UNIVERSITY OF M'SILA

Faculty of Technology

Department of Electrical Engineering

Final Study Dissertation

In The Aim Of Obtaining a Master's Degree

-Academic-

Domain: Science and Technology

Option: Automatic

Specialty: Automatic and Systems

Presented By:

ISSAME MAHDJOUBI

Topic:

Object Detection For Quadrotor Using Deep Learning

Composed From:

Noun	Grade	Quality	University
Mr. A. BENYOUNES	MCA	Advisor	Univ M'sila
Mr. L. BENYATTOU	MCA	President	Univ M'sila
Mr. S. ZEGHLACHE	PROF	Examiner	Univ M'sila
Mr. A. AIB	MCB	Examiner	Univ M'sila
Mr. H. FELOUAT	MCB	Co Advisor	Univ Blida

Academic year 2021/2022

Acknowledgments

First and foremost, I must acknowledge my limitless thanks to **ALLAH**, the Ever-magnificent, the Ever-Thankful, for His help and bless by giving me the opportunity, courage, patience and enough energy to carry out and complete the entire of this thesis.

I would like to express my special thanks of gratitude to my teacher **Dr. BENYOUNES Abdelhafid** for his supervision and support.

I would like to express my sincere appreciation to the examining committee, professor, **Mr. ZEGHLACHE Samir**, **Mr. BENYATTOU Lotfi** and **Mr. AIB Abdelghani** for their interest of the work.

Finally, I would like to express my tremendous gratitude to my family and friends for supporting me all the time.

Abstract

In this work, a type of unmanned aerial vehicle (UAV) called quadrotor with an object detection system is the subject that we will talk about. The main objectives of our work is the detection of chosen objects from the quadrotor. For that, there are two parts presented, the first one, a detailed description of the mathematical effects that applied to the structure of our system and how we implemented our system with showing all the parts, add to this we built it using simulink with the PID and we showed the results. In the second part, we will talk about artificial intelligence generally and deep learning specifically and we will show how exactly the detection happen using the right technics and algorithms with Yolov5 and which version we will use. Finally we will make a real test with the real time showing our prototype working in the field.

Keywords: UAV, Object Detection, Quadrotor, Deep Learning, YOLO.

ملخص

في هذا العمل ، سوف نتحدث عن نوع من المركبات الجوية غير المأهولة يسمى كوادروتور مع نظام الكشف عن الأجسام. تتمثل الأهداف الرئيسية لعملنا في الكشف عن الكائنات المختارة من خلال الطائرة بدون طيار. لذلك ، حيث ينقسم هذا العمل اي قسمين ، الأول ، تقديم نموذج رياضي كامل للكوادروتور مع الاخذ بعين الاعتبار جميع الظواهر الفزيائية و الايروديناميكية التي يخضع لها هذا النظام و كيف قمنا بتركيب النموذج مع شرح كامل للقطع المستعملة بالاضافة الى اننا قمنا ببناء النظام ايضا على برنامج simulink و قمنا باختباره باستعمال ال PID و استخراج النتائج. في الجزء الثاني ، سنتحدث عن الذكاء الاصطناعي بشكل عام والتعلم العميق على وجه التحديد وسنوضح كيف يحدث اكتشاف الاشياء بالضبط باستخدام تقنيات و خوارزميات Yolo النسخة الخامسة. أخيرًا ، سنقوم بإجراء اختبار حقيقي في الوقت الفعلي لإظهار نموذجنا الأولي الذي يعمل في المجال.

الكلمات المفتاحية: الطائرات بدون طيار ، الكشف عن الاشياء ، كوادروتور ، التعلم العميق, يولو.

Résumé

Dans ce travail, un type de véhicule aérien sans pilote (UAV) appelé quadrotor avec un système de détection d'objets est le sujet dont nous parlerons. L'objectif principal de notre travail est la détection d'objets choisis à partir du quadrirotor. Pour cela, il y a deux parties présentées, la première, une description détaillée des effets mathématiques appliqués à la structure de notre système et comment nous avons implémenté notre système en montrant toutes les parties, ajouter à cela nous l'avons construit en utilisant simulink avec le PID et nous avons montré les résultats. Dans la deuxième partie, nous parlerons de l'intelligence artificielle en général et de l'apprentissage en profondeur en particulier et nous montrerons exactement comment la détection se produit en utilisant les bonnes techniques et algorithmes avec Yolov5 et quelle version nous utiliserons. Enfin, nous ferons un test réel avec le temps réel montrant notre prototype fonctionnant sur le terrain.

Mots-clés : drone, détection d'objets, quadrirotor, profonds d'apprentissage.

Content

List of figures, ... 2
List of Tables, ... 4
List of Symbols, ... 5

General Introduction, ... 7

CHAPTER I QUADROTOR MODELLING
I. Introduction, ... 7
I.2. Definition of Drone, ... 10
I.3. History of Drone, ... 10
I.4. Classification of Drone, ... 11
I.5. Drone Types, ... 12
I.6. Drone Applications, ... 13
I.6.1. Military Applications, ... 13
I.6.2. Civilian Applications, ... 13
I.7. UAV Instrumentation, ... 14
I.7.1. Inertial Units, ... 14
I.7.2. Accelerometer, ... 14
I.7.3. Magnometer, ... 14
I.7.4. Gyrometers and Gyroscopes, ... 15
I.7.5. Geolocation System, ... 15
I.7.6. Altimeter Barometric Pressure, ... 15
I.8. Dynamic Modeling Of The Quadrotor, ... 15
I.8.1. The Flight Possibilities Of The Quadrotor, ... 15
I.8.2. Dynamic Quadrotor Modeling, ... 17
I.8.3. Rotation Matrix, ... 18
I.8.4. Physical Effects Acting On The Quadrotor, ... 20
I.9. Mathematical Model According To Newton-Euler Formalism, ... 22
I.10. Nonholonomic Constraints, ... 24
I.11. Rotor Dynamics, ... 25
I.12. State Representation, ... 26
I.13. Conclustion, ... 28

CHAPTER II IMPLEMENTATION AND CONTROLLING OF QUADROTOR

II.1. Introduction, ... 29
II.4. Hardware Components, ... 29
II.4.1. Frame, ... 29
II.4.2. Rotors, ... 30
II.4.3. Propellers, ... 31
II.4.4. Electronic Speed Controllers (ESC), ... 31
II.4.5. Power, ... 32
II.4.6. Flight Controller Pixhawk, ... 33
II.4.6.1 Why Pixhawk?, ... 33
II.4.6.2. Why not arduino?, ... 35
II.4.7. Firmware, ... 35
II.4.8. Pixhawk sensors, ... 35
II.4.9. Telemetry, ... 36
II.4.10. Radio transmitter and receiver, ... 36
II.5. PX4 Software, ... 37
II.7. Mission Planner, ... 38
II.2. PID Controller, ... 39
II.3. Attitude Control of Quadrotor, ... 40
II.4. Simulation Result, ... 41
II.8. Conclusion, ... 44

CHAPTER III DEEP LEARNING FOR OBJECTDETECTION

III.1. Introduction, ... 45
III.2. Artificial Intelligence, ... 45
III.2.1. AI history, ... 45
III.2.2. What is machine learning, ... 46
III.2.3. Why we use machine learning, ... 46
III.2.4. Applications of Machine learning, ... 46
III.3. Deep Learning, ... 47
III.3.1. Neural Network, ... 47
III.3.2. Deep Neural Network, ... 47
III.3.3. Recurrent Neural Networks, ... 48
III.3.4. Convolution Neural Networks, ... 49

III.3.5. Fully Connected Layers, ... 51
III.4. Object Detection, ... 52
III.4.1. What is object detection?, ... 52
III.4.2. Modes and types of object detection, ... 52
III.4.3. Why is object detection important?, ... 53
III.4.4. Challenges, ... 53
III.4.4.1. Variable Number of Objects, ... 53
III.4.4.2. Sizing, ... 53
III.4.4.3. Modelling, ... 54
III.4.4.4. Illumination, ... 54
III.4.4.5. Occlusion, ... 54
III.4.5. preprocessing dataset (image labeling), ... 54
III.4.6. The Bounding boxes, ... 54
III.4.7. Object Detection Methods, ... 55
III.4.7.1. Object detection model evaluation, ... 55
III.4.7.2. mean Average Precision (mAP), ... 55
III.4.7.3. Classical Methods, ... 55
III.4.7.4. Deep Learning Methods, ... 56
III.4.7.5. Multimodal Methods, ... 57
III.4.8.Multiple Objects Detection, ... 57
III.5. Conclusion,........ 58

CHAPTER IV OBJECT DETECTION WITH YOLOV5
IV.1. Introduction, ... 59
IV.2. OBJECT DETECTION MODEL, ... 59
IV.2.1. What is Yolo?, ... 59
IV.2.2. Why YOLO?, ... 59
IV.2.2. YOLO Basics, ... 60
IV.3. IMPLEMENTING YOLOV5 ALGORITHM,........ 61
IV.3.1. Human/Person Detection, ... 61
IV.3.2. YOLO Releases,........ 62
IV.3.3. Yolov5 Setup,........ 63
IV.3.4. Example with Yolov5 using Google Colab, ... 66
IV.3.5. Video transferring from the quadrotor, ... 66
IV.3.6. Nvidia Jetson Embedded system, ... 66
IV.4. Conclusion,........ 68
General Conclusion, ... 71

List of figures

Figure I.1 Havilland Queen Bee K4227 of the first 10 production models, ... 11
Figure I.2 Kettering Bug (image: the one USA air force), ... 11
Figure I.3 (a) Hovereye (b) Seiko Epson _FR. (c) Proxflyer, ... 11
Figure I.4 TUAV: (a) Le X-47B– (b) Watchkeeper WK450– (c) Sperwer, ... 12
Figure I.5 (a) RQ-4 Global Hawk - (b) MQ-1 Predator - (c) Orion, ... 12
Figure I.6 (a) Carolo P50- (b) eBee, ... 12
Figure I.7 (a) quadrotor-(b) hexarotor-(c) octorotor, ... 13
Figure I.8 (a) Mine cleaner, (b) monitoring of power lines, ... 14
Figure I.9 Exemples of Inertial units, ... 14
Figure I.10 MEMS Sensors -(a) accelerometer- (b) gyrometer- (c) magnetometer, ... 15
Figure I.11 Roll motion of quadrotor, ... 16
Figure I.12 Pitch motion of quadrotor, ... 16
Figure I.13 Yaw motion of quadrotor, ... 16
Figure I.14 Vertical translation of quadrotor, ... 17
Figure I.15 The earth and body frames of a quadrotor, ... 17
Figure I.16 DC Motor Schematic Diagram 25
Figure II.1 quadrotor frame Types 29
Figure II.2 380mm Frame 30
Figure II.3 Rotor classification 30
Figure II.4 F15843 920 Kv Brushless rotor 31
Figure II.5 1045 Propellers 31
Figure II.6 Duty cycles 32
Figure II.7 Selected Electronic speed control 32
Figure II.8 Left: Lipo battery charger. Right: Lipo battery 2.65Ah 33
Figure II.9 Pixhawk 2.4.8 33
Figure II.10 Pixhawk Pinouts explain 34
Figure II.11 Pixhawk with the used Modules 34
Figure II.12 Pixhawk 2.4.8 pinouts 34
Figure II.13 Example of a mission 35
Figure II.14 Telemetry hardware description 36
Figure II.15 Radio Controller FlySky FS-i6X 37

Figure II.17 A schematic about the linked hardware to the quadrotor 37
Figure II.18 Mission Planner Software 38
Figure II.18 The prototype equipped. 39
Figure II.19 Simulink model used for quadrotor simulations.. 40
Figure II.20 Roll, Pitch and Yaw Angles stabilisation reponse (PID controller)................................ 42
Figure II.21 Roll, Pitch and Yaw Control Response using PID control 42
Figure II.22 Tracking responses results of the desired trajectories using PID control. 43
Figure II.23 Control Input Response of the desired trajectories using pid control............................ 43
Figure II.24 Altitude reponse using PID control 44
Figure II.25 Control Input Response using PID control.. 44
Figure III.1 Example of a simple neural network 47
Figure III.1 Example of a simple neural network 48
Figure III.3 Example of a Recurrent Neural Networks 48
Figure III.4 How the convolution works. 49
Figure III.5 Left: A 5x5 input feature map. Right: a 3x3 convolution. 50
Figure III.6 Left: input feature map. Right: Convolution result 50
Figure III.7 Left: Max pooling Right: the output max pooling operation 51
Figure III.8 illustrates the end-to-end structure of a convolutional neural network. 51
Figure III.9. an example of a detected object 52
Figure III.10 example about image labelling 54
Figure III.11 Intersection over Union of the model 55
Figure III.12 Precision recall for the dataset 55
Figure III.13 Fast R-CNN. 56
Figure IV.4 A comparative breakdown of YOLO and Fast R-CNN 60
Figure IV.5 Object detection with YOLO 61
Figure IV.6 YOLO three step object detection process 61
Figure IV.7 Example of Detecting People 62
Figure IV.8 Difference between Yolov5 versions 62
Figure IV.9 Python Installation 63
Figure IV.10 Cuda Downloading 64
Figure IV.11 Cuda Installation 64
Figure IV.12 The volume of Pytorch users 64
Figure IV.13 Pytorch Downloading 64

Figure IV.14 Pytorch Installing 63
Figure IV.15 Applying Yolov5 64
Figure IV.16 Testing Yolov5 on the front camera 64
Figure IV.17 Testing Yolov5 on Google Colab 66
Figure IV.18 Sample picture for detection on Google colab 67
Figure IV.18 Sample picture after running YOLO on Google colab 67
Figure IV.17 a)VTX Video Transmitter b) FPV camera c) VRX Video receiver 64
Figure IV.18 Jetson Nano Board 64
Figure IV.19 Jetson Nano Input/Output 64
Figure IV.20 The final prototype 68

List of tables

Table II.1 : Effects of increasing PID gains to closed-loop response 29
Table II.2 Mechanical and electrical parameters of quadrotor 31
Table II.3 PID control parameters 31
Table IV.1 Real-time object detection comparison of leading methods 60
Table IV.2 Details about the difference between Yolov5 versions 63
Table IV.3 All Coordinates of each object in our sample picture 63

List of symbols

R Rotation Matrix

Ω Angular velocity

F_g Gravity force

F_f Thrust forces

F_{dt} Drag forces

M_{gp} Gyroscopic moment of the propellers

M_{gm} Gyroscopic moment due to quadrotor movements

M_a Moments resulting from aerodynamic friction

M_f Moments caused by thrust and drag forces.

K_p Lift coefficient

d Drag coefficient

K_{dt} Translation drag coefficient

K_p Lift coefficient

K_{fa} Coefficient of aerodynamic friction

I Identity matrix

I_X Area moment of inertia about the x-axis

I_Y Area moment of inertia about the y-axis

I_Z Area moment of inertia about the z-axis

J Symmetrical inertia matrix

J_r Rotor's inertia

l Length of the arm between the rotor and the centre of gravity of the quadrotor.

k_{mot} Motor torque constant

L_{mot} Motor circuit inductance

R_{mot} Motor circuit resistance

U Control input vector

V Lyapunov Function

X State vector

r Rotors' relative speed

ω Angular body rates

φ Roll angle

φ_d Desired roll φ

ψ Yaw angle

ψ_d Desired yaw ψ

θ Pitch angle

θ_d Desired pitch

g Gravitational acceleration

m Mass of quadrotor

ξ Position vector of the quadrotor

r Quadrotor's position in the navigation frame

x_d Desired x position

y_d Desired y position

z_d Desired altitude z

ωi Angular speed of rotor i

Acronyms

UAV Unmanned Aerial Vehicle

DOF Degree of Freedom

EKF Extended Kalman Filter

CIA Central Intelligence Agency

AI Artificial Intelligence

COG Centre of gravity

COA Centroid of Area

FPV First Person View

VTX Video Transmitter

VRX Video Receiver

ML Machine Learning

RTL Return To Launch

General Introduction

In this thesis we will focus on the development of a quadrotor mounted with an AI camera to scan a specific area and detect objects there.

Quadrotor, as an Unmanned Aerial Vehicle (UAV), is a rotorcraft equipped with four powered rotors laid up symmetrically around its center and we can classify it as a Vertical Take-Off and Landing (VTOL) aircraft. Because of its unique characteristics, such as low dimension, good maneuverability, simple mechanics, payload capability, and vertical, stationary, and low-speed right, this structure recently gathered popularity among several applications, in particular for surveillance, imaging, dangerous environments, indoor navigation, mapping, pipeline inspection, football stadium security, etc. **[1]**

As the main drawback, the high energy consumption of the Quadrotor can be mentioned and because of the technical insufficiencies in manufacturing of highly durable batteries, the operational time of Quadrotors is restricted to barely more than an hour. In addition because of its payload limitations, it is impossible to mount spare power supplies on the Quadrotor.**[1]**

The quadrotor is an under-actuated system with six outputs and four inputs that can be controlled by changing the rotor's speed, which has a simple design and easy control.

However, Quadrotor is an inherently unstable system, based on several reasons elaborated as follows. First, their dynamic model features include multivariable, non-linearity, strong interconnections, being under-actuated and static instability. Secondly, the quadrotor is affected by body gravity, air resistance, propeller driving force, gyro impacts, and so on. Hence, there is uncertainty in the dynamic model, that makes the control of the quadrotor quite a complex and difficult task mainly due to its under-actuated properties and nonlinearities, which accentuates the need for a proper controller.**[2,3,4,5]**

Also we will talk about YOLO models which are used for Object detection with small size, more speed and high performance. YOLO divides an image into a grid system, and each grid detects objects within itself. They can be used for real-time object detection based on the data streams. They require very few computational resources.

In this work, a detailed model of the quadrotor will be given taking into account the high-order non-holonomic constraints as well as the various physical phenomena that act on this system. And we will focus more on the integrated AI system that will help us to find objects remotely and from the sky.

This thesis is composed into 4 chapters, which are briefly introduced in the following paragraph:

Chapter 1: the first chapter gives an overview of the state of the art of the drones, we will present a general description by define the drones and their history, and then we will present the different types of drones, their classification, and their fields of application, as well as their instrumentation. In addition the dynamic modeling of the quadrotor; And, we will present the flying possibilities of the quadrotor as well as the different forces and moments acting on the quadrotor. Then, we highlight the dynamic modeling of the quadrotor. Also the dynamics are explained from the basic concepts to the Newton-Euler formalism. Finally, we will give the state model for the control of this system.

Chapter 2: In this chapter we will talk about how the quadrotor can be controlled with PID, after that we will pass to the hardware requirements to build a drone as well a the main controlling board Pixhawk that we used and how it works

Chapter 3: Object detection has become ingrained in our daily lives, with applications ranging from security to automated vehicle systems. So in this chapter we will see the evolution of the artificial intelligence and how it became important in all technology fields, as well as deep learning and the ways and algorithms of detecting an object from the camera to the recognition of an object and making a bounding box to it.

Chapter 4: Computer vision technology uses a variety of imaging systems instead of visual organs as input means, using computers to replace the brain to complete the processing and interpretation of visual information. So in this chapter we will use the Algorithms of Yolo to detect the objects we need and explaining the components we used to transfer the real time video from the drone to the processing board Nvidia Jetson Nano.

Part I

Modelling and Control of a Quadrotor

Chapter I

Quadrotor modelling

I. Introduction

Aerial robotics is a vast and interdisciplinary field. Drones are unmanned flying crafts on board, remotely controlled and reusable. It can be defined as a mechanical, electronic, and computer system controlled or programmed to perform tasks that are too repetitive dangerous, or difficult to be done directly by human beings, They are also known under the name,” UAV(Unmanned Aerial Vehicle)”.

In this chapter, we will shortly present the definition of a drone and its history, after that we will explain the diverse types of drones, their classification according to their characteristic dimensions, endurance, and operational altitude, finally we will present their fields of applications and their instrumentation, which allow them to know their state and perform their tasks.

I.2 Definition of Drones

Unmanned Aerial Vehicle (UAV) is a generic term for unmanned aircraft, UAV or drone, is a flying vehicle, does not carry a human operator, uses aerodynamic forces to provide vehicle lift, can fly autonomously or be piloted remotely, can be expendable or recoverable, and can carry a fatal or nonfatal payload. Their principal use is military for reconnaissance or surveillance missions, without risk of loss of life, they are well adapted for the realization of missions that would potentially put a crew in danger or which require permanence on the zone, which would be tedious for a crew on board. **[5]**

I.3 History of Drones

The history of UAVs is one of the cyclical developments often centered on military conflicts. The appearance of the first UAVs flying robots without a human pilot onboard returns on the date of the First World War. Drones have been used mainly as a targeted military for combat training. Before the 1900s, balloons were used by Austria to drop bombs on Venice (Italy).

After 10 years, the United States produced the drone "Kettering Bug" during the First World War. Within 10 years, some "Sperry Messenger" were converted into bombs flying and thus became the first real unmanned aircraft. Between 1930 and 1940, British produced 40 unmanned aircraft as a practice target for defenses anti-aircraft Known as "Queen Bees", their name inspired the use of the term "drone", During the Second World War, Reginald Denny convinced the US army to use its radio-controlled planes "OQ-2 Radio Planes" to train their anti-aircraft defense.

While The Germans produced the bomber "V-1" to bomb London. Between 1950 and 1960, the "Firebee" was developed during the wars of Korea and Vietnam for surveillance missions entrusted to drones. Between 1960 and 1970, the US Army began production of its first stealth drone, the "AQM-34 Ryan Firebee".

