B. TECH. PROJECT REPORT

On

DRONE DESIGN WITH OBJECT DETECTION AND TRACKING ABILITY

BY

ANKIT GAJBHIYE
DEEPAK KUMAR
KAILASH LIMBA

DISCIPLINE OF ELECTRICAL ENGINEERING
INDIAN INSTITUTW OF TECHNOLOGY INDORE
December 2019

DRONE DESIGN WITH OBJECT DETECTION AND TRACKING ABILITY

A PROJECT REPORT

Submitted in partial fulfilment of the requirements for the award of the degrees

of
BACHELOR OF TECHNOLOGY
in

ELECTRICAL ENGINEERING

Submitted by:
ANKIT GAJBHIYE
DEEPAK KUMAR
KAILASH LIMBA

Guided by:
Dr ABHINOY KUMAR SINGH
INSPIRE FACULTY

INDIAN INSTITUTE OF TECHNOLOGY INDORE
DECEMBER 2019

CANDIDATE'S DECLARATION

We hereby declare that the project entitled **"Drone Designigning With Object Detection And Tracking Ability"** submitted in partial fulfilment for the award of the degree of Bachelor of Technology in 'Electrical Engineering' completed under the supervision of **Dr Abhinoy Kumar Singh, Inspire Faculty, Electrical Engineering, IIT Indore** is an authentic work.

Further, we declare that we have not submitted this work for the award of any other degree elsewhere.

Ankit Gajbhiye	**Deepak Kumar**	**Kailash Limba**
(160002006)	**(160002013)**	**(160002021)**

CERTIFICATE by BTP GUIDE

It is certified that the above statement made by the students is correct to the best of my knowledge.

_____

Dr Abhinoy Kumar Singh

Inspire Faculty

Electrical Engineering

Indian Institute of Technology, Indore

Preface

This report on "Drone Design with Object Detection and Tracking Ability" is prepared under the guidance of Dr Abhinoy Kumar Singh.

Through this report, we have tried to give detailed information about a drone with the ability of object detection and tracking, and tried to cover every aspect of the new design, if the design is technologically and economically sound and feasible.

We have tried to the best of our abilities and knowledge to explain the content in a lucid manner. We have also added 3-D models and figures to make it illustrate.

ANKIT GAJBHIYE

DEEPAK KUMAR

KAILASH LIMBA

B.Tech. IVth Year

Discipline of Electrical Engineering

IIT Indore

Acknowledgements

We would like to express our sincere gratitude to our supervisor Dr Abhinoy Kumar Singh and Stochastic Control Lab members for their invaluable guidance, comments and suggestions throughout the course of project.

I would also like to thank Col. Anant Bhatt for hosting us at Military College of Telecommunication Engineering for a period of 15 days and for his help during the comparative analysis of algorithms.

ANKIT GAJBHIYE

DEEPAK KUMAR

KAILASH LIMBA

B.Tech. IVth Year

Discipline of Electrical Engineering

Indian Institute of Technology, Indore

ABSTRACT

Object detection and tracking are very important components of computer vision system. It has attracted many engineers and researchers in this field as it has multiple applications in video surveillance, navigation, 3D image reconstruction, robotics etc. This report will give a brief idea about the algorithms used for object detection and tracking, hardware setup of drone and software required for this task. This report proposed the development of an unmanned aerial vehicle (UAV) which is controlled by wireless transmitter. This implemented design is capable of flying, graceful motion, accurate altitude hold performance and also capable to detect the objects in the frame of the camera and track the target given by user using machine learning algorithms. In this system we used Pixhawk Flight Controller (3-axis accelerometer, 3-axis gyroscope & 3-axis magnetometer) which ensure its smooth movement and graceful motion. All signals are processed by a flight controller board which makes it more efficient and effective. This project was aimed to design a drone from scratch that continuously detect the objects in the frame and track the target given by the user using a movable camera mounted on a drone. All data and results discussed at the end of every chapter.

CONTENTS

Candidate's Declaration……..……..ii

Certificate by BTP Guide……..……ii

Preface…….……iii

Acknowledgements…….……..……...iv

Abstract……...……..……...v

Chapter 1: Introduction…….……..1

1.1 Literature Review……..……..2
1.1.1 Oehmichen (1920)……..……....2
1.1.2 De Bothezat Helicopter (1922)……..……..3
1.1.3 Queen Bee……..……..4
1.2 Recent Advancements……4
1.3 TLD……...……....5
1.4 Mean Shift……..5
1.5 Compressive Tracking……5

Chapter 2: Drone Design……..…….……..6

2.1 Components Required……...……6
2.1.1 Frame…….....6
2.1.2 Flight Controller……7
2.1.3 Motors……...8
2.1.4 Electronic Speed Controller……..…9
2.1.5 Battery……...9
2.1.6 GPS……..……10
2.1.7 Remote Controller……..11
2.1.8 Gimbal……12
2.1.9 Camera……....13
2.1.10 Video Transmission……..…13
2.2 Components Setup……...14
2.3 Methodology……...…17
2.4 Software Setup……..…21
2.5 Results……..……26

Chapter 3: Object Detection…….…..29

3.1 Introduction……..……..29
3.2 What is YOLO & why is it very useful?.....30
3.3 How Does YOLO function?.....30
3.4 How to encode Bounding Boxes?.....34
3.5 Intersection over Union & Non-Max Suppression…….……36
3.6 Creating dataset to train new classes……38

3.7 Tools to create training dataset……..……..39
3.8 Implementation of YOLO……..……...……..39
3.9 Results…….40

Chapter 4: Object Tracking…….……..…….…….……..43

4.1 Introduction……..……43
4.2 Approach…….…….43
4.3 Experiments……..……43
4.3.1 SiamFC…….44
4.3.2 SiamRPN…….……44
4.3.3 SiamMask……...……..45
4.4 Implementation of SiamMask……...45
4.5 Results…….…….…….…….....47

Chapter 5: Conclusions…….…….…….…….…….50

References And Sources…….…….…….…….…51

LIST OF FIGURES

1.1 Oehmichen Helicopter…….....2
1.2 George de Bothezat helicopter…….……3
1.3 De Havilland DH82B "Queen Bee"…….….3
1.4 Amazon Prime Air drone for delivery…….…..…..….4
2.1 Frames for Quadcopter & Hexa-copter……..…6
2.2 Pixhawk 2.4.8 FC……...…7
2.3 Racerstar BR2212 Brushless Motors……8
2.4 Electronic Speed Controller……..9
2.5 Orange Li-Po battery……....9
2.6 GPS Module……...10
2.7(a) Transmitter Display……....11
2.7(b) Radiolink AT10II Transmitter……12
2.7(c) R12DS Receiver……12
2.8 Gimbal……12
2.9 sj4000Camera……12
2.10 RC832 Receiver and TS832 Transmitter……13
2.11 Pixahwk 2.1.8 FC……14
2.12 8 main-out Pin slot &6 aux-out Pin slot on PIxhawk…….……..15
2.13(a) integrated Power distribution board for Quadcopter……...15
2.13(b) integrated power distribution board for Hexa-copter……..16
2.14(a) Power module……...…16
2.14(b) connection between power and Pixhawk…….……16
2.15(a) Quadcopter motor rotation…….……..17
2.15(b) Hexa-copter motor rotation……..……17
2.16(a) Pitch, Yaw and Roll axes……18
2.16(b) movement against Pitch, Yaw & Roll axes…….……18
2.16(c) Pitch, Yaw and Roll axes of a plane……..……19
2.17(a) left and right side movement from rear-view…….……..19
2.17(b) Leftward Motion……20
2.17(c) Rightward Motion……...20
2.18 Forward and Backward motion of a drone from Rear-view……20
2.19(a) Forward Motion……..20
2.19(b) Backward Motion……...20
2.20(a) Clockwise Motion……...21
2.20(b) Anti-Clockwise Motion……..21
2.20(c) Clockwise Motion……...21
2.20(d) Anti-Clockwise Motion……..21
2.21 choosing frame type in Mission Planner……22
2.22 Accel Calibration in Mission Planner……22
2.23 Compass Calibration in Mission Planner…….....23
2.24 Radio Calibration……23
2.25 Selecting Flight Modes in Mission Planner……24
2.26 Choosing Failsafe values in Mission…….. 24
2.27 ESC calibration in Mission Planner……25
2.28 Message section……..25
2.29(a) Quadcopter from Top-View……26
2.29(b) Quadcopter from Side-View……...26
2.30 Testing of Quadcopter in MCTE…….……26

2.31 Testing of Hexa-copter in MCTE……..…..…27
2.32 Hexa-copter during flight……...….28
3.1 Object Detection…….…..29
3.2 Input image taken from our drone at IIT Indore campus……...…30
3.3 Input image divided into 3x3 grids…….……...31
3.4 Last grid image…….32
3.5 Bounding box on the car……...….32
3.6 Training Layers for YOLO model……....33
3.7 Bounding box on the car in a grid……..…...34
3.8 Coordinates assigned for a particular bounding……35
3.9 B_x and B_y value assigned for a particular bounding box……35
3.10 Intersection of actual bounding box and predicted box……....36
3.11 Yellow region is area of intersection……36
3.12 Predicted bounding boxes……..…37
3.13 Final bounding boxes……....38
3.14 Tool to create dataset……39
3.15 (a) and (b) are the inputs for object detection…….……...40
3.16 Object Detection output for (a)…….……41
3.17 Object Detection output for (b)…….41
3.18 (c) and (d) are the inputs for object detection…….……...41
3.19 Object Detection output for (c)…….42
3.20 Object Detection output for (d)…….42
4.1 Architecture of SiamFC……....44
4.2 Architecture of SiamRPN…….45
4.3 Architecture of SiamMask……46
4.4 Object tracking when drone height is more than 50m……..…..…47
4.5 Object tracking when another object clashes the target…….…....48
4.6 Object tracking when target goes out of the frame……49

LIST OF TABLES

2.1 Datasheet table for BR2212 motor……..8
2.2 Frequency for FPV transmitter and receiver……...13
3.1 Eight-dimensional coordinate values for each grid……..31
3.2 Eight-dimensional coordinate values for each grid……...32
3.3 Eight-dimensional coordinate values for centre right grid……....33
3.4 Eight-dimensional coordinate values for centre grid……34
3.5 Eight-dimensional coordinate values for centre grid……35

Chapter 1: Introduction

One might have heard about drones, the most used product in the field of UAVs. They have become a common sight over the past few years and people are using them for all sorts of purposes, aerial photography, inspection high altitude structures search-and-rescue missions, etc. and all other sorts of purposes.

The use of reconnaissance drones in the Vietnamese War highlighted the main purpose of drones, then and now: to gather information. As the modern technology make advancements and becomes available, various groups such as military, engineers, researchers and hobbyists have been developing new designs and their implementations keep coming out. These days, the drones are commonly used for aerial photography, drone racing, and many other purposes, and even multinational companies are investing in drone equipment and software development.

Visual surveillance in business environments attempts to detect, track, and recognize objects of interest from multiple videos, and more generally to interpret object behaviours and actions. For instance, it aims to automatically compute the flux of people at public areas such as stores and travel sites, and then attain congestion and demographic analysis to assist in crowd traffic management and targeted advertisement. Such intelligent systems would replace the traditional surveillance setups where the number of cameras exceeds the capacity of costly human operators to monitor them.

Proceeding with a low-level image features to high-level event understanding approach, there are three main steps of visual analytics: detection of objects and agents, tracking of such objects and indicators from frame to frame, and evaluating tracking results to describe the activity of the target. This analogy can be extended to other applications including motion-based recognition, access control, video indexing, human-computer interaction, and vehicle traffic monitoring and navigation.

Some work has been done in the past over object detection and tracking but with the use of a stationary camera. And that makes it quite difficult to locate the target if the target gets out of the frame of the camera mounted on the drone, so a system was needed that can follow the target if it tries to get out of the camera frame.

So, we decided to do a project on drone designing that can detect and track a target given by the user. In order to build the drone, FC, transmitter, receiver, camera, gimbal and a data

logging SD card was used. Each component was interfaced, tested and verified to be working properly and were compatible with each other. After that, we have gone through the comparative analysis of detection and tracking algorithms, and finalised the algorithm that we used.

1.1 Literature Review

The earliest unmanned aerial vehicle [1] in the history of drones was seen in 1849 ITALY, when both Venice and Austrian soldiers were fighting in war. Venice was fighting against Austria for its independence. Austrian soldiers attacked the city of Venice, with hot-air, hydrogen, helium filled balloons, with explosives.

The first auto-pilot or pilot-less radio-controlled aircraft was used in World War I. In 1918, the U.S. Military developed the Kettering Bug [2], an unmanned torpedo, a forerunner aircraft of present-day cruise missiles, which was never used in war.

1.1.1 Oehmichen (1920)

Etienne Oehmichen [3] had done experiments with rotorcraft design in 1920s. He tried six designs among which Helicopter No.2 had 4 rotors and eight propellers and all driven by a single engine. Oehmichen used a steel-tube frame in his helicopter No.2, with two bladed rotors at the end of each arms. To stabilise the helicopter laterally, five propellers spinning in horizontal plane, one propeller was mounted at the nose for steering and remaining pair of propellers were for forward motion as shown in Fig1.1.

