**Université Abou Bakr Belkaïd-Tlemcen**
**Faculté de Technologie**
**Laboratoire d'Automatique de Tlemcen LAT**

**MÉMOIRE DE FIN D'ÉTUDES**

Pour l'obtention du Diplôme de

**MASTER EN AUTOMATIQUE.**

**Spécialité: AUTOMATIQUE ET INFORMATIQUE INDUSTRIELLE**

Présenté par le binôme **KADOUCI Abdenasser and BENCHAO Abderrafiq**

# Remote sensing by embedded vision on an autonomous UAV for precision agriculture

**soutenu le 6 Juillet 2021 devant le jury:**

| | | |
|---|---|---|
| Mr HADJ ABDELKADER Amine | Pr | Président. |
| Mr BANSALAH Choukri | MCB | Examinateur. |
| Mme. CHOUKCHOU-BRAHAM Amal | Pr | Encadrante. |
| Mr. MOKRANE Adel | | Co-Encadrant. |

**Année universitaire 2020-2021.**

# Thanks

First of all we thank God who gave us the courage and the will to reach our goal.

We deeply thank the head of Laboratory d'Automatique de Tlemcen LAT **M. Amine HADJ ABDELKADER** for welcoming us to the laboratory and for the good working condition.

We would like to express our heartfelt thanks and our deep gratitude to **Mme. Amal CHOUKCHOU-BRAHAM** for her confidence granted to us by agreeing to supervise us, her listening and advices, and for her availability throughout the development of this master until the end, where she has been a great contribution to us.

We also thank **Mr. MOKRANE Adel** for his warm welcome, and his support from the start as well as for the trust he placed in us.

We extend our gratitude to the members of the jury.

We thank all our teachers at the GEE department of Abou bakr Belkaid University at Tlemcen.

Our sincere thanks to our parents that we hope they will always be proud of us.

Finally, we thank all those who have contributed from near and far to the achievement of this work.

# Dedication

“

*It is my honor, to dedicate this work to my parents for all the emotional support and guidance through my life time, the night they spent awake for me is a dept that no money can pay for so I can reach to this point.*

*To my friends Nirmo Yacine, Bob Marli, and the future doctor Ben-Abdelmoumen Mohamed for the mental support through my education life.*

*To Chihab-eddin and Mouad for working with us to improve our project.*

*To Ghizlene Ahmadouch for her support and being the kindest girl I ever met.*

*To all my teachers through primary school, middle school, high school up to the university for pushing me to present my best.*

*To my partner in this work Abderrafik that helped me making this project a reality.*

*There are many dear people that I forgot to mention, To all of them I say*

*Thank you.*

”

***- Abdenasser***

“

*C’est avec tout fierté, avec tout respect que je dédie ce modeste travail. A mes chers parents et ma grande mère. Qui ont allumés mon chemin du savoir et qui m’ont encouragés et soutenu depuis mon enfance jusqu’à mon soutenance. Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon formation. C’est grâce à vous que je suis arrivé aujourd’hui à ce niveau d’étude.*

*A mes deux frères Abdedjalile et Mohamed et mes chéres soeurs.*

*A les anges sont mes nièces : Hanae, Safae, Wafae, Wissal, meriem.*

*Je dédie aussi ce travail à tous mes amis Bilal, Khali msirdi, Omar, Saide-ahemed, Smail, Bouzide, Rachid, Alae, Houssem, Mohammed, Youcef.*

*Mes remerciements les plus sincères à Fatima Zohra qui m’a apportée son aide et qui est contribuée à l’élaboration de ce travail.*

*A mon binôme Abdenasser qui a partagé avec moi ce travail.*

*A tous ceux ou celles qui me sont chers et qui j’ai omis involontairement à citer.*

*Merci.*

”

***- Abderrafiq***

# Abstract

Farming is an essential work for the humankind and the main source of food. Precision farming is the beginning of a bigger project which is the farm of the future. It will use fewer labor workers better energy and water consumption resulting in high production, lower time and cost.

In our topic we will focus on using drones to analyze agricultural information where it will be used to make decisions, for example controlling the water supply if the plants are weak, fruit counting, launching the piking process using an unmanned ground vehicle UGV if the products are ready .

The mapping of an agricultural field based on the NDVI vegetation index is not always sufficient to identify areas of contamination or diseases detected in the vegetation. It sometimes requires the collection of samples in the field, for a more precise analysis in ex-situ. This project consists of equipping a quad-rotor drone made in a former PFE, an Odroid XU-4 electronic card, a camera, and an GPS to perform the task of taking samples in the field. Visual feedback and GPS data can position the platform on visual targets with precision (± 1cm). The optimal trajectory which passes through all the points of interest (way-points) avoiding obstacles is established by the application of CPP (Coverage path planning) where the vehicle must intelligently plan it trajectories while avoiding existing obstacles in the environment then proceed to fruit counting to estimate the necessary storage.

Google maps is great for general view of an area but the lack of resolution and update rate makes it unusable for precision agriculture, in this project we are able to use alternative maps with higher resolution and even apply coverage path planning algorithms on them and also create a flower counting mission at the end

**Keywords :** Quadrotor drone, autonomous piloting, Odroid XU-4 card, Computer vision, GPS, image processing, CPP, precision agriculture.

# Contents

# List of Figures

# List of Algorithms

# List of symbols

| | |
|---|---|
| **a** | *lift slope* |
| **b** | *Thrust factor* |
| **d** | *Drag factor* |
| **g** | *Acceleration due to gravity* |
| **h** | *Vertical distance: Propeller center to CoG* |
| **i** | *Motor current* |
| **J** | *The inertia of the system*p |
| $J_r$ | *Rotor inertia* |
| $k_{ft}$ | *The translational drag coefficient* |
| $K_{fa}$ | *The coefficient of aerodynamic friction* |
| **l** | *Horizontal distance: propeller center to CoG* |
| **R** | *Rotation matrix* |
| **T** | *Thrust force* |
| **U** | *Control inputs* |
| **x, y, z** | *Position in body coordinate frame* |
| **V** | *The linked linear speed* |
| **v** | *Speed vector* |
| **F** | *Force* |
| **m** | *Overall mass* |
| $F_f$ | *The total force generated by the four rotors.* |
| $F_g$ | *Force of gravity.* |
| $M_f$ | *Moment caused by forces* |

# Acronyms

| | |
|---|---|
| **UAV** | *Unmanned Aerial Vehicle* |
| **ODROID** | *Open source Android* |
| **RGB** | *Red, Green, Blue* |
| **OpenCV** | *Open-source Computer Vision* |
| **PID** | *Proportional, Integrator, Derivative regulator* |
| **MPU** | *Manuvl Programming Unit* |
| **NVDI** | *Normalized Difference Vegetation Index* |
| **GPU** | *Graphics Processing Unit* |
| **CUDA** | *Compute Unified Device Architecture* |
| **HSV** | *Hue, Saturation, Value* |
| **LiDAR** | *Light Detection and Ranging* |
| **RAO** | *Retinal Artery Occlusion* |
| **RVO** | *Retinal Vein Occlusion* |
| **CPP** | *Coverage Path Planning* |
| **GPS** | *Global Positioning System* |
| **UGV** | *Unmanned Ground Vehicle* |
| **BLDC** | *Brushless Direct Current electric motor* |

# General Introduction

## 0.1 Problem Statement

The human population is growing exponentially, we are closing on 8 billion, we will surpass that at a fast rate and that is terrifying. Traditional farming will not be sufficient to supply the general demand, relying on heavy gas-guzzling agricultural vehicles will not last forever cause the oil will be drained someday with other natural resources and we will be at higher numbers.

So we need to develop our food supply lines to secure the population feed by developing the technology in this domain to improve our productivity and avoid classical relaying on oil based techniques.

## 0.2 Proposition

Among the solutions we can use autonomous machines, in fact they work faster, use less energy, less time and give higher output and have the ability to analyse all the information in the environment.

Analyzing such huge data requires a system that has great accessibility to every part of the field and takes as much information as possible from the environment.
Drones have a high degree of freedom that allows them to reach all locations in an open space, the advantage that makes them very useful in agriculture applications for example crops rusting instead of using piloted aircraft or gathering data from the field.
To achieve that, we are using a drone that was built in a previous project based on the "pixhawk" platform and in our project we will connect it with a single-board computer called "ODROID", it allows us to analyze information from an "RGB" camera using "OpenCv", then use the data to control the movement of the drone or just save it to control other machines. In our application we are also going to build a fruit counting drone that will follow an optimal path to cover the entire area and return with precise data.

## 0.3 Thesis Organization

**State of the art :** in this chapter, we give a quick overview on drones, particularly those used in precision agriculture the projects that have been made at the past 5 years in this topic by our supervisor teams at the "Laboratoir d'Automatique de Tlemcen LAT" where we explain generally how they where able to make multiple functional drones with different platforms.
**Dynamic modeling of UAV :** in this chapter we will talk about modeling a quad-copter drone and the physics around it in a general manner.
**Computer vision :** in this chapter we will talk computer vision and give some examples of its applications and at the end we will explain a fraction of the software tools in our applications .
**Path planning  fruit counting :** in this chapter we will explain what is Coverage path planning, how we can optimize it, and the advantage we got from this update. Then we will move to fruit counting procedure and explain the reason behind it, how to preform it at low time and cost.
**Realisation and results:** in this chapter we will talk about how we are going to apply the information of the previous chapters, and the hardware that is being used to build it. we will talk about the "pixhawk" and the drone related components , the "ODROID" and the computer vision related components.
At the end we will discuss the results of our application and the inconveniences that we came across.

# Chapter 1

# State of the art

## 1.1 Introduction

Drones commonly referred, as UAVs are mostly associated with military, industry and other specialized operations but with recent developments in area of sensors and Information Technology in last two decades the scope of drones has been widened to other areas like Agriculture [**17**]. For years now, drone advocates have cited precision agriculture - crop management that uses GPS and big data - as a way to increase crop yield while resolving water and food crises [**24**]. The drone's application in precision farming varies from the ability to analyze images of crops taken from high altitude to rusting huge fields. In this chapter, we will talk about some real-world application in agriculture,and the previous projects regarding drones in our university and the results that they came up with and also the material that they used, and finally we will talk about the platform that we are going to use.

## 1.2 History of drones

The idea of drones started way back in the 1849 unfortunately it was used for military purposes, when a war was fought between Austria and Venice. The Austria attacked Venice through "unmanned balloons" Figure(1.1) which were filled with explosive bombs. It is said that the number of balloons used at that time were two hundred approximately (Kennett, 1982). Though, those air-balloons don't satisfy the conditions of the present UAVs which are now in use in combat but still the impression of those unmanned balloons was quite operational and different. That effective idea used in 1849 can now be observed in present world in the form of UAVs [1].

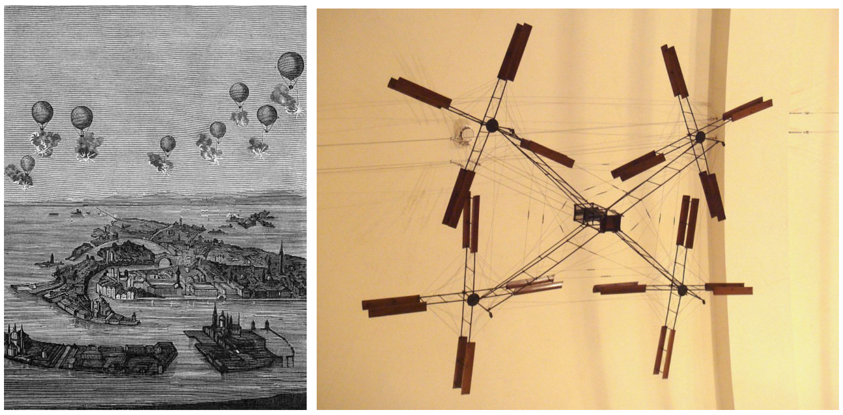


Figure 1.1: Venice attack 1849 and first quad copter

In 1907, the Breguet Brothers built their first human carrying helicopter–they called it the Breguet-Richet Gyroplane No.1, which was a quad-rotor. Clearly the Breguet's approached the problem of the helicopter more scientifically than others at the time, and thought hard about a configuration that was most likely to meet with success. In the words of Louis Breguet: "....Starting from from my calculations I was lead....to build an aircraft with four propellers [rotors] of 8.10 meter diameter." The issue of stability was a consideration in the design, but the first requirement for the machine was simply to lift itself and a pilot off the ground under its own [13]
Then technology saw a big leap forward in 1915 when the world introduced to the first unmanned areal vehicle after the outbreak of World War I.
Using A.M. Low's radio control techniques, the Royal Flying Corps' Ruston Proctor Aerial Target was built in 1916. The concept was to develop a small, very simple aircraft, pack it with explosives and then guide it into a designated target. Shortly after, on 12th of September, Hewitt-Sperry Automatic Airplane made its first flight. These two concept planes are considered the ancestors of today's cruise missiles.[22]

Figure 1.2: The first unmanned areal vehicle and the modern unmanned areal vehicle

Going on further to the 1990's where we can find that the drones application is thriving in the US army capable to fly for long range and deploy missiles or even searching for targets, Figure(1.2)
The drones development was focused on military application unfortunately for more than a century unfortunately. Now we can see that the world is open to other beneficial application for it in photography, agriculture, search and rescue or even fire fighting Figure(1.3)

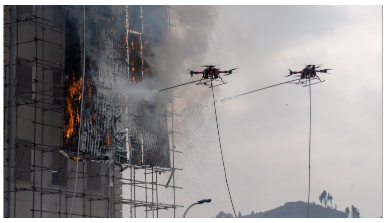


Figure 1.3: High-Rise Fire Fighting Drones

In this project we are interested in drone applications in precision farming, in the next section we will talk about some examples of agriculture drones.

## 1.3 Examples of agriculture drones

### 1.3.1 Soil and field analysis

Drones can be very handy at the start of the crop cycle, it can give an optimum strategy to use the provided ground and benefit from soil as much as possible, They build precise 3-D maps for the ground analysis, and determine seed planting patterns, Figure (1.4)

Figure 1.4: SOLO AGCO Edition

### 1.3.2 Crop monitoring:

We can optimise production by building vast fields but that will come with a cost of loosing the ability to monitor the entire crops add to that the unpredictable weather conditions which increase the maintenance costs.
Using satellite imagery is not very efficient method, the images needs to be pre-ordered and it can be taken only once a day, without mention of the high cost, bad precision, and the image quality on general.
Drones can give better image resolution and better update frequency at lower cost, Figure(1.5)

Figure 1.5: EBEE SQ-SenseFly & Lancaster 5 Precision Hawk

### 1.3.3 Planting :

Many rich lands around the world have great agricultural potential, but they are not being used because of the difficult landscape. Drones have high degree of freedom that allows them to reach such location and perform the planting process. It does not stop at that, there is a team in Australia who built a drone that shoots seed in the ground, by them it can plant up to 40,000 trees a day with a team of 2 people, Figure(1.6)

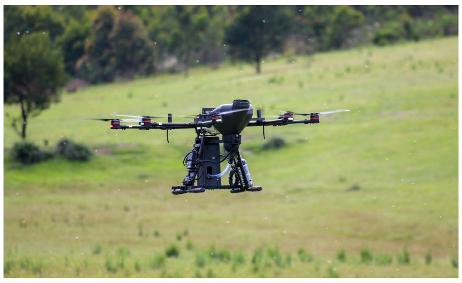


Figure 1.6: AirSeed planting drone

### 1.3.4 Crop spraying :

Today vast farms use heavy-duty tractors to spray the crops and for some places, they use airplanes to cover more space and gain time both of them have a common issue, both of them aren't environment friendly and they cost in term of energy and money.
Drones represent the optimal solution, use less energy and cost, and by keeping the development it will be more affordable in close future as the technology advance. It can perform completely autonomous or even coordinated as swarms to have faster results with more precision, Figure(1.7)

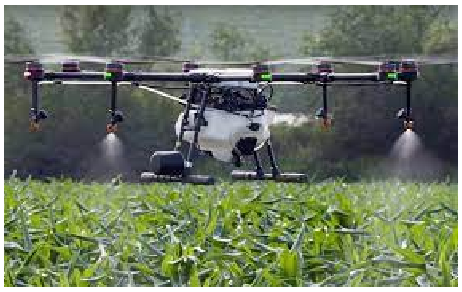


Figure 1.7: Agras MG-1-DJI

### 1.3.5 Irrigation :

We can equip drones with various types of sensors, from ultrasonic to infrared cameras, GPS, Lidar sensor. This will allow us to analyze and make decisions about the parts that need to be taken care of in the field, monitoring the growth of the crops, predicting the density and health of the crops, Figure(1.8)

Figure 1.8: DJI Matrice 210 V2 + Micasense RedEdge-MX + Zenmuse XT2

## 1.4 Benchmark

The vast applications of drones and their potentials in scientific research and technology led the automatic laboratory of Tlemcen to start looking at this domain. Many projects started in 2017 and continue until now.
3 projects have been made in this field in the last few years 2017, 2019, and 2020 without mention of the ones made at undergraduate projects, we will use these projects as a benchmark for our own.

