People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

AMO University of Bouira

Faculty of Sciences and Applied Sciences

GE Department

Master Thesis

in Electromechanics

Speciality: *Electromechanics*

Theme

Design Simulation and Realization of UAV

Advisors

- HAROUN Smail
- AIT ABBAS Hamou

Realized By

- BELHADJER Younes
- NACERI Billel

2020/2021

Acknowledgment

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout our research work to complete the research successfully.
We would like to express our deep and sincere gratitude to our research supervisors, **Dr. S. Haroun i** and **Dr. H. Ait Abbas i**, for giving us the opportunity to do this research and providing invaluable guidance throughout the whole research. Their dynamism, vision, sincerity and motivation have deeply inspired us.

They have taught us the methodology to carry out the research and to present the research works as clearly as possible. It was a great privilege and honor to work and study under their guidance.

We are extremely grateful for what they have offered us. We would also like to thank them for their friendship, empathy, and great sense of humor.

We are extending our heartfelt thanks to our classmates, friends, CRTI group and **Mr.Kechida, Mr.Kasri, Mr.Laribi, Mr.Gerbass, Mr.Badji and his family, Mr.Arbawi, Mr.Embarek, Mr.Madi, Mr.bahloul, Mr.Aisouni, Mr.Bouhedda, Mr.Fekkik** for their acceptance and patience during the discussion we had with them on our research work and thesis preparation.
We are extremely grateful to our parents for their love, prayers, caring and sacrifices for educating and preparing us for our future.

We are very much thankful to our friends for their love, understanding, prayers and continuing support to complete this research work. Also, we express our thanks to our sisters for their support and valuable prayers.

Our Special thanks goes to our friend Mr.Bibi Karim for the keen interest shown to complete this thesis successfully.

We would like to say thanks to our teachers throughout our years of study in the University of Bouira, for their constant
encouragement.
we express our special thanks to them for their genuine support throughout this research work.

Finally, our thanks go to all the people who have supported us to complete the research work directly or indirectly.

Dedication

TO My mother
a strong and gentle soul who taught me to trust Allah, believe in hard work and that so much could be done with little
My father for earning an honest living for us and for supporting and encouraging me to believe in myself
my beloved sisters, relatives , mentor, classmates and friends who have been my source of inspiration and gave me strength when I thought of giving up , who continually provide their moral, spiritual , emotional and financial support
No one who achieves success does so without the help of others. The wise and confident acknowledge this help with gratitude.
So I think everyone who helped me in this thesis directly or indirectly.

Belhadjer Younes.

Dedication

To My mother

the wisest and kindest, words just can't describe her beautiful soul. I can never thank her enough for being by my side through all my life,

and I will continue learning from her forever

My father

The most honest, humble and reliable. I can never forget the sacrifices he made throughout my whole life; I want to thank him for never letting me down.

My beloved two sisters and my brother for bearing me and supporting me and the huge moral support.

My mentors, Mr. A. Hamou and Mr. H. smail for helping me during my years of study and for their moral support during my hard times.

My classmates who became brothers and sisters, for being by my side through thick and thin, who continually provide their moral, spiritual and emotional support.

A wise man once said: "The greatest good is what we do for one another"

Naceri Billel.

Contents

List of Figures

List of Tables

Abreviations List

HALE	High Altitude Long Endurance
MALE	Medium Altitude Long Endurance
RF	Radio Frequency
UAV	Unnamed Air Vehicle
ESC	Electronic Speed Controller
RC	Radio Controller
FPV	First Person View
RPM	Rotation Per Minute
PDB	Power Distribution Bored
AMP	Aircraft Maintenance Program
AOA	Air Operations Area
Li-Po	lithium-poly
OSD	Operational System(s) Development
Tx	transmits
Rx	Receives
WDR	Weight Data Record
TVL	TV line
VTX	video transmitter
LTE	Loss of tail-rotor effectiveness
AIO	Analog Input/Output
VBAT	Voltage Batterie
PPM	Pulse-position modulation
SBus	Serial Bus

PWM	Pulse Width Modulation
GND	Ground
UART	Universal Asynchronous Receiver Transmitter
SITL	Software In The Loop
RTL	Return To Launch mode

Symbols List

${}^{b}\dot{V}^{b}_{CM\|i}$	Linear acceleration
CM|i	The inertial frame's center of mass
J^b	The quadcopter's inertia in relation to the body frame
C_T	The rotor's thrust coefficient
ρ	The density of air
A_r	The cross-sectional area of the propeller's rotation
r	The rotor's radius
ω	The rotor's angular velocity
T	Thrust Coefficient Relation
Q	Torque Coefficient Relation
C_Q	The torque coefficient for the motor/prop system
ω_{ss}	The projected steady-state motor RPM
C_R	The throttle percent to RPM conversion coefficient
(J_m)	The inertia of each motor's rotating components
ω_i	The speed of each motor/prop system
$M^b_{A,T}$	The moments present in the body frame as a function of the system's aerodynamics
$F^b_{A,T}$	The forces acting in the body frame on the quadcopter due to aerodynamics and thrust
P	Roll
Q	Pitch
R	Yaw
${}^{b}\dot{\omega}^{b}_{b\|i}$	The angular acceleration across each axis in the body frame with respect to the inertial frame

${}^{b}\Omega^{b}_{b\|i}$	A cross-product matrix for rotational velocity
$\omega^{b}_{b\|i}$	The rotational velocity of the quadcopter body within the body frame
$C_{b\|i}$	ZYX Sequence Rotation Matrix
Φ	Euler Kinematic Equation
${}^{b}\dot{V}^{b}_{CM\|i}$	Velocity State Equation
${}^{i}\dot{P}^{i}_{CM\|i}$	Position State Equation

Introduction

Recently, the control of flying machines has attracted the attention of automation researchers. Different approaches have been proposed to control planes, helicopters, rockets, satellites, mini helicopters, drones, etc. [1, 2, 3, 4]. Each of these devices does not have a precise model describing its behavior.

Interest in remote-controlled aerial drones seems to be growing more and more, particularly for military applications (demining, for example) and intervention in hostile environments (radioactive environments). One can imagine a drone to explore a contaminated building or tunnel and make a first observation before any human intervention.

Research in the field of autonomous aerial vehicles is essentially multidisciplinary. In fact, it involves a wide variety of fields such as aerodynamics, signal and image processing, automatic control, mechanics, composite materials, real-time computing.

In this thesis, we are particularly interested in air vehicles and more particularly in a quadrotor (with four propellers). Quadrotor drones are among the most complex of flying objects, because their flight dynamics are inherently nonlinear, and the variables are strongly coupled. The quadrotor has the ability to hover, which is required in some applications.

The objective of this project is building an autonomous quadrotor drone with maximum stability in different environments (like wind), implement a companion computer to our flight controller in order to script missions, the main goal is to implement advanced scripting on our companion computer with the associated camera, we will try to accomplish a python script in order to control our quadrotor with gesture commands.

The structure of the chapters:

Chapter 1: we will make an overview of UAVs.

Chapter 2: using AutoCAD and Ansys programs we will make a conception of the whole frame and test its durability.

Chapter 3: consists of studying the compatibility of the quadcopter components and the realization prosses.

Chapter 4: understanding and implementing the mathematical model of the quadrotor, optimizing the PIDs, making a Simulink model and interpretate the results.

Chapter 5: writing advanced python scripts in order to make a variety of programs, the main goal being image processing and gesture control.

Chapter 1

Introduction of UAVs

1.1 Introduction

A drone is a flying device which is capable of flying and carrying out a mission without human presence on board, it can fly autonomously or with the assistance of a pilot at a station. Drones are capable of carrying cameras, sensors, communication equipment or other devices. They are used to carry out reconnaissance missions, search for information or combat operations, etc.

1.2 State of the art

UAVs may be classified like any other aircraft, according to design configuration such as weight or engine type, maximum flight altitude, degree of operational autonomy, operational role, etc.

1.2.1 Brief history

1.2.1.1 Early UAVs

The earliest recorded use of an unmanned aerial vehicle for warfighting occurred in July 1849,[24] serving as a balloon carrier (the precursor to the aircraft carrier) [5] in the first offensive use of air power in naval aviation.[6] [27][8] Austrian forces besieging Venice attempted to launch some 200 incendiary balloons at the besieged city. The balloons were launched mainly from land; however, some were also launched from the Austrian ship

SMS Vulcano. At least one bomb fell in the city; however, due to the wind changing after launch, most of the balloons missed their target, and some drifted back over Austrian lines and the launching ship Vulcano.[9][10][11]

Significant development of drones started in the early 1900s, and originally focused on providing practice targets for training military personnel. The earliest attempt at a powered UAV was A. M. Low's "Aerial Target" in 1916.[12] Low confirmed that Geoffrey de Havilland's monoplane was the one that flew under control on 21 March 1917 using his radio system.[13] Other British unmanned developments followed during and after World War I leading to the fleet of over 400 de Havilland 82 Queen Bee aerial targets that went into service in 1935.

Nikola Tesla described a fleet of uncrewed aerial combat vehicles in 1915.[34] These developments also inspired the construction of the Kettering Bug by Charles Kettering from Dayton, Ohio and the Hewitt-Sperry Automatic Airplane . Initially meant as an uncrewed plane that would carry an explosive payload to a predetermined target. The first scaled remote piloted vehicle was developed by film star and model-airplane enthusiast Reginald Denny in 1935.[12]

1.2.1.2 Modern UAVs

With the maturing and miniaturization of applicable technologies in the 1980s and 1990s, interest in UAVs grew within the higher echelons of the U.S. military. In the 1990s, the U.S. DoD gave a contract to AAI Corporation along with the company Malat. The U.S. Navy bought the AAI Pioneer UAV that AAI and Malat developed jointly. Many of these UAVs saw service in the 1991 Gulf War. UAVs demonstrated the possibility of cheaper, more capable fighting machines, deployable without risk to aircrews. Initial generations primarily involved surveillance aircraft, but some carried armaments, such as the General Atomics MQ-1 Predator, that launched AGM-114 Hellfire air-to-ground missiles.

CAPECON was a European Union project to develop UAVs,[15] running from 1 May 2002 to 31 December 2005.[16] As of 2012, the USAF employed 7,494 UAVs – almost one in three USAF aircraft. [17] [18] The Central Intelligence Agency also operated UAVs.

[19] By 2013 at least 50 countries used UAVs. China, Iran, Pakistan, Turkey, and others designed and built their own varieties. The use of drones has continued to increase.[20] Due to their wide proliferation, no comprehensive list of UAV systems exists [18] [21].

The development of smart technologies and improved electrical power systems led to a parallel increase in the use of drones for consumer and general aviation activities. As of 2021, quadcopter drones exemplify the widespread popularity of hobby radio-controlled aircraft and toys, however the use of UAVs in commercial and general aviation is limited by a lack of autonomy and new regulatory environments which require line-of-sight contact with the pilot. In 2020 a Kargu 2 drone hunted down and attacked a human target in Libya, according to a report from the UN Security Council's Panel of Experts on Libya, published in March 2021. This may have been the first time an autonomous killer robot armed with lethal weaponry attacked human beings [22] [23].

1.2.2 Researchers projects

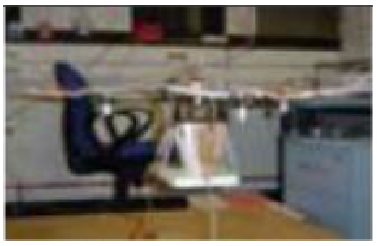
Although the " Quadrotor " configuration did not get much attention until the early 80s. Since then, several researchers have started to take an interest in this configuration for reasons of simplicity, capacity to support a payload and reduced cost.

Since 2001, several research centers and groups of specialists in aeronautics have started to publish the first results concerning the modeling and the control of this four-rotor helicopter such as: the work of "the University of Compiègne" and the project 'Robovolint' in France, and those from Lakehead University in Canada and many others.

Currently, many research projects are based on toys like the dragonflyer [14], the researchers have modified them by embedding more sensors and communication systems. Few groups are interested in the design and construction of quadrotors, and the minority of these groups do so optimally (design consideration and control) [24].

The table [1.1] shows the most interesting projects of the last 10 years:

.

project	**university**	**Photo**
Mesicopter	Stanford	
E. Altug's Thèse	Pennysylvanie	
P. Castillo's Thèse	Compiègne	
A. Clifton's Thèse	Vanderbilt	

P. Tournier's Thèse	MIT	
MID4-200	microDrones GmbH	

Table 1.1 – Some quadrotor projects

1.3 Classification of UAVs

UAVs may be classified like any other aircraft, according to design configuration such as weight or engine type, maximum flight altitude, degree of operational autonomy, operational role, etc.

1.3.1 Based on weight

Based on their weight, drones can be classified into five categories — nano (weighing up to 250 g), Micro air vehicles (MAV) (250 g - 2 kg), Miniature UAV or small (SUAV) (2-25 kg), medium (25-150 kg), and large (over 150 kg) [26].

1.3.2 Based on degree of autonomy

UAVs could also be classified based on the degree of autonomy in their flight operations. ICAO classifies uncrewed aircraft as either remotely piloted aircraft or fully autonomous. [45] Some UAVs offer intermediate degrees of autonomy. For example, a vehicle that is remotely piloted in most contexts but has an autonomous return-to-base operation. Some aircraft types may optionally fly manned or as UAVs, which may include manned aircraft transformed into uncrewed or Optionally Piloted UAVs (OPVs).

1.3.3 Based on altitude

The following UAV classifications have been used at industry events such as ParcAberporth Unmanned Systems forum:

- Hand-held 2,000 ft (600 m) altitude, about 2 km range Close 5,000 ft (1,500 m) altitude, up to 10 km range
- NATO type 10,000 ft (3,000 m) altitude, up to 50 km range Tactical 18,000 ft (5,500 m) altitude, about 160 km range
- MALE (medium altitude, long endurance) up to 30,000 ft (9,000 m) and range over 200 km
- HALE (high altitude, long endurance) over 30,000 ft (9,100 m) and indefinite range
- Hypersonic high-speed, supersonic (Mach 1–5) or hypersonic (Mach 5+) 50,000 ft (15,200 m) or suborbital altitude, range over 200 km
- Orbital low earth orbit (Mach 25+)
- CIS Lunar Earth-Moon transfer
- Computer Assisted Carrier Guidance System (CACGS) for UAVs

1.3.4 Based on the composite criteria

An example of classification based on the composite criteria is U.S. Military's unmanned aerial systems (UAS) classification of UAVs based on weight, maximum altitude and speed of the UAV component.

1.4 Types of Multirotors

Multicopters (as known as multirotors) often use fixed-pitch propellers, so the control of vehicle motion is achieved by varying the relative speed of each motor. Radio controlled multicopters are increasingly popular for aerial photography, and land surveying. More recently Drone Racing is a hot topics, where multicopters are used in racing and free-style competition.

There are many types of multirotor. They are generally categorized by the number of motors used, for example a three-motored multicopter is a called a tricopter, and the configuration can also be referred to as Y3. In this post we will discuss the following types of multirotors:

* Bicopter

* Tricopter (Y3, T3)

* Quadcopter (X4, Y4, V-Tail, A-Tail)

* Pentacopter

* Hexacopter (Y6)

* Octocopter (X8)

The number of motors and configuration have impact on flight performance, and each has its own advantages. For instance the more motors, the more power and lift capacity, which means you could carry more payload. More motors also means better redundancy in case of motor failure. But the downside is decrease in power efficiency, and increase in the cost of purchasing additional hardware and maintenance. We will also discuss the benefits of Coaxial Motor-Arrangement.

1.4.1 Bicopter – 2-Motor Multicopter

The BiCopter has two motors that can be moved by servos to desired tilt angles. It looks similar to the "Avatar Gunship" . Here is an example of the bicopter avatar gunship.

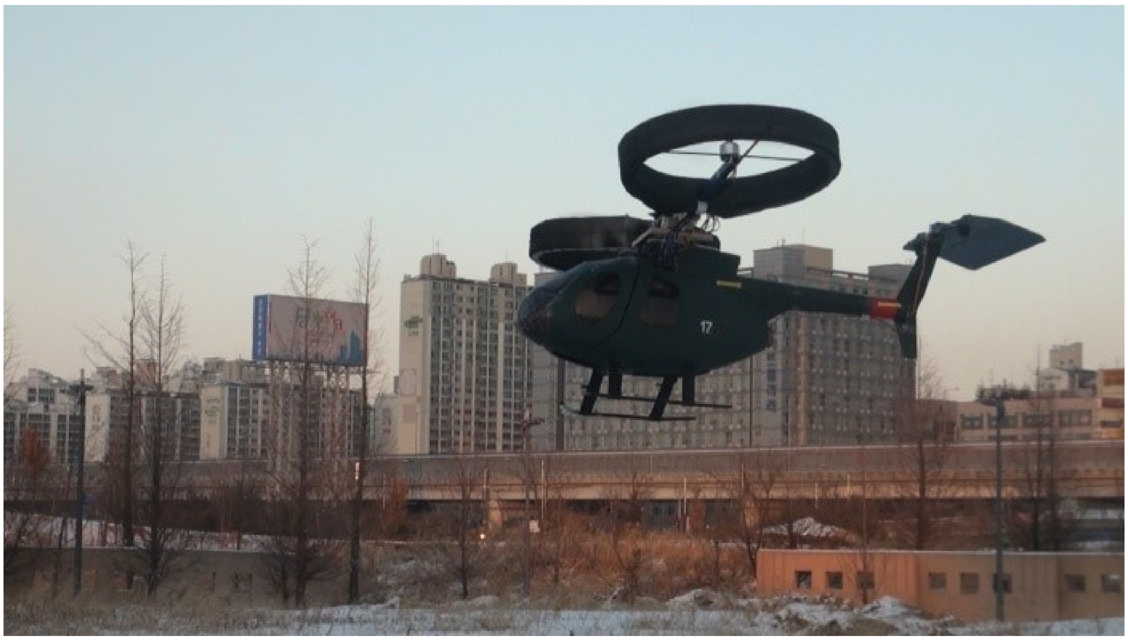


Figure 1.1 – Bicopter avatar gunship

Bicopter could be the cheapest multicopter config to build among all because it only uses two motors and two servos. But it's also the most difficult platform to stabilize in flight. It has the least lifting power given the fact that it only has 2 motors. Bicopter is not a very popular configuration for hobbyists, and there isn't much information to be found on the internet.

1.4.2 Tricopter – 3-Motor Multicopter

The Tricopter has 3 motors, and typically in a "Y" shape, where the arms are usually 120 degrees apart. Tricopters can sometimes be found in a "T" shape too. Two propellers on the front arms spins the opposite direction to counter each other out. The rear motor can be tilted left and right by a servo to enable the yaw mechanism. Here is a "T3" Tricopter example.

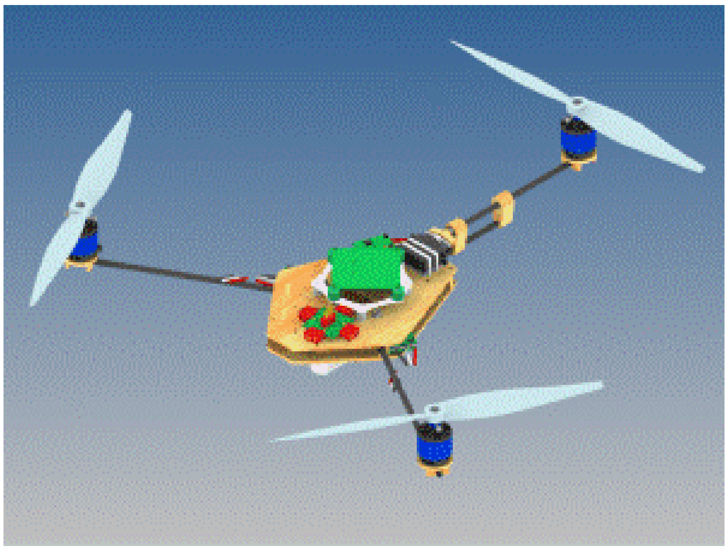


Figure 1.2 – ”T3” Tricopter

It’s a popular yet relatively cheap configuration because it only requires 3 motors, although you also need an additional servo but they are generally cheaper than brushless motors. Generally speaking, tricopters are less stable than other multirotors with more motors, and it’s not as robust due to the vulnerability of the tail servo and mechanics in crashes. For hobbyists, it’s also harder to build because of the yaw mechanism. Tricopter has more yaw authority comparing to a quadcopter. What that means is when a quadcopter or hexacopter yaws, they do so by slowing down half of the motors and speeding the other half. If the copter is already at full speed (all motors at 100%), it will have to lower the speed to make yaw happens. However on a Tricopter, it uses a servo to achieve yaw so it loses less thrust when the same situation happens. It also has lower lifting power because of the smaller motor numbers.

1.4.3 QuadCopter – 4-Motor multicopter

A quadcopter has 4 motors mounted on a symmetric frame, each arm is typically 90 degree apart for the X4 config. Two motors rotate CW (clockwise), and the other two rotate CCW (counter clockwise) to create opposite force to stay balance. Quadcopter is the most popular multirotor configuration, with the simplest mechanical structure. It’s widely used for drone racing in the form of “mini quad”.

1.4.4 Y4 – 4 motors

It looks like a tricopter but without the tail servo. There are two normal propellers and motors in front on separate arms and two coaxial motors in the rear mounted to one arm. Mechanically it's simpler than tricopters because of the absence of the and yaw mechanism.

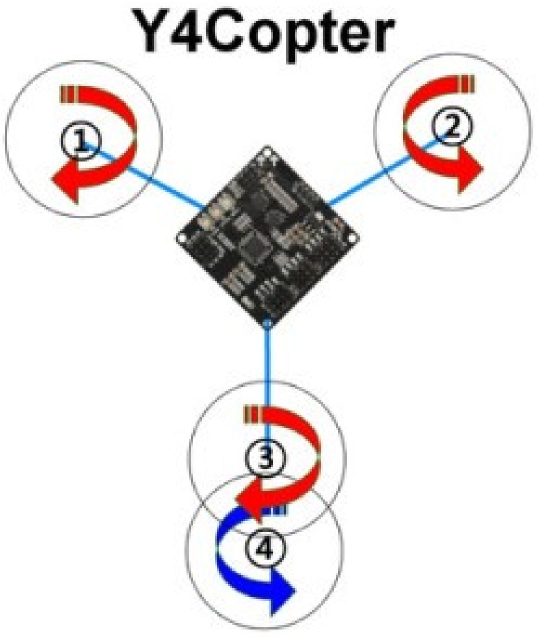


Figure 1.3 – Y4 Copter

While they weigh almost the same they have about 1/3 more lifting power than tricopters. They are usually more reliable than Tricopters because there is no potential servo issues.

1.4.5 Hexacopter – 6-Motor Multirotor

The hexacopter has 6 motors mounted typically 60 degree apart on a symmetric frame, with three sets of CW and CCW motors/propellers.

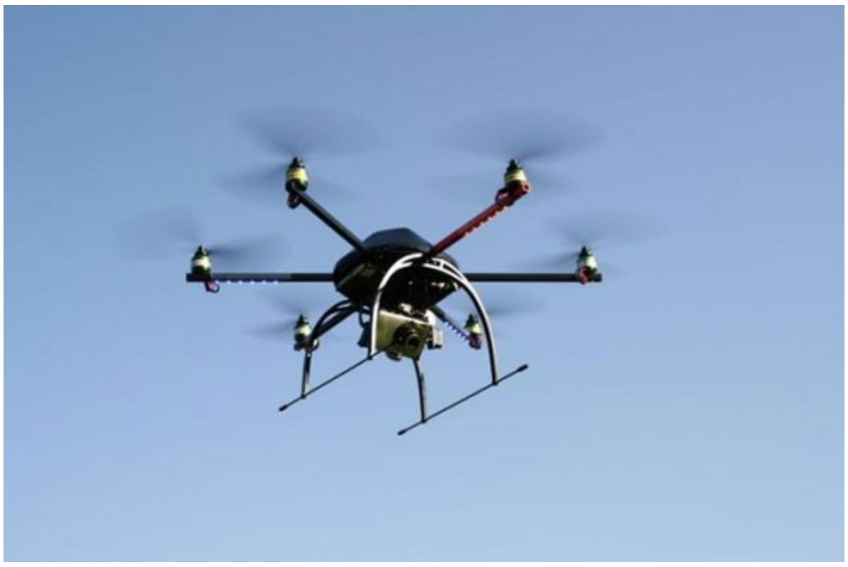


Figure 1.4 – Hexacopters

Hexacopters are very similar to the quadcopters, but they provide more lifting capacity with the extra motors. There is also improvement in redundancy: if one motor fails, the aircraft can still remain stable enough for a safe landing. The downside is that they tend to be larger in size and more expensive to build.

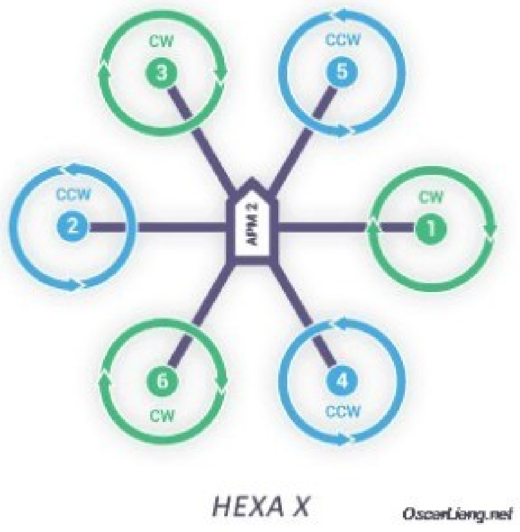


Figure 1.5 – HEXA X

1.4.6 Octocopter – 8 Motors

A typical octocopter has 8 motors on the same level with four sets of CW and CCW propellers.

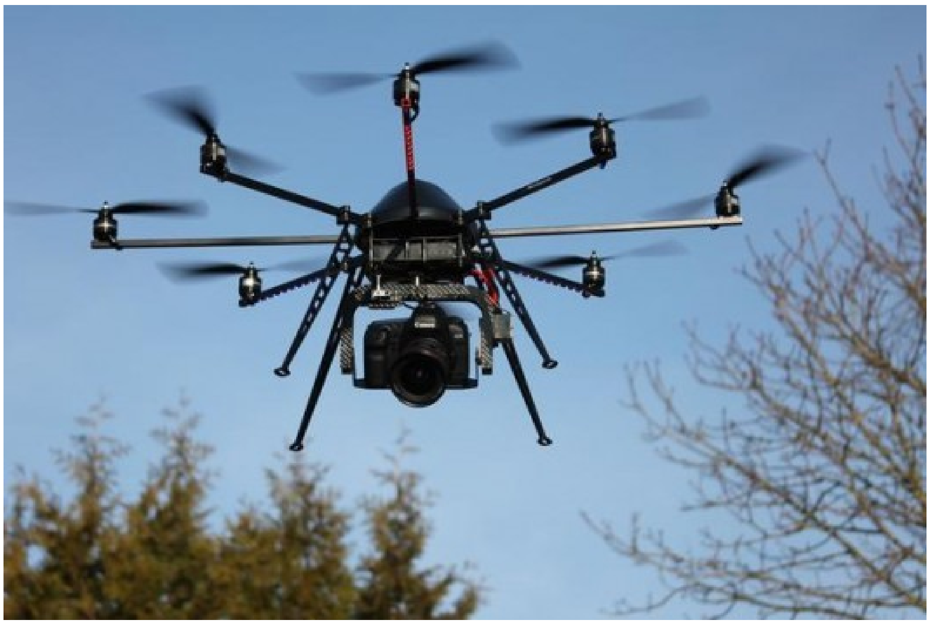


Figure 1.6 – Octocopter

Octocopters are similar to quadcopters and hexacopters. It's like an upgrade version of the hexacopter with even more lifting capacity and redundancy. However the large number of motors means they draw more current, and you will probably need to carry multiple battery packs. Also it's going to be expensive. They are very popular as aerial photography platforms and carrying heavy, professional filming gears [47].

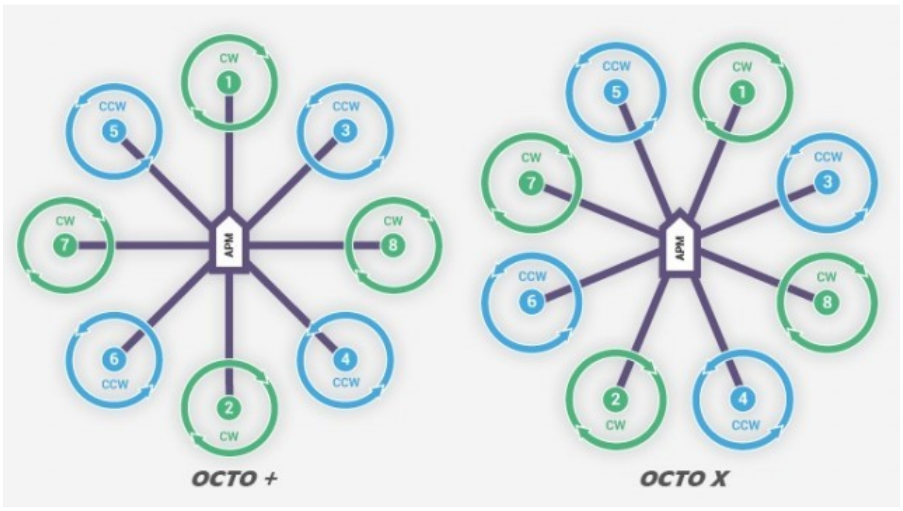


Figure 1.7 – Octo+ & Octo X

1.5 Uses of uavs

- **Military:** UAVs have been used widely in attack and combat roles. Military use of drones includes reconnaissance and observation from the sky. Cargo drones are used to supply weapons and cargo to military units.
- **Commercial:** There is a wide range of commercial applications of drones. A camera-equipped drone is used to map an area. It helps know if the proposed construction site is suitable for the construction of a particular structure. UAV is used in the commercial sector to take photos and videos of buildings, construction sites and ground areas. Real estate developers use such photos and videos to advertise their building projects.
- **Agricultural:** Farmers use drones to spray pesticides, fertilizers and other chemicals. Special cameras and sensors are used to spot problems in the crops. Diseased parts of the crop can be spotted early. Different types of data related to the farm, crop, land and atmospheric conditions can be collected. This data is used to ensure healthy crops and successful harvest.
- **Police:** Law enforcement agencies use drones to fight crimes. They use it for surveillance of a suspected target. Real-time surveillance is useful during active

crime scenes where sending the police personnel without knowing the ground situation can be dangerous.