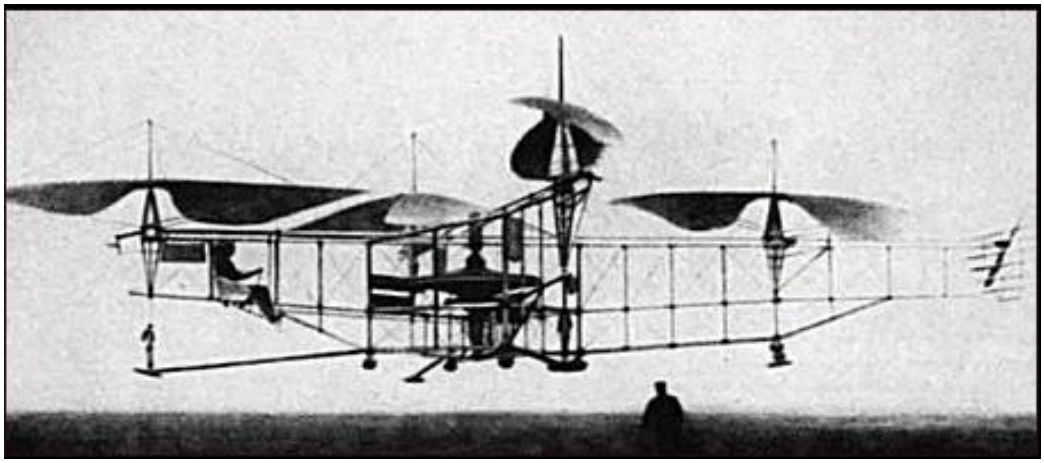


Fig 1.1: Oehmichen Helicopter

1.1.2 De Bothezat helicopter (1922)

Dr George De Bothezat and Ivan Jerome developed the aircraft vehicle [4], with six-bladed rotors at the end of a X-shaped structure. Two propellers with different pitch were used for thrust and yaw control (Fig1.2). It was built by US Air Service and made its first flight in October 1922. Its highest ever was around 5m (16 ft. 5 in.). It was unresponsive, mechanically complex and susceptible to reliability problem.

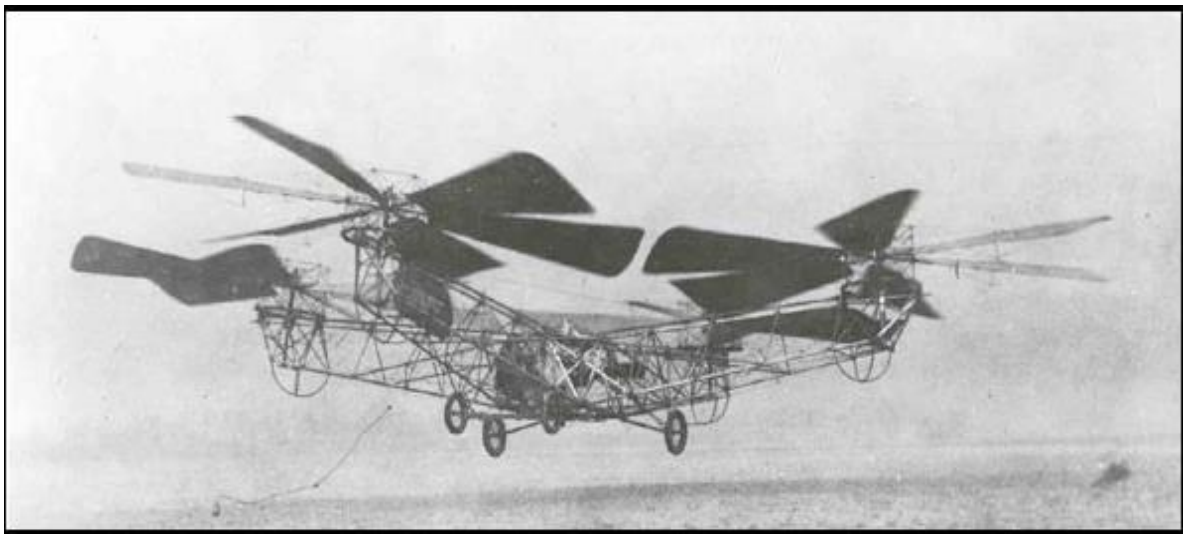


Fig1.2: George de Bothezat helicopter

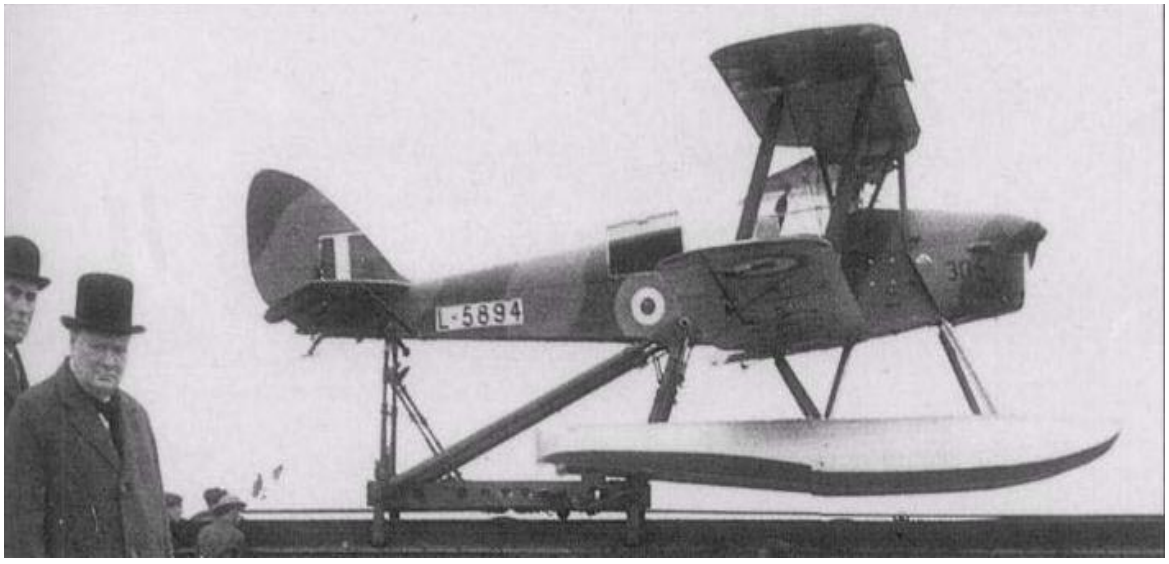


Fig1.3: De Havilland DH82B "Queen Bee"

1.1.3 Queen Bee

The first drone appeared in 1935 as a full-size retooling of the De Havilland DH82B "Queen Bee" [5] biplane (Fig1.3). This plane was fitted with a radio and servo-operated controls in the back seat of the drone. This plane could be conventional piloted from the front seat, but generally it flew pilotless. The term drone dates to this initial use, a play on the "Queen Bee" nomenclature.

1.2 Recent Advancements

According to a 2016 Wall Street Journal report claims that widespread drone use began 2006 when U.S. Customs and Border Protection Agency (BPA) introduced UAVs to monitor the U.S. and Mexico border [6].

In late 2012, Chris Anderson, chief editor of Wired Magazine, retired from his job to dedicate himself to his drone company, 3D Robotics. His drone company which started off specializing in hobbyist personal drones, now markets its drones to film companies and aerial photography, construction, telecom business and public safety companies, among others.

In late 2013, Amazon CEO, Jeff Bezos, announced a plan to use commercial drones for delivery purpose [7]. However, in July 2016, Reno-based Start-up company, Flirtey, beat Amazon to the punch, which it successfully delivered a package to a resident in Nevada through the use a commercial drone. Other companies also started taking interest in this idea. For example, in Sept. 2016, Virginia Polytechnic Institute and State University began a test with Project Wing, a unit of Google's parent company Alphabet Inc., to make deliveries through drone. Then, in Dec. 2016, Amazon delivered its first package in Cambridge, England and in March 2017, Amazon demonstrated a drone delivery in California.

Fig1.4: Amazon Prime Air drone for delivery

1.3 TLD (Track Learn Detect)

It is one of the best video object tracking algorithm. It performs its function in mainly three steps, i.e. tracking, learning and detection [8]. The bounding box will follow the trajectory of the target given by user. The detector monitors the all the displayed tasks that have been running on the backend and modify the bounding box if needed. The learning updates the errors of the detection to avoid tracking failure in future. The learning methods used for estimating the errors is P-N learning by a pair of "experts": (i) P-expert calculates all the mixed detections, and (ii) N- expert calculates all the false alarms.

1.4 Mean Shift

The bounding box for the target is defined in the first frame, as a region of interest. Mean shift [9] algorithm is responsible for separating the target from the background of frame. It moves tracked object data to the local maxima of probability density function. 'Aryabhatta' Coefficient [10] have been used to estimate the difference between two distribution.

1.5 Compressive Tracking

There are a number of tracking algorithms which have the disadvantage of lack of information of the target due to learning of false data. Compressive Tracking [11] algorithm works in a different way, it is a display model, based on the characteristics extracted from the multi scale information. An unique matrix is adopted to efficiently extract the characters. Some of the background and front targets are compressed using the above matrix mentioned. The video object tracking calculations are based on naive Bayes classifier and binary classification.

Chapter 2: Drone Design

In this chapter, we are going to discuss about the components that we require to build a fully functional drone (Quadcopter [12] and Hexa-copter) like flight controller, electronic speed controller, motors, etc. and their working. We are also going to discuss their working principle, about how they communicate with each other to work as a drone system and finally, about the software(s) we may need and use to configure the different configurable components.

2.1 Components Required

In this topic, we provide the necessary information about various parts required for better understanding and their working principle and their setup.

2.1.1 Frame

Frame is the structure that holds all the components of a drone together. One of the most important part of a drone is its frame as it supports all the motors and other electronics and prevents them from vibrations. So, one has to be very precise while making it as they need to be string but also light-weight. All parts are going to be attached to the frame, so one has to choose the frame size depending upon the purpose it is used for, for e.g. fpv, sport racing, or photography, etc. After deciding the size of the frame, one is recommended to use the recommended setup with his frame for better performance instead of choosing any random setup to avoid improper functioning of the drone. Nowadays, the frame is readily available in market with different materials like carbon fibre, aluminium, wood, fibreglass, etc. we are going to use 250mm size for quad-copter and 550mm size for hexa-copter.

Fig2.1: Frames for Quadcopter and Hexa-copter

2.1.2 Flight Controller

A drone flight controller, or FC, is an important part of a drone and controls most of the onboard electrical components with the assistance of Arduino-like processor and an array of sensors. A flight controller can be used with either a quadcopter, or a helicopter, or a hexa-copter, etc. with the help of associated firmware, which the FC is flashed with. The firmware helps it define the information which the FC provides to different motors for flying. The FC takes the input from RC receiver and feeds it to different sensors like IMU, gyroscope, compass. It then takes the output from these sensors and send it to each motor connected through ESCs. An FC can be flashed using various configurators available like Mission Planner, Betaflight, Cleanflight, Raceflight or KISS, etc., but one has to check if his/her FC is compatible with that configurator.

In this project, we have used a FC called Pixhawk 2.4.8 designed by 3D ROBOTICS in an open-source project. It has an autopilot system capable of autonomous stabilisation [13, 14], way-point based navigation. It also has support for two-way telemetry with radio telemetry module. It can support 8 RC channels with 4 ports. It is provided with a micro-SD card slot for logging purposes which is also called as black-box of drone. There's an external LED system provided with 5 different colours which are red, blue, green, yellow and purple which comes with a flash sequence. The light patterns [15] are associated with sound [16] / tone patterns, which give us the status of the flight controller.

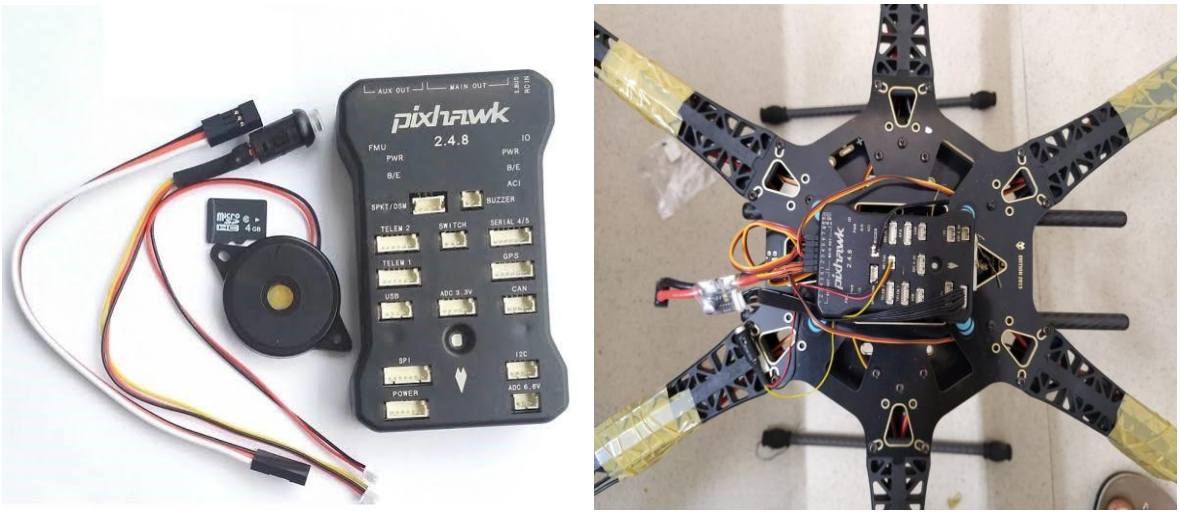


Fig2.2: Pixhawk 2.4.8 Flight controller

2.1.3 Motors

The drone motors are categorised in two parts, i.e., brushed and brushless motors [17]. We need reliable, high-quality brushless motors with rapid response for the drone system to fly smoothly. At some point of time while flying, if one or several of the motors faces any problem, as propellers [18] are attached to the motors, it would be a devastating occurrence for the drone system and it can at worst endanger the quadcopter itself, property, people and environment where it is being used. Furthermore, it's necessary for the motors to be powerful enough to be able to lift the quadcopter and perform various aerial movements required by the user. We need the motors to have a fast response to ensure a more stable flight. We require clockwise and counter-clockwise motor of the same quantity for stable flight.

