Software Requirements Specification

for

Autonomous Quadcopter

Prepared by:

Lakshya Sharma (101683039)

Ishaan Kohli (101503097)

Ishita Sharma (101503100)

Gaurav Dua (101503072)



Thapar University

May, 2018

Table of Contents

Table of Contents ii

1. Analysis …………………………………………………………………………………………1

2.1. Introduction 1

2.1.1 Purpose 1

2.1.2 Document Conventions 1

2.1.3 Intended Audience and Reading Suggestions 1

2.1.4 Product Scope 1

2.1.5 References 1

2.2. Overall Description 2

2.2.1 Product Perspective 2

2.2.2 Product Functions 2

2.2.3 User Classes and Characteristics 2

2.2.4 Operating Environment 2

2.2.5 Design and Implementation Constraints 2

2.2.6 User Documentation 2

2.2.7 Assumptions and Dependencies 3

2.3. External Interface Requirements 3

2.3.1 User Interfaces 3

2.3.2 Hardware Interfaces 3

2.3.3 Software Interfaces 3

2.3.4 Communications Interfaces 3

2.4. Other Nonfunctional Requirements 4

2.4.1 Performance Requirements 4

2.4.2 Safety Requirements 5

2.4.3 Security Requirements 5

2.5.  **Feasibility………………………………………………………..…………………………..6**

2.5.1 Operational Feasibility………………………………………..……….………………………….6

2.5.2 Technical Feasibility...………………………………………..……….………………………….6

2.5.3 Economical Feasibility……………………………………….……….………………………….6

2.5.4 Legal Feasibility……..………………………………………………….…..…………………….6

2.6. Other Requirements 5

**3. Assumptions and Constraints ………………………………………………………………...7**

**4. Cost Analysis ………………...………………………………………………………………...7**

**5. Detailed Architecture ……….………………………………………………………………...7**

Appendix A: Analysis Models 5

**1 Analysis**

1.1 Basic Introduction

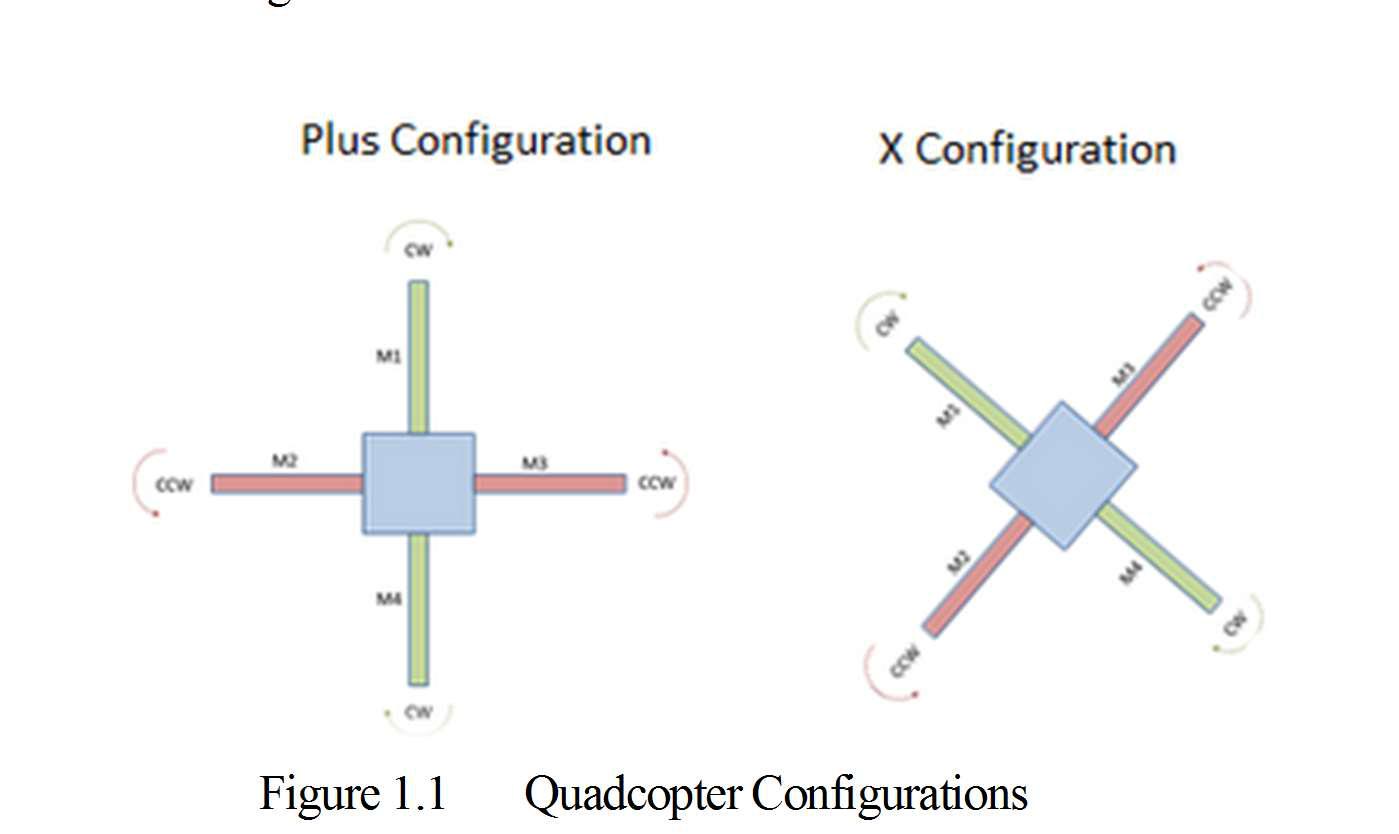
Over the last few years we have seen a massive growth in the manufacture and sales of remote control airborne vehicles known as Quadcopters. These Unmanned Aerial Vehicles have four arms and fixed pitch propellers which are set in an X or + configuration with X being the preferred configuration.

Figure 1.1 Quadcopter Configurations

They are sometimes referred to as Drones, Quadrotors or Quadrocopters. In the standard format two propellers will spin in a clockwise direction with the other two spinning in an anticlockwise direction allowing the craft to vertically ascend, hover in the air and fly in a designated direction. The Quadcopter is a simple format with very few moving parts and has rapidly become a favorite vehicle for remote control enthusiasts and is widely being used as an effective Aerial photographic platform. A large majority of the Quadcopters were originally built by hobbyists who understood the simplicity of the vehicle. By adding four motors and four propellers to a lightweight frame constructed of light wood, carbon fiber, or fiber glass then connecting it to a remote control transmitter via a small control board fitted with a gyroscopic stabilization system and connected to a Lipo battery these craft were relatively simple to construct. Experimentation has led to the configuration of variations of the Quadcopter by using different amounts of arms we have seen Tricopters, Hex copters and Octocopters (with eight arms).

The rapid advances in computing power, the efficiency of the coreless or brushless motors, smaller microprocessors the development of batteries and gyroscopic and accelerometer technology has all led to a proliferation of Quadcopter designs. The first Quadcopters were not designed for acrobatic flight as the development was concentrated on simple stable flight patterns but now this has all changed. Micro and even Nano Quadcopters are being produced mainly in China that can perform intricate aerobatic moves, flips and barrel rolls that years ago would have been unthinkable. Chinese companies like Hubsan have made tiny Nano Quadcopters. Quadcopters differ from conventional helicopters which use rotors which are able to vary the pitch of their blades dynamically as they move around the rotor hub. In the early days of flight, quadcopters (then referred to as 'quadrotors') were seen as possible solutions to some of the persistent problems in vertical flight; torque-induced control issues (as well as efficiency issues originating from the tail rotor, which generates no useful lift) can be eliminated by counter-rotation and the relatively short blades are much easier to construct.

1.2 About the Project

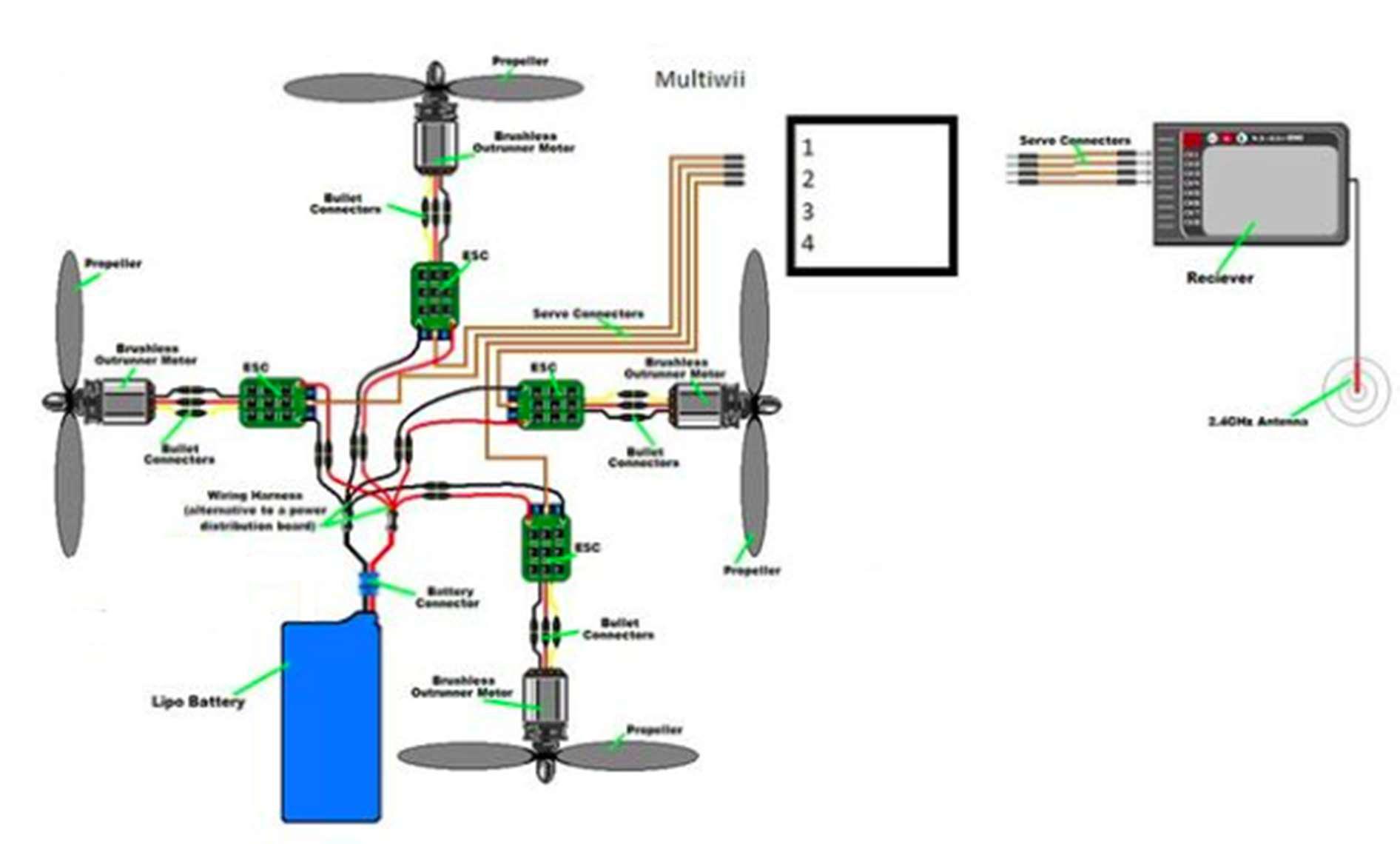
Quadcopter, also known as quadrotor helicopter or quadrotor, is a multirotor helicopter that is lifted and propelled by four rotors. Quadcopters are classified as rotorcraft, as opposed to fixed-wing aircraft, because their lift is generated by a set of rotors. In a quadcopter, two of the propellers spin in one direction (clockwise) and the other two spin the opposite direction (counterclockwise) and this enables the machine to hover in a stable formation.