- **3D Mapping:** Advance 3D imaging equipment installed in a drone is used to survey the landscape. Thousands of high-quality images are stitched together to create precise and high definition 3D maps of a ground area. It gives a better understanding of the geographical features of the area.
- **Disaster Relief:** It is difficult to know the magnitude of destruction immediately after a disaster. There is an urgent need to find ground information quickly. Sending search and rescue teams to such an area without prior knowledge of ground conditions may result in a waste of precious time. A UAV helps know exact locations where help is needed.
- **Hunting Hurricane:** Drones equipped with scientific equipment are used to observe storms and other natural disasters. The data collected and analyzed from such operations are used to develop predictive models that help predict an impending disaster with better accuracy.
- **Product Delivery:** This type of commercial venture is yet to take off due to regulatory constraints. However, many companies are working actively in this field. It is going to be a lucrative field for the sellers of products.
- **Research and Development:** Scientists use drones to gather different types of data related to the ground, sea and air. They can find useful data without sending several teams to the target locations. Accurate scientific data from various locations can be collected quickly and easily.
- **Reconnaissance:** UAVs are now used widely to protect border areas from intruders. It helps gather intelligence information on the battlefield. The information proves useful in protecting borders, combat units, and security installations. Military personnel can avoid high-risk missions or go to such missions with better information on the ground situation
- **General Users:** Hobbyists use small-size drones for recreational purposes. These units are used to enjoy the thrill of flying an aircraft. Now many UAVs made for

general users have a camera to take photos and videos. Some new UAV models can follow the moving drone pilot. There are strict drone flying rules and regulations that hobbyist drone operators must know [48].

1.6 Applications of UAVs

Multirotor UAV Drones are utilised in a huge variety of applications; we really are only limited by our imagination!

Drones offer the unprecedented ability to go where you could never go before, and experience perspectives never seen. There are so many video's appearing online capturing the beauty of planet earth from the air, which were just not possible previously.

Some of the most common commercial applications and uses for UAV Drones are [49]:

- Aerial Photography & Videography
- Real estate photography
- Mapping & Surveying
- Asset Inspection
- Payload carrying
- Agriculture
- Bird Control
- Crop spraying
- Crop monitoring
- Multispectral/thermal/NIR cameras
- Live streaming events
- Roof inspections
- Emergency Response

- Search and Rescue
- Marine Rescue
- Disaster zone mapping
- Disaster Relief
- Forenzics
- Mining
- Firefighting
- Monitoring Poachers
- Insurance
- Aviation
- Meterology
- Product Delivery

1.7 The regulations

In Algeria, the law applicable to drones includes provisions relating to both civil and criminal law. On the other hand, the piloting of drones is also governed by the transport code and the civil aviation code [25].

Both for the safety of aircraft and that of people on the ground, as well as for the respect of private life and image rights: it is very important that pilots know and apply these texts.

Due to their knowledge of these texts, BC Drone Réalisation pilots respect and apply these laws, which is your guarantee of safety [46].

1.8 Conclusion

In this chapter we provide a state of the art of drones. A description of the different helicopter drone configurations is given. We also offer more modern configurations made very recently either by university researchers or by the military sector.

Chapter 2

Design and Structural Analyses of Quadrotor

2.1 Introduction

The need of UAVs/quadcopters is tremendously increasing when it comes to the industry of aerospace. Making them robust and tough is a necessity in order to control and operate them in possibly tough or challenging environments to subdue the failures in flying. The physical structure and its design principles and modeling play an essential role in UAVs or Quadcopters due to its capability to hover in minimal areas and vertical takeoff and landing. Quadrotors are provided with four motors and it's a must that the propellers are locked to generate lift. The four motors handling the propellers are fixed on the end of the joist in the frame, usually forms a structure of a cross, made from four joists. In order to lift a quadrotor from the ground, two motors have to be fixed in the opposite direction of the other two motors (the first two rotates in clockwise direction and the other two in counter clockwise direction). To move and hover the quadrotor precisely, each motor will produce the necessary amount of thrust and torque in a spinning motion. Respecting the direction of revolving of the motors according to their Co-ordinates will cause the canceling of all the forces in order to maintain the same speeds. In the state of hovering, rotating the quadcopter in the desired direction some points must be took under consideration. Producing the 'yaw' in a direction requires the speed of two motors on any side (right/left/upper/bottom of the frame) to be greater than the remaining two. 'Roll' and 'pitch' are also produced by interchanging the speed of motors [27], as shown in figure 2.1

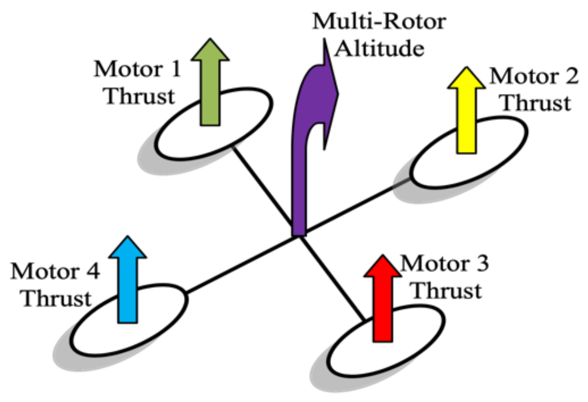


Figure 2.1 – arrangement of a quad copter

2.2 Literature Review

Jeong et al. [29] argued the unidirectional flying four-wheel drives with built-in propellers. This drive has the ability to hoveri in the air and maneuver on land. The stabilization of signals is controlled by Kalman filter and consequently fed to PID controllers.

Lim et al. [30] argued the layout and controlling effect of the flying robot with landing frame and double guided rotors. Ultrasonic and gyro-sensor are used to sense the altitude and the orientation of the situation. In order to control the robot, all the computations were used by the H8/3694F microcomputer.

Javir et al. [31] argued the quantity of thrust desired during various maneuvers of quadrotor. The study of the gravitational force, the weight of the frame and all components, stress, natural frequency, deformation is done by ANSYS software to assure the safe design.

Kumar et al. [32] argued a specific type of UAV with the ability of vertical takeoffs and smooth landing. The direction is varied by changing the individual propeller's speed without collective pitch & the cyclic control in this model.

Abishini et al. [33] developed a quadrotor specialized in underwater ship wrecks as well as aerial surveillance, crashes and other applications. certain aspects are considered to be important for developing such models like floatability and buoyancy for the integrity and power requirements of the quadrotor. This application explains the modeling of the frame in CATIA and later design is approved through stress analysis and comparison with other materials.

Prajwal et al. [34] developed a quadcopter for raising the greatest payload capacity. This study demonstrates the modulization of the frame in CATIA V5 R19 and ANSYS workbench were used for the static and dynamic analysis. CFD analysis as well was performed on the frame for calculating the maximum deformation and stress. The final results acquired were safe and within the limits.

2.3 The Design

2.3.1 AutoCAD Design

The frame we used for the quadrotor is a 500mm frame due to its vast range of usage and prospects, as well as swift weight and high durability and strength. Our 500mm frame takes a shape of 'X' geometrically which supply more room and space to mount additional components in the quadrotor. The model of the frame is developed in AutoCAD 2016 taking in consideration the dimensions from other well-known frames by taking attention to the safety measures, requirements and smooth functioning and optimal utilization of motors, propellers and electrical components.

The stands, layers, arms and spars are designed individually and assembled as shown in Figures. (2.2; 2.6). The dimensions used of the quadrotor frame are given in Table 2.1 .

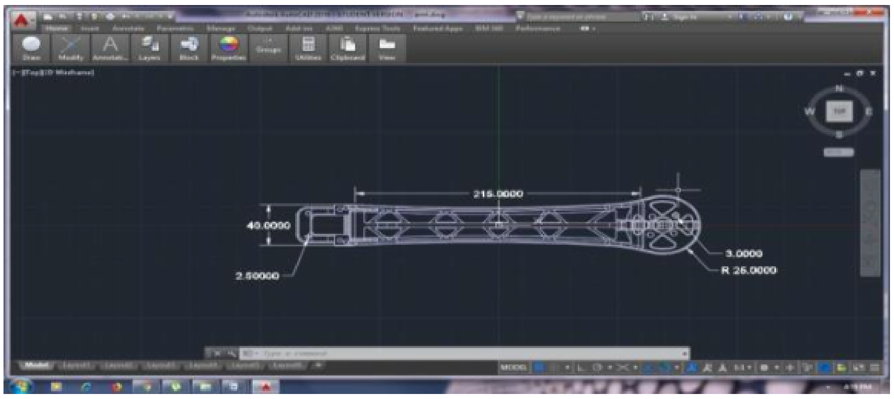


Figure 2.2 – Sketch of quadcopter arm

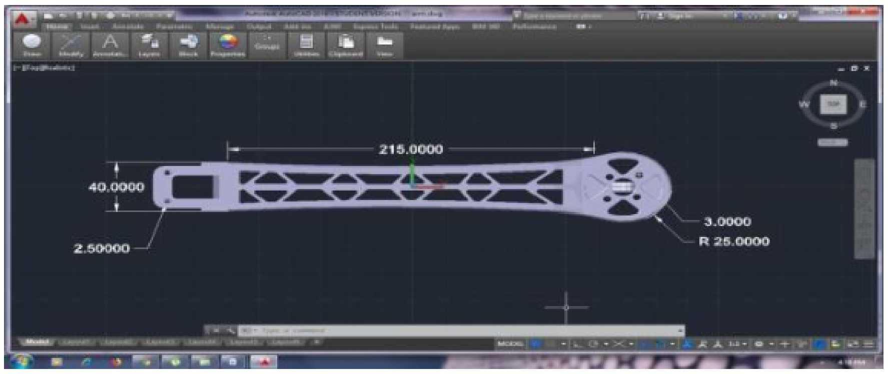


Figure 2.3 – Model of quadcopter arm in AutoCAD

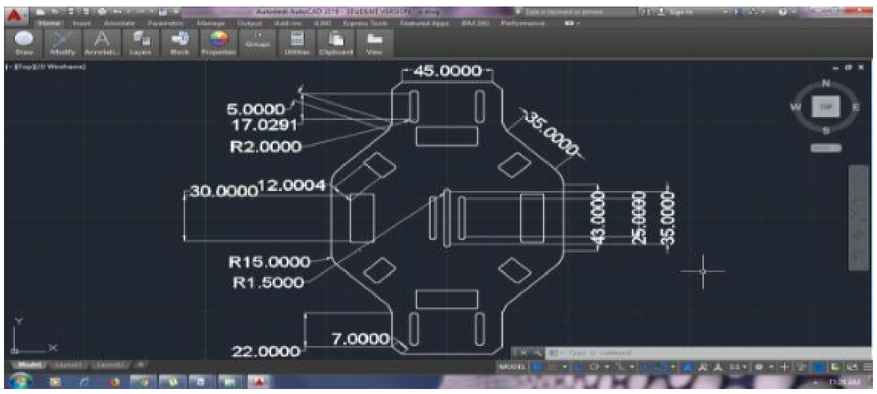


Figure 2.4 – Sketch of Base plate of quadcopter

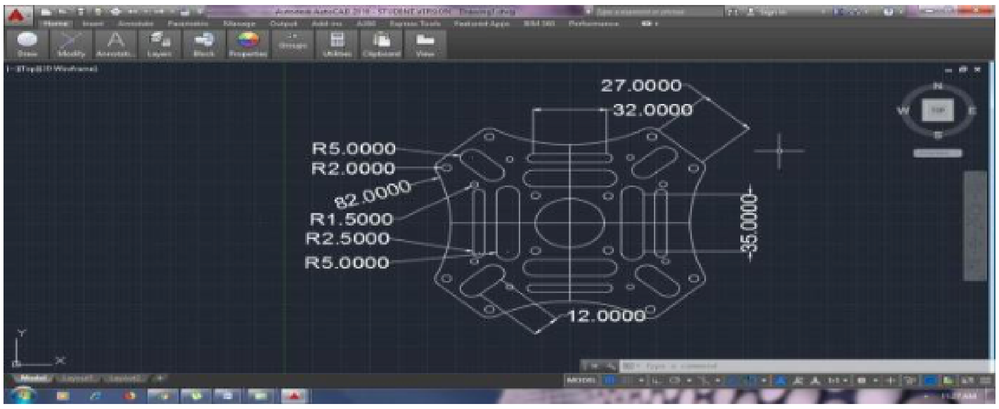


Figure 2.5 – Sketch of quadcopter top plate

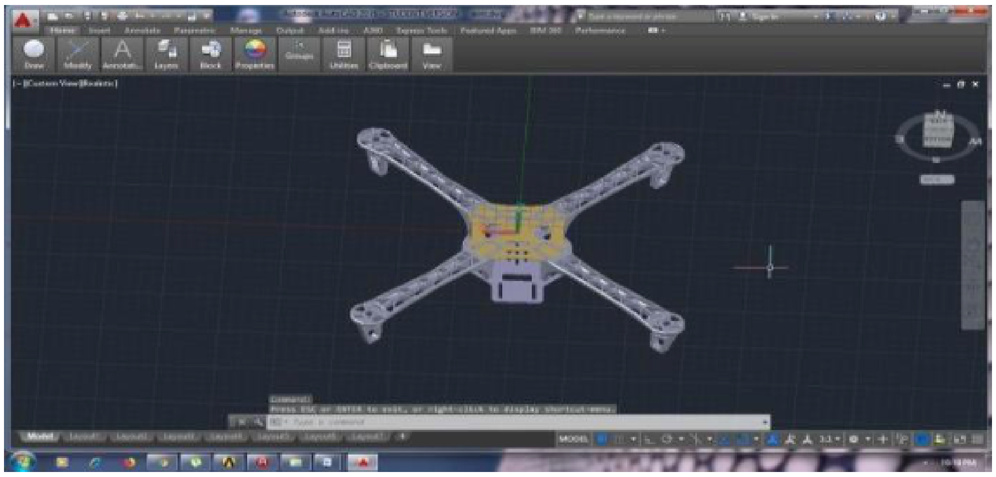


Figure 2.6 – Assembly of quadcopter frame in AutoCAD

.

Component number	**Quadcopter Components**	**Specifications(mm)**
1	Length of Arm (four arms)	220
2	Dimensions of Arms	40*13*0.6
3	Base plate dimensions (Length of vertical, horizontal sides, & thickness)	43*43*2
4	Rectangular drills on base plate	25*35
5	Radius of vertical and horizontal sides of top plate & thickness	82*2
6	Radius of circular drill on top plate	1.5*2
7	Length and breadth of square drill on top	5*32*35
8	Distance between two motors	500
9	Dimensions of Motor nut & screw	ϕ2.15

Table 2.1 – quadcopter dimensions

2.3.2 STRUCTURAL ANALYSIS OF QUADCOPTER

For a quadrotor, the frame is the main load carrying components. The ability to endure great compressive and tensile loads makes them more appealing. The analysis of the structure of the frame of the quadrotor prototype was carried out on ANSYS 17.0 software.

2.3.3 Von-mises stress analysis of base plate

the analysis of the structure is made out on the base plate of the quadrotor in ANSYS 17.0 software. The properties obtained for bearing static structural analysis are shown in Table 2.3. Meshing resolution used is good to obtain precise results. The greatest tensile strength of the base plate (PA66GF30) material is 160 MPa. The extreme equivalent stress that is acquired is 5.99 MPa as listed in figure 2.7. Therefore, the base plate is safe to endure the loads.

.

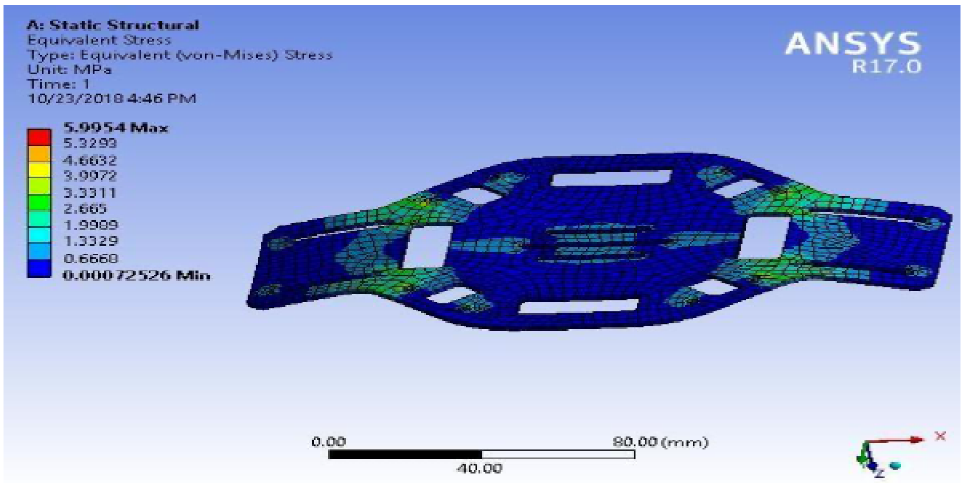


Figure 2.7 – Von-mises stress analysis for tensile load of 20N on base plate

material	Density (kg/m^3)	Young's Modulus (MPa)	Tensile strength (MPa)	Poisson's ratio
PA66GF30 Material	1370	15000	160	0.36

Table 2.2 – BOUNDARY CONDITIONS

2.3.4 Von-mises stress analysis of top plate and of our 500 mm frame

The absolute tensile strength of top plate (PA66GF30) material is 160 MPa. The greatest equivalent stress quired is 2.255 MPa as listed in Figure(2.8) therefore, the top plate is safe to endure the loads. The greatest equivalent stress that is acquired is 23.0 MPa as listed in Figure(2.9) therefore, our 500 mm frame is safe to endure the loads.

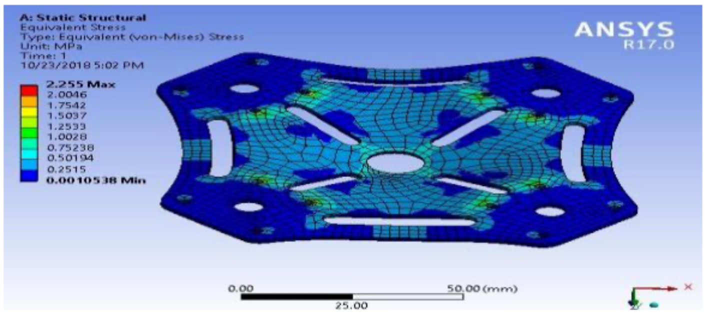


Figure 2.8 – Von-mises stress analysis for tensile load of 20N on top plate

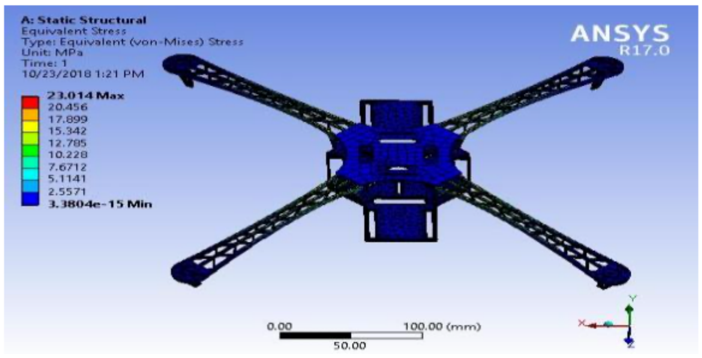


Figure 2.9 – Von-mises stress analysis for tensile load of 20N on our frame

2.3.5 Gross deformation analysis of base plate

The gross deformation acquired for the base plate is minimum 0 mm to maximum 0.14025 mm as shown in Figure(2.10).

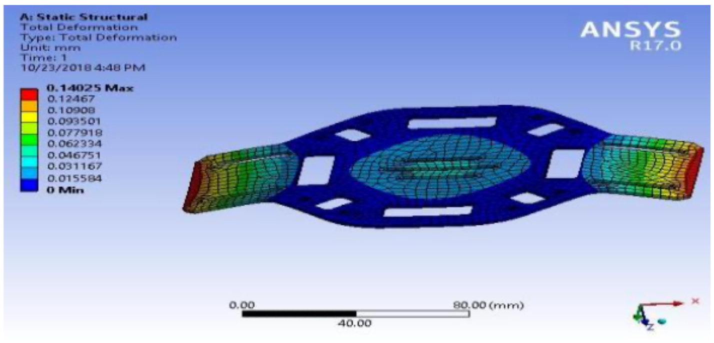


Figure 2.10 – Total deformation analysis for tensile load of 20 N on base plate

2.3.6 Gross deformation analysis of top plate and 500mm frame

The gross deformation acquired for top plate and 500mm frame is minimum 0 mm and maximum is 0.0097 mm and 4.135 mm as shown in Figures 2.11 & 2.12

The quadrotor frame prototype is subjected to static structural analysis in order to find the deformation when the load is applied on the 500mm frame. The results acquired are within the limits when compared to tensile strength of PA66GF30 material. Therefore, the frame is very safe to use and can endure the crashes.

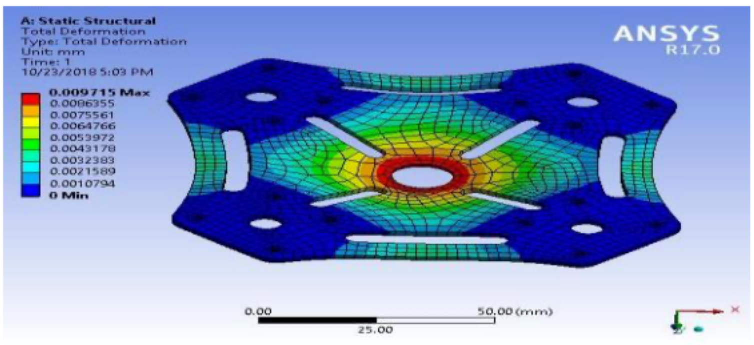


Figure 2.11 – Total deformation analysis for tensile load of 20 N on top plate

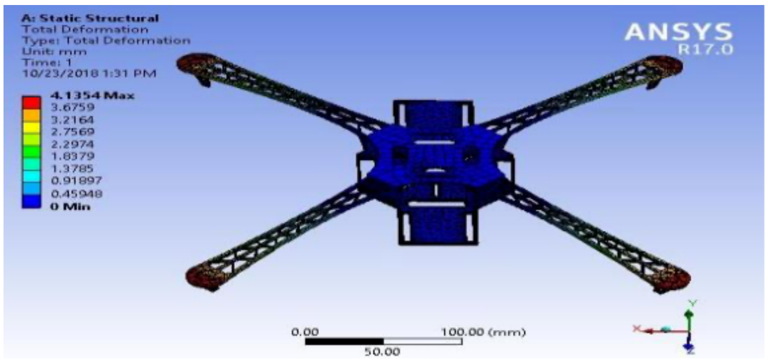
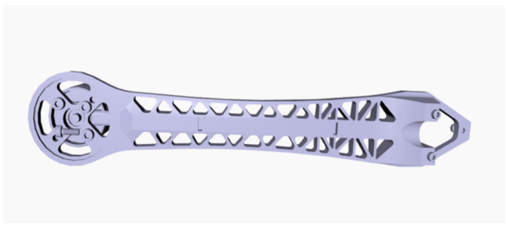
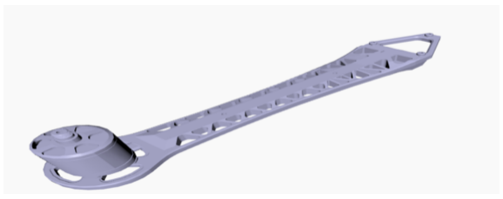
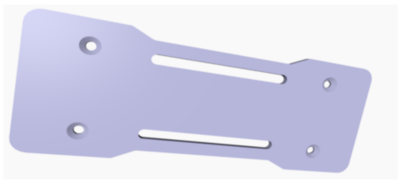
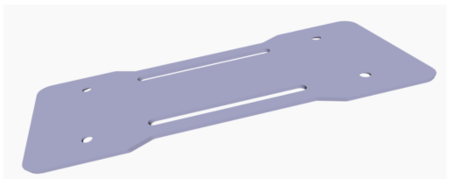
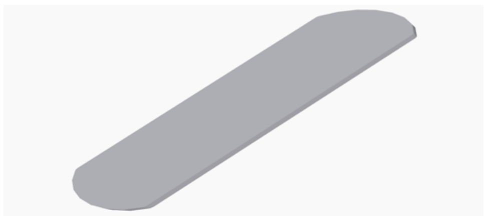
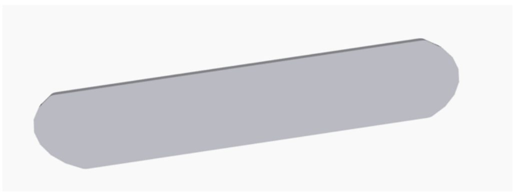
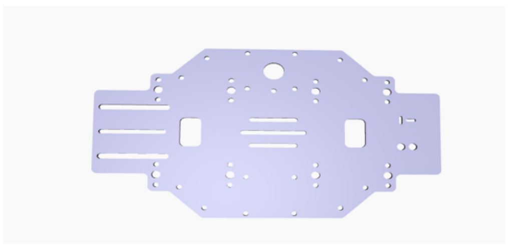
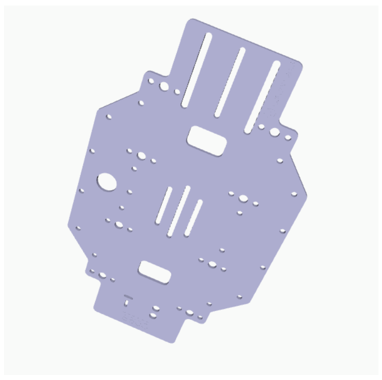
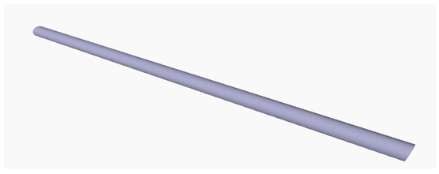
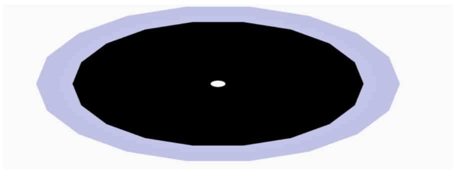
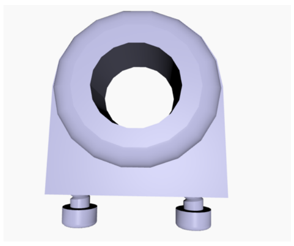
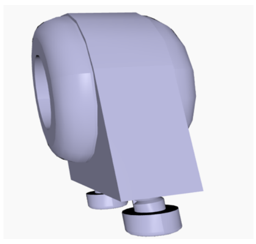
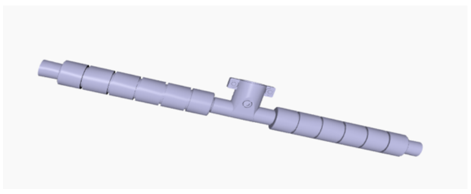
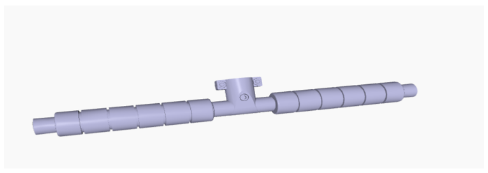
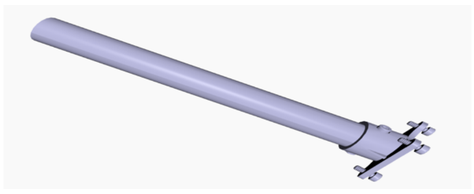
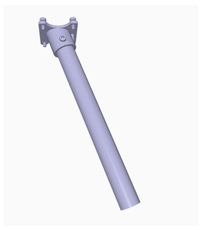
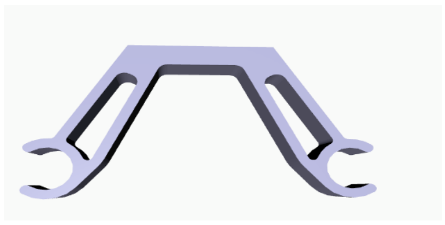
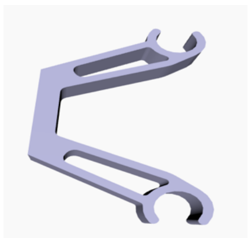


Figure 2.12 – Total deformation analysis for tensile load of 20 N on our frame

2.4 Final view of printable 3D components

Arm	1st view	2nd view
Arm		
Battery Ban		
Battery Pad		
Bottom plate		

Carbon fiber tube		
Hanger		
Landing gear bar		
Landing gear pole		
Pylons		

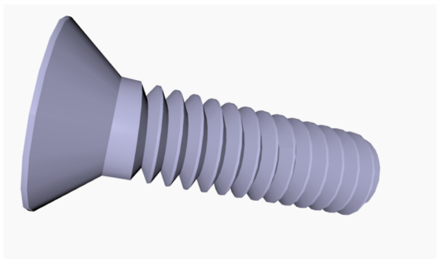
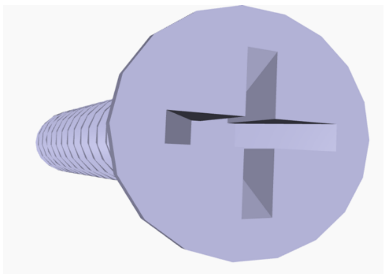
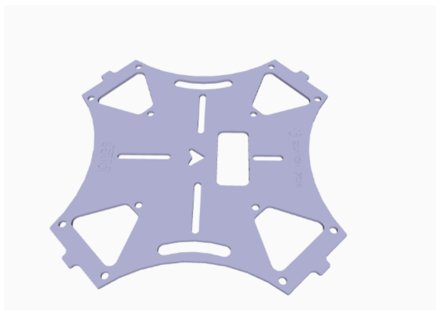
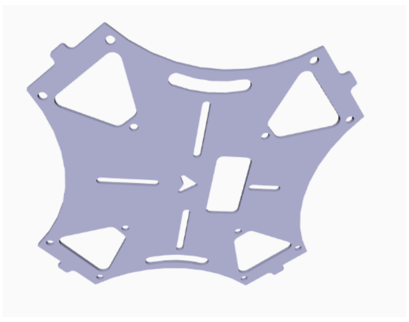
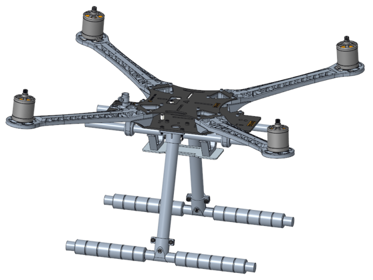
Screws		
Top plate		
The whole frame		

Table 2.3 – View of printable 3D components

2.5 Conclusion

Deformation and stresses induced are calculated using ANSYS Workbench 17.0. The obtained stress (23.0 MPa) and total deformation (4.135 mm) results for various type of analysis are within the required border.

