University of Glasgow

Department of Mechanical Engineering

Mechatronics Team Project 4

Design and Development of a UAV Tracking System

PTA

Kyle Arnott

Mikulas Cebecauer

Alexandre Cohen

Patrik Paulini

Filip Voudouris

Jack Wood

Academic Year 2018-2019

Contents

1 Acknowledgements **5**

2 Introduction **6**

3 Background **7**

3.1 Monopulse Radar 7

3.2 GPS Tracking 7

3.3 Optical Tracking 8

4 Aim **9**

5 Design **11**

5.1 Overview 11

5.2 Hardware 11

5.2.1 Mechanical Assembly Module 11

5.2.2 Optics Module 13

5.2.3 Detector Module 15

5.2.4 Lock-In Amplifier Module 16

5.2.5 Light Beacon Module 22

5.2.6 Control Module 24

5.2.7 Power Supply Module 24

5.2.8 Antenna Module 25

5.3 Software 26

5.3.1 GPS Program 26

5.3.2 Detector Sampling 27

5.3.3 Control 28

6 Implementation **30**

6.1 Overview 30

6.2 Hardware 30

6.2.1 Mechanical Assembly Module 30

6.2.2 Optics Module 32

6.2.3 Detector Module 33

6.2.4 Lock-In Amplifier Module 35

6.2.5 Light Beacon Module 45

6.2.6 Control Board Module 50

6.2.7 Power Supply Module 52

6.2.8 Antenna Module 52

6.3 Software 53

6.3.1 Software Systems Overview 53

6.3.2 Control Module Micro-controller 53

6.3.3 Lock-in Amplifier Module Micro-controller 60

7 Testing and Results **65**

7.1 Functionality Tests 65

7.1.1 Optical Range Testing 65

7.1.2 Optical Detector Dynamic and Static Characterisation 66

7.1.3 Lock-In Amplifier Performance Testing 70

7.1.4 Antenna Range Testing 70

7.1.5 Antenna Switching Testing 70

7.1.6 Mechanical Assembly Azimuth/Elevation Controller Testing 71

7.2 Field Tests 71

7.2.1 Short Range Tracking Field Test 72

7.2.2 Long Range Tracking Field Test 72

7.2.3 GPS Tracking Test 73

8 Conclusions **75**

8.1 Further Improvements 75

9 Bill of Materials **77**

10 Bibliography **79**

11 Appendices **81**

11.1 Software 81

11.2 Data 100

1 Acknowledgements

We would like to thank the following people for their support in completing this project:

- Dr. D. Muir
- Prof. J. Weaver
- Andrew Philips and all technicians in Rankine 709
- Everyone in the level 3 mechanical workshop
- Police Scotland
- Glasgow Riverside Museum
- Dr. D. Anderson

Without your help this project would not have been possible.

2 Introduction

This project was part of the University of Glasgow's Mechatronics engineering fourth year MEng course. This was a team project and thus was comprised of six fourth year Mechatronics students. The aims of this course were[14]:

1. to accumulate experience of working in a team focused on delivering specific objectives and requiring to design, plan, implement and test a challenging technical project;
2. to develop technical skills such as software writing, hardware design, construction and testing;
3. to make practical use of management and project planning skills;
4. to engage in a technically significant project which will require the full team in order to complete in time;
5. to work to a design brief and deliver a product which meets challenging specifications within a budget.

This project was sponsored by Police Scotland as part of an Unmanned Aerial Vehicle (UAV) development program. The supplied specification was:

"WiFi high gain tracking system. WiFi does not go to the 500m required by the application- so a high gain antenna needs to be developed. Commercial high gain antennas should suffice but they will need active tracking in order to fixate on the UAV".

From this it was inferred that we were to design and build a system capable of pointing a highly directional WiFi antenna at a UAV. This is in order to permit long range WiFi communications for the purpose of Police Scotland's search and rescue capabilities. This report details the development of this system and the results.

3 Background

There are always many different solutions to each problem, and this is especially applicable in engineering. In this section, the preliminary research will be discussed and the reasoning behind the approach that was chosen.

The team found that there were three possible methods to track the drone:

3.1 Monopulse Radar

Monopulse radar is one of the most prominent methods of target tracking. It consists of emitting a radio frequency (RF) pulse and determining the position of the target based on the strength of the reflected pulse. This method originally used conical scan tracking, which involves rotating the antenna and measuring the change in signal strength when the the antenna is going in one direction vs going in the other. This has many flaws such as putting large wear and tear on the mechanical system and the scan rate being limited by the scanning mechanism [8]. Therefore, an alternate method was developed which consists of having two variations of the signal (different polarisation) being emitted in slightly different directions. This is to have the maximum strength of each signal at a slight offset. Then, by comparing reflections, we can determine the direction that the target is moving [15].

As effective as this method is, there were several reasons it was not applicable to this project:

1. The system is based on an array of complex antennae that require significant RF design expertise and careful tuning[15]
2. This is a computationally heavy solution which would require highly experienced programmers
3. The solution is costly and the team had no previous exposure to this type of technology, making it a high risk solution

3.2 GPS Tracking

GPS (Global Positioning System) tracking is one of the most used technologies in drone tracking. It relies on standard GPS technology which makes use of satellites to triangulate the position of a

GPS emitter. While the benefits of this approach are its simplicity and ease of implementation, it comes with major drawbacks in terms of our specific goal. For this reason, GPS tracking shall be implemented as a back-up measure, to ensure a robust tracking system.

It was not chosen as a main method of tracking for the following reasons:

1. It requires GPS data from the drone to be transmitted through the WiFi connection. Therefore, if the connection is broken, no GPS data will be sent back and the ground station has no method of locating the drone until the connection is re-established
2. It was assumed that the system could be used in the Scottish Highlands where GPS is not guaranteed
3. GPS requires the drone to have nothing above it, as it might interfere with the accuracy of the position data

3.3 Optical Tracking

Optical tracking is the use of opto-electronics to identify a target by comparison against it's surroundings. For example if the target is uncooperative, it is illuminated with a bright light and the reflection is captured and used to infer positional data - similar to monopulse tracking mentioned earlier only using visible light rather than RF signals. For our application, the target is cooperative. This means that a light can be mounted on the drone, and the optical detection can use this as it's target. By employing four optical detectors that work in this way, the targets position and direction can be determined.

To accomplish this in the presence of noise, the target light source (the beacon) is given special characteristics such that it's signal can be extracted from the noise floor. Synchronous detection makes it possible to isolate an extremely small signal below the noise floor by comparing the frequency of the target signal against a known target frequency. The target signal is then down-mixed to a far lower frequency, lowering both signal bandwidth and allowing increased noise rejection. This approach offers an extremely robust method of tracking, but requires precise and sensitive analogue circuitry. Through research and recommendations from Professor Weaver, this approach was chosen as the predominant tracking method for our system.

4 Aim

From the description of the project, the user requirements and preliminary research, a wide variety of goals can be extracted. They fit into the development of two main systems which complement each other: The optical detection system and the GPS system. After further research into these two systems, specifications for the whole project were outlined and the success of the project was based on a comparison between the final project outcome and these specifications. A full specification of the system was presented and agreed upon with the academic supervisor. This full specification is given below:

Antennas:

- Use a directional WiFi antenna with sufficient gain to transmit and receive high-bandwidth communication over 500m with a narrow beam width
- Use a directional antenna in combination with an omni-directional antenna, switching between the two at short distance

Drone detection:

- Develop an optical detector capable of detecting modulated light, with sufficient resolution to robustly track the light source as it moves using synchronous detection
- Implement a redundant GPS system in case of optical photodiode detector implementation failure

Computing system:

- Low latency real-time software and microcontroller hardware delivering detection processing and subsequent control of the drives of the antenna gimbal assembly

Mechanical assembly:

- A mechanised azimuth and elevation rotational assembly is to be delivered, capable of connecting with the police vehicle and rotating 360° in the azimuth axis and reaching 100° in elevation angle

- A rotating platform able to reach angular velocity of 1 rad/s, with appropriate drives and control systems
- The weight of the system including antennas is to be up to 7.5kg with dimensions fitting into a volume of $1m^3$

5 Design

5.1 Overview

To simplify development, the system was broken down into hardware and software modules. This allowed concurrent design and implementation of multiple modules. An overall system diagram was created outlining all modules and interconnection among them (Figure 1). Overall, a budget driven design was adopted across all of these modules while still aiming to maximise performance. Note that only the final design is presented here and it was gradually refined in an iterative process by implementing and testing intermediate designs of each of the modules.

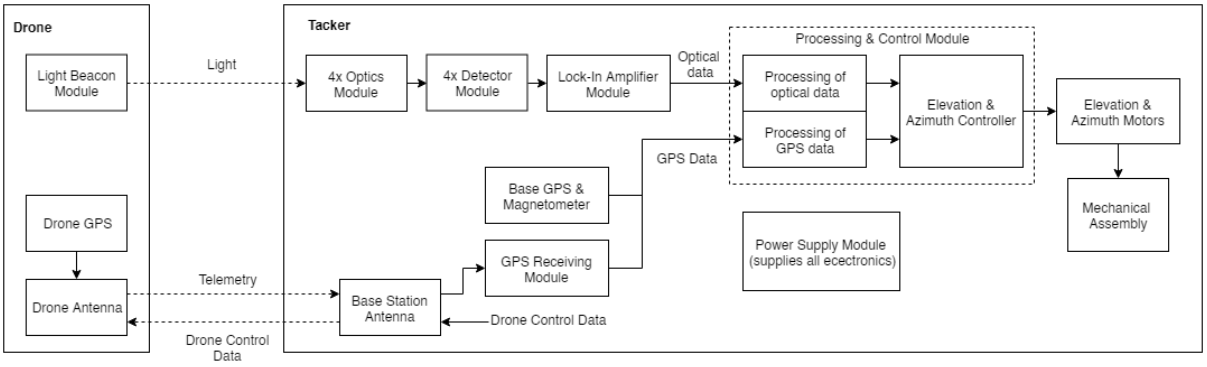


Figure 1: System block diagram, showing interconnections among subsystems.

5.2 Hardware

5.2.1 Mechanical Assembly Module

The main requirements of the mechanical assembly can be seen in the *Aims* section of the report. The design was created around this.

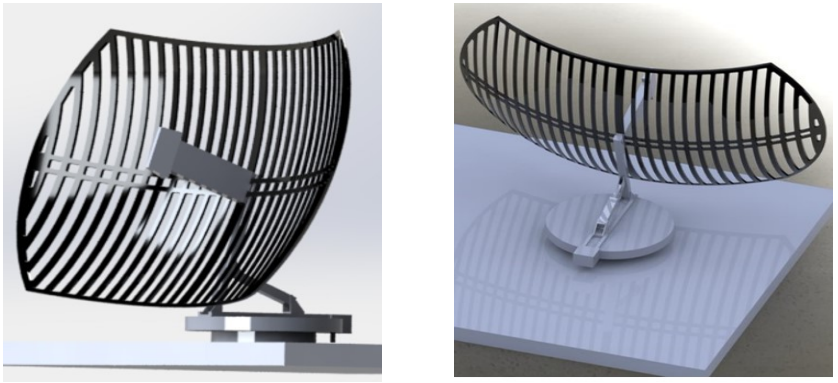


Figure 2: Solidworks model of rotation in azimuth

As can be seen from the images, the system has two platforms, one on top of the other. The bottom platform is a thrust bearing and the top platform is fastened to the bearing, thus allowing it to rotate through 360°. As a rotating mechanism, a gear will be fitted around the thrust bearing and a motor will be attached at the edge of the top platform. A belt will connect the two, thus allowing the rig to rotate and ensuring that the cables from the motor do not become twisted.

As the top platform needs to rotate at $1rad/s$, the gear attached to the platform must also rotate at that speed. A gear with a diameter of 30 cm with a pinion of 2cm was chosen as it would allow for a high torque output thus allowing the team to purchase a cheaper motor for less torque. Thus using:

$$\frac{\omega_{in}}{\omega_{out}} = \frac{d_{out}}{d_{in}} \tag{1}$$

It can be calculated that the RPM required by the motor is $110RPM$ ($200RPM$ was chosen to provide suitable overhead).

Although the previous image allows for a easy explanation of how the system rotates in azimuth, it does not have an accurate representation of the elevation drive system. The elevation system must be able to adjust the elevation angle of the lens assembly and the antenna. The design can be seen in the Figure 3.

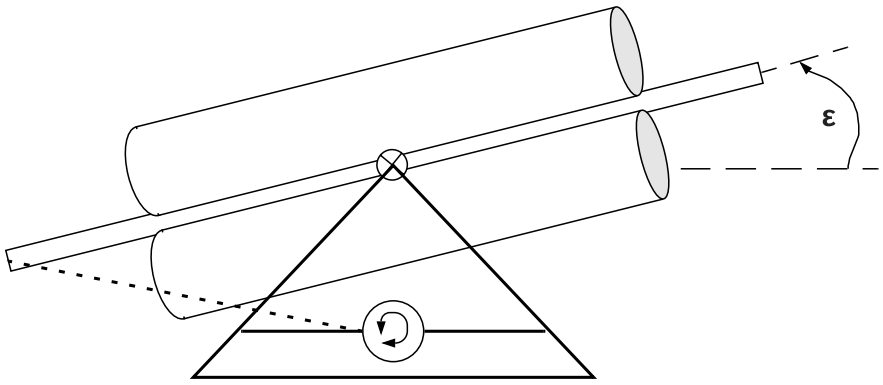


Figure 3: Elevation Drive Diagram.

The main concept behind this design is to have a motor that is attached to the end of the lens system via a string and by changing the length of the string, it is possible to control the elevation of the lens and antenna system.

5.2.2 Optics Module

From the beginning it was clear that the detectors will require optical peripherals to improve signal capture. These optics include lenses that concentrate the power delivered by the beacon on to the detectors sensor and a tube that limits field of view (FOV). The function of the lenses is to concentrate the light from area of the lens to the sensor. To accomplish this, a plano-convex or double convex lens is required. It is also desirable to minimise the size of the optics in order to lessen torque loading. This can be achieved by using a lens with minimal focal length.

These lenses can be made from lighter materials and theoretically can provide same optical characteristics as conventional glass lenses. The biggest advantage of the fresnel lenses is that they can be made much thinner than their convex alternatives. Thus, usage of fresnel lenses in this project minimises the weight of the optical modules and therefore reduce the torque specifications for the driving motors.

The FOV of the detectors θ_{view} was designed to be equal to the beam width of the long range WiFi antenna which defines the required accuracy limits for the detector. To limit the FOV, the approach angle of light θ_{app} to the lens needs to be limited. Figure 4 illustrates the design for limiting the field

of view of the detector.

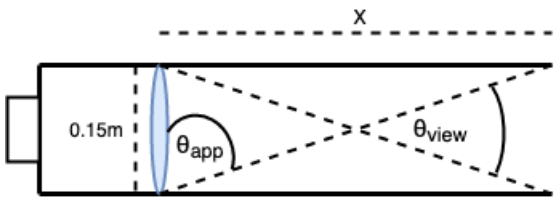


Figure 4: Optic Diagram.

In Figure 4, θ_{app} represents the approach angle, θ_{view} represents the field of view angle of the optic, and X represents the length of the tube in front of the lens. The length of the tube in front of the lens can be determined through geometry. It can be seen that:

$$180 = 2\theta_{app} + \theta_{view} \tag{2}$$

Note that to determine θ_{view}, it must be considered that the final system consists of four detectors that together localise the drone. This means that the actual field of view of the system is defined by the overlap of two of the detectors FOV. Figure 5 illustrates the expected alignment of the detector optics, where θ_{final} represents the final field of view that is shared by both detectors.

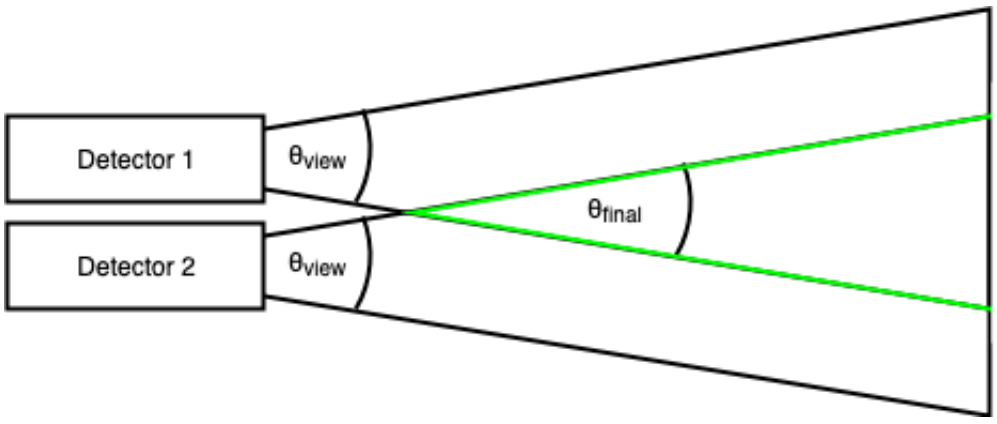


Figure 5: Illustration of the alignment and interaction between two detector optics.

As can be seen above the design expects the detectors to be mounted parallel to each other. Assuming a $\theta_{final} = 20°$ derived from the beam width of the directional antenna, the value for θ_{view} can be found. Finally, the value for θ_{app} and X can be then computed using trigonometry. Evaluating this gives $X = 850mm$ which is the required tube extension length in front of the lenses. This was

considered too much as it dramatically increases moment of inertia of the optics and therefore the torque specifications for the driving motors.

To reduce the length of the FOV limiting optics, improvements in the design had to be made. The optics limiting the FOV of each detector were mounted at an angle $\theta_{detector}$ shown in Figure 6. This allowed for larger FOV of each detector and therefore allowed for shorter optics. This was implemented by firstly setting a 70cm length specification for the optical setup. From this, a field of view of single optic was determined as $\theta_{view} \approx 30°$. This set the final length of the optics to 68cm. Finally, the mounting angle was determined experimentally. The process of determining of the mounting angle is discussed in the implementation section of this report.

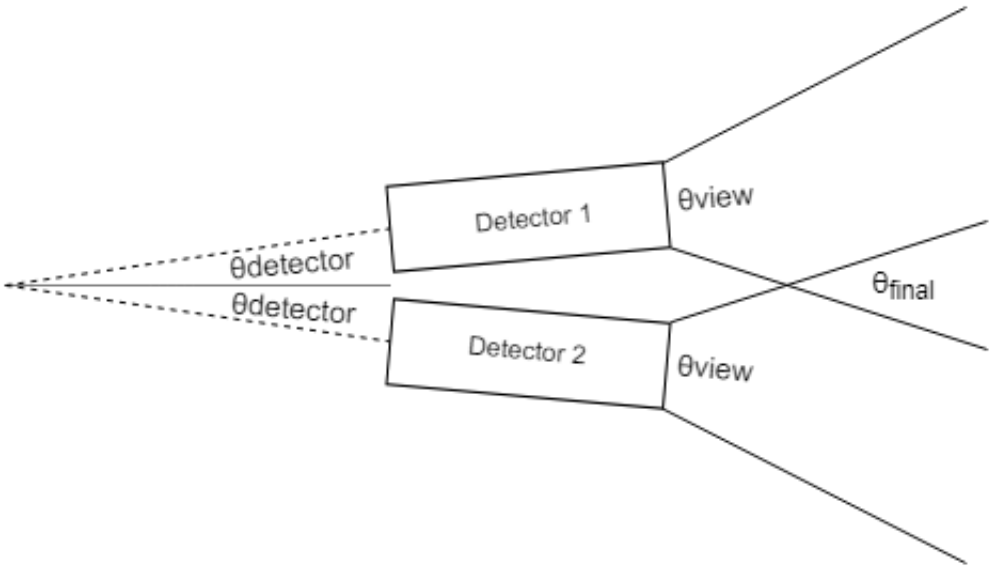


Figure 6: Illustration of additional $\theta_{detector}$ for detector alignment.

5.2.3 Detector Module

The detector module is responsible for capturing the signal emitted by the light beacon (modulated at ω_s), pre-conditioning it and supplying an input to the lock-in amplifier. The requirements in designing this module were to be very low noise and provide effective filtering. The designed architecture consists of a photodiode operating in photoconductive as part of a transimpedence amplifier (outlined in Philip Hobbs' paper [5]), and then filtering the output. The Philip Hobbs pre-amplification circuit effectively minimises any component noise and maintains a high SNR without sacrificing bandwidth. On the output of this transimpedence amplifier a band-pass filter was designed to remove any unwanted DC offsets, thus provide AC coupling. This ensure only the frequency component of

ω_s which corresponds to the light modulation frequency is passed to the lock-in amplifier. After ω_s is quantified, it is used to drive the design as it defines the pass band of the filter on the output.

5.2.4 Lock-In Amplifier Module

The principal task of this module is to perform synchronous demodulation of an input modulated voltage signal at specific frequency buried in noise, preconditioned by the detector module. Architecture of a typical lock-in amplifier was used as a starting point of the design process and the working principle was adapted for our application.

Typically, commercial lock-in amplifiers perform dual-phase demodulation where a sinusoidal response $V_s(t)$ from a device under test (with frequency ω_s, phase θ_s and magnitude R), is mixed with a sinusoidal reference signal $V_r(t)$ (with frequency ω_r) to create $Z(t)$. The reference signal is split according to phase-shift into a $\theta_r = 0°$ in-phase component and a $\theta_r = 90°$ quadrature component reference signal before mixing. Mixing of both reference signals and the input signal is done independently.

$$V_s(t) = Rcos(\omega_s t + \theta_s) = \frac{R}{2}e^{i(\omega_s t+\theta_s)} + \frac{R}{2}e^{-i(\omega_s t+\theta_s)} \tag{3}$$

$$V_r(t) = e^{-i(\omega_r t+\theta_r)} \tag{4}$$

$$Z(t) = V_r(t) * V_s(t) = \frac{R}{2}e^{i((\omega_s-\omega_r)t+\theta_s-\theta_r)} + \frac{R}{2}e^{-i((\omega_s+\omega_r)t+\theta_s+\theta_r)} \tag{5}$$

Following this, amplification by G and a low pass filter $LP\langle\rangle$ is applied such that down-mixed signals $O(t)$ are extracted and their bandwidths are decreased.

$$O(t) = G * LP\langle Z(t)\rangle = \frac{GR}{2}e^{i((\omega_s-\omega_r)t+\theta_s-\theta_r)} \tag{6}$$

By doing this, in-phase and quadrature components of the signal in cartesian coordinates can be recorded and by transformation into polar coordinates both the phase and the magnitude of the input signal can be reconstructed (Figure 7). Ideally, when the device under test response frequency and

the mixing frequency are equal $\omega_r = \omega_s$ and when the phase of the response input is equal to the phase of the reference signal $\theta_s = \theta_r$ a DC signal with amplified magnitude of the input signal is output, $Y(t)$. The output now has low bandwidth and overcomes problems with measurement noise.[7][11]

$$Y(t) = \frac{GR}{2} \tag{7}$$

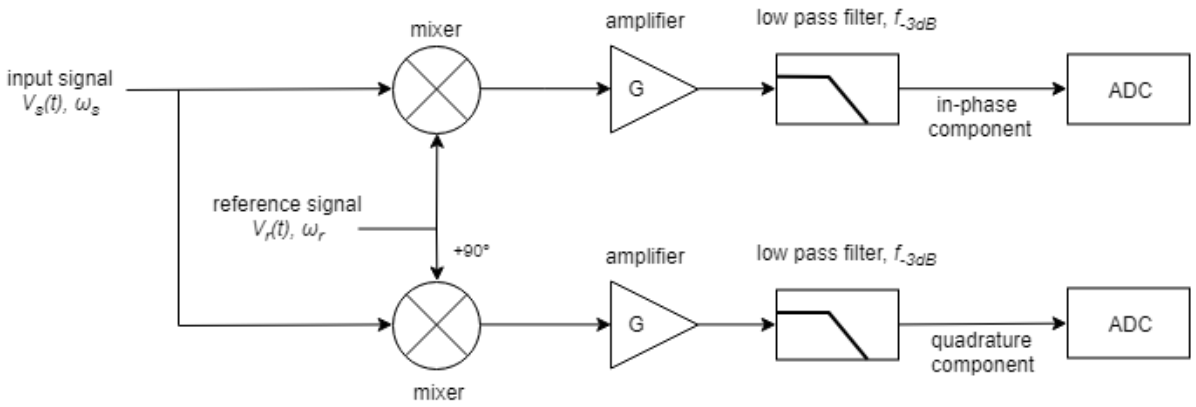


Figure 7: Block diagram of typical lock-in amplifier architecture.

For our application, this process contains a significant hurdle. A single reference signal is used to excite the device under test and is used as a reference signal for the mixing process. For the application of target tracking two separate reference signals are required - one for light beacon modulation on the drone (the target) and a second for the mixing reference signal (the tracker). Practically, the reference signals cannot be synchronised using external communication resulting in a finite frequency difference. This issue is worsened by frequency drift and noise. Thus the desired conditions, $\omega_r = \omega_s$ and $\theta_s = \theta_r$, cannot be achieved.

To address this issue, the typical architecture was altered. Instead of relying on equalising the reference frequency ω_r and input response frequency ω_s to obtain a DC output proportional to the magnitude of the AC input, a small static frequency difference between the two $\Delta\omega$ is created by choosing slightly different values for ω_r and ω_s. This gives an output signal proportional to the magnitude of the input, of frequency $\Delta\omega$. In practice, both the input and reference frequencies are susceptible to some error due to both drift and noise. These small errors are apparent in $\Delta\omega$, in the order of mHz.

$$\Delta\omega = \omega_s - \omega_r \pm \omega_{noise} \pm \omega_{drift} \tag{8}$$

Furthermore, the concept of the in-phase component and quadrature component of the reference signal for dual-phase demodulation is replaced by the concept of splitting the input response signal with frequency ω_s into two and phase shifting one of the input signals by 180°. Both of these signals are mixed with the same reference signal with frequency ω_r and are treated as differential inputs for differential amplification by G and low pass filtering $LP\langle\rangle$. Under these conditions the output $O(t)$ behaves as a signal with amplified magnitude of the input signal modulated by a sine wave at a frequency of $\Delta\omega$ and with phase $-\theta_r$.

$$O(t) = G * (LP\langle Z_0(t)\rangle - LP\langle Z_{180}(t)\rangle) = \frac{GR}{2}e^{i(\Delta\omega t + 0^\circ - \theta_r)} - \frac{GR}{2}e^{i(\Delta\omega t + 180^\circ - \theta_r)} \tag{9}$$

$$= \frac{GR}{2}e^{i\Delta\omega t}(e^{-i\theta_r} + e^{-i\theta_r}) \tag{10}$$

$$= GRe^{i(\Delta\omega t - \theta_r)} \tag{11}$$

Thus it can be seen that the equivalence condition of $\theta_s = \theta_r$ becomes unimportant as long as θ_r remains constant. By sampling this output at a higher rate (f_s) than the down-mixed frequency of the signal, and by detecting the maximum and minimum points of the signal and their time stamps, the magnitude of the signal GR as well as the frequency $\Delta\omega$ can be measured. These operations are digital processing routines performed after digitisation. Any small variations in the frequency on the output due to ω_{noise}, ω_{drift} or frequency synchronisation issues are overcome by finding the magnitude from at least one full period of the output signal sampled at a high rate.