Firstly the motors which we used have an obvious purpose: to spin the propellers. Motors are rated by kilovolts, the higher the kV rating, the faster the motor spins at a constant voltage. Next the Electric Speed controller or ESC, is what tells the motors how fast to spin at any given time. We need four ESCs for a quadcopter, one connected to each motor. The ESCs are then connected directly to the battery through either a wiring harness or power distribution board. Many ESC1s come with a built in battery eliminator circuit (BEC), which allows you to power things like your flight control board and radio receiver without connecting them directly to the battery. Because the motors on a quadcopter must all spin at precise speeds to achieve accurate flight, the ESC is very important.

Our Quadcopter uses four propellers, each controlled by its own motor and electronic speed controller. Using accelerometers we are able to measure the angle of the Quadcopter in terms of X<Y and Z and accordingly adjust the RPM of each motor in order to self -stabilize itself. the Quadcopter platform provides stability as a result of the counter rotating motors.

For Hovering over the skies the flight controller which is used is the ‘brain’ of the quadcopter. It houses the sensors such as gyroscopes and accelerometers that determine how fast each of the quadcopter’s motors spin. Its purpose is to stabilize the aircraft during flight and to do this, it takes signals from on-board gyroscopes (roll, pitch and yaw) and passes these signals to the Atmel644PA processor, which in-turn processes signals according the users selected firmware (e.g. Quadcopter) and passes the control signals to the installed Electronic Speed Controllers (ESCs) and the combination of these signals instructs the ESCs to make fine adjustments to the motors rotational speeds which in-turn stabilizes the craft. It can trace any path or can follow a GPS signal live by avoiding obstacles in the way by using sensor to detect objects realtime.

1.3 Block Diagram



3.1 Principle of Operation

Frame principle: Frame is the structure that holds all the components together. The Frame should be rigid, and be able to minimize the vibrations coming from the motors. Quadcopter frame consists of two to three parts which don’t necessarily have to be of the same material:

• The center plate where the electronics are mounted • Four arms mounted to the center plate

• Four motor brackets connecting the motors to the end of the arms

Most available materials for the frame are:

• Carbon Fiber

• Aluminum

• Wood, such as Plywood or MDF (Medium-density fiberboard)

Carbon fiber is most rigid and vibration absorbent out of the three materials but also the most expensive.

Hollow aluminum square rails are the most popular for the Quadcopters’ arms due to its relatively light weight, rigidness and affordability. However aluminum could suffer from motor vibrations, as the damping effect is not as good as carbon fiber. In cases of severe vibration problem, it could mess up sensor readings.

Wood board such as MDF plates could be cut out for the arms as they are better at absorbing the vibrations than aluminum. Unfortunately the wood is not a very rigid material and can break easily in Quadcopter crashes.

As for arm length, the term “motor-to-motor distance” is sometimes used, meaning the distance between the centers of one motor to that of another motor of the same arm in the Quadcopter terminology. The motor to motor distance usually depends on the diameter of the propellers. To make you have enough space between the propellers and they don’t get caught by each other

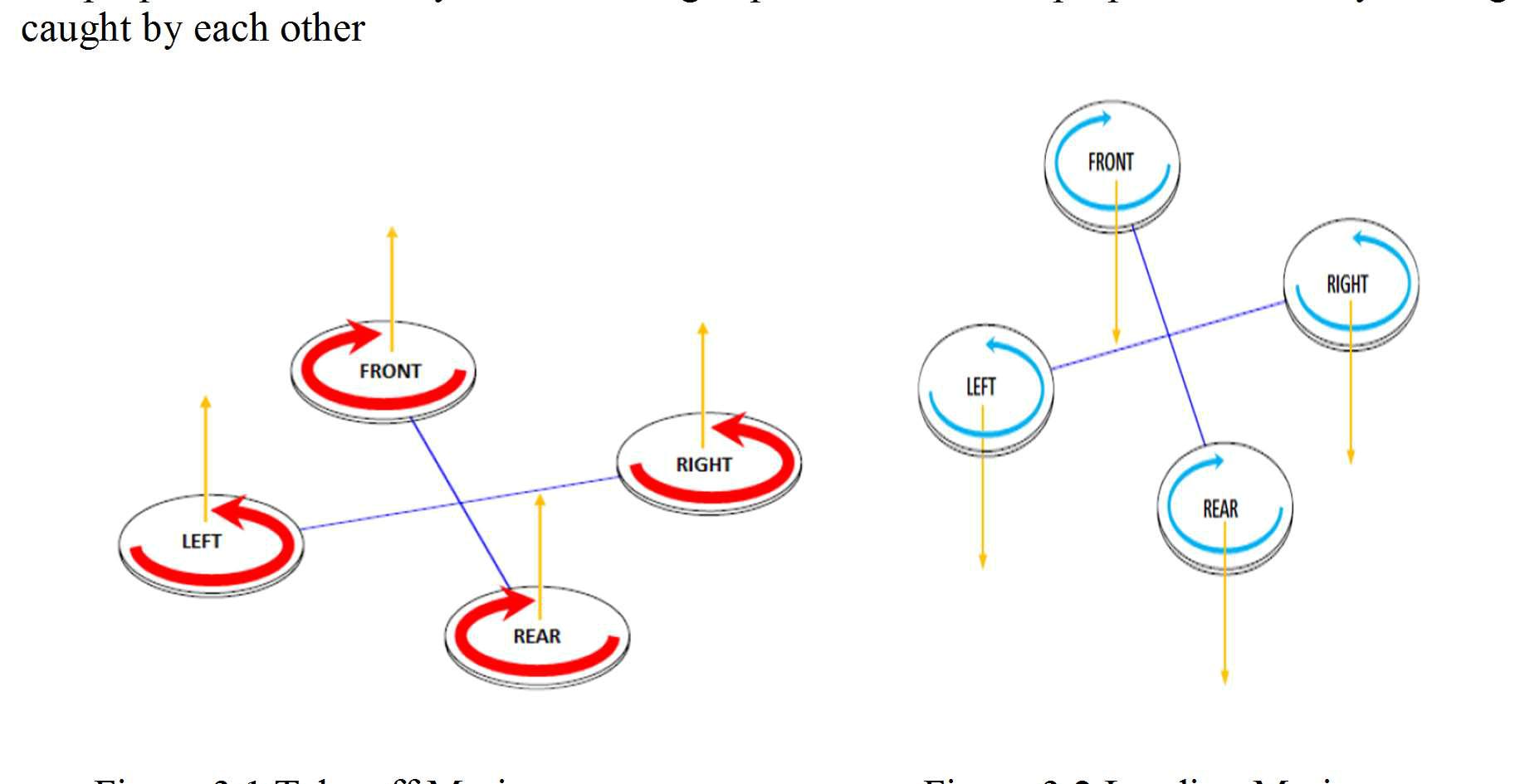


Figure 3.1 Take off Motion Figure 3.2 Landing Motion

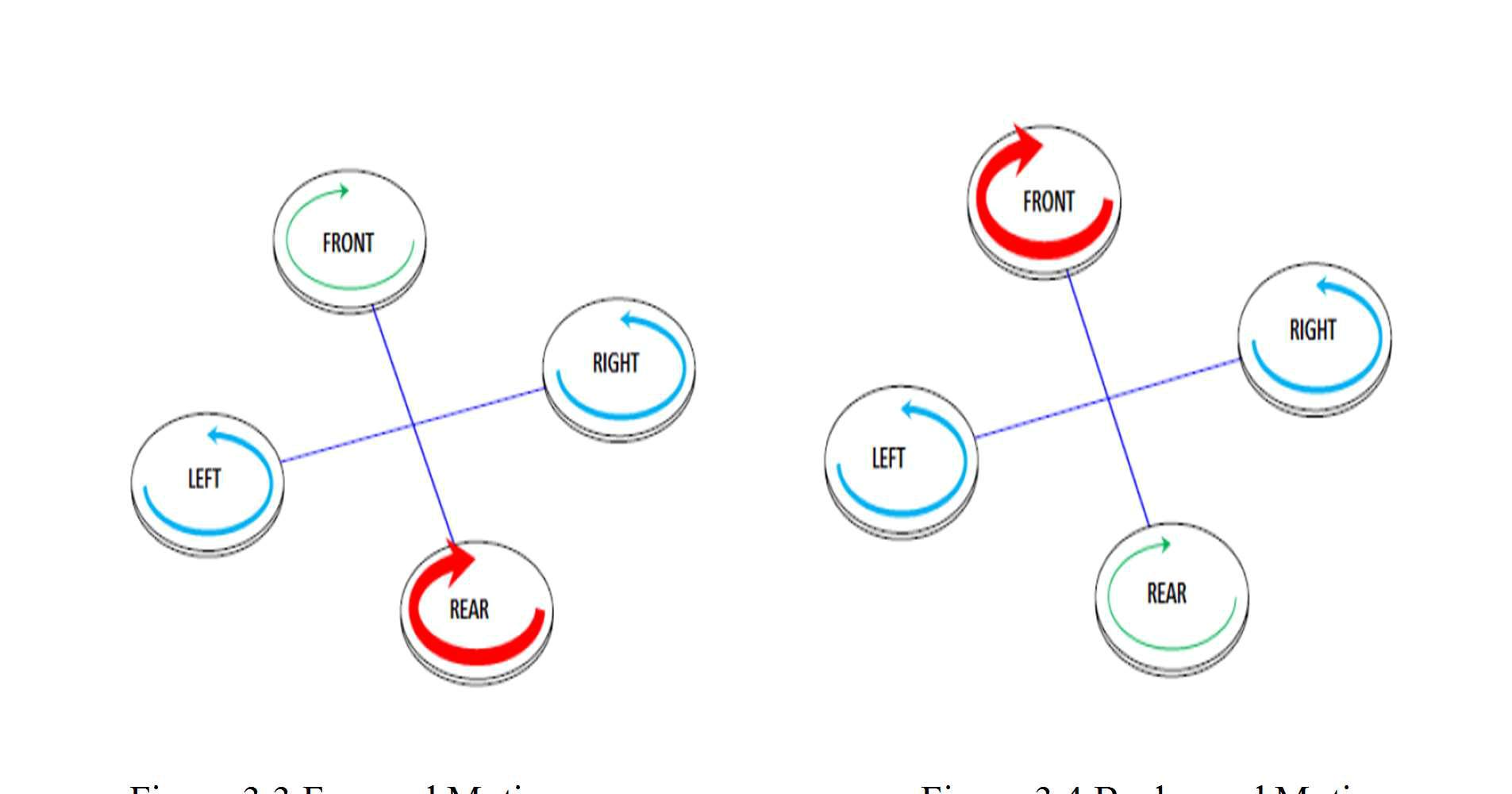


Figure 3.3 Forward Motion Figure 3.4 Back ward Motion

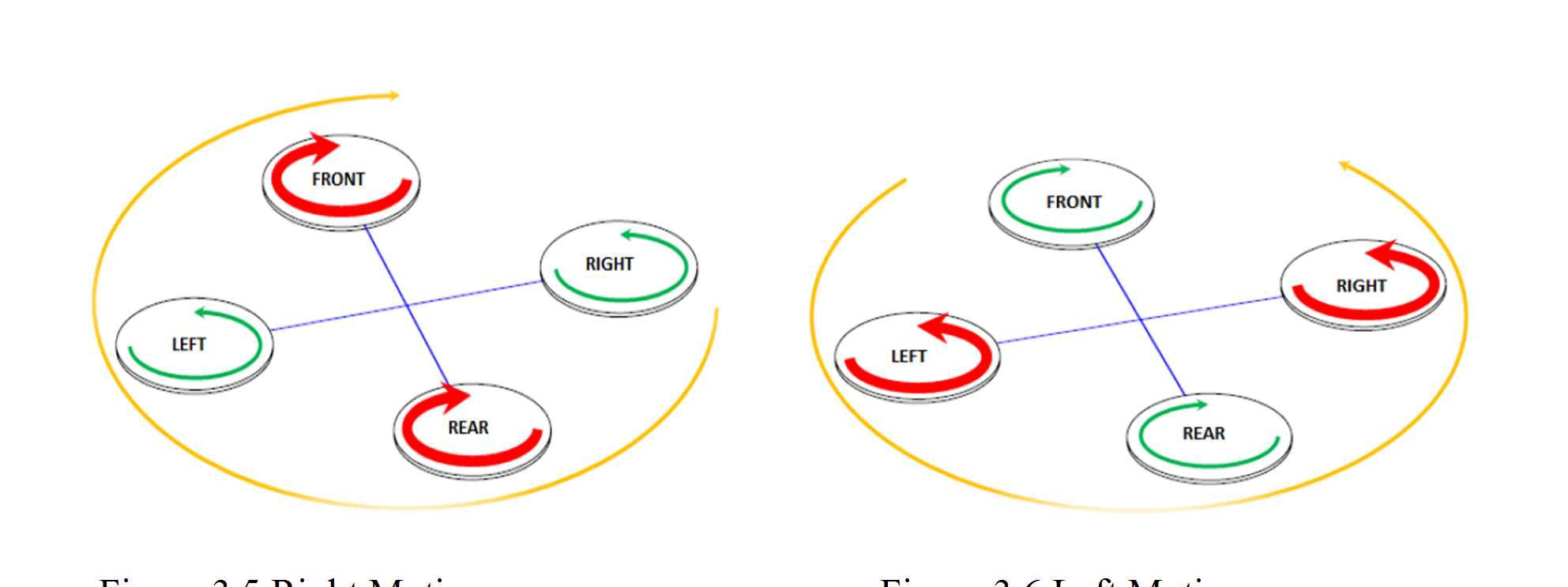


Figure 3.5 Right Motion Figure 3.6 Left Motion

3.1.1 Flying Principle

A propeller is a type of fan that transmits power by converting motion into thrust. Propeller dynamics can be modelled by both Bernoulli’s principle and Newton’s third law.