Based on these criteria, we decided to use the Race-star BR2212 brushless motor. It is a brushless motor designed for remote-controlled aeroplanes as well as a quadcopter, and are considered to be highly reliable. According to the specifications, each motor can give a thrust of 710 grams at 118 Watt with an efficiency of 6.0 g/W.

Fig2.3: Racerstar BR2212 brushless motors

MODEL	KV (rpm/V)	Voltage (A)	Prop	Load Current	Pull (g)	Power (W)	Efficiency (g/W)	LiPo Cell	Weight (g) Approx.
BR2212	920	11.1	8045	7.3	465	81	5.7	2-4S	50
			1045	9.5	642	105	6.1		
	980	11.1	8045	8.1	535	90	5.9		
			1045	10.6	710	118	6.0		

Table 2.1: Datasheet table for BR2212 motor

2.1.4 Electronic Speed Controller

Electronic speed controller (Fig.2.3) is an electronic circuit with the purpose to vary an electric motor's rotating speed and also to perform as a dynamic brake. ESCs are frequently used on radio-controlled models which are electrically powered, most commonly used with brushless motors, basically providing an electronically produced 3-phase electric power low voltage source of energy for the motor to work. An ESC has three types of wire. One is the set of 2-wires that connects the ESC with power supply, another set of 3-wires connects to the Throttle pins of the flight controller and another set of 3-wires connects to the motor. An ESC draws two kinds of current from the battery named as Burst current and constant current. Burst current is the current required to initiate the rotating of the motors. Constant current is the current needed after the motor rotation speed saturates to continue rotating. The current required depend on the task the system is performing. We are using Emax Blheli 30A electronic speed controller whose pic is given below.

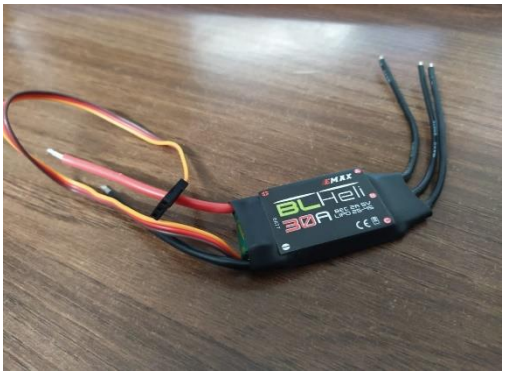


Fig2.4: Electronic Speed Controller (ESC)

Fig2.5: Orange Li-po battery

2.1.5 Battery

A battery (Fig.2.4) is a source of constant power supply which supplies the required input current by the circuit. Batteries are labelled in terms of discharge, i.e., max continuous discharge rate, burst discharge rate also the number of cells available in the battery. For example, some batteries come with 4-cell, others come with 3-cell, the difference between them is the number of cells associated with them, i.e., a 3-cell battery has three cells of battery plates connected in series and a 4-cell battery has four cells of battery plates in series connection. Discharge rate describes the current or power supplied, which are of 2 types, max burst and

max continuous discharge. Max continuous discharge rate are described as the maximum current or power that can be provided by the battery for some time continuously. Max Burst discharge are described as the maximum current or power that can be supplied to switch on the system. The battery we used is 5200 mAh 3S 40/80C LiPo battery which has three cells of batteries connected in series 40C as max continuous discharge rate and 80C as max burst discharge rate which can store 5200 mAh of power.

2.1.6 GPS Module

Global positioning system, or in other word GPS, is a satellite-based navigation system which was initially maintained by the US and designed to assist soldiers and military vehicles, but now it is readily available for common masses in the form of a GPS receiver. It is generally used for live positioning or location of a device wherever it may be present on earth. It has different uses based on the purpose required by the user, for example, monitoring, security, mapping and surveying, and other different purposes. In this project, we have used the GPS module [19] (Fig.2.5) mainly designed for the flight controller for various purposes like live positioning of our drone, waypoints mapping. It's also used as a failsafe method for our drone system, i.e., if the output voltage of the battery drops to a pre-fixed value, it will perform a set of function pre-fixed by the user like return-to-land, or it may even try to return to the position where the remote controller of the drone is positioned. It also has an attached compass with GPS module which helps us with better navigation of the drone system.

Fig2.6: GPS module

2.1.7 Remote Controller

A drone controller is a device used by the pilot to give a set of commands to the drone system. It works by sending a signal from the transmitter, which tells the drone what to do because of which it is also called remote controller. A remote controller has two elements, the transmitter [20] which the pilot holds in his hand to control the drone's movement [21] and the receiver is installed on the drone. The transmitter and the receiver communicate using a band of radio-frequency. Dramatically simplifying things here, the transmitter reads the sticks' inputs and sends them through the air to the receiver in near real-time. Once the receiver has this information, it passes it to the flight controller, which makes the drone move accordingly.

A drone transmitter comes with 6-8 frequency channels, and sometimes it also comes with 12 frequency channels. Each channel gives different feed based on the sticks' movement on the transmitter. The different frequency channels can be customized to perform a different function by the pilot as per his requirement. But it has four channels which give fixed input that is throttle, yaw, pitch, roll. Every transmitter has different compatibility and supports different communicating protocols like PPM, PWM, SBUS, IBUS, DSM2/X etc. We have used Radiolink AT10II remote controller (Fig2.7), which comes with 12 channels, has an inbuilt display (Fig.2.6), in which from channel 5 to 12 can be customized. It supports SBUS, PWM AND PPM protocol. It also has a function in which throttle to thrust ratio can be customized. It comes with R12DS as the compatible receiver. It also comes with OSD flight control telemetry module either PRM-01 or PRM-02 or PRM-03 depending on the model and requirement which gives us the information about throttle, speed, rise, voltage, longitude, latitude, altitude, GPS, etc. on display provided on the transmitter [22].

Fig 2.7(a): Transmitter display screen

Fig2.7(b): Radiolink AT10 transmitter

Fig2.7(c): R12DS receiver

2.1.8 Gimbal

The word 'gimbal' is defined as a pivoted support that allows rotation of any object in a single axis. A drone gimbal comes with either single axis, 2-axis or 3-axis freedom of movements or in other words, it has degree of freedom (dof) ranging from 1 to 3 different axes. So, a 3-axis gimbal allows any object mounted on the gimbal to be independent of movement of the one holding the gimbal. A drone gimbal is mainly used for stabilising the camera installed on it, enabling it to avoid facing any vibration occurred because of sudden movement during flight and hence allows the camera to capture videos or photos smoothly. We have used a 2-axis gimbal (Fig2.8), which enables the camera to have two dof enabling it to move freely in two different axes of rotation.

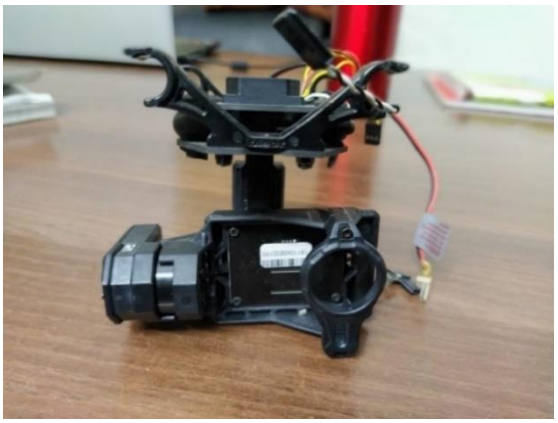


Fig2.8: Gimbal

Fig2.9: sj4000 Camera

2.1.9 Camera

The camera required by the drones is named as fpv camera [23] or first-person video camera, which means that it can send the video to the fpv display screen using a fpv radio. In this project, we have used sj4000 WIFI camera as it is fpv supported. It has a 14MP sensor and 170° wide-angle and 1280*720 HD resolution, which helps us to cover a relatively wide area with much clear resolution. It also comes with 90 min battery life which enables it to record longer without any headache that the camera's battery may die during our flight.

2.1.10 Video Transmission

Video transmission is achieved by using the compatible FPV transmitter and receiver. The camera sends the video in electrical signal to the fpv transmitter which then gets converted into radio signal by the fpv transmitter and transmitted. The fpv receiver then receives the transmitted radio signal and converts it to an electrical signal and sends it to the fpv display screen which the user holds. It has a different operating range (Table 2.2) from the remote controller to avoid any disturbance or mix-up between different radio signals and hence preventing any loss of video signal. In this project, we have used TS832 and RC832 (Fig2.10), which is commonly used for video transmission purposes.

CH / **FR**		**CH**							
		CH1	CH2	CH3	CH4	CH5	CH6	CH7	CH8
FR	FR1 or (A)	5895M	5845M	5825M	5805M	5785M	5765M	5745M	5725M
	FR2 or (B)	5733M	5752M	5771M	5790M	5809M	5828M	5847M	5866M
	FR3 or (C)	5705M	5685M	5665M	5645M	5885M	5905M	5925M	5940M
	FR4 or (D)	5740M	5760M	5780M	5800M	5820M	5840M	5860M	5880M

Table 2.2: Frequency for FPV transmitter and receiver

Fig2.10: RC 832 transmitter and TS 832 receiver

2.2 Components Setup

In this topic, we will be discussing how different components communicate with each other so that the final assembly becomes a fully functional quadcopter.

Fig2.11: Connections of Pixhawk

Fig2.11 gives us a rough idea about the connections of between Pixhawk board & GPS, Telemetry, buzzer and safety switch through the dedicated slots in pixhawk board.

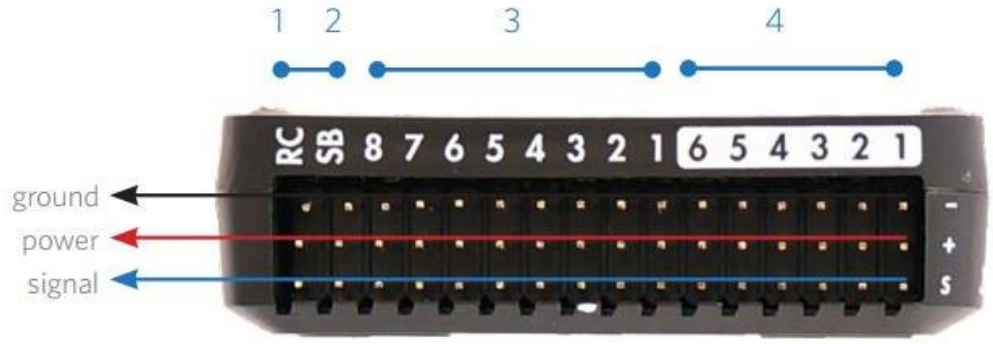


Fig2.12: 8 main-out pin slot and 6 aux-out pin slots on Pixhawk

Pixhawk board has been divided into 4 parts as labelled. No.1 represents the RC-in (RC) slot, No.2 represents S-bus (SB) slot, NO.3 has a total of 8 main-out pin slots, and No.4 has a total of 6 aux-out pins in which each slot has three pins. Each pin represents either power or ground or signal as mentioned. The remote-control receiver connects to either RC or SB, but it is advised to connect the receiver to RC pins. The eight main-out pin slots are output pins slot of pixhawk which are used to communicate with the ESCs.

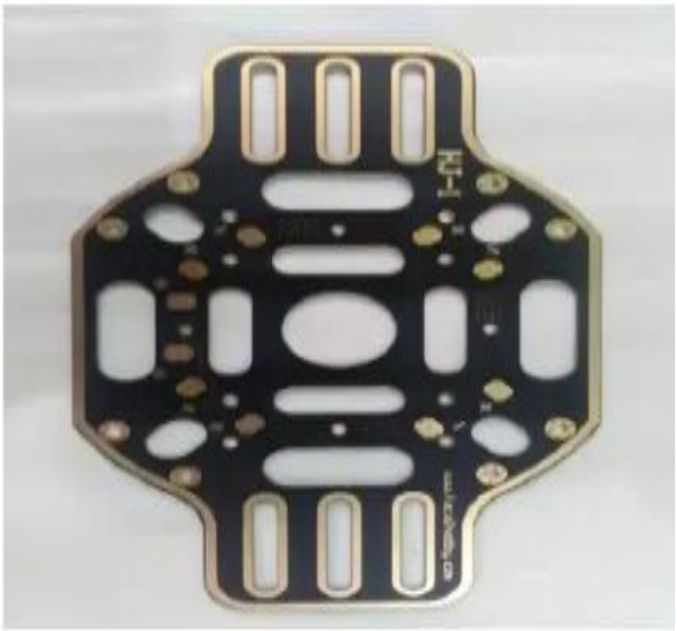


Fig2.13(a): Integrated power distribution board for quadcopter

Fig2.13(b): Integrated power distribution board for hexa-copter

Integrated power distribution board helps to reduce connections directly to the battery. This board provides a better solution by decreasing the number of wires used to supply power to ESCs, which are connected to the motors. Through the power distribution board, we can distribute the power to all different component of our quadcopter.

Fig2.14(a): power module and

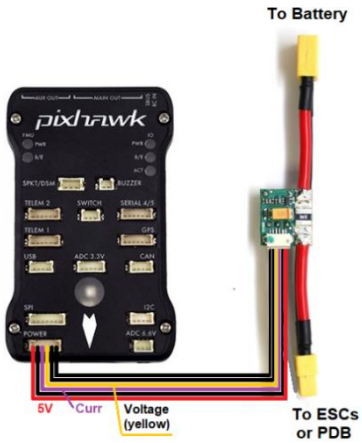


Fig2.14(b): connection between power module pixhawk board

The power module (Fig2.14) is an integral part of the quadcopter which regulates the power supply to the drone circuit. It has one end connected to the battery and another end to the power distribution board. But it also has a set of output wires which is connected directly to the pixhawk board through the dedicated slot as shown in the figure below.