The next issue addressed in our design was that it is required to perform the lock-in amplification process on four optical detector modules concurrently to obtain real-time data from each detector. As a cost driven design approach was adopted, manufacturing four equal lock-in amplifier modules and interfacing each individually was not viable. Instead, the architecture of the system was altered by adding a differential multiplexer stage onto the the input of the lock-in, capable of selecting one of the four differential inputs at any given time. This posed further constraints on the design, due to the finite response time t_r of the lock-in amplifier itself. Multiplexing frequency ω_m is introduced and must be larger or equal to $\Delta\omega$ to ensure that on the output, after t_r, at least one full period of multiplexed and conditioned signal can be sampled and from it the magnitude and frequency can

be inferred. Furthermore, it is important to quantify the response time t_r such that it is possible to identify the data from each detector after the digitisation. In general, it is desirable to achieve a low response time t_r since the reference input to the controller derived from the data is estimated to be supplied at a rate of $4t_r$ when using 4 detectors. This introduces lag which is to be minimised for smoother tracking (Figure 8).

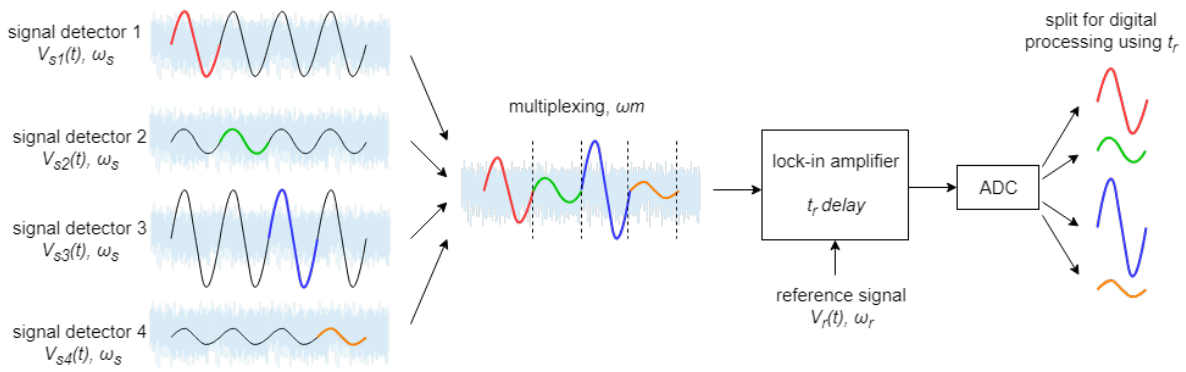


Figure 8: Diagram showing multiplexing of channels, timing and reconstruction of noise free signals after digitisation.

An additional design concern, which is critical for operation at large ranges where the signal strength of the beacon is small, is that an appropriate signal to noise ratio *SNR* is required and thus the noise of the system is to be minimised. This lead to the adoption of a noise budget. The frequency spectrum of the noise floor of outdoor ambient sunlight was measured with the detector module and based on this the design of the lock-in amplifier was optimised. At low frequencies 0Hz - 30kHz the noise floor is dominated by $1/f$ noise which corresponds to shot-noise and at larger frequencies 30kHz - 1MHz white noise corresponding to johnson-noise becomes dominant (Figure 9, Figure 10). Under laboratory lighting conditions, additional noise is present in the frequency spectrum due to interference from other instruments and systems, however similar trend of $1/f$ noise dominating at low frequencies and white noise at higher frequencies is observed (Figure 11). From these measurements it is clear that the lowest points on the noise floor are in the white noise region, measured to be on average -77dBV over a span of 1MHz. [10]

Two major methods of noise reduction are apparent from this. Firstly, by performing the demodulation with a mixing (reference) frequency in a region with minimum noise, the overall *SNR* can be increased, as this portion of the signals will be down-mixed. Therefore the reference signal with frequency ω_r is to be in the white noise region above 50kHz. Secondly, because noise is quantified in terms of power spectral density with units $nV/\sqrt{Hz}$ the noise present in the down-mixed the signal

which is filtered, is dependent on the bandwidth defined by the cut-off frequency f_{-3dB} of the applied lowpass filter $LP\langle\rangle$. Thus, by designing f_{-3dB} to be small, noise is reduced and SNR is increased at the expense of response time t_r of the system which increases as f_{-3dB} gets larger since the lowpass filter time constant τ dominates the system. Thus it is important to design f_{-3dB} appropriately.

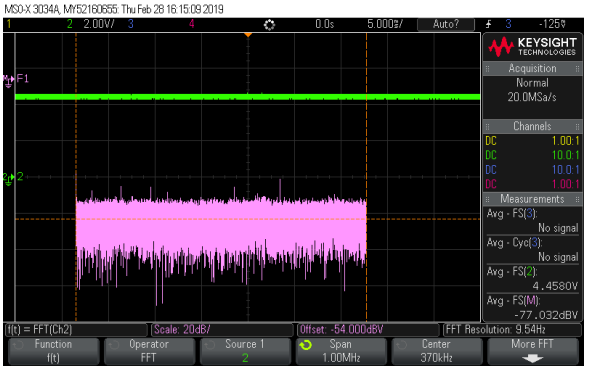


Figure 9: Frequency spectrum of ambient outside lighting noise floor. Range: 0Hz - 1MHz, Centre: 370kHz, Scale: 20dB/, Offset: -54dBV

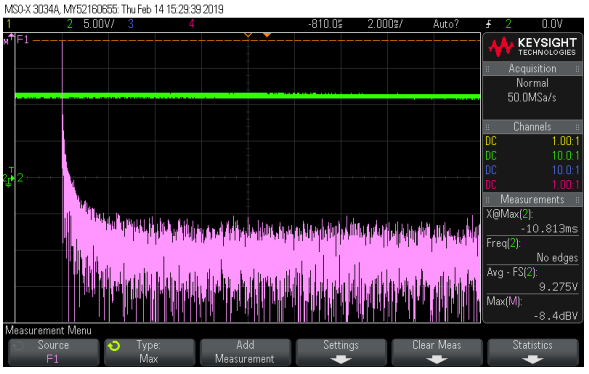


Figure 10: Zoomed in frequency spectrum of ambient outside lighting noise floor. Range: 0Hz - 200kHz, Centre: 80kHz, Scale: 10dB/, Offset: -46dBV

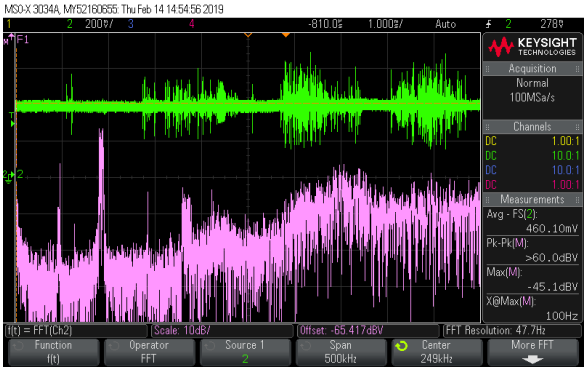


Figure 11: Zoomed in frequency spectrum of ambient laboratory lighting noise floor. Range: 0Hz - 500kHz, Centre: 250kHz, Scale: 10dB/, Offset: -65.417dBV

The final requirement of the system is to be able to work at small and large ranges. Thus the lock-in amplifier was designed such that it has wide input dynamic range. The gain G which determines the signal magnitude on the output had to be determined, to ensure that the signal is in the range of the ADC and does not saturate it. However, a single value of G does not provide the required dynamic range so the architecture of the differential input amplifier has to implement multiple gains. This is achieved by multiplexing gain circuitry so that at a specific range the specific G is selected. Gains have been selected based on measurements of the resolution of the lock-in amplifier defined at the

limit $SNR = 1$, the resolution of the ADC and the range of the ADC during implementation. Gains are switched to lower gain or a higher gain in real-time during digital processing when the signal magnitude at specific gain G crosses above or under predefined thresholds of 0.9 and 0.1 of ADC input range respectively.

Hence, the following main design principles for the Lock-In were identified:

- Pick a frequency difference $\Delta\omega$, in the order of tens of Hz. Depending on the precision of the devices used, make it larger than ω_{noise}, ω_{drift} and synchronisation frequency difference, so that it is dominant.
- Pick a sampling rate of the ADC f_s to be more than one order of magnitude higher than the frequency difference $\Delta\omega$ such that magnitude and frequency can be inferred from digitised data.
- Set the beacon frequency ω_s and the reference frequency ω_r such that they satisfy $\Delta\omega$. The higher the better, because of a lower ambient noise floor. Note that the required bandwidth of the lock-in amplifier input before down-mixing is set by higher of the two: ω_s or ω_r.
- Design the lowpass filter $LP\langle\rangle$ such that its f_{-3dB} is greater than $\Delta\omega$. Additionally, make f_{-3dB} small to improve SNR however make sure that filter time constant τ, which dominates the lock-in response time t_r, is in order of milliseconds so that tracking lag is minimised.
- Ensure the multiplexing frequency ω_m is larger or equal to $\Delta\omega$ to ensure that on the output, after t_r, at least one full period of multiplexed and conditioned signal can be captured and from it the magnitude and frequency can be inferred.
- Select multiple gains G such that wide input dynamic range is guaranteed by measuring the resolution of the lock-in amplifier defined at the limit $SNR = 1$, the resolution of the ADC and the input voltage range of the ADC during implementation.

Using the constraints and requirements on parameters appropriate values were selected for each parameter empirically: $\Delta\omega = 30Hz$, $\omega_s = 250kHz$, $\omega_r = 250.03kHz$, $f_{-3dB} = 40Hz$, $\omega_m = 20Hz$. Gain G was selected during implementation due to further practical constraints of a real (non-ideal) amplifier, and based on overall resolution measurements. Thus the main system architecture was established (Figure 12) along with its main parameters. Using these parameters the system was practically implemented.

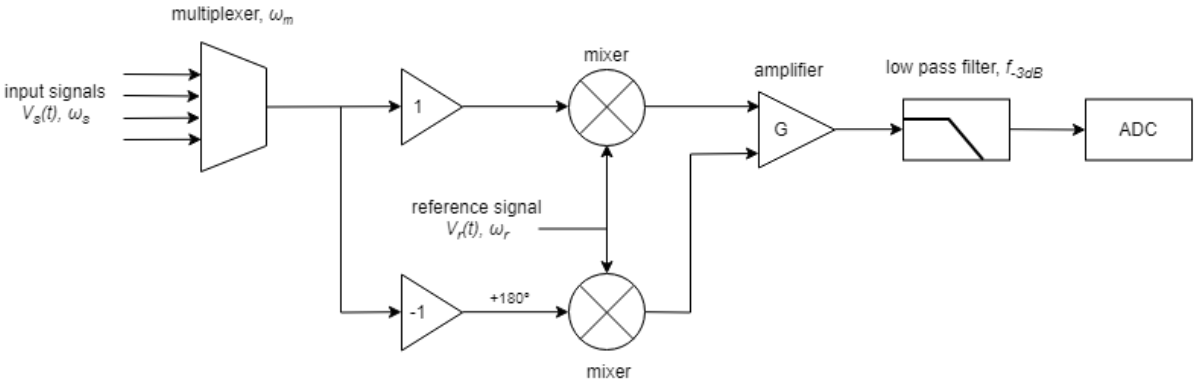


Figure 12: Block diagram of designed lock-in amplifier architecture.

5.2.5 Light Beacon Module

A fundamental requirement of the project is that the modulated light from the beacon will be recognisable over the specified distance of 500 meters to still be detectable by the sensor. To assure this, it is necessary to create a mathematical model that describes the relationship between the light intensity and the corresponding signal that will be detected. This will provide the specifications for the beacon light intensity, the sensitivity of the detector modules, and the size of the lens peripherals.

A candela Cd is the luminous intensity of a source that emits monochromatic radiation of frequency $540*10^{12}$Hz has a radiant intensity in a given direction of 1/683 watt per steradian [12]. This definition can be used directly since this project will use green light of frequency around $540*10^{12}$ hertz. A steradian Ω is defined as the solid angle subtended at the centre of a unit sphere by a unit area on its surface [13].

$$\Omega = \frac{A}{r^2} \tag{12}$$

This looks very similar to inverse square law that defines dissipation of light in atmosphere over distance. In fact, candela can therefore be defined as:

$$Cd = \frac{\frac{P}{683}}{\frac{A}{r^2}} \tag{13}$$

In this equation P represents the power delivered to certain area A over a distance r. If we solve for P

we get the following equation:

$$P = \frac{I(Cd) * 683 * A}{r^2} \tag{14}$$

In the above equation, $I(Cd)$ denotes the intensity of light in units of candelas. However, in general the specifications for LED's are given in lumens, Lm. The relationship between a candela and a lumen is defined below:

$$I(Cd) = \frac{I(Lm) * r^2}{A} \tag{15}$$

Similiarly to the previous equation, $I(Lm)$ here denotes the intensity of light but in units of lumens. In this case it is the area that is covered by the light in specific distance. This area can be defined by following expression.

$$A = \pi(d * tan(\frac{\theta_{apex}}{2}))^2 \tag{16}$$

When equations 15 and 16 are substituted into equation 14 we obtain flowing expression:

$$P = \frac{I(Lm) * 683 * A_{lens}}{\pi(d * tan(\frac{\theta_{apex}}{2}))^2} \tag{17}$$

This expression defines the power delivered by a light of intensity I in lumens, from light source with apex angle θ_{apex} over a distance d to an area of A_{Lens}. From this relationship the following specifications can be inferred. Firstly, maximising the intensity of light source will increase the power delivered to the sensor. Secondly, making the sensor area as big as possible will maximise signal capture. Finally, the apex angle of the light source should be minimise to direct most of the power towards the sensor. To generate specific requirements for light intensity, apex angle, sensor size and sensor sensitivity, further research of components and calculations were needed.

The chosen sensor is the Osram Opto BPW21 Full spectrum photo-diode with maximum sensitivity for light with frequency $550 * 10^{12} Hz$, a sensitivity of $0.34 A/W$ and with noise equivalent power of $0.074 pW/\sqrt{Hz}$. These values allow the resolution of the photo-diode to be determined.

$$P = 0.074 * 10^{-12} * (550 * 10^1 2)^{(1/2)} \tag{18}$$

$$P = 1.735 * 10^{-6} \tag{19}$$

In the above equation P is the smallest power difference detectable by the photo-diode. In order to ensure sufficient SNR and improve reliability, the assumed resolution will be 100 times this value.

In terms of the size of the optics needed in front of the detector, it was agreed that a diameter of 15 cm should be used. This decision was driven primarily by the fact that two $15cm$ diameter glass lenses

were provided by the university of Glasgow staff. This gives the area of the detector to be $0.01267m^2$. The distance d will be set to 1000m to once again ensure robustness and reliability. The apex angle will be set to 120 degrees as it is a fairly typical apex angle for flat high intensity chip on board (COB) LEDs. Equation 15 can then be manipulated to define the necessary intensity of the light from the beacon.

$$I(Lm) = \frac{P * \pi(d * tan(\frac{\theta_{apex}}{2}))^2}{683 * A_{lens}} \tag{20}$$

When the chosen values are inserted in this equation it claims that to detect the signal in distance of 1000m, by sensor with sensitive area of $0.01267m^2$ and resolution of $173.5 * 10^{-6}W$, the light intensity required is $188.96Lm$. This value is acceptable, and therefore the feasibility of the project can be theoretically confirmed.

5.2.6 Control Module

A module is required to run control software and permit GPS functionality. The basic design is given in Figure 13 and contains the main system micro-controller, the two motor drivers, a compass module,a GPS module and a compass module. This control module is responsible for accepting inputs from the lock-in and GPS peripherals and defining the control system for the system.

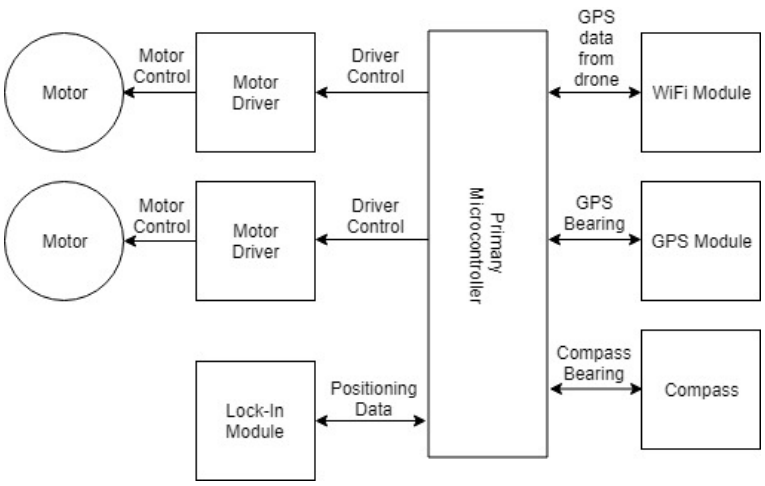


Figure 13: Control module architecture.

5.2.7 Power Supply Module

This module was designed based on the power requirements of all electrical modules used in the tracking system and the power resources available from Police Scotland's vehicles. The vehicles are

supplied with a generator producing mains AC at 230V/50Hz, thus input to our design was based around this specification. To make the design modular, it was decided to design the detector and the lock-in amplifier modules with combined power requirements of $500mA$ while being supplied from double rails $+18$ and $-18V$. Further, the power requirements for the control board module and the motors, were estimated to be $5A$ at $20V$ together. To accommodate these requirements, the architecture of the power supply module was designed. A main SMPS PSU provides isolation and an output with a voltage above $36V$ required for split rail generation. This output is input into a power distribution board which splits the input from the main SMPS PSU into the desired rails for all electronic modules. For the critical low noise precision operation of the detector and lock-in amplifier modules, a linear regulator generating the required split rails and with appropriate filtering to suppress ripple is to be implemented. For the high power requirements of motors, and the non-critical electronic modules, a SMPS is to be implemented to benefit from the smaller size and higher efficiency. (Figure 14)

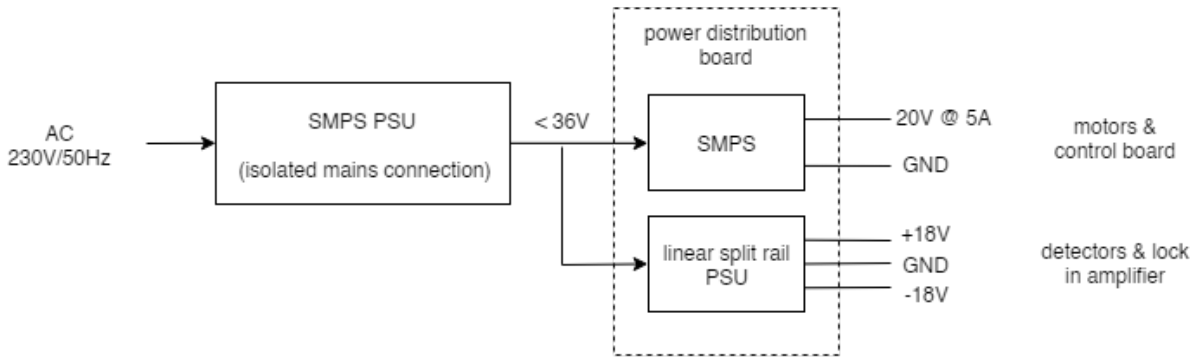


Figure 14: Designed power supply architecture.

5.2.8 Antenna Module

The design for this module was driven by requirements of Police Scotland the team responsible for the WiFi communications. It has to be able to transmit to a range of at least 500m with a narrow enough beam width to provide secure communications which cannot be intercepted.

5.3 Software

5.3.1 GPS Program

GPS data is communicated in a data structure known as NMEA. This contains a multitude of information most of which is surplus to requirement. A key requirement of the GPS system is the translation of a NMEA string such as below into a friendlier format.

*GPGGA,110617.00,4123.56789,N,00831.54761,W,1,05,2.68,129.0,M,50.1,M,,*42*

As the bearing from point A to B is required, the coordinates of both must be known. Once retrieved the following equations may be used to calculate the bearing from magnetic north between A and B:

$$x = \cos(\theta_B)\sin(\Delta L) \tag{21}$$

$$y = \cos(\theta_A)\sin(\theta_B) - \cos(\theta_B)\sin(\theta_A)\cos(\Delta L) \tag{22}$$

$$\beta = \arctan 2(x, y) \tag{23}$$

Where θ_A is the latitude of point A, θ_B is the latitude of point B and ΔL is the difference in longitude of point A and B. β is the angle in degrees from magnetic north to point B.

Now that a desired bearing is known, the system must be able to find its current position with regards to it. A magnetometer can be used to supply an angle from magnetic north, a, which may be used to find the platform's azimuth error from the desired position, $\varepsilon_{azimuth}$. This is used by the motor controller to drive the mechanical setup horizontally:

$$\varepsilon_{azimuth} = \beta_{azimuth} - a_{azimuth} \tag{24}$$

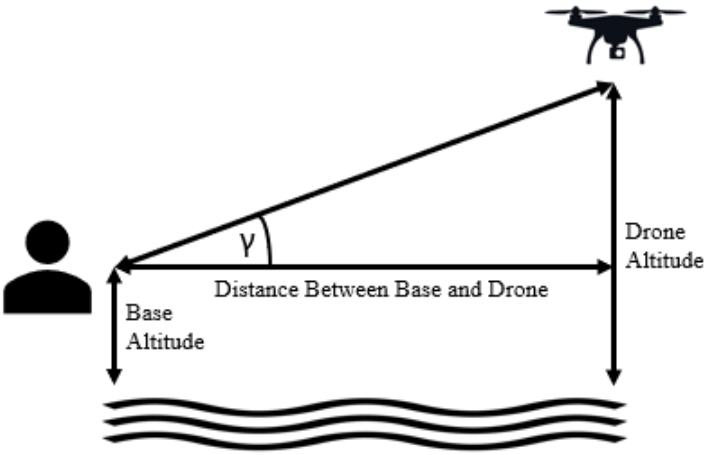


Figure 15: GPS Altitude

To find the altitude bearing, data from the GPS modules may be used once again. The following equation is used to find γ, the bearing in degrees from the horizon to the drone in the vertical axis.

$$\gamma = \arctan(\frac{h_{drone} - h_{base}}{d}) * \frac{180}{\pi} \tag{25}$$

Where h_{drone} and h_{base} are the height of the drone and base from sea-level, and d is the horizontal distance between them. The mechanical rig's current elevation must be found to create an error for the elevation control system.

5.3.2 Detector Sampling

The sampling program must be able to quickly and accurately measure the analogue output of an optical detector. This requires consistent and accurate timings, triggered automatically by the code through interrupts.

Acquiring this data from all four detectors requires two distinct operations:

1. Periodically sampling at the output of the lock-in amplifier.
2. Periodically switching the input of the lock-in to the next detector.

The sampling period of the ADC must be suitable for the signal created by the lock-in amplifier. At

the demodulated frequency of $\Delta\omega = 30Hz$, a suitable sample period is determined such that accurate digitisation is assured. It is estimated that 100 samples per period is sufficient, thus a sampling period $T_{sampling} = 333\mu s$ is to be used. The multiplexer switching period must be fast enough to capture a full period of the $\Delta\omega = 30Hz$ signal, so that the magnitude of the signal can be inferred and thus switching every $\omega_m = 20Hz$ is acceptable.

Once the signal magnitude is found for each detector, the corresponding axis may be compared to infer the beacon position relative to the detector array. By subtracting the 'right' detector's maximum from the 'left' detector's maximum, a differential value related to position is gained, denoted as $\delta_{azimuth}$. The same applies for elevation, the 'top' minus the 'bottom' detector yields a value for $\delta_{elevation}$.

When the light is centred to all four detectors $\delta_{azimuth}$ and $\delta_{elevation}$ are both zero. If the beacon moves left, the left detector to receives more signal and the right detector receives less, driving $\delta_{azimuth}$ positive. The magnitude of this positive value denotes the severity of the beacon's displacement. In a similar fashion, an increasingly negative value of $\delta_{azimuth}$ denotes movement to the right of the rig. The same methodology is applied for $\delta_{elevation}$.

As the beacon moves further away however the magnitude of the captured signal will decrease, causing smaller differentials and hence a less accurate control response. To combat this, two sets of gain multiplexers are controlled by the software, and will automatically increase/decrease the gains dependant on the maximum value seen in a given time frame.

5.3.3 Control

The fundamental design of the control system is to maintain a zero differential between the optical detectors. For rotation about the azimuth, the two horizontal pairs of detectors are considered (detectors 1 & 2 and 3 & 4). If the value of $\delta_{azimuth}$ is positive then the rig must rotate towards the left (-X direction), otherwise, if $\delta_{azimuth}$ is negative then the control algorithm will make the motor turn so the rig rotates towards the right (+X direction). The concept for the elevation is the same, where a positive $\delta_{elevation}$ results in the motor making the detectors and antenna point upwards and a negative $\delta_{elevation}$ makes the detectors and antenna point downwards.

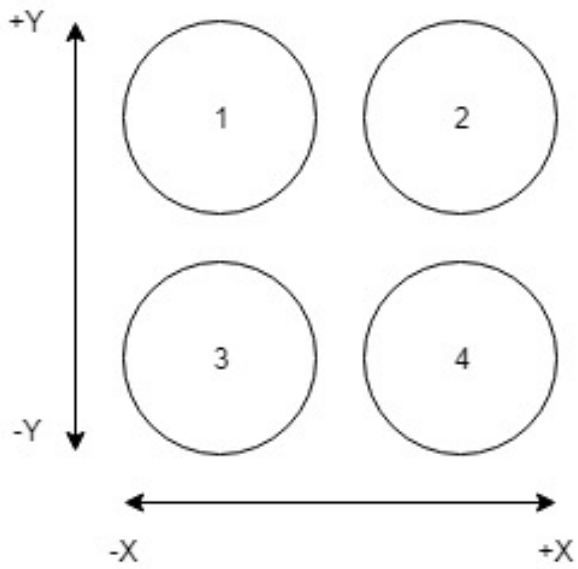


Figure 16: Detectors Layout

Additionally, the motor speed needs to be controlled according to how far from the centre the beacon is. A larger differential equates to a faster rotation to correct the error. Since the stepper motor drivers only allow for a preset rpm to be used, the only way to control the motor speed is to select the stepping mode. Three stepping modes were created:

- Full step for fast rotation
- $\frac{1}{4}$ step for medium rotation

Since 200RPM was the selected velocity, the above mentioned step sizes constitute to 200RPM and 50RPM respectively.

6 Implementation

6.1 Overview

Practical implementation of the designed architectures of all hardware and software modules is presented here.

6.2 Hardware

6.2.1 Mechanical Assembly Module

The mechanical platform consisted of T-slot arranged into a H-formation with a thrust bearing sitting on the bridge of the H. Attached to the bearing was a steel platform with a ring welded on it which was fastened to the bearing via set screws.Two equal lengths of T-slot were bolted horizontally to the steel sheet to form the base of the U-shaped structure. Long steel arms were attached at both ends of the T-slot in a vertical position and held by L-brackets, forming the sides of a U-Shape.

The detector tubes and antenna assembly were positioned in the centre of the U-shaped structure and held by a pin. This allowed the assembly to freely pivot up and down. The centre of mass of the tubes and antenna were focused just to the front of the U-shaped structure so that they could fall forwards due to their own weight, as shown in Figure 17 . A stepper motor with a spool of string was fixed on the platform on the rear side of the U-shaped structure, the motor can real in the string which would elevate the tubes and antenna or it could release the string to allow the detectors to lower themselves.