The Central Intelligence Agency (CIA) called for the production of a drone ultra-secret, which was able to fly at speeds of 3.3 Mach and altitudes of more than 27000 km. during the 1990s the UAVs gained wide acceptance as a useful military tool. Then until the year 2000, a variety of new drones was developed for military and non-military applications such as solar-powered drones "Helios" and "Pathfinder", and since 2000 so far more than 23 countries are developing drones and more than 41 have in services. The new drone brings many innovative ideas and uses. **[6]**

Figure I.1 Havilland Queen Bee K4227
One of the first 10 production models

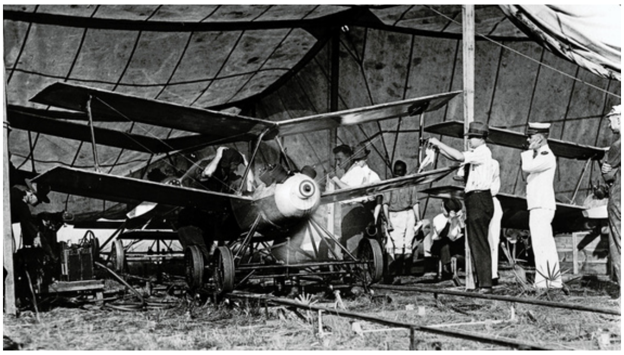

Figure I.2 Kettering Bug (the USA air force)

I.4 Classification of Drones

The classification of drones is a very hard practice, as far as it is different depending on the country and the organization. On the other hand, this classification depends on various parameters, including the flight range, the flight altitude, endurance, size (length, wingspan, etc.), weight, or their wings (fixed, rotating, or swinging). We will first present a general classification of drones, according to their radius action, endurance, altitude, and size:

- **Mini drones MAV (Mini Air Vehicule) & Micro / Nano UAVs:**

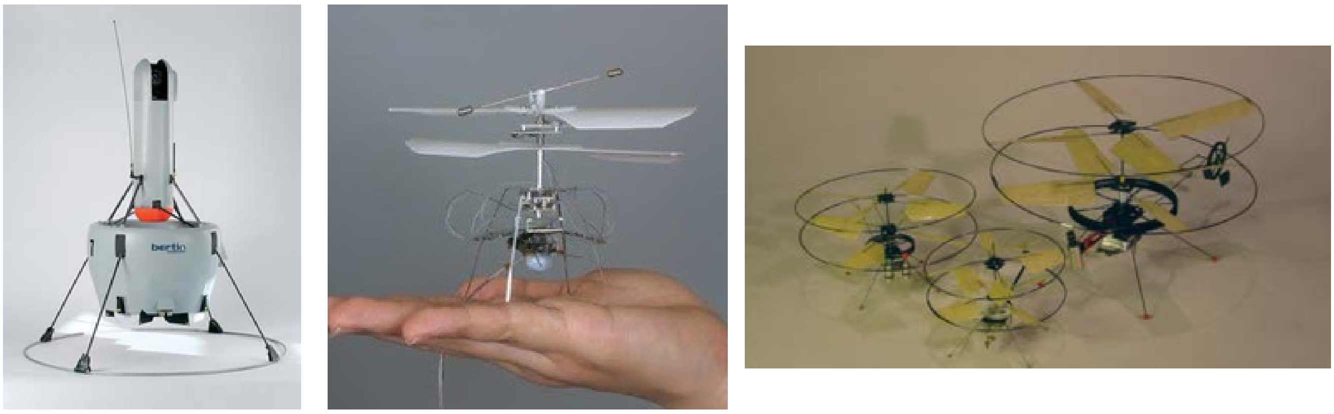

Figure I.3 – (a) Hovereye - (b) Seiko Epson μFR - (c) Proxflyer. **[7]**

- **TUAV tactical UAVs (Tactical Unmanned Aerial Vehicle):**

Figure I.4 TUAV: (a) Le X-47B - (b) Watchkeeper WK450 - (c) Sperwer **[7]**

- **HALE Drones (High Altitude Long Endurance) & MALE Drones (Medium Altitude Long Endurance):**

Figure 1.5: HALE & MALE : (a) RQ-4 Global Hawk - (b) MQ-1 Predator - (c) Orion. **[7]**

I.5 Drone Types

The important technical characteristic of drones is the type of drone. The main drone types are fixed-wing drones and rotary-wing drones. The majority of existing drones can be defined within these two types. However, there are other types like hybrid drone systems and ornithopters. These characteristics will be discussed to further visualize these technological characteristics.

- **Fixed Wing drones:** These drones consist of a pair of wings providing lift, with a fuselage, drifting, and a tail the propulsion is provided by one or more propellers.**[7]**

Figure 1.6 – (a) Carolo P50- (b) eBee. **[7]**

Multirotor: Multi-rotor UAVs are certainly the most common among autonomous air vehicles. These devices are usually equipped with four rotors (quadrotor), six and eight rotors, the mechanical simplicity makes this type of drone is widely used for the realization of experimental platforms at reduced costs **[8]**.

Figure I.7. (a) quadrotor-(b) hexarotor-(c) octorotor **[6]**.

I.6. Drone Applications

UAVs were developed to replace humans in hostile environments or dangerous situations. These vehicles have many advantages, it is very probably in the civil as in the military field, which these drones are called to play the biggest role, and this is because of their flexibility, and their versatility of use. In this section, we will present a set of their applications.

I.6.1. Military Applications

• **Mine clearance:** Drone contains an infrared camera that allows it to detect mines. Without risking a life.
• **Surveillance and intelligence:** Uses of drones for surveillance and intelligence are very diverse, they are equipped with optical sensors. These devices can collect lots of information on the enemies.
• **Combat support:** Drones can be armed with missiles that allows them to conduct high precision military attacks.
• **Search and rescue:** Drones can perform search missions in a wide area to locate casualties thus assisting the rescue team instead of a long search operation.

I.6.2. Civilian Applications

• **Science and research:** They can help scientists on their researches to observe different occurrences in nature or a particular environment from the sky.
• **Inspections:** checking many systems such as power lines, wind turbines, and pipelines
• **Unmanned cargo system:** They can serve in delivering of lightweight packages to customers.
• **Forest fire control:** Drones provide valuable help in the immediate response to a forest fire through early detection of fires.
• **Geographic Mapping:** Drones can reach difficult-to-access locations like eroded coastline or mountaintops and acquire very high-resolution data to create 3D maps.

Figure I. 8. (a) Mine cleaner, (b) monitoring of power lines .

I.7. UAV Instrumentation

I.7.1. Inertial units

An inertial measurement unit or also called (IMU) "Inertial Measurement Unit" is a navigation system providing the attitude, the vector of speed, and position of an object (see **figure I.10**). It mainly contains three accelerometers and three gyrometers measuring the non-gravitational acceleration and the instantaneous speed of rotation of the vehicle for an inertial frame of reference. MEMS technology is currently the most used technology, the sensors being simple, small, light, and economical. Some examples of these sensors are shown in **Figure I.11**. This technology makes it possible to have accelerometers, gyroscopes, and magnetometers integrated into an electronic circuit weighing around ten grams.

Figure 1.9 – Exemples of Inertial units

I.7.2. Accelerometer

An accelerometer allows, as its name suggests, to measure the nongravitational (or specific) acceleration of the object that supports it along with one, two, or three axes. A three-axis accelerometer makes it possible to measure the three linear accelerations along three orthogonal axes. Their principle is based on the deformation or displacement of a body during an acceleration. The advantage of an accelerometer is its great ease in revealing a multitude of data (acceleration, speed, displacement, force, etc.).

I.7.3. Magnetometer

A magnetometer, also called a magnetic compass, is a sensor allowing to measure the direction and/or the intensity of a magnetic field, and in particular, the direction of the earth's magnetic field also called magnetic induction. However, this sensor is sensitive to external magnetic disturbances from the drone.

I.7.4 Gyrometers and Gyroscopes

A gyrometer is a sensor that measures an instantaneous speed of rotation around an axis. A three-axis gyrometer makes it possible to measure the speed of rotation on the three axes of the moving frame (roll, pitch, and yaw speeds). A gyroscope measures the angular position along the axis where it is mounted in the inertial frame.

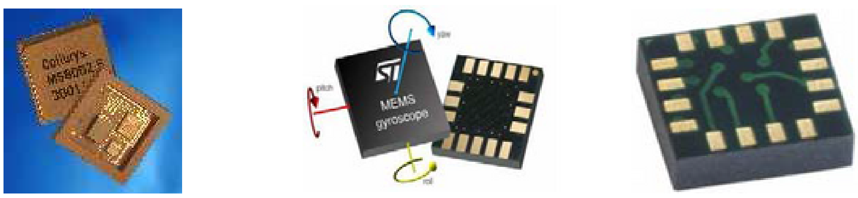

Figure I.10 – MEMS Sensors -(a) accelerometer- (b) gyrometer- (c) magnetometer

I.7.5 Geolocation system

The Global Navigation Satellite System (GNSS) encompasses all types of satellite navigation systems. GPS stands for “Global Positioning System” and is a satellite positioning and navigation system. This system is made up of twenty-four satellites spread over six orbits (four satellites per orbit) revolving around the globe (2 revolutions in 24 hours) and located at an altitude of 20,184 kilometers. Measurement errors are mainly due to interference which alters signal propagation.

I.7.6. Altimeter barometric pressure

A barometric altimeter is a sensor for determining the altitude between the sensor level and the reference level. The measurement of this sensor is very sensitive to changes in atmospheric conditions (gusts of wind). For miniature drones. It is generally best to use these sensors for interior applications.

I.8. Dynamic Modeling Of The Quadrotor

I.8.1. The flight possibilities of the quadrotor:

The quadrotor is a flying robot equipped with four fixed-pitch rotors, placed at the ends of a rigid cross-shaped structure, two rotors in the same pitch are placed opposite and rotate in the same direction, while the other two rotors rotate in the opposite direction, Clockwise and Counter Clockwise.

The base movements of the Quadrotor are achieved by varying the speed of each rotor, thereby changing the thrust produced, to get better information; we will discuss the possible movements of the Quadrotor.**[7]**

- **Roll Motion:**

This movement is created by the difference in lift force between the rotors 1 and 3; This difference provides a rotation around the x-axis coupled with a translation motion along the y-axis. Figure I.12 shows the roll motion of the quadrotor.

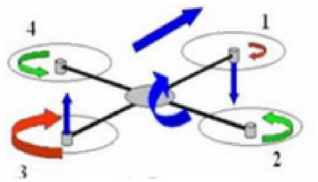

Figure I.11 – Roll motion of quadrotor.

- **Pitch Motion:**

This movement is created by the difference in the lift force between rotors 2 and 4, this difference provides a rotation around the y-axis coupled with a translation motion along the x-axis. Figure I.13 shows the pitch motion of the quadrotor, left and right rotations.

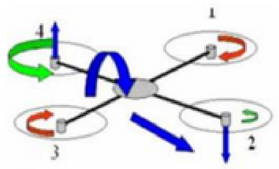

Figure I.12 – Pitch motion of quadrotor.

- **Yaw Motion:**

This movement is ensured by the difference in the velocity between the rotors which are on the same axis (1, 3) and (2, 4) This difference provides a rotation around the z-axis. Figure I.14 shows the yaw motion of the quadrotor.

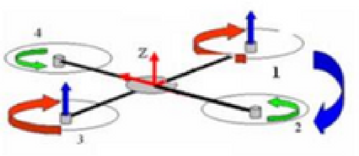

Figure I.13 – Yaw motion of quadrotor.

- **Vertical translation:**

The vertical movement is ensured by the lift forces of four rotors at the same time. To raise or lower the quadrotor, it is essential to increase or decrease the power of all the motors. Figure I.15 shows the vertical translation of quadrotor

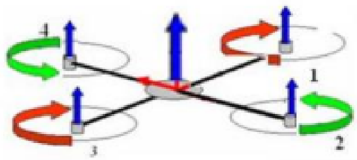

Figure I.14 – Vertical translation of quadrotor.

In the vertical translation we find:

• The lift force is greater than the weight force of the quadrotor > the movement is upward.

• The lift force is equal to the weight force of the quadrotor > the movement is hovering.

• The lift force is less than the weight force of the quadrotor > the vertical movement is downward along the z-axis.

I.8.2. Dynamic quadrotor modeling

In order to get the mathematical model of the quadrotor, first thing first we need to define two coordinates systems, which will be used **[9]**:

• Earth fixed frame (E-frame, $\mathscr{F}$E)

• Body fixed frame (B-frame, $\mathscr{F}$B)

Some quadrotor physical properties are measured in $\mathscr{F}$E (roll, pitch and yaw angles, angular velocities), while some properties are measured in $\mathscr{F}$B (linear accelerations) **[9]**.

The Earth reference frame is an inertial frame fixed on a specific place at ground level as its name implies, where the positive direction of the Z-axis is in the direction from the earth, while the body frame is at the center of the quadrotor body, with the positive direction of the XB axis with its x-axis pointing towards propeller 1. The positive direction of the YB axis points towards propeller 4, while the positive direction of the ZB axis is in the direction of the propeller's thrust forces. Figure I.16 shows the earth and body frames of the quadrotor.

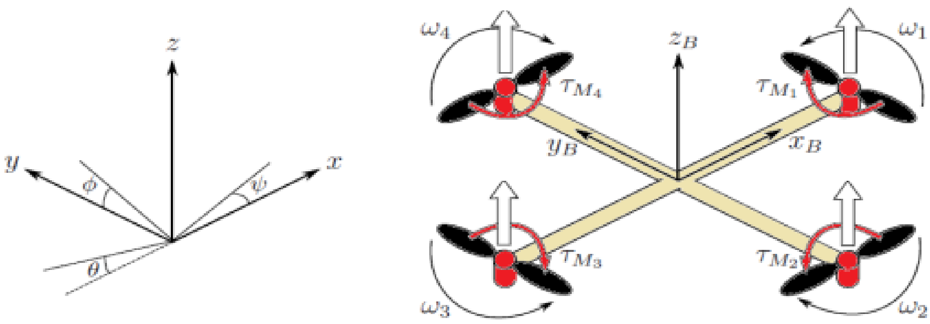

Figure I.15 The earth and body frames of a quadrotor.

The modeling of flying robots is a delicate task since the dynamics of the system are strongly nonlinear and fully coupled. To better understand; the dynamic model developed below, here are the different working hypotheses **[10]**:

- The quadrotor structure is rigid and symmetrical.
- The propellers are supposed solid.
- The mass center and Body fixed frame origin are supposed to coincide.
- Thrust and drag forces are proportional to the square of the propeller's speed.

I.8.3. Rotation Matrix

The distance between the Earth frame and the body frame describes the absolute position of the center of mass of the quadrotor $\xi = [x\ y\ z]T$. The rotation R from the body frame to the earth frame describes the orientation of the quadrotor.

The orientation between the references can be given by an orthogonal rotation matrix, the parameterization of the rotation matrix by Euler angles is often used in robotic applications. The orientation of the quadrotor is described using roll, pitch, and yaw angles (φ; θ and ψ) representing rotations about the X, Y, and Z-axes respectively. To find the elements of the rotation matrix R, we use the Euler angles:

➢ Roll angle φ such as $-\pi/2 < \varphi < \pi/2$

➢ Pitch angle θ such as $-\pi/2 < \theta < \pi/2$

➢ Yaw angle ψ such as $-\pi < \theta < \pi$

The rotation around the x- axis $R(x, \varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}$

The rotation around the y- axis $R(y, \theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$

The rotation around the z- axis $R(z,\ ) = \begin{bmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Therefore, the rotation matrix R is given by: R=*Rotz* ( )   (θ)*Rotx* (φ) Thereby:

$$R = \begin{bmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}$$

The rotation matrix from the body frame to the inertial frame is:

$$R=\begin{bmatrix} c\psi c\theta & c\psi s\theta s\varphi - s\psi c\varphi & c\psi s\theta c\varphi + s\psi s\varphi \\ s\psi c\theta & s\psi s\theta s\varphi - c\psi c\varphi & s\psi s\theta c\varphi - c\psi s\varphi \\ -s\theta & c\theta s\varphi & c\theta c\varphi \end{bmatrix} \quad \textbf{(I.1)}$$

Where: c=cos, and s=sin

The passage between the references is ensured by a matrix of transformation final T:

$$T=\begin{bmatrix} R & \xi \\ 0 & 1 \end{bmatrix} \quad \textbf{(I.2)}$$

Linear velocities

Linear velocities $vx^e\ vx^e\ vx^e$ in the fixed reference according to the linear velocities $vx^b\ vx^b\ vx^b$ in the body frame are given by:

$$v=\begin{bmatrix} vx^e \\ vy^e \\ vz^e \end{bmatrix} = R \times \begin{bmatrix} vx^b \\ vy^b \\ vz^b \end{bmatrix} \quad \textbf{(I.3)}$$

Angular velocities

The angular velocities $\Omega x\ \ \Omega y\ \ \Omega z$ in the earth frame are expressed as a function of the angular velocities $\dot{\varphi}$, $\dot{\theta}$, $\dot{\psi}$ in the body frame, we have:

$$\Omega=\begin{bmatrix} \Omega x \\ \Omega y \\ \Omega z \end{bmatrix} = \begin{bmatrix} \dot{\varphi} \\ 0 \\ 0 \end{bmatrix} + Rot_x(\varphi)^{-1}\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + (Rot_x(\theta)Rot_y(\varphi))^{-1}\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \quad \textbf{(I.4)}$$

After the simplification, we get:

$$\Omega=\begin{bmatrix} \Omega x \\ \Omega y \\ \Omega z \end{bmatrix} = \begin{bmatrix} \dot{\varphi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \dot{\theta} c\varphi \\ -\dot{\theta} s\varphi \end{bmatrix} + \begin{bmatrix} -\dot{\psi} s\theta \\ \dot{\psi} s\varphi\, c\theta \\ \dot{\psi} c\varphi\, c\theta \end{bmatrix} = \begin{bmatrix} \dot{\varphi} - \dot{\psi}\, s\theta \\ \dot{\theta} c\varphi + \dot{\psi} c\varphi\, c\theta \\ \dot{\psi} c\varphi\, c\theta - \dot{\theta} s\varphi \end{bmatrix} \quad \textbf{(I.5)}$$

$$\Omega=\begin{bmatrix} 1 & 0 & -s\theta \\ 0 & c\varphi & s\varphi c\theta \\ 0 & -s\varphi & c\varphi c\theta \end{bmatrix} \times \begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} \quad \textbf{(I.6)}$$

When the quadrotor makes small rotations, we can suppose the following approximations:

$\cos\varphi = \cos\theta = c\psi = 1$, and $\sin\varphi = \sin\theta = \sin\psi = 0$

Thus, the angular velocity will be: $\Omega = [\dot{\varphi}\ \dot{\theta}\ \dot{\psi}]^T$ **(I.7)**

I.8.4. Physical effects acting on the quadrotor

There are several physical effects acting on a quadrotor, including forces, moments and gyroscopic effect. We will talk about these effects:

- **Forces:**

- Gravity force: it is given by; $Fg = m \times g$, with: m is the mass of quadrotor and g the gravity.

- Thrust forces: which are forces caused by the rotation of the motors, they are perpendicular to the plane of the propellers. These forces are proportional to the square of the rotor's angular speed (ω):

$$F_f = k_p \omega_i^2 \quad \textbf{(I.8)}$$

Where $i = \overline{1:4}$ and kp Is the lift coefficient, it depends on the shape and the number of blades and the density of the air.

- Drag forces: The drag force is the coupling between a force of pressure and the force of viscous friction; in this case, we have two drag forces acting on the system, as the following:

✓ The drag in the propellers: it acts on the blades; it is proportional to the density of the air, to the shape of the blades and to the square of the angular velocity of the propeller, it is given by the following relation:

$$T_h = d\,\omega^2 \quad \textbf{(I.9)}$$

With d is the drag coefficient it depends on the fabrication of the propeller.

✓ The drag along the axes (x, y, z): it is due to the motion of the quadrotor body:

$$F_{dt} = K_{dt}\xi \quad \textbf{(I.10)}$$

With *Kdt* is the translation drag coefficient.

- **Moments**

There are several moments acting on the quadrotor, these moments are due to thrust and drag forces and gyroscopic effects

- Thrust moments

Starting with the moments around the body frame's x-axis, this moment is created by the difference between the lift forces of rotors 2 and 4, this moment is given by the following relation:

$$M_x = l(F_4 - F_2) = lk_p(\omega_4 - \omega_2) \quad \textbf{(I.11)}$$

Where *l*: is the length of the arm between the rotor and the centre of gravity of the quadrotor.

For the moments around the body frame's y-axis, also, this moment is created by the difference between the lift forces, but in this case, between rotors 1 and 3. This moment given by the following relation:

$$M_y = l(F_3 - F_1) = lk_p(\omega_3 - \omega_1) \quad \textbf{(I.12)}$$

- Drag moments:

as for the moments about the body frame's z-axis. This moment due to a reactive couple caused by the drag couples in each propeller. It given by the following relation:

$$M_z = K_d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \quad \textbf{(I.13)}$$

Moments resulting from aerodynamic friction, given by:

$$M_a = k_{fa}\Omega^2 \quad \textbf{(I.14)}$$

Where *kfa*: The coefficient of aerodynamic friction and is the angular velocity.