### 1.4.1 2017 project

This project is made by Meslouli Ismat & Mesli Riad Merwane it was the first project that succeeds to build the first functional drone in the region without prior experience. The project focused on identifying the optimal PID regulator for the drone and the optimal components by building a precise model for the entire system, the pixhawk adropilot is compatible with Mission planner software that means it is easy to configure for RC cars or boats, and in this case drones.

However the PID parameters that they came up with the identification where not accurate so they change them by testing until they found a suitable results .
the problem was in the handmade asymmetric structure and some identification techniques that resulted in wrong parameters in the end . Figure(1.9)

Figure 1.9: The test fly of the project of 2017

### 1.4.2 2019 project

This project was made by SEDINI Chahrazed & CHERIGUI Nasre-Eddine, the project focused on building the drone algorithm from scratch using the Arduino which is a basic micro-controller, it doesn't have the embedded gyroscope and some other components needed for the drone as the pixhawk.

They needed to add an external MPU and other circuits. the biggest challenge was the memory of the Arduino itself but they managed to go through and succeed to make it fly at the end and recording the flight information by the xbee telemetry, and the PID parameters where very close to the identification results they got by improving the techniques used in the prior project.

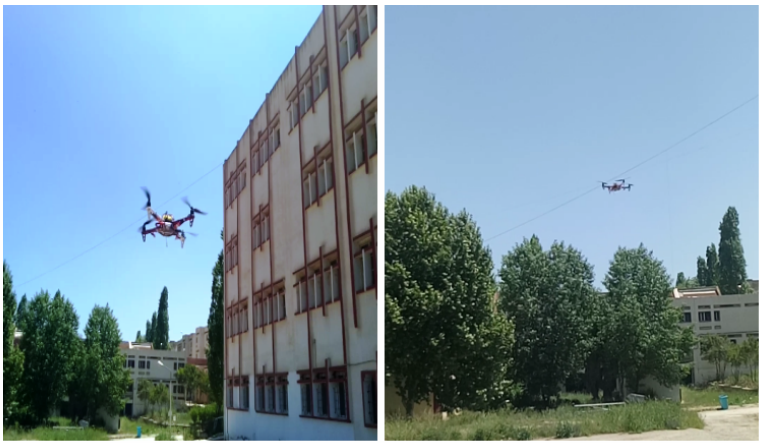


Figure 1.10: the test fly of the project of 2019

### 1.4.3 2020 project

This project was made by BOUGHRARA Samira Kawther & BOUTRIGUE Sarah, the project focused on NVDI detection to estimate the green areas density in a certain land and drones are perfect tools for such application.

The project uses pixhawk to control the drone connected with an on board computer to analyse the images taken for the field using opencv the biggest challenge was to make them communicate with each other

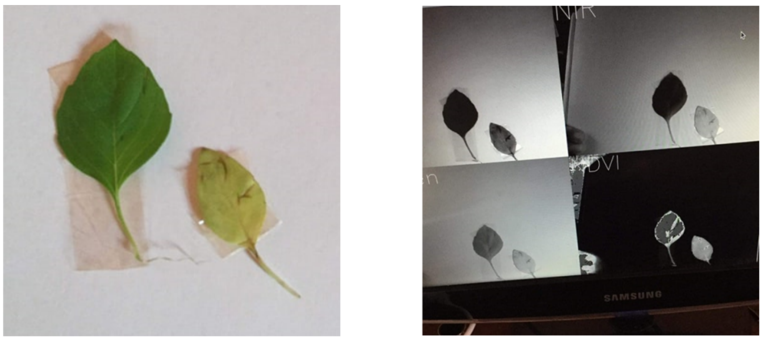


Figure 1.11: NVDI results of 2020 project

### 1.4.4 Coverage Path Planning

Mapping and Exploration of unknown environments are important missions for intelligent robots in order to attain complete autonomous behaviors. Maps are the most natural representation of robot's environment. In order to represent places in an environment, maps may contain other information like reflecting properties of objects, unsafe regions or difficult to traverse, and information gained from prior experiences.

There are many documents made about this process by Adel MOKRANE , Amal CHOUKCHOU-BRAHAM and Brahim CHERKI. these documents emphasize generating an optimal trajectory for drones in huge farming fields to analyze the data in it. Among these tasks, we propose to perform the fruit counting by taking images at low altitude. The documents indicate that satellite images have low resolution and give low details, so to give better results we can take high pictures of the field using the drones themselves and generate a trajectory from them with that we can have a higher resolution and enhance the details.

## 1.5 Conclusion

In this chapter we talked about the farming industry and the benefits of implementing drones in it, we also cited some examples of existing agriculture drones. We mentioned also some of the projects that have been made in this topic in the past years and the result that they came up with, at the end we talked about the current project and what we are going to do exactly.

# Chapter 2

# Dynamic modeling of UAV

## 2.1 Introduction

In order to use a flight controller, one must first deeply understand the movements of the aircraft, its dynamics and therefore its dynamic equations. This understanding is needed not just to control it, but also to ensure that simulations of UAV behavior are as close to reality as possible when the command is applied. In this chapter a description of the operating principle of the quadrotor drone is presented with a detailed description of its possible movements as well as the flight modes. The quadrotor is classified in the category of the most complex flying systems because of the number of physical effects that affect its dynamics, namely aerodynamic effects, gravity, gyroscopic effects, friction and moment of inertia [8], [9]. This complexity results essentially from the fact that the expression of these effects is different for each mode of flight. Indeed, the dynamic models of the quadrotor proposed in [2]change according to the planned tasks and according to the navigation environments defined a priori by the operator.
This chapter is devoted to the dynamic modeling for the control of the quad-rotor system.After having given the description of this system, the general structure and the principle of flight, we highlight our model adopted to this system.

## 2.2 General description of the quad-rotor

A quadrotor is an aerial mobile robot with four rotors[19]. These 4 rotors are usually placed at the endsof a cross, and the control electronics are usually placed in the center of the cross.In order to prevent the device from turning on itself on its yaw axis, it is necessary that two propellers turn in one direction, and the other two in the other direction. To be able to direct the device, it is necessary that each couple of propellers turning in the same direction is placed at the opposite ends of a branch of the cross.

The operation of a quadrotor is quite particular. By cleverly varying the power of the motors, it is possible to make it go up/down, to tilt it to the left/right (roll) or forwards/backwards (pitch) or even to make it rotate on itself (yaw) [18].

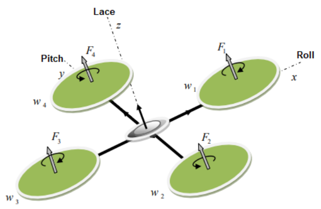


Figure 2.1: General structure of a quadrotor.

## 2.3 The Movements of the quadrotor

Basic quadrotor movements are achieved by varying the speed of each rotor thereby changing the thrust produced. The quadrotor tilts towards the direction of the rotor slower, which then takes into account the translation along this axis. Therefore, as to a conventional helicopter, the movements are coupled, meaning that the quadrotor cannot achieve travel without rolling or pitching, which means that a change in speed of a rotor translates into motion in at least three degrees of freedom. For example, increasing the left thruster speed will result in a roll motion (the quadrotor tilts towards the slower rotor, to the right), yawing (the balance between the rotors which rotates clockwise and the rotors which rotates clockwise. direction is disturbed resulting in a horizontal rotational movement), and a translation (the rolling motion tilts the armature and with it, the orientation of the push). This coupling is the reason why we can control the six degrees of quadrotor freedom with only four controls (the torque applied by the motors on each thruster)[21].

The quadrotor has five main movements:

-Vertical movement.

-Roll movement.

-Pitch movement.

-Yaw movement.

-Horizontal translations.

### 2.3.1 Vertical movement

In order to glide, the total lift force should only be along the z axis with a magnitude exactly opposite to the force of gravity. Besides, the lift force created by each rotor must be equal to prevent the vehicle from overturning more. Therefore, the thrust produced by each rotor must be the same.

The upward and downward movement is obtained by varying the speed of rotation of the motors (consequently the thrust produced), if the lift force is greater than the weight of the quadrotor themovement is upward, and if the lift force is less than weight of the quadrotor the movement is downward.

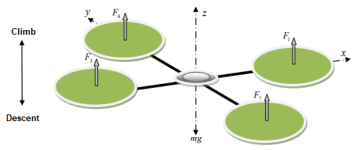


Figure 2.2: Illustration of vertical movement

### 2.3.2 Roll motion

Figure (2.3) shows how a rolling motion is obtained. In this case, we apply a torque around the x axis, i.e. by applying a difference in thrust between the rotor 2 and rotor 4. This movement (rotation around the x-axis) is coupled with a movement of translation along the y axis.

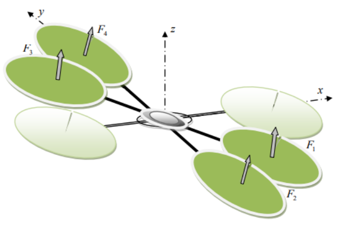


Figure 2.3: Illustration of the roll motion

### 2.3.3 Pitch movement

Figure (2.4) shows how a pitch motion is obtained. In this case, we apply a torque around the y-axis, i.e. by applying a difference in thrust between the rotor 1 and rotor 3. This movement (rotation around y) is coupled with a movement of translation along the x axis.

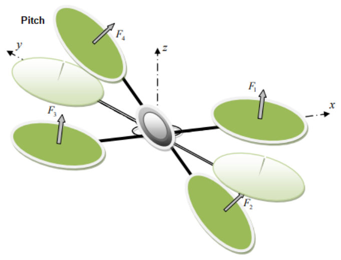


Figure 2.4: Illustration of the pitch motion

### 2.3.4 Yaw movement

Figure (2.5) shows how the yaw movement is obtained. In this case, we want to apply a torque around the z axis, which is done by applying a speed difference between the rotors 1,3 and 2,4. This movement is not a direct result of the thrust produced by the thrusters but by the reactive torques produced by the rotation of the rotors. The direction of the pushing force does not shift during movement, but the increasing force of lift in one pair of rotors must equal the decrease in other pairs to ensure that all the pushing force remains the same.

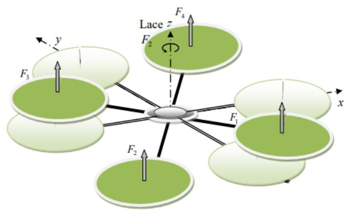


Figure 2.5: Illustration of the yaw movement

### 2.3.5 Transnational movements

Figure (2.6) shows how the horizontal translation is performed. In this case we want to apply a force along x or y which is done by tilting the body (by pitching or rolling) and increasing all the thrust produced to keep the importance of the z component of the thrust. equal to the force of gravity.

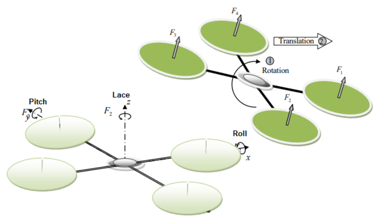


Figure 2.6: Illustration of the translational movement

## 2.4 Dynamic model of the quadrotor

The modeling of flying robots is a delicate task since the dynamics of the system are strongly nonlinear and fully coupled. In order to be able to better understand the model dynamic developed below, here are the different working hypotheses[9]:

-The structure of the quadrotor is assumed to be rigid and symmetrical, which implies that the matrix of inertia will be assumed to be diagonal.

-The propellers are supposed to be rigid in order to be able to neglect the effect of their deformation during of rotation.

-The center of mass and the origin of the coordinate system related to the structure coincide.

-The lift and drag forces are proportional to the squares of the rotation of the rotors, which is a very close approximation of the behavior aerodynamic.

### 2.4.1 Benchmarks used

A quadrotor requires two trihedrons to locate it in space, these landmarks are [12] [3]:

1. The terrestrial reference: It is noted: R0 (O0, X0, Y0, Z0). It is a landmark linked to the earth figure (2.7) assumed immobile.

2. The mark linked to the body of the quadrotor: The mark linked to the body of the quadrotor It is noted: R1 (O1, X1, Y1, Z1). It is a landmark whose origin O1 coincides with the center of gravity G of the quadrotor.

The parameters which describe the movement of the quadrotor are:

($\phi, \theta, \psi$, x, y, z,$\omega$, V) with:

-$\phi$(roll angle) : rotation around X1 ( $-\pi < \phi < \pi$).

-$\theta$ (pitch angle): rotation around Y1($-2\pi < \theta < 2\pi$).

- $\psi$(yaw angle) : rotation around Z1($-\pi < \psi < \pi$).

- x: coordinate of the center of gravity G of the following quadrotor X0.

- y: coordinate of the center of gravity G of the following quadrotor Y0.

- z: coordinate of the center of gravity G of the following quadrotor Z0.

-$\Omega$: [p, q, r] T $\in$ R0: the speed of rotation of the quadrotor with respect to inertial frame.

- V: [u, v, w] T ∈ R0: the linked linear speed of the quad-copter compared to inertial reference frame.

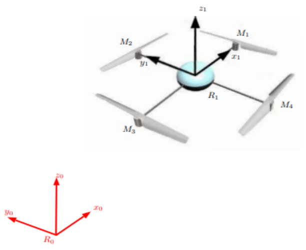


Figure 2.7: the quad-copter frames

### 2.4.2 Rotation matrix

It is considered that the centers O0 and O1 of the two references are the same, which means that the reference R1 only rotates with respect to the reference R0. the transition matrix from R1 to R0 is given by the rotation matrix[6]:

$$R = \begin{bmatrix} C\theta C\psi & -C\phi S\psi + S\phi S\theta C\psi & S\phi S\psi + C\phi S\theta C\psi \\ C\theta S\psi & C\phi C\psi + S\phi S\theta S\psi & -S\phi C\psi + C\phi S\theta S\psi \\ -S\theta & S\psi C\theta & C\psi C\theta \end{bmatrix} \tag{2.1}$$

$$with : C = cos, and S = sin$$

### 2.4.3 Angular speeds

The speeds of rotations $\Omega 1, \Omega 2, \Omega 3$ in the fixed reference frame are expressed according to the speeds of rotations $\phi, \theta, \psi$ in the mobile coordinate system, we have [2]:

$$\Omega = \begin{bmatrix} \Omega x \\ \Omega y \\ \Omega z \end{bmatrix} = \begin{bmatrix} \phi \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \theta C\phi \\ -\theta S\phi \end{bmatrix} + \begin{bmatrix} -\psi S\theta \\ \psi S\phi C\theta \\ \psi C\phi C\theta \end{bmatrix} = \begin{bmatrix} \phi - \psi S\theta \\ \psi S\phi C\theta + \theta C\phi \\ \psi C\phi C\theta - \theta S\phi \end{bmatrix} \tag{2.2}$$

$$\Omega = \begin{bmatrix} 1 & 0 & -S\theta \\ 0 & C\phi & S\phi C\theta \\ 0 & -S\phi & C\phi C\theta \end{bmatrix} \tag{2.3}$$

When the quadcopter makes small rotations, the following approximations can be made:
$C\phi = C\theta = C\psi = 1; S\phi = S\theta = S\psi = 0$

Therefore, the angular speed will be:

$$\Omega = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix} \tag{2.4}$$

### 2.4.4 Linear speeds

The linear speeds $V_x^b V_y^b V_z^b$ in the fixed coordinate system as a function of the linear speeds $V_x^m V_y^m V_z^m$ in the moving coordinate system are given by[2]:

$$V = \begin{bmatrix} V_x^b \\ V_y^b \\ V_z^b \end{bmatrix} = R \times \begin{bmatrix} V_x^m \\ V_y^m \\ V_z^m \end{bmatrix} \tag{2.5}$$

We define new the physical effects acting on the quad-rotor.