Principle and Working

The principle and working of a propeller is based on Bernoulli’s Principle and Newton’s Third Law Bernoulli’s principle states that for an inviscid flow, an increase in the speed of the fluid occurs simultaneously with a decrease in pressure or a decrease in the fluid’s potential energy. Newton’s third law states that every action has an equal and opposite reaction.

An aero foil of a propeller is shaped so that air flows faster over the top than under the bottom. There is, therefore, a greater pressure below the aero foil than above it. This difference in pressure produces the lift. Lift coefficient is a dimensionless coefficient that relates the lift generated by an aerodynamic body such as a wing or complete aircraft, the dynamic pressure of the fluid flow around the body, and a reference area associated with the body.

3.1.2 Mechanism

Quadcopter can be described as a small vehicle with four propellers attached to the root located at the cross frame. This aim for fixed rotors is used to control the vehicle motion. The speeds of these four rotors are independent. By independent pitch, roll and yaw attitude of the vehicle can be controlled easily. Pitch, roll and yaw attitude of Quadcopter.

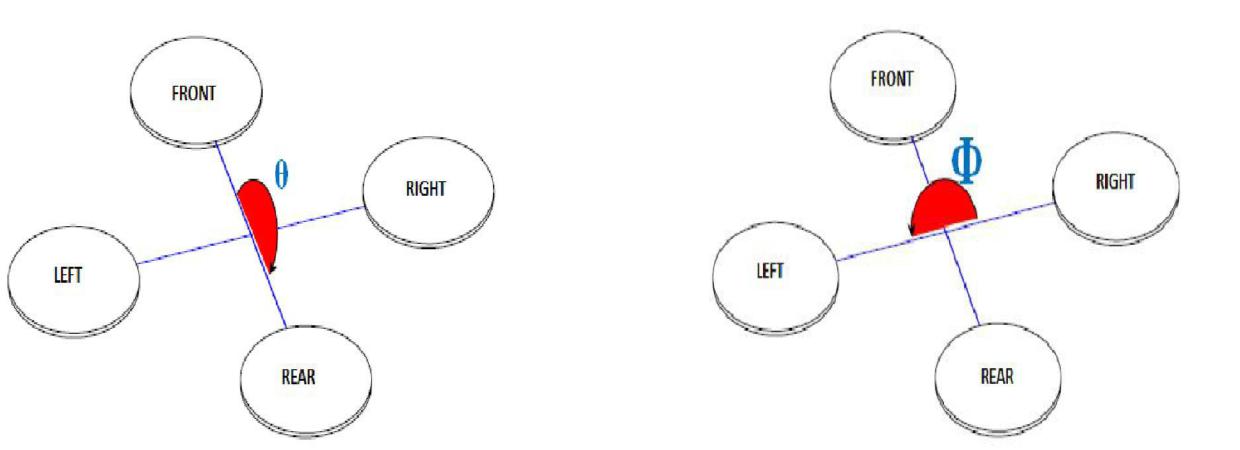


Figure 3.7 : pitch direction Figure 3.8 : roll direction

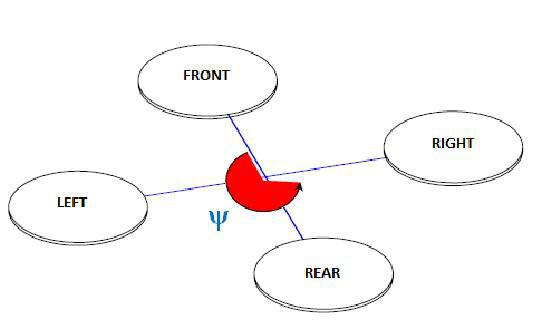


Figure 3.9 Yaw direction

3.1.3 Taking-off and landing motion mechanism

Quadcopter can be described as a small vehicle with four propellers attached to the root located at the cross frame. This aim for fixed rotors is used to control the vehicle motion. The speeds of these four rotors are independent. By independent pitch, roll and yaw attitude of the vehicle can be controlled easily. Pitch, roll and yaw attitude of Quadcopter.

Hovering or static position

The hovering or static position of the Quadcopter is done by two pairs of rotors, by rotating in clockwise or counter-clockwise respectively with the same speed. By two rotors rotating in clockwise and counter-clockwise position, the total sum of reaction torque is zero and this allows the Quadcopter to be in a hovering position.

Forward and backward motion

Forward (backward) motion is controlled by increasing (decreasing) speed of rear (front) rotor. Decreasing (increasing) rear (front) rotor’s speed simultaneously will affect the pitch angle of the Quadcopter.

Left and right motion

For left and right motion, it can be controlled by changing the yaw angle of the Quadcopter. Yaw angle can be controlled by increasing (decreasing) counter-clockwise rotors speed while decreasing (increasing) clockwise rotor speed.

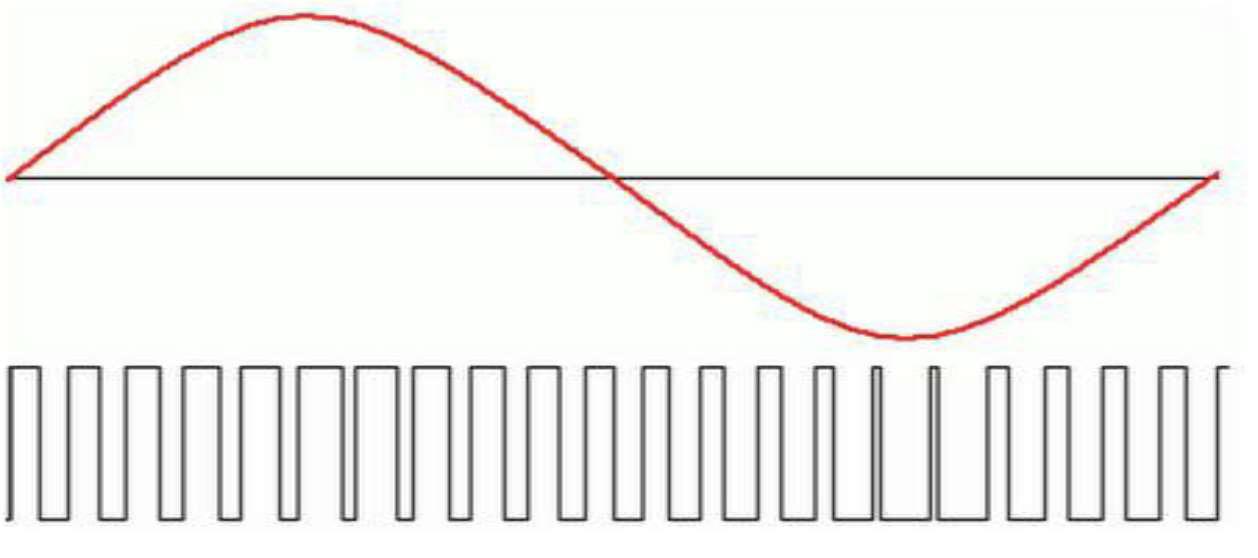
3.1.4 Processing of Receiving Signal

Modulation Technique in Quadcopter

Pulse Width Modulation

Modulation Technique in Quadcopter: Pulse Width Modulation

Pulse width modulation is a way of simulating an analog output by varying HIGH and LOW signals at intervals proportional to the value. Width of each pulse varies according to the amplitude of the analog signal.



Pulse Position Modulation Figure 3.10 Pulse width Modulation

Pulse-position modulation is a form of signal modulation in which M message bits are encoded by transmitting a single pulse in one of 2M possible time-shifts. This is repeated every T seconds, such that the transmitted bit rate is M/T bits per second.

.

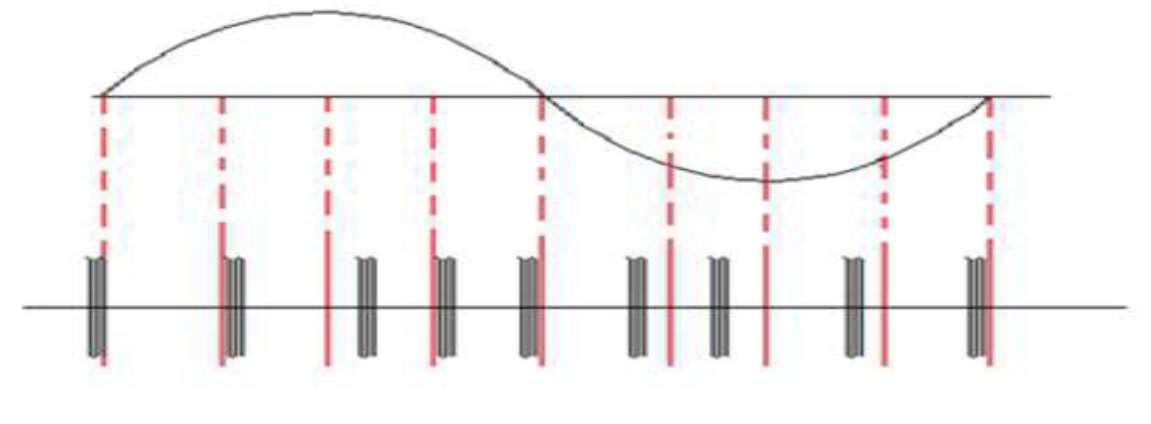


Figure 3.11 Pulse Position Modulation

PWM to PPM conversion

Pulse position modulation (PPM) is a pulse modulation technique that uses pulses that are of uniform height and width but displaced in time from some base position according to the amplitude of the signal at the instant of sampling. PWM refers to a pulse width modulation signal, where the width of each pulse changes according to the amplitude of an analog signal. PPM on the other hand refers to a pulse position modulation signal, where the width of each pulse remains the same, but each pulse is displaced by a certain position based on the analog signal amplitude. The basic need for conversion of a PWM signal received from a transmitter into a PPM signal arises due to the fact that the main controller board used (ArduIMU) on a Quadcopter can’t process a PWM signal and hence a converter is required to convert a PWM signal to a PPM signal.