2.3 Methodology

Understanding drone motor and propeller direction, along with the design, shows us how a drone works. Drones today, are very easy to fly in any direction. They can also hover in place smoothly. The engineering and design differ vastly to that of a helicopter or an aeroplane for flying.

So, flying process of quadcopter can be explained with the working of the motors and the rotating direction of propeller [24] attached to them as shown in Fig2.16(a) and (b), i.e., propeller design and motor thrust and their setup and the working of the sensors installed in the FC. The movement on the remote-control sticks sends information in the form of signals to the flight controller. This central flight controller sends this information to the ESCs of each motor, which in turn decreases or increases the rotating speed. This whole process can be depicted as:

Remote Control Sticks Movement →Central Flight Controller →ESC →Motors and Propellers →Drone movement or Hover.

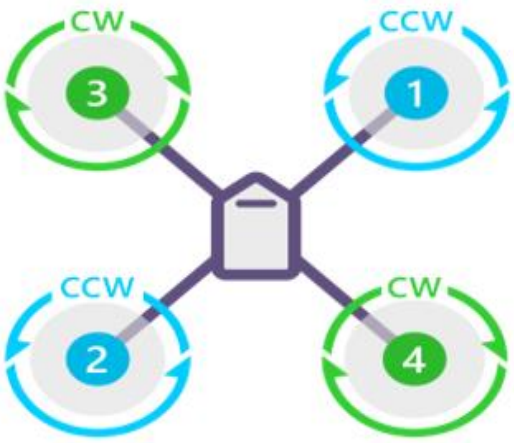


Fig2.15(a): Quadcopter motor rotation

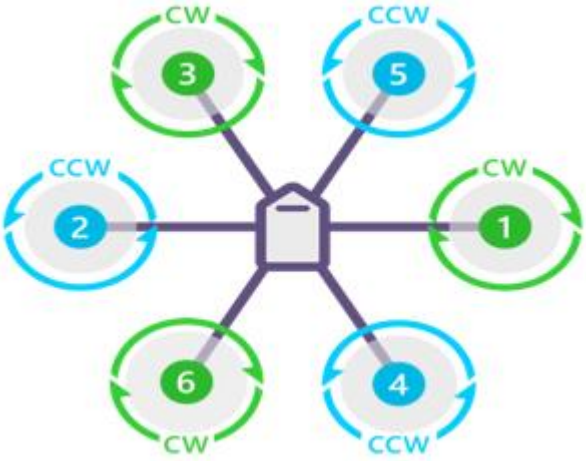


Fig2.15(b): Hexa-copter motor rotation

In the Fig2.16(a), one can see from the setup of motors in a quadcopter, that the alternate motors are rotating in one direction where the next to them motors are rotating in opposite direction to ensure the torques produced by each and every propellers cancels each other or in other words the net torque produced is zero and same can be said for the working of motors of hexa-copter. In order for the drone to rise in the air, a force must be created, which equals or exceeds the

gravitational pull on the quadcopter. The basic idea behind the aircraft-lift finally comes down to upward and downward force. So, based on the thrust produced by a drone, it can perform three actions in vertical plane: hover, climb and descend.

Before delving into the quadcopter motor and propeller setup, there are some key words [25] that one must keep in mind while building a quadcopter. These are known as Pitch, Roll and Yaw. Yaw is the basic movement to spin the quadcopter in clockwise or anti-clockwise direction. Pitch is the movement of the quadcopter either forward or to backward. Roll is the movement making the quadcopter fly sideways, either to right or left. People gets easily confused between Roll and Yaw movement so to get a clear idea of the all movements and the 3axes of movement can be known from the Fig.2.16(a), (b) and (c).

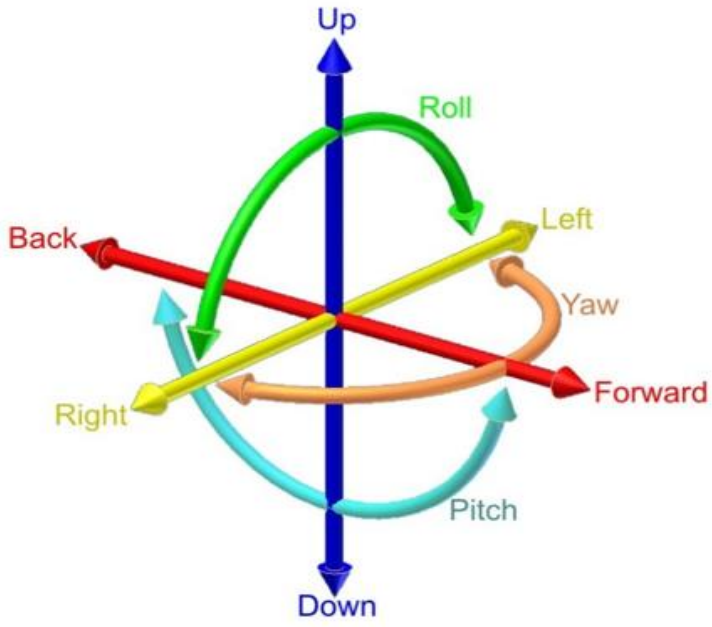


Fig2.16(a): Pitch, Yaw and Roll axes

Fig2.16(b): movement against Pitch, Yaw and Roll axes

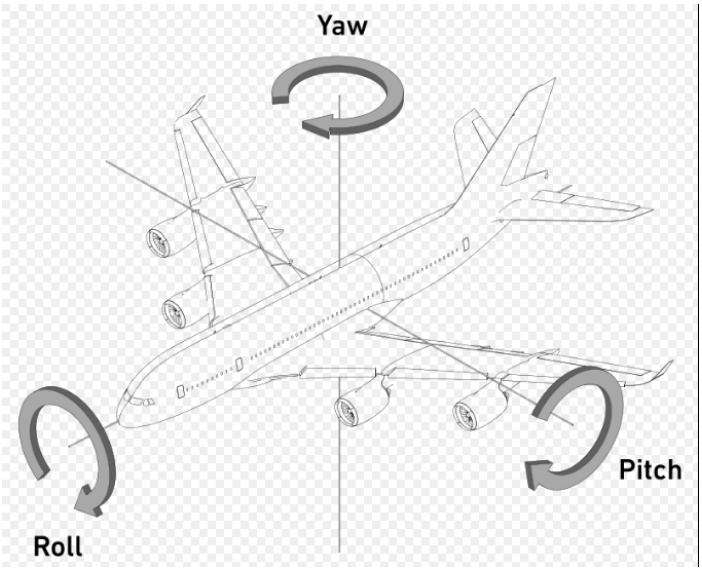


Fig2.16(c): Pitch, Yaw and Roll axis of a plane

Now, the question comes, how a quadcopter performs action according to Yaw, Pitch or Roll. the answer to this question comes from the different thrust produced by each and every propeller. The movement depending on the Roll-axis can be described as when the thrust produced by two right-most propellers exceeds to that of the to left-most motors, a net torque is produced by the thrust produced by the propellers which tilts the quadcopter towards left and similarly can be explained for hovering towards right by the thrust produced by different motors as shown in Fig2.20, 2.219(a) and (b).

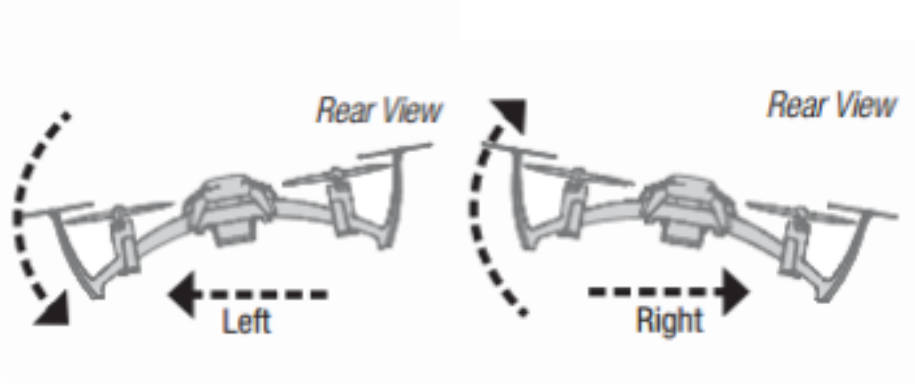


Fig2.17(a): left and right movement of a drone from reaer-view

Fig2.17(b): Leftward motion

Fig2.17(c): Rightward motion

The movement depending on the Pitch-axis can be described as when the thrust produced by two front-most propellers exceeds the thrust by two rear-most propellers then the quadcopter tilts towards and moves backwards. And the forward movement motion can be explained similarly.

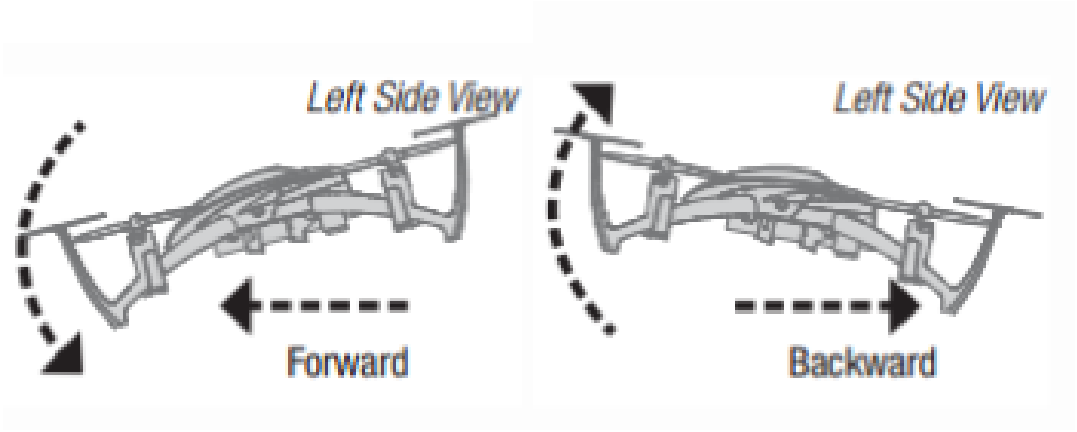


Fig2.18: Forward and Backward movement of a drone from Left-side view

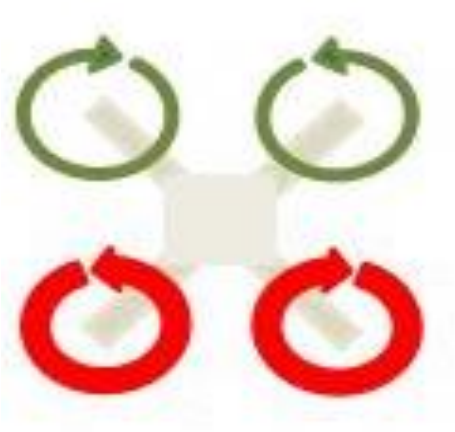


Fig2.19(a): Forward motion

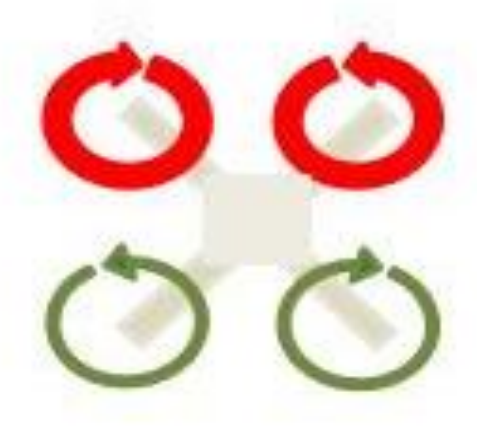


Fig2.19(b): Backward motion

And the motion based on Yaw-axis can be as when all the clockwise rotating propellers produce thrust greater than the thrust produced by all the anti-clockwise rotating propellers, then an angular velocity is generated by the net torque produced in clockwise direction and hence, the quadcopter rotates in clockwise direction.

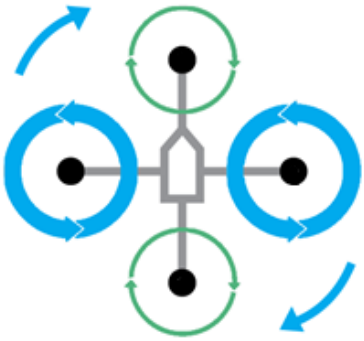


Fig2.20(a): Clockwise motion

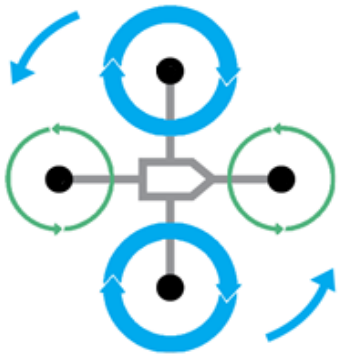


Fig2.20(b): Counter-clockwise motion

Fig2.20(c): Clockwise rotation

Fig2.20(d): Counter-Clockwise rotation

2.4 Software Setup

In this topic, we are going to see how a Pixhawk board is configured to fulfil our requirements. As Pixhawk board is an open-source project, we can use various configurators to configure our FC. Some of the configurators that can be used are QGroundControl, BetaFlight, or CleanFlight , etc. but we are going to be discussing about a software available for the same Mission Planner [26] (MP) , which is designed by the same company which designed Pixhawk board is also designed keeping Pixhawk and APM board in the view. With this software, we can change the variables and even PID values [27, 28] pre-set by the company to make the drone more stable while flying.

To connect the drone to a ground control system, for e.g. laptop, a mode of communication is set between them using a radio telemetry [29] called SiK Telemetry Radio. This telemetry set uses MAVlink protocol [30] for its communication. It consists two elements, the receiver and the transmitter. It has a range of (out of the box) better than 300m but can be changed by changing some PID values using MP. It is used to retrieve flight information of the drone to our laptop in order to follow several parameters of the drone on the ground.

The first step is to select the frame type and upload the firmware for the same that we are going to use, i.e., X-type Quadcopter.

