Open Source Mini Drone

Presented by:

Mogamad-Ameen Rawoot

Prepared for:

Dr. Jane Wyngaard and Mr. Justin Pead

November 7, 2021

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for the degree of Mechatronics Engineering.

Abstract

Unmanned Aircraft Systems (UAS) also known as aerial drones, are defined by the FAA (United States Federal Aviation Administration) as Aircraft (devices designed to fly in the air) which do not require a human pilot on-board to be able to fly [1]. It is further noted that aerial drones weighing less than 250g, known as mini-drones, are not required to be registered with the FAA before flying, as long as they are flown for the purposes of recreation [2].

Mini-drones have become increasingly more accessible to consumers in recent years and interest in the field has also improved. However, there is still a barrier between consumers and more feature-rich drones, where the typical accessible and affordable drones only offer basic flying and stunt abilities. The more features a drone has built-in, the more expensive and inaccessible they become and there are even fewer truly open source options, where every design file required to fully reproduce the drone, is shared.

This report details the research, design and building of an open source mini-drone, with an emphasis on sharing all of the available design files, documents and building procedures, such that anyone in the world could purchase the necessary components and build an identical drone. The designed drone, will also include features uncommonly seen in basic commercial drones, such as autopilot flight and stabilisation via optical flow technology. These features add a desirable aspect to the project, as they are typically only available in higher-end commercial drones and will be made available for anyone to implement in their own project.

Acknowledgments

I would like to acknowledge my family and friends for their continuous support throughout the last 4 years, obtaining this degree would simply not be possible without them. The amount of effort and long, tiring hours of continuous work, will never be truly understood by anyone other than them.

Also a big thank you to Dr. Jane Wyngaard and Mr. Justin Pead for guiding me through the process of this project and ultimately helping me through the designing and building of the Mini-Drone. I thought a set of final components for the design would never get chosen at a stage, yet it seems to have worked itself out in the end.

Finally, I would just like to congratulate myself on actually being able to complete a degree (fingers crossed). In high school I stressed myself out, thinking I wouldn't be able to get into Mechatronics. In first year I was certain I would fail many of my courses, I had the same thoughts throughout my second, third and fourth years, but here I am, hopefully completing my degree and not a single course failed, whoop whoop.

Plagiarism Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.

2. I have the IEEE convention for citation and referencing. Each contribution to, and quotation in, this final year project report from the work(s) of other people, has been attributed and has been cited and referenced.

3. This final year project report is my own work.

4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as there own work or part thereof.

Mogamad-Ameen Rawoot
November 7, 2021
Word Count: 19,813

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

This report covers the design, development and building of a fully open-source mini-drone, which will serve as a start to further development.

Unmanned Aircraft Systems (UAS) also known as aerial drones, are defined by the FAA (United States Federal Aviation Administration) as Aircraft (devices designed to fly in the air) which do not require any human pilot on-board, for the purposes of controlling the vehicle [1] [3]. It is further noted that aerial drones weighing less than 250g, known as mini-drones, are not required to be registered with the FAA before flying, as long as they are flown for the purposes of recreation [2]. Multi-rotors by comparison, are a sub section of UAS's/Drone's, which use multiple motors and propellers to achieve flight.

Recently these systems have become increasingly popular amongst consumers for many reasons, typically they are marketed as relatively affordable devices for entertainment purposes, or at a slight inflation are available as professional tools for video production, racing, surveillance, deliveries or even search and rescue operations.
These varied use cases have propelled the drone market toward a current value of an estimated 13.44 billion US dollars, a figure which is expected to expand even further

in upcoming years [4]. This is because of the numerous applications of these products and since prices are slowly decreasing and becoming more affordable for consumers.

1.2 Objectives

1.2.1 Motivation

The objective of this report is to explore the research surrounding the DIY mini-drone sphere, including functionality and components, currently available options, and current research related to optical flow sensors for location data. This research is done with the intention of designing a functioning and open source mini-drone with indoor autonomous features more readily seen on larger and higher-end drones.

The purpose of this design is to bring these uncommon features into the larger mainstream arena and for them to be more accessible to individuals interested in adding these features to their own custom drones. This research project will also act as a introductory blueprint, which will demonstrate to interested individuals the steps and considerations necessary when designing or customising a drone. This will hopefully encourage expansion and more ambitious projects in the drone sphere, adding additional and unconventional features to help with niche real-world tasks.

1.2.2 Problem Statement

The features mentioned before, such as indoor autonomous flight are not mainstream yet nor readily available. There is no formal instruction manual for designing and building these drones either, at least not in a way that can be interpreted for beginner development. The future prevalence of these features will help to advance the technologies such as optical flow sensors and other methods for spacial positioning much faster. Additionally, adding autonomous flight to more lower-end drones would make the piloting of drones easier to use and hence would attract more users to the technology.

This report aims to answer the following question: What are the steps required, to design, build and test a functioning mini-drone, which features open source firmware and an optical flow sensor?

1.3 Thesis Contributions

This report contributes the following outcomes:

- Conglomerating key concepts required to understand the flight of multi-rotor drones.
- Detailing the multiple different avenues and processes involved in designing a mini-drone.
- Showing the design steps and decisions needed to design a drone.
- Providing all of the files and instructions to reproduce and build the final product. These files and instructions will also be available to be modified if the reader wishes to expand the project and add additional features.

1.4 Scope and Limitations

Scope

The scope of this project, outlined in the project brief, is as follows:

- A literature review briefly summarising drone autopilot software, currently available mini-drones and recent advancements in the field that utilises optical flow sensors for localisation.
- A functioning mini-drone.
- All design files for the mini-drone, including mechanical, electrical, and software build instructions in a documented public GitHub repository.
- A qualification of the performance of the resulting design.

- The final thesis report, poster, and presentation.

Limitations

This project was constrained in many ways from the outset. The constraints affected the final decisions when designing and building the mini-drone. The main constraints were:

- The project had a budget of R1,500 for the purchase of components required (above what was freely available for the project); with an exception for projects which simply could not meet this requirement.
- A very strict deadline of 14 weeks meant that the components had to be chosen quickly and that there was no room for error.
- Since the project was conducted during the Covid-19 pandemic, work was mostly conducted remotely with the exception of 3D printing (and other prototyping services), and parts collection.
- The fact that the project was conducted in South Africa and during the pandemic meant that the parts' availability, pricing, shipping lead times and international shipping were all major factors in the decision making process when choosing the desired drone components.

1.5 Thesis Outline

The remainder of the report opens with a literature review, in which relevant literature covering the theory, important design considerations, the currently available drones and software, and up-to-date research on optical flow sensors for localisation is discussed.
The methodology in Chapter 3 follows, describing the intentions behind strategic choices in workflow, user requirements, design and testing, and details how each of the phases were conducted.
The design phase covered in Chapter 4 details the design of each subsection of the

drone, including the component selection, simulation, hardware and software design, and implementation.
The results in Chapter 5, then explore the testing procedures and cross-examines the results for further developments.
Finally, in Chapter 6, conclusions are derived from the results, and recommendations and future research is suggested.

Chapter 2

Literature Review

2.1 Proceedings

This Literature review considers a brief history of drones, particularly multi rotors, the basic functional components involved in the design of a drone, as well as the theory describing the requirements needed for a drone to operate. The following sections note the comparisons of various components, the various drone autopilot software systems currently available, as well as their differences, some of the presently purchasable mini-drones on the market and some recent projects and studies using optical flow sensors for the purposes of localisation are considered.

2.2 History

The first known instance of a UAV (Unmanned Aerial Vehicle)or drone being deployed was for military purposes. The Austrian army first used unmanned hot air balloons in 1849 with explosives attached to them, to attack Venice. However, the UAV technology available at the time was inadequate, with a change in wind direction sending many balloons off from their desired mark [5]. The first radio controlled drone was then created in 1916 by Archibald Low, for the purposes of World War One, but it wasn't until World War Two that drones started to become popular. In 1935 the British army started using what they referred to as "queen bee" drones

during their military target practices. The word used for UAVs stems from this relation to bees, since the drone amongst bees refers to those members that usually wait idly for their objective [6].

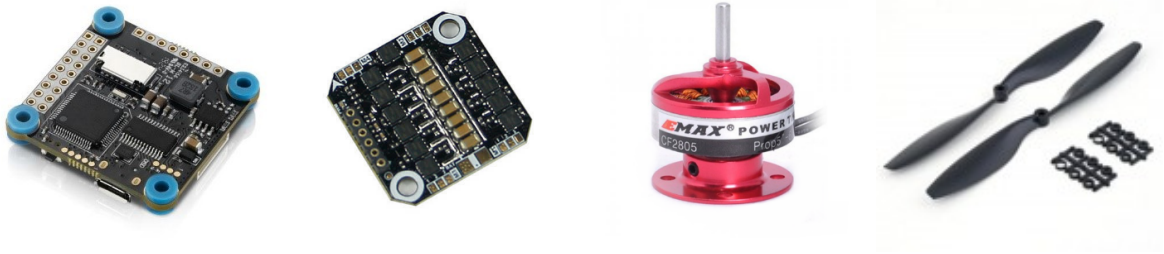
Later the German forces created a missile type drone, known as doodlebugs and in the Vietnam war, drones were used for surveillance and data capture of enemy terrain.
From the 1960's, RC hobbyist drones started to become more common due to the shrinkage of transistors, allowing drones to become increasingly smaller [7]. Currently, drones are being used for a variety of different purposes such as delivery, videography, and entertainment, with micro drones used for military applications. Drones are becoming increasingly popular as they become more accessible to the average consumer and as the use cases become more plentiful.

2.3 Theory

The basic architecture of a multi-rotor styled drone is seemingly simplistic when viewed from a consumer point of view, but careful consideration needs to be taken when choosing the main components. A strategic and calculated balance amongst the four necessary forces which act on an aircraft is essential, since the aircraft will be unable to take off/operate at all if there is an imbalance. [8].

For flight to occur, the weight of the system ('system' is used throughout the paper to refer to the fully outfitted mini-drone) must be lower than the amount of lift, and the drag must be lower than the thrust. Consequently, the design should maintain equilibrium between the increased weight and drag, which coincides with the different methods for improving thrust and lift; i.e. adding more motors or larger propellers improves thrust and lift, but adds weight and drag. The specific energy of a drone's energy source (motor/propeller combination) should also be large enough to ensure flight, despite efficiency losses caused by converting the energy into thrust/lift [9].

The basic operation of the components of a drone, includes the electronic components which are powered by the battery. The motors are physically attached to the frame along with the rest of the physical components, while the propellers are fitted onto the motor shafts. The motor spins these propellers in a specific orientation and speed (depending on their mounting position), causing the quadcopter to take flight using the physics principals discussed in the Physics Principles section below.



(a) Flight Controller [10] (b) ESC [11] (c) Motor [12] (d) Propeller [13]

Figure 2.1: Required Components 1

Flight controllers (autopilots/FC), seen in Figure 2.1a, control the functionality of the drone, and have a number of sensors such as a gyroscope, accelerometer, barometer and compass while others have additional GPS modules and other accessories. These sensors are used to determine a drone's position in a 3D space, and utilise the information gathered to control the motors to either lift, roll, pitch or yaw the drone. It does this by means of control loops and output PWM (pulse width modulation) signals, sensing changes in the position of the drone and altering the output PWM signals to account for them or for signals received by the pilot.

The electronic speed controllers (ESC), seen in Figure 2.1b, convert a PWM control signal, which the flight controller sends out, into a voltage and current that the motor can understand. The motor speed controllers differ depending on the type of motor being controlled. Brushed motors can be controlled with a single mosfet and diode combination and a simple PWM signal, from the flight controller, to control the armature speed [14]. Contrarily, a brushless motor requires the use of multiple mosfets, some motor position circuitry and the FC's PWM signal. This is because

brushless motors have multiple coils and permanent magnets.
Consequently, different coils need to be energised at the same time to rotate the motor, hence the need for multiple mosfets. The position of the coils relative to the magnets needs to be ascertained to energise the correct coils, hence the need for position circuitry. [15]

The choice of different motors is often reduced to brushed and brushless DC motors, which are the most commonly used options in the mini-drone field due to their size and performance. A brushless motor can be seen in Figure 2.1c. The propellers that the motors spin, seen in Figure 2.1d, cause the system to become airborne, the process of which will be detailed in Section 2.6 of the literature review.

The frame, seen in Figure 2.2a, is the rigid structure which holds all of the components together, it is critical that this constituent be as light as possible as it typically accounts for a large portion of the system's weight.

The battery, seen in Figure 2.2b, provides power to all of the system's electronics and can be constructed from various technologies such as LiPo and Li-Ion. In the event that the battery voltage available is not suitable for the electronics (apart from the motor and the ESC), a voltage regulator would be needed to either decrease or increase the voltage to a suitable level.

The receiver, as in Figure 2.2c, receives the signal from the transmitter, as in Figure 2.2d, and relays it to the flight controller. The transmitter in Figure 2.2d is a remote control, however, there is the potential for USB dongle usage that would connect to a laptop which would allow for equivalent control of the drone. The transmitter and receiver can operate using different protocols depending on the design, with the most common protocol being RF (Radio Frequency).

Finally, the camera, seen in Figure 2.2e, is optional but which offers additional useful features. The Camera can be used to capture high resolution video footage

which can be attached via a gimbal system to stabilise the footage. It can be used as an FPV (First person view) system which would connect to a VR Headset or screen, and can immerse the pilot for increased accuracy and complex manoeuvres. This project specifically utilises an optical flow camera, faced downward, for stabilisation features including being able to keep the drone better positioned.

(a) Frame [16] (b) Battery [17] (c) Receiver [18] (d) Transmitter [19] (e) Optical FLow [20]

Figure 2.2: Required Components 2

As can be seen by Figure 2.3, which shows the anatomy of a FPV quadcopter, the components detailed are connected strategically. The motors are wired to the ESC's, which are connected to the flight controller. The other electronics are all connected to the flight controller, and are battery powered.

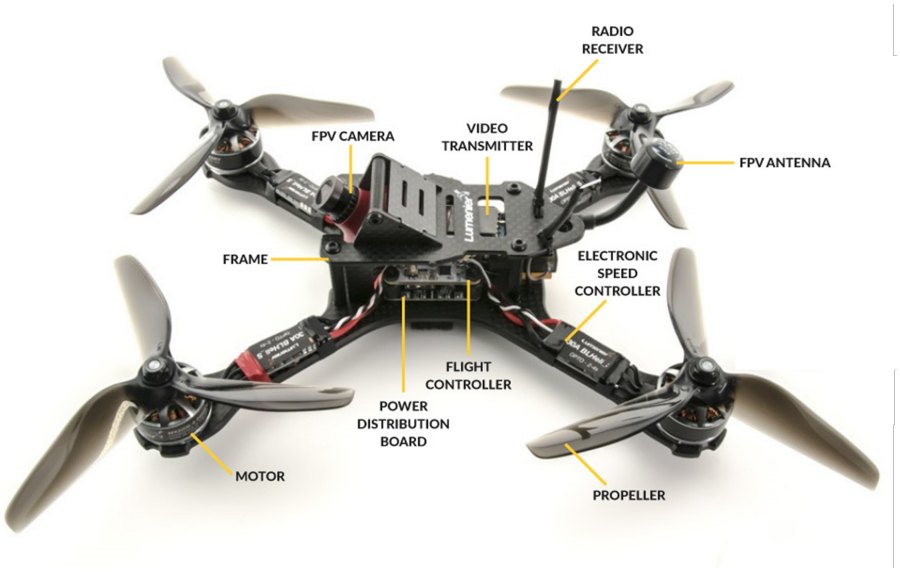


Figure 2.3: Anatomy of an FPV quadcopter [21]

Physics Principles

Quadcopters make use of simple physics principles to achieve flight. The rotors spin and create a downward force onto the air below, in accordance with Newton's 3rd law which dictates that every action must have an equal and opposite reaction [22]. Consequently, air pushes back with a countering upward force onto the propellers, which causes the aircraft to become airborne. When this reactionary force becomes stronger than the gravitational force and the rotors increase their rotational speed (also the more aerodynamic the rotors are), more force is applied and the drone is able to lift higher [23].

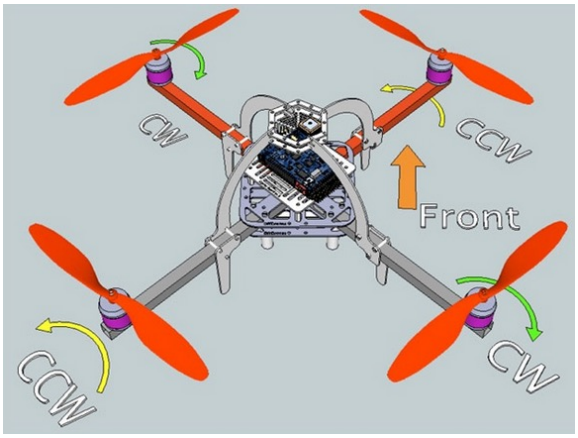


Figure 2.4: Rotation of quadcopter propellers [24]

However, if all the rotors were to spin in the same direction, the torque of the motors and the inertial component of the propellers would cause the quadcopter to spin its frame in the same direction. To counter this effect, two of the motors (diagonal to each other) spin in the same direction, with the remaining two motors spinning in the opposite direction. Yet, this poses a new problem, where two of the rotors would push the air upwards due to the propeller design. To resolve this, different propellers should be used for the specific directions, all of which need to push the air downward. [25]

2.4 Brushed versus Brushless Motors and their Controllers

Sincero, Cros and Viarouge, in their paper "Efficient simulation method for comparison of brush and brushless dc motors for light traction application", compare equivalent brushed permanent magnet DC motors against brushless PM motors, at the same operating point and while taking into account the mechanical, magnetic and power converter losses (inverter vs DC-DC) [26]. It was found that the DC-brushed and brushless motors tested had very different operating characteristics.

Brushless Motor								
Input Power	Diode losses	Mosfet Losses	Joule Losses	-	Output Power	Input Current	Torque	Efficiency
497.2W	1.15W	10.6W	57.3W	-	405W	11.1A	16.7N.m	81.5%
DC Brushed Motor								
Input Power	Arc Losses	Mosfet Losses	Joule Losses	Brush Losses	Output Power	Input Current	Torque	Efficiency
538.6W	29.3W	9.2W	58.9W	17.5W	403.9W	15.3A	15N.m	75%

Table 2.1: Comparison of Brushless Motor to DC Brushed Motor [26]

The brushless motor had a total power of 428.15W, with 69.05W in losses, while the brushed motor had a total power of 423.7W with 114.9W in losses, as seen in table 2.1.
These results show that brushless motors provide a higher torque and efficiency, while using less power/current and with fewer losses, albeit at a slightly higher weight. [27] Another design consideration is the cost of the motors and the controllers. Equivalent thrust brushed motors cost less than their brushless counterparts, due to the manufacturing being better established, easier and efficient.

The speed controllers that these motors use also have a cost difference, this is because of the different methods for controlling the motors, as mentioned in the ESC theory section above. The ESC's required for brushless motors cost more than the single mosfets needed for brushed motors.

2.5 Power to Weight Considerations

Research conducted at the University of Pennsylvania, in a paper titled "Power and weight considerations in small, agile quadrotors", provides many useful analytics [28]. It is essential to note that most components scale with the diameter of the quad, for instance the mass will increase the larger the frame of the quad is, by a factor of d^3, as will the thrust and drag, but by a factor of d^4, while the size of the propellers will scale linearly.

The paper notes that the battery weight contributes approximately 33% of a total system's weight, whereas the motors and propellers contribute 29%, while the frame and the electronics weigh 17% and 15% respectively. However, the motors and propellers contribute the most towards a system's inertia, since they are located at the outer diameter of the frame [28]. The weight of a system is critical to it being able to take flight and to the flight time, yet reducing the battery size results in the flight time being decreased. Weight and performance consequently need to be balanced. However, this can be challenging to achieve in an instance where the desired components cannot be physically weighed or where their size cannot be viewed; i.e. choosing components online is challenging when little information of the product is given.

The balance between weight and performance can be calculated based on the other system parameters. However, it requires the knowledge of many parameters which might be difficult to obtain for components sourced online, such as battery energy density, system energy consumption, system electronic efficiency, system flight height, flight distance, flight speed and total system mass excluding battery [29]. Unless these parameters are known, it would only be possible to optimise a system via experimental simulation in a system simulator such as eCalc. However, this method is flawed by component availability, since certain components are not realistically obtainable; i.e. a 100mah 6s battery appears to not exist on the market. It would be strategic to begin with a list of available and compatible components, and determine

the optimal combination of components with what is available.

2.6 Propeller Size versus Efficiency

The research paper "Experimental study on relation of small UAV propeller thrust generation and the mouth-ring", conducted at the University of Malaysia Perlis and the University of Sunderland, compares the size of propellers with their efficiency [30]. The authors compared propellers of various sizes, as well as ducted propellers with different spacing between the propeller tip and the ducting. It was found that using a 10 inch propeller, throughout a voltage range of 4-11V, consistently provided a higher thrust than a 9 inch propeller with the same pitch ratio, which is due to the increased surface area on the 10 inch propeller that pushes more air without adding excessive weight.

It was also noted that using a duct or a mouth ring on the propellers, which is a cone shaped structure surrounding the propeller, similar to that seen on aeroplanes, provides an increase in thrust when there was a large enough gap between the tip of the propeller and the inner diameter of the ducting for the specific voltage used. However, it is significant to note that an excessively large gap would negate the effects [30]. As such, using a duct to improve thrust is a viable option, but only if the gap and the weight of ducting is optimal.

Ultimately, a propeller with a larger surface area is able to produce more thrust per gram than a propeller with a smaller surface area. This is because the diameter of the propeller is related to its efficiency and velocity, while thrust is equal to the Power*efficiency/velocity. Consequently, a larger diameter results in more surface area, more thrust and higher efficiency [31].

2.7 Drone Software

Autopilot systems require coordination between hardware and software in order to function. This section covers the various software available for drone systems.

2.7.1 Flight Controller Firmware

Flight controllers are the main functional components of quadcopters as they contain the microcontroller and the inertial measurement sensors such as the accelerometer, gyroscope, magnetometer and barometer. They also host the connectors for various other important components such as receivers and regulators that are necessary for flight.

