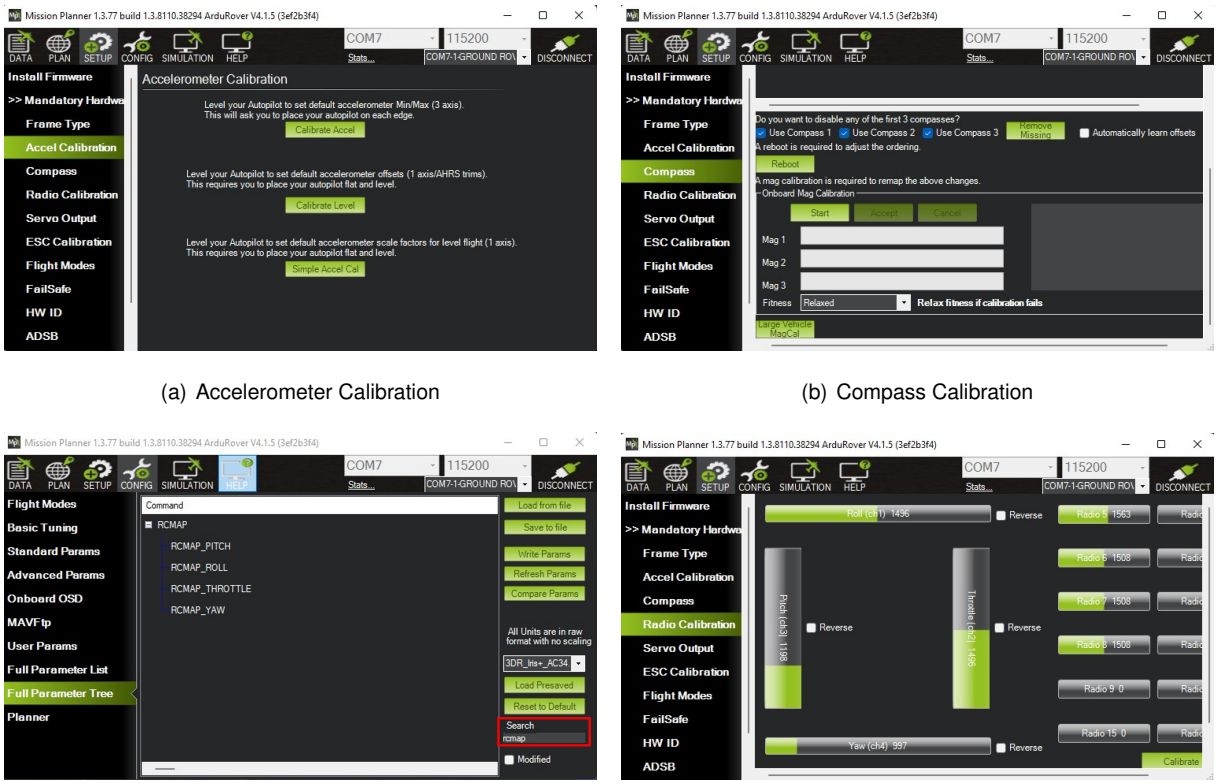


Figure 5.11: PPM encoder[40]

After completing the setup of the car it is necessary to install a suitable software onto the pixhawk in order to later test the rover's reaction to the presence of obstacles. Although until now PX44 was used as the firmware for the controller, it was concluded that a more vehicle-specific firmware such as ArduRover (one of the sub products of Ardupilot) would be easier to use for this particular experiment. With that being said, naturally the control station software was also changed to Mission Planner due to the greater support it provides to Ardupilot products.

The firmware setup was done through the ground control station software. Firstly, the flight controller internal sensors are calibrated as can be seen in Figure 5.12(a) and 5.12(b).

Secondly, each radio channel must be assigned to a specific task. To do that, the parameters of the type "RCMAP_xxx" must be edited with the desired channel number for each one of those functions using the parameter editor as seen in Figure 5.12(c). RCMAP_THROTTLE was assigned to channel 2 and RCMAP_ROLL to channel 1. This last parameter is actually referring to the Steering function, while the other two parameters are not used at all. These two channels are calibrated moving the radio transmitter sticks to their extreme positions within the Mission Planner's radio calibration tool as seen in Figure 5.12(d). Going back to the parameter editor, channel 3 is configured to change the drive mode of the vehicle, setting the parameter MODE_CH to 3, while channel 4 is configured to turn on/off the proximity avoidance, setting the parameter RC4_OPTION to 72.



