



**Senior Design Project Aero-497**

# Autonomous assembly of aircrafts midflight

## **Submitted by:**

Anees Peringal: 100045872

Abdulrahman Al Nuaimi: 100045412

Mohammed Al Yammahi: 100044788

Saeed Al Seriedi: 100044784

Sultan Al Hosani: 100045419

## **Project Supervisors:**

Dr Sean Swei

Dr Yahya Zweiri

## **Abstract**

The project aims to demonstrate the feasibility of assembling two aircrafts midflight. This is demonstrated by using standard form factor hexacopters in an Indoor setting. Aircrafts with large aspect ratio usually have a higher endurance. Manufacturing of such aircrafts is difficult due to the large space that is required. The project may see applications in manufacturing of such aircrafts or even in fault tolerant rotorcrafts due to the extra actuators after assembly. The Project uses Optitrack for localization and PX4 as the flight controller. We were able to demonstrate assembly with a stationary target in some limited conditions.

## Table of Contents

Abstract .....	2
Acknowledgements .....	<b>Error! Bookmark not defined.</b>
Chapter 1: Introduction .....	7
1.1 Goals and objectives .....	7
1.2 Motivation .....	7
1.3 Methods .....	8
1.4 Overview of the technical area .....	8
1.5 overview of the report .....	10
Chapter 2: System Specifications .....	10
2.1 Requirements .....	10
2.2. System Specification .....	12
The structure team system specifications: .....	12
Chapter 3: Theoretical Foundations .....	13
Introduction .....	13
Standards and regulations 3.2 .....	14
National flying regulations 3.2.1 .....	14
Design standards 3.2.2 .....	14
3.2 details of the theoretical model .....	14
State estimation .....	14
3.3.2 Discrete or other approximations .....	15
3.3.3 Limitations and assumptions .....	16
3.2 details of the theoretical model .....	16
State estimation .....	16

Controller Architecture .....	18
Chapter 4: concept Generation and design evaluation.....	19
4.1 different design options.....	19
4.2 Different design option .....	21
Docking system.....	23
4.3 Key design constraints .....	27
4.5 Assessment of design.....	28
Final System.....	30
4.3 Simulation .....	30
Gazebo Simulation.....	30
Chapter 5: Implementation and Fabrication .....	31
Hexacopter setup.....	31
Electrical connections .....	32
Data Streams .....	33
ROS configuration .....	34
Chapter 6: Testing.....	35
6.1 Verification .....	35
6.2 Validation.....	35
6.3 Evaluation .....	37
Chapter 7. Conclusion.....	37
7.1 Summary of work done.....	37
7.2 Conclusion .....	37
7.3 critical appraisal of work done.....	38
7.4 Recommendation for further work.....	38
Appendices.....	39

References .....	39
Software Listings .....	40
List of Tables .....	40
List of figures.....	41
Code: .....	41
Meeting notes.....	52

## **Acknowledgements**

We would like to express our gratitude towards Dr Sean Swei and Dr Yahya Zweiri for helping and guiding us throughout the year. Moreover, our appreciation goes to the researchers Ahmed Humais and Osama for their assistance with the code, and Mr Bittu Scaria for helping us print the parts.

We appreciate the department of Aerospace engineering for providing us the access to use the robotics lab which is a great and very safe environment to fly the drones, as well as their permission to use the tools needed.

# Chapter 1: Introduction

## 1.1 Goals and objectives

- To investigate the feasibility of autonomously assembling two UAVs.
- To integrate a large-scale air vehicle that can be assembled in mid-air.
- Develop and frequently test the docking mechanism to see how it performs.
- Analyze stability and flight controls during proximity maneuvers.
- Study the collaboration aerodynamic performance.

## 1.2 Motivation

With the development of technology, in recent years, unmanned aerial vehicles (UAVs) have been widely used as they attract extensive attention due to their flexibility and small size. With the rising accessibility of drones, many of the most dangerous and high-paying jobs within the commercial sector are ripe for displacement by drone technology [1]. The use cases for safe, cost-effective solutions range from data collection to delivery. And as autonomy and collision-avoidance technologies improve, so too will drones' ability to perform increasingly complex tasks.

The Aerospace department of Khalifa university and their appreciated efforts in offering such a great working environment from advanced and pretty safe labs not to mention how series risk and precious's procedure and polices is being taking care of to eliminate any expected hazard.

Moreover, in recent years, defenses companies around the world have focused heavily on developing drones as it might be the future of air battles. Thankfully, the UAE is host to one of the most prestigious and advanced companies (not to mention start-up local companies) like Lockheed and Raytheon technologies, this can only mean that the UAE has an ambitious to lead the transformation of the global aerospace industry.

The fundamental aerodynamics of a flying wing suggests that its aerodynamic efficiency improves as aspect ratio increases. However, fabricating, building, and assembling high aspect ratio air vehicles can be a challenge, especially when the advanced ultra-light materials are used for aerostructures. The Idea is to assemble any number of such smaller unmanned air vehicles together to create one large single UAV, a way to clear the confusion, think of it as assembling a hovering Lego parts together and create a big "spider net" out of drones. we can perform verity of task like increasing the payload(cargo) capacity.

### **1.3 Methods**

Our approach is to work with a “DJI f500” to be a stationary drone that would stay still(hovering) at a certain height with one of the docking arms attached to it. In addition, this specific docking mechanism would have an external wall assistance shaped like a cone, to guide the other arm to find its way in and get locked. Moreover, Using ROS software along with Gazebosim, the second drone would have the harder task where it will approach and attach to the Stationary drone.

### **1.4 Overview of the technical area**

A project that is related is the ModQuad, a novel flying modular robotic structure that can self-assemble in mid-air and cooperatively fly [2]. The project was developed in university of Pennsylvania. The structure is composed of agile flying modules that can easily move in all directions [2]. The module is based on a quadrotor platform within a cuboid frame which allows it to connect to other modules by matching vertical faces. Using this mechanism, a ModQuad swarm can quickly assemble flying structures in mid-air using the robot bodies as building units [2]. There are two important tasks for modular flying structures. First, they propose a decentralized modular attitude controller to allow a team of physically connected modules to fly cooperatively. Second, they develop a docking method that drives pairs of structures to be connected in mid-air [2]. finally, method precisely aligns, and corrects motion errors during the docking process they made. In their experiments, they analyzed and tested the performance of the cooperative flying method for multiple configurations. They also tested the docking method and they had successful results.

Another research that has been done to investigate the optimal gap between two propellers. Simulations were made for lifting force determination and gap distance optimization on the base of a simplified quadrotor helicopter model with 254 mm propellers [3]. They did separate simulations for four different rotation speeds 1500, 3000, 4000 and 5000 rpm. For each rotation speed, the distance between the two propellers changes from 5 mm to 140 mm (Figure 1). Results of the simulations illustrate that the lifting force produced by one propeller in quadrotor increased



on distances from 5 mm to 35 mm (Figure 1) by about 15%. Whereas at 70 mm, the lifting force will decrease by about 2% and then will become stagnant [3].

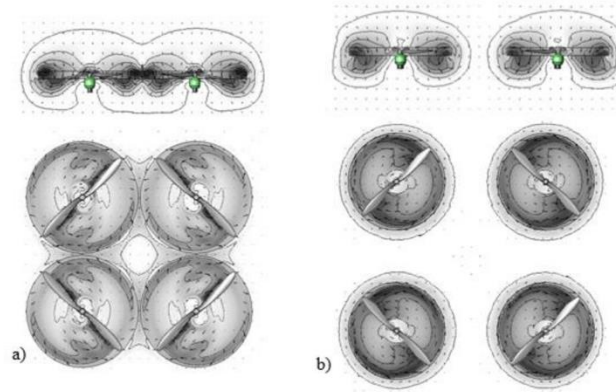


Figure 1: turbulent flow from Propeller

(Figure2) Shows the stream is partially compensating the lifting force. (a) Small turbulent areas appeared near the propeller edge where the air flow was twisting upwards. There was a small space between the propellers where airflows were running into each other and the resulting flow moved upwards. (b) illustrates airflow velocities and their directions when the distance between the propellers was about 140 mm. At this separation distance, the influence of airflows from the propellers was insufficient. Each propeller can be considered as separately working and with total lifting forces. [3]

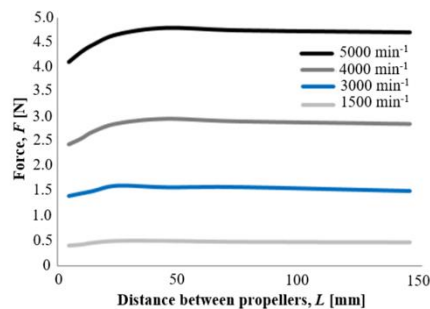


Figure 2: relation between lifting force and distance between propellers

They measured for a quadcopter with propeller of 254 mm diameter and pitch of 127 mm and total mass of the UAV equal to 1 kg, the optimal gap between propellers would be about 33 mm. “If the mass of the same design of UAV is equal to 2.5 kg, then the optimal gap would be 50mm.” [3]

So comparing with our UAV which is different in shape and design, but has a mass of 2.6 kg and with propeller 304.8mm x 106.68mm the optimal gap can be about 65mm or more.

## **1.5 overview of the report**

This report covers the processes that investigate the feasibility of autonomously assembling two UAVs by initially creating the right code to move the drone and maneuver, Afterward, the report will cover the finalized docking mechanism and the idea behind the locking part and how it works. How we developed and enhanced the docking mechanism throughout the year from a heavy arm with a rotating device to a much simpler and lighter locking mechanism.

## **Chapter 2: System Specifications**

### **2.1 Requirements**

- Design a connection arm for each UAV.
- Create a program that allows the actuator integrated within the lock to operate.
- Make sure that the 2 UAVs hover at a specific point.

Our project has different parts that the team needs to work on to build it, and we can group them into two groups structures and control. Structure components include parts such as the material and docking mechanism. Moreover, control deal with ROS, mavlink, Optitrack and coding. The basic requirements for this system to operate are two UAVs that need to be connected simultaneously while airborne. So, a connecting arm is being developed to achieve this concept. Moreover, during the development process, ROS is used as a middleware to make the two drones hover simultaneously and making the connecting arms lock during the process when connected. In addition, Optitrack compatible markers have been fitted on one of the drones to specify the location of our rigid body and we defined the other mechanism system as a rigid body using the same Optitrack compatible markers to get the coordinates properly. The drone will be autonomous as well as communicate and transfer data from the Optitrack. At last, it is required to connect two hovering UAVs using a connecting arm that is being developed and operated by ROS and Ubuntu.

Number	Requirement	priority
1	Gazebo simulation	High
2	Stream the data that coming from Optitrack motive	High
3	Include mavlink to ROS	High
4	Stop the drone if it flies improperly, using a safety button	High

*Table 1: Requirements for control team*

The most important requirement of the docking mechanism design is to make sure that the mechanism can maintain the integrity of the drone in case of any failure in the docking process, we know that the approach of the two drones to each other is a dangerous endeavor, so safety must be of paramount importance. Several tests will be run to ensure that the drone is able to move and dock to the other arm safely.

Number	Requirement	Priority
1	handle severe dangerous conditions	high
2	Minimum weight	high
3	Damp the vibration	high
4	Minimum distance	high
5	Safety	High

*Table 2: Requirements for the structure team*

From our initial flight test we found that we should build our mechanism to resist the vibration that could affect the stability of the drone and this contributes to obtaining inaccurate results that can affect the movement of the drone and its inaccurate arrive to the other drone. The minimum weight is of high importance because our drone can hold a specific weight.