Depending on the results obtained it is stated that modeling of quadcopter frame is safe and can be used for different applications of payload category.

Chapter 3

Component Selection and Optimization

3.1 Introduction

In a simple manner, to build a quadcopter a simple understanding of all the components that are used to fly is needed, a quadrotor consists of many essential parts; the frame, the motors, ESCs (Electronic Speed Controllers), propellers, battery, flight controller as well as an RC controller [35]; there are many additional components that can be added in order to improve the quadcopter, in our case we are aiming to make a programmable drone, therefore we used a raspberry pi 4, FPV camera and a pi camera v.1.3

The first step when building a quadcopter is choosing the desired size of the frame, we need to know approximately how much weight its going to handle in total, according to that, the motors can now be chosen, the motors can support a maximal amperage depending on their RPM , now the selection of the ESC's is based on the motors, the last crucial element is the battery, the ESC's support different kind of batteries (2S,3S,4S . . . . .etc)

3.2 Choosing quadcopter components

3.2.1 Frame

A quadcopter's frame is the core structure, or skeleton, to which the remaining components will be connected. Once you've selected what you want to do with your craft (aerial photography, racing, micro freestyle, etc.), you'll need to figure out what size is best for you. The size of the props you use will be determined by the size of the frame (or vice versa) [35], in our case after making the conception; we couldn't find where to print our 3D model so we got a model that is close geometrically to it and we landed on the 'S500' as our frame, as shown in figure(3.1).

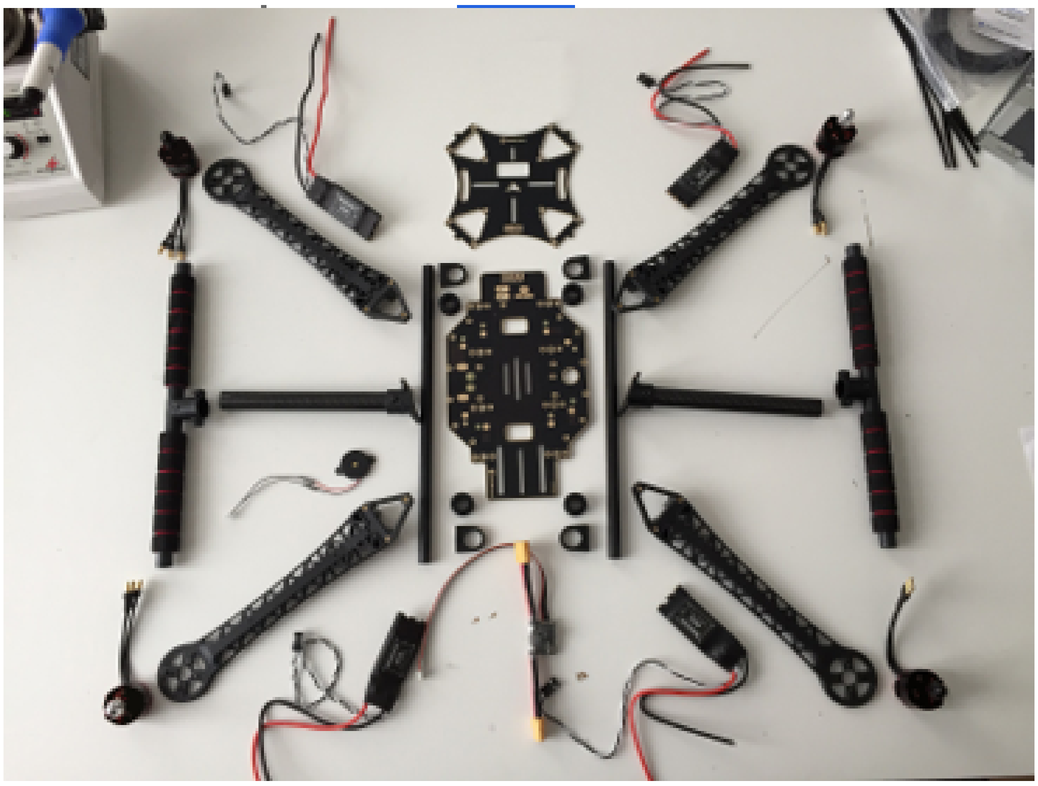


Figure 3.1 – The frame and the components that will be fixed on

This frame is spacious and has a built-in PDB which makes the connection of the other components easier.

3.2.2 Motors

Given your quad's motors are the primary drain on battery power, finding an efficient propeller and motor combination is critical. The motor speed is measured in kV; a lower kV motor produces greater torque, while a higher kV spins faster; however, this is without the propellers connected.

Aside from pure thrust, there are a number of factors that affect motor performance, one of which is how much current the motor draws from the battery. Check the specs of your motors for the maximum amp draw, and make sure your ESCs are rated to handle that amount of power [35].

The desired weight aimed for our build is approximately 2 KGs, therefore we had to choose powerful motors, we picked the sunnysky 1000 KV "brush-less DC motor", keep in mind that 'KV' doesn't mean kilovolts, however it determines the relationship between the motors input, its supplied voltage and its speed , '1000 KV' essentially means 1000 RPM per volt of input, in other words the bigger the 'KV' the faster the motor is going to spin for a given voltage and it works out that motors that spin faster tend to be most efficient with smaller propellers and better suited for smaller quadcopters like racing drones , in our case we needed a motor that is efficient in the ratio of weight-power as shown in figure(3.2)

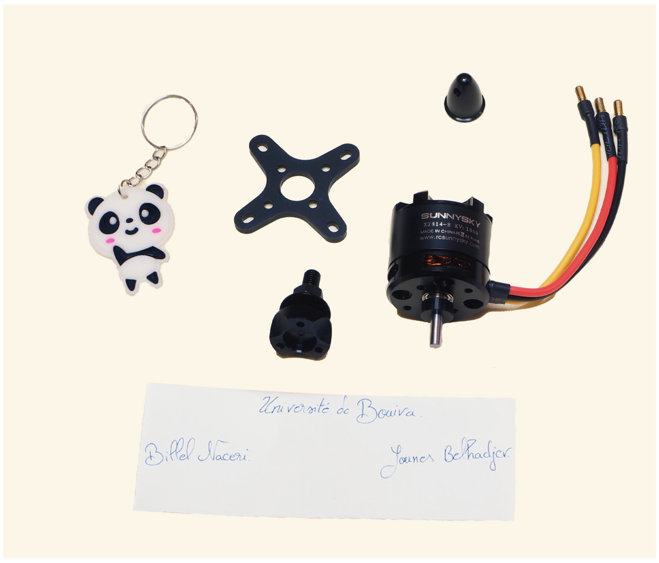


Figure 3.2 – Sunnysky 1000 KV "brush-less DC motor"

3.2.3 ESC's

An ESC is a device that analyzes data from the flight controller and converts them into phased electrical pulses in order to determine the brushless motor's speed. It's a must to make sure your FC and ESCs can both run the same ESC protocol. When choosing an

ESC, keep in mind that the current rating must be more than the amperage drawn by your motors and propellers. An ESC has four input terminals, two of which are for signals from the FC. The FC is wired with signal and signal ground; the two heavier wires are for Positive and Negative, and they transport the high current to the ESC to power the motor. Positive and negative wires are connected to the PDB. ESC's has three outputs that are terminals and connected to the motors [35].

We used Sunnysky 1000 KV "brush-less DC motors", they support a current of 30A-40A, therefore we have chosen the 'Simonk 30A ESCs' with 30A of continuous current rating, they also have a peak current of 45A, therefore a peak for a short period of time will not overheating them ,besides they are light in weight ,their voltage rating is 2S to 4S (lithium cells in series), during the process and tries we burned some of our ESCs and we had to change them, the only alternatives we found were 'EMAX 30A ESC's' as shown in figure(3.3)

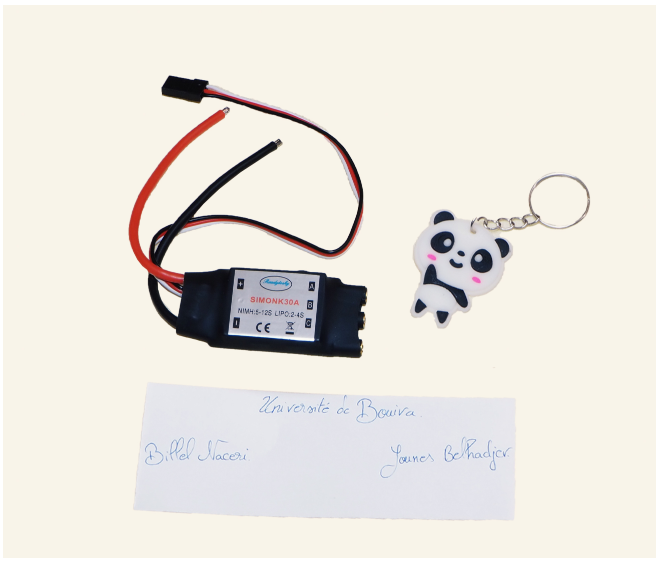


Figure 3.3 – Simonk 30A ESC's

3.2.4 Propellers

propellers are generally specified in terms of inches diameter and pitch, for quadcopters, there are probably thousands of various kinds of propellers, with many alternatives in practically every size. A larger propeller will require more torque from the motor than a lighter one; also, blades with a higher AOA (Angle of Attack – nicknamed ”aggressive props”) will meet more air resistance and require more torque. A motor consumes more Amps when it has to work harder to turn [35]. Finding the appropriate balance between thrust produced and amperage utilized by the prop and motor combination is a balancing act that every quad pilot goes through; there is no ”right answer.”

we used propellers with 10 inches in diameter and a pitch of 4,5 inches which produce a good amount of thrust. The model of our propellers is the ‘1045’. As shown in figure(3.4)

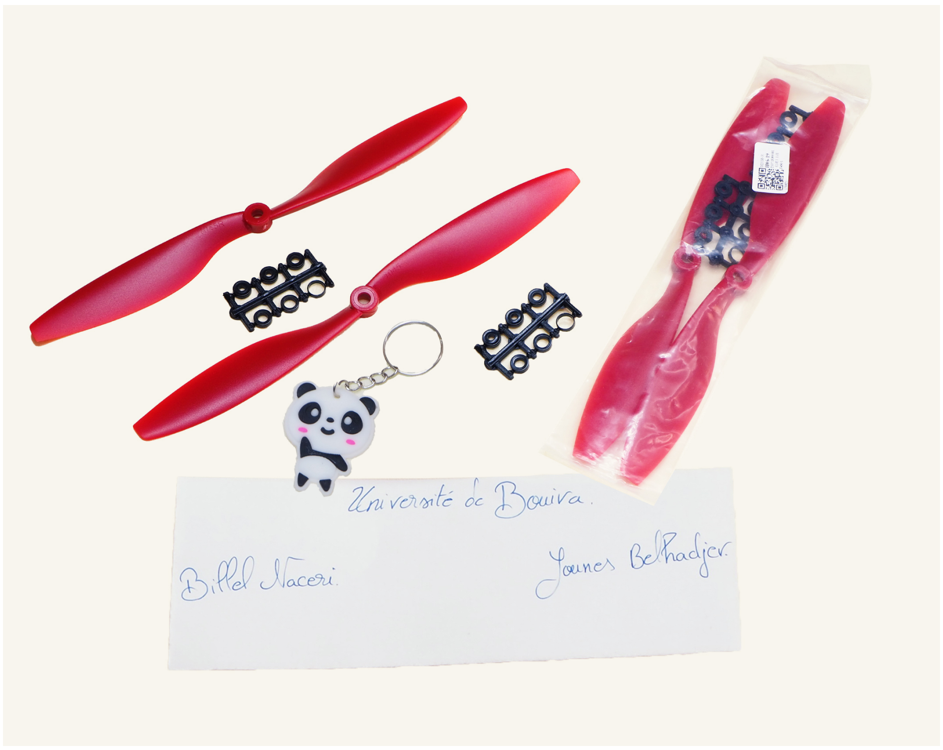


Figure 3.4 – ”1045” propellers

3.2.5 Battery

The quadcopters’ power sources are LiPo batteries. Because of its high energy density and discharge rate, LiPo is used. The nominal voltage (3.7v per cell), cell count in series (indicated as a number followed by a ‘S’) ie 4S = 14.8v, capacity in mAh (ie.1300mAh), and discharge rate or 'C' rating are all used to rate LiPo batteries (ie. 75C) [35].

keeping in mind that our "ESC's" support 3-4s batteries, we ended up buying two batteries ;the first being Youme lipo 3S 5200mAh with nominal voltage of 11.1V , and the second one being turnigy lipo 3S 5000mAh, we chose the 5000mah models specifically as they have the best power-weight ratio.

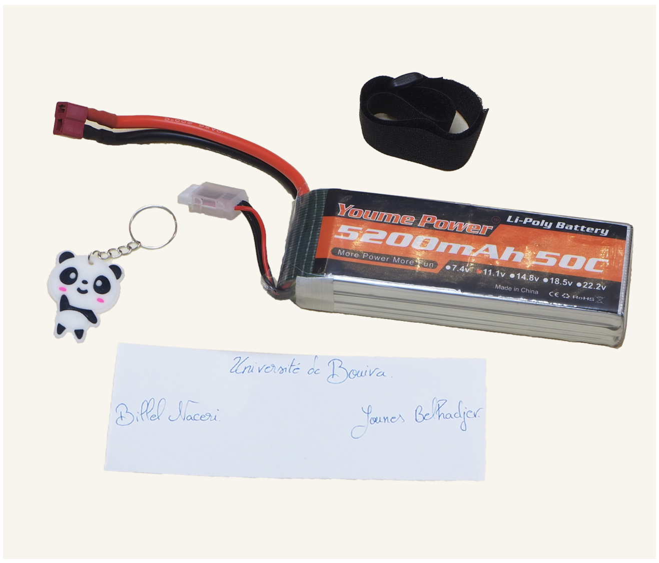


Figure 3.5 – Youme lipo 3S 5200mAh

3.2.6 Flight controller

The Flight Controller (or "FC") is a quadcopter's brain; it includes sensors on board that allow it to understand how the craft is moving. The FC employs algorithms to compute how fast each motor should be spinning based on the data provided by these sensors in order for the craft to behave as the pilot instructs via stick inputs on the TX (Radio Transmitter). The FC will be the main focus of the majority of your quad's wiring. The RX (receiver) must be connected so that the craft can be taught what the pilot wants it to execute. For the motors to carry out the FC commands, each of the ESC signal and ground wires must be connected. BetaFlight OSD was introduced with the debut of BetaFlight OSD (On Screen Display), every component on the quadcopter is highly dependable on the FC, because if there are any time delays at any time the drone will crash [35].

On our build we used the pixhawk 2.4.8 as shown in figure(3.6), This flight controller is based on the rapidly evolving and refined Ardupilot mega or "APM", an open-source project from 3DR robotics. This amazing flight controller allows us to control any type of multirotor vehicle or even turn it into a fully autonomous vehicle; capable of performing a wide range of tasks even programmed GPS missions with way-points with the optional GPS Module, it's a full autopilot capable of autonomous stabilization, way-point based navigation and support for two-way telemetry with radio telemetry modules. It Supports 8 RC channels with 4 serial ports [35].

Various user interfaces are available for programming, even some Apps for smartphones and tablets. The optional telemetry radio allows for two-way radio communication giving us full control and provides live data back to our computer.

Benefits of the Pixhawk system include integrated multi-threading, a Unix/Linux-like programming environment, completely new autopilot functions such as sophisticated scripting of missions and flight behavior, and a custom PX4 driver layer ensuring tight timing across all processes[36].

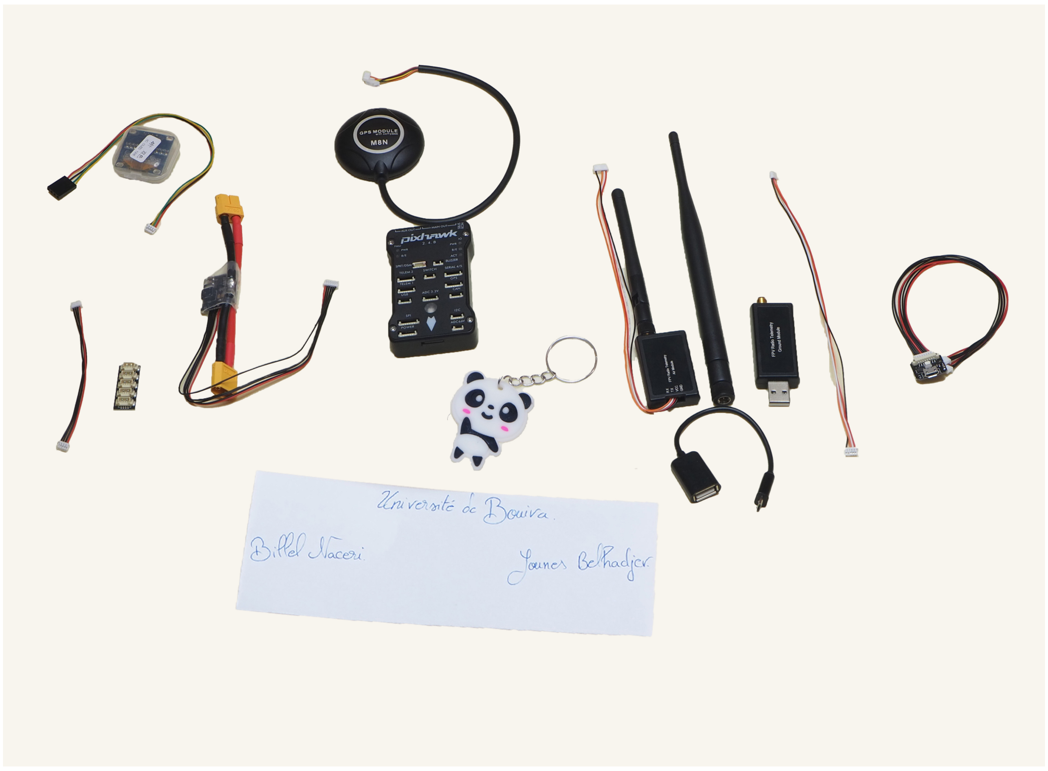


Figure 3.6 – pixhawk 2.4.8

3.2.7 FPV Camera

The pilot can observe the scene from onboard the drone using an FPV camera. There are usually two cameras on an FPV quad, one for real-time video streaming and the other for recording HD footage.

FPV cameras aren't known for their video quality; instead, they're built for WDR (Wide Dynamic Range) and low latency, which are critical in FPV. The capacity of a camera to depict variations in lighting conditions, as well as areas of shade and light, in the same image is referred to as WDR. The length of time it takes for your FPV camera to capture a picture and display it on your screen or in your goggles is known as latency [35].

We used Razer micro 1200 TVL fpv camera from Foxeer which has a great performance in a low light environment. It features a 1200TVL definition with an advanced natural image. The wide voltage range of 4.5V 25V, provides a wide range of compatibility.

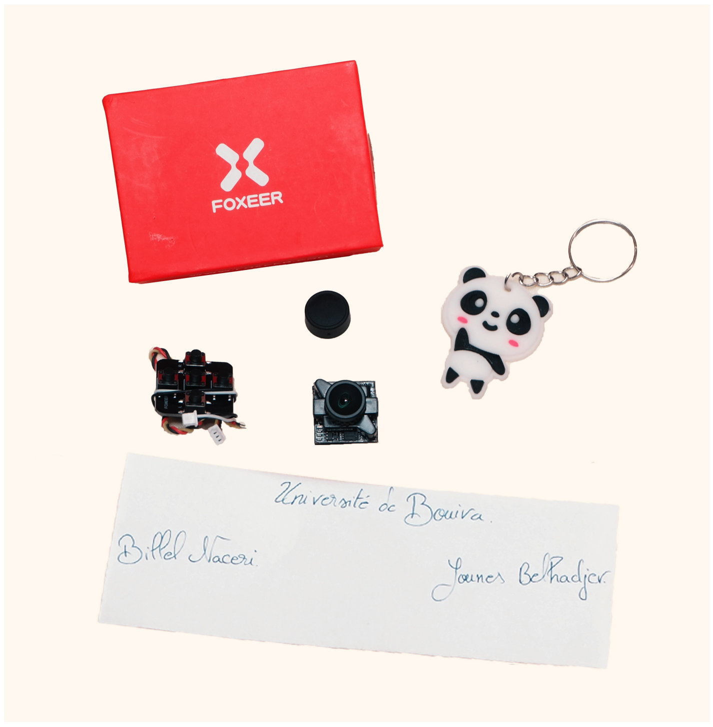


Figure 3.7 – Razer micro 1200 TVL fpv camera from Foxeer

3.2.8 Video transmitter

The video transmitter, often known as the VTX, connects to the FPV camera and sends video to the FPV goggles or monitor. The 5.8GHz frequency is now used by the majority of quadcopters for video transmission. You might discover that your VTX has other features, such as a regulated 5v output for powering your FPV camera. It's important to keep in mind that if you power your VTX without an antenna connected, it could burn out! [35].

The VTX receives a signal from the FPV camera (typically via the FC), which it then broadcasts on one of the 5.8gHz frequency bracket's several channels. Some VTX are powered by 5 volts, while some require more. If your VTX runs on 5v, be aware that it will be active when you connect your FC to USB. When establishing BetaFlight, you need have an antenna connected. Remember that if your VTX requires more than

5 volts, you'll need to attach a battery to set up your channel, band, and output power [35].

In our build we used the RC832, as it's one of the best transmitters, as shown in figure (3.8).

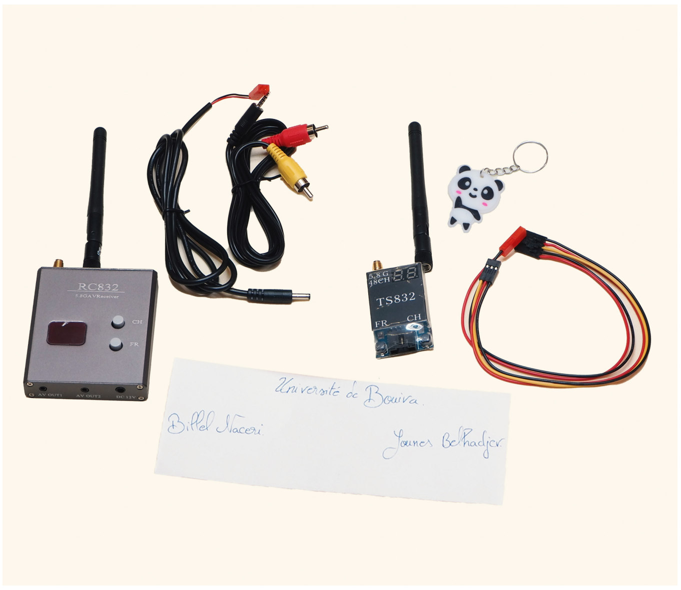


Figure 3.8 – Video transmitter

3.2.9 Companion computer

The companion computer can essentially run high level scripts and communicate commands to the flight controller using python coding [37].

Interestingly enough, we could fly the drone without a companion computer, and even do basic autonomous missions on a simple Pixhawk FC, we used a raspberry pi 4 model B along with pi camera v1.3 as shown in figure (3.9), the main reason of using a raspberry pi is that it unleashes advanced functionality and endless possibilities.

Here are some projects that raspberry pi is capable of:

- Object Recognition with TensorFlow
- Drone Fishing
- Drone Surveillance

- Motion Controlled Drone
- 4G LTE Drone for Limitless Range
- Obstacle Avoidance

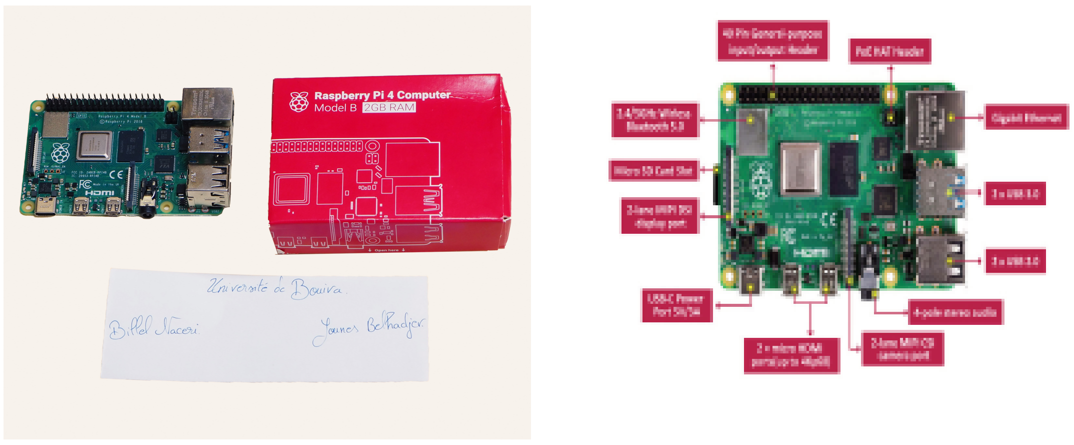
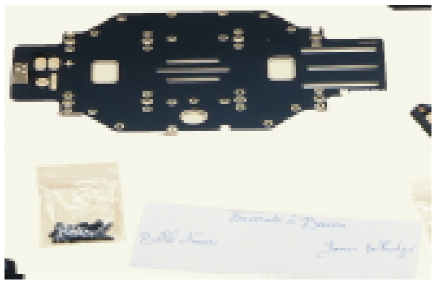


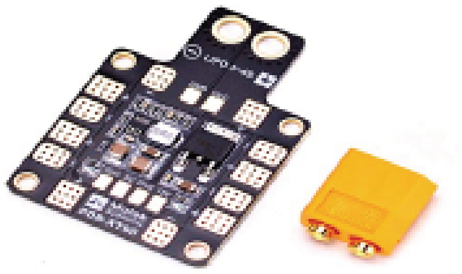
Figure 3.9 – raspberry pi 4

3.2.10 Power distribution board

The battery power lead (i.e. XT60) is usually connected to the PDB (Power Distribution Board). The PDB, as the name implies, distributes power to components at the voltages they require. FCs, ESCs, and other (called AIO or All-In-One) components that provide the same purpose have rendered the use of a PDB obsolete. These components can be linked to battery voltage (called VBAT) and then produce a stable voltage, such as 5v, to power an FPV camera or other components [35]. we used the provided PDB that comes with the s 500 drone frame we also got an additional PDB in case of encountering any problems, the other one being the Matek – Mini Hub PDB which has 5V/12V as shown In figure (3.10)



(a) S500 PDB



(b) extra PDB

Figure 3.10 – Power Distrubition Board

3.2.11 RX (Radio Receiver)

Transmitters (TX) and receivers (RX) are not interchangeable; therefore, you'll need to acquire an RX that works with your TX. For example, a FrSky Taranis transmitter will not operate with a FlySky receiver. These days, you're more than likely to use PPM or a digital Serial protocol, which just require one signal line for all channels, as well as power (3.3v or 5v) and GND. One of the UART terminals on your FC will be connected to the signal line (Flight Controller). Some FCs come with built-in receivers; if you go this route, ensure sure it uses a protocol that is compatible [35].

