

uc3m | Universidad **Carlos III** de Madrid

BACHELOR'S THESIS

DEGREE IN AEROSPACE ENGINEERING

**Obstacle alert and collision avoidance
system development for UAVs
with Pixhawk flight controller**

Cristóbal Cuevas García

JUNE 2018

Supervisors

Xin Chen

David Morante González



This work is licensed under Creative Commons **Attribution – Non
Commercial – Non Derivatives**

Abstract

In recent years, the unmanned aerial vehicles sector has been characterized by its sharp growth, spreading its line of applications and becoming one of the cutting edge technologies in the world. However, this exponential advancement would have been even more extreme but for the restrictive existing legislation that limits its operations.

That constraints imposed to drone operations are not legislated in vain. Specially in populated areas, flying drones is a dangerous service that entails risks for the safety of the population. Useless have been the attempts of many major online selling companies to make use of unmanned aerial vehicles serving as dealers of parcels. Further technology needs still to be implemented, for guarantying standards levels of safety in urban zones.

This thesis aims to contribute to this required development of drone technology by proposing a preliminary collision avoidance system for unmanned aerial vehicles. The project involves assembling a flight-capable quadcopter from scratch and implementing the collision avoidance as an additional subsystem. To that end, a set of ultrasonic range finders are located around the quadcopter. Their acquired raw data is processed in an auxiliary arduino microcontroller board that send trajectory corrections to the flight controller of the quadcopter based on the distance information.

As a result, a concept proof of an autonomous collision avoidance system is integrated into an unmanned aerial vehicle system. Results are obtained and consecutively analyzed based on ground and flight tests. For that purpose, the flight capabilities of the built quadcopter are proved, and afterwards, the collision avoidance is tested. Overall, the collision avoidance system effectiveness was achieved. The collision avoidance work principle consisted on warding off from obstacles when detected. Major faced problems are related to stability recover after the collision is avoided. That problem was solved by proving different flight modes, but that issue needs future work to make the collision avoidance more reliable.

Keywords: collision avoidance, ultrasonic range finder, Arduino board, Pixhawk flight controller, obstacle detection, environment monitoring, overwritten pitch/roll angles

Acknowledgements

Firstly, I would like to dedicate this thesis to my family, for their endless support and dedication, always providing me with the best education and resources at their disposal.

Also, thanks to Xin Chen for your great support and measureless patience through the development of this thesis. A big thank you also to David Morante, for providing me with the best technical support and advice always I needed it.

I would also like to dedicate this thesis to my cousin Damián, for providing with a place to test my quadcopter.

I want to thank all my classmates and friends who encouraged me to keep working on this thesis.

Finally, but not less important, I want to thank Mercedes, for being the best travel partner one could ever imagine.

Contents

Abstract	iii
Acknowledgements	v
List of Figures	x
List of Tables	xii
List of abbreviations 1	xiv
List of abbreviations 2	xv
1 Introduction	1
1.1 First approach to Collision Avoidance and in UAS	1
1.2 Socioeconomic Impact	4
1.3 Legal Framework	4
1.4 Project Objectives	6
1.5 Budget	7
1.5.1 Software Cost	7
1.5.2 Hardware Cost	7
1.5.3 Personnel expenses	7
1.6 Outline of the document	8
2 State of The Art	9
2.1 Unmanned aerial vehicles	9
2.1.1 MAV classification	9
2.2 Distance Sensing	11
2.2.1 Light Detection and Ranging (LiDAR)	11
2.2.2 Sound Navigation and Ranging (SoNAR)	12
2.2.3 Radio Detection and Ranging (RaDAR)	12
2.3 Collision Avoidance in UAVs	13
3 UAV components	15
3.1 Hardware Components	15
3.1.1 Frame	15
3.1.2 Rotors	17
3.1.3 Propellers	18
3.1.4 Electronic Speed Controllers (ESC)	19
3.1.5 Power	19
3.1.6 Flight Controller	20
3.1.7 Arduino Mega 2560	22
3.1.8 Telemetry	23
3.1.9 Radio transmitter and receiver	23
3.1.10 Ultrasonic range finder	24

3.2	Software elements	24
3.2.1	Ubuntu 16.04	24
3.2.2	APM Planner	25
3.2.3	Arduino IDE	27
3.3	Communications	27
3.3.1	RC control	27
3.3.2	MAVLink	28
4	Collision Avoidance System Design	31
4.1	Collision Avoidance System Requirements	31
4.2	Collision Avoidance System Stages	34
4.2.1	Minimum altitude activation level	34
4.2.2	Horizontal environment sensing	35
4.2.3	Obstacle detection	35
4.2.4	Collision avoidance Maneuver	36
4.3	Collision Avoidance Algorithm flow diagram	39
5	Collision Avoidance System Implementation	40
5.1	Quadcopter assembly, wiring and calibration-making the drone flight capable	40
5.2	CAS Hardware Integration	45
5.2.1	Arduino Mega 2560-Pixhawk flight controller	46
5.2.2	Arduino Mega 2560-Ultrasonic range finder HC-SR04	47
5.2.3	F450 kit-Ultrasonic range finder HC-SR04	48
5.2.4	F450 kit-Arduino Mega 2560	49
5.3	CAS software integration	51
5.3.1	Distance data acquisition and processing	51
5.3.2	Arduino board as an UAV system	52
5.3.3	Obstacle alert and collision avoidance	54
6	Testing and Results	56
6.1	Flight capabilities tests and results	56
6.1.1	Stabilize flight mode	56
6.1.2	Altitude hold mode	57
6.2	Component testing and results	58
6.2.1	HC-SR04 Ultrasonic range finders	58
6.2.2	Arduino Mega 2560	60
6.3	MAVLink communication testing and results	60
6.4	Obstacle collision avoidance ground tests	60
6.4.1	Arming and disarming test	60
6.4.2	Collision avoidance system ground test	61
6.5	Obstacle collision avoidance flight tests	66
6.5.1	Stabilize flight mode test	66
6.5.2	Altitude hold flight mode test	67

7	Summary, Conclusions and Future work	69
7.1	Project summary	69
7.2	Extracted conclusions	70
7.3	Future work	71
	References	73

List of Figures

1	Layered approach to avoid collision between manned aircraft	1
2	General collision avoidance procedure	3
3	Classification of MAVs: (a) fixed wing, (b) flapping wing, (c) fixed/flapping-wing, (d) rotary wing , (e) VTOL, (f) ducted fan, (g) tilt-rotor, (h) helicopter, (i) unconventional, (j) ornicopter	10
4	Two UAV applied LiDAR sensors currently in the market	11
5	MB1010 LV-MaxSonar-EZ1[14]	12
6	Arducopter collision avoidance environment sector divisions [17]	14
7	High performance LiDAR sensors	14
8	Different rotor configurations	16
9	F450 kit	16
10	Rotor classification [21]	17
11	2212 900 Kv Brushless rotor	18
12	High Pitch and Low Pitch values [22]	18
13	Selected Propellers [23]	19
14	Duty cycles [24]	19
15	Selected Battery [25]	20
16	Pixhawk 2.4.8 [27]	21
17	Arduino Mega 2560 [30]	23
18	Sik Telemetry Radio [31]	23
19	Flysky FS-i6 RC transmitter and receiver [32]	24
20	HC-SR04 Ultrasonic Finder [33]	24
21	Arduino IDE interface[34]	25
22	APM Planner interface	26
23	RC transmitter Euler angles and throttle sticks [37]	28
24	First stage	34
25	Second stage	35
26	Third stage	36
27	Collision avoidance roll maneuvers phases 1 and 2	37
28	Collision avoidance roll maneuver phase 3	37
29	Collision avoidance pitch maneuvers phases 1 and 2	38
30	Collision avoidance pitch maneuver phase 3	38
31	Collision avoidance algorithm flow chart	39
32	ESCs connections	40
33	XT60 connector	41
34	F450 assembled frame	41
35	Spinning criteria and rotor assembly	42
36	F450 legs	42
37	Wiring order	43
38	Cushioning sponge	43
39	Radio receiver and PPM encoder	44
40	Pixhawk flight controller pins	44

41	GCS calibration display	45
42	Arduino Mega 2560-Pixhawk flight controller pin connection	46
43	Arduino Mega 2560-HC-SR04 range finder pin connection	48
44	Lateral and bottom ultrasonic range finders	49
45	Arduino Mega 2560 assembled to the bottom of the power distribution board	50
46	Collision avoidance hardware layout	50
47	Final prototype	51
48	UAS linked to the GCS	53
49	Collision avoidance system communication lost alert	53
50	Ultrasonic range finders locations	54
51	Video footage frame of stabilize mode test	56
52	PID recommended parameters [41]	57
53	Prompted distances (cm) from the 5 ultrasonic range finders	59
54	Arming and disarming test	61
55	Prompted PWM commands	62
56	Drone response to pitch PWM outputs based on distances to front sensor 1	63
57	Drone response to pitch PWM outputs based on distances to back sensor 3	64
58	Drone response to roll PWM outputs based on distances to right sensor 2	65
59	Drone response to roll PWM outputs based on distances to left sensor 4	66
60	Collision avoidance flight test video footage frame	68

List of Tables

1	Estimated Budget	7
2	MAVLink bytes explanation	29
3	Collision avoidance system requirements	31
4	Arduino Mega 2560-Pixhawk flight controller wiring connections . . .	46
5	Arduino Mega 2560-HC SR04 pin connections	48
6	PWM overridden values	55

List of abbreviations 1

Acronym	Stands for
UAS	Unmanned aerial systems
UAV	Unmanned aerial vehicles
TOL	Target levels of safety
PSR	Primary Surveillance Radars
SSR	Secondary Surveillance Radars
TCAS	Traffic Alert and Collision Avoidance System
ATCO	Air Traffic Controller Operator
VMC	Visual meteorological conditions
IMC	Instrumental meteorological conditions
AESA	Agencia Estatal de Seguridad Aérea
AGL	Above ground level
MTOM	Maximum take-off mass
BVLOS	Beyond Visual Line of Sight
EVLOS	Extended Visual Line of Sight
MUAV	Mini Unmanned Aerial Vehicles
MAV	Micro UAV
LiDAR	Light Detection and Ranging
SoNAR	Sound Navigation and Ranging
RaDAR	Radio Detection and Ranging
BLDC	Brushless Direct Current
ESC	Electronic Speed Controllers
LiPo	Lithium Polimer
IMU	Inertia Measurement Unit

List of abbreviations 2

Acronym	Stands for
GCS	Ground Control Station
RC	Radio Control
PPM	Pulse Position Modulation
MAVLink	Micro Air Vehicle Link
CAS	Collision Avoidance System
ESC	Electronic speed controllers

1 Introduction

The purpose of this section lies on approaching the reader to collision avoidance adaption to UAS and its socioeconomic impact. Then, a legal framework about UAVs in Spain is given; and finally, and more directly related with this project, the project objectives, the budget and the outline of the document are presented.

1.1 First approach to Collision Avoidance and in UAS

Collision avoidance development in UAS is coming out as an essential solution for the introduction of UAS traffic in the civil airspace. Technologies in this field are being developed worldwide. Prerequisites for its implementation are highly intricate and vary depending on the target airspace. The collision hazards when mixing with manned air vehicles are way too diverse, and makes this development a laborious task.

The airspace has an international set of standards and regulations to guarantee and ensure that the risk of collision for manned aircraft is consistent with an acceptable Target Level of Safety (TLOS).[1].

For a collision to occur, failures must happen in multiple layers, as depicted in Figure 1.

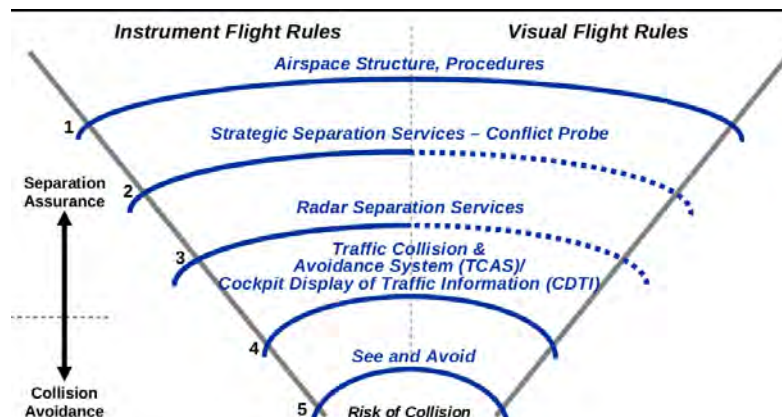


Figure 1: Layered approach to avoid collision between manned aircraft [1]

Layer 1 is not specifically focused on avoiding collision, while layers two to five follow the same procedure to prevent collision between two or several aircraft/airborne elements. That procedure is shown in Figure 2 where the following elements are delineated [1]:

- **Surveillance:** surveillance consist of determining the position (lateral or vertical) of any vehicle or object within any sector of the airspace [2]. Among the

developed systems to perform this task, those ones can be found:

1. Visual observation: mostly in the vicinity of airports.
 2. Noise detection: at the very commencement of aviation, this method was used to prevent airspace incursions by enemies. Today is still used for noise detection, particularly in populated areas.
 3. Radio reports: pilot based-reports through radio let air traffic controllers take notes about the current position of an aircraft within a given sector of a given airspace.
 4. Primary Surveillance Radars (PSR): those were the first "Radio Detecting and Ranging" and its principle of functioning consisted of high frequency radio transmitter broadcasting high-energy short-length radio waves through a rotating antenna, with a tiny portion of the sent wave coming back after being reflected by the detected object.
 5. Secondary Surveillance Radars (SSR): those radars were developed some years later than PSR and need active collaboration from the target, equipped with a transponder.
- **Risk identification:** this task is usually performed by air traffic controllers. Position and future position awareness based from flight information such as relative position between two aircraft and relative speed let ATC operators be aware of the potential risk of mid-air impact if the situation deteriorates.
 - **Determination of appropriate avoidance maneuver:** this is the next step right after identification of risk and can be developed by air traffic controllers by radar separation clearance; or by an on-board automation system called Traffic Alert and Collision Avoidance System (TCAS).
 - **Maneuver:** performed by the pilot based on the instructions received from ATCOs or the display of the TCAS.
 - **Return to course:** once the danger disappears, the aircraft carries on with its designated route.



Figure 2: General collision avoidance procedure
[1]

As the reader could observe, collision avoidance within a sector of the airspace is a complex system, where assistance of many players is needed, specially in the vicinity of airports where air traffic gets too busy. This makes the introduction of UAS into the air traffic a much more complicated system, where fulfillment of TLOS is the main challenge. Up to know, pilots were the last and crucial executors of the collision avoidance procedure, as explained above. Its absence would mean developing different procedures to remain under the required levels of safety. Among the several challenges that this proposes, here are some enumerated [1]:

- **Catastrophic consequences of mid-air collision:** the out-coming disasters produced by a mid-air collision lead to have very high target levels of safety and the acceptable probability of risk currently is 1×10^{-9} , which means one flight collision for every billion of flight hours. Introducing UAS into airspace maintaining this low risk level is difficult.
- **Difficulties on certification and verification:** collision avoidance system performance is already difficult to verify and certificate even whit manned aircraft.
- **Complexity of requirements:** UAS collision avoidance would need to be operative in Visual Meteorological Conditions (VMC), in Instrumental Meteorological conditions (IMC), in any altitude or flight level; as well as detecting any airborne object, not necessarily other aircraft, but balloons, walls trees and any urban element in the case of urban applications.
- **Lack of industry:** there are not many manufacturers working on this technology. The reason maybe lies in that manufacturers know its difficulties and potential limitations.
- **Community agreement:** ICAO, that stands for International Civil Aviation Organization, would be the responsible to set the certification procedures applied to this technology.

1.2 Socioeconomic Impact

During the last 10 years, technology development has allowed clients to get front line products at reasonable prices. Originally, drones were a technology that was only under the scope of military institutions due to its elevated cost and its leading technology. Nevertheless, economies of scale has lead to the possibility of purchasing those flight devices for prices ranging between \$50 and \$100 [3].

Companies such as Amazon or Google have already in mind further projects related to drones. They are just waiting for a suitable development of the legal framework. Amazon has promised to its clients a future system of parcel (under 51 lb) delivery based on the use of UAS [4]. Google, in contrast, is developing an aerial system of drones with the scope of environmental conservation and medicine deliver to unreachable and remote zones. Drones are considered as environmentally-friendly devices, since they do not require fuel to operate, and the majority of them are electrically powered.

The commercial benefits from drones is irrefutable. A recent investigation [3] has estimated that from 2015 to 2025, UAV implementation into national USA airspace would lead to \$82.1 billion in terms of economic growth and job creation, with the totality of 100,000 new jobs.

The main UAV markets are oriented to infrastructures, agriculture, transport, security, media, insurance, telecommunications and mining. Agriculture is one of the UAV sectors with more future, due to the UAV abilities to cover large areas, serving by feeding and hydrating plants while monitoring spread of disease and their health.

Jobs created within the mentioned range of 10 years would be primarily in the manufacturing sector. States would undoubtedly get benefits from tax windfalls from the increased economic activity. Commercial drones would also allow industries save expenses in terms of inventory, transportation and distribution.

1.3 Legal Framework

Up to this section, this thesis has refereed to the legality of drones based on international laws and international civil aviation institutions. However, national civil aviation institutions have not yet been able to harmonize a common legal framework valid for all countries. Therefore, since this project has been developed in an University located in Spain; the national legal situation concerning UAVs will be presented in this unit. All the information here presented has been retrieved from *Real Decreto 1036/2017* [5]. This is the current law that applies for the civil use of remotely piloted aircraft in Spain. In other words, it is the regulation for the use of drones within the boundaries of the country.

- **License requisites for piloting drones**

License is only required for individuals who are going to perform professional tasks by direct manipulation of drones. In those cases, an issued title by a certificated academy needs to be provided, in which the person demonstrates a practical and theoretical knowledge about piloting of drones as well as a medical certificate.

- **Requirements to be fulfilled to operate professionally with drones**

1. Registration in AESA as a UAV operator.
2. Holding a civil responsibility insurance.
3. Holding UAV pilot title.
4. Holding medical certificate into effect.

- **Requirements to be fulfilled to operate recreationally with drones**

1. Flying within a distance under 8 kilometers to any aerodrome or airport.
2. Flying outside of a controlled airspace.
3. Flying under 120 meters of height (AGL)
4. Flying daily and under proper meteorological conditions. In case of piloting an UAV under 2 Kg, night flights are allowed but never over 50 meters (AGL).
5. Flying always within the visual observation range of the pilot.
6. UAVs under 250 g can overfly cities and crowds but never over 20 meters.
7. Even though it is not mandatory, holding a civil responsibility insurance is strongly recommended.

- **Any UAV will need to hold a data plate with the following information**

1. Manufacturer
2. Type
3. Model
4. Serial number
5. Name and contact details of the operator

- **Requirements to operate UAVs professionally in urban zones and over crowds**

1. MTOM under 10 Kg
2. Operation performed within the visual range of the pilot.

3. The flight zone must be ribbed by the competent authorities. If that is not the case, a minimum horizontal distance of 50 meters must be preserved from buildings and people.
 4. The UAV must have a system of impact energy absorption (parachute, airbag...).
 5. AESA must authorize the operation.
- **Requirements to operate night UAV flights**
 1. AESA must authorize the flight based on a specific safety study.
 - **Requirements to operate UAV in controlled airspace**
 1. The UAV must be equipped with a transponder in mode S.
 2. The pilot must hold a radiophonist license.
 3. The pilot must accredit to know the language of communication between the ATCO and the UAV.
 4. AESA must authorize the flight.
 - **Requirements to operate UAV with MTOW over 2 Kg over the visual range of the pilot (BVLOS)**
 1. The UAV must be equipped with systems able to detect and avoid other users of the airspace.
 2. The UAV must be equipped with a vision device oriented facing forward.
 - **Requirements to operate UAV within EVLOS**
 1. The flights will be permitted under the presence of intermediate observers.
 2. The intermediate observers will remain in constant radio communication with the pilot and will accredit the theoretical knowledge of a remote pilot.

1.4 Project Objectives

As stated in the Abstract, this project aims to assemble and testing a flight-capable quadcopter from scratch and implementing a collision avoidance subsystem into it. Inside this scope, the following objectives can be stated:

- Building from scratch a flight capable drone to be used as a prototype in which implementing afterwards a Collision Avoidance System.
- Proving the flight capabilities of the prototype based on flight testing.
- Designing an Obstacle Collision System.