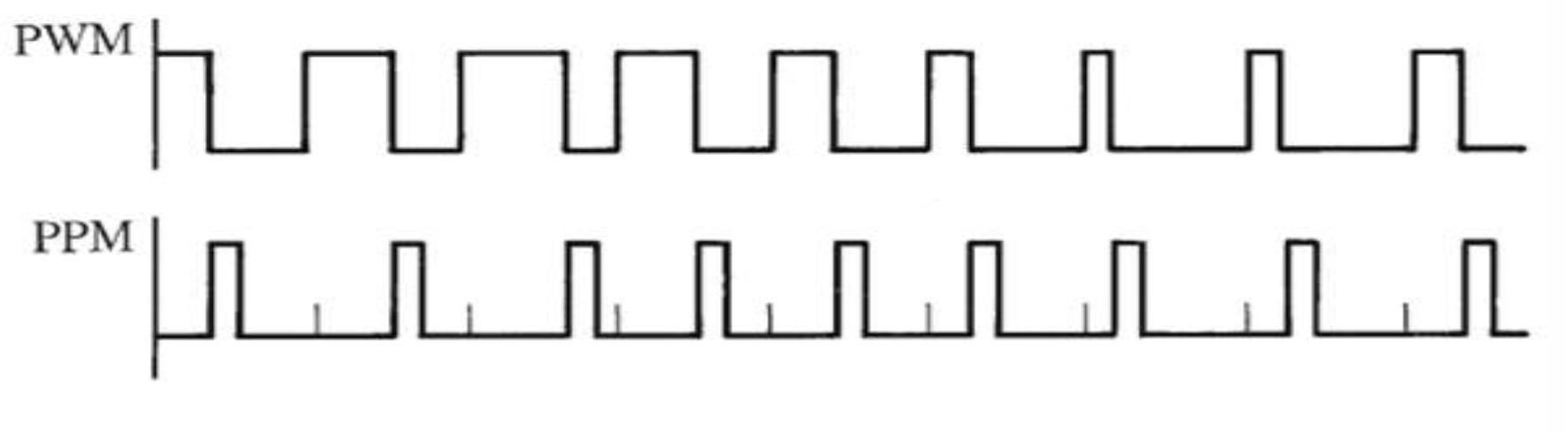


Figure 3.12: PWM to PPM conversion

3.2 Software Analysis

In this project we are using Arduino and Digital Radio Software

3.2.1 Arduino Analysis

In the project the program is dumped to the controller through Arduino. Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

3.2.2 Digital Radio Software

This is the software which is used to set PID Control Settings. Here we can set the different channels to be used for Radio transmitter and Receiver. Model that is used is MODEL-2. Different types of settings are available as:

1. ACRO

2. HELI-120

3. HELI-90

4. HELI-140

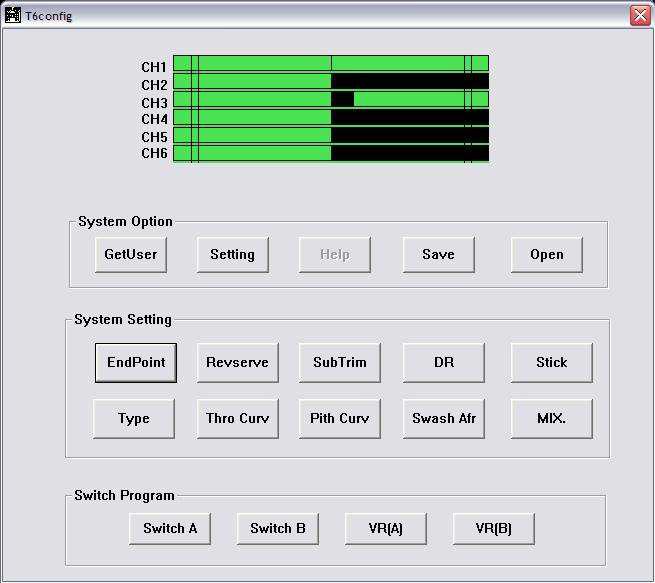


Figure 3.13 Digital Radio Software

3.3 Hardware Analysis

There are different steps to be followed in this analysis.

1. Assembling of Frame

2. Soldering for Chassis

3. Connection of ESC’S

4. Fixing of Brushless motors

5. Propellers fixing

6. Attachment of Flight controller KK2.1.5

7. Synchronization of Transmitter and Receiver

8. Checking receiver test and calibration

9. Adjusting the receiver parameters to IDLE

10. Testing the Quadcopter

3.3.1 Frame

Quadcopter frame can be called as the chassis of the quadcopter. The frame can be achieved in different configurations such as +, X, H, etc…the selection of the frame is totally a user defined choice based on his own purposes.

We used HJ 450 Frame. FlameWheel450 (F450) is a multi-rotor designed for all pilots for fun. It can achieve hovering, cruising, even rolling and other flight elements. It can be applied for entertainment, aerial photography, FPV and other aero-modeling activities.

When flying, the fast rotating propellers of FlameWheel450 will cause serious damage.

Safety precautions to be taken are:

1. Keep flying multi -rotor away from objects, such as obstacles, human beings..

2. Do not get close to or even touch the working motors and propellers, which will cause Serious injury.

3. Do not over load the multi-rotor.

4. Check that the propellers and the motors are installed correctly and firmly before flight.

5. Make sure the rotation direction of each propeller is correct

6. Check whether all parts of multi-rotor are in good condition before flight. Do not fly with old or broken parts.

7. Use DJI parts as much as possible.



Figure 3.14 HJ-450 4 axis Frame

3.3.2 Soldering

Chassis which is inbuilt with HJ-450 frame has to be soldered for connecting ESC’S. Chassis works as a PCB printed Board for power supply.

We have used Insulating material for soldering. While soldering we must make sure that there is no open or close circuit.

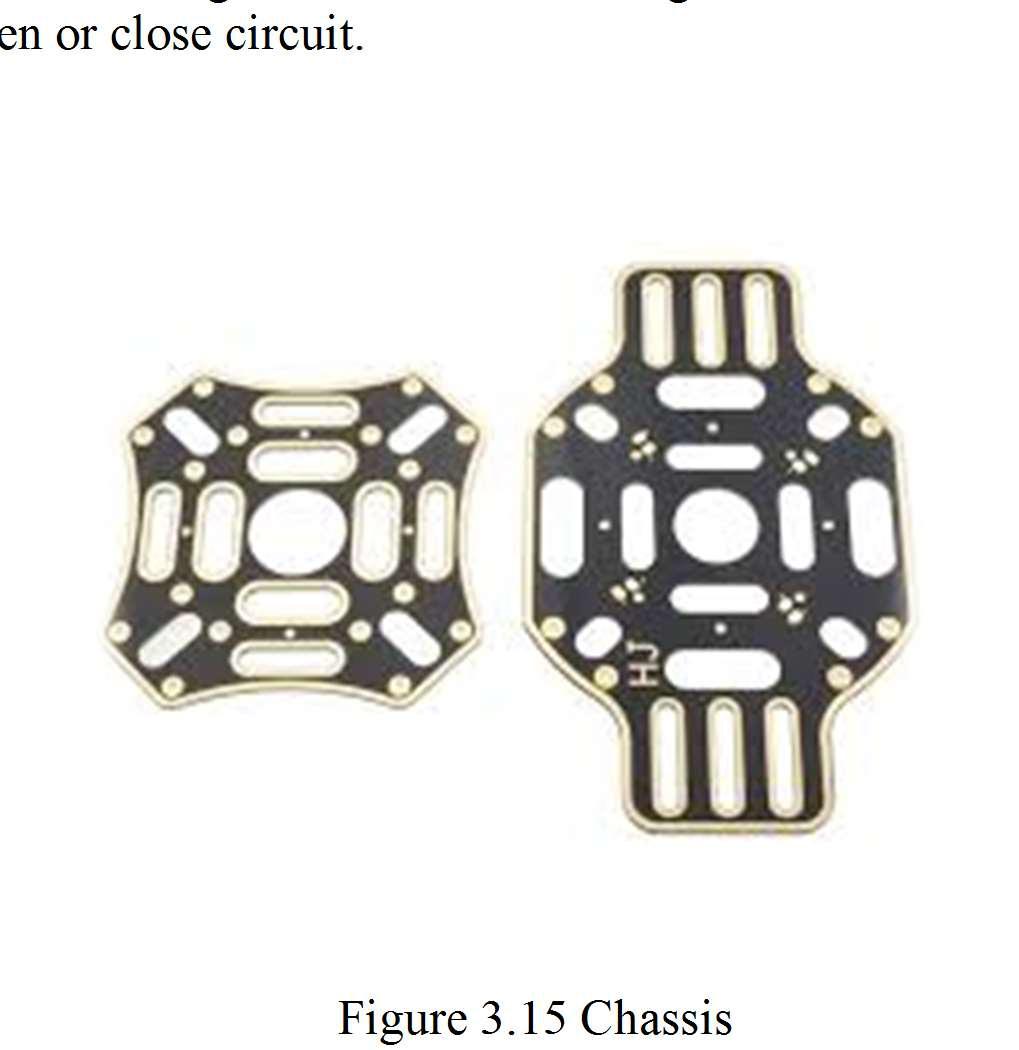


Figure 3.15 Chassis

3.3.3 Connection of ESC’S

After Soldering is done 4 ESC’S has to be connected to Chassis of HJ-450 frame. Proper Care should be taken so as not to get short circuit.

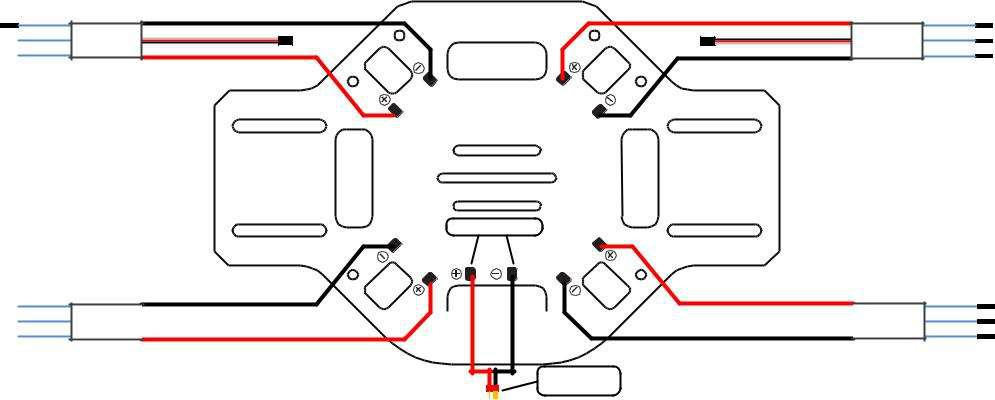
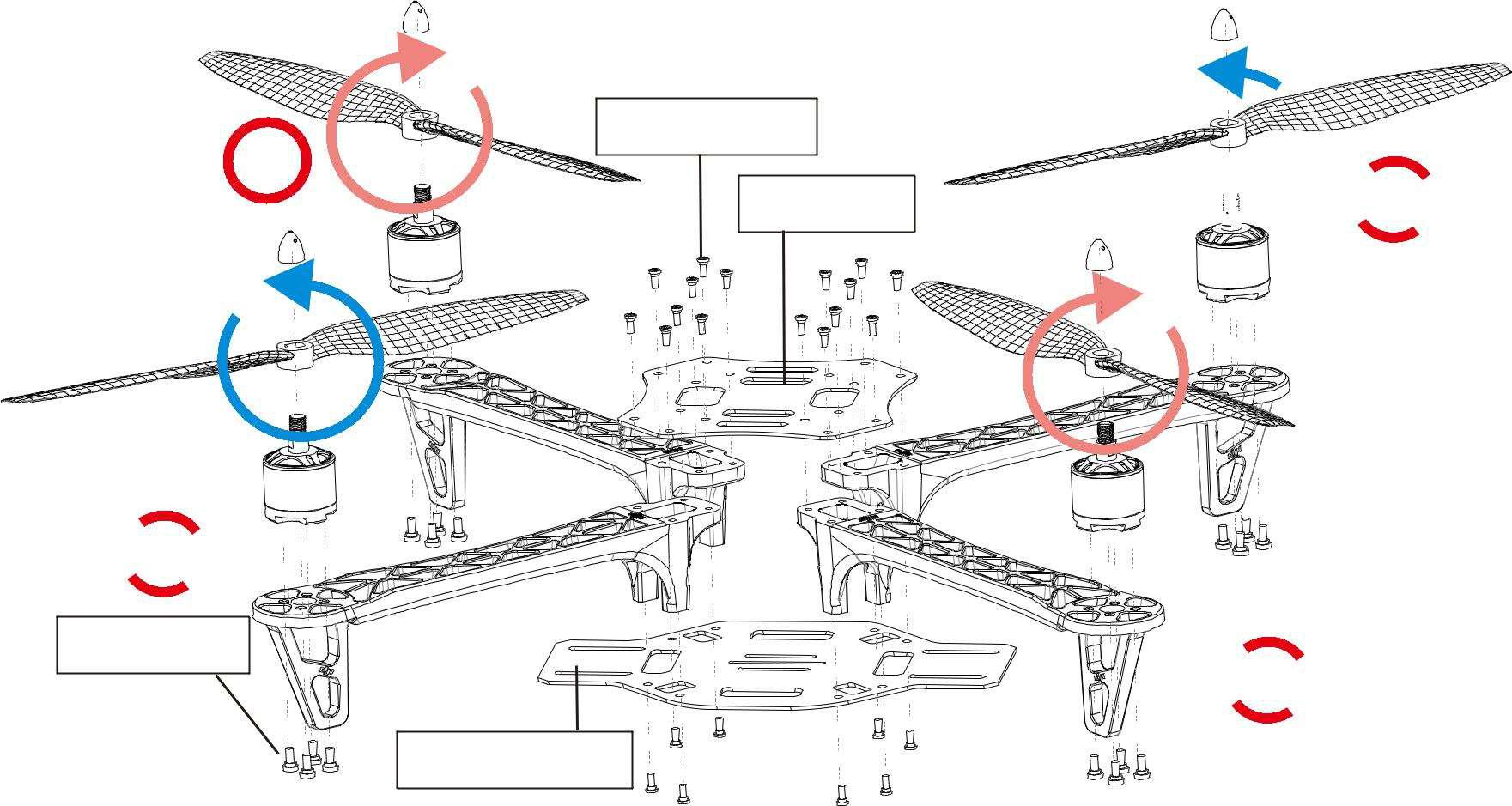