Fig2.21: Choosing frame type in Mission Planner

After choosing the type and uploading the firmware, we need to calibrate the accelerometer sensor present in the FC so that The FC sets its reference point.

Fig2.22: Accel Calibration in Mission Planner

Then comes the part for compass calibration, in which the inbuilt compass aligns itself with the ever-present magnetic field of the earth for better navigation. One can use externally mounted GPS/compass to reduce the error and better compass alignment.

Fig2.23: Compass Calibration in Mission Planner

After compass calibration, comes radio calibration in which we need to check if all the channels present on the RC are functioning properly and each channel has different function. It is also used for the FC to get the max and min value of the sticks if they can be varied.

Fig2.24: Radio Calibration in Mission Planner

After Radio calibration, we need to set the Flight Modes [31]. Flight Modes is a feature available in which we can choose one of the transmitter sticks to perform each function based on its positions, like Stabilise, Alt. Hold, Autotune, Training, Land, etc. We can choose at-most three different flight modes available for our purpose.

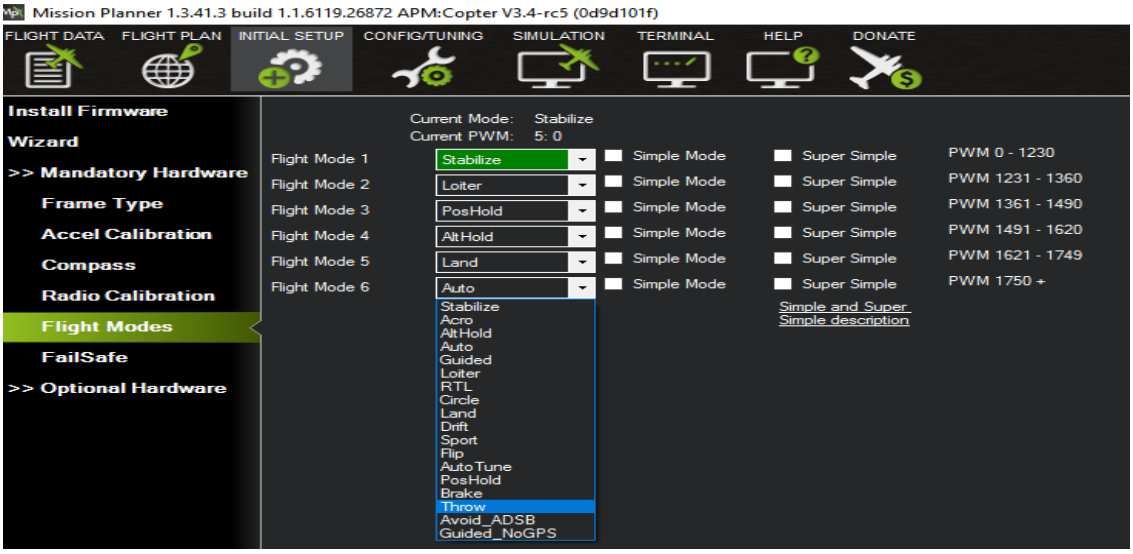


Fig2.25: Selecting Flight modes in Mission Planner

After setting different FlighModes, we set the FailSafe value of the battery, i.e., if the output voltage of the battery drops to a certain value, the drone performs a set of pre-loaded commands. It can be set even if you are not using the GPS, but the available set of commands that the drone can perform gets limited.

Fig2.26: Choosing Failsafe values in Mission Planner

After setting FailSafe value, we need to Calibrate all the ESC [32] we are using. ESCs are designed with different firmwares. The installed firmware decides the performance of the ESC. This gives information about which protocols it supports and what configuration interface can be used. There are different types of firmware defined for ESC, which are BLHeli, BLHeli_S, SimonK, KISS, BLHeli_32 and other manufacturer's own software.

The ESC protocols [33] are the language that the FC and ESC use to communicate, one of the basic tasks is to tell how fast the motor should be spinning. There are different protocols defined for ESCs like Analog PWM, Standard PWM, Oneshot125, etc.

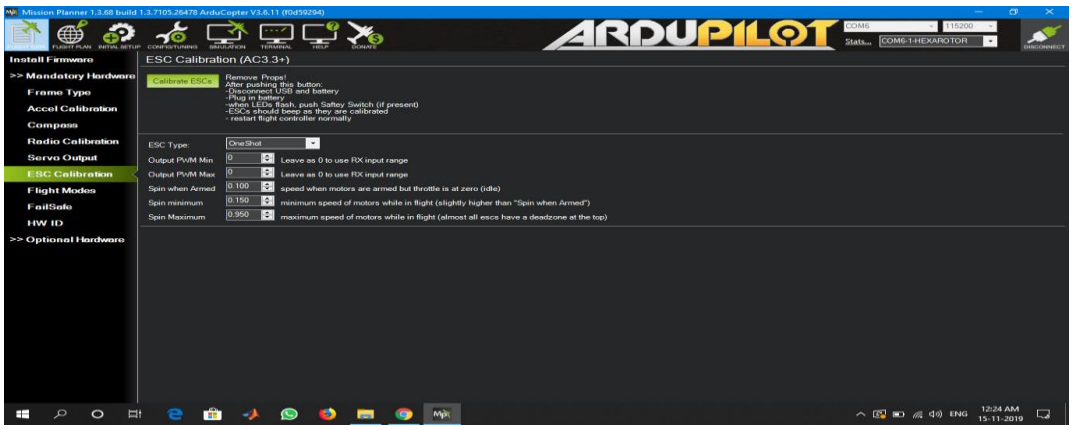


Fig2.27: ESC Calibration in Mission Planner

After all these processes, our quadcopter is ready to fly. But one can get the live status of the drone on the mission planner before or even while the drone is flying in MESSAGES part under the FLIGHT DATA section in MP.

Fig2.28: Message section

2.5 RESULTS

We have successfully achieved to fly our quadcopter using Radiolink AT10II transmitter with Pixhawk FC.

Fig2.29(a): Quadcopter from top-view

Fig2.29(b): Quadcopter from side-view

Fig2.30: Testing of quadcopter in MCTE

Fig2.31: Testing of hexa-copter in IIT Indore campus

Fig2.32: Hexa-copter during flight

Chapter 3: Object Detection

3.1 Introduction

It would be very easy for us if we get already designed algorithms and execute it, and get the required result. Maximum result accuracy and minimum effort. Isn't it what we look for in any field?

I feel very lucky that machine learning researchers embrace the open source technology that helps college students or concerned industries to use it for the fulfillment of their project need. But, it is also important to understand the concepts behind these algorithms. If we see broad picture and time constraint it is very handy and easy to use for a person or industry when research work has been done by our top tech researchers.

This is especially very true for Artificial Intelligence domains like computer vision. Not everyone has the computational knowledge and resources to build a model from scratch. That is where open source algorithms and pretrained models come into picture. And in this report, we will look at one such algorithm for object detection – YOLO (You Only Look Once) [34, 35] as shown in fig 3.1. It's a supremely fast and accurate technique, as we will see soon.

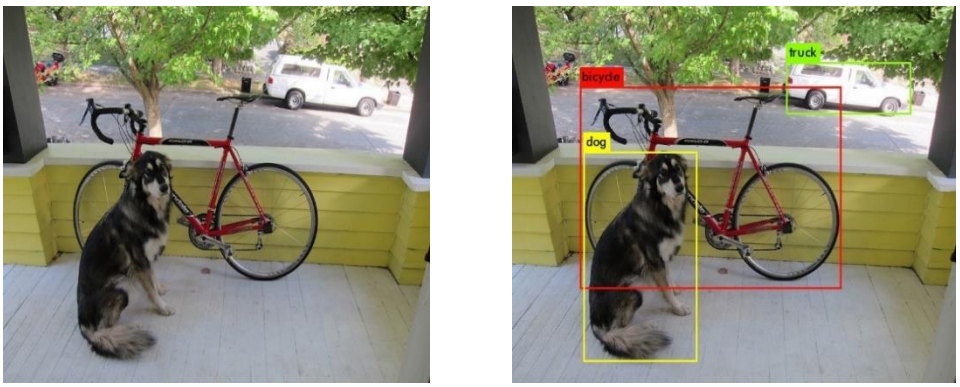


Fig 3.1: Object Detection

we will learn what makes YOLO algorithm easy to use, why we should use this over other object detection algorithms, and the different techniques used by YOLO algorithm. Once we have understood the concept and working principle, then we will then implement it in Linux environment.

3.2 What is YOLO and Why is it very useful?

Unlike other object detection techniques, YOLO algorithm deals with object detection in a much different way. YOLO will take the full image in a single example and predicts the class probabilities and bounding box coordinates for all the boxes. The main advantage of using YOLO algorithm is its high speed – it is extremely fast and can run over 45 FPS. It also represents the object in the frame which is related to known class.

This is one of the best algorithms for object recognition and detection and has shown a comparatively better output to the R-CNN techniques [36, 37, 38]. Now, we will see some of the different methods/techniques used in YOLO.

3.3 How does YOLO function?

We have discussed that why YOLO is such a useful algorithm, later we will understand the functioning of it. In this section, I have mentioned the major steps followed by YOLO for detecting objects in a given frame.

- It will take an input image from user as shown in fig 3.2.

Fig 3.2: Input image taken from our drone at IIT Indore campus

- The algorithm then divides the input image into SxS grids (S = 3) as shown in fig 3.3.

Fig 3.3: Input image divided into 3x3 grids

- Image localization and classification will be applied on every single grid. It then predicts the corresponding class probabilities and bounding box [39] for the objects, if found.

We will go through step by step to for more understanding of what we have just learned.

We need to pass labelled data to the model for the training purpose. Suppose we have divided the image into 3 X 3 matrix and there are total three classes which we want objects to be classified into. Let's say the classes we have Person, Car, and Motorbike respectively. So, for each grid, the label y will be defined as eight-dimensional as shown in table 3.1.

Y =	P_c
	B_x
	B_y
	B_h
	B_w
	C_1
	C_2
	C_3

Table 3.1: Eight-dimensional coordinate values for each grid

Here,

- P_c defines whether there is an object is present in the grid or not.
- B_x, B_y, B_h and B_w will specify the bounding box if there is an object present.
- C_1, C_2, C_3 will represent the detected classes. So, if the object found is a car, C_2 will be 1 and C_1 & C_3 will be 0, and so on.

We have selected the last grid from the above example as shown in fig 3.4:

Fig 3.4: Last grid image

Since there is no object found in this grid, P_c will be zero and y label for this particular grid as shown in table 3.2.

Y =	0
	?
	?
	?
	?
	?
	?
	?

Table 3.2: Eight-dimensional coordinate values for each grid

Here, '?' shows that it does not matter what all the coordinate values are, as there is no object found in the grid. We have taken another grid in which we have found a car ($C_2 = 1$) as shown in fig 3.5.

Fig 3.5: Bounding box on the car

Before going further and calculate y label, it is important to first understand the concept of YOLO. How it decides if there is an object found in the grid. In the input image, there are two

objects (two cars), so YOLO specify the center point of these two objects and these objects will be marked to the grid which contains the mid-point of these objects. The y label for the center right grid with the car as shown in table 3.3.

Y =	1
	B_x
	B_y
	B_h
	B_w
	0
	1
	0

Table 3.3: Eight-dimensional coordinate values for center right grid

Since we have found an object in the grid. So, P_c will be equal to 1. Rest x, y, h, w coordinates for bounding box will be calculated relative to the particular grid cell we are considering. Since car is the second class, $C_2 = 1$ and C_3 and $C_1 = 0$. So, for each of the 9 grids, we will have an eight-dimensional output vector. The output will have a shape of 3 X 3 X 8.

So now we have an input image and its corresponding target vector. Using the above example (input image – 100 X 100 X 3, output – 3 X 3 X 8), our model will be trained, as shown in fig 3.6.

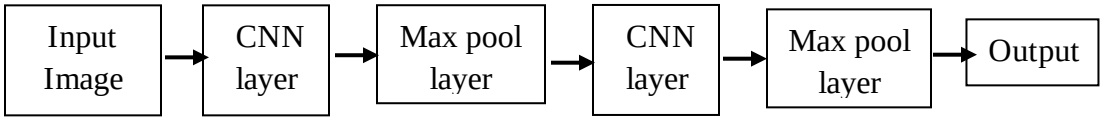


Fig3.6: Training Layers for YOLO model

We will run both forward and backward propagation to train our model. During the testing phase, we pass an image to the model and run forward propagation until we get an output y. In order to keep things simple, I have explained this using a 3 X 3 grid here, but generally in real-world scenarios we take larger grids (perhaps 19 X 19).

Even if an object spans out to more than one grid, it will only be assigned to a single grid in which its mid-point is located. We can reduce the chances of multiple objects appearing in the same grid cell by increasing the number of grids (19 X 19, for instance).

3.4 How to encode Bounding Boxes?

As we have discussed earlier, B_x, B_y, B_h and B_w are calculated relative to the grid cell we are dealing with. We will understand this concept with an example. Consider another grid which contains a car, as shown in fig 3.7.

Fig 3.7: Bounding box on the car in a grid

So, B_x, B_y, B_h and B_w will be calculated relative to this grid only. The y label for this grid as shown in table 3.4.

Y =	1
	B_x
	B_y
	B_h
	B_w
	0
	1
	0

Table 3.4: Eight-dimensional coordinate values for center grid

$P_c = 1$ since there is an object in this grid and since it is a car, $C_2 = 1$. We will see how to decide B_x, B_y, B_h and B_w. In YOLO, the coordinates assigned to all the grids, as shown in fig 3.8

(0,0)

(1,1)

Fig 3.8: Coordinates assigned for a particular bounding box

B_x, B_y are the x and y coordinates of the midpoint of the object with respect to this grid. In this case, it will be (around) $B_x = 0.4$ and $B_y = 0.3$ as shown in fig 3.9.