## **2.2. System Specification.**

The control and the structure systems consist of many different parts. However, the main parts for the control are the controller, Optitrack, communication and battery. Also, the main part of the structure system the body and docking mechanism. The team is also required to make sure that the mechanism is built to maintain the safety of the drones when they approach each other and build enough protection to keep propeller safe. such system needs to be well installed and properly functioning for the success of the mission.

### **The structure team system specifications:**

#### **Body**

The body is the main component in the drone as it supports the whole system by carrying most of the drone components including the payload and protect the components from the external effects. And based on this specific information we build our mechanism. Moreover, our durable drone body, helped us to distribute the weights correctly and place the mechanism in the right place because it provides the ability to install any excess parts or any extra arms.

#### **Docking Mechanism**

The mechanism is the component that we build to supports the whole system by carrying most of the docking mechanism components including the locking system and arms. The geometry of the mechanism is determined based on the vibration that may occur due to the extra weight and the type of material. In addition, it supports stability as well as the mobility of the system. The mechanism will have a long arm to maintain a safe distance, which could affect the performance and stability of the other drone.

### **The control team system specifications are:**

#### **Optitrack**

The Optitrack system is used for localization of the two drones. It is an array of cameras that find the position and orientation of each of the vehicles. Its specifications are as follows:

- **Primex 13**

High speed tracking with excellent precision in medium-sized capture volumes.

- Resolution: 1.3 MP.
- Frame Rate: 240 FPS.

## **Controller**

The controller is basically the brain of the drone which has several responsibilities. Raspberry Pi 3 Model B+ will be the controller used for the rover because of the multitasking capabilities, it has four cores that runs on 1.4GHz and has 1GB SDRAM. Can communicate using 2.4GHz and 5GHz. Require input voltage of 5V.

## **Communication (QGroundControl)**

QGroundControl provides full flight control and vehicle setup for PX4 vehicles. It provides easy and straightforward usage for us. This program provides easy access to all flight data that we have made using our vehicle and this helps us in studying the condition of our vehicle.

## **Chapter 3: Theoretical Foundations**

This chapter discusses the calculations made while creating the autonomous UAV assembly when it comes to the battery consumption during flight, flight path taken and other aerodynamic effects and interactions that are occurring during the two UAV assembly. Moreover, discuss the approximations made to achieve the required results. And finally, review the limitations that were faced while fabricating and testing and the assumptions that were made.

## **Introduction**

The aerospace and aviation industries are very large fields where there are a lot of opportunities and innovations to be discovered and made. Our UAV autonomous assembly uncovers new possibilities such as maintenance and part replacements can happen during flight. This will enable us to discover possibilities such as replacing broken parts on a large aircraft without using individuals which will increase the safety margin, replacing damaged drone parts by sending another drone to the specified location and using different autonomous assembly methods to replace the part during flight, using the UAVs to assemble small parts into a large usable part, and much more possibilities involving the autonomous assembly method.

## **Standards and regulations 3.2**

### **National flying regulations 3.2.1**

There are certain rules and regulations that are needed to be followed when attempting to fly a drone in the UAE. First, the pilot needs to be registered at the GCAA with the drones that are going to be used. Moreover, drones weighing 5 kg or less are only permitted to fly in the specified green zones with no recording devices fitted to it with the exception of registered individuals at the GCAA with UAE IDs and with no drop and release devices are allowed to be fitted to the drone. The flying range should be kept less than 122 meters during daytime and should be kept away from public and private properties. Drones shall be kept away from airports and helicopter flight zones with a circular diameter of 5 kilometers with radio control link maintained between the pilot and the drone. Drones weighing more than 25 Kg shall only be flown by registered GCAA individuals with a minimum age of 21 years old. And finally, collisions between objects and/or people are predominantly to be avoided at all costs and to be reported to the GCAA if any transpire [4].

### **Design standards 3.2.2**

Following the regular standards in the battery used, it is a rechargeable lithium-ion battery, and it follows the IEC 62133-2:2017 is a standard that defines the specifications and tests for the safe operation of portable sealed secondary lithium cells and batteries with a non-acid electrolyte in both planned and reasonably foreseeable situations [6]. When it comes to performance checks, designations, markings, dimensions, and other specifications for secondary lithium single cells and batteries for portable applications are defined in IEC 61960-3:2017. Furthermore, it provides buyers and consumers of secondary lithium cells and batteries with a set of parameters by which they can compare the output of different manufacturers' secondary lithium cells and batteries [5].

## **3.2 details of the theoretical model**

### **State estimation**

The PX4 flight controller offers three methods of estimation:

- 1) EKF2 estimator: estimates attitude, position, velocity, and wind states
- 2) LPE estimator: Estimates Position and velocity states

3) Q attitude Estimator: Quaternion based complementary filter for estimating attitude state. The EKF2 and LPE estimators are based on extended Kalman filtering. In our project, we use the EKF2 estimator because it is the most supported and tested on the PX4 platform. [6]

Extended Kalman filter is a non-linear version of the ordinary Kalman filter which is a method of fusing different sensors and the previous estimation to estimate the next step. This is done by taking a weighted average of the previous estimation and the current sensor measurements. The weights decide which we should trust more than others. In our Project, since we are running the experiment indoors, we are not trusting the gps for location and the barometer for altitude estimation. We have a more trusted sensor for position and altitude from the optitrack which we will use for the state estimation.

The EKF2 Estimation system is an extended Kalman filter with 24 states that helps identify the state of the altitude. Also, it helps identify the position and velocity by using the North, East, and the down directions. But our flight system is designed so that it goes and follows the required path according to the coordinates provided by the Optitrack system. The EKF2 also estimates the Gyro bias offset and scale factors plus the Z acceleration bias. The EKF2 has a lot of other capabilities like estimating the earth and body magnetic fields and also the wind velocity. But for our system, we are not going to put those properties in use. [7]

The EKF2 has a lot of benefits that can be taken advantage of. It can improve the accuracy during high-rate maneuvers by estimating the gyro scale factors. Moreover, it can switch between magnetometers if a fault is detected. Additionally, during flight, it can control large gyro bias changes while also providing a slightly smoother output and accuracy while also using less computing power. Another capability is that it can recover faster from bad sensor data and automatically starts using GPS rather than waiting for the vehicle motor to arm to improve the waiting times as soon as the checks pass. [7]

### 3.3.2 Discrete or other approximations

It is approximated that the UAV has a perfect weight distribution that is positioned at midpoint. Which means insignificant input from the stabilization system that is programmed in the

controller. Another approximation that was made is that during the connection using the docking system between the two UAVs, the down wash of the two UAVs can interact which causes some disruption that affects the stability of both UAVs making them propel each other. This can cause a lot of difficulty when connecting the two drones, also due to the air flow going from one propeller and hitting the other drone causing it to move further from the specified coordinate. These points are mostly disregarded during the connection and are mainly trying to be avoided.

### **3.3.3 Limitations and assumptions**

Limitations were faced during the during the process of making the project. For the drone to take flight manually. A person with a drone flight license is always needed to take control of the drone flight. Which caused a lot of setbacks and made us get behind schedule. During the development of the project, it was only allowed to fly the drone within a specific space covered with nets to help improve the safety. This factor restricted the exploration of the potential of what the drone can achieve when it comes to flight paths and high altitudes.

Another limitation that was faced is that for the UAV to perform tasks, markers were placed on top of the drone at different positions to create a structure that can be visible by the Optitrack cameras to provide the exact position of the UAV. And due to that reason, the drone can only perform tasks under the Optitrack position cameras which cost a lot to be obtained and positioned in deferent places.

It is assumed that the UAV has a perfect weight distribution that is placed in the exact center. Which means minimal input from the stabilization system that is programmed in the controller. Moreover, the facing direction of the drone was always limited due to the Qgroundcontrol program requiring a certain face direction at the start. Moreover, any small interruptions externally made like the drone falling on the ground or different placement always required a new calibration which was time consuming.

## **3.2 details of the theoretical model**

### **State estimation**

The PX4 flight controller offers three methods of estimation:



- 1) EKF2 estimator: estimates attitude, position, velocity, and wind states
- 2) LPE estimator: Estimates Position and velocity states
- 3) Q attitude Estimator: Quaternion based complementary filter for estimating attitude state.

The EKF2 and LPE estimators are based on extended Kalman filtering. In our project, we use the EKF2 estimator because it is the most supported and tested on the PX4 platform. [6]

Extended Kalman filter is a non-linear version of the ordinary Kalman filter which is a method of fusing different sensors and the previous estimation to estimate the next step. This is done by taking a weighted average of the previous estimation and the current sensor measurements. The weights decide which we should trust more than others. In our Project, since we are running the experiment indoors, we are not trusting the gps for location and the barometer for altitude estimation. We have a more trusted sensor for position and altitude from the optitrack which we will use for the state estimation.

The EKF2 Estimation system is an extended Kalman filter with 24 states that helps identify the state of the altitude. Also, it helps identify the position and velocity by using the North, East, and the down directions. But our flight system is designed so that it goes and follows the required path according to the coordinates provided by the Optitrack system. The EKF2 also estimates the Gyro bias offset and scale factors plus the Z acceleration bias. The EKF2 has a lot of other capabilities like estimating the earth and body magnetic fields and the wind velocity. But for our system, we are not going to put those properties in use. [7]

The EKF2 has a lot of benefits that can be taken advantage of. It can improve the accuracy during high-rate maneuvers by estimating the gyro scale factors. Moreover, it can switch between magnetometers if a fault is detected. Additionally, during flight, it can control large gyro bias changes while also providing a slightly smoother output and accuracy while also using less computing power. Another capability is that it can recover faster from bad sensor data and automatically starts using GPS rather than waiting for the vehicle motor to arm to improve the waiting times as soon as the checks pass. [7]

## Controller Architecture

The PX4 implements a cascaded control Architecture where the position controls are in the outer loop and the attitude controllers are in the inner loop.

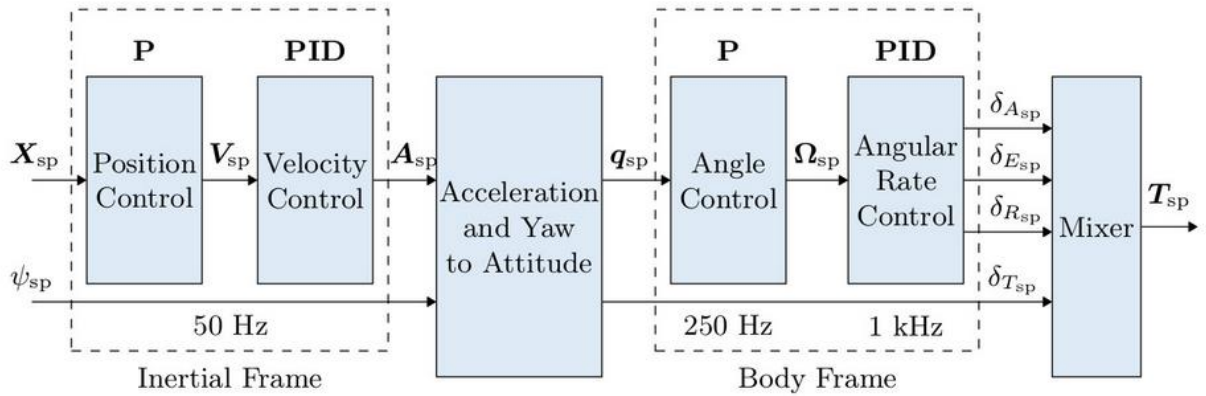


Figure 3: controller architecture of PX4 [8]

The position controller is a simple Proportional controller. Because of this, we can expect some difference from the commanded position.

The controller gains we used initially were too high to be used in our application. We had too much overshoot and this caused us to crash into the target rather than docking smoothly.