The initial design included the use of a gear with 30cm diameter fixed at the centre of the bearing on the base. However, producing a gear of this size meant that any small errors in the dimensions would add up and result in a faulty product, which led to the SolidWorks file of a smaller gear with diameter 14.3cm and 90 teeth being sourced from the internet and laser cut out of Perspex. The reduction in size of the large gear did not compromise the rigs ability to keep up with the drone in the horizontal; this can be seen from the following calculations:

$$\frac{N_S}{N_L} = \frac{\omega_L}{\omega_S} \tag{26}$$

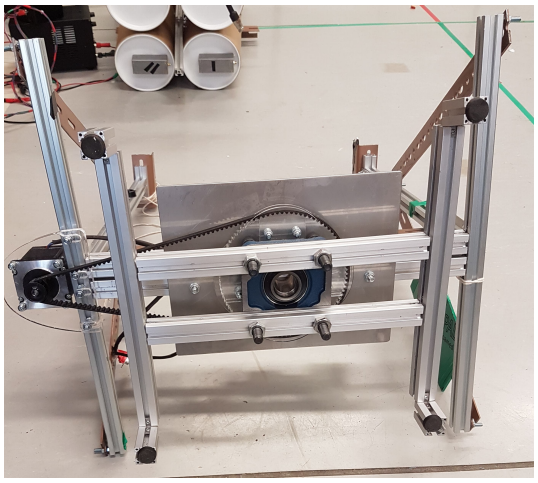
Where N_S is the number of teeth on the small gear, N_L is the number of teeth on the large gear, $\omega_L$is the large gear's RPM and ω_S is the small gear's RPM: $\omega_S = 200RPM$. Substituting into equation 27 gives:

$$\omega_L = \omega_S * \frac{N_S}{N_L} \tag{27}$$

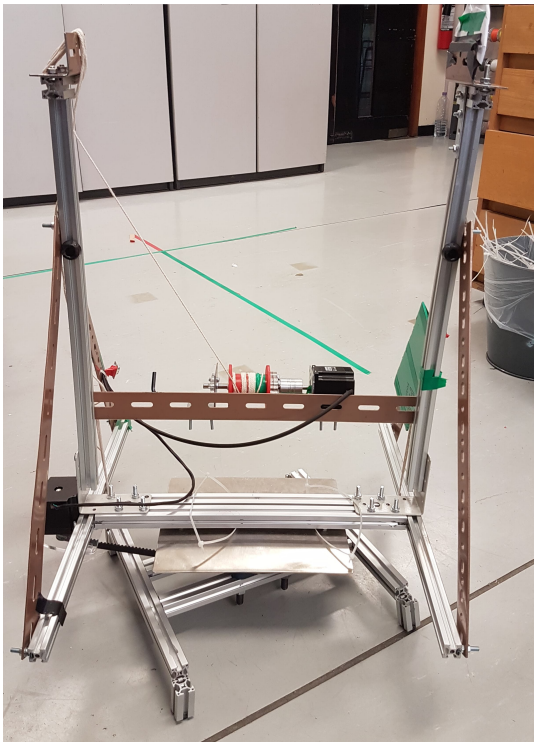
$$\omega_L = (2/15) * 200 = 26.67RPM \tag{28}$$

This speed is still sufficient to maintain target tracking.

To move the rig, two stepper motors were used. One to control the rotation in azimuth and the other to control the elevation. The UEETEK NMEA 23 stepper motors were chosen as they provide 1.26Nm of Torque and are the most economical that could be found. To drive these the TB67S249FTG Stepper Motor Driver were chosen as they can provide the necessary power to the motors. These motors provide more torque than the original estimation but they proved to be cheaper than lower power stepper motors.



(a) Base final assembly



(b) Platform final assembly

Figure 17: Final mechanical construction

6.2.2 Optics Module

The final optical set up consists of four individual detector tubes mounted together. It was impossible to source four glass lenses due to financial constraints, therefore two fresnel lenses were used in combination with two glass lenses. These lenses can be made from lighter materials and theoretically can provide same optical characteristics as conventional glass lenses. The biggest advantage of the fresnel lenses is that they can be made much thinner than their convex alternatives. Thus, usage of fresnel lenses in this project minimises the weight of the optical modules and therefore reduce the torque specifications for the driving motors. Using these lenses was not an ideal solution however and will be discussed later.

The glass plano-convex lenses are mounted at a distance of 200mm from the detector photo-diode, while the fresnel lenses were mounted at a distance of 140mm which corresponds to the focal lengths of the lenses respectively. The lenses were subsequently mounted into a tube with extension length of X=680mm in front of the lenses, as discussed in the design to achieve desired θ_{view} and θ_{app}. In order to correctly align the lenses with the photo-diodes grooves were cut into the interface between the optic and detector, allowing the detector modules position to be adjusted as needed relative the centre of the lens.

Aligning the optics relative to each other was achieved by attaching a metal framework to either side of the tubes. The beam width was measured during implementation, and this was used to determine $\theta_{detector}$ for the alignment of multiple detector tubes in a single assembly. With this setup, the measured beam width was approximately $\theta_{view} = 7°$. Thus, $\theta_{detector} = 3.5°$ was used among all 4 detector tubes which was designed into the metal framework geometry. To mount the optical setup to rest of the moving mechanical assembly, a shaft running between the upper and lower pair of optics was used. The final construction can be seen in Figure 18.

Figure 18: Developed detector assembly.

6.2.3 Detector Module

The finalised detector design utilises the Philip Hobbs photo-diode front end [5] with an additional bandpass filter on the output. The schematic is given in Figure 19, with the PCB design given in Figure 20. Although the schematic is fundamentally the photo-diode front end recommended by Philip Hobbs, his design used legacy components which are now unavailable. Similar components were chosen with respect to bandwidth and noise. The ADA4637-1ARZ (see Figure 19, component U9) proved to be an extremely effective transimpedance amplifier, and was utilised again in the lock-in amplifier due to its low noise characteristics. The photo-diode BPW21 was selected due to its high sensitivity $S = 10nA/Ix$, low parasitic capacitance $C_p = 580pF$ which would otherwise limit the bandwidth, low equivalent noise power $NEP = 0.074pW/\sqrt{Hz}$ and wavelength of maximum sensitivity $\lambda_{max} = 550nm$. Green light with wavelengths around $550nm$ was selected for tracking because in natural environments it has useful properties. Ambient sun light is concentrated on wavelengths of red light and operating here would add more noise, while wavelengths corresponding to blue light suffer from high scattering as they propagate through atmosphere and thus the signal would be attenuated.

The Analogue Devices Filter Wizard was used initially to design the filter, however it recommended an expensive chip, the ADA797. The OP27G (see Figure 25, component U18) was chosen as a cheap alternative and the filter design tweaked as the system was unstable when the new chip was introduced. The pass band of this filter was designed as $5kHz$ around $\omega_s = 250kHz$ as a 2nd order Chebyshev active filter. The designed cut of frequencies are $f_{1_{-3dB}} = 247.5kHz$ and $f_{2_{-3dB}} = 252.5kHz$ which allows this filter to perform AC coupling, and pass only the desired signal to the lock-in amplifier.

The board design minimises board area, populated with components on both faces. This allows the detectors to be mounted onto the optics themselves, increasing the robustness of the design. Another requirement for this board was that it must be stable over time, hence the separation of the regulation and detection circuitry. The board features two isolated ground planes, furthering the modules robustness as sensitive analogue signals are provided with a clean ground reference. The output is given as a differential to the Lock-In Amplifier using this clean ground, through a SMA Coaxial cable. Four of these detectors were manufactured, each with its own metal enclosure and optic peripherals. The photo-diode is mounted inside a 3D printed bracket, and the PCB is fixed on top. This in then attached inside the enclosure. The use of SMA cables and metal enclosures were to further minimise the effects of noise within the detection system. The enclosure containing the detector is then mounted onto a disk which attaches to the optics, shown in Figure 21.

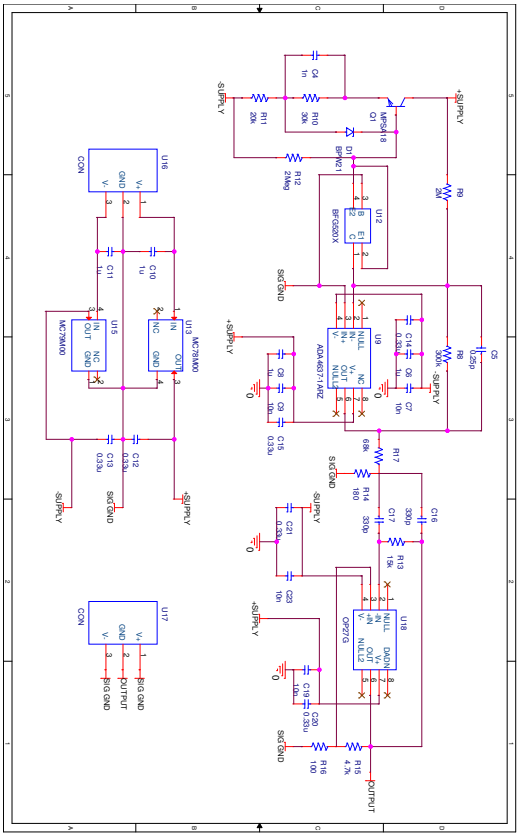
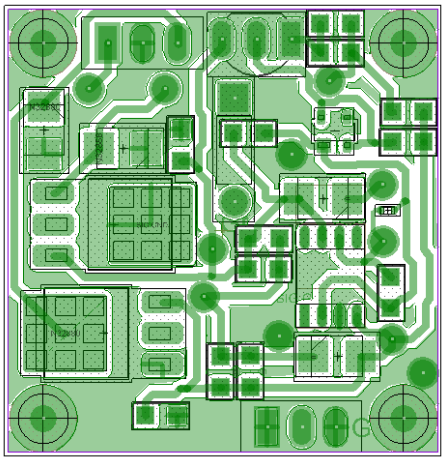
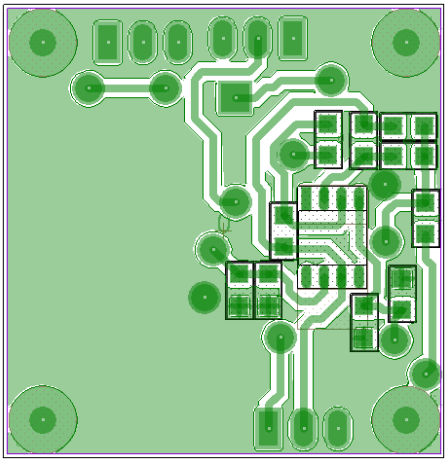


Figure 19: Detector Schematic

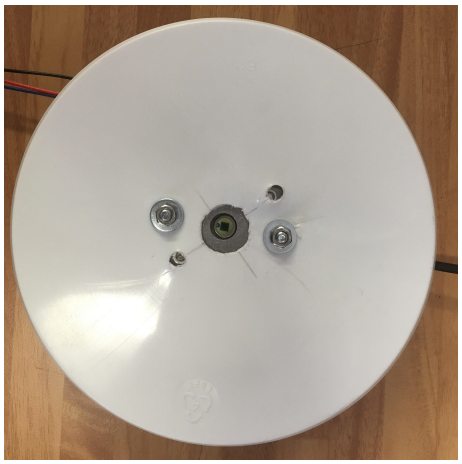


(a) Top



(b) Bottom

Figure 20: Detector PCB



(a) Lens side showing photo-diode window

(b) Outside

Figure 21: Detector Assembly

6.2.4 Lock-In Amplifier Module

Practical implementation of the lock-in amplifier is the culmination of numerous iterations. It contains multiple improvements in the form of additional filtering and gain stages as compared to the initial design architecture. The final schematic is given in Figure 25 and discussion of each main block follows.

The first stage of this system is the multiplexing of differential detector inputs. Differential inputs signals consisting of the signal and its reference ground are passed to the lock-in, which selects one of the four detectors using two multiplexer. These multiplexers are in parallel for selection of both a signal line and reference ground.

The next stage is a differential amplifier which addresses practical problems associated with grounding and inputs to the lock in amplifier as this system relies on precision low-noise design and provides phase shifting of the input signals. Grounding issues arising from interconnecting of multiple PCB's and EMI problems where addressed. Generally, star point grounding was adopted across all the individual PCB's which make up the system, with each of the systems having local regulation with local star point grounding configuration. However, having multiple PCB's sharing precision signals referenced to local grounds becomes problematic as these introduce ground loops offsets, allow common mode noise to couple onto the system and breaking the overall star point architecture. (Figure 22) Thus, it was crucial to isolate the PCB's such that their local grounds are not tied together and to

introduce measures that minimise EMI coupling. [6][4] To achieve this, differential inputs to the lock in amplifier and coaxial cables providing shielding were used to pass signals from detector modules. The action of a unity gain differential amplifier references input signals from their local ground to the local ground of the lock-in amplifier without tying them together and thus avoiding ground loops (see Figure 25, component U25). The ADA4084 was chosen as it meets bandwidth, low-noise and precision requirements at the input, and to minimise the errors resistors of $R = 10k\Omega \pm 0.1\%$ are used. This package contains two difference amplifiers which are connected such that their outputs are inverted. This splits the input into two equal components in antiphase as required (See Figure 12), creating a differential signal for the demodulation process.

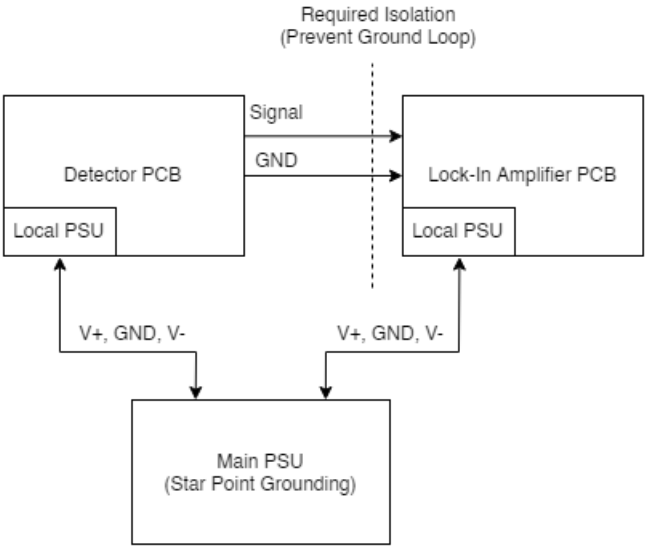


Figure 22: Example of star point grounding, among two PCB's and the main power supply unit. Any signals shared across PCB's must be isolated to prevent ground loops.

The following stage implements the mixers performing the synchronous demodulation critical to the overall detection system. Firstly, both the inverted and non-inverted input signals are passed into an analogue double-pole double-throw (DPDT) switch used as a mixer. This switch is the DG403DJ+ (see Figure 25, component U5), chosen for its system compatible attributes such as low on resistance and +/- 15V supply rails. The two signals are connected to the DG403DJ+ as shown Figure 23. The four switch architecture of this device defeats any possible distortion of the output due to unmatched turn off/turn on times of the internal switches.

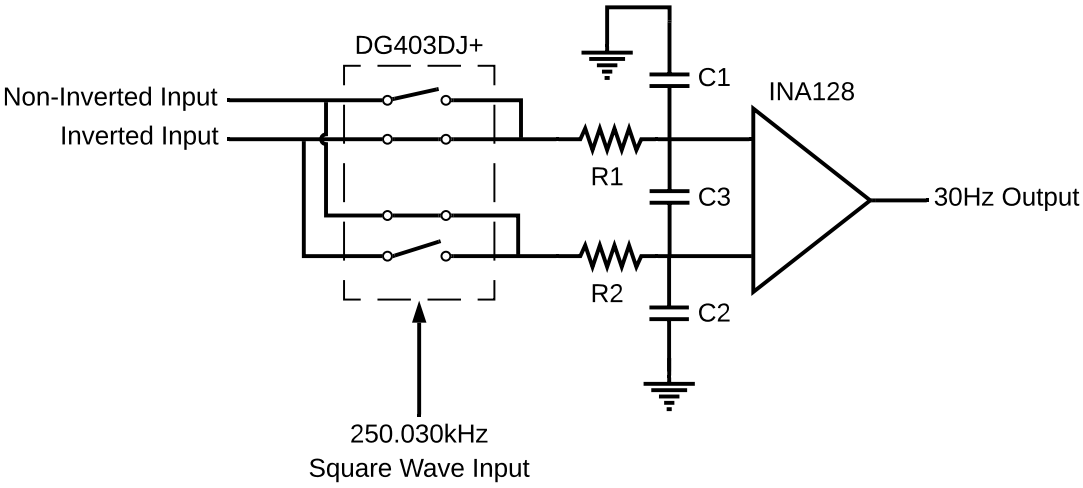


Figure 23: Switching circuit, along with LP filter and an instrumentation amplifier.

The switch is driven by circuitry containing a 20MHz crystal oscillator and the AD9833 function generator chip (see Figure 25, components U27, U26). This allows the switching frequency ω_r to be configured using SPI communication between a micro-controller (Teensy 3.2) and the AD9833 allowing for flexibility as it can be used to reconfigure the resultant $\Delta\omega$ if needed. The signal obtained from this circuit can be regarded as reliable and stable since the quoted accuracy with 20MHz clock rate is $0.07Hz$. Since the output frequency of this chip can be changed easily the stability of the output frequency can be assured by using a feedback loop in case the frequency drift is large. This can be understood to be a digital version of voltage controlled oscillator and can be implemented to improve reliability and robust operation.

With the input signals modulated at a frequency of $\omega_s = 250kHz$ and the switching performed at $\omega_r = 250.030kHz$, the switch output are two signals comprising of complementary portions taken from the inverted and non-inverted inputs. Because of $\Delta\omega = 30Hz$, these signals are modulated at $30Hz$ and $500.030kHz$ as expected from a mixer. These complementary signals are input into inverting and non-inverting inputs of a differential amplifier, amplifying only the differential components of the signal (sinusoidal modulation at frequencies $30Hz$ and $500.030kHz$) while rejecting common-mode components (the complementary portions of original signal). Because of this, it is desirable to use an instrumentation amplifier with high $CMRR$ for this process.

The instrumentation amplifier selected is the INA128 (see Figure 25, component U6) due to typical $CMRR = 130dB$ at gain $G = 1000$ and frequency $f = 100Hz$. Additional reasons for selection of INA128 include other specifications of this amplifier giving rise to negligible errors in practical

implementation within our circuit. As outlined during design, to achieve high dynamic range, variable gains had to be implemented. This was achieved by using a multiplexer interfaced by a micro-controller (Teensy 3.2) which can select appropriate gain resistor and thus set desired gain. Resistor values R used were selected selected based on desired gain G using equation:

$$G = 1 + \frac{50k\Omega}{R} \tag{29}$$

Estimating the desired gains G was done by assuming the amplitude of smallest detectable signal by the lock-in amplifier to be $50nV$ and assuming that the largest detectable signal will have amplitude of $3.3V$, which defined the wide input range at which the input is to operate. Then using an inbuilt micro-controller (Teensy 3.2) ADC with $12bits$ and input range of $0V - 3.3V$, multiple required gains G are determined by splitting the required wide input range $50nV - 3.3V$ into multiple smaller input ranges in ascending order were each smaller range consists out of minimum and maximum signal magnitudes which are amplified into $0.5V$ and $1.5V$ respectively by a specific gain G. (Figure 24) Thus, each smaller range can be attributed to a specific gain G which amplifies the signal this specific range into the ADC range. Because only a finite gain can be provided by the instrumentation amplifier, and because the maximum required amplification is $146.02dB$, the overall gain was split into two gain stages. Instrumentation amplifier variable gain provides $G_1 = 10000/3030/920/279/84/26/7.7/2.4/1$, and a secondary non-inverting operational amplifier gain stage with a secondary multiplexing (see Figure 25, component U50, U51) provides $G_2 = 909/3/1$. By combining the two gain stages, the whole wide input range is covered, providing high dynamic range of the lock-in amplifier.

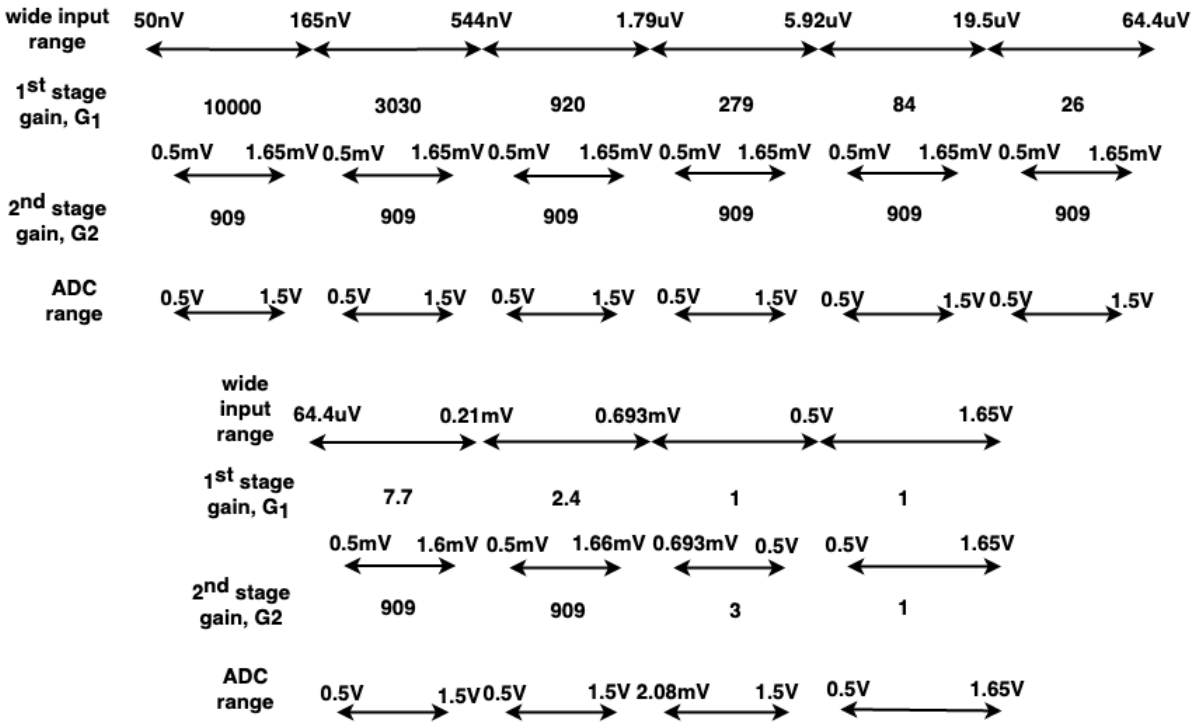


Figure 24: Splitting wide input range into smaller ranges using specific gains on 1st and 2nd amplification stages.

Practical implementation of the lowpass filter was based on the design parameter $f_{-3dB} = 40Hz$ selected earlier. During implementation process it became clear that the filter order and the overall time constant τ need to be determined. Furthermore, because an instrumentation amplifier was used, pre-filtering the signal after demodulation with a $1st$ order differential lowpass RC EMI filter adds benefit of easing the amplification process. (Figure 23) [1]

This filter was designed by selecting differential bandwidth to be:

$$BW_{Diff} = \frac{1}{2\pi(R_1 + R_2)(\frac{C_1 C_2}{C_1 + C_2} + C_3)} = 40Hz \tag{30}$$

Assuming $R_1 = R_2$ and $C_3 >> C_1, C_2$ this simplifies to:

$$BW_{Diff} \approx \frac{1}{4\pi R_1 C_3} = 40Hz \tag{31}$$

Choosing $R_1 = R_2 = 2k\Omega \pm 1\%$ low resistance to minimise johnson noise, $C_3 \approx 1\mu F \pm 1\%$

and in practise to maintain high CCMR of the instrumentation amplifier, $C_2 = C_1 \approx 0.1\mu F \pm 1\%$ are selected to be one order of magnitude lower than C_3. These values yield $\tau_{Diff} = 26.3ms$ and $BW_{Diff} = 37.89Hz$ which is acceptable (see Figure 25, component U6).

After amplification, a higher order lowpass filter was implemented which provides the main filtering. It was important to make the filter sharp such that, large attenuation is achieved quickly. The Analogue Devices Filter Wizard was used initially to design a 3rd order Bessel active lowpass filter with two operational amplifier stages. (see Figure 25, component U18) Again $f_{-3dB} = 40Hz$ was used and the ideal filter was then tweaked by changing the component values slightly to get optimal response. The filters response time adds additional delay into the system quantified during testing.

Following the conditioning stages the signal is biased by passing it onto the non-inverting input of a precision unity gain difference amplifier INA154 (see Figure 25, component U49) with a $1.6V$ zener diode connected to the inverting input. The output of the INA154 is then clipped using a pair of schottky diodes, the BAT54S dual package, so ensure the signal does not saturate the ADC. The reference pin of the INA154 is tied to the reference ground of the ADC and only then to the nominal ground, to minimise errors from currents flowing in ground paths and retain high ADC accuracy. The micro-controller used is the Teensy 3.2 (see Figure 25, component U52) due to its high performance as mentioned earlier. After it has performed digital processing of the data is complete, the Teensy 3.2 communicates with a Teensy 3.6 housed on the control board PCB. This communication is mediated by an optocoupler FOD8012A (see Figure 25, component U54), ensuring that the two PCB's are isolated from one another.