In our build, we used RadioLink AT9S PRO 2.4GHz 12CH transmitter with R9DS receiver due to:

- Control distance: about 900m ground and 3000m air
- Supports signal strength and voltage monitoring
- Offers vibrating alarm for noisy environment
- Includes hardware to add auto-centering for the left (throttle) joystick
- Supports both ppm and sbus connections

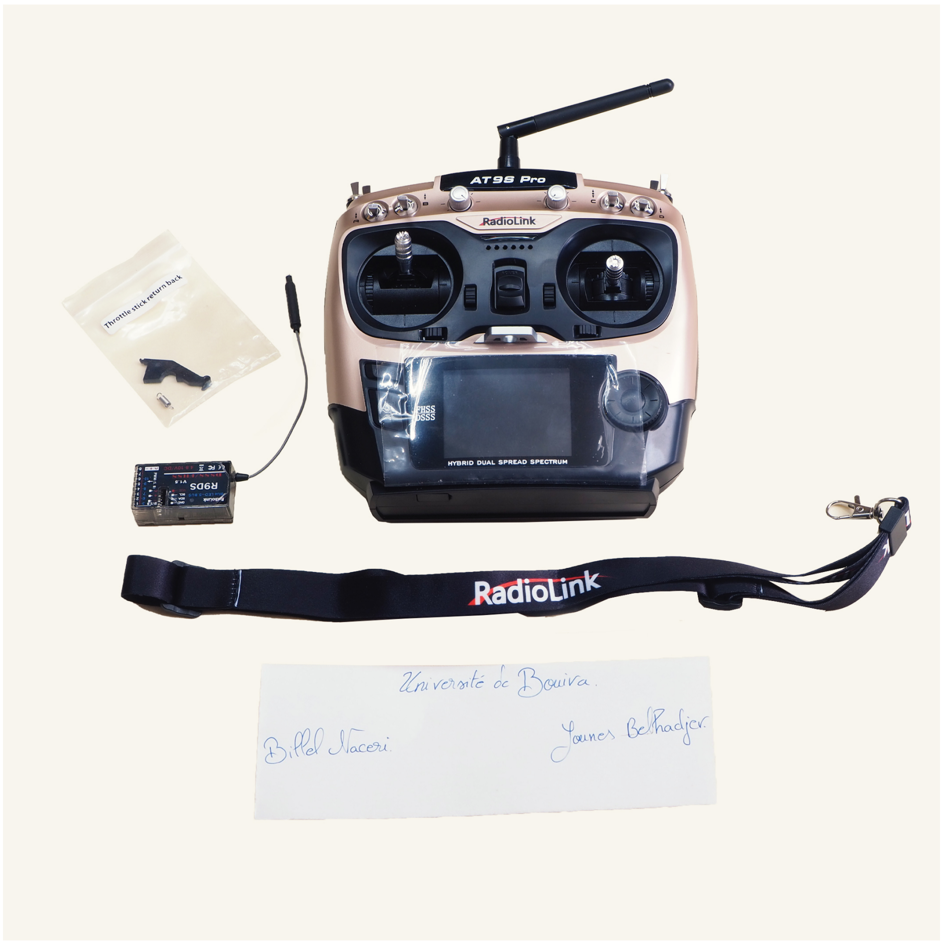


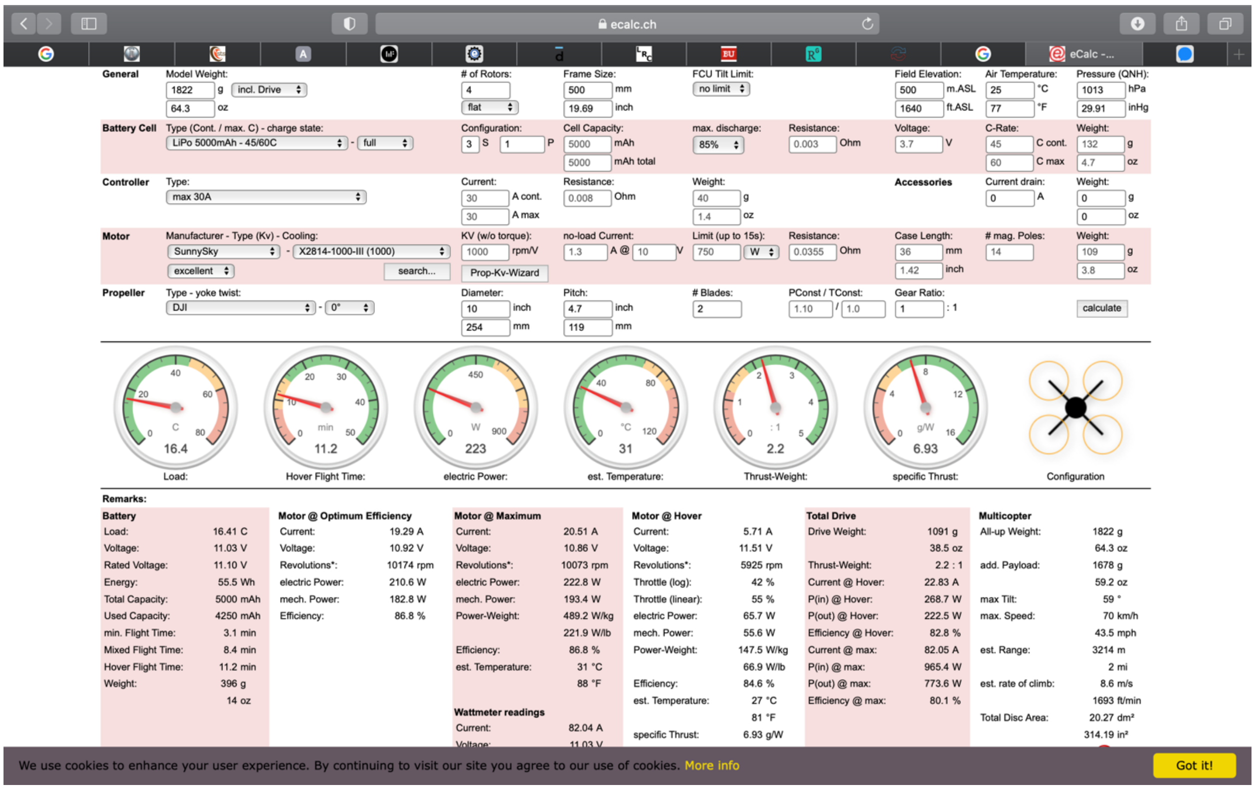
Figure 3.11 – RadioLink AT9S PRO 2.4GHz 12CH

3.2.12 Compatibility check using eCalc

Since 2004 eCalc provides web-based quality services to simulate, calculate, evaluate and design electric brushless motor drives for RC pilots of airplane, multi rotor, UAV, helicopter and EDF jets.

After selecting all our components, we can check the compatibility, estimated flight time and even the maximum payload on eCalc; after entering all of the components here are the results we got.

As the figure (3.12) shows, our quadrotor can get up to 11.2 min of flight time and can add a payload of 1.678 kilograms, based on these statistics our quadrotor will not overheat.



ecalc.ch

General	Model Weight: 1822 g incl. Drive; 64.3 oz	# of Rotors: 4; flat	Frame Size: 500 mm; 19.69 inch	FCU Tilt Limit: no limit		Field Elevation: 500 m.ASL; 1640 ft.ASL	Air Temperature: 25 °C; 77 °F	Pressure (QNH): 1013 hPa; 29.91 inHg
Battery Cell	Type (Cont. / max. C) - charge state: LiPo 5000mAh - 45/60C - full	Configuration: 3 S 1 P	Cell Capacity: 5000 mAh; 5000 mAh total	max. discharge: 85%	Resistance: 0.003 Ohm	Voltage: 3.7 V	C-Rate: 45 C cont.; 60 C max	Weight: 132 g; 4.7 oz
Controller	Type: max 30A	Current: 30 A cont.; 30 A max	Resistance: 0.008 Ohm	Weight: 40 g; 1.4 oz		**Accessories**	Current drain: 0 A	Weight: 0 g; 0 oz
Motor	Manufacturer - Type (Kv) - Cooling: SunnySky - X2814-1000-III (1000); excellent; search...	KV (w/o torque): 1000 rpm/V; Prop-Kv-Wizard	no-load Current: 1.3 A @ 10 V	Limit (up to 15s): 750 W	Resistance: 0.0355 Ohm	Case Length: 36 mm; 1.42 inch	# mag. Poles: 14	Weight: 109 g; 3.8 oz
Propeller	Type - yoke twist: DJI - 0°	Diameter: 10 inch; 254 mm	Pitch: 4.7 inch; 119 mm	# Blades: 2	PConst / TConst: 1.10 / 1.0	Gear Ratio: 1 : 1		calculate

Load:	Hover Flight Time:	electric Power:	est. Temperature:	Thrust-Weight:	specific Thrust:	Configuration
16.4 C	11.2 min	223 W	31 °C	2.2 : 1	6.93 g/W	

Remarks:

Battery		**Motor @ Optimum Efficiency**		**Motor @ Maximum**		**Motor @ Hover**		**Total Drive**		**Multicopter**	
Load:	16.41 C	Current:	19.29 A	Current:	20.51 A	Current:	5.71 A	Drive Weight:	1091 g	All-up Weight:	1822 g
Voltage:	11.03 V	Voltage:	10.92 V	Voltage:	10.86 V	Voltage:	11.51 V		38.5 oz		64.3 oz
Rated Voltage:	11.10 V	Revolutions*:	10174 rpm	Revolutions*:	10073 rpm	Revolutions*:	5925 rpm	Thrust-Weight:	2.2 : 1	add. Payload:	1678 g
Energy:	55.5 Wh	electric Power:	210.6 W	electric Power:	222.8 W	Throttle (log):	42 %	Current @ Hover:	22.83 A		59.2 oz
Total Capacity:	5000 mAh	mech. Power:	182.8 W	mech. Power:	193.4 W	Throttle (linear):	55 %	P(in) @ Hover:	268.7 W	max Tilt:	59 °
Used Capacity:	4250 mAh	Efficiency:	86.8 %	Power-Weight:	489.2 W/kg	electric Power:	65.7 W	P(out) @ Hover:	222.5 W	max. Speed:	70 km/h
min. Flight Time:	3.1 min				221.9 W/lb	mech. Power:	55.6 W	Efficiency @ Hover:	82.8 %		43.5 mph
Mixed Flight Time:	8.4 min			Efficiency:	86.8 %	Power-Weight:	147.5 W/kg	Current @ max:	82.05 A	est. Range:	3214 m
Hover Flight Time:	11.2 min			est. Temperature:	31 °C		66.9 W/lb	P(in) @ max:	965.4 W		2 mi
Weight:	396 g				88 °F	Efficiency:	84.6 %	P(out) @ max:	773.6 W	est. rate of climb:	8.6 m/s
	14 oz					est. Temperature:	27 °C	Efficiency @ max:	80.1 %		1693 ft/min
				Wattmeter readings			81 °F			Total Disc Area:	20.27 dm²
				Current:	82.04 A	specific Thrust:	6.93 g/W				314.19 in²
				Voltage:	11.03 V						

We use cookies to enhance your user experience. By continuing to visit our site you agree to our use of cookies. More info

Got it!

Figure 3.12 – eCalc results

3.3 Assembly and connections

3.3.1 Hardware Setup

The assembly part needs a lot of patience, and some wielding skills. As shown in figure (3.13) ; we made this block diagram for better understanding of the connection of components [36].

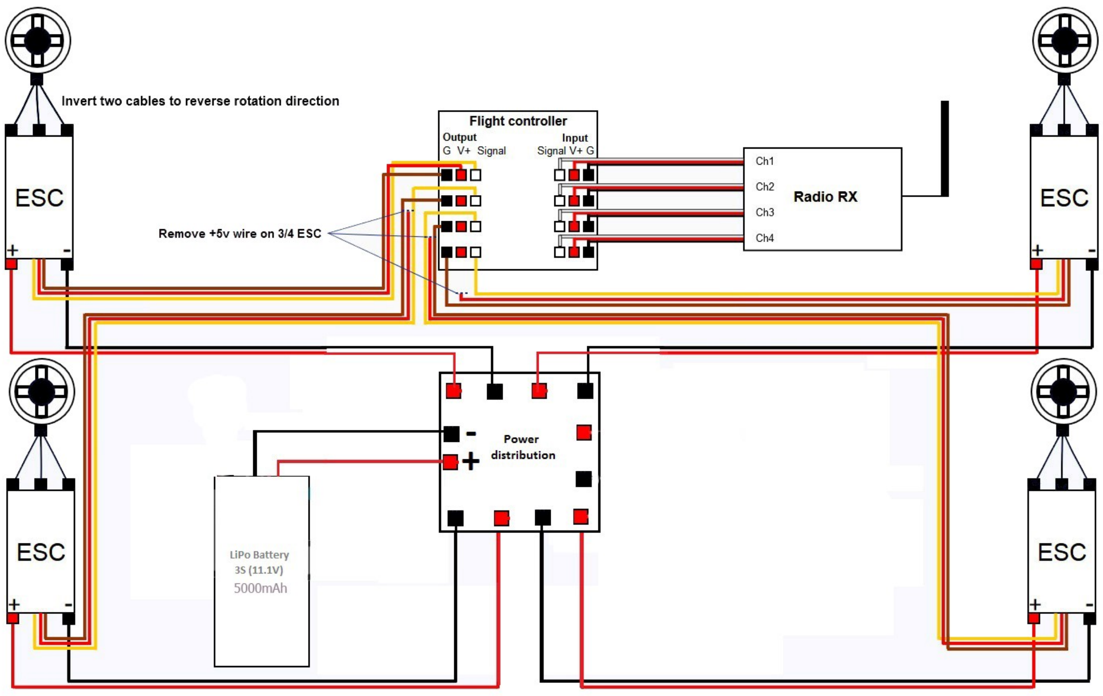


Figure 3.13 – Connections block diagram

3.3.1.1 Assembling primary electronic components ESC's

Due to many factors that affect the drone's performance, it is recommended that the speed controllers be connected on the bottom side of the frame. This is because it will "unload" the upper side of the drone, which is where further components should be put, among other things.

Zip ties are required for a secure fit of the ESC to the frame. ESCs will be properly secured and strapped down when flying this way.
The ESCs are wielded directly to the PDB, as shown in figure (3.14).

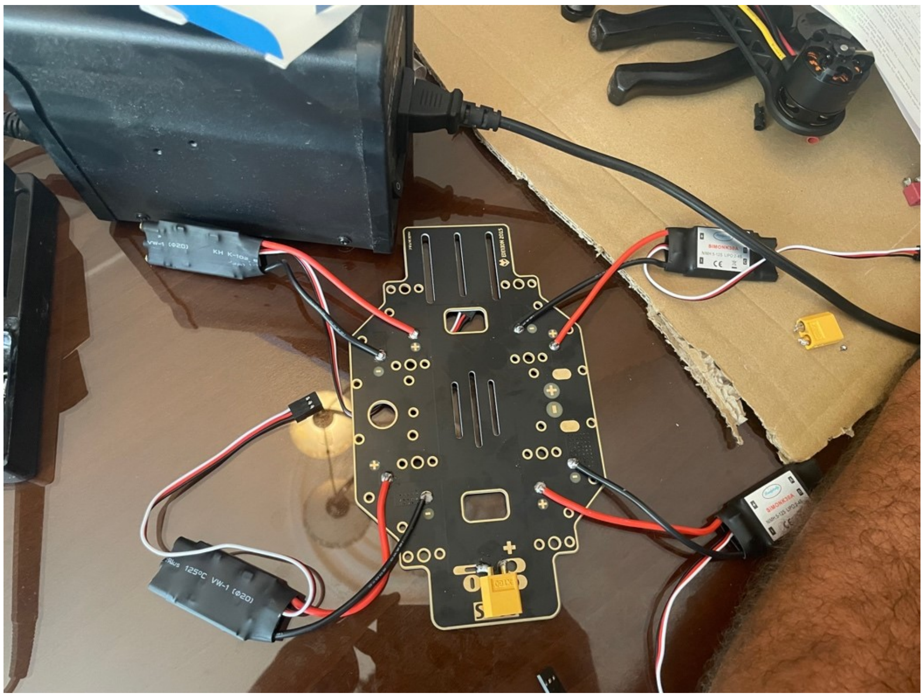


Figure 3.14 – ESCs wielded to the PDB

3.3.1.2 Assembling the motors on the frame

The next step is to drill the holes for the motors in the frame, based on the spacing between the screw's holes on the motors. Another hole should be drilled to allow the clip and shaft of the motor to move freely.

If the motors already came with mountings, we can skip this step. Place the motors in their proper location and secure it to the frame with screws and a screwdriver.

As shown in figure (3.15), our motors fit perfectly in the assembly.

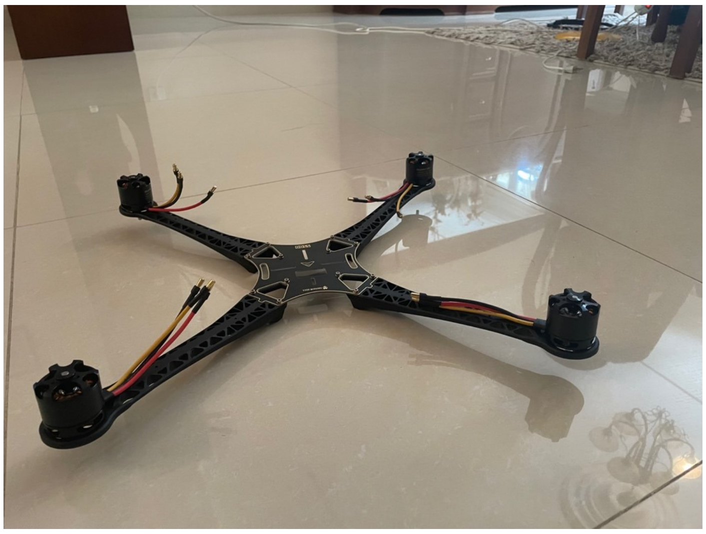


Figure 3.15 – Motors assembly on the upper frame

3.3.1.3 Testing the motors

It's essential to test the motors before mounting the FC, we used our RC to test each motor before mounting them, figure (3.16).

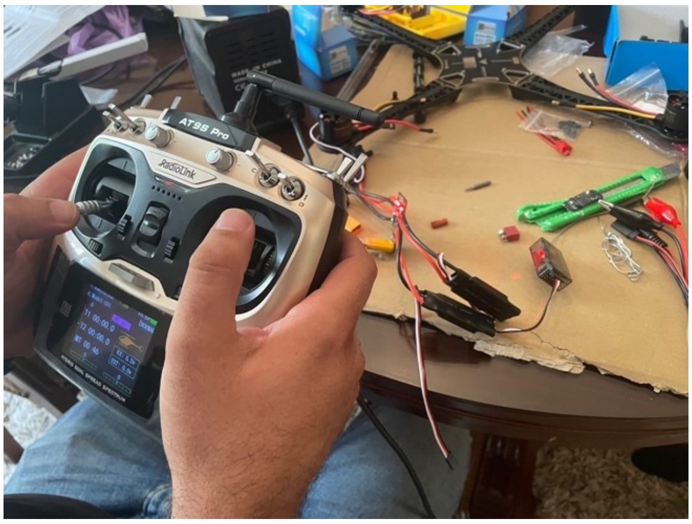


Figure 3.16 – Testing motors via RF

3.3.1.4 Connecting motors and ESCs

We connected the ESCs, motors and propellers to our Pixhawk, we connected the power (+), ground (-), and signal (s) wires for each ESC to the autopilot's main output pins by motor number [36]. As shown in figure (3.17).

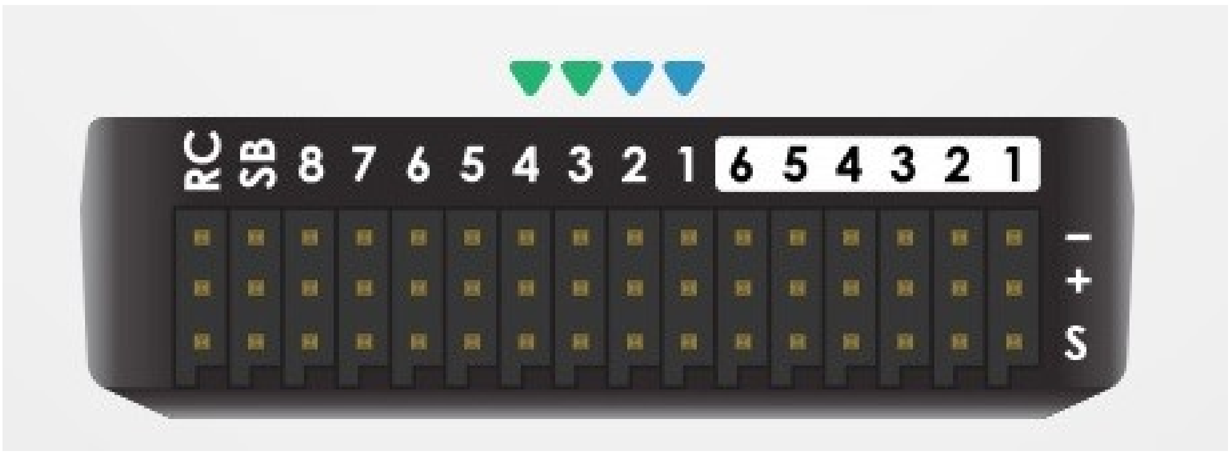


Figure 3.17 – Pixhawk motor entries

We had to respect the motors order for our frame type as shown below in figure (3.18).

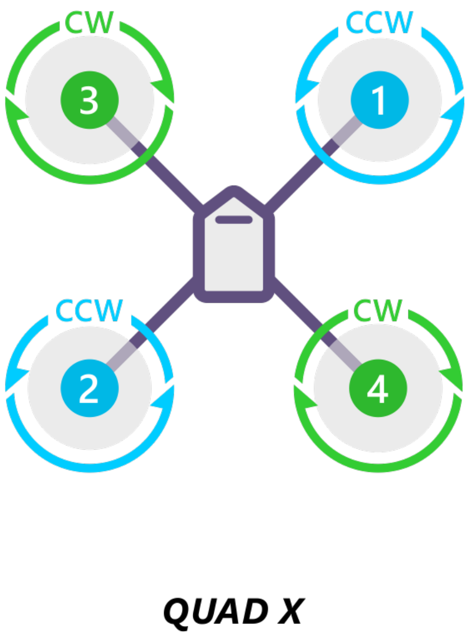


Figure 3.18 – Motors order for 'X' quad

3.3.1.5 Adding the landing gear

When landing your UAV, this gear is essential because it considerably decreases the shock when the drone descends on firm ground. It can be created in a variety of ways; you can also be inventive and make it your own.

You can also utilize alternative flexible but sturdy materials, such as new plastics, or anything else that reduces shock.

In our build, we have got carbon fiber landing gear, the full frame assembly with motors, ESCs and PDB is shown in figure (3.19).

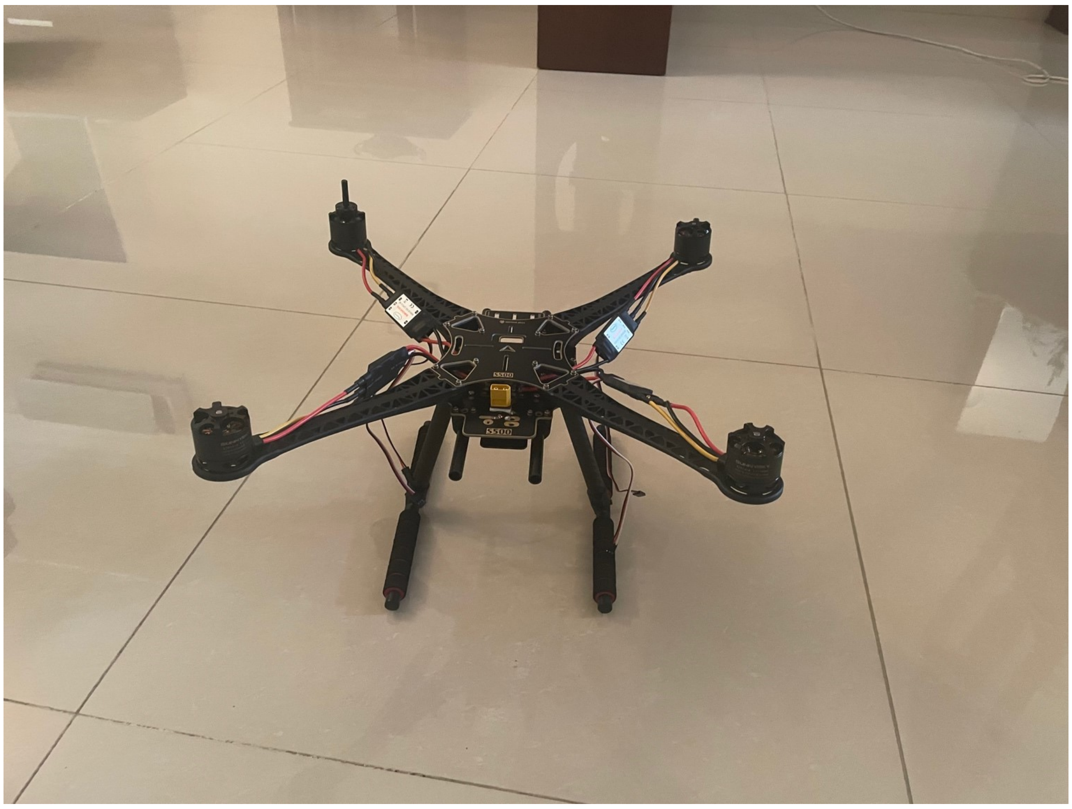


Figure 3.19 – frame assembly with motors, ESCs and PDB

3.3.1.6 Mounting the FC and the Raspberry pi

The flight controller has to be in the center of the quadcopter as it contains a compass and all the connections required for the calculations and orders. We put the anti-vibration printed tool under the raspberry pi-pixhawk combo, the GPS module has to be placed in parallel with the Pixhawk, the buzzer and the safety switch alongside with the R9DS are placed on the top board as well, figure (3.20).

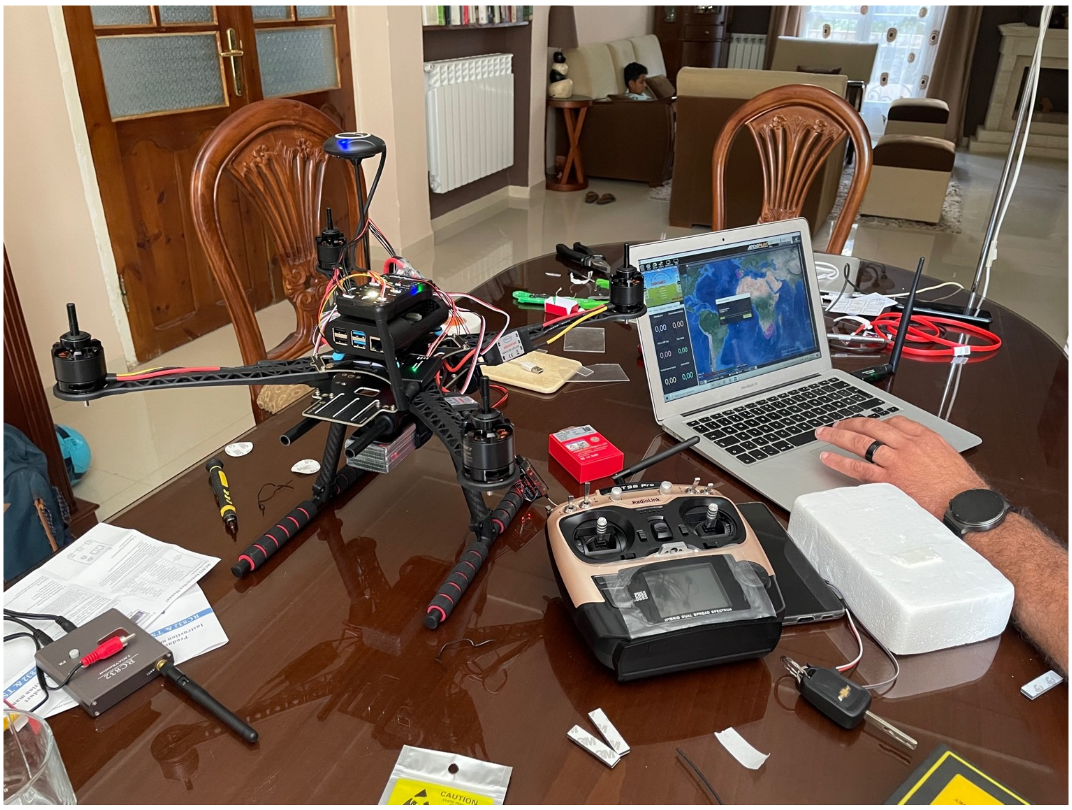
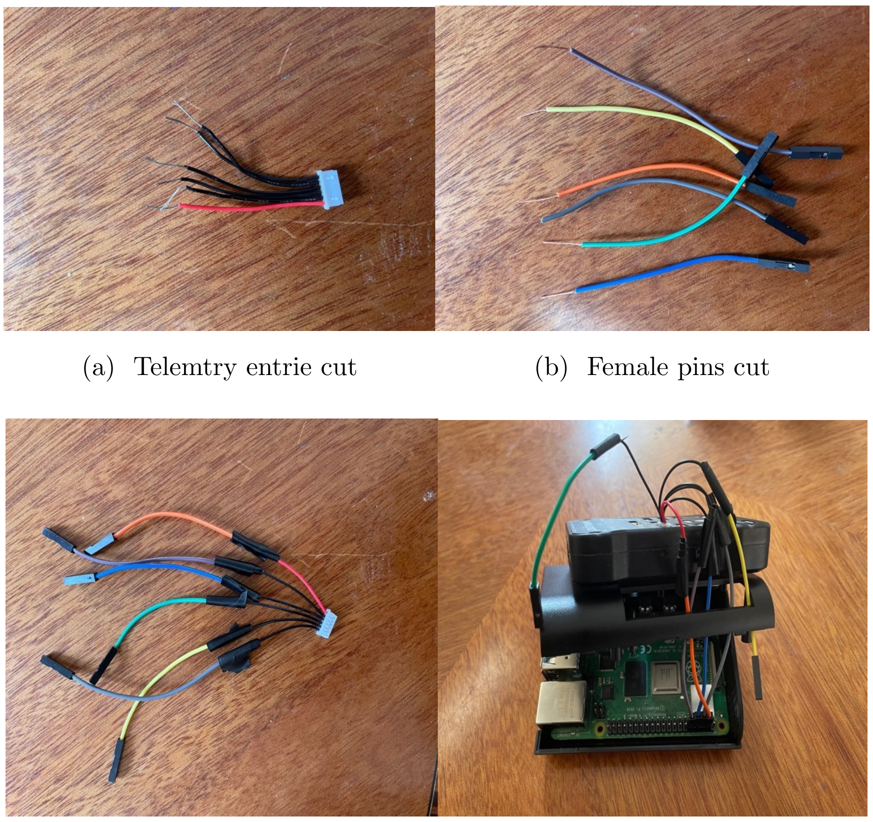


Figure 3.20 – Full assembly of the quad-copter

3.3.1.7 Connecting the FC and the Raspberry pi

To connect a raspberry pi to a drone is a simple task, we use the telemetry number 2 input in order to power up the raspberry pi as well as taking commands from its pins (RX and TX) [37]. Figure (3.21).


(a) Telemtry entrie cut (b) Female pins cut

(c) Assembly of pins to telemetry (d) cable connection

Figure 3.21 – Pixhawk-Raspberry pi connection

NOTE : the programming of the raspberry and its commands will be present in the last chapter

3.3.2 Software Setup

3.3.2.1 Mission planner overview

The ArduPilot open-source autopilot project's Mission Planner is a full-featured ground station program[38].

Michael Oborne's Mission Planner provides a lot more than its name suggests. Here are a few of the highlights:

- Using Google Maps/Bing/Open Street Maps/Custom WMS, enter waypoints, fences, and rally points with a few clicks.
- From the drop-down menus we can select mission commands.

- Download and examine mission log files.
- Configure your vehicle's autopilot settings.
- Interface with a PC flight simulator to create a full software-in-the-loop (SITL) UAV simulator.
- Run its own SITL simulation for all ArduPilot vehicles, with a variety of frame types.