(a) Accelerometer Calibration

(b) Compass Calibration

(c) Parameters editor

(d) Mapping radio channels

Figure 5.12: Mission Planner environment

Finally, the rover's cruise speed, cruise throttle and max turn rate were defined to allow the firmware to control the steering and throttle of the vehicle in automatic and semi-automatic modes. he avoidance algorithm calibration must be done, defining which sensor to use as rangefinder, its orientation, desired safety margin against the obstacle, the activation of this algorithm and the desired reaction of the rover (simple stop or a proper avoidance). Table 5.2 summarizes some of the most important parameters mentioned before and their values. From left to right, the parameters define the throttle percentage to reach the cruise speed, the actual cruise speed, the rangefinder type used (7 for LW20/C or 2 for MB1242), the object avoidance type to use (0 defines a simple stop) and lastly a switch that turns the object avoidance on and off. All firmware parameters not mentioned before or in Table 5.2 can be found in [41].

Table 5.2: Firmware setup parameters

Calibration Parameters		**Collision Avoidance Parameters**		
CRUISE_THROTTLE	CRUISE_SPEED	RNGFND1_TYPE	OA_TYPE	AVOID_ENABLE
30%	3.67 m/s	7/2	0	1

After analysing the ArduRover repository available on github [42], some main ideas regarding its behaviour were extracted, resulting in the logical scheme presented in Figure 5.13. In Rover.cpp the

code starts by scheduling several functions at specific frequencies and that will be run simultaneously while constantly updating information about the system. These functions may be reading sensors, radio signals, sending/receiving telemetry signals, updating the output for servos etc.

All those functions could be dissected, however, the most important one in the context of this thesis is the one where the object avoidance is incorporated. This function is defined in Rover.cpp and basically calls a function of the type ”Modexxx.update” where xxx can be whatever mode the rover is operating (Manual, Hold, Auto, Acro...). The main idea of this type of functions is to update the PWM signals that control the steering servo and the motor ESC. The difference between them is, fundamentally, the way they achieve this goal. For example, the Modemanual.update() or the Modehold.update() can do this directly. The first one reads the radio channels received by the pixhawk and outputs the corresponding PWM signal to the steering servo and motor ESC while the Hold mode outputs only PWM signals with a 0 % duty cycle, keeping the vehicle still. All the other modes require an intermediate step, determining the desired speed and heading and then calculating the correspondent throttle and steering that achieves that reference. For example, the auto mode would calculate the desired speed and heading using information regarding the rover current position, its next waypoint and the cruise speed it is trying to achieve while in acro mode the idea is to interpret the radio channel allocated for the throttle as the desired speed and the one allocated for the steering as the desired turn rate.

After determining the reference for speed and heading, these drive modes may immediately calculate the steering and throttle required for those references if the avoidance algorithm is not enabled or assess first its surroundings using at least one distance sensor and then adjust its speed and heading accordingly.

For the purpose of this thesis the avoidance will result solely in a cautious stop before breaching a given safety margin so after reading the distance sensor, the algorithm will evaluate if there is any risk of breaching that safety margin and adjust only the target speed resulting in a gradual stop.