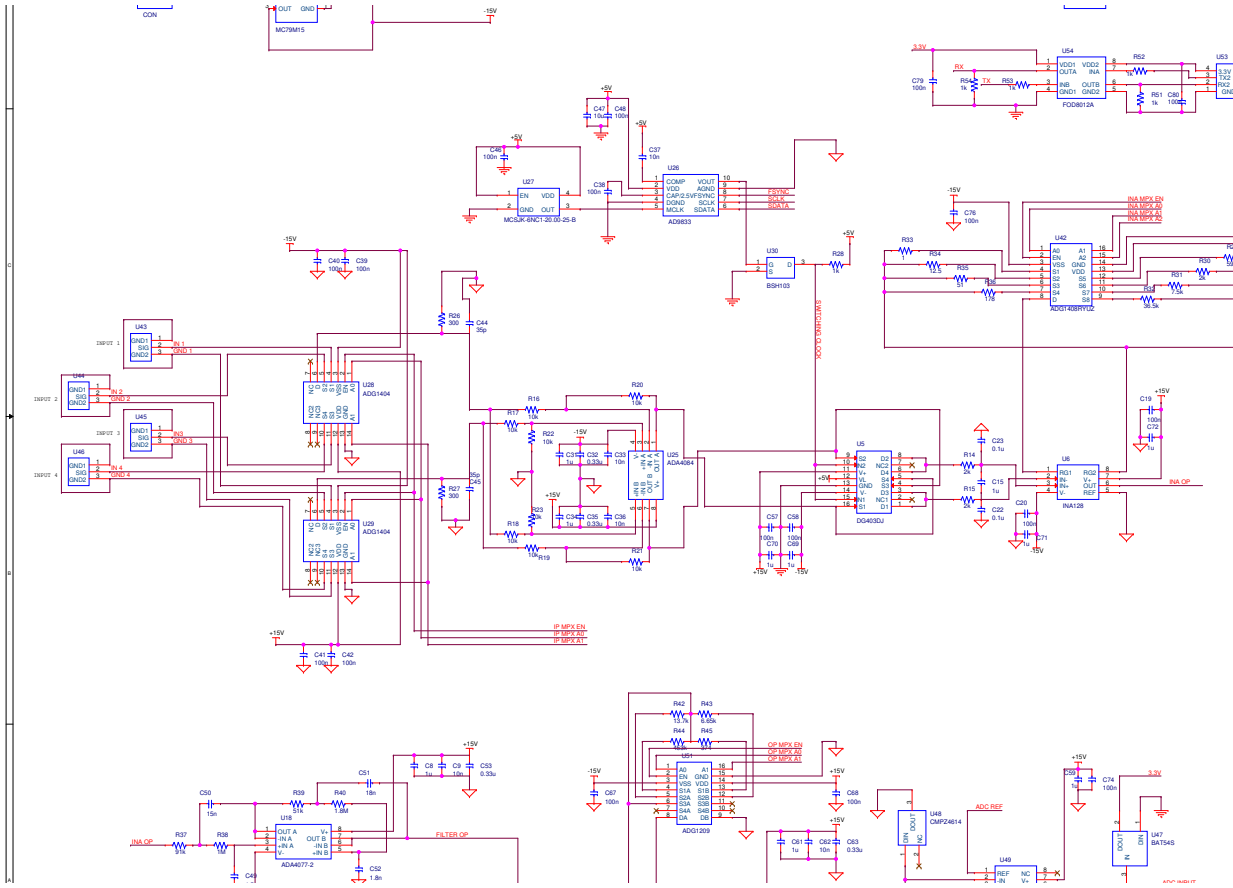
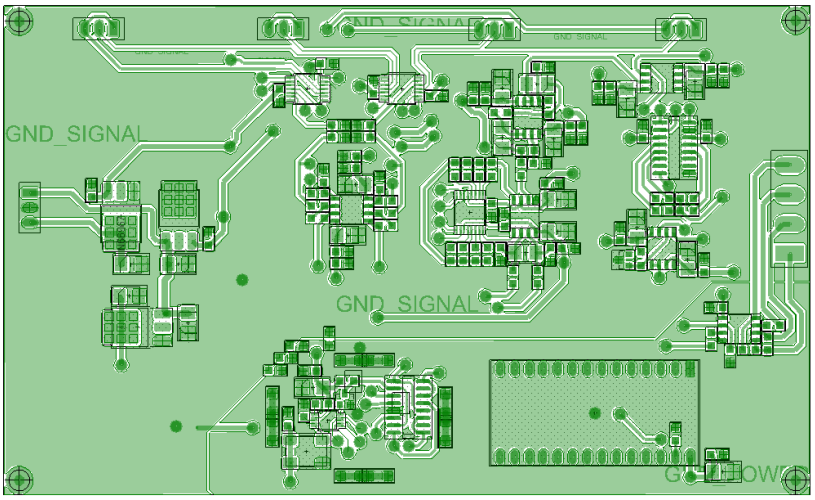
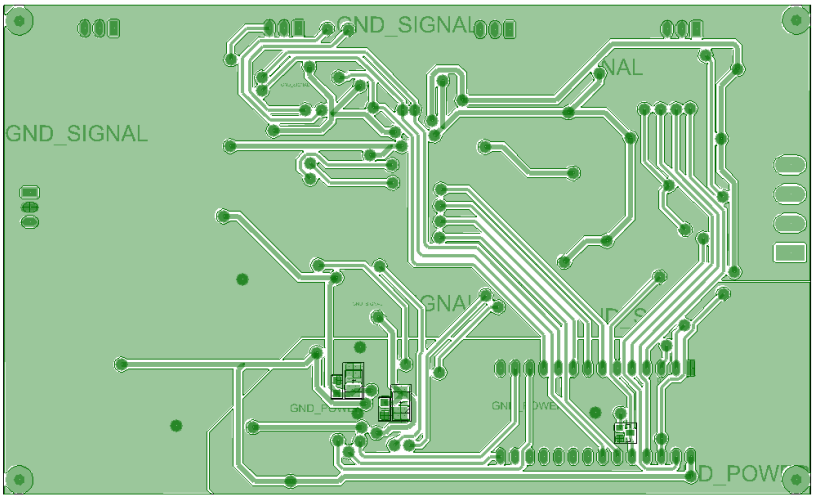


Figure 25: Lock-In Amplifier Schematic



(a) Top



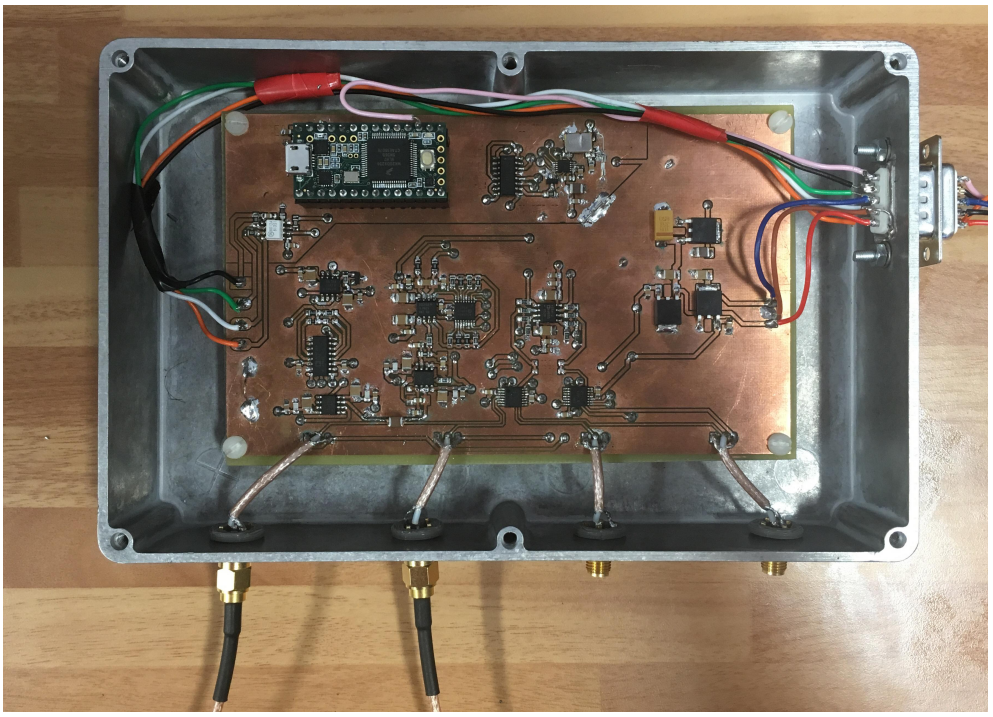
(b) Bottom

Figure 26: Lock-In Amplifier PCB

The final lock-in amplifier PCB is divided into two sections containing the analogue and digital elements of the system. This effectively isolates the sensitive analogue circuitry from the high frequency digital processes, and can be seen on the PCB below. The lower portion of the PCB is reserved for digital purposes, such as generation of the demodulation frequency and sampling of the final output signal. The upper portion houses all analogue circuitry and provides a clean ground reference. On the left of the board the power circuitry is given plenty of copper to dissipate excess heat, protecting the stability of the analogue and digital subsystems. Routing of the board is also divided into analogue, digital, and power. Analogue signals primarily run across the top face of the board, while digital signals are confined to the bottom layer. Any interactions between the analogue and digital

sections are also placed on the underside, to ensure plenty of ground paths for the sensitive analogue circuits on the top side. The power tracks travel on both faces of the PCB, and kept as short as possible. The flow of the PCB runs from the four coaxial inputs on the upper side (looking at Figure 26a), down through the initial difference amplifier stage and into the digital section for synchronous demodulation. This digital section contains the crystal oscillator and function generator chip, which control the analogue DPDT switch. The 'chopped' signal is then passed back up into the analogue section where it meets the INA128, followed by filtering and then a final gain stage. The signal is now clipped and biased so that it maximises the ADC of the Teensy 3.2, which then captures the signal. The data is then transferred off-board via serial communications through the optocoupler and output from the right hand side of the board.

This PCB is housed in a metal enclosure to ensure protection for EMI. It has four SMA coaxial inputs to accept four detector modules. The output and power of this board is supplied using a 9-way D-type. Final construction of this module can seen in Figure 27.



(a) Open

(b) Closed

Figure 27: Lock-In Amplifier Construction

6.2.5 Light Beacon Module

The design of the Beacon module was driven by the mathematical approximation of attenuation of light in the atmosphere mentioned in the design section of this report. It was also necessary to take into account weight specifications given by the drone assembly team as well as the need for an omni-directional light source. Additionally to this, to ensure correct operation of the optical localisation it was crucial to implement a stable and accurate modulation frequency with which to drive the light. The modulation frequency was set at $\omega_s = 250kHz$ as explained in the design. This was achieved using a crystal oscillator with fixed $1MHz$ frequency and two D-type flip-flops as frequency dividers. Thanks to the $15ppm$ stability of the crystal oscillator a stable and accurate square wave signal was obtained.

All of the components of the circuit require 3.3V supply, supplied from a 18.5V LiPo battery. Two stage voltage conversion was implemented. The first stage generated 5V from the 18.5V supply using a switch mode power supply (SMPS) from TRACO Power. To protect the SMPS from high voltage, two standard recovery diodes with voltage drop of 1V were placed on the input. The second stage was a fixed low drop out voltage regulator which provided 3.3V. The described circuit is given in Figure 28.

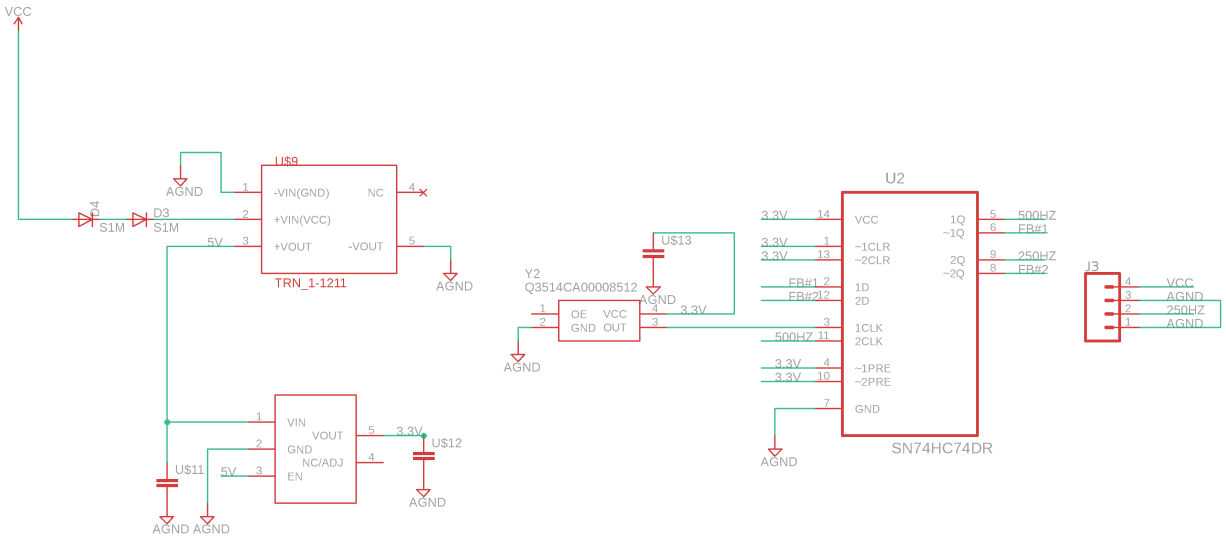


Figure 28: Beacon frequency generation schematic.

The already mentioned analysis of light dissipation in the atmosphere suggests that the intensity of the light should be set at $189Lm$. Unfortunately this value did not provide signal strong enough to be seen from much shorter distance than 1000m. This was attributed to approximations in the

model. Therefore, it was agreed that light intensity should be maximised to ensure functionality as well as improve *SNR*. Since the value acquired from the mathematical model provided only the lower limit for the light beacon intensity, other aspects like price and size had to be used to chose adequate component. Maximising the light intensity while minimising costs and size lead to the use of the $10W$, $720Lm$, high power, green COB LED from Futureeden. Since a single LED would not be enough to meet the specification it was decided that up to 18 LED's should be used for the final product. The LED has a beam width of $120°$. To ensure omni-directionality of the light source the LED is mounted on 3 separate faces forming a equilateral triangle. Even distribution of the 18 LEDs will ensure sufficient light intensity in every direction and therefore ensure constant visibility of the beacon. To enable mounting of the beacon on to the drone an enclosure was designed that would hold 3 sub-modules and have mounting points in every corner. The mentioned sub modules will be described later in this section. The beacon assembly design is given in Figure 29.

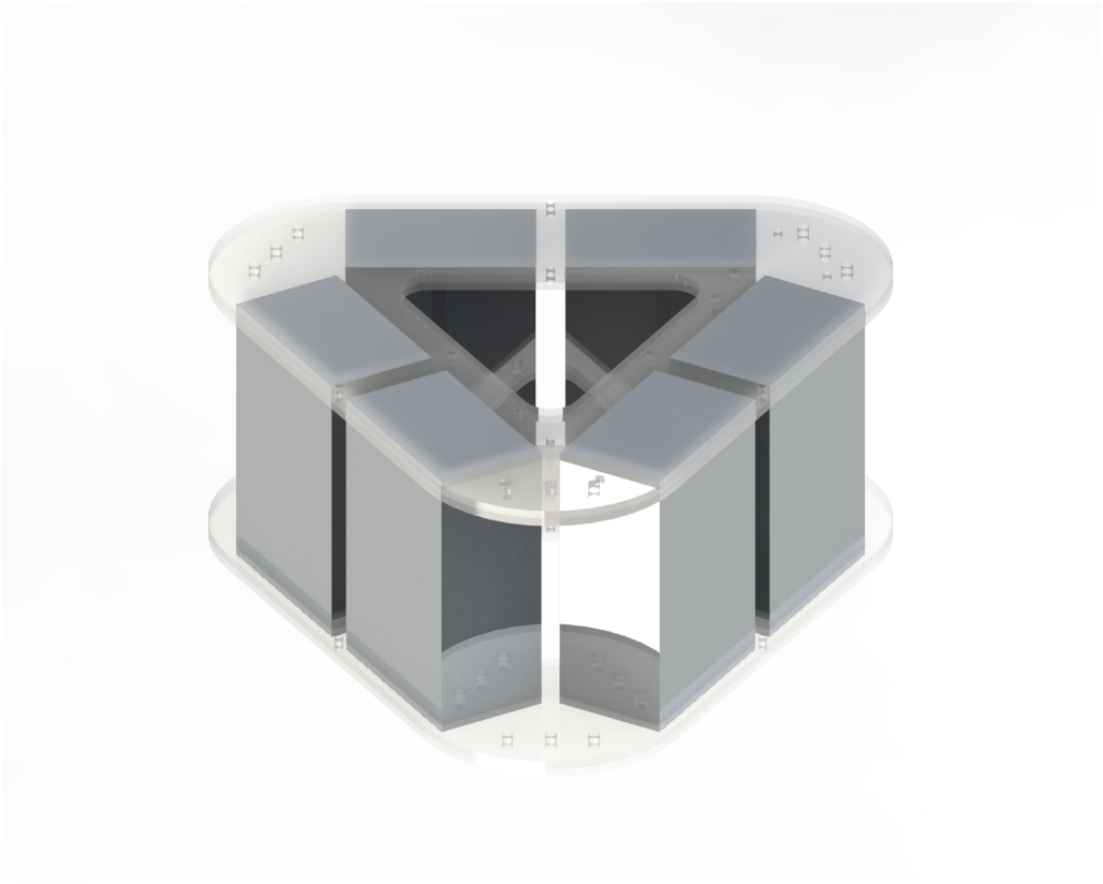


Figure 29: SolidWorks model of final beacon assembly.

Choice of the mentioned High power LED's resulted in current and thermal management specifications. To drive The LED's at $720Lm$ they require a current of 1A. The maximum junction tem-

perature of the LED's is 115°C and maximum power dissipation is 10.5W. With a forward voltage drop between 8 and 10 V it was agreed to use parallel connection between the LED's. This would require up to 18A if all LED's were connected to one power supply unit. This was considered to be not feasible and possibly dangerous. Therefore it was decided to create modular units of LED's where each unit would be able to accommodate up to 6 LED's. The design expects 3 units to form one omni-directional light source. Each sub-module would have a SMPS from GE Critical Power providing up to 6A at 12V, 3 power MOSFET's driving 2 LED's each and one 2Ω power resistor. MOSFET's were used in a low side switch configuration with up to two LED's connected to a single MOSFET. The 250kHz driving frequency is delivered to the MOSFET's from the beacon frequency generation board. Due to high voltages the power resistors were chosen with 75W power rating. This should be sufficient to withstand the 72W expected power dissipation. The resistor is connected to the 12VOUT line and to the anodes of all LEDs. The schematic for the described circuit can be seen below in Figure 30.

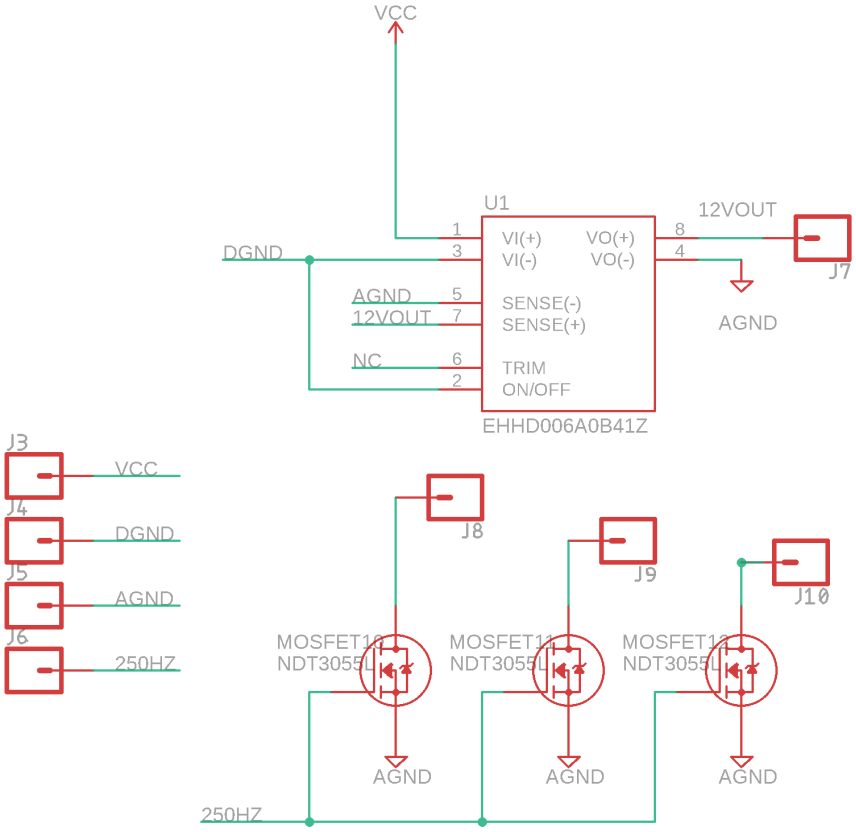


Figure 30: Beacon sub-module schematic.

In terms of thermal management the modules were designed for outdoor use with assumed tur-

bulent air flow as one might expect on a moving drone. The calculation below was performed to determine the necessary heat sink characteristics for the thermal management of the LED's. Upon research it was decided that each heat sink will accommodate 3 LED's.

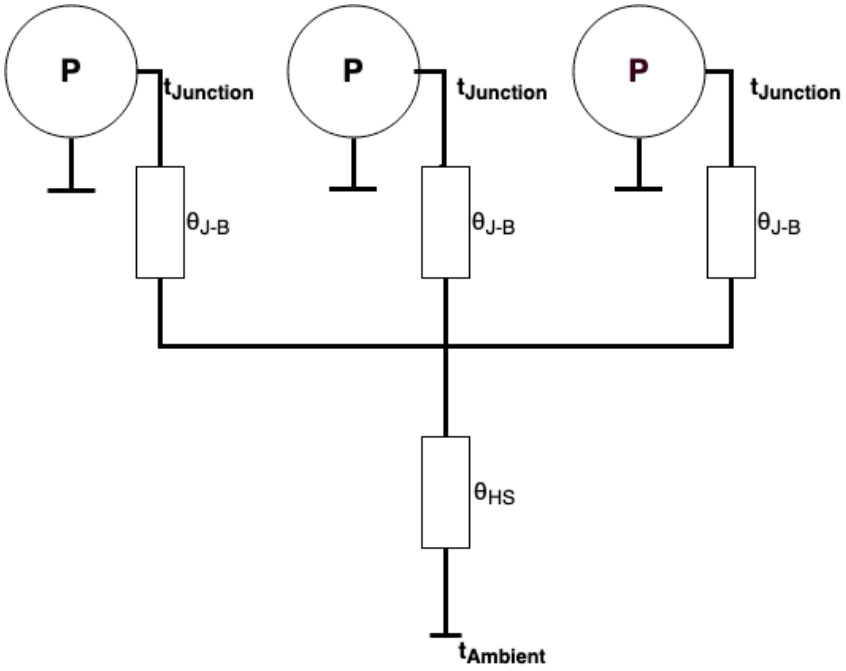


Figure 31: Standard thermal management diagram of 3 power LED's on a single heat sink.

Standard static thermal analysis was used in order to determine the necessary thermal resistance of the heat sink.

$$\Delta t = P_{Total} * \theta_{Total} \tag{32}$$

$$\Delta t = t_{Junction} - t_{Ambient} \tag{33}$$

$$P_{Total} = 3 * P_{LED} \tag{34}$$

$$\theta_{Total} = \frac{1}{\frac{1}{\theta_{J-B}} + \frac{1}{\theta_{J-B}} + \frac{1}{\theta_{J-B}}} + \theta_{HS} \tag{35}$$

This gives the following equation, where the only unknown is the heat sink thermal resistance.

$$t_{Junction} - t_{Ambient} = 3 * P_{LED} * (\frac{1}{\frac{1}{\theta_{J-B}} + \frac{1}{\theta_{J-B}} + \frac{1}{\theta_{J-B}}} + \theta_{HS}) \tag{36}$$

In order to find the value for thermal resistance of the heat sink θ_{HS} the values below are used in the equation above. These values for thermal resistance, power and maximum junction temperature

were found from the LED data sheet. The ambient temperature was assumed as 20°C.

$$P_{LED} = 10W \tag{37}$$

$$\theta_{J-B} = 1.33W/^{\circ}C \tag{38}$$

$$\theta_{HS} = xW/^{\circ}C \tag{39}$$

$$t_{Junction} = 115^{\circ}C \tag{40}$$

$$t_{Ambient} = 20^{\circ}C \tag{41}$$

Completing this calculation provides us with the largest possible thermal resistance for the heat sink to ensure the LED's remain under their maximum junction temperature.

$$\theta_{HS} = 2.723W/^{\circ}C \tag{42}$$

This value can not be trusted completely since multiple values were provided for the junction thermal resistance. This introduces a certain doubt in this calculation. To ensure reliable operation a heat sink with thermal resistance of $1.5W/^{\circ}C$ was selected. This overhead should ensure that the LED's will not overheat.

In terms of the Thermal management of the SMPS's it was decided that an additional heat sink is not required. The SMPS has its own heat sink and is expected to be in an ambient temperature not exceeding 20 degrees with turbulent air flow. This will ensure sufficient cooling for correct operation. The same logic applies also to the 2Ω power resistors. Overall weight of the beacon was evaluated at 2.874 Kg. This satisfies the given specification. It has to be mentioned that due to the modular nature or design it is possible to reduce the wight of the beacon in exchange for reduction in signal strength. For precise evaluation of this compromise additional range tests are needed.

Due to cost restraints it was agreed that only a single sub-module of the beacon will be needed for performing the required experiments. Other compromises ensured efficient use of resources and also provided limitations that guaranteed robustness of final product. Firstly, only three LED's were used in testing. This ensures that the range of detection can be extended by simple addition of more LED's on the beacon. Secondly, since the module does not require current of 6A, a smaller power resistor can be used. Finally, the battery used in testing has capacity of only $1300mAh$. This was sufficient for testing but would be very limiting for the final product as it would limit the flight time to something around 4 minutes when powering 3 sub-modules. These compromises had no major influence over the results of the completed tests and ensured efficiency of the project.

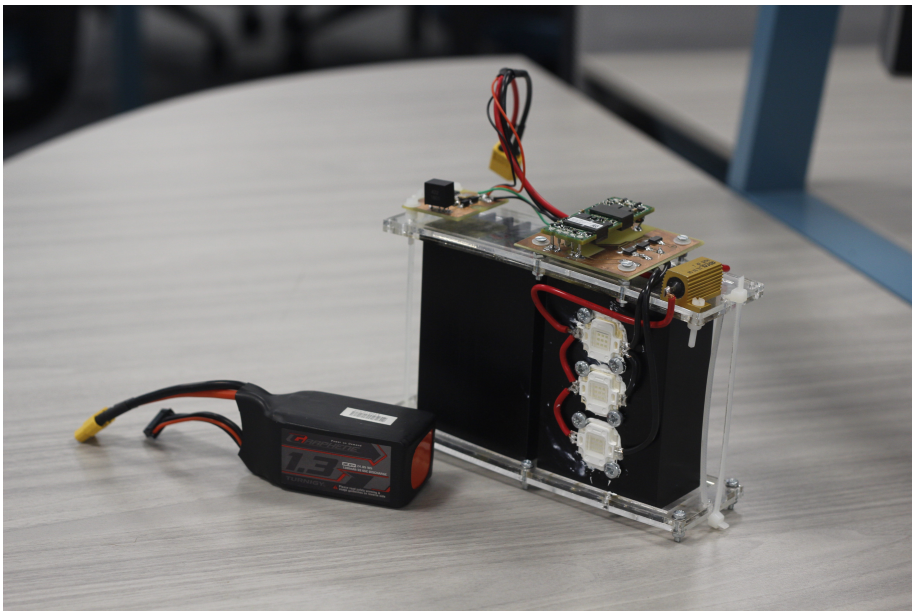


Figure 32: Beacon sub-module final construction.

6.2.6 Control Board Module

Following the design parameters for this module, the various peripherals necessary for the optical and GPS tracking systems were chosen. The final schematic and PCB for this module can be found in Figures 33 and 34 respectively.

Due to the large amount of sampling and communications required from the MCUs, the Teensy 3.6 was chosen over their official Arduino counterparts. At a comparable cost, the Teensy brand is able to sample at a higher resolution, clock significantly faster and communicate with many more devices. Further, the Teensy controllers have interrupts on all pins and the 3.6 contains a dedicated floating point unit.

To accomplish the GPS bearing calculations, coordinate data must be taken from both 'home' and the 'drone'. Receiving the home base data is trivial, requiring only a GPS module. For this task the SODIAL GY-GPS6MV2 was chosen thanks to its low cost and built in ceramic antenna. The module uses UART to communicate a NMEA sting which can be decoded by the Teensy 3.6.

Receiving the drone's GPS data is less easy however and required the WiFi team's assistance. The ESP8266 was chosen, a ubiquitous chip which can be regarded as a micro-controller in its own right. Due to this, setup was simple through the Arduino interface, functionality could be exactly controlled and the data sent to the Teensy 3.6 can be edited at will.

When both sets of GPS are combined, the desired bearings for azimuth and altitude can be found. This is of course useless however without a current bearing to compare to. Fulfilling this role in the azimuth plane is the Freescale MAG3110 on a SparkFun breakout board.

Driving the stepper motors is performed by a pair of Pololu TB67S249FTG drivers, which were chosen due to their large maximum current per phase and useful built-in automatic control systems. The motors and control board are powered by a single power supply which provides 20 Volts and 1.1 Amps directly to the motors and, through the use of a buck converter, 3.3 Volts to the Teensy 3.6.