- Implementing the Obstacle Collision System into the prototype.
- Proving the proper communication between the Obstacle Collision Avoidance system and the rest of the systems of the drone.
- Testing the proof of concept of the Obstacle Collision Avoidance System once implemented into the drone.

1.5 Budget

This sections states the costs associated within the project and estimates the associated budget.

1.5.1 Software Cost

All the software related tasks has been carried out making use of open-source software systems, such as Ardupilot, Arduino IDE, APM Planner and Ubuntu 16.04.

Therefore, the resulting costs in software stands for 0 €.

1.5.2 Hardware Cost

Table 1 shows the estimated costs in hardware expenses.

Table 1: Estimated Budget

Component	Unitary Price (€)	Units	Total (€)
F450 kit	227.76	1	227.76
Propeller Blades	3	6	18
Radio transmitter/receiver	51	1	51
Telemetry radio	48	1	48
Battery	41	2	82
Ulstrasonic Range Finder HC SR04	2.5	10	25
Wires	0.05	30	1.5
PPM encoder	8.90	1	8.90
Pixhawk Flight Controller	95	1	95
Arduino Mega 2560	34.65	1	34.65
TOTAL			591.81

1.5.3 Personnel expenses

According to [6], the mean salary of a recent graduate junior engineer in Spain stands for 24501 €, for full time eight-hour workday. The computed medium salary

per hour is computed as 13.61 €.

The estimated time dedicated to this project is over 650 hours. Therefore, it would correspond to a personnel expenses of 8846.50 € if a recent graduate engineer simulated this project.

1.6 Outline of the document

This document is structured in several sections.

- Section 2 introduces to the reader the current state of art about collision avoidance and environment sensing applied to UAVs.
- In section 3, all the quadcopter hardware components and software elements are presented, together with the communications among the UAV subsystems.
- In section 4, the collision avoidance system design is covered, defining the different collision avoidance stages, the sense algorithm and the system requirements.
- Section 5 covers the quadcopter building process, together with the obstacle collision avoidance system hardware and software implementation into it.
- In section 6, the ground and flight tests are detailed, analyzing and exposing the obtained results.
- Finally, in section 7 the outcoming conclusions and future work purposes are presented.

2 State of The Art

Unmanned aerial Vehicles (UAVs) has a huge range of applications. Companies of this sector have concentrated their efforts on developing UAVs capable of successfully perform their demanded tasks; as well as a software to establish communication among the hardware components and to provide the users with a reliable interface to communicate and operate the UAVs.

2.1 Unmanned aerial vehicles

Unmanned Aerial Vehicles (UAVs) stand for aircraft without an human pilot aboard. UAVs are a component of an UAS, that include the UAV, a ground-based controller and the system of communication between those two [7]. In this project, a Micro Air Vehicle (MAV) is utilized, a F450 quadcopter under 2 Kg. MAV stands for drones under 2 Kg and over 50 g of weight and ranging between 15 cm and 100 cm of size [8].

2.1.1 MAV classification

There are different kinds of MAV. According to a classification made in a research from 2017 [8], the following ones are highlighted.

- **Fixed wing MAVs:** those kind of MAVs are commonly formed by a rigid wing, a fuselage and tails, where motor with a propeller is placed. These MAVs are adaptable to many kind of environments, such as jungle, urban zones, mountains and even arctic environs. Among its applications, data acquisition and patrolling are the most relevant ones. Those kind of MAV are the ones with the greatest range and endurance, compared to the rest of the existing MAV.
- **Flapping wing MAVs:** the main feature of these MAVs are their flexible and flapper wings. They are inspired in animals ranging from very tiny insects to small birds. A key fundamental advantage among this kind of MAV lies on their brilliant capacity of hovering. That feature makes them ideal for search and rescue missions. These MAV are produced in different types of wing configurations: tandem, monoplane and biplane.
- **Rotatory wing MAVs):** those MAVs are characterized by its excellent hover capability and their maneuverability. A rotatory MAV is a MAV with rotatory blades and propeller-based systems. Their capacity of moving towards any desired direction, makes them unique for approachless environments. According to the number of blades of the propellers, they can be classified as twincopters, tricopters, quadcopters, pentacopters, hexacopters, octocopters, decacopters and dodecacopters.



Figure 3: Classification of MAVs: (a) fixed wing, (b) flapping wing, (c) fixed/flapping-wing, (d) rotary wing , (e) VTOL, (f) ducted fan, (g) tilt-rotor, (h) helicopter, (i) unconventional, (j) ornicopter [8]

2.2 Distance Sensing

Distance sensing is a leading technology that has application in many sectors of the engineering. Applied to UAVs, current state of art shows that their main utilization applies for environment mapping, terrain shape following, altitude measuring and collision avoidance [9].

2.2.1 Light Detection and Ranging (LiDAR)

Light Detection and Ranging (LiDAR) sensor is a technology of precise measurement based on the time elapsed between transmitting a pulsed optical laser signal and receiving its reflection. Thus, there are two signals types in this kind of measurement method: reference signal from the transmitter and reflected received signal from target. This process is called correlation, where the distance to the target is calculated based on the time delay and using the speed of light constant [10] .

LiDARs work at different electromagnetic wavelengths depending of its line of applications. For instance, meteorology LiDARs work always on infrared (1500 nm to 2000 nm) and on ultraviolet (250 nm) wavelengths.

The work principle of LiDARs is based on this equation:

$$L = c \frac{t}{2} \quad (1)$$

Where L is the distance, t is the time elapsed between sending and receiving the pulse and c is the light velocity constant.

There currently exist in the market LiDAR sensors for drones with high outstanding performances.

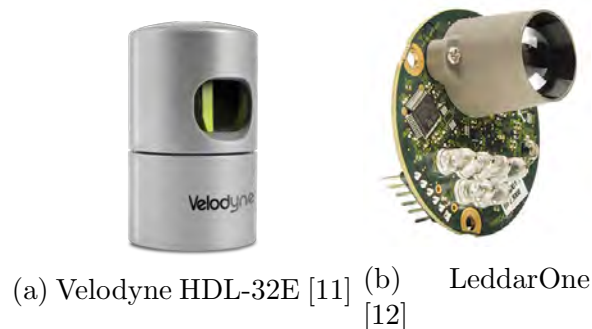


Figure 4: Two UAV applied LiDAR sensors currently in the market

2.2.2 Sound Navigation and Ranging (SoNAR)

SoNAR is based on emitting and receiving sound waves signals. By measuring the time delay between the sent pulse and received pulse it is possible to compute the distance from the target [13].

Among its applications, a significant one is measuring the depth of water surfaces when installed on ships. SoNAR sensors are much more useful on water environments rather than outdoor environments, since the propagation of sound through water is faster. It is likewise employed in submarines to detect vessels and in ships in order to locate fish targets. On medicine field, SoNARs sensors are usually employed for disease detection and monitoring unborn children.

Distance measurement is computed similarly as in the case of LiDARs. The only element that changes is the velocity, that in this case is the one of the sound, which changes depending on the medium of propagation.

A high-performance SoNAR employed in the UAV sector is illustrated in figure 5



Figure 5: MB1010 LV-MaxSonar-EZ1[14]

2.2.3 Radio Detection and Ranging (RaDAR)

Radar is an alternative technology that makes use of radio waves to measure distances. The principle of functioning is based on the reflected emitted signals. RaDAR sensors are compounded of an antenna, a transmitter and a receiver. The transmitter sends the radio-wave that propagates at the speed of light c . When reaching the objective, the wave hits the target and comes back reflected to the receiver. The reflected signal is detected and analyzed as radio echo by the antenna, and the signal is amplified by the receiver.

Civilian and military applications are within the line of applications of RaDARs [15]. They are usually employed in aircraft to measure distances from ground control stations and other objects within the airspace. Additionally, they are utilized in geology for identification of areas for mineral prospection; in agriculture for crop monitoring; in cartography for altimetric elevations; in hydrology for soil humidity detection; or in oceanography for ships detection and illegal fishing monitoring [16].

2.3 Collision Avoidance in UAVs

As mentioned in Section 1.1, the most employed collision avoidance system in current aviation is TCAS. And according to current legislation, drones aiming to cover routes within a controlled airspace should have it equipped.

However, industry is developing drones for recreative use with collision avoidance technology incorporated. The benefits are patent. For the pilot, sometimes it is every easy to get carried away in the middle of an operation. If the pilot get distracted for a while, the drone could easily fly into an object.

Drones are also employed in many crowded areas and events since they can capture films with unreachable angles for the traditional cam operators. Unfortunately, there have been some accidents and safety of the people is a must that should be preserved without exceptions.

In recent years, some models with collision avoidance equipment have been released to the market. They are all equipped with one to six sensors located in the fours sides, top and bottom. The most popular ones are Kespry 2, DJI Spark and DJI Phantom 4 Pro. The most prominent one is Kespry 2, which has 5 directions of obstacle sensing and 4 directions of obstacle avoidance. Apart from that, some intelligent flight modes are provided, as well as a super smooth stability and a top 4k camera.

Ardupilot, an open-source unmanned vehicle Autopilot Software, have an extension for Collision Avoidance algorithms in multicopters, by using cutting edge proximity sensors such as Lighthware SF40C and TeraRanger Tower. This is a feasible way for drone amateurs to incorporate collision avoidance functionality into their home-made multicopters. Current state of development of this technology lets arducopter users divide the area around the drone into 8 sectors. Each sector covers 45 °. The collision avoidance algorithm of the drone is programmed to stop the MAV when an obstacle is detected below a range of 2 meters.

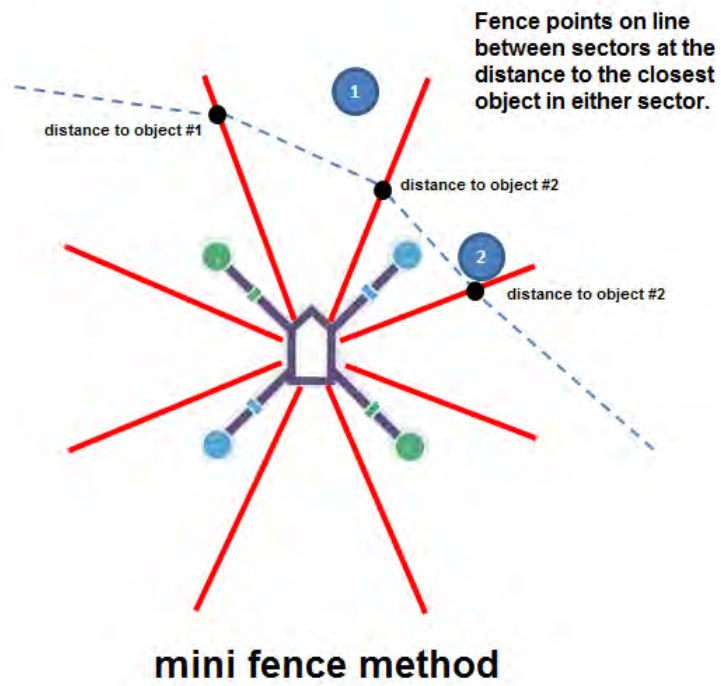


Figure 6: Arducopter collision avoidance environment sector divisions [17]



(a) Lighthouse SF40C [18]



(b) TeraRanger Tower [19]

Figure 7: High performance LiDAR sensors

3 UAV components

In this section, the UAV hardware components, the software elements and the communications among all the subsystems of the quadcopter will be presented.

3.1 Hardware Components

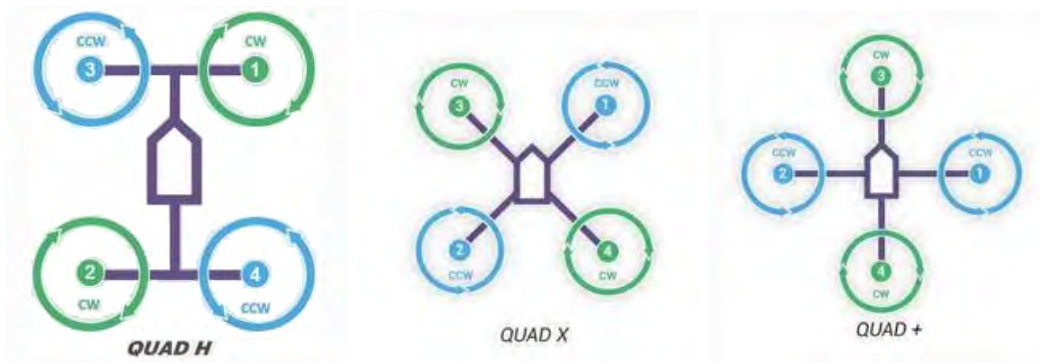
As previously explained, in order to implement the collision avoidance system into the drone, it was first needed to assembly and testing a capable flight drone. The selected drone was a quadcopter F450. It is important to remark that most of the components used within this project belongs to the Bioengineering and Aerospace Engineering Department of the University Carlos III of Madrid. Only the Pixhawk flight controller and the PPM encoder were bought to complete the scope of this project. Hence, a quadcopter was selected due to the fact that all the needed components to mount it were already present. Moreover, a quadcopter is one of the most proper MAV types to prove and implement such a complex system. Quadrotors provide the users with the possibility of hovering flights, making the collision avoidance testing more feasible and simple, taking into the account the null experience within piloting those vehicles of the writer of this thesis before immersing in this project.

Following sections introduce to the reader the hardware components employed in the execution of this project.

3.1.1 Frame

Frame is the element of MAV in charge on linking all the subsystems and providing the MAV with a physical integrity . In the current market, there exist several kinds of multirotor frames, from bicopter (two propelled MAV) to octacopter (eight propelled MAV). The frame is basically the MAV structure that holds all the components within the MAV. It is compounded by arms, landing gears, rotor mounts and central plates. Central plates usually have a dual function. It is the assembly node where landing gears and arms connect, and also include the power distribution wiring for powering the remaining components. The F450 kit frame employed within this project has two central plates. The upper plate works as linking node and the bottom plate works as PDB and as linking node as well.

There exist a relevant selection choice regarding frame: the orientation of the frame with respect to the position of the flight controller. In the case of quadcopters, three possible configurations can be selected: +, H and X.



(a) H configuration [20] (b) X Configuration [20] (c) + Configuration [20]

Figure 8: Different rotor configurations

Frame selection must be based on the further applications of the drone. The number of motors selected must be consistent with the future mission performed by the MAV. Two levers are varied depending on the number of motors: stability and endurance. For missions where endurance is a major concern, it is convenient to use copters with no more than four motors. For missions such as infrastructure inspection or crop monitoring, the stability of the drone is a significant issue that can be strongly improved by increasing the number of rotors. There exist two different frame configurations: flat and coaxial. Flat configuration only allows each arm to have one motor mounted, while coaxial configuration is designed to incorporate two similar coaxial motors on each arm.

The selected frame for this project was a quadcopter kit F450. The orientation of the rotors was X and the selected configuration was flat.



Figure 9: F450 kit

3.1.2 Rotors

The aim of rotors is lifting the quadcopter and letting it move within the whole space. The flight controller adjusts the angular velocities of the motors to move the vehicle towards the desired direction.

Motors differ according to their power type (AC or DC). AC motors are driven by alternating current (AC), while DC motors convert direct current electrical energy into mechanical energy [21]. Rotors are also classified by their mechanism to generate rotation. Figure 10 illustrates classification of rotors.

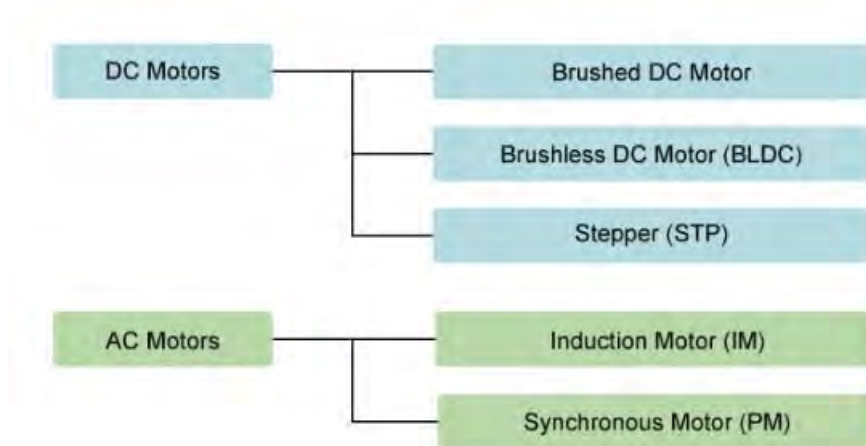


Figure 10: Rotor classification [21]

In the UAV industry, DC motors are used, specially brushless DC motors (BLDC). Brushed rotors are formed by an armature, brushes, a field magnet, an axle and a commutator. Their work principle lies on that the brushes charges the commutator with a polarity inversely to the permanent magnet, provoking a rotation of the armature. On the other hand, brushless motors don't have brushes. They are mounted in permanent magnets, normally more than four, around its perimeter. Those kind of rotors make use of control circuit. They are powered by DC electricity through an inverter or switching power supply, that produces AC electric current to control the rotor phases by closed loop controller.

The third kind of DC motors, stepped motors, are driven by pulses. They rotate at an angle (step) depending on the pulse. The rotation is commanded by the amount of received pulses. A relevant application of those kind of motors is paper feeding in printers, due to the fact that in those machines feeding occurs in fixed steps correlated with pulse count.

The size of the brushless rotors is measured by height and diameter. Rotation speeds are measured by KV rating. The KV value is given in rotations per minute (rpm)

per volt (V). For instance, the motor employed within this project has a KV of 900. A given voltage of 14 V would mean $14 \times 900 = 12600$ rpm.

The selected rotors were four units of 2212 900 Kv Brushless rotor; two clockwise rotating and the other two counter-clockwise rotating.



Figure 11: 2212 900 Kv Brushless rotor

3.1.3 Propellers

The propellers are the uppermost part of rotors. Propellers provide a downwards air flow providing a thrust to lift the quadcopter. For quadcopters, 2-blade propellers are commonly used. But propellers employed in UAV Industry can have up to 4 blades.

A significant design parameter of propellers is the pitch value. The pitch value is the angle that form the blade with the propeller diameter center. The higher the value of the pitch, the higher the angle the attack of the air with the propeller. Higher pitch values mean higher distance advanced by the vehicle in one rotation. Figure 12 illustrates the difference between high and low pitch values.

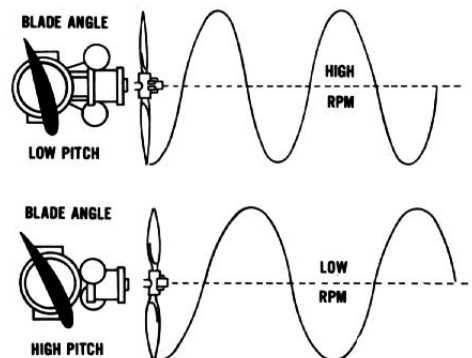


Figure 2-17 Propeller Blade Angle VS RPM

Figure 12: High Pitch and Low Pitch values [22]

The most relevant design option when manufacturing propellers is their rotation sense. Propellers can be clockwise or counter-clockwise rotating. Clockwise propellers must be always mounted on clockwise brushless motors, and the same happens with counter-clockwise propellers. A wrong mounting of the propellers could

lead to a thrust force pushing the quadcopter to the ground instead of pushing it upwards. This mistake is really common in UAV beginners.

The propellers employed in this project were Gemfan 10x4.5 APC SF style, where the first number corresponds to the diameter in inches, and the second value stands for the pitch value.



Figure 13: Selected Propellers [23]

3.1.4 Electronic Speed Controllers (ESC)

Electronic Speed Controllers (ESC) are the bond between the flight controller and the motors. What they do is transforming the electric current from battery (Direct Current) to Alternating Current (AC). The angular speed of the rotors depends on the Pulse Width Modulated (PWM) sent by the ESC controllers.

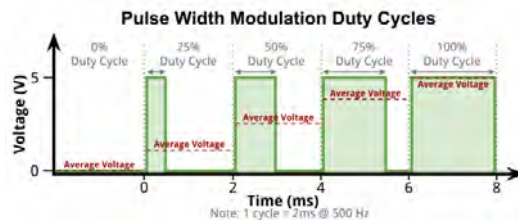


Figure 14: Duty cycles [24]

ESCs has maximum current and tension values permitted as well as rotors. That is an important issue to take into consideration when selecting a rotor compatible with an ESC. The maximum current supported by the ESC must be always equal or higher than the maximum one supported by the rotor. It is also strongly suggested that ESCs have a threshold margin between maximum allowed current between rotors and ESCs.