Figure 3.16 ESC Wiring



|  |  |  |
| --- | --- | --- |
| 1 | Screw M2 .5x6 |  |
| Top board | 2 |
|  |

3

Screw M3x8

4

Bottom Board

Figure 3.17 Over View of HJ 450

3.3.4 Fixing of Brushless motors

After fixing of ESC’S we need to attach 4 brushless motors of 1000 kv each.Attachment of Brushless motor to ESC’S is to be done carefully so as not to get burst of winding.The three bullets which are attached to brushless motore is to be connected with ESC’S.

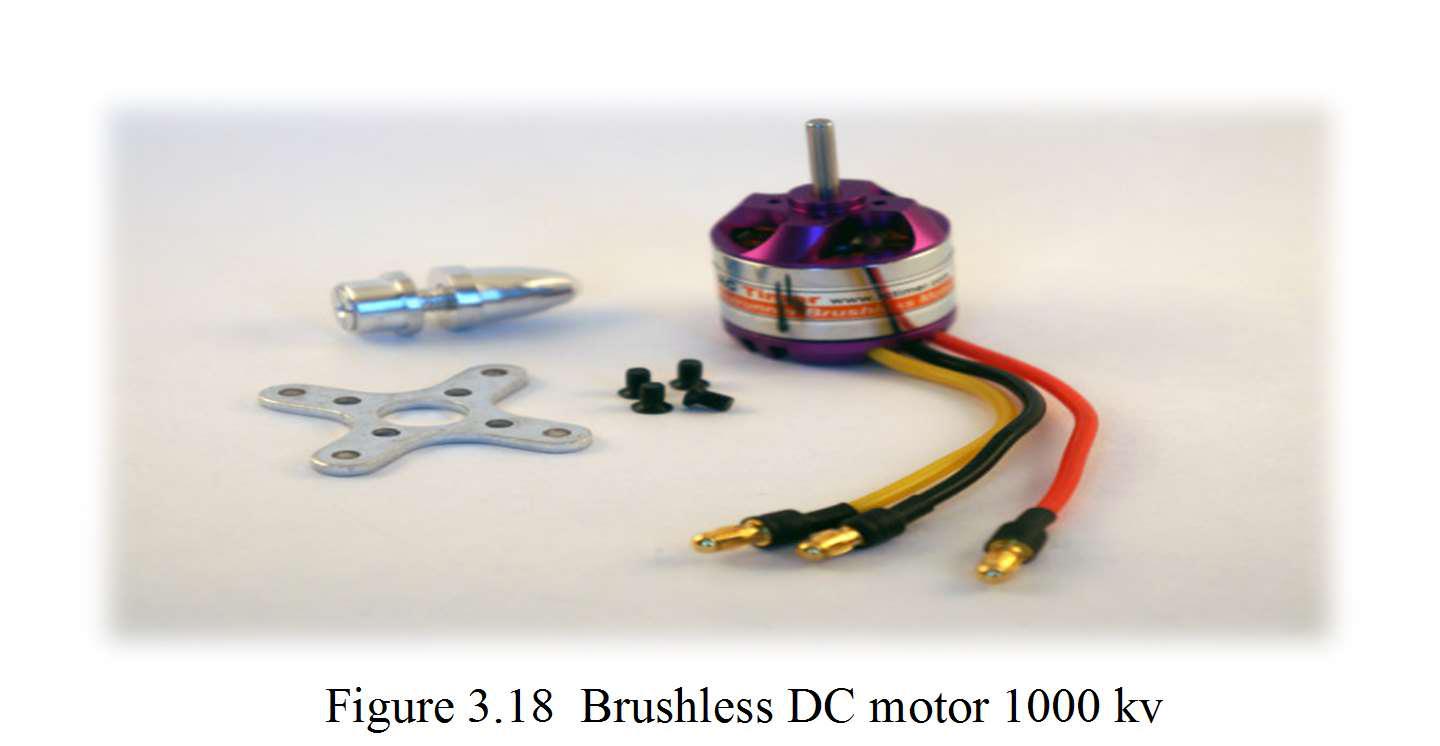


Figure 3.18 Brushless DC motor 1000 kv

3.3.5 Fixing of Propellers

After attaching brushless DC motors we need to 4 propellers.Two of them in clockwise and two of them in counter clock wise direction.

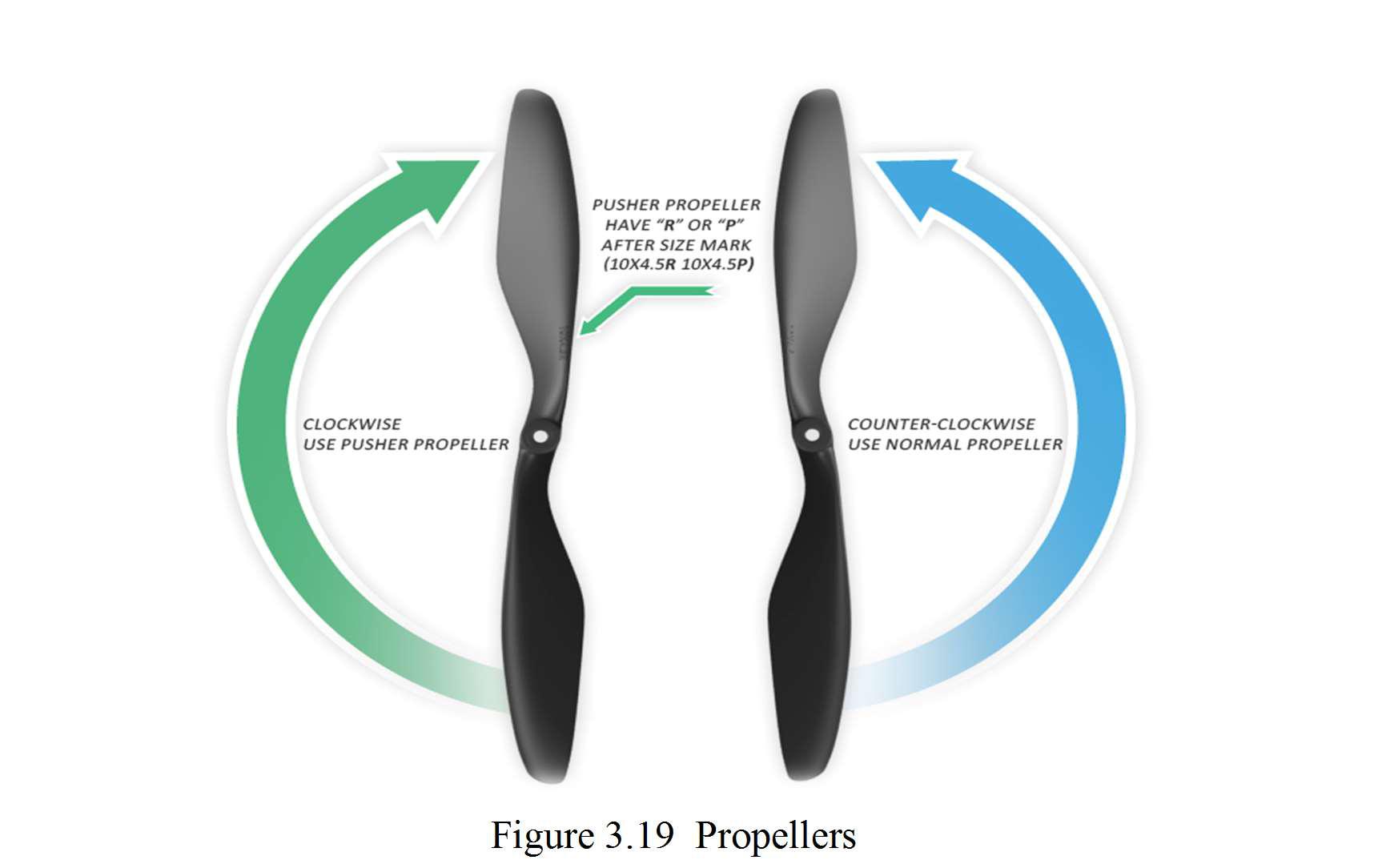


Figure 3.19 Propellers

3.3.6 Flight controller KK2.1.5

Firstly we need to “LOAD MOTOR LAYOUT”,checking for the directions of 4 motors of clock wise and counter clock wise direction.

Secondly we need to set PI controller settings as:

Roll/Pitch Axis:

Pgain = 50

Plimit = 100

Igain = 25

Ilimit = 20

Yaw Axis:

Pgain = 50

Plimit = 20

Igain = 25

Ilimit = 10

Default gains are set to 50/50/50 (roll/pitch/yaw) P-term, and 25, 25, 50 I-term. Limits are used to set the maximum value of the available motor power to be used for correction, so for example 100 is 100%. The I limit value is also known as anti wind-up in PID theory. The use of Limits is most important on the yaw axis and to prevent a large yaw correction from saturating the motors causing no control of the roll/pitch axis. The default values permit 30% (P Limit 20 + I limit 10) of the motor power to be used to make a yaw correction, making 70% available for the roll/pitch axis, the most important ones. You can increase Yaw P Limit for faster Yaw response. Note that Yaw response is also limited by the craft dynamics itself. Accessing the Self-Levelling Mode

1. You can access the self-levelling mode either from the settings of STICK or AUX channel.

2. When set to AUX Mode you must connect a spare channel usually CH5 or Ch6 and changing the Transmitter switch position will enable/disable Self- Levelling mode.

3. When set to STICK Mode to go into Self-Levelling Mode, you must set the Throttle to Minimum and set maximum Left Rudder whilst at the same time, setting maximum Left Aileron to disable SL or maximum Right Aileron to enable SL.

Flight Controller Sounds

1. One Beep (short beep, 2 sec delay) is emitted when the board is armed and the throttle is closed, this is for safety reasons so you know it’s armed.

2. One Long Beep is emitted when the board is either Armed or Disarmed.

Status Screen

1. Displays the message "SAFE" and the KK2.1.5 will not arm unless it says "OK"

Thirdly caliberation of flight controller has to be done from Ground Level according to surface area.

The Flight Controller Board must always have a source of +5v from an ESC, either one of the motors ESC or from a separate unit feeding the Receiver. If each ESC has a BEC (normal unless OPTO types) then it may be necessary to remove the power feed from the other ESC, usually by cutting the power line (RED) Cable on the other ESC.