- **Gyroscopic effect**

The gyroscopic effect is a physical effect in which gyroscopic torque or moments attempt to align the spin axis of the rotor along the inertial z-axis.
In our case there are two gyroscopic moments, the first is the gyroscopic moment propellers, the other is the gyroscopic moment due to quadrotor movements.

- The gyroscopic moment of the propellers: it is given by the following relation:

$$M = \Sigma_1^4 \Omega \wedge J_r [0 \quad 0 \quad (-1)^{i+1}\omega_i]^T \quad \textbf{(I.15)}$$

With: ω_i is angular speed of rotor i, *Jr* is the rotor inertia and $\wedge$ is the vector product.

- The gyroscopic moment due to quadrotor movements: it is given by the following relation:

$$M_{gm} = \Omega \wedge J\Omega \quad \textbf{(I.16)}$$

With: J is the inertia of the system.

I.9. Mathematical Model according to Newton-Euler Formalism

After using Newton-Euler formulation, the equations will be written in the following form **[8,11]**:

$$\begin{cases} \dot{\xi} = V \\ m\ddot{\xi} = F_f + F_{dt} + F_g \\ J\dot{\Omega} = -M_{gm} - M_{gp} - M_a + M_f \end{cases} \qquad \textbf{(I.17)}$$

Where ξ is the position vector of the quadrotor and **m** is the mass of the quadrotor

J: is the symmetrical inertia matrix of dimensions (3x3), it is provided by:

$$J = \begin{bmatrix} I_x & 0 & 0 \\ 0 & Iy & 0 \\ 0 & 0 & Iz \end{bmatrix} \qquad \textbf{(I.18)}$$

F_f: is the total force generated by the four rotors and we write it as follows :

$$F_f = R[\,0 \quad 0 \quad \Sigma_{i=1}^{4} F_i\,]^T \qquad \textbf{(I.19)}$$

$$F_f = \begin{bmatrix} c\psi s\theta c\varphi + s\psi s\varphi \\ s\psi s\theta c\varphi - c\psi s\varphi \\ s\theta c\varphi \end{bmatrix} k_p \Sigma_{i=1}^{4} \omega_i^2 \qquad \textbf{(I.20)}$$

F_{dt} : drag force along the axes (x, y, z) given by:

$$F_{dt} = \begin{bmatrix} -K_{dtx} & 0 & 0 \\ 0 & -K_{dty} & 0 \\ 0 & 0 & -K_{dtz} \end{bmatrix} \dot{\xi} \qquad \textbf{(I.21)}$$

Fg: The gravity force:

$$Fg = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \qquad \textbf{(I.22)}$$

M_f : The moments caused by thrust and drag forces

$$M_f = \begin{bmatrix} lk_p(\omega_3^2 - \omega_1^2) \\ lk_p(\omega_4^2 - \omega_2^2) \\ K_d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \qquad \textbf{(I.23)}$$

Ma: Is the moments resulting from the aerodynamic friction and it is provided by:

$$\mathrm{M}a = \begin{bmatrix} -K_{fax}\dot{\varphi}^2 \\ -K_{fay}\dot{\theta}^2 \\ -K_{faz}\dot{\psi}^2 \end{bmatrix} \quad \textbf{(I.24)}$$

I.9.1 - Translational Movement

The translation equations of movement for the quadrotor are primarily based totally on Newton`s 2nd regulation and they're derived the Earth inertial frame, we obtain:

$$m\ddot{\xi} = F_f + F_{dt} + F_g \quad \textbf{(I.25)}$$

And after we replace forces by their formula, we get:

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} cos\psi sin\theta cos\varphi + sin\psi sin\varphi \\ sin\psi sin\theta cos\varphi - cos\psi sin\varphi \\ sin\theta cos\varphi \end{bmatrix} K_p(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) + \begin{bmatrix} -K_{dtx}\dot{X} \\ -K_{dty}\dot{y} \\ -K_{dtz}\dot{Z} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \quad \textbf{(I.26)}$$

Therefore, we get the differential equations that define the translation motion:

$$\begin{cases} \ddot{x} = \frac{1}{m}\,(cos\psi sin\theta cos\varphi + sin\psi cos\varphi)\;\; K_p(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)\; \frac{-K_{dtx}\dot{X}}{m} \\ \ddot{y} = \frac{1}{m}\,(sin\psi sin\theta cos\varphi - cos\psi sin\varphi)\;\; K_p(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)\; \frac{-K_{dtx}\dot{y}}{m} \\ \ddot{z} = \frac{1}{m}\,(sin\theta cos\varphi)\;\; K_p(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)\; \frac{-K_{dtx}\dot{Z}}{m} - g \end{cases} \quad \textbf{(I.27)}$$

I.9.2 - Rotational movement

The rotational equations of movement are derived withinside the body frame, the use of Newton's law, about rotation movement, we find :

$$\mathrm{J}\ddot{\Omega} = -\mathrm{M}_{gm} - \mathrm{M}_{gp} - \mathrm{M}_{a} + \mathrm{M}_{f} \quad \textbf{(I.28)}$$

We replace each moment by its formula, we get

$$\mathrm{J}\ddot{\Omega} = -\begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \wedge \left(\begin{bmatrix} I_x & 0 & 0 \\ 0 & Iy & 0 \\ 0 & 0 & Iz \end{bmatrix} \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \right) \begin{bmatrix} -J_r\bar{\Omega}_r\dot{\varphi} \\ -J_r\bar{\Omega}_r\dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} -K_{fax}\dot{\varphi}^2 \\ -K_{fay}\dot{\theta}^2 \\ -K_{faz}\dot{\psi}^2 \end{bmatrix} + \begin{bmatrix} lk_p(\omega_3^2 - \omega_1^2) \\ lk_p(\omega_4^2 - \omega_2^2) \\ K_d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \quad \textbf{(I.29)}$$

After that we get the differential equations that define the rotation movement:

$$\begin{cases} I_X\ddot{\varphi} = \dot{\theta}\dot{\psi}[(I_Z - I_Y) + J\bar{\Omega}_r\dot{\theta} - K_{fax}\dot{\varphi}^2 + lk_p(\omega_3^2 - \omega_1^2) \\ I_Y\ddot{\theta} = \dot{\theta}\dot{\psi}[(I_Z - I_X) + J\bar{\Omega}_r\dot{\varphi} - K_{fay}\dot{\theta}^2 + lk_p(\omega_4^2 - \omega_2^2) \\ I_Z\ddot{\psi} = \dot{\theta}\dot{\psi}[(I_Y - I_Y) - K_{faz}\dot{\psi}^2 + K_d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{cases} \quad \textbf{(I.30)}$$

With: $\bar{\Omega}_r = (\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)$

$$\begin{bmatrix} U1 \\ U2 \\ U3 \\ U4 \end{bmatrix} = \begin{bmatrix} K_p & K_p & K_p & K_p \\ -K_p & 0 & K_p & 0 \\ 0 & -K_p & 0 & K_p \\ K_d & -K_d & K_d & -K_d \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \rightarrow \begin{cases} U1 = K_p(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\ U2 = K_p(\omega_3^2 - \omega_1^2) \\ U3 \ K_p(\omega_4^2 - \omega_2^2) \\ U4 = K_p(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{cases} \quad \textbf{(I.31)}$$

As a result, the dynamic model governing the quadrotor is as follows:

$$\begin{cases} \ddot{\varphi} = \frac{1}{I_x}\{(I_Y \ I_Z)\dot{\theta}\dot{\psi} + J\bar{\Omega}_r\dot{\theta} - K_{fax}\dot{\varphi}^2 + lU_2\} \\ \ddot{\theta} = \frac{1}{I_Y}\{(I_Z \ I_X)\dot{\varphi}\dot{\psi} - J\bar{\Omega}_r\dot{\varphi} - K_{fay}\dot{\theta}^2 + lU_3\} \\ \ddot{\psi} = \frac{1}{I_Z}\{(I_X \ I_Y)\dot{\theta}\dot{\psi} - K_{faz}\dot{\psi}^2 + lU_4\} \\ \ddot{x} = \frac{1}{m}\{-K_{dtx}\dot{X} + U_x U_1\} \\ \ddot{y} = \frac{1}{m}\{-K_{dtx}\dot{y} + U_y U_1\} \\ \ddot{z} = \frac{1}{m}\{-K_{dtx}\dot{Z} + cos\varphi cos\theta U_1\} - g \end{cases} \quad \textbf{(I.32)}$$

With:

$$\begin{cases} U_x = (cos\psi sin\theta cos\varphi + sin\psi sin\varphi) \\ U_y = (sin\psi sin\theta cos\varphi - cos\psi sin\varphi) \end{cases} \quad \textbf{(I.33)}$$

I.10. Nonholonomic constraints

In a perfect case therobot can move in any directionand that means the robot has many degrees of freedom as parameters of configuration, the number of degrees of freedom representing all the movements that the robot can achieve through its joints. Regardless, a real robot is subject to constraints during its movement. We can consider two major classes:

- The holonomic constraints, which reduces the set of admissible configurations without calling into question the existence of a path.
- The nonholonomic constraints that constrain the field of robot velocities.

In the context of our project, we will focus on the nonholonomic constraints characterizing the relations which link the angles φ and θ and the components of the vector of acceleration. From the dynamic equations of translation, we can extract the expressions of the nonholonomic constraints **[12]** :

We notice :

$$\begin{cases} U_x = (\cos\psi\sin\theta\cos\varphi + \sin\psi\sin\varphi) & \text{(a)} \\ U_y = (\sin\psi\sin\theta\cos\varphi - \cos\psi\sin\varphi) & \text{(b)} \end{cases}$$

Complete the following manipulation :

$$\sin\psi \times \text{(a)} - \cos\psi \times \text{(b)} \Rightarrow \sin\varphi = U_x \cdot \sin\psi - U_y \cdot \cos\psi$$

and

$$\frac{\text{(a)}}{\cos\varphi\cos\psi} \Rightarrow \sin\theta = \frac{U_x}{\cos\varphi\cos\psi} - \frac{\sin\theta\sin\psi}{\cos\varphi\cos\psi}$$

We get

$$\begin{cases} \sin\varphi = U_x \cdot \sin\psi - U_y \cdot \cos\psi \\ \sin\theta = \dfrac{U_x}{\cos\varphi\cos\psi} - \dfrac{\sin\theta\sin\psi}{\cos\varphi\cos\psi} \end{cases} \quad \textbf{(I.34)}$$

I.11. Rotor Dynamics

The motors usually utilized in quadrotors are brushless DC Motors, which provide little friction and high torque. In the following derivation, it is supposed that the rotors are not geared with rigid mechanical coupling between the motors and the propellers **[12]**. The schematic for a brushless DC motor at stable state as the figure I.17 shows.

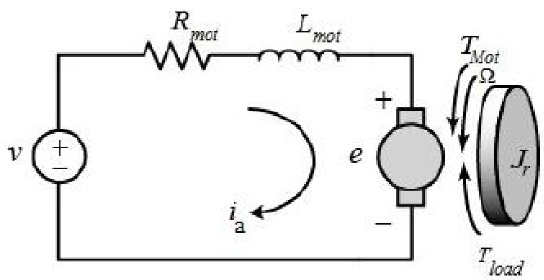

Figure I.16 DC Motor Schematic Diagram

Based on Kirchhoff's voltage law, we get the following equation:

$$v = R_{mot} i_a + L_{mot} \frac{di_a}{dt} + K_{mot} \omega \qquad \textbf{(I.35)}$$

With: v is the input voltage, R_{mot} motor's resistance, L_{mot} motor's inductance and i_a is the armature current, while $K_{mot} \omega$ represents the generated emf, and K_{mot} is the motor's torque constant.

Since the quadrotor depends on small motors, their inductance is low therefore it can be neglected, we get:

$$v = R_{mot} i_a + K_{mot} \omega_i \qquad \textbf{(I.36)}$$

Or

$$i_a = \frac{v - K_{mot} \omega_i}{R_{mot}} \qquad \textbf{(I.37)}$$

In addition, we have the motor's dynamic equation as follows:

$$J_r \dot{\omega}_i = T_{mot} - T_{lood} \qquad \textbf{(I.38)}$$

With: T_{mot} is the torque caused by the motor that is equal to $K_e i_a$, where K_e is the motor's electric constant and for small motors it is approximately equal to K_{mot}
T_{lood} is the load torque which is the torque generated from the propeller system, it equals to $K_M \omega^2$

Substituting the equations of T_{mot} and T_{lood} together with the current equation from the eqaution (I.37), we get:

$$J_r \dot{\omega}_i = K_{mot} \frac{v - K_{mot} \omega_i}{R_{mot}} - K_M \omega^2 \qquad \textbf{(I.39)}$$

After we simplify it, we can write the voltage as a function of the rotor's velocity as follows:

$$v = \frac{R_{mot}}{K_{mot}} J_r \dot{\omega}_i + K_{mot} \omega_i + K_M R_{mot} \omega_i^2 \qquad \textbf{(I.40)}$$

1.12. State Representation

Developing the acquired mathematical model for the quadrotor into a state-space model helps make the control problem easier to tackle

➢ **State Vector X:** Representing the state vector of the quadrotor to be,

$$X = [\, x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7 \;\; x_8 \;\; x_9 \;\; x_{10} \;\; x_{11} \;\; x_{12} \,]^T \tag{I.41}$$

Which is mapped to the degrees of freedom of the quadrotor as follows :

$$X = [\, \varphi \;\; \dot{\varphi} \;\; \theta \;\; \dot{\theta} \;\; \psi \;\; \dot{\psi} \;\; z \;\; \dot{z} \;\; x \;\; \dot{x} \;\; y \;\; \dot{y} \,]^T \tag{I.42}$$

The state vector represents the position of the quadrotor in space and its linear and angular velocities.

➢ **State Space Representation**: For a physical system, there is a multitude of state representations, in our case, we can write the system in state-space representation as the form

$\dot{x} = F(x) + G(x,U)$ **[13]**

By using the equation of motion (I.32), we can write the mathematical model in a state space representation as follows

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = a_1 x_4 x_6 + a_2 x_2^2 + a_3 \bar{\Omega}_r x_4 + b_1 u_2 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = a_4 x_2 x_6 + a_5 x_4^2 + a_6 \bar{\Omega}_r x_2 + b_2 u_3 \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = a_7 x_2 x_4 + a_8 x_6^2 + b_3 u_4 \\ \dot{x}_7 = x_8 \\ \dot{x}_8 = a_9 x_8 + \frac{1}{m} u_x u_1 \\ \dot{x}_9 = x_{10} \\ \dot{x}_{10} = a_{10} x_{10} + \frac{1}{m} u_x u_1 \\ \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = a_{11} x_{12} + \frac{cos\theta cos\varphi}{m} u_1 - g \end{cases} \tag{I.43}$$

Where:

$$\begin{cases} a_1 = \frac{(I_Y - I_Z)}{I_X} \,,\; a_2 = \frac{-K_{fax}}{I_X} \,,\; a_3 = \frac{J_r}{I_X} \,,\; a_4 = \frac{(I_Z - I_X)}{I_Y} \,,\; a_5 = \frac{-K_{fay}}{I_Y} \,,\; a_6 = \frac{J_r}{I_Y} \\ a_7 = \frac{(I_X - I_Y)}{I_Z} \,,\; a_8 = \frac{-K_{fax}}{I_Z} \,,\; a_9 = \frac{-K_{dtx}}{m} \,,\; a_{10} = \frac{-K_{dty}}{m} \,,\; a_{11} = \frac{-K_{dtz}}{m} \\ b_1 = \frac{l}{I_X} \,,\; b_2 = \frac{l}{I_Y} \,,\; b_3 = \frac{l}{I_Z} \end{cases} \tag{I.44}$$

I.13. Conclusion:

Through various translational and rotational movements, we have demonstrated the quadrotor's flying capabilities in this chapter. The applied forces and moments on the quadrotor's structure were then presented, and in order to define the mathematical model of our system, we formulated the equations of motion representing the quadrotor's dynamics, allowing us to describe the system's behavior with sufficient precision. The Newton-Euler method was used to develop the mathematical modeling strategy. We conclude that the quadrotor is an under-actuated system based on the model. Furthermore, the model's complexity, nonlinearity, and interconnections between system states can all be clearly seen.

Chapter II

Implementation and Controlling of quadrotor

II.1. Introduction:

In this chapter we will talk about our implementation and which parts are good for our prototype as well as we will explain why we chose the board Pixhawk as a microcontroller and its powerful benefits, after that we will pass to the control part, we built it using simulink with the PID and we presented the results of it.

II.1. Hardware Components

II.2.1. Frame

The frame is the component responsible for connecting all of the subsystems and ensuring physical integrity. Also the frame is what holds all of the quadrotor's components together. Arms, landing gears, rotor mounts, and central plates add to the complexity. In most cases, central plates serve two purposes. It's where the landing gears and arms connect, as well as the power distribution wiring that powers the remaining components. The JMT 380 kit frame employed within this project has a central plate that power all the ESCs. There are relevant selection choices regarding the frame: the frame's orientation in relation to the position of the flight controller In the case of quadrotors, there are three possible configurations: +, H, and X. And in our thesis we will use the X type.

Figure II.1 quadrotor frame Types

The selected frame for this project was a quadrotor. The orientation of the rotors was **X**.

Figure II.2 380mm Frame

II.2.2. Rotors

The purpose of rotors is to lift the quadcopter and allow it to move around the entire space. To move the vehicle in the desired direction, the flight controller adjusts the angular velocities of the motors. Motors are classified by the amount of power they produce (AC or DC). Alternating current (AC) drives AC motors, while direct current (DC) motors convert electrical energy into mechanical energy**[18]**. The mechanism by which rotors generate rotation is also classified. Rotor classification is depicted in the diagram.

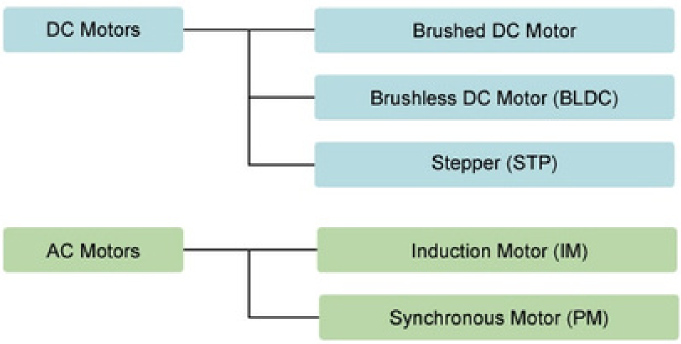

Figure II.3 Rotor classification

In the UAV industry, DC motors are used, specially brushless DC motors (BLDC). An armature, brushes, a field magnet, an axle, and a commutator make up brushed rotors. Their working principle is that the brushes charge the commutator with a polarity that is inverse to that of the permanent magnet, causing the armature to rotate. On the other hand, brushless motors don't have brushes. They are mounted in permanent magnets, normally more than four, around its perimeter. Those kind of rotors make use of control circuit. They are powered by DC electricity through an inverter or switching power supply, that produces AC electric current to control the rotor phases by closed loop controller. The size of the brushless rotors is measured by height and diameter. Rotation speeds are measured by KV rating. .

The KV value is given in rotations per minute (rpm) per volt (V). For instance, the motor employed within this project has a KV of 920. A given voltage of 11 V would mean 11 x 920 = 10120 rpm. The selected rotors were four units of F15843 920 Kv Brushless rotor; two clockwise rotating and the other two counter-clockwise rotating

Figure II.4 F15843 920 Kv Brushless rotor

II.2.3. Propellers

The propellers are the rotors' uppermost parts. Propellers create a downward air flow that provides lift for the quadcopter. 2-blade propellers are commonly used in quadcopters. However, propellers used in the UAV industry can have up to four blades.

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

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

Figure II.5 1045 Propellers

II.2.4. Electronic Speed Controllers (ESC)

Electronic Speed Controllers (ESC) are the bond between the flight controller and the motors. What they do is transforming the electric current from battery (Direct Current) to Alternating Current (AC).

The angular speed of the rotors depends on the Pulse Width Modulated (PWM) sent by the ESC controllers.

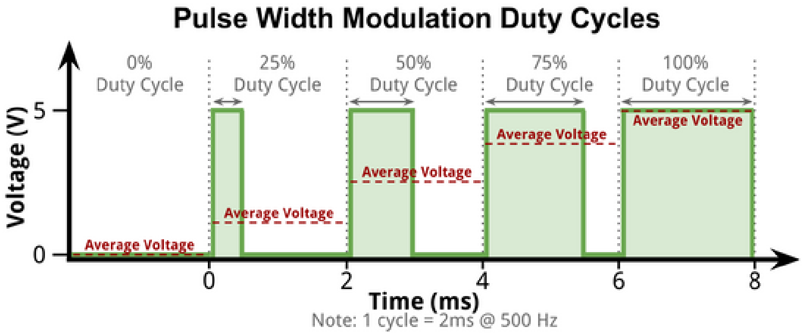

Figure II.6 Duty cycles

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

The selected ESCs for this project were four units of 40A

Figure II.7 Selected Electronic speed control

II.2.5. Power

Nearly all UAV are powered by lithium polymer batteries (LiPo). Those kind of batteries have great power capacity and light weights. Those two features make them perfect to be the source of energy of a device that is going to fly and that need to have low-weight payloads to increase its endurance and range. in our project we will use a Turnigy 2.65Ah with a capacity of 40.