The selected ESCs for this project were four units of Flysky 30A.

3.1.5 Power

Nearly all UAV are powered by lithium polymer batteries (LiPo). Those kind of batteries have great power capacity and light weights. Those two features make

them perfect to be the source of energy of a device that is going to fly and that need to have low-weight payloads to increase its endurance and range.

LiPo batteries are compounded by cells, each of one with a nominal voltage and are commonly plugged in series. The nomenclature for the number of cells in a LiPo battery is XS, where X stands for the number of cells and S for the serial connection. For instance (the battery employed in the elaboration of this project), a battery with three cells connected in series is denominated as a 3S LiPo battery.

LiPo batteries are classified by its power capacity and discharge (C) rating. Power capacity is given in miliamperers per hours (mAh). C rating basically defines the rate at which the battery is able to output its power. The battery selected in this project has a capacity of 4000 mAh and a C rating of 30C; so the maximum current draw it can output is $4 \times 30 = 120$ A.



Figure 15: Selected Battery [25]

3.1.6 Flight Controller

The flight controller is mandated to control all the quadcopter behaviors. In this project, a Pixhawk 2.4.8 was employed. Pixhawk is defined as an independent open hardware project to provide UAV autopilot features commonly employed in civil, industrial and military purposes [26].

Pixhawk flight controller runs an internal real-time operating system denominated NuttX. The alternative flight controller currently in the market is the one produced and developed by ArduPilot project: the Ardupilot Mega (APM). Pixhawk is considered as an improved flight controller with respect to APM. It has internal compass, additional telemetry ports and Floating Point Unit co-processor, that speeds up all mathematical computations.



Figure 16: Pixhawk 2.4.8 [27]

- **Firmware**

Flight controllers need an internal firmware. Firmware is defined as software written into hardware device's nonvolatile memory [28]. The firmware used in this project belongs to the open-source software provided by ArduPilot. Ardupilot provides a firmware for each of their supported vehicles: multi-copters, fixed plane UAVs, helicopters, rovers and underwater vehicles. In this project APM Copter 3.4.6 is the run firmware into the quadcopter. This firmware, which is freely available in the official page of ardupilot, is already fixed to be installed in the flight controller and start setting the first-time configuration tasks. It already includes its own flight modes to perform the desired operations.

- **Pixhawk sensors**

Pixhawk flight controller is equipped with internal sensors that retrieve the required information for its proper performance.

1. Inertial sensors: the inertial sensors are mounted in an electronic board called Inertia Measurement Unit (IMU). First inertia sensor is a 3-axis gyroscope which retrieves 3-axis angular accelerations. Second inertia sensor is a 3-axis accelerometer, which retrieves linear accelerations. The IMU purpose lies on retrieving the required information to maintain the quadcopter stabled during its operation. The flight controller processes that information and outputs required angular velocities throw the ESCs to maintain the drone stable reaching an horizontal balance in the longitudinal and lateral axis.
2. Barometer: a barometer is used to retrieve data of the air pressure to the flight controller. This data is processed and an altitude above ground level is estimated. This sensor is really useful for instance in *altitude hold mode*, where throttle is adjusted to maintain a desired altitude.
3. Magnetometer: a magnetometer is employed to compute the magnetic heading of the flight controller with respect to the magnetic north.

4. External sensors: external sensors can be plugged into the Pixhawk flight controller through the I2C, SPI and Serial ports. A critical sensor needed for the proper performance of guided flight modes is a GPS sensor, which is plugged into the Pixhawk by the GPS port. It can also be connected an additional compass(magnetometer) through the I2C port. An additional magnetometer helps the system to retrieve more precise headings, based on the comparison of the electromagnetic fields measured in the internal magnetometer of the Pixhawk and the external one.

- **Safety switch and Buzzer**

Pixhawk flight controller kit includes a safety switch and a buzzer.

The aim of the safety switch is blocking the PWM communication between the flight controller and the ESCs, to allow safe UAV manipulations even when the battery is connected.

The buzzer is a device that emits different kind of sounds depending on the status of the quadcopter. It is a manner of communication between the flight controller and the pilot without the use of the Ground Control station. For instance, when the UAV is armed, it emits a characteristic bip for two seconds.

3.1.7 Arduino Mega 2560

Arduino Mega 2560 is a single-board microcontroller. It is the main hardware device of the collision avoidance system together with the ultrasonic rangefinders.

Arduino is an open source computer hardware and software company that produces microcontroller kits employed on building digital devices whose main applications are sensing and object controlling in the physical and digital world [29].

In this project, the purpose of the Arduino Mega 2560 board is processing all the distance sensor information, and based on that, emitting the collision avoidance motion maneuver order to the flight controller. In particular, it sends overwriting values of the pitch or roll (depending of the location of the obstacle) in order to make the drone move away from the obstacle. The link between the arduino and the Pixhawk flight controller is established through serial communication, by the Universal Asynchronous Receiver/Transmitter (UART) hardware-integrated circuit, used for serial communication through the serial port. Further details about the connections between the components are given in following sections.



Figure 17: Arduino Mega 2560 [30]

3.1.8 Telemetry

Telemetry connection between the ground control station and the drone is needed in order to monitor the flight attitude of the drone as well as notifying any failure or incident during the flight. Also, for some flight modes, telemetry is crucial, specially for autonomous flight modes, where the UAV follows a flight route based on way-points set in the ground control station.

In this project, a SiK Telemetry Radio was used. This model allows ranges under 300 m. The telemetry kit has two transmitter and receivers; one on-board the UAV and the other one on the GCS. The frequency is 433 MHz and the protocol of communication with the Pixhawk flight controller is MAVLink.



Figure 18: SiK Telemetry Radio [31]

3.1.9 Radio transmitter and receiver

For pilot commanded flight modes, one of the most relevant aspects is having a proper communication between the pilot and the quadcopter. This task is carried out by a Radio Control (RC) transmitter and a Radio Control (RC) receiver. The frequency of the radio signal is 2.4 GHz. The receiver is responsible of converting the received radio signal into a Pulse Width Modulated Signal (PWM). In this project, the Pixhawk flight controller is used, and it has not PWM input, but PPM. PPM

stands for pulse position modulation. In PPM, the signal is conformed by a set of pulses of fixed length. Between the pulses, there are pauses of variable length.

Therefore, a PPM encoder was used to convert the PWM signal from the RC receiver to PPM to input the Pixhawk flight controller. So, the PWM signals of each of the 5 used channels (roll, pitch, yaw, throttle and flight mode) are all encoded in a single PPM signal. Since the the Flysky FS-i6 RC transmitter employed in this project has six channels, the remaining one is not used. This channel without use could be applied for servo motor control for instance, usually used when a camera as payload is on-board the quadcopter.



Figure 19: Flysky FS-i6 RC transmitter and receiver [32]

3.1.10 Ultrasonic range finder

Ultrasonic range finder is a SoNAR sensor and it has the work principle explained in Section 2.2.2. The utilized sensor in this project is the HC-SR04 sonar, which is connected to the Arduino Mega 2560 microcontroller board, where the sensor distance processing takes place.



Figure 20: HC-SR04 Ultrasonic Finder [33]

3.2 Software elements

3.2.1 Ubuntu 16.04

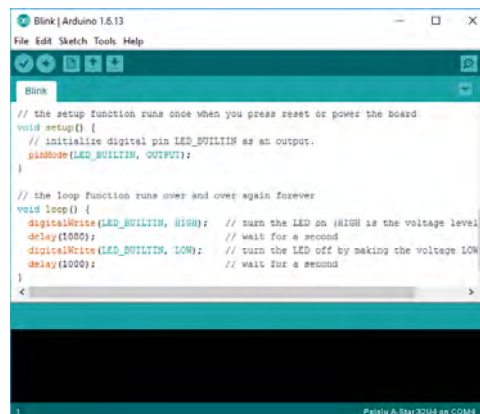
The operating system employed in this project is Ubuntu 16.04. It is a free and open source operating system based on Debian. The open-source software used within the

development of this project is run in this operating system.

3.2.2 APM Planner

ArduPilot is an open source, unmanned vehicle AutoPilot Software suite, developed with the main objective of guiding and controlling unmanned systems. It provides software running on the vehicle controller (i.e. the firmware); as well as ground control station software such as Mission Planner, APM Planner, QGroundControl or MavProxy. Mission Planner is the Ground Control Station software utilized by most of the users. However, it is only supported in Windows. Since the operating system employed within this project is Ubuntu 16.04; the available and compatible GCS software has been APM Planner 2.

Ground control station software is very useful for advanced missions and proper calibration of the vehicles, when it is necessary to tune the control parameters that control loop necessitate for real-time computations. When connecting the vehicle to a GCS, the control does not lie only on the 6 channels of the RC transmitter. High-level commands can be sent from Ground Control station to the vehicle, as well as monitoring real-time information of the attitude and state of the UAV. The vehicle can be connected to the GCS by USB wire connection to upload the firmware and perform the calibration procedures. But when the quadcopter is on the air, all real time information monitoring and advanced commands sending (for guiding modes) is performed via telemetry connection, whose hardware was introduced in section 3.1.8.



```
Arduino IDE interface showing a sketch for a Blink LED. The code includes comments and functions for setup and loop, with delays of 1000ms.
```

```
Arduino IDE interface showing a sketch for a Blink LED. The code includes comments and functions for setup and loop, with delays of 1000ms.
```

Figure 21: Arduino IDE interface[34]

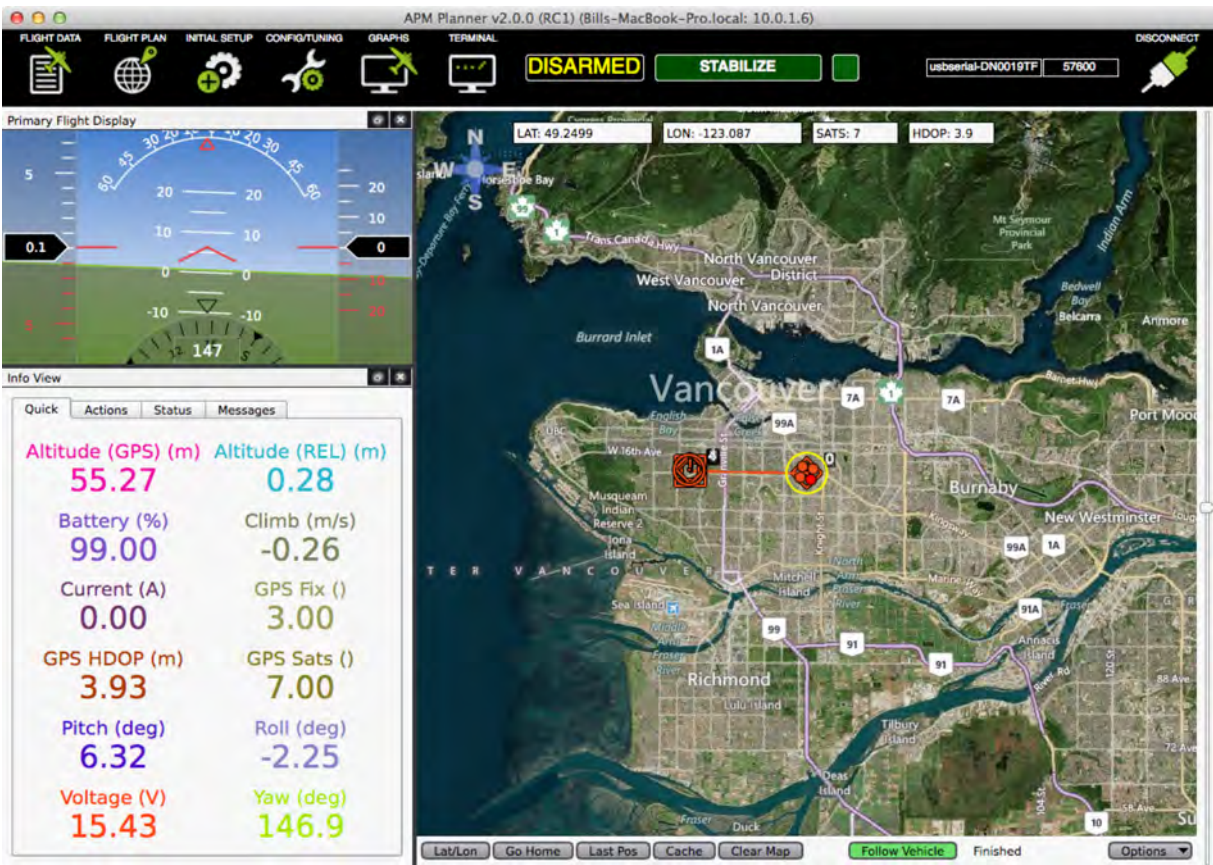


Figure 22: APM Planner interface

Arducopter firmware, which is the specific software uploaded to the quadcopter, has different flight modes implemented, each of them with different advanced features. There are many flight modes, some of them for very advanced and complex missions. In this project, only two of them were used to check the proper functioning and proof concept of the obstacle collision avoidance system implemented into the quadcopter. The utilized flight modes were STABILIZE and ALTITUDE HOLD modes [35].

- STABILIZE MODE:** this flight mode uses the attitude information given by the IMU installed in the flight controller, to control the quadcopter. Roll and pitch angles input by the pilot control the lean angle of the copter. When the pilot releases the sticks of its corresponding channels (channel 1 and channel 2), the vehicle levels itself by yielding pitch and roll values computed based on control open and close loops run inside the flight controller. However, when flying outside, the pilot will need to adjust pitch and roll values to maintain the vehicle on its position to compensate wind effects. Yaw input from the pilot controls the rate of change of the heading. If the stick of this channel (channel 3) is realized, the quadcopter should maintain constant its heading.

Pilot's throttle input controls the average motors speed, and therefore, its constant adjustment is necessary to maintain the altitude of the vehicle. The

throttle sent to the rotors is adjusted automatically based on the tilt angle of the vehicle, to reduce compensation from the pilot when attitude changes [36].

- **ALTITUDE HOLD MODE:** yaw, pitch and roll angles operate in the same way as in the Stabilize Mode. However, in this mode, the throttle is adjusted to maintain a constant altitude, based on the tilt angle and pressure information retrieved from IMU and barometer sensors installed inside the flight controller.

3.2.3 Arduino IDE

The Arduino integrated development environment (IDE) is a cross platform application written in Java. It provides a code editor, with simple one-click mechanisms to compile and upload programs to the arduino board [29]. Additionally, it provides message area, text console, toolbar with buttons for usual functions and hierarchy of operation menus.

Arduino IDE supports C and C++ programming languages. The code writing only entails two main functions:

- Setup: initializing the variables and constants.
- Loop: main program loop.

Arduino IDE utilizes the program 'avrdude' to transform the executable code into an hexadecimal text file loaded to the Arduino micro-controller board by a loader program.

In this project, a program is developed in an arduino sketch to process the distance input data from the sensors and output specific overriding values to change the trajectory of the quadcopter and avoid a collision.

3.3 Communications

As mentioned in previous sections, a quadcopter can be controlled by two possible ways. First one is by direct control of the pilot through a RC transmitter. Second one is by ground control station link through a telemetry radio installed in both, GCS and the vehicle.

3.3.1 RC control

This is the simplest way of controlling an UAV. It is mostly employed in civil use and leisure flights. All the channels remain under the control of the pilot, which should be trained properly to command the drone.

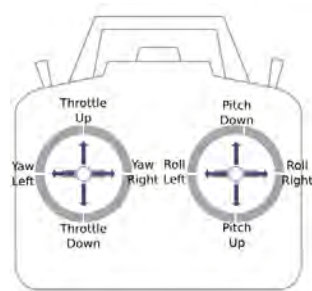


Figure 23: RC transmitter Euler angles and throttle sticks [37]

The four basic necessary channels to command a drone are the Euler angles and the throttle, as depicted in Figure 23. In this project, it is also used the fifth channel to control the flight mode of the drone. The information is sent to the RC receiver through radio waves. The receiver converts this signal to PWM for each of the six channels and sends it to the PPM encoder, which transfers the PPM signal to the flight controller.

3.3.2 MAVLink

MAVLink stands for Micro Air Vehicle Link, and is a protocol of communication developed with the purpose of information exchange between a GCS and MAVs. It was designed by creating small and light libraries containing the needed information for the interchange of sent and receive messages during GCS-MAV communication. MAVLink is pretty feasibly adaptable in any kind of GCS and vehicle, since the message definition is written in C. In spite of the current libraries have a wide range of messages, new messages can be created by users by developing new libraries [38].

The messages are byte-encrypted, and can be understood and interpreted by the GCS, the flight controller, or an additional micro-controller board as Arduino Mega 2560, the one utilized within this project. Messages such as heartbeat, arming, disarming, flight mode change or overwriting channels can be instructed by the GCS or a micro-controller board, and immediately thereafter interpreted and executed by the flight controller.

The MAVLink package consists of a codified sequence of bytes. Table 2 shows the structure and function of each of the bytes that conform a message package.

Among the autopilots that support this protocol of communication, there are many of them [38]:

- **Autopilots:** ArduPilot Mega, pxIMU Autopilot, SLUGS Autopilot, FLEX-IPILOT, MatrixPilot, SenseSoar Autopilot, SmartAP Autopilot, AUtoQuad 6 AutoPilot, Pixhawk Autopilot.

Table 2: MAVLink bytes explanation

Byte Index	Byte	Explanation
0	Start	Flag the start of a new packet
1	Longitude	Gives the longitude of the payload, in number of bytes, from 0 to 255 bytes.
2	Sequence	Gives a sequence number to the send message in order to order them in the receiver target system.
3	System ID	Identifies the number of the sending system, letting to distinguish between different sending systems.
4	Component ID	Identifies the component inside the sending system that has sent the command.
5	Message ID	Identifies the kind of sent message, so the receiving target knows how to read and interpret the received message.
6+n n=[0-255 bytes]	Payload	The payload can be conformed from 0 to 255 bytes and contains the content of the message itself.
[n+7,n+8]	Checksum	This last part of the message can weigh 1 or 2 bytes and its purpose is the control of mistakes, protecting the packet from decoding it in the wrong way.

- **GCS Software:** iDroneCtrl, QGroundControl, HK Ground Control Station, APM Planner, Mission Planner, MAVProxy.

One of the key aims of this project, as well as a tough objective, is creating a proper communication link among the main components of the UAV. Integrating the Obstacle Collision Avoidance System into the rest of the UAV had many difficulties. Undoubtedly, one of them was establish a MAVLink communication between the arduino board and the flight controller. The required maneuver execution order were sent from the Arduino Mega 2560 by MAVLink protocol of communication.

This task was done following a systematic approach, from very simple commands testing, to more complicated ones, finalizing in a successful concept proof of a collision avoidance maneuver.

It is also remarkable to take into the account that the distance data processing did not have nothing to do with MAVLink. A program was created to process that information inside the arduino board, and based on that, sending through MAVLink overwriting values of Euler angles to perform the maneuver. Therefore, the utility of MAVLink protocol of communication within this project lied on sending attitude corrections, instead of distances.

MAVLink communication is established in this project throw two physical connections:

- **USB connection:** established through serial communication, by the Universal Asynchronous Receiver/Transmitter (UART) hardware-integrated circuit, when the Pixhawk flight controller is connected to the GCS to perform calibrations and firmware updates; and for the arduino Mega 2560-flight controller connection.
- **Radio telemetry connection:** when the quadcopter is flying, to monitor in the GCS real-time information and to define control parameter values in the GCS without the need of disarming the vehicle.

4 Collision Avoidance System Design

In this section, the sensing and collision avoidance system design will be introduced.

4.1 Collision Avoidance System Requirements

First step prior to the design of any subsystem, consist on setting the requirements to be fulfilled. A model of classification based on INCOSE system requirements [39] was followed, as shown in table 3.

Table 3: Collision avoidance system requirements

Req. ID	Requirement
Functionality	
1.1	The collision avoidance system should detect the obstacles located on each of its four sides.
1.2	The collision avoidance system should constantly detect the altitude at which the MAV flies.
1.3	The collision avoidance system should maintain the MAVLINK communication link with the flight controller every time.
1.4	The collision avoidance system should successfully process the received distance data.
1.5	The collision avoidance system should successfully command a maneuver to avoid a collision.
Integration	
2.1	The collision avoidance system shall be successfully implemented in the MAV.
Performance	
3.1	Each of the four lateral ultrasonic range finders should be operative among the whole mission.
3.2	Each of the four lateral ultrasonic range finders should be powered among the whole mission.
3.3	The ultrasonic waves should propagate successfully within the environment where the MAV operates.
3.4	The ultrasonic range finder located in the bottom of the quadcopter should be operative among the whole mission.
3.5	The ultrasonic range finder located in the bottom of the quadcopter should be powered among the whole mission.