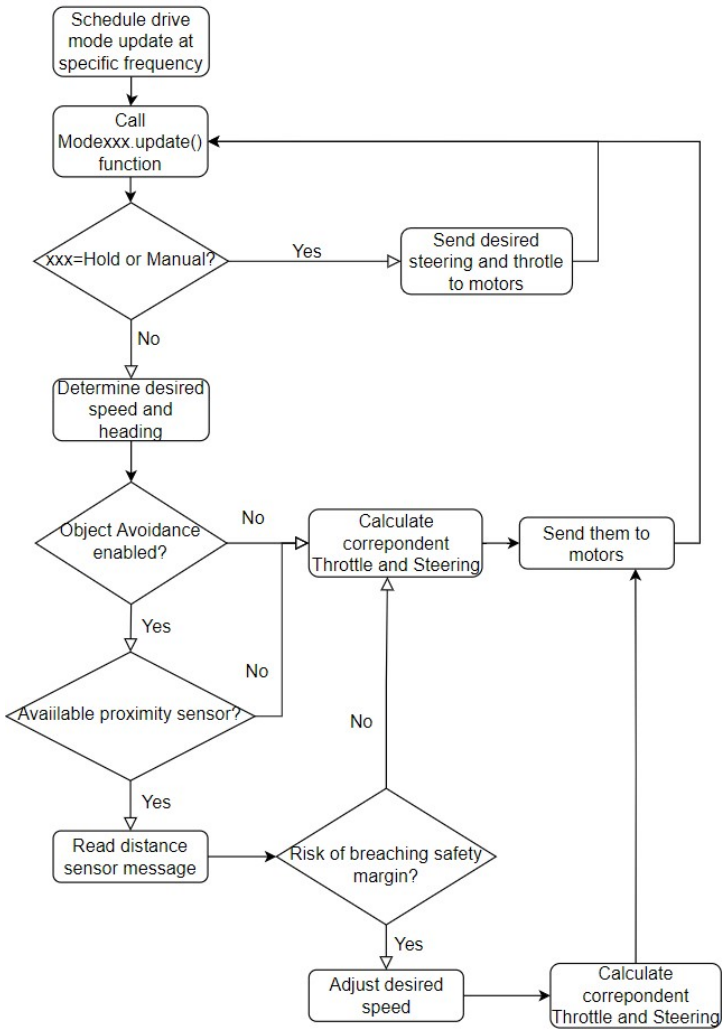


Figure 5.13: Logical scheme of the ArduRover object avoidance algorithm

After presenting the hardware connections and the way the avoidance algorithm works, we can move on to the presentation of the experiments to be performed. The idea is to put the rover travelling along a rectilinear trajectory with a distance sensor pointing forward and place an obstacle in the middle of the trajectory as seen in Figure 5.14.