### 2.4.5 Forces

**The forces acting on the system are[2]:**
**weight of the quadrotor:** it is given by $P = mg$, where: m is the total mass and g is 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 rotational speed of the motors:

$$F_i = b\omega_i^2 \tag{2.6}$$

With $i = 1 : 4$ and b is the coefficient of lift, it depends on the shape and number of the blades and the density of the air.
**Drag forces:** drag force is the coupling between a compressive force and the force viscous friction, in this case there are two drag forces acting on the system that they are:
-The drag in the propellers: it acts on the blades, it is proportional to the density of air, the shape of the blades and the square of the speed of rotation of the propeller, it is given by the following relation:

$$T_h = d\omega^2 \tag{2.7}$$

Where d is the drag coefficient it depends on the manufacture of the propeller.

-The drag along the axes $(x, y, z)$ : it is due to the movement of the body of the quadrotor

$$F_t = K_{ft} V \tag{2.8}$$

with: $K_{ft}$ the translational drag coefficient and V linear speed

### 2.4.6 The Moments

There are several moments acting on the quadrotor, these moments are due to the forces of thrust and drag and gyroscopic effects[2].

**Moments due to thrust forces:** -The rotation around the x axis: it is due to the moment shouted by the difference between the forces of lift of rotors 2 and 4, this moment is given by the following relation:

$$M_x = l(F_4 - F_2) = lb(\omega_4^2 - \omega_2^2) \tag{2.9}$$

where l is the length of the arm between the rotor and the center of gravity of the quadrotor.

-The rotation around the y axis: it is due to the moment shouted by the difference between the forces lift of rotors 1 and 3, this moment is given by the following relation:

$$F_y = l(F_3 - F_1) = lb(\omega_3^2 - \omega_3^2) \tag{2.10}$$

**Moments due to drag forces:**

-The rotation around the z axis: it is due to a reactive torque caused by the torques of drag in each propeller, this moment is given by the following relation:

$$M_z = (\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \tag{2.11}$$

-Moment resulting from aerodynamic friction, it is given by:

$$M_a = K_{fa} \Omega^2 \tag{2.12}$$

With $K_{fa}$ : The coefficient of aerodynamic friction and $\Omega$ is the angular velocity.

### 2.4.7 Gyroscopic effect

The gyroscopic effect is defined as the difficulty of modifying the position or the orientation of the plane of rotation of a rotating mass. The gyroscopic effect is so named with reference to the operating mode of the gyroscope, motion control device used in aviation (from the Greek gyro which means rotation and scope, to observe). In our case there are two gyroscopic moments, the first is the gyroscopic moment propellers, the other is the gyroscopic moment due to the quadrotor movements[2].

-Gyroscopic moment of the helices: it is given by the following relation:

$$M_{gh} = \sum_{1}^{4} \Omega \wedge J_r \begin{bmatrix} 0 & 0 & (-1)^{i+1}\omega_i \end{bmatrix}^T \tag{2.13}$$

with $J_r$ is the inertia of the rotors.
-Gyroscopic moment due to quadcopter movements: it is given by the relation next :

$$M_{gm} = \Omega \wedge J\Omega \tag{2.14}$$

with J is the inertia of the system.

## 2.5 Development of the mathematical model according to Newton-Euler

The different dynamic models of the quadrotor developed in the literature which can be classified into two categories: modeling according to Euler-Lagrange and modeling according to Newton-Euler. The Old PFE who carried out in this subject, in this part are taken the Newton-Euler method[20], [14].

### 2.5.1 The Newton-Euler formulation

For the development of the mathematical model, the drag force along the axes and the moment due to aerodynamic friction were neglected. The inertia tensor is assumed to be symmetrical and therefore the inertia matrix is diagonal. Using Newton-Euler's formulation, the equations are written in the following form:

$$\begin{cases} \xi' = V \\ m\xi'' = F_f + F_g \\ R' = RS(\Omega) \\ J\Omega' = -\Omega \wedge J\Omega + M_f - M_gh \end{cases} \tag{2.15}$$

With:
$\xi$ : The position vector of the quadrotor.
m : the total mass of the quadrotor.
$\Omega$ : The angular speed expressed in the fixed reference.

R ; The rotation matrix.
$\wedge$ : The cross product.
J : symmetrical inertia matrix of dimension (3x3).
S($\Omega$) : The antisymmetric matrix.
$F_f$: The total force generated by the four rotors.
$F_g$: Force of gravity.
$M_f$: Moment caused by forces.

### 2.5.2 System state representation

Our nonlinear state space x' = f (x; u) with the 12-dimensional state vector can be represented as follows[14]:
$x = (x_1; x_2; \dots x_{12})^T = (\phi; \phi'; \theta; \theta'; \psi; \psi'; x; x'; y; y'; z; z')^T$
So the state representation is as follows:

$$\begin{cases} x_1' = x_2 \\ x_2' = \frac{J_{yy}-J_{zz}}{J_{xx}} x_4 x_6 - \frac{J_r g(u)}{J_{xx}} + \frac{U_2}{J_{xx}} \\ x_3' = x_4 \\ x_4' = \frac{J_{zz}-J_{xx}}{J_{yy}} x_2 x_6 - \frac{J_r g(u)}{J_{yy}} + \frac{U_3}{J_{yy}} \\ x_5' = x_6 \\ x_6' = \frac{J_{xx}-J_{yy}}{J_{yy}} x_2 x_6 + \frac{U_4}{J_{zz}} \\ x_7' = x_8 \\ x_8' = \frac{U_1}{m}(C(x_1)S(x_3)C(x_5) + S(x_1)S(x_5)) \\ x_9' = x_{10} \\ x_{10}' = \frac{U_1}{m}(C(x_1)S(x_3)S(x_5) + S(x_1)C(x_5)) \\ x_{11}' = x_{12} \\ x_{12}' = \frac{U_1}{m}(C(x_1)C(x_3)) - g \end{cases} \tag{2.16}$$

with : C = cos, S = sin

## 2.6 Conclusion

This chapter provides the reader with some preliminary concepts about flying robots and how they work. The quadrotor is presented with a detailed description of its possible movements citing landmarks then use the flight modes. we have presented the differential equations representing the dynamics of the quadrotor as well as the forces, moments generated and we have cited the acting physical effects. We used the Newton-Euler formalism which allowed us to obtain a simplified model. The following chapter, we will present the integration of more specific components (Pixhawk, ODROID UX4, camera, GPS) in order to create our project.

# Chapter 3

# Computer vision

## 3.1 Introduction

Every living creature with eyes can analyze the 3-dimensional world with ease especially humankind, we are able to identify each other and even other objects around us. It does not stop at that, we can determine the speed, sizes, and even position of these objects. Computer vision refers to the theory and implementation of artificial systems that extract information from images to understand their content [5]. The ultimate goal of machine vision is image understanding the ability not only to recover image structure but also to know what it represents. By definition, this involves the use of models which describe and label the expected structure of the world [4]. Computer vision is also a main tool in machine learning and data science, stacking a set of pictures to train your computer to recognize a certain object or a certain action mimic the learning properties of biological creatures. Thankfully the computer vision has huge support from the scientific community and provided with great open-source platforms specifically opencv that allows the developers to be creative Figure(3.1).

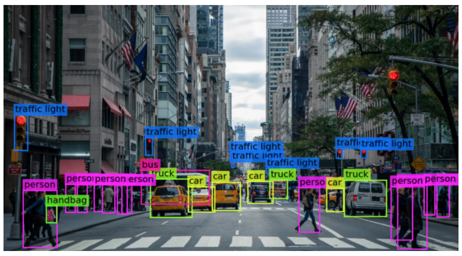


Figure 3.1: An application of computer vision

Computer vision is a rapidly growing field devoted to analyzing, modifying, and high-level understanding of images. Its objective is to determine what is happening in front of a camera and use that understanding to control a computer or robotic system, or to provide people with new images that are more informative [16].
In this chapter, we will explain computer vision, its capabilities and the many applications in the engineering domain it can offer and also talk about a great open source library called OpenCV and potentials it can offer for new developers.

## 3.2 OpenCV

Opencv is an abbreviation for open-source computer vision library Figure(3.2), it was started at Intel in 1999 by Gary Bradsky, and the first release came out in 2000.
Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day. OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

### 3.2.1 OpenCV-Python

The most famous program that is being used with it is Python, huge resources and fan base are using this platform to do different applications in GitHub, stack-flow and even YouTube tutorials, it varies from facial recognition to 3D reconstruction of an image or object tracking and more.

Python is a general purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation, Figure(3.2).

Figure 3.2: Opencv-Python

OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

### 3.2.2 Some operations of images

#### RGB images

An image is a combination of pixels forming a matrix where every pixel represent the intensity of the color in that position.
We have 3 channels represent : Red, Green, blue, by combining these 3 main colors with different intensities we can form all visible color combination possible forming what is called RGB images, Figure(3.3). For black  white images has a single channel forming the Grey-scale images. We can generate the grayscale images from the RGB images but that is not applied to the other way around.

$$greyscale = \frac{(Red + Green + Blue)}{3} \tag{3.1}$$

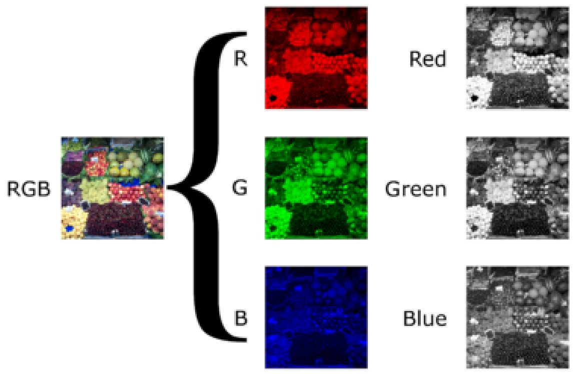


Figure 3.3: RGB image channels

#### Binary images

There is also the binary images where the pixels have only two possible intensity values either 1 or 0 by applying a threshold to the image or extracting a certain color range, it is very handy for detecting and tracking objects in an open environment making it very useful for our application, Figure(3.4).

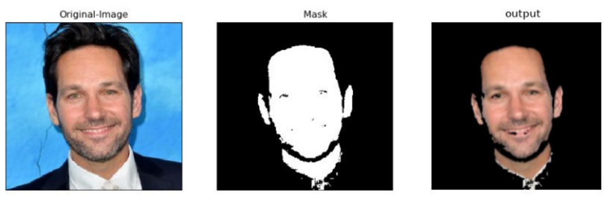


Figure 3.4: binary image

### HSV images

HSV or HSB refers to Hue, Saturation, Value or Brightness are alternative dimension for RGB images, it is used in drawing or image modification applications because it simplify the configuration of any color needed. If we represent an RGB pixel by a Cartesian coordinates, the HSV graph will give us the cylindrical coordinates.
**Hue :** it represent the color spectrum itself by forming a circle that contain all colors possible, the circle start from red at 0 degree up to blue at 240 degree and back to red Algorithm(1).
**Saturation :** it represent how far the color is from the grey-scale, for example if it is 0 we will be at the grey image algorithm(2).
**Value :** it represent the effect of the lighting at the image, that will be very handy when we need to extract a certain color in diffident brightness situations Algorithm(3).
we can represent the HSV form easily in this figure(3.5)

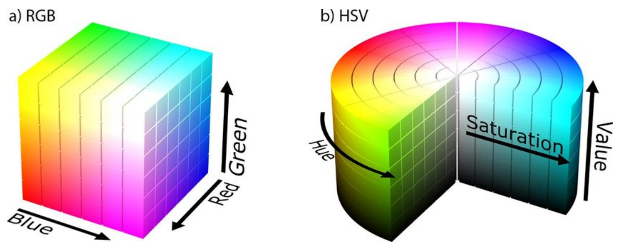


Figure 3.5: HSV image

### Erosion and Dilation

Erosion and Dilation are binary image operations that form the base of almost all other morphological operations. It works by calculating the gradient of the image to detect the edges and apply the Erosion or the Dilation. **Erosion** remove pixels from the boundaries of the white object figure(3.6) and **Dilation** add pixels to the boundaries of the white object Figure(3.6).

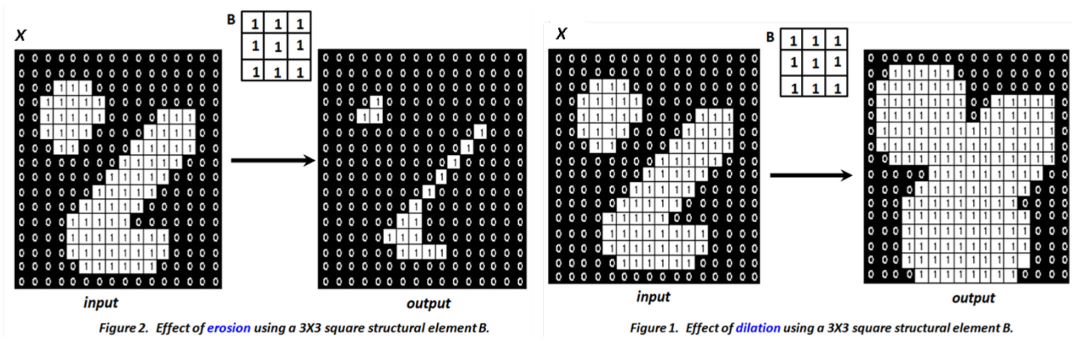

X
B
input
output
Figure 2. Effect of erosion using a 3X3 square structural element B.
X
B
input
output
Figure 1. Effect of dilation using a 3X3 square structural element B.

Figure 3.6: Erosion and Dilation

### Opening and closing morphology

Opening and closing morphology works in binary images, it represent a low pass filter that allows us to get red off small noises. **The opening** is formed by applying erosion then dilation that will allow us to remove white noises. **The closing** is formed by applying dilation then erosion that will allow us to remove black noises.
we can see the difference in this Figure(3.7)

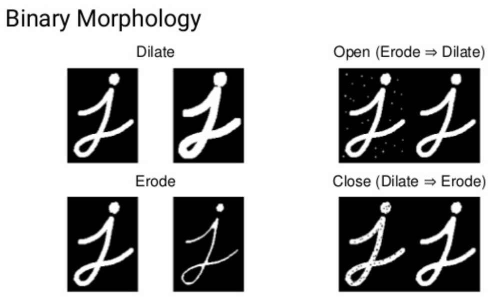


Figure 3.7: Opening and Closing morphology

## 3.3 Computer vision application examples

Images provide more data than other types of sensors in a compact form, but this data needs treatment to suit the applications that it is made for. For that computer vision is a very handy tool in industrial applications, autonomous vehicles, robotic applications. Here some examples of real word application that we found about computer vision :

### Robot vision

Rn 3 dimension robots, we can calculate and determine its exact position, velocity, orientation in the environment, the problem is detecting other objects in the surrounding of it. Using sensors is effective but it works only for certain tasks, where computer vision can give it a global aspect.
For example, if we need to detect an object that we need to pick up with a robotic arm, a camera with known coordination can give a 2D position for our object and with that, we can do the task. we will present the detection process later on in our application.