Req. ID	Requirement
3.6	The microcontroller board responsible of the maneuver commanding should constantly maintain the communication link with the flight controller by sending heartbeat messages.
3.7	The collision avoidance system should get the control of MAV and overwrite commands to the ones sent by the pilot.
3.8	The collision avoidance system should operate autonomously.
Usability Requirements	
4.1	The collision avoidance system should be tested in flight operation.
4.2	The collision avoidance should not consume energy destined for the rest of subsystems.
Interface Requirements	
5.1	Communication among the collision avoidance system and the rest of the MAV should be successfully established.
5.2	The collision avoidance system should be activated prior to the start of the operation.
5.3	The collision avoidance system should operate independently of the pilot commanding.
5.4	The collision avoidance system should be detected by the GCS
Operational Requirements	
6.1	The collision avoidance system should not obstruct the pilot's execution.
6.2	The collision avoidance system should not require further maintenance than checking the proper state of the sensors and the wiring connections.
6.3	The collision avoidance system should detect any obstacle located in front any of the four sides of the MAV.
6.4	The collision avoidance system should increment the safety of the MAV operation.
States requirements	
7.1	There should be three main stages in flight operation: minimum altitude activation system, obstacle monitoring and collision avoidance.
Physical requirements	
8.1	Once the collision avoidance is implemented, the MAV should remain under 2 Kg MTOM
Design requirements	
9.1	The collision avoidance system admits up to 6 ultrasonic range finders installed simultaneously.

Req. ID	Requirement
Policies and regulations requirements	
10.1	The MAV should comply with the Spanish legislation related to RPAs
Cost requirements	
11.1	The collision avoidance system elements should not cost more than 100 €

The requirements described in the table were almost completely covered during the development of this project. There is one important issue that was not accomplished due to its technical difficulties. Having the four sensors working simultaneously was a tough task, which was not achieved. Lateral sensors were tried and tested separately, in different flight tests. Therefore, system requirements 1.1, 3.1, 3.2, and 6.3 were not considered to be properly fulfilled.

4.2 Collision Avoidance System Stages

The collision avoidance system follows four stages, considering a regular operation from take off to obstacle collision maneuver: minimum altitude clearance, horizontal environment sensing, obstacle detection and collision avoidance maneuver.

4.2.1 Minimum altitude activation level

This stage is defined as a necessary barrier for the safety and successful operation of the collision avoidance system. Its purpose consists on activating the collision avoidance system once the quadcopter has reached a minimum altitude of one meter AGL. Without this stage, the collision avoidance system would be active from the very beginning of the operation (i.e. 0 meters AGL). This could lead to a collision avoidance maneuver too close to the ground, risking the physical integrity of the quadcopter.

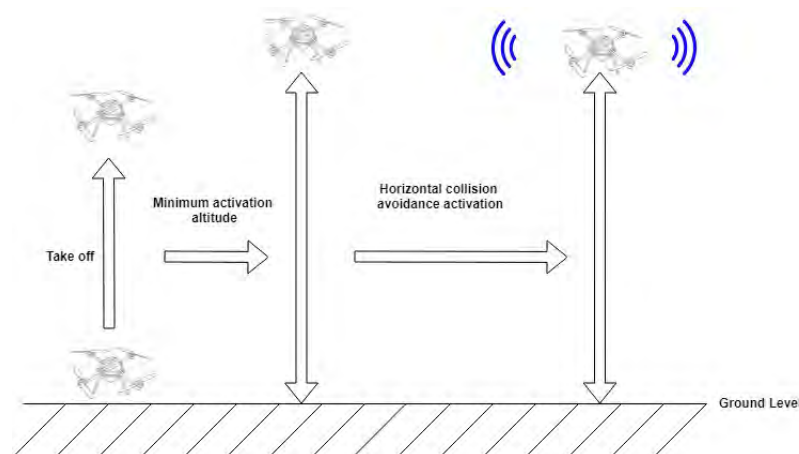


Figure 24: First stage

This task was performed by installing an ultrasonic range finder located at the bottom of the quadcopter. By constant ultrasonic waves sending, the Arduino microcontroller board processes the distance to ground information and activates the horizontal obstacle detecting system once a minimum altitude of one meter is reached.

It is important to take into the account the possible limitations of this stage. The ideal environment would be a flat ground surface, parallel to the x-y plane of the quadcopter, where ultrasonic waves could 'bounce' properly and come back to the ultrasonic rangefinder receiver. However, most of the tests were performed in a grass ground, due to its soft surface in case of ground impact. The irregularity of the grass could have changed the vertical direction of propagation of the waves, letting reach back the receiver just a portion of the sent pulses.

4.2.2 Horizontal environment sensing

Once the minimum altitude threshold has been reached, the horizontal environment sensing is activated. This task is performed by locating ultrasonic range finders on each of its four sides. Those sensors were constantly emitting pulses searching for potential obstacles. Based on the time the waves took to come back to the receivers, distance data is obtained and processed within the arduino microcontroller board. Hence, this is a stage that lasts when an obstacle is detected to be a potential risk for the safety of the operation.

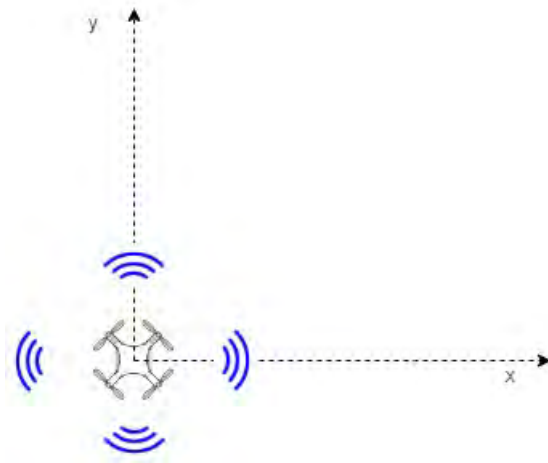


Figure 25: Second stage

It is worth to mention that this functionality was not totally implemented within this project. The horizontal detecting system was developed following a systematic procedure. Firstly, it was installed just one sensor on one side, to test if the obstacle distance was properly measured. Then, that sensor was located on the remaining three sides of the quadcopter, to follow the same procedure. The collision avoidance algorithm was designed to just operate in one side of the quadcopter on each single operation. A significant future work would lie on mixing the four sides in one single operation. To do that, the code run inside the arduino board must be recoded to monitor the four distances at the same time and emit collision avoidance maneuvers, depending on the side where the obstacle is detected. This a tough task, specially under situations where the drone encounters with more than one potential obstacle present in more than just one side.

4.2.3 Obstacle detection

The distances to the objects around the quadcopter are constantly updated inside the arduino microcontroller board. When that distance is lower than a threshold value of *1.2 meters*, a potential risky obstacle is detected, and the last stage takes place.

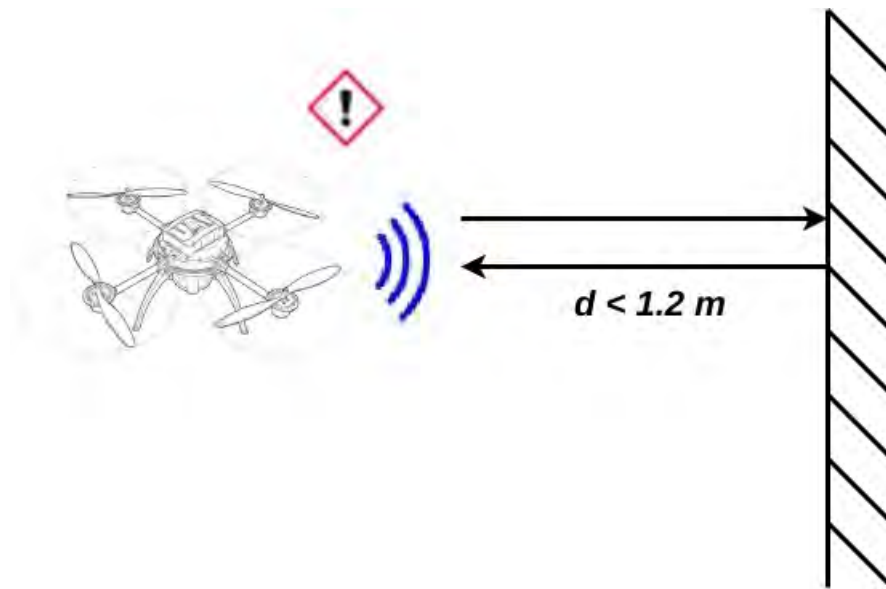


Figure 26: Third stage

4.2.4 Collision avoidance Maneuver

The collision avoidance maneuver is the last and most critical stage. It starts when an obstacle lies inside the threshold range below *1.2 meters*. The arduino microcontroller board sends collision avoidance maneuver commands to the Pixhawk flight controller through MAVLink in order to ward the MAV off the obstacle. The maneuver commands are sent until the obstacle is considered to be far enough from the vehicle and ceases to be a potential risk for the physical integrity of the quadcopter. In other words, the collision avoidance maneuver lasts when the obstacle is outside the threshold range of *1.2 meters*.

During this stage, the quadcopter remains under the exclusive control of the collision avoidance system, which operates autonomously without the need of any activation of the pilot.

The collision avoidance maneuver consists on rotating the vehicle around the pitch or roll angle, depending on the location of the obstacle with respect to the quadcopter. By rotating around the pitch or roll angle, the lift force direction is no longer parallel to the *z-axis*, generating then two force components: an *xy-plane* parallel force and a *z-axis* parallel force. The first component is the target of the pitch/roll maneuver; and its direction should points oppositely to the obstacle. That force component is named as L'_y or L'_x , depending on the modified Euler angle, roll or pitch, respectively. Figures 27 and 28 illustrate the collision maneuver phases for roll maneuver. Figures 29 and 30 illustrate the collision maneuver phases for pitch maneuver.

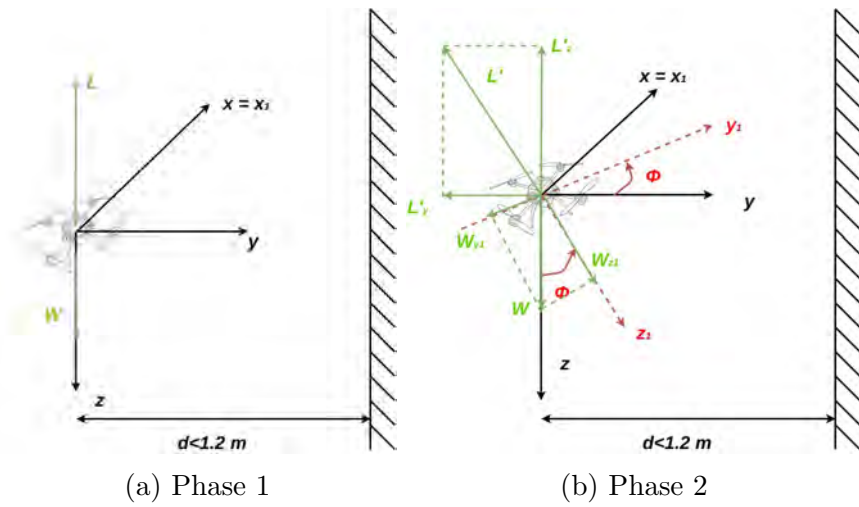


Figure 27: Collision avoidance roll maneuvers phases 1 and 2

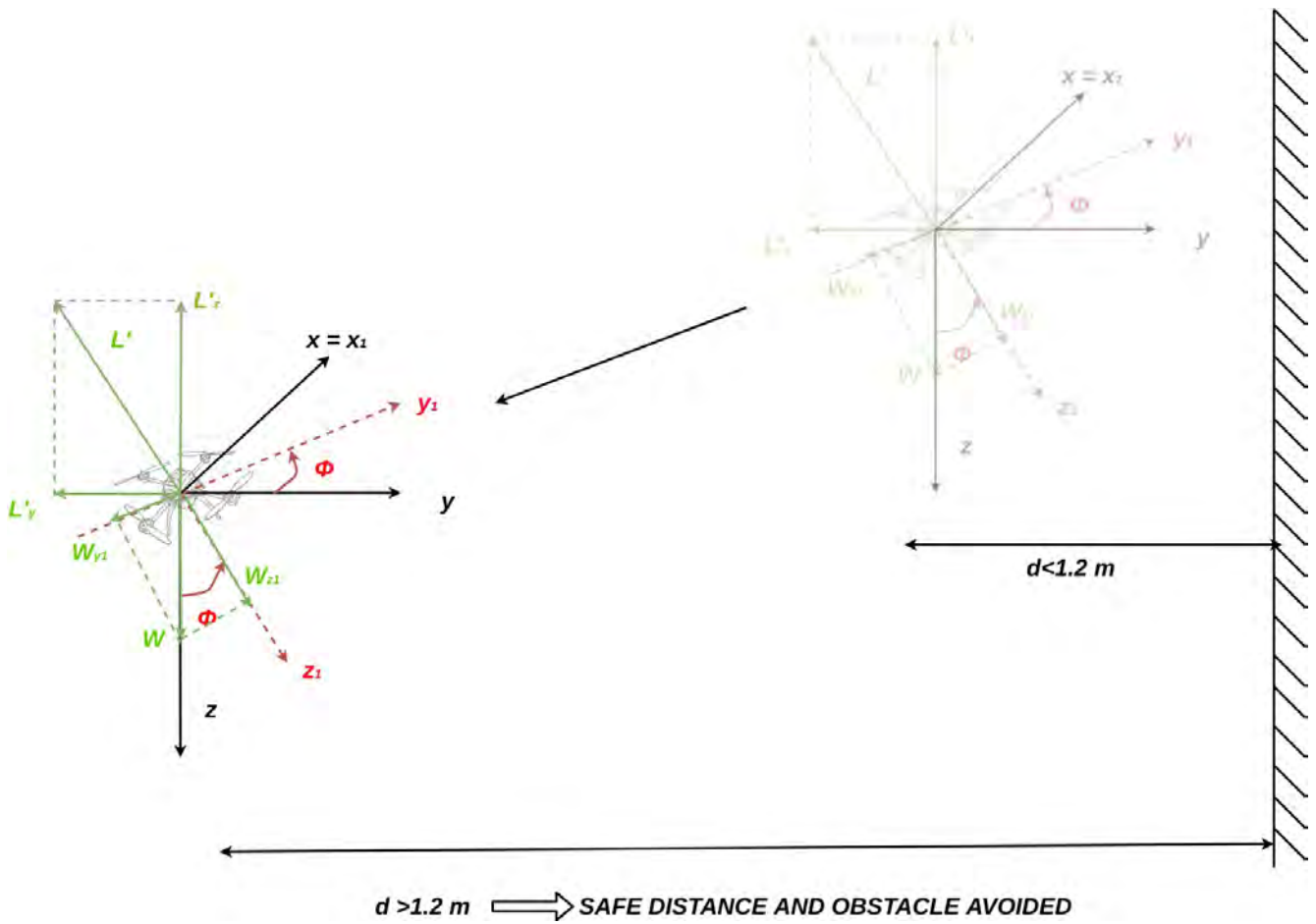


Figure 28: Collision avoidance roll maneuver phase 3

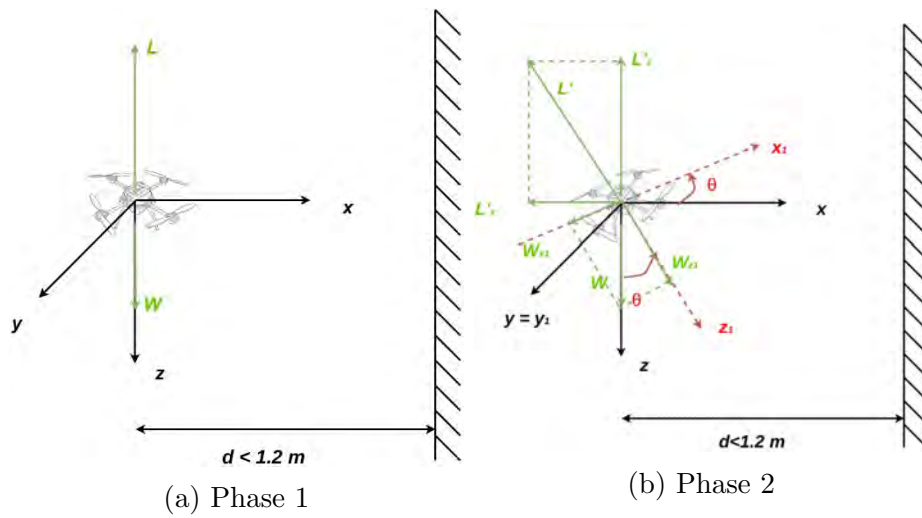


Figure 29: Collision avoidance pitch maneuvers phases 1 and 2

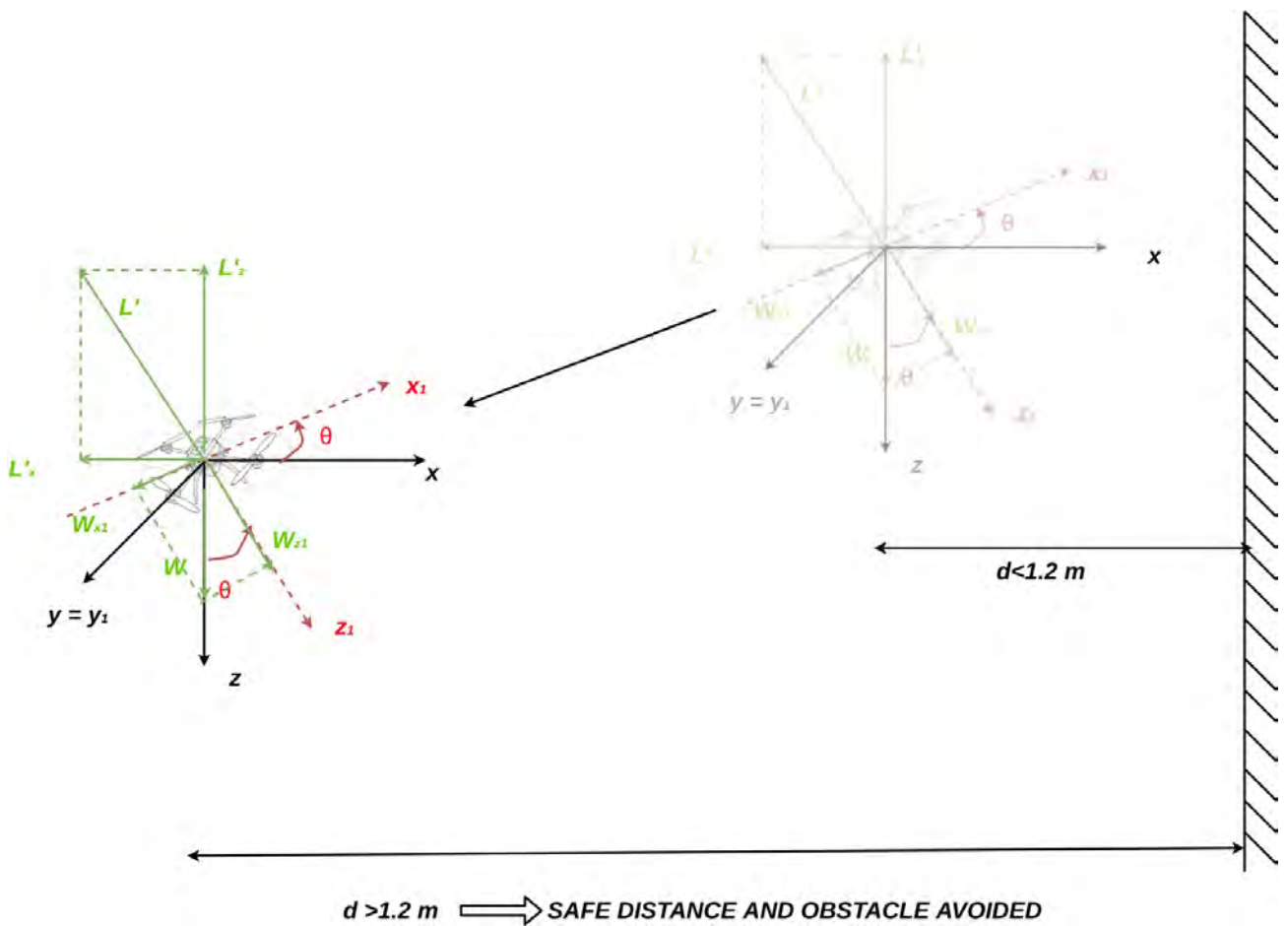


Figure 30: Collision avoidance pitch maneuver phase 3

As shown, L'_y or L'_x lift forces are the ones that push the vehicle away from the

obstacle. However, there is a drawback in this solution. Initially, the drone in phase 1 is considered to be hovering, existing total equilibrium of forces, and the weight of the vehicle is compensated by the lift force provided by the rotors. When the vehicle rotates around the roll or pitch angle, the lift force changes its direction, and the lift force component along the initial z -axis is no longer valid to compensate the weight of the quadcopter. This provokes a descent of the vehicle towards the ground, unless the lift (i.e.thrust) force is manually compensated by the pilot. This relevant issue can be solved by changing the flight mode to altitude hold mode, as will be explained in following sections.

