

UCF Senior Design I

Optical Beacon Tracker for Optical Wireless Communication

Department of Electrical Engineering and Computer Science

The College of Optics and Photonics

University of Central Florida

Dr. Chung Yong Chan

Dr Aravinda Kar

Sponsored by the Knight Vision Lab

Reviewers

Dr. Kyle Renshaw CREOL

Mark Maddox ECE

Dr. Justin Phelps ECE

Group 4

Seth Johnson Photonic Science and Engineering

Raymond Loudis Electrical Engineering

Blake Baez Computer Engineering

Table of Contents

Chapter 1: Executive Summary

Chapter 2: Project Description

- 2.1: Motivation and background**
- 2.2: Goals and Objectives**
- 2.3: Requirements and Specifications**
- 2.4: House of Quality**
- 2.5: Hardware and Software**
 - 2.5.1: Hardware**
 - 2.5.2: Optical Diagram**
 - 2.5.3: Software**

Chapter 3: Research and Part Selection

- 3.1: Similar Projects**
- 3.2: Constraints Due to Engineering Specifications**
- 3.3: Technology Investigation and Selection**
 - 3.3.1 Exploring Motion Control Solutions**
 - 3.3.2 Exploring MCU and IMU Solutions**
 - 3.3.3 Exploring GPS Solutions**
 - 3.3.4: Optical Components**
 - 3.3.5: Photodetectors**
 - 3.3.6: Light Sources**
 - 3.3.7: Energy Source Solutions**
 - 3.3.8: Voltage Regulators**
- 3.4 Coding Languages**

Chapter 4: Design Constraints and Standards

- 4.1: Standards**
 - 4.1.1 Laser Safety Standards**
 - 4.1.2 Software and Communication Standards**
- 4.2: Constraints**
 - 4.2.1 Budgetary Constraints**
 - 4.2.2 Weight Constraints**
 - 4.2.3 Environmental Constraints**
 - 4.2.4 Location Constraints**
 - 4.2.5 Technical Constraints**
 - 4.2.6 Integration Constraints**
 - 4.2.7 Regulatory Constraints**
 - 4.2.7 Time Constraints**

Chapter 5: Comparison of LLM Platforms

Chapter 6: Design

- 6.1: Optical**

6.2: Electrical

6.2.1 MCU Design

6.2.2 Receiver PCBs

6.2.3 PSU's

6.2.4 LED Array

6.2.5 MOSFET Switches, IMU, GPS and LoRa Modules

6.3: Software

6.3.1 Circuit Design

6.3.2 Software Use Case Diagram

6.3.3 Software: Motor Control

6.3.4 Software: MCU and PCB Connection

6.3.5 Microcontroller programming

Chapter 7: Integration and Prototype Construction

7.1 Microcontroller and GPS Integration

7.2 IMU integration

7.3 Telemetry Communication

7.4 Telemetry and GPS Integration

7.5 Laser Diode Array Control

7.6 Quadrant Photodiode (QPD) Integration

7.1 Microcontroller and GPS Integration

7.7 Motor Integration

7.8 Motor, Laser DiodeArray, and Quadrant Photodiode Integration

7.9 Final System Integration

Chapter 8: Administrative Content

8.1: Budget Estimates and Funding

8.2: Deadlines and Milestones

8.3: Work Distribution

Chapter 9: Conclusion

Appendix A: Citation

Appendix B: LLM Prompts and Outputs

Acknowledgments

We'd like to thank our professors, Dr. Chung Yong Chan, Dr. Lei Wei, and Dr. Aravinda Kar for their guidance and reviews during the process of writing this paper. We'd also like to thank our reviewers Dr. Kyle Renshaw, Mark Maddox, and Justin Phelps for taking the time out of their schedules to review our project and offer their own assistance when we've needed it. We'd like to thank Dr. Jeremy Mares, Clay Chester, and Changkee Hong for offering their guidance for designing the project. Lastly, we'd like to thank the entirety of the Knight Vision Lab for sponsoring this project and lending us their equipment for the construction and testing of the project.

1 Executive Summary

By the end of this project our goal is to build a device capable of locating and tracking an optical beacon. We'll accomplish this by using an InGaAs photodiode coupled to a fiber which is connected to the center of an Si annular quadrant photodiode. Both photodiodes will be attached to a set of motors, with the signal outputs from the photodiodes giving directions to said motors.

We'll also create a beacon using an array of LEDs that emit 850nm light and a fiber couple laser diode that emits 1550nm light. The beacon will be attached to a set of servo motors so that both systems can track each other whilst mobile. The tracker/receiver system will have an IMU, and both systems will possess GPS modules for the purpose of general alignment within a few meters. Once the 850nm light is incident onto the quadrant photodiode, the output signal will align the motors until the 1550nm light is incident onto the optic fiber. When the 1550nm light passes through the fiber and onto the InGaAs photodiode, we'll know we have acceptable alignment.

In this paper, we'll be discussing our motivations for the project, comparing projects that serve a similar purpose, and how we plan on accomplishing our goals. We'll also go into extensive detail about our research into technologies we can use for our design and the selection of components for the device.

In chapter 2, we'll discuss in detail the motivation behind the project, as well as list our basic, advance, and stretch goals. We'll also list the objectives we'll have to accomplish in order to meet those goals, and briefly outline the methods we'll use to design the optical, electrical, and software components of the device. In chapter 3, we'll discuss our investigation into similar projects as well as technology and chosen components for building our device.

In chapter 4 we'll cover the various standards and constraints we'll be limited by, as well as the impacts they'll have on how we'll design and test the device safely in a given environment. In chapter 5 we'll do an analysis and comparison of outputs from Chat GPT and other LLM softwares in order to see how helpful they can be for planning engineering projects. In chapter 6 we'll go into more detail about the electrical, optical, and software design processes than what was discussed in chapter 2.

In chapter 7, we'll cover the overall integration of each sub-system for the project. Since the actual building of the project will be done in Senior Design 2, more of chapter 7 will simply be our plan for actually assembling the beacon and the tracker respectively, when the time comes. Chapter 8 will cover the administrative contents for the project, such as budget estimates, workload distributions, and milestones for Senior Design 1 and 2. Lastly, chapter 9 will be the conclusion and summary of our efforts towards the project.

While some of the physics and engineering subjects will already be familiar to our team members, we'll still have to extensive research into topics such as position sensing using quadrant photodiodes, designing and choosing laser drivers, and programming motors to

move based on inputs from photodiodes as well as GPS and IMU modules. These are just a few of the new skills the team will have to learn in order for the project to be a success.

2 Project Description

2.1 Motivation and Background

The idea of optical wireless communication (OWC) has existed for decades and comes with many advantages. Signals can be driven by light sources across the electromagnetic spectrum, granting a large range of frequencies to choose from, and the speed of signal transfer is only limited by the pulse width of the light source since the signal travels at the speed of light. Signals also cannot be intercepted unless the potential spy is within the line of sight of the transmitter, which makes OWC one of the most secure ways to send communications. Unfortunately, clear signals require precise alignment and are weak to interference from atmospheric loss [4]. Also, if the transmitter or receiver are mobile then maintaining that precise alignment becomes even more difficult, which is the problem we will try to solve.

The goal of this project is to build a device that can track an optical beacon while both the receiver and the beacon are mobile. When it comes to position tracking, one of the most obvious solutions would be GPS, and in fact we'll even be including GPS in the design for general alignment. However, most GPS systems are only accurate down to a few meters, while optical communications require precision down the centimeter. With a quadrant photodiode (QPD) we can accurately track the position of any light source at a certain wavelength as long as the receiver optics are properly designed. With an array of NIR LEDs, we can transmit light over long distances with enough optical power to be detected by the QPD and align both the receiver and the beacon. We'll be using a QPD sensitive to 850nm for alignment, and a standard photodiode sensitive to 1550nm to confirm perfect alignment of the system.

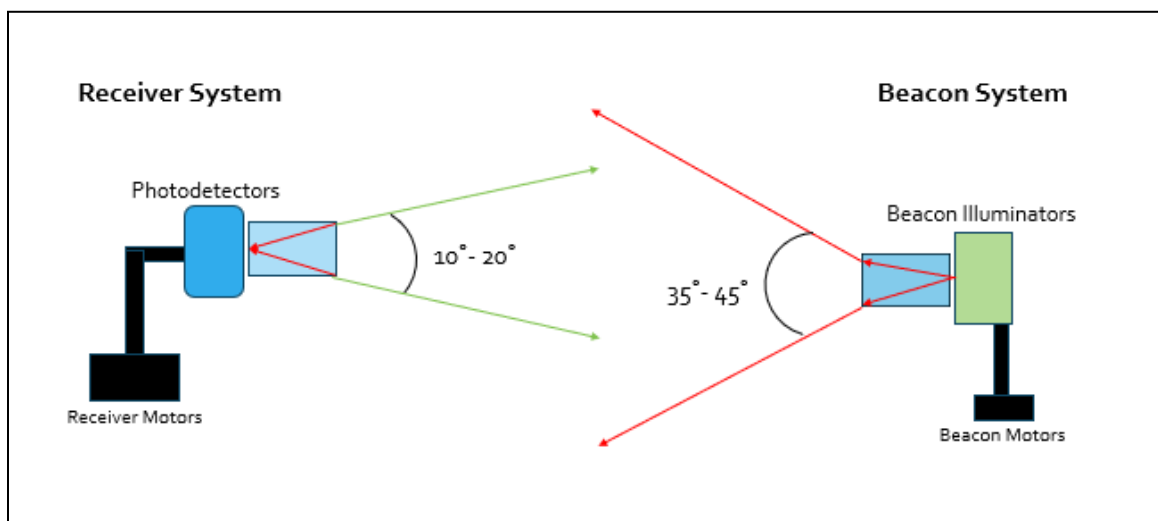


Figure 2.1-1: System diagram of the optical beacon tracker.

Using two separate wavelengths is essential. If we only used one wavelength, the beacon and the transmitted signal would interfere with each other. By using diodes that emit 850nm and 1550nm infrared light, we can avoid loss due to atmospheric interference. The lack of susceptibility to atmospheric loss is also one of the reasons that 1550nm is the most common wavelength for OWC, which is why it's the wavelength we'll use to confirm alignment. We'd like to stress that we are not building a communication system, only a tracker that can be easily integrated into a communication system.

2.2 Goals and Objectives

The goals we have for this project span from immediate basic goals, to advanced and stretch goals. The basic goals define the bare minimum for the project to be successful, and can easily be accomplished by our project deadline. The basic goals for this project can be summarized as follows:

- Create a beacon with enough power and divergence to be detected up to 3 meters away whilst indoors
- Create a receiver that can actively track the beacon whilst mobile, up to 3 meters away when indoors

The advance goals for the project will be more difficult and still need to be completed before our allotted time for the project is up, but they are still reasonably achievable. The advance goals are:

- Capable of tracking up to 1 km away whilst outdoors.
- Tracking the beacon with the accuracy stated in the requirements and specifications section.

The stretch goals would only be attainable if we stay very far ahead on this project. Since we only have 3 team members, the second stretch goal in particular would be difficult to achieve during our allotted time. The stretch goals are:

- Capable of tracking up to 10 km away in ideal weather conditions.
- Capable of transmitting and decoding a simple signal (like an SOS) from the beacon.

For now, the goal is simply to be capable of tracking a beacon attached to the object transmitting the signals. To be able to send and decode actual signals from the beacon would reasonably require an extra team member, which makes it a far stretching goal. Tracking the beacon at 10 km should be possible in theory, however proving it in any way other than mathematically would be nearly impossible without the use of drones.

The objectives for this project will all add up to help us accomplish our goals.

- Perform safety calculations so that the 850 nm diodes can be safely used indoors.
- Calculate the required optical output power for the beacon to be detectable by the receiver at a given distance.
- Design the focusing, diverging, and collimating lens systems on the beacon and receiver.
- Design a PCB for the beacon and receiver.

- Design a power supply for the beacon and receiver respectively, capable of driving the laser diodes and servo motors, as well as biasing the photodiodes.
- Create an algorithm that enables the servo motors to use outputs from the QPD, GPS, and IMU as inputs for positioning.

2.3 Requirements and Specifications

Like any engineering project, we'll have to meet certain design specifications by the end of the year. This is a sponsored project, so certain specifications such as outdoor functional distance and track accuracy are set by the sponsor. Other specifications were set by ourselves to ensure we can meet our goals for the project.

#:	Requirement	Specification	Description	Priority
1	Receiver FOV	10-20 degrees	The viewing angle from the receiver	Mid
2	Beacon FOV	35-45 degrees	The divergence angle of the beacon	Mid
3	Stationary track accuracy	10-20 μ Rad	The accuracy of the system when stationary	High
4	Mobile track accuracy	500-600 μ Rad	The accuracy of the system when mobile	High
5	Track loop without IMU compensation	10-15 kHz	Signal processing loop without using an IMU	Mid
6	Track loop with IMU compensation	100-200 Hz	Signal processing loop while using an IMU	Mid
7	Receiver and Beacon weight	Less than 5kg Each	The receiver and beacon must weigh less than 5kg respectively	Low
8	Indoor functional distance	2-3 meters	Functional distance while indoors	High
9	Outdoor functional distance	0.1-1 kilometers	Functional distance while outdoors	Mid
10	Battery life	30-60 minutes	How long the system will be able to run on battery supply	High
11	Reaction speed	300-600 ms	Time delay between detecting an optical signal and moving the motors	High
12	GPS Accuracy	2-3 meters	Accuracy of the GPS system when outdoors.	mid

Figure 2.3-1: Table of project specifications.

2.4 House of Quality

The house of quality diagram is essential for making sure our engineering specifications meet the needs of our clients or sponsors. Our quality house is shown below.

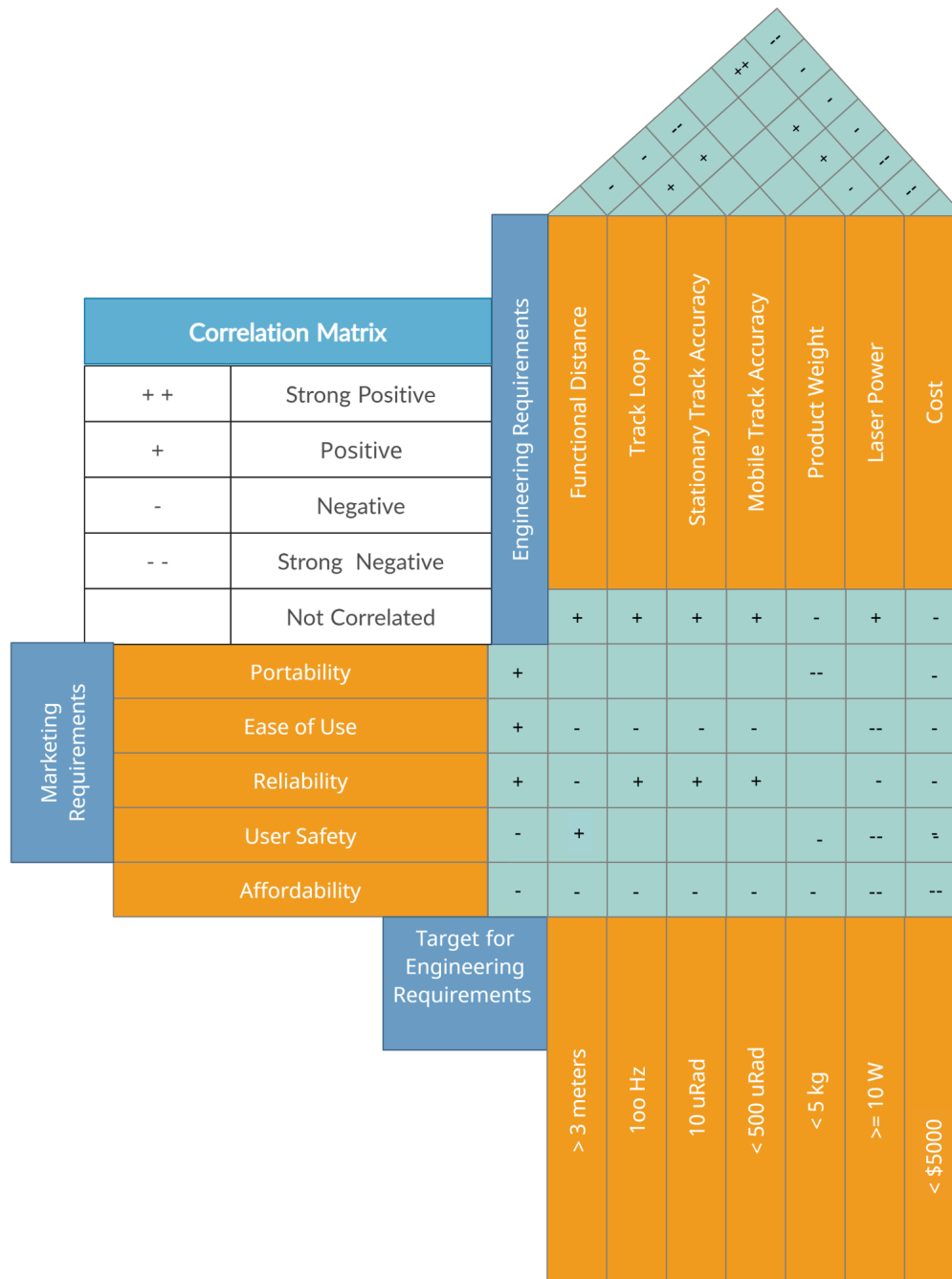
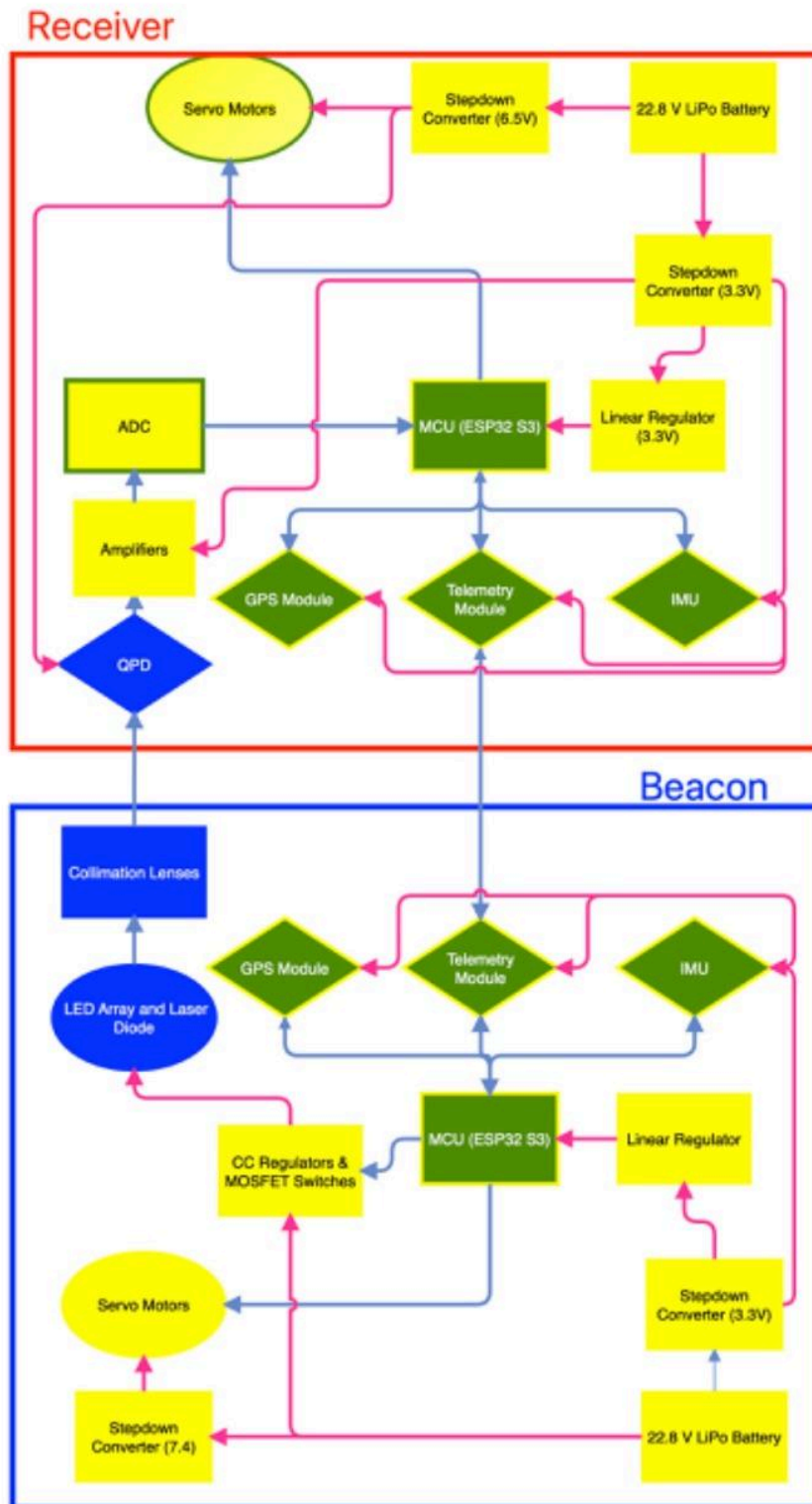


Figure 2.4-1: House of quality diagram

2.5 Hardware and Software

2.5.1 Hardware



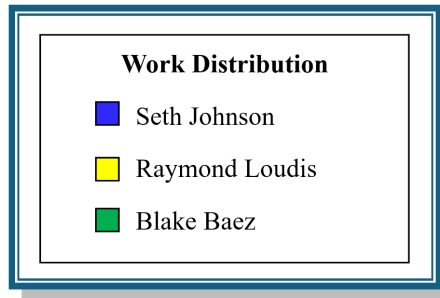


Figure 2.5.1-1: Hardware block diagram.

In the flowchart above blue arrows and traces represent the exchange of analog and digital signals while the red arrows and traces represent the power distribution for each of the components.

The GPS, telemetry and IMU modules along with the esp32 will all be powered by an LM2567 fixed switching regulator at 3.3V while the servo motors and QPD will receive their power via an adjustable LM2567 set at 6.5V. Furthermore, the LED array which is made up of three series of LEDs will be using CC buck regulators to step the voltage down to each threshold and hold the current at 80mA per series.

The quadrant photodiode or QPD, consists of four separate photodiode segments or quadrants. When the light source of the beacon illuminates the photodiode each segment generates a current proportional to the light received. By comparing these currents, we hope to determine the position of the light source relative to the photodiode and align it with the beacon's transmitter/LED array. In order to perform this comparison we will be sending the QPD signals through two stages of operational amplifiers, the first will be a transimpedance stage which will be used to convert the small signal current into a measurable voltage, the signal will be outputted into a negative voltage here. The second stage of operational amplifiers will be used to re-invert the signal and pass it to the ESP32s ADC as a positive voltage. To ensure that the signal does not surpass the 3.3V maximum magnitude allowed by the ADC the second stage of op-amps will be supplied by 3.3V. More about obtaining alignment via optical signal will be included in the software section.

For general alignment purposes we have decided to use 2 different methods, the first involves implementing gps and IMU modules on both the Beacon and the Receiver and have them share one another's location over LoRa. From this data both devices will be able to determine a bearing angle in which to turn for alignment. The second method is to be implemented while location data is not available. When this occurs both devices will enter into the "lighthouse protocol" in which they will pan and tilt through their full range of motion and search for an optical signal. When the optical signal is sensed by the receiver a command to stop is sent over LoRa to have the beacon hold its position while the receiver fine tunes the alignment using the QPD.

In regards to the LED array and laser diode, one of our stretch goals was to implement digital modulation for the transmitter so that the user can modulate the laser diodes output

via digital signal, which would be useful for pulsating or changing intensity based on user control. However, we did implement PWM by using MOSFET switches in our final design but did not include real time user adjustments. Essentially a duty cycle can be set on the MCU via alterations in the code and flashed as needed but implementing user control became less of a priority when designing GUI. For our live demo indoors we plan on pulsing the LEDs leaving them on for 2-3 seconds and off for the same duration. This duration will allow for strong and clear signals to be read by the receiver while still demonstrating the PWM capabilities. Whilst outdoors however they will be pulsed at a much faster rate to compensate for unwanted light received by the QPD, allowing for a signal of a certain frequency to be passed and ignoring all other signals.

In conclusion, the power source that we are using is a 22.8 Li-Po battery which will be readily available and mounted on the drones in which we designed our project to be outfitted to. The power distribution will be handled by CC and CV regulators as needed and the ESP32 S3 will be responsible for handling the computations and data relaying between the different modules and components.

2.5.2 Optical Diagram

The optical design is crucial to this project. We must be able to project the light of our beacon with enough divergence to cover a significant area. We must then be able to collect the light from the beacon so that it illuminates the active cell of our QPD while also focusing onto a separate detector.

Laser diodes naturally diverge unevenly on the horizontal and vertical axis, so both arrays will have to be collimated into smaller, more evenly dispersed beams. They will then both pass through a dichroic mirror and a diffuser plate so that the arrays appear as a single spatially coherent beam, and then diverged upon exiting the beacon aperture. Upon reaching the receiver, the two wavelengths must be separated again so that they can be focused onto their own respective detectors. The diagram below illustrates the necessary optical setup for the beacon and receiver system.

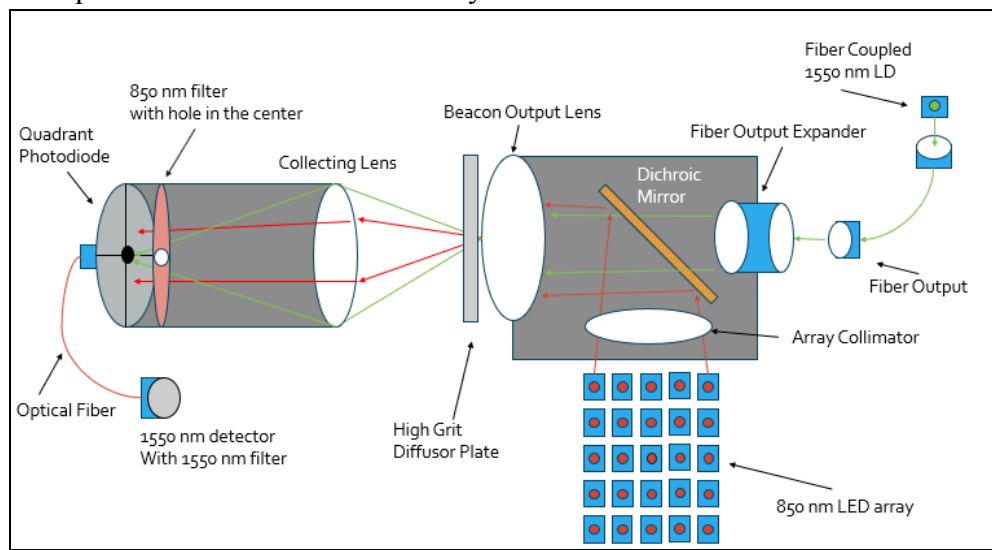


Figure 2.5.2-1: Diagram of the optical design

2.5.3 Software

There will be two programmable MCU's controlling both the beacon side and the receiver side. This will be the bulk of the programmable software in this project. The first step is to initialize all system components, including the microcontroller (MCU), the laser diodes, quadrant photodiode (QPD), gimbals, and communication interfaces. To do this, we need to set up communication protocols (e.g., I2C, SPI), configure the QPD for signal detection, and set up control parameters for the motors. The next step is detecting the signals from the 25 lasers diodes (850 nm) and the telecomm tracking lasers (1550 nm).