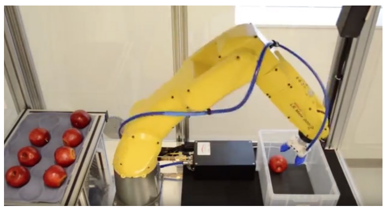


Figure 3.8: Aruit organiser robot

### Autonomous vehicle

Modern cars are using cameras as the main sensors to understand the surrounding environment, they are being used to identify signs, road lines, other cars, traffic lights and even the conditions of the driver, and the main goal for that is reaching full autonomous figure(3.9) and providing better safety features. Many technologies exist to calculate distances between cars such as LIDAR sensors, it is effective but it is positioned in the roof of the car making it less aerodynamic without mention that it cost a fortune, or ultrasonic sensor for a cheaper solution but it have limitations of the range and distance that it can cover so it is used as a basic parking sensor. There is also a different approach used by huge manufacturers such as Tesla using computer vision with better results and more affordable, it is called stereo vision it concentrates on 3D reconstruction of images using 2 cameras that take pictures for the same object in different angles the same way human eyes work Figure(3.10)

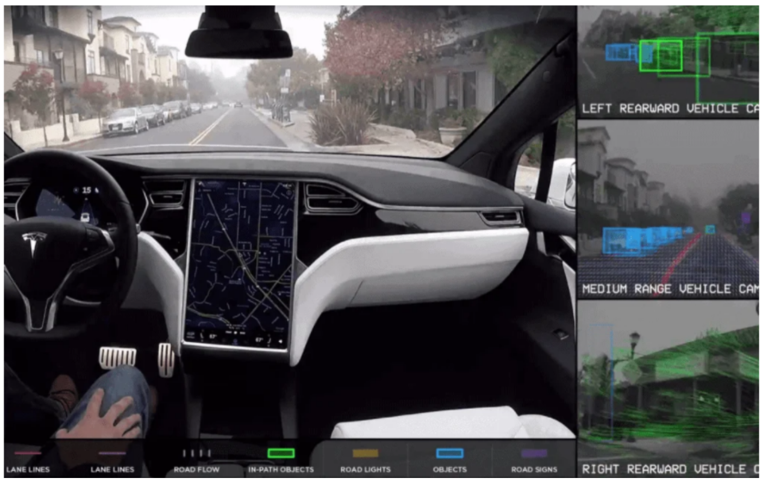


Figure 3.9: Surroundings recognition by camera

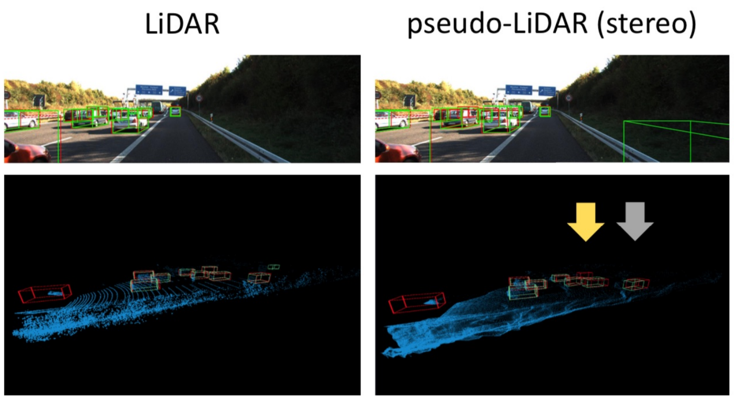


Figure 3.10: LIDAR vs stereo vision

### Medical images

There is a remarkable interest in using artificial intelligence AI for diagnostic analysis of various types of medical images, primarily through convolution neural networks, a type of deep learning technology referred to as “computer vision” [11].

Computer vision can make the process of analyzing the medical images faster and gain time for patients specially for urgent situations here is some examples of it :
Retinal vessel network analysis gives us information about the status of general system and conditions of the eyes. Ophthalmologists can diagnose early sign of vascular burden due to hypertension and diabetes as well as vision threatening retinal diseases like Retinal Artery Occlusion (RAO) and Retinal Vein Occlusion (RVO) from abnormality in vascular structure [10].

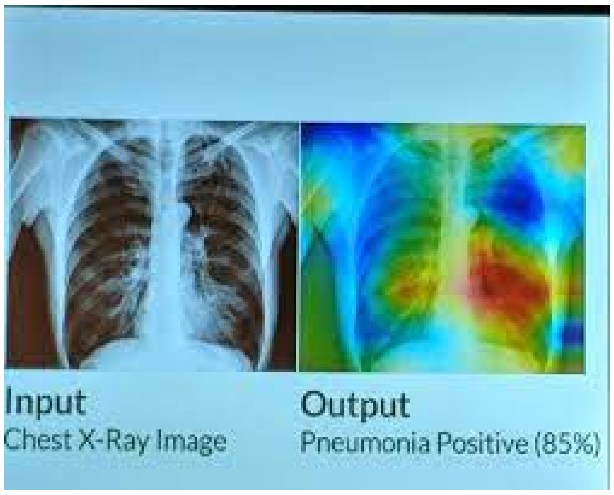


Figure 3.11: Pneumonia detection using computer vision

## 3.4 Conclusion

Computer vision has vast fields of applications and it appears to be the way that the world is going to work with, it seeks to reach the capabilities of what the human eye can do. The applications we talked about above are just a fraction of what this type of knowledge is capable of. In our project, we try to implement some of these techniques in the agricultural domain.

# Chapter 4

# Path planning & fruit counting

## 4.1 Coverage path planning

### 4.1.1 Introduction

Coverage Path Planning (CPP) is the task of determining a path that passes over all points of an area or volume of interest while avoiding obstacles [7].
Autonomous flight will allow us to preform the counting process in massive fields efficiently in cost of time, energy, and simplicity, and also it allows us to preform a sample gathering for analysing a certain area detected by NVDI ”The Normalized Difference Vegetation Index” for example, this application can be preformed using Mission planner directly or by using Boustrophedon algorithm.
In this part we will talk about autonomous flight, optimal path for maximum land coverage at minimum energy consumption, and finally we will talk about how to plan a mission using mission planner and the inconveniences that comes with it, and the solution that we applied these problems.
Next, we will explain the steps we took to create a fruit counting program using Opencv followed by example images for demonstration purposes.

### 4.1.2 Mission planner

Mission planner Figure(4.1) is an open source autopilot platform suited for all different form of unmanned vehicles. It provide us with all the data we need about the drone during the flight such as the GPS location, the orientation, the speed and even if there are some sort of failures, also can be used as a configuration utility or as a dynamic control supplement for your autonomous vehicle, Setup, configure, and tune a vehicle for optimum performance. With Mission planner we can build our path then upload it to the controller, and we can start the mission when ready just by switching to auto mode.

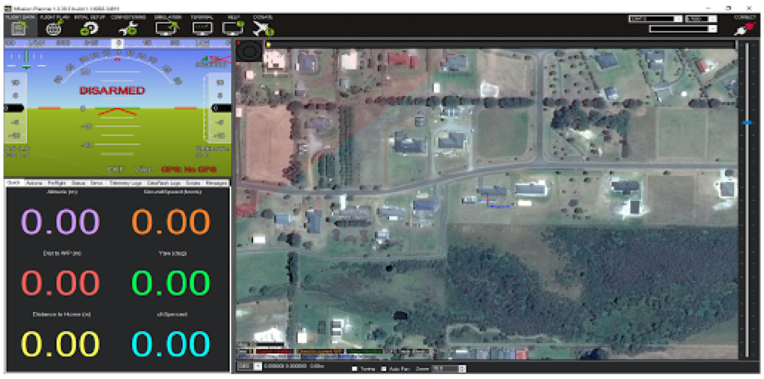
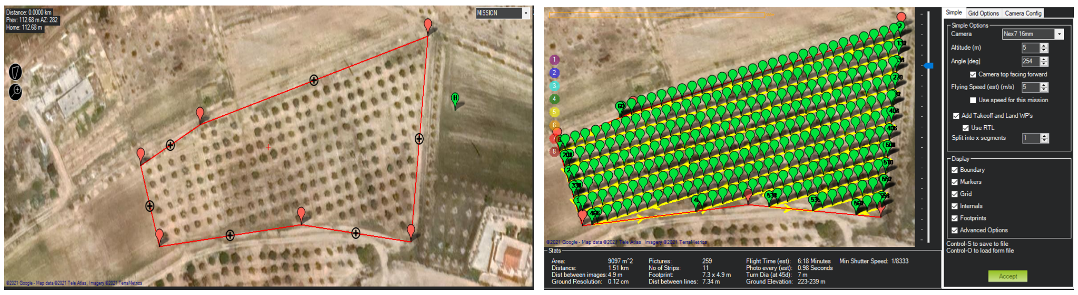


Figure 4.1: Mission planner.

## Mission planning

There are 2 main approaches for planning a mission **direct mission** by setting way-points that the drone will follow or **sampling grid** by setting the area that we need to cover then generate an optimal path that will cover all the zone and satisfy any requirement we are looking for as : altitude, speed, sampling angle, images overlap and return estimations about flight time, generated distance, distance between images following a couple of steps :

- Creating a convex polygon that represents the area, we took an example at figure(4.2).
- Sampling the area, it is an automated process that allow for optimum coverage as needed figure(4.2)


MISSION
Simple
Grid Options
Camera Config
Accept
©2021 Google - Map data ©2021 Tele Atlas, Imagery ©2021 TerraMetrics

```
Simple Options
Camera                      Nex7 16mm
Altitude (m)                5
Angle [deg]                 254
    [✓] Camera top facing forward
Flying Speed (est) (m/s)    5
    [ ] Use speed for this mission
[✓] Add Takeoff and Land WP's
  [✓] Use RTL
Split into x segments       1

Display
[✓] Boundary
[✓] Markers
[✓] Grid
[✓] Internals
[✓] Footprints
[✓] Advanced Options
Control-S to save to file
Control-O to load form file
```

```
Stats
Area:                 9097 m^2     Pictures:            259          Flight Time (est):   6:18 Minutes   Min Shutter Speed:  1/8333
Distance:             1.51 km      No of Strips:        11           Photo every (est):   0.98 Seconds
Dist between images:  4.9 m        Footprint:           7.3 x 4.9 m  Turn Dia (at 45d):   7 m
Ground Resolution:    0.12 cm      Dist between lines:  7.34 m       Ground Elevation:    223-239 m
```

Figure 4.2: Sampling grid

However using google maps implies low image resolution and very low update frequency from 1 to 3 years that means it is not suitable for a precision application.
For that we can generate our own map using images taken by a drone at high altitude resulting in greater image resolution and controllable update rate. With that, we can even preform an NVDI map for field integrity analysis (this project is being worked on by Mouad Dali Yahya and Chihab Eddin Brahim simultaneously). We are going to work with this figure that the other team builds for explaining purposes.

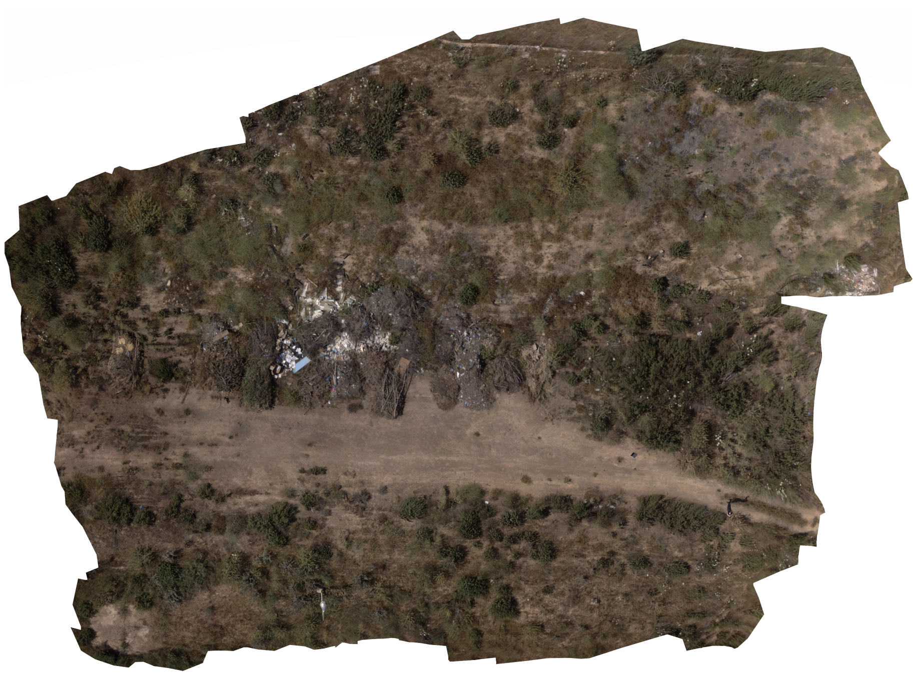


Figure 4.3: Drone map made by Chihab and Mouad

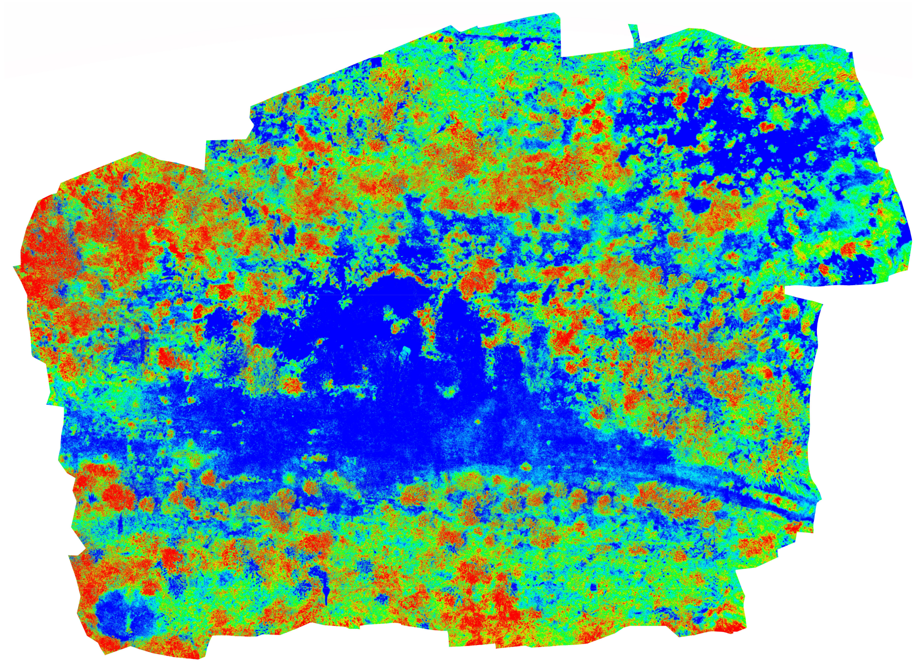


Figure 4.4: NVDI map made by Chihab and Mouad

### 4.1.3 CPP (coverage path planning)

Mission planner can generate the coverage path for different areas without considering the obstacles inside the fly zone, even if we attempted to plan it manually it will be difficult especially because of the low satellite image resolution.
The general problem is modeled as a rural postman problem which has been demonstrated to be NP-Hard (non-deterministic polynomial-time hardness). Our solution relies on dividing the problem into two steps: optimization of the visiting order and optimization of the flight lines orientation. The method is validated through several simulations using

real parameters. In addition, it is fast enough for being implemented on-board [23].

### Application

The algorithm of J. Irving Vasquez-Gomez requires the map as a binary image (the black represents the the zone that we need to cover and the white represents the obstacles that we need to avoid) and in result we will have the way-points as pixel coordinates. We will explain the steps that we followed to obtain a real world coordinates that we can apply in mission planner :
**The coverage zone :** in this part we can determine the obstacle that we need to avoid, using the previous map Figure(4.4) we will create a binary image and we will add some obstacles just for demonstration purposes. The result image represents the coverage map that can be used in the algorithm figure(4.5).