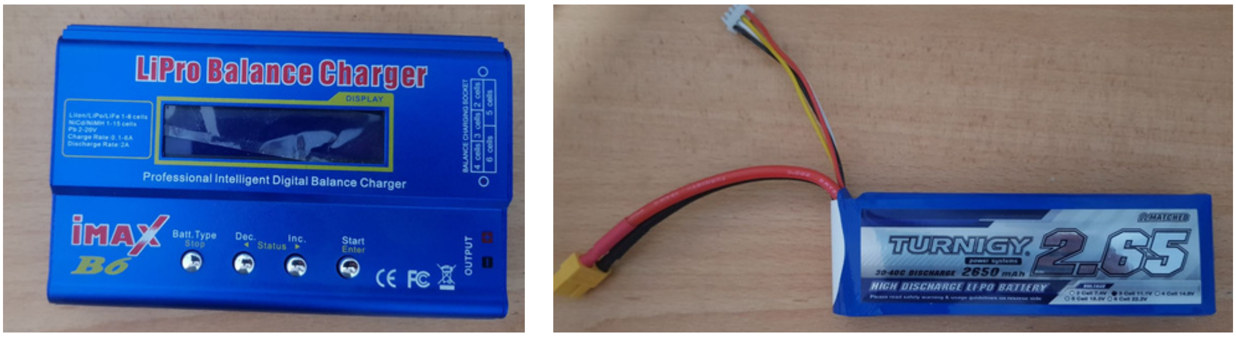

Figure II.8 Left: Lipo battery charger. Right: Lipo battery 2.65Ah

II.2.6. Flight Controller Pixhawk

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

Pixhawk flight controller runs an internal real-time operating system denominated NuttX. The alternative flight controller currently in the market is the one produced and developed by ArduPilot project.

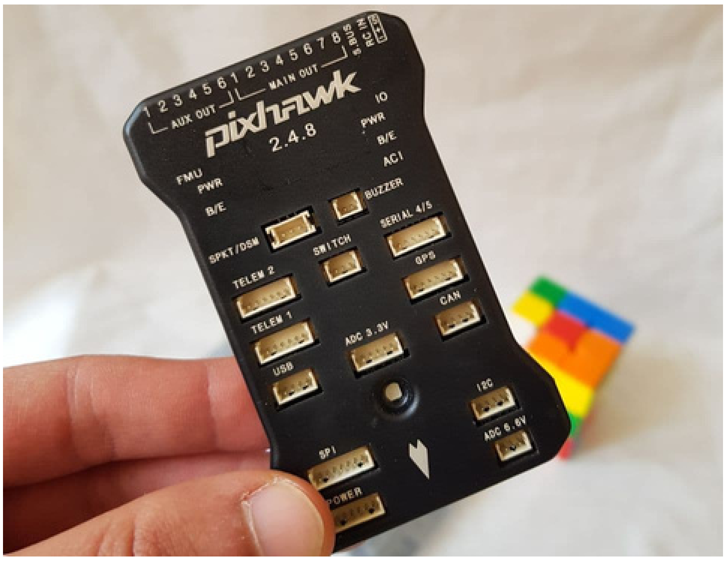

Figure II.9 Pixhawk 2.4.8

II.2.6.1 Why Pixhawk?

Pixhawk is an independent open-hardware project providing readily-available that developed by a group of professional engineers and in the end it comes with low-cost, and high-end, Flexibility in terms of hardware peripherals that can be attached, Widely-used and thus well-tested/stable, autopilot hardware designs to the academic, hobby and industrial communities**[19]**.

Here are Pixhawk's pinouts

Figure II.10 Pixhawk Pinouts explain

And here as we can see, the buzzer can be used to audibly indicate status changes for the vehicle, the safety switch allowing us to handle the vehicle safety by disabling the motors until we activate the switch, GPS module locks to satellites by itself and sends data to pixhawk, Telemetry Radios can be used to provide a wireless connection between a ground control station and the vehicle, and lastly the power module will provide a steady ~5V to Pixhawk and allow the Pixhawk to measure the current and/or voltage of the main battery.

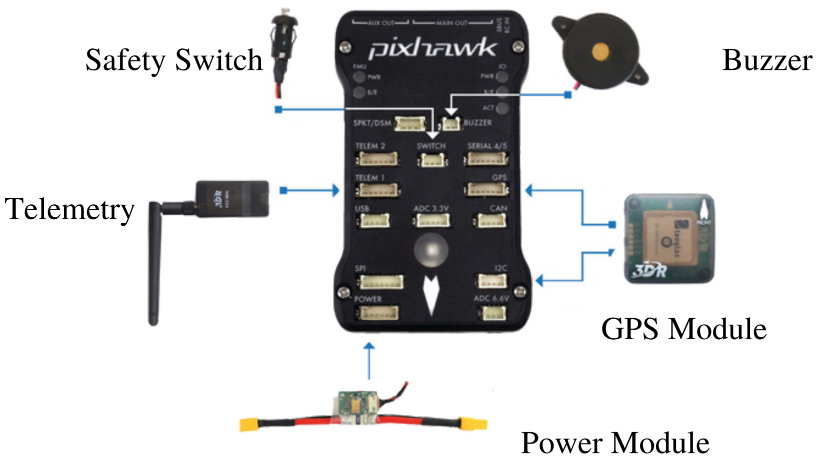

Figure II.11 Pixhawk with the used Modules

it also comes with auxiliary outputs that can control additional servos

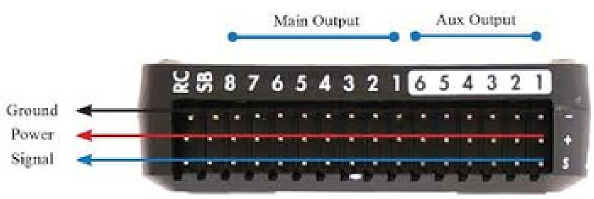

Figure II.12 Pixhawk 2.4.8 pinouts

we can make different modes as RTL (return to launch) when the board lose the signal or getting a low battery level. add to that we can send it in a mission using mission planner software after planning a trajectory and following it on the computer with it's position step by step.

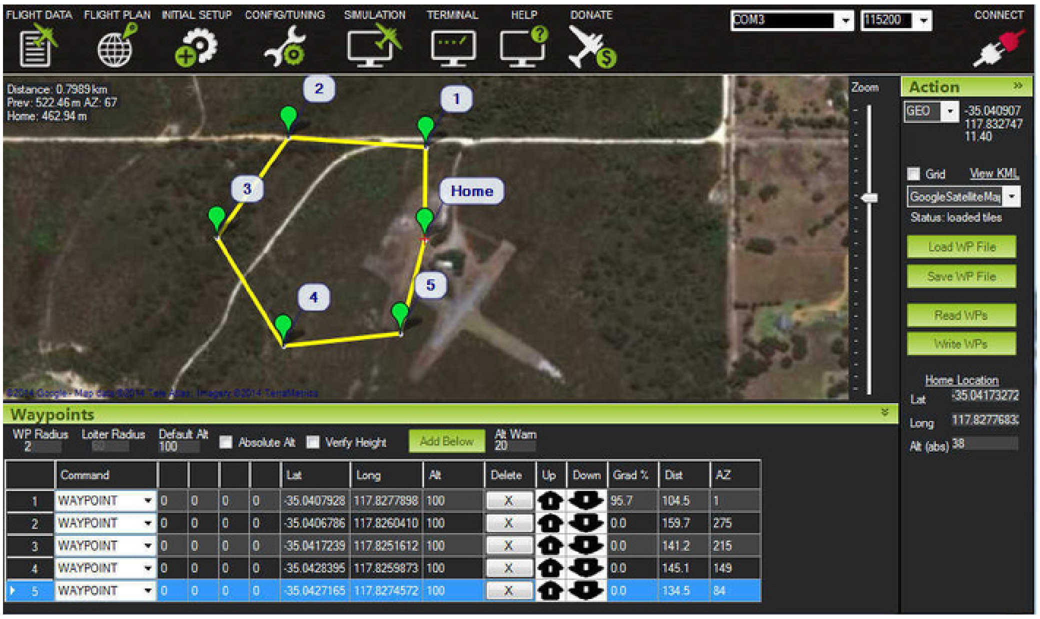

Figure II.13 Example of a mission

II.2.6.2. Why not arduino?

We didn't use arduino because it can't do as much as Pixhawk can do, because we alredy made a test with it and didn't give a good results especially on the response time, it can't also reach the level of using Pixhawk, Pixhawk is easy to use and give much more better results and gave us the hand of editing whatever we need to succeed the project.

II.2.7. Firmware

Internal firmware is required for flight controllers. Firmware is software that is written into the nonvolatile memory of a hardware device. The firmware used in this project is part of the ArduPilot open-source software. Each of Ardupilot's supported vehicles has its own firmware: multicopters, fixed plane UAVs, helicopters, rovers and underwater vehicles. This firmware is already fixed to be installed in the flight controller and start setting the first-time configuration tasks, and it is freely available on the ardupilot official page. It already has its own flight modes for carrying out the required tasks.

II.2.8. Pixhawk sensors

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

- Barometer: a barometer is used to retrieve data of the air pressure to the flight controller. This data is processed and an altitude above ground level is estimated. This sensor is really useful for instance in altitude hold mode, where throttle is adjusted to maintain a desired altitude.
- Magnetometer: a magnetometer is employed to compute the magnetic heading of the flight controller with respect to the magnetic north.

- Inertial sensors: the inertial sensors are mounted in an electronic board called Inertia Measurement Unit (IMU). First inertia sensor is a 3-axis gyroscope which retrieves 3-axis angular accelerations. A 3-axis accelerometer is the second inertia sensor, and it measures linear accelerations. The IMU's purpose is to retrieve the necessary data in order to keep the quadcopter stable during operation. The flight controller analyzes the data and sends the necessary angular velocities to the ESCs in order to keep the drone stable and achieve horizontal balance in the longitudinal and lateral axes.
- External sensors: external sensors can be plugged into the Pixhawk flight controller throw the I2C, SPI and Serial ports. A critical sensor needed for the proper performance of guided flight modes is a GPS sensor, which is plugged into the Pixhawk by the GPS port. It can also be connected an additional compass(magnetometer) through the I2C port. An additional magnetometer helps the system to retrieve more precised headings, based on the comparison of the electromagnetic fields measured in the internal magnetometer of the Pixhawk and the external one.
- The safety switch's purpose is to prevent PWM communication between the flight controller and the ESCs, allowing for safe UAV manipulations even when the battery is connected. The buzzer is a device that emits various sounds depending on the quadcopter's status. It's a way for the flight controller and the pilot to communicate without using the Ground Control station. When the UAV is armed, for example, it emits a distinctive bip for two seconds.

II.2.9. Telemetry

A telemetry connection between the ground control station and the drone is required to monitor the drone's flight attitude and to report any failures or incidents that occur during the flight. Telemetry is also important for some flight modes, particularly autonomous flight modes, in which the quadrotor follows a flight path based on waypoints set in the ground control station.

Figure II.14 Telemetry hardware description

II.2.10. Radio transmitter and receiver

A Radio Control (RC) transmitter and a Radio Control (RC) receiver are used to complete this task. The radio signal's frequency is 2.4 GHz. The receiver is in charge of converting the radio signal received into a Pulse Width Modulated Signal (PWM) (PWM). The Pixhawk flight controller is used in this project, and it has PPM input rather than PWM. PPM stands for pulse position modulation. In PPM, the signal is conformed by a set of pulses of fixed length. Between the pulses, there are pauses of variable length. To input the Pixhawk flight controller, a PPM encoder was used to convert the PWM signal from the RC receiver to PPM. As a result, each of the five used channels' PWM signals (roll, pitch, yaw, throttle, and flight mode) is encoded in a single PPM signal[20]. The remaining channel will be used to control additional servos or mode changing, the Flysky FS-i6X RC transmitter used in this project has ten channels. This channel could be used for servo motor control, which is commonly used when a camera is mounted as a payload on a quadcopter.

Figure II.15 Radio Controller FlySky FS-i6X

II.3. PX4 Software

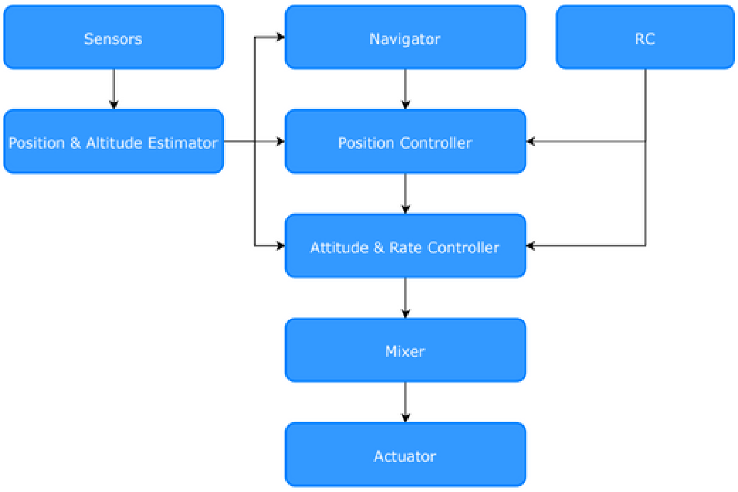

Figure II.16 A schematic about the linked hardware to the quadrotor

PX4 is an open-source community-based autopilot platform founded in 2008 by Lorenz Meier. It was originally known as the "ETH Pixhawk project". One of the most vital parts of the PX4 platform is the autopilot firmware. This software acts as a pilot, monitoring all sensor values and responding to commands. It generates outputs to all actuators to control the unmanned vehicle based on that data. The Autopilot system is made up of two layers: Middleware and Flight Stack.

Sensor and user input fusion, estimations, and flight control are all done by the flight stack. Middleware creates a layer that handles all internal and external communication as well as driver support for supported hardware. All estimation and control algorithms are collected in the flight stack layer.**[21]**

There are controller algorithms for rovers, multi-rotors, helicopters, and VTOL planes, among other vehicles.

II.4. Mission Planner

The ground control station for Plane, Copter, and Rover is called Mission Planner. 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 do for you:

- Load the firmware (the software) into the autopilot board (i.e. Pixhawk series) that controls your vehicle.
- Setup, configure, and tune your vehicle for optimum performance.
- Plan, save and load autonomous missions into you autopilot with simple point-and-click way-point entry on Google or other maps.
- Download and analyze mission logs created by your autopilot.
- Interface with a PC flight simulator to create a full hardware-in-the-loop UAV simulator.
- With appropriate telemetry hardware you can:

1. Monitor your vehicle's status while in operation.
2. Record telemetry logs which contain much more information the the on-board autopilot logs.
3. View and analyze the telemetry logs.
4. Operate your vehicle in FPV (first person view)

Figure II.17 Mission Planner Software

And here is our frame equipped with all the parts that we talked about before:

Figure II.18 The prototype equipped.

II.5. PID Controller

Proportional-Integral-Derivative controller (or PID controller) is a control loop feedback algorithm used widely in industrial control systems. Advantages of PID controller are : simple structure, good performance for several processes and tunable even without a specific model of the controlled system. There are 3 components in a PID controller: Proportional term (P), Integral term (I) and Derivative term (D). P term depends on the present error, I term is affected by the accumulation of the past errors and D term is a prediction of the future error based on the current rate of change. Each of these terms has its own gain to determine how much it affects the controlled system. The structure of PID controller can be expressed with reference signal r(t), system output y(t), error e(t), and the controller output u(t) as follows**[17]**:

$$u(t)=K_p \cdot e(t) + K_i \cdot \int_0^t e(t)dt + K \cdot \frac{de(t)}{dt} \qquad \textbf{(I.2)}$$

$$e(t) = r(t) - y(t)$$

Closed-loop response	Rise time	Overshoot	Settling time	Steady-state error
Kp	Decrease	Increase	Small change	Decrease
Ki	Decrease	Increase	Increase	Eliminate
Kd	Small change	Decrease	Decrease	No effect

Table II.1 : Effects of increasing PID gains to closed-loop response

Table II.1 shows how each of the PID gains affects the closed-loop response.

II.6. Attitude Control of Quadrotor

The model developed in chapter 1 describes the differential equations of the system. It is advisable for control design to simplify the model in order to comply with the real-time constraints of the embedded control loop. Hence, hub forces and rolling moments are neglected, and thrust and drag coefficients are supposed constant.

$$\begin{cases} \ddot{\varphi} = \frac{d}{J_x} \cdot U_2 \\ \ddot{\theta} = \frac{d}{J_y} \cdot U_3 \\ \ddot{\psi} = \frac{d}{J_z} \cdot U_4 \\ \ddot{Z} = -g + \frac{U_1}{m} \end{cases} \qquad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = b_1 u_2 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = b_2 u_3 \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = b_3 u_4 \\ \dot{x}_7 = x_8 \\ \dot{x}_8 = a_9 x_8 + \frac{1}{m} u_x u_1 \end{cases} \qquad \textbf{(II.2)}$$

For each input, a PID controller is built and based on equation (II.2). Thus, there are four sub PID controllers for the entire system. They are:

$$\begin{aligned} U_1 &= K_{p2}(z_d - z) + K_{d2}(\dot{z}_d - \dot{z}) + K_{i2}\int (z_d - z(t))\,dt \\ U_2 &= K_{p2}(\varphi_d - \varphi) + K_{d2}(\dot{\varphi}_d - \dot{\varphi}) + K_{i2}\int (\varphi_d - \varphi(t))\,dt \\ U_3 &= K_{p2}(\theta_d - \theta) + K_{d2}(\dot{\theta}_d - \dot{\theta}) + K_{i2}\int (\theta_d - \theta(t))\,dt \\ U_4 &= K_{p2}(\psi_d - \psi) + K_{d2}(\dot{\psi}_d - \dot{\psi}) + K_{i2}\int (\psi_d - \psi(t))\,dt \end{aligned} \qquad \textbf{(II.3)}$$

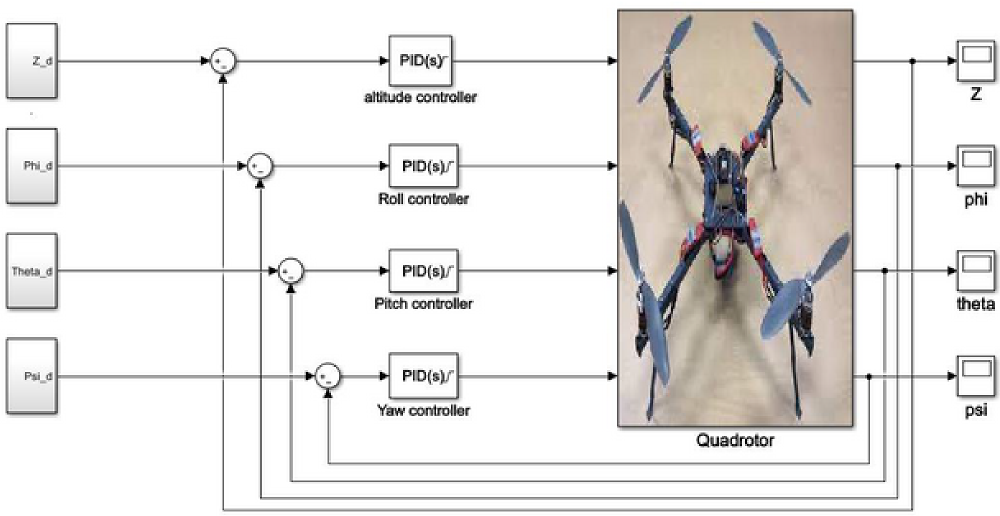

Figure II.19 Simulink model used for quadrotor simulations

The quadrotor attitude dynamics allows the quadrotor to be directly controlled by a PID designed for angle stabilization.