4.3 Collision Avoidance Algorithm flow diagram

An explanatory flow diagram architecture has been drawn, where the stages of the collision avoidance implemented in a custom MAV mission are shown.

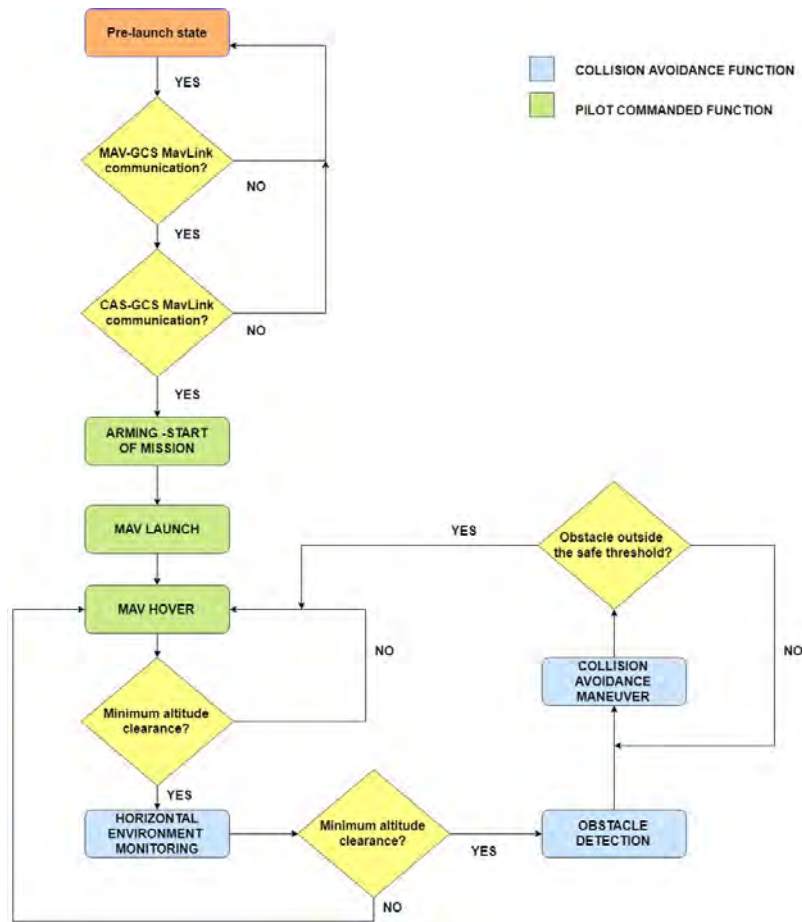


Figure 31: Collision avoidance algorithm flow chart

5 Collision Avoidance System Implementation

In this section, the physical and software integration of the Collision Avoidance System elements will be detailed.

5.1 Quadcopter assembly, wiring and calibration-making the drone flight capable

As mentioned in section 1.4, first objective within this project is building a flight-capable drone from scratch, making use of the hardware introduced in section 3. Although building the drone is not the main purpose of this project, it is a necessary and intermediate step to have a testing platform on which integrating the CAS and proving its performance. Some of the main assembly phases and principal connections will be detailed in this section. The followed procedure is based on the quadcopter assembly lab notes from the Master in Aeronautical Engineering at University Carlos III of Madrid [40].

- **Power distribution board soldering:**

First step when mounting a quadcopter consists on soldering the electronic speed controllers and the battery connector port to the power distribution board. Each ESC is plugged to its corresponding pad in the power distribution board, soldering the black wire to the negative (-) pad and the red one to the positive pad (+). Since the PWB belongs to the F450 kit, it has four pads for the four ESCs. After that, the power cable is soldered to the main power board, following the same procedure: red cable to positive pad (+) and black cable to negative pad (-). The power cable is soldered on its other end to a XT60 connector to receive energy from the battery. Figures 32 and 33 illustrate the ESCs soldering and the XT60 connector.

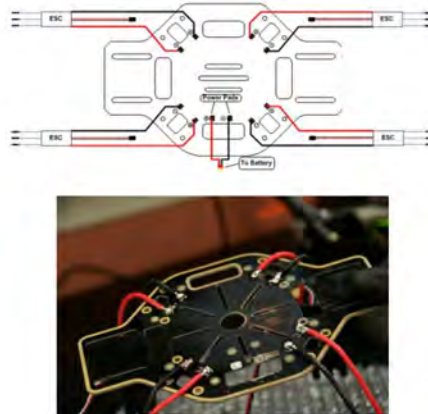


Figure 32: ESCs connections



Figure 33: XT60 connector

- **Frame assembly**

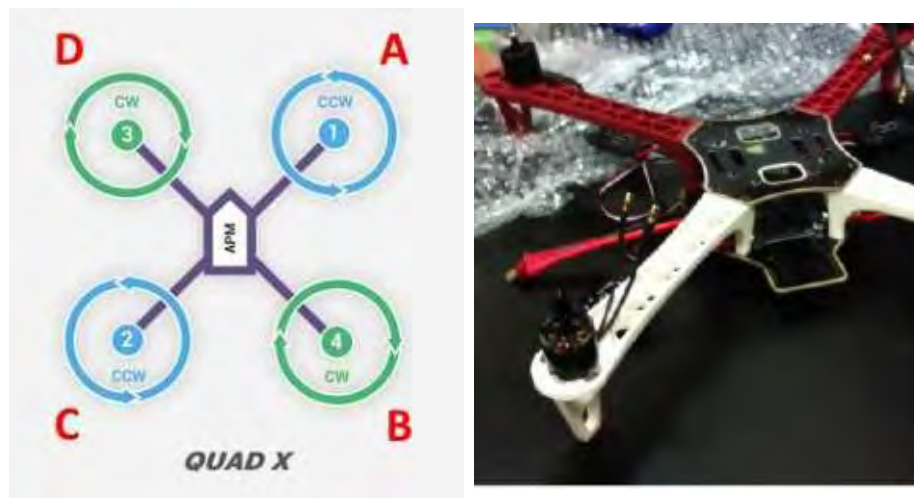
Next step consists on assembling the F450 frame, ergo the four arms to the bottom (i.e. PWD) and upper plate. Each arm is screwed to the plates by using M2.5*6 mounting screws and the hex-wrench. Figure 34 shows the resulting frame after being assembled.



Figure 34: F450 assembled frame

- **Motor assembly**

Next step consists on screwing the four motors to the end of each arm by utilizing M3*6 mounting screws. It is relevant to remark that two kind of motors are used: clockwise rotating and counter-clockwise rotating. To distinguish them, black-head motors are counter clockwise rotating and silver-head motors are clockwise rotating. Figure 35a illustrates the selected criteria of rotation for the quadcopter. As observed, diagonally aligned motors rotate in the same direction. This motor disposition is not selected in vain, it has its physical reason. When a rotor actuates, it creates a torque effect on the vehicle body. Therefore, if all the motors rotate in the same direction, the quadcopter will yaw due to the effect of this torque. To counteract this effect, two rotors are spin in one direction, and the other two in the opposite one, to get an stable quadcopter, whose yaw rotation only depends on the flight controller inputs.



(a) Spinning criteria

(b) Rotors assembly

Figure 35: Spinning criteria and rotor assembly

- **Propellers to motors**

Next step is assembling the propellers to the motors. As mentioned in the previous section, their rotating direction must be consistent with the disposition of the rotors. Propellers are manufactured depending of its rotation. Therefore, counter clockwise rotating propellers are coiled in black-head rotors and clockwise propellers are coiled in silver-head rotors.

- **Legs to frame**

Legs provide the quadcopter protection in landing operations as well as stability in take off operations. Additionally, its use is crucial in this project, considering that the main element of the CAS (the arduino microcontroller) is located in the bottom plate. Landing without legs would damage the microcontroller board since it would touch directly the ground.



Figure 36: F450 legs

- **ESCs to motors**

ESCs has three wire inputs from the motors. The order of the connections varies the rotation of the rotors, according to figure 37. Therefore, the wiring must be done consistent with the desired rotation.

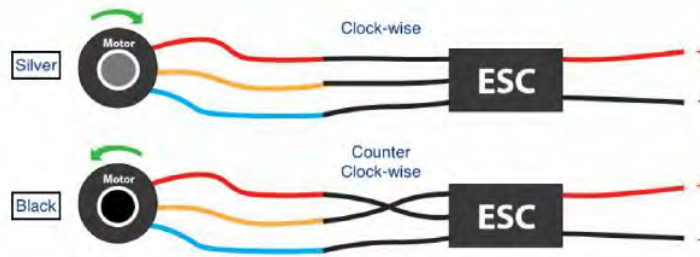


Figure 37: Wiring order

- **Pixhawk flight controller to frame**

The Pixhawk flight controller is not taped directly to the upper plate of the frame. The upper plate has lots of vibrations during flight. That vibrations would surely affect the performance of the flight controller IMU sensor. To avoid this issue, a cushioning sponge is used in the middle to absorb those vibrations.



Figure 38: Cushioning sponge

- **Radio receiver and PPM encoder assembly**

The radio receiver is taped to one of the arms of the quadcopter. It must be ensured that the blades do not touch this item when rotating. Next to it, the PPM encoder is also taped.

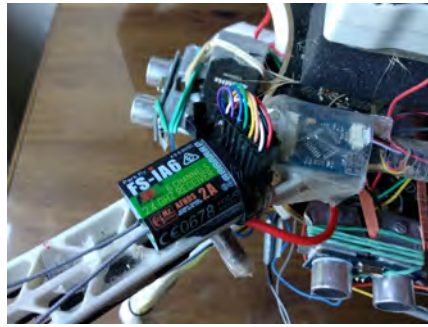


Figure 39: Radio receiver and PPM encoder

- **Pixhawk flight controller wirings**

Figure 40 shows the pins inputs and outputs of the Pixhawk flight controller. The ESCs are plugged in 1 to 4 output pins, and the PPM encoder is plugged to the RC pins.

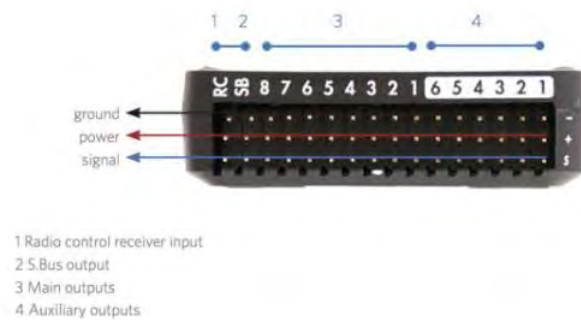


Figure 40: Pixhawk flight controller pins

- **Calibrations**

In order to make the quadcopter flight capable, several calibrations need to be performed prior to flight testing.

First of all, the frame configuration must be selected. For the quadcopter used within this project, X frame configuration was employed.

After that, the internal compass of the Pixhawk flight controller must be properly calibrated. It must be done by connecting the flight controller to the ground control station through the USB cable or through telemetry link. APM planner 2 ground control station has among its functionalities a pre-set calibration process to calibrate the internal sensors of the selected flight controller and the selected RC transmitter.

During the compass calibration, the GCS asks the user for rotating the drone around the Euler angles for 60 seconds. By doing this, the GCS acquires cali-

bration samples and at the end of the process, the GCS displays the resulting offset values.

Then, the accelerometer calibration must be performed. The GCS has also a pre-set functionality to do it. It prompts the user to locate the drone on six orientations: nose up, nose down, left, right, level and back. Once completed, the accelerometer should be precisely calibrated.

Next step is calibrating the RC transmitter. The purpose of this calibration is setting the maximum and minimum possible values input by the user into the four channels of the RC transmitter. The GCS prompts the user to move the sticks on all directions reaching the maximums and minimums. Once the process is completed, the GCS displays the medium, maximum and minimum values of roll, pitch, yaw and thrust channels.

Finally, the last mandatory step consists on setting the flight modes. In this project, the flight mode option was assigned to the fifth channel of the RC transmitter. Depending of the position of the fifth channel stick, the quadcopter will fly on the pre-selected flight modes. Only two modes were used in this project: stabilize and altitude hold mode. Therefore, there were only two optional positions assigned to the fifth channel stick.

Once completed the calibration process, the quadcopter should be ready to fly at first.



Figure 41: GCS calibration display

5.2 CAS Hardware Integration

The hardware integration of the collision avoidance system elements is defined in next subsections.

5.2.1 Arduino Mega 2560-Pixhawk flight controller

Figure 42 shows the physical wiring connections between Arduino Mega 2560 and Pixhawk flight controller. The communication type is serial, by UART. Table 4 details each pin connection.

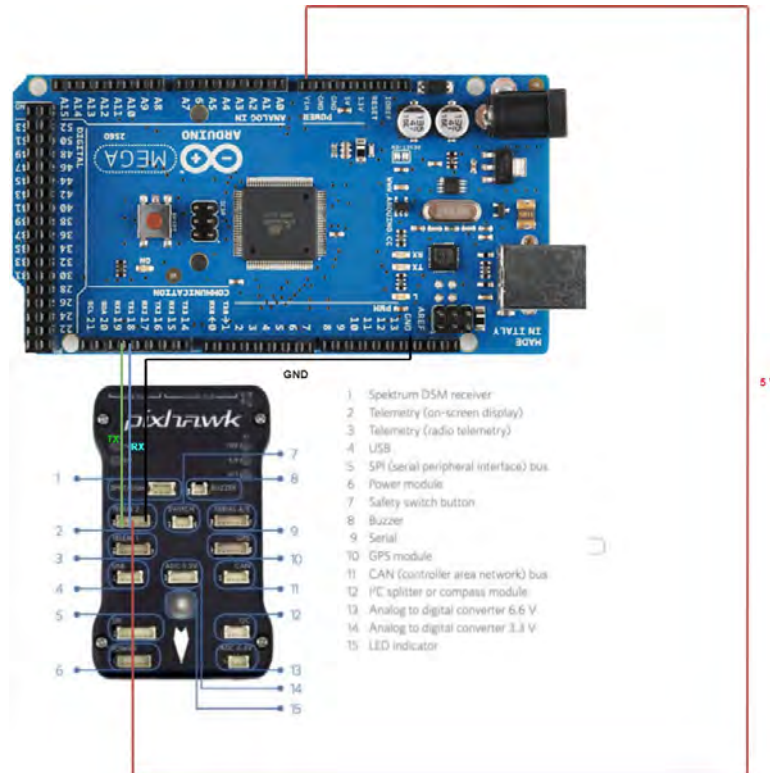


Figure 42: Arduino Mega 2560-Pixhawk flight controller pin connection

Table 4: Arduino Mega 2560-Pixhawk flight controller wiring connections

Connection Number	Type	Color	Pixhawk Pin	Arduino Mega 2560 Pin
1	Power	Red	5 V (OUT)	5 V (IN)
2	Ground	Black	GND	GND
3	Serial	Green	TX	RX
4	Serial	Cyan	RX	TX

In serial communications, the bit transmission is one-bit data at a time, and all the bit transmission is operated on one wire. The serial protocol of communication between the Pixhawk Flight Controller and the Arduino 2560 is UART. It is an asynchronous communication, without a clock line. In the Arduino Mega 2560, there exist 4 UART ports; in this project the serial 1 port was used, making use of the open-source Serial Library provided by arduino.

5.2.2 Arduino Mega 2560-Ultrasonic range finder HC-SR04

The HC-SR04 ultrasonic rangefinder works emitting a 40000 Hz ultrasonic wave that propagates through the medium until reaching an obstacle, where it 'bounces' back to receiver sensor. The sensor is compounded by four pins: VCC, Ground, Trig and Echo:

- **VCC:** this is the input voltage pin. It must be connected to the 5 volts pin of the arduino board.
- **GND:** this is the ground pin. This pin acts as the reference point in the electrical circuit from which the 5 voltage is measured. This pin must be connected to the available GND pins in the Arduino Board.
- **Trig pin:** this pin must be plugged to any digital pin of the arduino board. In order to send a pulse, the digital pin of the arduino must be trig on high state for at least 10 μ s. By this, it will send a 8 cycle burst that will come back after bouncing.
- **Echo Pin:** this pin must be connected to any digital pin of the arduino board. When the wave signals comes back; this pin will output in μ s the time the time difference between sending the pulse and receiving its reflection. By proper calculations as shown in section 2.2.2, the distance from the obstacle can be computed.

Taking into the account that the arduino board has 12 digital pins, it can be connected up to 6 sensors (2 pins per sensor) and be operative simultaneously. This chance can extend the obstacle collision avoidance problem to the six faces of the vehicle: four sides, bottom and top. In this project, five ultrasonic range finders were used. One of them was located in the bottom of the quadcopter, working as a minimum altitude clearance device. The other four remaining ones were located on each of the four sides of the quadcopter, working as horizontal environmental sensing devices.

Table 5: Arduino Mega 2560-HC SR04 pin connections

Connection Number	Type	Color	Arduino Mega 2560 Pin	HC-SR04 Pin
1	Power	Red	5 V	VCC
2	Ground	Black	GND	GND
3	Trig	Green	Any digital Pin	TRIG
4	Echo	Cyan	Any Digital Pin	ECHO

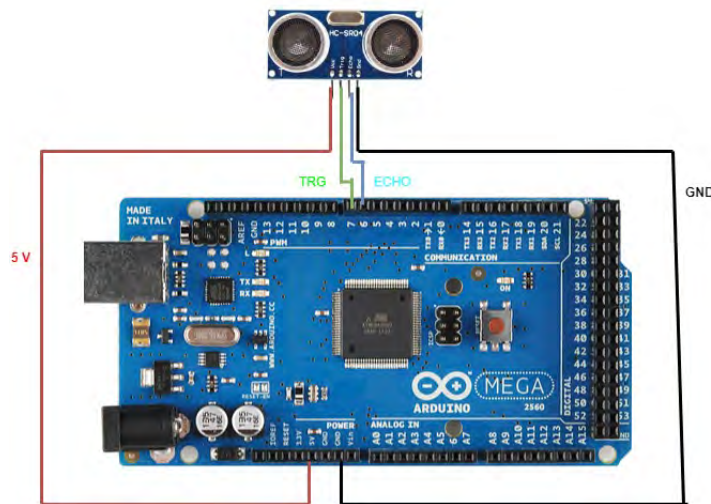


Figure 43: Arduino Mega 2560-HC-SR04 range finder pin connection

Table 5 details the pin connections between the Arduino board 2560 and the ultrasonic range finder.

5.2.3 F450 kit-Ultrasonic range finder HC-SR04

As already stated, five ultrasonic range finders were implemented into the quadcopter. They were all assembled to the power distribution board of the quadcopter as shown in figure 44. The four lateral ones were carefully taped to maintain a vertical alignment with the upward direction of the quadcopter. In that way, the ultrasonic waves travel following a parallel direction to the ground and perpendicular to the obstacles test devices employed within the testing phase. Perpendicular 'impact' of the ultrasonic waves against the obstacles ensures that the waves are going to travel back to the receiver. Oblique impact makes lose some of the sent pulses, coming back just a portion of the reflected beats.