The response can be seen in the figure below:

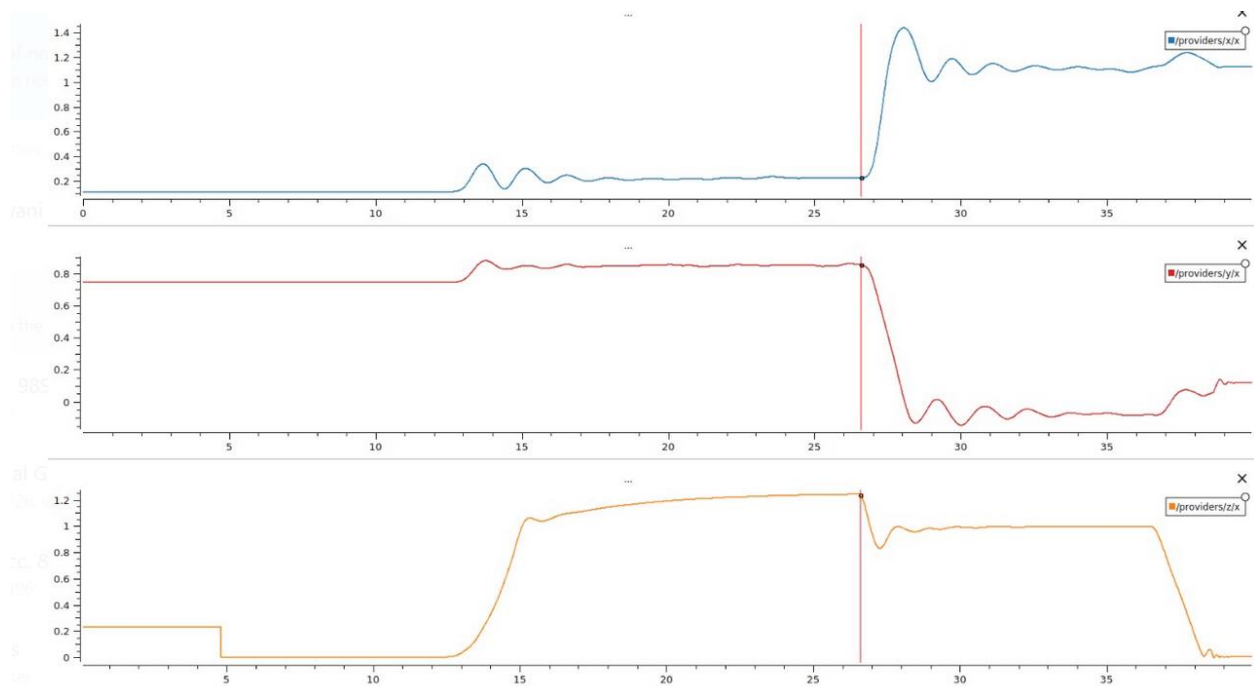


Figure 4: Response from the hexacopter

The figure shows response in X, Y and Z directions, respectively. The red marker shows the point when the set point was defined. We can see that there is about 30% overshoot in the x direction in this setup. The controller gains were tuned later to decrease the overshoot. We expect that the new performance will offer a more stable flight for our requirement.

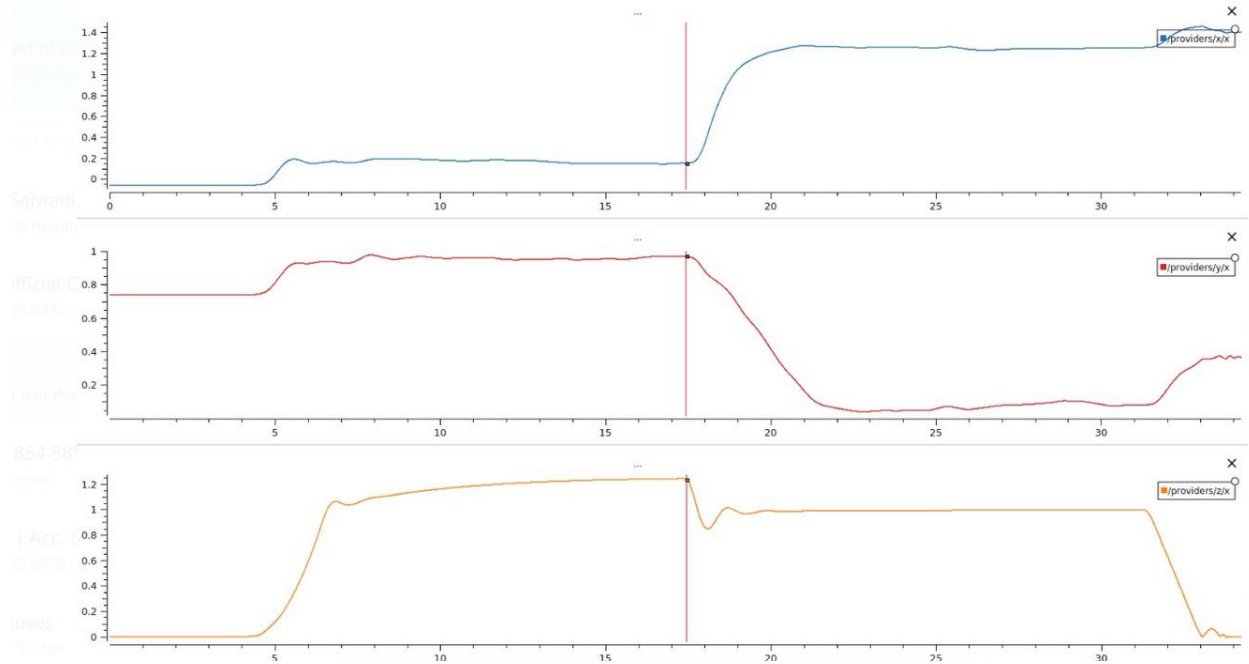


Figure 5: PID response after tuning to remove overshoot.

We decreased the proportional gain to remove the overshoot. In the response plots above we can see that there is no overshoot in the x and y directions. It must also be noted that the response is slower in this case. The overshoot in the z direction does not pose any problems, because we can see that the z achieves steady state before x and y.

## Chapter 4: concept Generation and design evaluation

### 4.1 Introduction

We have an idea that we want to implement, so we wanted to build a docking system design that help to merge the two drones, and this merging is flexible and smooth and can solve the errors that may face while movement during contact, and we were able to make the docking system as a single body with the drone, despite the increase in the weight of the plane. And frequent vibration and imbalance, but we were able to correct these defects by choosing the right design for the operation. The designed docking design systems will consist of two arms which will be attached in two

drones, in front of the arms a part will be designed using Solidworks and 3D printed, and it will be composed of a slot and the other part is a lock, as shown in the table 1 (Simple docking system), and we will add a spring in the lock to make the movement easier between the two drones while hovering. The mechanism is simple, it just needs a push to close and dock the other drone and a pull force to open the lock. As well as we made the slot in the front wider to increase the probability of hitting the target, as shown in the table 1 (Simple docking system). Moreover, the structure of the drone went through several stages, where different materials and designs were taken into consideration. We learned that the applicable companion computer would be the raspberry pi 3 is better suited for our project application because it has multitasking abilities.

## 4.2 Different design option

### Platform



Configuration	Description	Picture
Crazyflie 2.0	The Crazyflie 2.0 is a palm sized quadcopter weighing 27 grams supporting wireless control over radio and Bluetooth Low Energy (LE). It has a flight time of 7 minutes and a charge time of 40 minutes [9].	
DJI F550 Hexacopter	The features are Flame Wheel motors F550 airframe with landing gear, weight of 478g. it is a platform that you can add on it for example put a battery, control system, GPS, and others it depends on your project needs.	

Table 3: Platform comparison

We have challenged ourselves and decided to make our project for larger and heavier UAV's for their usual uses in the future, despite the existence of experiments and research for small UAV's with magnets ModQuad [2], the module is based on a Crazyflie 2.0 quadrotor platform within a cuboid frame which allows it to attach to other modules by matching vertical faces. But we found the drone suitable for long tasks and the larger size suitable for most tasks such as strength and weight bearing.

## Companion Computer


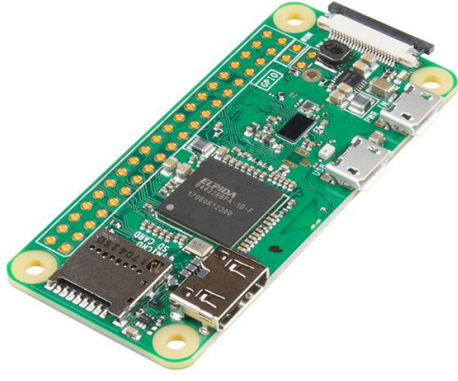
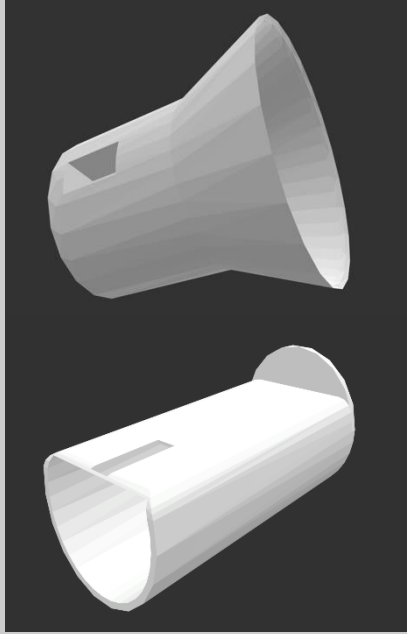
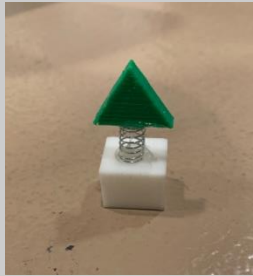
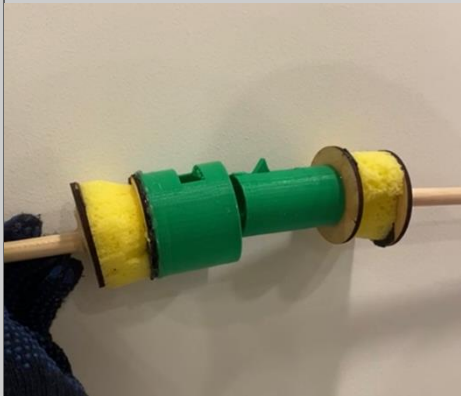
Configuration	Description	Control model
Raspberry Pi 3 Model B+	It has four cores that runs on 1.4GHz and has 1GB SDRAM. Can communicate using 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE. Support Power over Ethernet(PoE) but require a separate PoE hat. Require input voltage of 5V.	
Raspberry Pi Zero W	It is largely rein reduced size of only 65mm long by 30mm wide and at a very economical price. Mini HDMI to HDMI cable or adapter to connect to your monitor. It has a USB device, as well as a unique CSI camera cable. you will need a microSD card with an operating system and a high-quality 5V power supply.	