The mechanical and electrical parametrs of quadrotor used in the simulation,are given in the following table

Parameters	Designation	Value
K_p	lift coefficient	2.9842×10^{-5}N. rad^2/s^2
K_d	drag coefficient	3.2320×10^{-7}N. rad^2/s^2
m	weight of the quadrotor	785g
d	Distance between the center of mass of the system and the axis of rotation of the motors	25cm
J	Quadrotor Inertia Matrix	diag(3.8278;3.828;7.656)× 10^{-3}N.m/ rad /s^2
K_{fa}	Aerodynamic friction coefficients	diag(5.5670;5.5670;6.3540) $\times10^{-4}$N / rad /s
K_{ft}	Coefficients of drag forces according to (X ,Y, Z)	diag(5.5670;5.5670;6.3540) $\times 10^{-4}$N / rad / s
Jr	Rotor inertia	2.8385×10 N.m/ rad / s^2
K_e	Electrical torque constant	0.0216
Cr	The friction	5.3826×10^{-3}
Kr	Load torque constant	3.4629×10^{-7}
β_0 , β_1 , β_2 , b	Motor Parameters	189.63, 6.0612, 0.0122, 280.19

Table II.2 Mechanical and electrical parameters of quadrotor

II.7. Simulation Result

The following PID gains is used in the simulation

Controller	K_p	K_i	K_d
Altitude	1.6	0.5	0.002
Roll	2.3	0.6	0.005
Pitch	1.9	0.7	0.006
Yaw	1.2	0.5	0.002

Table II.3 PID control parameters

The simulation results are shown from **Figure II.20** to **Figure II.25**

- **Stabilization of attitude**

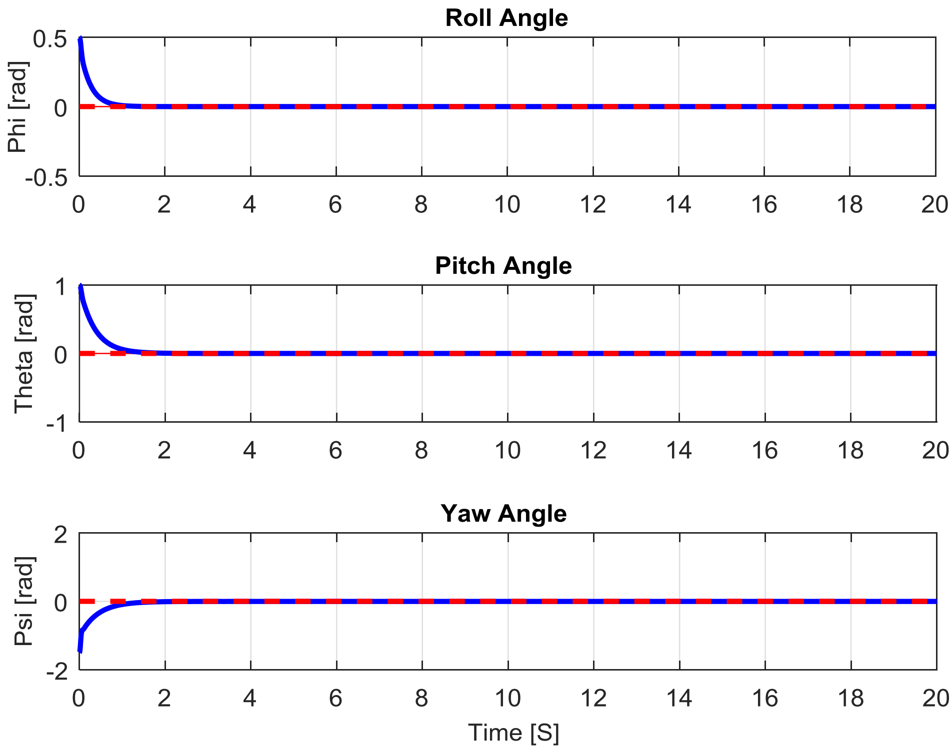

Figure II.20 Roll, Pitch and Yaw Angles stabilisation reponse (PID controller)

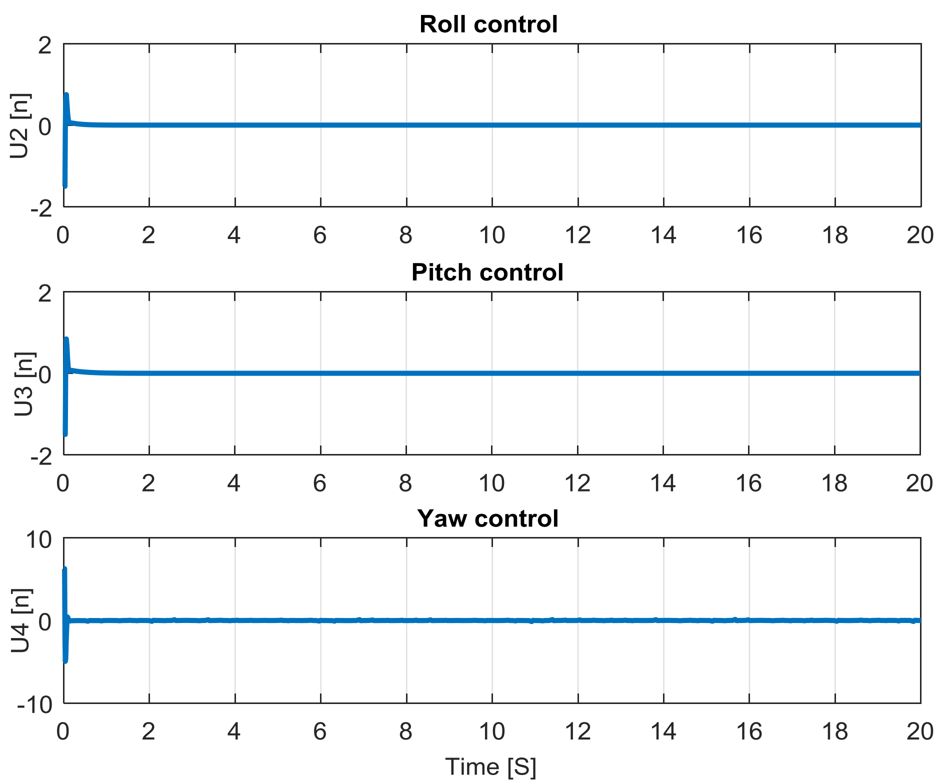

Figure II.21 Roll, Pitch and Yaw Control Response using PID control

- **Varying Trajectory**

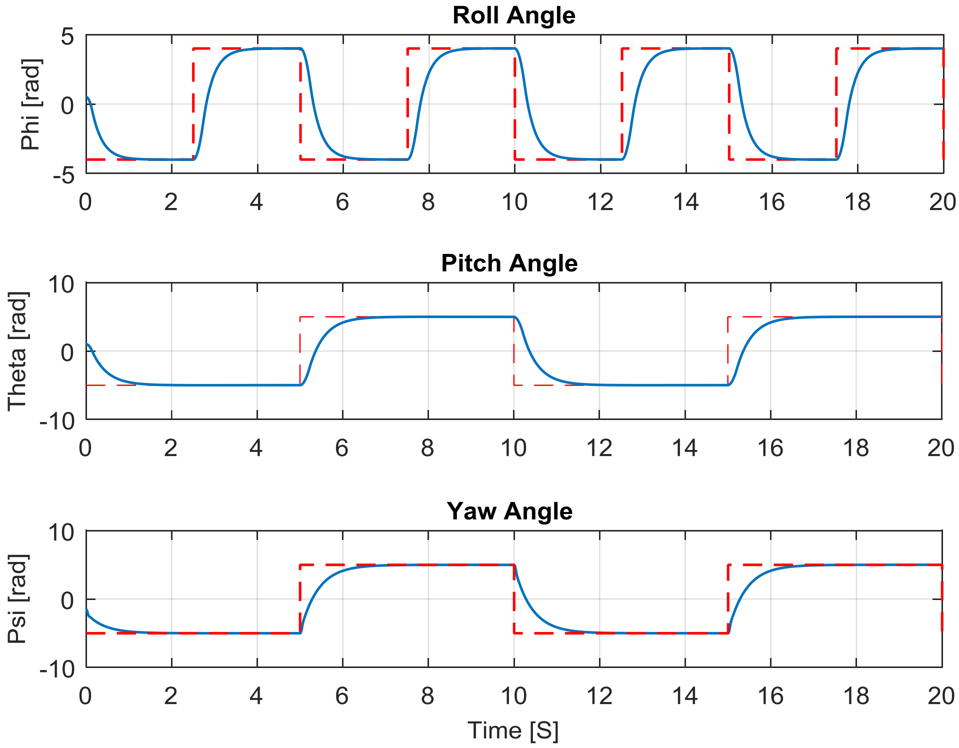

Figure II.22 Tracking responses results of the desired trajectories using PID control.

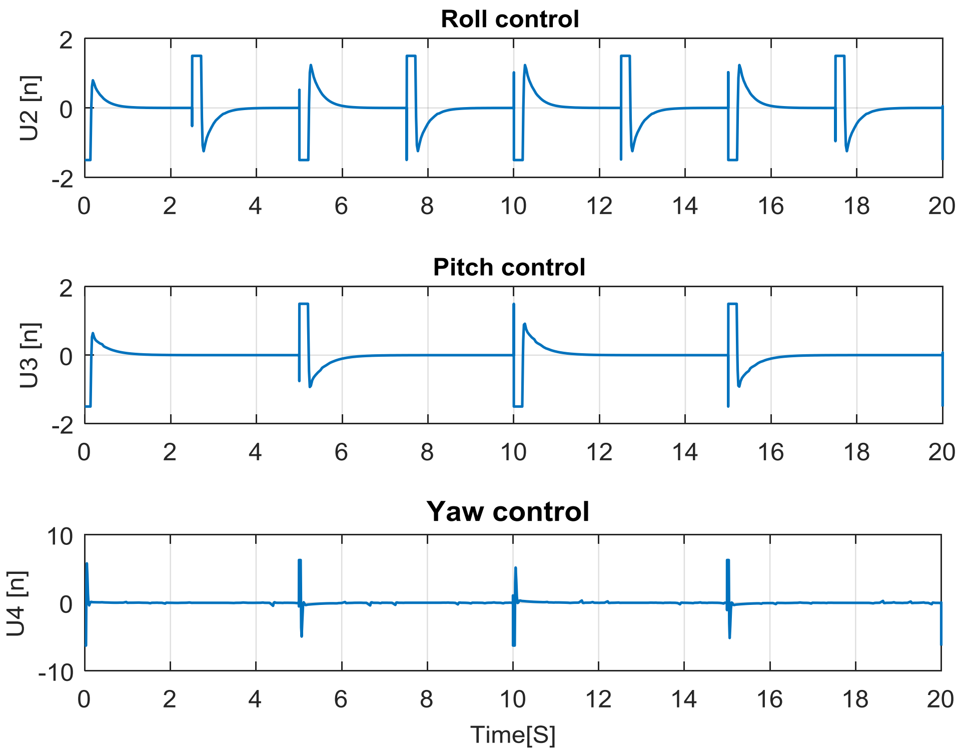

Figure II.23 Control Input Response of the desired trajectories using pid control

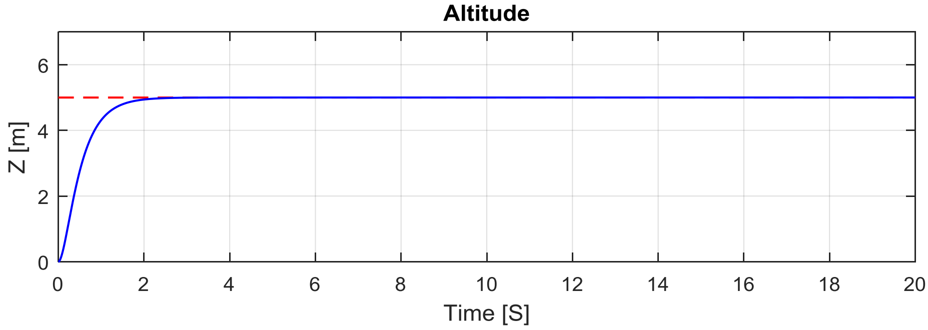

Figure II.24 Altitude reponse using PID control

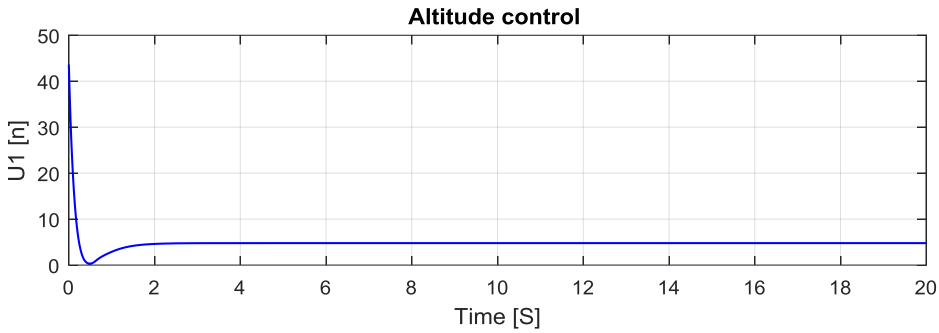

Figure II.25 Control Input Response using PID control

II.8. Conclusion:

In this chapter, we have presented our equipment which we will use from the frame to the remote control as well as the software of the used microcontroller Pixhawk. Adding to that we showed how the PID system works on the quadrotor and how the change happen with the three angles and the axe z, we also showed our output using Matlab.

Part II

Deep Learning For Object Detection

Chapter III

Deep Learning For Object Detection

III.1 Introduction

Object detection has become ingrained in our daily lives, with applications ranging from security to automated vehicle systems. So in this chapter we will see the evolution of the artificial intelligence and how it became important in all technology fields, as well as deep learning and the ways and algorithms of detecting an object from the camera to the recognition of an object and making a bounding box to it.

III.2 Artificial Intelligence:

III.2.1 AI history:

Artificial intelligence (AI) has a long history, beginning with myths, stories, and rumors about master craftsmen endowing artificial beings with intelligence or consciousness. Philosophers who attempted to describe the process of human thinking as the mechanical manipulation of symbols sowed the seeds of modern AI. The programmable digital computer, a machine based on the abstract essence of mathematical reasoning, was invented in the 1940s as a result of this work. This device and its concepts prompted a group of scientists to seriously consider the possibility of creating an electronic brain.

During the summer of 1956, a workshop on the campus of Dartmouth College in the United States established the field of AI research. For decades, those who attended would be at the forefront of AI research. Many of them predicted that within a generation, a machine as intelligent as a human would exist, and they were given millions of dollars to make this vision a reality.

Eventually, it became clear that the project's difficulty had been grossly underestimated by commercial developers and researchers. The US and British governments stopped funding undirected artificial intelligence research in 1974, in response to criticism from James Lighthill and ongoing congressional pressure, and the difficult years that followed became known as the "AI winter." Seven years later, a visionary initiative by the Japanese government inspired governments and industry to invest billions of dollars in AI, but investors became disillusioned by the late 1980s and withdrew funding once more.

Machine learning was successfully applied to many problems in academia and industry in the first decades of the twenty-first century, thanks to new methods, the use of powerful computer hardware, and the collection of massive data sets.**[21]**

III.2.2 What is machine learning:

Machine learning (ML) is the study of computer algorithms that can learn and develop on their own with experience and data. It is considered to be a component of artificial intelligence. Machine learning algorithms create a model based on training data to make predictions or judgments without having to be explicitly programmed to do so.

Machine learning algorithms are utilized in a wide range of applications, including medicine, email filtering, speech recognition, and computer vision, where developing traditional algorithms to do the required tasks is difficult or impossible. However, not all machine learning is statistical learning. A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers. The field of machine learning benefits from the study of mathematical optimization because it provides methods, theory, and application domains. Data mining is a related field of study that focuses on unsupervised learning for exploratory data analysis. Data and neural networks are used in some machine learning implementations to mimic the functioning of a biological brain. Machine learning is also known as predictive analytics when it is used to solve business problems.**[22]**

III.2.3 Why we use machine learning:

Machine learning technology is necessarily new; machine learning algorithms have existed for years, but machine learning procedures have lately risen to prominence as a result of a number of significant technological advancements, including:

- Access to vast amounts and types of data, especially the creation and widespread use of "big data."
- Much more cost-effective data storage options helped made massive data sets more accessible to more businesses and for a wider range of applications.
- Increasing processing power that allows computers and specifically AI applications to complete calculations much faster than ever before.

These advancements have paved the way for machine learning to deliver far better results than it has in the past, allowing machine learning applications to benefit almost every industry and business activity.

III.2.4 Applications of Machine learning:

- **Healthcare**: ML is used to analyze massive healthcare data sets to accelerate discovery of treatments and cures, improve patient outcomes, and automate routine processes to prevent human error. For example, IBM’s Watson uses data mining to provide physicians data they can use to personalize patient treatment.**[23]**
- **Data security**: Machine learning algorithms can detect data security flaws before they become breaches. Machine learning algorithms can predict future high-risk activities based on prior experiences, allowing risk to be managed proactively.
- **Military**: Also the military uses ML for their autonomous drones to target recognition, survey or perform perilous missions while the men would take care of the less dangerous support and support missions.

- **Fraud detection**: ML is being used in the financial and banking sector to autonomously analyze large numbers of transactions to uncover fraudulent activity in real time. Technology services firm Capgemini claims that fraud detection systems using machine learning and analytics minimize fraud investigation time by 70% and improve detection accuracy by 90%.

III.3 Deep Learning:

III.3.1 Neural Network

A neural network is a set of algorithms that attempts to recognize underlying relationships in a batch of data using a method that resembles how the human brain works. Neural networks, in this context, refer to systems of neurons that can be organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems.**[24]**

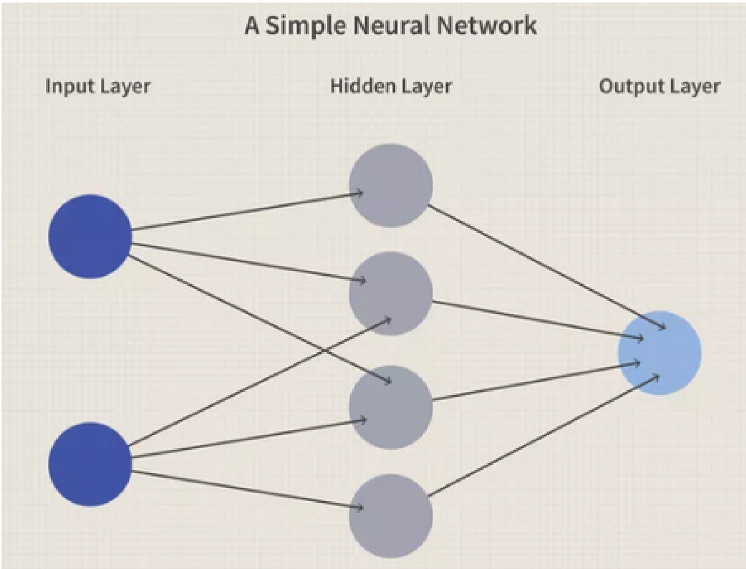

Figure III.1 Example of a simple neural network

III.3.2. Deep Neural Network

Nodes are little components of the system that function similarly to neurons in the human brain. When they are stimulated, a reaction occurs in these nodes. Some are connected and marked, while others are not, although nodes are generally grouped into layers.

To complete a task, the system must process layers of data between the input and output. The deeper the network is evaluated, the more layers it must process to obtain the outcome. Credit Assignment Path (CAP) is a notion that refers to the number of such levels required for the system to finish the assignment. If the CAP index is greater than two, the neural network is considered deep.

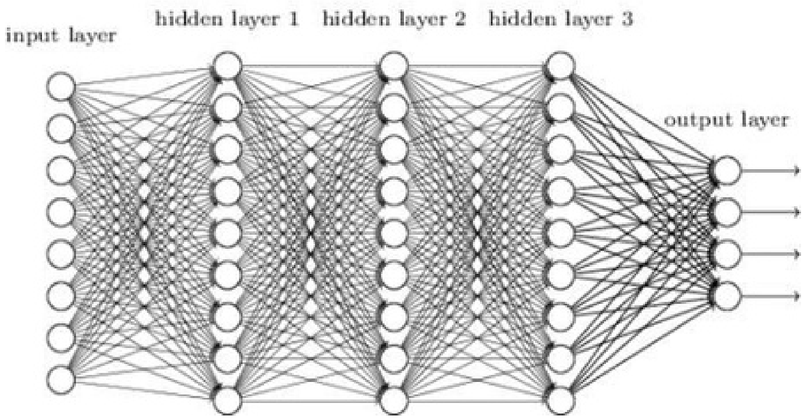

Figure III.2 Example of a deep neural network

When you need to replace human labor with autonomous work without sacrificing efficiency, a deep neural network can help. In real life, deep neural networks can be used in a variety of ways. For example, SenseTime, a Chinese startup, developed an automatic face recognition system to identify criminals that employs real-time cameras to locate an offender in a crowd. It is now standard procedure with the police and other government agencies. The famous company UbiTech creates AI robots. One of their creations is the Alpha 2 robot that can live in a family, speak with its members, search for information, write messages, and execute voice commands.**[25]**

III.3.3. Recurrent Neural Networks

RNN is a type of neural network that uses hidden units to analyze data streams. The output of several programs, such as text processing, speech recognition, and DNA sequencing, is dependent on earlier calculations. RNNs are highly suited for the health informatics area, where huge amounts of sequential data are accessible to process, because they deal with sequential data. In general, RNNs are provided with the input samples which contain more interdependencies. Also they have a significant representation for keeping the information about the past time steps. The output produced at time T1 affects the parameter available at time T1 + 1. In this manner, RNNs keep two kinds of input such as the present one and the past recent one to produce the output for the new data. As deep autoencoders. **[26]**

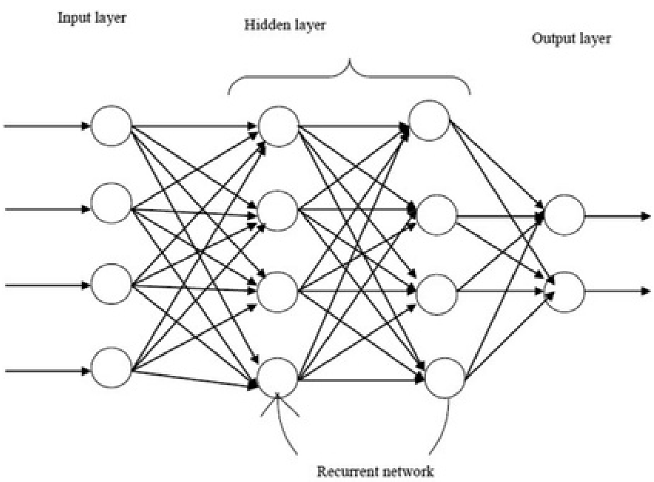

Figure III.3 Example of a Recurrent Neural Networks

III.3.4 Convolution Neural Networks

The discovery that a convolutional neural network (CNN) could be used to extract higher- and higher-level representations of image content was a breakthrough in building models for image classification. Rather than preprocessing the data to extract features such as textures and shapes, a CNN uses the image's raw pixel data as input and "learns" how to extract these features and, in turn, infer what object they represent.