The ground control station for Plane, Copter, and Rover is called Mission Planner. It's only compatible with Windows. For your autonomous vehicle, Mission Planner can be used as a configuration tool or as a dynamic control supplement. Here are a few examples of what Mission Planner can achieve for you:

- Load the firmware (software) into the autopilot board that controls your vehicle (i.e. Pixhawk series).
- Set up, adjust, and optimize your car to its maximum potential.
- With simple point-and-click way-point entry on Google or other maps, you can plan, save, and load autonomous trips into your autopilot.
- Your autopilot's mission logs can be downloaded and analyzed.
- Interface with a PC flight simulator to create a full hardware-in-the-loop UAV simulator.
- With appropriate telemetry hardware you can:
  * Monitor your vehicle's status while in operation.
  * Record telemetry logs which contain much more information the the on-board autopilot logs.
  * View and analyze the telemetry logs.
  * Operate your vehicle in FPV (first person view)

3.3.3 Configuration of mission planner

Once we have installed a ground station on our computer, we connected the autopilot using the micro-USB cable as shown in figure (3.22) below.

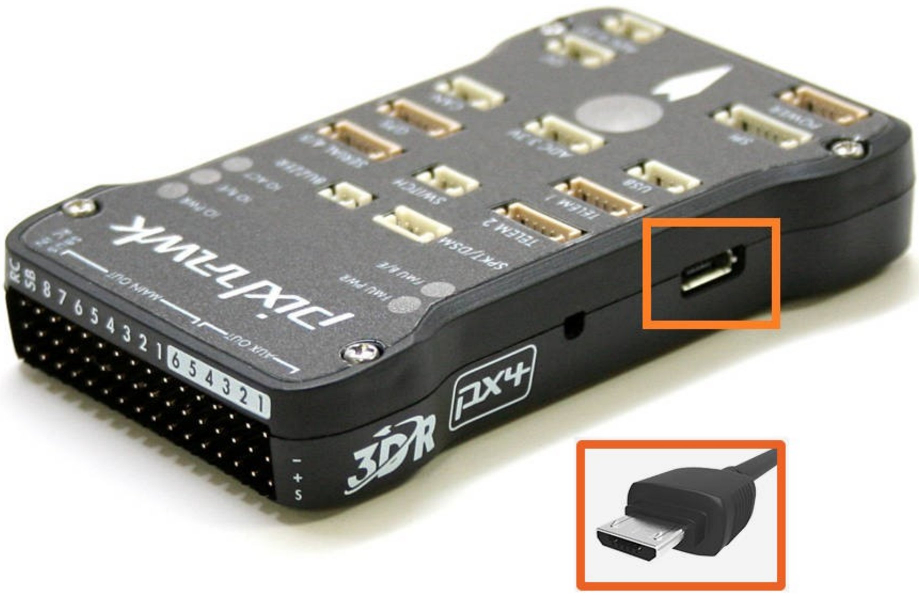


Figure 3.22 – Pixhawk micro-USB port

Windows should automatically detect and install the correct driver software.

In Mission Planner's SETUP — Install Firmware screen select the appropriate icon that matches our frame (i.e. Quad, Hexa). Answer Yes. As shown in figure (3.23). In our case it's the arducopter quad. After installing the latest firmware, with the right

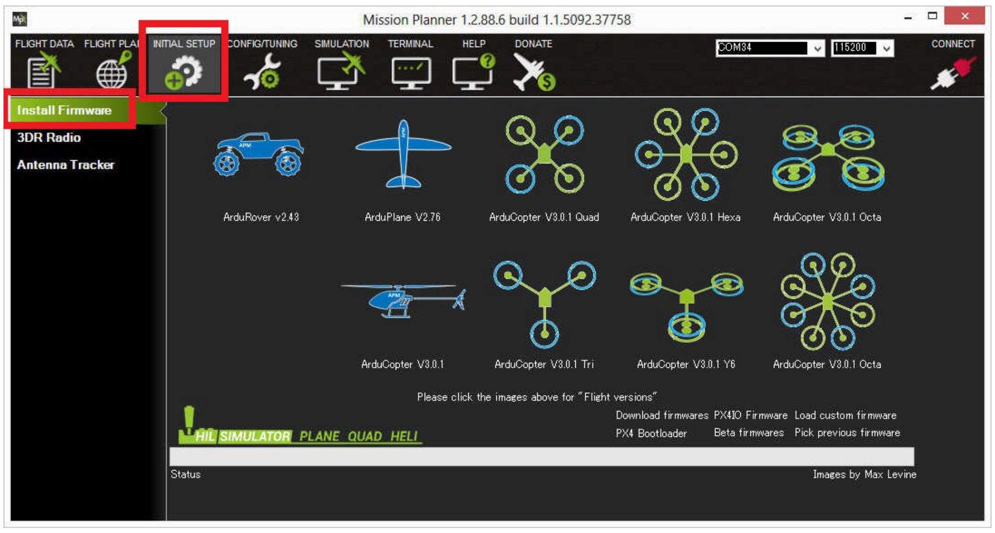


Figure 3.23 – Mission planner GUI

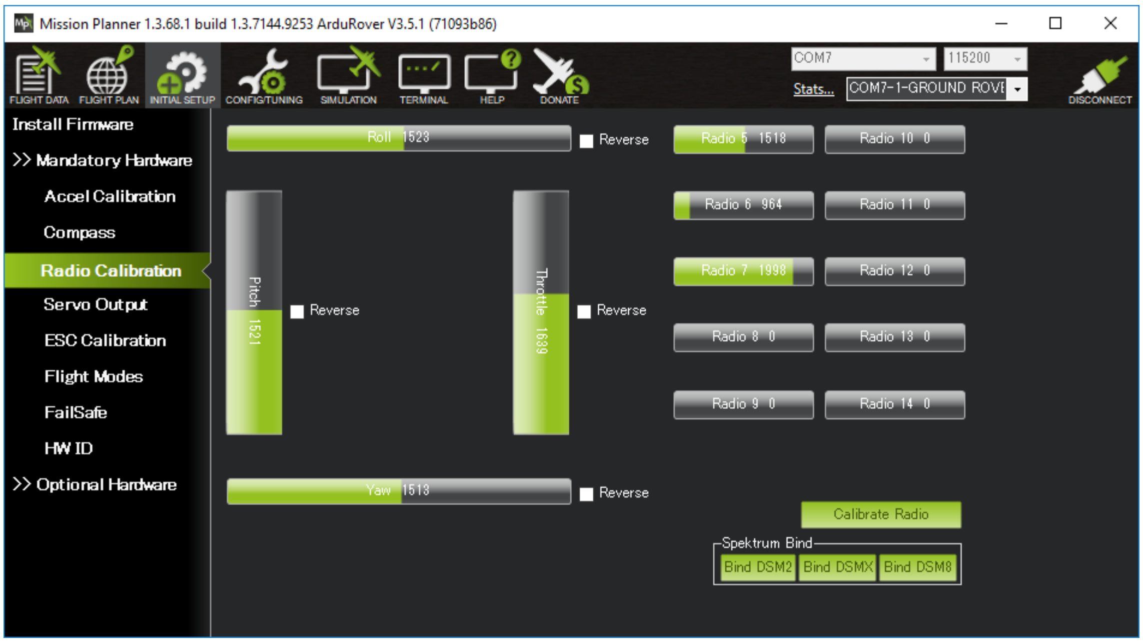
telemetry we can now safely connect our FC to our Pixhawk, now we can see that all the data of our drone imported to mission planner, we can now import our PIDs, get instant gyroscopic view, GPS information and even send it in missions, however before all of this we need to run some calibrations first.

3.3.4 Component's calibration on mission planner

3.3.4.1 Radio control calibration

RC transmitters allow the pilot to set the flight mode, control the vehicle's movement and orientation and also turn on/off auxiliary functions (i.e., raising and lowering landing gear, etc.). RC Calibration involves capturing each RC input channel's minimum, maximum and "trim" values so that ArduPilot can correctly interpret the input[38].

On the Mission Planner we opened INITIAL SETUP — Mandatory Hardware — Radio Calibration screen, now we can calibrate our RC transmitter easily. As shown in figure (3.24)


Mission Planner 1.3.68.1 build 1.3.7144.9253 ArduRover V3.5.1 (71093b86)

COM7 115200 COM7-1-GROUND ROVE

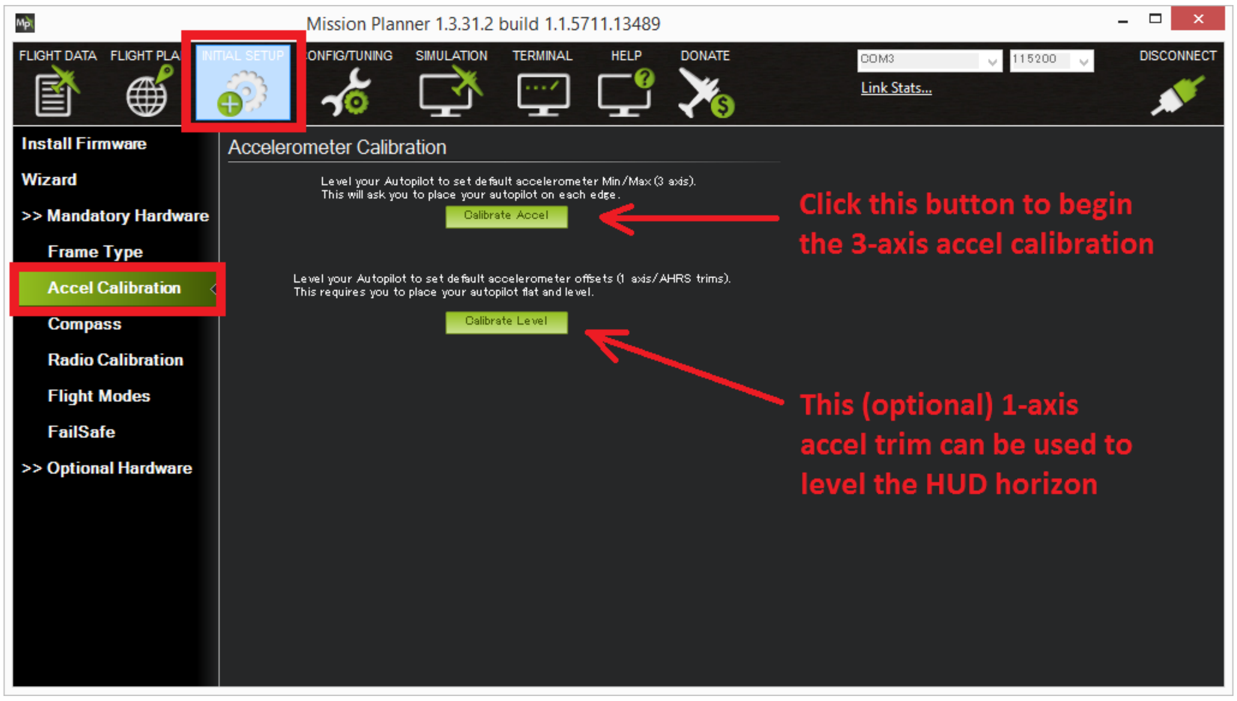
Roll 1523 · Pitch 1521 · Throttle 1639 · Yaw 1513

Radio 5 1518 · Radio 6 964 · Radio 7 1998 · Radio 8 0 · Radio 9 0 · Radio 10 0 · Radio 11 0 · Radio 12 0 · Radio 13 0 · Radio 14 0

Figure 3.24 – Radio Calibration

3.3.4.2 Accelerometer Calibration

Under Setup — Mandatory Hardware, select Accel Calibration from the left-side menu as show in figure (3.25).



Accelerometer Calibration

Level your Autopilot to set default accelerometer Min/Max (3 axis).
This will ask you to place your autopilot on each edge.

Level your Autopilot to set default accelerometer offsets (1 axis/AHRS trims).
This requires you to place your autopilot flat and level.

Click this button to begin the 3-axis accel calibration

This (optional) 1-axis accel trim can be used to level the HUD horizon

Figure 3.25 – Accel Calibration

When we click ‘Calibrate Accel’ the calibration starts, Mission Planner will prompt us to place the quadcopter in each calibration position [38]. Press ‘Calibrate Accel’ key to indicate that the autopilot is in position and then we proceeded to the next orientation. The calibration positions are: level, on right side, left side, nose down, nose up and on its back. As shown in figure (3.26).

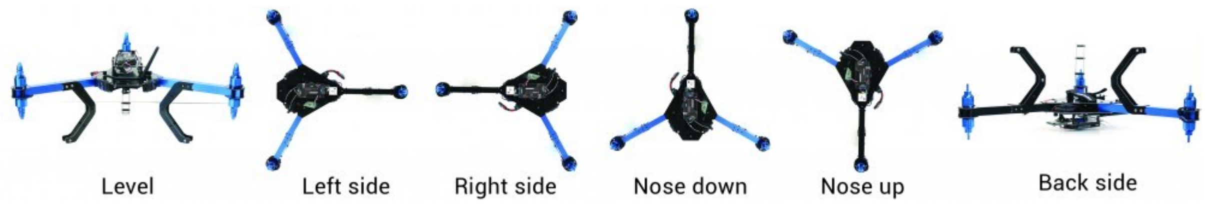
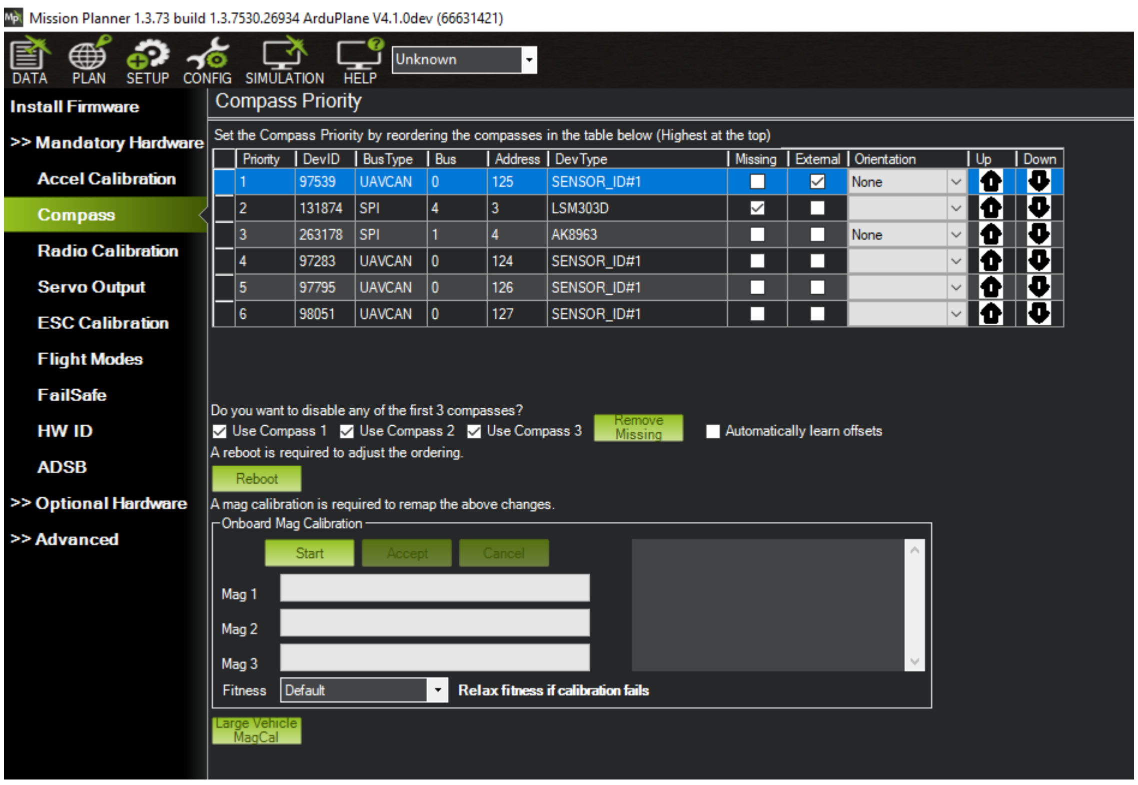


Figure 3.26 – Calibration positions

3.3.4.3 Compass Calibration

It is not necessary to recalibrate the compass when the vehicle is flown at a new location because ArduPilot includes a "world magnetic model" which allows converting the location's magnetic North to true North without recalibrating. In addition, the location's "inclination" is calibrated at startup and then again soon after takeoff. It is important that when compass calibration is done, the vehicle have a good 3D GPS lock, in order to assure the best setup. If necessary, move outdoors in order to get a good 3D GPS lock before doing the compass calibration [38].

Under SETUP— Mandatory Hardware select Compass, our build has 2 compasses; one on the Pixhawk and the other on the GPS module, select the first 2 compasses and we start the onboard mag calibration, once the calibration starts we had to move our quadcopter in place in rotational and upside movements in order for the calibration to be completed, as the vehicle is rotated the green bars should extend further and further to the right until the calibration completes. figure (3.27).



Compass Priority

Set the Compass Priority by reordering the compasses in the table below (Highest at the top)

Priority	DevID	BusType	Bus	Address	DevType	Missing	External	Orientation	Up	Down
1	97539	UAVCAN	0	125	SENSOR_ID#1	☐	✓	None	⇧	⇩
2	131874	SPI	4	3	LSM303D	✓	☐		⇧	⇩
3	263178	SPI	1	4	AK8963	☐	☐	None	⇧	⇩
4	97283	UAVCAN	0	124	SENSOR_ID#1	☐	☐		⇧	⇩
5	97795	UAVCAN	0	126	SENSOR_ID#1	☐	☐		⇧	⇩
6	98051	UAVCAN	0	127	SENSOR_ID#1	☐	☐		⇧	⇩

Do you want to disable any of the first 3 compasses?

✓ Use Compass 1 ✓ Use Compass 2 ✓ Use Compass 3 Remove Missing ☐ Automatically learn offsets

A reboot is required to adjust the ordering.

Reboot

A mag calibration is required to remap the above changes.

Onboard Mag Calibration

Start Accept Cancel

Mag 1

Mag 2

Mag 3

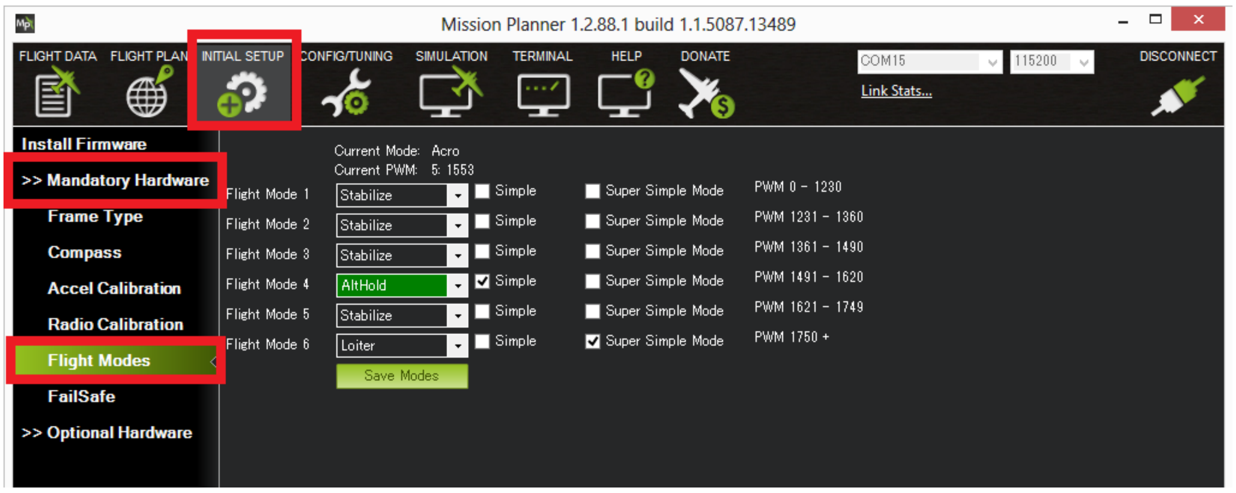
Fitness: Default **Relax fitness if calibration fails**

Large Vehicle MagCal

Figure 3.27 – Compass calibration

3.3.4.4 RC Transmitter Flight Mode Configuration

The mapping between switch position and flight mode is set in the Mission Planner Flight Mode screen. We can set up the flight modes available on the transmitter, The flight mode channel is the input radio channel that ArduPilot monitors for mode changes. In our build we chose to put 'Stabilize', 'alt hold' and 'RTL' as our flight modes as those modes suit our uses of our quadcopter [38]. Figure (3.28)



Current Mode: Acro
Current PWM: 5: 1553

Flight Mode 1	Stabilize	☐ Simple	☐ Super Simple Mode	PWM 0 - 1230
Flight Mode 2	Stabilize	☐ Simple	☐ Super Simple Mode	PWM 1231 - 1360
Flight Mode 3	Stabilize	☐ Simple	☐ Super Simple Mode	PWM 1361 - 1490
Flight Mode 4	AltHold	✓ Simple	☐ Super Simple Mode	PWM 1491 - 1620
Flight Mode 5	Stabilize	☐ Simple	☐ Super Simple Mode	PWM 1621 - 1749
Flight Mode 6	Loiter	☐ Simple	✓ Super Simple Mode	PWM 1750 +

Figure 3.28 – Flight Modes

3.3.4.5 Electronic Speed Controller (ESC) Calibration

Electronic speed controllers are responsible for spinning the motors at the speed requested by the autopilot. Most ESCs need to be calibrated so that they know the minimum and maximum PMW values that the flight controller will send, however it's always recommended to manually calibrate the ESCs to prevent any inconvenience, the steps to calibrate ESCs are as follows:

- Connect to the autopilot from a ground station such as the Mission Planner and set the ESC_CALIBRATION parameter to 3
- Disconnect the battery and USB cable so the autopilot powers down
- Connect the battery The arming tone will be played (buzzer attached), press the safety button (attached the Pixhawk telemetry) until it displays solid red You will

hear a musical tone then two beeps

- A few seconds later you should hear a number of beeps (one for each battery cell you're using) and finally a single long beep indicating the end points have been set and the ESC is calibrated Disconnect the battery and power up again normally and test as described below

3.4 Conclusion

Now that everything is set up, we can test our quadcopter, however an implementation of our desired PIDs (roll, pitch and yaw) is required, in the next chapter as we simulate our mathematical model, we will be able to implement our PIDs into the FC and our quadcopter will be ready for a stable flight.

Chapter 4

Dynamic Modeling and Simulation Using MATLAB and Simulink

4.1 Introduction

Quad copters are becoming more popular as a result of their small size and good mobility, and they have a wide range of uses. A quadcopter's dynamics are highly nonlinear. Furthermore, with six degrees of freedom and four control inputs, it is an under-actuated system. The control inputs that determine the vehicle's motion are the thrust and torques required for tilting the quadcopter. The thrust and torque are generated by varying the rotor speeds. The thrust created by the quadcopter's rotor blades is always directed in the direction of the quadcopter's central axis. As a result, the quadcopter's axis should be tilted with respect to the vertical to produce propulsion in a specific direction. As a result, a quadcopter's translational motion is linked to its angular orientation, making quadcopter dynamics and control very complex [50].

4.2 Mathematical model

4.2.1 Introduction

A discussion of notation is required before moving on to the mathematical model. Due to the obvious complexity of a system with six degrees of freedom, numerous notation systems have been created and are required to properly express the essential variables. An example of the notation we've selected is shown below:

$$ {}^{b}\dot{V}^{b}_{CM|i} \tag{4.1}$$

The basis variable in this case is linear acceleration, or. The variable also has two superscripts and one subscript, as you can see, to further define what we're talking about. The top left superscript, b, indicates that the derivative was performed in the body frame of reference, while the top right superscript, indicates that the acceleration is expressed in terms of body frame vector components, and the subscript, CM|i, indicates that this variable refers to the inertial frame's center of mass [50].

The coordinate system that is used is another key part of the math model. As you progress through your model, the model and conventions you choose will become increasingly critical, so be sure to keep them in mind and well-documented. Depending on whether you choose a plus ("+") or an X ("X") arrangement, the coordinate system will change, as described below Figure (4.1) :

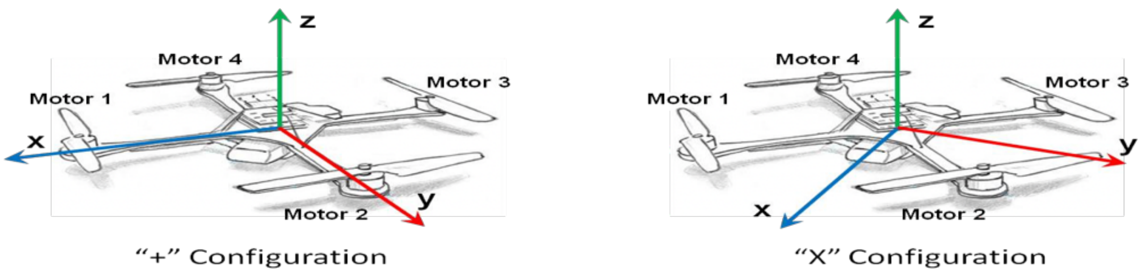


Figure 4.1 – Plus and X configuration diagram

The plus configuration, as shown in Fig. (4.1), consists of the X axis running along the arm of motor 1 (which, by convention, spins counter-clockwise from above), the Y axis running along the arm of motor 2 (spinning in the opposite direction as the adjacent motors), and the Z axis pointing upward. The value represents the distance between a motor and its rotation axis, and it should be the same for all motors. If you use an x configuration, which is defined as a 45-degree rotation in the X-Y plane in the positive yaw direction with the X axis lying between motors 1 and 2, this value will change. The x axis is believed to be the positive forward direction for vehicle movement in either design. Our rotation conventions are presented in Figure (4.2) below for clarity .

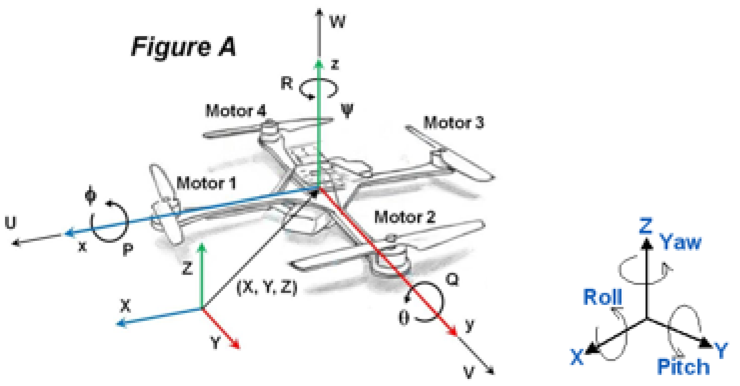


Figure 4.2 – Axis labels and conventions

4.2.2 Mass Moment of Inertia Matrix

The inertia matrix is an important component of the system. The inertia matrix is vital to the system's flight dynamics since it describes the quadcopters mass moment of inertia across the predefined axes. To populate the needed inertia matrix, you can compute the mass moment of inertia across the X, Y, and Z axes with some approximations. The Mass Moment of Inertia documentation goes into much information about this process [50]. The inertia matrix will look like this once you've determined it using either the "+" or "X" configuration:

$$J^b = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix} (MassMomentofInertiaMatrix) \tag{4.2}$$

J^b denotes the quadcopter's inertia in relation to the body frame, while J_{xx} , J_{YY} , and J_{zz} denote the quadcopter's inertia along each axis. The matrix is diagonal due to the system's symmetry and will be same in either a "+" or "X" configuration. Due to the need to invert the matrix for use in the angular velocity state equation, the diagonal form of the matrix is convenient.

4.2.3 Thrust Coefficient

The thrust of the motors is the driving force behind all quadcopter maneuvers, making control design and modeling critical. The thrust generated by a single motor/prop system can be computed using the formula :

$$T = C_T \rho A_r r^2 \omega^2 \tag{4.3}$$

Where C_T is the rotor's thrust coefficient, ρ is the density of air, A_r is the cross-sectional area of the propeller's rotation, r is the rotor's radius, and ω is the rotor's angular velocity [50]. A lumped parameter approach can be utilized to ease the characterization process for simple flight modeling :

$$T = C_T \omega^2 (ThrustCoefficientRelation) \tag{4.4}$$

The lumped parameter thrust coefficient for the individual motor/prop system is indicated by C_T . The motor/prop thrust produces a force in the positive Z direction that is perpendicular to the X-Y plane of the body frame.

4.2.4 Torque Coefficient

The torque force of the motor/prop system must also be determined in order to understand the motor effect on yaw, which may be done in a similar manner to the thrust testing. The following is the lumped parameter equation :

$$Q = C_Q \omega^2 (TorqueCoefficientRelation) \tag{4.5}$$

The torque created by the motor is Q, and the torque coefficient for the motor/prop system is C_Q .This torque provides a force that acts to yaw the system about the z-axis [50].

4.2.5 Initial Matrix Construction

The offered data analysis applications can assist you in calculating these coefficients for characterizing your system after you have performed a variety of tests with each of the test stands. With this information, we can design a matrix like the one below to describe the system's thrusts and torques.

$$\begin{bmatrix} \sum T \\ \Im\phi \\ \Im\theta \\ \Im\psi \end{bmatrix} = \begin{bmatrix} C_T & C_T & C_T & C_T \\ 0 & d_+C_T & 0 & -d_+C_T \\ -d_+C_T & 0 & d_+C_T & 0 \\ -C_Q & C_Q & -C_Q & C_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} ("+"configuration) \tag{4.6}$$

It should be noted that d, is just the distance between the motors and their respective axes of rotation, where d_+ is the arm length from the quadcopter hub center to the motor/prop, all of the current values have been explained thus far [50].

If you're utilizing an x configuration,d_x can be calculated using $d_+ \sin{(45)}$, which is the distance between the motor/prop and the body's axes of rotation. As a result, this configuration adjustment has no influence on C_Q , whereas the effect of C_T will be divided across all four motors for both pitch and roll.

$$\begin{bmatrix} \sum T \\ \Im\phi \\ \Im\theta \\ \Im\psi \end{bmatrix} = \begin{bmatrix} C_T & C_T & C_T & C_T \\ -d_XC_T & d_XC_T & d_XC_T & -d_XC_T \\ -d_XC_T & -d_XC_T & d_XC_T & d_XC_T \\ -C_Q & C_Q & -C_Q & C_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} (\backslash X"configuration) \tag{4.7}$$

4.2.6 Throttle Command Relation

The coefficients of thrust and torque are based on a relationship with the motors' RPM rather than being directly calculated by the control system, which is a significant factor for control purposes (such as throttle command). As a result, a linear regression is required to convert throttle command values (measured in percent throttle) to RPM values. For this reason, the following regression was created :

$$\omega_{ss} = (Throttle\%)C_R + b \tag{4.8}$$

The projected steady-state motor RPM ω_{ss} , the throttle percentage command is Throttle %, the throttle percent to RPM conversion coefficient is C_R , and the y-intercept of the linear regression relationship is b. The given data analysis application can be used to perform linear regression, allowing your controller to use the right coefficients specified by your motor tests for optimal accuracy and realism.