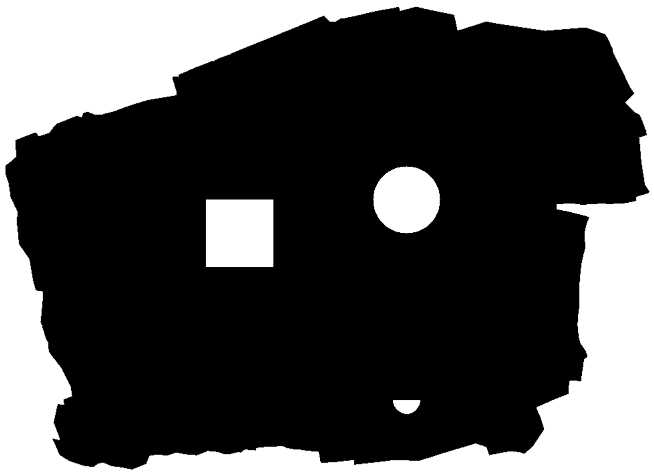


Figure 4.5: Coverage map

**Generating the path :** in this part Adel Mokrane [15] inserts the map in the Boustrophedon algorithm to generate an optimal path, we chose a distance between lines of 60 pixels just for demonstration Figure(4.6)

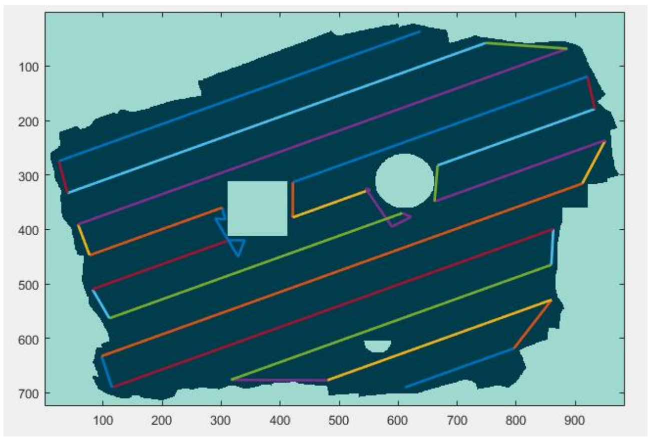


Figure 4.6: Result path

**Pixels to real world coordinates :** For the moment we have only an image without any relation to real world position. The advantage of drone images is that it records the position of every image taken to build the map Figure(4.7) with that we can take 2 points and calibrate the image to real world coordinates since the earth is massive the curvature is basically flat so we can consider that the relation between pixels and degrees linear Y=AX+B and using simple relations we can determine A and B:

| | |
|---|---|
| RCX | Real World Coordinate x axis |
| RCY | Real World Coordinate y axis |
| PCX | Pixel Coordinate x axis |
| PCY | Pixel Coordinate y axis |

$$Ax = \frac{RCX2 - RCX1}{PCX2 - PCX1} \tag{4.1}$$

$$Ay = \frac{RCY2 - RCY1}{PCY2 - PCY1} \tag{4.2}$$

For B we can use one of the 2 points:

$$RCX1 = Ax * PCX1 + Bx \longrightarrow Bx = RCX1 - Ax * PCX1 \tag{4.3}$$

$$RCY1 = Ay * PCY1 + By \longrightarrow By = RCY1 - Ay * PCY1 \tag{4.4}$$

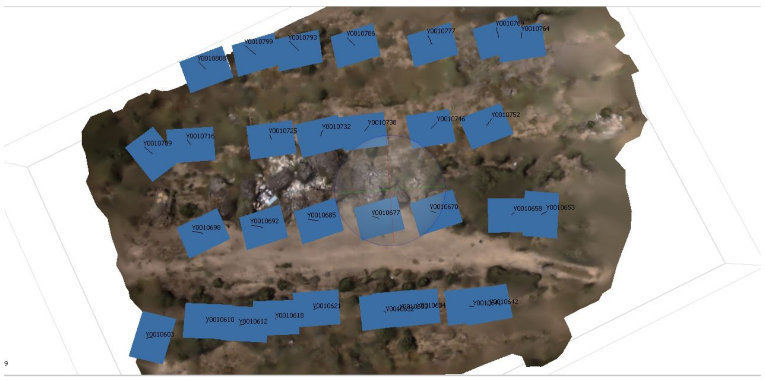


Figure 4.7: Coordinates of the taken images

**Result**

Thanks to the algorithm of Adel Mokrane we are able to generate an optimal path to cover all required spots of an image even with obstacles Figure(4.6) and we can attach this image with real world coordinates using the flight log and the images taken by Chihab and Mouad Figure(4.7) that they used to build the map.

## 4.2 Fruit counting

### 4.2.1 Introduction

Fruit counting is a great way to assess the production and get a general view of the storage capacity needed, in fact there is no sensor that can perform this application better than computer vision. For this application we can follow 2 paths to reach our goal either by using machine learning or using image masks.

### 4.2.2 Machine learning

It works by collecting a database with negative and positive images, negative means the images that don't have the object needed, and positive means the images that contains the object needed.
Next, we need to select the object from the positive images then extract the features that all have in common. One of the simplest ways to do so is haar cascade. It is easy and applied in many applications from face detection to cars, animals, and more.
**Haar cascade classifier :** Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001 [25]. It works by finding an optimal threshold for the images to extract a certain features that define the object. The prediction is going to be more accurate if the object has many features for example in a face we can see the change in the threshold of the eyebrows, the nose, and the lips and more Figure(4.8).

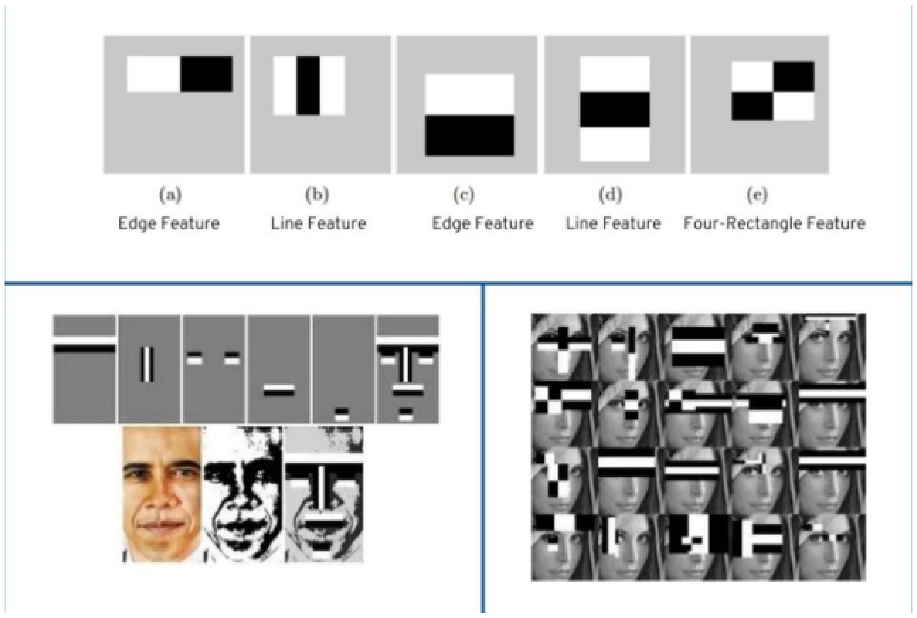


Figure 4.8: Haar cascade classifier.

### 4.2.3 Advantages

- Detecting complex objects.
- Instant detection during the application.
- High detection accuracy.

### 4.2.4 Disadvantages

- Training time, for the best results we need to select our object from around 1000 images.
- Over-fitting, the images forming the data base need to be different from each other in different environments with different lighting situations or the detection will be focused only on the samples in the database.
- Low accuracy in detecting objects with low number of features or data set.

In our application we need to detect fruits, we are going to work with orange in our case. It has simple round shape making haar cascade have low accuracy for that we are going to use color based detection, so we are going to talk about it next on.

### 4.2.5 Color based detection

Now we will explain how to preform object detection based on the color, this method will allow us to extract the object from the environment by saving the region of interest that contains the colors forming the object by applying a **mask**.

#### Mask

A mask allows us to focus only on the region of interest, it requires 2 information :
**Low range** it represents a vector of the lowest intensity of the 3 RGB colors that we are looking for.
**High range** it represents a vector of the highest intensity of the 3 RGB colors that we are looking for.
With that every pixel that satisfies this criteria will result as 1 in the mask and 0 if it doesn't at the end forming a binary image.
Making the mask using the RGB form is not convenient cause in different lighting condition the RGB values change drastically for all channels that means we will build a bigger range resulting in low accuracy for that we use HSV transformation.

#### HSV

The HSV format will allow us to extract the color needed even at different lighting conditions letting us use tighter range to increase accuracy and precision, the results we found are a bit noisy we can eliminate them using low pass filters Figures(4.9)(4.10)

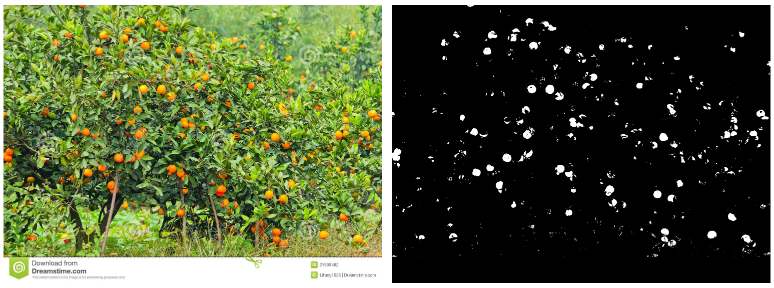


Figure 4.9: Masked image 1

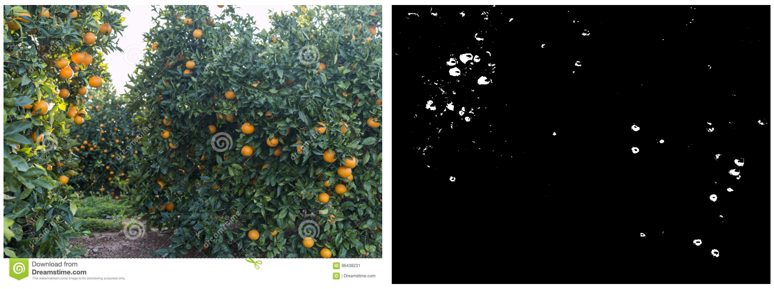


Figure 4.10: Masked image 2

### Opening morphology

Opening morphology is the optimal low pass filter in a binary image to remove white noises without losing the size and form of the objects and keep the main data intact as shown in this Figures(4.11)(4.12), but still we will have another issue some single objects are separated by small spaces and that will effect the counting process.

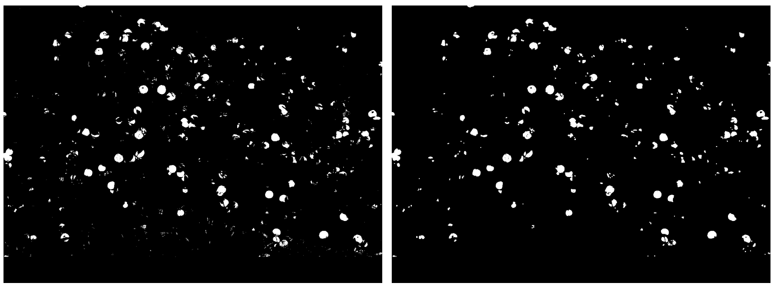


Figure 4.11: Open morphology 1

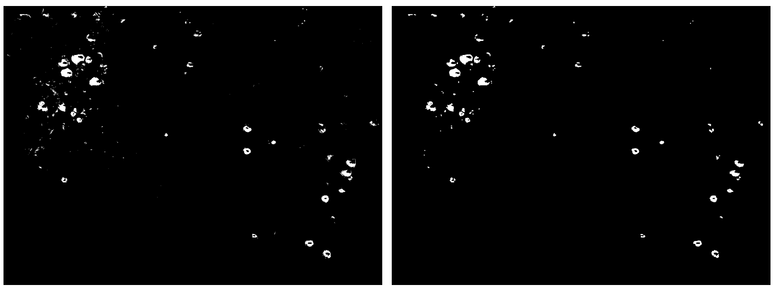


Figure 4.12: Open morphology 2

### Dilation

Dilation will be ideal for attaching separated parts and closing internal spaces but using high number of iterations will result a fusion between objects causing it to become a large single one. The results shown in Figures(4.13)(4.14) represent the effect of this operation, it might look similar but it have a great effect on the next step where we didn't want to cause a fusion between the elements by choosing high iteration number.

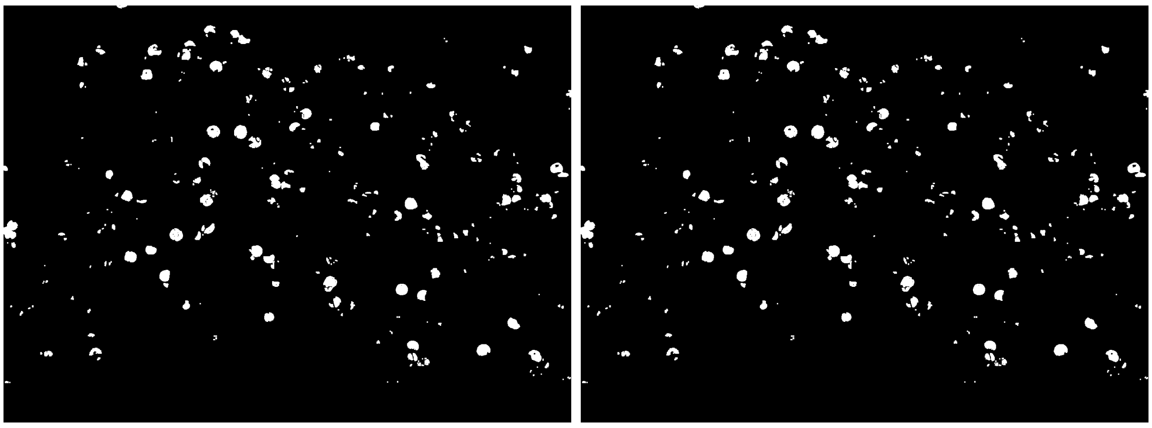


Figure 4.13: Dilation 1

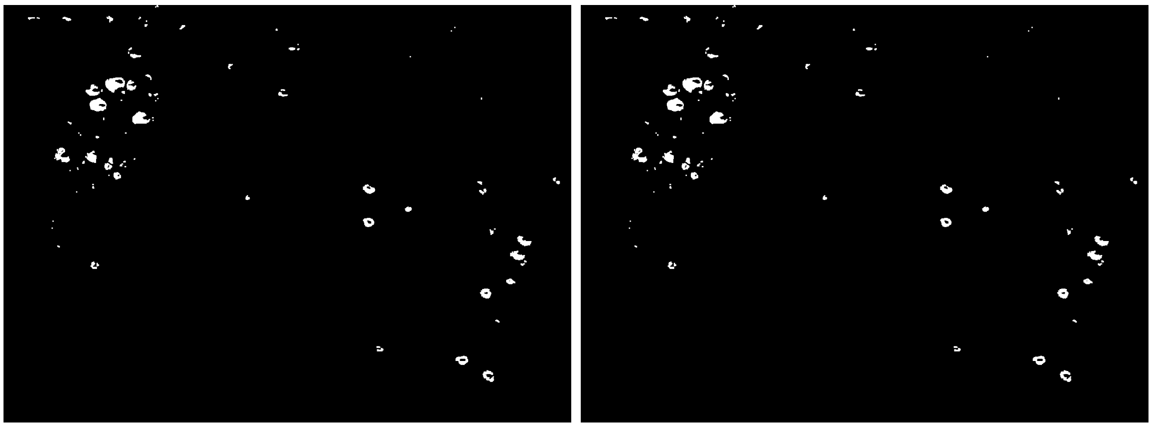


Figure 4.14: Dilation 2

### Contours

The optimal contour method for fruit counting is the SIMPLE approximation method, it uses less memory and it is enough to extract the number of fruits and their position. Applying a bounding rectangle on The contours can give us the coordinates and dimensions of every object we can detect, that will be very helpful to start an autonomous fruit picking process using a UGV fitted with a robotic arm, the final results are shown in the Figure(4.15) we can see that we counted 233 at the first image and 83 at the second image. We can see that in the second image there is some fruits that we couldn't detect, the reason for that is the huge lighting difference between the 2 images, the resolution of the camera and the environment