The flight controller firmware is run on the microcontroller. The firmware is responsible for many tasks such as interpreting the receivers' signals from the transmitter (or laptop) and creating and adjusting the PWM signal that is emitted to the mosfets or ESCs of the motors, which in turn changes the position of the drone. The Firmware also caries out different calculations to stabilise the drone, and can further interpret the signals from an optical flow sensor to stabilise the drone when engaging in autonomous flight. [32]

Some of the various flight controller firmware options on the market are listed below:

BetaFlight

The BetaFlight firmware is the most popular quadcopter firmware available and is derivative of the cleanFlight firmware. It has a very large selection of configuration items in its arsenal, such as PID calibration, different drone configurations, and most importantly has one of the largest selections of supported flight controllers [33]. It does not, however, have many autonomous flight features.

MultiWii

While MultiWii has been partially phased out, it maintains viable applications such as being compatible with smaller, less powerful microcontrollers such as the Arduino (Atmega 328p). MultiWii was initially designed to make use of the IMU sensors in the Nintendo Wii remote, alongside a microcontroller for use as a flight controller for a quadcopter [34].

2.7.2 Autopilot Focused Flight Controller Firmware

Autopilot components control the trajectory of an aircraft, with only the location or path being determined by a pilot. This allows the pilot to focus on other aspects, such as camera angle in the case of a film quadcopter, and so that the pilot does not need to input controls continuously [35].

In terms of multi-rotors, the flight controller should have a minimum set of sensors for autopilot functionality, and needs an autopilot focused firmware to control the drone. The autopilot firmware receives flight path instructions via communications with the ground control software, and converts these instructions to motor inputs.

The following two firmware are the most popular autopilot focused firmware currently available.

PX4

PX4 is an open-source autopilot firmware that is aimed at lower cost autonomous flight for drones. It offers features such as 2D and 3D aerial maps as well as drag and drop waypoints. PX4 has a large support community and is compatible with a number of flight controller boards; it is a viable option offering open-source flexibility [36] [37].

ArduPilot

ArduPilot is also an open-source autopilot firmware, but is split into five different sub-programs, namely ArduCopter, ArduPlane, ArduRover, ArduSub and Antenna Tracker. ArduCopter claims to support functionality for FPV focused drones, photography/videography focused drones as well as fully autonomous flight. It also has a more well-established and large community following for support, which is what contributed to utilising it for this project [38].

2.7.3 Autopilot Ground Control Software

A ground control station (GCS) is typically a control centre for UAVs, which is unattached to the physical drone. The "pilot" of the UAV can send commands to the vehicle via these stations, essentially controlling it wirelessly [39].

In terms of smaller DIY drones, the ground control station refers to the software that communicates with the drone for instructions. The software dictates to the "pilot" where the drone is and other telemetry information including the battery life reading and the altitude. They also allow for live setpoint changes to move the drone and for settings to be adjusted [40].

The autopilot firmware mentioned are typically used with the following GCSs respectively, but can be used with others.

QGround Control

QGround Control offers support for PX4 and ArduPilot based drones, and has features to setup vehicles using these firmware as well. It features mission planning capabilities, flight maps, video streaming, multiple vehicle controls and runs on a wide range of platforms [41]. It is typically used alongside PX4 drones.

Mission Planner

Mission Planner is a part of the ArduPilot programme suite and consequently has features that interface with all the different ArduPilot based vehicles (for example multi-copters, rovers and planes). It includes tuning functionality for its vehicles, and the ability to save setpoint missions which is useful for videography drones trying to capture a set video sequence. It also has most of the features available which QGround Control has. Additionally, it has a first person mode, telemetry support and works with simulation software to simulate a UAV's flight [42]. Mission Planner was chosen since it is part of the ArduPilot suite, and so is much more integrated with the ArduCopter firmware than QGround Control.

2.8 Available Mini-Drones

For the purposes of this report, a mini drone will be considered as any multi-rotor that weighs less than 250g.
There are currently many mini-drone options available, most of which are more affordable, offering minimal functionality, and which rather focuses on speed and stunts. However, in terms of task specific mini-drones that can handle an exact workload, such as sufficient thrust to carry a high quality camera and gimbal for videography or being light and agile enough for FPV racing, building a custom drone is better suited to fulfilling these criteria. This needs to be evaluated against the other available options and their prices, as drones such as the DJI mavic mini 2 and the Flywoo Explora LR4 are suited for the two drone classes above.

Hubsan Range

Hubsan offers a wide variety of drones and mini-drones. Their Zino Mini SE drone focuses on videography and features a 4K camera, brushless motors, very stable flight and other autonomous flight features. The SE marginally fits the mini-drone category criteria, weighing 249g, however, the high quality and stable footage are features that determine a final price of 6800 rand. Hubsan has some FPV mini-drones like the

H123D and the H122D, which are light weight (sub 150g), brushless motor drones used for racing, each costing less than 2800 rand. They also have more affordable, entry-level indoor drones like the H502E and H502T, which are also light weight but which use cheaper brushed motors to lower the cost [43].

The DJI mavic mini 2 offers a similar design to the Hubsan Zino Mini SE, as well as many similar features, but many users consider the DJI drones to be superior regardless of the cost disparity.

BetaFPV Range

BetaFPV have many mini-drones available, which use the betaflight firmware mentioned earlier. BetaFPV offers a beginner FPV series, which focusses on affordable, lightweight FPV packages, lowering the barrier of entry into FPV drones, and which cost less than 200 dollars each. They also have a Toothpick and Twig series, which are ultra-lightweight drones each made for a different purpose, such as distance flying, FPV racing or 360 degree filming. Most of these cost less than 250 dollars, with few costing more. There is then the Cinewhoop series (all under 290 dollars each), with drones aimed at carrying smaller, high-quality cameras, such as the SMO 4K Camera, for filming video sequences. Their brushed series is then the most cost-effective series, with the smallest of these drones costing less than 130 dollars, and which are mainly aimed at entry-level pilots for indoor use [44].

2.8.1 ArduPilot Mini-Drones

The above mentioned drones all utilise their own custom firmware, or use betaflight and other open-source firmware, but none of them were designed to use ArduPilot. This is because ArduPilot is typically used with larger, fully featured drones that have faster processors, and which are commonly used by hobbyists with components such as a Raspberry Pi or other mini computers which ArduPilot is equipped to interface with. The following two mini-drones pose as exceptions to this.

Sky-Viper Range: Fury and SkyRocket

Sky-Viper were pioneers in using ArduPilot as the main firmware on their commercial mini-drones, initially including it on their SkyRocket drone in 2017. The SkyRocket or the V2450GPS was less than 150 dollars and featured more affordable brushed motors, GPS, video streaming, position hold and return to home features, which is considerable given the price.

The SkyRocket drone has since been discontinued, but Sky-viper released the Fury, Scout and Journey drones which similarly all use the ArduPilot firmware. Fury is the entry-level drone costing less than 75 dollars and features auto-hover and position hold. The Scout drone is their intermediate model that costs less than 80 dollars and has an additional camera with streaming functionality. Journey is then Sky-Viper's premium offering (most similar to v2450GPS) costing approximately 100 dollars, and has a GPS module, follow the pilot mode, waypoint flying and extended range features [45] [46].

BitCraze Crazyflie 2.1

The Crazyflie 2.1 is the most up-to-date version of Bitcraze's mini drone at present, weighing 27 grams excluding the battery and costs 225 dollars. The Crazyflie also uses the ArduPilot firmware, however, it is far more open-source than the Sky-Rocket alternative, with nearly all of the schematics and files required to construct it freely available. It features Bluetooth low-energy as well as radio, meaning the drone can be flown either with a smartphone, or via a computer that has an adaptor and an Xbox or Playstation controller.

The Crazyflie also features an expansion port with an SPI and I2C connection, which allows a number of their expansion decks to be connected to the drone allowing for many additional features. Some of the expansions decks include a "flow deck", which includes an optical flow sensor and range finder, and which allows for stabilisation, position hold and somewhat autonomous flight. Other expansion decks include an "AI deck" which adds an artificial intelligence core and ESP32 WiFi connectivity, whilst there are also a number of positioning decks which detect nearby objects,

or which can track the position of the Crazyflie in 3D using an HTC Vive system [47].

Table 2.2 below summarises the discussed features of these drones. The Sky-Viper drone is the most cost effective with many features.

Drone	Price	Weight	Motors	Flight time	Materials	Features	Link
DJI Mini 2	R9000	242g	Brushless	31 min	Plastic	4k video, autonomous flight	Link
Flywoo Explora LR4	R3200	-	Brushless	15 min	Carbon Fibre, Plastic	4k video, GPS	Link
Hubsan Zino Mini SE	R6800	249g	Brushless	45 min	Plastic	4k video, GPS, Optical flow	Link
Hubsan H123D	R2700	145g	Brushless	10 min	Carbon Fibre	FPV camera	Link
BetaFPV Pavo30	R5360	173g	Brushless	6-8 min	Carbon Fibre, Plastic	4k Video	Link
Sky-Viper Journey	R1500	=198g	Brushed	15 min	Plastic	GPS, optical flow, Video	Link
Bitcraze Crazyflie 2.1	R3400	27g	Brushed	7 min	FR4, Plastic	Expansion slot for optical flow or GPS	Link

Table 2.2: Comparison of currently available mini-drones

2.9 Recent Work using Optical Flow Sensors

"Localization system for indoor service robots" & "Afocal Optical flow sensor" (Dong-Hoon Yi, Tae-Jae Lee and Dong-Il "Dan" Cho)

These papers detail the designing of an afocal optical flow sensor for range approximation, as well as the utilisation of that sensor as the base of a localisation system for use with indoor robots in undesirable conditions.

The first paper uses the sensor which would typically be used in a computer mouse as the base of the optical flow sensor system. It details a solution to reducing the vertical height variance error of the sensor, through the use of an infinite effective focal length system (afocal), with promising results [48].

The second paper elaborates on using the above system on an indoor robot in a low luminance and slippery environment. The slippery environment avoidance was achieved using wheel encoders to track the position of the motors and compare them to the desired position, as well as using a gyroscope and front facing camera to determine the exact position of the robot. The low luminance determined that the image obtained from the robot be passed through a rolling guidance filter after histogram equalisation. The whole system was designed to be used with affordable processors in everyday robots [49].

Autonomous UAV using Optical flow sensor for positioning and navigation (Nils Gageik, Michael Strohmeier and Sergio Montenegro)

This paper details strategising a solution to the problem of controlling all degrees of freedom of a UAV without using a reference system that needs to be setup independently from the drone; i.e. the entire positioning system would be processed on the drone itself and would not need external sensors. This technique makes use of an optical flow sensor for the 2D positioning of the drone, while height sensing is achieved with ultrasonic, infrared and pressure sensors to compute the full 3D position of the drone in space. The paper was designed for the AQopterI8 project, where the objective was to design an autonomous indoor drone (much like the objective of this research report) [50].

Indoor Location-Based Positioning System Using Stereo Vision with the Drone Camera (Young-Hoon Jin, Kwang-Woo Ko, and Won-Hyung Lee)

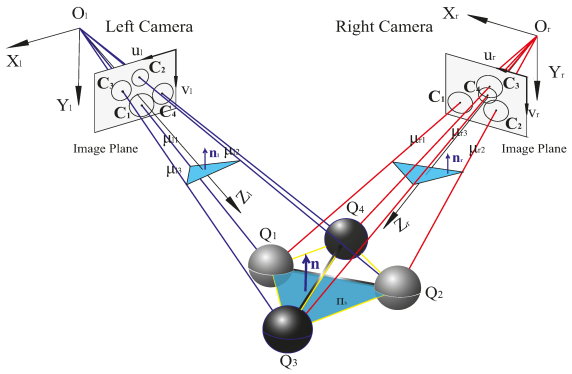


Figure 2.5: 3D Stereo Vision [51]

This project involved using the camera of a drone as a 3D stereo vision sensor, allowing for objects or even entire areas, to be modelled in 3D space for use with indoor robots. This was achieved by estimating the depth value from the camera module as described in Figure 2.5, where the footage from multiple cameras is used to reconstruct a 3D model of an object. The paper focused on deriving a method to control the drone indoors, by using structure from motion (sfm) algorithms. SfM obtains the relative position between cameras. Initially a 3D space was reconstructed using images, and then the drone's position was determined by comparing its own camera feed to the 3D space created from those images. This technique is useful for indoor autonomous flight or controlling multiple drones in a single area; a research area which is suitable for further development of this project [52].

2.10 Summary

Considering the currently available drones and recent research on optical flow sensors, there is a niche gap in the research which this report aims to fill. The literature review covered the available products and software which can be used in the making of a custom drone, and unpacked papers regarding the design of a custom drone. The report will focus on designing a self-assembly drone, using as many readily-available

components as possible and keeping all of the design files and documents open-source. The drone should attempt to include those features which are commonly available on more expensive and closed-source products, such as autonomous flight within a certain area (GPS required for further distances) and the inclusion of an optical flow sensor for the purposes of stabilisation and assisting autonomous flight.

The system will make use of the ArduPilot firmware stack as well as the Mission Planer ground control station for the purposes of controlling the drone in an autonomous manner. The design of the drone will also take into account the various factors of research which were mentioned in this section, namely the propeller size, weight considerations and the comparisons of motors.

Chapter 3

Methodology

This project involved various phases of implementation. Initially the project requirements were defined to ensure that the final user specifications be met when completing the remaining phases. The literature surrounding the desired topic and requirements was then reviewed and compiled into a cohesive chapter, which was followed by the design of the drone, the physical implementation and the iterative process between these phases.

The topics covered in the literature review were critical to the component selection process as well as the design, as certain studies such as the power to weight ratio, brushless versus brushed motors and propeller size, influenced the specifications that were determined.
Finally, the results were obtained through testing of the system and conclusions were drawn from the results. This section specifies these phases, so as to detail the constituents and aims of the project comprehensively.

Figure 3.1 details the workflow phases used throughout the project, which were influenced by the agile model. The agile model focuses on continuous development through design, implementation, testing and review and uses an iterative approach to its development. This model was used as it allowed for the final system design used, to be a culmination of knowledge derived from various successful and unsuccessful

design attempts. The model achieved this by implementing an iterative loop of the design, implementation and testing phases, which improved the overall system through real world experience and results.

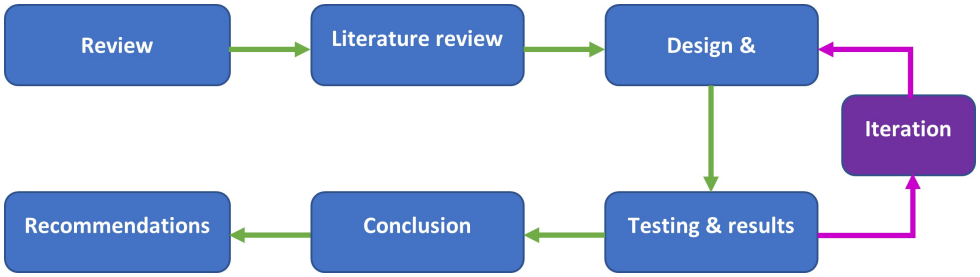


Figure 3.1: Project work flow

3.1 Project Requirements and Terms of Reference

The main aim of the project was to design an open-source mini-drone (sub 250g), using the ArduPilot firmware stack and incorporate a downward facing optical flow sensor (Bitcrze Flowdeck) to allow for the drone to hold its position and maintain stability. ArduPilot is a professional grade open-source autopilot firmware for autonomous vehicles. The initial deliverables as per the project brief were listed in the project scope.

After assessing the project scope and reviewing literature regarding mini-drone theory, hardware, software, optical flow sensors and currently available products, some requirements were produced to elaborate on the scope of the project and were later used to determine the testing procedures for the drone. Initially a set of user requirements were constructed, as detailed in Table 3.1.

Requirement Number	User Requirement
UR01	Develop and build a functioning mini-drone which uses an open source firmware.
UR02	The drone should be untethered, i.e. it should not be powered via a power source on the ground
UR03	The system should be somewhat scaleable, i.e. additional features added at a later stage and core components be used on larger drones.
UR04	The drone should be able to at least hover for 10 minutes of continuous flight
UR05	The drone should be easily controllable by a human pilot
UR06	The full weight of the system should be less than 250g to classify as a mini-drone
UR07	The system should have some autopilot functionality, such as flying in a circle without user input or following a preset path
UR08	It should be able to automatically stabilise itself in the air, i.e. no oscillations but drift is acceptable
UR09	It should be able to hold its position in 3D space on command
UR10	The full project should be open source
UR11	The project should be well documented

Table 3.1: User requirements for the mini-drone system

Functional Requirements

The user requirements listed in Table 3.1 were further processed into the functional requirements of Table 3.2.

Requirement Number	Relevant User Requirements	Functional Requirement
FR01	UR02 and UR04	The system should have a large enough battery to power the system for 10 minutes, while having sufficient thrust to handle the batteries weight.
FR02	UR02 and UR05	The system should be controllable by a wireless transmitter with joysticks for easy learning.
FR03	UR06	The design of the systems frame and component choices should be very strict on weight to remain under the 250g limit.
FR04	UR01 and UR07	The system should use the ArduPilot firmware which is an open source autopilot software for UAV's.
FR05	UR08 and UR07	The system should have 10 degrees of freedom in sensors to know its exact orientation in space, such that the system can counter any stability issues.
FR06	UR07 and UR09	The system should include an optical flow sensor so that it can hold its horizontal position steady.
FR07	UR08 and UR09	The system should also include a range finder so that its vertical position can be held steady.
FR08	UR10 and UR11	All of the project files and information should be uploaded to GitHub so that the project can be reproduced and expanded upon by other interested parties.
FR09	UR03	The core components should have additional ports available for expansion and should be compatible with various motor controllers.

Table 3.2: Functional requirements of mini-drone system

Design Specifications

Finally the design requirements, which were used to further the systems design, were decided upon after reviewing the functional requirements of Table 3.2. The design requirements created are detailed in Table 3.3.

Requirement Number	Relevant User Requirements	Design Requirement
DR01	FR01	The systems design must include an efficient power regulator (Power Brick mini) to power all of the smaller electronics as well as the motors without too much wasted energy.
DR02	FR02	The system must include a SBUS receiver (FrSky XM+) to function with the vast majority of flight controllers and transmitters.
DR03	FR03	The frame should be made with either carbon fibre or light 3D printed plastic to ensure it meets weight requirements. The size of other components should also be kept small and light.
DR04	FR04	The flight controller of the system should be compatible with the ArduCopter firmware of the ArduPilot stack.
DR05	FR05	The system/flight controller must include a 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer (compass) and an altimeter/barometer.
DR06	FR06 and FR07	The Bitcraze Flowdeck v2 should be used as it has both a PMW3901MB Optical flow and a VL53L1X Time of flight sensor on board.
DR07	FR08	The project should consist of mostly off the shelf parts such that it can easily be reproduced by another person

Table 3.3: Design requirements for the mini-drone system

3.2 Design and Implementation

The component selection for the system similarly followed an iterative approach, where the selection process was repeated numerous times due to component unavailability, compatibility issues and weight adjustments. These changes were inevitable during the component selection process because of the high level of interdependence of the individual components, where any minor changes would have reverberations on the functionality of the system.

For instance only ArduPilot compatible flight controllers were set as a design requirement, but there were only a few compatible devices, not all of which were readily available.

An initial design was created which optimistically used brushed motors which were freely available for use during this project, however this was ultimately deemed unfeasible due to their low thrust and incompatibility with the propellers readily available. As a result, multiple other simulations were considered, which were created using the component specifications provided by online product pages and manuals, to calculate the overall thrust, flight time, weight and controllability of the systems.

The multi-rotor drone simulation program, eCalc, was used to determine the simulations' performances for the various designs, and these values were tabulated in excel for comparison. Following this, the compatibility of the components for the most successful simulated designs were confirmed and the relevant changes were made to resolve issues that arose. On various occasions this lead to the need for re-simulation, since small changes inevitably affected performance. A final design was then chosen and components were ordered.

Thereafter the mechanical frame design was completed to meet its weight specifications and to hold all of the elected components. Finally the system was assembled after obtaining all of the necessary connectors and manufacturing equipment.

3.3 Testing Procedures

The testing of the final mini-drone was conducted through various experiments. These experiments were decided as optimal for testing that the user requirements were met. The results from the final system, simulated in eCalc, were included as a base for some of the performance tests conducted to be compared to.
The test for speed and acceleration was included to ascertain whether the drone was fully functioning and whether it had a speed relative to other drones currently

available.

The hover time was tested to ensure that the drone met its 10 minute hover requirement mentioned in the user requirements set. The payload experiment was conducted to test the number of additional sensors or components the drone could carry should the feature set of the drone need to be expanded.

The tests involving the optical flow sensor were included to discern whether the sensor was preferable over the on-board position estimation systems of the flight controller and under which conditions the sensor would function correctly. The order in which these tests were conducted is as follows:

- Final Component eCalc Results
- Test hover flight time.
- Testing speed and acceleration
- Judging agility/ease of control
- Payload testing
- Quantifying holding position accuracy using different sensors (barometer, range finder, optical flow)
- Optical flow terrain testing
- Additional modes

Chapter 4

Design

4.1 Component Selection

The selection of the system's core components, as outlined in the description of the drone design, involves the comparison of complex variables. This is due to the fact that many of the parameters in determining whether the drone will fly are very sensitive; i.e. should the overall weight be slightly over the limit, the whole system will not be able to overcome gravity and hover.

Determining the drone components requires careful evaluation of weight, motor thrust, price and functionality, which can be challenging. This section covers some aspects that need to be taken into consideration for each component, and is followed by simulations of some complete systems using eCalc.

This project considered components that have technical specifications on the listing (such as a datasheet since many online products do not list the specifications), then utilised those values in a simulation to determine the performance of the system, modified the values to achieve a better outcome, and finally attempted to source components which have values near those calculated.

Additionally, the design requirements for this project included a pre-selected optical flow sensor, the Flowdeck V2.

Flight controller

The flight controller connects all of the components and ensures that they can communicate with each other effectively. Choosing the correct flight controller is critical to building a successful flying drone. This project is restricted in its flight controller choice since the brief stipulated the use of the ArduPilot firmware, which precludes that only flight controllers compatible with ArduPilot can be used. Nonetheless, ArduPilot was preferable over other options for its open-source platform with a large support community, and since it is well established and stable. It also has many features including optical flow support which not many other firmware supports, and as it works with a number of hardware platforms. An alternative to ArduPilot would be PX4, which is another autopilot focused firmware with support for optical flow sensors. However, there are not many other well known firmware which support these features.