To begin, the CNN is given an input feature map, which is a three-dimensional matrix whose first two dimensions correspond to the image length and width in pixels. The third dimension is 3 in length (corresponding to the 3 channels of a color image: red, green, and blue). The CNN is made up of a series of modules, each of which performs three tasks.**[27]**

1. Convolution

Convolution takes tiles from an input feature map and applies filters to them to create new features, resulting in a convolved feature map (which may have a different size and depth than the input feature map). There are two parameters that define convolutions:

- Size of the tiles that are extracted (typically 3x3 or 5x5 pixels).
- The depth of the output feature map, which corresponds to the number of filters that are applied.

During a convolution, the filters (matrices the same size as the tile size) effectively slide over the input feature map's grid horizontally and vertically, one pixel at a time, extracting each corresponding tile

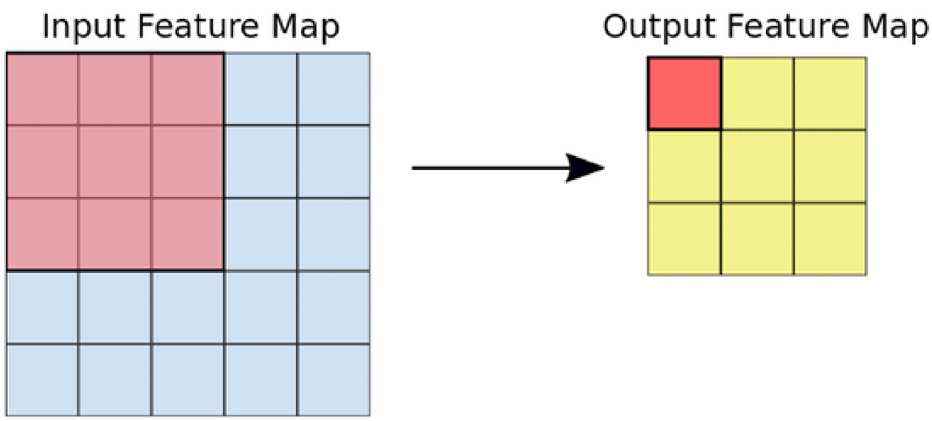

Figure III.4 How the convolution works.

FIGURE - A 3x3 convolution of depth 1 performed over a 5x5 input feature map, also of depth 1. There are nine possible 3x3 locations to extract tiles from the 5x5 feature map, so this convolution produces a 3x3 output feature map.

The CNN performs element-wise multiplication of the filter and tile matrices for each filter-tile pair, then sums all the elements of the resulting matrix to get a single value. The convolved feature matrix then outputs each of these resulting values for each filter-tile pair.

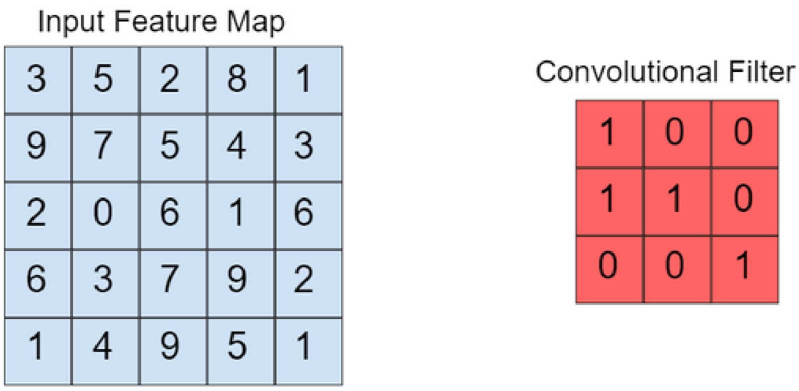

Figure III.5 Left: A 5x5 input feature map. Right: a 3x3 convolution

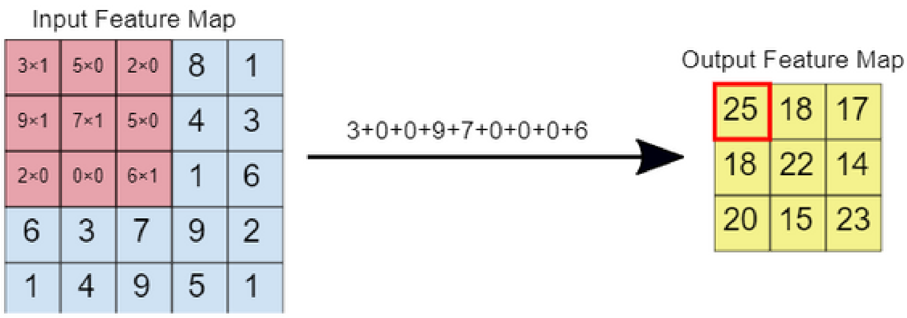

Figure III.6 Left: input feature map. Right: Convolved result

The CNN "learns" the best values for the filter matrices that allow it to extract meaningful features (textures, edges, and shapes) from the input feature map during training. The number of features the CNN can extract grows as the number of filters (output feature map depth) applied to the input grows. However, because filters account for the vast majority of the CNN's resources, training time increases as more filters are added. Furthermore, because each new filter added to the network adds less incremental value than the previous one, engineers strive to build networks with the fewest possible filters to extract the features required for accurate image classification.

2. ReLU

Following each convolution operation, the CNN applies a Rectified Linear Unit (ReLU) transformation to the convolved feature, in order to introduce nonlinearity into the model.

3. Pooling

That after ReLU, the CNN downsamples the convolved feature (to reduce processing time), reducing the number of dimensions of the feature map while preserving the most critical feature information. Max pooling is a common algorithm used in this process.

Max pooling works in the same way that convolution does. We move our mouse over the feature map and extract tiles of a certain size. The maximum value is output to a new feature map for each tile, while all other values are discarded. Two parameters are required for max pooling operations:

- **Size** of the max-pooling filter (typically 2x2 pixels)
- **Stride**: the distance, in pixels, separating each extracted tile. Unlike with convolution, where filters slide over the feature map pixel by pixel, in max pooling, the stride determines the locations where each tile is extracted. For a 2x2 filter, a stride of 2 specifies that the max pooling operation will extract all nonoverlapping 2x2 tiles from the feature map

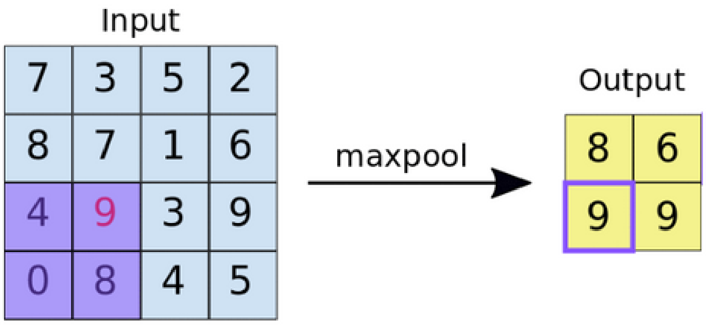

Figure III.7 Left: Max pooling Right: the output max pooling operation

III.3.5. Fully Connected Layers

One or more fully connected layers are found at the end of a convolutional neural network (when two layers are "fully connected," every node in the first layer is connected to every node in the second layer). Their task is to classify data using the features extracted by the convolutions. A softmax activation function, which outputs a probability value from 0 to 1 for each of the classification labels the model is attempting to predict, is typically found in the final fully connected layer.

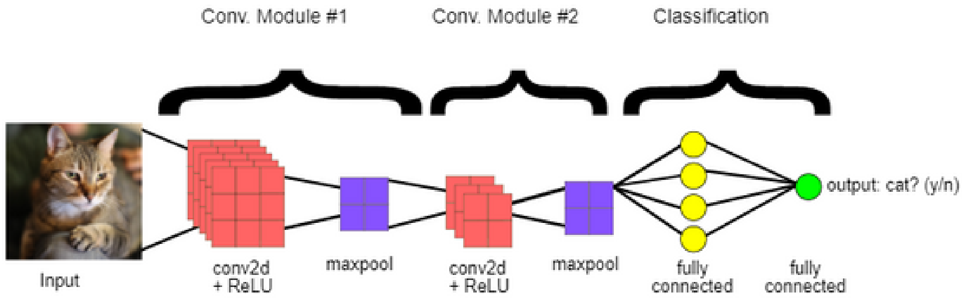

Figure III.8 illustrates the end-to-end structure of a convolutional neural network.

For feature extraction, the CNN uses two convolution modules (convolution + ReLU + pooling) and two fully connected layers for classification. Other CNNs may have more or fewer convolutional modules, as well as more or fewer fully connected layers. Engineers frequently conduct experiments to determine the best configuration for their model.**[28]**

III.4. Object Detection:

Object detection is a computer vision technique for identifying and locating objects in images and videos. Object detection can be used to count objects in a scene, determine and track their precise locations, and accurately label them using this type of identification and localization.
In this chapter we will talk about what object detection is, and the different modes as well as the methods we can find and which method we used and why.**[29]**

III.4.1. What is object detection?

Object detection is a computer vision technique for identifying and locating objects in images and videos. Object detection, in particular, creates bounding boxes around detected objects, allowing us to see where they are in (and how they move through) a scene.Because object detection and image recognition are often confused, it's important to understand the differences before moving on.**[29]**
Image recognition assigns a label to an image. A picture of a dog receives the label “dog”. A picture of two dogs, still receives the label “dog”. Object detection, on the other hand, draws a box around each dog and labels the box “dog”. The model predicts where each object is and what label should be applied. In that way, object detection provides more information about an image than recognition.

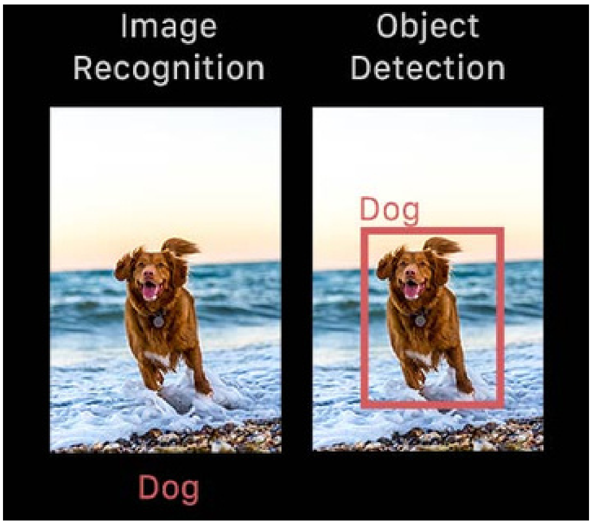

Figure III.9. an example of a detected object

An image is labeled using image recognition. The label "dog" is applied to a photograph of a dog. The label "dog" is still applied to a photograph of two dogs. In contrast, object detection draws a box around each dog and labels it "dog." The model forecasts the location of each object and the label that should be applied. Object detection, in this sense, provides more information about an image than recognition.

III.4.2. Modes and types of object detection

Object detection can be divided into two categories: machine learning-based approaches and deep learning-based approaches. In more traditional ML-based approaches, computer vision techniques are used to look at various features of an image, such as the color histogram or edges, to identify groups of pixels that may belong to an object. These features are then fed into a regression model that predicts the location of the object along with its label. Deep learning-based approaches, on the other hand, use convolutional neural networks (CNNs) to perform end-to-end, unsupervised object detection, which eliminates the need to define and extract features separately.

III.4.3. Why is object detection important?

Object detection is inextricably linked to other similar computer vision techniques like image recognition and image segmentation in that it aids in the comprehension and analysis of scenes in images or video. However, there are significant differences. Image segmentation creates a pixel-level understanding of a scene's elements, while image recognition only produces a class label for an identified object. Object detection is distinguished from these other tasks by its ability to locate objects within an image or video. As a result, we can count and track those objects.**[30]**

Given these key distinctions and object detection's unique capabilities, we can see how it can be applied in a number of ways:

- Crowd counting
- Self-driving cars
- Video surveillance
- Face detection
- Anomaly detection

Of course, this isn't an exhaustive list, but it includes some of the primary ways in which object detection is shaping our future.

III.4.4. Challenges

There are a lot of issues and challenges related to object recognition.

III.4.4.1 Variable Number of Objects

We need to represent data into fixedsized vectors when training a machine learning modal We can't tell the correct number of outputs if we don't know the number of objects in an image. It will be difficult to determine the vector size. Although postprocessing is required to solve this problem, it increases the model's complexity.

III.4.4.2 Sizing

The objects in a photograph are not all the same size. Object recognition models face a significant challenge due to object size differences. When doing classification, we usually want to identify the object that occupies the majority of the image. However, some objects may only cover a small portion of an image. Variable size sliding windows could be used to solve this problem, but this is a very inefficient solution.

III.4.4.3 Modelling

Another challenge is solving two problems at the same time. It's difficult to combine classification and localization into a single model.

III.4.4.4. Illumination

The same object may appear differently in different images depending on the lighting conditions. Regardless of the lighting, the system must be able to recognize the object.

III.4.4.5. Occlusion

Objects on the image are sometimes partially visible. Some objects are partially hidden behind another. The model must be capable of dealing with these situations. Object recognition problems include noise, blurry images, deformation, interclass variation, background clutter, and so on.

III.4.5. preprocessing dataset (image labeling)

the dataset does not have bounding boxes around the objects. We need to add them ourselves. This is often one of the hardest and most costly parts of a Machine Learning project: getting the labels. It is a good idea to spend time looking for the right tools. An image labeling or annotation tool is used to label the images for bounding box object detection and segmentation. Open-source image labeling tool like :(VGG Image, Annatator, labelimg, openlabeler, imglab)

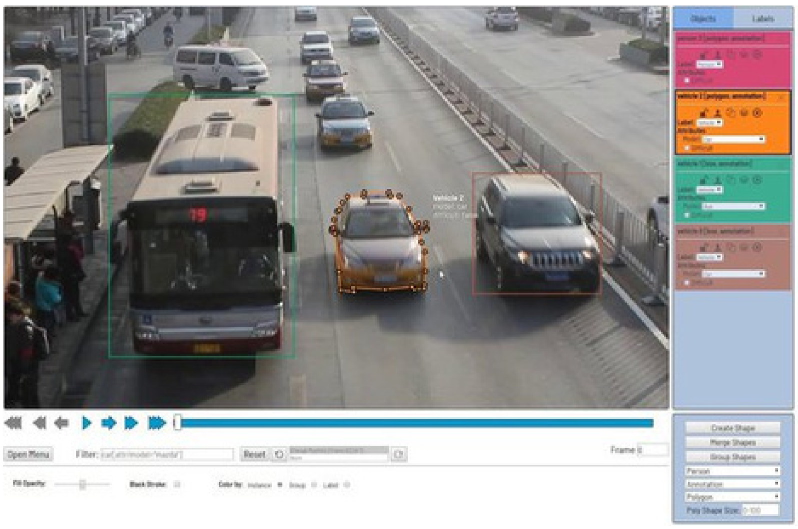

Figure III.10 example about image labelling

III.4.6. The Bounding boxes

The bounding boxes should be normalized so that the horizontal and vertical coordinates, as well as the height and width, all range from 0 to 1. It is common to predict the square root of the height and width rather than the height and width directly: this way, a 10-pixel error for a large bounding box will not be penalized as much as a 10-pixel error for a small bounding box.

III.4.7. Object Detection Methods

III.4.7.1. Object detection model evaluation

The MSE often works fairly well as a cost function to train the model, but it is not a great metric to evaluate how well the model can predict bounding boxes. The most common metric for this is the Intersection over Union (IoU).

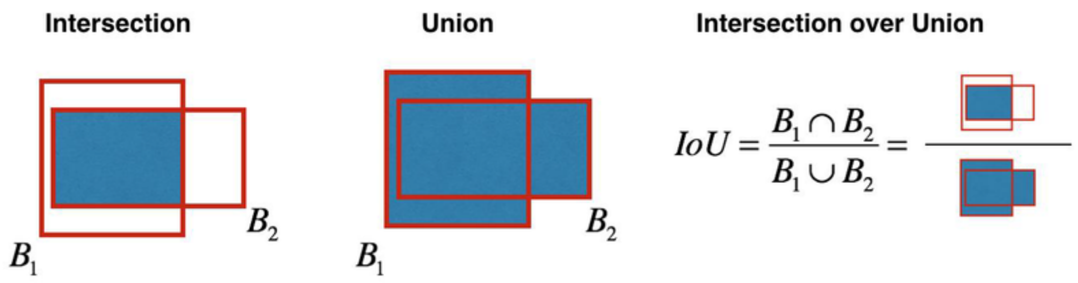

Figure III.11 Intersection over Union of the model

III.4.7.2. mean Average Precision (mAP)

In order to calculate mAP, we draw a series of precision-recall curves with the IoU threshold set at varying levels of difficulty. In COCO evaluation, the IoU threshold ranges from 0.5 to 0.95 with a step size of 0.05 represented as AP@[.5:.05:.95]

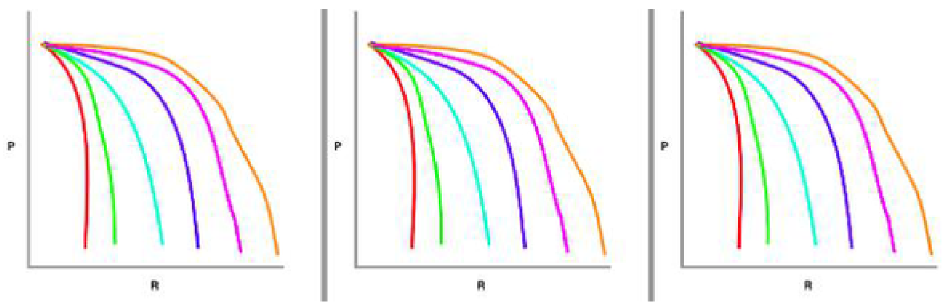

Figure III.12 Precision recall for the dataset

We draw these precision-recall curves for the dataset split out by class type (for example 3 classes and 6 threshold ranges).

III.4.7.3. Classical Methods

Many different types of methods for solving object recognition problems have been proposed over the years. But two methods stand out above the rest. Paul Viola made the first appearance in 2001. "Robust Real-Time Object Detection" was the title of his paper. This method is quick and straightforward. It is used to detect faces in real time in point-and-shoot cameras. Using the Haar feature, this method generates different binary classifiers.

These binary classifiers are evaluated using a multi-scale sliding window in cascade and are dropped early if the classification is negative. Navneet Dalal and Bill Triggs proposed the Histogram of Oriented Gradient or HOG feature as the second method. For classification, they use the Support Vector Machine (SVM). Similar to Paul's method, it uses a multiscale sliding window. It outperforms the first method in terms of accuracy, but it is much slower than Paul's method.

III.4.7.4. Deep Learning Methods

Deep learning has revolutionized machine learning, particularly in the field of computer vision. Deep learning models have outperformed other traditional object recognition methods. With the development of the convolutional neural network in 2012, modern history or object recognition began. AlexNet took first place in the ILSVRC 2012 by a large margin. AlexNet was built on the decades-old LeNet, with data augmentation, ReLU, dropout, and GPU implementation. It demonstrated the power of a convolutional neural network and ushered in a new era in computer vision.**[31]**

OverFeat: In 2013, P Sermanet of NYU proposed a multiscale sliding window algorithm for extracting features from an input image using AlexNet.

R-CNN: A region-based convolutional neural network (R-CNN) is a natural combination of a heuristic region proposal method and a CNN feature extractor. A region proposal method, such as selective search, is used to extract possible objects from an image. After that, the regions are cropped and warped to a predetermined size. Each region's features are extracted using CNN. Then, to classify each region, a support vector machine model is trained. Although training an R-CNN is difficult, it can yield excellent results.**[30]**

R-CNN Fast: R-CNN Fast is similar to R-CNN. It uses selective search to extract possible objects, just like R-CNN. However, there is a distinction in the feature extraction step. Instead of using SVM in individual regions, CNN is used to extract features across the entire image. And, on the feature map, uses Region of Interest (ROI) pooling with a final feed forward network for classification and regression. The system's biggest flaw is that it still relies on selective search for region proposals.**[30]**

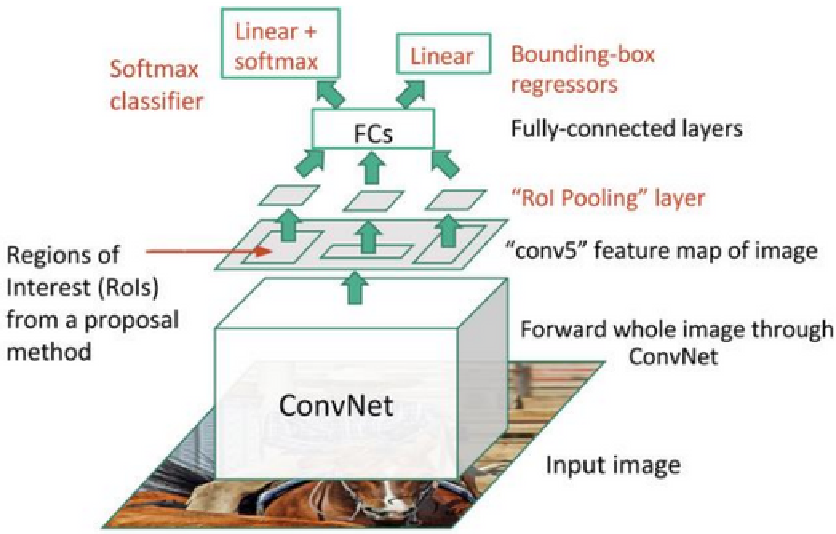

Figure III.13 Fast R-CNN.