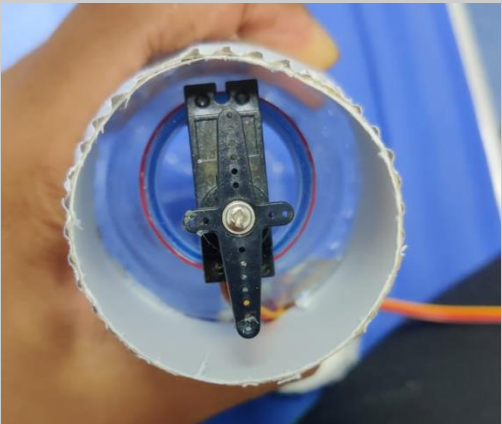
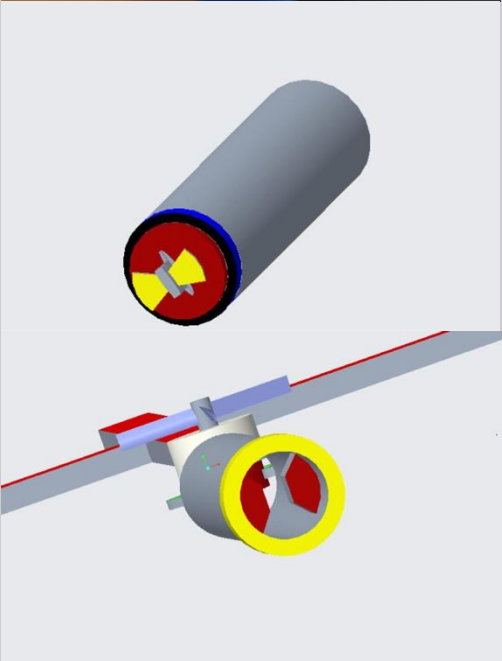
Table 4: Companion Computer comparison

We want to fly the drone in offboard mode with the help of a companion computer. We faced some difficulty in choosing the right operating system. So, through our research and the help of colleagues in the laboratory, they advised us to use Raspberry pi and we had several options of the same type. We first tried Raspberry Pi Zero but the setup for it little more complicated than on

other pieces. Because of the small size, many of the connectors on the Pi Zero are not standard. Therefore, after our experience, we found Raspberry pi3 is better and can handle the tasks.

### Docking mechanism

Configuration	Description	control model
Simple docking system	We designed this simple device to increase the possibility to go through each other and lock and open easily and its appropriate size and lightweight material PLA 3D printed, and a small part with a spring is add in the big part from the top, that helps to lock and open.	 
Composite docking system	This design consists of PLA 3D printed, sponge, spring and wood and is flexible, medium size and small probability of entering the target.	

Configuration	Description	control model
Motor docking system	This design has a motor that rotates to close the docking system, and it consists of foam and plastic materials, and its size is small and the possibility of successful closure and open is rare.	 

*Table 5: docking mechanism Comparison*

We have come up with many ideas that may work with this project successfully, and the best of them are three designs, which are the simple docking system to merge the two drones, also the composite docking system and the other design was supported by a locking motor. We have learned through experiments that the simple docking system is the best solution for our project for the design ability to perform the task perfectly by increase the possibility to go through each other easily by making the diameter of the slot bigger and the spring has low stiffness about 503 N/m



which can bend the locking pin easily by law force from the drone and to open the dock it only needs a small force in the opposite side. The design is uncomplicated it's fluctuation of electricity use.

### Arm material

Configuration	Description	Picture
Carbon fiber rod	High performance the carbon fiber rod has good heat resistance, corrosion. Lightweight lighter than aluminum, yet stronger than steel, stable performance. The price is suitable about 80 AED for 12mmx10mmx420mm ,2 pieces.	
Wood rod	Strong wood rod, lightweight, stable the size is 615mm in length and 16mm width, the price for one piece is 17 AED.	
Foam sheet	Lightweight and can be shaped easily, 5mm thick foam sheet. The size of the board is 50x70cm. A pack has 5 pieces about 14 AED.	

Table 6: Arm material comparison

In the beginning, we used foam for its low weight to connect the drone to the device, and the use was difficult due to its flexibility and lack of stability, so we changed the material for wood, which was more solid, stable, and less weight. In the end, we used carbon fiber because it is better than other materials for its stability, high strength, and vibration damping [10].

### Drone protection



Configuration	Description	Picture
Drone cage	a 360° shell around the drone enables safe collisions with objects that would otherwise get in the way whilst carrying out inspections indoors, in confined spaces, or hard-to-reach places. DRONECAGE's Collision Rods are made from ultra-lightweight with advanced aerospace-grade carbon fiber memory materials. This gives DRONECAGE ultimate strength and flexibility whilst on missions. It is so expensive it costs 4,734.19 AED.	
Anti-collision ring	6 pieces propeller protection Anti-collision ring for F550 drone, it is lightweight and cheap price about 50 AED. The material is flexible and if one piece is braked, we can just replace the broken one only. We will 3D print it because it is simple.	

Table 7: propeller protection system comparison

To protect the drone, we looked for a way to repel the strikes to mitigate the damage that would be caused to the drone. we had several options, the propeller protection Anti-collision ring or Protective Cage, the full cover. After research and consultation, we used Anti-collision ring, because it is low in weight, ease of manufacturability, fends bruises with flexibility, and reduces damage when doing tasks.

### **4.3 Key design constraints**

While manufacturing the parts that we attached to the drone we took into consideration that the material must suit our requirements, that is why we selected composite for our first docking system since it is lightweight, and it will give us absolute freedom to shape and gives flexible movement. we chose carbon fiber for the rod arm and (PLA) for our second design which is our final docking system to make it lighter, fatigue resistance compressive strength, more stable, and does not vibrate or bend.

Here we are trying as much as possible to provide a design, which is characterized by its simple mechanism and performs tasks that are required from it completely and this will be achieved by using sophisticated devices that is available in the market and affordable.

Our project requires its tasks to merge with another drone, and this makes our project exposed to the risk of collision and difficulty in using the perfect design that can protect the drone and makes its movement easy and stable without vibration of the attached docking system.

It is also important that we choose low-weight materials and high-precision quality control devices for the success of our project.

## 4.5 Assessment of design

### Material Selection

Materials	Advantages	Disadvantages
Foam	<ul style="list-style-type: none"><li>○ Lightweight</li><li>○ Easy to shape.</li></ul>	<ul style="list-style-type: none"><li>○ Not stable while hovering.</li></ul>
Aluminum metal:	<ul style="list-style-type: none"><li>○ Corrosion-resistant.</li><li>○ Lightweight</li><li>○ Strong</li><li>○ Recyclable</li></ul>	<ul style="list-style-type: none"><li>○ More expensive than steel</li><li>○ Lower strength than steel</li></ul>
Steel:	<ul style="list-style-type: none"><li>○ Tough</li></ul>	<ul style="list-style-type: none"><li>○ Bends faster than aluminum.</li><li>○ Brittle</li><li>○ Heavy</li></ul>
Wood	<ul style="list-style-type: none"><li>○ Easy to shape and cut.</li><li>○ Low cost</li></ul>	<ul style="list-style-type: none"><li>○ brittle</li><li>○ Weak</li></ul>
Carbon fiber	<ul style="list-style-type: none"><li>○ Lightweight</li><li>○ High strength</li></ul>	<ul style="list-style-type: none"><li>○ Expensive</li></ul>
PLA	<ul style="list-style-type: none"><li>○ Lightweight</li><li>○ Low cost</li></ul>	<ul style="list-style-type: none"><li>○ Limited geometry size and structure for 3D printer</li></ul>

Table 8: Material advantages and disadvantages

## Scoring

Score	Description
0	Poor
0.5	Average
1	Excellent

Figure of Merit	Score Weight	Aluminum metal	Steel	Wood	PLA	Carbon fiber	foam
Weight	0.25	0.5	1	0.5	1	1	1
Strength	0.25	0.5	1	0.5	1	1	0
Durability	0.20	1	0	0	1	1	0.5
Manufacturing	0.20	1	0.5	1	0.5	1	1
Cost	0.10	1	0.5	1	1	0.5	1
	<b>Total Score</b>	0.75	0.6	0.55	<b>0.9</b>	<b>0.9</b>	0.625

Table 9: scoring

Based on the comparison above, it can be observed that PLA and carbon fiber are the most suitable material for the docking system and the arm rod that will be attached to the drone, so we choose carbon fiber for our arm rod where carbon fiber because of higher strength and stiffness and better creep and fatigue resistance and more stable (no vibration). While PLA is the most suitable material for the docking system.

## Final System

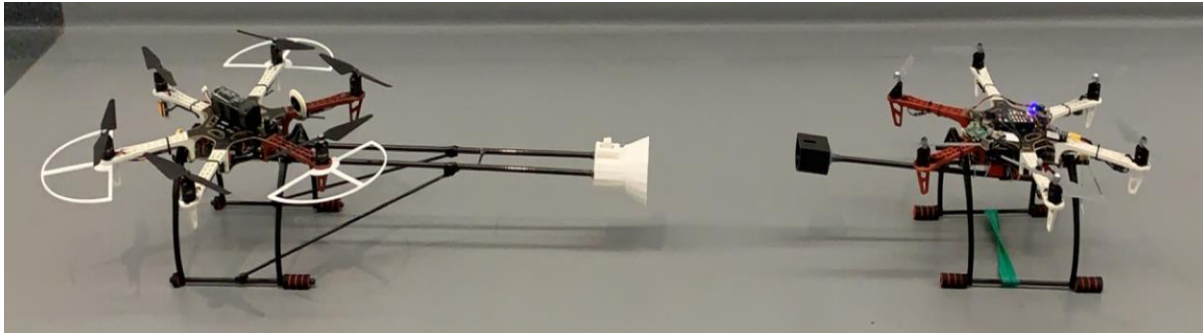
Propeller protection

Carbon fiber rod

Docking system (slot)

Docking system (lock)

spring



Stationary drone

Arm length: 82 cm

Arm length: 37 cm

Moving drone

## 4.3 Simulation

### Gazebo Simulation

We implemented the algorithm on a physics simulation engine called GazeboSim. With the help of GazeboSim, we can simulate the vehicle and simulate the firmware that is running on the simulator. There are some differences between the simulation and the actual hardware simulation. Some of the differences are:

- 1) The simulation is based on a different platform. The simulation uses typhoon h480 hexacopter. The actual hardware implementation is based on dji f550 Hexacopter.
- 2) The simulation runs on ubuntu 18 and ROS melodic whereas the hardware implementation runs on ubuntu20 and ROS Noetic.
- 3) The simulation uses a simulated gps for position estimation. In the actual Hardware implementation, we use the Optitrack system.

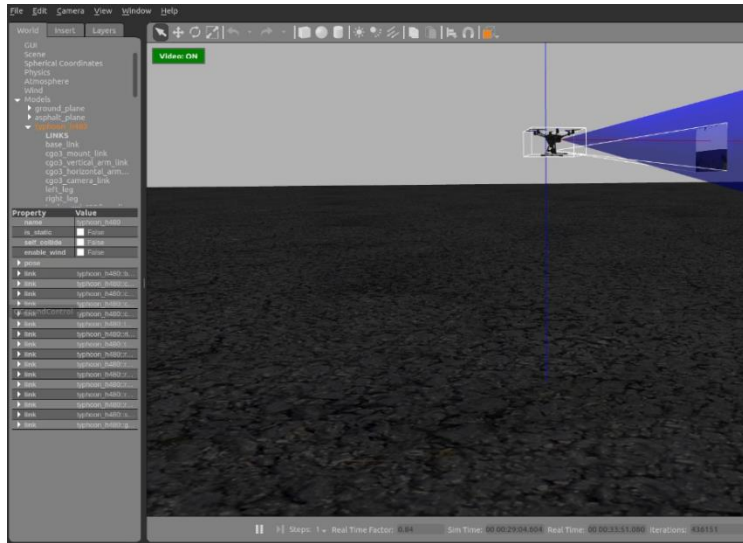


Figure 6: Simulated hexacopter in flying in offboard mode.

We were able to fly the hexacopter platform in offboard mode in the simulation and we were able to see that we can use the same code for different platforms. This is because regardless of the type of platform is being used, mavros behaves similarly for all the different supported platform types. Px4 accounts for the different platform and behaves according to the setpoint we have sent.