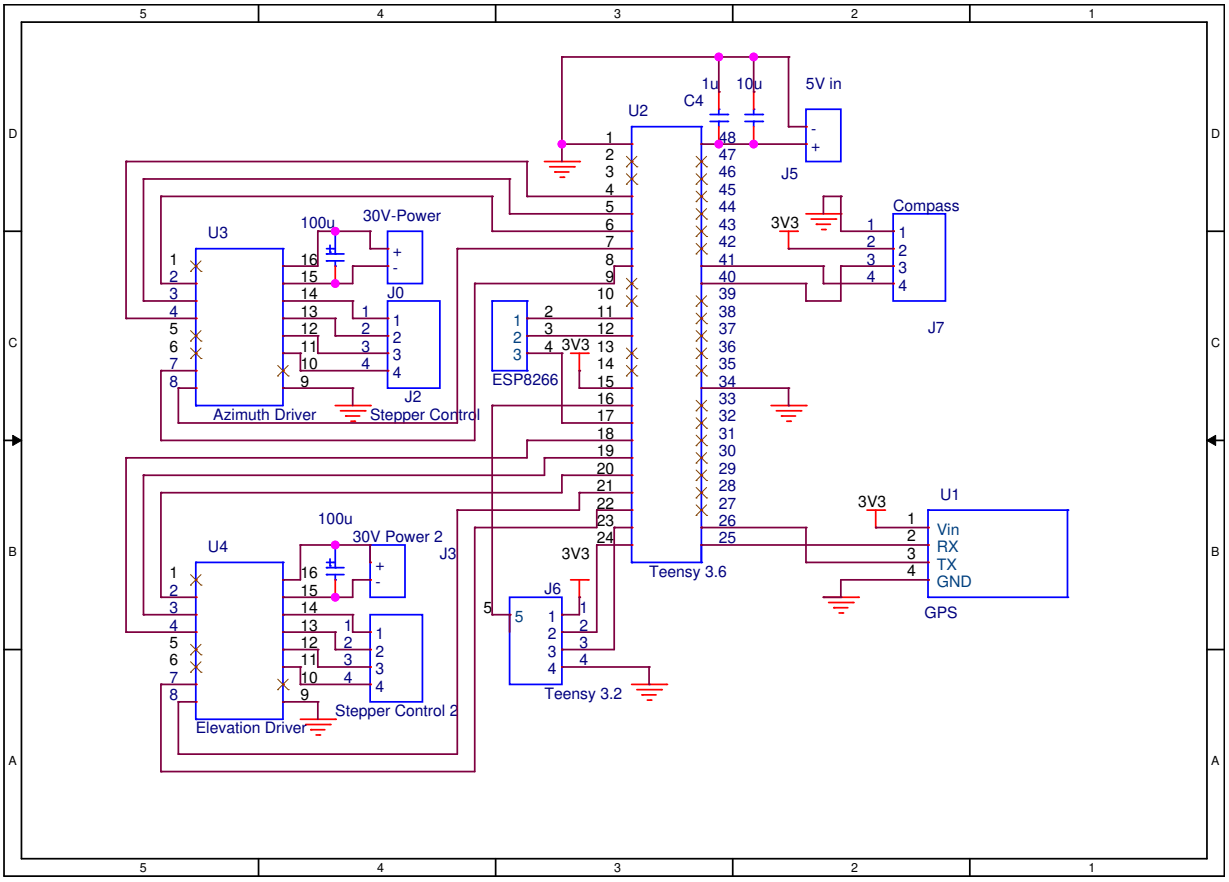
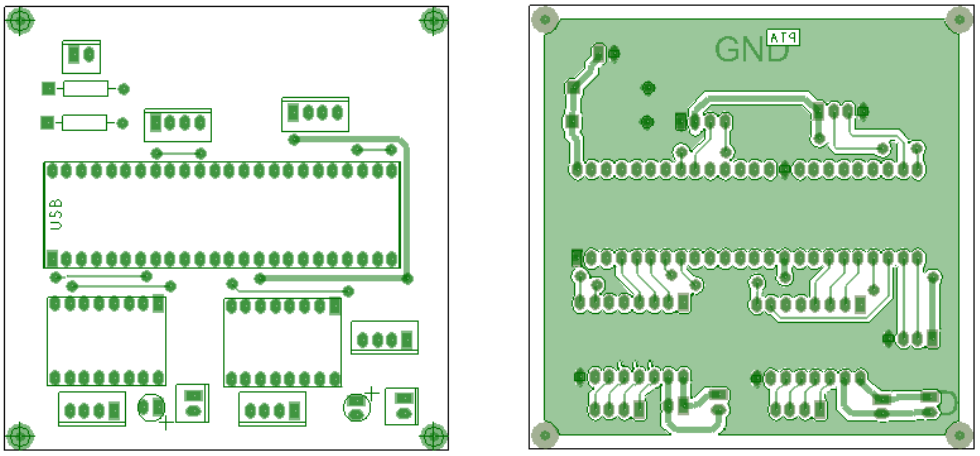


Figure 33: Control Board Schematic.



(a) Top (b) Bottom

Figure 34: Control PCB

6.2.7 Power Supply Module

This module was not implemented due to time constraints. The design stage was complete meaning that given more time this module can be developed.

6.2.8 Antenna Module

The antenna chosen was a high gain 17dBi, 2.4G WiFi grid parabolic antenna [3]. It has a horizontal beam width of 21 degrees and a vertical beam width of 16 degrees.

Transmitting and receiving modules interacting with the antenna also had to be chosen. After extensive research the XBee WiFi transmission modules were chosen as they have they the highest output power that can be found off the shelf, while still respecting UK regulations[2]. In later stages of the project these were changed due to their transmission rate being too slow. This decision was not made by PTA but was made by one of the other teams working with Police Scotland. Instead, a WiFi router and a raspberry pi module was used as a transmitter and receiver. Regardless of this not being the original solution, it was tested and a range of *330m* was attained. Another advantage of this system is that it allows for an omnidirectional antenna to be mounted to the router. This provides an area of around 40m in which the drone and the base station can connect, regardless of where the parabolic antenna is pointing.

6.3 Software

6.3.1 Software Systems Overview

Whilst official Arduino hardware was not used, the Arduino integrated development environment (IDE) was. Pre-existing libraries designed for Arduino compatible boards were readily available and large amounts of documentation and support exist aiding development.

The software duties were separated into two distinct systems, the control board micro-controller (MCU) which communicates with all systems and drives the motors, and the detector MCU which samples the optical detector and calculates the required differentials.

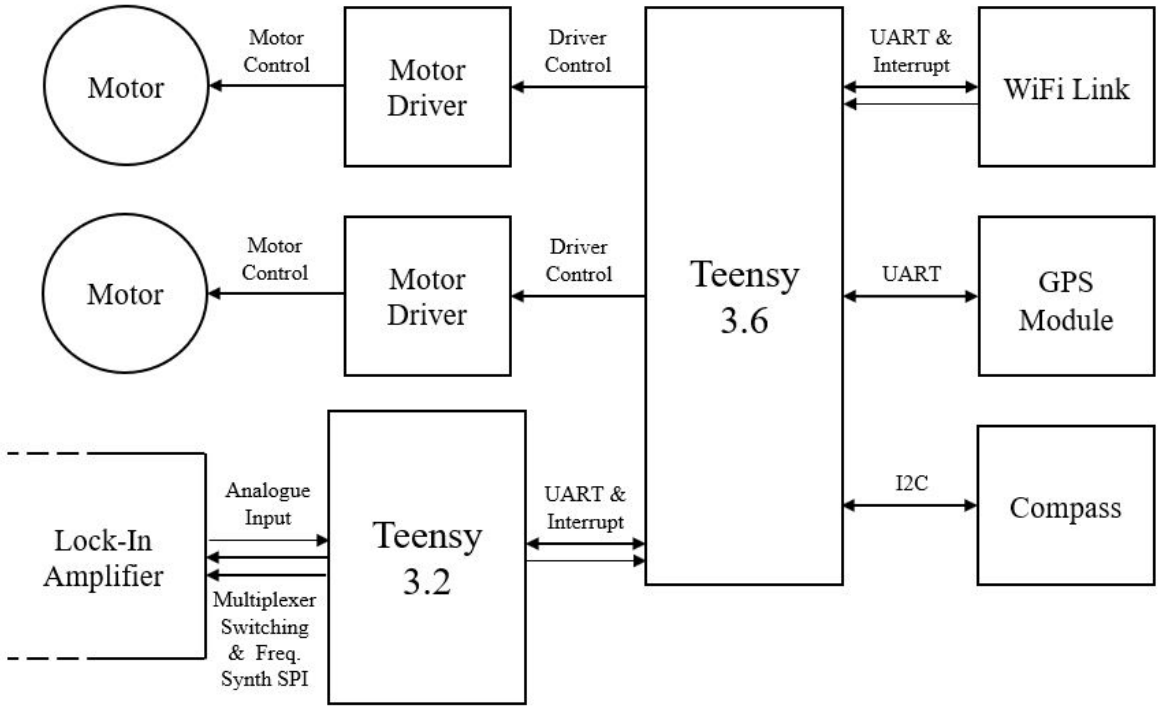


Figure 35: Software Systems Overview

6.3.2 Control Module Micro-controller

Due to time constraints both detection systems could not be integrated into one main program. As such two main programs run on the Teensy 3.6 depending on the current task.

GPS Program

As mentioned previously, key to the GPS sub system is the retrieval of NMEA data. Algorithm 1 below performs this operation and is called through a timer interrupt every second.

```
1  void baseGPSData() {
2
3    cli();
4    unsigned long start = millis();
5
6    do {
7      //If the serial port connected to the base GPS has data
8      while (baseGPSPort.available()) {
9
10       //Feed the gps.base object information via .encode
11       gpsBase.encode(baseGPSPort.read());
12       BASE_LAT = gpsBase.location.lat();      //Latitude?
13       BASE_LON = gpsBase.location.lng();      //Longitude?
14       BASE_ALT = gpsBase.altitude.meters();   //Altitude?
15       baseDataFlag = true;                    //data flag high
16
17     }
18   //Allow time for a full NMEA sentence to arrive
19   } while (millis() - start < 1000);
20
21   baseDataFlag = false;
22   sei();
23
24 }
```

Algorithm 1: Base GPS Data Retrieval

A library called "TinyGPS++" is utilised here for the parsing of the raw NMEA data. Line 9 shows the '.encode' function, which creates the sentences from the received chars, and checks that the sentences are valid. Whilst NMEA contains a multitude of information, it is only the GPGGA string that is required, and it is from this that lines 10, 11 and 12 pick out the latitude, longitude, and altitude of the base station.

Data is fed to the encoding function for a whole second, to allow complete sentences to be formed from the data. A *baseDataFlag* is returned with a boolean value depending on if the operation was successful or not. As the function handles shared global variables, cli(); and sei(); must be used at the start and end of the function to disable and re-enable further interrupts from accessing (and corrupting) the data.

Receiving GPS data from the drone requires Wi-Fi communications through the ESP8266 chip.

Algorithm 2 given below receives and formats the drone's GPS data, and is called every time the ESP8266 sets a 'data available' pin high.

```
1  void serialEvent() {
2
3    if (esp.available() > 0) {
4      BUFFER[j] = esp.read();
5      if (BUFFER[j] == 't') {
6        //If the string in the buffer terminates with a t it is the laTitude data
7        //Convert the current contents of the buffer into an int, save it and reset the buffer
8        long LAT = atoi(BUFFER);
9        DRONE_LAT = (double)LAT / 1000000;
10       bufferReset(BUFFER);
11       j = 0;
12
13     } else if (BUFFER[j] == 'n') {
14
15       ...same idea
16
17     } else if (BUFFER[j] == 'a') {
18
19       ...same idea
20
21     } else if (BUFFER[j] == '\n') {
22       //If a new line character is received trigger a data ready flag
23       j = 0;
24       flag = 1;
25     } else {
26       j++;
27     }
28   }
29 }
```

Algorithm 2: Drone GPS Data Retrieval

One consideration is that the ESP8266 may receive chars only. This means the code was required to recombine the individual chars into a string, then convert the sting into a float. Recombining the chars was no issue, as in *C* a string is simply an array of chars. However, recombining the chars into the distinct latitude, longitude, and altitude variables proved troublesome.

As implemented above, all characters are placed into a buffer array (line 4) at a position dictated by index 'j'. If a predesignated character is found (such as 't') the buffer is converted into an integer then saved into the corresponding container (line 9). This container now holds the correct data, however it must be cast into the correct coordinate format (line 10). The buffer and its index are reset (line 11, 12). If no predesignated character is found, the index increases and data is placed into the next place in the array (line 35). An example of a string received by the ESP8266 is below:

Received	12345678t	12345678n	20a
Translated	Lat 12.345678	Lon 1.2345678	Alt 20

From the above, the latitude, longitude and altitude coordinates are found. To calculate both our current and desired bearings we must call the *gpsBearing()* function.

```
1  void gpsBearing() {
2
3    //Calculates the distance to the drone station. In implementation drone_lat/drone_lon will
       be recieved in an interrupt
4    unsigned long distanceKmToBase =
5    (unsigned long)TinyGPSPlus::distanceBetween(BASE_LAT, BASE_LON, DRONE_LAT, DRONE_LON);
6
7    //Calculates the initial bearing that would lead to the drone
8    double courseToBase =
9      TinyGPSPlus::courseTo(BASE_LAT, BASE_LON, DRONE_LAT, DRONE_LON);
10
11   //Calculates the vertical bearing
12   double altitudeDifference = BASE_ALT - DRONE_ALT;
13   float verticalBearing = (atan(altitudeDifference / distanceKmToBase)) * 180 / PI;
14
15   //Calculates the horizontal bearing
16   mag.readMag(&x, &y, &z);
17
18   //Calculates the horizontal error...
19   float horizontalError = (courseToBase - (360 - magHeading));
20   //...and moves the rig by it
21   azimuthRotate(horizontalError);
22
23   flag = 0;
24
25 }
```

Algorithm 3: GPS Bearing Calculations

Again the TinyGPS++ library is used to find the distance and bearing between the base station and the drone on lines 5-8 and 11-14.

The method for calculating the vertical bearing, as laid out in the design section of the report, is implemented lines 17-18.

To read the compass bearing, the "SparkFun MAG3110" library was used. When testing the magnetometer on the Arduino Uno test MCU, the bearing was found without a hitch. When ported to the Teensy 3.6 however seemingly nonsensical results were found. The magnetometer has a 16-bit architecture and sends signed integers of the following form:

1000 1010 1101 0000

This number represents -30000 using 2's complement, however, if not explicitly defined, integers are as long as the CPU's architecture defines, and as such the 32-bit Teensy saves the value as follows:

0000 0000 0000 0000 1000 1010 1101 0000

When converted using 2's compliment this is equal to 35536. To fix this bug, the library had to be edited to explicitly state that all integers are signed 16 bits wide (int16_t), removing the 16 empty bits at the start of the number, and allowing the correct bearing value to be found.

After some processing both the current and desired azimuth bearing are compared to calculate the error by which the azimuth controller must move (line 36). This angle is sent through *azimuthRotate()* which simply scales the value by the gear ratio between pinion and belt. A flow diagram illustrating the GPS program is given in Figure 36.

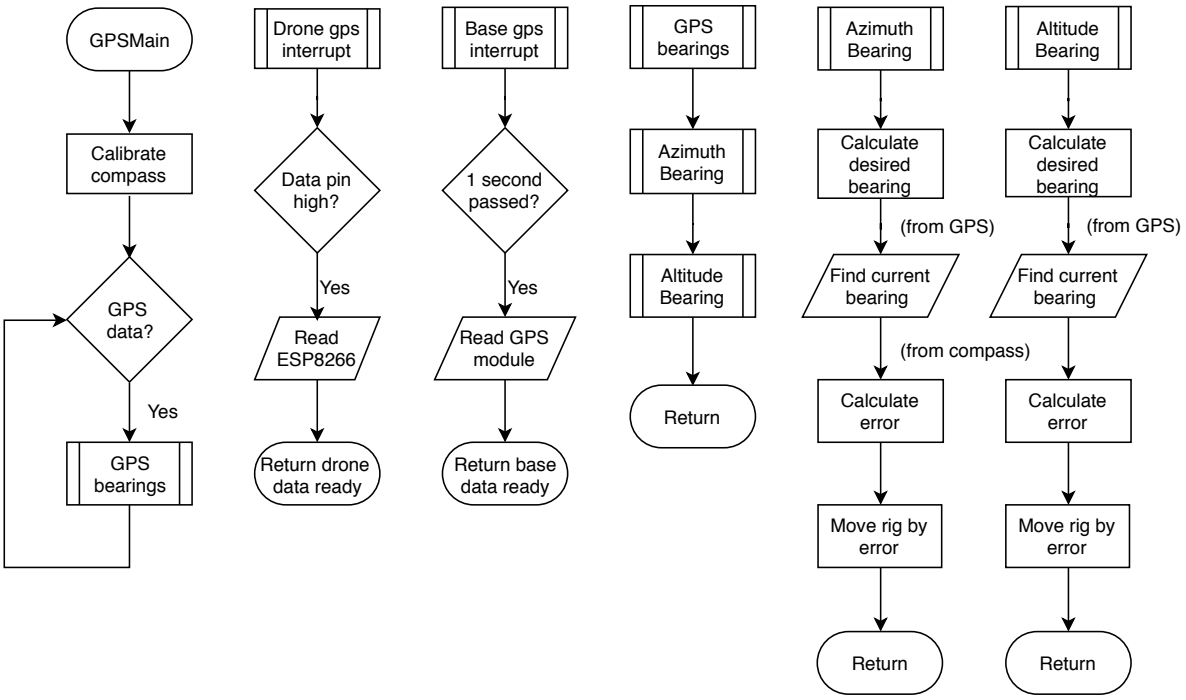


Figure 36: GPS program flow diagram.

Optical Tracking Program

The main program for the optical detection system is somewhat simpler thanks to the reduced number of peripherals that it must communicate with.

Algorithm 4 is triggered by the Teensy 3.2 every time data is available.

```
1  void detectorData() {
2
3    static byte ndx = 0;        //Recieved char index
4    char endMarker = '\n';      //endMarker = newline char
5    char rc;                    //Recieved char
6
7    //Is there data on the detector UART port?
8    if (detectorUART.available() > 0) {
9      //If there is read the port and save it into rc
10     rc = detectorUART.read();
11
12     if (rc != endMarker) {          //If we have not reached the end of the sentence
13       receivedChars[ndx] = rc;    // then fill the current position of the array with rc
14       ndx++;                      //Increment ndx ready for the next char
15       if (ndx >= numChars) {      //If the index is greater than or equal to the predefined
             size of the char array
16         ndx = numChars - 1;       //Set the index to (size of array - 1)
17       }
18     } else {
19       receivedChars[ndx] = '\0';    //Otherwise terminate the array
20       ndx = 0;                      //Reset the counter
21
22       //Convert the received char array into an integer
23       differential = atoi(receivedChars);
24       Serial.println(differential);
25     }
26   }
27 }
```

Algorithm 4: Optical Data Retrieval

This is similar to the WiFi serial event detailed in the GPS section. Here, if data is available (line 7) it is read and saved to 'rc' (line 8). If this received character is not a new line character, then it is saved into the current index in the character array (line 11).

If the current index is greater than or equal to the size of the array, then the index is reduced by 1 (line 14) so that the array does not overflow before a new line is found. When it is, the terminator character is placed into the array and the index is reset (line 17-18). Finally the string is cast into an integer and saved as the detectors 'differential' value.

This differential value is used to drive the motors through the control algorithm, a snippet of which is given below.

```
1  //If at the centre do not move
2  if ((differential > 0) && (differential < 30)) {
```

```
3     digitalWrite(MODE2, 0);
4     digitalWrite(MODE1, 0);
5     digitalWrite(MODE0, 0);
6
7     //Tell motor to stop moving
8     stepper.disable();
9   }
10
11  //If at the far left
12  else if (differential > 400) {
13    //Set to full step mode
14    digitalWrite(MODE2, 1);
15    digitalWrite(MODE1, 0);
16    digitalWrite(MODE0, 0);
17
18    //Tell motor to rotate by 5 steps * scaler
19    stepper.enable();
20    azimuthRotate(5);
21  }
```

Algorithm 5: Differential Motor Control

Depending on the differentials magnitude and sign, the motors step mode (line 3-5, 14-16) and the amount of steps is makes (line 8, 19-20) is set. In the above, if the differential is in the 'zero' band, the motor is disabled. If the differential is greater than 400, this indicates a strong signal is the left detector, requiring a large movement to the left accomplished by a full step mode. The flow diagram of this program is found in Figure 37.

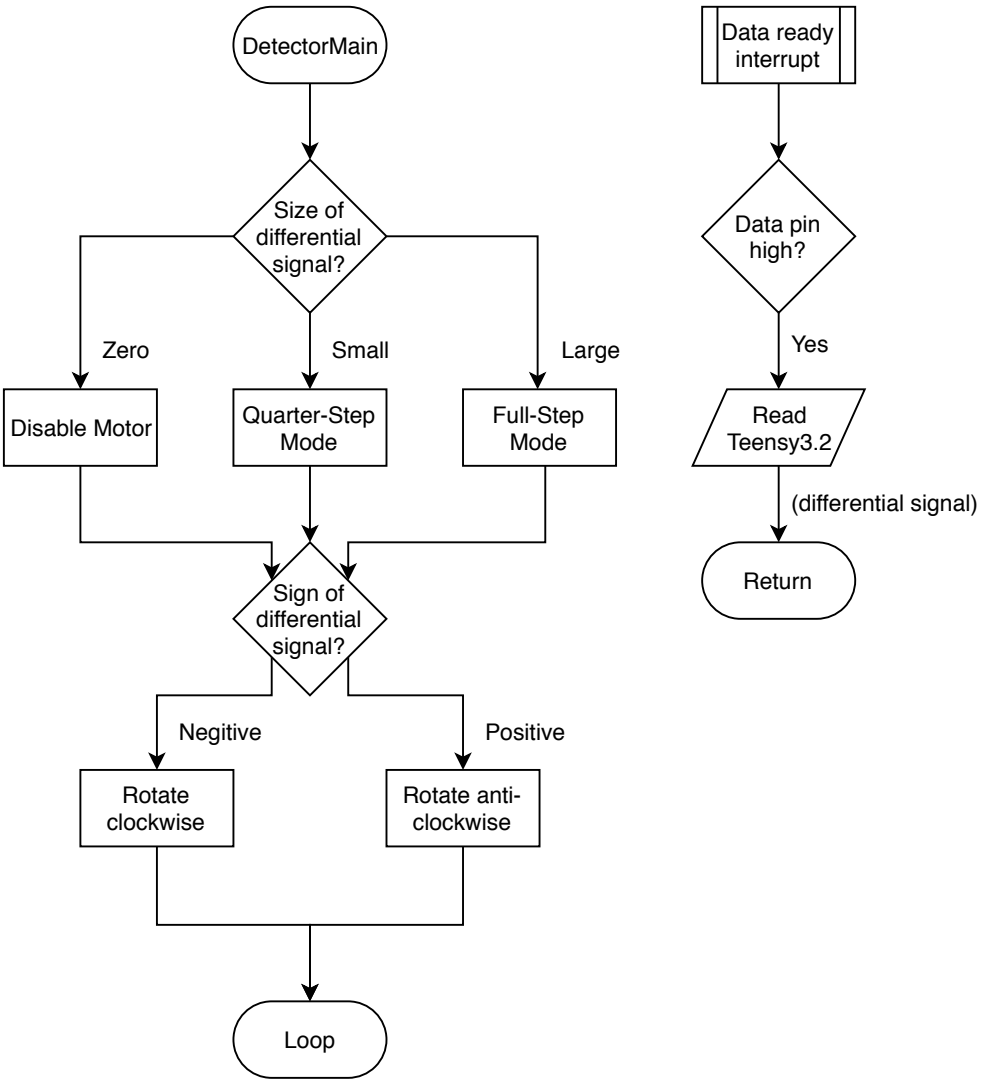


Figure 37: Optical tracking program flow diagram.

6.3.3 Lock-in Amplifier Module Micro-controller

Central to the detector software is the sampling of the lock-in amplifier. Whilst a simple command such as *analogRead()* could be used, its resolution and speed cannot be controlled. Instead, the hardware ADC is directly manipulated.

```
adc->setAveraging(16);
adc->setResolution(12);
adc->setConversionSpeed(ADC_CONVERSION_SPEED::HIGH_SPEED);
adc->setSamplingSpeed(ADC_SAMPLING_SPEED::HIGH_SPEED);
```

Algorithm 6: ADC Setup

An advantage of using the Teensy 3.2 is it's ability to average a defined previous number of samples and return the result. This is set up on line 1, and gives a slight low-pass filter characteristic to the sampling operation. This is advantageous as it lessens noise present in the signal, especially as the resolution is quite high at 12 bits (line 2). From the chip manufacturer, the max ADC sampling and conversion speed recommended for a resolution of 12 is "HIGH" which is set on lines 3 and 4.

Once the ADC is setup, the sampling is preformed by *"timer0 callback()"* every 500μs.

```
1 //Begins reading the ADC, preforms 16 sample averaging
2 void timer0_callback(void) {
3   adc->startSingleRead(adcInput, ADC_0);
4 }
```

Algorithm 7: ADC Sampling

To change the detector circuit being sampled the *"muxSwitch()"* function is called every 50ms. The function changes the active buffer accordingly and also calculates maximums as well as differentials when appropriate.

```
1  void muxSwitch() {
2    rubbishFlag = 1;                                //Set the rubbish buffer to receive data
3
4    switch (muxIndex) {
5
6      case 0:
7
8        detectorMux(1);                           //Switch to detector 1
9        left4diff = maximum(leftBufferArray);     //Find max of leftBuffer
10
11       rightFlag = 1;                      //Set the right array to collect data
12       leftFlag = 0;                       //...and disable the left array
13       muxIndex++;                         //Force the next switch statement to run next call
14
15     case 1:
16
17       detectorMux(2);                          //Switch to detector 2
18       right4diff = maximum(rightBufferArray);  //Find max of rightBuffer
19       aziDiff = left4diff - right4diff;        //Calculate the azimuth differential
20
21       rightFlag = 0;                      //Disable the right array
22       leftFlag = 1;                       //...endable the left to collect data
23       diffFlag = 1;                       //...and notify the main loop data is ready
24       muxIndex = 0;                       //Reset the switch index
25   }
26 }
```

Algorithm 8: Detector Multiplexer Switching

In the times between the function calls, sample data is placed in an array. Every 50ms, Algorithm 8 changes the active array (line 11-12, 21-22), and attributes the maximum value of the last 50ms sampling period to whichever detector was last active (line 9 and 18). When the function reaches the last detector, the difference is found (line 19) and the main loop is instructed that data is ready to be sent (line 23). The method by which the maximum value of the array is found can be found in Algorithm 10 in the appendix.

In Algorithm 8 there are two detectors being sampled and three arrays being populated: 'rubbish', 'left' and 'right'. The 'rubbish' buffer simply throws away a portion of the received samples after switching. This is to account for the propagation delay of the signal through the lock-in amplifier as after switching, signal is still momentarily received from the previous detector.