A significant complication with using ArduPilot was that the compatible flight controllers were either outdated or inaccessible, or they were fully featured but large, heavy and significantly more expensive. A flight controller with ArduPilot's minimum functioning requirements and optical flow connectivity was necessary. This task was difficult since building AutoPilot focused drones is a niche market, and the controllers compatible with optical flow were costly.

ArduPilot requires an accelerometer, gyroscope, barometer as well as a compass, GPS (recommended) or both. This removes many non-autonomous hardware options since non-autoPilot focused controllers do not include a barometer or compass, and GPS modules can be expensive.
The benefit of using an autopilot controller over a typical controller is that the drone can be controlled autonomously. For instance, simply setting the start and end locations will result in the controller and ground control station combination calculating the necessary information to fly the drone, which reduces pilot tasks and consequently pilot fatigue as well; such as keeping the drone steady during flight.
Optical flow sensors could potentially replace the compass for indoor flight, with the

camera acting as a compass, but this functionality is not well documented. Furthermore, the latest versions of ArduPilot which supports optical flow, at a minimum, requires a STM32F4 microprocessor at 168MHZ with 1Mb of flash storage, which means that their older flight controllers which do not meet these requirements cannot support optical flow functionality [53].

Two options were decided on, namely being the more affordable Omnibus F4 derivative [54], the Omnibus F4 SD, which is coupled with an Arduino and telemetry module [55], and the Pixracer R15.
This Arduino and telemetry setup is required because the F4 SD does not have an SPI connection available (only I2C), which is required for the Optical flow sensor provided for the project (FlowDeck V2). It also does not have a compass, so modifications would need to be implemented to use the optical flow sensor for indoor autopilot functionality and an external compass would be needed.
The alternative was the Mro Pixracer R15, which met the requirements of the accelerometer, gyroscope, barometer, compass, SPI and I2C all being present on the board, and of meeting the minimum ArduPilot requirements, however, at a much higher cost.

Frame

After conducting tests between different sizes and combinations of components using eCalc (which is discussed in the Simulation Modelling section), the frame diameter of 100mm was decided upon due to the more affordable motors involved and the lower weight. Regarding the weight, an educated guess of 10g for quads [56] and 15g for hexacopters [57] was made according to the expected weight which was based on other frames on the market. Once all the components were chosen, a frame design was created that attempted to meet the desired strength and weight requirements.

Motors

Motors spin the drone propellers which allow it to fly, however optimising the relationship between thrust, weight and torque is critical to a successful drone. As discussed in the literature review, brushless motors are more robust than brushed motors at the cost of being slightly heavier and more expensive. Yet, this negative aspect is typically outweighed by the benefits. The 100mm frame size decided upon determined that brushed motors would be a potentially viable option depending on the weight of the overall system.

Drone motors are typically categorised by size, including 1103 and 1104 which were the motor sizes applicable to the frame size decided upon for this project. These categories narrow down the motor choices, since the size of a motor delimits the maximum capabilities achievable. Once the appropriate size category was selected, the most suitable motor for the use case was chosen; i.e. more powerful motors for a heavier system, or weaker motors to promote longer battery life in a lighter system. However, simulation in eCalc was necessary before a decision could be made.

It should be noted that when choosing motors, only those which list their full specifications should be considered. The specifications should include the KV rating or rpm/V, the no-load current and the voltage it was tested at, the maximum current or power draw, the winding resistance, the number of magnetic poles, and the physical dimensions of the motor [58]. Without these specifications, the motor cannot be used with eCalc (unless tested by the eCalc team and listed in eCalc itself) and the performance of the motors would be an estimate.

Propellers

Choosing propellers depends on the physical design and weight of the system as well as the chosen motors. The mounting of the propellers must be compatible with the motors, for instance the 1103 motors have a 1.5mm shaft and two M2 mounting holes on either side. Whereas, small brushed motors have a thin 0.8-1mm shaft and

no screw holes, and are consequently only compatible with smaller propellers. The frame design dictates the size of the propellers, since the intersection of propellers would be detrimental as would be the propellers intersecting with the electronics mount.

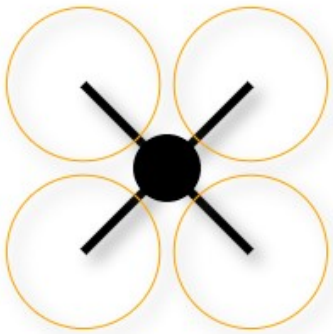


Figure 4.1: Maximising Propeller Size to Frame Size [58]

Consequently, it would optimise efficiency if the maximum propeller size corresponding to the frame size is chosen, as mentioned in the literature review, since propeller efficiency improves with size; as is evident in Figure 4.1. The 100mm frame size chosen, dictated that 65mm propellers be used to maximise the system's efficiency. Selecting the pitch of the propeller blades follows a similar rule, where the steeper pitch results in a higher thrust.
However, the pitch should not exceed 66% of the diameter of the propeller, as this can cause the system to enter a vortex ring state [59] which would cause the drone to experience turbulence or become unstable [60]. By availability, the propeller pitch of 38mm was chosen.

Mosfets/ESC

When considering motor controllers for drones, the two most commonly used motor types are brushed and brushless DC motors. The controllers for these motors are either single mosfet drivers or ESCs respectively. When choosing mosfets for brushed motors, the maximum output current of the mosfet should be higher than the maximum current draw rating of the motors, since running a mosfet at its maximum

rated current for extended periods will lead to failure by overheating. The switching speed of the mosfet should also be fast enough to spin the motor at its max speed. A flyback diode is also needed, to prevent current flowing back through the circuit when the mosfet is in the off state (when the motor acts as a generator) [61].

When choosing ESCs for a brushless motor, the maximum current of the mosfets and the ESC technology should be considered. The most common ESC types used in drones are PWM controlled ESCs which respond to standard analog PWM signals [62]; Oneshot ESCs which use a faster PWM protocol than analog PWM ESCs; multishot ESCs which are similar to Oneshot but which are faster (not many multishot ESCs exist, which means they have limited support); and Dshot (Digital shot) ESCs which use the latest protocol that sends digital packets rather than PWM signals. Dshot ESCs are faster, have a higher resolution, are less susceptible to noise, are safer (microcontroller can detect errors) and do not require calibration [63].

Dshot ESCs are preferable when available since they are usually similarly priced to the alternatives while offering the above benefits. However the protocols supported by the chosen flight controller should be noted before purchasing.

Radio/Telemetry

Radio receivers and telemetry modules are dependant on the transmitter being used and the desired range of flight. Since the transmitters are typically expensive, they should be of the initially selected components in order to fit within the pilot's budget. There are various transmitter options available which offer different build quality, features, radio technology and potentially a screen for telemetry data.

The receiver chosen needs to use the same protocol as the transmitter so that they can communicate. The most common frequency is 2.5Ghz, while lower frequencies offer greater range. The receiver's number of supported channels should correspond with the number of desired inputs and outputs on the transmitter [64].

The telemetry module is typically an independent device operating over a different protocol with its transceiver connected to either a laptop, tablet or VR device, for viewing the telemetry data, such as the battery voltage or errors and signal strength during flight. However should the transmitter and receiver support telemetry, this would eliminate the need for an independent telemetry module and laptop or tablet.

The transmitter available for this project was a FrSky Taranis Q X7S, which dictated that a SBUS FrSky receiver should be used. The FrSky XM+ was chosen since it was the lightest and most affordable compatible option. The telemetry module selected was the ESP-01 WiFi module, since it was lightweight and since the project was mainly designed for indoor flight where the range of WiFi technology was not a hindrance.

Power management

The battery is the largest and heaviest component on the drone and thus has a large effect on the performance of the system. When simulating the battery, its weight should be overestimated to ensure the drone can still take flight should the real-world desired battery weight be unobtainable.

The different chemical structures of batteries should also be considered. Typically LiPo batteries are used for RC (Radio Control) applications such as drones and cars because of their high energy density to weight ratio compared to NiCAD, NiMH or Li-ion batteries [65] [66] which leaves the LiPo batteries comparatively lighter for the same capacity and with higher discharge rates.

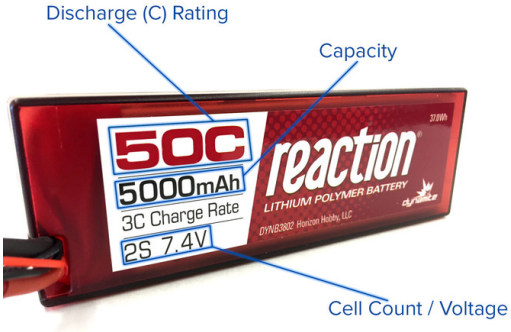


Figure 4.2: Battery Diagram [67]

Performance of LiPo batteries is denoted by the discharge rating and capacity. The capacity is rated in mAh (milliampere/hour) which is the amount of current that can be drawn from the battery to discharge within one hour. The higher the capacity of the battery the longer it is able to supply power, albeit at a higher weight.

The discharge rating is measured in Amps, and refers to the maximum current that can be drawn from the battery without it damaging itself. The Amperage value is typically listed in terms of C (capacity); i.e. a 75C 460mAh battery can be discharged at 75 x 460mAh = 34.5A. LiPo batteries have a standard nominal voltage per cell of 3.7V, which means a 2S battery will have 7.4V nominal, this should be chosen in accordance with the selected motor's desired voltage [67].

A possible design approach would be to calculate the performance of the system using a theoretical battery that meets the current draw requirements of the motors (and voltage), the desired flight time, and which is of a heavier weight than a real-world battery of that specification. Thereafter, a real-world battery which closely resembles the requirements should be found.

The electronics, other than the ESC and motors, also need to be powered by the battery. However, they will typically require between 3.3 and 5V, which is lower than the battery voltage in most cases. A BEC (Battery Eliminator Circuit) is required to reduce the battery voltage to the necessary value. Some BEC units will include a voltage and current sensor to note the battery's charge status during the flight and to ensure the motors are not drawing excessive current for the battery rating.

4.1.1 Simulation modelling

The eCalc [58] software was used throughout, as it allows users to simulate the performance of a drone configuration without having to buy, create and test the components. It spares time and effort that would otherwise be lost to numerous costly, failed attempts.

A typical eCalc configuration can be seen in Figure 4.3 below (hexacopter design), where the frame size, weight, number of motors, battery size and specifications, ESC weight and specifications, motor weight and specifications, and propeller diameter, pitch and weight is entered into the simulator:

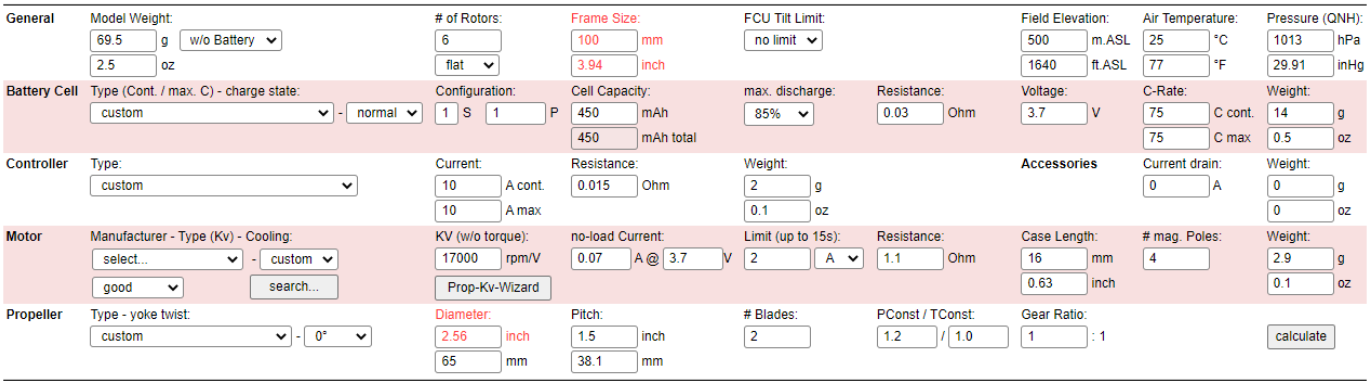


Figure 4.3: eCalc Configuration

The performance output from the above inputs can be seen in Figure 4.4, with the battery load, hover flight time, power draw, thrust to weight ratio, and specific thrust performance metrics being included amongst numerous others:

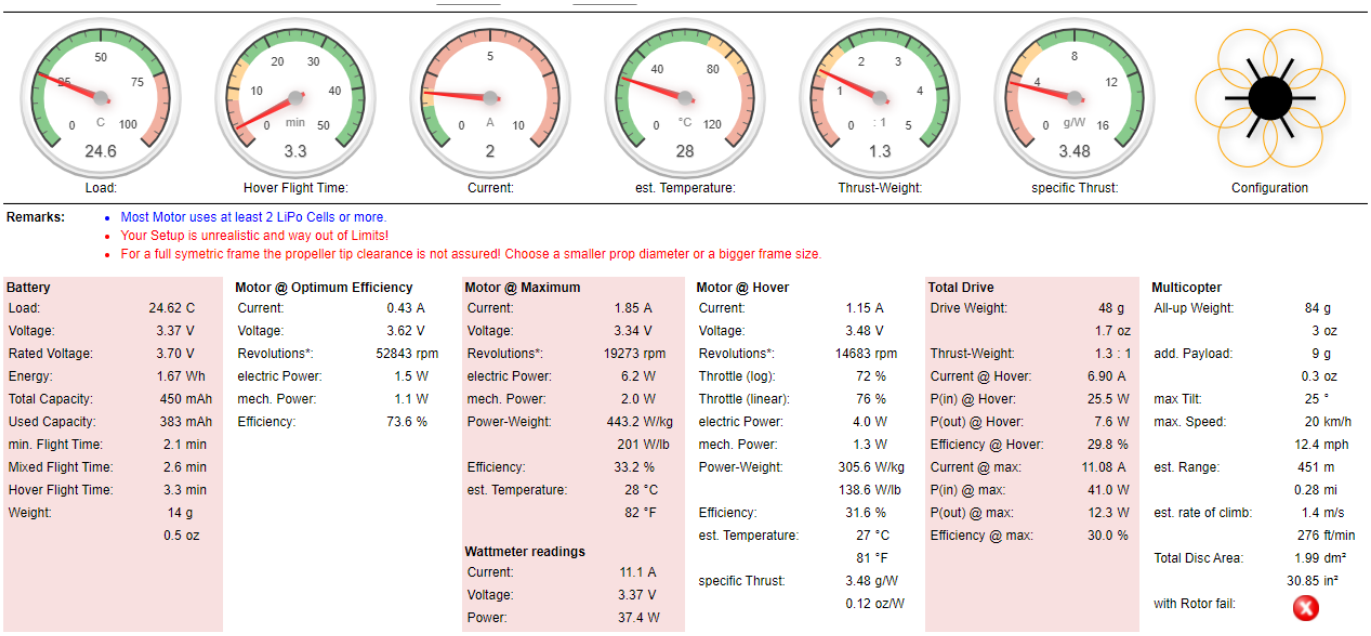


Figure 4.4: eCalc Output

Component Selection Phase 1

Many systems were tested using eCalc until a final choice was decided upon. All of the simulated brushed motor systems mentioned below used the same 1s 450maH 75c (3.7V) LiPo battery with an estimated weight of 14g (1s 450mAh 75C), although the systems with brushless motors used a 2s configuration and needed a C rating of 40+ C for approximately 460mAh (460mAh 2S 75C), but could work with a higher capacity battery such as a 25C for 1000+ mAh (2s 1000mAh 25C). Alternatively, two of the 1s batteries could be placed in series.
They all used the same receiver for the flight controller, the FrSky XM+ (FrSky XM). All of the systems use the same 65mm diameter, 1.5 inch pitch, polycarbonate propellers (65mm 1.5in Prop), even the hexacopter configurations where two propellers would be raised above the rest.
Note that in the comparison tables the price and weight calculations are approximate as some products available for purchase do not explicitly clarify weight. The prices of the products in the comparison tables do not include shipping costs, approximate values have been included for international customs and duties.
All the necessary simulation tables for each system are placed in Appendix A.

The systems are summarised and compared in the following table:

System	Price	Weight w/o bat	Flight time	Throttle	Recommend?
Sys 1 - Cheaper hex, brushed	1444	69,5	3.3min	76%	No
Sys 2 - Cheaper quad, brushed	1389	57	0	0	No
Sys 3 - Cheaper quad, brushless	2440	95	6.1min	45%	Yes
Sys 4 - Expensive hex, brushed	3599	61,5	4min	70%	Maybe
Sys 5 - Expensive quad, brushed	3544	49	4min	84%	No
Sys 6 - Expensive quad, brushless	4595	87	6.7min	43%	Yes

Table 4.1: Component system table 1

System 1, Cheaper drone with Hexacopter configuration (6 motors):

This simulated drone used smaller brushed DC motors, which were made available for this project (0716 Motor), a 6 Mosfet motor driver board (Mosfet board), and was based around the estimated frame weight of 15g (Frame). The flight controller used did not feature a built in Compass, which ArduPilot necessitates, nor did it have SPI and I2C pins available for the Flow Deck (Hobbywing Xrotor F4 G3 (Local Omnibus clone)). To utilise the flow deck, an additional Arduino was necessary for its I2C, SPI and UART connections, as mentioned previously in the Flight Controller section. The system would use an ESP-01 to communicate its telemetry data with the Ground control software (ESP). The flight controller also needed an external step-up regulator for when powered by its 1s battery (Regulator). An adaptor would also need to be printed, converting the 0.8mm shaft of the motor to the 1.5mm mount on the propellers.

System 2, Cheaper drone with Quadcopter configuration (4 motors):

This simulated drone was exactly the same as system 1 except that it used 4 motors instead of 6, hence there were only 4 propellers and the frame was slightly lighter.
Note that this system was unable to hover when simulated, exaggerating the fragility of the required balance between thrust and weight when designing a drone.

System 3, Cheaper drone with Quadcopter configuration (4 brushless motors):

This simulated drone was similar to system 2, but the brushed motors were replaced with brushless motors (T-motor F10), which required the Mosfet board to be replaced with four, 20 amp ESCs. It should be noted, that the weight of this system could have been reduced by roughly 18g by using the (DYS ELF ESC) 4 in 1, 10A ESC, but which would have cost more.

System 4, More expensive drone with Hexacopter configuration (6 motors):

This simulated drone had the same configuration as system 1, but the flight controller and Arduino setup were replaced with a single flight controller (Mro Pixracer R15), which had a built in compass, and which included a telemetry transceiver, and had SPI and I2C pins to connect the Flow Deck to.

System 5, More expensive drone with Quadcopter configuration (4 motors):

This simulated drone was exactly the same as system 4, except that it used 4 motors instead of 6, hence there were only 4 propellers and the frame was slightly lighter.

System 6, More expensive drone with Quadcopter configuration (4 brushless motors):

This simulated drone was similar to system 5, but the brushed motors were replaced with the same brushless motors as in system 3, which required the Mosfet board be replaced with 4x20 amp ESCs. The weight of this system could also be reduced by roughly 18g by using the same 4 in 1 ESC as mentioned in system 3.

Another Option could have been to use a RadioLink MiniPix flight controller (Link), which would cost approximately 1500 rand in place of the Mro Pixracer R15 used in Systems 4, 5 and 6 which costs approximately 2700 rand. Their respective weights were similar but since the MiniPix did not have an SPI pinout, but had extra UARTs, an SPI to UART converter would have been needed (Link), which would have cost 200 rand and would weigh between 2-5g. However, it was questionable whether this method would work with the SPI converter.

From the 6 available options, both of the quadcopters that used smaller motors (brushed)(systems 2 and 5) were not recommendable, as they simply were unable to

bear the weight of the full system. The Hexacopter with the cheaper flight controller combination (system 1) was also not recommendable, as the hover throttle was too high to be able to maintain control of the drone. The Hexacopter with the more expensive flight controller (system 4) could have possibly worked as it was lighter, but the hover throttle was still relatively high.

Either of the brushless options were recommendable, as they outperformed the other systems to an acceptable degree. It was then still necessary to determine which flight controller to use in the design. It was anticipate that the more affordable option would require more work to implement, yet there was also a lack of confidence in whether it would be possible to have the system functioning within the time confines of this project.

The MRo flight controller was more expensive than the alternative entry-level option as it had more features. This signified that there was no intermediate controller available on the market, which would have had to have only the minimum features required for this specific project. Only entry-level and advanced flight controllers were available. A solution to this problem would have been to design a custom flight controller, which would have helped reduce the cost of the overall system. However, designing such a controller would have been a more involved process and would have taken more time than was made available by the constraints of the project, which determined the choice.

Component Selection Phase 2

Following the initial selection phase, Further calculations with numerous additional system configurations and components were conducted, including systems with gear reductions in an attempt to get the smaller, more affordable brushed motors working with the desired system payload. Ultimately, three systems were selected to be compared toward a decision. The systems with gear reductions were deemed unfeasible, due to the higher risk of destruction upon crashing, as well as the difficulty

in sourcing the exact size of small plastic gears (without having them made, which would have been costly).

Two of the systems that were most feasible required the purchase of new motors, instead of being freely supplied for this project. Additionally, the fact that the MRo flight controller was deemed the most likely to succeed in the time limit and given the specific project requirements (i.e, ArduPilot compatible and have support for I2c and SPI for optical flow sensor), meant that the budget for this project was extended. As mentioned previously, a solution would have been to create a custom PCB which would have included all of the desired features for this specific project, where the PCB acts as the frame in a similar way to the Bitcraze Crazyflie.

The final three systems that were chosen, were a cheaper brushless quadopter (system A), a more expensive brushless quadcopter (system B) and an expensive brushed Hexacopter option (system C).
Using the brushed motors with the inexpensive Omnibus flight controller and separate Arduino for the Optical flow, was deemed unfeasible because the system setup weighed too much for the capacity of the motors, even when in a Hexacopter configuration. The hover throttle of the systems were too high to have any control, as they would only be able to hover but be uncontrollable when moving around.

System A used the inexpensive Omnibus F4 flight controller and Arduino setup mentioned previously, along with the brushless T-motor F10 motors and the 4 in 1 DYS ELF 10A ESC board. System B used the expensive Mro Pixracer R15 flight controller and the same brushless setup as System A. System C used the Mro flight controller as well, but used 8520 brushed motors (link) and the 6 in 1 Mosfet board.

These systems are compared further in Table 4.2 and 4.3 below.