## Chapter 5: Implementation and Fabrication

There are two different parts that must be implemented separately. We need to setup the hexacopter platform to be able to achieve autonomous flight. We also need to fabricate the docking mechanism so that we can dock autonomously.

### Hexacopter setup

As mentioned earlier, we need a companion computer to achieve autonomous flight and for this we had chosen the raspberry pi3 (Rpi). The raspberry pi requires 5v and 2.5Amps of current. The power supply for the Rpi is crucial in this project because having under voltage will cause the Rpi to work inefficiently and can even cause file system corruption. We attempted different methods to supply this reliable power. And these were:

- 1) Power supplied from the holybro power distribution board. This is where the Pixhawk flight controller gets 5v power from.

- 2) Additional 5v battery to be used only by the Rpi.
- 3) Buck converter to convert 12v from the Lipo Battery to 5v that can be used by the Rpi [11].

In our tests, we were consistently getting a low voltage alert, so in the end we used the Buck converter because it gives us a good control over the output voltage. This helped alleviate the undervoltage problems we were having.

### Electrical connections

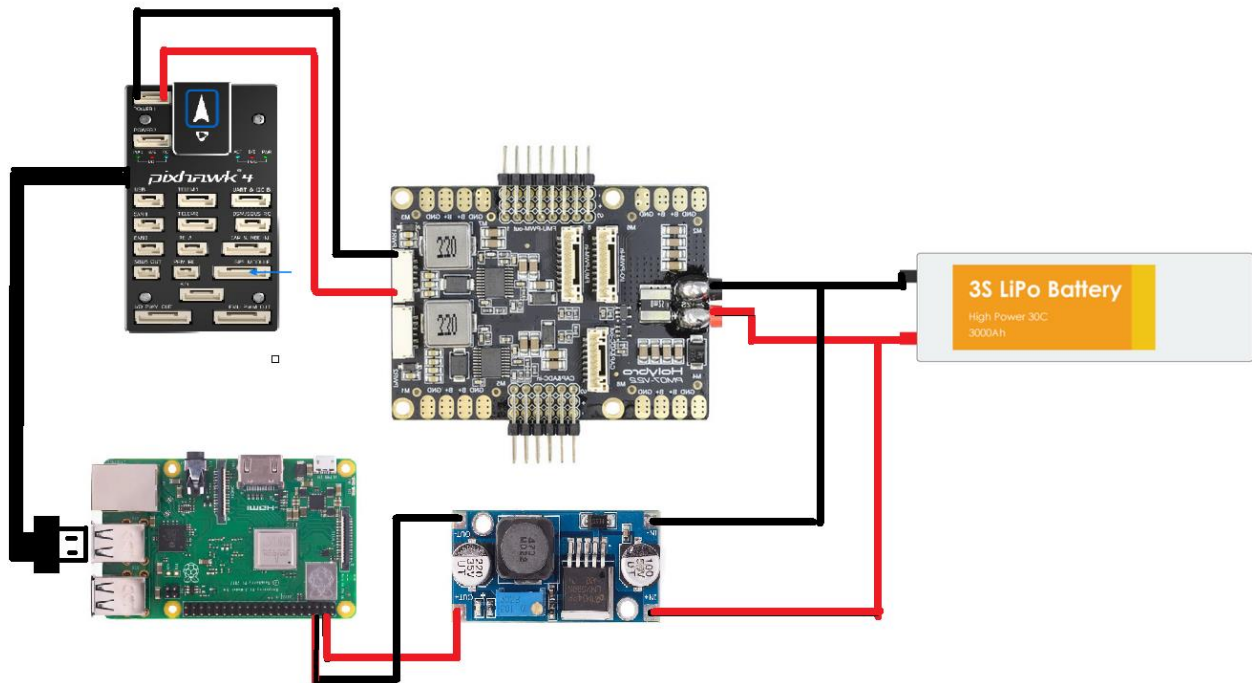
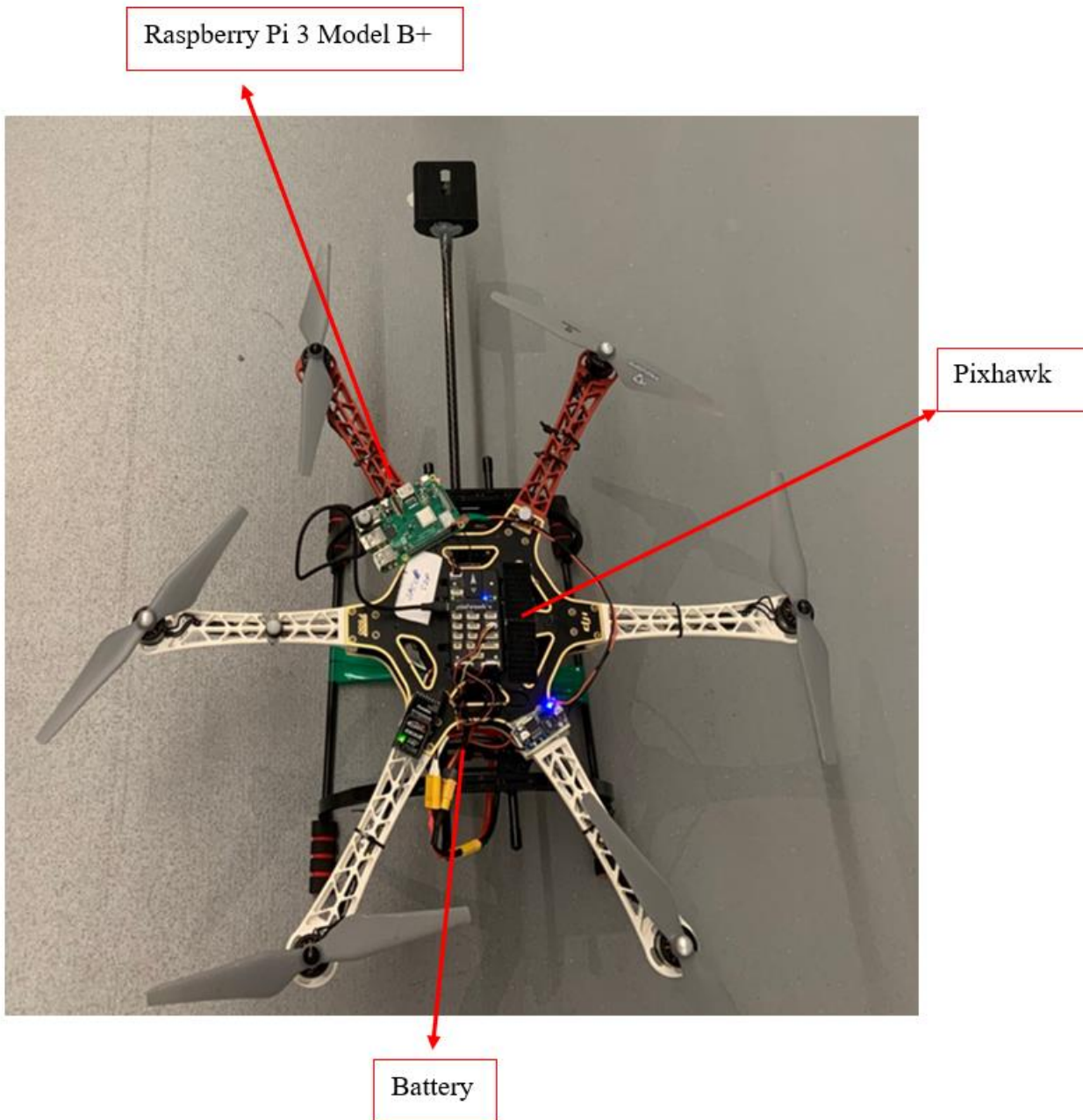


Figure 7: Electrical connections

The connections from the Rpi to the Pixhawk could be made through the USB or through the serial port in the GPIO pins. During our tests we found that attempting to boot the Rpi with the GPIO serial port connected had trouble booting until the GPIO pins were unplugged. We decided to use a USB connection to avoid this problem. The PX4 will not let us takeoff while a USB is connected due to safety reasons, but this can be easily overridden.

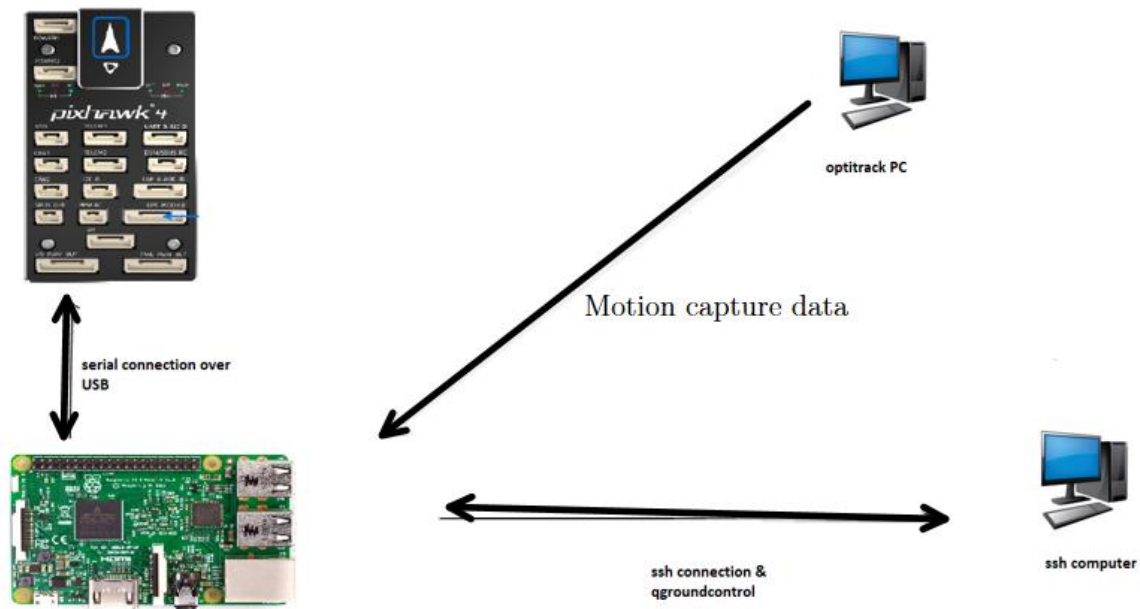




*Figure 8: Hardware implementation of the circuit diagram*

### **Data Streams**

In this Project there are a lot of data that must be sent to different components in real time and with high latency. The figure below shows how all the data is sent between different computers that are used in this project.



*Figure 9: Data streams between different parts*

In this figure, the left-hand side shows all the connections on the drone itself and the right hand side shows all the connections on the ground station. It must be noted that the connections are not direct between computers, the connections are made over a high speed 5GHz router. This gives us a low latency connection between the computers.

When the Rpi is on the platform, we cannot give it commands, so to give commands remotely, we connect the Rpi to a computer over SSh [12]. With the help of the SSh computer, we can connect the Rpi to the optitrack Pc for localization. With the help of the SSh computer, we can also use mavros to connect the Rpi with the Pixhawk over serial and Rpi with QgroundControl over UDP. In our project we run QGroundControl from the same pc that is using ssh. This is not a necessary condition, but it allows us to have fewer computers in the network.

## **ROS configuration**

We have the optitrack data coming from the optitrack that is published to the topic “/vrpn\_client\_node/RigidBody01/pose”. We need to publish the data to a mavros topic to use the data for state estimation. As mentioned before, we are using EKF2 state estimator, so for this

estimator we need to publish the optitrack data to “/mavros/vision\_pose/pose” the pose data that is published on this topic will be used for position estimation of the drone.