(0,0) B_x B_y

(1,1)

Fig 3.9: B_x and B_y value assigned for a particular bounding box

B_h is the ratio of the height of the bounding box to the height of the corresponding grid cell, which in our case is around 0.9. So, $B_h = 0.9$. B_w is the ratio of the width of the bounding box to the width of the grid cell. So, $B_w = 0.5$ (approximately). The y label for this grid as shown in table 3.5.

Y =	1
	0.4
	0.3
	0.9
	0.5
	0
	1
	0

Table 3.5: Eight-dimensional coordinate values for center grid

We have seen here that B_x and B_y will always range between 0 and 1 as the midpoint will always lie within the grid. Whereas B_h and B_w can be more than 1 in case the dimensions of the bounding box are more than the dimension of the grid.

In the next section, we will look at more ideas that can potentially help us in making this algorithm's performance even better.

3.5 Intersection Over Union & Non-Max Suppression

Here's some food for thought – how can we decide whether the predicted bounding box is giving us a good outcome (or a bad one)? This is where Intersection over Union [40] comes into the picture. It calculates the Intersection over Union of the actual bounding box and the predicted bonding box. Consider the actual and predicted bounding boxes for a car as shown in fig 3.10.

Fig 3.10: Intersection of actual bounding box and predicted box

Here, the blue box is the actual bounding box and the black box is the predicted one. How can we decide whether it is a good prediction or not? IoU, or Intersection over Union, will calculate the area of the intersection over union of these two boxes as shown in fig 3.11.

Fig 3.11: Yellow region is area of intersection

IoU = Area of the intersection / Area of the union, i.e.

IoU = Area of yellow box / Area of green box

If IoU is greater than 0.5, we can say that the prediction is good enough. 0.5 is an arbitrary threshold we have taken here, but it can be changed according to your specific problem. Intuitively, the more you increase the threshold, the better the predictions become.

There is one more technique that can improve the output of YOLO significantly – Non-Max Suppression [41].

One of the most common problems with object detection algorithms is that rather than detecting an object just once, they might detect it multiple times as shown in fig 3.12.

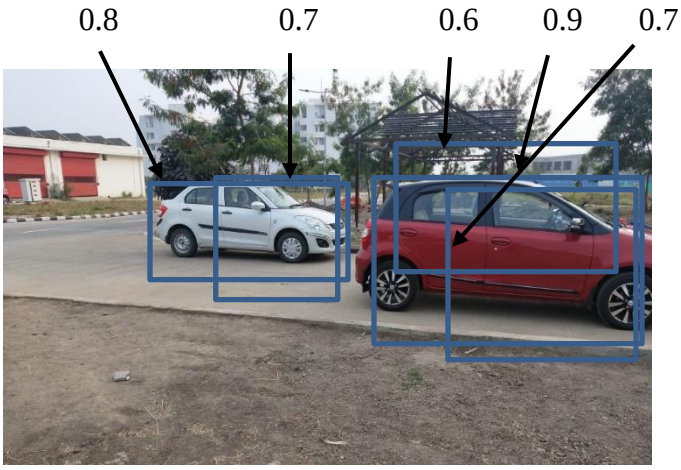


Fig 3.12: Predicted bounding boxes

As we can see cars are detected more than once. The Non-Max Suppression approach is responsible for cleaning the extra bounding boxes. We will see how this technique works

1. It will take the bounding box which have highest probability. Like in above image (fig 3.12), maximum probability is 0.9, so the corresponding box will be selected.
2. Now, it will go through all the other bounding boxes and select the box which the high IoU are compressed. Like in our example, bounding boxes with 0.6 and 0.7 are compressed.
3. After suppressing the bounding boxes, it will select the bounding box which have the next larger probability.
4. Then, it will look again at the IoU of the boxes and compress the bounding box having the high IoU.
5. We will repeat all the above steps until we get the until we get the final output as shown in fig 3.13.

Fig 3.13: Final bounding boxes

This is what Non-Max Suppression is all about. We are taking the boxes with maximum probability and suppressing the close-by boxes with non-max probabilities. We will quickly summarize the points which we've seen in this section about the Non-Max Suppression algorithm:

1. Discard all the boxes having probabilities less than or equal to a pre-defined threshold (say, 0.5).
2. For the remaining boxes:
    a. Pick the box with the highest probability and take that as the output prediction.
    b. Discard any other box which has IoU greater than the threshold with the output box from the above step.
    c. Repeat step 2 until all the boxes are either taken as the output prediction or discarded.

3.6 Creating the Dataset to train new classes

A training set for YOLO consists of a series of images, each one will come with a text file indicating the co-ordinates and the class of each of the objects found in the image.

Unfortunately, right now there seems to be no standard look on how to represent this information. In other words, each of the framework expect a different format to represent the bounding box and class of each object:

- Darknet uses a text file, with the following format:

[category number] [object center - X] [object center -Y] [object width -X] [object width - Y]

3.7 Tools to create training Dataset

There are several tools that can be used to create the coordinates for each one of the images that will be part of the training set. This means, to manually identify the bounding box containing each one of the objects in the image and indicate it to the corresponding class as fig 3.14.

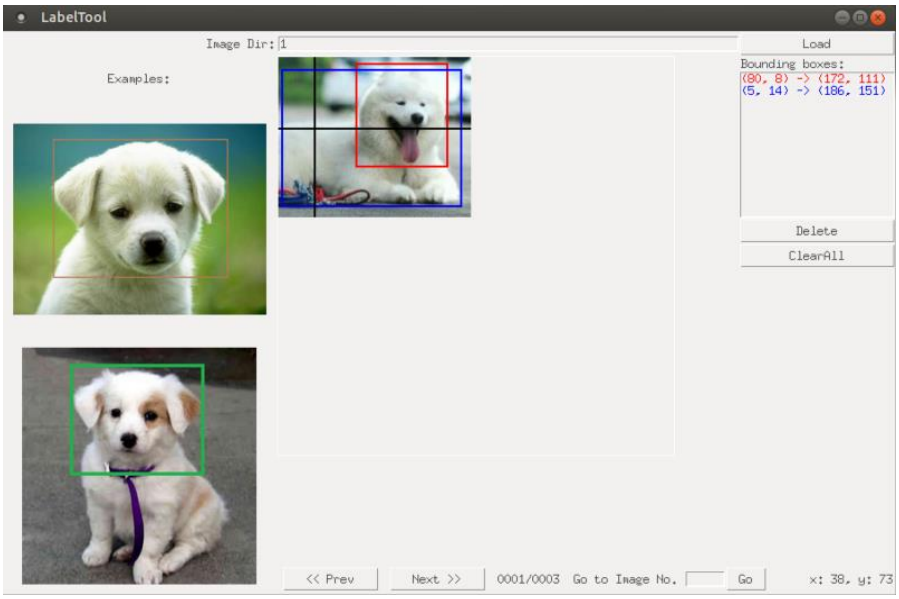


Fig3.14: Tool to create dataset

- **Yolo Mark**: Created by the same people behind the Windows fork. Saves the annotations in the format expected by Darknet [42].
- **LabelImg**: Saves the annotations in the PASCAL VOC format, the one used natively by Darkflow [43].
- **BBox-Label-Tool**: Saves the annotations in a format that as far as I know is not natively used by any of the YOLO implementations, besides, the master branch does not allow multiple classes per image [44].

3.8 Implementation of YOLO

Currently we know three implementations of YOLO each of them has advantages and disadvantages as follows:

- **Darknet**: This is the official implementation, created by the same researchers behind the algorithm. It was written in C with CUDA enabled, hence it does support GPU computation. It is a complete neural network framework, so it really can be used for other tasks besides YOLO detection. The disadvantage is that, since it is written from the scratch

(not based on established neural network framework) it will be more difficult to find answers for errors we might found [45].

- **AlexeyAB/darknet**: Here I am actually cheating a little bit because it is actually a fork of Darknet to support Windows and Linux environment. I have also used this one to check its compatibility with my system and also for the variation in outputs, I have checked the README many times, it is an excellent source to find tips and recommendations about YOLO in general, how to prepare you training set, how to train the network etc [46].
- **Darkflow**: This is the port of Darknet to work over TensorFlow. This is the system I have also implemented before, mainly because I started this project without having a GPU to train the network and apparently using CPU-only Darkflow is several times faster than the original Darknet. As far as I know, the main disadvantage is that it has not been updated to YOLOv3 [47].

All these implementations come "ready to use", which means you only need to download and install them to start detecting images or videos right away using already trained weights available to download. Naturally this detection will be limited to classes contained in the datasets used to obtain these weights.

3.9 Results

We have successfully achieved object detection in images and real-time videos. These are some of the outputs we have got with YOLO algorithm. Some of the input videos and images have taken from our newly built drone.

(a) (b)

Fig 3.15: (a) and (b) are the inputs for object detection

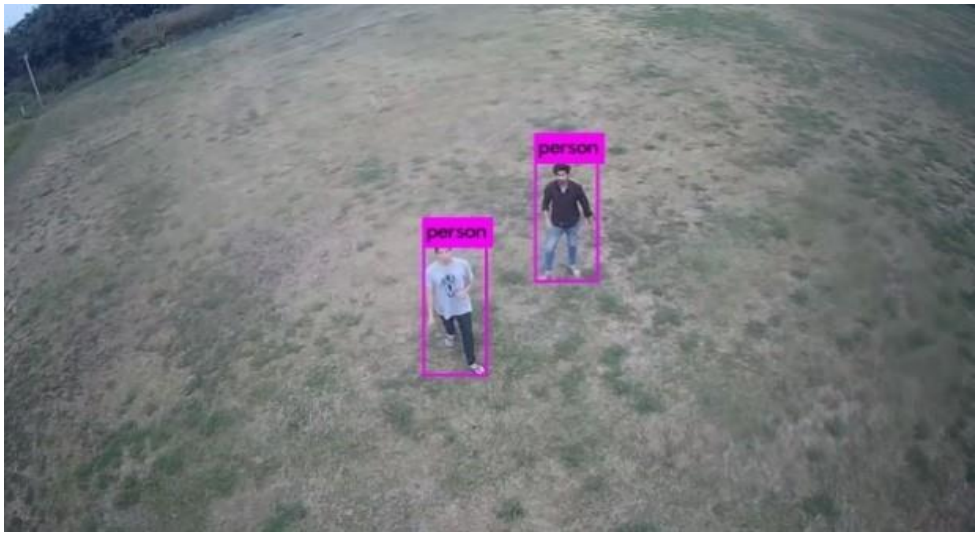
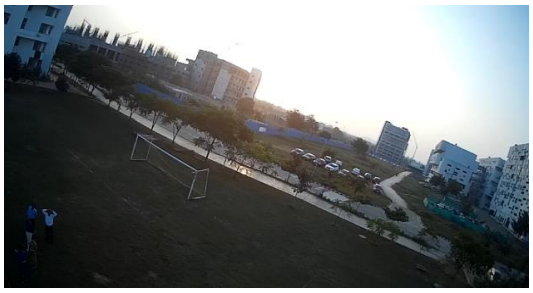


Fig3.16: Object Detection output for (a)

Fig 3.17: Object Detection output for (b)



(c) (d)

Fig 3.18: (c) and (d) are the inputs for object detection

Fig 3.19: Object detection output for (c)

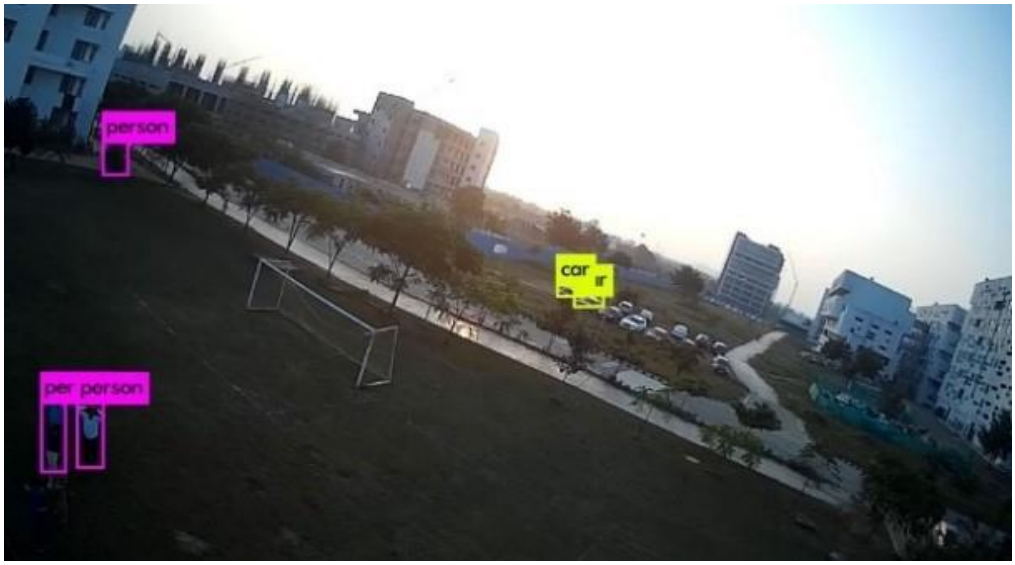


Fig 3.20: Object detection output for (d)

Chapter 4: Object Tracking

4.1 Introduction

We are going to implement video object tracking on a target given by the user. Yes, this can be achieved by the algorithm we are going to use by making a bounding box on the target, and we can effectively track the target in real-time video.

Inspired by the observations from human perception, in this chapter, we will discuss about the two-stage object tracking method driven by prediction. Given a tracking output in time t, we first predict the approximate object location in the next frame (t+1) without seeing it. Based on the prediction outputs, we then further refine the localization as well as the segmentation outputs by using the appearance input in time t+1. The refined tracking and segmentation outputs can help us back to update a better prediction model, which will be applied again in the upcoming frames.