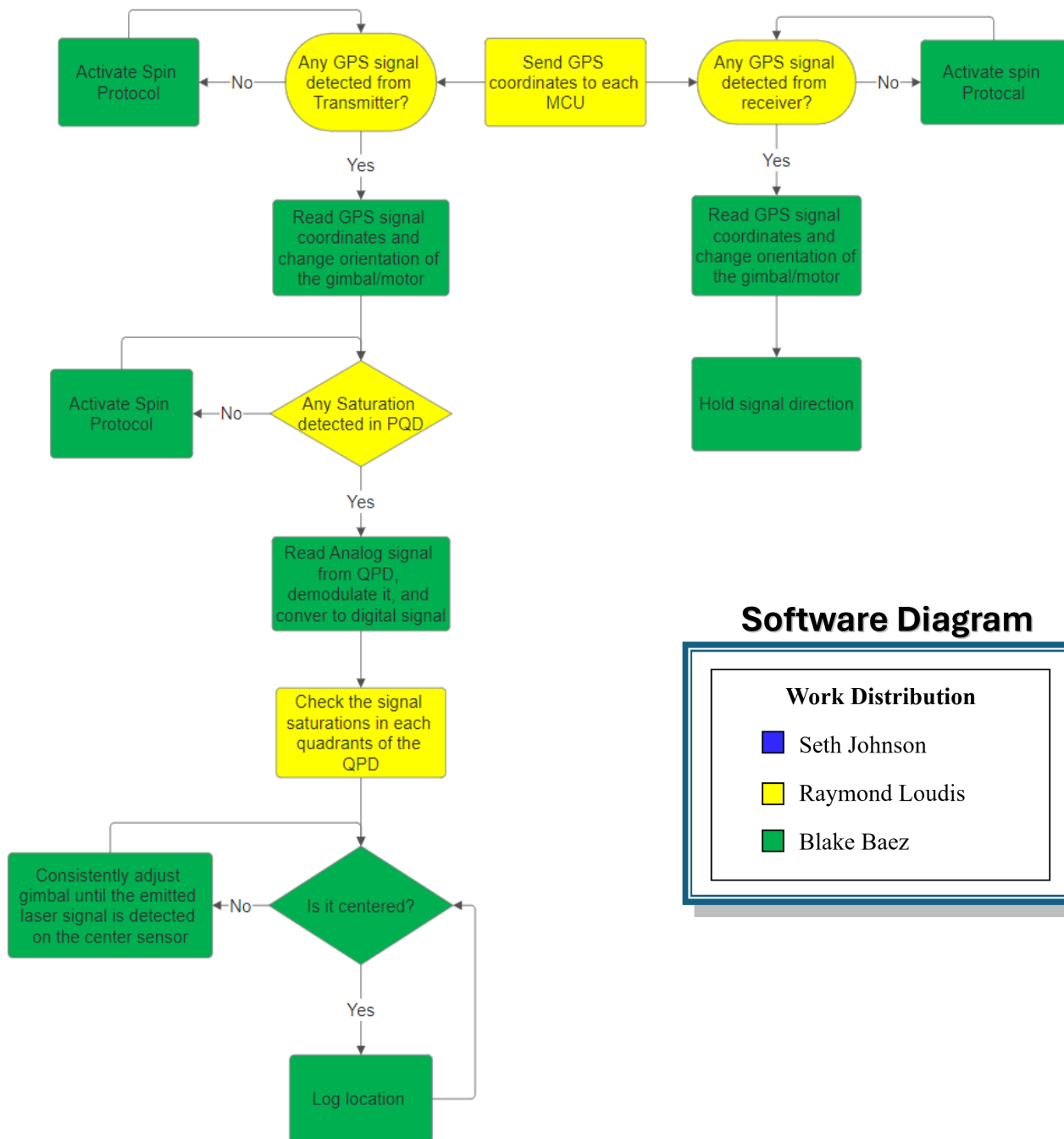


Figure 2.5.3-1: Software flowchart

To accomplish this, the MCU must take the analog signals from the QPD and use ADCs to convert analog signals to digital format. One must implement demodulation techniques to extract the pulsed signals from the noise. Then filter out ambient infrared sources. The software side of that is to apply digital filtering to eliminate remaining ambient IR interference. The tracking algorithm must also only accept the 1550nm, while the 850nm laser input is used for centering from a general location. The way it will be done is by computing the position of the laser beam on the QPD using the differences in signal intensities from the four quadrants. To move the servos, The MCU will contain a PID control algorithm to generate motor movement commands based on the position data. This PID control algorithm will be done using Ardupilot. It is important that the movement is smooth and precise to avoid signal loss.

As this project goes on, it's important that the code logs critical data and ensures reliable communication. This means logging position data, signal strength, motor movements, and system status for debugging and analysis. The system must be able to handle faults effectively to ensure its reliability. This can be done by implementing self-check routines to verify the functionality of all components at startup. It is also somewhat important to efficiently manage power consumption. This system can be used with a battery.

3 Research and Part Selections

3.1 Similar Projects

There are a total of 3 comparative projects that will be addressed in this section. The first two share likeness in optical alignment using a QPD while the last one focuses on real time tracking using a 2-axis gimbal such as the Gremsy T3V3 proposed to be used on our receiver. However, the gimbal was dismissed and found to be too difficult to implement.

The first related project that we will be addressing is a study conducted for Free-Space Optical (FSO) communication between a Ground Control Station (GCS) and a UAV [1]. I bring this study up first because it shares likeness in both optical tracking and using a 2-axis gimbal for tracking in real-time. This study proposes, much like our own, that we will be able to use a QPD to align the laser diodes on our beacon. Additionally, the proposal of a circular search pattern to obtain line of sight is similar to our "spin-protocol" listed above in the software flow chart. The experiments conducted in this study show that by using a laser generator, QPD sensor and driving actuators it is possible to control each side of their GCS and UAV independently to achieve perfect optical alignment for FSO communication [1]. However, there are some differences between our approach and the one used in this study.

The first key difference is that their design is focused on a communication between a GCS and UAV where our project hopes to be able to achieve this alignment between two UAVs. Another key difference is that we are hoping to reduce the time and increase the

range of target acquisition using GPS and LoRa modules. Furthermore, where this study transmitted the positional information received from the GCS photo diode to the UAV, we believe that positioning the beacon with the GPS and IMU will be sufficient for optical alignment and tracking. It should be noted that if our approach lacks the necessary accuracy we may consider a similar solution in both the beacon and receiver utilizing the data obtained from the QPD. Additionally, the distance between the transmitter and receiver for this study were only a distance of 100m, about a tenth of what we hope to demonstrate on our finished product.

The next comparative project that will be addressed is a study to construct a linear closed loop feedback system that compensates for beam wander in medium-range optical link applications or a Laser Beam Stabilization (LBS) system [2]. Unlike the previous study, the distance implemented here is 10km, a bit closer to our desired distance. Issues of longer distance transmission are brought to attention such as beam wander, which is caused as the optical beam propagates through free space and encounters atmospheric interference or turbulence.

The turbulent medium consists of large and small scale eddies that act as prisms of varying refractive nature. The LBS system is to compensate for beam wander in a closed loop fashion, the methodology used in their optical alignment are of note and may need to be implemented into later designs for our own project given further testing. This case study shows the implementation of fast steering mirrors mounted onto actuators that reflects the focused beam onto a QPD. Essentially the technique of actuating a mirror for fine tune adjustments could be promising for our attempt of obtaining and maintaining line of sight between our beacon and receiver.

Moving on to our last comparative project, a paper on the development of a gimbal control algorithm for real-time object tracking and location acquisition from a UAV [3]. The algorithm proposed by this paper deals with two scenarios, one offering gimbal pointing at a given coordinate and the other is to align an optically with a certain point in an image plane. This paper covers the implementation of GPS systems with an IMU for position orientation for their first scenario, similar to our own, however their algorithm for tracking is designed around user target selection. Essentially the gimbal is rotated and moved into position until the user has determined line of sight, once the object of interest is selected it is able to control the gimbal to keep the object within a designated camera frame. Although our project aims to be a bit more autonomous, the algorithms used for gimbal tracking control and the use of GPS systems for positional approximation, provide us insight into what lies ahead for our own project.

3.2 Constraints Due to Engineering Specifications

The first minor constraint this project must deal with is keeping the receiver and the transmitter each under 5kg. This was more of a soft constraint that was mentioned by this project's sponsor. All the components including the power supply, servo motors, MCU/IMU, LED array, GPS and telemetry modules, and lenses must be under 4kg for

the transmitter side and 6kg for the receiver side. This will ensure that our 20kg and 35kg servo motors will be able to move at their desired speed and not run into stalling issues.

Although not a strict requirement, this constraint drives the selection of lightweight materials and components. To ensure that the beacon and receiver devices are able to meet this requirement we plan to 3D print lightweight chassis which all of our PCBs and modules will be outfitted/secured to. If issues in weight would have arisen we then would have simply swapped out to larger servo motors but luckily this did not occur.

Most of the weight will be reserved for the mounting components, lenses, and laser diodes for the transmitter and receiver. The reason is that the rest of these parts will not reach 1kg for the receiver or transmitter side. To give a part selection example for the transmitter side:

- **Micro-Controller:** ESP32 S3 Mini (12g)
- **GPS module:** GY-NEO-6M (17g)
- **telemetry module:** RYLR890 (3g)
- **IMU:** QMC5883L (.5g)
- **PSU:** 7x 13g (91g)
- **LED Array:** (13g)
- **Optical Rig:** (1.2 kg)

As for the receiver side part selection:

- **Micro-Controller:** ESP 32 S3 Mini
- **GPS Receiver module:** GY-NEO-6M
- **Telemetry module:** RYLR890
- **IMU:** QMC5883L (.5g)
- **PSU:** 2x 13g (26g)
- **Receiver PCBs:** (42g)
- **Optical Rig:** (.75 kg)

The total weight for either side does not come close to encroaching the limits previously mentioned. From this we were able to determine that our choice in motors was feasible in regards to torque, however accuracy is an issue that will be addressed in a later section.

With the weight limit in mind, the system must be robust. It will be constantly moving around, and it must be able to consistently keep the same precision. It is imperative that mounting parts for the receiver and transmitter contain lightweight alloys or composite materials that can help meet the weight constraint without compromising strength.

Another minor constraint is power consumption. The employer did not give this project a restriction on power limits. It was only mentioned in passing that this system will never be used for more than half an hour. It was also stated that running the system for half an hour was not an actual requirement either. Regardless, the battery must be large enough to run the system for a couple of minutes. The bigger the battery, the more it could take

out of our budget. The battery is not factored into our weight limit however, as the power supply will be changed around for this system when in use for certain applications as stated by our employer.

Efficiency is also important. While the system must have a moderate amount of power usage so that a battery with a huge amount of Watt-hours is not needed, the system must also operate within the power limits of available batteries or power supplies. Components must be chosen for their energy efficiency. Low-power microcontrollers, energy-efficient sensors, and optimized firmware can help reduce overall power consumption.

For that reason, we have chosen the ESP 32 S3, which not only consume little power, but they also both offer power-efficient modes as well. At peak usage the ESP32 S3 Mini draws about 240 mA, at an operating voltage of 3.3V that gives us roughly .8Wh without using power-efficient modes.

One of the most important parts in terms of power usage is the LED array. Unlike the motors , GPS, microcontroller, etc., the laser diodes will constantly be running. The average consumption of a one-watt 850nm laser diode is 1 Watt-hour. There will be an array of 25 of these diodes so that is 25 Watt-hour on top of the 1 Watt-hour 1550nm laser diode. It is also important to note that these laser diodes will have 10 kHz track loop (without IMU compensation) and 100 Hz track loop (with IMU compensation). This changes the average power consumption to as much as 10% from the given numbers above. This is based on the duty cycle of the laser diodes for both cases.

The most important constraint that this group had to work with is the precision and accuracy of the system. This is the focal point of the project. The receiver side must be able to accurately track the transmitter side up to 10 microrad track accuracy (stationary), 500 microrad track accuracy (moving), and capable of tracking to within 5-degrees of the solar corridor. The 850nm LED signal must evenly saturate the QPD and the 1550nm laser signal must go through a center pinhole and onto the 1550 nm detector. This will drastically raise the costs of the parts.

Since the parts of the system must not only be precise, but accurate as well, the mounting parts must be fast, smooth, and stable. It will mean nothing if the receiver can accurately track the laser signal stationary but cannot hold onto that signal because it isn't stable, or the signal is moving around. This meant that the encoders with dc motors and or gimbals must have brushless dc motors or servo motors. This limits a lot of potential motors and gimbals and raises the price.

To meet the 10 microrad track accuracy (stationary), 500 microrad track accuracy (moving), and capable of tracking to within 5-degrees of the solar corridor, the resolution of the encoders must be around 4096 pulses per revolution on the receiver side. It is not feasible to have servo motors with this level of accuracy. This is why the group decided to have the receiver mount outfitted with gears to compensate for a lack of accuracy. As of now designing and printing the gears have begun but if the implementation becomes

too difficult or out right infeasible we will abandon this idea and focus on specifying the pulse widths rather than passing angle values to the `servo.write()` function.

Signal processing was also important and even factored into the precision and accuracy of the system. The MCU must process the signals accurately and quickly. The telemetry module was to be also able to properly send the signal out, so the microcontroller can achieve this. For this reason, the module should be rated above distances that the project is aiming to do. This will give the system insurance that it can successfully send signals despite interference.

As for the MCU/IMU, it must be able to handle real-time signal processing, requiring efficient coding practices and possibly the use of hardware accelerators. All the while filtering algorithms must distinguish the laser signal from noise with precision. This aspect of the project must be hard real-time. Failure could mean incorrect tracking and no signal, making the system a failure.

For the optical design, the main constraints we'll have to work around are the divergence angle of the beacon, the field of view of the receiver, and the optical path length of the receiver. The respective values for these constraints are:

- 45-degree divergence angle of the beacon.
- 10-degree field of view angle of the receiver.
- 2 inch optical path length of the receiver optics.

These constraints are necessary for the success of the project, even if they seem arbitrary. For the beacon, the divergence angle is such that there's a wide angle where the light source is visible, but not so wide that the output power is spread too far. If the divergence were 30-degrees, it would be too narrow for practical detection. If the divergence were 60-degrees, the power would be spread too much and would be too difficult to detect. A divergence angle of 45-degrees is a good in-between so that the beacon is visible and detectable. Similarly, for the field of view of the receiver, 5-degrees would be too narrow, and 15 or 20-degrees would be too broad for accurate tracking, which makes 10-degrees a good in-between for accurate tracking.

The optical path length for the receiver is arguably the most arbitrary, however keeping the path length short allows for more portability and smaller focused spot sizes for the beam, which lead to more accurate tracking and more efficient product.

3.3 Technology Investigation and Selection

3.3.1 Exploring Motion Control Solutions

One of the first and biggest considerations that group had to make is how to move the QPD and the laser diode array. There are many ways we can practically move objects in the x and y plane but there are many complex-mirror based options as well. Some of the options we look at are galvanometer-based systems, voice coil actuators, MEMS (Micro-Electro-mechanical Systems) mirrors, stepper motor systems with precision

stages, piezoelectric actuators, electro-optical deflectors (EOD), rotary encoders with servo motors.

The Galvanometer-based system, although highly precise with extremely fast response times, was immediately out due to complexity and cost. We believed it would put us over budget when looking up parts. Even if we could fit it into our budget, the online support and complexity of these would hinder the development of our project. Even without the high cost, they would not be practical. They work based on mirrors, motors, and the servo. The problem is if one wants to have a wider range of motion, you must have more motors and mirrors. This would just be impractical.

The electro-optical deflector is similar in the sense that it uses mirrors as well. However, it uses electro-optic effects instead to deflect the light beam without moving parts. This cannot work because of several reasons: it is too expensive, complex, and it can only move the laser from specific predetermined angles.

Voice coil actuators are used for other applications besides sound. They provide high precision and fast response times as well. Their biggest downside is limited range of motion. The average range of motion for a small VCA is in the millimeter range. That will not work for this project. Somewhat similar is the Piezoelectric actuator. Piezoelectric actuators are a device that use piezoelectric effect to provide small, precise movements. These are also in the microscopic effective range and would not meet the needs of this project.

Another mirror-based system is the MEMS. This was another method that was immediately dismissed. The degree of motion was exceedingly small and for much more precise movement that was what we needed. There are no ways for the average person to even build this machine for an application such as this.

The last two options are the ones this group has settled on, gimbals and rotary encoder with DC motors. They both meet the requirements for this project. They are both affordable, have the right amount of precision we need, and are far less complex than most of these options. The gimbal would be ready as is, but the difficult part would be coding the algorithm to use it. Another problem was being able to communicate with the gimbal. Most gimbals have their own software. Some of them will have an SDK or a library for communication available on github. The problem with the motor we are using is that, although there is an SDK available for it, it was not developed by the company itself, but by some college student in the US. DC motors are simpler in nature, but they would need some assembly and mounting work. They only spin on one axis so two are needed. In the context of servo motors, they do not have an actual rotation of 360 degrees, despite what they are labeled as, and will be typically 20 or more degrees off of the number mentioned in the datasheet. It is a matter of hooking them up together and being able to maintain that precision.

The group decided to pool all the data that was possible on each system for a receiver mount available on the market. All the irrelevant data was removed and the relevant data and or most important data was compiled into a comprehensive list. Every system that

was explored in the above paragraphs was included in this list. The graph comparison can be found below:

Receiver Mount Options					
	AVG Precision	AVG Power needed	AVG range of motion	Cost	AVG Weight
Galvanometer-based system	±0.5 mm	around 10-50 watts	Can cover 360 degrees in a 2D plane.	\$2,000 - \$10,000	1kg - 3 kg
Voice coil actuator	Sub-micron to micron-level precision	Around 10-100 watts	Limited to a few millimeters to centimeters in linear motion.	\$1,000 - \$5,000	0.5 kg - 2kg
Piezoelectric actuator	Nanometer to sub-micron precision	less than 10 watts	Limited to a few micrometers in linear motion.	\$500 - \$2,000	0.1kg - 0.5kg
MEMS	Sub-micron to micron-level accuracy	less than 1 watt	Limited range, usually a few degrees of rotation or micrometers of linear travel.	\$10 - \$500	Less than 0.1 kg
Gimbals	Around 0.01 degrees	5-100 watts	360 degrees in yaw, pitch, and roll	\$500 - \$5,000	1kg - 5 kg
Rotary encoder with DC motors	Around 0.01 degrees	5-100 watts	360 degrees in one 2D plane.	\$100 - \$1,000	0.5kg - 2 kg

Table 3.3.1-1: Comparison of mounts for the receiver.

For the receiver rotary encoder with DC motor, the group came up with several different brands that could work with this project. Two of the biggest considerations in choosing rotary DC motors is, one the motion profile of the application and two its stall torque. Stall torque is the torque load that causes the motor to stop moving or “stall”. The brands are as follows: Maxon Motor, Faulhaber, Oriental Motor, Dynamotion, Pololu, and US Digital. We have chosen two options: the Maxon Motor and the Faulhaber.

The reason we didn’t pick US digital is because they generally don’t sell motors on the level of precision we need for the receiver, just the encoders that they can be paired with. We have picked two brushless DC motors from the Maxon motor and Faulhaber Motor companies that we think would suit our needs. The rotary encoder with motor we picked from Maxon Motor is the DCX series with encoder (specifically the Maxon DCX 22 S with Encoder (High Precision Version)). Below is a graph comparison of the receiver side motors and encoders:

Receiver Side Servo Motor Consideration	
Brand/Type	Brand/Type
<u>Feetech</u> <ul style="list-style-type: none"> • Motor Type: DCX 22 S 	<u>Mallofusa</u> <ul style="list-style-type: none"> • Motor Type: Mg995 Servos
Voltage	Voltage
<ul style="list-style-type: none"> • Nominal Voltage (motor): 5 – 7.4 V • Supply Voltage (encoder): 3.3 V or 5 V 	<ul style="list-style-type: none"> • Nominal Voltage (motor): 4.8 – 7.2 V • Supply Voltage (encoder): 3.3 V or 5.5 V
Resolution	Resolution
<ul style="list-style-type: none"> • Dead band width: alleged 3 microseconds (20 microseconds actual) 	<ul style="list-style-type: none"> • Dead band width: alleged 5 microseconds (20 microseconds actual)
Operating Temperature	Operating Temperature
<ul style="list-style-type: none"> • 0°C to +55°C 	<ul style="list-style-type: none"> • 0°C to +55°C
Weight	Weight
<ul style="list-style-type: none"> • Motor: 66g 	<ul style="list-style-type: none"> • Motor: 55g
Current	Current
<ul style="list-style-type: none"> • No load current (motor): 19.6 mA • Stall current (motor): 2 A 	<ul style="list-style-type: none"> • No load current (motor): 39 mA • Stall current (motor): 1.3 A
Cost	Cost
<ul style="list-style-type: none"> • Around \$35 dollars per motor 	<ul style="list-style-type: none"> • Around \$25 dollars per motor

Table 3.3.1-2: Comparison of DC motors/encoders for the receiver.

These are the two brands that had one of the highest resolutions that met within our budget, met within our loose weight requirement, and consumed a reasonable amount of power. Since the sponsor would not pay for this project and we did not have a lot of money, price became a much more important factor. At this price point, a lot of the motors were very similar and likely from the same factory. This posed many problems such as the illusion of choice. Another issue in this price point was getting knockoff motors. We initially got the Tower Pro's, but immediately realized that it was not working off of 3.3v logic. It was a rebranded old servo motor that operated off of 5v logic.

Time was another constraint. Since we abandoned the gimbal so late, we did not have a lot of time to select many of our parts. This is confined to Amazon.com as well. This meant the life cycle of many of the motors we checked out was very short. For example, Our first choice of servo motors was the feetech. It met our budget as well as provided plenty of torque and strength in order to hold our mounts with no worries. It was no longer available on amazon and there were no other 35kg digital servo motors available in that price range either. This meant we had to select between the other 20 - 20kg motors.

It is important to note that this project needs two of these motors since it needs to be able to rotate in the x-plane and y-plane. This meant that each motor needed a mount. This mount is on a standard called an aluminum multifunction servo bracket set. This includes an aluminum mounting bracket, screws, nuts, a ball bearing, and an aluminium u mount. Four of these were bought for the initial gimbal mount.

The transmitter rotary encoder with a DC motor could be much cheaper and less precise. This means we had a much wider market and much more brands to choose from. We also had the option of stepper motors, dc motors, and servo motors instead of just brushless since high precision is not as required. Many of the initial brands we looked at were: NEMA Motors, Pololu Motors, Oriental Motor, Hitec, Adafruit Motors, Odrive, StepperOnline, TowerPro, TT Motor, DSServo, and Seed Studio Motors.

Since these motors were bought much further in advance, we decided to choose the most widely available and cost effective parts. These brands are the most widely available for precise applications (they both offer brushless DC motors or servo motors). Although some of the other motor options are for precise applications such as TT Motor, and Stepper Online. The motors they offer are more geared towards other applications, eg. robotics. DSServo and Tower Pro also support a wide range of connectors so the motors can be mounted onto each other as well as other objects.

While high-end motors like those from Oriental Motor offer superior accuracy, they significantly increase the project's cost. Bellow is a comparison of the specifications of the two encoders with dc motors that we chose:

Transmitter Side Servo Motor Consideration	
Brand/Type	Brand/Type
<u>DSServo</u> <ul style="list-style-type: none"> Motor Type: DS3218MG 	Tower Pro <ul style="list-style-type: none"> Motor Type: Mg995 Servos
Voltage	Voltage
<ul style="list-style-type: none"> Nominal Voltage (motor): 12 V Supply Voltage (encoder): 3.3 V or 5 V 	<ul style="list-style-type: none"> Nominal Voltage (motor): 4.8 – 7.2 V Supply Voltage (encoder): 3.3 V or 5.5 V
Resolution	Resolution
<ul style="list-style-type: none"> Dead band width: alleged 3 microseconds (20 microseconds actual) 	<ul style="list-style-type: none"> Dead band width: alleged 5 microseconds (20 microseconds actual)
Operating Temperature	Operating Temperature
<ul style="list-style-type: none"> -15°C to +70°C 	<ul style="list-style-type: none"> 0°C to +55°C
Weight	Weight
<ul style="list-style-type: none"> Motor: 60g 	<ul style="list-style-type: none"> Motor: 55g
Current	Current
<ul style="list-style-type: none"> No load current (motor): 200 mA Stall current (motor): 5.5 A Supply current (encoder): 22 mA 	<ul style="list-style-type: none"> No load current (motor): 39 mA Stall current (motor): 1.3 A
Cost	Cost
<ul style="list-style-type: none"> Around \$30 per motor 	<ul style="list-style-type: none"> Around \$25 dollars per motor

Table 3.3.1-3: Comparison of DC motors/encoders for the transmitter.

Sadly, one of the main reasons we went with DSServo, is that we got scammed from a Tower Pro knockoff. Time and money was wasted, however there was one upside.

DSServo offered up to 270 degrees of alleged motion (this ended up being 243 degrees and was within expectation). Most other motors listed within this price range on amazon only had about 180 degrees of motion like the Tower Pro. So although a little extra money was spent, this allowed a greater range of motion for our gimbal to operate in.

3.3.2 Exploring MCU and IMU Solutions

The next step for this project was to find a suitable MCU and IMU. Our employer wanted a reliable setup. Some of the models that were considered are as follows: the Pixhawk 4, Holybro Durandal, and the Cube Orange+ Standard Set ADS-B. All three of these brands and their corresponding products are an MCU and IMU combination. All of these brands contain a magnetometer, CAN, I2C, UART, SPI, USB, as well as a barometer chip.

The main consideration that this group has to take into account is what standards and software would be ideal choices to accomplish the requirements of this project. This is important because each of these IMU/MCU combos uses different software and standards. The Pixhawk 4 uses that hardware platform PX4 with the NuttX OS. We would have to choose the QGroundControl software to work with as well. Meanwhile, the Cube Orange+ can work with both the PX4 hardware platform and Arduipilot. This flexibility makes it an enticing candidate.