STEP-1 Mount the FC on the frame with the LCD facing front and the buttons facing back. You can use the supplied anti-static foam container as a form of protective case for the Flight Controller on the craft.

STEP-2 Connect the receiver outputs to the corresponding left-hand side of the controller board.

STEP-3 Connect the ESC’s to the right side of the Flight Controller Board. M1 is towards the front of the board and M8 is nearest to the push buttons. The negative (black or brown) lead towards the edge of the FC. The negative (black or brown) lead is connected to the edge of the Flight Controller.

The Flight Controller Board must always have a source of +5v from an ESC, either one of the motors ESC or from a separate unit feeding the Receiver. If each ESC has a BEC (normal unless OPTO types) then it may be necessary to remove the power feed from the other ESC, usually by cutting the power line (RED) Cable on the other ESC.

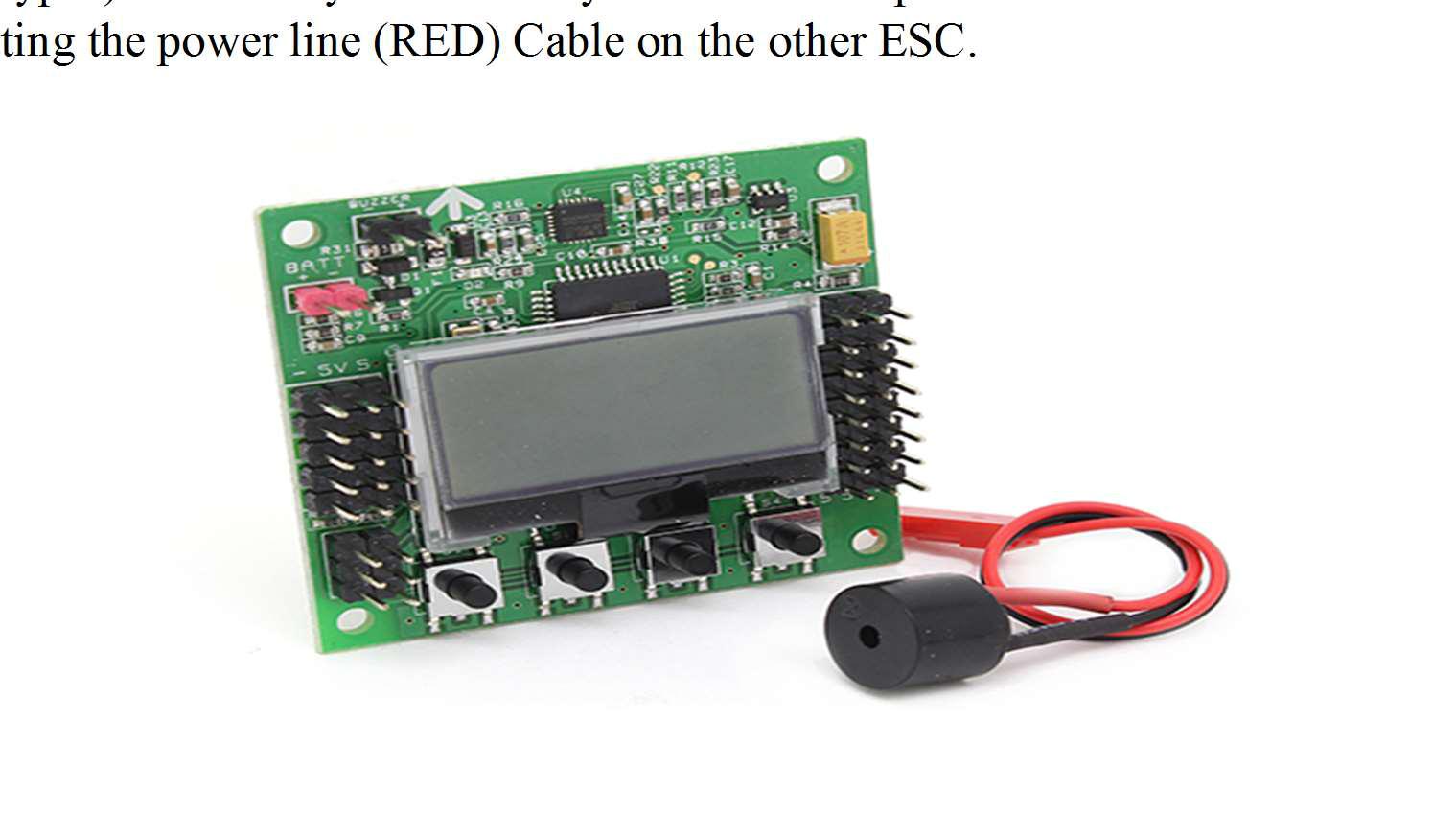


Figure 3.20 Flight Controller KK.2.1.5

3.3.7 Synchronization of Transmitter and Receiver

Receiver which is connected to Flight controller has to be synchronized with transmitter fly sky-CT6B.

Receiver has Six channels and BAT.Each has three pins as:

1. Signal

2. Ground

3. Supply

Dummy Wire has to be connected to BAT Of Signal and Supply.After connecting it has to be synchronized with transmitter by pressing the BEEP button present.Along with it any one of the four ESC’S has to be connected to any of the channel present in the receiver.

3.3.8 Receiver Test

Once it is synchronized,in the flight control board the receiver parameters will start changing according to adjustment in the fly sky transmitter-CT6B.

Aileron

Elevator

Throttle

Rudder

AUX

3.3.9 Receiver Parameters Set to IDLE

Once receiver test is performed,all the receiver parameters has to be set to IDLE for proper configuration.That means all parameters must be set to Zero.

3.3.10 Testing

After all the operations are performed,transmitter should be On and then moving the throttle

up and down for about 3 times and then moving left to right,will make the flight controller to change to ARMED state and hence quadcopter hover the skies.

3.4 Static thrust Calculation

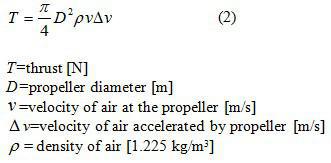
Calculations of static thrust are needed in order to ensure that the proper propellers and motors have been selected.

Static thrust is defined as the amount of thrust produced by a propeller which is located stationary to the earth. This calculation is particularly important for this project because Quadcopter are more likely to perform at low speeds relative to the earth. This low-speed performance ensures that the calculations of static thrust can be applied to a wide range of flight conditions. Also, it is important to note that the final calculations of static thrust are estimates and not actual values.

The first step in calculating static thrust is determining the power transmitted by the motors to the propellers in terms of rpm. To calculate power, the formula used for their datasheet is given in Equation 1.

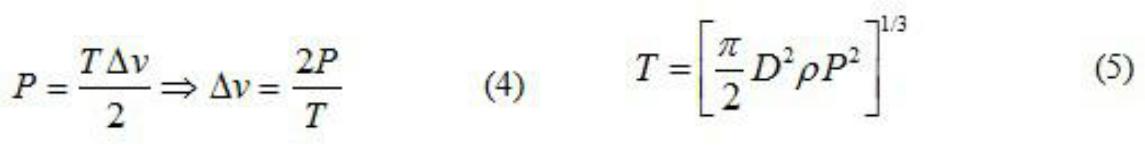
Where power is in watts and rpm is in thousands. For example, a 6X4 APC propeller has a propeller constant of 0.015 and a power factor of 3.2. Given a rotational speed of 10,000 rpm, the calculation goes as follows:

Power=0.015X103.2=24 W.

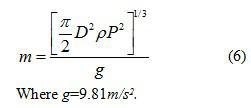
The next step is to determine the thrust produced by a propeller. Equation 2 gives thrust based on the Momentum Theory commonly used rule is that velocity of the air at the propeller is v=½Δv of the total change in air velocity: Therefore, and equation 3 is derived.



Equation 4 gives the power that is absorbed by the propeller from the motor. Equation 5 shows the result of solving equation 4 for Δv and substituting it into equation 3. In doing so, Δv is eliminated and torque can be calculated.



Finally, it is advantageous to express the results of equation 5 in terms of mass. Newton’s Law, F=ma, is used to obtain equation 6.



Solving for mass is useful for Quadcopter because it can be directly related to the mass of the aircraft. In particular, a thrust (mass) that equals the mass of the aircraft is needed for hovering.

3.5 Trouble shooting of the project and Difficulties Faced

We faces several problems through the journey of making this project which demanded our proper attention towards the project. Some of them are discussed in this section.

Arming the motors

Building the frame

Interfacing of the board

Achieving the stability

Arming the motors:

We are not aware of arming the motors and how to do it at the initial stage of the project and for this reason we ended up with an ideal copter without any response on throttle. So we went looking back on our work and found no troubles. After watching several videos from internet we came to learn about the process of arming and the problem is solved.

Building the frame:

Several hurdles are passed in building the frame for achieving the center of gravity, center of mass at the desired positions designed. The problem faced even in selecting the material for building the frame. When we went on building the frame using balsa wood we ended up with a hung frame on mounting the frames. So made some research and finally decided to build frame using Al foils.

Interfacing the board:

To program the board as per our purpose we need to interface it to the PC which needs a cable and a burner. In the beginning we were in search of board drivers and ended up with an error message on PC. After looking through the manual properly we understood about looking for the drivers of burner instead of board, problem solved.

Achieving the stability:

Once we were ready with our drone we started to take trials and we ended up with unstable flight which kept went around without giving any directional inputs other than throttle this is the major problem that we were stuck with. So we looked back on shifting the weights on frame and taking trial and from each trial we used to take feedback of the performance and after such several trials we were able to achieve up to the mark.

3.6 Disadvantages

Limited Abilities: Drones have obvious limitations. For example, they cannot communicate with civilians for more detailed intelligence. Drones cannot capture

surrendering military personnel, abandoned hardware, or military bases. Drones cannot go from door to door, at least, not yet.

2. Civilian Losses: Drone warfare often causes collateral damages in civilian lives and property, as well as traditional warfare too.

3. Counterproductive and Destabilizing: Civilian opinions about drones are typically negative, since they are viewed as an invasion force. The mere presence of drones has been known to convert civilians into military combats. Furthermore, when drones cause collateral damage, such as killing civilians and damaging civilian property, the opinions of civilians decrease even more so. Additionally, some cultures believe the use of drones as not brave and cold hearted. As a result, drones are sometimes counterproductive by more destabilizing some regions.

4. Too Easy: By making drone warfare very similar to video games, drone warfare makes combat too easy by diminishing ethical decisions.

5. Work and Personal Life Balance: Some drone pilots or operators have difficulty switching between combat mode at work and civilian mode while not working. This is especially difficulty when drone pilots have minimal transition periods between work and personal, if any at all.

6. Take Over: Finally, the worst case scenario is when drones or a fleet of drones have been commandeered or taken control by the enemy. While security measures help make this possibility more difficult, it will never be impossible.

**2 SRS**

# Introduction

## Purpose

The purpose of this project is to develop Semi-Autonomous Quadcopter. This UAV will be able to connect to the google maps online api to find a way between two GPS locations and be able to follow that path to reach the destination. Also, to counter the obstacles it finds during its flight, there would be a suitable algorithm which helps it to avoid those obstacles and re route it’s way accordingly to have a successful flight.

## Document Conventions

The font used during the SRS was Times New Roman. For normal text, the font size was 12, and 14 for headings. The headings were in bold. The text was left indented.

## Intended Audience and Reading Suggestions

The intended audience for this document would be the students who want to learn more about this project or want to use it as a reference for their own project.

## Product Scope

The scope of this technology would be future students who are doing a similar project. Additionally, it would be for people who would want to automate their day to day task by using a quadcopter.

# Overall Description

## Product Perspective

This product is a new, self-contained product.

As an intuitive capstone project, we wanted to do something which would require give us a fruitful experience to enter the world of making project. There are various ways in which you can automate a quadcopter, for example self-flight, security UAV etc. Our autonomous quadcopter would be able to move through two Given GPS coordinates by itself and also be able to avoid the given obstacles on its own.

The basic use of such a system is sending any kind of item from one point to another, basically as a delivery drone. This can help a user in day to day task like for watering plants, spraying insectescide on the crops.