YOLO: In 2015, Joseph Redmon published a paper titled You Only Look Once: Unified, Real-Time Object Detection (YOLO). YOLO is a multibox development that is a CNN-based region proposal solution. It converts multibox from a region proposal system to an object recognition system by adding a softmax layer in front of the box regressor and box classifier layers to predict the object class directly. It produces excellent results while also being quick.

Faster R-CNN: Faster R-CNN is a Fast R-CNN in which selective search is replaced by Region Proposal Network (RPN) for region proposal. Multibox has also influenced RPN. This makes the modal trainable from beginning to end.

SSD: The RPN of Faster R-CNN is used in the single shot detector. RPN is used to give object confidence score in Faster RCNN, but it is used directly to classify objects inside the prior box here.

Mask R-CNN: is a segmentation variant of Faster R-CNN. A branch has been added here to predict class-specific object masks. Because RoIPooling was not designed for pixel to pixel alignment, Mask RCNN replaced it with RoIAlign.

III.4.7.5. Multimodal Methods

The results of deep learning methods for object recognition are impressive. However, a large amount of data is required to train a deep learning model. Labeled data is required for classification problems. However, labeling an image takes time. Images from the internet are sometimes mislabeled or incorrectly labeled. The solution to this problem is multimodal machine learning. The type of information or data representation format in which information is stored is referred to as modality. The way we saw the world was multimodal. Things are visible, sounds are heard, odors are smelled, and texture is felt. This is how we derive knowledge from the environment. The same principle applies to multimodal machine learning. Different modalities of information complement each other. Image classification and captioning models, for example, rely on labelled input data. However, labels may be inaccurate or unavailable. In such cases, the model could be trained using the descriptions and tags that came with the image.

Multimodal learning faces some significant technical challenges:

- Representation,
- Alignment,
- Translation,
- Fusion,
- Co-learning

III.4.8.Multiple Objects Detection

In general, object detectors have three (3) main components:

1) The backbone that extracts features from the given image.
2) The feature network that takes multiple levels of features from the backbone as input and outputs a list of fused features that represent salient characteristics of the image.
3) The final class/box network that uses the fused features to predict the class and location of each object.

III.5.Conclusion:

In this chapter, we have talked about how artificial intelligence started and it's revolution on this world, we listed the differences and the importance of using deep learning methods, as well as we explained how the neurons works and how it developed itself, and finally we presented the ways of the object detection, how the bounding box work and seeing the different algorithms and which is the good and the compatible to our project.

Chapter IV

Object detection with YOLOv5

IV.1. Introduction

Computer vision technology uses a variety of imaging systems instead of visual organs as input means, using computers to replace the brain to complete the processing and interpretation of visual information. So in this chapter we will use the Algorithms of Yolo to detect the objects we need and explaining the components we used to transfer the real time video from the drone to the processing board Nvidia Jetson Nano

IV.2. OBJECT DETECTION MODEL

IV.2.1. What is Yolo?

You Only Look Once (YOLO) is an algorithm that uses convolutional neural networks for object detection.

It is one of the faster object detection algorithms out there.

It is a very good choice when we need real-time detection, without loss of too much accuracy.

YOLO has only 9 convolutional layers, so it's less accurate but faster and better suited for mobile and embedded projects. Darknet53 (The backbone used in YOLOV3) has 53 convolutional layers, so it's more accurate but slower. In YOLOv5, backbone can be CSPNet.

IV.2.2. Why YOLO?

YOLO was chosen as the object detection algorithm for this study (You Only Look Once). This algorithm detects and classifies objects quickly using a single convolutional neural network (CNN). In comparison to other object detection methods, it was chosen for its real-time accuracy as well as the public availability of its source code. The structure allows for real-time object classification at 45 frames per second, making this algorithm ideal for human detection on surveillance video in the event of a disaster or emergency.

YOLO accounts for the entire image (rather than focusing on just one region of interest), resulting in fewer background errors than other state-of-the-art detection methods like Fast R-CNN. Fast R-CNN is regarded as one of the most effective object detection algorithms (Redmon et al. 2016). When comparing the performance of Fast R-CNN and YOLO, it was discovered that Fast C-RNN had nearly three times as many background errors as YOLO.

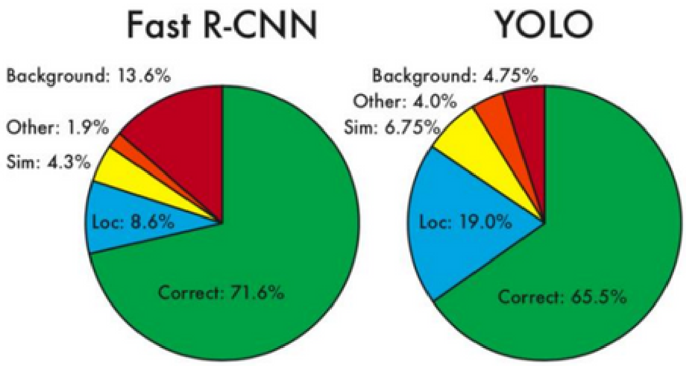

Figure IV.4 A comparative breakdown of YOLO and Fast R-CNN

When an algorithm misaligns the predicted bounding box with the ground truth, it is known as a localization error (the algorithm thinks the object is in a slightly different location than it is in reality). When the algorithm detects an object that does not exist, background errors occur. Higher background accuracy outweighs the negative effects of localization error for human detection in search and rescue.

The real-time time detection threshold is also missed by Fast R-CNN (30 frames per second). On the well-known Pascal VOC 2007 dataset, the figure shows a comparison of state-of-the-art real-time object detection methods. For real-time object detection, YOLO had the highest mAP (mean Average Precision) value. Fast R-CNN achieved higher mean average precision at the expense of computing time, but fell short of real-time detection classification at 0.5 frames per second (frames per second). When it comes to real-time search and rescue, YOLO remains the best option for object detection.

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

Table IV.1 Real-time object detection comparison of leading methods

IV.2.3. YOLO Basics

Due to its high speed and average precision, as well as its open-source availability, YOLO remains a top choice algorithm for object detection, according to research. The algorithm's functionality can be broken down into three basic steps.

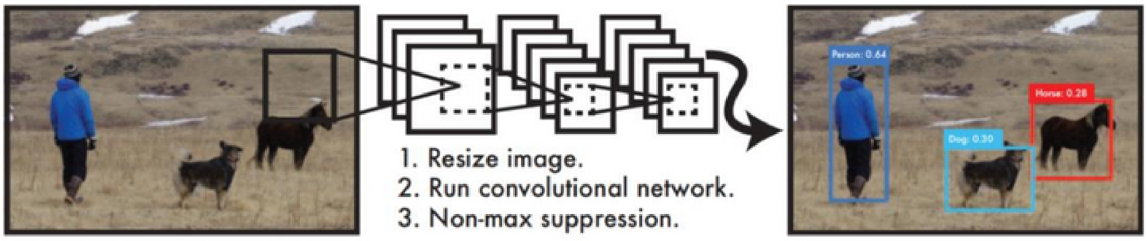

Figure IV.5 Object detection with YOLO

The next diagrams show YOLO's three-step process for detecting and classifying objects. The image is first divided into a grid with a S X S dimension. Each cell predicts the number of bounding boxes, which is represented by the variable B, after the image has been divided into a grid. Grid cells with fully enclosed objects are in charge of detecting the object inside.

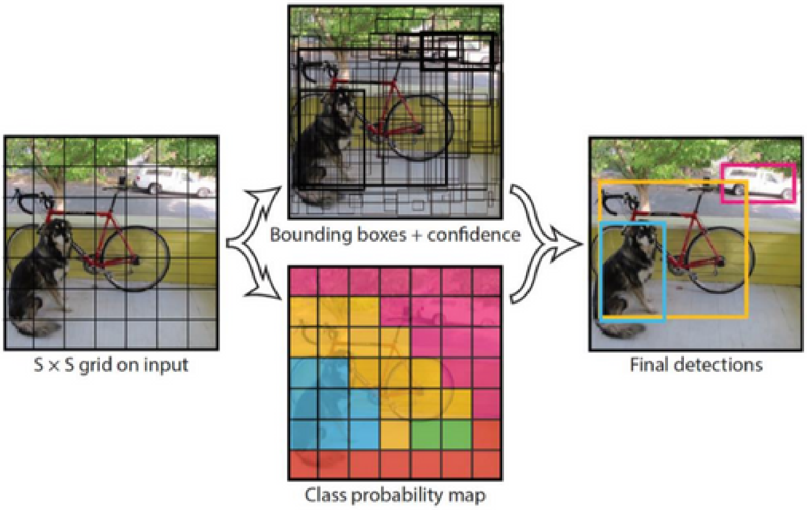

Figure IV.6 YOLO three step object detection process

For each box, confidence scores are predicted, reflecting the model's level of confidence in determining if the box indeed bounds an object, as well as in predicting what the object is.

IV.3 IMPLEMENTING YOLOV5 ALGORITHM

IV.3.1. Human/Person Detection:

Computer vision systems use human detection and tracking to locate and follow people in video imagery. Human detection is the process of locating all instances of human beings in an image by searching all locations in the image at all possible scales and comparing a small area at each location with known templates or patterns of people. The process of temporally associating human detections within a video sequence to generate persistent paths, or trajectories, of people is known as human tracking. Human detection and tracking are generally considered the first two processes in a video surveillance pipeline, and can feed into higher-level reasoning modules such as action recognition and dynamic scene analysis.

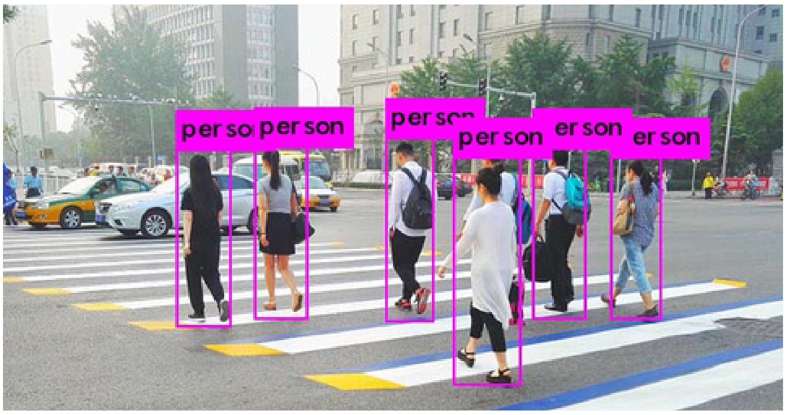

Figure IV.7 Example of Detecting People

IV.3.2. YOLO Releases

YOLO v1 was introduced in May 2016 by Joseph Redmon with paper "You Only Look Once: Unified, Real-Time Object Detection." This was one of the biggest evolution in real-time object detection.

In December 2017, Joseph introduced another version of YOLOv2 with paper "YOLO9000: Better, Faster, Stronger." it was also known as YOLO 9000.

After a year in April 2018, the most popular and stable version of YOLO was introduced. Joseph had a partner this time and they released YOLOv3 with paper "YOLOv3: An Incremental Improvement".

in April 2020, YOLOv4 was introduced with some astounding new things, It outperformed YOLOv3 with a high margin and also has a significant amount of average precision when compared to EfficientDet Family.

After a few weeks, exactly on 9 June 2020, just four days back another unofficial author Glenn Jocher released YOLOv5. There are lots of controversies about the selection of the name "YOLOv5" and other stuff. Glenn introduced PyTorch based version of YOLOv5 with exceptional improvements. Hence he has not released any official paper yet.**[32]**

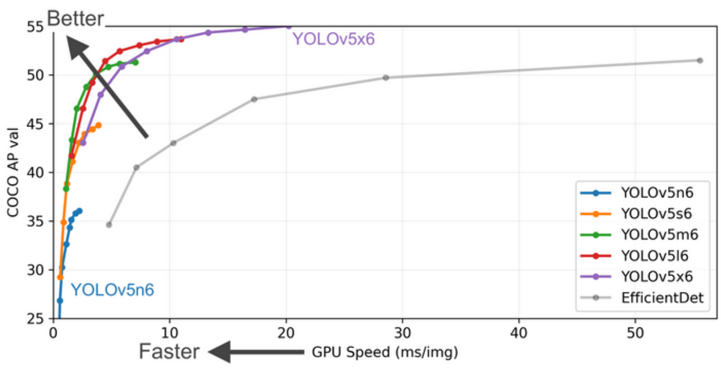

Figure IV.8 Difference between Yolov5 versions **[32]**

- yolov5-n which is a nano version
- yolov5-s which is a small version
- yolov5-m which is a medium version
- yolov5-l which is a large version
- yolov5-x which is an extra-large version

YOLOv5's initial release is extremely fast, performant, and simple to use. While YOLOv5 does not add any new model architecture to the YOLO family, it does add a new PyTorch training and deployment framework that improves the state of the art for object detectors. Furthermore, YOLOv5 is extremely user-friendly and comes "out of the box" ready to use on custom objects.

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	36.0	54.4	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.8	63.7	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.3	69.3	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.7	71.3	1784	15.8	10.5	76.8	111.4
YOLOv5x6	1280	55.0	72.7	3136	26.2	19.4	140.7	209.8
+ TTA	1536	**55.8**	**72.7**	-	-	-	-	-

Table IV.2 Details about the difference between Yolov5 versions **[32]**

IV.3.3. Yolov5 Setup

The first thing we want to do is install YOLOv5. You'll get this from the GitHub's repository

https://github.com/ultralytics/yolov5/releases

Next, we need to install Python, they recommend installing the version 3.7 and up, but we will install the latest version from python's website.

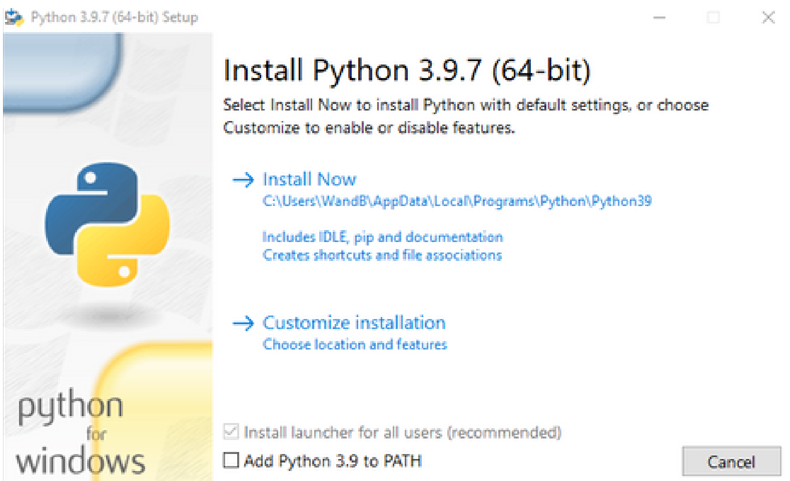

Figure IV. 9 Python Installation

Now we pass to CUDA installation.

CUDA is NVIDIA's parallel computing platform for their GPUs, basically it allows your machine to run parts of the computation in parallel "which it provides a lot of speed"

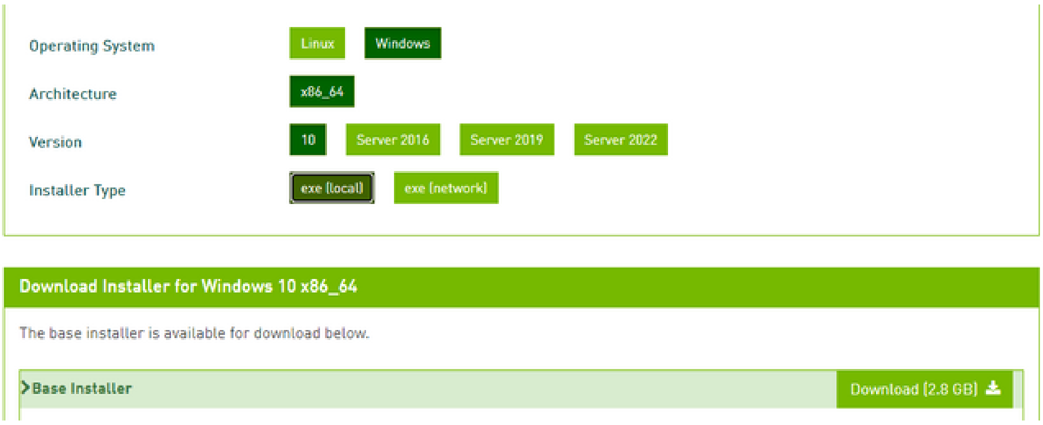

Figure IV.10 Cuda Downloading

After it's downloaded we launch the file and install:

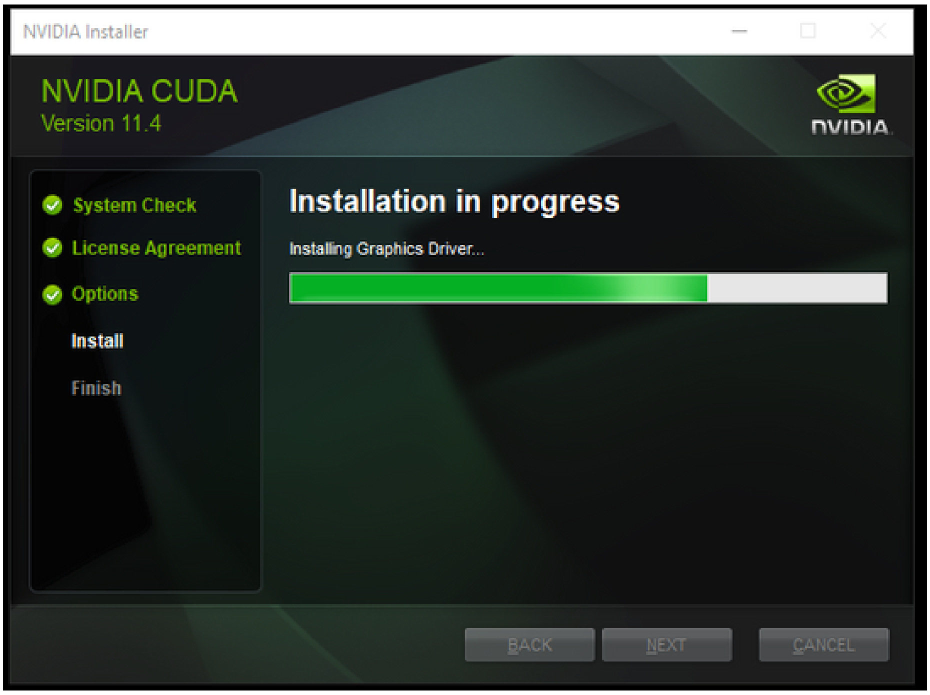

Figure IV.11 Cuda Installation

Installing PyTorch

PyTorch is the most popular machine learning frameworks to define models, run inference (what we'll be doing here) and perform training. and as the graph shows how trendy PyTorch was.