All that research was for naught however, as the Cube Orange and Pixhawk we had were not able to send data using the MavLink library over UART. Both MCU boards were extensively tested and both were able to communicate on a basic level, such as the esp32 receiving heartbeats. The problem arose when we connected our GPS and IMU combo, the Here 4 +. The exact problem was never pinpointed but the issues were that there were problems interpreting the signal.

Similar issues were happening with the Gremsy T3V3. This was one of the gimbals that were donated to the project but could not be used because it had issues communicating with the esp 32. Even when connected to our MCU's, the Mavlink protocol also had issues reading the data.

Since we already had both of our potential MCUs, we had to take another hit to our personal budget. Although we didn't need to buy an MCU, we needed to buy an IMU chip and couldn't use any that Knights Lab could donate. This left us with limited time and we searched for the cheapest IMU since accuracy was no longer our biggest concern for the IMU data. This left two possible chips: The GY-271 QMC5883L from WWZMDiB and the BMI270 (Qwiic) from SparkFun. Both were around the cheapest we could find on amazon and that would ship within the week. The biggest difference was that the Sparkfun was a well known and reputable brand and WWZMDiB was not (just a random custom chip made by a factory).

IMU Chip	
Brand/Type	Brand/Type
<u>WWZMDiB</u> <ul style="list-style-type: none"> IMU: GY-271 QMC5883L 	<u>SparkFun</u> <ul style="list-style-type: none"> IMU: BMI270 (<u>Qwiic</u>)
Gyroscope	Gyroscope
<ul style="list-style-type: none"> 3-axis 	<ul style="list-style-type: none"> 3-axis
Accelerometer	Accelerometer
<ul style="list-style-type: none"> 3-axis 	<ul style="list-style-type: none"> 3-axis
Magnetometer	Magnetometer
<ul style="list-style-type: none"> 3-axis 	<ul style="list-style-type: none"> 3-axis
Communication Protocol	Communication Protocol
<ul style="list-style-type: none"> I2C 	<ul style="list-style-type: none"> CAN (or I2C)
Cost	Cost
<ul style="list-style-type: none"> \$8 	<ul style="list-style-type: none"> \$15

Table 3.3.2-2: Comparison of IMU chips.

We ultimately decided to go with the QMC5883L model. The reason is because of the price and the communication protocol. We didn't want too many parts that used UART, but also a more robust and supported communication protocol for IMU chips.

The receiver microcontroller must be robust and powerful. The reason is that the group decided that it would be most efficient if it did the heavy lifting in terms of precision. It needs to accurately track the laser signal, which includes fine motor control as well as quick and accurate reading of the QPD.

The transmitter microcontroller is different. It does need the same level of communication protocol, processing speed, or robustness. This does not mean that our choices can have slow processors or low flash memory. The good news is that the cheap microcontroller market is saturated, and innovation is booming. This allows the project to much more easily reach budget goals.

The brands that got consideration had to be easily bought, like the transmitter, had to be Arduino compatible (this means that we are using C code for this microcontroller as well), and compatible with all of the communication protocols the group will be using. Luckily, most microcontrollers under \$50 dollars use UART, USB, and I2C. All of the chosen ones use relatively low power and contain SRAM. The following microcontrollers are the types that the group has decided on: ESP 32, Arduino Due, Teensy 4.0, and Raspberry Pi Pico. Their specifications are listed below:

Transmitter Microcontroller Consideration Part 1					
Brand/Type	Processor	Flash Memory	SRAM	Communication Protocol	Average Cost
ESP 32	Dual-core Tensilica LX6 microprocessor, up to 240 MHz	4 MB	520 KB	3 UART, 3 SPI, 2 I2S, 2 I2C, 1 USB	\$10 - \$20
Arduino Due	Atmel SAM3X8E ARM Cortex-M3, 84 MHz	512 KB	96 KB	4 UART, 2 SPI, 2 I2C, 1 CAN, 1 USB	\$35 - \$50
Teensy 4.0	ARM Cortex-M7, 600 MHz	2 MB	1 MB	7 UART, 3 SPI, 3 I2C, 1 I2S, 1 CAN, 1 USB	\$20 - \$25
Raspberry Pi Pico	Dual-core ARM Cortex-M0+, up to 133 MHz	2 MB	264 KB	2 UART, 2 SPI, 2 I2C, 1 USB	\$4 - \$6

Table 3.3.2-3: Comparison of transmitter microcontrollers Part 1.

In terms of CPU performance, all the microcontrollers seem sufficient, even the Arduino Due. The issue comes when looking at flash memory and SRAM. With so little at 512 KB of Flash Memory and 96 KB of SRAM, the group is not confident it could complete its necessary calculations or algorithms before overflowing and having to reboot. For unknown reasons as well, it is by far the highest in price among the options. For these main two reasons, it's the least likely to be chosen.

In terms of communication protocol, all the microcontrollers are excellent except the Raspberry Pi Pico. Two UART connections do not leave much in terms of headroom. It also only has 26 GPIO and 3 channels of 12-bit ADC. Although it is unlikely this group will use all of these pins and channels, it is important to note it has much less than its competitors. As this project is further developed. It may be necessary to attach parts that require analog connections or GPIO pins.

One standout specification to note is that there is only one microcontroller type that has wireless connectivity built in: the ESP 32. This is important because it allows for future

possible connectivity options. Some of the parts this group already selected can use Wi-Fi and or Bluetooth. The remaining specs on the microcontrollers are listed in the graph below:

Transmitter Microcontroller Consideration Part 2						
Brand/Type	Input Voltage	Wireless Connectivity	Current consumption	Operating Temperature	Digital Inputs/Outputs	Analog Inputs
ESP32	3.3V	Wi-Fi 802.11 b/g/n, Bluetooth v4.2 BR/EDR and BLE	Deep sleep: 10 μ A Light sleep: 0.8 mA Active: 160-240 mA	-40°C - +125°C	34 GPIO pins	18 channels of 12-bit ADC
Arduino Due	3.3V	None (requires external module)	Active: 130 mA	-40°C - +85°C	54 GPIO pins	12 channels of 12-bit ADC
Teensy 4.0	2.8 to 5.5 VDC	None (requires external module)	Active: 100 mA	-40°C - +85°C	40 GPIO pins	14 channels of 12-bit ADC
Raspberry Pi Pico	1.8V to 3.3V	None (requires external module)	Active: 91 mA	-20°C - +85°C	26 GPIO pins	3 channels of 12-bit ADC

Table 3.3.2-4: Comparison of transmitter microcontrollers Part 2.

The operating temperature is also extremely high on the ESP 32. It can reach 125 degrees Celsius. This is a cut above the other microcontrollers. This is all rolled in a package cheaper than almost every other microcontroller except the Raspberry Pi Pico.

3.3.3 Exploring GPS Solutions

GPS was another secondary addition as it will be used to support the precise tracking. Nonetheless, the GPS must be accurate enough so that the receiver side does not have trouble locating the laser signal. Any module that has accuracy within meters or above will not be sufficient. The GPS should allow the laser diode array to reliably contact the QPD so that the receiver does not have to search for a signal in the general area that the GPS reported. A more accurate GPS also means that it could calculate the position quicker. The group decided that highly reliable positioning and accuracy within centimeters was required.

Update rate was another important consideration. The update rate of the GPS module

determines how frequently it provides position updates. Higher update rates are required in this project because the transmitter can be moving as well as said application requiring real-time position data. The ability for the system to reliably transmit between microcontrollers while they are being moved also requires a module with great sensitivity.

The project description notes that not only is the system a real-time system, it must be secure and ideally encrypted. With accuracy and stability in mind, these GPS modules will need to have their own suitable GNSS module. With this, a selection such as the Here 4+ was available. The problem with this is that it used the CAN protocol for communication and was set up to communicate via MavLink as well. It was initially meant to be run on the Ardupilot software that we were no longer using. This meant we had to buy our own gps modules in a limited amount of time.

Since our budget was low, we had to go through 3 main considerations, speed, accuracy, and cost. Cost would be the most significant factor and the next two being about equal. Since the gps update would be rather slow at this price point no matter what gps module we bought at this price point, we decided to go for the one that had the most accuracy listed on its spec sheet.

GPS Module Consideration						
Brand/Type	Communication Protocol	Channels	RTK Accuracy	Constellations Supported	Update Rate	Cost
GY-NEO6MV2 NEO-6M GPS Module	UART, I2C	50 channels	Not supported	GPS	Up to 5 Hz	About \$10
Ublox NEO-M8N GPS Module	UART, I2C, SPI	72 channels	0.025 m (with RTK corrections)	GPS, GLONASS, Galileo, QZSS, SBAS	Up to 10 Hz	Around \$30
Ublox ZED-F9P	UART, USB, Bluetooth, Wi-Fi (via expansion)	184 channels	Horizontal 10 mm + 1 ppm, Vertical 20 mm + 1 ppm	GPS L1C/A, L2C; GLONASS L1OF, L2OF; Galileo E1-B/C, E5b; BeiDou B1I, B2I; QZSS L1C/A, L2C	Up to 20 Hz	Around \$200
Quectel L76-M	UART	33 channels	Not supported	GPS, GLONASS, QZSS	Up to 10 Hz	Around \$15 - \$20
Adafruit Ultimate GPS (MTK3339 chipset)	CAN, UART, I2C	66 channels	Not supported	GPS, SBAS	Up to 10 Hz	\$30 - \$40

Table 3.3.3-1: Comparison of GPS modules.

The unlisted main reason we chose the first module is because the accuracy out of the box can reach within 3 meters. For that price, it is plenty good and much better than the Quectel and adafruit. The one large downside is the update rate, or speed. It is 5 Hz, which is painfully slow. However we accepted this and designed the code around this to make up for the shortcoming. The algorithm I developed essentially tries to avoid pulling for gps data as much as possible. Once a lock is established, it will save the coordinates and move onto the next task.

3.3.4 Exploring Telemetry Module Solutions

Once the research was finalized on the GPS module, we began research on how to communicate between the two MCUs. The first thing each micro controller needs to do is find its own location, altitude, and acceleration. Once it has done that, it needs to be able to communicate said information to the other microcontroller. This project requires this to be done long distance and wirelessly. Some of the options we researched are listed below:

MCU Communications Consideration		
Type	Distance	Average Cost
Wi-Fi	Typically up to 100 meters indoors, longer outdoors depending on environmental conditions and antennas.	\$5 - \$20
Bluetooth	Around 10 meters for Class 2 devices (standard), up to 100 meters for Class 1 devices (extended range).	\$5 - \$10 (depending on version)
RF (Radio Frequency) Modules	Around a couple 100 meters, can be extended with external antennas.	\$5 - \$10
Cellular Modules	Coverage area depends on cellular network provider, can range from local to global coverage.	\$20 - \$60
LoRa (Long Range)	Can reach several kilometers in rural areas on average. This depends on how much noise is in the area.	\$20 - \$60
Satellite Communication	Global coverage, can communicate from anywhere on Earth.	\$400 - \$2000
Visible Light Communication (VLC)	Typically up to 10 meters, depending on the intensity of the light source and ambient light conditions.	\$20 - \$200
Infrared (IR) Communication	Generally up to a few meters (line-of-sight).	Less than \$5

Table 3.3.4-1: Comparison of Telemetry Modules.

The first requirement listed in this project requires that the system to work around 1 kilometer. This was later lowered to a 100 meter demonstration due to the power of our optical setup, but if we had stronger LEDs and laser diode, this system would work at 1 kilometer. This already excludes Wi-Fi, Bluetooth, most RF modules, visible light communication, and infrared communication. None of these options could reach over a couple hundred meters, especially with extensive setup and or, in the case of Wi-Fi, a physical connection of ethernet to the board itself. The group did see some modules done using Wi-Fi to connect to the microcontroller remotely and then said the micro controller communicates with the other microcontroller with Wi-Fi as well. This, however, would require the system to be in range of a base station. This system should be autonomous. This left the group with three options: LoRa, satellite communication, and Cellular modules. This was easily changed to two options as satellite communication was too expensive.

One important thing the group had to consider is that the system should be super reliable in any location as it is a real-time system. For one, the system should be able to work in most locations on the planet, including rural areas. Of the two options left that the group was considering, using a cellular module would require cell towers. LoRa is also one of the lowest power consuming of the researched modules.

It is also not necessary that the system can communicate worldwide as the limit is how far the QPD can reliably read the laser diode array. LoRa modulation is designed to be robust against interference and noise, which can be beneficial in environments where there might be other sources of electromagnetic interference (EMI). This includes near-infrared light sources or other telemetry signals.

Cellular modules can easily be affected by interference, such as buildings and other signals, especially when using LTE and above. Choosing modules that are compatible with multiple wavelengths adds to cost. The added complications with cellular. They tend to be lacking incompatible or easy to use communication protocols such as UART. For this reason, the group chose LoRa as our telemetry module.

There are many different LoRa modules in terms of performance. The ones the group looked at had to have a certain frequency band and were at least FCC compliant. Among the topmost used brands in this category, the group had to choose ones that had a balance in data rate, range and power consumption. Among these, some were knocked out that had no compatible communication protocol. The following were some of the telemetry modules that this group was considering: REYAX RYLR896, REYAX RYLR689, HopeRF RFM95W, Ebyte E32-433T30D, Ai-Thinker Ra-02, Dragino LoRa Shield, and TTGO LoRa32. Although there are many more available modules these are the ones we found within our budget and listed as in stock. A graph comparing their specs is listed below:

Telemetry Module Consideration Part 1					
Brand/Type	Input Power	Frequency Range	Operating Temp	Range	Cost
REYAX RYLR896	3.3–5.5 V	868 – 915 MHz	-40°C to 85°C	Up to 10 km	\$12–\$18
REYAX RYLR689	2.8–3.6 V	868 – 915 MHz	-40°C to 85°C	Up to 10 km	\$12–\$20
HopeRF RFM95W	1.8–3.7 V	137 – 1020 MHz	-40°C to 85°C	Up to 15 km	\$6–\$10
Ebyte E32- 433T30D	3.3–5.5 V	410 – 441 MHz	-40°C to 85°C	Up to 8 km	\$15–\$25
Ai-Thinker Ra- 02	1.8–3.7 V	410–525 / 862–1020 MHz	-40°C to 85°C	Up to 10 km	\$5–\$10
Dragino LoRa Shield	3.3 V	868 – 915 MHz	-40°C to 85°C	Up to 15 km	\$18–\$30
TTGO LoRa32	3.3–5 V	868 – 915 MHz	-40°C to 85°C	Up to 10 km	\$15–\$25

Table 3.3.4-2: Comparison of LoRa Modules part 1.

Concerning the range, just being able to transmit up to 1 or 2 kilometers does not mean it is good enough for our application. It needs to reliably communicate up to that distance without using a low power. This meant that the Ebyte was potentially not reliably enough for this project, theoretically (if it were later tested at around 1 kilometer or two).

Another issue is the cost. Our budget was very tight and every dollar counted. There was also the issue of buying multiple modules and or extras in case something went wrong. For example, we initially bought the REYAX RYLR689. The issue there was attaching pins to it. With so many pins being wired up so close, we had certain complications and ultimately decided that the SPI interface chips would just be a hassle. We then bought the UART version and a newer model from the same brand. Luckily it was around the same price

Although UART was much easier to connect pin wise, the actual code would be harder. We had to make sure we had compatible libraries for the chip. Each of the libraries we used had their issues, but we ultimately got it working. Another small issue was the trade off. Although these LoRas had excellent range, the trade off was speed. Most of the LoRa modules in the category had slow transmission speed so picking the most reliable one at long range seemed like the right choice.

Below are additional specs that the group weighed on when considering the right module for this project:

Telemetry Module Consideration Part 2					
Brand/Type	Current Consumption	Air Data Rate	Communication Protocol	Weight	Receive Sensitivity
REYAX RYLR689	Tx: ~120 mA Rx: ~12.5 mA	Up to 300 kbps	SPI	5g	-137 dBm
HopeRF RFM95W	Tx: ~120 mA at +20 dBm Rx: ~10.3 mA	0.3–300 kbps	SPI	1.4g	-148 dBm
Ebyte E32-433T30D	Tx: ~650 mA at +30 dBm Rx: ~15 mA	0.3–19.2 kbps	UART	8g	-138 dBm
Ai-Thinker Ra-02	Tx: ~120 mA at +20 dBm Rx: ~10.8 mA	0.018–37.5 kbps	SPI	4.5 g	-148 dBm
REYAX RYLR896	Tx: ~120 mA Rx: ~12.5 mA	Up to 300 kbps	UART	5g	-137 dBm
Dragino LoRa Shield (RFM95)	Tx: ~120 mA at +20 dBm Rx: ~10.3 mA	0.3–300 kbps	SPI	11g	-148 dBm
TTGO LoRa32 (SX1276)	120 mA / 10.8 mA	0.3–37.5 kbps	UART, SPI	~7 g	-148 dBm

Table 3.3.4-3: Comparison of LoRa Modules part 2

3.3.4 Optical Components

When choosing optical components, especially lenses, there are much less comparisons to make. For lenses, the only real factor to consider is the material, since other factors like focal length, diameter, and anti-reflection coating are design choices. The most common lens materials are N-BK7 and UV Fused Silica. Both materials can transmit the wavelengths we require, so choosing between them will be based on cost and damage threshold. The general pros and cons of both materials can be summarized in the table below.

Pros/Cons	N-BK7	UV Fused Silica
Pro	Lower Cost	Higher Chemical Resistance
Pro	Higher Transmission Band	Lower Scattering Loss
Pro	High Mechanical Damage Threshold	Higher Thermal Damage Threshold
Con	Lower Thermal Damage Threshold	Lower Mechanical Damage Threshold
Con	Higher Scattering Loss	Higher Cost
Con	High Density	Smaller Transmission Band

Table 3.3.4-1: Comparison of common lens materials.

Since UV Fused Silica has a higher damage threshold but costs more, we'll likely use that material for most of the transmitter system since the beacon will emit a high power. However, we can use the cheaper N-BK7 for the receiver side since the incident power will be much lower.

The other optical components that require comparison are the dichroic mirrors and diffuser plate. Dichroic mirrors will transmit a certain wavelength band while reflecting another, which makes them essential components to the optical design. Proper selection for dichroic mirrors is important for wavelength specificity, optical performance, quality/durability, and cost effectiveness. For these reasons we took special care in our selection and used the websites of reputable and modestly priced vendors like Thorlabs and Edmund Optics. The mirrors that we will be comparing will be 1 inch in diameter and must be capable of transmitting and reflecting the required wavelengths mentioned earlier in this paper. Furthermore, no diodes over the cost of \$350 were considered in the following comparison, this was to ensure that our budget was not overly tight this early on in the design process. Below is the table for our dichroic mirror comparison:

Company	Product ID	Transmission Band (nm)	Reflection Band (nm)	Cost
Thorlabs	DMLP1180	1260 - 1700	750 - 1100	\$230.46
Thorlabs	DMLP1000	1020 - 1550	520 - 985	\$310.06
Edmund Optics	87-039	985 - 1600	715 - 900	\$171.00
Edmund Optics	87-040	1035 - 1600	750 - 950	\$176.00

Table 3.3.4-2: Comparison of dichroic mirrors

While the Edmund Optics mirrors may appear to be much cheaper whilst transmitting and reflecting the required wavelengths, their listed diameters are 25mm rather than 1 inch (25.4mm). It's not a huge difference, however it can affect their compatibility with different optomechanical components. If compatibility is a non-issue, then we'll likely use the Edmund Optics 87-039. If there is a problem with compatibility, then we'll use the Thorlabs DMLP1180. Both mirrors are the cheaper option from their respective companies that will meet the requirements of this project.

The last optical component that requires comparison is the diffuser plate. Diffuser plates will help blend the laser diode array so that it appears as a single light source, however they can also absorb a lot of optical power whilst adding extra divergence to the beam. We'll have to select a diffuser plate that transmits the maximum amount of power whilst also applying the least amount of divergence. The plate will also require a higher damage threshold since it will be placed directly in front of the laser array. There are engineered diffuser plates that can accomplish this, however they are much more expensive than simple sanded glass. We'll once again be turning to Thorlabs and Edmund Optics and comparing the best options for engineered diffusers.

Company	Product ID	Transmission Efficiency (%)	Dispersion Angle (Deg)	Cost
Thorlabs	ED1-C20	90	20	\$142.56
Thorlabs	ED1-C50	90	50	\$142.56
Thorlabs	EDG5C20	90	20	\$280
Edmund Optics	47-680	85	0.5	\$457.50
Edmund Optics	55-849	85	5	\$457.50
Edmund Optics	48-507	85	15	\$457.50

Table 3.3.4-3: Comparison of engineered diffuser plates.

The Edmund Optics diffusers have a slightly lower transmission efficiency, but the dispersion angles are much better than their Thorlabs counterparts. The dispersion angles

being so much lower would be worth the lower transmission efficiency, but unfortunately the Edmund Optics diffusers are nearly double the cost of the most expensive Thorlabs diffusers. Because of the cost difference, it's most likely we'll use either Thorlabs ED1-C20, or Thorlabs EDG5C20.

3.3.5 Photodetectors.

There are a few critical pieces of optoelectronics for this project, and in this section we'll be comparing quadrant photodiodes sensitive to 850nm and standard photodiodes sensitive to 1550nm. When choosing photodetectors, the most important parameter will be the minimum detectable power. In order to determine the necessary optical output power for our beacon, we must calculate how the power disperses with distance as well as the minimum detectable power of our respective photodiodes. The minimum detectable power (MDP) of a photodiode is determined by its noise equivalent power (NEP) and the frequency of the incident signal. The relation between the MDP and the NEP is characterized by equations 3.3.5-1 and 3.3.5-2:

$$f_{BW} = \frac{0.35}{t_R} \quad (\text{Equation 3.3.5-1})$$

$$MDP = NEP \cdot \sqrt{f_{BW}} \quad (\text{Equation 3.3.5-2})$$

Where t_R is the rise time of the photodiode, and f_{BW} is the bandwidth of the system in Hz. Photodiodes, quadrant or standard, that are sensitive to 850nm are typically made of Si. While Si photodiodes aren't necessarily rare, we specifically need an annular QPD so that we can couple a fiber optic cable to the center. Unfortunately, we've only been able to find two photodiodes that meet this requirement and they're both from the same company. As such, we'll be comparing them both based on their respective NEPs and active area sizes. The Si annular quadrant photodetectors we have to choose from are listed below.

Company	Product ID	Active Area per Element (mm ²)	NEP (W/√Hz)	Rise Time (ns)	Cost
Advanced Photonix	SPOT-9D-0	19.5	1.9e-14	6	\$125.86
Advanced Photonix	SPOT-4D-0	1.7	8.7e-15	6	\$69.50

Table 3.3.5-1: Comparison of Si annular quadrant photodetectors.

Between both annular QPDs, the SPOT-9D-0 will serve our purposes better. With its larger active region size, it'll be easier to shape the 850nm beam to cover the whole active region whilst still keeping the 1550nm beam smaller enough to fit into the optic fiber. With the larger NEP, the SPOT-9D-0 has a minimum detectable power of 145pW, which will be important when choosing the LED array in section 3.3.8.

The next critical item for comparison is the 1550nm detector. Most photodiodes that are sensitive to 1550nm are made of InGaAs and have high responsivities at that wavelength. Luckily, 1550nm is one of the most common wavelengths for telecommunications. This makes it easy to find components with high responsivities and low dark currents for reasonable prices. Much like the Si quadrant photodiode, the key parameters we're comparing are responsivity at the specified wavelength, dark current, active region size, and cost. A collection of InGaAs photodetectors we've investigated are tabulated below:

Company	Product ID	Responsivity (A/W)	NEP (W/√Hz)	Rise Time (ns)	Cost
Thorlabs	FGA015	0.95	1.3e-14	0.3	\$65
Thorlabs	FGA01FC	1.00	4.5e-15	0.3	\$174.62
Edmund Optics	#55-754	0.95	4.5e-15	0.3	\$95
Edmund Optics	#62-271	0.95	8.42e-15	10	\$100
Excelitas Technologies	C30645EH	9.3	60e-15	40	\$318.15
Excelitas Technologies	C30662L-200	9.4	9e-14	0.3	\$203.7

Table 3.3.5-2: Comparison of InGaAs photodiodes sensitive to 1550nm

Based on our research, the Thorlabs FGA015 will be our best option for InGaAs photodiodes. The Excelitas Technologies photodiodes have better specs, however for the extra expense they aren't worth it when the other brands will suit our needs just fine. The Edmund Optics photodiodes have similar specs to the Thorlabs models, however the FGA015 will be easier to couple into a fiber optic cable and has satisfactory specs for a more reasonable price. Using our equations above, the FGA015 will have a minimum detectable power of 15nW.

3.3.6 Light Sources

When determining our light sources, we'll have to choose a component that can be detected by the minimum detectable power of the photodiode. We can predict how the output optical power will change with distance, as well as how much of that power will be picked up from by the tracker, by using equations 3.3.6-1, 3.3.6-2, and 3.3.6-3:

$$A_t = \pi \left(\frac{D_t}{2} \right)^2 \quad (\text{Equation 3.3.6-1})$$

$$A_B = \pi \left(d \cdot \left(\tan \left(\frac{\theta_{div}}{2} \right) \right) \right)^2 \quad (\text{Equation 3.3.6-2})$$

$$P_{inc} = P_o \frac{A_t}{A_B} \quad (\text{Equation 3.3.6-3})$$

In equations 3.3.6-1 and 3.3.6-2, D_t and A_t are the diameter and area of our tracker aperture respectively, A_B is the area of the beam projected by the beacon, d is the distance from the beacon to the tracker, and θ_{div} is the output divergence angle of the beacon. We know that the tracker aperture diameter will be 1 inch (25.4mm), the output divergence angle will be 45 degrees, and the distance from the beacon to the tracker will be 3m for indoor demonstrations and 1km for outdoor tests.

In equation 3.3.6-3, P_{inc} is the incident optical power on the tracker, and P_o is the output optical power of the beacon. Using our equations and known values, and the MDP of our photodiodes, we can determine that the output power of our light sources can be no less than 154mW and 164mW for the LED array and 1550nm laser respectively in order for the beacon to be detectable at 1km. Even if we meet the MDP of our photodiodes, it would be a noisy signal at that power level. Therefore we'll prioritize having slightly more power than we need since the output power can be limited simply by limiting the current across the light sources.