This process of attributing samples to detectors (and rubbish heaps) is carried out in the code below, which is called after each sample operation is completed by the ADC.

```
//Called when sampling is complete
void adc0_isr() {

    //Place sample into buffer0
    int16_t buffer0 = adc->readSingle();

    if (rubbishFlag == 1) {
      if (rubbishBufferIndex < (rubbishArraySize - 1)) {
        rubbishBufferArray[rubbishBufferIndex] = buffer0;
      }
      rubbishBufferIndex = 0;
      rubbishFlag = 0;
    }

    if (leftFlag == 1) {
      if (leftBufferIndex == (leftArraySize - 1)) {
        leftBufferArray[leftBufferIndex] = buffer0;
        leftBufferIndex = 0;
      } else {
        leftBufferArray[leftBufferIndex] = buffer0;
        leftBufferIndex++;
      }
    }
}
```

Algorithm 9: ADC Sample Saving

The calculated sample is stored in a buffer, and then depending on the flag values set by *muxSwitch()* it is placed into one of 3 arrays. The rubbish buffer is always triggered at the start of each new detector sampling period, and consumes 30 out of 100 samples taken per sampling period. After this the *detectorArray* is populated until full.

Due to time constraints, automatic gain control was not able to be implemented as requested by the design. Instead, manual adjustments are made through the functions below.

```
inaGainMux(inaGain);
outputGainMux(outputGain);
```

Controlling the AD9833 frequency synthesizer chip on the lock-in board required on chip registers to be written to. Example code created by the chip manufacturer may be found in Algorithm 12 in the appendix.

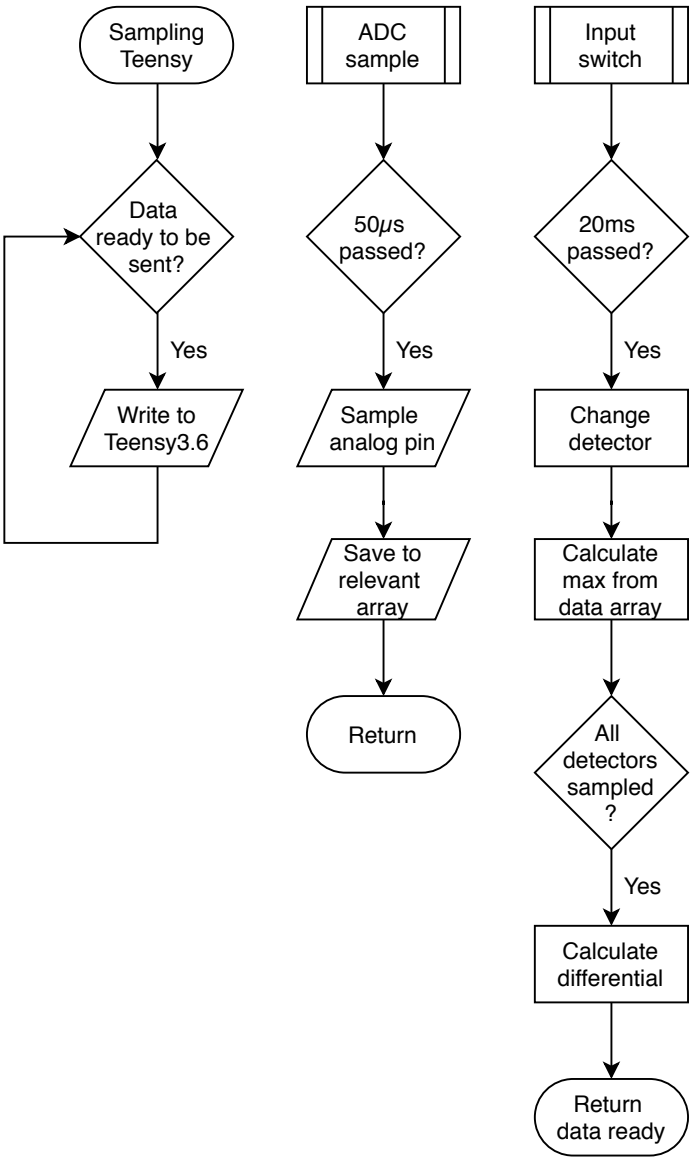


Figure 38: Lock-in amplifier program flow diagram.

7 Testing and Results

Testing comprised of development tests, functionality tests and field tests. Results of each test are presented and discussed under each individual test. Development tests were adopted due to iterative design thus they consisted out of experiments done on each iteration of the hardware and software to support the development. These will not be described further as they mainly consisted out of debugging and measurements discussed during design. Functionality tests were used to confirm expected operation of final versions of subsystems. Field tests were performed to validate full operation of the whole tracking system under real operating conditions. Note that all field tests where optical detectors were used were performed only with a pair of detectors able to drive the azimuth axis only due to only two lenses being available for our use. Nevertheless, the same process is used to drive the elevation axis with additional lenses hence testing with only azimuth is sufficient to validate the design.

7.1 Functionality Tests

Functionality of all the main modules was tested to assure reliable and correct operation before integration with other modules. Furthermore, the performance of individual modules developed was measured during these tests. This included: optical range testing, optical detector dynamic and static characterisation, lock-in amplifier performance testing, base GPS testing, antenna range testing, antenna switching testing, and mechanical assembly azimuth/elevation control testing.

7.1.1 Optical Range Testing

The purpose of this test was to measure the range of the optical detector interfaced with the developed lock-in amplifier. The test setup consisted out of a stationary light beacon positioned at a certain distance from a single detector. On the detector, the amplitude of the received signal was inspected using an oscilloscope and the signal was sampled to be recorded. By visual inspection the signal was assessed to determine the presence of noise at a given distance and whether the signal was usable. This test was performed multiple times with increasing distances and throughout the development cycle as the detectors were improved. In practise, due to limited testing space with clear line of sight in the city landscape, a continuous distance sweep was impossible to perform and thus only specific distances were used during testing.

Distances of 30m (University of Glasgow Hunter Halls), 110m (University of Glasgow Cloisters and Quadrants), 250m (Rankine Building roof to Kelvingrove Park) and 330m (Rankine Building roof to Kelvingrove Park) were successfully tested with the signal being fully received. Figure 39 indicates the test range used for the 330m test. The lock-in amplifier gain settings used at the largest tested range (330m) were $G_1 = 7.7$ and $G_2 = 909$ and the signals recorded had peak-to-peak value estimated to be equal to full range of the ADC of $3.3V$. This means that the lock-in was successfully capturing a signal with a magnitude of around $240\mu V$ with the gain set to about 7000. This maximum gain available within the lock in is in excess of $9,000,000$. Hence it can be inferred from this test that the optical detection system should be capable of reaching the 500m range specification. This is a motivation for further range testing in the future, however suitable test ranges must be identified.

Figure 39: 330m range test from Rankine Building roof to Kelvingrove Park flag pole.

7.1.2 Optical Detector Dynamic and Static Characterisation

The purpose of this test was to quantify characteristics of the optical detection module, including the dynamic response, differential output, FOV, alignment, and lens performance. Note that this testing was performed with a set of two optical detectors with the digitally processed differential signal as the output.

Firstly, the dynamic response of the optical detection was measured by fixing the detector to a static position and moving the light beacon at a constant speed $\approx 1.4m/s$ across the full detector FOV. The distance between the optical detector and the light beacon was $30m$ and the distance

moved was $6m$. The results (Figure 40 and Figure 41) show differential signals which were clearly able to detect the moving light. Three major conclusion were drawn from this data - the response time of the detector is sufficient to track a moving target, the detector is highly directional with the FOV estimated to be 7°, and within this FOV sharp signal peaking is observed. Thus, it became clear that the sensitivity of the detector is a function of the light position in space which can be approximated by a highly narrow and peaking Gaussian. This nature of the signal makes the alignment of the lenses with the photo-diode and the alignment of the differential optical detectors extremely sensitive, creating uncertainty in the observed peak which is the reason why the peaks magnitudes recorded vary between around 800 to 1500. Hence further measurements to quantify the detectors Gaussian response as a function of spatial position, and a versatile model of the signal are needed in future. Additionally, design improvements focusing on minimising the misalignment errors and achieving precise alignment must be implemented.

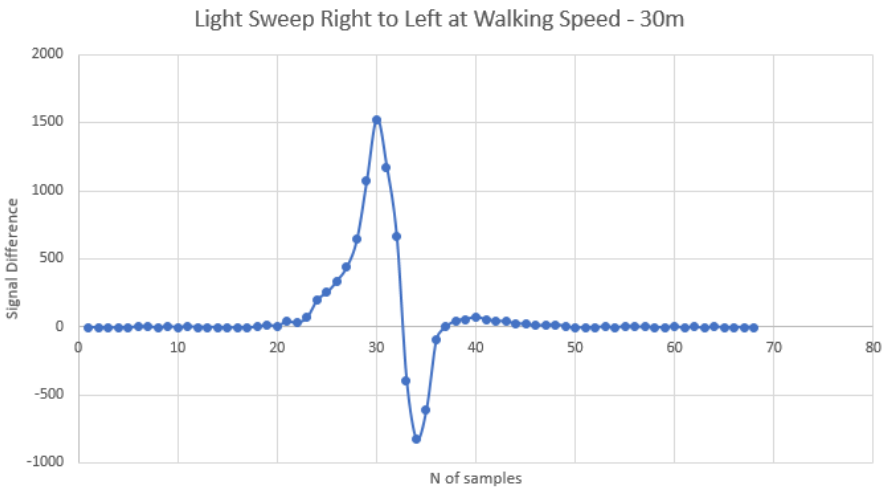


Figure 40: Dynamic response of differential pair optical module as light is swept through detectors field of view form right to left.

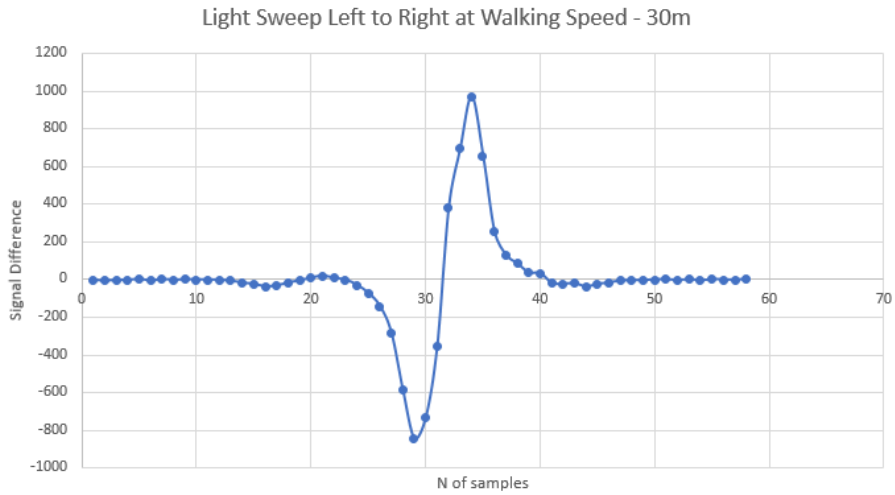


Figure 41: Dynamic response of differential pair optical module as light is swept through detectors field of view form left to right.

A secondary test was performed where the light beacon was moved to predefined positions in the detectors field of view and kept at these positions stationary for 30*sec*. The distance between the optical detector and the light beacon was 15*m* and the set positions were: centre 30s, 2*m* left 30s, centre 30s, 2*m* right 30s, centre 30s. This test was performed with glass lenses and subsequently with the fresnel lenses for comparison. From the results, (Figure 42 and Figure 43) the misalignment uncertainty had a clear effect on the recorded peak values which differed across the two detectors. As can be seen from results of both types of lenses, at stationary discrete positions a clear and stable difference from the centre positions 0 differential signal was observed but the magnitude varies greatly. Furthermore, a periodic variation in the output differential signal superimposed on the desired response is observed. After analysis of this data, it became apparent that this is due to an error in the input multiplexing, splitting of the sampled data and the finite delay of the lock-in amplifier response. When this delay is quantified imprecisely, splitting of data causes the initial portion of data to be attributed to the wrong detector at every split which results in a periodic differential signal output. To mitigate this issue the delay presented in the response is to be quantified more precisely. Most importantly, collection of data showing distinct positions of the light beacon was still achieved.

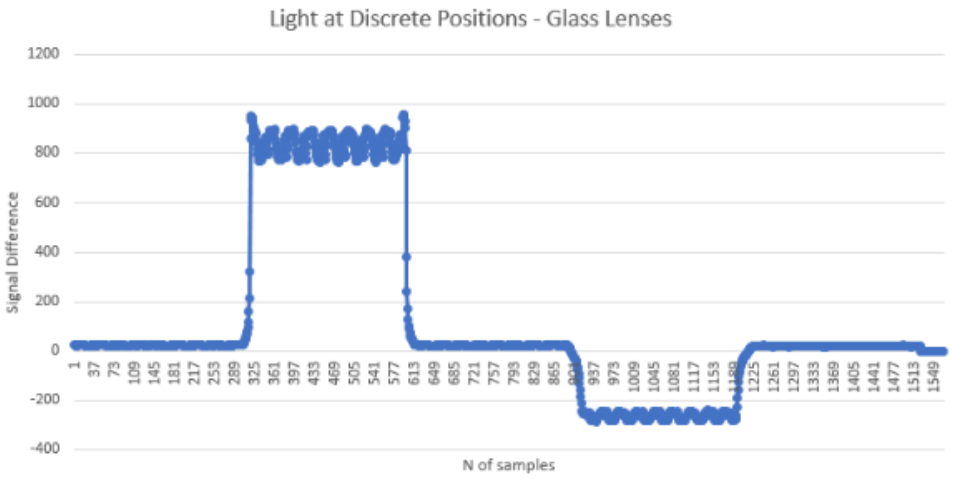


Figure 42: Data set of differential signals when using glass lenses from light positioned at discrete positions in detectors field of view.

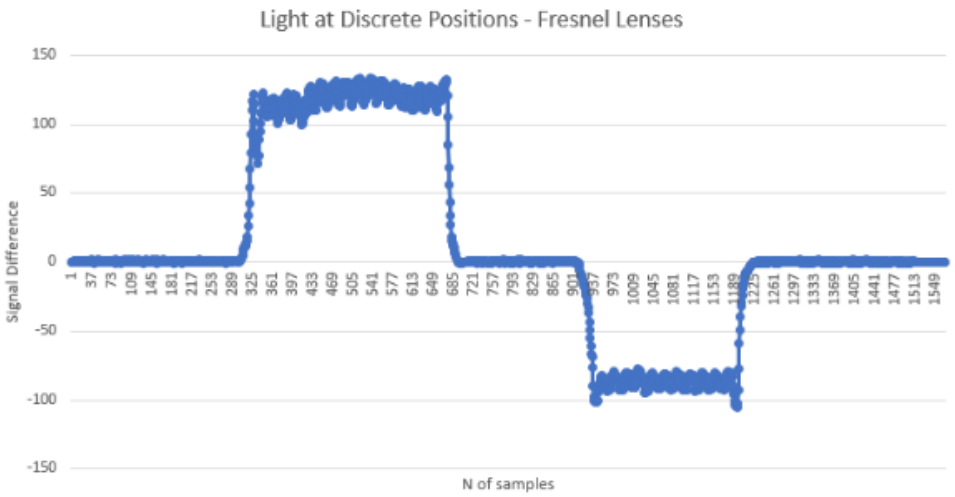


Figure 43: Data set of differential signals when using Fresnel lenses from light positioned at discrete positions in detectors field of view.

Additionally, a clear distinction between the performance of the lenses was observed. The differential signal from the fresnel lenses was recorded to be an order magnitude less than their glass counterparts. Due to this large difference, at large ranges using fresnel lenses will not suffice and thus their use was discouraged.

7.1.3 Lock-In Amplifier Performance Testing

The testing of the lock-in amplifier to determine its performance was done by measuring its resolution using data from the optical detector characterisation as well as optical range testing data.

During the design, the lock-in amplifier gains were designed to be able to amplify a minimum signal of $50nV$, given ideal zero noise floor. At the largest range test of 330m, a signal with magnitude of $3.3V$ with insignificant noise was sampled on the ADC input, while using cascaded gains $G_1 = 7.7$ and $G_2 = 909$. Thus, the smallest tested signal was amplified by an overall gain of 6999.3 which equates to a signal $471\mu V$ which is considered to be the stable and tested resolution of the lock-in amplifier. However further tests should be performed until $SNR \approx 1$ is reached which defines the true minimal resolution of the instrument.

7.1.4 Antenna Range Testing

The long range directional antenna was tested in the same location as the optical range test (Figure 39). It was tested with the team that was working on the WiFi router, which acted as a TX and RX module at the ground station. The test consisted of having one team with the antenna on the roof and another team with a smart-phone, connected to the router. The smart-phone team had an app downloaded on their phone as to monitor the strength of the connection.

The maximum distance reached was 330m due to lack of space available. At 330m a connection of $43Mbps$ with a latency of $5ms$ was achieved. This is a very strong connection and the team is confident that a greater distance could be reached if a larger test range could be found. Furthermore, the dedicated WiFi receiving peripheral on the drone is better at receiving a signal than the standard module found in smart-phones.

7.1.5 Antenna Switching Testing

This test was also conducted with the team working on the WiFi router, as the router was introduced as to allow switching between the omni-directional antenna and the parabolic main antenna. This test was conducted the same day the large scale test was conducted.

The test consisted of moving around the router/antenna system with a smart-phone and then moving out of the range of the omni-directional antenna but still within the beam width of the directional

antenna and ensuring that the connection is constant. This test confirmed the viability of antenna switching at close range.

7.1.6 Mechanical Assembly Azimuth/Elevation Controller Testing

During horizontal actuation testing the assembly was made to turn 90°, 180° and 360°. In each case a video was recorded to allow more precise measurements. Video analysis for all three cases allowed an average angular velocity to be calculated. This process determined the angular velocity to be $46.67°/s$, or $7.79RPM$. The maximal angular velocity will be used when the drone being tracked is moving at the maximum velocity at short ranges of the directional antenna. The shortest range is set at $30m$. At this distance if the drone flies in a circle around the assembly the maximum allowable drone velocity is around 24m/s. This is dramatically more than the quoted 2m/s given in the initial specification.

The vertical actuation test was performed similarly to the horizontal one. The angle used for testing was 60 degrees. Video analysis was utilised again to ensure accurate and reliable test results. An angular velocity of 28.57 degrees/s was found. This evaluates the allowable velocity for the drone flying upwards in distance of 30m to be around 15m/s which is once again much more than required.

The mechanical assembly passed the angular velocity tests, the functionality is not limited and was proven to work meaning that tracking the drone should be reliable. Despite this fact, angular velocity generated and angular position reached suffer from inaccuracies due to structural instability and lack of structural stiffness of the mechanical assembly. Additionally, the inertia of the system is large so a more ridged assembly and possibly stronger motors capable of handling the resultant torques are could be implemented in future.

7.2 Field Tests

Three main tests of all the modules integrated together were performed. Testing of the optical tracking and GPS tracking were performed individually. Optical tracking was only tested in the azimuth axis as only one pair of glass lenses were available.

7.2.1 Short Range Tracking Field Test

This test was performed indoors at a distance of $15m$ from the detector. Tracking of the light in the azimuth axis was successfully performed in both directions. The light source was swiped before the tracker performing a 90° arc at a quick walking speed ($\approx 2m/s$) and the tracker followed the beacon with minor difficulty. Due to the detectors being highly directional, imprecisely aligned, and the use of only azimuth tracking, difficulties arose when the light was not directly pointed into the detectors at the same height from the ground. The detector misalignment caused a different control response of the tracker as the light was moved from left to right and then backwards left to right. The results can be found in the Appendices section 2 (Data). In the latter, the tracking performance was worse as the zero point was not tracked as well, and the response suffered from spikes because of the wrong relative alignment between the two lenses (Figure 44). This should be resolved by adding the elevation axis into the system to minimise these effects. For robust operation, improvements in the stiffness of the mechanical assembly, control system and alignment of the sensors are needed.

Ultimately, this proved that all of the optical modules developed were integrated successfully with the software and mechanical assembly and that the controllers handled the optical data correctly.

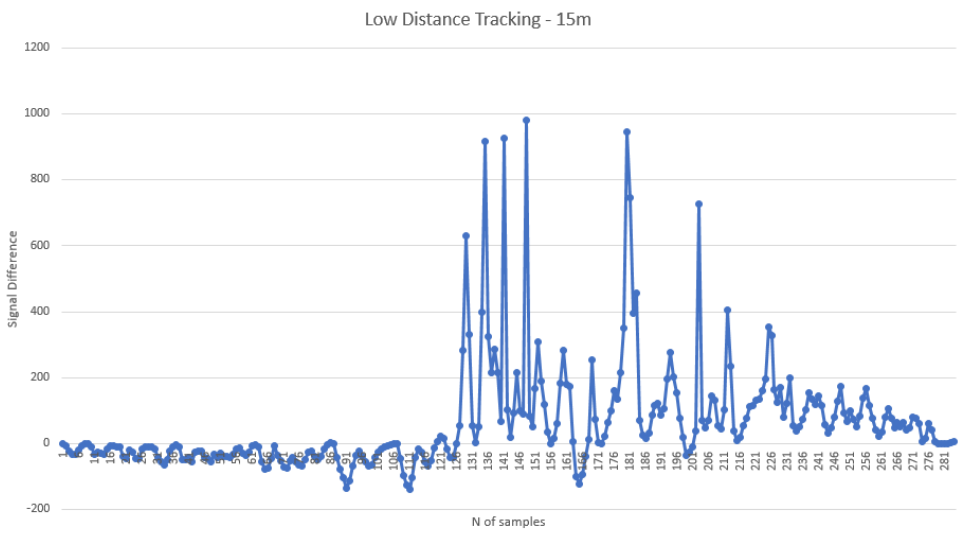


Figure 44: Differential Signals Moving Left to Right then Right to Left.

7.2.2 Long Range Tracking Field Test

A full range test was performed outside in real operating conditions, but issues were encountered. This test was performed under direct sunlight at a location with white surroundings. The issues were

however troubleshooted fully due to lack of time, and further testing under these conditions is required to draw conclusions.

7.2.3 GPS Tracking Test

To test the functionality of the GPS localisation system, simulated GPS data was sent by the WiFi team to actuate the system. This experiment was divided in to three parts: base GPS data collection, drone communication testing, and horizontal/vertical control.

Base GPS Testing

To test the accuracy of the base GPS data, tests were carried out at the two locations given in Figure 45.

	54 Henderson Street			74 Oakfield Avenue		
	Latitide	Longitude	Altitude	Latitide	Longitude	Altitude
Real	55.876738	-4.273595	33.000000	55.872631	-4.2859601	35.000000
	55.876777	-4.273671	33.600000	55.872661	-4.285974	9.000000
	55.876777	-4.273669	34.100000	55.872665	-4.285979	8.700000
Measured	55.876777	-4.273667	34.000000	55.872665	-4.285984	8.500000
	55.876781	-4.273666	34.100000	55.872669	-4.285988	7.300000
	55.876781	-4.273664	34.400000	55.872673	-4.285997	6.000000
Average	55.876779	-4.273667	34.040000	55.872667	-4.285984	7.900000

Figure 45: GPS Positions as Found by Base GPS Receiver

By plotting and comparing the averaged measured data and the real positional data given in Figure 45, we may see that the error in the position is 9m and 4m respectively. Discrepancies in the altitude position data are far larger however reaching around 25m for the data gathered at 74 Oakfield Avenue. Although not perfect this is within defined acceptable errors for GPS altitude data [9]. Full GPS logs for both locations can be found in the Appendices section 2.

Drone Communication Testing

This test consisted of sending simulated data in a particular format to our system through WiFi and subsequently converting the data to coordinates. This system was implemented and worked reliably, proving the Wifi GPS communication successful. Unfortunately it was not possible to test this with the control software due lack of time for testing with the team responsible for the WiFi

communication. This is a test of full operation and should be performed in the future.

Combined Horizontal and Vertical Testing

In this test the system's ability to point at coordinates was tested. The main program was given the home coordinates shown below, and was sent locations to move to.

	Base	Main Tower	Library	James Watt
Latitude	55.872217	55.871157	55.873216	55.871205
Longitude	-4.2858481	-4.2886805	-4.2890882	-4.2863738
Altitude	36	100	75	50

Due to the magnetometers failure prior to the test, active error feedback was not possible. As such the station's azimuth was moved to point at the first target (the Main Tower given above), its elevation was set to zero degrees.

Once this calibration was complete the controller calculated bearings, shown below. These show the required change in angle to move from the current target to the next target. The rig was actuated and pointed correctly at the targets.

Target	Main tower	Library	James Watt
Change in Azimuth (°)	0	62.42	-102.54
Change in Altitude (°)	16.80	-7.17	-2.80

In reality the system turned by 61 degrees when turning towards the Library and by -106 degrees when turning towards the James Watt. Therefore the error observed is approximately 4 degrees. This is not a significant issue as the beam width of the antenna in the horizontal direction is 22 degrees. In terms of vertical control, an accuracy of 1 degree was achieved. Since the beam width of the antenna in the vertical direction is around 16 degrees, robust operation can be assumed. In general the control of the mechanical assembly had sufficient accuracy to maintain connection with the drone assuming correct calibration. In future tests a magnetometer should be utilised to provide a feedback to both the azimuth and elevation control and therefore avoid the need for calibration.

8 Conclusions

In conclusion this project has been a success as all major specifications that were outlined at the beginning of the project have been met.

All systems were designed, developed and integrated proving that optical tracking is effective and applicable to Police Scotland's needs. However, for a fully finalised and robust tracking system, more time and appropriate test locations are needed in order to fully validate the product. Nevertheless, the team is confident that further tests would be successful.

8.1 Further Improvements

The system was able to meet specifications but through testing a number of possible improvements were identified. Firstly, the base platform would be re-designed so that it's wider. Since the original base was designed for a smaller system, the current design oscillates a significant amount during tracking. Secondly, a wider bearing would be used to hold the top platform so that the oscillations would be further reduced. Encoders or limit switches would also be added to the system to provide absolute orientation measurements, which could be utilised to form a more robust control framework. The two fresnel lenses in the current prototype must also be replaced with glass plano-convex lenses for the next iteration of the system.

In terms of the electronics, further measures should be taken to protect the subsystems from damage and increase system robustness. This would include weather-proofing of the optics to ensure Police Scotland are not limited in their operations. Additionally, the design included automatic gain switching within the lock-in amplifier. While this was implemented in the hardware there was insufficient time to tune the relevant software.

With implementation of the improvements outlined above, the designed system would be ready for another set of tests and ultimately use in search and rescue operations.