4.2.7 Gyroscopic Forces

Before we generate our moment matrix, we must account for one more set of forces: the forces resulting from gyroscopic precession. Gyroscopic precession is a phenomenon that occurs when the axis of rotation of a rotating body is shifted, and the effects are often counterintuitive to people who are inexperienced with them. The inertia of each motor's rotating components (J_m), the rolling and pitching rates (P and Q), and the speed of each motor/prop system (ω_i). all influence the gyroscopic forces on the body. The motors' gyroscopic torques for pitch and roll action are illustrated below :

$$\Im_{\phi gyro} = J_m Q \frac{\pi}{30}(\omega_1 - \omega_2 + \omega_3 - \omega_4) \tag{4.9}$$

$$\Im_{\theta gyro} = J_m P \frac{\pi}{30}(-\omega_1 + \omega_2 - \omega_3 + \omega_4) \tag{4.10}$$

The $\pi/30$ term corresponds to the transition from RPM to radians that must occur for the gyroscopic force to be calculated.

4.2.8 Final Matrix Construction

We may reorganize the equations in matrix form for our simulation purposes by adding these motor/prop forces to the appropriate terms. For a "+" configuration, the resulting matrix will account for the above aerodynamic, gyroscopic, and thrust moments caused by the quadcopter's motor/prop systems :

$$M^b_{A,T} = \begin{bmatrix} d_+ C_T \omega_2^2 - d_+ C_T \omega_4^2 + J_m Q(\frac{\pi}{30})(\omega_1 - \omega_2 + \omega_3 - \omega_4) \\ -d_+ C_T \omega_2^1 - d_+ C_T \omega_3^2 + J_m P(\frac{\pi}{30})(-\omega_1 + \omega_2 - \omega_3 + \omega_4) \\ -C_Q \omega_1^2 + C_Q \omega_2^2 - C_Q \omega_3^2 + C_Q \omega_4^2 \end{bmatrix} \tag{4.11}$$

$M^b_{A,T}$ indicates the moments present in the body frame as a function of the system's aerodynamics, thrusts, and torques. Gravity and the lift of the quadcopter's rotors both exert forces on the quadcopter's body. The lift force is calculated as follows :

$$F^b_{A,T} = \begin{bmatrix} 0 \\ 0 \\ C_T(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \end{bmatrix} \tag{4.12}$$

$F^b_{A,T}$ refers to the forces acting in the body frame on the quadcopter due to aerodynamics and thrust (assumed oriented strictly in the positive z direction). It should be noted that while we say there are acting aerodynamic forces, it is assumed that the static thrust and torque tests capture the elements of aerodynamics that we are interested in. Additional effects (such as blade flapping, frame aerodynamic drag, etc.) could be added to the model after additional research and testing.

4.2.9 State Equations

The state equations that define the dynamics model are next. The Angular Velocity State Equation is the first thing we'll go over [50].

$$^b\dot{\omega}^b_{b|i} = (J^b)^{-1}[M^b_{A,T} - \Omega^b_{b|i} J^b \omega^b_{b|i}] = \begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} (AngularVelocityStateEquation) \tag{4.13}$$

This equation takes into account the quadcopter's inertia, angular velocity, and the moments applied by the motor/prop systems to represent the change in roll (P), pitch (Q), and yaw (R) rates.$^b\dot{\omega}^b_{b|i}$ is the angular acceleration across each axis in the body frame with respect to the inertial frame, and can also be written as:

$$^b\dot{\omega}^b_{b|i} = \begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} \tag{4.14}$$

Having already the inertia matrix and moment matrices, we move on to $^b\Omega^b_{b|i}$ which is a cross-product matrix for rotational velocity. The form of this matrix is shown below:

$$^b\Omega^b_{b|i} = \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & p & 0 \end{bmatrix} \tag{4.15}$$

Here, P, Q, and R are again the rotation rates about the X, Y, and Z axis, respectively. The $\omega^b_{b|i}$ term is the rotational velocity of the quadcopter body within the body frame and is defined directly by P, Q, and R.

$$\omega^b_{b|i} = \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \tag{4.16}$$

The Euler Kinematic Equation is the second state equation defined, and it allows us to calculate the rate of change of the Euler angles in the inertial frame:

$$\Phi = H(\Phi)\omega^b_{b|i} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{4.17}$$

however, we'll go over rotation matrices before we get into this equation. An aircraft's rotation is described as a rotation around the z-axis (yaw), then a rotation around the y-axis (pitch), and finally a rotation around the x-axis, according to the aerospace rotation sequence (roll). Each rotation is done in a single plane and in a right-handed method. A composite rotation matrix can be formed using these three rotations to translate the motion of the aircraft from the body frame to a new reference frame. Matrix multiplication can be used to find the resulting rotation matrix, which translates rotations from the body frame to the inertial frame. The letters s, c, and t, respectively, stand for sine, cosine, and tangent functions.

$$u^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & s(\phi) \\ 0 & -s(\phi) & c(\phi) \end{bmatrix} \begin{bmatrix} c(\theta) & 0 & -s(\theta) \\ 0 & 1 & 0 \\ s(\theta) & 0 & c(\theta) \end{bmatrix} \begin{bmatrix} c(\psi) & s(\psi) & 0 \\ -s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} u^i \tag{4.18}$$

Following through with the matrix multiplication yields the rotation matrix from the inertial to the body frame using the aerospace rotation sequence :

$$C_{b|i} = \begin{bmatrix} c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\ (-c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi)) & (c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi)) & s(\phi)c(\theta) \\ (s(\phi)s(\psi) + c(\phi)s(\theta)c(\psi)) & (-s(\phi)c(\psi) + c(\phi)s(\theta)s(\psi)) & c(\phi)c(\theta) \end{bmatrix} \tag{4.19}$$

(ZYX Sequence Rotation Matrix)

When it comes to solving the velocity and position state equations, this rotation matrix is crucial. The scope of this document does not allow for a comprehensive discussion of rotation matrices. The angular velocity of the aircraft in the body frame can be connected to changes in angle rotation using consecutive rotation matrices, as shown below, where the C matrices of ϕ and θ are those from u^b .

$$\omega^b_{b|i} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + C_\phi \left(\begin{bmatrix} 0 \\ \dot{\phi} \\ 0 \end{bmatrix} + C_\theta \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \right) \tag{4.20}$$

Performing matrix multiplication and addition and taking the derivative the Euler Kinematic Equation can be found (details not given here):

$$\Phi = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & t(\theta)S(\phi) & t(\theta)c(\phi) \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi)/c(\theta) & c(\phi)/c(\theta) \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} = H(\Phi)\omega^b_{b|i} \tag{4.21}$$

(Euler Kinematic Equation)

While this method is effective, it has one significant flaw: when is equal to 90^o, a singularity develops. As a result, if the aircraft's pitch approaches or exceeds 90 degrees, the simulation's accuracy and numerical stability may be jeopardized. This will not be an issue for most users, given the simulation's minimal control design aims. However, there are various ways to circumvent this problem, including employing quaternions in the simulation, and motivated users may opt to adapt the simulation to use this or another method to avoid the singularity.

The Velocity State Equation, which explains the acceleration of the rigid body quadcopter model's center of mass depending on the forces and accelerations operating on the body, is the next state equation we'll look at :

$${}^b\dot{V}^b_{CM|i} = (\frac{1}{m})F^b_{A,T} + g^b - \Omega^b_{b|i}\omega^b_{CM|i} = \begin{bmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \end{bmatrix} \tag{4.22}$$

(Velocity State Equation)

Here, $b\dot{V}^b_{CM|i}$ is the linear acceleration of the center of mass in the body frame with respect to the inertial frame. The variable m is the total mass of the quadcopter, while g^b is the acceleration of gravity translated to act in the body frame by the rotation matrix $C_{b|i}$.

$$g^b = C_{b|i} g^i \tag{4.23}$$

Using these equations you can find the linear acceleration of the quadcopter in the X, Y, and Z directions of the body frame.

The Position State Equation, which describes the linear velocity of the quadcopter's center of mass in the inertial frame, is the final state equation to be addressed:

$$^{i}\dot{P}^{i}_{CM|i} = C_{i|b} V^{b}_{CM|i} = \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} \tag{4.24}$$

(Position State Equation)

Here, $V^{b}_{CM|i}$ is simply the velocity of the quadcopter in the body frame that is rotated into the inertial frame using the transpose of $C_{b|i}$, which is $C_{i|b}$. This state equation allows us to determine the velocity of the quadcopter in the X, Y, and Z directions of the inertial frame.

4.2.10 Conclusion

Now that our mathematical model is ready, we need to build a control system which will allow us to control the quadcopter, given that our model can be simulated in MATLAB Simulink, we will try to implement the equations stated above with our quadcopter characteristics and optimize the results, however due to the short time we will only simulate the 'altitude hold' of the quadcopter and try to get the best stabilization possible for a stable flight.

4.3 Simulation and Control

4.3.1 Introduction

During our research for the best quadcopter simulation and control systems, we landed on an open-source Quadcopter Dynamic Modeling and Simulation made by 'MEM senior design team 37' at Drexel university, we took the model and implemented our drone characteristics and we are trying to optimize our PIDs accordingly.

This Simulink based quadcopter simulation is intended to get us up and running with a simulation representative of our vehicle's dynamic performance. The simulation represents a synthesis of several author's approaches to modeling the behavior of a quadcopter, many important effects (most obviously aerodynamic effects such as blade flapping) are ignored or dramatically simplified.

The knowledge about PIDS, Feedback control design and state space modeling from a mathematical perspective is highly required for making modification and getting the best results out of this simulation [50].

4.3.2 The Simulink model

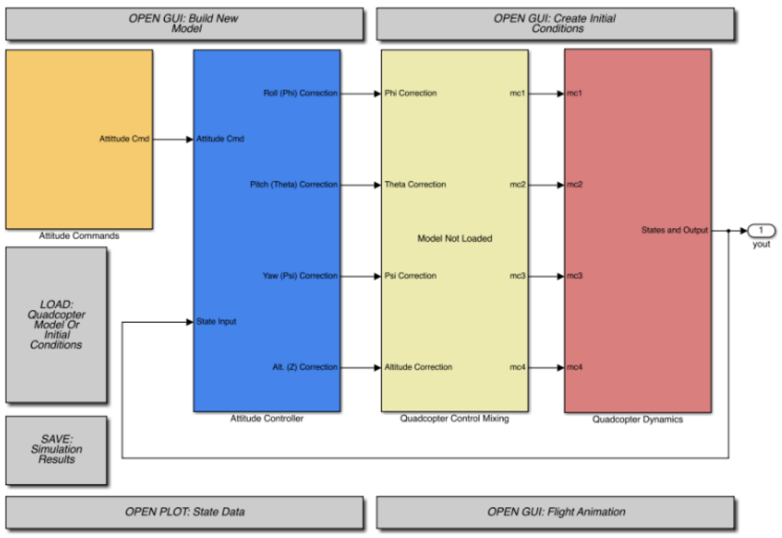
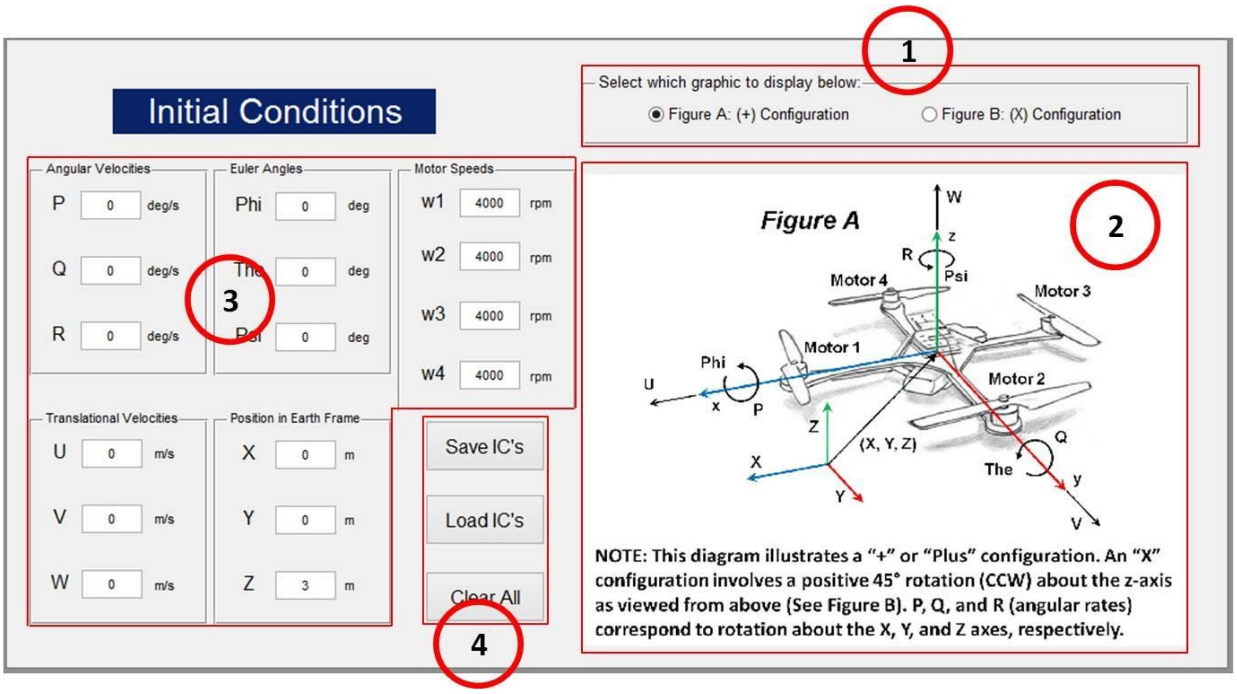


Figure 4.3 – Simulation overview

Figure (4.3), shows the whole Simulink model, every block states exactly its purpose, Notice the grey blocks; they act as buttons, by double clicking on them we can integrate our quadcopter's characteristics.

4.3.2.1 Creating initial conditions

The first step is creating the initial conditions of our quadcopter in the open GUI, for usage in the simulation, this GUI accepts user inputs of initial conditions for their quadcopter vehicle. Once all of the fields have been filled in, the interface saves a MATLAB "structure" that contains all of the simulation's parameters. A structure is a data type that allows us to group together different sorts of data into a single variable name that can be transported across the MATLAB environment easily. In Figure (4.4), you can see the Initial Conditions GUI.



Initial Conditions

Select which graphic to display below: [✓ Figure A: (+) Configuration] Figure B: (X) Configuration

Angular Velocities		
P	0	deg/s
Q	0	deg/s
R	0	deg/s
Euler Angles		
Phi	0	deg
The	0	deg
Psi	0	deg
Motor Speeds		
w1	4000	rpm
w2	4000	rpm
w3	4000	rpm
w4	4000	rpm
Translational Velocities		
U	0	m/s
V	0	m/s
W	0	m/s
Position in Earth Frame		
X	0	m
Y	0	m
Z	3	m

NOTE: This diagram illustrates a "+" or "Plus" configuration. An "X" configuration involves a positive 45° rotation (CCW) about the z-axis as viewed from above (See Figure B). P, Q, and R (angular rates) correspond to rotation about the X, Y, and Z axes, respectively.

Figure 4.4 – Initial conditions GUI

1. As stated before about the "+" and "x" configurations, Graphic display toggle: Figure A displays the initial condition parameters on a quadcopter vehicle in a "+" configuration. This configuration has Motor 1 located directly on the x-axis and Motor 2 located directly on the y-axis. Figure B displays the same vehicle except in an "X" configuration, where the x-axis is located directly between Motors 1 and 2. This involves a positive 45^o rotation (CCW) of the coordinate system about the z-axis as viewed from above. Toggle between Figure A and B to see the differences visually.

2. Toggle between Figure A and B to display a visual demonstrating the differences between the "+" and "X" configurations, our model is the "+" model.

3. Initial Condition inputs for :

   o Angular Velocities (P, Q, R) in degrees/second :

   Initial Roll rate, pitch rate, and yaw rate respectively.

   o Euler angles (Phi, Theta, Psi) in degrees:

   Initial orientation of the quadcopter vehicle with respect to the x, y, and z axes (roll, pitch, yaw) of the inertial frame.

   o Translational Velocities (U, V, W) in meters/second:

   Initial velocities of the body in the x, y, and z directions of the body frame.

   o Position in Inertial Frame (meters): Initial position of the body in the inertial frame. The inertial frame is a fixed set of axes that are used as a reference and don't accelerate or rotate.

   o Motor speeds (rpm): Initial speeds of the 4 motors on the vehicle. For example, if the vehicle is to start on the ground with motors off (w1 = w2 = w3 = w4 = 0 rpm). Or, if the vehicle is to start in a hover (w1 = w2 = w3 = w4 $\cong$ 1000 rpm for our quadcopter.

4 The "Save IC's" button is used to save the conditions inside a MATLAB structure once all of the initial condition fields have been filled in. If we want to load an IC structure that we already saved, we can click "Load IC's," choose the structure we want, and the GUI will populate the fields with the loaded parameters.

4.3.2.2 Modeling GUI

This user interface accepts physical measures from our quadcopter vehicle. The major goal of this interface is to calculate the vehicle's moment of inertia matrix based on the quadcopter's exact dimensions and measurements. The performance coefficients and other parameters generated from motor testing are also accepted by the interface. Once all of the fields have been filled out, the interface saves a MATLAB "structure" (a "+ structure in our case) with all of the parameters needed to perform the simulation. A structure is a data type that allows us to aggregate numerous pieces of data into a single variable that can be transferred around easily in the MATLAB environment. As shown in figure (4.5).

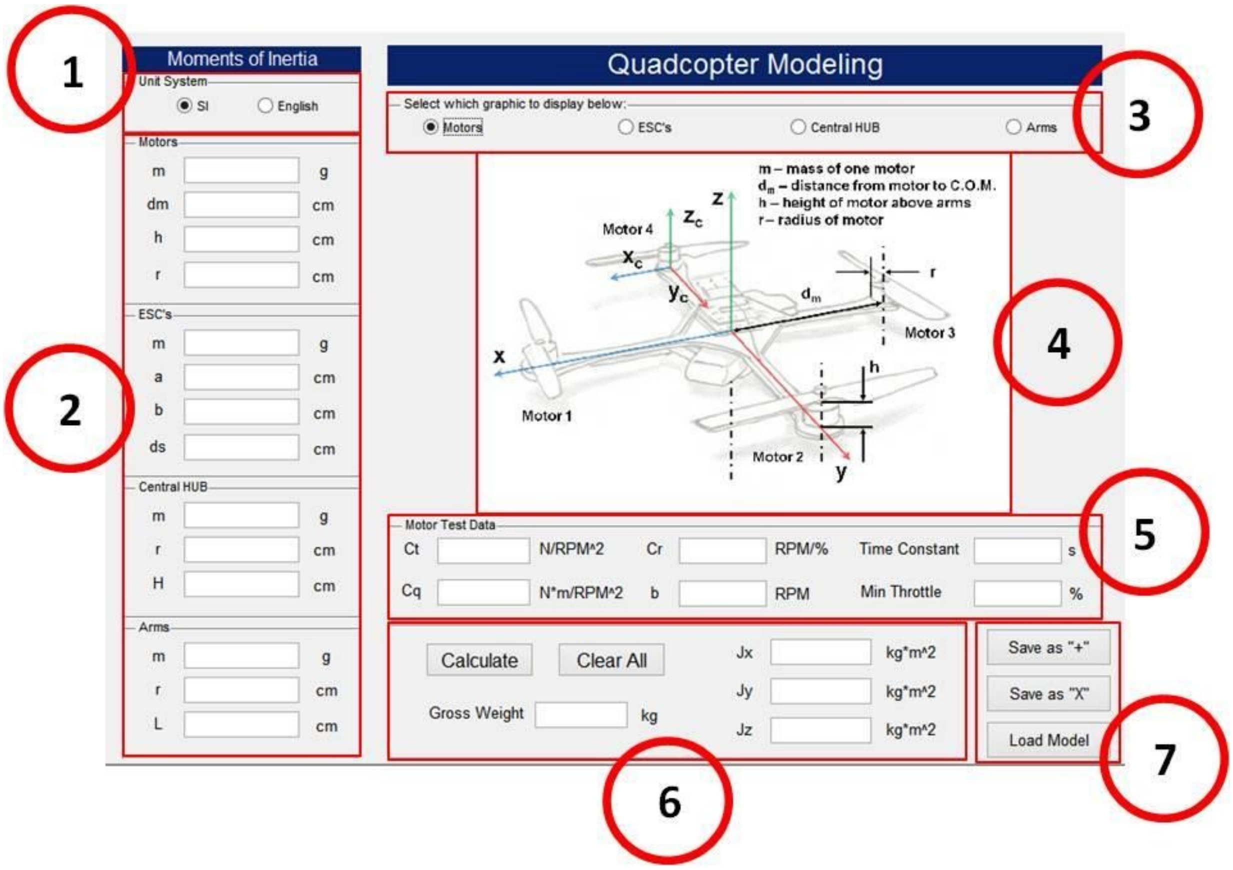


Figure 4.5 – Structure quad-copter modeling

1. Toggle between SI and English units depending on which measuring tools are being used. IMPORTANT: The application will convert all inputs into the SI unit system (kilograms/meters) for use in the simulation, regardless of whether they are in grams/centimeters or ounces/inches.

2. Inputs for Motor, ESC, HUB, and Arm dimensions (either in g/cm or oz./in.).

    o Motors:

    ∗ m: mass of one motor (with propeller) using scale

    ∗ dm: distance from the center of mass (COM) of the motor to the COM of the quad vehicle (COM is assumed to coincide with geometric center!)

∗ h: height of the motor above the arm (not including prop axel)

∗ r: radius of a motor

o ESC's

∗ m: mass of 1 ESC

∗ a: width of an ESC

∗ b: length of an ESC

∗ ds: distance from the COM of the ESC to the COM the quad vehicle

o Central HUB

∗ m: total mass of the central HUB (including battery, controller, power distribution board, etc.). This value might be most easily obtained by subtracting the mass of the individual components mentioned herein by the total weight of the quad in a "ready-to-fly" state (i.e. Hub mass = Total mass – (Motors + ESC's + Arms).

∗ r:: radius of the central HUB (modeled as a cylinder, estimate as needed)

∗ H:: height of the central hub (total height, modeled as a cylinder, see Hub diagram)

o Arms:

∗ m:: mass of (1) arm (motor, ESC, etc. excluded)

∗ r:: radius of an arm (modeled as a cylindrical rod)

∗ L: length of an arm (arm only up to attachment to HUB)

3. Toggle between graphics to display in center of GUI to assist in measuring parameters.

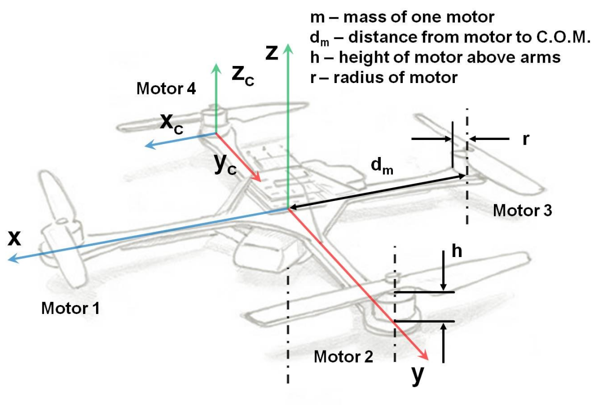
4. Motor, ESC, HUB, and Arm graphics displayed in center of GUI

5. Motor Test Data inputs: Coefficients obtained from motor test data are entered here. Also, "Min Throttle" (the minimum throttle setting for which actual motor rotation is achieved) can be entered.

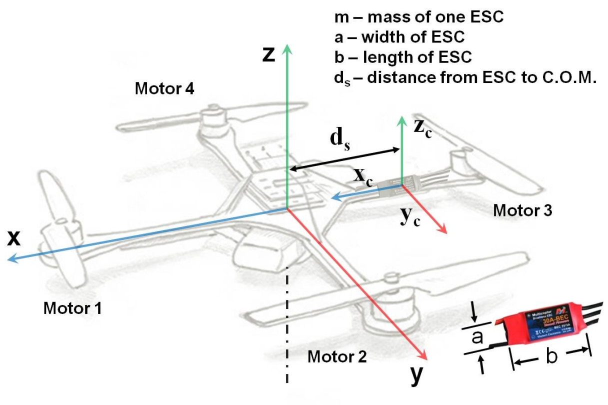
6. the "Calculate" button calculates the Gross Weight of the vehicle, and the J_x, J_y, and J_z values of the inertia matrix. Or, "Clear All" clears all of the GUI fields.

7. Once the fields are populated, we can select “Save as +” (in our case) or “Save as X” to save the model in a “plus” configuration or in an “X” configuration. The parameters are saved inside a MATLAB “structure” from which the Simulink simulation extracts them during simulation. If a model is already saved, we can select “Load Model” to select a desired structure and the GUI will populate the fields with these saved parameters.

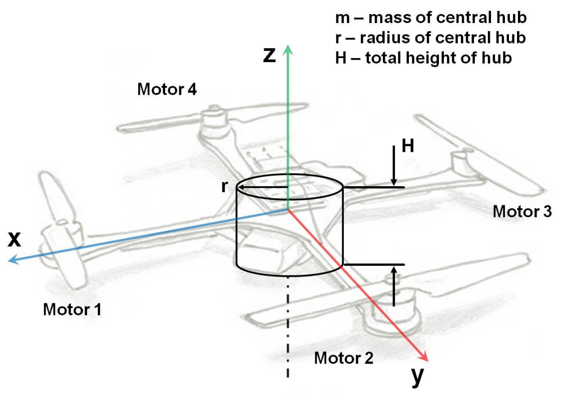
The figure (4.6) states the graphics of the GUI model:



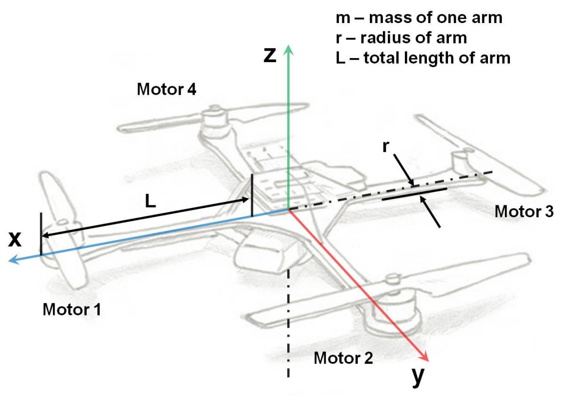
(a) Motors



(b) ESC’s



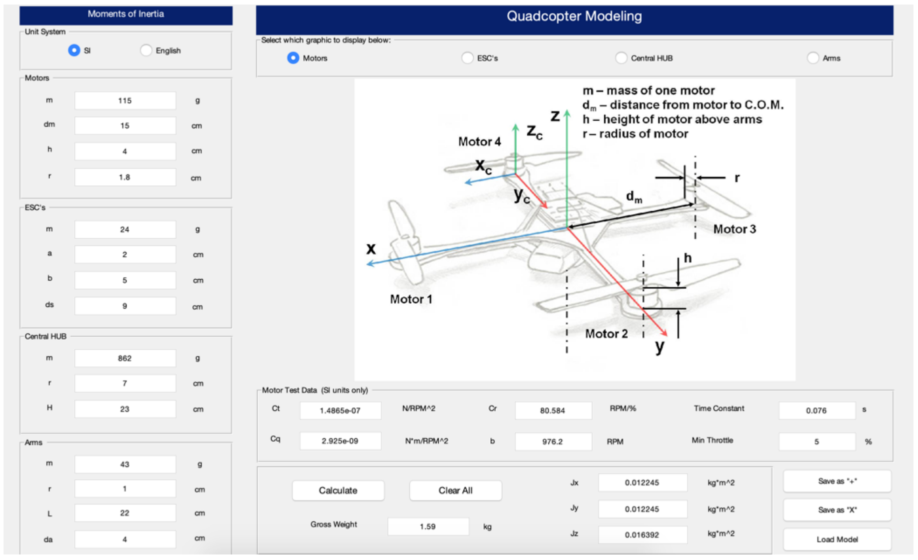
(c) Central hub



(d) Arms

Figure 4.6 – Structure modeling GUI

Now we can enter all our quadcopter characteristics and save them as “+ model”, figure (4.7).



Moments of Inertia

Unit System: [✓ SI] English

Motors

m	115	g
dm	15	cm
h	4	cm
r	1.8	cm

ESC's

m	24	g
a	2	cm
b	5	cm
ds	9	cm

Central HUB

m	862	g
r	7	cm
H	23	cm

Arms

m	43	g
r	1	cm
L	22	cm
da	4	cm

Quadcopter Modeling

Select which graphic to display below: [✓ Motors] ESC's Central HUB Arms

m – mass of one motor
d_m – distance from motor to C.O.M.
h – height of motor above arms
r – radius of motor

Motor Test Data (SI units only)

Ct	1.4865e-07	N/RPM^2	Cr	80.584	RPM/%	Time Constant	0.076	s
Cq	2.925e-09	N*m/RPM^2	b	976.2	RPM	Min Throttle	5	%

Gross Weight 1.59 kg

Jx	0.012245	kg*m^2
Jy	0.012245	kg*m^2
Jz	0.016392	kg*m^2

Figure 4.7 – Structure modeling characteristics of our model

4.3.2.3 Altitude command only model

This is an altitude-command-only model. In other words, there is no control system to track position. Instead, the controller only tries to track altitude (φ, θ, ψ) and altitude (Z) commands using a PID controller (we will make our PIDs later on).

Once we have integrated all of the above, we can now load our saved "quad model +" and the open-source "hover.mat" (it's an altitude command only) by double clicking on "LOAD quadcopter model or initial conditions).

4.3.2.4 Altitude command blocks

With the models loaded the simulation can be run, now we need check out a few things first. the Altitude Commands block contains our PIDs for the roll, pitch, yaw and altitude control.