## **Chapter 6: Testing**

### **6.1 Verification**

As stated in the requirements, the system is constructed with two drones. A stationary drone and another drone who performs the connection autonomously. This connection can be made by the design of the docking system where it allows a seamless connection. But due to some shortcomings, during the testing, only one drone has been used to perform the connection with a stationary dock. This is due to the high safety risks linked to the connection of the two drones. Small malfunctions can be catastrophic if we are using two drones. So, using a stationary dock with only one drone can help imitate the environment to a certain degree while still maintaining the required safety margins.

### **6.2 Validation**

For the first couple of flights, a couple of issues were faced where we were trying to keep the drone at a certain altitude and other issues regarding the stability. A specific coordinate was provided using the program linked to the QGroundControl, but the drone always kept yawing 90 degrees to the left for no visible reason causing a tremendous directional error. Moreover, a certain roll effect due to the inconsistent stability showed a translational motion.

While testing, a couple of collisions were recorded since the drone kept losing stability and causing it to have an uncontrollable translational motion toward the safety nets. Causing a free fall when the drone is at a high altitude and damaging the drone's legs. The legs have been replaced. The final leg design used was more compliant and able to handle and accidental pressure applied to it.

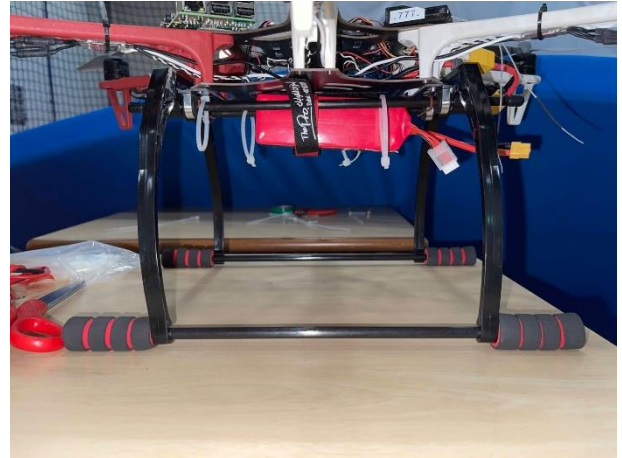


Figure 10: Unsuccessful test

In the final test, we can see that the flight was more stable. The position of the drone is plotted in the figure below.

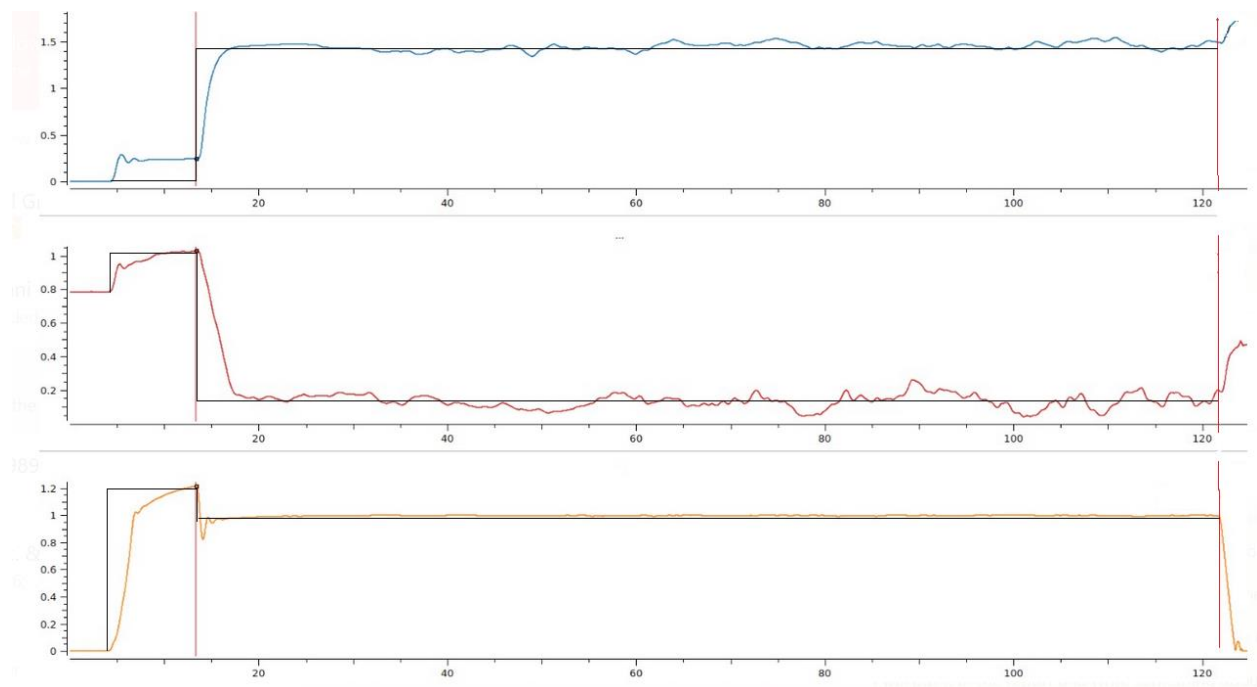


Figure 11: flight test (X, Y, Z)

we can see that the vehicle follows the setpoint tightly in the beginning, but there is considerable drift in the x and y directions. This might be because of the effects of wind that is reflected from the stationary docking system. We should notice that the discrepancy is more noticeable the longer the flight test. The voltage drop from the battery might cause poorer controller

performance. We can account for this slight drift with the mechanical design of the docking system.

### **6.3 Evaluation**

The closest system that is available in the literature to our system is the ModQuad [2]. In that project they are using magnets to attach two systems accurately. This might cause problems down the line if this system is to be implemented with the help of GPS. They also use small quadcopters. This might have been done to minimize the interaction between the two drones. This has the disadvantage of not being able to carry weight. As discussed earlier, the idea was born out of having a high aspect ratio wing. To meet this end goal, each drone will be required to carry some weight by itself.

## **Chapter 7. Conclusion**

### **7.1 Summary of work done.**

There were many things that the team learned during the work on this project. From the very initial definition of the problem to the very end of the project, there were many changes that were made to the project and many different design options that were explored before we settled on the final design. During each test that we did for each of the parts, we faced some problems, and these problems were opportunity for us to improve on the design. Using an open-source platform like Px4 gave us many resources like forums where we could interact with the community that are already using Px4 for troubleshooting and exchanging information. We were also able to learn other skills such as project planning and teamwork while working on this Project.

### **7.2 Conclusion**

To conclude the work done, the progress of the project was influenced by the tests, as well as the meetings, discussions, and sources of support the group held as a team, all of which contributed to the various ideas supporting the structure. It is believed that the ideal approach was trial and error and considering all ideas to ensure teamwork and creativity. Other modifications and tests were carried out to ensure that the components were functioning properly and that the structure was fully maintained and operated. The project has been finalized with various tests to ensure that the system works as intended.

### **7.3 critical appraisal of work done**

Considering Safety, we were not able to test the system on actual two drones. We were however able to emulate the working environment, by placing the docking system on a stationary object and docking with the stationary object. This is different from having two drones because even when we say that one of the drones are “stationary” the position hold is not perfect on drones. The drone that is executing position hold will drift small distances about the setpoint. When we have two drones, we will have twice the inaccuracy. The airflow from one of the drones will influence the other drone. In the case where the docking system is placed on a stationary platform, we do not have this effect. The test can be made more accurate by providing some external wind to judge how our system will work in the presence of some external wind.

Although the system is close to the required system and can be used to show that such docking is possible, there may be problems and bugs further down the road when the system is attempted on two drones.

### **7.4 Recommendation for further work**

There is no one right approach to dealing with these design projects, but better solutions are realized through trial and error. After our experiences, we concluded that the idea needs a lot of additions and research support and development necessary for the success of the project on a larger scale, then the project will be very useful in most sectors of aircrafts where the project can be applied with a group of drones combined them to fly autonomously together in a big aspect ratio and this will benefit a lot in many faces, one of them is if one of the drones break down, the other will take its place. In addition to the ability to make an integrated plane from collecting drones together and connecting each other by the arms (docking system), this will help reduce dropping some ruined parts of the aircraft and provide a maintenance mechanism in the air by flying one of the drones and merging them with others and fixing the malfunction or removing and repairing it.

## Appendices

## References

- [1] J. Euchi, "Do drones have a realistic place in a pandemic fight for delivering medical supplies in healthcare systems problems?," *Chinese Journal of Aeronautics*, vol. 34, no. 2, 2020.
- [2] B. G. G. L. M. Y. V. K. David Saldana, "ModQuad: The Flying Modular Structure that Self-Assembles in Midair," in *IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, 2018.
- [3] D. A. I. Penkov, "Analysis and study of the influence of the geometrical parameters of mini unmanned quad-rotor helicopters to optimise energy saving," *International Journal of Automotive and Mechanical Engineering*, vol. 14, no. 4, pp. 4730-4746, 2017.
- [4] UAV Coach, "UAV Coach," 21 January 2021. [Online]. Available: <https://uavcoach.com/drone-laws-in-uae/>. [Accessed 4 April 2021].
- [5] IEC, *IEC 61960-3 Edition 1.0 2017-02 INTERNATIONAL STANDARD NORME INTERNATIONALE Secondary cells and batteries containing alkaline or other non-acid electrolytes – Secondary lithium cells and batteries for portable applications*, Online, 2017.
- [6] PX4, "PX4 documentation," PX4, 12 february 2020. [Online]. Available: [https://docs.px4.io/master/en/advanced/switching\\_state\\_estimators.html](https://docs.px4.io/master/en/advanced/switching_state_estimators.html). [Accessed 4 January 2021].
- [7] Arducopter, "Arducopter documentation," 2020. [Online]. Available: <https://ardupilot.org/dev/docs/ekf2-estimation-system.html>. [Accessed 19 January 2021].
- [8] PX4, "PX4 documentation," 27 january 2021. [Online]. Available: [https://docs.px4.io/master/en/flight\\_stack/controller\\_diagrams.html](https://docs.px4.io/master/en/flight_stack/controller_diagrams.html). [Accessed 15 February 2020].
- [9] bitcraze, bitcraze, 2021. [Online]. Available: <https://store.bitcraze.io/products/crazyflie-2-0>. [Accessed 21 december 2020].

- [10] I. V. M. H. Zuzana Murčinková, "Damping properties of fibre composite and conventional materials measured by free damped vibration response," *Advances in Mechanical Engineering*, vol. 11, no. 5, 2019.
- [11] Components101, "Buck Converter: Basics, Working, Design and Operation," 26 April 2019. [Online]. Available: <https://components101.com/articles/buck-converter-basics-working-design-and-operation>. [Accessed 27 April 2021].
- [12] RISC Lab, "Risc-docs," 2018. [Online]. Available: <https://risc.readthedocs.io/1-indoor-flight.html>. [Accessed 23 December 2020].