The User will have to feed two GPS coordinates and a desired height and the quadcopter will automatically move in that direction. Then it would take off and land on the respective gps coordinates. The same is given in the Use Case diagram(See Appendix A)

## Product Function

The product performs the following functions:

* Get the GPS coordinates from the user using the computer software interface.
* Get the Height from the user.
* Take-off from the starting point, and land on the desired final gps coordinate
* Would be able to detect obstacles and avoid them.
* To have a stop command which would immediately stop the Quadcoper and land it safely on the same location.

## Operating Environment

This is an embedded system, so there are no constraints with respect to the running environment. However, the microcontroller that we would be using is the Arduino Uno and use the Ardu Pilot flight controller board. All the sensors are processed in FMS itself.

The software on the computer would be able to run on windows only since we are developing the software for that platform.

## Design and Implementation Constraints

The design constraints that we faced where the dimensions of the Quadcopter frame, and various other factors like centre of gravity, weight etc for a smooth flight. We have used the F450 quadcopter frame to build our quadcopter.

The only legal constraint that we can face is the restriction of UAV in India. The maximum height that we can test our quadcopter is 200m and that would be kept in mind while designing the code.

## User Documentation

We referred to Stack Overflow (*https://www.stackoverflow.com/*) for looking up for algorithms and ciruits.io(<https://circuits.io>) for simulations and debugging.

## Assumptions and Dependencies

We have assumed that the flight environment will not be turbulent while designing the algorithm. We are dependent for routes on the google maps api.

We are also going to use the laptop for image processing and making judgements about obstacles, and hence there is an active network connection between the Camera and the PC.

# Interface

## User Interfaces

The User interface will be very simple. It would only include the 2 options, first would be entering the GPS Coordinates, and second would be setting the height. The same would be developed as an Application or a web interface.

## Hardware Interfaces

The hardware interface would consist of an Arduino Uno board which would help us to interface the various sensors, which are Gyroscopic, Accelerometric, GPS, Barometric sensors to find the correct orientation of the Quadcopter and also to correctly stabilize the quadcopter when left in idle state. The Arduino official software would be used to upload the code onto the flight management sensor.

## Software Interfaces

The software interfaces that we would be using would mainly include Google Maps Api to find the route between the Start and the destination and the other interfaces that we would be including are ArduPilot and MultiWii which are open source projects that are specifically designed for the better stabilization of the Quadcopter and make changes to the quadcopter to our need.

## Communications Interfaces

The communication interfaces would include a Bluetooth module and a wifi module. The commands would be sent by both the Bluetooth and the wifi module. However, the wifi module would also be specifically designed for network communication for attaining correct user data.

# Other Nonfunctional Requirements

## Performance Requirements

The system must be user-friendly and the delays involved must be less. So in every action-response of the system, there are no immediate delays. In case of opening forms, of popping error messages and saving the settings or sessions there are no noticeable delays.

## Safety Requirements

Information transmission should be securely transmitted to server without any changes in information.

## Security Requirements

The main safety requirement for this project is that mid-flight, there should be no way that there can be sent a command that can alter the flight route or any behavior that the UAV must not show.

# Feasibility

## Operational Feasibility

The quadcopter is an example of an embedded system. So, the first thing in operational feasibility is that how good an embedded system is a quadcopter. The main operation of the quadcopter is flying. The flying can be easily achieved using an Arduino or any other microcontroller.

The other operations that are also the main functional requirements of the Project are: -

1. GPS point to point movement
2. Detection of Obstacles using camera

The completed project would consist of the above two actions resulting in the flying of the quadcopter, or one can say that these actions would in turn give commands to the quadcopter to fly according and adjust its course. With due help of various research papers, it has been found that this is possible. The first thing, GPS movement can be implemented, and secondly, detecting object can also be done using a camera or an Ultra Sonic Sensor. The challenge is to combine these two and find whether which input is more favorable at a point of time and making decision. It has been seen that it is possible to implement such an algorithm in which, whenever an obstacle is detected, the quadcopter first readjusts itself to avoid the obstacle and then it corrects its orientation and then continue its course from that point. This can easily be implemented and is operationally feasible and hence can be implemented.

## Technical Feasibility

The technical feasibility can be divided into parts:

### Hardware Feasibility

The hardware required for this project is a microcontroller, a gyroscope and accelerometer sensor, a gps sensor, a wifi module and a Bluetooth module. All this will comprise of the Flight management system or the flight controller.

The other parts that are required are 4 brushless motors, 4 ESC (Electronic Speed Controllers), rotors and a quadcopter frame and a battery.

All these parts are readily available and are also in the process of being ordered. The connections are also readily available on the internet for use.

### Software Feasibility

The help from three things would be required, Python-For image processing, Arduino(C/C++) for controlling the Arduino and Html, CSS and a backend to write a web interface.

For proper functioning, the interface and image recognition should be able to send commands over to the Arduino board so that the Arduino can take necessary steps. This is possible with the help of libraries found on Pypi, and NPM and can be done easiliy.

## Economic Feasibility

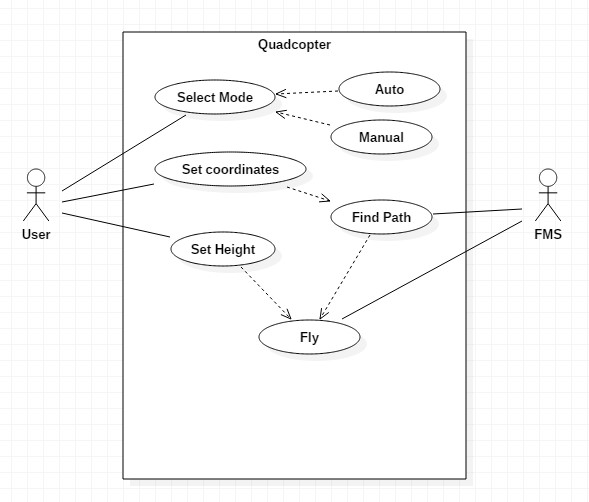
All in all, the project would cost around Rs. 10000 which is not very much and is being provided by the college and hence is feasible economically.

## Legal Feasibility

The project is Legally feasible in other countries, but in India, there is a restriction of height that the quadcopter can reach and hence the software would be designed such that restriction is not Crossed.

**Appendix A-Analysis Models**

1. Use Case Diagram



1. Use Case Templates

|  |  |
| --- | --- |
| Use Case ID: | 1 |
| Use Case Name: | Select Mode |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Select the mode in which the quadcopter has to fly (auto/manual) |
| Preconditions: | - |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | First event |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | Mode A (manual), Mode B (Auto) |
| Special Requirements: | - |
| Assumptions: | The quadcopter has just started and is waiting for first instructions from user |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| Use Case ID: | 2 |
| Use Case Name: | Set Coordinates |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Set the latitude and longitude of destination |
| Preconditions: | Mode should be auto |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | The Quadcopter has been set to auto mode and is waiting for instructions from user |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| Use Case ID: | 3 |
| Use Case Name: | Set Height |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Select the height at which the quadcopter has to fly (max 3m) |
| Preconditions: | - |
| Postconditions: | - |
| Priority: | Low |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

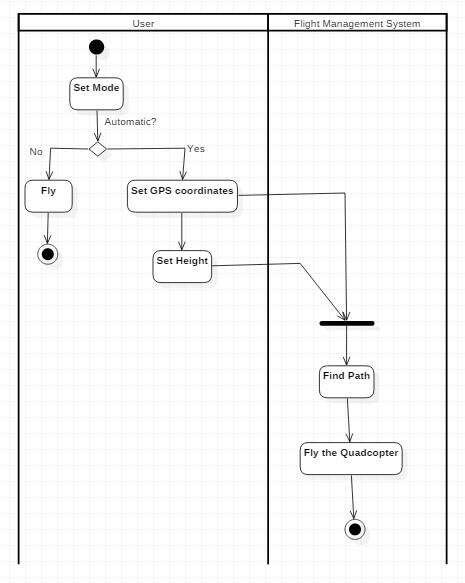
|  |  |
| --- | --- |
| Use Case ID: | 4 |
| Use Case Name: | Integrate with Google Maps |

|  |  |
| --- | --- |
| Actor: | Flight Management System |
| Description: | Find path which it has to follow |
| Preconditions: | Coordinates must have been set and mode should be auto |
| Postconditions: | - |
| Priority: | Medium |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height (Optional) -> Integrate with Google Maps |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | Coordinates must have been set and mode should be auto |
| Assumptions: | - |
| Notes and Issues: | - |

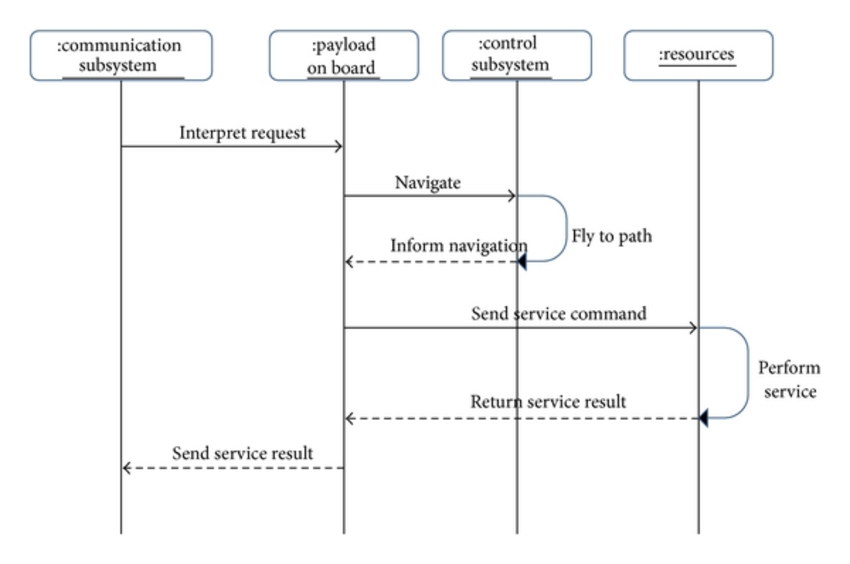
|  |  |
| --- | --- |
| Use Case ID: | 5 |
| Use Case Name: | Fly |

|  |  |
| --- | --- |
| Actor: | Flight Management System |
| Description: | Fly with the given instructions |
| Preconditions: | Coordinates must have been set if mode is auto |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Continuous |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height (Optional) -> Integrate with Google Maps -> Fly |
| Alternative Courses: | Mode (manual) -> Fly |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | Coordinates must have been set if mode is auto |
| Assumptions: | Default height is 3m |
| Notes and Issues: | - |