On the beacon end of the system, choosing the correct light sources is critical to ensuring that the signal is detectable by the receiver. We'll investigate different LEDs that emit 850nm light, laser diodes that emit 1550nm with adequate power, drivers for that laser diode. When choosing LEDs, the key parameters are the driving current and output divergence, also known as viewing angle or viewing half angle. Since the LEDs will be in a 5x5 array, the output optical power is not quite as important as the viewing angle. A higher output power will be a nice bonus, but a larger viewing angle would make the optical design exceptionally difficult. Most LEDs are usually cheap, so cost will also bear less weight in this comparison. The best LEDs we found are summarized in the table below:

Company	Product ID	Output Irradiance (mW/sr)	Driving Current (mA)	Viewing Angle (deg)	Unit Cost
Lumileds	L110-0850060200000	1150	1000	60	\$4.04
ams OSRAM	SFH 4258S	100	100	30	\$1.10
Vishay	VSMY98545ADS	380	1000	45	\$5.98
Würth Elektronik	15427285BA240	160	1000	120	\$2.46
Vishay	VSMY12850	16	70	80	\$0.71
Broadcom	ARE1-89D0-00000	600	1000	90	\$4.76

Table 3.3.6-1: Comparison of 850nm light emitting diodes

From our investigations, the ams OSRAM SFH 4258S appears to be the best option. All other LEDs either require too much input current, or the viewing angle is too high. For the SFH 4258S, the driving current is reasonable, and the viewing angle is small compared to other LEDs which will make the optical design a reasonable endeavor. With an output irradiance of 100mW/sr (which roughly equates to 21mW of output power), the array will have an approximate total output power of 535mW which will be detectable by the tracker at long distances.

The other key light source we'll have to compare is the 1550nm laser diode. Similar to the LED, the parameters we'll be comparing are output optical power, output divergence angle, driving current, and cost. Unlike the LEDs, output optical power will hold significantly more weight than viewing angle since the laser diode will be coupled to an optical fiber. 1550nm laser diodes are also significantly more expensive than average LEDs, so cost will also bear more weight in this comparison. The laser diodes we've researched can be summarized in the table below:

Company	Product ID	Output Power (W)	Driving Current (A)	Fast Divergence Angle (deg)	Unit Cost
Advanced Photonix	CVLL 95-90	10	30	24	\$236
Advanced Photonix	CVLL 350-90	25	75	24	\$254
Thorlabs	FPL1055T	0.3	0.75	23	\$424
Thorlabs	L1550G1	1.7	5	28	\$355

Table 3.3.6-2: Comparison of 1550nm laser diodes.

Based on our research, the Advanced Photonix CVLL-350-90 laser diode appears to be the best option. Despite the high driving current, we can still get a few watts of power by driving it at 2-5 A. This gives us a better ratio of watts-per-dollar than any other laser diode we've researched.

3.3.7 Energy Source Solutions

The choice for using a battery-based power should come as no surprise, seeing as the tracking system needs to be mobile it is only natural the energy source too should be mobile. However, there are many factors at play in choosing the correct battery for our power supply. These include but are not limited to: capacity, discharge rate, weight and compatibility with components. Our primary goal in battery selection was to ensure that it could supply both the drone and our tracking apparatus without compromising the safety of our components or the drone. Our reasoning behind using the same battery for both the tracker and the drone stems from our need to minimize weight and impact on

aerodynamics while ensuring a large enough capacity. Unfortunately, as we continue our design process a specific load has yet to be confirmed for the beacon. We have yet to reach a final decision on servo motors or determine the efficiency we can drive our 25 laser diodes. Despite these uncertainties we have been able to come up with a rough estimate for both our beacon and receiver, these approximations are made based on data sheets and specs listed in the Appendix . Also, we assume below that we can drive our diode array at an 85% efficiency and that our dynamic loads for both the gimbal and purposed servo motors are at 25% to accommodate a possibly long “spin-protocol” for target acquisition.

Device	Component	Needed Power (mW)	Operating Voltage (V)	Needed Charge (mAh)
Receiver	Aurelia X6 Standard [24]	444k	22.2	20k
	Gremsey T3V3 [25]	5450	14.5	375
	MCU, GPS, IMU, ADC [23][27]	1,052-1,500	4-5.7	263
	TOTAL			20,625
Beacon	Aurelia X6 Standard [24]	444k	22.2	20k
	Laser Diodes** [26]	6,000	3V	2000
	Servo Motors*		3-6	2000 (Max Rated)
	MCU, GPS, IMU [23]	1,000-1,425	4-5.7	250
	TOTAL			24,250

Table 3.3.7-1: Load Approximations

**The Servo motor Load is based on the average allotted current from parts researched in 3.3.1*

***The power consumption listed above does not account for modulation. Constant current value given.*

From the table above we can approximate that the receiver battery should have a capacity no smaller than 20,625 mAh and the battery for the beacon should have a minimum capacitance of 24,250 mAh. Additionally the size of the battery must be no larger than 200mm x 160mm x 325mm per the specifications provided on the Aurelia X6.

Furthermore, inconsideration of discharge rate, many drones use lithium polymer batteries which have appealing attributes such as higher discharge rates and lower weight in comparison to lithium ion batteries, despite the Li-Ion batteries being of smaller form factor. This high discharge rate is beneficial for both motors and driving laser diodes, as they require bursts of high current to operate. A brief comparison between battery technologies is given below. It should be noted that Li-Po batteries are most commonly used for drone applications and that this table considers technologies that would be implemented if the on-board battery was found insufficient and needed an additional power source.

Attribute	Li-Po	Li-Ion	NiMH	NiCd
Energy Density	High	High	Moderate	Low
Weight	Light	Moderate	Heavy	Heavy
Discharge Rate	High	Moderate	Low	Moderate
Cost	Moderate	High	Low	Low
Charging Time	Fast	Fast	Slow	Moderate
Operating Temperature (C)	-20 to 60 C *	-20 to 60 C *	0 to 50 C	0 to 45 C

Table 3.3.7-2: Battery Technology Comparison

**While comparing specific batteries there was variance found in safe operating temperatures. Please refer to the operating temperatures listed in the part comparison.*

Lithium battery types are the clear choice, they are lighter and have higher energy density. From this realization a part comparison was conducted while considering the minimum mAh for the beacon and receiver, which are 24,250 and 20,625 respectively. Below are some of the comparisons made while selecting the batteries.

Manufacturer	Part Number	Capacity (mAh)	Weight (g)	Connection Type	Output Voltage (V)	Discharge Rate* (c)	Cost (\$)
Tattu	TA-10C-25000-6S1P-HV-XT90	25000	2577	XT90-S	22.8	10/20	504.99
Aurelia	-	10000 (x2)	-	XT90-S	22.2		549.95
Lumenier	721248878748	22000	2478	EC5	22.2	10/40	399.99

LiperiAir	LPSS2200 06S10	22000	1910	XT90-S	22.2	10/25	269.99
Tattu	TA-25C-30 000-6S1P- AS150U-F	30000	3500	XT90-S	22.2	25/50	576.00 (Out of Stock)

Table 3.3.7-3: Battery Comparison

**Discharge Rate/Max Discharge Rate.*

In sum, while comparing batteries we looked primarily at the capacity, weight, discharge rate and cost/availability. We have decided to go with the Tattu 25000 mAh battery because it meets our needs and is readily available. If we find that later on in the design process this will not be sufficient we will re-address the issue of battery selection or going with a separate battery for the tracker all together.

3.4 Coding Languages

As for the software that will be used in this project, several languages have been considered. These languages are C, C++, Java, VHDL/Verilog (for FPGA-based systems), Python, and MicroPython. Some aspects of this system will be a hard real-time system. That means that making the deadline is very important, as well as making the correct calculations. To do this, one would need a deterministic and stable code. Object-oriented code tends to not be predictable, despite this, a large part of the industry still uses JAVA in a lot of microcontroller applications. Despite the use of C for important real-time tasks, the group has considered other options.

Below is a brief comparison (pros and cons) of each language:

1. C

- **Pros:**
 - High performance and low-level memory control, essential for real-time systems.
 - Extensive support for embedded systems and microcontrollers.
 - Well-established for use with real-time operating systems (RTOS).
- **Cons:**
 - More complex and error-prone due to manual memory management.
 - Less abstraction, which can increase development time.

2. Java

- **Pros:**
 - High-level language with built-in memory management and garbage collection, reducing memory leaks.

- Object-oriented programming (OOP) features for better code organization and reuse.
- Extensive libraries and frameworks.
- **Cons:**
 - Higher overhead and slower performance compared to C due to the Java Virtual Machine (JVM).
 - Less common in embedded systems and real-time applications, though possible with technologies like Real-Time Specification for Java (RTSJ).
 - Increased latency and unpredictable garbage collection times can be problematic in hard real-time systems.

3. C++

- **Pros:**
 - Combines the performance and low-level control of C with object-oriented features.
 - Supports both procedural and object-oriented programming, offering greater flexibility.
 - Extensive libraries and support for embedded systems.
- **Cons:**
 - Manual memory management, similar to C, which can be error-prone.
 - May introduce performance overhead compared to pure C.

4. Python

- **Pros:**
 - High-level language with simple syntax, making it easy to write and maintain code.
 - Extensive standard library and third-party modules.
 - Strong community support and rapid development capabilities.
- **Cons:**
 - Interpreted language with significant performance overhead, unsuitable for hard real-time requirements.
 - Limited support for low-level hardware interaction.
 - Not typically used for real-time systems due to garbage collection and interpreter overhead.

5. MicroPython

- **Pros:**
 - Python implementation optimized for microcontrollers.
 - Easy to write and understand, with rapid development capabilities.
 - Good for prototyping and educational purposes.
- **Cons:**
 - Performance limitations due to being an interpreted language.
 - Not suitable for hard real-time systems due to garbage collection.

- Limited functionality compared to full Python and other compiled languages.

6. VHDL/Verilog (for FPGA-based systems)

- **Pros:**
 - Allows designing custom hardware circuits for real-time performance.
 - Deterministic and highly predictable timing behavior.
- **Cons:**
 - Requires hardware design knowledge and is complex to learn.
 - Longer development cycles compared to software-based solutions.
 - Limited to FPGA-based systems and not applicable to general-purpose microcontrollers.

It is important to note that VHDL/Verilog has an entirely different application than our traditional functional programming language, and especially object-oriented programming. It is based on C but its job is to help design our PCB.

If this group wanted to use object-oriented language for a certain application, it would not use Python. It has very little support for low-level hardware interaction. Even compared to Java, there is way too much performance overhead that the easier coding would go to waste. It also has garbage collection just like microPython. Both of these languages are almost never used and for good reason.

The top three languages that are used in the industry for real-time systems and microcontrollers are C, C++, and JAVA. C++ and JAVA are less ideal because of the fact that they are object-oriented, however they do have support for microcontrollers. Out of these two, JAVA would be the top choice since there is indeed more support on top of Real-Time Specification for Java.

C was already the top choice for many reasons. If one could manage memory well with proper code, one of the major downsides to using C would be fixed. It also has the most support of any language by far for embedded systems and microcontrollers (this includes Real-Time Operating Systems (RTOS) Support). It has the one of highest performance and efficiency, with minimal overhead and deterministic execution. It also has low-level access to memory and hardware.

4 Standards and Constraints

Standards and constraints can range from useful tools, to inconvenient obstacles. Standard hardware and software such as USB and . Constraints, such as time and environment, are less useful and more inconvenient for the team. In this chapter we'll be discussing the standards and constraints for the project as well as how they'll affect our design and progress.

4.1 Standards

4.1.1 Laser Safety Standards

There are a number of standards we'll have to follow throughout the project, chief among them being laser safety [22]. We'll be working with two different wavelengths, 850nm and 1550nm, and while 1550nm is generally considered to be "eye safe" during normal operation even at high powers, 850nm light sources are not. Because 850nm light sources are not eye safe, it's even more important for us to follow laser safety guidelines, especially during any indoor live demos. Laser safety standards will normally include items such as laser classification, safety protocols, and control measures.

Laser classification is determined by how dangerous a laser can be towards living things, whether to the skin or retina, when under normal operation. The classes for lasers range from Class 1 to Class 4, and can be summarized as follows:

- Class 1: Completely safe under normal operation.
- Class 2: Can cause damage to the retina, however damage is minimized by the human blink response (0.25ms).
- Class 2B: Can cause damage to the retina, however damage is minimized by the human blink response, unless the beam is amplified by optical components.
- Class 3: Can cause damage if handled inappropriately. Generally safe, however the human blink response is not fast enough to prevent damage to the retina if the beam is directly incident upon the eye. Beams reflected off of non-mirror surfaces are usually safe for viewing.
- Class 3B: Hazardous to the eye. Any exposure without eye protection will cause damage, and prolonged exposure on the skin or flammable materials may cause burning.
- Class 4: Extremely hazardous. Even reflected beams will damage the eye without protection, and exposure to the skin or flammable materials will cause burning. Class 4 lasers should only be used by trained operators.

With the output power we will require in order for the beacon to be detectable 10km away (1W or more), the beacon would technically be considered a Class 4 danger. However, with the high divergence from the optical design, the actual incident power will be very low. Even still, all team members will take the required training courses on safe laser operation, and we will only test the long range functionality in a controlled environment.

For indoor demonstrations, we will have to ensure the beacon only emits enough power to be considered Class 1. 850nm light sources can be especially dangerous to the eye, since it's a non-visible wavelength but still capable of damaging the retina. However, 850nm light sources can still be considered eye safe as long as they output less than 1mW. By limiting the input current as well as applying neutral density (ND) filters, we can limit the amount of optical output power. The limited power, as well as the

divergence from the optical design, will ensure that the laser can be safely used for indoor demonstrations.

The other laser safety standards, control measures and safety protocols, are the standards that must be in place in order for the use of hazardous lasers to be considered “normal operation”. We’ll be putting an extensive amount of time into ensuring that the laser can be considered Class 1 for indoor operations, however for outdoor testing we’ll have no choice but to use the Class 4 power. For our outdoor tests, we’ll have to make sure we use safety protocols and control measures so that no team members or bystanders are harmed. Since we’ll only be using the Class 4 power for private field tests, the control measures will mostly be for our own protection. The control measures we can employ for safe laser operation are:

- Personal Protective Equipment (PPE). These include optical safety goggles rated for the wavelength and optical power we’ll be working with.
- Engineering blocks, such as protective housings, beam blockers, and shutters over the output beam.
- Safety training. All team members will take a training course on laser safety so that everyone knows what steps to take for operating a hazardous laser safely.

Outside of control measures, there are also the safety protocols we’ll follow. The safety protocols will apply to our private field tests, as well as indoor demos. Even though the indoor demos will only use Class 1 laser power, the safety protocols will still be important to ensure a safe demonstration for everyone in the room. The safety protocols we’ll follow are:

- Environmental controls. Making sure the area around the operating laser is clear of bystanders, or that all bystanders are aware of the laser being operated.
- Written policies. All safety requirements will be written on paper so that everyone within the vicinity has a clear understanding of the laser being operated.
- Visible alerts. Items such as bright signs or blinking lights so that anyone in the vicinity that is not aware a laser is being operated, is made aware.

4.1.2 Software and Communication Standards

The following is a list of software and communication protocol standards that will be used, experimented with, or could be used in the project:

Standard: FCC Part 15 (Radio Frequency Devices)

Impact on Design:

- Ensures the telemetry modules (RFD900x) comply with RF emissions regulations.
- Requires testing for interference and certification, impacting design timelines and costs.

Standard: GNSS (Global Navigation Satellite System)

- **Standard:** IS-GPS-200, IS-GPS-705, IS-GPS-800
- **Description:** These standards define the interfaces and performance for GPS signals, ensuring interoperability and reliability of GNSS devices.
- **Impact:** Ensures the Here+ GPS module accurately receives and processes satellite signals for precise positioning and timing, critical for both transmitter and receiver alignment.

Standard: GPS (Global Positioning System)

- **Standard:** ICD-GPS-870 (Interface Control Document for GPS)
- **Description:** Defines the protocols and data formats used in GPS receivers.
- **Impact:** Ensures the Cube Orange+ and Here+ GPS modules can effectively communicate and process location data, crucial for maintaining accurate positioning of the gimbal and laser alignment.

Arduino and C Code

- **Standard:** ISO/IEC 9899:2018 (C18)
- **Description:** Specifies the requirements for the C programming language, including syntax, libraries, and compiler behavior.
- **Impact:** Ensures code portability and reliability across different microcontroller platforms, including the Cube Orange+ and ESP32.

Bluetooth

- **Standard:** IEEE 802.15.1
- **Description:** Specifies the protocol and architecture for short-range wireless communication using Bluetooth technology.
- **Impact:** Ensures any Bluetooth-enabled devices used in the project (e.g., for diagnostics or additional telemetry) are compatible and can communicate effectively.

Wi-Fi

Standard: IEEE 802.11

Description: Specifies the protocol and architecture for wireless internet. Specifies the frequency bands used (e.g., 2.4 GHz, 5 GHz). Defines the supported data rates for different versions (e.g., 802.11a/b/g/n/ac/ax). Includes security protocols such as WPA, WPA2, and WPA3 to ensure secure communication. Ensures devices from different manufacturers can communicate effectively.

Impact: It has allowed many devices on a local area connect seamlessly, regardless of location (as long as its in range) and could work universally without any plug in. It has vastly assisted in the improvement of modern communication.

Electromagnetic Compatibility - EMC

Standard: IEC 61000

Description: The IEC 61000 series of standards focus on electromagnetic compatibility (EMC) to ensure that electronic devices function properly without causing or suffering from electromagnetic interference. It limits the amount of electromagnetic interference that a device can emit. It ensures devices can operate correctly in the presence of certain levels of electromagnetic interference. It specifies the testing procedures for evaluating the EMC performance of devices. Devices must also be tested and certified to meet these standards to prevent and mitigate interference issues.

Impact: These standards are important for nearly all working devices on the planet and their ability to work around each other. Without this standard, many devices would likely interfere with each other and many signals would get interrupted or also interfered with.

CAN (Controller Area Network)

- **Standard:** ISO 11898
- **Description:** Defines the standards for CAN bus communication, including physical layer, data link layer, and protocol.
- **Impact:** Ensures reliable communication between different microcontrollers and sensors, particularly if CAN bus is used for data transfer between the Cube Orange+ and other components.

USB (Universal Serial Bus)

- **Standard:** USB 2.0 (High-Speed), USB 3.0 (SuperSpeed)
- **Description:** Defines the specifications for USB interfaces, including data transfer rates, power supply, and physical connections.
- **Impact:** Ensures reliable data transfer and power supply for USB-connected devices, such as the ESP32 or other peripheral devices used in the project.

IEEE 1588 (Precision Time Protocol - PTP)

Description: The IEEE 1588 standard, known as the Precision Time Protocol (PTP), is used for precise synchronization of clocks in computer networks. This is particularly useful for applications that require high precision time synchronization.

Key Points:

- **Clock Synchronization:** Provides mechanisms to synchronize clocks in a network with sub-microsecond accuracy.
- **Master-Slave Hierarchy:** Describes a master-slave hierarchy for distributing precise time across network nodes.
- **Profiles:** Defines different profiles for various applications, such as telecommunications and power systems, to tailor the synchronization mechanism to specific needs.

Unlike the other standards, this one is not optional. The very basis of the usable microcontrollers in this project will use the precision time protocol. Any other part that has logic circuitry also uses PTP. Its ability to provide sub-microsecond accuracy ensures that systems requiring precise timing can operate efficiently and reliably. Whether in telecommunications, power grids, financial services, industrial automation, or other sectors, PTP plays a crucial role in enhancing performance, ensuring compliance, and enabling the synchronization of complex, distributed systems.

4.1.3 Radio Communication standards

The FCC (Federal Communications Commission) set the law for anything related to communications in the United States. One of the most critical standards for this project involves radio communication, which falls under the purview of the Federal Communications Commission (FCC) in the United States. The FCC regulates the use of the electromagnetic spectrum to avoid interference between different communication systems and to ensure that all radio communications are conducted safely and efficiently.

On the FCC website, they have a detailed criteria of their regulations:

- Frequency Allocation
- Power Limits
- Emission Standards
- Modulation Techniques
- Certification and Labeling

Frequency allocation:

- The FCC allocates specific frequency bands for different types of radio communication. For this project, the telemetry module RFD 900x-US operates within the 900 MHz band. This band is typically used for industrial, scientific, and medical (ISM) applications, which are subject to specific regulations to minimize interference.
- Part 15 of the FCC rules outlines the operational guidelines for unlicensed devices, including the ISM band. Devices operating in this band must adhere to power limits and emission requirements to prevent harmful interference with other devices.

Power Limits:

- The FCC sets limits on the output power of radio devices to control the range and reduce the likelihood of interference. For the 900 MHz ISM band, the maximum allowable effective radiated power (ERP) is typically 1 watt (30 dBm) for point-to-point communication.
- The RFD 900x-US telemetry module must be configured to ensure that it does not exceed this power limit. Additionally, antenna gain must be taken into account, as high-gain antennas can effectively increase the ERP.

Emissions Standards:

- The FCC requires that devices operating in the 900 MHz band comply with specific emission standards to limit out-of-band emissions and minimize interference. These requirements are detailed in Part 15, Section 15.247, which stipulates the maximum permissible levels of spurious emissions.
- Testing and certification are necessary to demonstrate compliance with these emission standards. Manufacturers must submit their devices for testing to ensure they meet the FCC's requirements, and this process results in the issuance of an FCC ID, signifying compliance.

Modulation Techniques:

- The FCC also regulates the modulation techniques used in radio communication. The RFD 900x-US uses frequency hopping spread spectrum (FHSS) and direct sequence spread spectrum (DSSS) techniques, which are permitted under Part 15.247.
- These techniques help minimize interference and maximize the efficient use of the spectrum. The telemetry module must be configured to use these techniques within the guidelines specified by the FCC.

Certification and Labeling:

- Devices that comply with FCC regulations receive an FCC certification, indicated by an FCC ID on the device. This certification process involves rigorous testing to ensure that the device adheres to all relevant standards.
- Proper labeling is essential for compliance. Devices must display their FCC ID and any other relevant information, such as warning labels for users. This labeling ensures that users and regulators can quickly identify compliant devices.

4.1.4 PCB Design Standards

While many of the FCC standards relate to RF and intentional radiators there are constraints on unintentional radiators, like noise from a power supply. EMC testing is conducted by the FCC to combat unintentional radiation, this test is to measure the strength of the EMF field that is emitted from the device under varying test conditions. According to Intertek nearly 50% of all first time electromagnetic compatibility (EMC) tests result in failure [69]. The following restraints on unintentional radiators are provided in the table below.

Equipment Class	Frequency of Emission (MHz)	Radiated Limit (dBμV/m)
A (10m distance)	30-88	39
	88-216	43.5
	216-960	46.4

	>960	49.5
B (3m distance)	30-88	40
	88-216	43.5
	216-960	46
	>960	49.5

Table 4.1.4: Limits of unintentional radiators.

4.1.5 Optical Design Standards

When designing an optical system, there are two sets of standards we'll want to follow. First is the ISO 10110 Series, which is a series of standards for drawing optical diagrams and schematics. Optical diagrams sketched under the ISO10110 typically include information like general terminology surface form tolerances, centering tolerances, and surface imperfections

The other standard we should follow when designing our optical system is the ISO 14999 Series. The ISO 14999 series includes standards for evaluating and testing optical components and equipment. The series is useful when searching for things like surface imperfections and faulty sensors [37].

4.2 Constraints

The Optical Beacon Tracker for Optical Wireless Communication project presents a range of constraints that must be carefully considered to ensure the system's successful implementation and operation. These constraints include budgetary limitations, weight restrictions, environmental factors, time, and technical challenges. Understanding and addressing these constraints is crucial for the project's feasibility and performance.

4.2.1 Budgetary Constraints

One of the primary constraints for this project is the budget. The allocation of financial resources impacts the quality and capabilities of the components used. Some key budgetary considerations include:

- **Component Costs:** High-quality components such as laser diodes, gimbals, GPS modules, and certain microcontrollers can be expensive. Ensuring the project remains within a couple of hundred dollars for critical parts like the gimbal requires careful selection and potential compromises on performance.
- **Development and Testing:** Budget must be allocated for prototyping, testing, and potential iterations of the design. This includes costs for additional equipment, testing environments, and possibly external services for advanced testing and calibration.

- **Maintenance:** Long-term operation requires more robust parts. Budgeting for these future expenses is necessary to sustain the project's viability over time.

4.2.2 Mobility Constraints

The weight of the system is a critical constraint, especially if the receiver is mounted on a mobile platform such as a drone or a vehicle. Factors influencing weight include:

- **Component Selection:** Lightweight components are preferred to minimize the overall weight. For instance, using compact and lightweight microcontrollers, laser diodes, and gimbals is essential.
- **Structural Materials:** The choice of materials for mounting and housing the components affects the weight. Materials like aluminum or carbon fiber can provide strength while being lightweight.
- **Battery and Power Supply:** Batteries and power supply units must be sufficient to power the system while not adding excessive weight. Balancing energy capacity and weight is a key challenge.

4.2.3 Environmental Constraints

Environmental factors play a significant role in the design and operation of the optical beacon tracker. These factors include:

- **Weather Conditions:** The system should be able to operate reliably under various weather conditions, including rain, fog, and high temperatures (The Florida sun). Some weatherproofing components may need to be added to ensure stable operation in adverse conditions are crucial.
- **Temperature Variations:** Extreme temperatures can affect the performance of electronic components. Implementing thermal management solutions such as heat sinks may be necessary.
- **Lighting Conditions:** The presence of ambient light (sunlight) and other sources of optical noise can interfere with the laser signal. Ensuring robust signal detection and filtering mechanisms is vital for reliable operation.

4.2.4 Location Constraints

The geographical location where the system will be deployed affects several aspects of its design and functionality:

- **GPS Accuracy:** The accuracy of the GPS modules is influenced by the geographical location. Urban environments with tall buildings may cause multipath errors, while open areas provide better GPS performance.
- **Line of Sight:** The optical communication system relies on a clear line of sight between the transmitter and receiver. Obstacles such as buildings, trees, or terrain can obstruct the signal path.
- **Mobility:** If the receiver is mobile, such as on a drone or vehicle, the system must be designed to handle dynamic changes in location and orientation.

4.2.5 Technical Constraints

Technical constraints encompass the limitations and challenges associated with the components design, system's design, and operation:

- **Precision and Accuracy:** Achieving high precision in pointing the QPD towards the laser signal is critical. The gimbal and control system must provide fine adjustments to maintain alignment. The components such as the photodiodes are also only so accurate. Not to mention the pinhole that has been machined at the center of the QPD may be crooked, affecting the systems ability to detect the laser signal.
- **Signal Processing:** The microcontroller must efficiently process the analog signals from the QPD, demodulate them, and convert them to digital signals. This requires adequate processing power and efficient algorithms. By extension, a weak cpu will not be able to do this in a fast enough manner.
- **Power Consumption:** The system's power consumption must be managed to ensure long operational periods, especially if battery-powered. Energy-efficient components and power management strategies are essential. This also ties in with weight because although a bigger battery can be used to compensate for a more power hungry system, that creates a heavier overall system and the max weight is 5kg for the transmitter and receiver each.
- **Latency:** The communication and control loop must have low latency to ensure real-time adjustments. Delays in processing or communication can lead to misalignment and signal loss. Time and money were also spent in getting a telemetry module that could meet the latency requirements that this project would need.