System	Price	Weight excl battery	Flight time	Throttle	Complexity	Recommend?
A	2710	77	7.9min	40%	40/50	Yes
B	4870	69	8.2min	39%	31/50	Yes
C	4081	73,4	4.5min	61%	35/50	No

Table 4.2: Final Systems performance comparison

	System A			System B			System C		
Components	Weight	Price	Link	Weight	Price	Link	Weight	Price	Link
Motors	28	840	Brushless Motor	28	840	Brushless Motor	29,4	500	Brushed Motor
Frame	10	105	Rod	10	105	Rod	15	105	Rod
FC	8	500	Xrotor FC/Omnibus	11	2700	Mro FC	11	2700	Mro FC
Receiver	3	300	Rc	3	300	Rc	3	300	Rc
Flow Deck	2		Flow	2		Flow	2		Flow
Battery		300	2s Bat		300	2s Bat		155	1s Bat
Props	4	55	Prop 65/1.4	4	55	Prop 65/1.4	6	110	Prop 65/1
ESC\Mosfet Board	6	570	ESC	6	570	ESC	2	180	Mos
Arduino + Telem	11	40	Telem						
Wires + Misc	5			5			5	31	5V
Total	77	2710		69	4870		73,4	4081	
Including Battery	101			99			85,4		

Table 4.3: Final component choice, price and weight comparison

Initially it was decided that system C, using the brushed motors, was not as robust as the brushless motor systems, and that being a fraction over the weight stipulated in Table 4.3 meant it would not take off. Whereas the brushless systems were able to bear much heavier payloads, allowing for more flexibility in terms of the frame design. An additional reason for deciding against the brushed motors was that the propeller shaft diameter was 1mm, which meant finding propellers in the required coinciding size of 65mm was difficult; they were only available abroad. The mosfet board would also have to have been imported from abroad. The long lead times for the most high risk components, and the lack of robustness, lead to the removal of system C as an option.

When deciding between Systems A and B, the price of system A was more reasonable, but the risk of the final product not working sufficiently or meeting the goals set out in the beginning of this report was higher (due to the possible incompatibility with ArduPilot and the workaround required for the lack of I2C and SPI for Optical flow sensor and rangefinder to work). Consequently, system B was the option selected despite its price, because it was the option that would most likely succeed within the given time limit and the requirements of needing to use ArduPilot and the Flow Deck.

However, unforeseen issues arose when components were not purchased timeously (components needed for system B were sold out) such that new components had to be chosen. The chosen motor needed to be changed to a unit which used the same form factor and had similar performance (unfortunately the exact performance could not be matched) [68].

It was also assumed that a simple and lightweight voltage regulator would be used to power the flight controller and its electronics (ESC and motors would be powered directly by the battery), however this would not feature battery voltage and current monitoring which were necessary to ensure the drone would not crash should the battery die. However, the only component which was in stock had a weight of 15g (later reduced to 11g by desoldering connectors) which reiterates the hindering of the brushed motors in system C. Fortunately, the selected brushless motors could bear the added weight effectively.

4.2 Physical Design

The design of the system frame involved measuring the physical sensors and components, by using calipers to obtain their major dimensions. These dimensions were then transferred into the CAD software, Fusion 360, to be used as to-scale templates when designing the frame and other mounts (all CAD files included in GitHub).

Each piece of material on the frame was incrementally added in consideration

of the light weight requirement of the frame (10g). The weight of the frame was simulated using the built in material system in Fusion 360. An initial design was then created using carbon fibre tubes and a 3D printed structure, as seen in Figure 4.5. However, this design proved inefficient when considering the small size of the components, since assembly of this type of frame would be challenging.

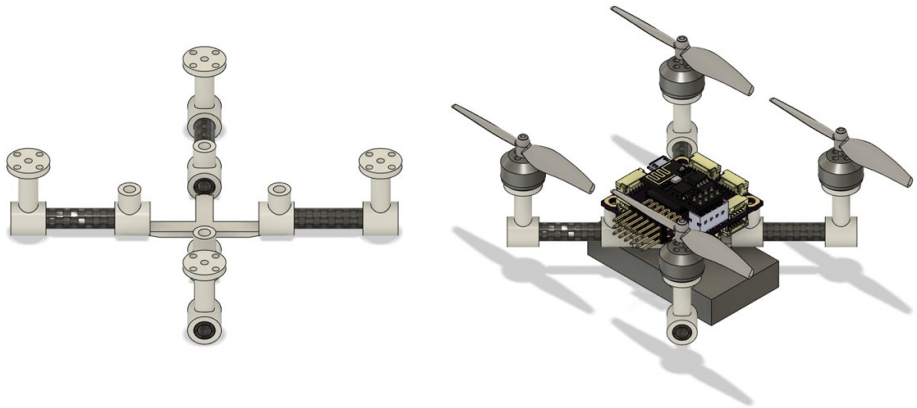


Figure 4.5: Frame design 1 - with carbon tubes

Accordingly, it was decided that the frame be 3D printed, due to the availability of printers for this project and since 3D printing allows for rapid iterations and testing of the design. A prototype was designed with the flight controller located above the main structural plate, the motors positioned higher than the flight controller, and with the rest of the electronics placed below the flight controller, as seen in Figure 4.6. However, this design proved insufficient due to the plastic filaments not being sufficiently rigid, resulting in the pillars, which the motors were mounted onto, causing severe oscillations during flight.

The designs which follow all include triangulation of the top structural plate, to rigidify the frame and prevent it from being warped by the thrust of the motors [69]. Had the frame not been reinforced in this way, the drone would have experienced oscillations that would have impeded its flight. Overbearing oscillations have the potential to create interference between the propellers (propellers hitting each other).

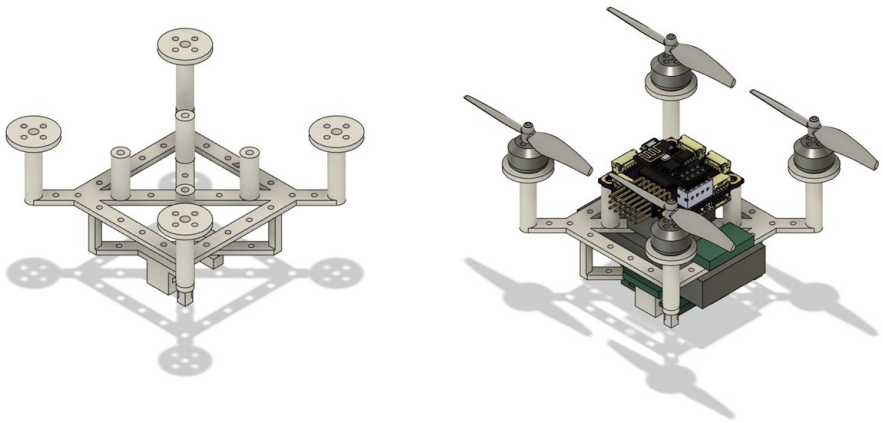


Figure 4.6: Frame design 2 - 3D print with FC top side

The design seen in Figure 4.7 positioned the electronics beneath the top structural plate, and attached the motors directly onto the top plate for a stronger design. A thin I-beam was included, connecting the motor mount to the flight controller pillar mount. This I-beam removed the flexibility on the motor mounting surface and improved the vertical strength of the frame.

The battery and sensors were then positioned beneath the flight controller using thin rectangular posts and a mounting plate. However, the weight of this frame was 11g (1g over desired weight), which prescribed that no landing gear be added and that the voltage regulator and battery sensor module would be resting on the ground when stationary.

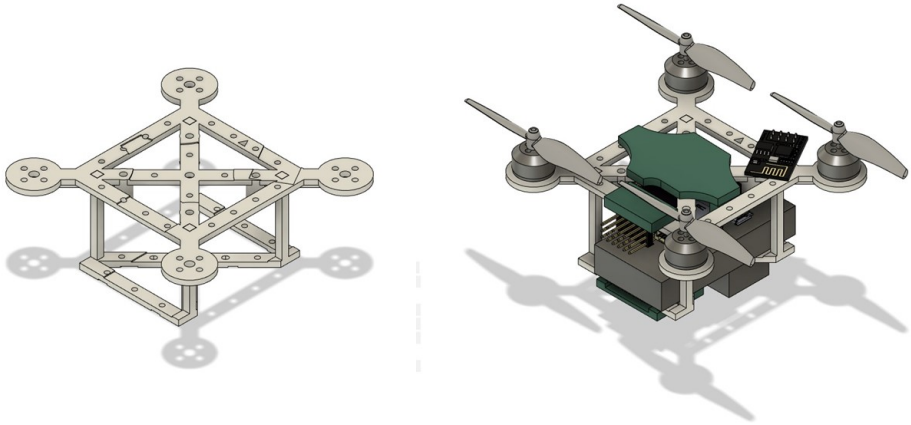


Figure 4.7: Frame design 3 - 3D print with FC bottom side

The design in Figure 4.7 was used extensively for testing because it was anticipated to be the final design. However, the drone was crashed numerous times (flying in a confined space), repeatedly snapping the thin posts holding the battery and sensors which then had to be repaired or reprinted. This prompted a new design, which featured thicker and stronger tubular posts, and which is seen in Figure 4.8. An unfavourable repercussion was that the new posts added 4g to the weight of the frame, making it 15g (5g over the desired weight).

A base was designed for the drone to rest on when idle or before flight, also seen in Figure 4.8. This was implemented due to the weight limit of the frame dictating a lack of landing gear. Regardless, the drone was still piloted to land on a cushioned surface so as to reduce strain. Also, manoeuvring the drone proved to be challenging.

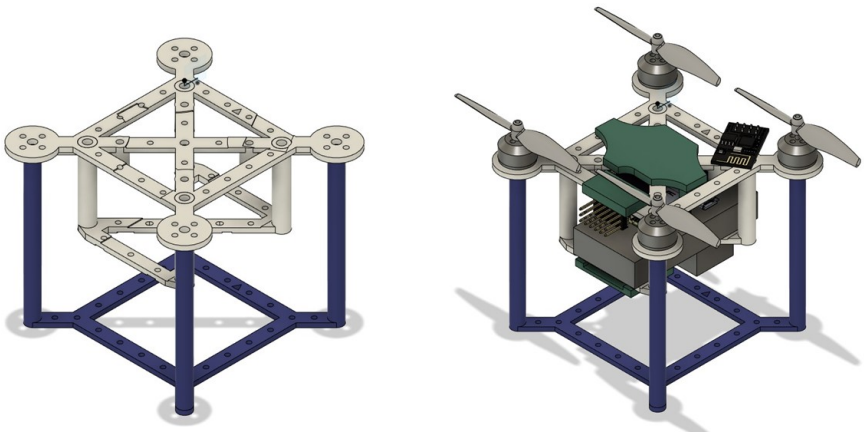


Figure 4.8: Frame design 4 - 3D print with FC bottom side and stand

It is worth noting that the final frame design used could be improved on, since the progress of the project and its potential were restricted by the time constraints. Specifically, the posts used to hold the battery and sensors could have been substituted with stronger carbon fibre tubes bonded to the top and bottom frame plates. These carbon rods could also be used in the design of the landing posts. The frame's structural plate could also be substituted with carbon fibre sheeting, which is especially strong, allowing for more material to be removed to meet the frame's weight requirement.

4.3 Software Design

Setting up the WiFi telemetry required the MAVESP8266 firmware to be flashed onto the ESP-01 device, using an FTDI programmer connected to the ESP unit as in Figure 4.9.

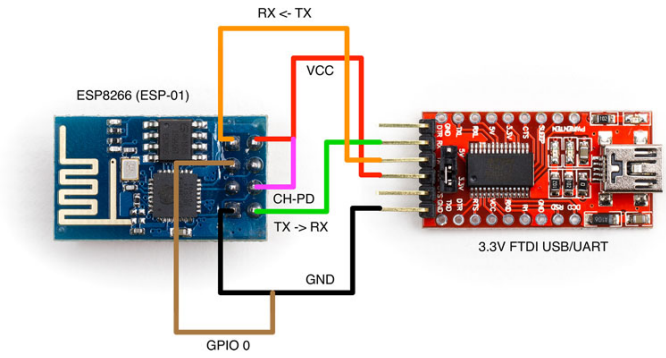


Figure 4.9: ESP-01 FTDI connection [70]

The NodeMCU ESP8266 flasher [71] was used to upload the MAVESP8266 firmware onto the device [72].
Thereafter, while the flight controller was connected via USB, the Serial1 (telemetry 1 port) parameters in Mission Planner were set to use MAVLink2 with a baud rate of 921600.
Once reconnected to the drone, the ESP-01 emitted a WiFi hotspot that the laptop was connected to and a connection was established with Mission Planner. This telemetry connection allowed for the flight controller to be configured in the same capacity as when connected via USB, and to display the sensor data and status of the drone.

The firmware of the flight controller was modified to allow communication with the optical flow sensor. The modifications included re-defining the pin-out of the

physical hardware ports on the flight controller.

The datasheet of the flight controller's microcontroller was examined (STM32F427VIT6) [73] along with the schematic of the Pixracer flight controller (available on GitHub). This was conducted in order to obtain the necessary pins which had the potential to be remapped and used for SPI communication (the protocol which the optical flow sensor communicates with). This is possible because certain pins on a microcontroller can be configured to use different protocols based on the desired application.

The SPI compatible pins obtained were compared to the pins in the FC's schematic. It was found that the pins on the ESP-01 module's port (which uses the UART protocol) could be re-assigned as SPI pins. The compatible pins were PB4, which corresponded to the CS pin for SPI 4, PE2 to SCK, PE5 to MISO and PE6 to MOSI [74].

The full development environment for the ArduPilot stack was installed in accordance with the instructions on their website [75], to modify the firmware. The ArduPilot source code was downloaded from their Github repository [76]. The latest stable version of Arducopter was selected as the firmware. This was, at the time of writing, Copter 4.1 which was selected for editing using the command "git checkout Copter-4.1.0".

The files were downloaded and setup in the Linux environment for windows. The "hwdef.dat" file (hardware definition file), for the Pixracer flight controller, was navigated to at "libraries/AP_HAL_ChibiOS/hwdef/Pixracer" and edited.

The pin definitions for the ESP-01 were commented out and the new SPI definitions were added, as seen in Figure 4.10. Subsequently, the SPIDEV settings for the pixartflow (the brand of optical flow sensor on the Flowdeck) were added to the file. The SPIDEV settings included the frequency being set to 2Mhz, the mode being set to 3 (relative to ArduPilot mode definitions), as well as setting the SPI CS pin and the name of the sensor, as seen in Figure 4.11a. A definition was also added at the end of the file, indicating to the flight controller that a pixartflow sensor is attached,

evident in Figure 4.11b [77]. The edited files are available on Github

```
# USART1 is disabled (was ESP8266)
PB6 USART1_TX USART1
PB7 USART1_RX USART1
PA8 USART1_RTS USART1
# PE10 is not a hw CTS pin for USART1
PE10 8266_CTS INPUT

# old (make GPIOs for ESP8266 available via mavlink relay control as pins)
# 60 to 63
# comment out these four lines to enable SPI
#PB4 8266_GPIO2 OUTPUT GPIO(60)
#PE2 8266_GPI0 INPUT PULLUP GPIO(61)
#PE5 8266_PD OUTPUT HIGH GPIO(62)
#PE6 8266_RST OUTPUT HIGH GPIO(63)

# Setup the SPI4 GPIO's for flowdeck v2 (just the PMW3901 optical flow sensor)
# Uncomment these 4 lines and the lines further below to enable pixartflow
PB4 FLOW_CS CS
PE2 SPI4_SCK SPI4
PE5 SPI4_MISO SPI4
PE6 SPI4_MOSI SPI4

# I2C for flowdeck v2 (just the VL53L1x range finder)
PB8 I2C1_SCL I2C1
PB9 I2C1_SDA I2C1

# SPI2 is FRAM
PB10 SPI2_SCK SPI2
PB12 CAN2_RX CAN2
PB13 CAN2_TX CAN2 # this is SPI2_SCK on beta board
```

Figure 4.10: Edited hwdef.dat file, re-configuring the pin definitions of the Pixracer

```
# SPI device table. The DEVID values are chosen to match the PX4 port
# of ArduPilot so users don't need to re-do their accel and compass calibrations
# when moving to ChibiOS
SPIDEV ms5611_int  SPI2 DEVID3  BARO_CS    MODE3  20*MHZ  20*MHZ
SPIDEV mpu9250     SPI1 DEVID4  MPU9250_CS MODE3   2*MHZ   4*MHZ
SPIDEV icm20608    SPI1 DEVID6  20608_CS   MODE3   2*MHZ   8*MHZ
SPIDEV hmc5843     SPI1 DEVID5  MAG_CS     MODE3  11*MHZ  11*MHZ
SPIDEV lis3mdl     SPI1 DEVID5  MAG_CS     MODE3 500*KHZ 500*KHZ
SPIDEV ramtron     SPI2 DEVID10 FRAM_CS    MODE3   8*MHZ   8*MHZ
# Enable pixartflow
SPIDEV pixartflow  SPI4 DEVID2  FLOW_CS    MODE3   2*MHZ   2*MHZ
```

(a) SPIDEV configuration

```
# 6 PWM available by default
define BOARD_PWM_COUNT_DEFAULT 6

# Define pixartflow
define HAL_HAVE_PIXARTFLOW_SPI

# two IMUs
IMU Invensense SPI:icm20608 ROTATION_ROLL_180_YAW_90
IMU Invensense SPI:mpu9250 ROTATION_ROLL_180_YAW_90
define HAL_DEFAULT_INS_FAST_SAMPLE 1
```

(b) PixartFlow Definition

Figure 4.11: Edited hwdef.dat file, re-configuring the SPIDEV and definitions of the Pixracer

The firmware was then configured and built using Linux Waf, and the Pixracer was booted into "flash mode" using Mission Planner to upload and install the compiled .apj file.

4.4 Wiring

The individual component datasheets/pinout and the flight controllers pinout diagram [78] were compared throughout the wiring process. The overall system diagram, seen in Figure 4.12, describes the system's flow of operation with the red arrows and wireless symbols indicating connections between components.

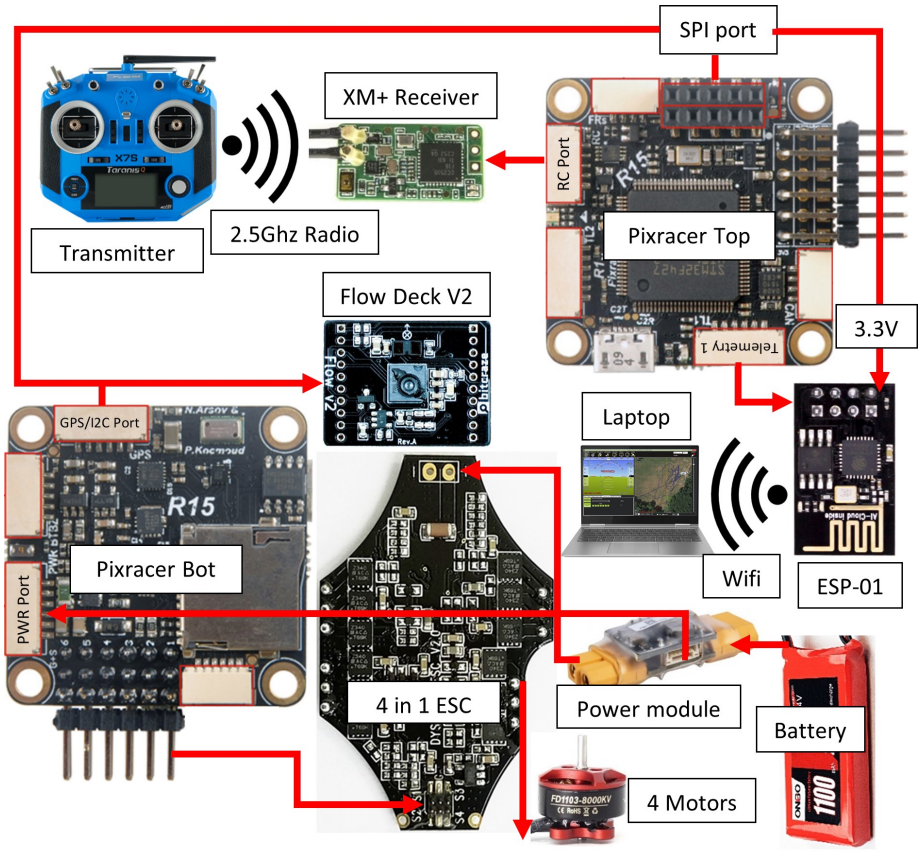


Figure 4.12: Full System flow diagram [79] [80] [68] [81] [82] [83] [84] [85] [86] [87]

As mentioned in the Software Design section, the Pixracer required pins to be reconfigured for its SPI connection. The new, reconfigured connector pinout (originally ESP-01 port), is shown in Figure 4.13, where the 3.3V pin provides power to multiple modules, namely the ESP-01, as well as the VL53L1X and PMW3901 sensors on the flow deck.

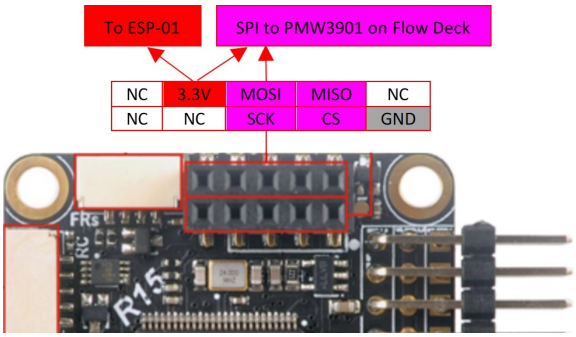


Figure 4.13: SPI Pinout for Flow Deck (Pixracer side) [87]

Three wires were connected directly between the I2C connection of the VL53L1X rangefinder on the Flow Deck module, and the I2C port on the Pixracer, evident in Figure 4.14. The 3.3V source for this module was obtained from the SPI connector as mentioned before.