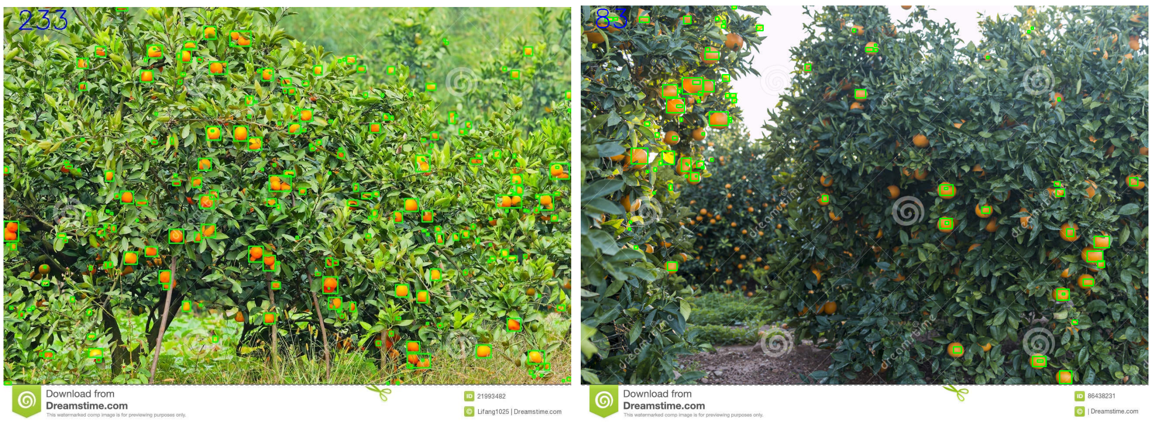
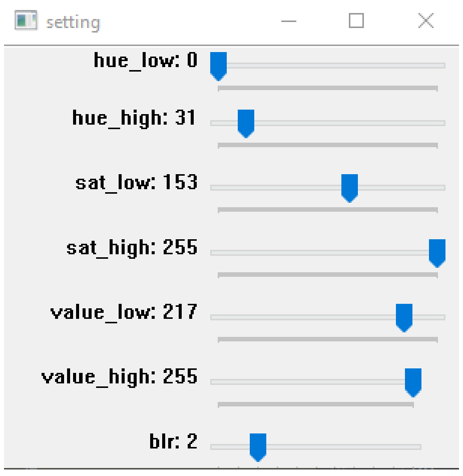


Figure 4.15: The final results

### Opencv track bar

There are many parameters that can optimize the results for a common environment, we can use opencv to build a track-bar that allows us to change the parameters to suit the fruit we are looking for and the optimal filters needed Figure(4.16).


setting

hue_low: 0

hue_high: 31

sat_low: 153

sat_high: 255

value_low: 217

value_high: 255

blr: 2

Figure 4.16: The track bar

## 4.3 Conclusion

As a conclusion, in this chapter we explained how we are able to generate an optimal coverage path using mission planner then added a different approach that allows us to use a map built by drone images where it allows us to generate the optimal path even with obstacles.
Then, we explained how we can perform the fruit counting using computer vision using a step by step examples. Next, we will talk about the hardware we need to realize this application.

# Chapter 5

# Realization and results

In this chapter we will talk about the hardware that Ismat Meslouli picked to build the drone, how thy work, their role, the reason behind these choices, the performance we can expect from it. At the end we will finalize our thesis by discussing the results that we came up with in the coverage path planning and the fruit counting projects also we will mention the difficulties that we encountered.

## 5.1 The drone

### 5.1.1 The controller

The controller is the main element of a drone in sense of intelligent part, it connects the components together. There is many types of controllers to pick from, in our case we are using the pixhawk.
The pixhawk is a great controller platform designed for building any type of unmanned vehicles from submarines to winged plains and in our case a quadrocopter, it is easy to use and have huge community to back it up with a reasonable price tag.

#### Spec functions

**-Processor:**

- 32-bit ARM Cortex M4 core with FPU
- 168 MHz/256 KB RAM/2 MB Flash
- 32-bit fail-safe co-processor

**-Sensors:**

- MPU6000 as main accel and gyro
- ST Micro 16-bit gyroscope

- ST Micro 14-bit accelerometer/compass (magnetometer)
- MEAS barometer

**-Power:**

- Ideal diode controller with automatic fail-over
- Servo rail high-power (7 V) and high-current ready
- All peripheral outputs over-current protected, all inputs ESD protected.

**The connections**

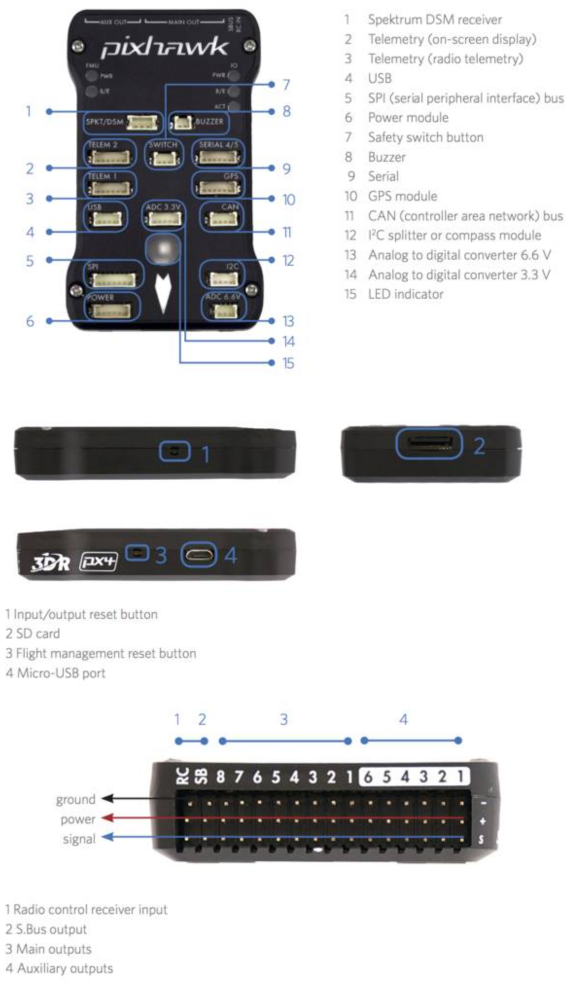

1 Spektrum DSM receiver
2 Telemetry (on-screen display)
3 Telemetry (radio telemetry)
4 USB
5 SPI (serial peripheral interface) bus
6 Power module
7 Safety switch button
8 Buzzer
9 Serial
10 GPS module
11 CAN (controller area network) bus
12 I²C splitter or compass module
13 Analog to digital converter 6.6 V
14 Analog to digital converter 3.3 V
15 LED indicator
1 Input/output reset button
2 SD card
3 Flight management reset button
4 Micro-USB port
ground
power
signal
1 Radio control receiver input
2 S.Bus output
3 Main outputs
4 Auxiliary outputs

Figure 5.1: pixhawk 1 connections

### 5.1.2 Chassis

We are looking for building an agricultural drone with heavy-duty potential for that we need a large size chassis with a rigid frame, like the Tarot IRONMAN-650 Figure(5.2).

It comes with a carbon-fiber frame fitted with foldable tube axes to enhance transportation and save weight without losing rigidity, as a result it comes with a 650mm wheelbase at a weight of 450g. It is a large frame making it very handy in agriculture and heavy duty applications it is also fitted with mounts suitable for any flight controller and other accessories.

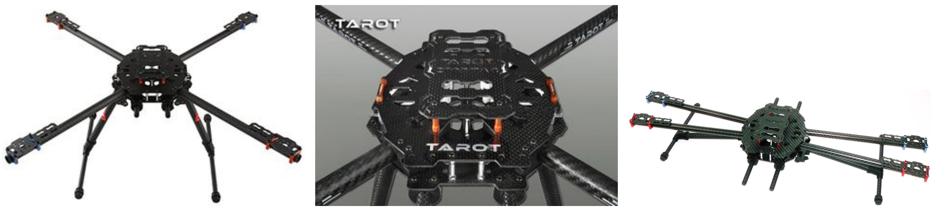


Figure 5.2: Tarot IRONMAN 650

### 5.1.3 Motors

We have a large size frame with all the additions on it, which means that we need high torque motors for it to be able to fly. For that, we are going for a low kv BLDC motors of exactly 800kv. There is another advantage for that, it draws less current than higher kv motors which means it is more efficient and will give us more flight time.

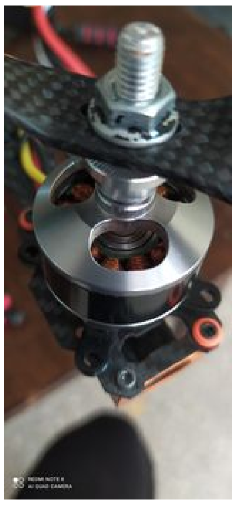


Figure 5.3: 800 kv motors.

---

**ps :** kv actually refers to the RPM of the motor per volt for example in our case 800kv motor at 11.1v can turn at 8880 RPM at no load.

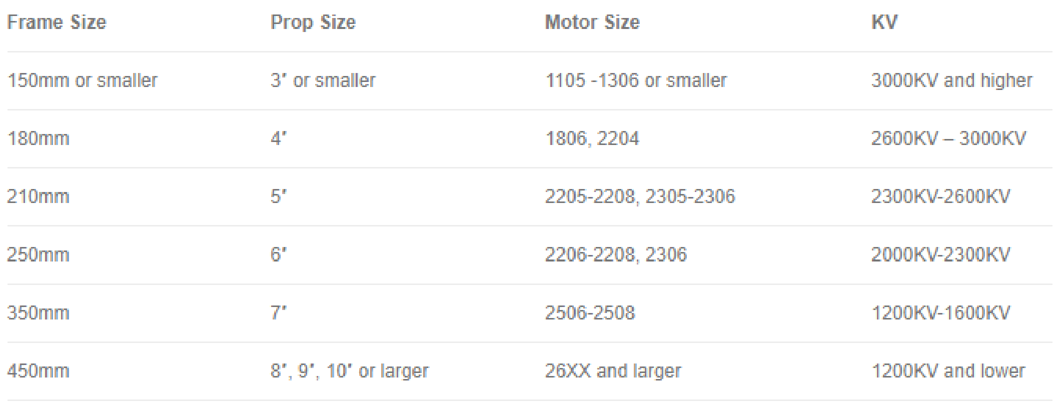
---

### 5.1.4 Propellers

Propellers rotation forces the ear to go down and create thrust to lift the drone up, the dimension of the propeller is inversely proportional to the motor kv rating value, a low

RPM motor will need a longer propeller, and vice versa. Each one has its own proprieties the first one have more torque that gives it heavy-duty potentials and it is perfect for agriculture application, the second one has more agility making it perfect for acrobatic applications or photography. The propellers are also linked to the frame size, we have chosen a large chassis with a 650mm wheelbase and a weight of 450g, which is a bit massive for a drone without talking about other additions such as the battery, the motors them-self and other accessories with it.

Following this table Figure(5.4) we have 800kv motors with a 650mm frame for that we had 3 choices and we have gone with the longest, the "10 inches" (25cm) propellers in carbon fiber Figure(5.5) to enhance rigidity without gaining weight giving it more heavy-duty capabilities for future projects.



| Frame Size | Prop Size | Motor Size | KV |
|---|---|---|---|
| 150mm or smaller | 3' or smaller | 1105 -1306 or smaller | 3000KV and higher |
| 180mm | 4' | 1806, 2204 | 2600KV – 3000KV |
| 210mm | 5' | 2205-2208, 2305-2306 | 2300KV-2600KV |
| 250mm | 6' | 2206-2208, 2306 | 2000KV-2300KV |
| 350mm | 7' | 2506-2508 | 1200KV-1600KV |
| 450mm | 8', 9', 10' or larger | 26XX and larger | 1200KV and lower |

Figure 5.4: propellers,frame,motors relation table.

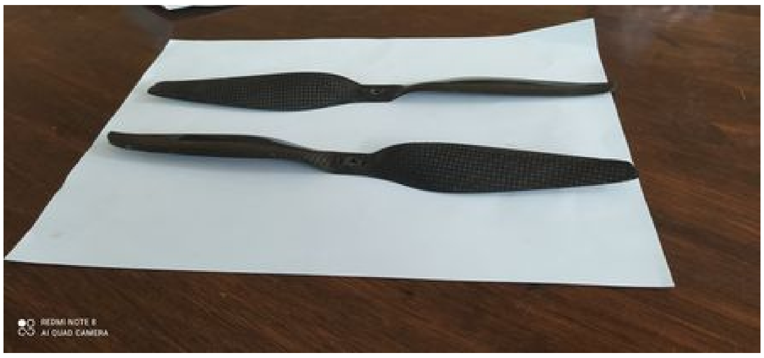


Figure 5.5: 10 inch carbon fiber propellers.

---

**ps :** Propellers come in pairs, built to rotate in the opposite direction of each other, in the quad-copter we use that to eliminate the centrifugal force .

---

### 5.1.5 ECS "electronic speed controller"

The BLDC motors can provide higher RPM and generate less heat and noise than brushed motors because there is no friction by the brushes, but they necessitate working with 3 phase electrical supply.

For that, we are using ESCs, they transform DC to 3 phase AC with controllable frequency. That means, we are able to control the RPM of the motors by changing the PMW ”Pulse Width Modulation” signal sent to the control pins of the ESC in result we will have.

The motor direction is determined by the connection between the ESC and the motor, so if we need to change the direction all what we have to do is swapping between 2 connections as shown in Figure(5.6).

For current rating we are working with a 30A rated speed controllers that means it can withstand a current up to 30A , so it will be more than enough to support any high current draw by the motors without any problem or overheating Figure(5.6).

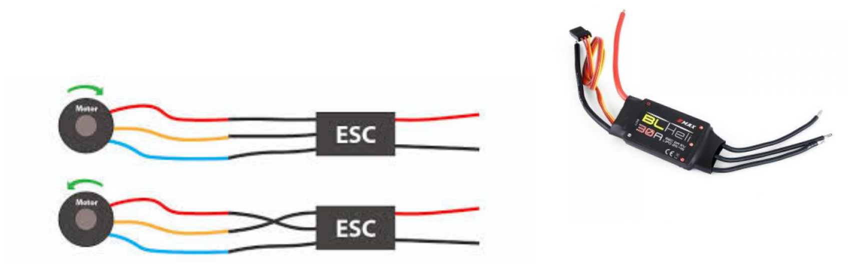


Figure 5.6: 30A rated ESC

### 5.1.6 GPS

Flying a drone manually is possible but it is not recommended, losing the signal from the ground station would mean losing the drone completely especially if it was flying at high altitudes. A GPS can trigger a failsafe and switch to RTL mode ”return to launch” and save the situation, add to that it provides a backup compass to assist the main one of the pixhawk. Moreover, it is an essential tool for planning a flight mission autonomously and allows us to perform our application. Thankfully, the pixhawk is fitted with connection pins specified for the GPS shown in Figure(5.7)

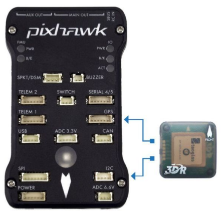


Figure 5.7: GPS-pixhawk connection

### 5.1.7 buzzer & safety switch

There are many accessories that can be added to the pixhawk, a **passive buzzer** is added to differentiate between a problem and another by playing a specific built-in rhythm or playing a satisfying sounds if everything is fine Figure(5.8). A **safety switch** can be added as an essential security feature to disarm the motors when we are working on it even if it was armed by the remote Figure(5.8).

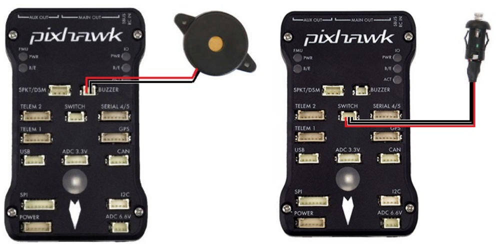


Figure 5.8: buzzer and safety switch-pixhawk connection

### 5.1.8 Communication

We can connect our ground station with the drone using USB cable and accessing the data in ”mission planner” that allows us to monitor the GPS location and the orientations of our drone or if there is any system failing.
But that will not work if the pixhawk is disconnected from computer, for that we use a telemetry device for wireless connection works in 433MHz frequency, with that we will be able to monitor the movement of the drone and it is possible to be done even by phone by downloading mission planner application but it is not recommended cause we found that it can change the settings randomly sometimes.