4.2.6 Integration Constraints

Integrating various components into a cohesive system presents its own set of challenges:

- **Compatibility:** Ensuring that all components, such as the QPD, ESP32 microcontroller, GPS modules, and gimbal, are compatible and can communicate effectively.
- **Synchronization:** The timing and synchronization between the transmitter's pulse width modulation and the receiver's detection and processing must be precise.
- **Software Development:** Developing robust software for control, signal processing, and communication is a significant task. This includes handling real-time data, implementing control algorithms, and ensuring reliable operation.

4.2.7 Regulatory Constraints

Compliance with regulatory requirements is essential for legal operation:

- **Frequency Regulations:** The use of laser diodes and wireless communication modules must comply with frequency regulations and safety standards to avoid interference and ensure safe operation.

- **Safety Standards:** The system must adhere to safety standards to prevent harm to humans and animals. This includes measures to prevent accidental exposure to laser radiation.

Both of these have been explained more in detail previously under laser safety standards and frequency Regulations.

4.2.8 Time Constraints

As a senior design project at the University of Central Florida (UCF), the timeline is a critical constraint. The project must be completed within the summer and fall terms of 2024. Key time-related considerations include:

- **Project Phases:** The project must be broken down into manageable phases, including research, design, prototyping, testing, and final implementation. Each phase must be carefully planned to ensure timely completion.
- **Milestones and Deadlines:** Setting and adhering to milestones and deadlines is crucial for maintaining progress. Regular progress reviews and adjustments to the project plan may be necessary to stay on track.
- **Resource Allocation:** Efficient allocation of time and resources, including team members' skills and availability, is essential. Balancing the workload and ensuring each team member contributes effectively can help meet deadlines.
- **Contingency Planning:** Allowing time for unexpected challenges and setbacks is important. Building in buffer periods for troubleshooting and problem-solving can help mitigate delays.

4.2.9 Component Safety Constraints

In addition to laser safety there are also precautions that should be taken to ensure personal and component safety alike. These include proper handling of the power-supply/battery, ensuring that components are safe to connect or disconnect before attempting to do so, testing all components in a lab or controlled environment and so on.

- **Electrical Safety:** It is important that all components receive the correct input voltage and are safely supplied from the 22.8 V battery. To ensure this all regulators should be tested in the lab before hooking them to the battery supply while using resistors to simulate the operational load.
- **Thermal Safety:** Due to the large amount of diodes and the need for linear regulation to drive them, temperature monitoring is essential for the beacon. If the thermals were to reach a critical level they could cause catastrophic failures and c
- **Mechanical Safety:** Range of motion for the gimbal and servo motors should not conflict with the position of other components. More importantly, all components should be securely mounted in place and capable of withstanding the forces of the drones flight

- **Operational Safety:** It is important to ensure proper and secure connections when powering up and down the beacon and tracker/receiver. Furthermore, initial testing should be done under the lowest power settings possible while slowly increasing the power input to desired specification, concerning the diodes and motors.

5 Comparison of LLM Platforms

In this section we hope to provide insight and comparison of OpenAI's ChatGPT and other platforms such as BARD or Gemini developed by google. To begin, both services can be useful and offer a similar product but one of the first things that will stand out to any user is that Gemini offers a bit more accuracy or rather fact checking. To elaborate, when you get a response from Gemini it will highlight certain statements and flag them for credibility whether it can or cannot find relevant content via a simple google search. This helps handle the accuracy issues and concerns we are all aware of while using a LLM. However, the use case for either platform has revolved around brainstorming at this point in our design. The reason being is that even with credibility checking there is no assurance that the suggested solutions generated by either ChatGPT or Gemini are even possible. Furthermore, compatibility in part selections can be questionable at best. To exemplify these concerns and others there will be four different prompts and responses compared below. The first is going to cover a generic question applicable to the concept of the project, the next topic/question will be more specific and in depth and so on. The responses generated by ChatGPT and Gemini will be marked with [GPT] and [GEM] respectively and be recorded in the appendix at the end of this paper.

Comparison 1: Both of the responses resulted in beneficial information on how to approach the problem without overextending into part selection and direct implementation. The inclusion of this conversation was to illuminate the value of LLM in brainstorming and conceptual approach in the design process. While the results between the two are similar there are some key differences to note.

The first notable difference being a pro for Gemini is that it recommended a better optical design, calling for collimator lenses to combat divergence as well as mentioning a photodiode able to receive the NIR spectrum. Another favorable component in Gemini's response was the overall brevity itself. While the prompt was admittedly vague, Gemini seemed to give a conceptual prompt a short conceptual answer. Comparatively ChatGPT seemed a bit on the redundant side using twice as many words with little additional substance to show for it. For example in the "Set Up the Tracker" section if the GPT response has two bullet points telling the user to first mount the photodiode to the gimbal and then the gimbal to the drone, simply a blatant waste of space.

However, there are some notable drawbacks for Gemini too, the failure to recommend an ADC, even at the conceptual level, can be considered a significant oversight considering the clear use of a quadrant photodiode in optical tracking. Another con found when comparing the gemini response to ChatGPT's is the resources for further information suggested simply google searching for various retailers to find part specifications.

Although an important step in the design process is part selection and comparison, a more conceptual resource would be arguably more useful and important at the brainstorming stage of a project.

The next notable difference is a pro for ChatGPT in its recommendation for a communication fail safe using redundancy in critical components, similar to the spin protocol we plan to implement. This in addition to the recommendation of RF technology, which again is similar to our design in implementing a telemetry module, described in our hardware section.

In sum, both of the platforms identified a course of action that could be useful in starting the design process. While ChatGPT generated more redundant and superfluous information Gemini and also attempted to give overly specific step-by-step instructions it did make up for it in recommending a communication fail safe, an RF based telemetry module, and some useful ideas for possible implementation. Contrastingly, Gemini provided a more concise response while addressing the need for collimator lenses and the need for a photodiode that can handle the appropriate wavelengths. It also wasn't short of flaws, considering the absence of an ADC to control the gimbal with analog signals received from the quadrant photodiode. While both LLM platforms shared strengths and weaknesses I would say that the more beneficial response to the prompt came from ChatGPT. This is due not only to the pros listed above but because one of its major flaws was over description. Additionally, during this description there was a separation between beacon and receiver components and function which made it easier to decipher which end was being referenced during the response.

Comparison 2: In this second prompt we are a bit more specific in a desired task and can begin to see a larger divergence between the two responses. Unlike the last prompt we are now asking how to implement specific types of technologies to transmit location and orientation data of the beacon. Because this is an implementation prompt certain aspects that were viewed as a con in the previous section actually work as a pro in this one. For example, in this response ChatGPT again went into much greater detail than Gemini, where in the last example this was a major con here it is more applicable and can be seen as a possible pro.

While the lists of various parts and model numbers for GPS modules, IMUs, MCUs, quadrant photodiodes and more are provided, this shouldn't be viewed as an exact shopping list because there aren't any guarantees that the combination of hardware recommended will work. The information is useful in guiding research in potential parts and components along with their capabilities. While building the hardware diagram ChatGPT proved useful in explaining various components and technologies that could be used and how they could interconnect.

Furthermore, another pro outside of part suggestions is the provided pseudo code for both the beacon and the receiver to communicate position between one another. Despite only being pseudo code it is a visual aid and linear reference to how implementation of the recommended components could be possible. For example, the beacon code is simple, a

loop transmitting the GPS and IMU data with a delay to ensure a proper update rate. The pseudo provided for the tracker is a bit more complicated, it essentially loops a continuous read of the GPS and IMU data from its own modules while also receiving the data from the beacon modules. This loop also includes instructions to calculate the necessary gimbal angles and to set them. The only way that this loop gets broken is if the quadrant photodiode detects the laser diodes of the beacon. Obviously this code is bare bones and doesn't account for the adjusting of the laser diode array but it provides valuable insight into the programming and implementation of the diodes and gimbal.

Now considering Gemini, some of the pros in its response are that it mentions the possibility of atmospheric interference and the need for sensor accuracy in pre-aligning the gimbal. To elaborate on the need for accuracy Gemini states that factors like satellite visibility along with atmospheric conditions can greatly hinder performance. Additionally, Gemini states that proper calibration of components like the IMU is needed to ensure proper operation. These considerations are important in moving forward with the design process and can be viewed as valuable pointers.

However, despite the valuable insight in implementation, Gemini still gave a short and conceptual answer despite the prompt calling for specific technologies and devices. The previous prompt did suggest part research via retailers websites but made no mention of it here, which is odd to say the least. Furthermore, Gemini also included a recommendation for a telemetry module but left the type of technology open ended. A preferable response would have done a brief comparison between radio, cellular and satellite technologies mentioned in its recommendation for telemetry models.

To conclude, the better response once again came from ChatGPT. This conclusion can be made from the additional information provided about parts along with the pseudo code. It is important to note that the response from ChatGPT was once again flawed in its redundancy and excessive length, while more information tends to be beneficial in this instance the listing of implementation steps could have been halved while still relaying the same amount of information.

Comparison 3: This third and final case exemplifies a complete contrast in responses due to comprehension. The third prompt inquired whether it was best to route power through the MCU to route the power directly from the battery and use the MCU to switch the power supply to the batteries. This prompt was chosen to show the suggestions made by each LLM given a question asking for insight of approach.

However, there appeared to be an issue in comprehension, ChatGPT understood that the drivers would be modulated through the MCU regardless and that I was asking if the drivers should be powered directly from the power supply or have it routed through the MCUs PCB for direct digital modulation. On the other hand Gemini perceived that I wanted to directly supply the laser diodes from the power supply and modulate that directly. However, despite the misinterpretation Gemini recommended a solution similar to that of ChatGPTs in that it called for routing the power through the MCU circuit board with a separate regulated power supply. Unfortunately due to the misinterpretation it also

generated a warning about the inability to modulate if powered directly from the battery, which was never considered to be an option.

Another notable difference proving to be a pro for ChatGPT and a con for Gemini is the mention of current control and power regulation. These are key components of powering our laser diodes and Gemini made mention of neither. Overall, it seems that no matter the subject, whether it be vague or specific, ChatGPT generates more useful responses than Gemini and was better for the overall brainstorming for design. While Gemini provided fact-checking and warnings of credibility this project or rather the prompts provided seemed to be too niche for it to flag any of the information it provided. Furthermore, GPT was able to make good on its oversized responses by including crucial information that was all but overlooked by its counterpart. In sum, for the first three prompts on design and implementation ChatGPT was the clear winner.

Comparison 4: Once again ChatGPT provides a much lengthier response with little other than pseudo code to show for it. However, the point of this prompt was to include specific components that should be recognizable to both platforms. Stating that we plan to implement the Orange+, Here4 and a telemetry module, all flagship products, it should exemplify each platform's ability to provide specific and topical solutions given components with highly publicized specifications.

To begin our comparison it is important to once again point out the similarities in the two responses to begin extracting useful information. The first similarity at the top of each response was in regards to software, such as Mission Planner and QGroundControl were suggested by both platforms. Besides being listed at the top of both responses it is brought to attention here to show the similarities in our choice in using Mission planner and the intuitiveness of both platforms to recognize that both Mission Planner and QGroundControl are recommended by the manufacturers. The second similarity worth mentioning is the implementation of using the GCS to send positional between the beacon and receiver. This is of note because although it was not our idea of original implementation we had quickly realized that it is the best course of action for our hardware selection up to this point. However, these are the only two similarities that held useful information, far less than our last three prompts.

Next, we will begin to look at the pros and cons of the Gemini response and how it was either helpful or hindering. One of the hindering differences was that Gemini had mentioned an alternative plan of action, one that did not require a GCS. This was particularly interesting because our original design for implementation had the two drones communicating seamlessly with one another. However, we found that this approach was much harder to implement using the orange+ and here4, which are designed to be used with either a GCS or HereLink handheld controller. In the response it did mention that this method would be more difficult but it seems as though the severe difference in difficulty was not properly expressed. A helpful difference between the two responses was that Gemini was able to provide a URL to a developer guide, this guide was able to then provide various resources through hyperlinks in each of its descriptions. Compared to ChatGPT omitting URLs of any kind. While the resources provided may

have been satisfactory, overall Gemini was not able to answer the prompt with the expected amount of detail, proving again that it is an ideal platform for brainstorming rather than direct implementation.

Moving on, we will now begin to address the pros and cons of the ChatGPT response. At first glance it is astonishing the difference in length. ChatGPT was able to generate a three page response in comparison to the half page that Gemini had responded with. Furthermore, a more concise and informed answer was given on the capabilities of Mission Planner and QGroundControl to handle multiple drones, something that we had come to realize through research just days prior to using this as a prompt example. One of the aspects that seemed glossed over was the positional relay section, simply stating that the data request and relay configuration was dependent upon the GCS.

Despite the lackluster response to a would be useful section ChatGPT seemed to be accurate enough and overall helpful. One last thing to consider is both a pro and con, the response from prompt 2 simply had generated pseudo code, which was acceptable had the language not been known or specified, in this response ChatGPT took it upon itself to generate python code. This can be considered as a con because the language used for the Cube Orange+ and the ESP32 is C++ and this could cause confusion. However, the pro to this is, like the pseudo code from prompt 2, it can provide an idea of how to implement the script to relay telemetry data between the two drones via a GCS.

Conclusion: Overall it is apparent that ChatGPT gives lengthier and more descriptive responses, on the other hand this also leads to an over extension of its actual knowledge. Meaning that although many of the responses from ChatGPT were usually better it did contain more misinformation/misguidance at times. Despite the Gemini platform being designed more for enterprise solutions and enhanced context understanding, ChatGPT was the better solution for questions on implementation techniques and part information. However Gemini wasn't without its pros, in addition to misinformation flags in its response it seemed to be more useful for brainstorming and approaching a problem rather than directly solving it.

Truly, at times, less really is more and Gemini had proved that in the first prompt response and was again viewed as a pro in the second. In fact, it wasn't until prompts three and four was the brevity of the response considered a con. To conclude both platforms had their pros and cons, but in the end it is ChatGPT that wins the day. This was concluded based upon the responses provided here and other prompts that had been given to both platforms during our design process.

6 Design

6.1 Optical

In this section, we'll discuss how we designed the optical setup shown in figure 2.5.2-1. The optical design of the project was crucial to ensure we can transmit the light from the beacon at the desired divergence without any unnecessary power loss. We are then able to collect the light from the beacon at a given distance and be able to focus it onto our QPD

and optic fiber aperture. On the receiver/tracker, we wanted to keep the optical path length less than two inches, maintain a relatively large spot size for the beacon array, and spot size for the 1550nm laser diode small enough to fit through the optic fiber coupled into the annular QPD.

According to the spec sheet for the SPOT-9D-0, the diameter of the whole QPD is 10mm across. This means we wanted the final spot size for the 850nm LED array to have a diameter approximately 10mm across. Whilst the hole in the annular QPD may be 0.2mm, the entrance for the single mode optical fiber that it's coupled to is only 0.126mm. So, we'll want the spot size for the 1550nm laser diode to be less than 0.12mm across. The minimum spot size of a focused laser was determined by equation 6.1.1. [38]

$$w_o = \frac{2 \cdot \lambda \cdot f}{\pi \cdot D} \quad (\text{Equation 6.1.1})$$

Where w_o is the minimum spot size of the laser, λ is the wavelength of the incident light, f is the focal length of the lens, and D is the size of the beam incident upon the lens. Since we're using a 1 inch aperture for the receiver, we can use a condenser lens with a focal length of 16mm in order to shorten the optical path length and get the smallest possible spot size for the 1550nm laser diode.

This disparity in spot sizes may seem impractical, however this is the main reason we're using an array for the 850nm beacon. Since we're effectively imaging the lights coming from the beacon onto the annular QPD plane, if the array is several times larger than the output of the 1550nm laser diode, then the focused spot size will also be several times larger. This is characterized by equations 6.1.2 and 6.1.3, and [40]:

$$r = 1.22 \frac{\lambda f}{D} \quad (\text{Equation 6.1.2})$$

$$w_{o, array} = \sqrt{(r)^2 + (N \cdot l)^2} \quad (\text{Equation 6.1.3})$$

Where r is the radius of a single focused LED, N is the number of LEDs in a single row or column, l is the distance between the centers of each LED, and r_{total} is the total spot size of the array. Equation 6.1.2 is very similar to 6.1.1, but not quite the same. We already know the focal length of our lens and the diameter of the lens aperture, but this time the wavelength will be 850nm instead of 1550nm. Our chosen LED, the ams OSRAM SFH 4258S, has a casing that is 3.7mm wide [39], so the centers of each LED will be at least 3.7mm apart from each other.

With our known and desired values we can calculate the spot size of a single LED and the spot size of an LED array, with each of those values being 0.21um and 18.5mm respectively. The spot size of the LED array may seem far too large, however since the incident beam should have a gaussian profile it will still be effective for position sensing using the annular QPD.

On the beacon end, we'll have to collimate the array of LEDs long enough to reflect off of the standard and dichroic mirrors, and then pass through the output lens of the beacon. While this would be relatively simple for a single point source, it's much more difficult for an array due to the conservation of etendue. Dichroic mirrors are limited to a one inch diameter, so we'll have to use a lens with a large enough diameter to fully encompass the whole array and focus it quickly. When collimating a single LED, we would use equation 6.1.4, but since we're working with an array we'll have to also use equations 6.1.5 and 6.1.6 [42]:

$$\theta_{col} = \frac{d}{f} \quad (\text{Equation 6.1.4})$$

$$\theta_{col,array} = \frac{d_{array}}{f} \quad (\text{Equation 6.1.5})$$

$$d_{array} = \sqrt{(N \cdot l)^2 + (N \cdot l)^2} \quad (\text{Equation 6.1.6})$$

Where d is the size of the LED emission area, θ_{col} is the divergence angle of the collimated array, $\theta_{col,array}$ is the divergence angle of the collimated array, and d_{array} is the size of the array emission area. We want the lowest divergence angle possible, so we start by calculating an array emission area of 26.16mm. Commercial lenses only come in diameters of 25.4mm or 30mm, so we know we'll use a 30mm lens to ensure we capture the whole area. Due to the nature of collimated arrays, it's impossible to actually collimate the whole thing. However, by solving equation 6.1.5 for $\theta_{col,array} = 45$ degrees, we can get the longest and smallest collimated beam size until the focal length ends.

Solving the equation gives a focal length of 33.3mm, but 30mm is what's commercially available. We'll be limited in collimated distance due to the small focal length so the will be coupled directly to the LED output with the standard mirror placed directly in alignment with the center of the beam and directed to the dichroic mirror which then directs the beam to the output lens. By the time the beam reaches the output lens it will be diverging so the output lens will have to control the spread.

We also had to expand the output beam from the fiber coupled laser. If the beam size and divergence of the 1550nm laser diode don't match that of the LED array by the time they reach the beacon output lens, then the total divergence of the beacon will be unequal. To start, we found the output waist and divergence from the fiber output, which can be done with equations 6.1.5 and 6.1.6 [41]:

$$w_o = \frac{MFD}{2} \quad (\text{Equation 6.1.5})$$

$$\theta_{fdiv} = \frac{\lambda}{\pi \cdot w_o} \quad (\text{Equation 6.1.6})$$

Where w_o and θ_{fdiv} are the beam waist and divergence of the fiber output, respectively. MFD is the mode field diameter of the fiber, which is determined by the properties of the optic fiber and the wavelength of light being used. Since the optic fiber SMF-28 has an

MFD of 10.5um at 1550nm, we can calculate the output divergence angle of the 1550nm fiber laser to be 0.094 radians, or 5.4 degrees.

We wanted the beam of the fiber to be the same size as the array after a few millimeters of propagation, so we'll use a bi-concave lens with a short negative focal length to diverge the beam even faster. The beam is small, so a 6mm diameter lens will serve well and be cheaper in material cost. The relation between the focal length and output divergence of a bi-concave lens and how it affects the divergence of an incident beam is given by equations 6.1.7 and 6.1.8:

$$\theta_{Ldiv} = \frac{1}{|f|} \quad (\text{Equation 6.1.7})$$

$$\theta_{out} = \theta_{fdiv} + \theta_{Ldiv} \quad (\text{Equation 6.1.8})$$

Where f is the absolute value of the lens focal length, θ_{Ldiv} is the output divergence of the lens in radians, and θ_{total} is the output divergence after the beam passes through it. We want the beam to be roughly 25mm in diameter by the time it reaches the output lens, and there would be no more than 18mm of propagation distance since the expanding lens has to be in-between the fiber output and the dichroic mirror. Solving the equations gives us an output divergence of 15 degrees and a focal length of -18mm for the fiber diverging lens.

For the beacon output lens, we want to limit the divergence of the LED array whilst increasing the divergence of the fiber output. The easiest way to control the output divergence of an array is to refocus it so that the desired divergence is reached after passing the focal length. For this we used a 30mm diameter plano-convex lens with a focal length of 30mm. The combination of all of these lenses gives us our desired output divergence with minimal loss.

6.2 Electrical

In this section we will introduce and describe our designs for our PSU, MCU, LED array and Receiver PCBs. Each subsection will provide a verbal description along with the schematic and PCB layout for each component.

6.2.1 MCU Design

While designing the MCU it was apparent that a small footprint and versatility were key in order for the same design to be used for both the beacon and the receiver. This ultimately led us to developing a layout that uses male pin headers for all GPIO pins to allow for maximum flexibility in implementation. Furthermore, we were able to reduce the size of the board down to 62.8mm x 25.5mm, small enough to be easily placed on either of our devices. In addition to small form factor and maximum pin outs we also implemented a temperature and moisture sensor that uses an RGB LED to indicate if the board is in danger of failure. Lastly, an LDO has been placed on the board to manage the

Temperature and Moisture Sensor

The diagram shows a 5V USB input connected to a 10uF capacitor (C6) and the IN pin of a TLV75801PDRV LDO (U3). The EN pin is connected to GND. The FB pin is connected to the OUT pin. The OUT pin is connected to a 3V3 output, which is also connected to a 10uF capacitor (C7) and a 5.1k resistor (R8). The 3V3 output is connected to the LED (D2) through R8. The LED is connected to GND. The circuit is labeled with component values and pin numbers.

58

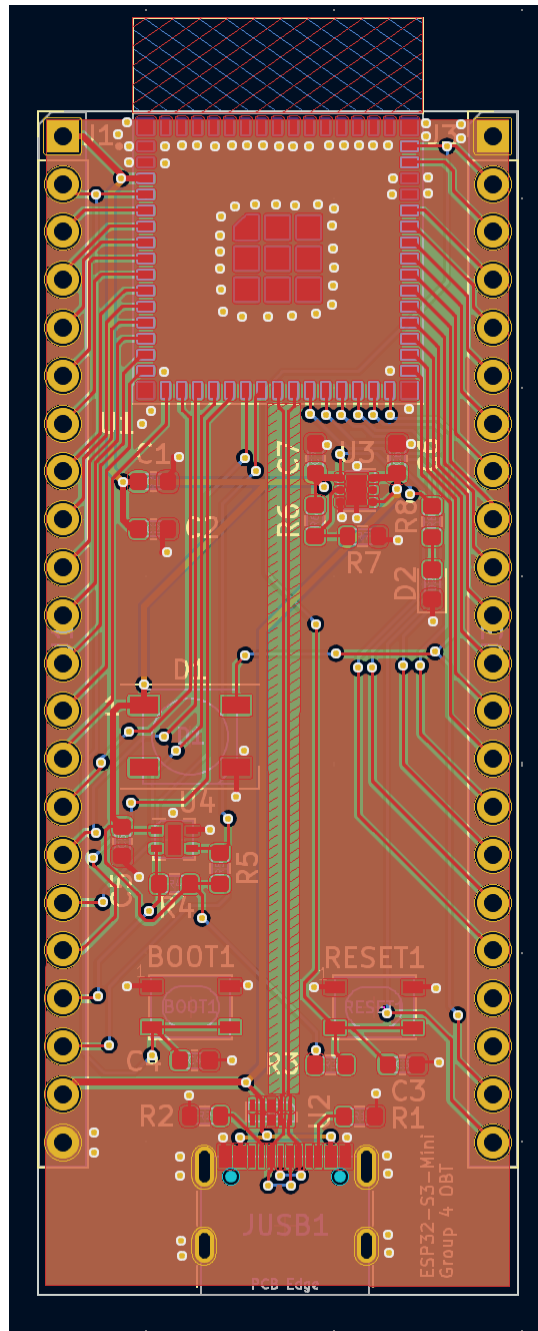


Fig 6.2.1-4: MCU PCB Layout

6.2.2 Receiver PCBs

In order to receive the signals from the QPD properly they must be amplified but also kept within the range of 0-3.3V. To accomplish this we will be using two stages of op-amps in cascade with a 3.3V supply. The first stage will be made of transimpedance op-amps and output a negative voltage, while the other stage will

be used for additional gain and inverting the signal back to a positive voltage. Originally the second stage was designed to be summing and differential amplifiers, to cut down on signal computation done by the MCU, but after thorough testing we found that there was a discrepancy in the sums and differences received across the circuit. This issue was resolved by simply using inverting amplifiers after the transimpedance stage to invert the signal and apply further gain. This solution was implemented by making a second PCB that used two more ADA4625 op-amps that received the 4 signals from the primary receiver board and passed the signal to the MCU. Whilst we would have rather redesigned the original board and omit the need for a second we found that unsoldering the QPD could prove problematic, possibly damaging the sensor and rendering it useless. Below are the schematics of the two boards and the circuit that they realize together.