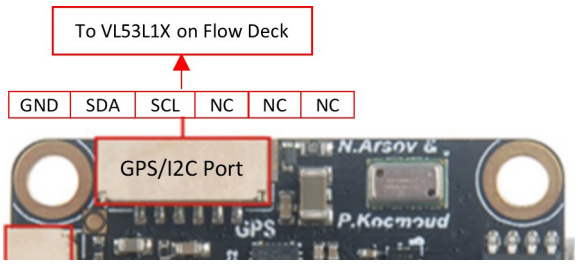


Figure 4.14: I2C Pinout for Flow Deck (Pixracer side) [87]

When determining the pinout for the flow deck, it was necessary to acknowledge it uses a standard connector pinout for the Bitcraze Crazyflie, which has many additional unused pins. The schematic [88] was compared to the Bitcraze Flow Breakout (the sensor version which was meant for external development and not intended to be attached to the Bitcraze Crazyflie 2) [89], to determine the minimum pins necessary. The ports on the Pixracer were then compared, noting the connections required, as seen in Figure 4.15.

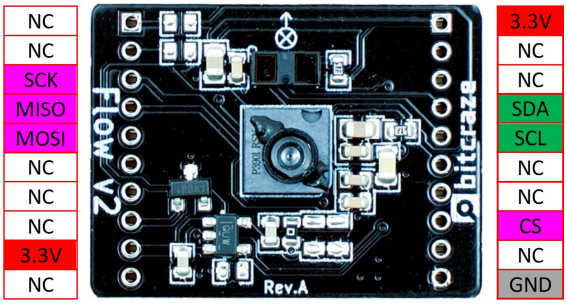


Figure 4.15: Flow Deck V2 pinout [84]

The Telemetry 1 port on the Pixracer was used to communicate with the ESP-01 module over a UART connection, as in Figure 4.16.

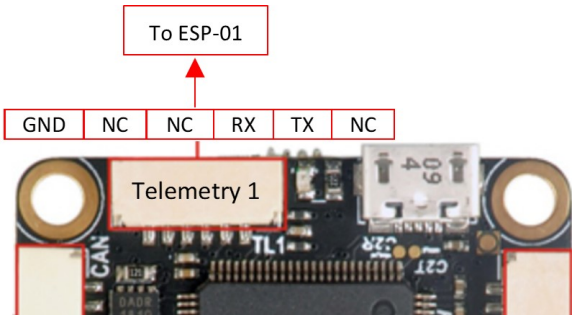


Figure 4.16: UART Pinout for ESP-01 (Pixracer side) [87]

The Telemetry 1 port, was connected to the ESP module in accordance with the pinout seen in Figure 4.17. The 3.3V source for the module was connected to the SPI connector on the Pixracer as shown in Figure 4.13. This was required because the ESP module was not 5V tolerant. The Enable pin on the module was connected to the 3.3V pin (set as high), ensuring that the module turned on when connected to power.

Further development of this project, would have allowed for the remaining pins of the ESP-01 to be implemented; necessitating further editing of the ArduPilot hardware definition file. Enabling the reset pin would allow the ESP module to restart when the flight controller is soft restarted (firmware is restarted but power is not disconnected). At present, the module stops working after a soft restart and requires that power be cycled for it to work again. The GPIO pins could also be used to connect LEDs, which would indicate of data transfer or errors.

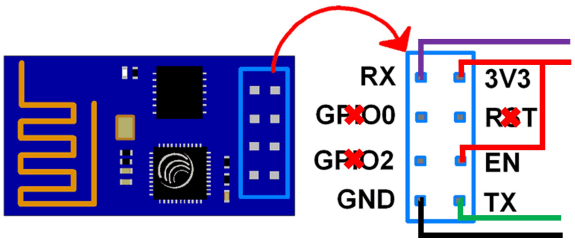


Figure 4.17: ESP-01 pinout [90]

The RC port on the Pixracer was used to communicate with the XM+ receiver, with the pinout described in Figure 4.18. The FrSky telemetry port alongside it could not be used for the same functionality since it is only compatible with specific FrSky telemetry transceivers.

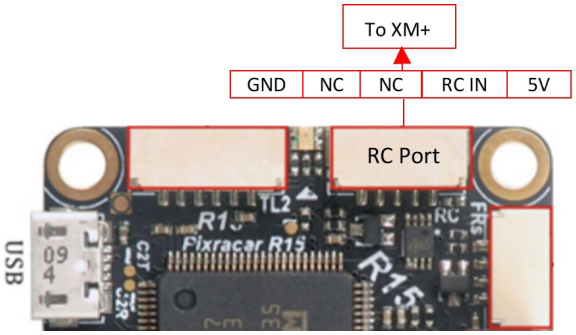


Figure 4.18: RC Pinout for XM+ Receiver (Pixracer side) [87]

The RC port on the aforementioned Pixracer was connected to the XM+ receiver module, according to the pinout in Figure 4.19.

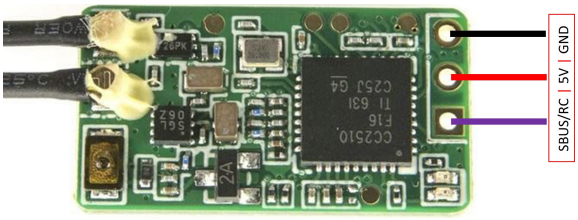


Figure 4.19: XM+ Receiver pinout [82]

The servo header on the Pixracer was used as the signal output for the ESC. Figure 4.20 illustrates that the top row of pins was used for the servo signals, the middle row for Vdd which were in fact not needed for the ESC (they connect directly

to servo motors), and the bottom row was used as ground. Only a single ground wire was needed for the chosen ESC, as it was a 4 in 1 module (not four individual ESCs).

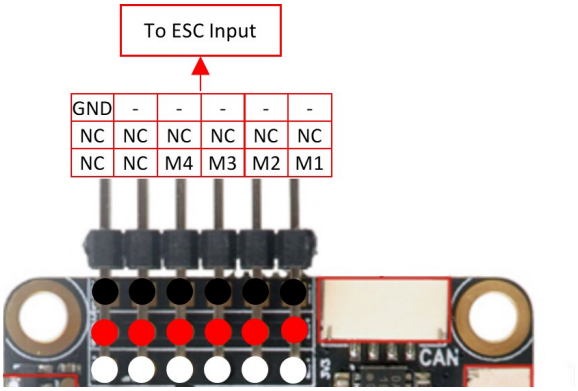


Figure 4.20: Servo pinout to ESC input (Pixracer side) [87]

The ESC was connected to each of the signal wires as mentioned above, which it used as inputs to control the motors. Each of the motors had specifically numbered soldering pads to connect to on the ESC, as seen in Figure 4.21. The ESC was powered via a connection with the Power Brick Mini's high voltage output (the battery connection passed through a shunt resistor, allowing the current to be measured).

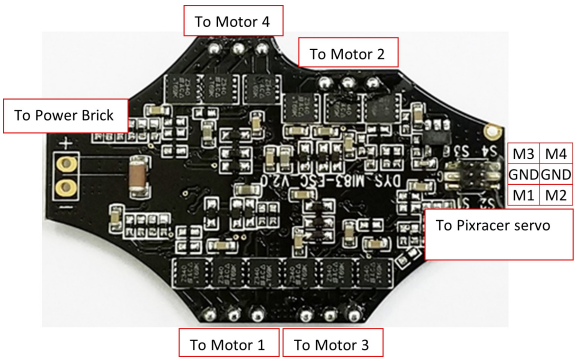


Figure 4.21: ESC Inputs [81]

The power cable required no modification and plugged directly into the Power Brick Mini. This was because the Pixracer followed the same pinouts as the Pixhawk 2.1 (The Cube), which the Power Brick Mini was especially designed for.

4.5 Implementation and Setup

4.5.1 Build Process

The frame components were 3D printed and the system was assembled. The assembly included the creation of various custom wires (various lengths and terminations), such as a Dupont 8 pin header (from the SPI port) soldered directly to the ESP-01 and flow deck. Various JST-GH connection cables were also created. However, since these connectors were not readily available locally, a Pixhawk cable pack (Pixracer used the same standard connectors) was purchased and the cables cut and modified to connect to the various components.

The motors were screwed into the frame with the provided M2 screws, and the propellers were attached onto the shafts of the motors and screwed in with the longer M2 screws that were included. The Pixracer was bolted to the frame as well. The heavy XT60 connectors included on the power module and battery were desoldered and replaced with lighter XT30 connectors. The motors were soldered to the ESC, as were the signal wires. The polar ends of the signal wires were then soldered to a Dupont header. The wires connecting the flow deck were also soldered.

Initially the design used thick double-sided tape to attach the ESC, Flow deck, XM+, and power module to the frame. This was replaced with thin double-sided tape, but clear glue was ultimately used since it held the components well and could be neatly removed when frames were changed.
The iteration of two different frames and the final design are shown in Figures 4.22 and 4.23.

Figure 4.22: Iteration of frames and final system

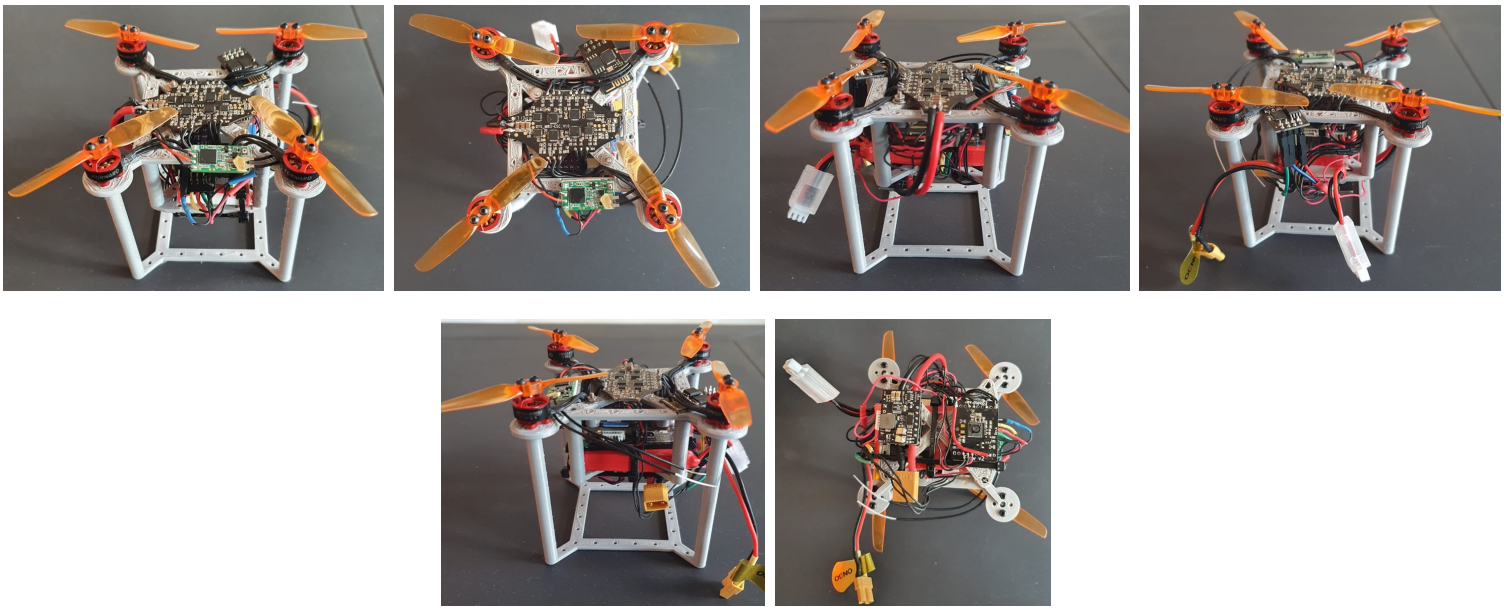


Figure 4.23: Final frame design

It should be noted that knowledge of the efforts necessary for achieving SPI functionality on the Pixracer, by modifying the firmware's hardware definitions, clarified that it was possible to get the cheaper Omnibus F4 Pro flight controller to work with the optical flow sensor and rangefinder on the flow deck V2. This would have required the Omnibus' on-board OSD (on-screen display) be desoldered, and the SPI connection from the PMW3901 on the flow deck be soldered to those pins. The rangefinder would then need to be connected to the GPS I2C pins on the Omnibus.

However, this was not well documented when selecting the components and would have been a risk to the project's success given the time limit, hence why the Pixracer was chosen. The Omnibus does not include a compass, necessary for autonomous flight, which meant that a UART compatible external compass would have been needed (or an I2C extension board).

The firmware was configured in order for the components on the drone to work together. The Pixracer was connected either via WiFi (ESP-01/8266 telemetry module) or USB, allowing Mission Planner to edit the firmware parameters of the drone.

All of the parameters changed and used are available in the GitHub repository, formatted as a file which can be directly uploaded to another Pixracer or ArduPilot flight controller.

4.5.2 Setup Procedure:

The initial setup procedure was completed in Mission Planner, according to ArduPilot's website [91]. The frame type was set as a quad X design, since it appropriately informed the firmware of the correct amount of throttle to send to each motor to keep the drone level and hovering. Other configurations, such as "V", would have motors positioned with different spacing between each other, which would change the kinematic model of the drone and the resulting thrust required from each motor.

The accelerometer, gyroscope and compass were calibrated using Mission Planners built in wizards. The drone was required to be placed in various orientations, such as nose down and on its left side. This allowed the flight controller to determine the correct range of sensor values required for the individual sensors. The compass wizard followed a similar procedure, but needed more rotations to calibrate.

The transmitter and receiver were setup by first creating a new profile on the

transmitter, so that it could be used for multiple RC systems. The profile was setup to be used in conjunction with a quadcopter which was achieved by first assigning the joysticks to particular functions; i.e. the left joystick moving up and down controlled the throttle, while the left and right motion of the right joystick controlled the roll axis of the drone. The switches were subsequently assigned to the desired inputs; i.e. the 2 way switch was used for arming and disarming, while the 3 way switches were assigned to the different flight modes [92].

The inputs created were assigned names and "mixed" to a specific radio channel. By default ArduPilot requires that roll, pitch, throttle and yaw be assigned to channels 1, 2, 3 and 4 respectively [93], and that the autotune mode be set as channel 7 or 8 (typically a temporary assignment), but the channels following these were assigned randomly.

The transmitter was then bound to the receiver. For the XM+ used, this required the bind button to be held before power was applied, in order to boot the receiver into bind mode, which allowed the transmitter to be bound to the receiver upon pressing a similar button [94]. The transmitter was then calibrated in Mission Planner for maximum and minimum joystick values. The different flight modes were also assigned to the desired channels.

The fail-safe mechanisms were chosen and setup. The "radio fail-safe" specifically was included to force the drone to complete a new task when the radio signal lost connection or became too weak. Typically this new task would be for the drone to fly back to a set home position. However, this setting was not implemented since this drone did not include a GPS module which might result in the firmware attempting to fly the drone home despite not having any GPS location data. The drone was instead instructed to land safely and immediately.

The "battery level fail-safe" was also enabled, as it prevented irreversible battery damage from occurring if the voltage were to drop too low. The "EKF fail-safe"

(Extended Klaman Filter) should have been setup on this drone (discussed in the Results chapter), to note any large variance in the position data from the positioning sensors, and would have landed the drone if there was a variance of more than one second.

The pre-arm bitmask was adjusted, to set which pre-flight checks the firmware would do. Most of the checks were turned off (notably the arming safety switch) except for the barometer, compass, rangefinder and battery level. The other checks were either redundant or were intended for hardware which the system did not have.

The battery voltage and current monitoring sensors of the Power Brick Mini were then setup and calibrated using Mission Planner. This was achieved by adjusting the multiplier such that the reading displayed by the sensor (via telemetry) accurately matched the real world value that was tested using a multi-meter.

The ESC protocol was setup as DShot300, to correspond with the protocol used by the ESC, before the motors were tested. Calibration would have been required if the ESC used a different protocol, but DShot does not require calibration as was previously discussed.

The motor order and spin direction were changed on the software instead of resoldering the pins on the ESC, due to wire routing constraints. This was necessary since the motor order indicated on the PCB of the 4 in 1 ESC was inaccurate. Mission Planner's motor test allowed each motor to be individually spun at a slow speed, to test that the correct motor was spinning and in the correct direction. The direction and order of motors were then adjusted according to Figure 4.24; i.e. Motor A in Mission Planner refers to Motor 1 on the drone, and Motor B in Mission Planner refers to Motor 4.

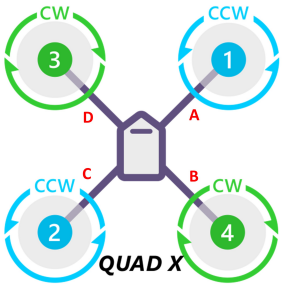


Figure 4.24: Motor Order

The optical flow sensor and rangefinder were configured by adjusting certain parameters. The "flow_type" parameter was set to "pixart", since the sensor on the Flow Deck V2 is a PMW3901MB made by pixart. Its physical position relative to the origin of the drone was set as well, and the EKF flow delay parameter was set to 80 for this sensor.

The rangefinder address was set to the I2C bus address 41 (specific to the Pixracer). The ground clearance parameter was also set to account for the distance away from the ground that the sensor should be at when stationary. The minimum value allowed was 5cm dictated by the sensors minimum functioning distance [95], however, the frame design used did not have landing gear which subsequently meant that the sensor made contact with the ground when stationary. This necessitated having to raise the drone away from the ground before takeoff in loiter mode.

The maximum height which the rangefinder supported was set to 360cm for the VL53l1X sensor, the rangefinder type was set as 16 (for VL53l0X and VL53l1x), the scaling factor was changed to 1, the orientation set as down, and the position relative to the origin set.

4.6 Tuning

Once the initial settings were configured, the flight tuning process began. Following the ArduCopter guide [96], initial settings needed to be inputted for the first flight to take place. These settings were dependant on the battery voltage, propeller size, the ESC used and the firmware version used. The settings determined the

maximum and minimum PWM signals allowed, the motor spin, hover value, and frequency needed for the gyroscope and accelerometer to function appropriately (differs depending on size of drone).

These settings were obtained by approximate relationship graphs previously tested by the developers of the firmware; i.e. the gyroscope filter frequency versus propeller size graph in Figure 4.25. For the approximately 2.55 inch (65mm) propellers used on the drone, the frequency was set to 100Hz.
Typically the frequency (related to sensor accuracy) for smaller drones is set higher than for larger drones, since small joystick adjustments can result in large movements for these small drones.

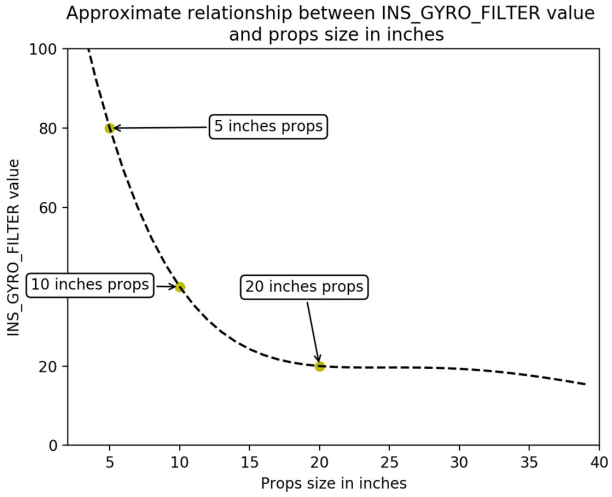


Figure 4.25: Gyro filter frequency(Hz) vs Propeller size (inches) [96]

After the first flight procedure was conducted, the drone was tuned to remove any oscillations observed. The tuning process consisted of reducing the PID values for the control loop until the oscillations were eliminated, and was tested in the "stabilise" flight mode. Once completed, the drone was tested in "althold" mode with the learn parameter enabled, so that the firmware could calculate an appropriate hover parameter.

After being evaluated the tune was found to be unsatisfactory, necessitating a

more accurate manual tune of PID values and of transmitter trim settings. Use of the autotune feature should have ensued to set a more accurate tune, however it demanded the use of a large, empty field. This was disregarded as a viable option due to perilous winds, rain and generally unpredictable weather conditions, which could have damaged the drone's electronics and would have nullified the autotune results (autotune is considered inaccurate in windy conditions, especially in mini-drones).

The optical flow sensor required a first time setup procedure for calibration. Initially, the sensor data was recorded during a flight in stabilise mode. These values were used to manually calibrate the sensors' measurements until they resembled the gyroscope's measurements. This was achieved by means of adjusting scalars and yaw modifiers. The sensor was then enabled for stabilisation and tested in the loiter flight mode [97]. However, this test was unsuccessful, as is further detailed in the Results chapter.

Chapter 5

Results

5.1 Results and Discussion

5.1.1 Final Component eCalc Results

The eCalc simulation input values used for the final system design are shown below in Figure 5.1.

It should be noted that the no-load current value used for the motors was estimated from various similar sized motors, KV and resistance. This was done since the actual value could not be obtained (the original motor manufacturer was liaised with via email: The product has since been discontinued and they do not have the values on hand). A more effective technique for obtaining the values would have been to measure the no-load current manually, however, additional equipment would be needed to measure the value correctly.