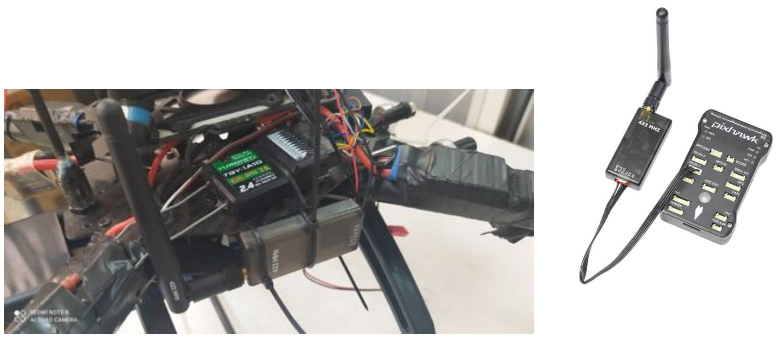


Figure 5.9: Telemetry communication.

### 5.1.9 Remote

Autonomous flight is not simple, for that we need to integrate it with manual control at the start to ensure the development without losing safety. For a remote, we have gone with Turnigy TGY-i10 it has a tactical screen and it comes with 10 channels and every channel represents a variable. There are 4 main channels, and others are auxiliary channels shown in this table Figure(5.10). the $5^{th}$ channel is made for choosing the flight mode, for that we have we will work with :

- AUTO : it requires GPS and it uses a pre-planned mission, it is the main mode for our application.
- RTL : very handy safety mode, it returns the drone to launch position in case of losing the control or it can be triggered automatically in case of a failsafe.
- ALTHOLD : this mode does not need a GPS position and it is very stable, it comes in very handy if there is a problem with the GPS.

Figure 5.10: Remote configuration

| | |
|---|---|
| channel 1 | pitch |
| channel 2 | roll |
| channel 3 | throttle |
| channel 4 | yaw |
| channel 5 | flight modes |
| channel 6 | IOC |
| channel 7 | free channel |
| channel 8 | free channel |

### 5.1.10 Power supply

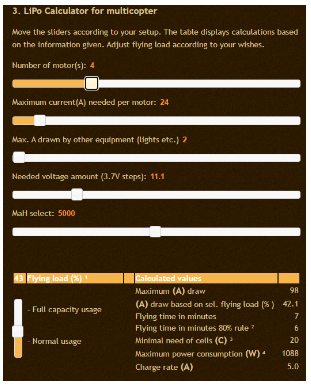
The drone gets its power from a 3S LIPO battery outputting over 12 v when fully charged with a capacity of 5000mAh and a discharge rate of 50c. With that, we can calculate how much current it can produce:

$$5Ah * 50c = 250A \tag{5.1}$$

that is enough current to supply all the motors and the pixhawk even with that heavy chassis Ismat picked.

Figure 5.11: LIPO battery 5000mAh

We can estimate that we can get around 7 min of flight time depending on our application using this calculator:



**3. LiPo Calculator for multicopter**

Move the sliders according to your setup. The table displays calculations based on the information given. Adjust flying load according to your wishes.

Number of motor(s): 4

Maximum current(A) needed per motor: 24

Max. A drawn by other equipment (lights etc.) 2

Needed voltage amount (3.7V steps): 11.1

MaH select: 5000

| 43 | Flying load (%) ¹ | Calculated values | |
|---|---|---|---|
| | - Full capacity usage | Maximum **(A)** draw | 98 |
| | | **(A)** draw based on sel. flying load (% ) | 42.1 |
| | | Flying time in minutes | 7 |
| | | Flying time in minutes 80% rule ² | 6 |
| | - Normal usage | Minimal need of cells **(C)** ³ | 20 |
| | | Maximum power consumption **(W)** ⁴ | 1088 |
| | | Charge rate **(A)** | 5.0 |

### Power distribution

For the motors we have a distribution board it is connected to the battery and the 4 ESCs Figure(5.12).
and for powering the pixhawk we used a voltage regulator with 6 pins output compatible with the pixhawk to measure the voltage and current of the battery and providing 5v output Figure(5.12).

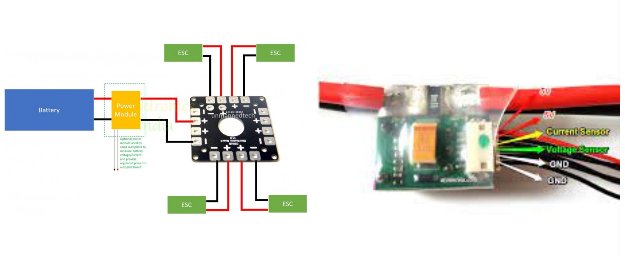

ESC
ESC
Battery
Power Module
Optional power module used by some autopilots to measure battery voltage/current and provide regulated power to autopilot board
unmannedtech
ESC
ESC
5V
5V
Current Sensor
Voltage Sensor
GND
GND

Figure 5.12: Power distribution and Voltage regulator for the pixhawk

## 5.2 Computer vision

In this part we need a single board computer that can be mounted on the drone and analyse the images for that we picked the Odroid XU-4

### 5.2.1 Odroid XU-4

ODROID-XU4 is a new generation of computing device with more powerful, energy-efficient hardware and a smaller form factor. Offering open source support, the board can run various flavors of Linux, including the latest Ubuntu 16.04 and Android 4.4 KitKat and 7.1 Nougat. By implementing the eMMC 5.0, USB 3.0 and Gigabit Ethernet interfaces, the ODROID-XU4 boasts amazing data transfer speeds, a feature that is increasingly required to support advanced processing power on ARM devices. This allows users to truly experience an upgrade in computing, especially with faster booting, web browsing, networking and 3D games.

Figure 5.13: Odroid XU4

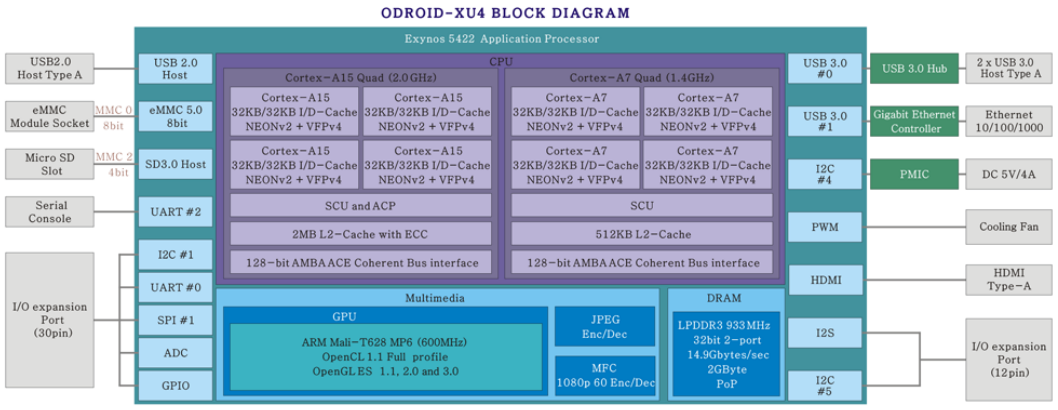


Figure 5.14: Odroid diagram

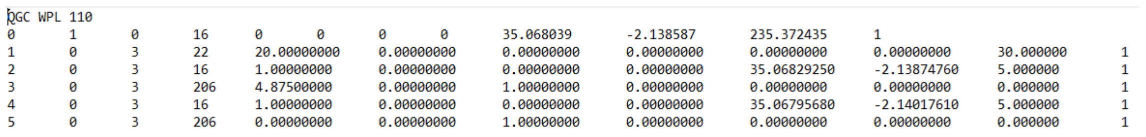
### 5.2.2 Odroid power

The Odroid XU4 requires a 5v DC with current draw up to 4A a LiPo battery can provide this current easily but we need to regulate the voltage, for that we picked an extra battery and fitted it with a voltage regulator

## 5.3 CPP (Coverage Path Planning)

Working on the map manually by taking the coordinates of every way-point from the image to mission planer isn't very practical at all, we needed to come up with a method to create the mission directly from the image and apply it to mission planner. The great thing is that mission planner saves the mission as a text file with ".waypoints" extension Figure(5.15) the general form is :
QGC WPL VERSION
INDEX | CWP | CF | CMND | PAR1 | PAR2 | PAR3 | PAR4 | X | Y | Z | AUTOCONTINUE
**INDEX :** represent the number of the step
**CWD :** Current Way-Point it takes the position of the drone before takeoff set to 1 to activate it is used for "set-home"

**CF :** Frame Coordinate it is connected to the type of the UAV 3 for the quad copter
**CMND :** represent the state command, in our platform we picked the main 4 (16:way-point, 22:takeoff, 206:camera-trigger, 22:return-to-launch)
**PAR1 :** the time delay in seconds
**PAR3 :** trigger stat either 1 or 0
**X,Y,Z :** the coordinates of the waypoint (latitude, longitude, altitude)
**AUTOCONTINUE :** when done with the current step go to next step

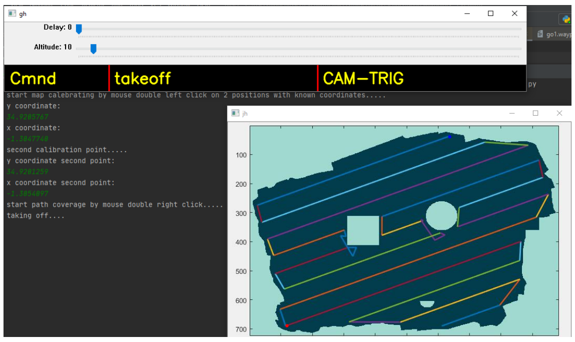


```
QGC WPL 110
0	1	0	16	0	0	0	0	35.068039	-2.138587	235.372435	1
1	0	3	22	20.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	30.000000	1
2	0	3	16	1.00000000	0.00000000	0.00000000	0.00000000	35.06829250	-2.13874760	5.000000	1
3	0	3	206	4.87500000	0.00000000	1.00000000	0.00000000	0.00000000	0.00000000	0.000000	1
4	0	3	16	1.00000000	0.00000000	0.00000000	0.00000000	35.06795680	-2.14017610	5.000000	1
5	0	3	206	0.00000000	0.00000000	1.00000000	0.00000000	0.00000000	0.00000000	0.000000	1
```

Figure 5.15: mission example

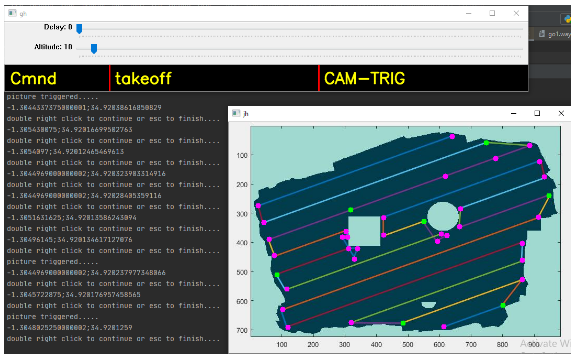
For sake of simplicity, we created a python script that creates a mission similar to mission planner using opencv by following simple steps.

## 5.3.1 Mapping



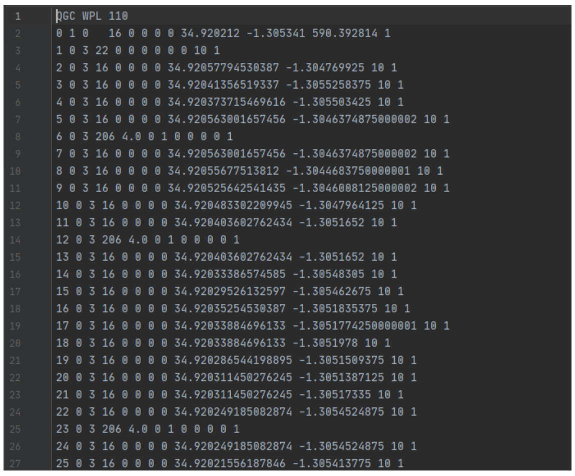
```
start map calebrating by mouse double left click on 2 positions with known coordinates.....
y coordinate:
34.9205767
x coordinate:
-1.3047740
second calibration point.....
y coordinate second point:
34.9201259
x coordinate second point:
-1.3054097
start path coverage by mouse double right click.....
taking off....
```

Figure 5.16: Map calibrating



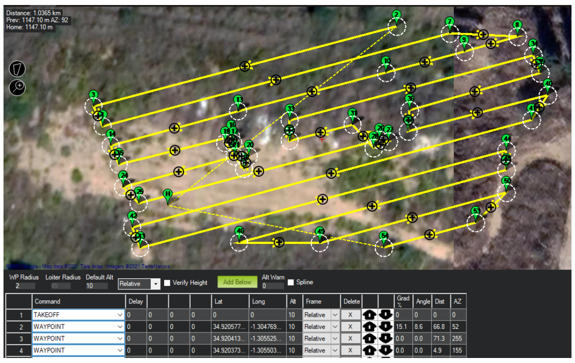
```
picture triggered.....
-1.3044337375000001;34.92038616850829
double right click to continue or esc to finish....
-1.305430075;34.92016699502763
double right click to continue or esc to finish....
-1.3054097;34.92012465469613
double right click to continue or esc to finish....
-1.3044969000000002;34.92032390331491
double right click to continue or esc to finish....
-1.3044969000000002;34.92028405359116
double right click to continue or esc to finish....
-1.3051631625;34.92013586243094
double right click to continue or esc to finish....
-1.30496145;34.920134617127076
double right click to continue or esc to finish....
picture triggered.....
-1.3044969000000002;34.92023797734806
double right click to continue or esc to finish....
-1.3045722875;34.92017695745856
double right click to continue or esc to finish....
picture triggered.....
-1.3048025250000002;34.9201259
double right click to continue or esc to finish....
```

Figure 5.17: Creating waypoints



```
QGC WPL 110
0 1 0   16 0 0 0 0 34.920212 -1.305341 590.392814 1
1 0 3 22 0 0 0 0 0 0 10 1
2 0 3 16 0 0 0 0 34.92057794530387 -1.304769925 10 1
3 0 3 16 0 0 0 0 34.92041356519337 -1.3055258375 10 1
4 0 3 16 0 0 0 0 34.920373715469616 -1.305503425 10 1
5 0 3 16 0 0 0 0 34.920563001657456 -1.3046374875000002 10 1
6 0 3 206 4.0 0 1 0 0 0 0 1
7 0 3 16 0 0 0 0 34.920563001657456 -1.3046374875000002 10 1
8 0 3 16 0 0 0 0 34.92055677513812 -1.3044683750000001 10 1
9 0 3 16 0 0 0 0 34.920525642541435 -1.3046008125000002 10 1
10 0 3 16 0 0 0 0 34.920483302209945 -1.3047964125 10 1
11 0 3 16 0 0 0 0 34.920403602762434 -1.3051652 10 1
12 0 3 206 4.0 0 1 0 0 0 0 1
13 0 3 16 0 0 0 0 34.920403602762434 -1.3051652 10 1
14 0 3 16 0 0 0 0 34.92033386574585 -1.30548305 10 1
15 0 3 16 0 0 0 0 34.92029526132597 -1.305462675 10 1
16 0 3 16 0 0 0 0 34.92035254530387 -1.3051835375 10 1
17 0 3 16 0 0 0 0 34.92033884696133 -1.3051774250000001 10 1
18 0 3 16 0 0 0 0 34.92033884696133 -1.3051978 10 1
19 0 3 16 0 0 0 0 34.920286544198895 -1.3051509375 10 1
20 0 3 16 0 0 0 0 34.920311450276245 -1.3051387125 10 1
21 0 3 16 0 0 0 0 34.920311450276245 -1.30517335 10 1
22 0 3 16 0 0 0 0 34.920249185082874 -1.3054524875 10 1
23 0 3 206 4.0 0 1 0 0 0 0 1
24 0 3 16 0 0 0 0 34.920249185082874 -1.3054524875 10 1
25 0 3 16 0 0 0 0 34.92021556187846 -1.305413775 10 1
```

Figure 5.18: Resulting text file



WP Radius 2 | Loiter Radius 45 | Default Alt 10 | Relative | Verify Height | Add Below | Alt Warn 0 | Spline

| | Command | Delay | | | | Lat | Long | Alt | Frame | Delete | | | Grad % | Angle | Dist | AZ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | TAKEOFF | 0 | 0 | 0 | 0 | 0 | 0 | 10 | Relative | X | | | 0 | 0 | 0 | 0 |
| 2 | WAYPOINT | 0 | 0 | 0 | 0 | 34.920577... | -1.304769... | 10 | Relative | X | | | 15.1 | 8.6 | 66.8 | 52 |
| 3 | WAYPOINT | 0 | 0 | 0 | 0 | 34.920413... | -1.305525... | 10 | Relative | X | | | 0.0 | 0.0 | 71.3 | 255 |
| 4 | WAYPOINT | 0 | 0 | 0 | 0 | 34.920373... | -1.305503... | 10 | Relative | X | | | 0.0 | 0.0 | 4.9 | 155 |

Figure 5.19: Mission planner result

### 5.3.2 results discussion

**Map calibrating :** Figure(5.16) this is the first step by double right mouse click we can select 2 points. Next, the program requires to insert the real world coordinates of these points. Finally, press takeoff after adjusting the required altitude and delay.
**Creating waypoints :** Figure(5.17) using double right click we can create the waypoints we need to cover if we need to take a picture for a certain area press on CAM-TRIG to finish up press ESC and the program will create a return to home sequence, in this example the green dotes represent the image trigger areas.
**Mission planner result:** Figure(5.19) the final result was pretty accurate, it can be used to improve Coverage path planning significantly but unfortunate we couldn't use it in a real farm but we where able to test it in the university.