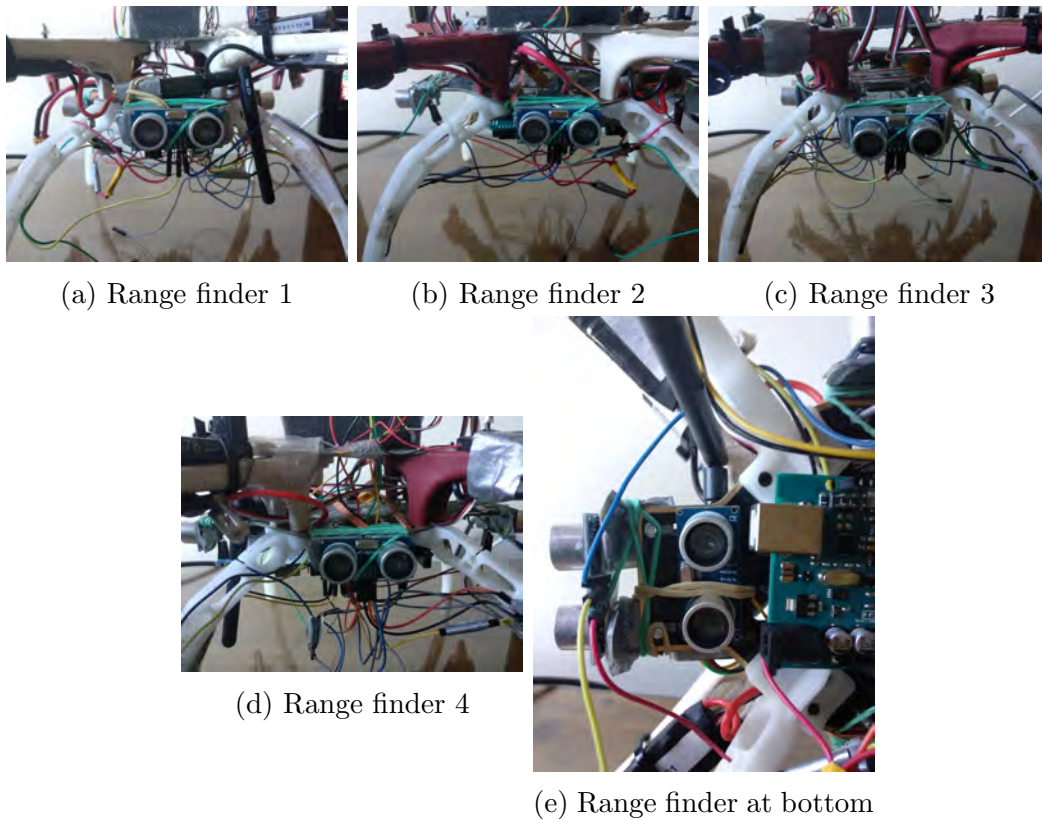


Figure 44: Lateral and bottom ultrasonic range finders

5.2.4 F450 kit-Arduino Mega 2560

The Arduino Mega 2560 microcontroller board was assembled to the bottom of the quadcopter, tapped to the power distribution board. That location gave many advantages in terms of the wiring tasks. During flight tests, some wires came off due mainly to collisions against the ground. Having the arduino microcontroller board located in the bottom of the MAV made the wiring arrangements an easy task, since they were feasibly accessible for manipulation.

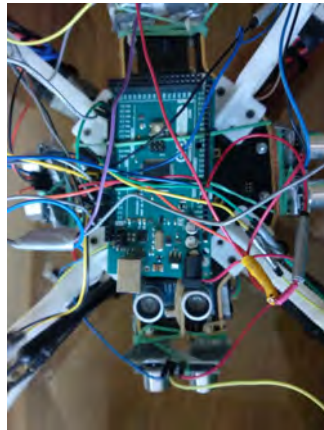


Figure 45: Arduino Mega 2560 assembled to the bottom of the power distribution board

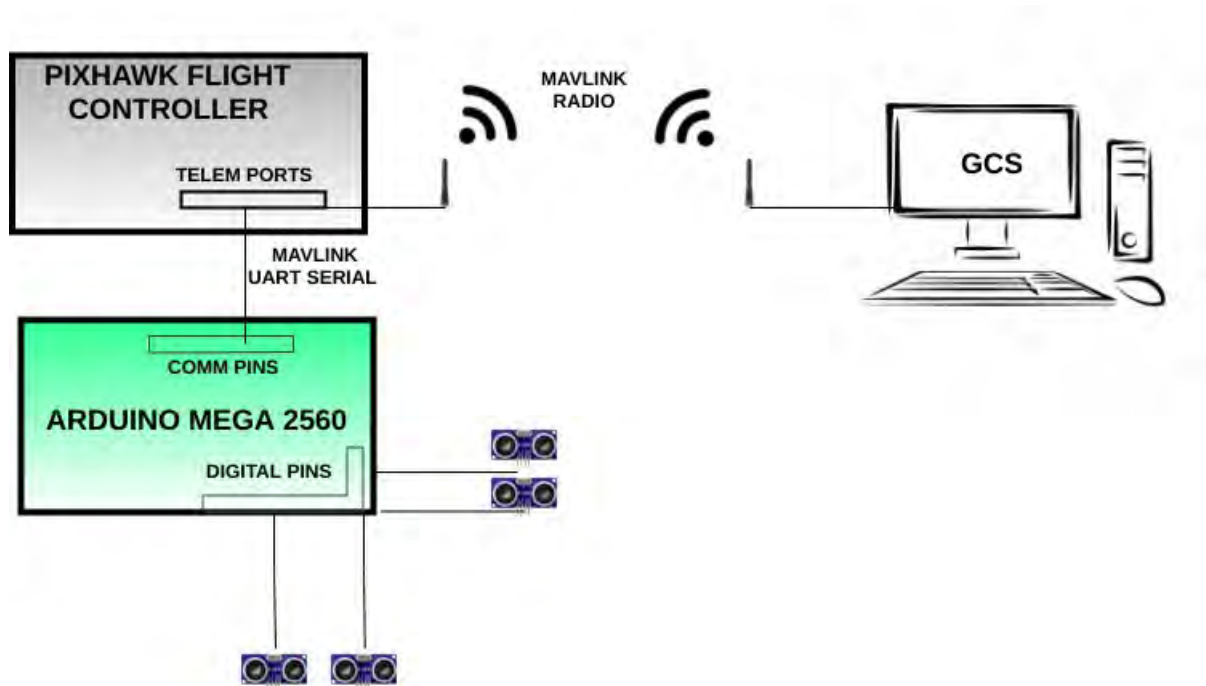


Figure 46: Collision avoidance hardware layout

Once all the hardware components are integrated into the quadcopter, the resulting prototype is illustrated in figure 47



Figure 47: Final prototype

5.3 CAS software integration

The title of this section represents undoubtedly one of the most challenging parts within this project. The key and fundamental duty to correctly implement the CAS into the UAV system lies on establishing a proper communication between the flight controller and the arduino microcontroller board. As previously mentioned, the pitch and roll correction orders are sent from the arduino board, based on the distance from the quadcopter to an object. As introduced in section 3.3.2, MAVLink protocol of communication is the tool utilized by those two devices to send and receive messages. Following sections will cover the software related tasks to make the CAS successfully operate in the quadcopter.

5.3.1 Distance data acquisition and processing

First task carried out was establishing communication between the ultrasonic range finders and the Arduino 2560 microcontroller. The HC-SR04 sensors emit a pulse and compute the time it takes to reflect and come back to the sensor receiver. This is the time output to the microcontroller board.

A program was created to process that time information and convert it to distance. For that purpose, arduino IDE was utilized. As mentioned in section 3.2.3, the code writing just entails two basic functions: setup and loop. Inside those functions, further and complex functions can be used, to be run just one time if inside the setup function; or rather being run all the time if included inside the loop function.

The distance function was included inside the loop of the arduino sketch; hence, pulses were constantly sent to monitor the environment.

As explained in section 4.2.1 the quadcopter needs to exceed a minimum altitude of 1 meter to start monitoring the horizontal surrounding environment. That minimum altitude level requisite is implemented as a mandatory condition to make the lateral ultrasonic range finders work.

The collision avoidance system was integrated into the four sides of the quadcopter. However, the arduino program was designed to just activate one of the four sides on each flight. In other words, four different programs were created depending on the desired side of the drone where testing the collision avoidance system.

5.3.2 Arduino board as an UAV system

The collision avoidance system works sending pitch and roll corrections to change the trajectory of the drone and ward it off from obstacles. Those pitch and roll corrections are overwritten to the ones imposed by the pilot through the RC transmitter. To achieve fortunate overwriting messages, the collision avoidance system must be firstly identified by the GCS as an additional UAV system. That identification is carried out by establishing communication between the arduino microcontroller and the Pixhawk flight controller. For that end, MAVLink protocol of communication is employed.

MAVLink has a wide range of messages to send and receive from GCS, flight controllers and additional microcontroller boards. For the identification of the Arduino Mega 2560, the heartbeat message is utilized. Heartbeat message purpose lies on identification of two subsystems and establishing proper link between them. This message must be sent from the arduino board to the Pixhawk flight controller constantly, to inform the flight controller that the collision avoidance subsystem is still alive. In case of not receiving the heartbeat pulses, the flight controller would initiate a fail-safe system and would inform the GCS about the failure of the collision avoidance system. Figure 48 and 49 show the proper identification of the collision avoidance as a subsystem and its failure when the communications are lost, respectively.

The heartbeat message has the byte structure introduced in table 2 on section 3.3.2. The Pixhawk flight controller must be informed about the start of the message, its longitude, the sequence number, the sending system ID, the sending component ID (the arduino board), the message ID (heartbeat message is enumerated as 0), the payload and the checksum. All this information is prepacked in a function run in the arduino board and sent through the serial1 port to the Pixhawk flight controller.



Figure 48: UAS linked to the GCS

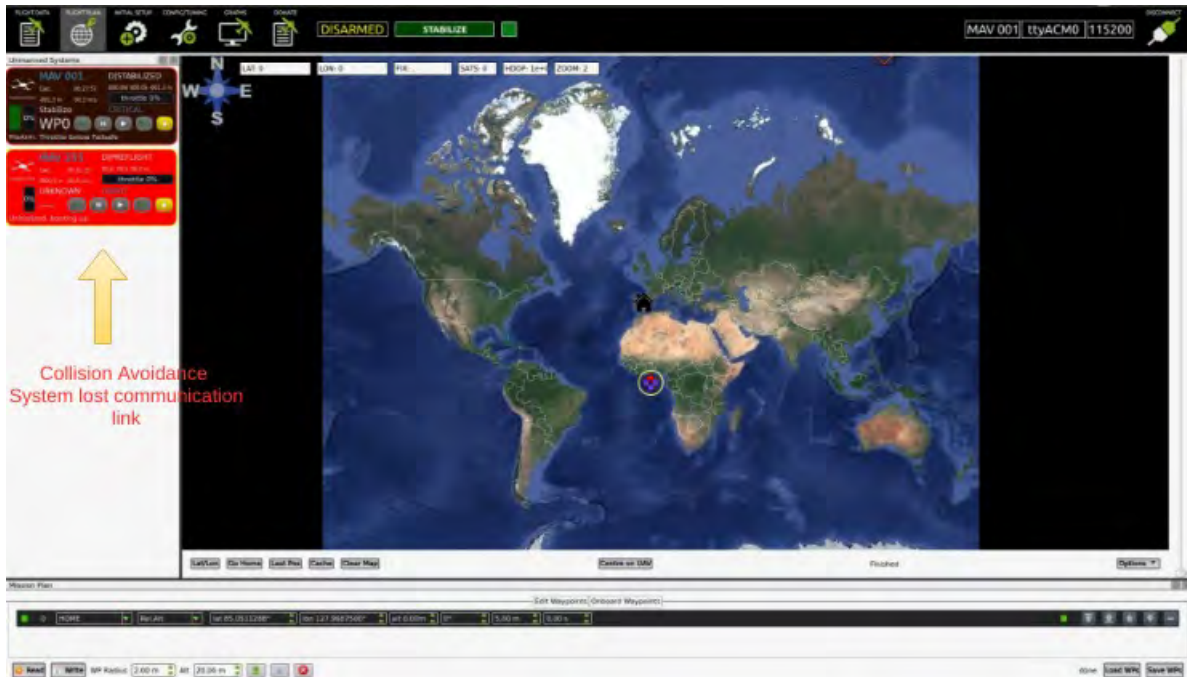


Figure 49: Collision avoidance system communication lost alert

5.3.3 Obstacle alert and collision avoidance

Four different programs were created to be uploaded inside the arduino microcontroller board, depending on the side of the quadcopter to be tested. On each of these four codes, the four stages of the collision avoidance system were implemented: minimum altitude activation level, horizontal environment sensing, obstacle detection and collision avoidance maneuver; following the logical sketch illustrated on figure 31.

The distance data acquisition and processing function is run repeatedly in the arduino board, and thus, the selected lateral sensor is constantly measuring the distance to a potential obstacle. When it detects a distance below 1.2 m, the program executes an order to override a computed pitch or roll value, depending on the side where the object is detected. Figure 50 illustrates the disposition of the four sensors around the quadcopter.

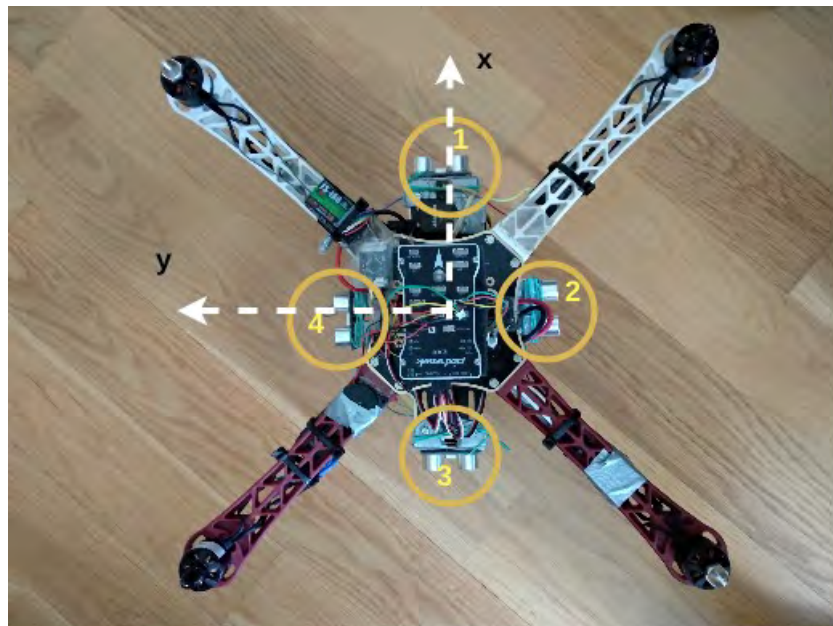


Figure 50: Ultrasonic range finders locations

When sensors 1 or 3 are active, the program outputs pitch values to incline the quadcopter and ward it off from the obstacle, according to figure 30. When sensors 2 or 4 are active, the program outputs roll values to incline the quadcopter and ward it off from the obstacle according to figure 28.

Roll and pitch values correspond to channels 1 and 2 of the radio transmitter. The radio receiver outputs to the PPM encoder 6 PWM signals, one for each channel. The PWM signals on each channel range between 1000 and 2000, and have a linear relation with the commanded variable that depends on them. For instance, assuming that the pitch angle has a minimum value of -45° and a maximum value of 45° , their

corresponding PWM signals would be 1000 and 2000 respectively. When a PWM signal is sent by the roll, pitch or roll channels, the PID controller of the flight controller demands a rotational angular velocity of the drone to achieve the required angle. The maximum reachable angles can be set through the GCS software, APM Planner 2 in this project. The maximum and minimum values for pitch and roll values are set as 45° and -45° .

The pitch and roll corrections are sent to the flight controller based on the distance at which the obstacle is. An obstacle detection distance is set in the arduino program, below which the collision avoidance system activates. That distance is 1.2 m. Below this distance, pitch and roll values are sent to the flight controller to induce a pitch or roll correction. But this correction is programmed to be proportional, depending on the safe distance below 1.2 meters, and therefore make a soft or aggressive maneuver based on the level of risk. Table 6 shows the induced PWM signals and its corresponding angle values, depending on the distance to the obstacle.

Table 6: PWM overridden values

Distance (cm)	Sensor 1		Sensor 2		Sensor 3		Sensor 4	
	PWM value	Pitch Value ($^\circ$)	PWM Value	Roll Value ($^\circ$)	PWM Value	Pitch Value ($^\circ$)	PWM Value	Roll Value ($^\circ$)
$70 < d < 120$	1600	9	1400	-9	1400	-9	1600	9
$40 < d < 70$	1675	15.75	1325	-15.75	1325	-15.75	1675	15.75
$d < 40$	1750	22.5	1250	-22.5	1250	-22.5	1750	22.5

6 Testing and Results

This section covers the experimental tests performed to prove the quadcopter flight capabilities, as well as the collision avoidance system performance.

For the quadcopter, the experimental tests consist basically on flight tests to prove the two utilized flight modes: stabilize and altitude hold.

For the collision avoidance system, a systematic approach is carried out, testing firstly the components separately, then the communications, and finally the system implemented in the MAV, by ground tests and flight tests too.

6.1 Flight capabilities tests and results

As introduced some lines above, the flight capabilities of the F450 quadcopter need to be tested in order to later test the collision avoidance system once implemented.

6.1.1 Stabilize flight mode

As already introduced in section 3.2.2, stabilize mode levels automatically the vehicle, based on IMU sensor readings, and the vehicle change its trajectory only under pilot desired inputs of roll, pitch, thrust and yaw.

Once the drone was totally calibrated by following the procedure covered in section 5.1, outside flights were carried out, in a private preserve located in Parla.

Initially, the drone's behavior was pretty worse than expected, since it was not even able to hover for some seconds.



Figure 51: Video footage frame of stabilize mode test

This issue was solved by checking the PID parameters, which can be adjusted in the ground control station. The PID controllers are the mechanism employed inside

the Pixhawk flight controller to constantly calculate the error value as the difference between the desired output variables and the sensor-measured ones. Based on that difference, the controllers apply rectifications based on proportional, integral and derivative terms. Figure 52 illustrates the recommended PID parameter values by the official Ardupilot webpage [41].



Figure 52: PID recommended parameters [41]

Once those values were corrected, the quadcopter flight performance was increasingly improved. Several flight tests were performed and recorded, to check its stability. Notice that stability during flight was a major concern to prove in flight the collision avoidance capabilities. The ultrasonic range finders need a relative stable hover to send and receive pulses in a proper manner, and output time elapsed data to the arduino microcontroller board.

6.1.2 Altitude hold mode

As already mentioned in section 3.2.2, altitude hold flight mode operates in the same way as stabilize flight mode for the Euler angles control. The main difference lies on that throttle is automatically adjusted by the flight controller to maintain a fixed altitude.

This flight mode was tried and tested in the F450 quadcopter. The main reason was having an alternative flight mode on which proving the collision avoidance system. In section 4.2.4, the collision avoidance maneuver stage was explained. A main problem is caused when the vehicle pitches or rolls. The lift force is no longer aligned with the z-axis and the drone falls down. When altitude mode is activated, the drone should

maintain its altitude, independently of the commanded pitch or roll maneuvers. In that way, the collision avoidance maneuver could be performed at a fixed altitude.

First of all, altitude hold mode was tested without the collision avoidance system integrated. Overall, the results were good, in spite of some usual problems.

As mentioned, the drone maintains the altitude based on pressure readings from the barometer installed inside the Pixhawk flight controller. This has a major and common drawback. When the flight controller is directly exposed to propeller wash, the internal barometer is subjected to pressure changes that confuse the flight controller and make it oscillate in a range between one and two meters. This issue could be solved by locating the Pixhawk flight controller between the power distribution board plate and the upper link plate. In that way, the propeller wash effect does not affect directly to the flight controller since it would be protected. That solution was tried but a new issue appeared. The separation between the upper and bottom plates is not too wide, and therefore, the flight controller was pretty tightly tapped. This caused a major transmission of vibrations to the flight controller during flying tests. Vibrations in the flight controller are not desirable at all. They cause the drone to fly with lots of vibration and shaking unpredictable movements, that make it manually uncontrollable.

A solution to successfully operate altitude hold flight mode in the quadcopter built throughout this project is proposed. The flight controller should remain on its current position, in the top of the upper link together with a cushioning sponge to absorb the vibrations. To wipe out the propeller wash effect, a small case could be printed to be adapted on the F450 kit. That would solve the propeller wash effect and at the same time would prevent the flight controller from potential collisions, since it would cover it completely. That solution lies on future work to be developed on this quadcopter.