4.3.2.5 Altitude controller

Figure (4.8) demonstrates the altitude controller block, there are four blocks "Roll, Pitch, Yaw and altitude control blocks each one of them contains a PID controller that we can tune, figure (4.9)

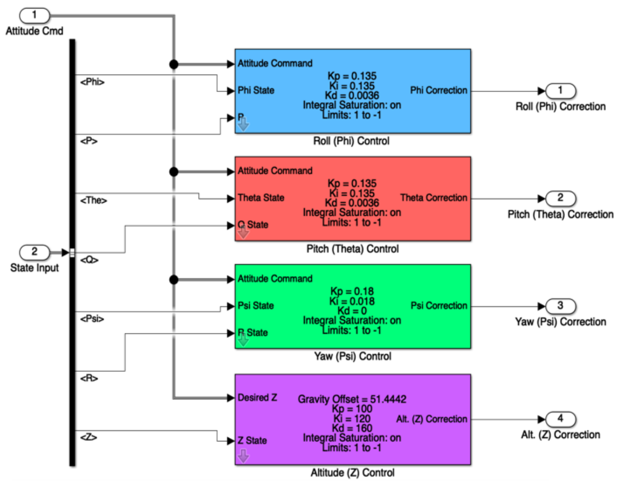


Figure 4.8 – Attitude control block interior

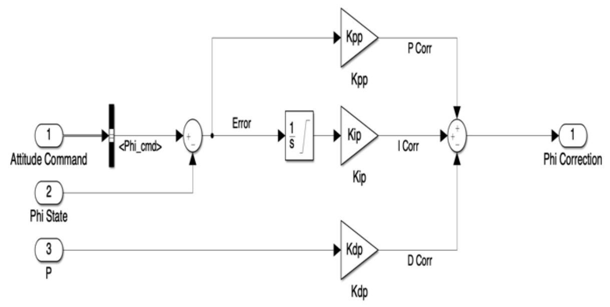


Figure 4.9 – PID controller diagram

4.3.2.5.1 Altitude command blocks The Ziegler-Nichols rule is a heuristic PID tuning rule that attempts to produce good values for the three PID gain parameters, we gradually increase the gain "k_p " acting alone until the correct oscillation of the loop (pumping), k_{pc} is the limit gain co-ordinates the pumping while T_c is the period of oscillations. The Ziegler and Nichols Method proposing to calculate the PID corrector gains using the following recommendations [51]:

Régulateur	k_p	k_i	k_d
P	0.5* k_{pc}		
PI	0.45 k_{pc}	0.38* T_c	
PID	0.6* k_{pc}	0.5*T_c	0.125* T_c

Table 4.1 – Ziegler and Nichols Method table

.

Since we are using PID we can start calculating based on our simulation:

- Roll and Pitch PIDs: since we have approximately the same k_{pc} and T_c values for both Pitch and Roll we are going to use the same calculation for both: our k_{pc} and T_c are respectively: k_{pc}=0.225 T_c=0.27
  using the table equations:

$$k_p=0.6*k_{pc}=0.6*0.225=0.135$$
$$k_i=0.5*T_c=0.25*0.27=0.135$$
$$k_d=0.125*T_c=0.125*0.27=0.0036$$

- Repeating the same method, we can get the PIDs of the yaw and altitude Z:

yaw	altitude
k_p=0.18	k_p=100
k_i=0.018	k_i=120
k_d=0	k_d=160

Table 4.2 – PID values of our model

.

4.3.2.6 Quadcopter Control Mixing block

This block takes the correction commands for the Phi, Theta, Psi and Z and "mixes" them by letting each correction be sent to the correct motor (sign is very important). It should say "Configuration "+"" on the front, indicating that it has recognized that you loaded a "+" configuration model. You should see something like this inside. Figure (4.10)

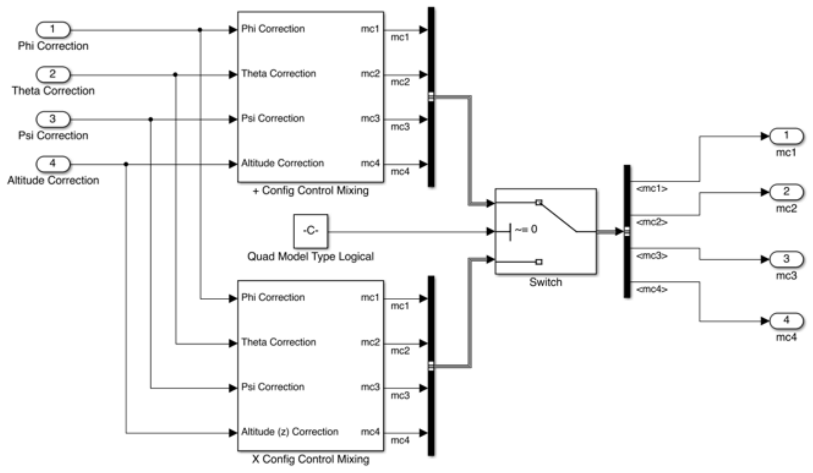


Figure 4.10 – Quadcopter control mixing overview

The purpose of the switch mechanism is to allow both "X"-configuration and "+"-configuration vehicles to be controlled correctly without having to use a different block. Take a look inside both the "+" and "X" blocks. The plus block is the one that will actually be used right now since we loaded a "+" quadcopter model. The equations are written next to each output for reference, Figure (4.11).

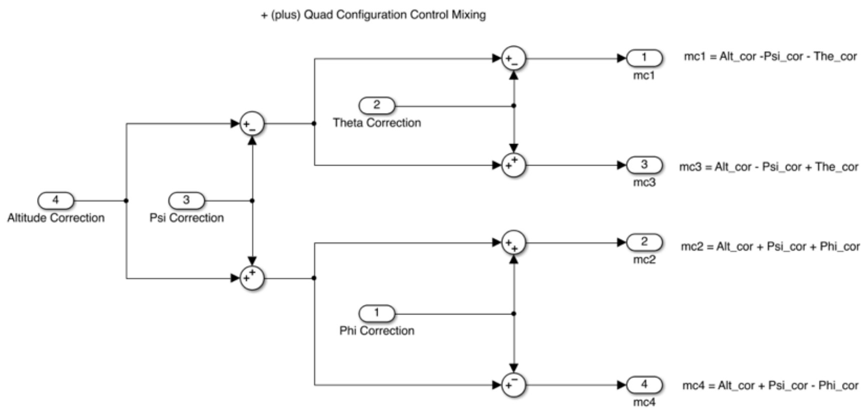

+ (plus) Quad Configuration Control Mixing
Altitude Correction
Psi Correction
Theta Correction
Phi Correction
mc1
mc2
mc3
mc4
mc1 = Alt_cor -Psi_cor - The_cor
mc3 = Alt_cor - Psi_cor + The_cor
mc2 = Alt_cor + Psi_cor + Phi_cor
mc4 = Alt_cor + Psi_cor - Phi_cor

Figure 4.11 – Plus configuration control mixing

4.3.2.7 Quadcopter dynamics block

As shown in Figure (4.12), The motor dynamics block restricts input commands to between 0% and 100% throttle (which is obviously the maximum possible range of throttle command signals), simulates the motor cutoff behavior at very low throttle, and most importantly applies the and linear relation to the percent throttle signal and simulates the first order delay. The output of the block is the RPM for each motor at any given moment in time.

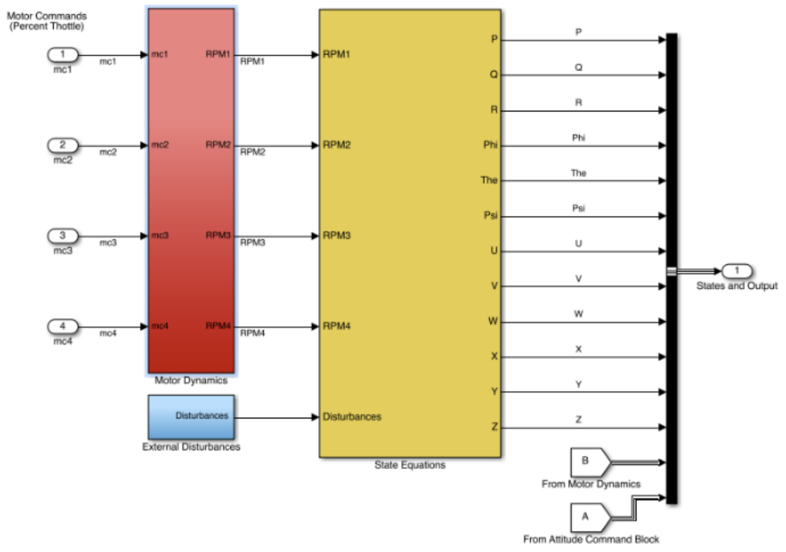


Figure 4.12 – Quadcopter Dynamics block

The disturbance block is not something we will use very much. It was added in order to make it easier to add external disturbance effects (such as wind forces on the vehicle) to the simulation.

4.3.3 MATLAB Simulation

Now that we are done with the implementation of our quadcopter characteristics, we can safely start the simulation and see how it goes.

Since this is an altitude only command system, we have set 10ft as our desired altitude; the flight animation GUI will allow us to see how our drone reacts, and 'open plot: state data' will allow us to see the graphs of our motors and other characteristics.

In Figure (4.13), we can see the flight animation GUI at a 10s simulation with 3frame skips:

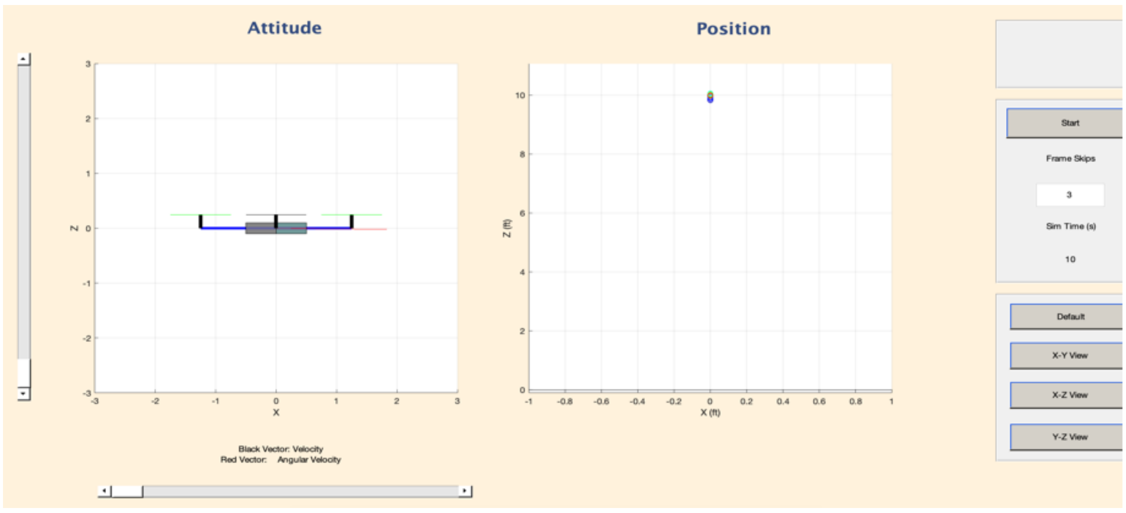


Figure 4.13 – Simulation of attitude hold of our quadcopter

Figure (4.13) , shows great results of position altitude hold and stability of the quadcopter, the quadcopter held its 10ft altitude with small margin of error.

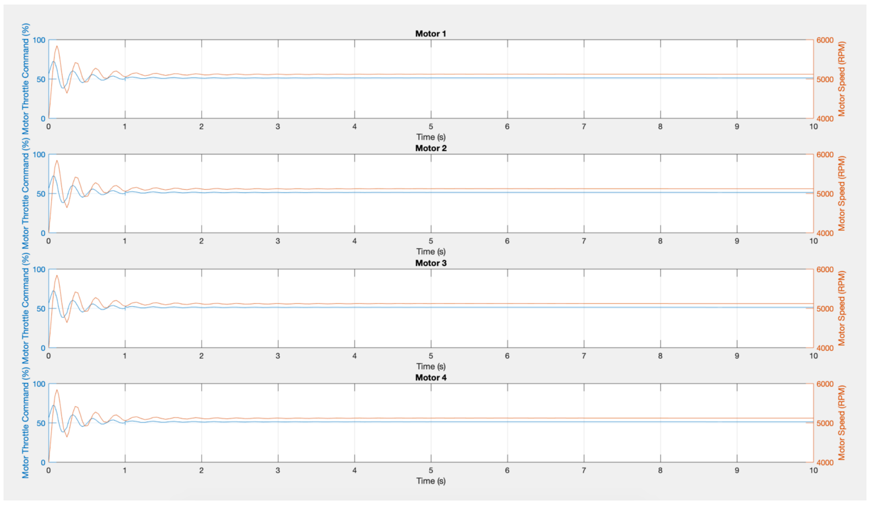


Figure 4.14 – Motor commands

Figure (4.14) represents a simulation graph of the throttle command percentage alongside the motors speed in RPM, it's using a conventional PID control to make the dynamic self-balancing.

Since the quadcopter takes off at almost 10ft the throttle command and the motor speed increases for the first 0.2 seconds in order to reach the desired altitude (10ft), the throttle command reaches a peak of 90% and the speed peak reaches 5250 RPMs, from 0.2 to 1.3 seconds the oscillations indicate that the motors are seeking stability in the transitional regime, after that our quadcopter reaches the permanent regime as the motors speed stabilize at 5000 RPMs with the throttle command percentage stabilizing at 55%, The system overshoot is small, at the same time the steady- state error is almost zero, and the system response is fast. In the PID design some modification can be done to make it more effectiveness and eliminates some constrains of conventional PID.

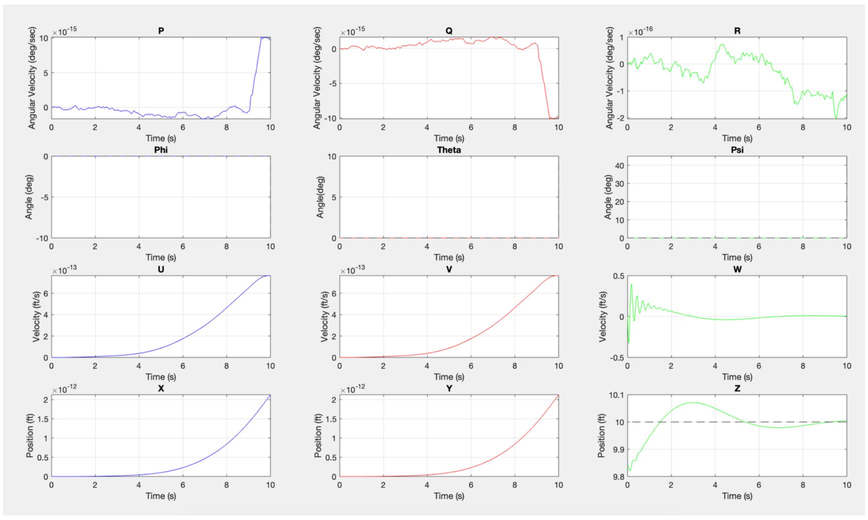


Figure 4.15 – Movement and speed of the UAV

Figure (4.15) represents a simulation graph of the movement of the quadcopter on the three axis (XYZ) it's using a conventional PID control to make the dynamic self-balancing, the results show that the angular velocity of P, Q and R according to the XYZ axis respectively is very small and can be considered negligible; U, V and W are the actual speed of the quadcopter according to the XYZ axis respectively, the velocity U and V are too small and considered negligible, the results show that the quadcopter movement on the X and Y axis is too small meaning that the quadcopter isn't moving forward, backward, left or right, however the velocity W varies for the first 1.5 seconds reaching a peak of 0.5 ft/s, as the quadcopter comes to stabilization the velocity W decreases.

Note: Phi, Theta and Psi represent the commands of movements according to X, Y and Z axis respectively, in this simulation we didn't give any commands in any of the axis.

4.4 Conclusion

This Simulink model helped in order to get good PIDs parameters, the simulated results presented great stabilization of the quadcopter.

We implemented the PIDs to our flight controller and the actual tests were convenient.

Chapter 5

Programming of the Quadcopter

5.1 Introduction

The flight controller (FC) is the low-level device that operates the hardware. It is what allows the drone to self-level itself hundreds of times a second. The FC needs to be highly dependable, because if there are any time delays at all, the quadcopter will crash.

The companion computer is a little different. This device can essentially run high level scripts and communicate commands to the flight controller. In any raspberry pi quadcopter, the raspberry pi board will be acting as the companion computer.

Interestingly enough, we can fly the quadcopter without a companion computer/raspberry pi quadcopter, and even do basic autonomous missions on a Pixhawk quadcopter. The point of putting a companion computer is that it unleashes advanced functionality to the quadcopter.

5.2 Raspberry pi Commands

5.2.1 Setting up the raspberry pi

5.2.1.1 Raspberry pi OS

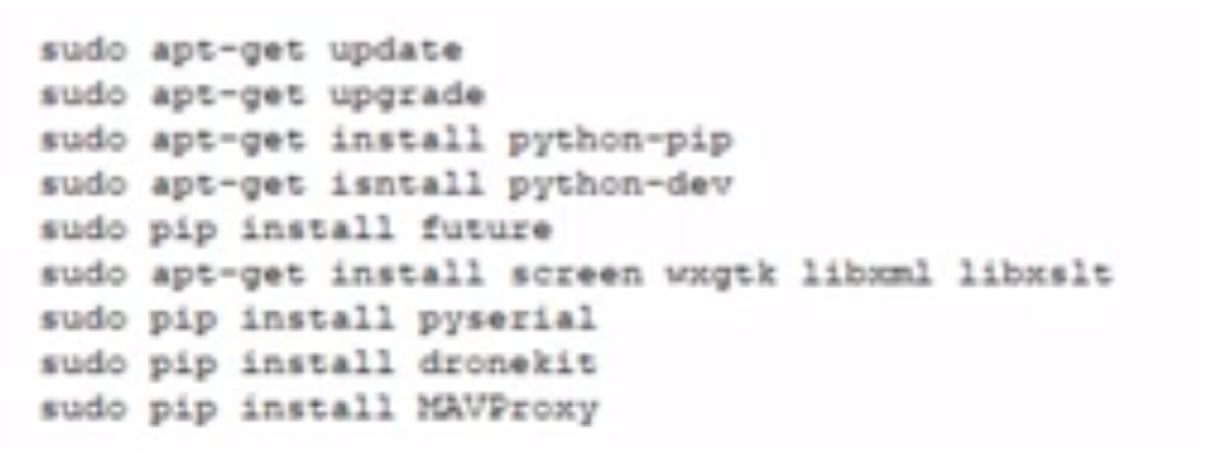
Since our hardware setup is done, we need to setup our software on our raspberry pi in order to be able to execute high end scripting. We need to install the raspberry pi OS using Raspberry pi manager to write the OS to our SD card [52].

Note: we must enable 'SSH' in the configuration menu before burning our image, since it's the best way to send orders using the WIFI connection, also we need to setup our WIFI SSID and password for an easy connection.

Our Raspberry pi OS is ready and can be run now, and we are able to send commands, install packages and write scripts on an 'SSH' connection.

5.2.2 Raspberry pi packages

The OS is ready, however it's an empty space, we need to install all the essential packages in order to be able to control the quadcopter, using the 'terminal command window' we can build an 'SSH' connection by writing the commands 'pi@raspberrypi.local', once the raspberry pi is detected we enter our Password which is 'raspberry' by default, now we can install the packages needed, figure (5.1).



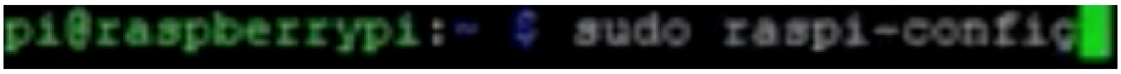
```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install python-pip
sudo apt-get isntall python-dev
sudo pip install future
sudo apt-get install screen wxgtk libxml libxslt
sudo pip install pyserial
sudo pip install dronekit
sudo pip install MAVProxy
```

Figure 5.1 – Raspberry pi packages

Every package above is a system package and is needed to facilitate the running of the OS except the last two 'dronekit' and 'MAVProxy' which will allow us to import the drone commands from its equations, once all our packages are installed, we can now write simple scripts and command our quad copter.

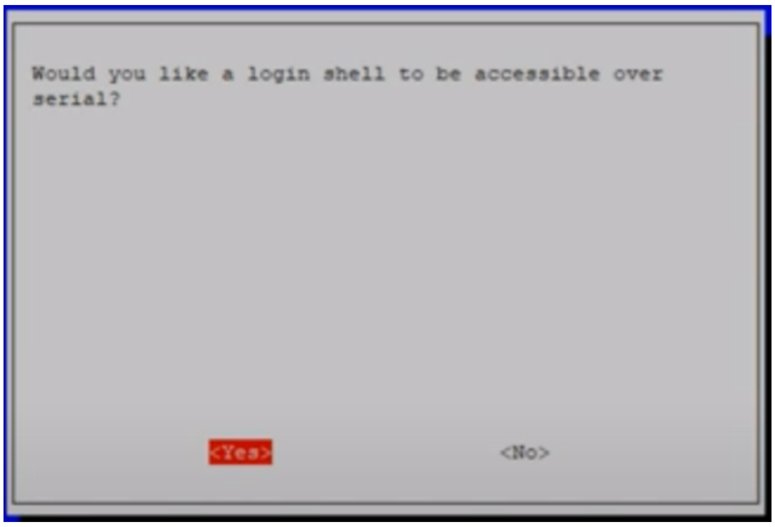
5.2.3 Raspberry pi python scripts

Our commands go through a ‘UART’ hardware connection, in order to control our quadcopter through python scripting we need to run a simple configuration first by typing ‘sudo raspi-config’, figure (5.2), in the menu through ‘interface options’ ‘serial ports’ we disabled the ‘login shield over serial’, figure (5.3), and enable the ‘serial port hardware’, figure (5.4).



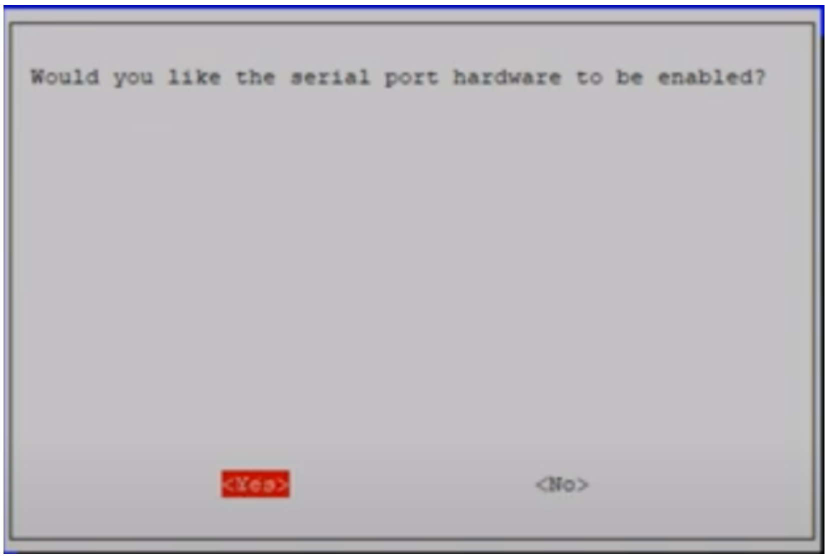
```
pi@raspberrypi:~ $ sudo raspi-config
```

Figure 5.2 – sudo raspi-config



Would you like a login shell to be accessible over serial?

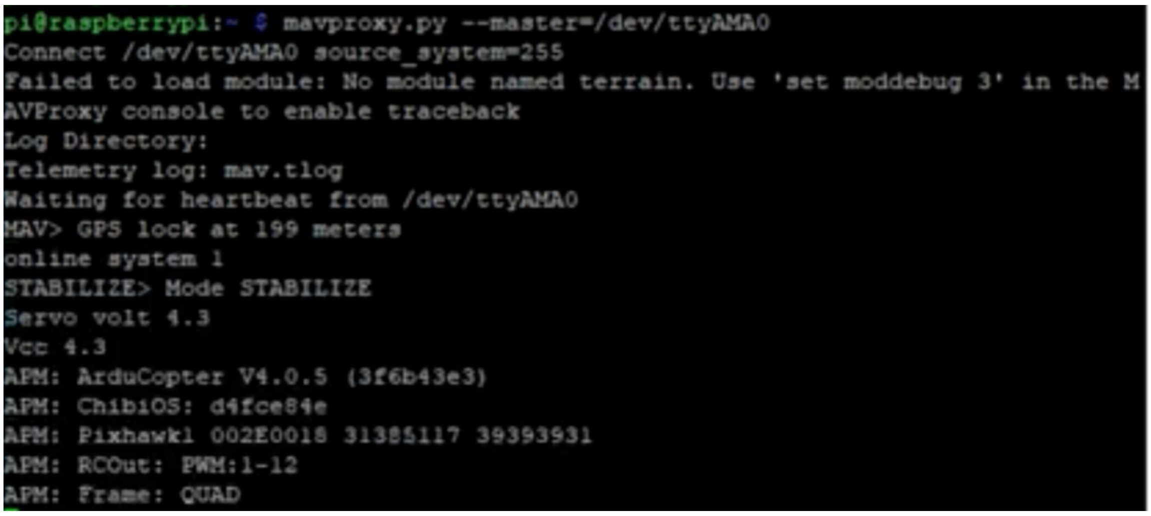
Figure 5.3 – disable the ‘login shield over serial



```
Would you like the serial port hardware to be enabled?

          <Yes>                    <No>
```

Figure 5.4 – enable the ‘serial port hardware’

In figure (5.5), by writing the command ‘mavproxy.py –master=/dev/ttyAMA0’ we can see all our quadcopter real-time specifications [53].



```
pi@raspberrypi:~ $ mavproxy.py --master=/dev/ttyAMA0
Connect /dev/ttyAMA0 source_system=255
Failed to load module: No module named terrain. Use 'set moddebug 3' in the M
AVProxy console to enable traceback
Log Directory:
Telemetry log: mav.tlog
Waiting for heartbeat from /dev/ttyAMA0
MAV> GPS lock at 199 meters
online system 1
STABILIZE> Mode STABILIZE
Servo volt 4.3
Vcc 4.3
APM: ArduCopter V4.0.5 (3f6b43e3)
APM: ChibiOS: d4fce84e
APM: Pixhawk1 002E0018 31385117 39393931
APM: RCOut: PWM:1-12
APM: Frame: QUAD
```

Figure 5.5 – quadcopter real-time specifications

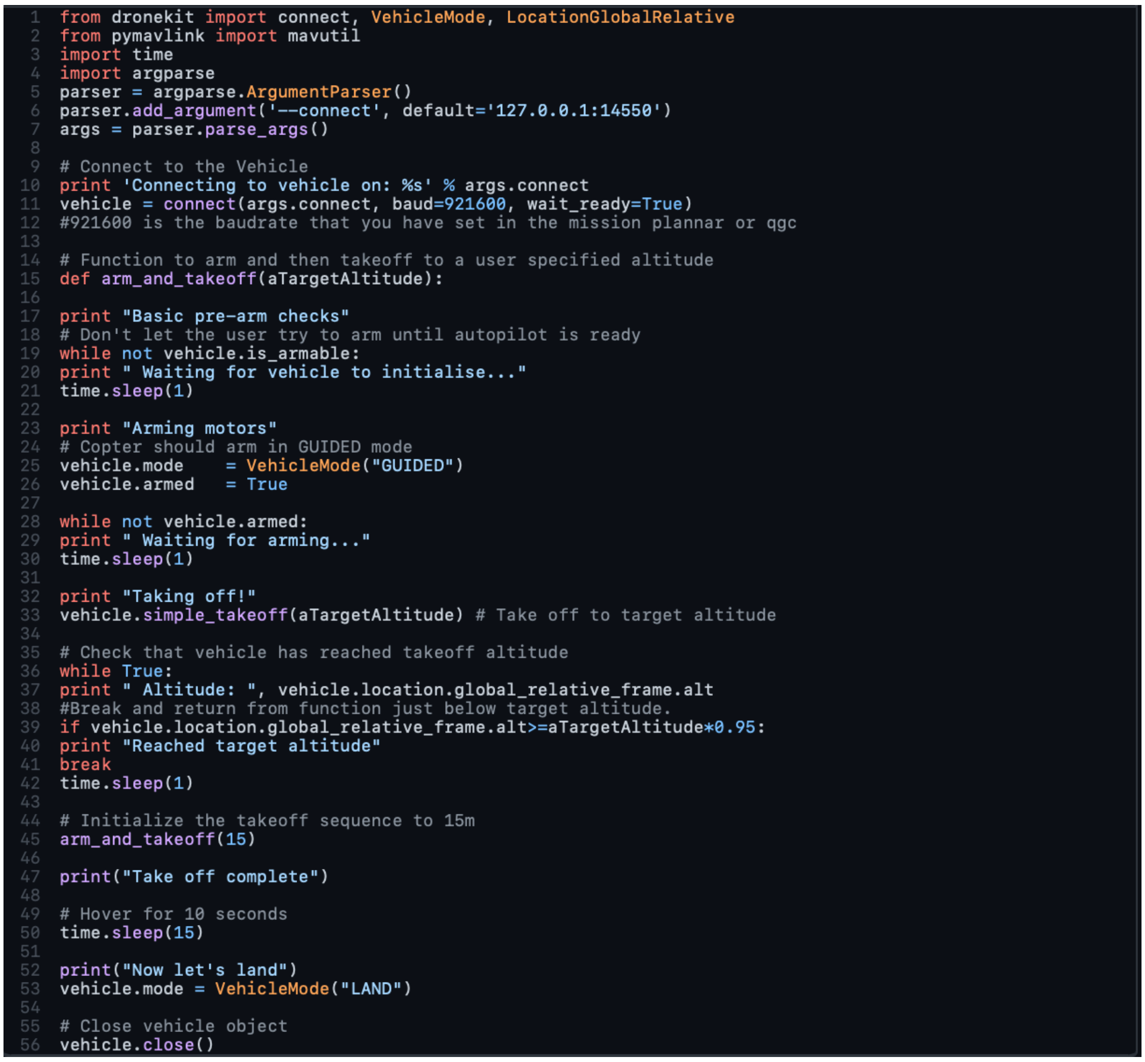
Now we can write our scripts, using the command ‘vi project-name.py’,

5.3 Python scripts and programs

Now that we can code in python, there are no limits when it comes to the applications that can be realized, we made some interesting python coding and we made the following progams.