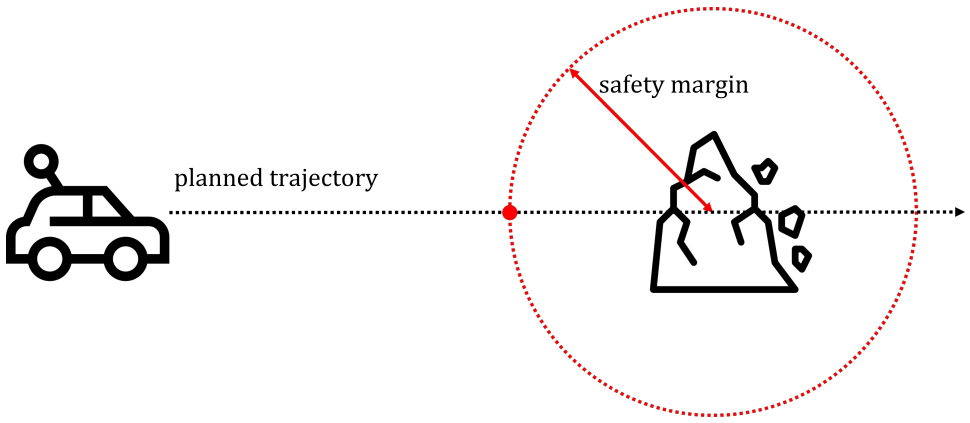


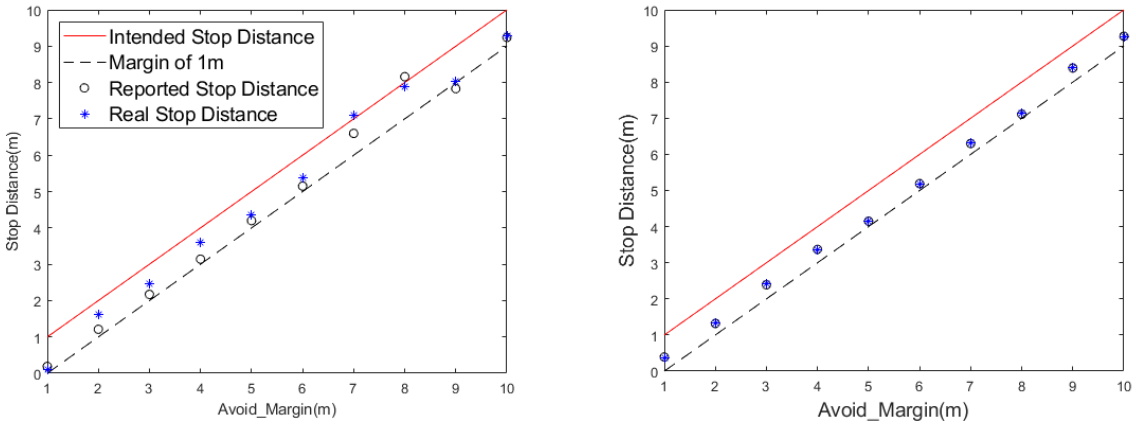
Figure 5.14: Rover tests scheme

Either in Acro or Auto modes, the rover can be placed on that collision path and, if the avoidance algorithm is active, the rover will try to stop at a certain point (red dot in Figure 5.14) of the trajectory depending on the safety margin given as input. In order to test this tool, the rover was equipped with all the necessary hardware present in Figure 5.10 and then taken to the field as seen in Figure 5.15. The parameter that controls the safety margin (Avoid_Margin) started at one meter and was then incremented by one meter until it reached its maximum value of $10\,\mathrm{m}$. For each value of Avoid_Margin, both reported and real stop distances were registered. Figure 5.16 shows precisely this information when the rover was equipped with the laser rangefinder and the ultrasonic sensor.

(a) On bench

(b) In testing

Figure 5.15: Final rover configuration



(a) Laser Rangefinder

(b) Ultrasonic Sensor

Figure 5.16: Simple avoidance results

Through the analysis of Figure 5.16, it is clear the inability of the rover to stop before the safety margin defined by the Avoid_Margin parameter regardless of the sensor used. Nevertheless, the rover has indeed reacted to the presence of the obstacle and was able to effectively stop, which can be seen as a success considering that the rover never exceeded in more than one meter the stipulated

point for its stop, as seen by all the dots in both graphics above the dashed line. Although there are reasons to validate the system used, it is important to understand why it is not performing at its maximum capabilities. The most probable reason for this outcome is a faulty calibration, more specifically, the definition of CUISE_THROTTLE and CRUISE_SPEED present in Table 5.2. These parameters were set using an ArduRover functionality, which autonomously calculates the vehicle's cruise speed for the throttle command that was manually being send to the pixhawk. Without an accurate relation between these two parameters, auto modes will be affected as the controller will not be able to make the rover follow accurately the desired speed variable mentioned in Figure 5.13.

5.3 Summary

Throughout this chapter, all the hardware and software required for future more complex experiments were thoroughly tested. In the case of the laser rangefinder, its actual maximum range was found to be $15\,\mathrm{m}$ bellow the $100\,\mathrm{m}$ stated in its data sheet while the ultrasonic sensor surpassed by $30\,\mathrm{cm}$ its previously reported $630\,\mathrm{cm}$ maximum range and proved to have a very directional beam pattern after additional tests.

As for the final tests with the rover, it was evident that the avoidance algorithm used works, however some limitations were detected forcing the definition of an error margin of $1\,\mathrm{m}$ for this system. The reasons for these failures are not evident but there is a strong possibility that this is due to a poor calibration of the vehicle, making it unable to precisely reach the speed that the algorithm calculates at each iteration.

Chapter 6

Conclusions

6.1 Achievements

The initial idea for this thesis was to deepen a previously initiated study on Sense and Avoidance systems for small fixed wing UAVs with special focus on the detection phase. After a brief introduction and the consequent delimitation of the objectives for this thesis, we proceeded with an analysis about sense and avoidance systems, listing the various types of sensors that can be used, underlining the most appropriate for this specific problem and presenting various types of avoidance algorithms with special focus on the Potential Fields Method that was already implemented in [13]

This was followed by a simulation phase in which the goal was to define a series of parameters that perfectly characterize the configuration of a sensing system. Hereupon, these parameters were tested in a series of simulations that forced the UAV to avoid various obstacles with random sizes, positions and speeds. The two types of sensors tested were laser and sonar, since they would be the ones available for future real-world tests. The avoidance algorithm was previously developed, as it was the laser rangefinder model. The sonar model was built from scratch for these tests. After completing these simulations it was concluded that augmenting both ultrasonic sensors or laser rangefinders range will have positive effects on the systems performance. Nevertheless, this was not the preponderant parameter. The sensor's orientation proved to be the variable with a more well defined minimum. Additionally, it was also concluded that a bad orientation can solely define the systems outcome, making the maximum range influence disappear.