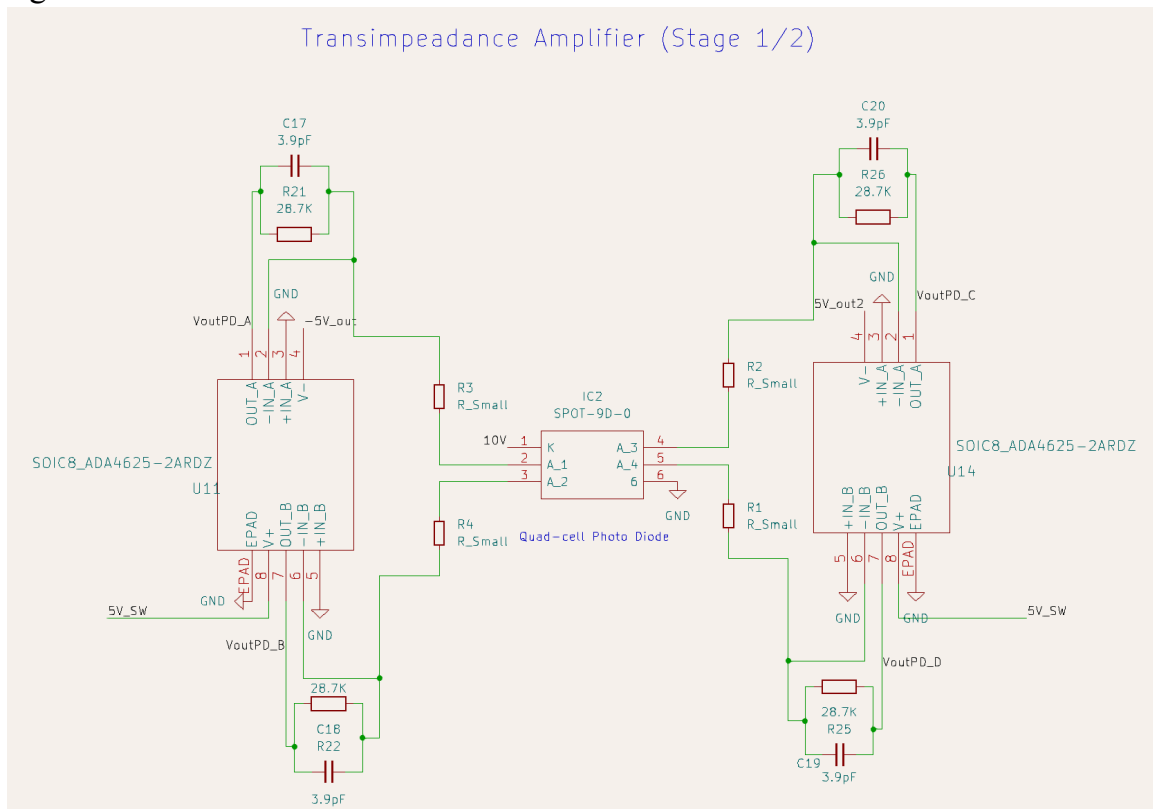


Fig 6.2.2-1: Receiver Schematic I

Inverting and 1.5x Gain (Stage 2/2)

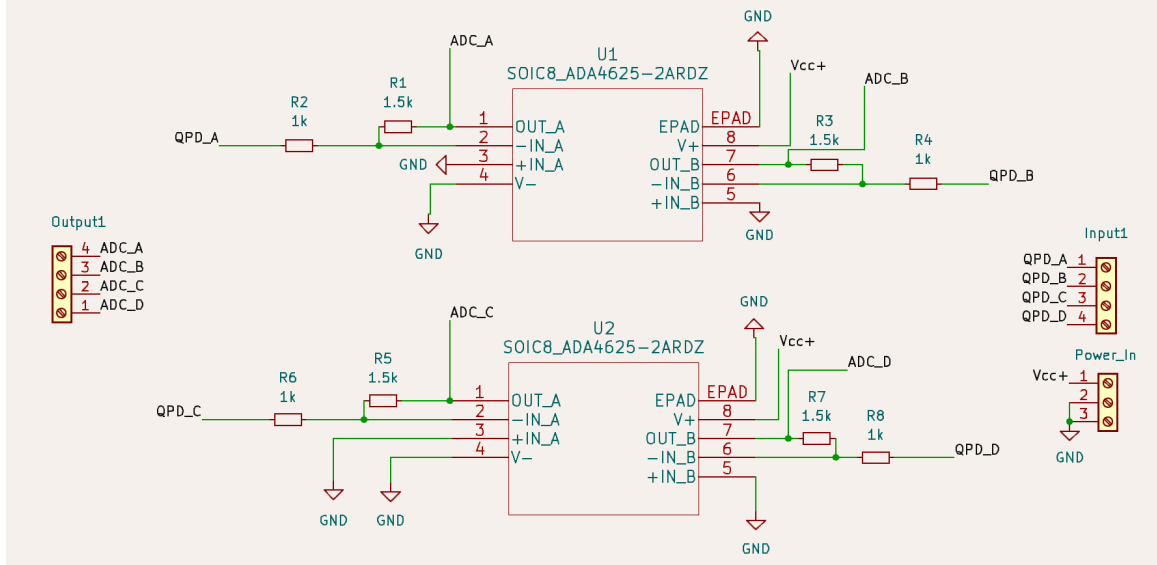


Fig 6.2.2-2: Receiver Schematic II

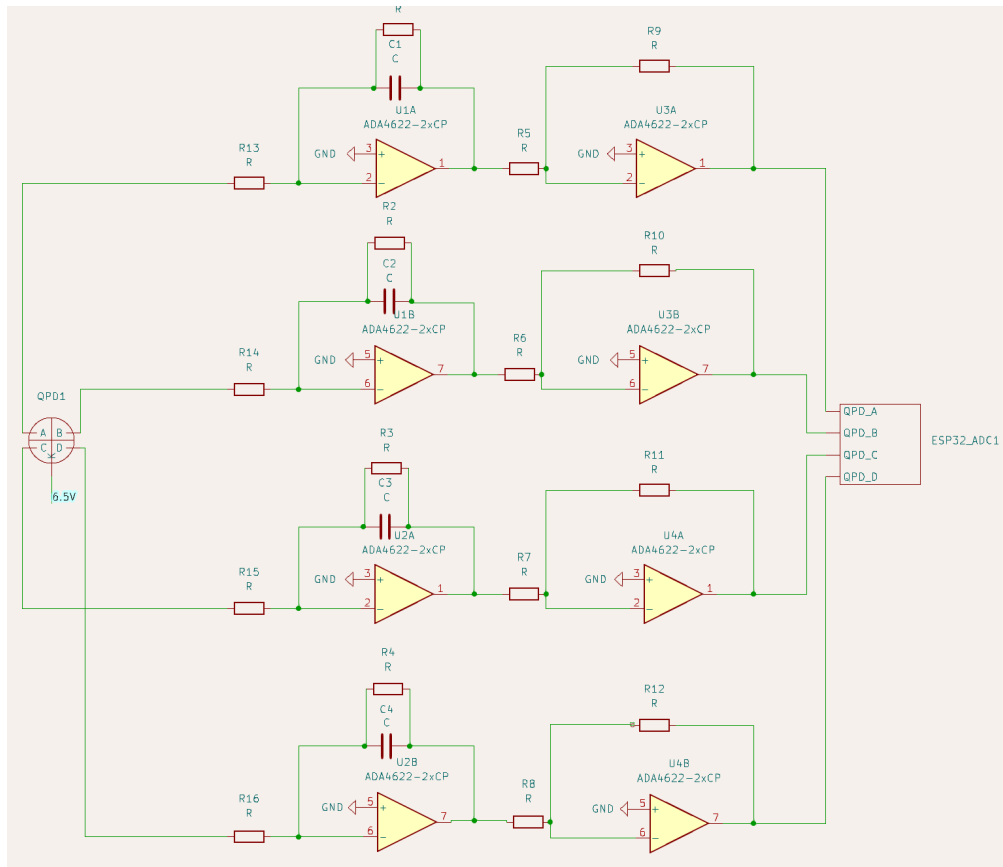


Fig 6.2.2-3: Realized Circuit of the Receiver PSUs

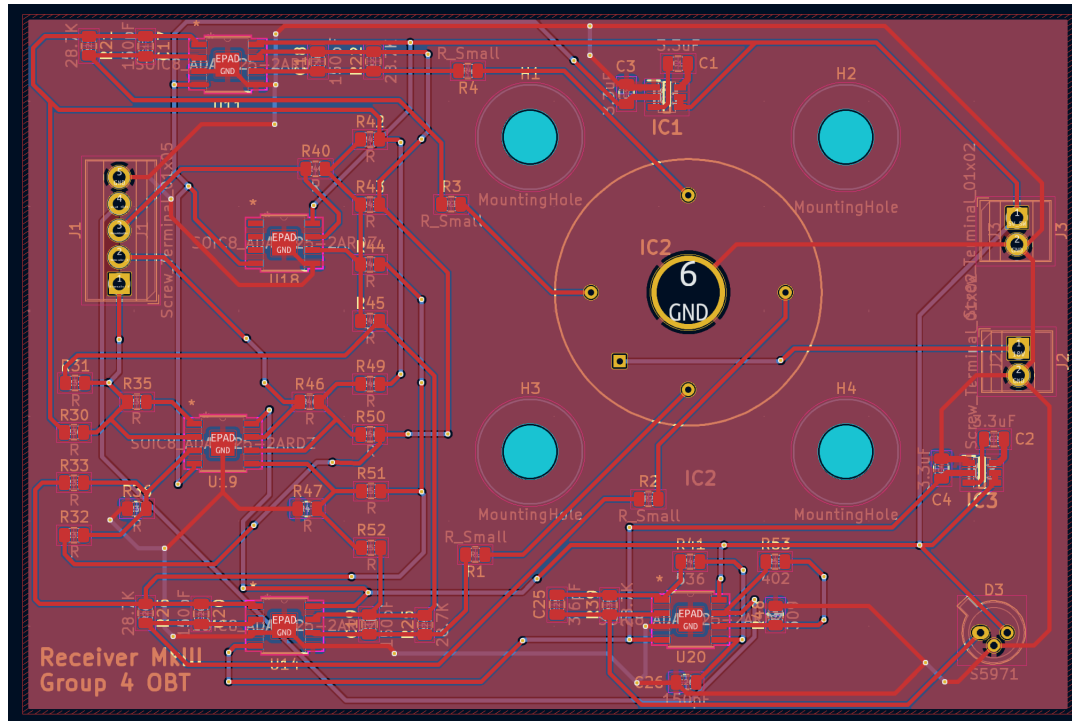


Fig 6.2.2-4: PCB Layout of Receiver I

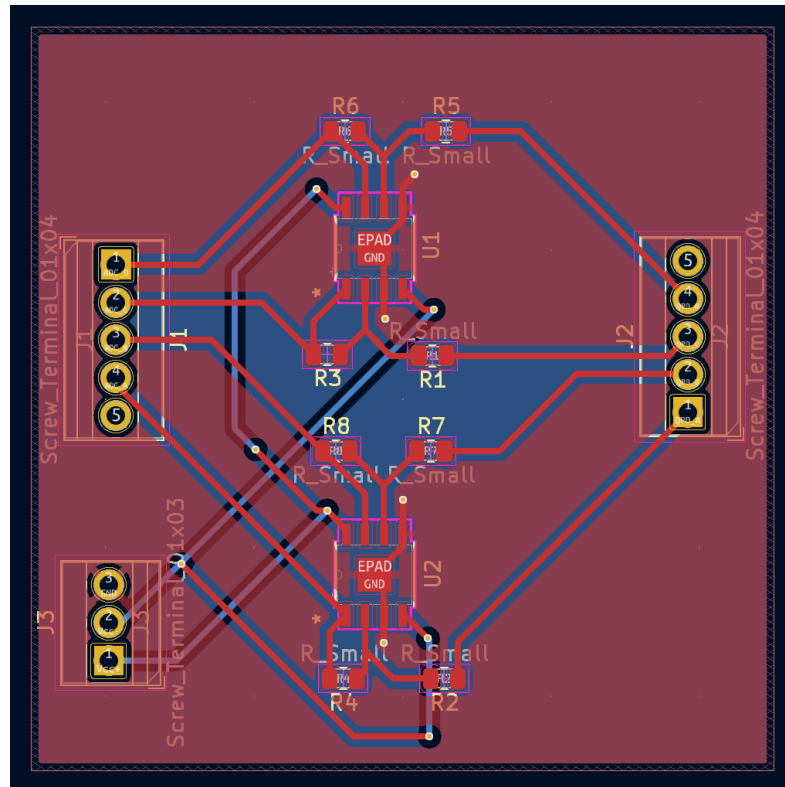


Fig 6.2.2-5: PCB Layout of Receiver II

6.2.3 PSUs

To account for the various modules and motors we needed to be able to supply two different regulated voltages, 3.3V and 6.5V. To do this we designed our PCB to house one voltage regulator and take advantage of the 5 board minimum from the manufacturer. Both the beacon and receiver are to be outfitted with two iterations of the PSU board to attain the two different output voltages needed.

Deciding the output voltages was simply based on component requirements. As the current draw for the GPS, IMU and LoRa modules came to about 500mA and therefore could not all use the ESP32s 3.3V pin out alone it became apparent that a voltage regulator capable of a 1 amp minimum was needed. To accommodate this we decided to use an LM2657 switching regulator to step down the 22.8V from the battery to 3.3V, the LM2657 is capable of up to three amps and was suitable for our needs. The servo motors on the other hand had different operating voltages, the beacon motors requiring anywhere from 5 to 6.5V and the receiver motors requiring a voltage within the range of 6 - 7.4V. To accommodate both motors we decided to use the same LM2567 used for our 3.3V regulator and simply set it to output 6.5V. The largest current draw from a single motor was found to be just over 800mA from the FeeTech motors used on the receiver. The largest current draw we expect to see is 1.6A from the FeeTech motors, which again is well within the output range of our voltage regulator. Below is the schematic and diagram for the PSU board in its 6.5V configuration.

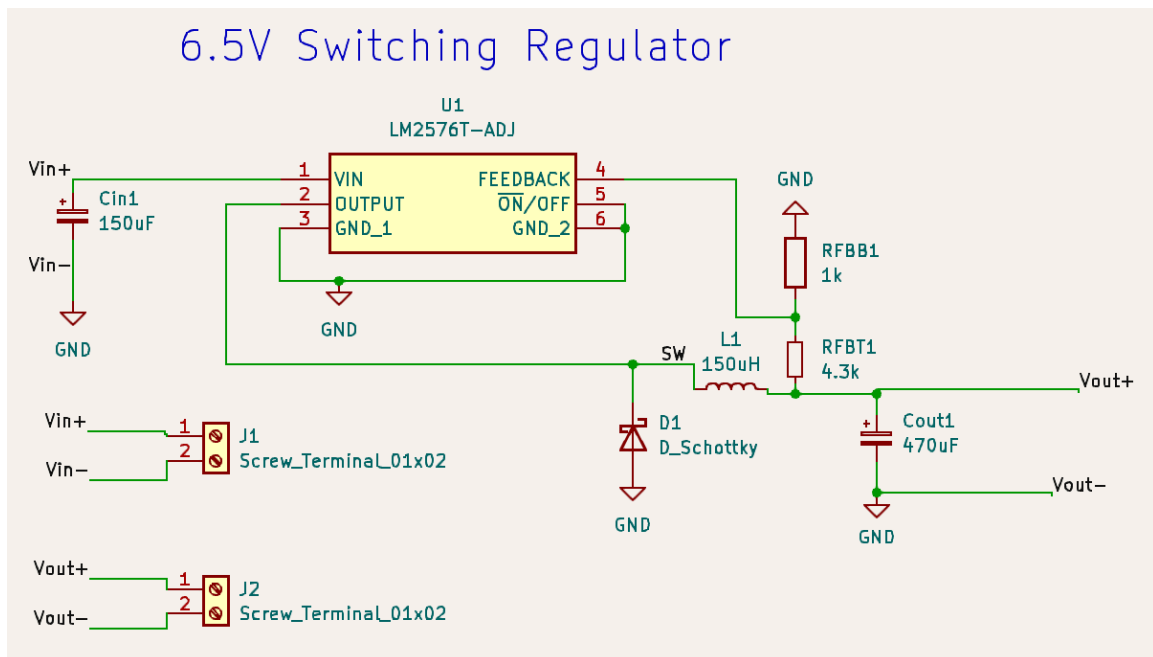


Fig 6.2.3-1: Schematic of Voltage Regulator

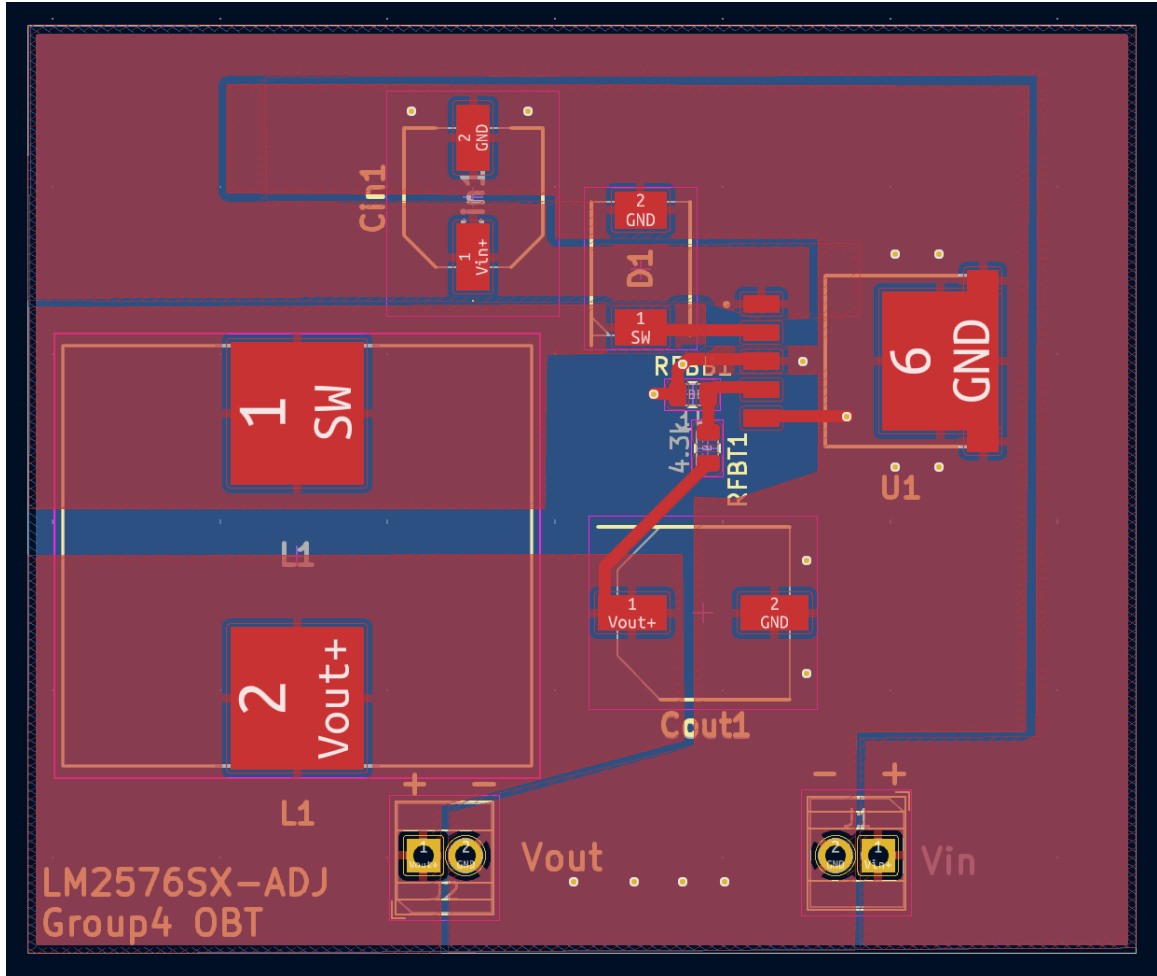
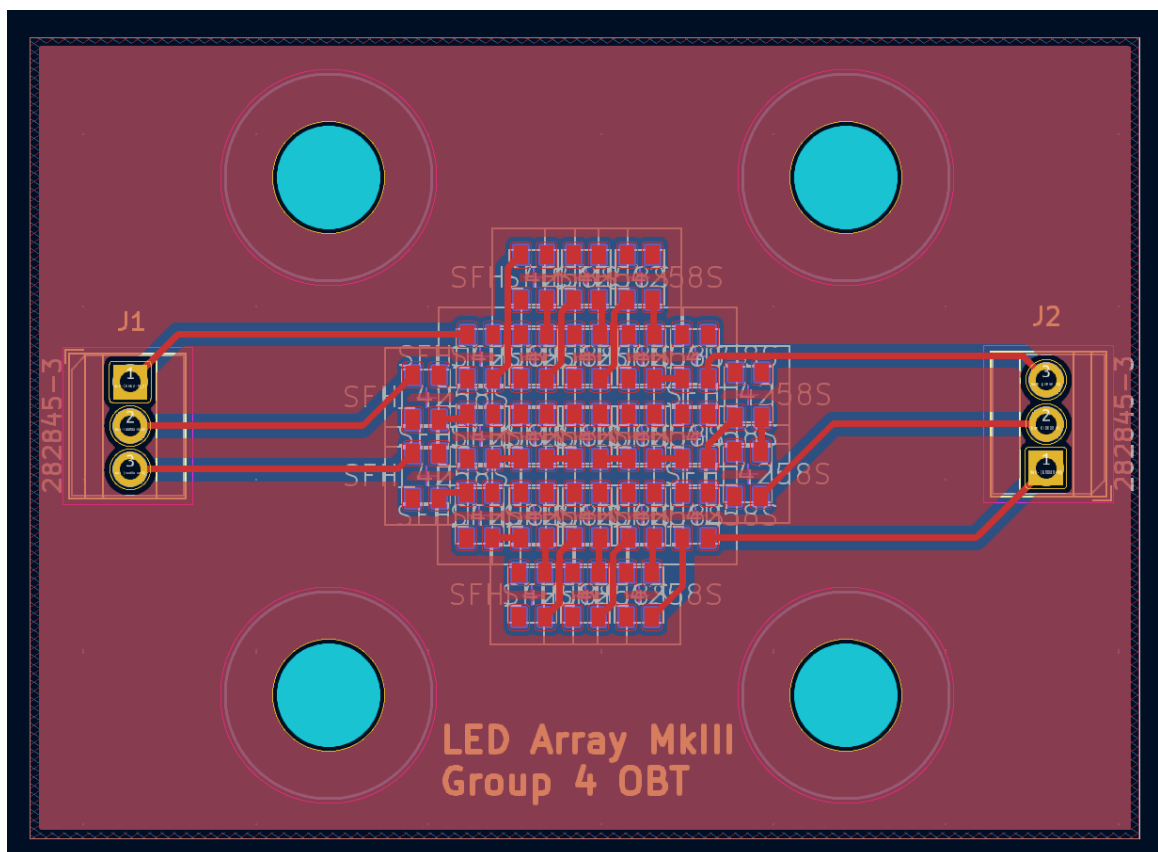
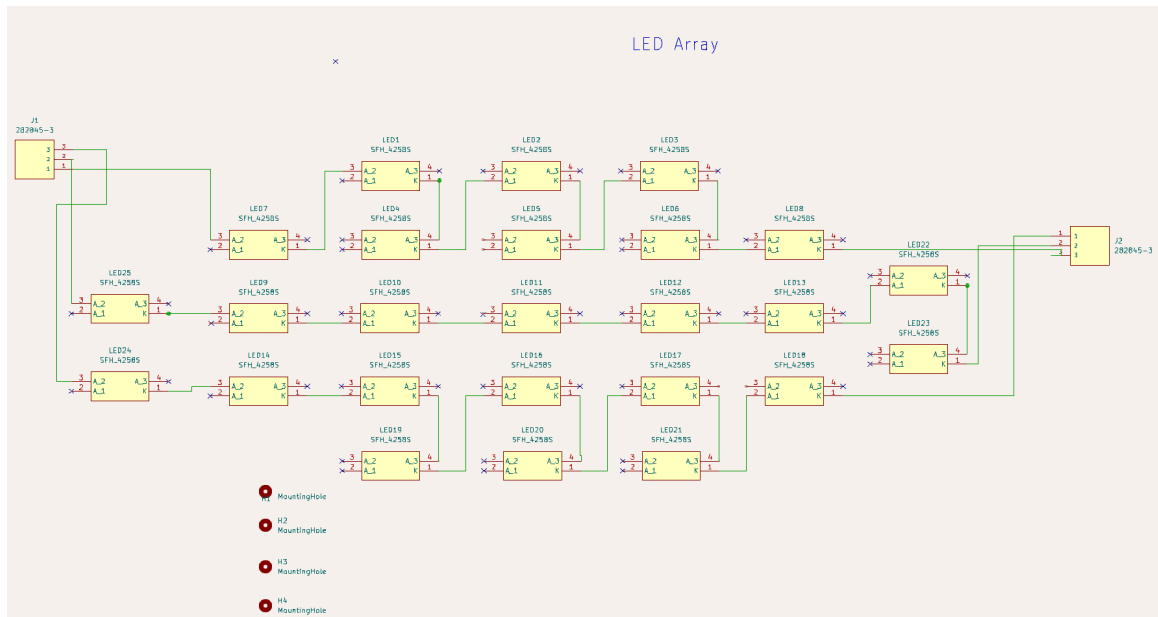


Fig 6.2.3-2: PCB Layout of Voltage Regulator

6.2.4 LED Array

The LED array is composed of 25 NIR LEDs made up of 3 series runs (8,8, and 9). To drive the LED array we are using 3 off the shelf CC buck regulators. Each regulator is set for the forward voltage of each series and a current limit of 80mA. This current limiting causes the output voltage to drop as the LEDs become hotter and their forward voltage begins to drop. Additionally, copper fills were added to the top and bottom layer along with a heat sink attached to the back of the board to combat the high thermals outputted by the array. To modulate the array we have used MOSFET switches that can be turned on and off via the ESP32s 3.3V logic signals. Below are the schematic and PCB layout of the LED Array. It should be noted that the close proximity and shape of the array were determined by the size and orientation of the collimator lens.



6.2.5 MOSFET Switches, IMU, GPS, and LoRa Modules

In this subsection we will briefly discuss the different off the shelf modules used to realize our project and how they interact with the other components already mentioned. To begin, in order to achieve various pulse widths for our LEDs we selected the Anmbest MD114. This module consists of two MOSFETs in parallel, this design allows for loads as high as 15A or 400W under normal temperature conditions. The gates are biased using a 3.3V signal from the ESP32, when the voltage is applied the switch becomes open and allows for current to flow through. If the modulation is high enough the LEDs will simply dim but for our application we ensure that the pulse widths are wide enough to achieve full output power, allowing for the receiver to recognize the signal at longer ranges.

Next, we have the IMU (QMC5883L) and GPS modules (GY-NEO6MV2). Both require a 3.3V power source which is supplied by our LM2567 voltage regulator. Both of the modules will need to relay positional data to the ESP32 that will then in turn be used to calculate a bearing angle and sent off to the other MCU via the LoRa module. Positional data being received in a timely manner is essential to the general alignment between the beacon and receiver. The part selection section will explain more on the specific modules and the reason behind their selection.

For our last module we have the LoRa module (RYLR890). These modules will need to be able to both transmit and receive positional data from one ESP32 and relay it to the other MCU. With a communication range of up to 4 km depending on the antenna it is outfitted with, it made perfect sense to implement this specific module. Furthermore, the wide frequency range provided many options for transmission, 820 to 1020 MHz.

In conclusion the modules selected were chosen based on their specifications and capabilities. The integration of these modules were made fairly easy based upon their common operating voltage and compatibility with ESP32 libraries.