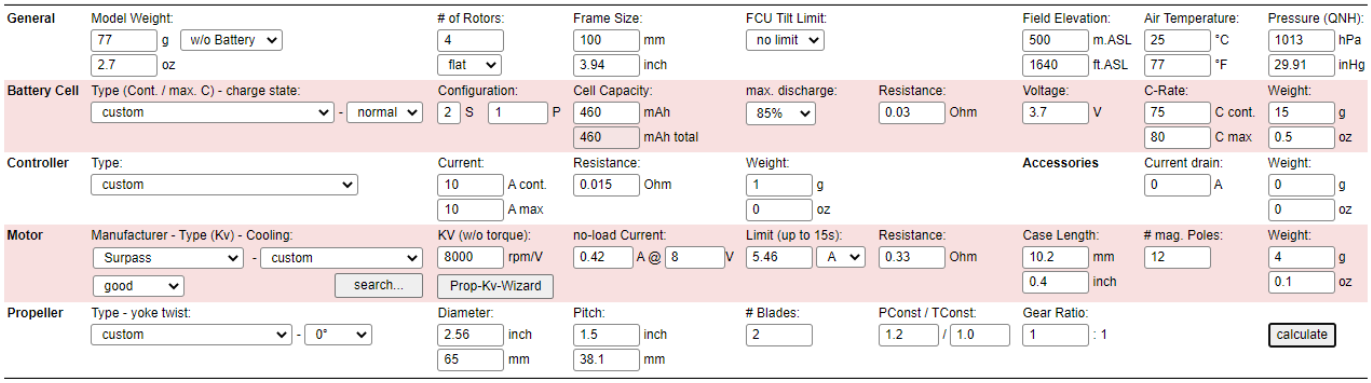
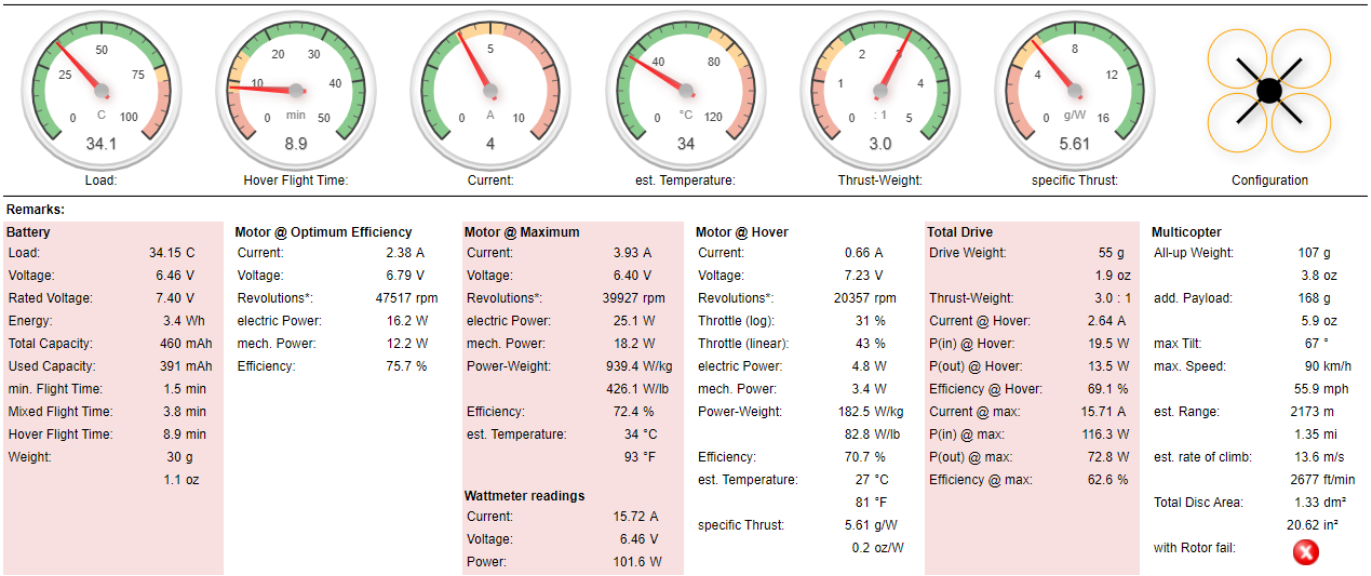


Figure 5.1: eCalc final component input values

From the calculated output seen in Figure 5.2, the expected hover flight time of the system was approximately 8.9 minutes, which was close to the 10 minute user requirement. Having anticipated that eCalc underestimated the achievable time, meant the system would most likely achieve its goal. The expected load on the battery was 34C, meaning the 75C battery chosen would be able to bear the discharge rate.

The maximum current the motors were expected to draw at full throttle was 3.93A, which was lower than the 5.46A rating which they each had. The specific thrust and the thrust to weight ratio were estimated to be 5.61g/W and 3.0:1 respectively, which were higher than the values obtained from the systems' simulated during the design phase. Additionally, eCalc claimed that the drone would be able to carry an additional 173g pay load for a short period of time, although this seemed like an unlikely real world outcome.



Remarks:

Battery		Motor @ Optimum Efficiency		Motor @ Maximum		Motor @ Hover		Total Drive		Multicopter	
Load:	34.15 C	Current:	2.38 A	Current:	3.93 A	Current:	0.66 A	Drive Weight:	55 g	All-up Weight:	107 g
Voltage:	6.46 V	Voltage:	6.79 V	Voltage:	6.40 V	Voltage:	7.23 V		1.9 oz		3.8 oz
Rated Voltage:	7.40 V	Revolutions*:	47517 rpm	Revolutions*:	39927 rpm	Revolutions*:	20357 rpm	Thrust-Weight:	3.0 : 1	add. Payload:	168 g
Energy:	3.4 Wh	electric Power:	16.2 W	electric Power:	25.1 W	Throttle (log):	31 %	Current @ Hover:	2.64 A		5.9 oz
Total Capacity:	460 mAh	mech. Power:	12.2 W	mech. Power:	18.2 W	Throttle (linear):	43 %	P(in) @ Hover:	19.5 W	max Tilt:	67 °
Used Capacity:	391 mAh	Efficiency:	75.7 %	Power-Weight:	939.4 W/kg	electric Power:	4.8 W	P(out) @ Hover:	13.5 W	max. Speed:	90 km/h
min. Flight Time:	1.5 min				426.1 W/lb	mech. Power:	3.4 W	Efficiency @ Hover:	69.1 %		55.9 mph
Mixed Flight Time:	3.8 min			Efficiency:	72.4 %	Power-Weight:	182.5 W/kg	Current @ max:	15.71 A	est. Range:	2173 m
Hover Flight Time:	8.9 min			est. Temperature:	34 °C		82.8 W/lb	P(in) @ max:	116.3 W		1.35 mi
Weight:	30 g				93 °F	Efficiency:	70.7 %	P(out) @ max:	72.8 W	est. rate of climb:	13.6 m/s
	1.1 oz			**Wattmeter readings**		est. Temperature:	27 °C	Efficiency @ max:	62.6 %		2677 ft/min
				Current:	15.72 A		81 °F			Total Disc Area:	1.33 dm²
				Voltage:	6.46 V	specific Thrust:	5.61 g/W				20.62 in²
				Power:	101.6 W		0.2 oz/W			with Rotor fail:	

Figure 5.2: eCalc final component output performance

The range estimation in Figure 5.3 showed that when travelling at the optimum speed of approximately 25Km/h, the drone would be able to cover a distance of approximately 2.17 Km, while taking into account standard drag values; i.e. on a day with clear skies and minimal wind speed.

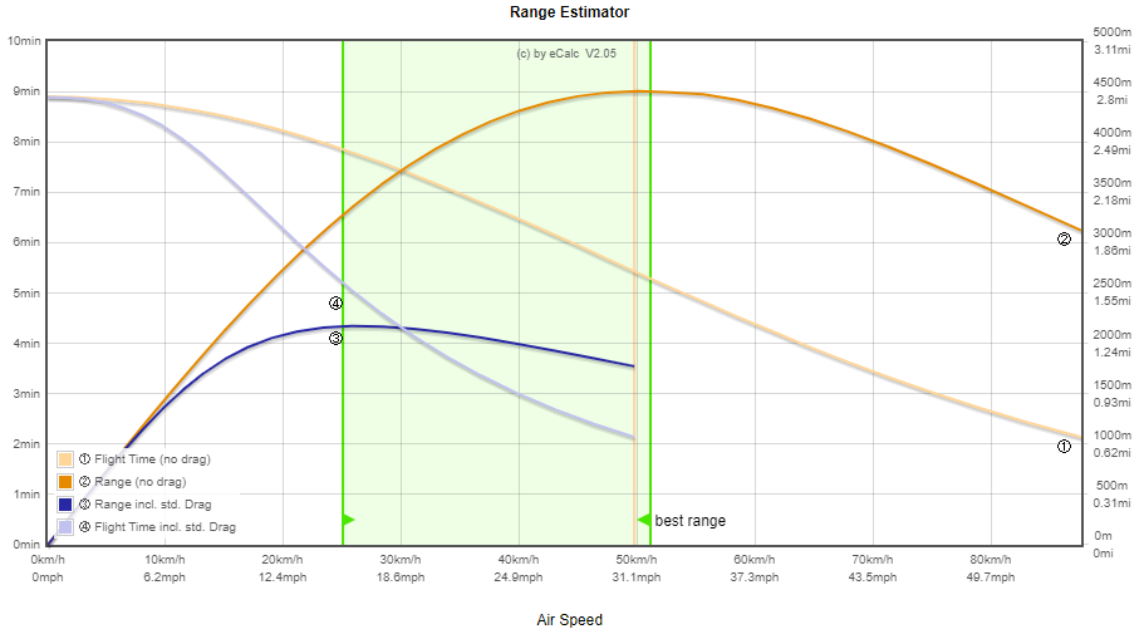


Figure 5.3: eCalc final component output range estimation curve

When observing the curve of efficiency versus motor current drawn, seen in Figure 5.4, the most efficient current draw for the chosen motors was 2.38A, while the current draw during hover was approximately 0.66A. This meant that these motors would be more efficient with a slightly larger sized frame and propeller combination, and that the configuration chosen was sub-optimal for the chosen motors.

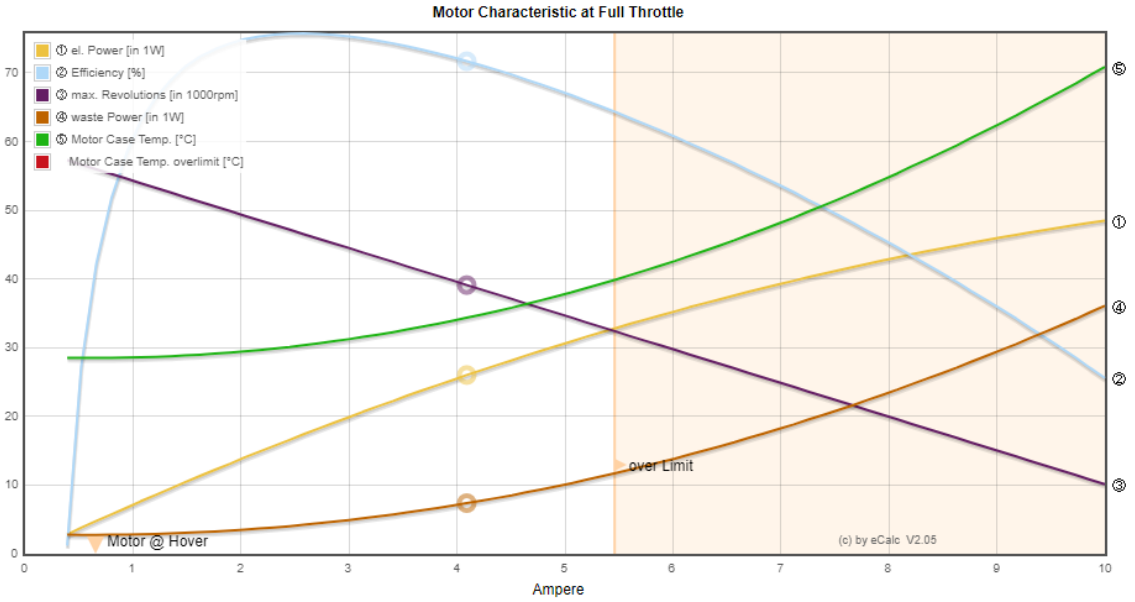


Figure 5.4: eCalc final component output motor characteristic curve at full throttle

5.1.2 Testing

During the first flight, the drone armed successfully but after increasing the throttle it accelerated out of control and into a roof, and was manually forced to a stop. This was as a consequence of using an Xbox controller as the joystick (over telemetry) in the interim, instead of the designated drone transmitter which was not yet made available by that point. The Xbox controller's throttle trigger had a very short travel, meaning that even incremental effort on the controller correlated to a large change in throttle.

After the tuning flight, and some adjustments, the drone appeared to be flying well, but crashed into a wall once again. It is anticipated that this crash was caused

by the autopilot since this drone does not feature a GPS module. The GPS module is highly recommended by the ArduPilot developers, the firmware by default uses the GPS to navigate in autonomous modes. This meant that when the firmware could not find GPS positioning data during the flight, it would automatically revert to getting its positioning data from the optical flow sensor which had not yet been calibrated, causing the drone to collide with the wall.

Another possible cause could be the GPS or throttle return to landing (RTL) fail-safes, which force the drone to return to its home position should the requirements be met but which cannot be set without a GPS. After disabling all of these features, the drone tuning process was completed as expected. However, completing an autotune in a confined space was an impossible task considering the drone needed to move around and adjust its roll, pitch and yaw, to test different PID values.
This meant that the remaining testing was completed with an imperfect PID tune as mentioned previously, which was controllable but which had drift and was sensitive to inputs from the joysticks.

Testing note

The testing videos concerning the following data are available for viewing using the following YouTube link and in the Github page for this project.

Testing flight time:

The drone's hover flight time testing was completed in stabilise mode, and both of the conducted tests, shown in Table 5.1, were done with fully charged batteries. The time was recorded from the instant the drone lifted off from the ground, to the moment the motors decreased in power output.

Mode	Flight Time (s)
Stabilize with only small adjustments for drift, Test 1	7.192
Stabilize with only small adjustments for drift, Test 2	7.18

Table 5.1: Drone flight time results

The final hover time was approximately 7.19 minutes, which is lower than the expected hover time of 8.9 minutes by 1.7 minutes. Some of this loss in time was caused by the drone having to continuously be centred to correct for drift and to account for not being entirely still. The drift was most likely caused by an inaccurate tune and radio trim settings. Although, another contributor of the reduced flight time could have been the imperfect no-load current used when simulating the final system's expected hover time, as mentioned before.

The hover time was also approximately 2.8 minutes lower than the desired hover time of 10 minutes that was set out in the user requirements. This time could possibly have been achieved by reducing the overall weight of the system, or by increasing the propeller size; since propellers are more efficient the larger their diameter. The frame diameter would also then have to be increased while attempting to maintain the current weight as much as possible.

Alternatively, a larger capacity battery could be used. Although, the battery should then have a similar weight to the current battery in order to be effective. This is a plausible option given that the C rating for the current 460mAh battery (75C) is higher than what is required for the system (40C), according to eCalc, as it was the only 2S battery available with a high enough discharge rating at the desired capacity of 460mAh. A higher capacity battery which meets the desired discharge rating (C rating would be lower at a higher capacity) at the same weight is likely to exist.

Testing speed over a 2m distance:

The average speed and acceleration of the drone were tested by having the drone start from various hover heights in stabilise mode, and thereafter pushing the directional joystick on the transmitter to the far left until the drone flew across the desired distance. The tests were conducted indoors between two window panes that were exactly 2.4m apart. The tests were recorded using a camera at 50 frames per second, and a video editing software was used to measure the time from the start point to crossing the end point.

An example of the start and end points of the real world setup are described in Figure 5.5.

Figure 5.5: Speed test start and end points

The average speed of the drone across the measured distance was calculated using the test distance of 2.4m, divided by the time taken from passing the first window pane until completely passing the second window pane. The average acceleration was calculated by dividing the average speed (calculated above) by the time at initial movement until passing the second window pane. However, this was not a very accurate acceleration calculation due to it relying on the average speed calculated before, which could not accurately be obtained without proper measurement equipment.

Number	Speed (m/s)	Acceleration m/s/s
1	3.87	3.76
2	4.36	4.4
3	3.08	2.68
4	3.33	3.62
Average	3.66	3.615

Table 5.2: Drone speed results

From the results seen in Table 5.2, it is evident that the speed values obtained vary by approximately 1.28m/s, which is significant. These variations were most likely caused by the drone not having flown perfectly horizontally during some of the tests; it moved perpendicularly inward and outward slightly during its flight (inward and outward of the screen if viewing the video). This had an effect on the time taken to cover the test distance, and since the time values were so short (sub 1 second) any variance would have a substantial impact on the calculated speed.

The speed values were also lower than expected, when compared to other quadcopters of a similar size. This is because the testing area (indoors) was far too small to set the drone's throttle to its maximum value and test its maximum speed.

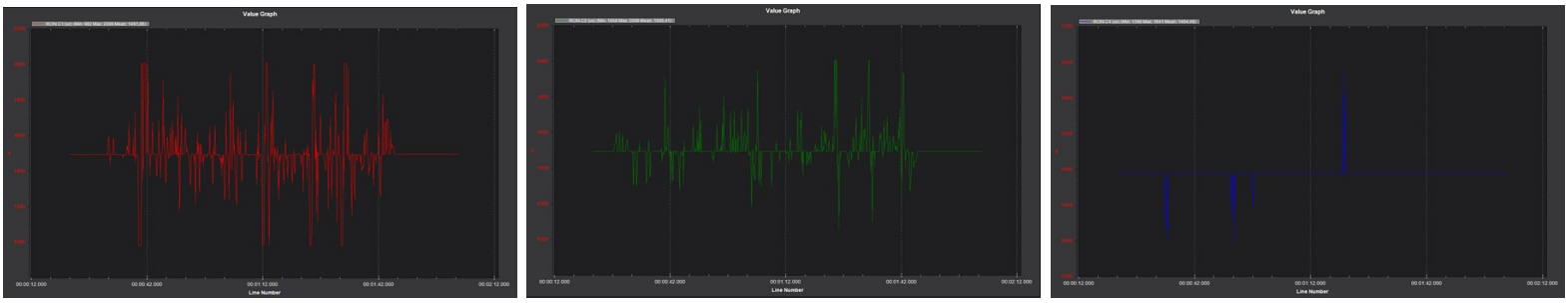
Figure 5.6: Log data, horizontal acceleration and motor throttle

The graph of horizontal motor acceleration and motor throttle is seen in Figure 5.6. The throttle is represented by the spiked, red line as it is being sent to the motors for the speed tests that were conducted. The acceleration is then represented by the spiked, green line, and is measured from the Pixracer's accelerometer, in the direction of the drone's movement. The measured values of acceleration ranged from 3.4m/s/s to 2.23m/s/s for the 4 tests, which are close to the calculated values anticipated. However, there is a large error margin, because the values calculated were using the inaccurate average velocities obtained before.

More accurate tests could have been conducted if the speed was measured using a speed camera or radar gun in a large open field, with the drone flying at full speed past the sensor.

Judging agility/ease of control:

The quadcopter was in stabilise mode when testing its ease of control and agility. It responded quickly to adjustments in roll, pitch and yaw and was very directionally accurate; it flew according to its transmitter input accurately. However, as mentioned before, the drone had a significant amount of drift, caused by the imperfect tune, which meant that adjustments to its position needed to be made constantly. These adjustments can be seen by the roll, pitch and yaw inputs in Figure 5.7, with the roll and pitch being used more than the yaw.
This could additionally have been caused by the system's weight balance being slightly off. Considering the small size of the drone and motors, any imbalance in weight could have an effect on the amount of drift.



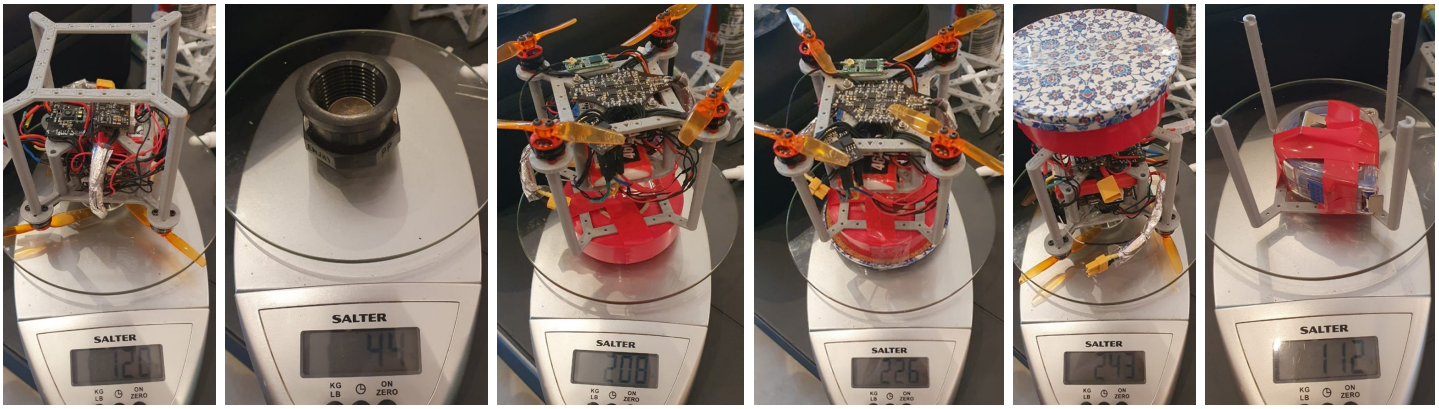
(a) Roll (b) Pitch (c) Yaw

Figure 5.7: Log data, roll, pitch and yaw inputs to drone

To account for this drift the frame could be redesigned, such that the weight of each component be more centralised. For instance, the battery was not perfectly centred due to the SPI port on the Pixracer protruding downwards on one side, forcing the battery to be placed besides it. Certain components such as the Power Brick could potentially be separated into two by desoldering. Wires could then be used to replace the header which connected the two halves, which would allow for more even distribution of the weight of the component across the frame.

Payload testing:

The payload capacity of the drone was tested by incrementally adding weights onto the drone until it was unable to hover. The stand designed for takeoff procedures was glued to the frame and used as a support for the weights. The full weight of the system was 107g while the stand weighed 13g, as such a 13g weight was added to the weight of each of the payloads.
The stand attached to the frame and each of the payloads tested (consisting of round objects with even weight distribution) are shown in Figure 5.8. The uneven weight used for testing was a tape measure, which had loose loose internal components and which had an unsymmetrical shape.



(a) System w stand (b) 59g w/o stand (c) 101g payload (d) 119g payload (e) 136g payload (f) 112g uneven

Figure 5.8: Payloads tested

An attempt to fly the drone with the payloads attached was conducted for each of the different weights, with the test filmed by camera. The footage was reviewed and tabulated in Table 5.3. From the results, it is evident that the maximum payload the system was able to carry, while still being able to hover and manoeuvre sufficiently, was between 60 and 100g. Any weight above 100g required too high a throttle to manoeuvre the drone while hovering. The maximum weight the drone was able to sufficiently hover with was between 119g and 136g, which was lower than the 173g expected from the eCalc calculation; although the eCalc value anticipated was optimistic.

Payload	Comment
59g	Stable and able to fly
101g	Stable but needs higher throttle
119g	Stable but needs higher throttle
136g	Stable but throttle maxing out
112g uneven distribution	Unstable

Table 5.3: Drone maximum payload capacity test

The performance of the uneven payload, despite it being less weight than the maximum payload which the drone could hover with, was very unstable and caused

the drone to crash. This reiterates that the positioning, distribution and secure attachment of the weight affects the stability of the drone. This again emphasises that any incorrect weight distribution of the system could be a cause for the drift evident while hovering.