## 5.4 Fruit counting

For testing the project we decided to count flowers in the university, we choose that because we had 2 choices either painting objects in the usual test zone (the zone that Mouad and Chihab made the map in) or waiting until the university is empty and fly it around a flower zone, it seems to be the best option and we can see the results in the

Figures(5.20)(5.21)(5.22)

## 5.4.1 Counting

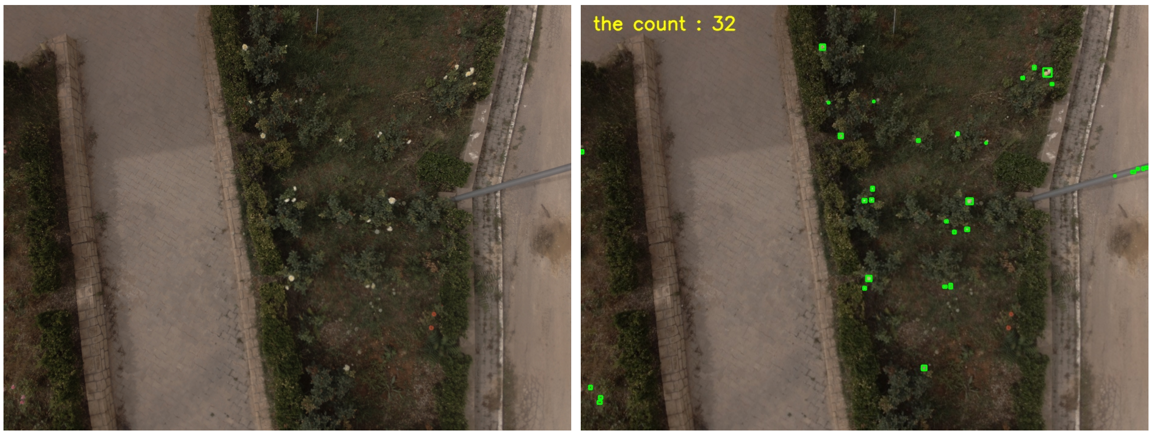


Figure 5.20: Result 1

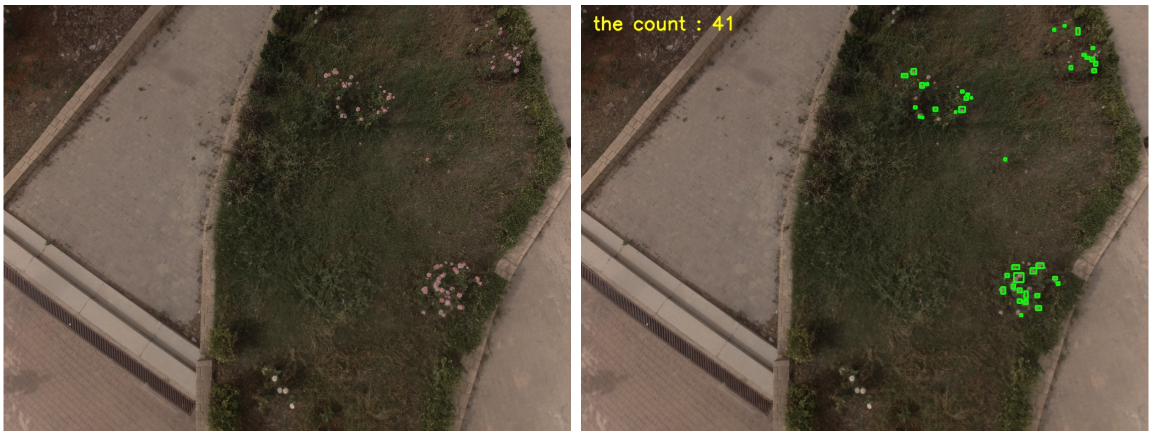


Figure 5.21: Result 2

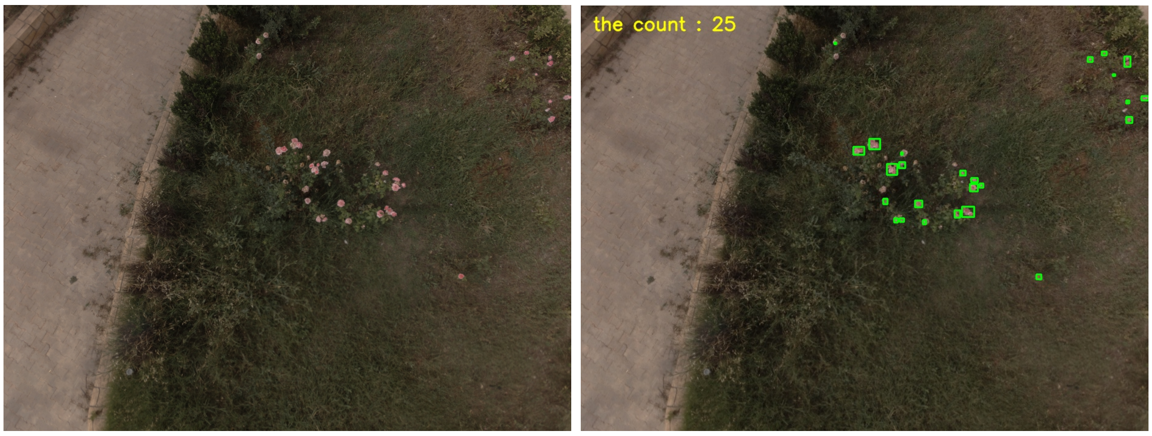


Figure 5.22: Result 3

### 5.4.2 Results discussion

From the results below we can see that there are some small flaws in the prediction, in result 1 we can see some parts of the pole detected as flowers and for result 2 we can see that some flowers undetected, these small misinterpretation are essential to avoid over-fitting, that means we can use it in different environments with the same parameters in result we got a decent estimation that can be used in real world application for large scale fruit counting.

### 5.4.3 Inconveniences

The main issues in building this project are the hardware, either they take long time to reach or costly expensive or hardly accessible.
Even if you are lucky like us and find everything set and ready to fly from previous project and thanks to Laboratoire d'Automatique de Tlemcen, unfortunately we are not able to test it outside the university in a real farming field.

## 5.5 Conclusion

In this chapter we detailed the hardware used to realize this project and the performances it can provide, we also explained how we where able get highly precise accuracy for autonomous drone flight using Adel MOKRANE algorithms instead of using low resolution google map. Finally, we shared the results we got from the fruit counting application and discussed the flaws it came with and why we kept them for and talked about the inconveniences that we came across

# General conclusion

Drone technologies are spreading across the world in various applications and mentioned a small fraction in our introduction and concerned more about precision agriculture unmanned areal vehicles, we explained how the quad-copter dynamics work in a general manner and the physics behind it. Next, we introduced computer vision and the massive fields it can effect from medical to autonomous cars, also we explained briefly some of the algorithms we are going to use in our application. Finally, we shared how we were able to realize this project, now we were able to work with any map image and get better resolution than satellite images also we could determine our own update rate and create an optimal coverage path even with obstacles, we can preform a fruit counting to estimate the possible production all of that in matter of an hour or less. With the help of Allah and everyone involved in this project we were successful to create these applications and Inchallaah it will be a benchmark for bigger projects to come by the next generations. Drones are helpful means that take great importance in many applications particularly in environment and precision agriculture where there still be many progression. Fruit counting is a small application that can be optimized using the drone made map instead of google map, there is huge application range for it for example we can build an optimal trajectory for a low altitude that preform a task of spraying planets or using the NVDI map, locate the low vegetation index zone and take samples, or changing lanes to fire-fighting and create a drone that localize potential wildfire starting since there is no limit for imagination then there is no limit for drone applications.

# Bibliography

[1] W. Azhar and K. Ejaz. “Legality of US Drone Strikes in Pakistan”. In: ().

[2] S. Bouabdallah. “design and control of quadrotors with application to autonomous flying.” In: 15 (2007), p. 127.

[3] Z. Chekakta and A. Zoubiri. “Conception, Modélisation et Commande d’un UAV de type Quadrirotor”. In: 20 (2016), p. 104.

[4] Timothy F Cootes, Cristopher J Taylor, et al. *Statistical models of appearance for computer vision.* 2004.

[5] G. Danuser. “Computer vision in cell biology”. In: *Cell* 147.5 (2011), pp. 973–978.

[6] A. Frenot, A. Gossmann, and R. Guillerm. “STABILISATION D’UN QUADRIROTOR”. In: 21 (2005), p. 93.

[7] E. Galceran and M. Carreras. “A survey on coverage path planning for robotics”. In: *Robotics and Autonomous systems* 61.12 (2013), pp. 1258–1276.

[8] et M. Tadjine H. Bouadi M. Bouchoucha. “Modelling and Stabilizing Control Laws Design Based on Sliding Mode for an UAV Type-Quadrotor”. In: 13 (2007), p. 7.

[9] et M. Tadjine H. Bouadi M. Bouchoucha. “Sliding Mode Control Based on Backstepping Approach for an UAV Type-Quadrotor”. In: 14 (2007), p. 7.

[10] T. Iqbal and H. Ali. “Generative adversarial network for medical images (MI-GAN)”. In: *Journal of medical systems* 42.11 (2018), pp. 1–11.

[11] Dong Wook Kim et al. “Design characteristics of studies reporting the performance of artificial intelligence algorithms for diagnostic analysis of medical images: results from recently published papers”. In: *Korean journal of radiology* 20.3 (2019), p. 405.

[12] K. Laib and D. Maamria. “Commande d’un QUADRIROTOR”. In: 19 (2011), p. 104.

[13] J Gordon Leishman. “The breguet-richet quad-rotor helicopter of 1907”. In: *Vertiflite* 47.3 (2002), pp. 58–60.

[14] I. Meslouli and R. Mesli. “Réalisation et pilotage d’un drone à quatre rotors”. In: 23 (2017), p. 85.

[15] A. Mokrane, A. Braham, and B. Cherki. “UAV Path Planning Based on Dynamic Programming Algorithm On Photogrammetric DEMs”. In: *2020 International Conference on Electrical Engineering (ICEE)*. IEEE. 2020, pp. 1–5.

[16] K. Pulli et al. “Real-time computer vision with OpenCV”. In: *Communications of the ACM* 55.6 (2012), pp. 61–69.

[17] V. Puri, A. Nayyar, and L. Raja. “Agriculture drones: A modern breakthrough in precision agriculture”. In: *Journal of Statistics and Management Systems* 20.4 (2017), pp. 507–518.

[18] F. Mohamedi N. Saci. “simulation-dun-drone-sous-matlab”. In: 17 (2015), p. 90.

[19] Z. Satla. “Contribution à la modélisation et à la commande d’un drone miniature”. In: 16 (2017), p. 113.

[20] C. Sedini and N. Cherigui. “Conception et commande d’un quadrotor UAV à base d’Arduino”. In: 22 (2019), p. 94.

[21] A. Torche and F. Aftis. “simulation dymanique et realisation d’un drone quadrirotor a base de carte autopilote PIXHAWK”. In: 18 (2019), p. 93.

[22] G. Udeanu, A. Dobrescu, and M. Oltean. “Unmanned aerial vehicle in military operations”. In: *Scientific research and education in the air force* 18.1 (2016), pp. 199–206.

[23] Juan I. Vasquez-Gomez, Juan C. Herrera-Lozada, and M. Olguin-Carbajal. “Coverage path planning for surveying disjoint areas”. In: *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2018, pp. 899–904.

[24] F. Veroustraete. “The rise of the drones in agriculture”. In: *EC agriculture* 2.2 (2015), pp. 325–327.

[25] P. Viola and M. Jones. “Rapid object detection using a boosted cascade of simple features”. In: *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*. Vol. 1. IEEE. 2001, pp. I–I.

# Appendix A

# Algorithms

## A.1 Mission planning

```
Result: Map calibrating
m=0
if Left mouse double click on the map and m==0 then
    save x,y as x1, y1 position
    enter real world coordinates xr1, yr1
    m=1
end
if Left mouse double click one the map and m==1 then
    save x,y as x2, y2 position
    enter real world coordinates xr2, yr2
    m=2
    Ax=(xr2-xr1)/(x2-x1)
    Ay=(yr2-yr1)/(y2-y1)
    Bx=-(Ax*x1)+xr1 By=-(Ay*y1)+yr1
end
```

```
Result: Path planning
step=0
if Right mouse double click on the map then
    xr = Ax*x+Bx
    yr = Ay*y+By
    input(altitude)
    input(delay)
    command=16
    trigger=0
    create text line
    step++
end
```

```
   Result: Taking-off and Image trigger
1  create track bare for altitude create track bar for delay time if Right mouse
     double click on takeoff then
2      read altitude from track bar
3      create takeoff text line
4  end
5  if Right mouse double click on CAM-TRIG then
6      command=206
7      Delay=2
8      trigger=1
9      create text line
10     command=16
11     Delay=0
12     trigger=0
13     create text line
14 end
```

## A.2 HSV

**Algorithm 1:** HSV channel 1

```
Result: Hue channel [0:360] "H"
for [X,Y] pixel in image do
    [R;G;B] = RGB
    max = max(R;G;B)
    min = min(R;G;B)
    if max == R then
        H = 60*(G-B)/(max-min)
    end
    if max == G then
        H = 60*(2+(B-R)/(max-min))
    end
    else
        H = 60*(4+(R-G)/(max-min))
    end
    if H¡0 then
        H=360-H
    end
end
```

**Algorithm 2:** HSV channel 2 "S"

```
Result: Saturation channel [0:1]
for [X,Y] pixel in image do
    [R;G;B] = RGB
    max = max(R;G;B)
    min = min(R;G;B)
    S = (max-min)/max
end
```

**Algorithm 3:** HSV channel 3 "V"

```
Result: Value channel [0:1]
for [X,Y] pixel in image do
    [R;G;B] = RGB
    max = max(R;G;B)
    V = max
end
```

## A.3 Detection

**Algorithm 4:** Object detection

```
Result: List of contours
while True do
    Read signal from GPIO
    if signal is high then
        take image
        for [X, Y] pixel in image do
            RGBtoHSV (algorithms(1,2,3)
            low H,S,V Range
            high H,S,V Range
            if $H,S,V \leq high$ & $H,S,V \geq low$ then
                output[X, Y]=1
            else
                output[X, Y]=0
            end
            Contour(output)
            draw rectangle around object
        end
    end
end
```