6.3 Software

6.3.1 Circuit Design

This group plans to use two main softwares for designing the circuitry in this project:

- MATLAB
- LTSpice

LTSpice is one of the programs the group will be using in this project. LTSpice is a powerful, free-to-use SPICE (Simulation Program with Integrated Circuit Emphasis) software tool developed by Analog Devices (formerly Linear Technology) for simulating electronic circuits. The Optical Beacon Tracker project involves multiple electronic components, such as microcontrollers, servo motors, laser diodes, and sensors, all requiring stable power supplies. The power supplies will be from a battery so a necessary buck conversion will also have to be designed. This software will help us in the testing

phase when choosing appropriate components, such as voltage regulators, capacitors, and inductors.

To get into specifics, when creating a voltage in regulator circuits in LTSpice, one would run transient and steady-state simulations to analyze the output voltage and current stability. When filtering and processing the signals from the Quadrant Photodiode (QPD). LTSpice can be used to design and simulate these analog signal processing circuits, such as amplifiers and filters, to ensure they perform correctly before implementation. It will also use dot design and simulate the laser diode driver circuits, ensuring they provide the correct current and pulsing characteristics. This will be used more so in the testing phase. It will be used in conjunction with another software, namely MATLAB.

MATLAB (Matrix Laboratory) is a high-performance language and environment developed by MathWorks, primarily used for technical computing, data analysis, algorithm development, and simulation. It offers extensive libraries and toolboxes for various applications, including signal processing, control systems, and data visualization. MATLAB will mostly be used in this project for coding and data verification. The software has a lot of support for working with electrical components and signals. The main thing that this group will be using it for is simulating the laser diode characteristics. Unlike LTSpice, MATLAB can easily simulate de-modularization as well as implement virtual filters to process the raw data to filter out noise and extract meaningful information.

Although the other abilities of MATLAB allow you to use it with a microcontroller, the group wants to avoid this. Regardless, MATLAB's Control System Toolbox can be used to design and simulate control systems for the servo motors and gimbals used in the project. This helps ensure precise and stable control of the tracking system.

One thing to note that can be useful is the Hardware-in-the-Loop (HIL) Simulation. The team is also not planning to use that, but MATLAB can be used for hardware-in-the-loop (HIL) simulation, where parts of the system are simulated in MATLAB while interfacing with actual hardware components. This allows for testing and validation of algorithms and control systems with real-world signals.

6.3.2 Software Use Case Diagram

The Software Use Case Diagram outlines how the users will interface with the system. It depicts all the different ways the user can use the system. In this case, the two sides the User can interact with the system is the GCS on the receiver and transmitter side. The user can check GPS coordinates on either side. For the requirements of this project, the user will only be able to communicate one way. That is through the transmitter side. Here the user can start up the system, transmit the LED and laser diode, and start the

communication by typing in their desired strings through the Terminal. This terminal will use the UART protocol to interface with the ESP32 and convert it to binary. It will then transmit the data using the laser signal, which should be locked on through the QPD.

Software Use Case Diagram

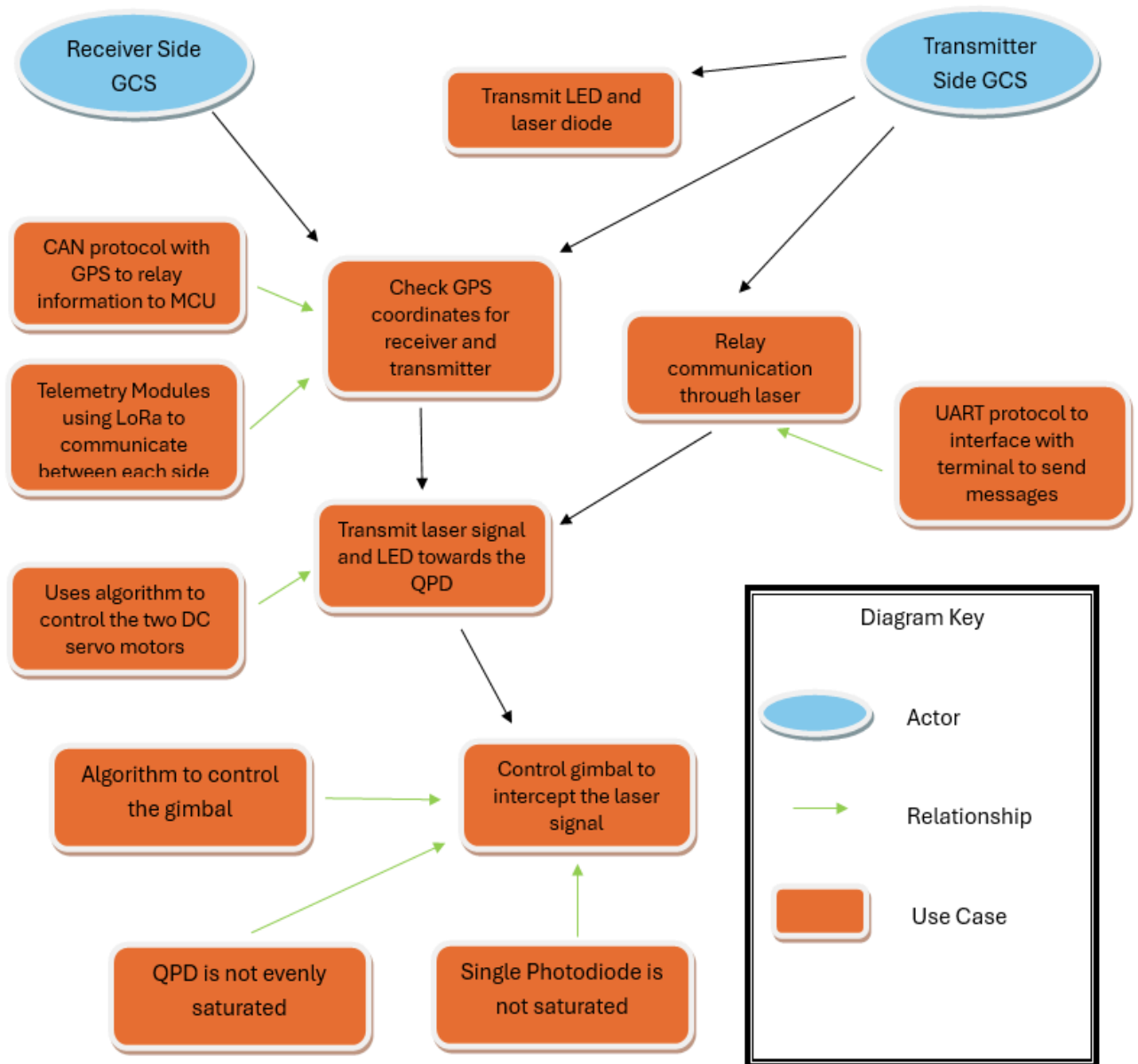


Fig 6.3.2: Software Use Case Diagram

6.3.3 Software: Motor Control

This diagram outlines the state diagram of the DC motor and gimbal. On the transmitter side, the ESP32 only has to interpret the GPS coordinates and then command the X and Y axis motors to move to said coordinates. Unless new GPS coordinates are received, the motors will no longer move. On the Receiver side, the initial objectives, up until moving the gimbal to the desired coordinates, are the same as the transmitter side. The big difference is the gimbal is planned to constantly be moving, adjusting its position to ensure that a stable connection with the single photodiode and the laser signal is maintained.

Motor and Gimbal Control

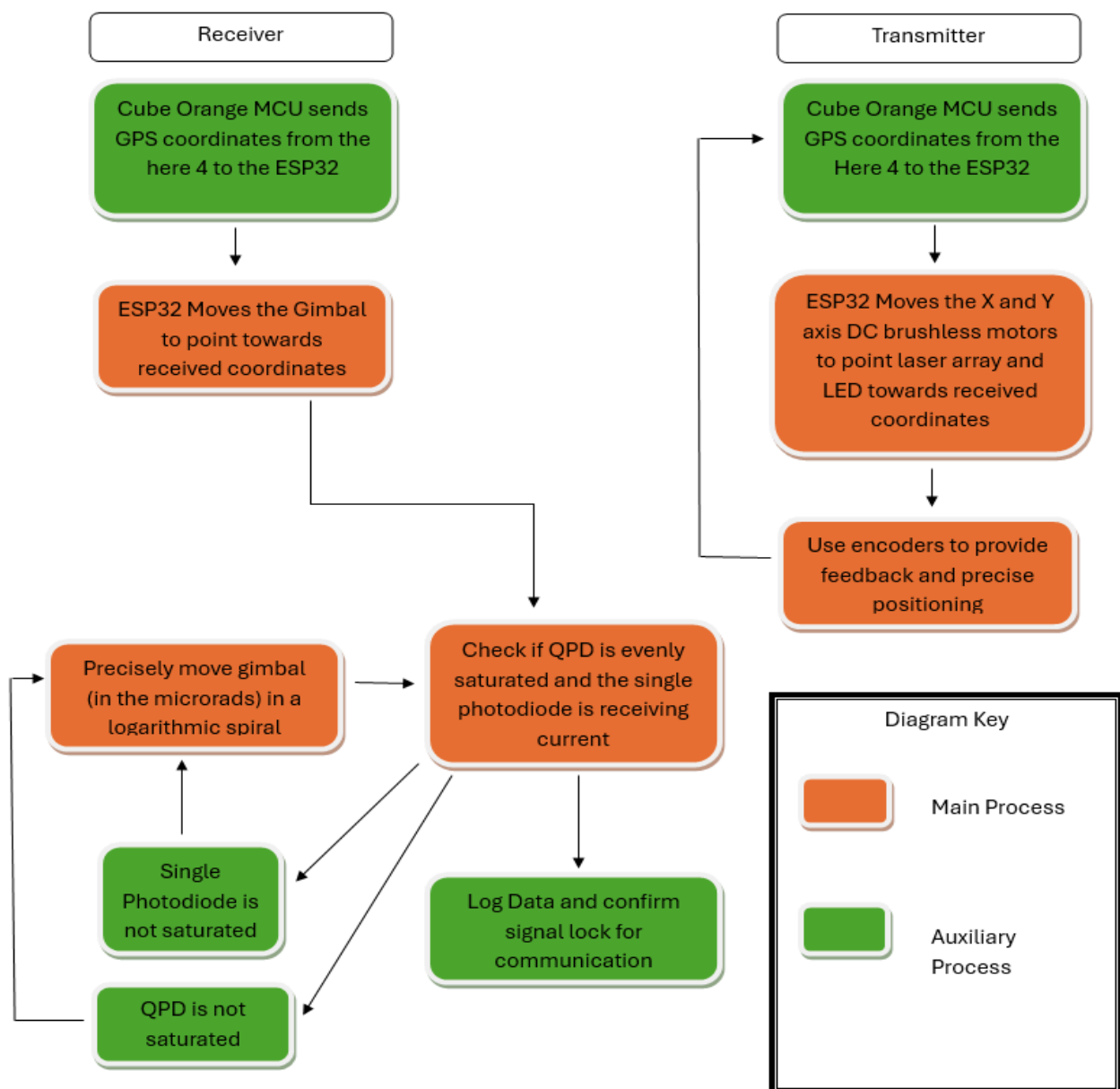


Fig 6.3.3: Motor and Gimbal Control

6.3.3 Software: Laser Diode and LED Control

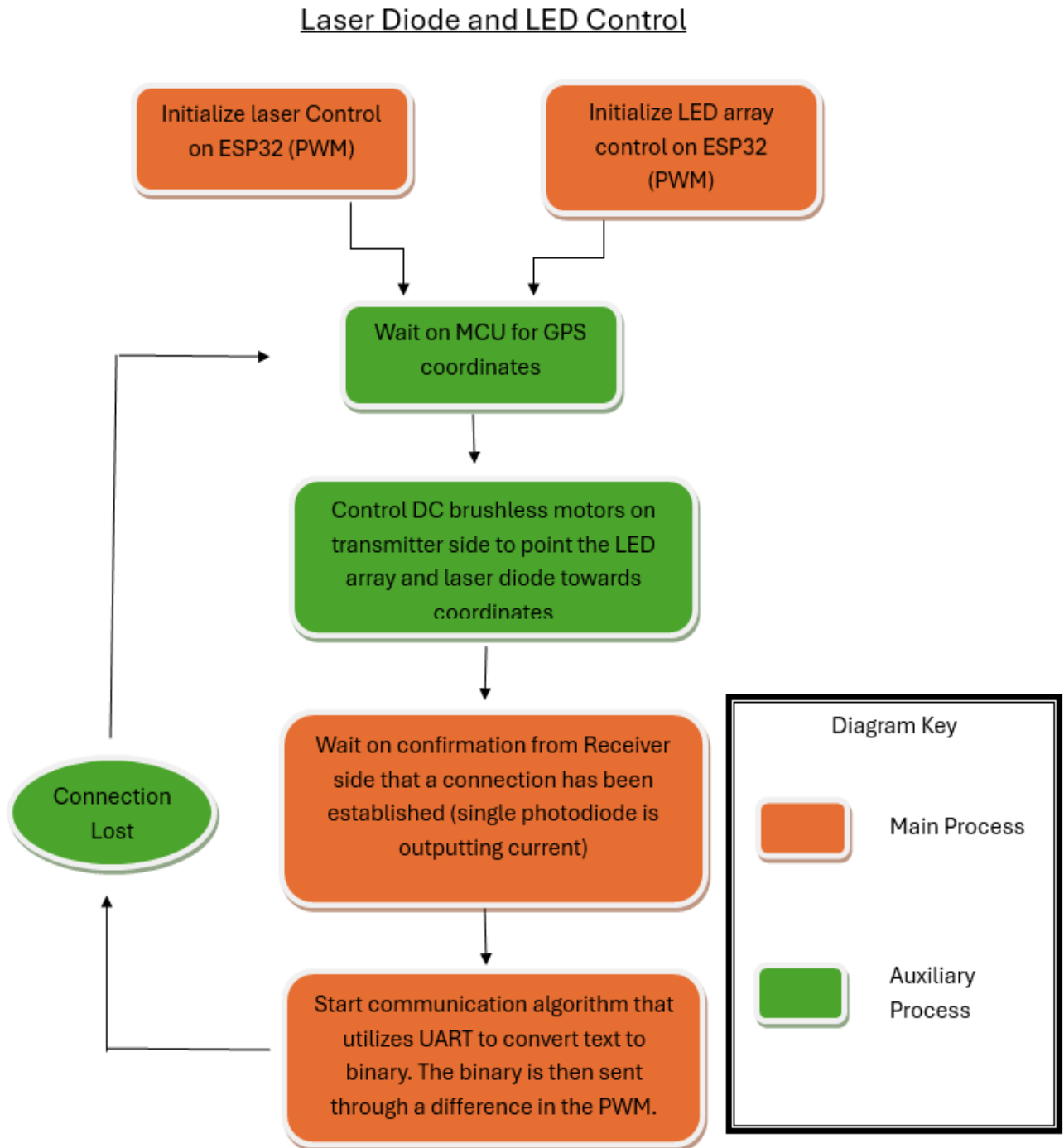


Fig 6.3.4: Laser Diode and LED Control

Pulse Width Modulation (PWM) is critical in this project, as it modulates the intensity of the light sources (laser diode and LED array) to convey data and ensure correct alignment. Setting this up for the Laser Diode is quite simple. The 'ledc' library defines the frequency and resolution of the PWM signal. The 'ledcWrite' function then could be used to adjust the duty cycle according to the data that is being transmitted.

The LED will be set up in a similar way (libraries are not needed and all the needed functions are intrinsically supported). Once initialized, the duty cycle will be set and not changed. The laser will be modulated in a similar manner (PWM) but will still require a laser driver.

For our stretch goal, the project should encompass communication. To achieve this, the Ground Control System (GCS) needs to communicate with the ESP32 microcontrollers on both the transmitter and receiver sides. The UART protocol is used for this serial communication. The User will enter in their desired string of characters and the code will convert it to binary. If the photodiode detects light (voltage above a certain threshold), interpret this as a binary "1." Since the laser signal must be present to confirm a stable connection is there, a binary "0" cannot be 0. To work around this, the software will interpret the "voltage below a certain threshold" as a binary "0".

An even bigger stretch goal to ensure that the data transmitted is reliable:

- The timing must be correct
- The synchronization must be the same for the transmitter and receiver
- Error handling must be implemented

For timing, one must ensure that the reading frequency matches the transmission rate to correctly interpret each bit. This can be decided beforehand, like the LoRa, and established during the LoRa datatransfer or just hard coded into the ESP 32. Just as important, is the bit duration. Both the transmitter and receiver must agree on the duration of each bit. Otherwise not a single bit will be read correctly. This can also be established the same way as the bit interpretation. Start and stop bits can be implemented at the beginning and ending of a data packet.

For error handling, redundancy implementation is essential. One could add parity bits or checksum to ensure the data integrity. To reiterate, the goal of the project for our team is not communication. It is optical tracking to help establish communication. This project will be handed off to someone else who will develop the communication aspect. At that point, it will be a mostly software issue.

6.3.4 Software: MCU and PCB Connection

MCUs & PCBs Connection

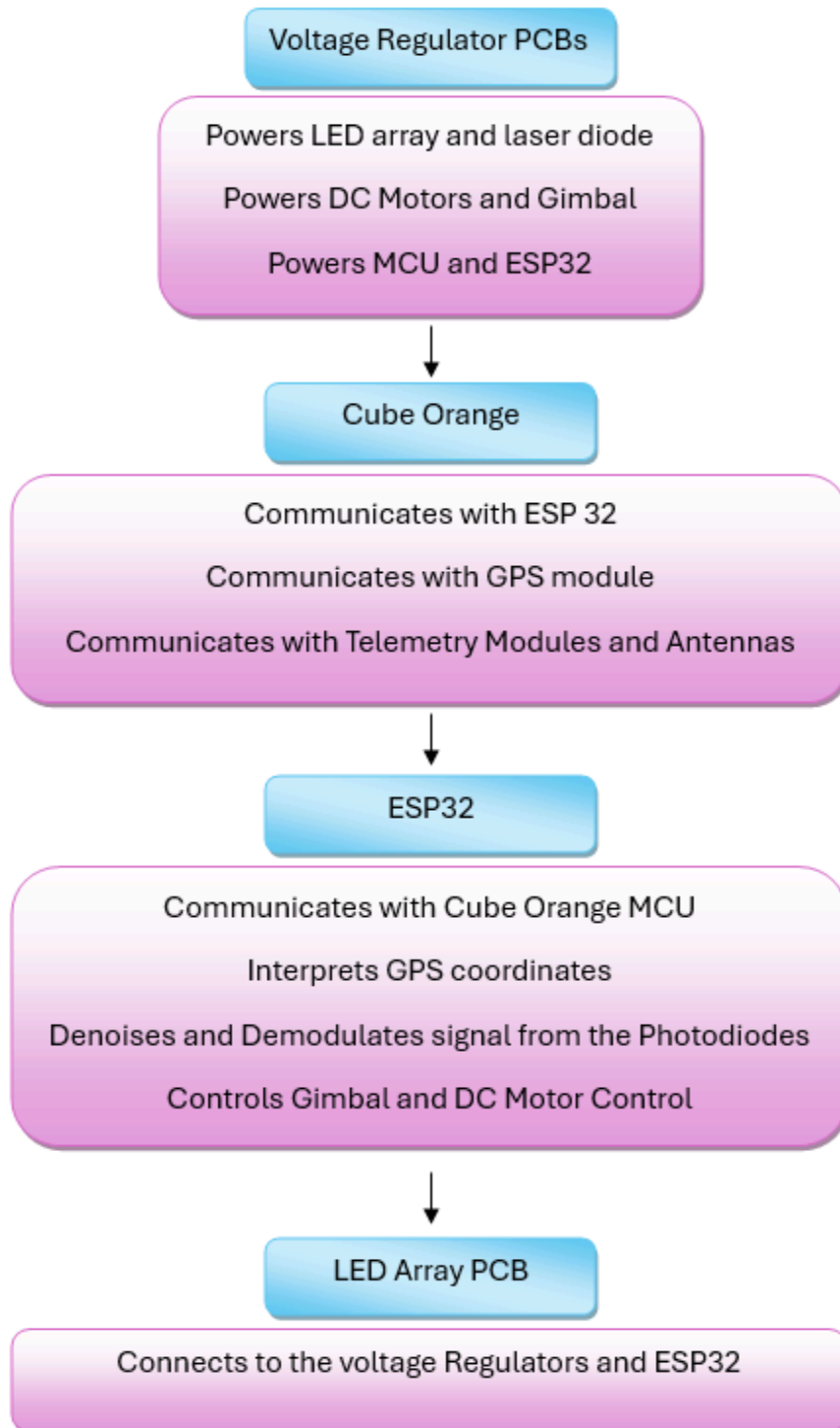


Fig 6.3.2-1: Software Use Case Diagram

This state diagram shows how all the custom PCBs will interface with the MCUs. The custom voltage regulators are simply designed to meet the power needs of all our components while using a simple large battery as a power supply. The Cube Orange's job is to interface directly with the GPS and telemetry modules. It cannot be directly coded on with Arduino since Ardupilot is installed on its firmware. The ESP32's job is to communicate with the Cube Orange. It has to take in all the raw data from its GPS. The ESP 32 also has to interface with the QPD and single Photodiode. It must denoise and demodulate the signal from them and use that data in the tracker algorithm used to move the gimbal.

6.3.5 Microcontroller programming

It has been established that C will be the code of choice for the microcontrollers in this project. This language will be used on these two potential platforms:

- Arduino IDE
- MATLAB

As mentioned earlier, the group did not want to use MATLAB for controlling the microcontroller, so all of the experimentation and testing will be done using Arduino IDE.

Besides the mentioned capabilities of the base program, such as interfacing with all the pins of the microcontroller, built in functions, and extensions, the program allows one to add many libraries.

The Arduino IDE software has a wide variety of libraries that can be used for being able to interface with our components as well as streamlining our code. Some of the following libraries are ones our group looked at and considered using:

- Async TCP
- ESP Async WebServer
- ESP32Servo
- ESPUI
- LittleFS_ESP32
- LoRa
- QMC5883LCompass
- TFT_eSPI
- TinyGPSPlus
- Math
- WiFi
- umm_malloc/umm_heap_select
- DNSServer

- ArduinoJson

More than half of these libraries will be used to create our web based GUI. That is discussed in its respective section. The reason we decided to do that is because of the complications that we ran into using our ILI9341 display. For some reason, there's specific issues with the 2.8 inch version and an esp32 dev kit. They were eventually resolved, but that solution involved directly plugging in the led pin to our dev kit, which promptly killed the onboard psu. For that reason, we ditched the LCD touch screen and went to a web based version. The downside is that it doesn't work on wifi setups like the school. A workaround is to create a hotspot off the computer and connect to that.

The BH1750 is another library we are using. The library uses 0x23 address as default, but it is possible to define any other address. It is also recommended to change to 0x5C if any issues are found with the default value. Like most other programs, the I2C needs to be initialized. Here the user can select between six different measurement modes. Each mode is divided into two groups: continuous and one-time measurements. In 'Continuous' mode, the sensor continuously measures lightness value. In one-time mode the sensor makes only one measurement and then goes into 'Power Down' mode. This project will be using continuous mode as the laser signal needs to be constantly analyzed in order to establish a good connection. Each mode also has three different provisions:

- Low Resolution Mode - (4 lx precision, 16ms measurement time)
- High Resolution Mode - (1 lx precision, 120ms measurement time)
- High Resolution Mode 2 - (0.5 lx precision, 120ms measurement time)

Each has their own tradeoff and has not been experimented with yet to make a decision.

TSL235R is a light to frequency converter library. This library does not measure the frequency. That should be done with an interrupt function (user code). The library provides functions to calculate the radiation and with functions to compensate e.g. for wavelength and voltage used.

Some of the libraries are unknowingly necessary. For example, the esp32 dev kit does not have automatic support for controlling servos. The servo.h library that's commonly used for many arduino micro controllers does not support the Expressif architecture.

Id2410 is a library that is used for tracking radar and provides some useful built in functions. The group is also not sure it wants to utilize this library as it may not help it with this project specific use case scenario (tracks distance and other things using radar and not GPS for example). For consideration and research will have to be done in SD 2.

6.3.5 Web Based GUI

Our web based GUI was designed off of the ESPUI library. It is an extensive library that offers a lot of quality of life features. Adding color or changing the borders was very simple. These simple features do not allow for a lot of customization. That must be done by the user. The library felt similar to CS in many ways.

This library needed a lot of other libraries to work, one such library being the ArduinoJson library which contained most of the predesigned models, such as colors, and box shapes. Another one is malloc and heap select. This library was absolutely needed so that ESPUI could function.

There were several libraries that were downloaded that were not needed. One such library is the LittleFS_esp32. The only purpose of this library is to make more room for the esp32s memory. It takes unused data in the onboard files and stores them on files in the arduino IDE.

The purpose of this UI was to add some user interaction despite the system being most automatic. This meant that the user could stop/start the project. The type of information that will be displayed is the transmitter and receiver azimuth, the longitude, latitude, and altitude of the transmitter and receiver, as well as the response time to the tracking (the length of time it takes for the receiver to get information for adjusting the alignment for tracking to sending a signal to the motors to move).

7 Integration and Prototype Construction

7.1 Microcontroller and GPS Integration

The group's first task was to integrate the Microcontroller and GPS integration. Some of the listed parts are bought for testing. These were the first parts chosen for testing:

- **Microcontroller:** ESP32 Dev Kit
- **GPS Module:** GY-NEO6MV2 NEO-6M GPS

Interfacing ESP32 with the NEO6MV2

Connections:

- Connect the GPS module's **TX pin** to the ESP32's **RX pin (e.g., GPIO16)** and the **RX pin** to the ESP32's **TX pin (e.g., GPIO17)**.
- Power the GPS module using the ESP32's **3.3V or 5V** pin and connect the grounds (GND).

Libraries:

Use the **TinyGPS++** or **NeoGPS** library for parsing GPS data like latitude, longitude, speed, and time.

Code:

Initialize the ESP32's UART communication using the HardwareSerial library. Set the baud rate of the GPS module (in this case 9600) and write code to read and parse NMEA sentences from the GPS.

Testing:

Use a serial monitor to display parsed GPS data. Ensure the module has a clear view of the sky for satellite locks.

In the case of the TinyGPS++ library, it can parse most of the code for us and store it in a struct. All we have to do is call from that struct and specify that accuracy to a certain decimal point. Although slow (at best the data was received about once a second). The data seemed fairly accurate and the code was clean. This allowed testing to go smoothly.

The longest part was waiting for the first lock. Whenever a GPS module is making a lock for the first time, it can take up to 30 minutes, especially when the module is cold.

7.2 IMU integration

Our next priority was connecting the imu to the microcontroller. Since the first subsystem was simple enough, we decided to combine it with that sub-system. Here are the parts chosen for testing:

- **Microcontroller:** ESP32 Dev Kit
- **GPS Module:** GY-NEO6MV2 NEO-6M GPS
- **IMU Module:** GY-271 QMC5883L

Connections:

- Connect **VCC** to the ESP32's **3.3V pin**.
- Connect **GND** to the ESP32's **GND**.
- Connect **SCL** and **SDA** to the ESP32's **I2C pins** (e.g., GPIO22 for SCL and GPIO21 for SDA).