Ultimately, this test could have been improved with a more secure weight mounting method and by having smaller weights, so that the payload could be more finely incremented to determine a more accurate maximum payload.

Quantifying holding position accuracy:

When testing the height position accuracy, the ArduCopter firmware automatically employed the rangefinder for height position up to its maximum usable height (in althold mode) of 3.6m for the VL53L1x sensor used, and thereafter would switch to the barometer for its height positioning. To test the rangefinder alone, the drone was not flown above the height of 3.6m. And, when testing the barometer alone, the rangefinder and optical flow were disabled so that the drone would not have to be flown above 3.6m. To test the optical flow sensor exclusively, the flowhold mode was used as it obtained its positioning and velocity data from the flow sensor only. When testing the rangefinder and optical flow sensor together, loiter mode was used below the height of 3.6m, as the two sensors are used in conjunction by default in this mode [98].

However, when attempting the tests which involved the optical flow sensor, the drone flew erratically and crashed. The flight logs showed that the optical flow sensor data was not tracking the changes in the drone's movement correctly. The inaccurate tracking mentioned can be seen by the X-axis flow rate versus derived body velocity graph in Figure 5.9. The flow rate data, shown in orange, should have closely resembled the derived velocity data, shown in blue, but was instead much smaller and somewhat uncorrelated, despite having been tuned appropriately before flight testing.

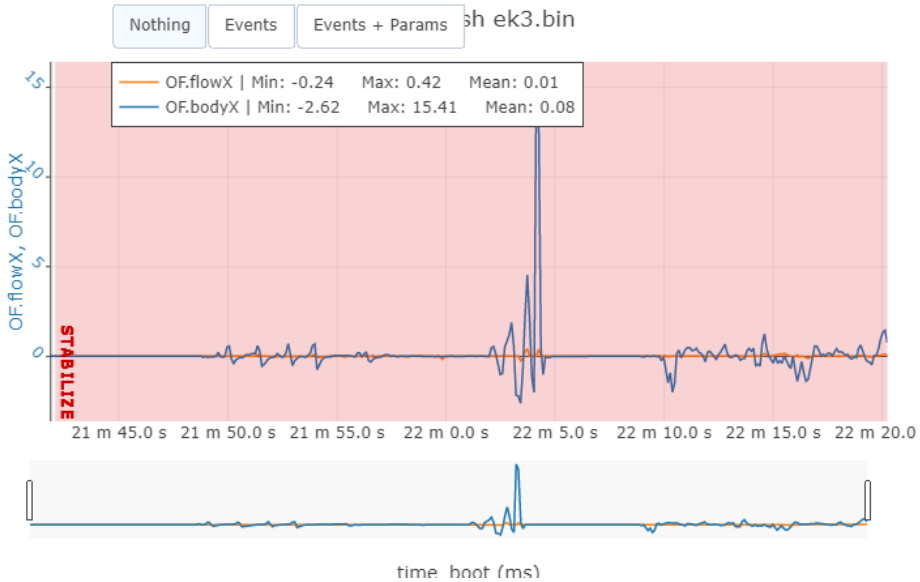


Figure 5.9: Optical flow sensor and derived body velocity data of crashed flight

The setup procedure for the optical flow sensor, as mentioned in the design section, required that the flow scalars be adjusted until the flow rate in the X and Y axes corresponded to their respective derived velocities. The correlation was tested in a number of scenarios, to find the cause of the malfunction. Initially the sensor was tested with the Pixracer only powered via USB, and the data from this test showed the sensor to be functioning much more appropriately, as seen in Figure 5.10. It was then suspected that the cause of the malfunction was either EMF interference from the battery, motors or ESC (as the motors were spinning during the test), from noise generated by the Power Brick module's switch mode regulator, or from the SPI link being noisy, since the sensor was seemingly unaffected when powered via USB.

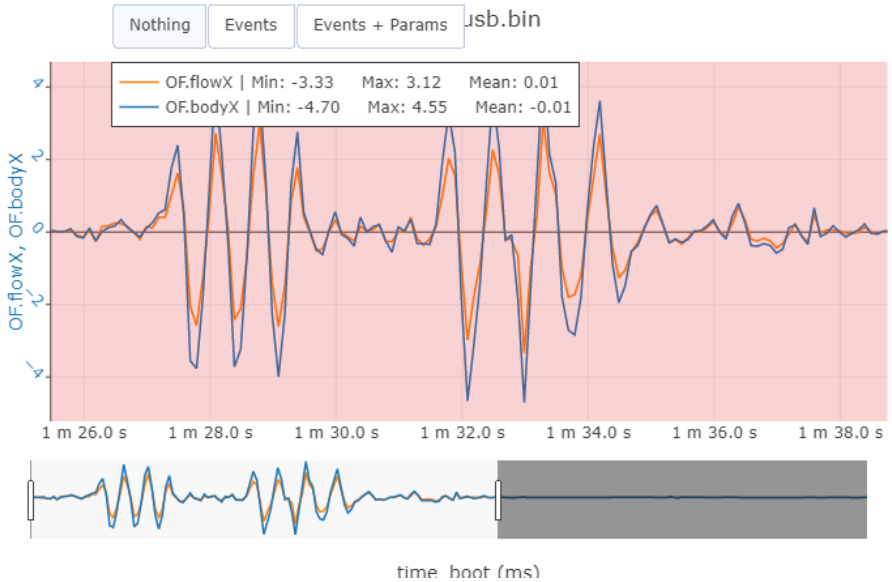


Figure 5.10: Optical flow sensor and derived body velocity data (powered only via USB)

The sensor was then tested with the battery powering the system, as seen in Figure 5.11. The sensor data was more erratic in this instance and did not correlate to the body velocity as it did when the system was powered by USB. Noise or interference from the power module, which was physically located near the flow sensor, were anticipated possible causes for this behaviour. However, in attempt to remove noise which may have been caused by the switching regulator in the Power Brick module, similar behaviour was observed when testing the system with the battery connected and the Pixracer powered by a LM7805 linear voltage regulator (non-switching regulator).

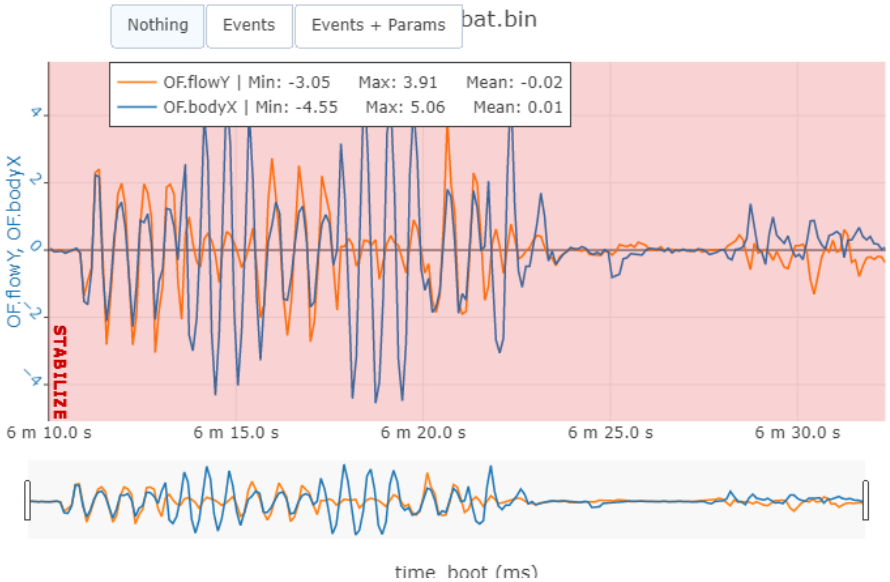


Figure 5.11: Optical flow sensor and derived velocity data (powered only via battery)

Upon further investigation of the issue, the wiring for the system was re-evaluated and it was found that the Flow Deck was mistakenly connected to a 5V source instead of the 3.3V designated in the design chapter. When observing the Flow Deck's datasheet, it was noted that a 1.8V regulator was present for the logic power on the PMW3901 sensor, as well as the fact that the SPI and I2C communication pins on the Pixracer used 3.3V logic. This was anticipated to be the reason why the sensors continued to function despite the erroneous power source. Switching to the 3.3V source reduced the error to a more acceptable level, however it was not fully eliminated, evident in the fluctuations toward the end of the test in Figure 5.12. It was anticipated that some interference was still causing imperfect behaviour with the optical flow sensor.

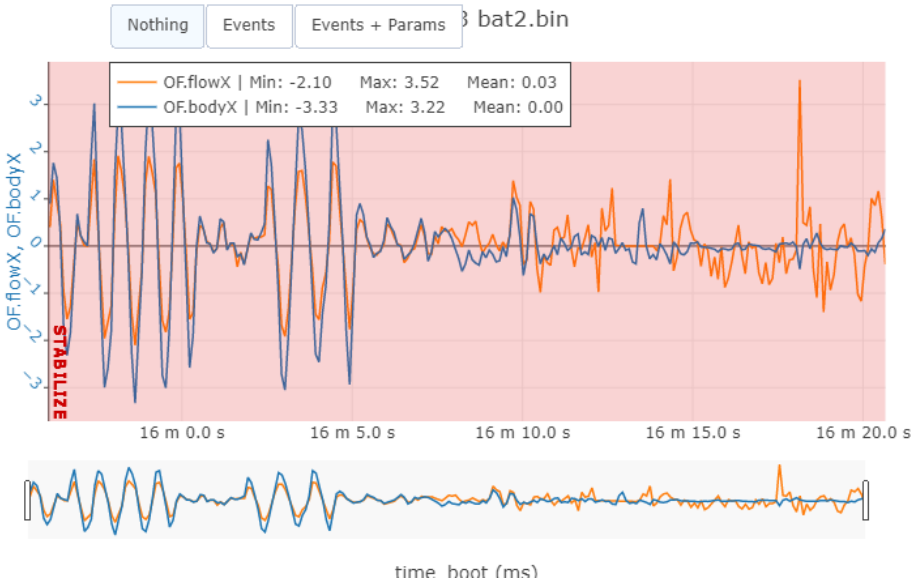


Figure 5.12: Optical flow sensor and derived velocity data (powered only via battery, with corrected input voltage)

The sensor error was reduced to an acceptable level after this change, however, time did not permit the holding position tests to be conducted, and a large empty area would have been required to safely test the drone in the desired modes.

Regardless, it was anticipated that the drone would be most stable when flying in loiter mode using both the optical flow sensor and rangefinder for position tracking. This is because the rangefinder was observed to have a more accurate height estimation compared to the barometer, as seen in the comparison graph in Figure 5.13. The rangefinder was much more consistent in its readings and was far more accurate in describing the actual height of the drone, which was approximately 800mm above the ground during this test. The barometer reading did not correlate with the real world altitude.

The optical flow sensor was also anticipated to provide a better horizontal position estimation compared to the IMUs on the Pixracer. This is because the optical flow sensor is a form of camera, meaning it would be able to physically see if the drone was changing position, as opposed to estimating position based on acceleration

and orientation, which is the method the IMU sensors use.

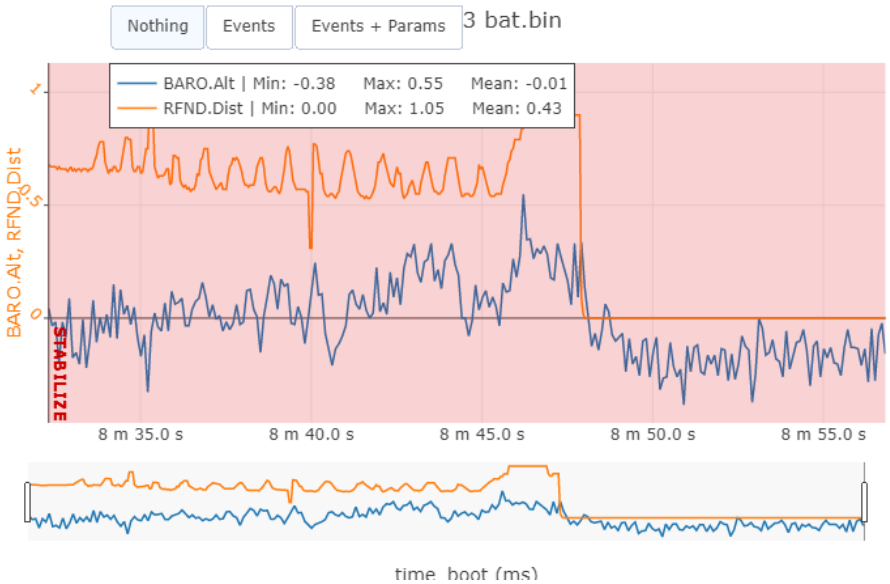


Figure 5.13: Rangefinder height estimation versus barometer during test flight

The system managed to be tested in althold mode as this did not require the optical flow sensor, and it was able to hold its position using the rangefinder. However, above the rangefinder's maximum reading distance of 3.6m, the drone's altitude was not as stable without the rangefinder. This aligns with the observation that the rangefinder's altitude accuracy was superior to the barometer's. However, this was tested under imperfect conditions, as it was conducted outdoors with some level of wind (the footage for this test is unavailable due to human error in the recording procedure).

Optical flow condition test:

Due to the malfunction observed when using the optical flow sensor during flight, as mentioned above, the optical flow condition testing was not conducted. Nonetheless, the procedure for the test would have involved hovering the drone in flowhold and loiter modes over various ground surfaces, including reflective tiles, semi-reflective tiles, bricks and grass, and would have been tested in light and darkness for the

various surfaces.

The performance would have been measured by filming the tests and measuring how well the drone could maintain its vertical and horizontal position relative to a set point in the video frame. It was anticipated that the drone would function well above textured surfaces, such as grass, but only in certain lighting conditions. If the surrounding light level was too bright the sensor would not be able to correctly view the ground surface as the image would either be saturated or washed out. Or if the surrounding light level was too dark, the variations of the ground would be indistinguishable. These conditions would be worse since the sensor does not have any auto-focus or white balance correction capabilities.

Additional modes testing:

The circle mode was also not tested because of the fault with the optical flow sensor. However, had the sensor functioned correctly, loiter mode would have been usable, as well as circle mode since it functions with the same principles as loiter mode [99]. The other modes available in ArduPilot required either a GPS or the optical flow sensor and as such were not tested within the available time for this project [100].

Chapter 6

Conclusions

This project aimed to answer the question: What are the steps required, to design, build and test a functioning mini-drone, which features an open source firmware and an optical flow sensor? In response, a drone needed to be built and tested, and which needed to achieve the desired user requirements set out in the methodology section. The main requirements stipulated that the drone needed to be battery powered, use open source firmware, be able to hover for approximately 10 minutes, be easily controllable, weigh less than 250g, have some autonomous features, and be fully open source and well documented.

Initially a literature review was undertaken to gain a better understanding of the physics of a mini-drone and to determine the necessities for designing and developing one. The design process was then initiated, which included various system simulations using eCalc software, and employing the performance values supplied on locally available component websites. This process was repeated until suitable systems were obtained. The pros and cons of each system were examined and a final system was decided upon. Due to unforeseen circumstances when purchasing the chosen components, the system had to be marginally modified to use different motors and a different power module.

Once the components arrived, they were each measured with calipers. Suitable

frames were iteratively designed and 3D printed in accordance with these measurements, and for use as a base for all the components. The system was assembled, and certain components were modified to operate more effectively in accordance with the rest of the system. Namely, the firmware on the flight controller had to be modified to enable an SPI connection on one of the connectors, for use with the provided optical flow sensor. The system was then setup in software and flight tuning was conducted, which resulted in a number of issues and crashes. This resulted in the redesign of the frame and implementation of more software adjustments. Finally the system was tested with an imperfect flight tune, and the results discussed.

6.1 Functionality Assessment

The system met the first user requirement, to develop and build a functioning mini-drone which uses an open source firmware (UR01). The system was designed to use a battery to power it; i.e. it was untethered (UR02). The system was also scaleable in that the core components, namely the flight controller, telemetry, receiver, optical flow sensor and Power Brick, could be used with a different motor, propeller, ESC and battery combination so that the entire system could be made as a much larger version (UR03). The flight controller also had unused ports for feature expansion (UR03). It was easily controllable by a human pilot (UR05), despite having hover drift. The weight was 107g which was less than half of the maximum allowable system weight of 250g (UR06).

It was able to automatically stabilise itself in the air without oscillations (UR08). The project is fully open-source (UR10) and is well documented (UR11), with a GitHub page containing all of the design files and instructions required to fully recreate the system.

The system was also able to hold its position in 3D space on command when tested in althold mode, but it did have some level of drift (UR09). The system was not tested in its ability to maintain its position using the optical flow sensor and rangefinder together, which would have produced better holding results.

The drone was unable to meet the 10 minute hover (UR04) established in the user requirements, only being able to hover for approximately 7 minutes and 20 seconds before the battery was fully depleted. A larger capacity battery would be a solution to this issue.
The system was not able to meet its budget target of 1500 Rand either, in part due to increased pricing, but mainly due to time constraints. With more time, an all in one PCB and frame could have been designed, which would have been lighter, have had all the desired features, and would have been cheaper if the premium priced Pixracer was avoided. If the whole system was lighter, smaller brushed motors and mosfets could have been used and would have reduced costs as well.

The system was not tested in circle mode, to test whether the drone could fly in a circle without user input, and to test whether the drone could be programmed to follow a certain pre-set path (UR07). This was because of the optical flow sensor malfunction mentioned in the results chapter.

6.2 Reliability Assessment

The system was reliable in terms of its ability to fly in stabilise mode every time it was turned on. However, some other modes, such as loiter mode, which used the optical flow sensor and rangefinder to hold its position, required that the position of the system be set before launch, which did not always work immediately and often required a reboot of the system. The system did not include a GPS module in its design, which is what ArduPilot is typically used with. As such, certain autonomous features could not be tested, including autonomous long distance flight.

When observing the crash resistance of the system, the 3D printed material used for the frame was not very strong, which meant that the system was unable to withstand the crashes, necessitating the frame be repaired or replaced after each crash. The frame also did not have any landing gear due to design weight constraints, which meant that if the system stopped working mid-air, it would land on some electronics

upon crashing which could damage them.

Some elements which worked well for this design were the brushless motors, which had sufficient power to allow the addition of a 60 to 100g payload onto the system, such as a small camera for video capture. The system was easy to control and the height estimation from the range finder worked well to maintain the altitude of the drone in althold mode.

6.3 Future Work and Recommendations

This project was originally aimed to be a basis for further research and development, as such there are a number of recommendations for further work and improvements.

The main avenue of improvement for this project would be to reduce the price of the system. This could be done by designing a custom PCB, which would include all of the electronics for the desired features, such as a micro controller with SPI, I2C and SBUS connections available via headers, an ESP 8266 IC for telemetry, as well as a barometer, gyroscope, accelerometer and compass. Additionally, power regulation and distribution would be built into the design, as well as mosfets for the desired brushed motors and mounting holes for them. Having all the components integrated into a single PCB as opposed to having them on separate modules, would significantly reduce the weight and price of the system.

Alternatively, should the system be kept more configurable and usable with different frame, motor and propeller sizes, the more affordable Omnibus flight controller could be explored to reduce costs. Carbon fibre could be used for the frame, with either delrin studs or carbon tubes used to hold the remaining components in place. Another frame design would involve the frame being injection moulded, as this would allow the frame to be a single piece of plastic instead of multiple pieces which need to be assembled. This would be stronger than the 3D printed frame, since the 3D

printed parts are created in layers which can often lead to cracking along the print seams, while a moulded part would consist of one piece of hardened plastic.

A study into the effects of 3D printing infill settings, on the strength and weight of the printed parts, could be conducted. When Slicing a 3D model to be printed (converting the file into instructions for the printer), the setting for the amount of infill material used was present. Typically prototype 3D printed parts use a 20% infill, however this project used 100% infill, as it was assumed the parts would be stronger. Reducing the infill could potentially allow for weight savings.

Another area of future research to further this project, would be to investigate the EMF and radio interference emitted by the various components on the drone, as they have an effect on the positioning accuracy of the IMU and other positioning sensors. Observations could also be conducted on the effects of magnetic interference on the compass sensor, or magnetometer, as it was observed that the system would have magnetic field variance when the drone was situated in close proximity to certain metal objects including a metal table frame. An investigation could also be conducted into methods for minimising or mitigating the effects of the various interference classifications mentioned above, as they could affect the functionality of the drone.

Some possible improvements for the final system design used:

- Slight ameliorations in weight could be made by improving the cable management of the drone, as some wires were excessively long to accommodate easier installation but which could be shortened to improve weight.
- The system could also greatly be improved by implementing some form of landing gear, to prevent the drone from landing on its electronics and potentially damaging them.
- A WiFi repeater could be made using a Raspberry Pi, which would allow for

the telemetry link to function across further distances. It could also act as a WiFi bridge, allowing the ground control laptop to be connected to both the drone's telemetry module and an internet source. The method used for connection saw the telemetry module emitting a hot-spot which the laptop would have to connect to, forcing it to disconnect from its internet connection.

- A different ESC (one meant for custom drones) should be used, as the ESC chosen was designed to be used with a mass produced mini-drone, the DYS Elf. The chosen ESC had a very small header for the signal wires, which were difficult to solder to even with a corresponding female header attached onto it.

- Better mounting for the components to the frame could be designed, as opposed to using clear glue. Although, this would add weight which is why it was not implemented in this project. This means that the drone would need to be scaled up in size, so that the added mounting weight becomes insignificant.

- Increasing the system's size with larger motors, propellers and frame, toward meeting the mini-drone weight limit of 250g, would also result in more airborne stability and would be easier to assemble. This is because the 100mm frame and small components made assembly difficult, with wire lengths repeatedly needing to be redone for certain components to fit.

Bibliography

[1] U. F. A. Administration, "Law enforcement guidance for suspected unauthorised uas opertions." U.S Department of transportation, Dec 2015, (Accessed on 06/11/2021).

[2] ——, "How to register your drone," https://www.faa.gov/uas/getting_started/register_drone/, Oct 2021, (Accessed on 06/11/2021).

[3] S. Herrick, "What's the difference between a drone, uav and uas," https://botlink.com/blog/whats-the-difference-between-a-drone-uav-and-uas, November 2017, (Accessed on 11/09/2021).

[4] "Commercial drone market size, share & trends analysis report," https://www.grandviewresearch.com/industry-analysis/global-commercial-drones-market, (Accessed on 11/09/2021).