Figure 46: Final Build

9 Bill of Materials

Module	Quantity	Description	Supplier	Unit Cost	Total	
Detector Module (Values here enough to build 4 as required)	24	0.33uF Cap	RS	0.206	4.944	
	16	1uF Cap	RS	1.032	16.512	
	16	10nF Cap	RS	0.206	3.296	
	4	1nF Cap	RS	0.284	1.136	
	4	4.7K Res	In stores	0.03	0.12	
	4	15K Res	In stores	0.03	0.12	
	4	178 Ohm Res	Farnell	0.047	0.188	
	4	20K Ohm Res	Mouser	0.076	0.304	
	4	30K Ohm Res	Mouser	0.145	0.58	
	8	2M Ohms Res	Mouser	0.129	1.032	
	4	100 Ohms Res	Farnell	0.0782	0.3128	
	4	68.1K Ohms Res	Farnell	0.0081	0.0324	
	4	300K Ohms Res	Mouser	0.076	0.304	
	4	BPW 21 Photodiode	RS	6.78	27.12	
	4	MPSA18 Transistor	Mouser	0.664	2.656	
	4	ADA4637-1ARZ	RS	6.05	24.2	
	4	BFG520/X NPN Transistor	RS	0.128	0.512	
	4	MC78M15CDTRKG 15V Lin. Reg	Mouser	0.321	1.284	
	4	MC79M15CDTRKG -15V Lin. Reg	Mouser	0.419	1.676	
	4	OP27	RS	3.09	12.36	
	4	9-way Male D-type	In stores	0.092	0.368	
	4	9-way Female D-type	In stores	0.11	0.44	
	4	SMA Bulkhead	RS	4.55	18.2	
		SMA Cable	Professor Weaver	0	0	
	4	SMA Crimp Termination	RS	2.4	9.6	
	4	Aluminium enclosure 92 x 38 x 31mm	RS	4.09	16.36	Total: 143.6572
Mechanical Module	1	4 Hole Flanged Bearing	RS	7.11	7.11	
	1	7mm modified bore, 2 off 2mm grubscrew holes	Belting Online	7.62	7.62	
	1	Timing belt 800mm length, 9mm width	Belting Online	4.68	4.68	
	2	UEETEK Nema 23 Stepper Motor 2.8A 178.5oz.in/1.26Nm	Amazon	23.99	47.98	
	1	Aluminium Extrusion Profile T-slot 20mm x 20mm	Amazon	23	23	
		T-slot	lab		0	
		Laser cut gear from Perspex sheet	MR EWAN RUSSELL		0	
		Laser cut sheet for azimuth motor, Perspex sheet	MR EWAN RUSSELL		0	
	4	Postal tubes 1250mm x 152mm	Kite Packaging	4.06	16.24	Total: 106.63
Control Module	1	Teensy 3.6 Microcontroller	Hobbytronics	24.95	24.95	
	1	Triple axis magnometer breakout - MAG3110	hobbytronics	11.4	11.4	
	2	Stepper motor driver - TB67S249FTG	Cool Components	7.49	14.98	
	2	100µF Cap	In stores	0.19	0.38	
	1	Thermal Paste Adhesive Tape (10mm 25m)	Amazon	9.99	9.99	
	2	Heatsink, BGA, 28.5K/W, 10 x 10 x 10mm	RS	1.24	2.48	
	1	NEO-6M 56 Channel GPS Receiver	Hobbytronics	13.2	13.2	
		Buck Converter module,3.2V-40V to 1.25V-35V				
	1	Adjustable Voltage Regulator	Amazon	7.99	7.99	
	1	5V1 Zenner diode	In stores	0.011	0.46	Total: 85.83
Light Beacon	1	EHHD006A0B41Z - DC/DC Converter	Farnell	21.43	21.43	
	5	NDT3055L N-Channel MOSFET	RS	0.708	3.54	
	1	TRN 1-1211 DC/DC Converter	Farnell	8.49	8.49	
	5	NCP718ASN330T1G - Fixed LDO Voltage Regulator	Farnell	0.111	0.555	
	5	1 µF Cap	Farnell	0.175	0.875	
	2	74HC74D - Flip-Flop	Farnell	0.262	0.524	
	3	10W Emerald Green LED (EPILED 520-530nm)	Future Eden	5.49	16.47	
	5	S1M - Standard Recovery Diode	Farnell	0.185	0.925	
	2	SK 100/100 SA - Heat Sink	Farnell	7.21	14.42	
	1	Arcol HS25 2 Ohm Panel Resistor	RS	2.13	2.13	Total: 69.359

Module	Quantity	Description	Supplier	Unit Cost	Total	
Lock-In Module	1	91K Ohm Res	Farnell	0.0086	0.0086	
	1	51K Ohm Res	Farnell	0.0088	0.0088	
	1	1M Ohm Res	Farnell	0.0263	0.0263	
	1	1.8M Ohm Res	Farnell	0.0141	0.0141	
	3	2K Ohm Res	Farnell	0.0263	0.0789	
	1	453K Ohm Res	Farnell	0.0452	0.0452	
	1	13.7K Ohm Res	Farnell	0.841	0.841	
	1	6.65K Ohm Res	Farnell	0.0184	0.0184	
	1	374 Ohm Res	Farnell	0.0107	0.0107	
	1	1 Ohm Res	Farnell	0.0183	0.0183	
	1	12.5 Ohm Res	Farnell	0.0407	0.0407	
	1	51 Ohm Res	Farnell	0.0138	0.0138	
	1	178 Ohm Res	Farnell	0.047	0.047	
	1	590 Ohm Res	Farnell	0.0794	0.0794	
	1	7.5K Ohm Res	Farnell	0.0263	0.0263	
	1	36.5K Ohm Res	Farnell	0.0235	0.0235	
	8	10K Ohm Res	RS	0.522	4.176	
	1	100K Ohm Res	In stores	0.03	0.03	
	5	1K Ohm Res	In stores	0.03	0.15	
	2	300 Ohm Res	In stores	0.03	0.06	
	1	15nF Cap	Farnell	0.0501	0.0501	
	1	1.5nF Cap	Farnell	0.0426	0.0426	
	1	18nF Cap	Farnell	0.0704	0.0704	
	1	1.8nF Cap	Farnell	0.0687	0.0687	
	16	1µF Cap	Farnell	0.13	2.08	
	20	100nF Cap	Farnell	0.0914	1.828	
	7	10nF Cap	In stores	0.64	4.48	
	8	0.33µF Cap	In stores	0.06	0.48	
	2	0.1µF Cap	In stores	0.085	0.17	
	2	35pF Cap	In stores	0.034	0.068	
	2	10µF Cap	In stores	0.11	0.22	
	1	Teensy 3.2 Microcontroller	hobbytronics	19.8	19.8	
	1	FOD8012A 2-Channel Optocoupler	Farnell	3.23	3.23	
	2	ADA4077-2ARZ	Farnell	3.52	7.04	
	1	ADG1209YRZ - 4:1 Analog Multiplexer	Farnell	5.14	5.14	
	1	CMPZ4614 Zener diode	Mouser	0.321	0.321	
	1	INA154 Difference Amp	Mouser	2.89	2.89	
	1	BAT54S Schottky Diode	Farnell	0.0409	0.0409	
	2	ADG1404YRUZ - 4:1 Analog Multiplexer	Farnell	6.09	12.18	
	1	ADA4084-2ARZ	RS	6.04	6.04	
	1	DG403DJ DPDT Switch	Farnell	6.21	6.21	
	1	INA128	RS	8.93	8.93	
	1	ADG1408 Multiplexer	Mouser	5.62	5.62	
	1	BSH103 MOSFET	Farnell	0.245	0.245	
	1	AD9833 Progammable waveform generator	Farnell	7.98	7.98	
	1	MCSJK-6NC1-20.00-25-B 1Mhz Crystal Oscillator	Farnell	0.66	0.66	
	1	LDO Reg	RS	0.4	0.4	
	1	MC78M15CDTRKG 15V Lin. Reg	Mouser	0.321	0.321	
	1	MC79M15CDTRKG -15V Lin. Reg	Mouser	0.419	0.419	
	4	SMA Bulkhead	RS	4.55	18.2	
	4	SMA Crimp Termination	RS	2.4	9.6	
	1	Aluminium enclosure 188 x 120 x 37mm	RS	14.74	14.74	Total: 145.2817
Other	1	High Gain 17dBi 2.4G WIFI Grid Parabolic Antenna Female TDJ-2400SPD4	Ebay	41.99	41.99	
	1	WiFi Antenna Cable 10M / 393.7 inch N Male to RP-SMA Male RG58 50 Ohm Low Loss Cable	Amazon	13.43	13.43	
	2	150mm Glass Plano-Conves Lenses	Rankine 709	0	0	Total: 55.42

Grand Total: **£606.18**

10 Bibliography

References

[1] Analog Devices. "In-Amp Input RFI Protection". In: *MT-070 Tutorial* (2009).

[2] DIGI. *Digi XBee Wi-Fi Module*. URL: https://www.digi.com/products/embedded-systems/rf-modules/2-4-ghz-modules/xbee-wi-fi.

[3] Egasky. *Antenna data sheet*. 2019. URL: https://www.ebay.co.uk/itm/High-Gain-17dBi-2-4G-WIFI-Wireless-Grid-Parabolic-Antenna-N-Female-TDJ-2400SPD4/281368405854?epid=1238954587%5C&hash=item4182dd275e:g:ACcAAOSwo8hTp6Ja:rk:5:pf:0.

[4] Genzo. *Confusion with different recommendations about wiring floating single-ended output sources to differential inputs*. 2018. URL: https://electronics.stackexchange.com/questions/350799/confusion-with-different-recommendations-about-wiring-floating-single-ended-outp.

[5] Philip C.D. Hobbs. "Photodiode Front Ends - The Real Story". In: *Optics & Photonics News* (2001).

[6] Mosaic Industries. *Precision Measurement Without Ground Offsets*. 2014. URL: http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/measurement-techniques/instrumentation-amplifier-ground-loop-isolator.

[7] Zurich Instruments. "Principles of lock-in detection and the state of the art". In: *White Paper* (2016).

[8] Rahul Karanam. *Monopulse Radar Presentation*. 2009. URL: https://www.slideshare.net/phanirahul/monopulse-radar.

[9] Joe Mehaffey. *GPS Altitude Readout > How Accurate?* URL: http://gpsinformation.net/main/altitude.htm.

[10] Luis Orozco. "Synchronous Detectors Facilitate Precision, Low-Level Measurements". In: *Analog Dialogue* (2014).

[11] Standford Research. "What is a lock in amplifier?" In: *SR830 Basics* ().

[12] National Institute of Standards and Technology (US). *Base Unit Definitions: Candela*. URL: https://physics.nist.gov/cuu/Units/candela.html.

[13] *Steradian*. URL: https://en.wikipedia.org/wiki/Steradian.

[14] D. Muir University of Glasgow. *Course Description*. 2019. URL: https://www.gla.ac.uk/coursecatalogue/course/?code=ENG417.

[15] Christian Wolff. *Radartutorial.eu*. 2019. URL: http://www.radartutorial.eu/06.antennas/Monopulse%5C%20Antenna.en.html.

11 Appendices

11.1 Software

```
//Set up required libraries
#include <TinyGPS++.h>                          //for interfacing with the base GPS module
#include <math.h>                               //for using arctan()
#include <SparkFun_MAG3110.h>                   //for using the magnometer
#include "DRV8825.h"                            //Driver for the stepper motors

//Define the serial ports as something more descriptive
#define baseGPSPort Serial5
#define droneGPSPort Serial1

//Defining Stepper motor items for azimuth
#define DIRaz 29
#define STEPaz 30
#define MODE0az 28
#define MODE1az 27
#define MODE2az 26
#define MOTOR_STEPSaz 200

//Defining Stepper motor items for altitude
#define DIRalt 5
#define STEPalt 6
#define MODE0alt 4
#define MODE1alt 3
#define MODE2alt 2
#define MOTOR_STEPSalt 200
#define droneDataReady 25

double avgBASE_LAT, avgBASE_LON, avgBASE_ALT;           //Average GPS variables
int avgCounter = 0;                                     //Counts the number of running averages

//For drone communication
volatile int j = 0;
volatile bool droneDataFlag = false;
char *rubbish = NULL;
char BUFFER[11] =  {'0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0'};
volatile long data = 0;

//GPS data variables
volatile double DRONE_LAT;
volatile double DRONE_LON;
volatile double DRONE_ALT;
volatile double BASE_LAT;
volatile double BASE_LON;
volatile double BASE_ALT;
```

```
45
46
47 //For home base GPS averaging
48 float verticalGPSBearing;
49 float horizontaGPSBearing;
50 volatile bool baseDataFlag = false;
51
52 //Interupt variables
53 int GPSperiod = 5000000;
54 int startTimerValue = 0;
55 IntervalTimer baseGPSDataPull;
56
57 //GPS object setup
58 TinyGPSPlus gpsBase;
59 TinyGPSPlus gpsDrone;
60
61 //Magnometer object setup
62 MAG3110 mag = MAG3110();
63
64 //Stepper motor driver object setup
65 DRV8825 stepperAzimuth(MOTOR_STEPSaz, DIRaz, STEPaz, MODE0az, MODE1az, MODE2az);
66 DRV8825 stepperAltitude(MOTOR_STEPSalt, DIRalt, STEPalt, MODE0alt, MODE1alt, MODE2alt);
67
68 ////////////////////////////////////////////////////////////////////
69 // Recieve GPS data from the base reciever. Interupts every second //
70 ////////////////////////////////////////////////////////////////////
71
72 void baseGPSData() {
73
74   cli();
75
76   unsigned long start = millis();
77
78   do {
79     while (baseGPSPort.available()) {             //If the serial port connected to the base
         GPS has data
80       gpsBase.encode(baseGPSPort.read());         //If the data is complete and ready to be
           used
81       BASE_LAT = gpsBase.location.lat();          //Find the latitude
82       BASE_LON = gpsBase.location.lng();          //Find the longitude
83       BASE_ALT = gpsBase.altitude.meters();       //Find the altitude
84
85       baseDataFlag = true;                    //Set the data flag high to allow further
           processing
86
87     }
88   } while (millis() - start < 1000);
89
90   baseDataFlag = false;
91   sei();
92
93 }
94
95 ////////////////////////////////////////////////////////////////////
```

```
// Recieves GPS data from ESP8266. Called via a pin high interrupt //
//////////////////////////////////////////////////////////////////////

void droneGPSData() {
  if (droneGPSPort.available() > 0) {
    BUFFER[j] = droneGPSPort.read();
    if (BUFFER[j] == 't') {
      //If the string in the buffer terminates with a t it is the laTitude da
      //Convert the current contents of the buffer into an int, save it and reset the buffer
      long LAT = atoi(BUFFER);
      DRONE_LAT = (double)LAT / 1000000;
      bufferReset(BUFFER);
      j = 0;

    } else if (BUFFER[j] == 'n') {
      //If the string in the buffer terminates with an n it is the loNgitude data
      long LON = atoi(BUFFER);
      DRONE_LON = (double)LON / 10000000;
      bufferReset(BUFFER);
      j = 0;

    } else if (BUFFER[j] == 'a') {
      //If the string in the buffer terminates with an a it is the Altiude data
      long ALT = atoi(BUFFER);
      DRONE_ALT = (double)ALT;
      bufferReset(BUFFER);
      j = 0;

    } else if (BUFFER[j] == '\n') {
      j = 0;
      droneDataFlag = true;
    } else {
      j++;
    }
  }
}


///////////
// Setup //
///////////

void setup()
{
  Serial.be
  gin(9600);                              //Initialise the USB serial monitor
  baseGPSPort.begin(9600);                //Initialise serial5 UART port for the GPS
      module data
  droneGPSPort.begin(9600);               //Initialise serial4 UART port for the detecotr
      teensy data

  pinMode(droneDataReady, INPUT);

  //Initialise the azimuth stepper motor
```

```
148   stepperAzimuth.begin(200, 1);
149   stepperAzimuth.enable();
150   //And its behaviour
151   digitalWrite(MODE2az, 1);
152   digitalWrite(MODE1az, 0);
153   digitalWrite(MODE0az, 1);
154
155   //Initialise the altitude stepper motor
156   stepperAltitude.begin(250, 1);
157   stepperAltitude.enable();
158   //And its behaviour
159   digitalWrite(MODE2alt, 1);
160   digitalWrite(MODE1alt, 0);
161   digitalWrite(MODE0alt, 1);
162
163
164  mag.initialize();       //Initialise the magnometer
165
166   //Setup intrrupt for recieving drone GPS data
167   attachInterrupt(digitalPinToInterrupt(droneDataReady), droneGPSData, HIGH);
168   //Setup timer interrupt for recieving base GPS data
169   startTimerValue = baseGPSDataPull.begin(baseGPSData, GPSperiod);
170
171 }
172
173 ///////////////
174 // Main loop //
175 ///////////////
176
177 void loop() {
178
179   int16_t x, y, z;                    //x, y, z values for the magnometer
180
181   //Calibrate the magnometer
182   if (!mag.isCalibrated()) {          //If we're not calibrated
183
184     if (!mag.isCalibrating()) {       //And we're not currently calibrating
185       mag.enterCalMode();             //Enter calbiration
186       stepper.rotate(360);            //Rotate platform 360 degrees
187       delay(1000);
188
189     } else {
190       mag.calibrate();
191     }
192   }
193
194   //Create a running average of the stations location IF there is data form the base station
195   if (baseDataFlag == true) {
196     avgCounter++;
197     baseRunningAverage();
198     //If there is also data from the drone then calculate the bearings between them
199     if (droneDataFlag == true ) {
200       gpsBearing();
201       droneDataFlag = false;
```

```
202     }
203     baseDataFlag = false;
204   }
205
206 }
207
208 ////////////////////////////////////////////////////
209 // Create a running average of the baseGPS coords //
210 ////////////////////////////////////////////////////
211
212 void baseRunningAverage(double BASE_LAT, double BASE_LON, double DRONE_ALT, int avgCounter)
      {
213
214   if (avgCounter <= 60) {
215
216     double netBASE_LAT = + BASE_LAT;
217     avgBASE_LAT = netBASE_LAT / avgCounter;
218
219     double netBASE_LON = + BASE_LON;
220     avgBASE_LON = netBASE_LON / avgCounter;
221
222     double netBASE_ALT = + BASE_ALT;
223     avgBASE_ALT = netBASE_ALT / avgCounter;
224
225   } else {
226
227     //Resets the averages every 60 seconds
228
229     avgBASE_LAT = 0;
230     avgBASE_LON = 0;
231     avgBASE_ALT = 0;
232
233     avgCounter = 0;
234   }
235
236   return;
237
238 }
239
240 ////////////////////////////////////////////////////
241 // Calculates desired bearings based on GPS data //
242 ////////////////////////////////////////////////////
243
244 void gpsBearing() {
245
246   //Calculates the distance to the drone station.
247   unsigned long distanceKmToBase =
248     (unsigned long)TinyGPSPlus::distanceBetween(
249       BASE_LAT,
250       BASE_LON,
251       DRONE_LAT,
252       DRONE_LON);
253
254   //Calculates the initial bearing that would lead to the drone
```

```
  double courseToBase =
    TinyGPSPlus::courseTo(
      BASE_LAT,
      BASE_LON,
      DRONE_LAT,
      DRONE_LON);

  //Calculates the vertical bearing
  double altitudeDifference = BASE_ALT - DRONE_ALT;
  float verticalBearing = (atan(altitudeDifference / distanceKmToBase)) * 180 / PI;

  int16_t x, y, z;
  float magHeading;

  //Calculates the horizontal bearing
  mag.readMag(&x, &y, &z);

  if (mag.readHeading() <= 0) {
    magHeading = mag.readHeading();
    magHeading += 360;
  } else {
    magHeading = mag.readHeading();
  }

  float horizontalError = (courseToBase - (360 - magHeading));

  azimuthRotate(horizontalError);
  altitudeRotate(verticalBearing);

}

//////////////////////////////////////////////////////////////
// Converts the bearings found into degrees of the rig motor //
//////////////////////////////////////////////////////////////

void azimuthRotate(int oldBearing) {
  int newBearing = oldBearing * 60;
  stepper.rotate(newBearing);
}

void altitudeRotate(int oldBearing) {
  int newBearing = oldBearing * 140;
  stepper.rotate(newBearing);
}

void bufferReset(char *BUFFER) {
  for (int i = 0; i == 10; i++) {
    BUFFER[i] = 0;
  }
}
```

Algorithm 10: GPSmain

```
#include <Arduino.h>
#include "DRV8825.h"

//Using a 200-step motor (most common)
#define MOTOR_STEPS 200

//Define the serial port as something more descriptive
#define detectorUART Serial4

//Configure the pins used by the motor
#define DIR 29
#define STEP 30
#define MODE0 28
#define MODE1 27
#define MODE2 26

//Pin used by the esp to trigger the data pull from the teensy
#define detectorInteruptPin 24

//Initialise the stepper motor object
DRV8825 stepper(MOTOR_STEPS, DIR, STEP, MODE0, MODE1, MODE2);

//Variables for recieving data from detector teensy
const byte numChars = 32;             //Define the size of the char array
char receivedChars[numChars];         //Initialise an array to store the received data
volatile int differential = 0;        //Where the recieved data is stored

//////////////////////////////////////////////////////////////////////
// Recieves data from the detector teensy. Called via a pin interupt //
//////////////////////////////////////////////////////////////////////

void detectorData() {

  static byte ndx = 0;                    //Recieved char index
  char endMarker = '\n';                  //endMarker = newline char
  char rc;                                //Recieved char

  if (detectorUART.available() > 0) {     //Is there data on the detector UART port?
    rc = detectorUART.read();             //Read the port and save it into rc

    if (rc != endMarker) {                //Have we reached the end of the sentance?
      receivedChars[ndx] = rc;            //If we have not then fill the current position in the char array with the rc
      ndx++;                              //Incriment ndx ready for the next char
      if (ndx >= numChars) {              //If the index is greater than or equal to the predefined size of the char array
        ndx = numChars - 1;               //Set the index to (size of array - 1)
      }
    } else {
      receivedChars[ndx] = '\0';              //If \n has been recieved then terminate the array
      ndx = 0;                                //Reset the counter
```

```
      differential = atoi(receivedChars);     //Convert the received char array into an intiger
      digitalWriteFast(LED_BUILTIN, !digitalReadFast(LED_BUILTIN) );
    }
  }
}

//////////
// Setup //
//////////

void setup() {

  //Begin serial monitor and detector UART
  Serial.begin(9600);
  detectorUART.begin(9600);

  //Initialise the stepper with RPM = 200
  stepper.begin(200, 1);

  //Setup the interrupt pin
  pinMode(detectorInteruptPin, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);

  //Create an interrupt triggered by the above pin, set it to trigger 'detector data' on a pin = HIGH
  attachInterrupt(digitalPinToInterrupt(detectorInteruptPin), detectorData, HIGH);
}

//////////////////////////////////////////////////////////////////////
// Main loop - takes received data and moves the motor accordingly //
//////////////////////////////////////////////////////////////////////

void loop() {

  cli();

  //If at the centre do not move
  if ((differential > 0) && (differential < 30)) {

    digitalWrite(MODE2, 0);
    digitalWrite(MODE1, 0);
    digitalWrite(MODE0, 0);

    stepper.disable();
    azimuthRotate(20);
    delay(1000);
    azimuthRotate(-20);
    delay(1000);
  }

  //If at the far left
  else if (differential > 400) {

```

```
    //Set to full step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 0);
    digitalWrite(MODE0, 0);

    // Tell motor to rotate by 5 * the scaler contained in azimuthRotate
    stepper.enable();
    azimuthRotate(5);
  }
  //left
  else if ((differential > 300) && (differential < 400)) {

    // 1/4 step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 1);
    digitalWrite(MODE0, 0);

    stepper.enable();
    azimuthRotate(20);
  }
  //slight left
  else if ((differential > 30) && (differential < 300)) {
    // 1/8 step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 1);
    digitalWrite(MODE0, 1);

    stepper.enable();
    azimuthRotate(40);
  }

  //far right
  else if (differential < -400) {
    //full step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 0);
    digitalWrite(MODE0, 0);

    // Tell motor to rotate 1 degree. That's it.
    stepper.enable();
    azimuthRotate(-5);
  }
  //right
  else if ((differential < -300) && (differential > -400)) {
    // 1/4 step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 1);
    digitalWrite(MODE0, 0);

    stepper.enable();
    azimuthRotate(-20);
  }
  //slight right
  else if ((differential < -30) && (differential > -300)) {
```

```
    // 1/8 step mode
    digitalWrite(MODE2, 1);
    digitalWrite(MODE1, 1);
    digitalWrite(MODE0, 1);

    stepper.enable();
    azimuthRotate(-40);
  }

  sei();
  delay(1);
}

////////////////////////////////////////////////////////////////////////////////
// Takes in an number of steps, scales it, then makes the motor rotate by that many //
////////////////////////////////////////////////////////////////////////////////

void azimuthRotate(int oldBearing) {
  int newBearing = oldBearing * 60;
  stepper.rotate(newBearing);
}
```