Next, several options were presented for the hardware and software to be used in future real tests. Firstly all the hardware was chosen and gathered in a final electrical scheme. It was concluded that the best approach for this problem would be to use the Lightware LW20/C as laser rangefinder, Maxbotix MB1242 as the ultrasonic sensor and the pixhawk 2 (or cube black) as flight controller. Concerning software, the two best known options for controller firmware and ground control station software were presented and the final decision was to choose px4 and QGroundControl respectively as they offer the most active user development community.

Finally, the sensors were tested individually reporting their real range and measurement errors. In

the specific case of the ultrasonic sensor more exhaustive experiments were done in order to obtain an experimental beam pattern. It was concluded that the laser rangefinder maximum range is slightly less than what was stated in the data sheet (-15%) and ultrasonic sensor was actually above the expectations by almost 5%. The ultrasonic sensor experimental beam pattern proved to be extremely directional confirming the data sheet information. Regarding, the sensor's measurement errors, it was concluded that this would not be a problem in the future as neither one of them passed 0.7% of their maximum range. Knowing the specific limitations of the available sensors, the next step was to integrate them in a small rover, testing the interaction of these sensors with the controller and the rover's ability to stop autonomously in case of an imminent collision. It was concluded that the system was able to react to the presence of obstacles, however, its reaction was not ideal as it never managed to stop before the established safety margin. This error is believed to be directly linked to a bad calibration.

6.2 Future Work

In order to continue this work, there are several steps ahead. Still in the simulation phase, more sensors can and should be simulated, also considering more complex configurations that incorporate more sensors mixing several types if necessary. As for the experimental part, it is necessary to categorize more distance sensors as wells as testing the rover avoidance algorithm at its maximum capabilities, by trying to make the rover completely avoid an obstacle and follow the route previously planned.

Finally, all hardware and software used for the rover tests must be adapted in order to finally implement a sense and avoidance system in a small fixed wing UAV as planned since the beginning.

Bibliography

[1] J. F. Keane and S. S. Carr. A brief history of early unmanned aircraft. *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)*, 32(3):558–571, 2013. ISSN 02705214.

[2] C. Barroso. Firefront: an inteligent system that will help detect and fight forest fires. Technical report, Lusa, July 2020. URL:https://tecnico.ulisboa.pt/en/news/firefront-an-intelligent-system-that-will-help-detect-and-fight-forest-fires/.

[3] S. M. Hader. Cargo drones: The future of parcel delivery. Technical report, Roland Berger, February 2020. URL:https://www.rolandberger.com/en/Insights/Publications/Cargo-drones-The-future-of-parcel-delivery.html.

[4] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis. A review on UAV-based applications for precision agriculture. *Information (Switzerland)*, 10, 2019. ISSN 20782489. doi: 10.3390/info10110349.

[5] P. Finnegan. 2020/2021 world civil unmanned aerial systems. Technical report, Teal Group, 2020.

[6] K. Dalamagkidis. Classification of uavs. pages 83–91, 2015. in Handbook of Unmanned Aerial Vehicles, doi:10.1007/978-90-481-9707-1_94.

[7] N. Alturas. Modeling and optimization of an obstacle detection system for small uav's. Master's thesis, IST, January 2021.

[8] Tekever ar4. URL:http://uas.tekever.com/ar4-evo/.

[9] K. Squires. Nasa-led airborne mission studies storm intensification in northern hemisphere, August 2017. URL:https://www.nasa.gov/centers/armstrong/features/airborne-mission-studies-northern-hemisphere.html.

[10] Nasa's global hawk in the eye of hurricane earl on september 2, September 2010. URL:https://www.nasa.gov/mission_pages/hurricanes/missions/grip/multimedia/GH_EyeOfEarl.html.