5.3.1 Take off-Hover-Land program

Using the 'MAVProxy' and 'dronekit' commands, we wrote a simple python script that will allow our drone to take off to fifteen meters and then hover for ten seconds, right after that our quadrotor will perform a smooth landing [54]. figure (5.6)



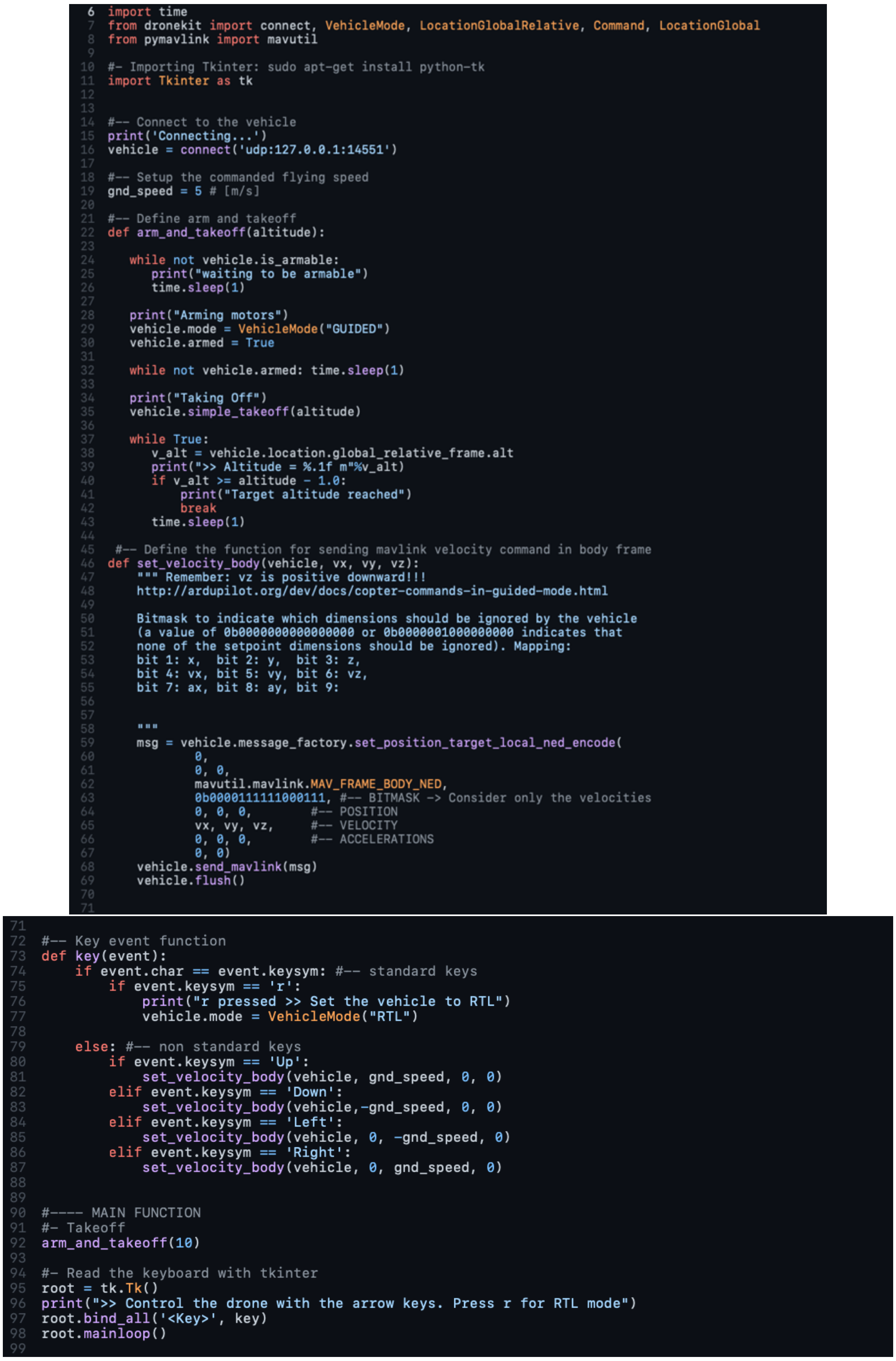
```
from dronekit import connect, VehicleMode, LocationGlobalRelative
from pymavlink import mavutil
import time
import argparse
parser = argparse.ArgumentParser()
parser.add_argument('--connect', default='127.0.0.1:14550')
args = parser.parse_args()

# Connect to the Vehicle
print 'Connecting to vehicle on: %s' % args.connect
vehicle = connect(args.connect, baud=921600, wait_ready=True)
#921600 is the baudrate that you have set in the mission plannar or qgc

# Function to arm and then takeoff to a user specified altitude
def arm_and_takeoff(aTargetAltitude):

print "Basic pre-arm checks"
# Don't let the user try to arm until autopilot is ready
while not vehicle.is_armable:
print " Waiting for vehicle to initialise..."
time.sleep(1)

print "Arming motors"
# Copter should arm in GUIDED mode
vehicle.mode    = VehicleMode("GUIDED")
vehicle.armed   = True

while not vehicle.armed:
print " Waiting for arming..."
time.sleep(1)

print "Taking off!"
vehicle.simple_takeoff(aTargetAltitude) # Take off to target altitude

# Check that vehicle has reached takeoff altitude
while True:
print " Altitude: ", vehicle.location.global_relative_frame.alt
#Break and return from function just below target altitude.
if vehicle.location.global_relative_frame.alt>=aTargetAltitude*0.95:
print "Reached target altitude"
break
time.sleep(1)

# Initialize the takeoff sequence to 15m
arm_and_takeoff(15)

print("Take off complete")

# Hover for 10 seconds
time.sleep(15)

print("Now let's land")
vehicle.mode = VehicleMode("LAND")

# Close vehicle object
vehicle.close()
```

Figure 5.6 – Python script Take off-Hover-Land program

Now we execute our script by typing ‘python project-name.py –connect /dev/ttyAMA0’, if everything runs smoothly our drone can start the autonomous mission of taking off and hover then land on the ground.

5.3.2 Keyboard quadcopter control

As mentioned before the possibilities are endless, this time we made a python script that will allow us to take control of our drone using our arrow keys in our keyboard, the script below is self-explanatory as it has comments, this script will allow us to control the quad copter in the X-Y axis using arrow key and by pressing the ‘r’ button on our keyboard our quadcopter will perform a smooth landing. Figure(5.7)



```python
import time
from dronekit import connect, VehicleMode, LocationGlobalRelative, Command, LocationGlobal
from pymavlink import mavutil

#- Importing Tkinter: sudo apt-get install python-tk
import Tkinter as tk


#-- Connect to the vehicle
print('Connecting...')
vehicle = connect('udp:127.0.0.1:14551')

#-- Setup the commanded flying speed
gnd_speed = 5 # [m/s]

#-- Define arm and takeoff
def arm_and_takeoff(altitude):

   while not vehicle.is_armable:
      print("waiting to be armable")
      time.sleep(1)

   print("Arming motors")
   vehicle.mode = VehicleMode("GUIDED")
   vehicle.armed = True

   while not vehicle.armed: time.sleep(1)

   print("Taking Off")
   vehicle.simple_takeoff(altitude)

   while True:
      v_alt = vehicle.location.global_relative_frame.alt
      print(">> Altitude = %.1f m"%v_alt)
      if v_alt >= altitude - 1.0:
          print("Target altitude reached")
          break
      time.sleep(1)

 #-- Define the function for sending mavlink velocity command in body frame
def set_velocity_body(vehicle, vx, vy, vz):
    """ Remember: vz is positive downward!!!
    http://ardupilot.org/dev/docs/copter-commands-in-guided-mode.html

    Bitmask to indicate which dimensions should be ignored by the vehicle
    (a value of 0b0000000000000000 or 0b0000001000000000 indicates that
    none of the setpoint dimensions should be ignored). Mapping:
    bit 1: x,  bit 2: y,  bit 3: z,
    bit 4: vx, bit 5: vy, bit 6: vz,
    bit 7: ax, bit 8: ay, bit 9:


    """
    msg = vehicle.message_factory.set_position_target_local_ned_encode(
            0,
            0, 0,
            mavutil.mavlink.MAV_FRAME_BODY_NED,
            0b0000111111000111, #-- BITMASK -> Consider only the velocities
            0, 0, 0,        #-- POSITION
            vx, vy, vz,     #-- VELOCITY
            0, 0, 0,        #-- ACCELERATIONS
            0, 0)
    vehicle.send_mavlink(msg)
    vehicle.flush()


#-- Key event function
def key(event):
    if event.char == event.keysym: #-- standard keys
        if event.keysym == 'r':
            print("r pressed >> Set the vehicle to RTL")
            vehicle.mode = VehicleMode("RTL")

    else: #-- non standard keys
        if event.keysym == 'Up':
            set_velocity_body(vehicle, gnd_speed, 0, 0)
        elif event.keysym == 'Down':
            set_velocity_body(vehicle,-gnd_speed, 0, 0)
        elif event.keysym == 'Left':
            set_velocity_body(vehicle, 0, -gnd_speed, 0)
        elif event.keysym == 'Right':
            set_velocity_body(vehicle, 0, gnd_speed, 0)


#---- MAIN FUNCTION
#- Takeoff
arm_and_takeoff(10)

#- Read the keyboard with tkinter
root = tk.Tk()
print(">> Control the drone with the arrow keys. Press r for RTL mode")
root.bind_all('<Key>', key)
root.mainloop()
```

Figure 5.7 – Python script Keyboard quadcopter control

5.3.3 Surveillance quadcopter

After making the first two scripts, we wanted to realize the idea of using an FPV camera, however this time using the "Pi camera", python scripts and adding some improvements. The idea is based on transmitting real-time video feed through a "URL", this application is open source and capable of taking a video in a surveillance mode, meaning that it will only take a video if a new object passed by the camera. Figure (5.8).

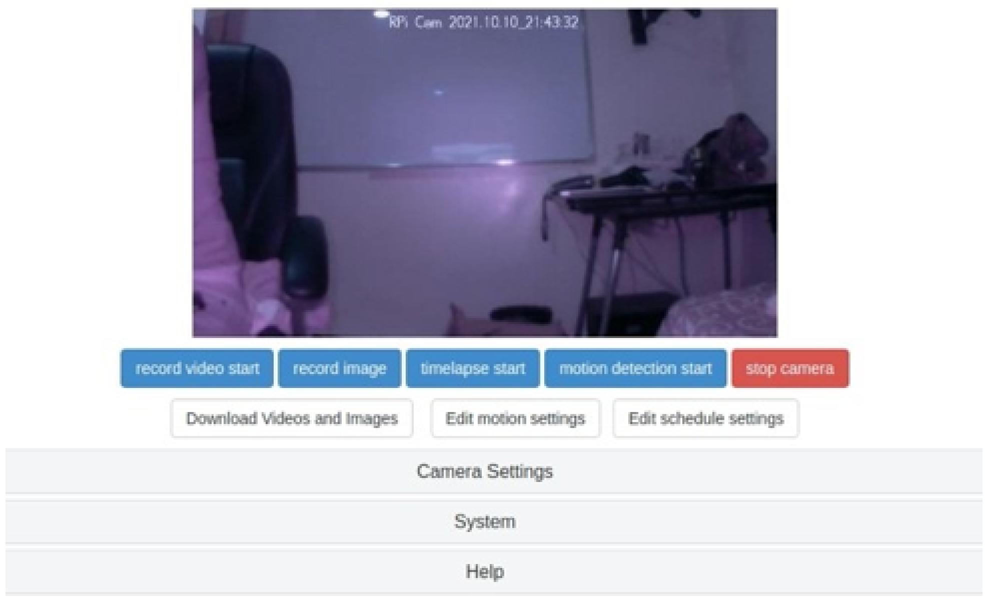


Figure 5.8 – Surveillance quadcopter

5.3.4 Gesture Control

Implementing gesture control was really challenging, we needed to import a lot of libraries like 'mediapipe' and 'opencv', since we had our base model of controlling the drone with keyboard commands, we needed to switch these commands with gestures, however it was not easy, we needed to train our model to avoid any mis readings of our hand gestures [55] [56]. Figure(5.9)

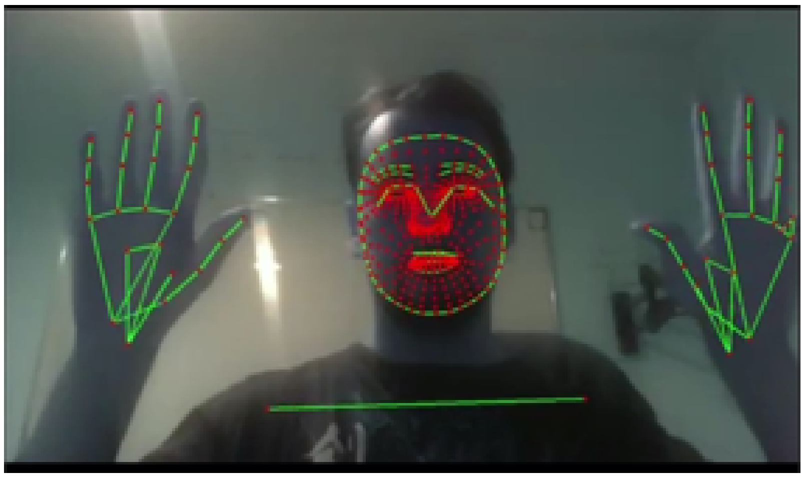


Figure 5.9 – Gesture Control

when the camera sees one finger: the quadrotor goes forward, two fingers: the quadrotor goes backwards, three fingers: the quadrotor goes left, four fingers: the quadrotor goes right and finally when the hand is closed the quadrotor will land smoothly, down below is the diagram of command .

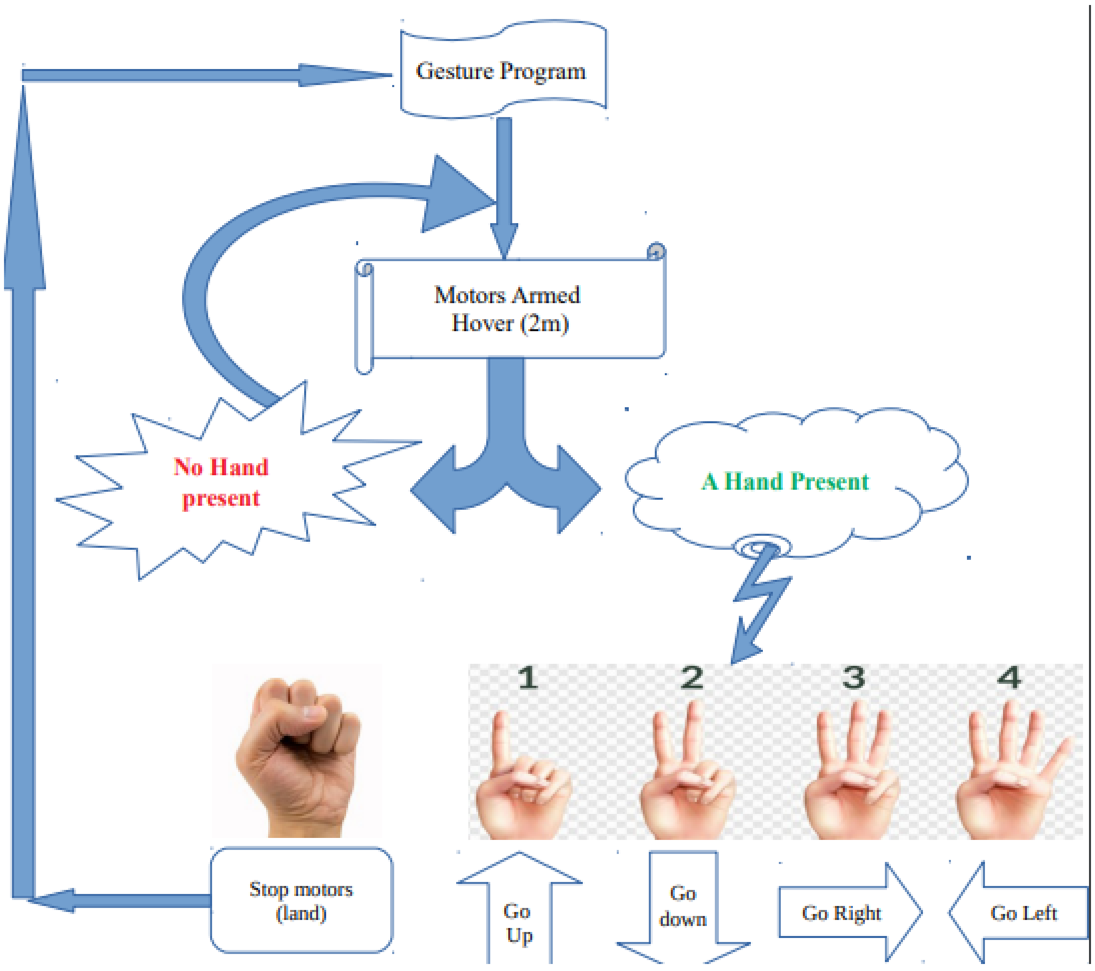


Figure 5.10 – diagram of command

5.4 Conclusion

The Pixhawk-raspberry pi combination is a powerful combination that basically let us do whatever we want through scripting using python commands, we are currently working on object image processing and tracking, in hopes that we can complete our task in the near future.

Conclusion and Future Work

Conclusion

Main aim of this project was to develop a Drone which can be used in several control purposes. For controlling the Drone, 2.4 GHz radio frequency transmitter, receiver, microcontroller, companion computer, electronic speed controller, brushless DC motors.

AutoCAD 2016 and ANSYS 17.0 software were used to make the conception and durability tests, the results were convenient and safe to use, it took a month of work. The components selection was based on making a mini drone of 500mm diameter, the realization took about two days to accomplish.

MATLAB ®/Simulink® has been used to developed the Drone roll, yaw and pitch control system simulation for altitude control only using an open source model created by MEM group from Drexel university, by implementing our model characteristics we managed to regulate our PIDs (The proportional, integral, derivative controller), the results shows good performance of controlling the roll, pitch and yaw while maintaining a 10ft altitude of developed Drone, we also tested the quadcopter's stability even while adding exterior effects.

For live video footage feedback, image processing and commanding the quadcopter autonomously a raspberry pi 4b was used, the scripting part was not easy by any means as we didn't have a background about python scripting. Demonstration shows the successful operation of quadcopter taking commands from the raspberry python scripts (take-off land mission, controlling the quadcopter using keyboard commands and controlling the quadcopter using gesture commands).

Limitations

Though we were trying to make it modest, we had some limitations. Firstly, the power issues got priority. Our prototype can fly up to 12 minutes with fully charged battery. However, we can overcome such issues by using more powerful batteries and motors but that will increase the cost approximately 35% of the overall cost. Secondly, the cost was expensive as we spent about 1150 euros, the delivery of the components took longer than expected, we had only 40 days do accomplish all of this work.

Some of our ESCs were burnt during the flying tests so we had to buy new ones, the quadcopter crashed many times, the GPS accuracy became poor.

Since our radio controller's range is approximately 3 km so we cannot operate this vehicle beyond this range.

Recommendations

This work could be improved by including the precise mechanical instruments made of carbon fiber. In addition, to have a more precise PID parameters, new methods of PID tuning (the use of genetic algorithms) could be employed for optimal values. Yaw mechanism for tilting rotors can be more improved and well controlled by using gear mechanism. On the other hand, to increase the flight duration 5 cell 5000 mAh Li-Po batteries can be used and it is highly recommended that a vibration absorber needs to be attached to remove vibration effects and get a stable flight. For the GPS a LIDAR sensor can be added to get accurate altitude measurements.

Future work

The **future work** is to firstly enhance the range of our quadcopter using 4G LTE connection to get unlimited range with live video feed, we will also try to optimize our gesture control model, as for other projects, object detection and tracking is our goal at the moment, getting longer flight time and better maximum weight load by improving our components.

Bibliography

[1] Michel Asensio, les drones et les opérations en réseau, segmentation mission, fondation pour la recherche stratégique ,2008

[2] Rodrigo Martinez-Val, Carlos Hernandez, Preliminary design of a low speed, long endurance remote piloted vehicles (RPV) for civil applications, Aircraft Design 2 (1999) ,167-182

[3] G, Romeo, G, Frulla, and E, Cestino, Design of a high-altitude long-endurance solar-powered unmanned air vehicle for multi-payload and operations Proc. IMechE Vol. 221 Part G : J. Aerospace Engineering 2007

[4] Air and Space europe, Current and Future UAV Military Users and Applications, VOL .1, NO 5/6, 1999

[5] Kaplan, Philip (2013). Naval Aviation in the Second World War. Pen and Sword. p. 19. ISBN 978-1-4738-2997-8

[6] Hallion, Richard P. (2003). Taking Flight: Inventing the Aerial Age, from Antiquity through the First World War. Oxford University Press. p. 66. ISBN 978-0-19-028959-1.

[7] Naval Aviation in the First World War: Its Impact and Influence, R. D. Layman, page 56

[8] Renner, Stephen L. (2016). Broken Wings: The Hungarian Air Force, 1918–45. Indiana University Press. p. 2. ISBN 978-0-253-02339-1.

[9] Murphy, Justin D. (2005). Military Aircraft, Origins to 1918: An Illustrated History of Their Impact. ABC-CLIO. pp. 9–10. ISBN 978-1-85109-488-2.

[10] Haydon, F. Stansbury (2000). Military Ballooning During the Early Civil War. JHU Press. pp. 18–20. ISBN 978-0-8018-6442-1.

[11] ”Mikesh, Robert C. ”Japan’s World War II balloon bomb attacks on North America.” (1973)” (PDF).

[12] Jump up to:a b c Taylor, John W. R.. Jane’s Pocket Book of Remotely Piloted Vehicles.

[13] Professor A. M. Low FLIGHT, 3 October 1952 page 436 “The First Guided Missile”

[14] Wagner 1982, p. 79.

[15] Z. Goraj; A. Frydrychewicz; R. Świtkiewicz; B. Hernik; J. Gadomski; T. Goetzendorf-Grabowski; M. Figat; St Suchodolski; W. Chajec. report (PDF). Bulletin of the Polish Academy of Sciences, Technical Sciences, Volume 52. Number 3, 2004. Retrieved 9 December 2015.

[16] Community Research and Development Information Service. Civil uav application and economic effectiveness of potential configuration solutions. published by the Publications Office of the European Union. Retrieved 9 December 2015.

[17] Ackerman, Spencer; Shachtman, Noah (9 January 2012). ”Almost 1 in 3 U.S. Warplanes Is a Robot”. WIRED. Retrieved 8 January 2015.

[18] Jump up to:a b Singer, Peter W. ”A Revolution Once More: Unmanned Systems and the Middle East” Archived 6 August 2011 at the Wayback Machine, The Brookings Institution, November 2009.

[19] Radsan, AJ; Murphy (2011). ”Measure Twice, Shoot Once: Higher Care for Cia-Targeted Killing”. Univ. Ill. Law Rev.:1201–1241.

[20] Sayler (2015)

[21] Franke, Ulrike Esther [”The global diffusion of unmanned aerial vehicles (UAVs) or ’drones’”], in Mike Aaronson (ed) Precision Strike Warfare and International Intervention, Routledge 2015.

[22] Hambling, David. ”Drones may have attacked humans fully autonomously for the first time”. New Scientist. Retrieved 30 May 2021.

[23] ”Killer drone ’hunted down a human target’ without being told to”. New York Post. 29 May 2021. Retrieved 30 May 2021.

[24] Wagner 1982, p. 78, 79.

[25] Dunstan, Simon (2013). Israeli Fortifications of the October War 1973. Osprey Publishing. p. 16. ISBN 9781782004318. Retrieved 25 October 2015. “The War of Attrition was also notable for the first use of UAVs, or unmanned aerial vehicles, carrying reconnaissance cameras in combat.”

[26] Saxena, V. K. (2013). The Amazing Growth and Journey of UAV’s and Ballistic Missile Defence Capabilities: Where the Technology is Leading to?. Vij Books India Pvt Ltd. p. 6. ISBN 9789382573807. Retrieved 25 October 2015. “During the Yom Kippur War the Israelis used Teledyne Ryan 124 R RPVs along with the home-grown Scout and Mastiff UAVs for reconnaissance, surveillance, and as decoys to draw fire from Arab SAMs. This resulted in Arab forces expending costly and scarce missiles on inappropriate targets [...].”

[27] Dirman Hanafi, Mongkhun Qetkeaw, Mohd Nor, Mohd Than, “Simple GUI Wireless Controller of Quadcopter” International Journal of Communications, Network and System Sciences, vol. 06, No.1, 2013, pp.52-59.

[28] Mohd khan., “Quad copter Flight Dynamics”, International Journal of Scientific and Technology Research, vol. 3, Issue 8, 2014.

[29] S. H. Jeong, & S. Jung, “Novel design and position control of an omni-directional flying automobile (Omni-fly mobile), International Conference on Control Automation and Systems, 2010, pp.2480-2484.

[30] H. O. Lim & S. Machida, “Mechanism and control of coaxial double contra-rotation flying robot,” International Conference on Control Automation and Systems, 2010, pp.1109-1114.

[31] Prof. A.V. Javir, Ketan Pawar, Santosh Dhudum, Nitin Patel, “Design and fabrication of Quad copter”, Journal of the International Association of Advanced Technology and Science, vol. 16, 2015.

[32] Prem Kumar, Amirtharaja.S, Harie. S, Kishore Rohy. S, Kiruba. V, “Quad copter Video surveillance and control using computer”, International Journal of Electrical and Electronics Engineers, vol. 7, Issue 01, 2015.

[33] Abishini A H, Priyanka Bas B, Raque Bertilla A, Haston Amit Kumar, “Design and static structural analysis of an aerial and underwater drone”, International Research Journal of Engineering and Technology, vol. 5, Issue 4, 2018.

[34] Prajwal kumar M. Patil, Mallik arjun B. Koujalagi, Krishna Toli, “Modeling, Analysis & Fabrication of Quadcopter (Uav) With Payload Drop Mechanism,” International Journal of Research in Advent Technology, Special Issue, 2019.

[35] https://oscarliang.com/quadcopter-hardware-overview/

[36] https://ardupilot.org/rover/docs/common-pixhawk-overview.html#common-pixhawk-overview

[37] https://dojofordrones.com/raspberry-pi-drone/

[38] https://ardupilot.org/planner/index.html

[39] RCtoys, http://www.rctoys.com/.

[40] S. Bouabdallah.” Design and control of quadrotors with application to autonomous

[41] Direction Cosine Matrix IMU : Theory. William Premerlani and Paul Bizard. Mai2009flying”

[42] Nano, micro, small: The different drone types in India & if Jammu-like strike can be averted, The Print, 29 June 2021.

[43] Drones, Percepto (3 January 2019). ”The Differences Between UAV, UAS, and Autonomous Drones”. Percepto.

[44] John K. Borchardt, Unmanned aerial vehicles spur composites use, 2004 Elsevier Ltd. All rights reserved.

[45] Blum, Howard (2003). The eve of destruction: the untold story of the Yom Kippur War. HarperCollins. ISBN 9780060013998.

[46] Wagner 1982, p. 202.

[47] R. Mahony, V. Kumar, and P. Corke, “Multirotor Aerial Vehicles: Modeling, Estimation, and Control of a Quadrotor,” in IEEE Robotics and Automation Magazine, vol. 19, Sept. 2012, pp. 20-32

[48] P. I. Corke, Robotics, Vision & Control: Fundamental Algorithms in MATLAB. Berlin: Springer-Verlag, 2011.

[49] L. K. Burkamshaw, “Towards a low-cost quadrotor research platform.” Naval Postgraduate School, Monterey, California. March 2010.

[50] https://ardupilot.org/planner/index.html

[51] Ziegler, J.G & Nichols, N. B. (1942). ”Optimum settings for automatic controllers” . Transactions of the ASME. 64: 759–768.

[52] Getting the Raspberry Pi up and running with Raspbian,” ardupilot.org, [Online]. Available: http://ardupilot.org/dev/docs/making-a-mavlink-wifi-bridge-using-the- Raspberry-Pi.html.

[53] ArduPilot, ”ArduPilot,” Dev Team, 2016. [Online]. Available: http://ardupilot.org/dev/docs/Raspberry-Pi-via-mavlink.html.

[54] 3DR, ”Dronekit-python,” [Online]. Available: http://python.dronekit.io

[55] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. ”MediaPipe Hands: On-device Real-time Hand Tracking.” 2020.

[56] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. ”BlazePose: On-device Real-time Body Pose tracking.” 2020.

Abstract

This thesis is about making the conception, modelling, design and control of a mini quadrotor with a focus on hovering state system. It introduces a conception of a 500mm quadcopter, mathematical model for simulation and control of such system. It then describes the realization of the prototype quadrotor. The methodology is subsequently applied to design an autonomous quadrotor, that is able to fly via RF and autonomously via the implementation of python scripting, which allowed us to control our quadrotor using gesture commands.

Résumé

Cette thèse porte sur la conception, la modélisation, le design et le contrôle d'un mini quadrotor avec un accent sur le système d'état de vol stationnaire. Elle présente la conception d'un quadcoptère de 500mm, le modèle mathématique pour la simulation et le contrôle d'un tel système. Il décrit ensuite la réalisation du prototype de quadrotor. La méthodologie est ensuite appliquée à la conception d'un quadrotor autonome, capable de voler via RF et de manière autonome via l'implémentation de script python, qui nous a permis de contrôler notre quadrotor en utilisant des commandes gestuelles.

ملخص

تدور هذه الأطروحة حول النمذجة، التصميم، المحاكاة والتحكم في طائرة بدون طيار ذات اربع محركات مع التركيز على نظام حالة الطوفان. قدّمنا مفهوما لطائرة كوادكوبتر ٥٠٠ مم،
حيث تمت دراسة النموذج الرياضي لمحاكاة هذا النظام والتحكم فيه. تم بعد ذلك تجسيد الأطروحة و تصميم طائرة قادرة على الطيران عبر التردد اللاسلكي او بشكل مستقل عن طريق تنفيذ البرمجة النصية بايثون، مما سمح لنا بالتحكم في الطائرة باستخدام أوامر الإيماءات.