## Software Listings

Some of the software that are used in the project are:

- 1) Ros Noetic: To send messages to different parts of the project.
- 2) PX4 flight stack: The flight controller that is onboard.
- 3) Ubuntu 20 server: The operating system used in the Companion computer.
- 4) Solidworks: Cad Software.
- 5) PTC Creo: CAD Software.
- 6) Cura: Slicer for 3d printing.
- 7) GazeboSim: Physics engine for simulating our vehicles and code.
- 8) Python: Programming language
- 9) Tmux: Terminal multiplexer to run multiple commands over the same ssh connection.
- 10) plotjuggler

## List of Tables

Table 1: Requirements for control team .....	11
Table 2: Requirements for the structure team.....	11
Table 3: Platform comparison.....	21
Table 4: Companion Computer comparison .....	22
Table 5: docking mechanism Comparison.....	24



Table 6: Arm material comparison .....	25
Table 7: propeller protection system comparison .....	26
Table 8: Assessment of design.....	28
Table 9: scoring.....	29

## List of figures

Figure 1: turbulent flow from Propeller.....	9
Figure 2: relation between lifting force and distance between propellers .....	9
Figure 3: controller architecture of PX4 [8] .....	18
Figure 4: Response from the hexacopter .....	18
Figure 5: PID response after tuning to remove overshoot. ....	19
Figure 6: Simulated hexacopter in flying in offboard mode.....	31
Figure 7: Electrical connections.....	32
Figure 8: Hardware implementation of the circuit diagram.....	33
Figure 9: Data streams between different parts .....	34
Figure 10: Unsuccessful test .....	36
Figure 11: flight test (X, Y, Z).....	36

## Code:

The code for initializing the PX4 and publishing all the ros topics is given below:

```
import rospy
from mavros_msgs.msg import GlobalPositionTarget, State, PositionTarget
from mavros_msgs.srv import CommandBool, CommandTOL, SetMode
from geometry_msgs.msg import PoseStamped, Twist
from sensor_msgs.msg import Imu, NavSatFix
from std_msgs.msg import Float32, Float64, String
import time
from pyquaternion import Quaternion
import math
import threading
```

```
class Px4Controller:
```

```
    def __init__(self):
```

```
        self.imu = None
```

```
        self.gps = None
```

```
        self.local_pose = None
```

```
        self.current_state = None
```

```
        self.current_heading = None
```

```
        self.takeoff_height = 3.2
```

```
        self.local_enumeration_position = None
```

```
        self.current_target_pose = None
```

```
        self.global_target = None
```

```
        self.received_new_task = False
```

```
        self.arm_state = False
```

```
        self.offboard_state = False
```

```
        self.received_imu = False
```

```
        self.frame = "BODY"
```

```
        self.state = None
```

```
'''
```

```
    ros subscribers
```

```
'''
```

```
        self.local_pose_sub = rospy.Subscriber("/mavros/local_position/pose", PoseStamped,  
self.local_pose_callback)
```

```
        self.mavros_sub = rospy.Subscriber("/mavros/state", State, self.mavros_state_callback)
```

```
        self.gps_sub = rospy.Subscriber("/mavros/global_position/global", NavSatFix,  
self.gps_callback)
```

```
        self.imu_sub = rospy.Subscriber("/mavros/imu/data", Imu, self.imu_callback)
```

```
        self.set_target_position_sub = rospy.Subscriber("gi/set_pose/position", PoseStamped,  
self.set_target_position_callback)
```

```
        self.set_target_yaw_sub = rospy.Subscriber("gi/set_pose/orientation", Float32,  
self.set_target_yaw_callback)
```

```
        self.custom_activity_sub = rospy.Subscriber("gi/set_activity/type", String,  
self.custom_activity_callback)
```

```

'''
ros publishers
'''

self.local_target_pub = rospy.Publisher('mavros/setpoint_raw/local', PositionTarget,
queue_size=10)

'''
ros services
'''

self.armService = rospy.ServiceProxy('/mavros/cmd/arming', CommandBool)
self.flightModeService = rospy.ServiceProxy('/mavros/set_mode', SetMode)


print("Px4 Controller Initialized!")


def start(self):
    rospy.init_node("offboard_node")
    for i in range(10):
        if self.current_heading is not None:
            break
        else:
            print("Waiting for initialization.")
            time.sleep(0.5)
    self.cur_target_pose = self.construct_target(0, 0, self.takeoff_height, self.current_heading)

    #print ("self.cur_target_pose:", self.cur_target_pose, type(self.cur_target_pose))

    for i in range(10):
        self.local_target_pub.publish(self.cur_target_pose)
        self.arm_state = self.arm()
        self.offboard_state = self.offboard()
        time.sleep(0.2)

    if self.takeoff_detection():
        print("Vehicle Took Off!")

    else:
        print("Vehicle Took Off Failed!")

```

```

        return

'''
main ROS thread
'''
while self.arm_state and self.offboard_state and (rospy.is_shutdown() is False):

    self.local_target_pub.publish(self.cur_target_pose)

    if (self.state is "LAND") and (self.local_pose.pose.position.z < 0.15):

        if(self.disarm()):

            self.state = "DISARMED"

        time.sleep(0.1)

def construct_target(self, x, y, z, yaw, yaw_rate = 1):
    target_raw_pose = PositionTarget()
    target_raw_pose.header.stamp = rospy.Time.now()

    target_raw_pose.coordinate_frame = 9

    target_raw_pose.position.x = x
    target_raw_pose.position.y = y
    target_raw_pose.position.z = z

    target_raw_pose.type_mask = PositionTarget.IGNORE_VX +
PositionTarget.IGNORE_VY + PositionTarget.IGNORE_VZ \
        + PositionTarget.IGNORE_AFX + PositionTarget.IGNORE_AFY +
PositionTarget.IGNORE_AFZ \
        + PositionTarget.FORCE

    target_raw_pose.yaw = yaw
    target_raw_pose.yaw_rate = yaw_rate

    return target_raw_pose

```

```

'''
cur_p : poseStamped
target_p: positionTarget
'''

def position_distance(self, cur_p, target_p, threshold=0.1):
    delta_x = math.fabs(cur_p.pose.position.x - target_p.position.x)
    delta_y = math.fabs(cur_p.pose.position.y - target_p.position.y)
    delta_z = math.fabs(cur_p.pose.position.z - target_p.position.z)

    if (delta_x + delta_y + delta_z < threshold):
        return True
    else:
        return False

def local_pose_callback(self, msg):
    self.local_pose = msg
    self.local_enu_position = msg

def mavros_state_callback(self, msg):
    self.mavros_state = msg.mode

def imu_callback(self, msg):
    global global_imu, current_heading
    self.imu = msg

    self.current_heading = self.q2yaw(self.imu.orientation)

    self.received_imu = True

def gps_callback(self, msg):
    self.gps = msg

def FLU2ENU(self, msg):

    FLU_x = msg.pose.position.x * math.cos(self.current_heading) - msg.pose.position.y *
    math.sin(self.current_heading)

```

```

        FLU_y = msg.pose.position.x * math.sin(self.current_heading) + msg.pose.position.y *
math.cos(self.current_heading)
        FLU_z = msg.pose.position.z

```

```

    return FLU_x, FLU_y, FLU_z

```

```

def set_target_position_callback(self, msg):
    print("Received New Position Task!")

```

```

    if msg.header.frame_id == 'base_link':

```

```

        """

```

```

        BODY_FLU

```

```

        """

```

```

        # For Body frame, we will use FLU (Forward, Left and Up)

```

```

        #           +Z      +X

```

```

        #           ^      ^

```

```

        #           |      /

```

```

        #           ||

```

```

        # +Y <-----body

```

```

        self.frame = "BODY"

```

```

        print("body FLU frame")

```

```

        ENU_X, ENU_Y, ENU_Z = self.FLU2ENU(msg)

```

```

        ENU_X = ENU_X + self.local_pose.pose.position.x

```

```

        ENU_Y = ENU_Y + self.local_pose.pose.position.y

```

```

        ENU_Z = ENU_Z + self.local_pose.pose.position.z

```

```

        self.cur_target_pose = self.construct_target(ENU_X,
                                                    ENU_Y,
                                                    ENU_Z,
                                                    self.current_heading)

```

```

    else:

```

```

        """

```

```

        LOCAL_ENU

```

```

        """

```

```

# For world frame, we will use ENU (EAST, NORTH and UP)
#   +Z   +Y
#   ^   ^
#   |   /
#   |  /
#   world-----> +X

self.frame = "LOCAL_ENU"
print("local ENU frame")

self.cur_target_pose = self.construct_target(msg.pose.position.x,
                                              msg.pose.position.y,
                                              msg.pose.position.z,
                                              self.current_heading)

'''
Receive A Custom Activity
'''

def custom_activity_callback(self, msg):

    print("Received Custom Activity:", msg.data)

    if msg.data == "LAND":
        print("LANDING!")
        self.state = "LAND"
        self.cur_target_pose = self.construct_target(self.local_pose.pose.position.x,
                                                    self.local_pose.pose.position.y,
                                                    0.1,
                                                    self.current_heading)

    if msg.data == "HOVER":
        print("HOVERING!")
        self.state = "HOVER"
        self.hover()

    else:
        print("Received Custom Activity:", msg.data, "not supported yet!")

def set_target_yaw_callback(self, msg):

```

```

print("Received New Yaw Task!")

yaw_deg = msg.data * math.pi / 180.0
self.cur_target_pose = self.construct_target(self.local_pose.pose.position.x,
                                              self.local_pose.pose.position.y,
                                              self.local_pose.pose.position.z,
                                              yaw_deg)

'''
return yaw from current IMU
'''

def q2yaw(self, q):
    if isinstance(q, Quaternion):
        rotate_z_rad = q.yaw_pitch_roll[0]
    else:
        q_ = Quaternion(q.w, q.x, q.y, q.z)
        rotate_z_rad = q_.yaw_pitch_roll[0]

    return rotate_z_rad

def arm(self):
    if self.armService(True):
        return True
    else:
        print("Vehicle arming failed!")
        return False

def disarm(self):
    if self.armService(False):
        return True
    else:
        print("Vehicle disarming failed!")
        return False

def offboard(self):
    if self.flightModeService(custom_mode='OFFBOARD'):
        return True
    else:
        print("Vechile Offboard failed")

```



```
    return False
```

```
def hover(self):
```

```
    self.cur_target_pose = self.construct_target(self.local_pose.pose.position.x,  
                                                  self.local_pose.pose.position.y,  
                                                  self.local_pose.pose.position.z,  
                                                  self.current_heading)
```

```
def takeoff_detection(self):
```

```
    if self.local_pose.pose.position.z > 0.1 and self.offboard_state and self.arm_state:  
        return True  
    else:  
        return False
```

```
if __name__ == '__main__':
```

```
    con = Px4Controller()  
    con.start()
```

-----end of code snippet -----

**The code defining the class for moving the vehicle is given below:**

```
import rospy  
from mavros_msgs.msg import GlobalPositionTarget, State  
from mavros_msgs.srv import CommandBool, CommandTOL, SetMode  
from geometry_msgs.msg import PoseStamped, Twist  
from sensor_msgs.msg import Imu, NavSatFix  
from std_msgs.msg import Float32, String  
from pyquaternion import Quaternion  
import time  
import math
```

```
class Commander:
```

```
    def __init__(self):  
        rospy.init_node("commander_node")
```

```

        rate = rospy.Rate(20)
        self.position_target_pub = rospy.Publisher('gi/set_pose/position', PoseStamped,
queue_size=10)
        self.yaw_target_pub = rospy.Publisher('gi/set_pose/orientation', Float32, queue_size=10)
        self.custom_activity_pub = rospy.Publisher('gi/set_activity/type', String, queue_size=10)

def move(self, x, y, z, BODY_OFFSET_ENU=True):
    self.position_target_pub.publish(self.set_pose(x, y, z, BODY_OFFSET_ENU))

def turn(self, yaw_degree):
    self.yaw_target_pub.publish(yaw_degree)

# land at current position
def land(self):
    self.custom_activity_pub.publish(String("LAND"))

# hover at current position
def hover(self):
    self.custom_activity_pub.publish(String("HOVER"))

# return to home position with defined height
def return_home(self, height):
    self.position_target_pub.publish(self.set_pose(0, 0, height, False))

def set_pose(self, x=0, y=0, z=2, BODY_FLU = True):
    pose = PoseStamped()
    pose.header.stamp = rospy.Time.now()

    # ROS uses ENU internally, so we will stick to this convention
    if BODY_FLU:
        pose.header.frame_id = 'base_link'

    else:
        pose.header.frame_id = 'map'