[11] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access*, 2019. doi:10.1109/ACCESS.2019.2909530.

[12] Legal issues arising from the adoption of unmanned aircraft systems, August 2015. URL:https://www.financierworldwide.com/legal-issues-arising-from-the-adoption-of-unmanned-aircraft-systems.YHcbVyWSlhE.

[13] J. Alves. Path planning and collision avoidance algorithms for small rpas. Master's thesis, IST, June 2017.

[14] X.Yu and Y.Zhang. Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects. *Progress in Aerospace Sciences*, 2015. https://doi.org/10.1016/j.paerosci.2015.01.001.

[15] K. G. M. K. M. Skowron, W. Chmielowiec and A.Srebro. Sense and avoid for small unmanned aircraft systems: Research on methods and best practises. *Journal of Aerospace Engineering*. doi:10.1177/0954410019867802.

[16] M. Skolnik. *Introduction to Radar systems*. 2 edition, 1981.

[17] J. Zeng. Ainstein enabled new drone water sampling application with its radar technology, November 2020. URL:https://www.dronecode.org/ainstein-enabled-new-drone-water-sampling-application-with-its-radar-technology/.

[18] 3d data. URL:https://www1.nyc.gov/site/doitt/residents/gis-3d-data.page.

[19] K. Gillespie. Ultrasonic sensors: Advantages and limitations, September 2019. URL:https://www.maxbotix.com/articles/advantages-limitations-ultrasonic-sensors.htm/.

[20] Ultrasonic module hc-sr04. URL:https://www.electronicwings.com/sensors-modules/ultrasonic-module-hc-sr04.

[21] C. Luo, S. I. McClean, G. Parr, L. Teacy, and R. De Nardi. UAV position estimation and collision avoidance using the extended kalman filter. *IEEE Transactions on Vehicular Technology*, 62(6): 2749–2762, 2013. ISSN 00189545. doi: 10.1109/TVT.2013.2243480.

[22] A. A. . A. M. . P. O. . E. Badreddin. A comparative study of collision avoidance techniques for unmanned aerial vehicles. *IEEE International Conference on Systems*, 2013. doi: 10.1109/SMC.2013.338.

[23] H.Safadi. Local path planning using virtual potential field. https://www.cs.mcgill.ca/ hsafad/robotics/, 2007. Accessed: 25-03-2021.

[24] B. Etkin and L. D. Reid. *Dynamics of Flight*, volume 2. 3 edition, 1959.

[25] Maxbotix. URL:https://www.maxbotix.com/.

[26] I2cxl-maxsonar-ez series data sheet, 2012. URL:https://www.maxbotix.com/documents/I2CXL-MaxSonar-EZ_Datasheet.pdf.

[27] Lightware. URL:https://lightware.com/.

[28] Lw20 lidar sensor datasheet, 2019.

[29] DroneCode. URL:https://www.dronecode.org/.

[30] Hexadrone. Pixhawk v2 feature overview. URL:https://hexadrone.fr/img/cms/DRS_Pixhawk-2.pdf.

[31] Mavlink developer guide. URL:https://mavlink.io/en/.

[32] Ardupilot, . URL:https://ardupilot.org/.

[33] About ardupilot, . URL:https://ardupilot.org/index.php/about.

[34] Px4, . URL:https://px4.io.

[35] PX4. Software overview, . URL:https://px4.io/software/software-overview/.

[36] Qgroundcontrol, . URL:http://qgroundcontrol.com/.

[37] Mission planner. URL:https://ardupilot.org/planner/.

[38] Qgroundcontrol user guide, . URL:https://docs.qgroundcontrol.com/master/en/index.html.

[39] R. Burnett. Frequently asked questions about ultrasonic sensors. URL:https://www.maxbotix.com/frequently-asked-questions.

[40] Ardupilot. Peripheral hardware-ppm encoder. URL:https://ardupilot.org/copter/docs/common-ppm-encoder.html.

[41] Ardurover-first time setup and configuration, . URL:https://ardupilot.org/rover/docs/apmrover-setup.html.

[42] Ardupilot code repository. URL:https://github.com/ArduPilot/ardupilot.