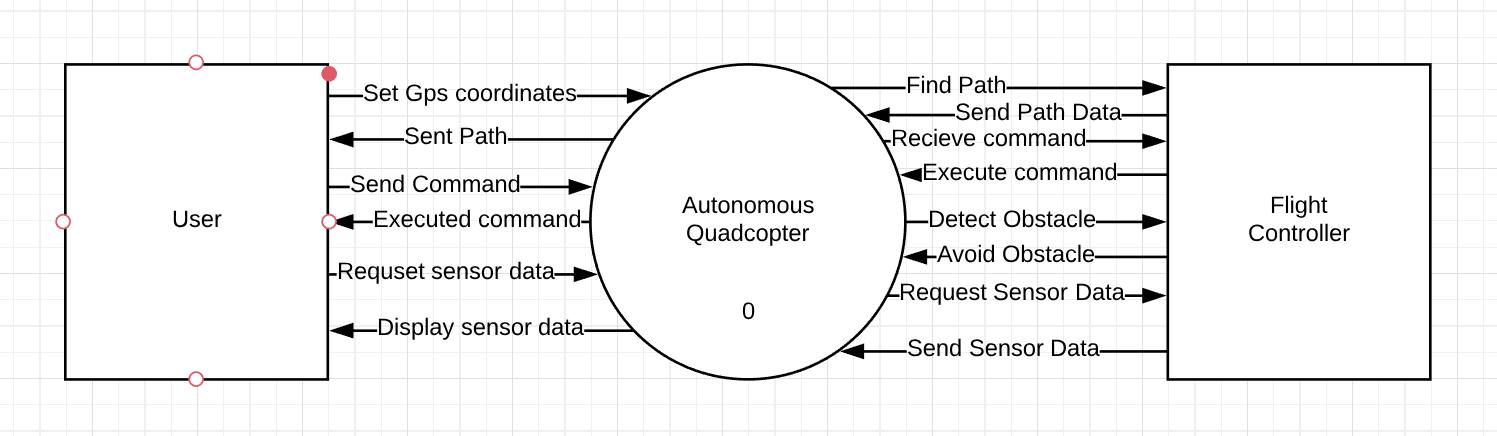
1. **Activity/Swimlane Diagram**



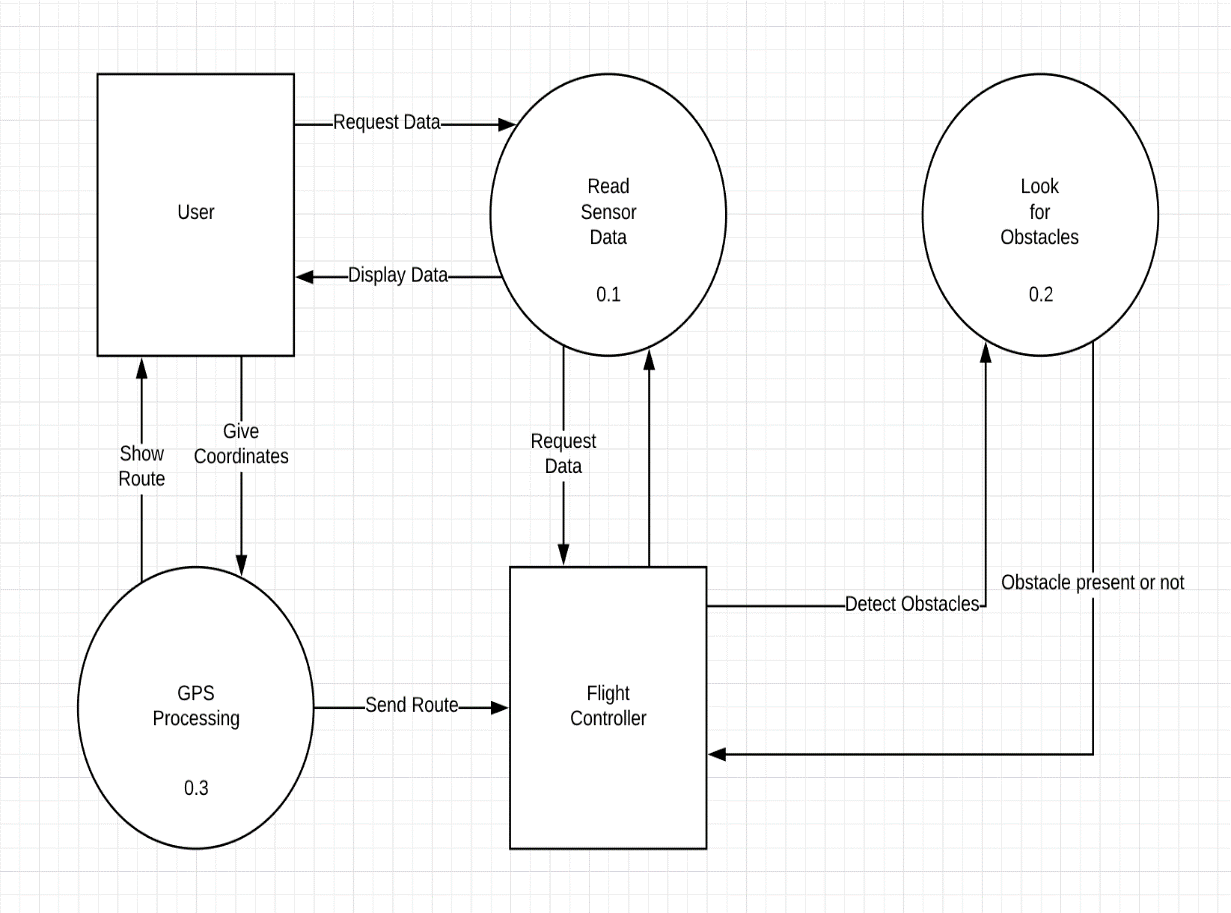
1. **Sequence Diagram**



1. **Data Flow Diagram**
   1. **Level 0**

****

* 1. **Level 1**

****

**3 Assumptions and Constraints**

1. **Assumptions:**

* Quad copter will only travel to coordinates of the address captured through Google maps.
* Quad copter and our laptop would remain within the same Wi-Fi network.
* Obstacles would only be detected through ultrasonic sensors due to cost constraint.

1. **Constraints:**

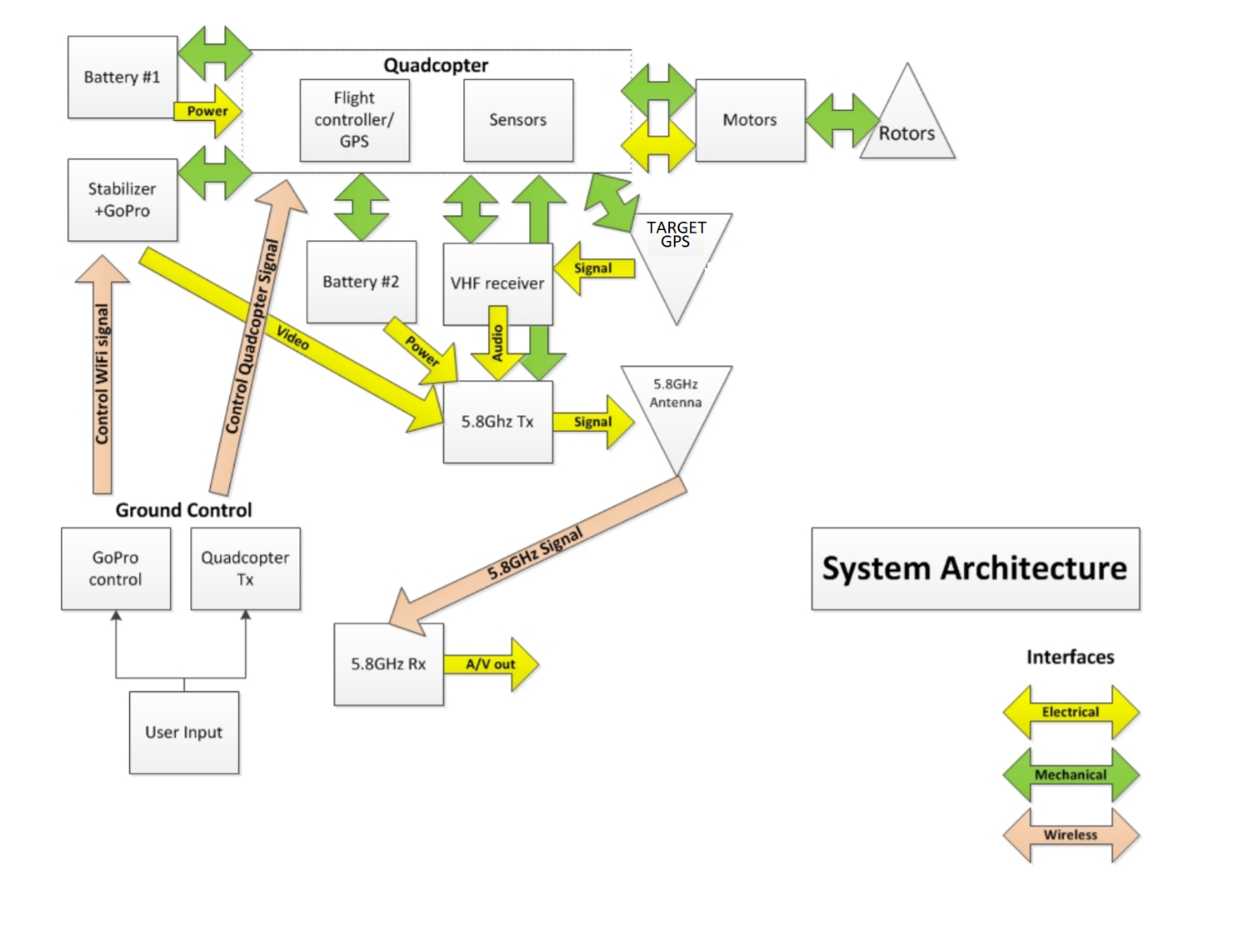
* Max flying height would be 3m.
* Cost of the project is a major constraint.
* It cannot hover to different coordinates of earth if they are not registered with an address on Google maps.

**4 Cost Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No. | Part | Model | Cost (Rs) |
| 1 | Bluetooth Module | HC05 | 380 |
| 2 | LiPo Battery | 11.1 V, 25C, 220mAh | 1730 |
| 3 | LiPo Battery Charger | REESS2 | 589 |
| 4 | Quadcopter Frame | F450 | 4410 |
| 5 | 4 Blushless Motors | A2212 KV1000 |
| 6 | 4 ESCs | 30A |
| 7 | 2 Pairs of Propellers | 1045 |
| 8 | GPS Positioning Module | NEO 6M U-Blox | 935 |
| 9 | Gyroscope + 3 Accelerometer Module | MPU6050 (GY521) | 229 |
| 10 | Serial Wi-Fi Transceiver Module | ESP8266 | 278 |
| 11 | Jumper Wires | M-M, M-F, F-F | 150 |
| 12 | Arduino Uno | Uno R3 | 526 |
| 13 | Ultrasonic Sensor | HC-SR04 | 245 |
|  |  | Total | 9472 |

**5 Detailed Architecture**

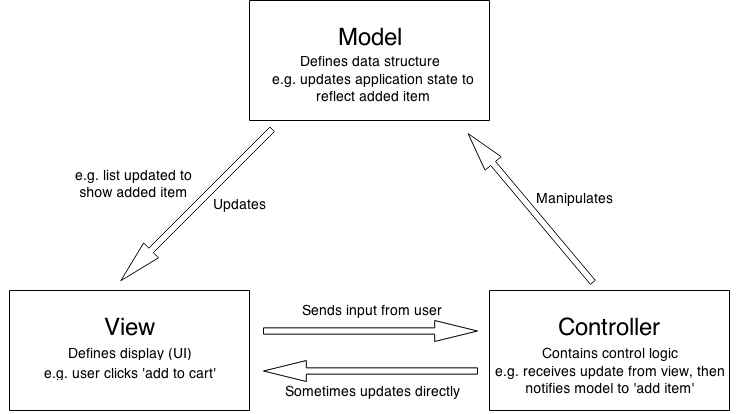
**5.1 Context Diagram**



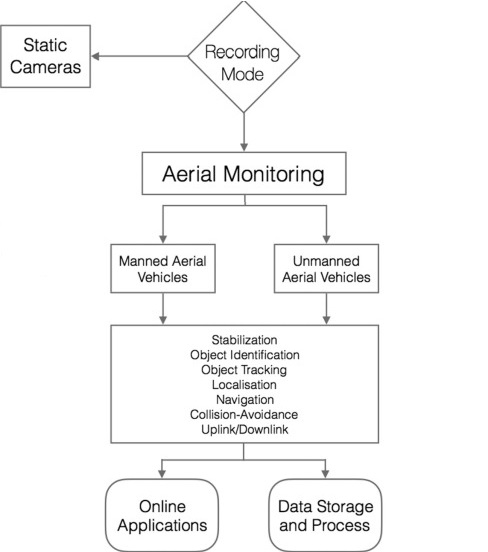
**5.2 MVC Diagram**

In this section, you will get an overview of MVC architecture. The MVC architectural pattern has existed for a long time in software engineering. All most all the languages use MVC with slight variation, but conceptually it remains the same.

MVC stands for Model, View and Controller. MVC separates application into three components - Model, View and Controller.



**Model**: Model represents shape of the data and business logic. It maintains the data of the application. Model objects retrieve and store model state in a database.



**View**: View is a user interface. View display data using model to the user and also enables them to modify the data.



**Controller**: Controller handles the user request. Typically, user interact with View, which in-tern raises appropriate URL request, this request will be handled by a controller. The controller renders the appropriate view with the model data as a response.