Object Tracking is the process of locating an object by creating a bounding box on it, and the bounding box will follow the trajectory of the target given by the user. It has number of uses – some of which are human-computer interaction, traffic control, surveillance etc. We are going to implement the object tracking algorithm with our newly designed drone.

4.2 Approach

To continuously follow the trajectory of the target given by user, we are going to implement a fully convolutional neural network algorithm called SiamMask [48]. In video object tracking, most methods don't consider the time continuity of object motion. In other words, most methods predict a zero-velocity-object and calculate according to the region centered on the last estimated position of the target [49, 50, 51, 52].

Inspired by human perception, we set up a new search region in the coming frame centered at the predicted object position. We project the estimated object center position back to the pending detection frame.

To make the SiamMask algorithm suitable for tracking task. We have used the VOT-2016 [53] dataset for the automatic bounding box generation as it offers the highest IOU as reported in [52].

4.3 Experiments

In this section, we will discuss our approaches on visual object tracking (VOT 2016 and VOT 2018). It is worth noticing that our method does not depend on the selection of tracking. To

better evaluate the efficiency of our method, we adopt SiamMask [52] with provided pretrained model as our online tracking.

We will see the algorithms that we have tested for our object tracking purpose as follows:

4.3.1 SiamFC

SiamFC [54] converts the tracking problem into a similarity learning problem, for which a fully convolutional Siamese network is trained to locate an exemplar image within a larger search image. The network compares an exemplar image z (initial appearance of the object) to a condidate image x of the similar size and returns a better score if the two images depict the same output image and a low score otherwise.

For training, pairs are obtained from a dataset of annotated videos by extracting exemplar and search images that are centered on the target. The images are extracted from the two frames of a video that contain the object and are at most T frames apart. The elements of the score are considered to belong to a positive instance if they are within radius R of the center. Note the network is learned offline and **does not** update via tracking.

Below is the neural network architecture of SiamFC [54] as shown in fig 4.1.

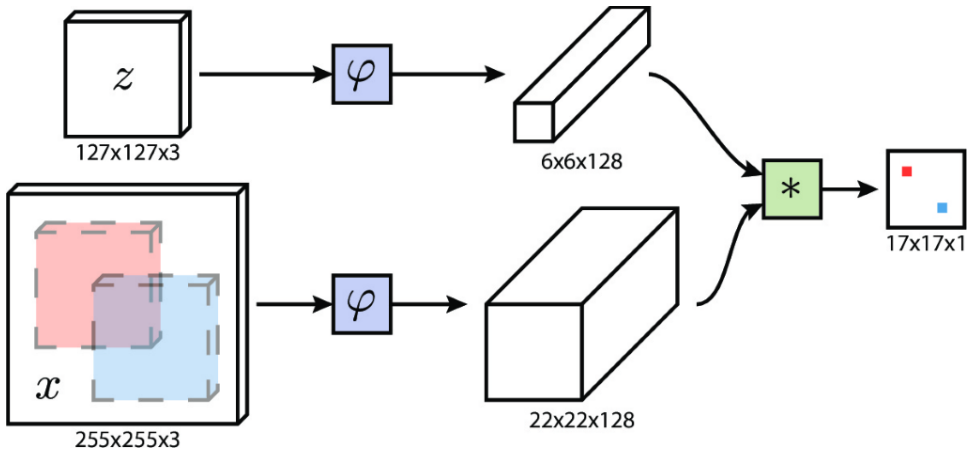


Fig 4.1: Architecture of SiamFC

4.3.2 SiamRPN

SiamRPN [55] focuses on the speed of the neural network-based tracking algorithm. It consists of Siamese framework for feature extraction and region proposal subnetwork including the template and detection branch. The proposed framework is formulated as a local one-shot object detection task, where the bounding box in the first frame is the only example.

Similar as Faster-RCNN, the template branch predicts background and the regression branch compute the bounding box offset for each of the k frames. The template branch of the Siamese subnetwork is pre-computed and the correlation layers (denoted as star) as trivial convolution layers is used to perform online tracking.

Below is the neural network architecture of SiamRPN as shown in fig 4.2.

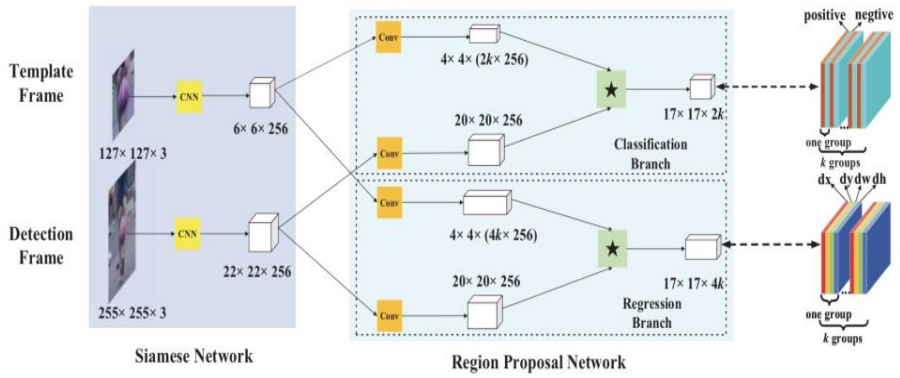


Fig 4.2: Architecture of SiamRPN

4.3.3 SiamMask

SiamMask [52] calculates the problems of visual tracking and visual object segmentation as a joint learning of three tasks:

- To learn a measure of similarities between the target and multiple candidates in a sliding window fashion.
- Bounding box regression using a Region Proposal Network.
- Class-agnostic binary segmentation: binary labels are only required during offline training to compute the segmentation loss and not online during tracking.

Once trained, SiamMask solely relies on a single bounding box initialisation, operates online without updates and produces rotated bounding boxes at 55 frames per second.

Below is the neural network architecture of SiamMask as shown in fig 4.3.

4.4 Implementation of SiamMask

We are going to implement the SiamMask algorithm in Linux environment. To run this algorithm, we need a highly configured computer system with GPU and CUDA enabled, as it is based on real time video object tracking.

Following are the steps to implement SiamMask:

1. Install Anaconda

   wget https://repo.continuum.io/archive/Anaconda3-5.3.0-Linux-x86_64.sh

   bash Anaconda3-5.3.0-Linux-x86_64.shsource ~/.profile

2. Clone the repository and build the environment and benchmark toolkits

   git clone https://github.com/xl-sr/THOR.git

   cd THOR

   conda env create -f environment.ymlconda activate THOR

   bash benchmark/make_toolkits.sh

3. Download the SiamMask model

   wget http://www.robots.ox.ac.uk/~qwang/SiamMask_VOT.pth

   mv SiamMask_VOT.pth trackers/SiamMask/model.pth

4. Get the datasets

   cd data/

   bash get_test_data.sh

5. Run your webcam

   python webcam_demo.py --tracker SiamRPN

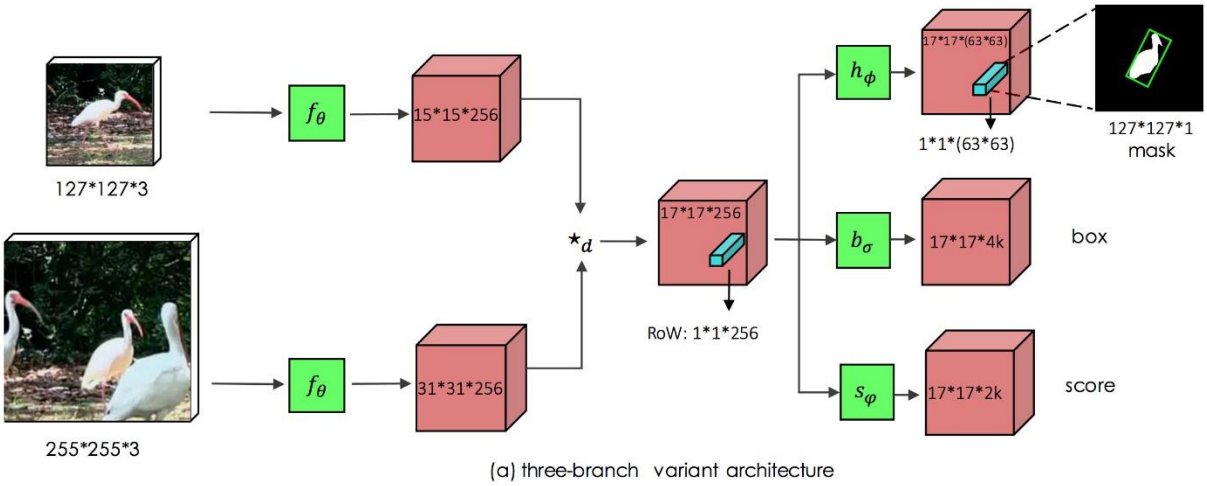


Fig 4.3: Architecture of SiamMask

4.5 Results

We have successfully achieved video object tracking on real-time videos. These are some of the outputs we have got with SiamMask algorithm. All the input videos are taken from our newly designed drone.