6.2 Component testing and results

The components that lay the obstacle collision avoidance must be initially tested isolated, to check its functionality before integrating them into the rest of the quadcopter.

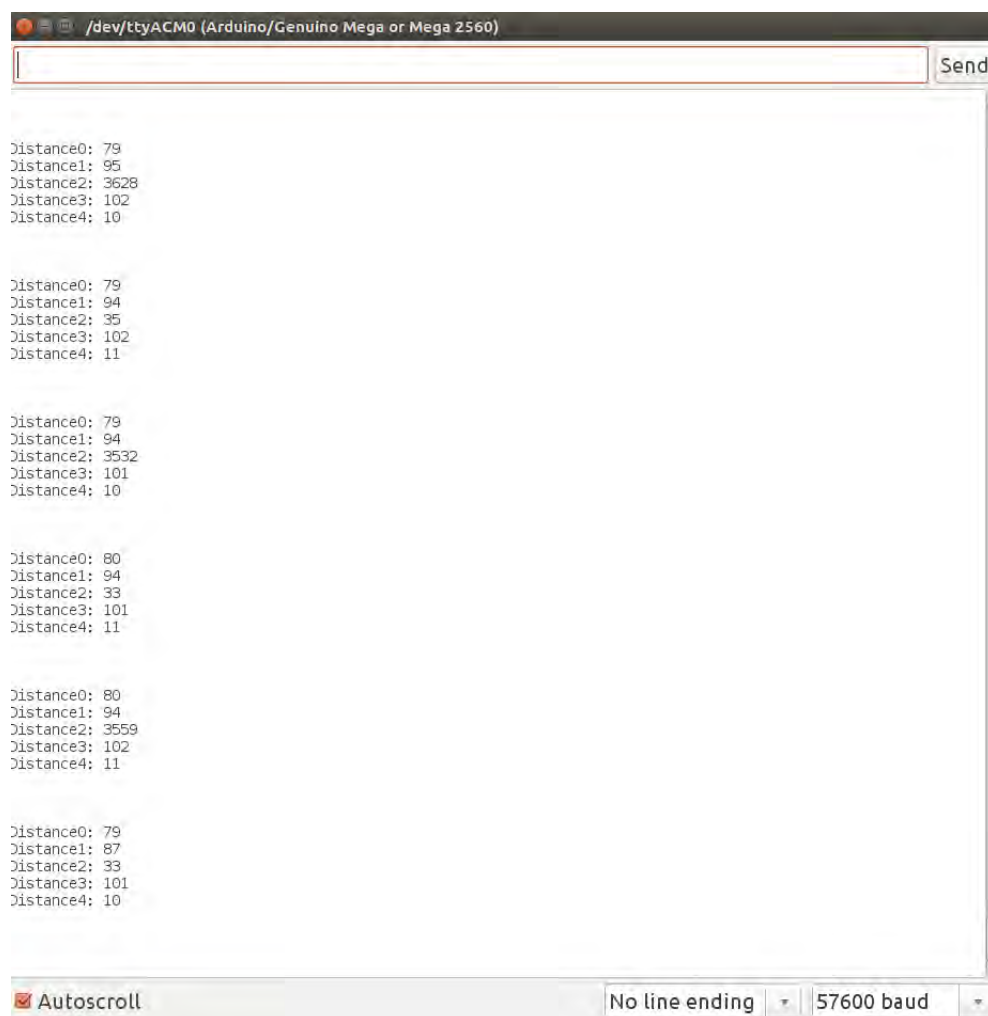
6.2.1 HC-SR04 Ultrasonic range finders

Five ultrasonic range finders were assembled to the quadcopter. The selected criteria of the range finders was based mainly on its price. HC-SR04 is the cheapest range finder in the market. Notice that one relevant requirement and limitation on the development of this project lies on the budget.

Firstly, the range finders were separately tested plugged to the Arduino Mega 2560 microcontroller board following the wiring sketch illustrated in figure 45. An arduino

program was created to output the measured distances on the serial monitor display. It is worth mentioning that the quality of the ultrasonic sensors has been a relevant constraint on the performance of the collision avoidance system. Some of the sensors did not even display distances when separately tested. Having taken this detail into consideration, more than 10 HC-SR04 were acquired, to just choose the ones with the best performances.

The test methodology of the ultrasonic range finders is simple. The serial arduino IDE monitor prompts the measured distances by the sensors. This task is implemented in the code by using the `Serial.write()` function. A rectangular wood veneer was employed as obstacle and was gradually ward off and draw near the sensors to check proper distance acquisition response in the serial monitor display. Figure 53 shows the prompted distances in cm in the arduino IDE serial monitor display.



The screenshot shows a serial monitor window titled "/dev/ttyACM0 (Arduino/Genuino Mega or Mega 2560)". The window contains a text input field at the top with a "Send" button. Below the input field, the serial monitor displays five groups of distance readings, each group containing five lines of data. The data is as follows:

```
Distance0: 79
Distance1: 95
Distance2: 3628
Distance3: 102
Distance4: 10

Distance0: 79
Distance1: 94
Distance2: 35
Distance3: 102
Distance4: 11

Distance0: 79
Distance1: 94
Distance2: 3532
Distance3: 101
Distance4: 10

Distance0: 80
Distance1: 94
Distance2: 33
Distance3: 101
Distance4: 11

Distance0: 80
Distance1: 94
Distance2: 3559
Distance3: 102
Distance4: 11

Distance0: 79
Distance1: 87
Distance2: 33
Distance3: 101
Distance4: 10
```

At the bottom of the window, there is a status bar with the following settings: Autoscroll, No line ending, and 57600 baud.

Figure 53: Prompted distances (cm) from the 5 ultrasonic range finders

6.2.2 Arduino Mega 2560

Arduino Mega 2560 micro controller board was implemented into the MAV system following the flight controller wiring scheme illustrated in figure 42. Proper powering was checked by repeating the HC-SR04 range finders tests when they were already assembled to the quadcopter. This test was just carried out to ensure proper power supply to all the collision avoidance system components.

6.3 MAVLink communication testing and results

A program was created to constantly send heartbeat messages from the Arduino Mega 2560 to the Pixhawk flight controller. The utility of this MAVLink messages lies on constantly monitoring the communication between arduino microcontroller board and Pixhawk flight controller. To check whether the link is established and the Arduino Mega 2560 is identified by the ground control station as an additional UAS system, the UAS systems monitor display in the APM Planner 2 GCS software is consulted. In case of proper communication link, the UAS display should remain as illustrated in figure 48.

Some difficulties were faced prior to getting successful communication. To send the Heartbeat message, a function called `mavlink_msg_heartbeat_pack()` must be employed. That function belongs to mavlink libraries and their inputs are given in table 2. Specially, System ID, Component ID and Message ID inputs are intricate to know. They have a predefined enumeration based on the specific components that form the UAS. That enumeration depends mainly on the hardware components being used, and sometimes gets tough to know if proper values are being input, specially when no communication seems to be established and nothing alerts about the reason.

6.4 Obstacle collision avoidance ground tests

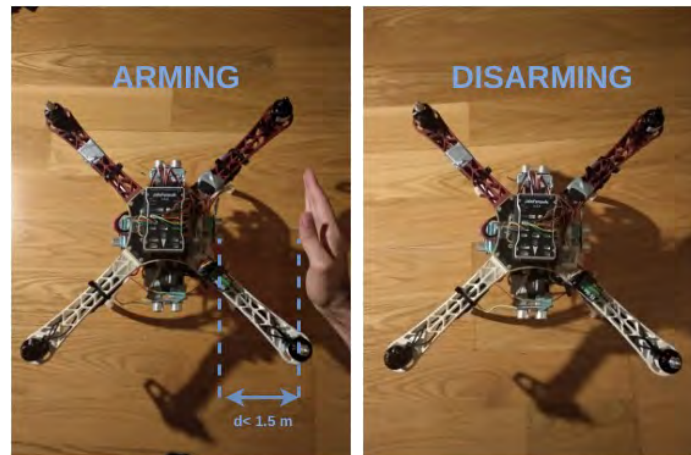
Some ground tests were performed to prove the effectiveness of the obstacle collision avoidance system. In that way, flight performance can be estimated and the collision avoidance system is not proved for the first time on flight tests, which could lead to failure and dramatic consequences for the physical integrity of the quadcopter.

6.4.1 Arming and disarming test

A sketch program to be run in the arduino board was created to arm and disarm the vehicle each time one of the lateral sensors detect an obstacle below the range of 1.2 meters. Based on the processed distance for a selected lateral sensor, the arduino

board send arming and disarming messages to the Pixhawk flight controller through MAVLink protocol of communication.

Successful results were obtained. The test procedure just consisted on putting and removing the hand from the sending direction of the activated ultrasonic range finder in the arduino code. The rotors started rotating each time the hand was located below a distance of 1.2 meters. This test was carried out without propellers, to guarantee the safety of the user in every moment.



(a) Arming the drone (b) Disarming the drone

Figure 54: Arming and disarming test

After completing this test, a principal task was proved: MAVLink message commands sent based on distance information. This is the key of the sensing and collision avoidance algorithm developed on this project. Although this test was much simpler than collision avoidance commands testing, a significant achievement was considered to be got, since integration of the collision avoidance hardware and communication with the flight controller was achieved.

6.4.2 Collision avoidance system ground test

The main collision avoidance arduino code was tested in ground prior to flight test. Notice that the flight can be simulated holding the vehicle and moving it manually inside a room, warding it off and drawing it near to the walls. The distance data is received by the sensors and processed inside the arduino microcontroller board, that send pitch and roll corrections to the flight controller. During those tests, the drone is disarmed, and it is impossible to visually check whether the test is having a result or not. However, there is a possibility to check if pitch and roll corrections are being output by the arduino microcontroller board. It is by monitoring the pitch and roll corrections in the serial monitor functionality of the arduino IDE. In order to do that, the arduino must be connected to the computer by USB, thus the drone

movement is limited by the longitude of the wire. Nevertheless, there is enough freedom of movement to place the drone at several distances from a wall and check whether pitch and roll commands are being sent.

During this test, the user must always hold the drone over an altitude above the ground of at least one meter. Otherwise, the horizontal environment sensing can not be activated, and no pitch or roll outputs will be prompted in the serial monitor.

The four codes corresponding to the four sides of the drone were run inside the arduino and successful results were obtained. Consistent PWM values corresponding to pitch and roll channels were prompted in the serial monitor of the arduino, according to the distances at which the drone was placed from a wall. When the drone was placed below an altitude of 1 meter with respect to the ground, no PWM signals to pitch and roll channels were output, hence the minimum altitude activation level was proved to be working as well. Figure 55 shows the serial monitor of the arduino IDE with some PWM signals sent to the Pixhawk flight controller. Notice that the PWM-distance-angle correlation was already included in table 6, section 5.3.3 .

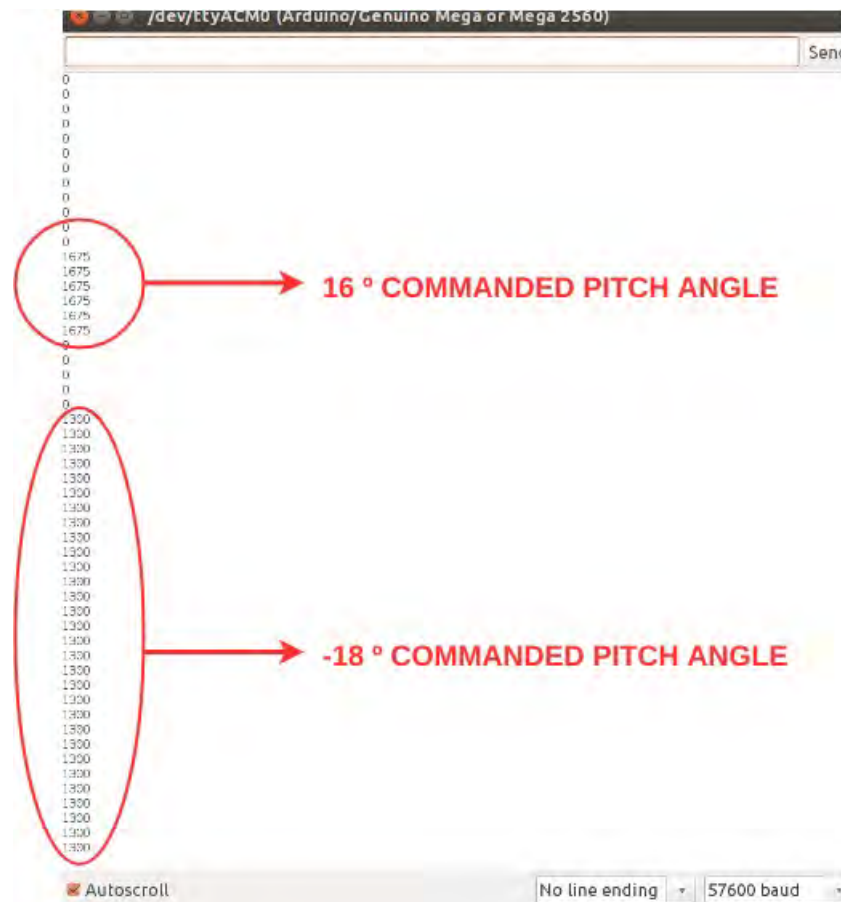


Figure 55: Prompted PWM commands

When the drone is armed, the PWM output values should have proper response

in the pitch and roll state of the drone. Figures 56, 57, 58 and 59 illustrate the expected response for each of the commanded PWM on each sensor. That response was tested during real flight, as explained in section 6.5.

- **Front sensor 1:**

1. 1600 PWM Value: 9° pitch angle, when obstacle distance is between 70 and 120 cm.
2. 1675 PWM Value: 15.75° pitch angle, when obstacle distance is between 40 and 70 cm.
3. 1750 PWM Value: 22.5° pitch angle, when obstacle distance is below 40 cm.

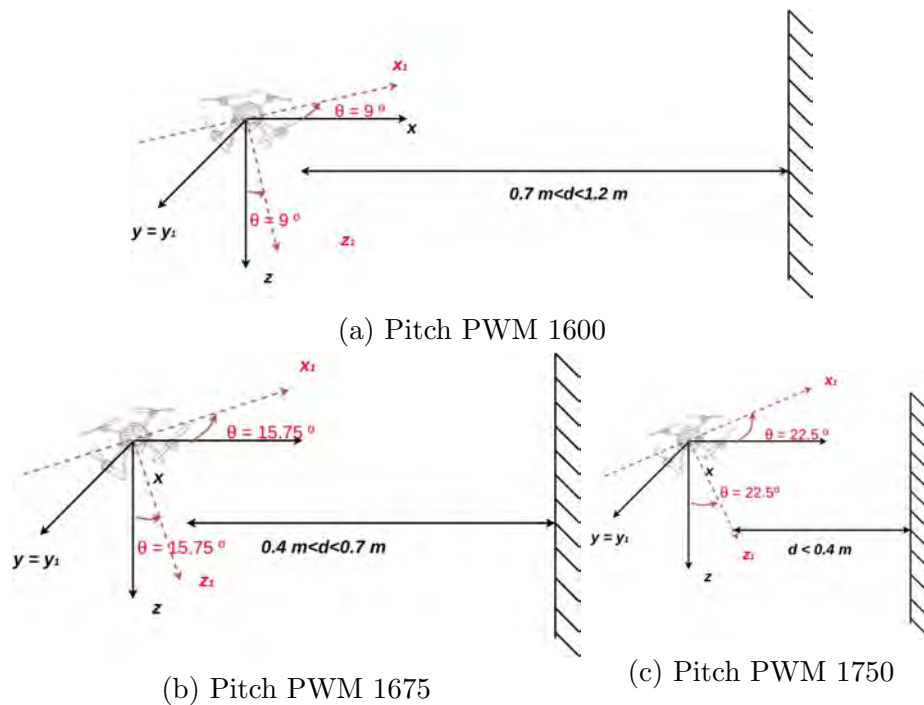


Figure 56: Drone response to pitch PWM outputs based on distances to front sensor 1

- **Back sensor 3:**

1. 1400 PWM Value: -9° pitch angle, when obstacle distance is between 70 and 120 cm.
2. 1325 PWM Value: -15.75° pitch angle, when obstacle distance is between 40 and 70 cm.
3. 1250 PWM Value: -22.5° pitch angle, when obstacle distance is below 40 cm.

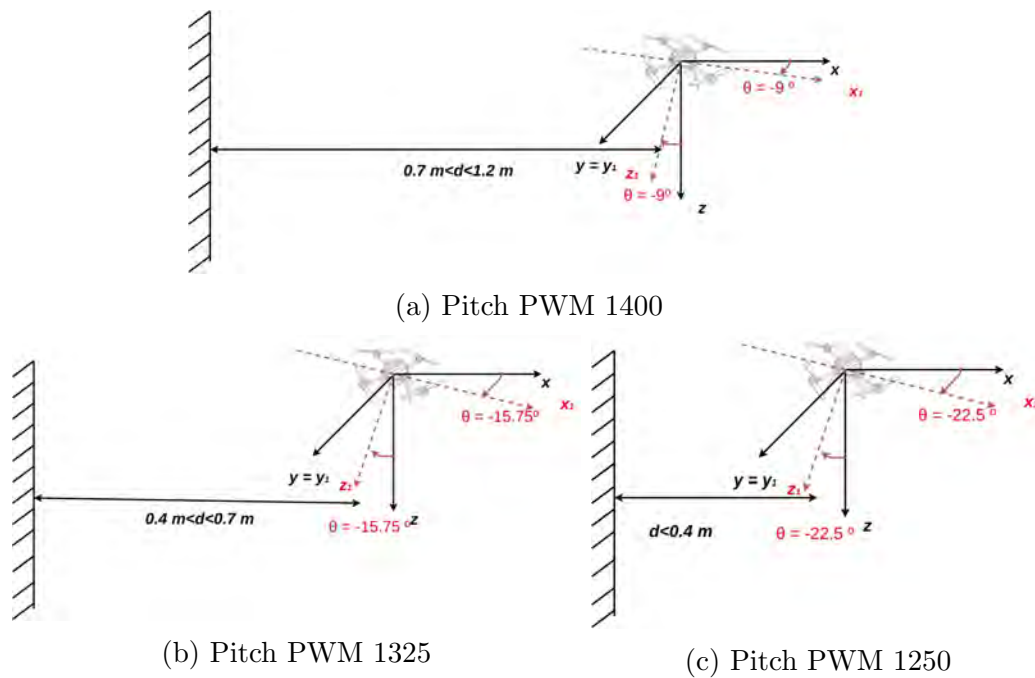


Figure 57: Drone response to pitch PWM outputs based on distances to back sensor
3

- **Right sensor 2:**

1. 1400 PWM Value: -9° roll angle, when obstacle distance is between 70 and 120 cm.
2. 1325 PWM Value: -15.75° roll angle, when obstacle distance is between 40 and 70 cm.
3. 1250 PWM Value: -22.5° roll angle, when obstacle distance is below 40 cm.