```

```

pose.pose.position.x = x
pose.pose.position.y = y
pose.pose.position.z = z
return pose

```

```

if __name__ == "__main__":

```

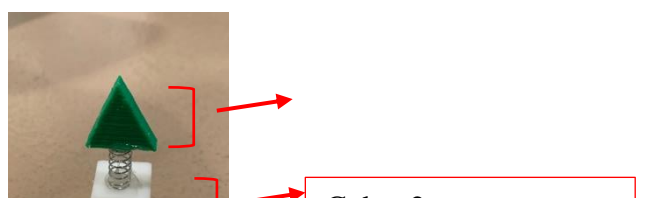
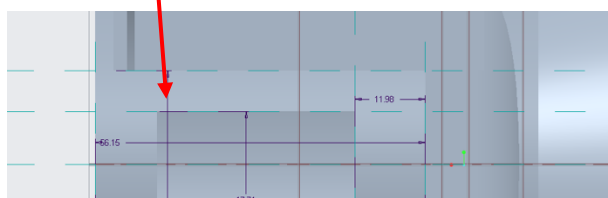
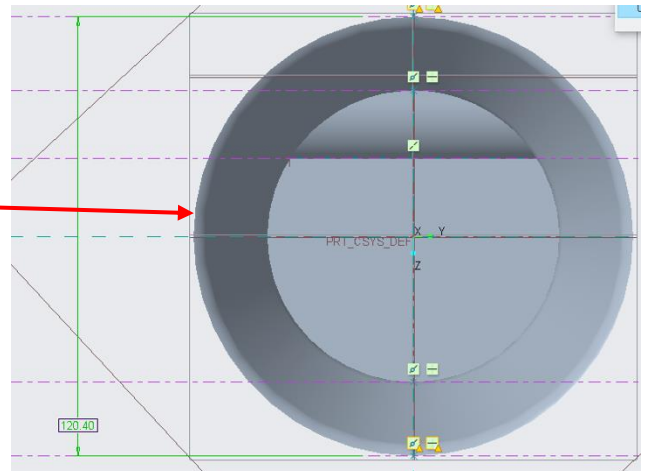
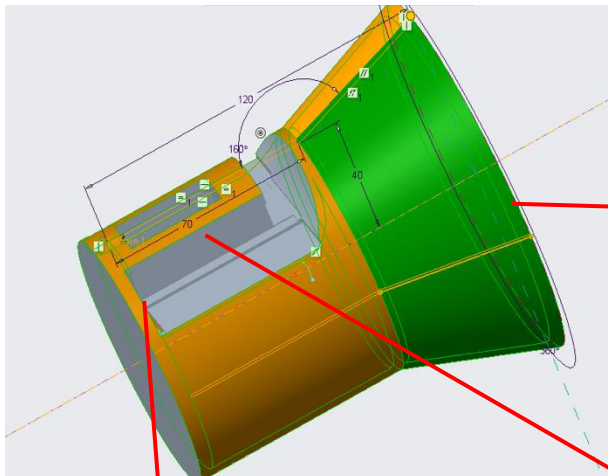
```

    con = Commander()
    time.sleep(2)
    con.move(1, 0, 0)
    time.sleep(2)
    con.turn(90)
    time.sleep(2)
    con.land()

```

## Docking system (slot)

Measurements in (mm)

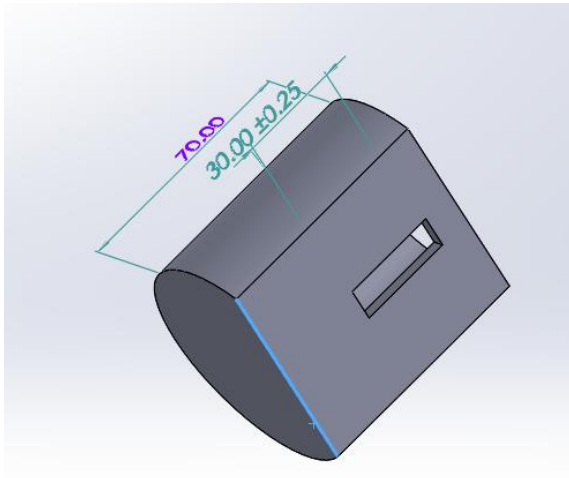


High & base: 2cm

Thickness: 0.5cm

## Docking system (lock)

Measurements in (mm)



## Meeting notes

### Meeting 1

Wednesday, September 16, 2020

11:42 AM

All the parts are made in different places, shipped and then assembled. So you need to make something bigger to accommodate the things to be shipped.

- Boeing 737, if you disassemble will give you about 1 to 2 million parts. Half million of them are unique parts.
- Having unique parts means you need to have unique manufacturing processes.
- Comparing with the automotive industry, they use common parts. So it is modular and you can share parts etc.
- Short distance personal vehicles in the future might require a paradigm shift. So we want to look into ways of building personal aircrafts like how it is done in the automotive industry.
- We don't want to be constrained by the size of the factory that is used to make these aircrafts. An idea is to make an aircraft in midflight.

- Not a new idea (vijay kumar grasp upenn look up <https://www.grasp.upenn.edu/>). The new idea is what we can do after we assemble the aircraft.
- Objectives
  - Investigate the aerodynamic efficiency of formation flying(example of formation of flock of migratory birds)
  - Investigate the aerodynamics of attaching them
  - Two UAVs that flying in close proximity and study the coupling effect of them.
  - Investigate if there is a position for one UAV with respect to the other UAV that gives a certain aerodynamic advantage
  - Investigate if there is a position that gives a disadvantage
  - Mechanism for attaching. (this will depend on the above mentioned positions, if any, because the UAV will need to move through those points when attaching)
  - Wind tunnel preferable
- Advantages:
  - Facebook has high altitude perpetual flying aircrafts.( they never land, solar powered)(aquila)
  - In aircrafts such as this, we can fix these perpetual flying aircrafts while it is flying

Make a project manager that changes every month

What each person will do

Literature review

Present what has been done from the previous meeting and what will be done before the next meeting

Consider what will be done when there is difference of opinion

What skills can you learn

## Meeting 2

Wednesday, September 23, 2020

8:42 PM

Should we purchase uavs and design a docking assembly?

We should think about the attachment mechanism.

It may refurbish an existing uav and put this mechanism on it

- Magnets are not reliable and it can disturb the signals from the

Look into LIDAR tech

Benefit of having multiple UAVs

- We can use the same modules for different purposes. This will reduce the waste and more environment friendly
- Overactuated system
- Maybe used in the air warehouses which amazon is building up

Things we are expected to do:

- Design a docking system(preferably not magnets because of the reasons above)

- We may have pilot in the loop atleast in the beginning
- CFD analysis at different distances(we may simplify as a fixed wing for the time atleast)

Proposal submission thursday

## Team meeting 1

Monday, October 5, 2020

9:55 PM

Quad copter gapter: <https://robots.ros.org/gapter/> works with ros, pixhawk( autopilot), mavlink to ask it to move around. Has a single board computer which might be useful for computer vision applications. Payload 0.6kg.

Hector quadrotor: [http://wiki.ros.org/hector\\_quadrotormodelling](http://wiki.ros.org/hector_quadrotormodelling) control and simulation of quadcopter. Might be useful

Whycon algorithm:

[https://www.researchgate.net/publication/311811749\\_WhyCon\\_An\\_Efficient\\_Marker-based\\_Localization\\_System](https://www.researchgate.net/publication/311811749_WhyCon_An_Efficient_Marker-based_Localization_System) I forgot to mention this in the meeting, but this might be useful for us for high precision alignment between the two drones.

Docking system: male-female. Maybe the umbrella looking thingy that I was talking about. Try and find prebuilt systems for the job.

Questions:

- Having the quadcopters on the same line might not give us any aerodynamic efficiency because it is not similar to the birds(requires further enquiry)
- Turning off the central rotors might be better for stability of the system after docking. Requires further inquiry as to whether the system will be able to give enough thrust with only half the rotors working
- Material to be used for the docking system(carbon fiber might be too expensive and plastic might be too easy to break)

## Meeting3

Wednesday, October 7, 2020

10:14 PM

Saeed: Quadcopter gapter: <http://edu.gaitech.hk/>. This will probably have less endurance than what we would preferably require.

- Dji 550 based hex copter(Dr. Yahya lab). Better performance overall. Uses pixhawk, ros etc. has onboard computer(NUC/Nvidia) might be useful for onboard video processing.
- Need certification to use the system.

- Technician available until the end of December 2020.

Anees: Whycon algorithm for finding relative position and rotation(might be useful)  
[https://www.researchgate.net/publication/311811749\\_WhyCon\\_An\\_Efficient\\_Marker-based\\_Localization\\_System](https://www.researchgate.net/publication/311811749_WhyCon_An_Efficient_Marker-based_Localization_System)

- Cnn based controller for precise docking
- Gazebo based simulator for good physics simulation. URDF should use the same dimensions, inertia etc. of the system we hope to use.

Abdulrahman: Docking system. Male female as was discussed previously, umbrella mechanism altered.

- Rotational actuator may be more compact and better suited for this use.
- Make models from cardboard or something before finalizing on the design.
- We may use 3-d printers if the size constraints permit it.

Mohamad: turbulences because of the docking system under the rotors.

- This can possibly be taken care of the flight controller itself.

Budget from uni is questionable, lookout for more details in the future  
 Split the team up so that each person can work on different aspects parallelly.

## Meeting 5

Sunday, November 15, 2020  
 9:11 PM

- We need to come up with all the literature review that we are doing
- We need to make a table of all the things that are to be done in the next semester.
- We may demonstrate the working of the project in a controlled environment(Lab)
- May make use of any available resources in the labs(optitrack, etc)
- Try with a single drone if we can work with the optitrack system.
- Think about having the locking actuator in the mother drone rather than the child drone.
- Outer loop optitrack info and use a camera for the inner loop, getting it aligned (d435)
- Design some kind of controller (PID) for the inner loop

## Meeting 6

Monday, January 11, 2021

5:01 PM

- **Sanad academy:** we can ask the technician to fly the drone for us. We need to make a formal request for the type of experiment we need to do explaining what equipment we will need and the lab to be used.
- **3d printer:** we have one that can do  $1m^3$  we can do 30by 30 in the printers in the lab. We should make a cad drawing that is perfect, try it on the smaller printer. If need be, we can put in a request for the bigger one.
- **Gazebo:** no need for Integral (no trajectory tracking) and proportional has to be smaller.
- Derivative is to be 15% of proportional and increase as necessary. We don't know all the details of the specific models, but we can probably get the details from the team and using it can get a transfer function. We can design a PID controller similar to what was done in the UAV control course.
- **Locking system:** we should look into providing some relative motion between them. That is it should not be rigid. Think compartments of a train. The way we have right now, might cause breakage. We should try and see how much rigidity there should be.
- We will make a formal powerpoint showing all the work that has been done from the previous meeting and then also talk about what will be done before the next meeting

## Team meeting 3

Tuesday, January 12, 2021

8:38 PM

Duties before the next meeting:

1. Mohamed Alyammahi: Vibration isolation(springs)
2. Abdulrahman Alnuaimi: update the cad with the new idea of the springs or the sponge or the aerobridge thingy as discussed
3. Sultan Alhosani: Calculation of the springs etc
4. Saeed Al Sereidi: Coupling of drones in close proximity
5. Anees Peringal: Correcting the gains in Gazebo

## Meeting 7

Tuesday, January 26, 2021

10:27 AM

Springs



Allows some relative motion.

But the vibrations will not be damped out

## Deformable material

Things like sponges can damp out vibrations between the two vehicles.

Add a structure so that we can mount the docking mechanism lower relative to the drone itself.

## Interaction between two propellers