Algorithm 11: detectorMain

```
#include "ADC.h"
#include "RingBuffer.h"
#include <IntervalTimer.h>
#include <SPI.h>

//Definitions for the output gain muiltiplexer pins
#define outputMuxA0 2
#define outputMuxA1 3
#define outputMuxEN 23

//Definitions for the ina gain multiplexer pins
#define inaMuxA2 5
#define inaMuxA1 6
#define inaMuxA0 7
#define inaMuxEN 21

//Definitions for the input multiplexer pins
#define inputMuxA0 8
#define inputMuxA1 9
#define inputMuxEN 20

//Pin for triggering the interupt on the main teensy
#define uartDataReady 15

//Detector data variables
//volatile float normalisedMaxAmplitude = 0;
volatile int16_t maxAmplitude = 0;
int16_t left4diff, right4diff, up4diff, down4diff;
```

```
int16_t aziDiff, altDiff;

//For automatically switching input channels
volatile int8_t muxIndex = 1;

//Constants for frequency synth
#define SPI_CLOCK_SPEED 12000000      // 12MHz SPI clock
const int FSYNC = 10;                 // pin 10 (SS)
unsigned long MCLK = 20000000;        // AD9833 onboard crystal reference frequency
unsigned long freq = 250030;          // set initial frequency

//Loop indicator LED
const int ledPin = LED_BUILTIN;

//sampling pin (A8 aka 22), and period between samples
const int adcInput = A8;
const int adcReadPeriod = 500; //us

//Period between writes to the serial port
const int mainLoopPeriod = 50000; //us

//Period between mux channel switching and array swapping
const int switchPeriod = 50000; //us

ADC *adc = new ADC(); // adc object

IntervalTimer timer0; //Timer for ADC reading
IntervalTimer timer1; //Timer for switching channels

int startTimerValue0 = 0, startTimerValue1 = 0;

//For indicating data is ready
volatile int diffFlag;

//Setup data arrays
volatile int16_t leftFlag = 0;
volatile int16_t leftBufferIndex = 0;
volatile int16_t leftBufferArray[leftArraySize];
volatile int16_t rightFlag = 0;
volatile int16_t rightBufferIndex = 0;
volatile int16_t rightBufferArray[rightArraySize];
volatile int16_t rubbishFlag = 0;
volatile int16_t rubbishBufferIndex = 0;
volatile int16_t rubbishBufferArray[rubbishArraySize];

#define leftArraySize 70
#define rightArraySize 70
#define rubbishArraySize 30

//////////////////////////////////////////////
// READ DATA DROM ADC AND CHECK FOR MAXIMUM //
//////////////////////////////////////////////

void timer0_callback(void) {
```

```
83   adc->startSingleRead(adcInput, ADC_0);
84 }
85
86 // when the measurement finishes this will be called
87 void adc0_isr() {
88
89   uint8_t pin = ADC::sc1a2channelADC0[ADC0_SC1A & ADC_SC1A_CHANNELS]; // the bits 0-4 of
       ADC0_SC1A have the channel
90
91   if (pin == adcInput) {
92
93     cli();
94
95     //Read analogue pin and save it
96     int16_t buffer0 = adc->readSingle();
97
98     if (rubbishFlag == 1) {
99       if (rubbishBufferIndex == (rubbishArraySize - 1)) {
100        rubbishBufferArray[rubbishBufferIndex] = buffer0;
101        rubbishBufferIndex = 0;
102      } else {
103        rubbishBufferArray[rubbishBufferIndex] = buffer0;
104        rubbishFlag++;
105      }
106    }
107
108    if (leftFlag == 1) {
109      if (leftBufferIndex == (leftArraySize - 1)) {
110        leftBufferArray[leftBufferIndex] = buffer0;
111        leftBufferIndex = 0;
112      } else {
113        leftBufferArray[leftBufferIndex] = buffer0;
114        leftBufferIndex++;
115      }
116    }
117
118    if (rightFlag == 1) {
119      if (rightBufferIndex == (rightArraySize - 1)) {
120        rightBufferArray[rightBufferIndex] = buffer0;
121        rightBufferIndex = 0;
122      } else {
123        rightBufferArray[rightBufferIndex] = buffer0;
124        rightBufferIndex++;
125      }
126    }
127
128
129    sei();
130
131  } else { // clear interrupt anyway
132    adc->readSingle();
133  }
134
135  // restore ADC config if it was in use before being interrupted by the analog timer
```

```
136   if (adc->adc0->adcWasInUse) {
137     adc->adc0->loadConfig(&adc->adc0->adc_config);  // restore ADC config, and restart
        conversion
138     adc->adc0->adcWasInUse = false;                 // avoid a conversion started by this
        isr to repeat itself
139   }
140
141 }
142
143 ////////////////////////////////////////////////////////////////
144 // SWITCH DETECTORS VIA ISR & SAVE RELEVENT VALUES FOR PERIOD //
145 ////////////////////////////////////////////////////////////////
146
147 void muxSwitch() {
148
149   digitalWriteFast(LED_BUILTIN, !digitalReadFast(LED_BUILTIN) );
150
151   cli();
152
153   rubbishFlag = 1;
154
155   switch (muxIndex) {
156
157     case 0:
158
159       detectorMux(1);
160
161       left4diff = maximum(leftBufferArray);
162
163       leftFlag = 0;
164       rightFlag = 1;
165       muxIndex++;
166
167       sei();
168
169       break;                                        //Return
170
171     case 1:
172
173       detectorMux(2);
174
175       right4diff = maximum(rightBufferArray);
176
177       aziDiff = left4diff - right4diff;             //Azimuth differencia
178
179       rightFlag = 0;
180       leftFlag = 1;
181       diffFlag = 1;
182       muxIndex = 0;
183
184       sei();
185
186       break;
187
```

```
188     default:
189
190       sei();
191
192       break;
193   }
194 }
195
196 ///////////
197 // SETUP //
198 ///////////
199
200 void setup() {
201
202   //Setup for the frequency synth
203   pinMode (FSYNC, OUTPUT);
204   digitalWrite(FSYNC, HIGH);
205   SPI.begin();
206   AD9833setFrequency(freq);
207
208   //ADC things that I dont fully understand
209   pinMode(ledPin, OUTPUT);         // led blinks every loop
210   pinMode(ledPin + 1, OUTPUT);     // timer0 starts a measurement
211   pinMode(ledPin + 3, OUTPUT);     // adc0_isr, measurement finished for readPin0
212
213   //Setup pins for the output gain multiplexer
214   pinMode(outputMuxA0, OUTPUT);
215   pinMode(outputMuxA1, OUTPUT);
216   pinMode(outputMuxEN, OUTPUT);
217
218   //Setup pins for the instrumentation gain amplifier
219   pinMode(inaMuxA0, OUTPUT);
220   pinMode(inaMuxA1, OUTPUT);
221   pinMode(inaMuxA2, OUTPUT);
222   pinMode(inaMuxEN, OUTPUT);
223
224   //PHPD detector selection
225   pinMode(inputMuxA0, OUTPUT);
226   pinMode(inputMuxA1, OUTPUT);
227   pinMode(inputMuxEN, OUTPUT);
228
229   //Sampling pin
230   pinMode(adcInput, INPUT);
231
232   //Start the UARTs port
233   Serial.begin(9600);
234   Serial1.begin(9600);
235   pinMode(uartDataReady, OUTPUT);
236
237   //Enable averaging and set the resolution to 12bits
238   adc->setAveraging(16); // set number of averages
239   adc->setResolution(12); // set bits of resolution
240
241   //Setup sampling and conversion speeds
```

```
  adc->setConversionSpeed(ADC_CONVERSION_SPEED::HIGH_SPEED); // change the conversion speed
  adc->setSamplingSpeed(ADC_SAMPLING_SPEED::HIGH_SPEED); // change the sampling speed

  //Start the timers
  startTimerValue0 = timer0.begin(timer0_callback, adcReadPeriod);
  startTimerValue1 = timer1.begin(muxSwitch, switchPeriod);

  delayMicroseconds(40); // if we wait less than 36us the timer1 will interrupt the
      conversion

  adc->enableInterrupts(ADC_0);

  delay(500);

  //Choose an instrumentation-amp gain
  //inaGain may be equal to: 10000, 3030, 920, 279, 84, 26, 7, 2, 1
  int inaGain = 1;
  inaGainMux(inaGain);

  //Choose an output-amp gain
  //outputGain may be equal to: 909, 3, 1
  float outputGain = 1;
  outputGainMux(outputGain);

}

//////////////
// MAIN LOOP //
//////////////

void loop() {


  cli();

  if (diffFlag == 1) {
    //Serial.println(aziDiff);
    writeInt(aziDiff);
    diffFlag = 0;
  }

  sei();
  delay(100);
}

//////////////////////////////////////////////
// SET OUTPUT GAIN VIA OUTPUT MULTIPLEXER //
//////////////////////////////////////////////

void outputGainMux(int outputGain) {

  switch (outputGain) {

    //Channel 1 - gain of 909
```

```
    case 909:
      digitalWrite(outputMuxEN, HIGH);
      digitalWrite(outputMuxA0, LOW);
      digitalWrite(outputMuxA1, LOW);
      break;

    //Channel 2 - gain of 1
    case 1:
      digitalWrite(outputMuxEN, HIGH);
      digitalWrite(outputMuxA0, HIGH);
      digitalWrite(outputMuxA1, LOW);
      break;

    //Channel 3 - gain of 3
    case 3:
      digitalWrite(outputMuxEN, HIGH);
      digitalWrite(outputMuxA0, LOW);
      digitalWrite(outputMuxA1, HIGH);
      break;
  }
}

//////////////////////////////////////////////////
// SET DETECTOR CHANNEL VIA DETECTOR MULTIPLEXER //
//////////////////////////////////////////////////

void detectorMux(int detectorChannel) {

  digitalWrite(inputMuxEN, HIGH);

  switch (detectorChannel) {

    case 1:
      digitalWrite(inputMuxA0, LOW);
      digitalWrite(inputMuxA1, LOW);
      break;

    case 2:
      digitalWrite(inputMuxA0, HIGH);
      digitalWrite(inputMuxA1, LOW);
      break;

    case 3:
      digitalWrite(inputMuxA0, LOW);
      digitalWrite(inputMuxA1, HIGH);
      break;

    case 4:
      digitalWrite(inputMuxA0, HIGH);
      digitalWrite(inputMuxA1, HIGH);
      break;
  }

}
```

```
349
350 ////////////////////////////////////////////////////////
351 // SET INSTRUMENTATION OPAMP GAIN VIA INA MULTIPLEXER //
352 ////////////////////////////////////////////////////////
353
354 void inaGainMux(int inaGain) {
355
356   switch (inaGain) {
357
358     //Channel 1 - gain of 10,000
359     case 10000:
360       digitalWrite(inaMuxEN, HIGH);
361       digitalWrite(inaMuxA0, LOW);
362       digitalWrite(inaMuxA1, LOW);
363       digitalWrite(inaMuxA2, LOW);
364       break;
365
366     //Channel 2 - gain of 3,030
367     case 3030:
368       digitalWrite(inaMuxEN, HIGH);
369       digitalWrite(inaMuxA0, HIGH);
370       digitalWrite(inaMuxA1, LOW);
371       digitalWrite(inaMuxA2, LOW);
372       break;
373
374     //Channel 3 - gain of 920
375     case 920:
376       digitalWrite(inaMuxEN, HIGH);
377       digitalWrite(inaMuxA0, LOW);
378       digitalWrite(inaMuxA1, HIGH);
379       digitalWrite(inaMuxA2, LOW);
380       break;
381
382     //Channel 4 - gain of 279
383     case 279:
384       digitalWrite(inaMuxEN, HIGH);
385       digitalWrite(inaMuxA0, HIGH);
386       digitalWrite(inaMuxA1, HIGH);
387       digitalWrite(inaMuxA2, LOW);
388       break;
389
390     //Channel 5 - gain of 84
391     case 84:
392       digitalWrite(inaMuxEN, HIGH);
393       digitalWrite(inaMuxA0, LOW);
394       digitalWrite(inaMuxA1, LOW);
395       digitalWrite(inaMuxA2, HIGH);
396       break;
397
398     //Channel 6 - gain of 26
399     case 26:
400       digitalWrite(inaMuxEN, HIGH);
401       digitalWrite(inaMuxA0, HIGH);
402       digitalWrite(inaMuxA1, LOW);
```

```
      digitalWrite(inaMuxA2, HIGH);
      break;

    //Channel 7 - gain of 7
    case 7:
      digitalWrite(inaMuxEN, HIGH);
      digitalWrite(inaMuxA0, LOW);
      digitalWrite(inaMuxA1, HIGH);
      digitalWrite(inaMuxA2, HIGH);
      break;

    //Channel 8 - gain of 2
    case 2:
      digitalWrite(inaMuxEN, HIGH);
      digitalWrite(inaMuxA0, HIGH);
      digitalWrite(inaMuxA1, HIGH);
      digitalWrite(inaMuxA2, HIGH);
      break;

    //Not connected - gain of 1
    case 1:
      digitalWrite(inaMuxEN, LOW);
      break;
  }
}

//////////////////////////////////////////////
// CHANGE FREQUENCIES OF AD9833 SIG GEN CHIP //
//////////////////////////////////////////////

void AD9833setFrequency(long frequency) {
  long FreqReg = (frequency * pow(2, 28)) / MCLK;
  int MSB = (int)((FreqReg & 0xFFFC000) >> 14);    // only lower 14 bits are used for data
  int LSB = (int)(FreqReg & 0x3FFF);

  LSB |= 0x4000;                        // DB 15=0, DB14=1
  MSB |= 0x4000;                        // DB 15=0, DB14=1

  WriteRegister(0x21E8);
  WriteRegister(LSB);                     // write lower 16 bits to AD9833 registers
  WriteRegister(MSB);                     // write upper 16 bits to AD9833 registers
  WriteRegister(0xC000);                  // write phase register
  WriteRegister(0x0068);
}

//Write to the frequnecy generators registers
void WriteRegister(int data) {
  SPI.beginTransaction(SPISettings(SPI_CLOCK_SPEED, MSBFIRST, SPI_MODE2));
  digitalWrite(FSYNC, LOW);             // set FSYNC low before writing to AD9833 registers
  SPI.transfer16(data);
  digitalWrite(FSYNC, HIGH);            // write done, set FSYNC high
  SPI.endTransaction();
}

```

```
////////////////////////////////////////////
// WRITE INTEGERS TO THE MAIN TEENSY 3.6 //
////////////////////////////////////////////

void writeInt(int number) {
  digitalWriteFast(uartDataReady, HIGH);              //Trigger the interupt on the 3.6 to start it collecting data
  Serial1.println(number);                            //Send the number
  digitalWriteFast(uartDataReady, LOW);               //Reset the pin whihc triggers the interupt
  return;
}

volatile int16_t maximum(volatile int16_t array[]){    //finds the max value of the last 16 data inputs
    volatile int16_t maxA = array[0];
    for (int x = 0; x<69; x++){    //cycles through the array
        if (array[x] != 0){   //filters out zero values
            if (array[x]>maxA){    //if the value in the array is more than the current max then replace it
                maxA = array[x];   //max = that element in the array
            }
        }
    }
    return maxA;
}

volatile int16_t  minimum(volatile int16_t array[]){    //finds the minimum value of the last 16 data inputs
    volatile int16_t minA = array[0];
    for (int x = 0; x<69; x++){
        if (array[x] != 0){   //filters 0
            if (array[x]<minA){
                minA = array[x];   //if the value in the array is less than the current min then replace it
            }
        }
    }
    return minA;
}
```

Algorithm 12: sampling

11.2 Data

Figure 47: Clockwise Optical Tracking at 15m,

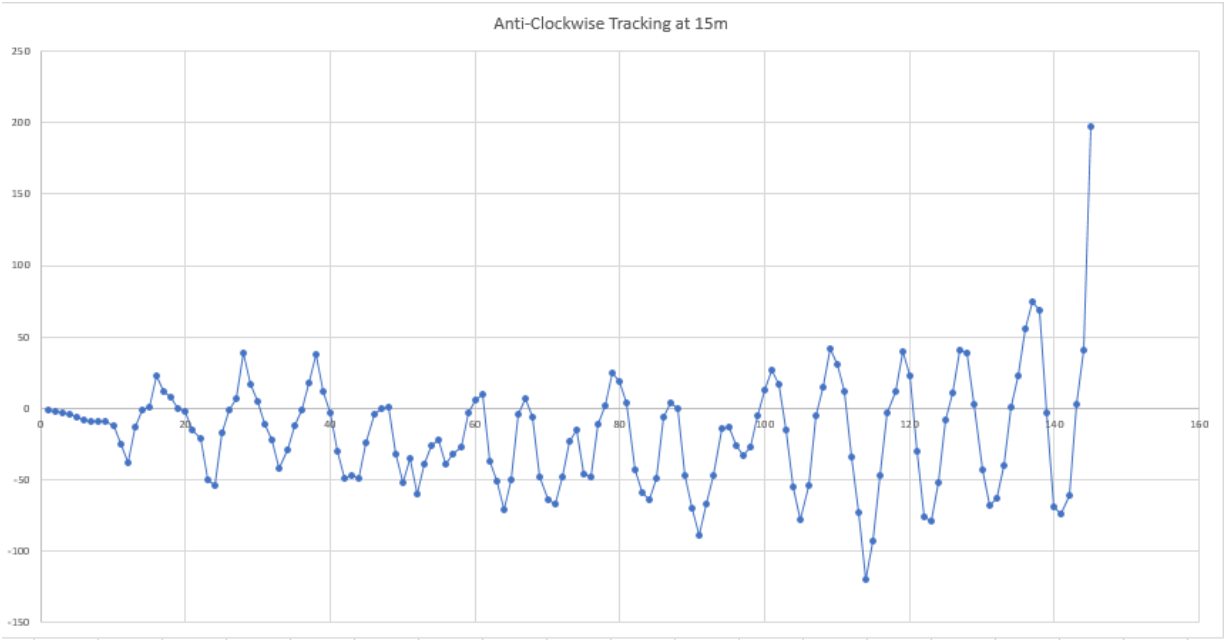


Figure 48: Anti-Clockwise Optical Tracking at 15m

Sats	HDOP	Latitude (deg)	Longitude (deg)	Fix Age	Date	Time	Date Age	Alt (m)	Course --- from	Speed GPS ----	Card	Distance ---- to	Course London ----	Card	Chars RX	Sentences RX	Checks Fail
****	*****	**********	**********	****	**********	********	****	******	******	*****	***	********	******	***	63	0	0
4	1.7	55.872661	-4.285974	452	01/11/2019	13:25:21	564	9.00	0.00	0.50	N	557	148.88	SSE	455	2	0
4	1.7	55.872665	-4.285979	528	01/11/2019	13:25:22	642	8.70	0.00	0.72	N	557	148.88	SSE	847	4	0
4	1.7	55.872665	-4.285984	605	01/11/2019	13:25:23	718	8.50	0.00	1.09	N	557	148.88	SSE	1239	6	0
4	1.7	55.872669	-4.285988	682	01/11/2019	13:25:24	796	7.30	0.00	0.44	N	557	148.88	SSE	1631	8	0
4	1.7	55.872673	-4.285997	758	01/11/2019	13:25:25	870	6.00	0.00	0.78	N	557	148.88	SSE	2023	10	0
4	1.7	55.872684	-4.286005	2	01/11/2019	13:25:27	16	4.90	358.93	2.57	N	557	148.88	SSE	2622	14	0
4	6.3	55.872688	-4.286009	1069	01/11/2019	13:25:28	244	4.20	358.93	2.57	N	557	148.88	SSE	3122	14	0
4	6.3	55.872688	-4.286009	2116	01/11/2019	13:25:29	277	4.20	358.93	2.57	N	557	148.88	SSE	3417	14	0
4	6.3	55.872688	-4.286009	3193	01/11/2019	13:25:30	354	4.20	358.93	2.57	N	557	148.88	SSE	3712	14	0
4	6.3	55.872688	-4.286009	4271	01/11/2019	13:25:31	432	4.20	358.93	2.57	N	557	148.88	SSE	4007	14	0
4	6.3	55.872688	-4.286009	5347	01/11/2019	13:25:32	507	4.20	358.93	2.57	N	557	148.88	SSE	4302	14	0
4	6.3	55.872688	-4.286009	6424	01/11/2019	13:25:33	583	4.20	358.93	2.57	N	557	148.88	SSE	4597	14	0
4	1.7	55.872665	-4.286069	499	01/11/2019	13:25:34	619	2.50	167.63	2.15	SSE	557	148.88	SSE	5001	16	0
4	1.7	55.872665	-4.286072	584	01/11/2019	13:25:35	702	2.50	50.78	3.32	NE	557	148.88	SSE	5403	18	0
4	1.7	55.872669	-4.286074	659	01/11/2019	13:25:36	776	2.30	22.18	3.26	NNE	557	148.88	SSE	5805	20	0
4	1.7	55.872676	-4.286074	739	01/11/2019	13:25:37	857	2.70	3.47	4.37	N	557	148.88	SSE	6205	22	0
4	1.7	55.872676	-4.286073	885	01/11/2019	13:25:38	1014	3.10	3.47	1.50	N	557	148.88	SSE	6781	26	0
4	1.7	55.872657	-4.286067	75	01/11/2019	13:25:40	202	3.00	3.47	0.78	N	557	148.88	SSE	7342	28	0
4	6.3	55.872657	-4.286067	1199	01/11/2019	13:25:41	339	3.00	3.47	0.78	N	557	148.88	SSE	7676	28	0
4	6.3	55.872657	-4.286067	2276	01/11/2019	13:25:42	418	3.00	3.47	0.78	N	557	148.88	SSE	7971	28	0
4	6.3	55.872657	-4.286067	3353	01/11/2019	13:25:43	492	3.00	3.47	0.78	N	557	148.88	SSE	8274	28	0
4	6.3	55.872657	-4.286067	4430	01/11/2019	13:25:44	569	3.00	3.47	0.78	N	557	148.88	SSE	8577	28	0
4	6.3	55.872657	-4.286067	5506	01/11/2019	13:25:45	644	3.00	3.47	0.78	N	557	148.88	SSE	8880	28	0
4	6.3	55.872657	-4.286067	6583	01/11/2019	13:25:46	718	3.00	3.47	0.78	N	557	148.88	SSE	9189	28	0
4	6.3	55.872657	-4.286067	7660	01/11/2019	13:25:47	796	3.00	3.47	0.78	N	557	148.88	SSE	9492	28	0
4	6.3	55.872657	-4.286067	8737	01/11/2019	13:25:48	877	3.00	3.47	0.78	N	557	148.88	SSE	9803	28	0
4	6.3	55.872657	-4.286067	9813	01/11/2019	13:25:49	952	3.00	3.47	0.78	N	557	148.88	SSE	10190	28	0
4	6.3	55.872608	-4.286026	77	01/11/2019	13:25:51	91	3.00	3.47	0.39	N	557	148.88	SSE	10671	30	0
4	1.7	55.872611	-4.286026	130	01/11/2019	13:25:52	264	0.80	188.87	2.02	S	557	148.88	SSE	11213	32	0
4	1.7	55.872608	-4.286022	246	01/11/2019	13:25:53	359	1.00	188.87	1.17	S	557	148.88	SSE	11605	34	0
4	1.7	55.872604	-4.286019	310	01/11/2019	13:25:54	429	1.00	182.90	1.87	S	557	148.88	SSE	12017	36	0
4	1.7	55.872600	-4.286016	394	01/11/2019	13:25:55	508	1.10	182.90	0.72	S	557	148.88	SSE	12417	38	0
4	1.7	55.872604	-4.286015	462	01/11/2019	13:25:56	582	1.20	295.91	2.61	WNW	557	148.88	SSE	12829	40	0
4	1.7	55.872604	-4.286013	539	01/11/2019	13:25:57	659	1.00	315.90	1.65	NW	557	148.88	SSE	13241	42	0
4	1.7	55.872604	-4.286012	631	01/11/2019	13:25:58	745	0.70	315.90	0.83	NW	557	148.88	SSE	13641	44	0

Figure 49: GPS Data for 74 Oakfield Avenue

Sats	HDOP	Latitude (deg)	Longitude (deg)	Fix Age	Date	Time	Date Age	Alt (m)	Course --- from GPS	Speed ----	Card	Distance ---- to London	Course ----	Card	Chars RX	Sentences RX	Checksum Fail
****	*****	**********	**********	****	**********	********	****	******	******	*****	***	********	******	***	63	0	0
7	1.0	55.876777	-4.273671	778	01/10/2019	17:21:39	893	33.60	0.00	0.39	N	557	148.98	SSE	590	2	0
7	1.0	55.876777	-4.273669	23	01/10/2019	17:21:41	37	34.10	0.00	0.19	N	557	148.98	SSE	1315	6	0
7	1.0	55.876777	-4.273667	99	01/10/2019	17:21:42	230	34.00	0.00	0.28	N	557	148.98	SSE	1995	8	0
7	1.0	55.876781	-4.273666	274	01/10/2019	17:21:43	401	34.10	0.00	0.11	N	557	148.98	SSE	2651	10	0
7	1.0	55.876781	-4.273664	356	01/10/2019	17:21:44	470	34.40	0.00	0.20	N	557	148.98	SSE	3167	12	0
7	1.0	55.876781	-4.273663	430	01/10/2019	17:21:45	544	34.60	0.00	0.76	N	557	148.98	SSE	3683	14	0
7	1.0	55.876785	-4.273662	498	01/10/2019	17:21:46	619	34.70	356.43	1.22	N	557	148.98	SSE	4211	16	0
7	1.0	55.876789	-4.273661	575	01/10/2019	17:21:47	696	35.40	359.23	1.28	N	557	148.98	SSE	4739	18	0
7	1.2	55.876792	-4.273660	664	01/10/2019	17:21:48	779	36.00	359.23	1.06	N	557	148.98	SSE	5255	20	0
7	1.2	55.876792	-4.273658	737	01/10/2019	17:21:49	850	36.20	359.23	0.72	N	557	148.98	SSE	5771	22	0
7	1.2	55.876792	-4.273657	878	01/10/2019	17:21:50	1009	36.40	359.23	0.04	N	557	148.98	SSE	6474	26	0
7	1.2	55.876796	-4.273654	71	01/10/2019	17:21:52	198	36.60	359.23	0.28	N	557	148.98	SSE	7152	28	0
7	1.2	55.876796	-4.273653	246	01/10/2019	17:21:53	374	36.50	359.23	0.30	N	557	148.98	SSE	7831	30	0
7	1.2	55.876796	-4.273653	356	01/10/2019	17:21:54	470	36.70	359.23	0.54	N	557	148.98	SSE	8347	32	0
7	1.2	55.876800	-4.273652	431	01/10/2019	17:21:55	545	37.40	359.23	0.78	N	557	148.98	SSE	8863	34	0
7	1.0	55.876804	-4.273652	508	01/10/2019	17:21:56	622	37.70	359.23	0.33	N	557	148.98	SSE	9381	36	0
7	1.0	55.876808	-4.273651	583	01/10/2019	17:21:57	698	38.20	359.23	0.37	N	557	148.98	SSE	9897	38	0
7	1.0	55.876808	-4.273650	659	01/10/2019	17:21:58	774	38.10	359.23	0.07	N	557	148.98	SSE	10413	40	0
8	1.0	55.876808	-4.273650	734	01/10/2019	17:21:59	849	38.20	359.23	0.11	N	557	148.98	SSE	10933	42	0
8	1.0	55.876808	-4.273649	807	01/10/2019	17:22:00	922	38.40	359.23	0.04	N	557	148.98	SSE	11547	45	0
7	1.0	55.876811	-4.273649	109	01/10/2019	17:22:02	123	38.50	359.23	0.19	N	557	148.98	SSE	12249	48	0
7	1.0	55.876811	-4.273649	172	01/10/2019	17:22:03	301	38.90	359.23	0.13	N	557	148.98	SSE	12935	50	0
7	1.0	55.876811	-4.273649	346	01/10/2019	17:22:04	467	38.80	359.23	0.20	N	557	148.98	SSE	13517	52	0
7	1.0	55.876815	-4.273650	393	01/10/2019	17:22:05	508	39.20	359.23	0.24	N	557	148.98	SSE	14033	54	0
7	1.0	55.876819	-4.273650	462	01/10/2019	17:22:06	577	39.70	359.23	0.33	N	557	148.98	SSE	14549	56	0
8	1.0	55.876819	-4.273650	535	01/10/2019	17:22:07	649	40.00	359.23	0.09	N	557	148.98	SSE	15069	58	0
7	1.0	55.876823	-4.273650	622	01/10/2019	17:22:08	736	40.50	359.23	0.30	N	557	148.98	SSE	15585	60	0
7	1.0	55.876823	-4.273650	699	01/10/2019	17:22:09	814	41.10	359.23	0.54	N	557	148.98	SSE	16101	62	0
7	1.0	55.876827	-4.273650	777	01/10/2019	17:22:10	891	41.40	359.23	0.31	N	557	148.98	SSE	16622	64	0
7	1.0	55.876827	-4.273650	20	01/10/2019	17:22:12	36	42.00	359.23	0.50	N	557	148.98	SSE	17348	68	0
7	1.0	55.876831	-4.273649	98	01/10/2019	17:22:13	226	42.60	359.23	0.24	N	557	148.98	SSE	18027	70	0
7	1.0	55.876831	-4.273649	282	01/10/2019	17:22:14	409	43.00	359.23	0.39	N	557	148.98	SSE	18681	72	0
7	1.0	55.876831	-4.273649	356	01/10/2019	17:22:15	470	42.50	359.23	0.09	N	557	148.98	SSE	19197	74	0
8	1.0	55.876827	-4.273648	429	01/10/2019	17:22:16	544	42.40	359.23	0.06	N	557	148.98	SSE	19717	76	0
7	1.0	55.876827	-4.273648	511	01/10/2019	17:22:17	625	42.30	359.23	0.07	N	557	148.98	SSE	20233	78	0
7	1.0	55.876827	-4.273648	591	01/10/2019	17:22:18	706	42.40	359.23	0.06	N	557	148.98	SSE	20749	80	0
7	1.0	55.876827	-4.273650	659	01/10/2019	17:22:19	773	42.10	359.23	0.30	N	557	148.98	SSE	21267	82	0
8	1.0	55.876827	-4.273651	736	01/10/2019	17:22:20	851	42.30	359.23	0.07	N	557	148.98	SSE	21787	84	0
7	1.8	55.876823	-4.273651	818	01/10/2019	17:22:21	931	41.90	359.23	0.20	N	557	148.98	SSE	22404	87	0
7	1.0	55.876827	-4.273653	109	01/10/2019	17:22:23	124	42.50	359.23	0.19	N	557	148.98	SSE	23101	90	0
7	1.0	55.876827	-4.273653	172	01/10/2019	17:22:24	302	42.70	359.23	0.13	N	557	148.98	SSE	23788	92	0

Figure 50: GPS Data for 54 Oakfield Avenue