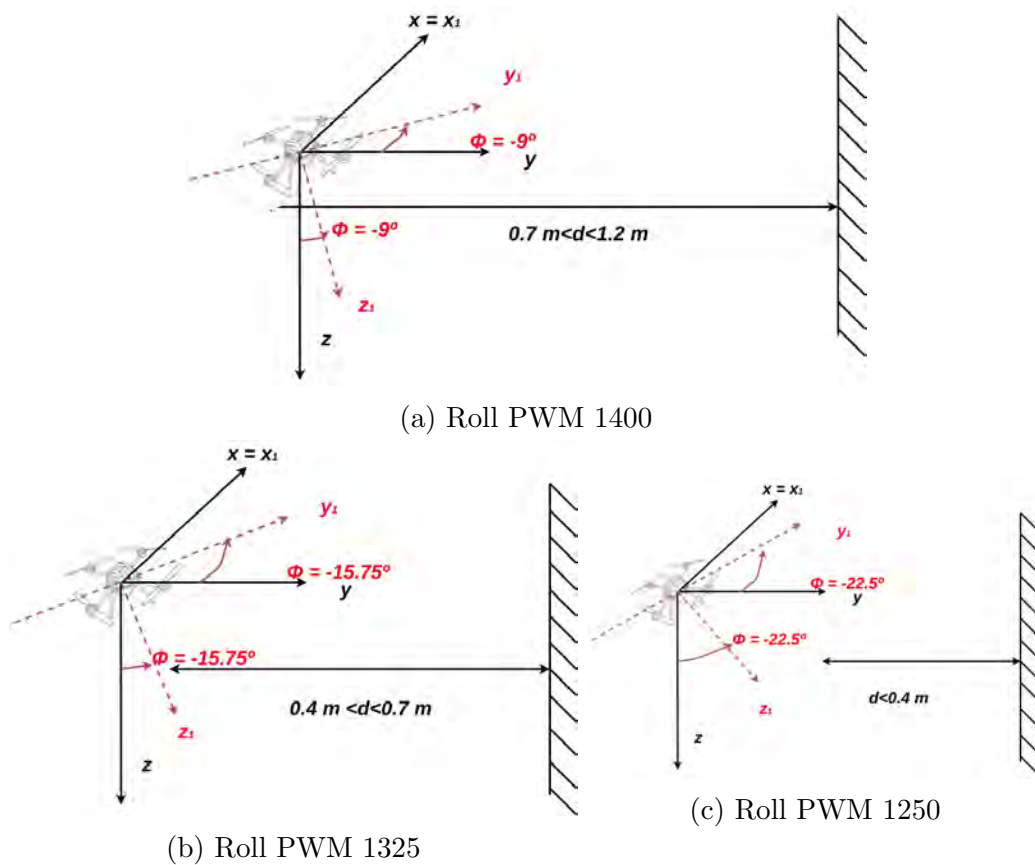


Figure 58: Drone response to roll PWM outputs based on distances to right sensor 2

- **Left sensor 4:**

1. 1600 PWM Value: 9° roll angle, when obstacle distance is between 70 and 120 cm.
2. 1675 PWM Value: 15.75° roll angle, when obstacle distance is between 40 and 70 cm.
3. 1750 PWM Value: 22.5° roll angle, when obstacle distance is below 40 cm.

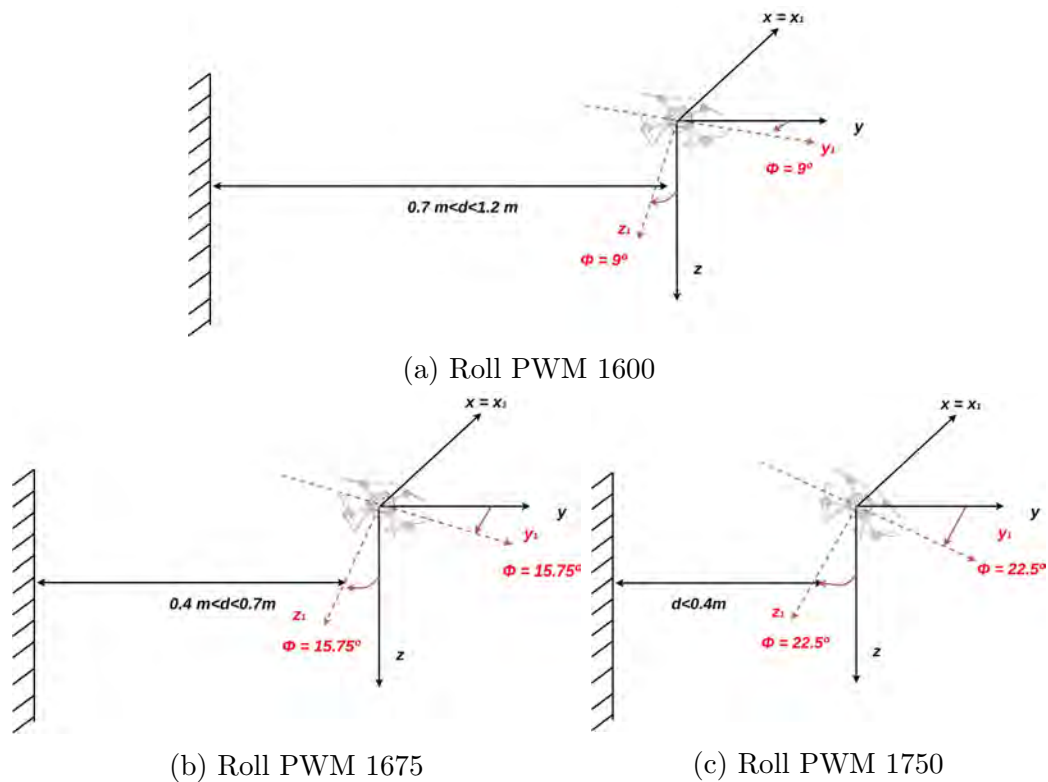


Figure 59: Drone response to roll PWM outputs based on distances to left sensor 4

6.5 Obstacle collision avoidance flight tests

Once the collision avoidance performance was proved on ground, it was time to try during flight. Several flight tests were performed, in two different flight modes: stabilize flight mode and altitude hold flight mode.

6.5.1 Stabilize flight mode test

First flight test consisted on flying the drone in stabilize mode and proving the collision avoidance performance on each of its four sides. Therefore, four different flights were performed, running their corresponding sensor code inside the arduino board, depending on the desired side of collision avoidance to be tested.

In order to do this test, two people are needed. One person is in charge of piloting the drone and the other person is in charge of placing an obstacle in the lateral side to be tested. The object used as obstacle was a cardboard plate. When the drone is considered to be hovering in a stable way, the cardboard plate is approximated to one of its sides. When the distance between the sensor and the obstacle is below 1.2 meters, the drone should perform the collision avoidance maneuver, almost with no delay of reaction. When the drone has warded off from the cardboard plate for a safe

distance higher than 1.2 meter, the drone should stabilize again and the pitch/roll control should return back to the pilot.

Results obtained during this flight test had a major issue. The collision avoidance actuated and the drone warded off from the obstacles. However, after performing the collision avoidance, sometimes, the drone fell to the ground with almost no time of reaction for the pilot to adjust the throttle and maintain an stable hover.

The collision avoidance concept proof was achieved, but stability after avoidance maneuver was a relevant issue to be solved in order to fulfill the initial requirements established for the object collision avoidance system.

6.5.2 Altitude hold flight mode test

To solve the stability issue of the drone after collision avoidance, altitude hold flight mode tests were done. That could prevent the drone from falling to the ground by fixing an altitude to hover during the test.

Same tests as in stabilize mode were performed: one flight for each of the four lateral sides to be proved.

Results were apparently good. The drone did not fall to the ground after avoiding the collision against the obstacle, since a fixed altitude was commanded to be maintained based on pressure readings from the barometer inside the flight controller. The drone performed the collision avoidance maneuver in the four sides letting the pilot continue with a stable hover once the obstacle was out of the margin distance of 1.2 meters.

There was only one issue regarding to this flight mode. The quadcopter tends to oscillate in a range of 1 to 2 meters when flying in altitude hold mode. The altitude should be completely fixed, but propeller's wash effect changes the pressure reading of the barometer that gets confused and believes that the quadcopter is changing its altitude. A solution as a future work is proposed, and lies on covering and protecting the Pixhawk flight controller with a case.



Figure 60: Collision avoidance flight test video footage frame

7 Summary, Conclusions and Future work

During this project, a preliminary design of a collision avoidance system for unmanned aerial vehicles with Pixhawk flight controller has been developed, implemented into a real quadcopter, and tested in realistic scenarios. Five ultrasonic range finder sensors were integrated into a F450 quadcopter, with the purpose of monitoring the horizontal environment and avoiding potential collisions with objects placed on its surroundings.

7.1 Project summary

The most significant objective of the project lied on the design and integration of the collision avoidance functionality as an additional subsystem to improve the safety in flight operations of unmanned aerial vehicles equipped with Pixhawk flight controller. This task implied the use of many hardware components and software elements in order to have a complete and reliable system to satisfy the initial objectives pretended to be fulfilled.

A set of requirements were established as target functionality, integration, performance, usability, interface, operational, states, physical, design, certification and cost preliminary conditions that orientated the development of the collision avoidance subsystem from hardware and software selection to the design and integration phases.

A collision avoidance algorithm was designed to follow four well differentiated stages from stable hover state to successful collision elusion. Firstly, a minimum altitude activation layer was introduced into the solution, consisting on constant monitoring of the altitude until a minimum height of 1 meter is reached and the subsequent horizontal environment monitoring stage starts. During the horizontal environment monitoring stage, the ultrasonic range finder sensors located in the four lateral sides of the quadcopter trigger constantly ultrasonic waves in order to obtain raw distance information of the objects placed around the vehicle. That raw data is processed in an auxiliary arduino microcontroller board. Next stage is the obstacle detection, and occurs when an processed distance is lower than a safe margin of 1.2 meters. Last stage is the collision avoidance maneuver and proceeds the obstacle detection stage with negligible margin of delay. When a distance is sensed to be under 1.2 meters, the arduino auxiliary board triggers pitch and roll corrections to the Pixhawk flight controller through MAVLink protocol of communication. Those corrections make the quadcopter ward off from the potential risky obstacle. Once the distance to the object is over 1.2 meters, the control of the pitch/roll angles of the vehicle returns to the pilot.

To implement the collision avoidance subsystem, it was firstly necessary to have a testing quadcopter. Therefore, a F450 flight capable quadcopter was built from

scratch and calibrated, achieving proper flight performances with stable hover capabilities. Afterwards, the collision avoidance subsystem integration was carried out, firstly by hardware implementation of the components and later by software integration; achieving proper communications among all the subsystems of the drone and specially, between the arduino microcontroller board and the Pixhawk flight controller. Arduino programs were developed to process the distance acquired by the ultrasonic range finder sensors and based on them; send trajectory corrections to the Pixhawk flight controller.

The testing phase was carried out following a systematic approach. Initially, the quadcopter testing platform was tested without the collision avoidance system implemented. Several flights were performed to prove its proper flight capabilities. Subsequently, the ultrasonic range finder sensors were tested connected to the arduino microcontroller board as an isolated system in order to check proper distance measurements. After that, the collision avoidance was tested being implemented into the quadcopter by ground tests. Successful trajectory corrections commands were monitored by the arduino IDE monitor display, and acceptable results were obtained to proceed to the final and main test phase: flight tests. The flight tests with the collision avoidance system integrated were carried out in outside environment scenario. Obstacles were simulated by an auxiliary person holding a cardboard plate and placing it in front of the lateral sides of the drone. Acceptable results were obtained to verify that the proof of concept of the collision avoidance system was working. The drone responded warding off from the obstacles and performing the pitch and roll correction maneuvers, in spite of some stability difficulties when returning the control to the pilot and ultrasonic range finder quality limitations that make some attempts not to succeed.

7.2 Extracted conclusions

Overall, the following isolated conclusions can be extracted:

- Current state of the art and the existing technology resources makes the collision avoidance being an implementable technology that can be integrated into unmanned aerial vehicles. Programming skills are needed to implement the avoidance algorithm, but in this project it has been demonstrated that basic aptitudes on arduino programming language are enough to test a proof of concept of collision avoidance.
- MAVLink protocol of communication is a huge and useful tool to implement any kind of capability to drones, prior compatibility of hardware components. Useful functionalities can be implemented to improve performance of drones in applications such as search and rescue, inspections and security.
- Hardware capabilities can seriously limit the performance of unmanned aerial vehicle systems. In this project, the HC-SR04 ultrasonic range finder sensors

were selected due to its cheap price. They had significant limitations in terms of precision in the measurements that make some flight tests not to succeed.

- Communications among subsystems is a tough and critical task for the proper development of capabilities in unmanned aerial vehicle systems. In this project, MAVLink communication between the arduino microcontroller board and the Pixhawk flight controller became difficult. MAVLink is a protocol of communication with the purpose of information exchange between unmanned aerial vehicle systems, which is constantly being updated and the libraries on which it is based are constantly being developed by users. However, there are not many teaching resources available to acquire a basis from which start developing. Further advances need to be done in this aspect.
- Hardware compatibility is also a critical aspect that needs to be checked prior to hardware selection for unmanned aerial vehicles. In this project, the compatibility of communications between the arduino microcontroller, the ultrasonic range finder sensors and the Pixhawk flight controller was checked prior to start working on it.
- Systematic approach in the test phase is necessary in order to achieve results. Specially for drones, where failures could deteriorate the physical integrity of a testing platform, it is very important to perform ground-based tests with the aim to ensure that communications and subsystems integration are ready.
- Overall, during this project, a baseline of work was established on which developing further improvements and ideas with the aim to get a more reliable and safe collision avoidance system.

7.3 Future work

During the development of this project, only a proof of concept of a preliminary collision avoidance system was tested. There are still many software and hardware related improvements and ideas that could be implemented. Some of them are here delineated:

- Two flight modes were proved during this project: stabilize flight mode and altitude hold flight mode. Altitude flight mode would be a much more efficient mode on which testing the collision avoidance system, since fixing an altitude prevents the drone from falling to the ground when pitch and roll corrections are overwritten. In the test phase, altitude hold flight mode gave a relevant issue. Propeller wash effect confused the pressure readings of the flight controller barometer, making it oscillate within a range between 1 and 2 meters. A further solution would lie on printing a case to cover the flight controller and protect it from this effect.
- The collision avoidance algorithm was just developed to be tested on one lat-

eral side on each flight test. Further arduino programming development could let proving flight tests with the four lateral sensors working at the same time. In that way, more than one obstacle could be placed and more complex trajectory corrections would need to be developed. By doing this, a 360°horizontal collision avoidance proof of concept would be achieved.

- The ultrasonic range finders can be assembled to the plates of the drone in a more efficient way. They are basically taped with elastic bands. This caused undesirable vibrations during flight that could lead to failure in distance readings.
- Hardware improvements could be implemented into the system. Higher performance ultrasonic range finder sensors acquisition would lead to more successful collision avoidance performances.
- The collision avoidance maneuver stage could be further improved. More precise angle computations could be done, based on the distance to obstacles. With the current avoidance maneuver arduino program, the drone usually responds with very aggressive pitch/roll maneuvers that make the vehicle destabilize and sometimes fall to the ground. Softer pitch/roll corrections would maintain the drone on its hovering position, making the control-return to the pilot a feasible task.

References

- [1] Andrew R Lacher, David R Maroney, Andrew D Zeitlin, et al. “Unmanned aircraft collision avoidance–technology assessment and evaluation methods”. In: *FAA EUROCONTROL ATM R&D Symposium, Barcelona, Spain*. 2007.
- [2] Javier Lloret. *Introduction to Air Navigation*. Create Space, 2016.
- [3] Bijan V Darryl J. “The economic impact of unmanned aircraft systems integration in the USA”. In: 2013.
- [4] Dane Bamburly. “Drones: Designed for product delivery”. In: *Design Management Review* 26.1 (2015), pp. 40–48.
- [5] *Real Decreto 1036/2017*. Ministerio de la presidencia y para las administraciones territoriales, Dec. 2017.
- [6] *Salario: Ingeniero Junior en España*. URL: <http://espana.jobtonic.es/salary/26526/74925.html> (visited on 06/02/2018).
- [7] *Unmanned Aircraft Systems*. International Civil Aviation Organization, Feb. 2016.
- [8] M Hassanalian and A Abdelkefi. “Classifications, applications, and design challenges of drones: a review”. In: *Progress in Aerospace Sciences* 91 (2017), pp. 99–131.
- [9] Gonzalo Pajares. “Overview and current status of remote sensing applications based on unmanned aerial vehicles (UAVs)”. In: *Photogrammetric Engineering & Remote Sensing* 81.4 (2015), pp. 281–329.
- [10] *How does LiDAR work? The science behind the technology*. URL: <http://www.lidar-uk.com/how-lidar-works/> (visited on 06/03/2018).
- [11] *HDL-32E lidar*. URL: <http://velodynelidar.com/hdl-32e.html> (visited on 03/02/2018).
- [12] *LeddarOne Single-Element Sensor Module*. URL: <https://leddartech.com/modules/leddarone/> (visited on 03/02/2018).
- [13] *Sonar working principle*. URL: https://www.zigya.com/study/book?class=9&board=cbse&subject=science&book=science&chapter=sound&q_type=&q_topic=&q_category=&question_id=SCEN9016176 (visited on 04/02/2018).
- [14] *MB1010 LV-MaxSonar-EZ1*. URL: https://www.maxbotix.com/Ultrasonic_Sensors/MB1010.htm (visited on 03/02/2018).
- [15] Samuel S Blackman. “Multiple-target tracking with radar applications”. In: *Dedham, MA, Artech House, Inc., 1986, 463 p.* (1986).
- [16] *PRINCIPALES APLICACIONES DE SISTEMAS DE RADAR*. URL: <http://elradar.50webs.com/aplicaciones.htm> (visited on 02/03/2018).
- [17] *Copter Object Avoidance*. URL: <http://ardupilot.org/dev/docs/code-overview-object-avoidance.html> (visited on 02/03/2018).
- [18] *LightWare SF40/C 100m Laser Scanner 360 SLAM*. URL: <https://www.3dxr.co.uk/product/lightware-sf40c-laser-scanner/> (visited on 02/03/2018).

-
- [19] *TeraRanger Tower*. URL: <http://www.teraranger.com/product/teraranger-tower/> (visited on 02/03/2018).
- [20] *Frame Class Configuration*. URL: <http://ardupilot.org/copter/docs/frame-type-configuration.html> (visited on 02/04/2018).
- [21] *BLDC motor basics*. URL: <https://www.renesas.com/en-eu/support/technical-resources/engineer-school/brushless-dc-motor-01-overview.html> (visited on 03/04/2018).
- [22] *Constant Speed Propellers*. URL: <http://www.dauntless-soft.com/PRODUCTS/Freebies/Library/books/FLT/Chapter2/Constant.htm> (visited on 04/04/2018).
- [23] *Helice Gemfan fibra de carbono 10x4.5 (pareja) Estilo APC SF*. URL: <http://www.dauntless-soft.com/PRODUCTS/Freebies/Library/books/FLT/Chapter2/Constant.htm><https://rc-innovations.es/Helices-de-carbono-aviones-rc-drones-gemfan/helice-fibra-carbono-10x4.5-pareja-multicoptero> (visited on 05/05/2018).
- [24] *Pulse Width Modulation with analog write*. URL: <http://robotic-controls.com/book/export/html/57> (visited on 06/04/2018).
- [25] *Desire power V8 Series 3s 4000mAh 30C*. URL: <https://rc-innovations.es/Baterias-lipo-3s-calidad-drones/desire-power-3s-4000-30c-bateria-lipo7> (visited on 03/23/2018).
- [26] *Pixhawk Overview*. URL: <https://pixhawk.org/> (visited on 04/21/2018).
- [27] *Pixhawk clones good/bad*. URL: <https://discuss.ardupilot.org/t/pixhawk-clones-good-bad/9552> (visited on 06/10/2018).
- [28] *Pixhawk clones good/bad*. URL: <https://whatis.techtarget.com/definition/firmware> (visited on 05/03/2018).
- [29] *Arduino*. URL: <https://en.wikipedia.org/wiki/Arduino> (visited on 04/06/2018).
- [30] *Arduino Mega 2560 REV3*. URL: <https://store.arduino.cc/usa/arduino-mega-2560-rev3> (visited on 03/05/2018).
- [31] *Telemetry Radio*. URL: <http://ardupilot.org/copter/docs/common-sik-telemetry-radio.html> (visited on 02/22/2018).
- [32] *Flysky FS-i6 AFHDS*. URL: <https://www.tomtop.com> (visited on 04/03/2018).
- [33] *HCSR04*. URL: <http://www.cordobatec.com/producto/sensor-ultrasonico-hc-sr04/> (visited on 04/05/2018).
- [34] *Arduino IDE interface*. URL: <https://www.pololu.com/docs/0J61/6.2> (visited on 03/04/2018).
- [35] *Flight Modes*. URL: <http://ardupilot.org/copter/docs/flight-modes.html> (visited on 03/06/2018).
- [36] *Stabilize Mode*. URL: <http://ardupilot.org/copter/docs/stabilize-mode.html> (visited on 02/05/2018).
- [37] *RC Hardware What is an RC transmitter?* URL: <https://www.pololu.com/docs/0J61/6.2> (visited on 04/05/2018).
- [38] Ernesto Santana. *MAVLink: Protocolo de comunicación para drones*. URL: <http://www.xdrones.es/mavlink/> (visited on 02/02/2018).

- [39] *System Requirements*. URL: http://www.sebokwiki.org/wiki/System_Requirements (visited on 04/04/2018).
- [40] David Morante Manuel Soler Xin Chen Gonzalo Sánchez. *Quadcopter assembly laboratory*. University Carlos III of Madrid. 2016.
- [41] *Advanced Tuning*. URL: <http://ardupilot.org/copter/docs/tuning.html> (visited on 02/06/2018).