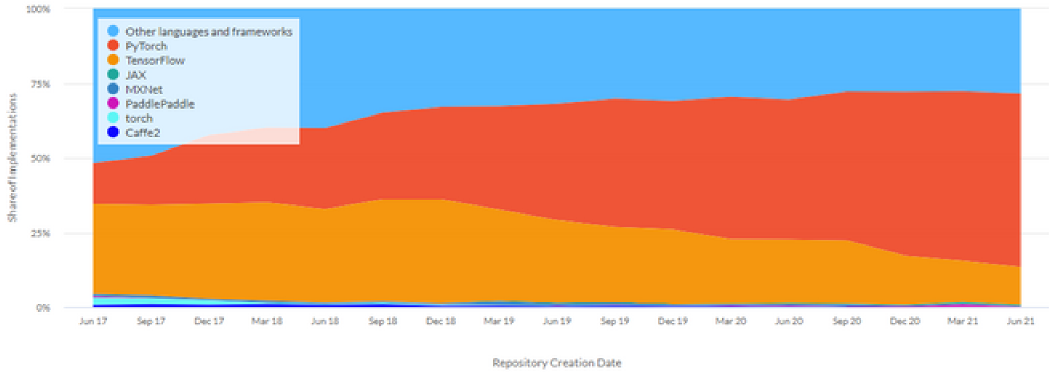

Figure IV.12 The volume of Pytorch users

PyTorch, like most machine learning frameworks, runs calculations and inference on the GPU which is significantly faster than running them on the CPU.
Installing PyTorch As A Python Module

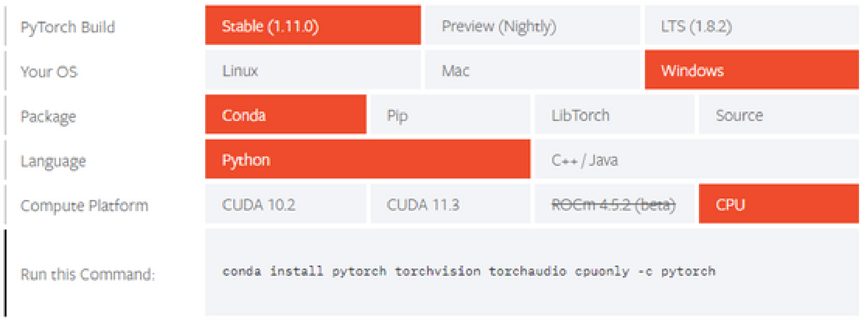

Figure IV.13 Pytorch Downloading

Now we need to install some additional modules that are found in the YOLOv5 requirements. From the Console we'll now run the following to install the required modules:

```
pip install -r requirements.txt
```

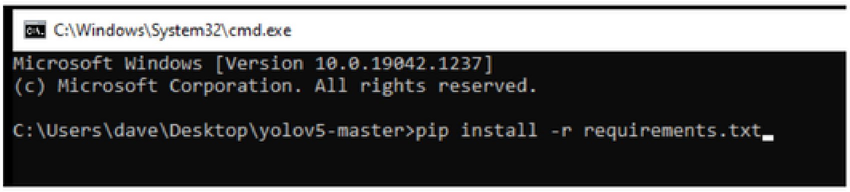

Figure IV.14 Pytorch Installing

Creating Bounding Boxes With YOLOv5 On Your Webcam
And Now we'll make a test with the internal webcam to check if the installation goes well

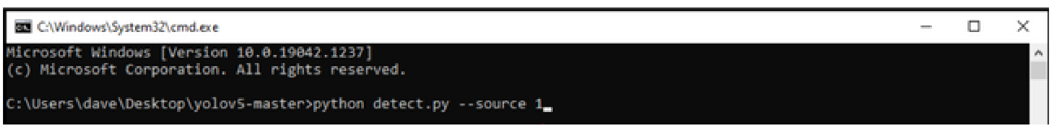

Figure IV.15 Applying Yolov5

After few moments from running it, we'll get this:

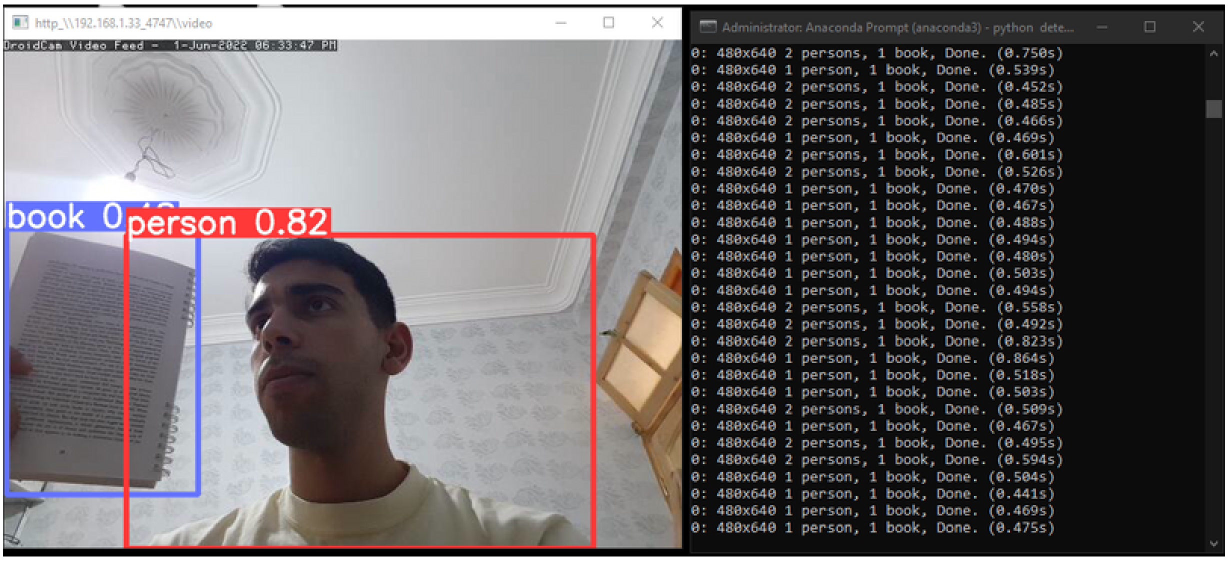

Figure IV.16 Testing Yolov5 on the front camera

IV.3.4. Example with Yolov5 using Google Colab

Here is an example of a picture includes multiple objects and running the algorithm of YOLOv5 on Google colab to detect them, not just detcting these objects but also getting the coordinates of each object, let's start with the algorithm:

```
[ ] import torch

    # Model
    model = torch.hub.load("ultralytics/yolov5", "yolov5l")  # or yolov5m, yolov5l, yolov5x, custom

    # Images
    img = "/content/image_filtering.jpeg"  # or file, Path, PIL, OpenCV, numpy, list

    # Inference
    results = model(img)

    # Results
    results.print()  # or .show(), .save(), .crop(), .pandas(), etc.
    results.save()
    print("----------------------------------------------------------------------")
    print(results.pandas().xyxy[0])
    print("----------------------------------------------------------------------")
    print(results.xyxy[0])
```

Figure IV.17 Testing Yolov5 on Google Colab

"/content/image_filtering.jpeg" this is the path of the location of the camera on Google colab, and here is our sample picture:

Figure IV.18 Sample picture for detection on Google colab

After running the algorithm, we will get a our sample picture with the detected objects, as the picture shows, there are 13 objects detected only in one picture. And it doesn't give us only the detected objects but their coordinates too, layers and the parameters that used to detect them. Here it is:

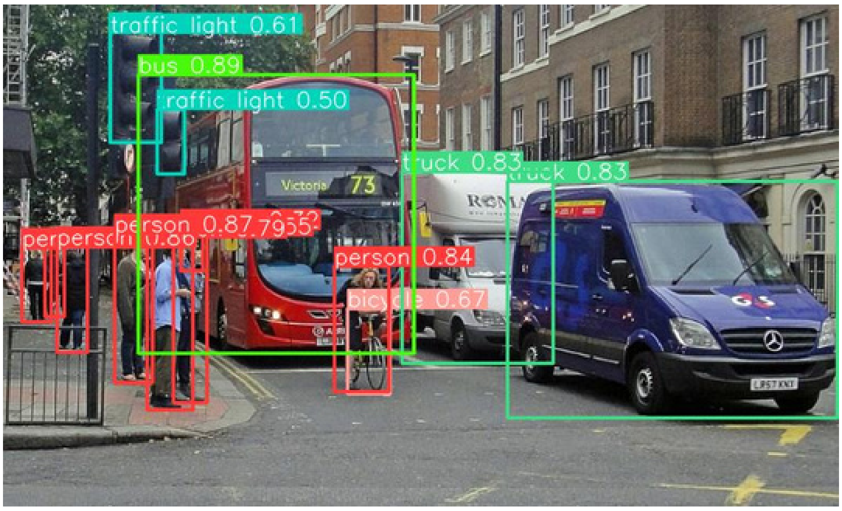

Figure IV.18 Sample picture after running YOLO on Google colab

and these are the information it showed :

```
Using cache found in /root/.cache/torch/hub/ultralytics_yolov5_master
YOLOv5 🚀 2022-2-7 torch 1.10.2+cu102 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model Summary: 367 layers, 46533693 parameters, 0 gradients
Adding AutoShape...
image 1/1: 461x768 8 persons, 1 bicycle, 1 bus, 2 trucks, 2 traffic lights
Speed: 6.6ms pre-process, 45.4ms inference, 1.6ms NMS per image at shape (1, 3, 416, 640)
Saved 1 image to runs/detect/exp11
```

Figure IV.18 All the info about the picture on Google colab result

And finally here are the coordinates of each object :

	name	xmin	ymin	xmax	ymax	confidence	class \
0	bus	124.740402	64.503998	377.141754	320.942322	0.890298	5
1	person	102.655739	210.870331	137.596695	347.077759	0.873126	0
2	person	49.363827	222.802841	77.182220	319.126892	0.857055	0
3	person	17.937988	223.077103	44.291721	291.614502	0.849582	0
4	person	304.224792	240.171555	354.014557	355.375183	0.838321	0
5	truck	366.640808	153.669647	505.511963	329.707367	0.831683	7
6	truck	463.062775	162.331863	766.483765	379.124023	0.831134	7
7	person	132.658600	214.482864	174.844482	372.915222	0.792098	0
8	bicycle	316.354584	279.445099	355.087708	357.004761	0.665685	1
9	traffic light	98.216553	8.914574	145.505112	126.925209	0.614283	9
10	person	156.006897	212.889847	187.324371	365.589661	0.545510	0
11	traffic light	141.735703	96.302147	166.756851	155.703598	0.500186	9
12	person	163.460709	206.455032	183.392334	244.459747	0.320147	0
13	person	38.316811	223.024246	56.312469	286.237793	0.302911	0

Table IV.3 All Coordinates of each object in our sample picture

IV.3.5. Video transferring from the quadrotor

In our project we will use an external camera that shows us the video in real-time from the quadrotor

VTX stands for video transmitter, it's a device that sends the video from the camera in the quadrotor to a video receiver which can display it on the computer.

FPV camera means first person view, it may has medium video quality but it has a high speed data transfer "video transfer" throw the radio transmitter and in real-time

RTX stands for video receiver, it's a device that receives the video from the transmitter that linked to camera in the quadrotor and we can display it on the pc with a USB cable easily

Figure IV.17 a)VTX Video Transmitter b) FPV camera c) VRX Video receiver

IV.3.6. Nvidia Jetson Embedded system

The Jetson Nano is a small, powerful computer designed to power entry-level edge AI applications and devices. It can makes us getting started quickly with the comprehensive NVIDIA JetPack™ SDK, which includes accelerated libraries for deep learning, computer vision, graphics, multimedia, and more, and we will use it to process center for our video using YOLO because of its powerful GPU.

Figure IV.18 Jetson Nano Board

Jetson Nano comes with a 4GB of Ram and a 128-core of NVIDIA Maxwell, a 4K HDMI video and 40 GPIO Pinouts, and here are the hardware pinouts:

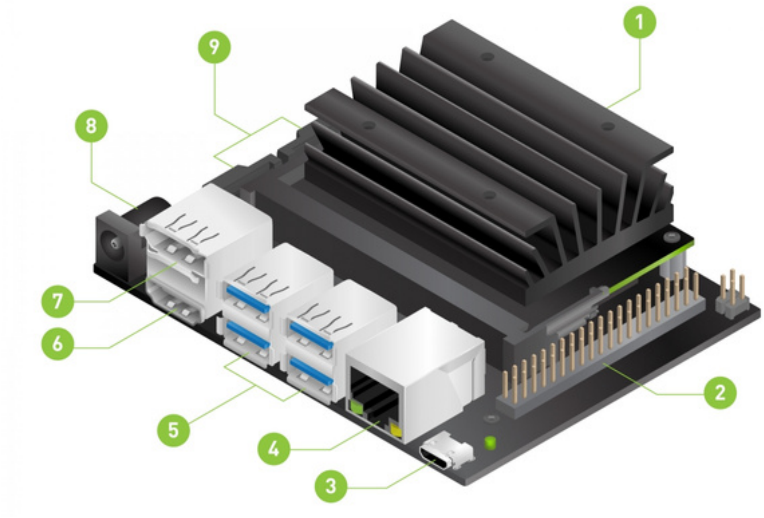

Figure IV.19 Jetson Nano Input/Output

1. microSD card slot for main storage
2. 40-pin expansion header
3. Micro-USB port for 5V power input, or for Device Mode
4. Gigabit Ethernet port
5. USB 3.0 ports (x4)
6. HDMI output port
7. DisplayPort connector
8. DC Barrel jack for 5V power input
9. MIPI CSI-2 camera connectors

Figure IV.20 The final prototype

IV.4. Conclusion

In this chapter we have presented Yolov5 and why exactly we chose it and the way to get started with it, we also saw the power and the rapidity of Yolo compared to other algorithms. we described our equipment of video transferring from the quadrotor to the computer or specifically to the algorithm, it may need a powerful computer but with the small version of Yolo "s" the algorithm worked well without any problems.

General Conclusion

As an Unmanned Aerial Vehicle (UAV), a quadrotor is a rotorcraft with four powered rotors arranged symmetrically around its center. It is classified as a Vertical Take-Off and Landing (VTOL) aircraft. This structure has recently gained popularity among several applications, in particular for surveillance, imaging, dangerous environments, indoor navigation, mapping, pipeline inspection, and football stadium security, due to its unique characteristics, such as low dimension, good maneuverability, simple mechanics, payload capability, and vertical, stationary, and low-speed.

The main focus of this thesis was to find objects as humans in a specific area, that is capable to process images with an FPV camera. the quadrotor platform was created for the purpose of using the mission mode to send it to any place we want. we used Pixhawk because of its stability and diffrent modes as well as the simplicity of using the mission planner software.

Before we developed the object detection system, we analyzed the controllers, the mathematical model of the quadrotor as well as the dynamics of the system. In order for the model to reflect the actual system, the inertia of the quadrotor was measured and the motor parameters were identified. A Simulink model is formed to be able to simulate quadrotor behavior and test the controllers within the 3 main angles and the axe z.

After that we passed to our main goal which is detecting objects and we focused on human and cars detection because it's guided to them, and the most problems it solve are losing them on the desert, forest or any difficult places to reach even tracking cars, or even detecting fires, or using agricultural crops to distinguish good crops from bad ones. We can say it can be a multi problem solver to any domain.

In conclusion, the quadrotor platform and the object detection system have performed successfully in altitude control and especially in the detection side, but could not reach the desired levels in terms of the components quality and the computer performance. It can be stated that the system or the prototype would work in the next level if the necessary parts were available.

As perspective for future work

- Seeking to improve the equipment, especially the camera and its transmitter.
- sending a localization with a picture throw a message after detecting an object while it is in a mission.

BIBLIOGRAPHY

[1] Behnam Nourghassemi "Development of the Control Algorithms for Autonomous Landing of Unmanned Aerial Vehicles" IEEE Conference Paper | Jan-Mar 2009

[2] Fouad Yacef, Omar Bouhali and Mustapha Hamerlain, "Adaptive Fuzzy Tracking Control of Unmanned Quadrotor via Backstepping" IEEE Conference Paper · June 2014.

[3] Fouad Yacef, Omar Bouhali and Mustapha Hamerlain "Adaptive Fuzzy Backstepping Control for Trajectory Tracking of Unmanned Aerial Quadrotor" 2014 International Conference on Unmanned Aircraft Systems (ICUAS), May 27-30, 2014. Orlando, FL, USA.

[4] Mohd Ariffanan Mohd Basri, Kumeresan A. Danapalasingam and Abdul Rashid Husain "Design and Optimization of Backstepping controller for an underactuated autonomous Quadrotor unmanned aerial vehicle." Transactions of famena 2014.

[4] Bousbaine, Bamgbose, T Poyi and and K. Joseph "Design of Self-tuning PID Controller Parameters Using Fuzzy Logic Controller for Quad-rotor Helicopter" International Journal of Trend in Research and Development, Volume 3(6), IJTRD | Nov-Dec 2016.

[5] LABIDI Hemza, ZEGHLACHE Hemza and MAZOUZ Aboubaker essaddiq "Methodes de commande par backstepping appliquees la stabilisation d'attitude a quatre rotor." Engineering thesis, University of M'sila 2010.

[6] Oussama BOUDGUIGA "Commande a saturation pour le contrôle de la position d'un robot volant type quadrotor '' Master thesis, Quebec University 2016.

[7] Mokhtari Mohammed Rida "Observation et Commande de Drones Miniatures à voilures tournantes '' PhD thesis, Tlemcen University 2015.

[8] Monia MECHIRGUI, ''Commande optimale minimisant la consommation d'energie diun drone utilisé comme relai de communication. '' Master thesis, Quebec University, 2014.

[9] Zoran Benić, Petar Piljek and Denis Kotarski "Mathematical modelling of Unmanned aerial vehicles with four rotors" Interdisciplinary Description of Complex Systems 14(1), 88-100, 2016.

[10] H. Bouadi, M. Bouchoucha, and M. Tadjine "Sliding Mode Control based on Backstepping Approach for an UAV Type-Quadrotor" International Journal of Mechanical and Mechatronics Engineering Vol:1, No:2, 2007.

[11] B. Safaee and S. Kamaleddin Mousavi Mashhadi "Optimization of fuzzy membership functions via PSO and GA with application to quadrotor" Journal of AI and Data Mining. Vol 5, No 1, 2017, 1-10.

[12] BOUADI Hakim, ‘’ synthese de lois de commande stabilisantes pour un uav type quadrotor.’’ Magister thesis, Ecole militaire polytechnique, Algeria.

[13] Heba talla Mohamed Nabil ElKholy ‘’ Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches" Master thesis, the American university in Cairo, 2014.

[14] L. Camilli, "Emerging technologies, applications, regulations, and market challenges in the consumer aerial drone industry," Technical Report, San Francisco State University College of Business, 2015.

[15] L. Camilli, "Emerging technologies, applications, regulations, and market challenges in the consumer aerial drone industry," Technical Report, San Francisco State University College of Business, 2015.

[16] Pounds P, Mahony R, Hynes P, Roberts J (2002). Design of a FourRotor Aerial Robot," Australian Conference on Robotics and Automation, Auckland.

[17] S. R. Barros dos Sanrors, C. L. Nascimento, S. N. Givigi, "Design of attitude and path tracking controllers for quadrotor robots using reinforcement learning", Proc. of the 2012 IEEE Aerospace Conference, pp. 1-16, 2012

[18] J. R. Handershot and T. J. E. Miller, Design of Brushless Permanent Magnet Motors. Oxford, U.K.: Oxford Univ. Press, 1994

[19] https://docs.px4.io/v1.12/en/flight_controller/pixhawk_series.html

[20] Matejka, Stepan (2015). [IEEE 2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA - Pardubice, Czech Republic) All digital FPGA based PWM modulator for radio frequency transmitters. , (), 244–247.

[21] Haenlein, Michael, and Andreas Kaplan. "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence." California management review 61.4 (2019): 5-14.

[22] El Naqa, Issam, and Martin J. Murphy. "What is machine learning?." In machine learning in radiation oncology, pp. 3-11. Springer, Cham, 2015.

[23] Callahan, Alison, and Nigam H. Shah. "Machine learning in healthcare." In Key Advances in Clinical Informatics, pp. 279-291. Academic Press, 2017.

[24] Wang, Sun-Chong. "Artificial neural network." In Interdisciplinary computing in java programming, pp. 81-100. Springer, Boston, MA, 2003.

[25] Szegedy, Christian, Alexander Toshev, and Dumitru Erhan. "Deep neural networks for object detection." Advances in neural information processing systems 26 (2013).

[26] Medsker, Larry R., and L. C. Jain. "Recurrent neural networks." Design and Applications 5 (2001): 64-67.

[27] Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network." In 2017 international conference on engineering and technology (ICET), pp. 1-6. Ieee, 2017.

[28] Li, Pengyu, Biao Wang, and Lei Zhang. "Virtual fully-connected layer: Training a large-scale face recognition dataset with limited computational resources." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13315-13324. 2021.

[29] Amit, Yali, Pedro Felzenszwalb, and Ross Girshick. "Object detection." Computer Vision: A Reference Guide (2020): 1-9.

[30] Rajeshwari, P., P. Abhishek, P. Srikanth, and T. Vinod. "Object detection: an overview." Int. J. Trend Sci. Res. Dev.(IJTSRD) 3, no. 1 (2019): 1663-1665.

[31] Deng, Li, and Dong Yu. "Deep learning: methods and applications." Foundations and trends® in signal processing 7, no. 3–4 (2014): 197-387.

[32] https://github.com/ultralytics/yolov5/releases

Abstract

In this work, a type of unmanned aerial vehicle (UAV) called quadrotor with an object detection system is the subject that we will talk about. The main objectives of our work is the detection of chosen objects from the quadrotor. For that, there are two parts presented, the first one, a detailed description of the mathematical effects that applied to the structure of our system. In the second part, we will talk about artificial intelligence generally and deep learning specifically and we will show how exactly the detection happen using the right technics and algorithms. Finally we will make a real test with the real time showing our prototype working in the field.

Keywords: UAV, Object Detection, Quadrotor, Deep Learning.

ملخص

في هذا العمل ، سوف نتحدث عن نوع من المركبات الجوية غير المأهولة يسمى كوادروتور مع نظام الكشف عن الأجسام. تتمثل الأهداف الرئيسية لعملنا في الكشف عن الكائنات المختارة من خلال الطائرة بدون طيار. لذلك ، حيث ينقسم هذا العمل اي قسمين ، الأول ، تقديم نموذج رياضي كامل للكوادروتور مع الاخذ بعين الاعتبار جميع الظواهر الفزيائية و الايروديناميكية التي يخضع لها هذا النظام. في الجزء الثاني ، سنتحدث عن الذكاء الاصطناعي بشكل عام والتعلم العميق على وجه التحديد وسنوضح كيف يحدث اكتشاف الاشياء بالضبط باستخدام التقنيات والخوارزميات المناسبة. أخيرًا ، سنقوم بإجراء اختبار حقيقي في الوقت الفعلي لإظهار نموذجنا الأولي الذي يعمل في المجال.

الكلمات المفتاحية: الطائرات بدون طيار ، الكشف عن الاشياء ، كوادروتور ، التعلم العميق.

Résumé

Dans ce travail, un type de véhicule aérien sans pilote (UAV) appelé quadrotor avec un système de détection d'objets est le sujet dont nous parlerons. L'objectif principal de notre travail est la détection d'objets choisis à partir du quadrirotor. Pour cela, deux parties sont présentées, la première, une description détaillée des effets mathématiques qui s'appliquent à la structure de notre système. Dans la deuxième partie, nous parlerons de l'intelligence artificielle en général et de l'apprentissage en profondeur en particulier et nous montrerons comment exactement la détection se produit en utilisant les bonnes techniques et algorithmes. Enfin, nous ferons un test réel avec le temps réel montrant notre prototype fonctionnant sur le terrain.

Mots-clés : drone, détection d'objets, quadrirotor, profonds d'apprentissage.