[5] K. Vyas, "A brief history of drones: The remote controlled unmanned aerial vehicles (uavs)," https://interestingengineering.com/a-brief-history-of-drones-the-remote-controlled-unmanned-aerial-vehicles-uavs, June 2020, (Accessed on 11/09/2021).

[6] J. Daniels, "How did drones get their name?" https://www.bestspy.org/drones-get-name/, April 2020, (Accessed on 11/09/2021).

[7] D. Daly, "A not-so-short history of unmanned aerial vehicles (uav)," https://consortiq.com/short-history-unmanned-aerial-vehicles-uavs/, (Accessed on 11/09/2021).

[8] B. D. Shelley Canright, "The four forces of flight," https://www.nasa.gov/audience/foreducators/k-4/features/F_Four_Forces_of_Flight.html, April 2009, (Accessed on 12/09/2021).

[9] D. S. Pachpute, "Working principle and components of drones," https://cfdflowengineering.com/working-principle-and-components-of-drone/, 2021, (Accessed on 5/10/2021).

[10] "Hobbywing xrotor f405 g3 omnibus f4 flight controller," https://www.rcworld.co.za/product_details.php?proid=1516, 2021.

[11] "Kingkong fly egg rc 2,3s blheli_s 10a esc," https://www.rcworld.co.za/product_details.php?proid=434, 2021.

[12] "Emax cf 28-05 brushless outrunner 1600kv," https://www.rcworld.co.za/product_details.php?proid=294, 2021.

[13] "10x4.5 propellers," https://www.rcworld.co.za/product_details.php?proid=7, 2021.

[14] "Driving using a single mosfet," http://hades.mech.northwestern.edu/index.php/Driving_using_a_single_MOSFET, 2006.

[15] Dejan, "How brushless motor and esc work," https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/, 2019, (Accessed on 12/09/2021).

[16] "Riva180 mini racer carbon fiber - frame," https://www.rcworld.co.za/product_details.php?proid=1174, 2021.

[17] "Riva180 mini racer carbon fiber - frame," https://www.rcworld.co.za/product_details.php?proid=672, 2021.

[18] "Turnigy ia6 receiver 6ch 2.4g afhds 2a receiver," https://www.rcworld.co.za/product_details.php?proid=850, 2021.

[19] "Radiomaster tx16s," https://www.rcworld.co.za/product_details.php?proid=1520, 2021.

[20] "Flow deck v2," https://store.bitcraze.io/collections/decks/products/flow-deck-v2, 2021.

[21] G. team, "New to fpv? start here!" https://www.getfpv.com/learn/new-to-fpv-beginner/, 2021, (Accessed on 5/10/2021).

[22] K. Sands, "Newton's laws of motion," https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/newtons-laws-of-motion/, May 2021, (Accessed on 12/09/2021).

[23] J. Maltos, "The flight physics of quadcopter drones," http://ffden-2.phys.uaf.edu/webproj/211_fall_2018/J-Rod_Maltos/physics_4th.html, Fall 2018, (Accessed on 12/09/2021).

[24] R. Gurung, "Tech talk quadcopter basics," http://xybernetics.com/techtalk/miscellaneous/tech-talk-quadcopter-basics/, April 2016, (Accessed on 12/09/2021).

[25] N. S. Gipson, "Advanced air mobility: The science behind quadcopters," https://www.nasa.gov/aeroresearch/stem/AAMhttps://www.nasa.gov/sites/default/files/atoms/files/aam-science-behind-quadcopters-reader-student-guide_0.pdf, July 2020, (Accessed on 12/09/2021).

[26] G. C. R. Sincero, J. Cros, and P. Viarouge, "Efficient simulation method for comparison of brush and brushless dc motors for light traction application," in *2009 13th European Conference on Power Electronics and Applications*, 2009, pp. 1–10.

[27] M. Yildirim, M. Polat, and H. Kürüm, "A survey on comparison of electric motor types and drives used for electric vehicles," in *2014 16th International Power Electronics and Motion Control Conference and Exposition*, 2014, pp. 218–223.

[28] Y. Mulgaonkar, M. Whitzer, B. Morgan, C. M. Kroninger, A. M. Harrington, and V. Kumar, "Power and weight considerations in small, agile quadrotors," in *Micro- and Nanotechnology Sensors, Systems, and Applications VI*, T. George, M. S. Islam, and A. K. Dutta, Eds., vol. 9083, International Society for Optics and Photonics. SPIE, 2014, pp. 376 – 391. [Online]. Available: https://doi.org/10.1117/12.2051112

[29] H. Yan, S.-H. Yang, Y. Chen, and S. A. Fahmy, "Optimum battery weight for maximizing available energy in uav-enabled wireless communications," *IEEE Wireless Communications Letters*, vol. 10, no. 7, pp. 1410–1413, 2021.

[30] M. N. K. Othman, Z. M. Razlan, M. S. M. Azmi, I. Zunaidi, W. K. Wan, A. B. Shahriman, M. S. M. Hashim, M. K. Faizi, I. Ibrahim, and N. S. Kamarrudin, "Experimental study on relation of small UAV propeller thrust generation and the mouth-ring," *IOP Conference Series: Materials Science and Engineering*, vol. 429, p. 012079, nov 2018. [Online]. Available: https://doi.org/10.1088/1757-899x/429/1/012079

[31] M. Herpperle, "How a propeller works," https://www.mh-aerotools.de/airfoils/propuls4.htm, May 2018, (Accessed on 12/09/2021).

[32] T. Gudde, "Flight controllers explained for everyone," https://fusion.engineering/flight-controllers-explained-for-everyone/, 2021, (Accessed on 14/09/2021).

[33] B. team, "Betaflight," https://betaflight.com/, 2021, (Accessed on 14/09/2021).

[34] M. team, "Multiwii," http://www.multiwii.com/, 2021, (Accessed on 14/09/2021).

[35] U. F. A. Administration, "Advanced avionics handbook." U.S Department of transportation, 2009, (Accessed on 14/09/2021).

[36] PX4, "Px4 and 3d robotics present: Pixhawk," https://diydrones.com/profiles/blogs/px4-and-3d-robotics-present-pixhawk-an-advanced-user-friendly, August 2013, (Accessed on 14/09/2021).

[37] ——, "Mav computer vision student team," https://web.archive.org/web/20170611002351/https://pixhawk.ethz.ch/, August 2013, (Accessed on 14/09/2021).

[38] A. Team, "Copter," https://ardupilot.org/copter/index.html, 2021, (Accessed on 14/09/2021).

[39] J. McHale, "Ground control stations for uavs," https://www.militaryaerospace.com/rf-analog/article/16723798/ground-control-stations-for-unmanned-aerial-vehicles-uavs-are-becoming-networkinghub-cockpit June 2010, (Accessed on 20/09/2021).

[40] A. team, "Choosing a gcs," https://ardupilot.org/copter/docs/common-choosing-a-ground-station.html, 2021, (Accessed on 20/09/2021).

[41] Q. Team, "Qgroundcontrol overview," https://docs.qgroundcontrol.com/master/en/index.html, 2021, (Accessed on 20/09/2021).

[42] A. Team, "Mission planner overview," https://ardupilot.org/planner/docs/mission-planner-overview.html, 2021, (Accessed on 20/09/2021).

[43] H. Team, "Hubsan range," https://www.hubsan.com/na/index.php?main_page=allkinds, 2021, (Accessed on 20/09/2021).

[44] B. Team, "Betafpv drones," https://betafpv.com/collections/all-drone, 2021, (Accessed on 20/09/2021).

[45] A. Team, "Skyrocket," https://ardupilot.org/copter/docs/skyrocket.html, 2021, (Accessed on 20/09/2021).

[46] S.-V. Team, "Sky-viper journey + comparison," https://sky-viper.com/journey/, 2018, (Accessed on 20/09/2021).

[47] B. Team, "Crazyflie 2.1," https://www.bitcraze.io/products/crazyflie-2-1/, 2021, (Accessed on 20/09/2021).

[48] D.-H. Yi, T.-J. Lee, and D.-I. Cho, "Afocal optical flow sensor for reducing vertical height sensitivity in indoor robot localization and navigation," *Sensors*, vol. 15, no. 5, pp. 11 208–11 221, 2015. [Online]. Available: https://www.mdpi.com/1424-8220/15/5/11208

[49] ——, "A new localization system for indoor service robots in low luminance and slippery indoor environment using afocal optical flow sensor based sensor fusion," *Sensors*, vol. 18, no. 1, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/1/171

[50] N. Gageik, M. Strohmeier, and S. Montenegro, "An autonomous uav with an optical flow sensor for positioning and navigation," *International Journal of Advanced Robotic Systems*, vol. 10, no. 10, p. 341, 2013. [Online]. Available: https://doi.org/10.5772/56813

[51] Z. Wei and K. Zhao, "Structural parameters calibration for binocular stereo vision sensors using a double-sphere target," *Sensors*, vol. 16, no. 7, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/7/1074

[52] Y.-H. Jin, K.-W. Ko, and W.-H. Lee, "An indoor location-based positioning system using stereo vision with the drone camera," *Mobile Information Systems*, vol. 2018, p. 5160543, Oct 2018. [Online]. Available: https://doi.org/10.1155/2018/5160543

[53] P. Poirier, "Ardupilot requirements," https://gitter.im/ArduPilot/ChibiOS?at=5c76d76e35c01307534a5d08, Feb 2019, (Accessed on 21/09/2021).

[54] A. Team, "Omnibus f4 pro," https://ardupilot.org/copter/docs/common-omnibusf4pro.html, 2021, (Accessed on 21/09/2021).

[55] B. Team, "Omnibus f4sd," https://www.banggood.com/Omnibus-F4SD-32K-Betaflight_3_2_

0-STM32-F405-Flight-Controller-OSD-5V-3A-BEC-30_5X30_5mm-p-1211677.html?cur_warehouse=CN&rmmds=search, 2021, (Accessed on 21/09/2021).

[56] R. Team, "X2 elf frame," https://www.rcworld.co.za/product_details.php?proid=298, 2021, (Accessed on 21/09/2021).

[57] B. Team, "Flywoo firefly hex nano frame," https://usa.banggood.com/Flywoo-Firefly-Hex-Nano-Analog-Spare-Part-90mm-Wheelbase-Hexacopter-Frame-Kit-for-RC-html?cur_warehouse=CN&rmmds=search, 2021, (Accessed on 21/09/2021).

[58] eCalc Team, "Ecalc maximize hover efficiency," https://www.ecalc.ch/, 2021, (Accessed on 26/09/2021).

[59] U. F. A. Administration, "Helicopter flying handbook." U.S Department of transportation, 2019, (Accessed on 14/09/2021).

[60] eCalc Team, "Ecalc maximize hover efficiency," https://www.youtube.com/watch?v=ZSjU8fW3U10&ab_channel=eCalc-gettherightdrive, Feb 2017, (Accessed on 25/09/2021).

[61] D. Birks, "Brushed dc motors and how to drive them," https://www.diodes.com/design/support/technical-articles/driving-brushed-dc-motors/, Sep 2020, (Accessed on 25/09/2021).

[62] A. Team, "Pwm,oneshot, and dshot protocol escs," https://ardupilot.org/copter/docs/common-brushless-escs.html#common-brushless-escs, 2017, (Accessed on 25/09/2021).

[63] O. Liang, "What is dshot," https://oscarliang.com/dshot/, July 2020, (Accessed on 25/09/2021).

[64] ——, "The best radio transmitter for fpv drones," https://oscarliang.com/radio-transmitter/#radio-receivers, June 2021, (Accessed on 25/09/2021).

[65] Large, "Lithium-ion polymer battery advantages and disadvantages," https://www.large.net/news/8ju43nv.html, Apr 2020, (Accessed on 25/09/2021).

[66] R. Triggs, "Lithium-ion vs lithium-polymer: What's the difference?" https://www.androidauthority.com/lithium-ion-vs-lithium-polymer-whats-the-difference-27608/, May 2021, (Accessed on 25/09/2021).

[67] B. Schneider, "A guide to understanding lipo batteries," https://rogershobbycenter.com/lipoguide, September 2021, (Accessed on 25/09/2021).

[68] B. Team, "Hglrc fd1103 8000kv brushless motor," https://boyztoyz.co.za/product/hglrc-fd1103-8000kv-brushless-motor/, Feb 2017, (Accessed on 25/09/2021).

[69] K. C. Kosmala and K. Kemmis, "Architechtural elements," https://buffaloarchitecture.org/wp-content/uploads/2015/08/AEPresentation-ArchitecturalElements.pdf, August 2015, (Accessed on 25/09/2021).

[70] R. Delport, "Connecting the esp-01 module to a bread-board and ftdi programmer," https://behind-the-scenes.net/connecting-the-esp8266-to-a-breadboard-and-ftdi-programmer/, June 2018, (Accessed on 25/09/2021).

[71] NodeMCU, "Nodemcu esp8266 flasher," https://github.com/nodemcu/nodemcu-flasher/blob/master/Win64/Release/ESP8266Flasher.exe, Sept 2015, (Accessed on 25/09/2021).

[72] A. Team, "Esp8266 wifi telemetry," https://ardupilot.org/copter/docs/common-esp8266-telemetry.html, 2021, (Accessed on 25/09/2021).

[73] S. Microelectronics, "Stm32f427xx stm32f429xx datasheet," https://www.st.com/content/ccc/resource/technical/document/datasheet/03/b4/b2/36/4c/72/49/29/DM00071990.pdf/files/DM00071990.pdf/jcr:content/translations/en.DM00071990.pdf, Jan 2018, (Accessed on 26/09/2021).

[74] Discussion, "Pixracer pmw3901 integration," https://discuss.ardupilot.org/t/integration-of-pixracer-with-optical-flow-pmw-3901/34864/29, 2018-2019, (Accessed on 26/09/2021).

[75] A. Team, "Setting up the build environment on windows10 using wsl1 or wsl2," https://ardupilot.org/dev/docs/building-setup-windows10.html#building-setup-windows10, 2021, (Accessed on 26/09/2021).

[76] ——, "Ardupilot project," https://github.com/ArduPilot/ardupilot, October 2021, (Accessed on 26/09/2021).

[77] Auturgy, "Ap hal chibos: fmuv4 add pixartflow to hwdef.dat #9754," https://github.com/ArduPilot/ardupilot/pull/9754/files, Nov 2018, (Accessed on 27/09/2021).

[78] P. Team, "mro pixracer," https://docs.px4.io/v1.12/en/flight_controller/pixracer.html, Feb 2021, (Accessed on 29/09/2021).

[79] P. Kantue and J. Pedro, "Real-time identification of faulty systems: Development of an aerial platform with emulated rotor faults," https://www.researchgate.net/publication/332409110_Real-Time_Identification_of_Faulty_Systems_Development_of_an_Aerial_Platform_with_Emulated_Rotor_Faults, April 2019, (Accessed on 27/09/2021).

[80] Mark, "Esp-01 esp8266 serial wifi module," https://kuongshun.en.made-in-china.com/product/kNxJlXyCMocS/China-Esp-01-Esp8266-Serial-WiFi-Module.html, 2021, (Accessed on 27/09/2021).

[81] R. Depot, "Dys elf-83mm micro drone - 4 in 1 10a esc (elf-f10a)," https://www.rcdepothobbies.com/product_p/elf-f10a.htm, 2021, (Accessed on 27/09/2021).

[82] G. Hobbies, "Frsky xm+ sbus mini receiver," https://www.goblinhobbies.co.za/frsky-xm-sbus-mini-receiver.html, 2021, (Accessed on 28/09/2021).

[83] R. W. Team, "Frsky taranis q x7s transmitter 2.4ghz," https://www.rcworld.co.za/mobile/product_details.php?proid=1418, 2021, (Accessed on 28/09/2021).

[84] B. Team, "Flow deck v2," https://store.bitcraze.io/products/flow-deck-v2, 2021, (Accessed on 28/09/2021).

[85] O. Team, "Proficnc/hex power brick mini," https://www.onedrone.com/store/index.php?route=product/product&product_id=3131, 2021, (Accessed on 28/09/2021).

[86] D. S. Team, "Onbo 25c 1100mah 2s lipo battery," https://radiosailing.net/products/25c-1100mah-2s-lipo-battery-w-xtc60-male-connector, 2021, (Accessed on 28/09/2021).

[87] F. T. Team, "Pixracer r15 mini flight controller with wifi module," https://www.flyingtech.co.uk/electronics/mini-px4-pixracer-r15-flight-controller-autopilot-wifi, 2021, (Accessed on 29/09/2021).

[88] B. Team, "Flow deck v2 schematic," https://wiki.bitcraze.io/_media/projects:crazyflie2:expansionboards:flow-deck-v2_rev-a_schematic.pdf, May 2018, (Accessed on 29/09/2021).

[89] ——, "Flow breakout," https://wiki.bitcraze.io/_media/breakout:flow_breakout_revb.pdf, June 2017, (Accessed on 29/09/2021).

[90] E. W. Team, "Esp8266 wifi module," https://www.electronicwings.com/sensors-modules/esp8266-wifi-module, Feb 2018, (Accessed on 29/09/2021).

[91] A. Team, "Mandatory hardware configuration," https://ardupilot.org/copter/docs/configuring-hardware.html, 2021, (Accessed on 1/10/2021).

[92] Painless360, "Taranis q x7 radio: Creating basic model types (quad, plane, v tail and wing)," https://www.youtube.com/watch?v=NIR85KOqIAo&t=652s&ab_channel=Painless360, Apr 2017, (Accessed on 1/10/2021).

[93] A. Team, "Rc input channel mapping (rcmap)," https://ardupilot.org/copter/docs/common-rcmap.html, 2021, (Accessed on 1/10/2021).

[94] F. Team, "Xm plus manual," https://www.frsky-rc.com/xm-plus-mini-sbus-non-telemetry-full-range/, Aug 2017, (Accessed on 1/10/2021).

[95] S. M. team, "Vl53l1x time-of-flight sensor," https://www.st.com/resource/en/datasheet/vl53l1x.pdf, October 2021, (Accessed on 11/05/2021).

[96] A. Team, "Tuning process instructions," https://ardupilot.org/copter/docs/tuning-process-instructions.html#tuning-process-instructions, 2021, (Accessed on 2/10/2021).

[97] ——, "Optical flow sensor testing and setup," https://ardupilot.org/copter/docs/common-optical-flow-sensor-setup.html, Feb 2017, (Accessed on 5/10/2021).

[98] ——, "Rangefinders," https://ardupilot.org/copter/docs/common-rangefinder-landingpage.html, 2021, (Accessed on 5/10/2021).

[99] ——, "Circle mode," https://ardupilot.org/copter/docs/circle-mode.html, 2021, (Accessed on 06/11/2021).

[100] ——, "Flight mode," https://ArduPilot.org/copter/docs/flight-modes.html, 2021, (Accessed on 06/11/2021).

Appendix A

Supporting Data

A.1 Component Choice Tables

Component	Weight	Price	Link
Motors	17,5		Motor
Frame	15	105	Rod
FC	8	500	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	14	173	Bat
Props	6	110	Prop
ESC\Mosfet Board	2	180	Mos
Arduino + Transceiver	11	45	Telem
Wires + Misc	5	31	5V
Total	83,5	1444	

Table A.1: System 1 (Affordable Hexacopter) - Components

Performance						
Hover Throttle	All-up Weight	Efficiency@Max	Mixed Flight time	Bat Load	Thrust-Weight	Specific Thrust
76%	83.5g	30%	3.3min	25C	1.3:1	3.48 g/W

Table A.2: System 1 Performance

Component	Weight	Price	Link
Motors	12		Motor
Frame	10	105	Rod
FC	8	500	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	14	173	Bat
Props	4	55	Prop
ESC\Mosfet Board	2	180	Mos
Arduino + Transceiver	11	45	Telem
Wires + Misc	5	31	5V
Total	71	1389	

Table A.3: System 2 (Affordable Quadcopter) - Components

System 2 had no performance metrics as it was unable to hover

Component	Weight	Price	Link
Motors	28	840	Motor
Frame	10	105	Rod
FC	8	500	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	30	235	Bat
Props	4	55	Prop
ESC\Mosfet Board	24	360	ESC
Arduino + Transceiver	11	45	Telem
Wires + Misc	5		
Total	125	2440	

Table A.4: System 3 (Affordable Brushless Quadcopter) - Components

Performance						
Hover Throttle	All-up Weight	Efficiency@Max	Mixed Flight time	Bat Load	Thrust-Weight	Specific Thrust
45%	125g	53.5%	6.1min	42.3C	2.6:1	4.57 g/W

Table A.5: System 3 Performance

Component	Weight	Price	Link
Motors	17,5		Motor
Frame	15	105	Rod
FC	11	2700	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	14	173	Bat
Props	6	110	Prop
ESC\Mosfet Board	2	180	Mos
Wires + Misc	5	31	5V
Total	75.5	3599	

Table A.6: System 4 (Expensive Hexacopter) - Components

Performance						
Hover Throttle	All-up Weight	Efficiency@Max	Mixed Flight time	Bat Load	Thrust-Weight	Specific Thrust
70%	76g	30%	4min	25C	1.5:1	3.76 g/W

Table A.7: System 4 Performance

Component	Weight	Price	Link
Motors	12		Motor
Frame	10	105	Rod
FC	11	2700	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	14	173	Bat
Props	4	55	Prop
ESC\Mosfet Board	2	180	Mos
Wires + Misc	5	31	5V
Total	63	3544	

Table A.8: System 5 (Expensive Quadcopter) - Components

Performance						
Hover Throttle	All-up Weight	Efficiency@Max	Mixed Flight time	Bat Load	Thrust-Weight	Specific Thrust
84%	63g	30.6%	4min	17C	1.2:1	3.15 g/W

Table A.9: System 5 Performance

Component	Weight	Price	Link
Motors	28	840	Motor
Frame	10	105	Rod
FC	11	2700	FC
Receiver	3	300	Rec
Flow Deck	2		Flow
Battery	30	235	Bat
Props	4	55	Prop
ESC\Mosfet Board	24	360	ESC
Wires + Misc	5		
Total	117	4595	

Table A.10: System 6 (Expensive Brushless Quadcopter) - Components

Performance						
Hover Throttle	All-up Weight	Efficiency@Max	Mixed Flight time	Bat Load	Thrust-Weight	Specific Thrust
43%	117g	53.5%	6.7min	42.3C	2.8:1	4.67 g/W

Table A.11: System 6 Performance

Appendix B

Ethics application form