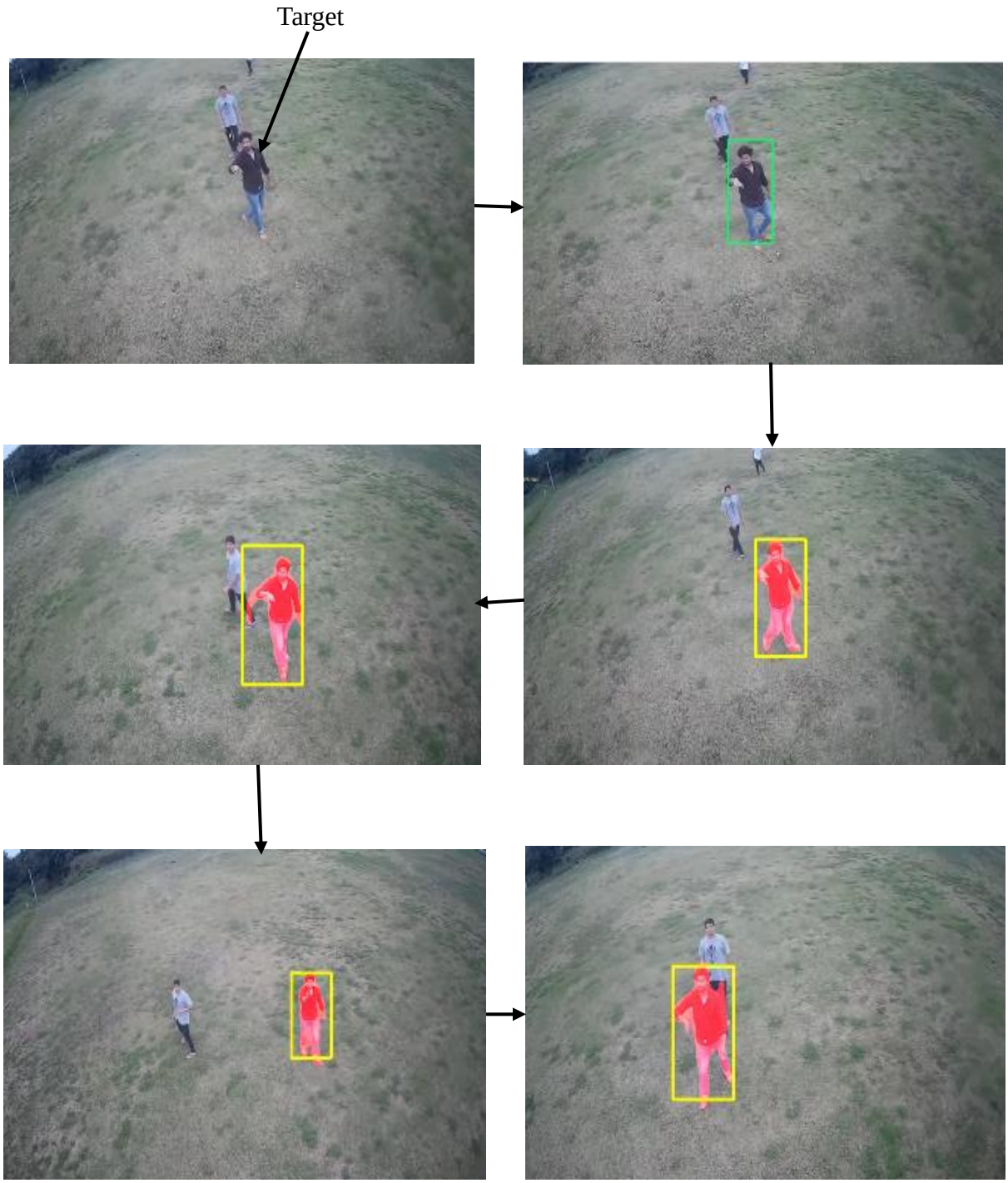


Fig 4.4: Object tracking when drone height is more than 50m

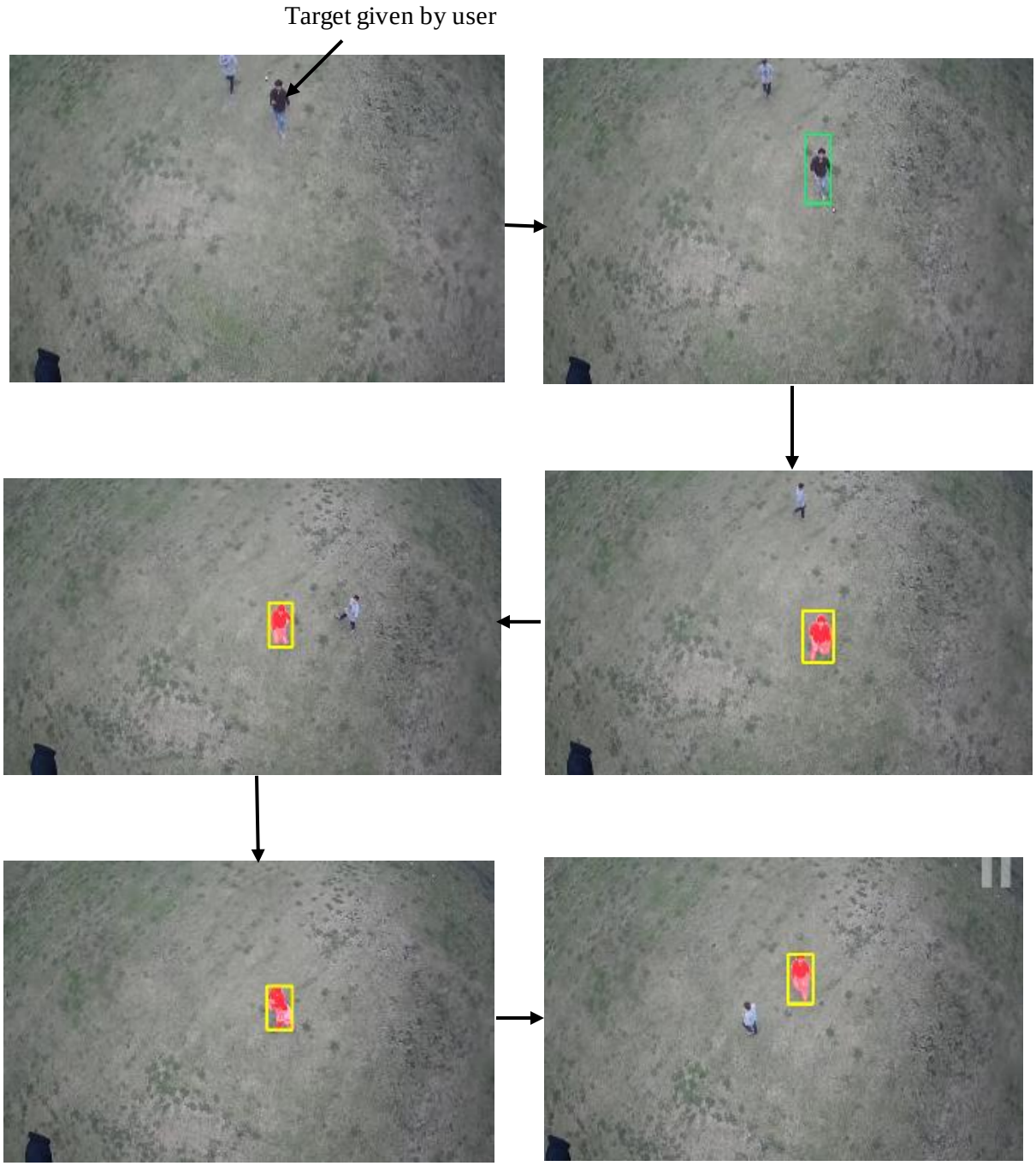


Fig 4.5: Object tracking when another object clashes the target. Here, the bounding box is not get replaced by another object different from the target.

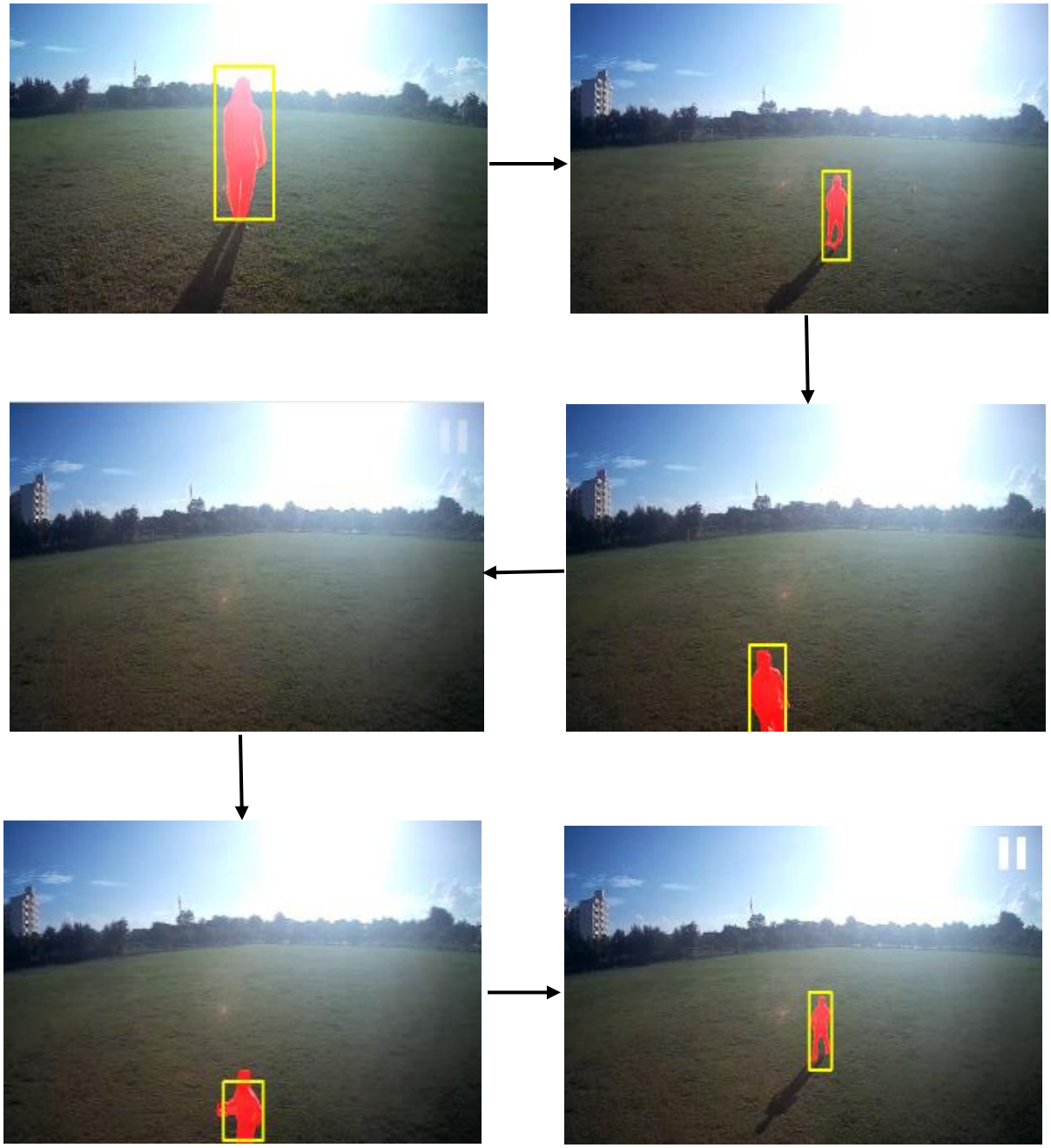


Fig 4.6: Object tracking when target goes out of the frame and comes back to frame. It is still tracking the same target when object comes back to the frame.

Chapter 5: Conclusions

In this report, we have overviewed the following points:

- We have developed the drone with real-time aerial detection and tracking ability. We have achieved the graceful motion and accurate altitude hold performance of drone while controlling from the ground.
- We were also able to achieve 2-axis movement of the gimbal mounted on a drone to set the target at the center of the frame of camera.
- Several features of our drone:
    - It can be controlled with a transmitter from ground.
    - Movement of the camera can be controlled from the ground.
    - Effective air to ground communication.
    - Real-time output for object detection and tracking.
- Drone is also capable of sharing the images and videos in real-time with ground computing and control unit with the help of compatible FPV transmitter and receiver.
- Images and videos are processed with machine learning algorithms for detection and tracking purpose.
- We have got high accuracy for object detection and tracking tested on COCO and VOT 2016 (better Intersection over Union) dataset respectively.

References And Image Sources

1. https://en.wikipedia.org/wiki/History_of_unmanned_aerial_vehicles
2. https://en.wikipedia.org/wiki/Kettering_Bug
3. https://en.wikipedia.org/wiki/%C3%89tienne_Oehmichen
4. https://en.wikipedia.org/wiki/De_Bothezat_helicopter
5. https://www.dehavillandmuseum.co.uk/aircraft/de-havilland-dh82b-queen-bee/
6. https://en.wikipedia.org/wiki/History_of_unmanned_aerial_vehicles
7. https://www.usatoday.com/story/money/2019/06/05/amazon-drone-delivery-shopping-giant-unveils-new-prime-air-drone/1358260001/
8. Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-Learning-Detection. In IEEE transactions on pattern analysis and machine learning, VOL. 6, NO. 1
9. Dorin Comaniciu. Mean Shift: A Robust Approach Towards Features Space Analysis. In IEEE transactions on pattern analysis and machine learning, VOL. 24, NO. 5, May 2002
10. https://en.wikipedia.org/wiki/Ku%E1%B9%AD%E1%B9%ADaka
11. http://www4.comp.polyu.edu.hk/~cslzhang/CT/eccv_ct_camera.pdf
12. Design of All-Terrain Rover Quadcopter for Military Engineering Services Shantanu Chaudhary, Arka Prava, Neha Nidhi, Vijay Nath. Lecture Notes in Electrical Engineering book series (LNEE, volume 511)
13. Building a quadcopter: An approach for an Autonomous Quadcopter. VNV Aditya Sharma ; M Rajesh. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)
14. Self-Navigating Quadcopter. O Guneshwor Singh / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (3), 2015
15. http://ardupilot.org/copter/docs/common-leds-pixhawk.html
16. http://ardupilot.org/copter/docs/common-sounds-pixhawkpx4.html#common-sounds-pixhawkpx4

17. https://www.getfpv.com/motors/what-is-a-drone-motor.html

18. Stabilizing and control of tilting-rotor Quadcopter in case of a propeller failure. Alireza Nemati, Rumit Kumar, Manish Kumar. Proceedings of the ASME Dynamic Systems and Control Division DSCC 2016

19. A low cost fully autonomous GPS (Global Positioning System) based quad copter for disaster management. HimadriNath Saha, Srijita Basu, Supratim Auddy, Ratul Dey. Arnab Nandy, Debjit Pal, Nirjhar Roy, Subhadeep Jasu, Ankita Saha, SoummyoPriyo Chattopadhyay, Tamanna Maity. 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)

20. https://www.getfpv.com/learn/new-to-fpv/all-about-multirotor-fpv-drone-radio-transmitter-and-receiver/

21. Experimental Validation of an Altitude Control for Quadcopter. Zaki Mustapa, Shakir Saat, A. M. Darsono and H. H. Yusof. ARPN Journal of Engineering and Applied Sciences.

22. Simple GUI Wireless Controller of Quadcopter Dirman Hanafi, Mongkhun Qetkeaw, Rozaimi Ghazali, Mohd Nor Mohd Than, Wahyu Mulyo Utomo, Rosli Omar. Int. J. Communications, Network and System Sciences, 2013

23. Visual tracking and control of a quadcopter using a stereo camera system and inertial sensors. 2009 International Conference on Mechatronics and Automation.

24. https://www.dronezon.com/learn-about-drones-quadcopters/how-a-quadcopter-works-with-propellers-and-motors-direction-design-explained/

25. https://emissarydrones.com/what-is-roll-pitch-and-yaw

26. https://www.youtube.com/watch?v=I8ugqKsMIpE

27. Design of Control System for Quadcopter using Complementary Filter and PID Controller. V.P. Kodgirwar, Vivek Kumar, Manish Shegokar, SushantSawant. International Journal of Engineering Research & Technology (IJERT), 2014

28. Modelling and Simulation of Quadcopter using PID Controller. Viswanadhapalli Praveen* and Anju S. Pillai. IJCTA, 9(15), 2016

29. http://ardupilot.org/copter/docs/common-sik-telemetry-radio.html

30. http://ardupilot.org/dev/docs/mavlink-basics.html

31. http://ardupilot.org/plane/docs/flight-modes.html

32. http://ardupilot.org/copter/docs/esc-calibration.html?

33. https://robu.in/what-is-oneshot-and-multishot-in-esc-difference-between-oneshot-and-multishot-esc-esc-calibration-protocol/

34. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection

35. https://pjreddie.com/darknet/yolo/

36. Rich feature hierarchies for accurate object detection and semantic segmentation. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik. Cornell University. CVPR 2014.

37. Ross Girshick. Fast R-CNN. Cornell University

38. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Cornell University

39. https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

40. Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, Silvio Savarese. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. Cornell University.

41. https://www.vision.ee.ethz.ch/publications/papers/proceedings/eth_biwi_01126.pdf

42. https://github.com/AlexeyAB/Yolo_mark

43. https://github.com/tzutalin/labelImg

44. https://github.com/puzzledqs/BBox-Label-Tool

45. https://pjreddie.com/darknet

46. https://github.com/AlexeyAB/darknet/

47. https://github.com/thtrieu/darkflow/

48. Fast Online Object Tracking and Segmentation: A Unifying Approach

49. Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In European conference on computer vision, pages 850–865. Springer, 2016.

50. Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8971–8980, 2018.

51. Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In European Conference on Computer Vision, pages 103–119. Springer, 2018.

52. Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. arXiv preprint arXiv:1812.05050, 2018.

53. Matej Kristan, Jiři Matas, Aleš Leonardis, Michael Felsberg, Gustavo Fernández, and et al. The visual object tracking vot2016 challenge results. In Proceedings of the European Conference on Computer Vision Workshop, pages 777–823. Springer International Publishing, 2016.

54. https://www.robots.ox.ac.uk/~luca/siamese-fc.html

55. http://openaccess.thecvf.com/content_CVPR_2019/papers/Li_SiamRPN_Evolution_of_Siamese_Visual_Tracking_With_Very_Deep_Networks_CVPR_2019_paper.pdf