Libraries:

Use the **QMC5883LCompass library** and it should come with everything you need, such as calibration and data smoothing.

Code:

The IMU needs to be initialized using the hardware serial interface, but for I2C. The address won't be specified as the library already does that for you. The mode can be changed depending on what address the user uses.

Testing:

The first step is to calibrate the IMU. This is because the IMU is very prone to changes such as electromagnetic and shock. Even when it is calibrated, it doesn't mean those parameters cannot change over time. The calibration is also very finicky. One of the

reasons we chose this specific library (QMC5883LCompass) is because it had a better calibration program. Regardless, nearly all of them require you to spin them along the x, y, and z axes that are labeled on the chip. The user should only rotate it in one direction at a time due to the nature of the code. Once this was done, the user can easily request the orientation, such as the azimuth and the x, y, and z from a struct, just like the GPS library.

The testing was briefly done as we requisition data at the same time from the GPS and IMU.

7.3 Telemetry Communication

This subsystem involves LoRa and is necessary for the transmitter and receiver side to communicate their position and movement data. For testing, the group chose these components:

- **Telemetry Modules:** RYLR896 Lora Module SX1276
- **Microcontroller:** Hiletgo ESP-WROOM-32

The manufacturer, REYEX, recommends that the users use the standard LoRa library for arduino IDE. It happens to be compatible with all esp32 and the Expressif architecture. Here is the pin assignment for the following module:

Telemetry Module: RFD 900x	Microcontroller: Cube Orange+	Direction
GND	GND	Either
5V	Vcc or 5v	Either
Serial 1 TX	RX	Input
Serial 1 RX	TX	Output

For the VCC and GND you just match these pins on most boards. The TX and RX crossover.

Install Required Libraries:

The ESP32's native HardwareSerial library is sufficient for UART communication with the RYLR896.

Initialize UART:

Use the ESP32's secondary UART interface (Serial2) by initializing it in the code.

LoRa AT Commands:

The Final step is to use AT commands for configuration and data transmission. This tends to be the case for most UART communication based LoRa modules. The following are the LoRa commands we used in our setup:

- AT: To check module communication.
- AT+ADDRESS=<ID>: Set the device address.
- AT+NETWORKID=<ID>: Set the network ID.
- AT+SEND=<address>,<data>: Send data to a specific address.

It is important that delays are set for transmission and receiving. This gives the modules on either end time to send and receive data. This also allows them a bigger window of time so that you aren't skipping transmissions on the receiving end for example. \

7.4 Telemetry, GPS, and IMU Integration

Once the GPS and microcontroller are confirmed to have worked, the system needs to be able to communicate its location to another microcontroller. The parts that will be used for this subsystem are as follows:

- **Microcontrollers:** Hiletgo ESP-WROOM-32
- **Telemetry Modules:** RYLR896 Lora Module SX1276
- **GPS Module:** GY-NEO6MV2 NEO-6M GPS
- **IMU Module:** GY-271 QMC5883L

For this subsystem, code should be written so that all of the positional data is taken from the GPS and transmitted over LoRa to the other telemetry module through the antenna. This data position will be verified by measuring in the real world and cross examining it with the outputs on Mission Planner.

To make efficient use of the code, everything was separated into its own separate functions. There was a separate function for sending and receiving gps data, as well as imu data. This was done in preparation to integrate this subsystem with the next.

Since the GPS and LoRA both take up quite a lot of time, the GPS data should be sent sparingly.

7.5 LED Array Control/Modulation

As of now our plan is to drive the 850nm array with 3 CC buck regulators set to each of the three series' threshold voltage. They will be current limited to 80mA per series for outdoor demos as well. To modulate the array we will be hooking 3 mosfet switches to the same GPIO pin of the ESP32 S3. This will turn the LEDs on when the signal received is high and off if no signal is received. We will be pulsing the LEDs as needed, indoor demos <1Hz along with a lower forward current (50mA) and outdoor demos at approximately 10Hz at 80mA.

7.6 Quadrant Photodiode (QPD) Integration

The Quadrant Photodiode and the InGaAs photodiode are another integral part of the system that may take the most amount of time later in the integration plan. Our first step however is making sure both detectors work and that it is outputting some data to the microcontroller. Since the single photodiode is such a small as well as similar integration to the QPD, it will be worked on at roughly the same time as the single photodiode. The parts that were chosen to integrate in this subsystem are as follows:

- **Microcontroller:** Hiletgo ESP-WROOM-32
- **QPD:** SPOT-4D-0: Annular Quadrant Silicon Photodiode
- **Photodetector:** FGA015 - InGaAs Photodiode
- **Laser diode:** FPL1055T
- **Simple Electrical components:** Resistors, capacitors, OP amps, wires etc.

Testing for the InGaAs photodiode is not that difficult as it can simply be attached to a breadboard, with a resistor, and connecting the 5v, GND, and analog input pin or an I2C pin. Once set up, it is important to note which ADC and which IO pin you plugged into. The ESP32 has two ADC modules (ADC1 and ADC2). There are 8 IO pins that are connected to ADC1 and 10 pins that are connected to ADC2. To measure analog input signals, you must connect them to IO pins that are connected to either of the ADC modules.

Libraries will also be an important contributing factor but for now, the group will download a simple and robust library, called BH1750, in the Arduino IDE program (can also be found on GitHub). Once a connection is established with the photodiode, ESP 32, and the Arduino IDE, the I2C address is declared and the I2C bus is initialized. Next a Boolean is written that can be used to detect setup problems. Then a loop simply prints out the lux that the sensor detects from the 1550 nm laser diode signal. This is simply a test to make sure everything works as measuring the lux is only one aspect of this system. It is necessary that this system detects a certain wavelength, its irradiance, and its track loop. This will be explored on the hardware side in Senior Design 2.

Once I2C is up and running and it is known that all the connections work, the group will now work with another library called TSL235R. This is a light to frequency convertor as mentioned before. This library will allow us to accurately detect our laser signal from FPL1055T, when the correct wavelength is put into the code. Once it is established that the data matches up, the group should test the frequency and pulse response of the photodetectors.

Once the laser diodes are set up to use pulse width modulation, they need to be demodulated on the receiver side. In this case, that is the QPD, circuit and microcontroller. It is possible to do this through the hardware side or the software side. Both will be explored in this experimentation.

For the software side, the group just must code the time between pulses, the carrier signal, and carrier frequency. Our code will be cross examined with MATLAB results. To

do that, we will enter all the input parameters that are read from the QPD on the serial monitor in the Arduino IDE. Then we will demodulate and print a graph that outputs the transmitted signal, received signal, and demodulation signal. It will display the power spectrum and frequency of each. It is highly important that the data signal data that is being input into the microcontroller is accurate. The whole system will not work otherwise.

If the software side does pose issues, there are chips that could be integrated into a circuit that could demodulate the signal. The group, however, wants to avoid this as that adds complexity, money, and overall wait times for the system (another PCB that must be ordered) as a whole.

Since InGaAs photodiode sensors have a typical sampling rate of 15 Hz, this testing will start out at 10kHz since that will cause the measurements to be more accurate. Once it is determined that the demodulation works and the irradiance is accurate, the testing is done, and the group can move on to the QPD.

The Annular Quadrant Silicon Photodiode initial testing will be the same as the single photodiode. It will be wired up with the low voltage, ultra-low power, low-noise quadrant photodiode circuit mentioned before, and the pins matched to any four open analog inputs. It is then checked with the test code used before for the single photodiode if there is any lux output under ambient light and direct light. The only difference in the code is including the multiple pins needed for the QPD. Next the group will test the irradiance in a similar fashion and verify that the system is accurate. Once established the group can start experimenting with the light spot position/saturation. Using the given equations mentioned before:

$$V_{PD1} = -R_L(i1+i2+i3+i4) - R_L i_1 \quad (\text{equation 7.6.1})$$

$$V_{PD2} = V_B - R_L(i1+i2+i3+i4) - R_L i_2 \quad (\text{equation 7.6.2})$$

$$V_{PD3} = -R_L(i1+i2+i3+i4) - R_L i_3 \quad (\text{equation 7.6.3})$$

$$V_{PD4} = V_B - R_L(i1+i2+i3+i4) - R_L i_4 \quad (\text{equation 7.6.4})$$

The group will be able to find the inverse polarization voltages of the photodiodes with our given circuitry set up. The values will be printed on the serial monitor and will be verified in real time by moving the laser diode around the QPD. Once we can verify that it's working, the group can employ the noise analysis equations into the code as well. The noise analysis helps get a more accurate reading of the signal.

Now that both parts have been tested and verified, the group will move onto combining both systems. The QPD has a small pinhole at the center that the single photodiode will be lined up with. It has not been decided how it will be kept in place, but some sort of adhesive has been suggested. Once they are both mounted, they will be rewired, and both algorithms for scanning the analog inputs will be implemented. The algorithm that centers the signal on the QPD should allow for the 1550nm laser signal to go through the pinhole and reach the single photodiode. This part is prone to human error so the

photodiode will be attached with tape first, then when the pinhole gets sufficient signal through it, the part can be glued.

Once we have verified that the system works and the data is accurate, the group will go ahead and finalize the PCB for the circuit that was used. Once the PCB is received, it will be tested in the same way as the circuit layout on the board.

7.7 Motor Integration

Initial set up with the motor should be the easiest part of the whole system. Once the ESP32Servo library is set up, the syntax is the exact same as the regular servo library for other Arduino boards. The proper pins just need to be connected and a proper voltage regulation. The parts the group will be using in this subsystem are as follows:

- **Microcontroller:** Hiletgo ESP-WROOM-32
- **DC Motor:** Towerpro Micro Servo SG90, Feetech DCX 22 S, and DSServo DS3218MG
- **Simple Electrical Components:** resistors, transistors, diodes, etc.

The first step was to experiment with controlling one motor (no IC). For this, we used a cheap motor: DC3V-6V DC drive motor. This was a cheap motor that the team had lying around, and this could be done with any dc motor without IC. Next the group connected the resistor to the base pin of the NPN transistor (this is to protect the ESP 32 Board from overvoltage), then connected it in series to pin D4 on the microcontroller. Next the collector pin was connected to ground, the emitter pin to the positive pin of the general-purpose diode, and the negative pin of the general-purpose diode to the 3.3 volts.

The next step was testing this setup on the Arduino IDE. A simple code was set up to spin the motor. After testing that, the team setup code to control the duration and distance for the motor. After this system is verified to work, the group moves on to the next dc motor set up.

The next DC motor set up, the Feetech 35kg servo motor, comes with two motors. They are mounted in a way that they act like a gimbal using the aluminum mounting brackets that were purchased. Once the two mounting brackets are put on the motors, the pan motor and screw right onto the tilt motor. Next a u bracket can be screwed onto the gear of the tilt motor on one side, and to the screw on bearing on the other side. This does have IC as an internal component so any extra electrical components will not be necessary here. To connect this motor to the ESP 32 microcontroller, we connected the power pin to a separate power source set to 6V, GND to GND, and for the analog connections, we connected the top tilt motor to port 18 and the bottom pan to port 19. Other ports can be used. The same tests, for controlling duration and distance were applied to this system.

The next step is to calibrate the motors. The `.write(angle)` is never accurate for any out of the box servo motor, so one must calibrate the angle themselves. This can be done using the `.writeMicroSeconds` function. The user just has to write to two separate microsecond

designations (for the pulse width modulation) and measure whatever angle the motor went to at said microsecond. The last step is to write the equation for the difference:

Desired angle = (zero position signal) + (target Degree / (MeasuredDeg1 / (TestRunMicroseconds1)))

Each motor needed to be calibrated differently. This was done using a printed out protractor and a needle that was secured onto the motor.

For the receiver side, we have a 360 degree pan motor (actually measured to be 330 degree) and a 180 degree tilt motor (167 degrees max). For the transmitter side, we assembled two 270 degree motors (actual measurement turned out to be 248 degrees in testing).

7.8 Motor, Laser Diode Array, and Quadrant Photodiode Integration

This subsystem combination will be the bulk of the project, regardless of what combination of motors are chosen. The laser Diode Array and the QPD need to first be mounted on their prospective hardware. More plastic and tape will be used to secure these parts to the receiver and transmitter ends.

The following parts will be used in this subsection:

- **Microcontroller:** Hiletgo ESP-WROOM-32
- **DC Motor:** DSServo DS3218MG and Feetech DCX 22 S
- **QPD:** SPOT-4D-0: Annular Quadrant Silicon Photodiode
- **Photodetector:** FGA015 - InGaAs Photodiode
- **Laser diode:** FPL1055T
- **Simple Electrical components:** Resistors, capacitors, OP amps, wires etc

The testing for this subsystem will become the most extensive part. Once everything is mounted, the receiver side (QPD and motors/gimbal) and the transmitter side (laser diode array and motors/gimbal) will be attached to their prospective microcontrollers.

All the code used in the final QPD system will be implemented here. That includes the demodulation, noise analysis, and light saturation to center position analysis. That data will be taken and put into the other algorithm. Said algorithm will determine how to move the motors/gimbals in relation to where the saturation lies in the quadrants for the QPD. The speed and duration that these motors move is highly important as the transmitter is expected to be moving.

Once the QPD is hooked up to our PCBs, the data for each quadrant will go to a separate pin (specifically the I2C). The intensity difference must be calculated. Here are the two equations used to get that data:

$$xDiff = (topRightValue + bottomRightValue) - (topLeftValue + bottomLeftValue)$$

$$yDiff = (topRightValue + topLeftValue) - (bottomRightValue + bottomLeftValue)$$

It was important that we set the limits for the motors as well as the threshold for the amount of degrees the motor can move. This is to remove jitter and get a more stable readout.

Since it is known that the QPD software works and is robust right now, getting accurate tracking when moving is the current top priority. The movement will be tested by moving the transmitter system by hand to different locations in a room while pointing the transmitter towards the receiver. This should also be done at various speeds to test every contingency. It is expected that a lot of tweaking will go into the algorithm for controlling the motors.

7.9 Final System Integration

Once subsystem 8 is completed, all that is left is to integrate the GPS into the system. The GPS allows the receiver system to find the laser diode signal even when none of it hits the QPD. This final integration will include all the researched parts:

- **Microcontroller:** Hiletgo ESP-WROOM-32
- **DC Motor:** DSServo DS3218MG and Feetech DCX 22 S
- **QPD:** SPOT-4D-0: Annular Quadrant Silicon Photodiode
- **Photodetector:** FGA015 - InGaAs Photodiode
- **Laser diode:** FPL1055T
- **GPS Module:** GY-NEO6MV2 NEO-6M GPS
- **IMU Module:** GY-271 QMC5883L
- **Simple Electrical components:** Resistors, capacitors, OP amps, wires etc

Here is where the real testing and final assembly will begin. Once we verify the indoor functionality and the outdoor functionality both function within specification when fully assembled, we can start developing the final rig. This includes a 3D printed mount to go along with the aluminum one. This is needed because we are using large 3D printed gears, two 6:1 gear ratios making a 30:1.

To accomplish that, we are printing an extension of sorts to give room for the gears and QPD housing to rotate. This will be done on the transmitter side as well where the laser diode and led array are. They will both have their own housing that will be 3D printed. The lenses and diffusers will also go inside these 3D printed housing. We then will make a stand that the transmitter and receiver motor gimbal can screw onto. All of our modules and PSUs will be attached to the base of this module for a portable design. The batter will attach to the bottom of this baseplate and a killswitch next to it just to save any components in case of a malfunction.

As far as testing goes, each component will be tested separately again and we will work our way through each subsystem, making sure it works as intended when fully integrated into this rig. Once it's established that everything works within spec, the final tweeking will begin in order to establish proper tracking.

8 Administrative Content

8.1 Budget Estimates and Funding

Our team is sponsored by the Knight Vision Lab (KVL), a research group at CREOL, and they will be providing the budget for this project. Our limit for the project is \$5000. Luckily the KVL will provide certain components, such as the receiver gimbal, so they won't come out of our budget. The table below consists of very rough cost estimates of the most critical components we'll need for the project.

Item	Quantity	Unit Cost	Total Cost	Acquisition Status
SFH 4258S 850nm LED	25	\$1.10	\$27.5	Acquired
FPL1055T 1550nm LD	1	\$424	\$424	Acquired
SPOT-9D-0 Annular QPD	1	\$150	\$150	Acquired
FGA01FC InGaAs PD	1	\$175	\$175	Acquired
Various Optical Lenses	4	\$43	\$172	Acquired
#38-788 Optical Diffuser	1	\$17	\$17	Acquired
87-040 Dichroic Mirror	1	\$176	\$176	Acquired
Gremsy T3V3 Receiver Gimbal	1	\$1800	\$1800	Acquired*
Feiyu G6 Max Beacon Motor	2	\$190	\$280	Ordered
Here3+ GPS Module	1	\$350	\$350	Acquired
Cube Orange+ IMU Module	1	\$350	\$350	Acquired
ESP32 MCU Module	1	\$45	\$45	Acquired

RDX900x-US LoRa Modules	2	\$96	\$192	Acquired
Tattu Batteries	2	\$405	\$810	Acquired
Cost of Major Components			\$3263.06	

8.1: Bill of Major components.

**The Gremsy T3V3 was gifted to us by the KVL group, so it won't come out of our budget*

8.2 Project Milestones

There are several milestones and deadlines we'll have to meet in order to finish the project. Each milestone has been addressed and its planned work time has been roughly established. The tables below roughly encompass these milestones.

Task	Start Date	Anticipated End Date	Duration
Project Brainstorming	Spring Semester	Spring Semester	10 weeks
Project details worked out (some optimizations)	05/13/2024	05/23/2024	1.5 week
Individual Research Delegations and Assessments	08/16/24	09/23/24	1 weeks
Initial Design Document (Based upon the D&C documents)	05/21/24	05/31/24	1.5 weeks
20-Page D&C (improvements made, more research done, and any recommended changes through consultation)	5/31/24	06/7/24	1 week
Meet and Discuss with Sponsor/mentor (seek guidance)	06/7/24	06/21/24	2 weeks
60-Page Milestone (improvements made, more research done, and any recommended changes through consultation)	06/21/24	07/5/24	2 weeks
Group Review: Final Draft (90 pages)	07/5/24	07/23/24	2.5 weeks

Table 8.2.1: Senior Design 1 deadline

Task	Start Date	Anticipated End Date	Duration
Final Selection of components and Ordering the Parts	05/24/24	06/7/24	2 weeks
System Design (designing schematics for PCB, filters, etc.)	05/27/24	06/17/24	3 weeks
PCB Design	06/07/24	06/17/24	2.5 weeks
Testing components, arrays, and filters	06/17/24	07/5/24	2.5 weeks
PCB and MCU Testing	06/17/24	07/5/24	2.5 weeks
Develop prototypes for Mini Demo Video	07/5/24	07/23/24	2.5 weeks

Table 8.2.2 Senior Design I Project Design Milestones

Task	Start Date	Anticipated End Date	Duration
PCB and Other Design Testing (improving on prototype)	08/21/24	09/11/24	3 weeks
Assembling Parts/System Integration for final prototype	09/11/24	09/23/24	3 weeks
Prototype testing for final product	09/23/24	11/15/24	7 weeks
Finalize the Documentation	11/1/24	11/15/24	2 weeks
Practice Final Presentation	11/15/24	11/22/24	1 weeks
Final Presentation	TBA	TBA	TBA

Table 8.2.3 Senior Design II Project Design & Documentation Milestones

8.3 Work Distributions

There are three members of our team, each from a different discipline. The PSE, Seth, will handle the design and assembly of all optical components in the system. This includes the lens systems as well as the selection of the laser, LEDs and photodetectors

based on divergence and output optical power. As the team leader, Seth will also be responsible for the administrative content and relaying information between the team and the sponsor. The EE, Raymond, will be in charge of designing the necessary power supply and integrating the GPS, IMU and LoRa modules into usable PCBs for both the receiver and beacon. Other responsibilities will be to support Blake with software implementation and Seth in properly driving the diodes. The CPE, Blake, will be responsible for most of the software work required for the project. This includes demodulating the signals from the QDP, as well as using those signals to give directions to the gimbal.

9 Conclusion

This project set out with the ambitious goal of developing an optical beacon tracker for optical wireless communication. Our primary objective was to create a system capable of locating and tracking an optical beacon, which would be achieved through the integration of a Quadrant Photodiode (QPD) and an InGaAs photodiode. Both sensors would be mounted on a gimbal, with the QPD detecting the incident light and the InGaAs photodiode confirming alignment. The beacon, emitting light at 850nm and 1550nm, would be mobile and capable of interacting with the tracking system, ensuring a dynamic and robust communication link.

Throughout the development process, we have and will encounter numerous challenges and learning opportunities. From the initial conceptualization and research phases to the detailed design and integration of optical, electrical, and software components, each step brought us closer to realizing our vision. The project's success hinged on the meticulous selection of components, the strategic planning of our design methods, and the rigorous testing of our system's performance.

We achieved several key milestones throughout this project, each have and will mark significant progress towards our end goal:

Design and Prototyping: We successfully designed and prototyped the transmitter and receiver systems, ensuring that the beacon emitted light at the specified wavelengths and that the receiver could effectively detect and track the signal.

Component Selection: Extensive research has led to the selection of suitable components, including the ones we have experimented with in hand: InGaAs photodiode, Si annular quadrant photodiode, ESP32 microcontroller, and gimbal system. Each component was chosen for its compatibility, performance, and cost-effectiveness.

Software Development: We developed robust software, as well as experimenting with other code, to process the signals from the QPD, demodulate them, denoise them, and control the gimbal and dc motors for accurate tracking. The Arduino code also integrated GPS data for initial alignment and utilized the built in IMU for stability during tracking.

System Integration: The integration of optical, electrical, and mechanical subsystems will be a critical milestone. Ensuring seamless communication between the components and achieving stable and accurate tracking will require iterative testing and refinement.

Testing and Validation: We are conducting comprehensive testing to validate the system's performance. This includes bench tests for individual components, as well as field tests to assess the overall functionality and reliability of the tracking system.

As far as lessons learned, throughout this project, we are encountering many challenges that provided valuable lessons such as:

Interference and Signal Clarity: One of the primary challenges was dealing with interference and ensuring the clarity of the signal detected by the QPD. This required fine-tuning the signal processing algorithms and implementing filters to enhance signal quality.

Precision and Stability: Achieving the required precision of the DC motor and the precision and stability of the gimbal system will be another significant challenge. We are learning the importance of calibrating the IMU and refining the control algorithms to maintain accurate tracking even in dynamic environments.

System Integration Complexity: Integrating multiple subsystems with varying specifications and ensuring seamless communication is already posing a problem. This was already known going into the project that connected subsystems, creating a more complex system, would make debugging issues more difficult. This taught us the importance of thorough planning, modular design, and iterative testing to identify and resolve integration issues.

Project Management: Managing the project timeline, budget, and workload distribution will be crucial to our success. We learned to prioritize tasks, allocate resources effectively, and maintain clear communication within the team to ensure timely progress.

While we are achieving our current primary objectives, there are several areas in our approach that leave room for future improvement and expansion:

Enhanced Signal Processing: Since the group has only developed a basic algorithm for processing the signal from the photodiodes, further refining the signal processing algorithms to improve noise reduction and signal clarity would enhance the system's performance, especially in challenging environments.

Advanced Gimbal and DC Motor Control: Implementing more advanced control techniques, such as adaptive control, could further improve the precision and stability of the gimbal system.

Extended Range and Mobility: Expanding the range and mobility of both the beacon and the tracker would enable more versatile applications. This could involve

incorporating more powerful emitters and sensors, as well as enhancing the mobility and robustness of the systems.

Robust Testing: Conducting more extensive and varied testing, including in different environmental conditions and with varying levels of interference, would provide a deeper understanding of the system's capabilities and limitations.'

The development of this optical beacon tracker will have significant implications for the field of optical wireless communication. Potential applications include:

Telecommunications: Enhancing optical communication systems for long-distance data transmission with high precision and reliability.

Autonomous Systems: Improving navigation and tracking capabilities for autonomous vehicles and drones, particularly in GPS-denied environments.

Industrial Automation: Enabling precise alignment and tracking for automated manufacturing and inspection systems, improving efficiency and accuracy.

Research and Exploration: Providing robust tracking solutions for scientific research, environmental monitoring, and space exploration missions, where reliable communication and alignment are critical.

In conclusion, this project will successfully demonstrate the feasibility and potential of an optical beacon tracker for optical wireless communication. Through meticulous design, rigorous testing, and innovative problem-solving, we are developing a system capable of accurately locating and tracking an optical beacon. The lessons we are learning and the achievements made throughout this project are laying a strong foundation for future advancements and applications in this exciting field. We look forward to further refining and expanding our system, exploring new applications, and contributing to the ongoing development of optical wireless communication technologies.

Appendix A: Citations

- [1] Tracking control for Free-Space optical communication of unmanned aerial vehicle. (2018, December 1). IEEE Conference Publication | IEEE Xplore.
<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8644959>
- [2] A linear closed loop feedback system for beam wander correction in Medium-Range Optical link. (2023, December 14). IEEE Conference Publication | IEEE Xplore.
<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10450056>
- [3] P. Kumar, S. Sonkar, A. K. Ghosh and D. Philip, "Real-time vision-based tracking of a moving terrain target from Light Weight Fixed Wing UAV using gimbal control," *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, Prague, Czech Republic, 2020, pp. 154-159, doi: 10.1109/CoDIT49905.2020.9263896
- [4] Uysal, Murat & Capsoni, C. & Boucouvalas, Anthony & Udvary, Eszter. (2017). Optical Wireless Communications – An Emerging Technology.
10.1007/978-3-319-30201-0.
- [5] Pololu - 50:1 Metal Gearmotor 37Dx70L mm 12V with 64 CPR Encoder (Helical Pinion). (n.d.). <https://www.pololu.com/product/4753>
- [6] 60 W (1/12 HP) BMU Series Brushless DC Motor Speed Control Systems. (n.d.). Oriental Motor USA.
<https://catalog.orientalmotor.com/viewitems/bmu-series-brushless-dc-motor-speed-control/60w-bmu-series-brushless-dc-motors>
- [7] Maxon 500 (n.d.).
<https://www.maxongroup.com/maxon/view/product/motor/dcmotor/DCX/DCX22/DCX22L01GBSL450>
- [8] Series 3242 . . . BX4. (n.d.). Dr. Fritz Faulhaber GmbH & Co. KG.
<https://www.faulhaber.com/en/products/series/3242bx4/>
- [9] Gremsy T3V3 Spec - Gremsy. (2022, July 11). <https://gremsy.com/gremsy-t3v3-spec>
- [10] Support for DJI OM 4 SE - DJI. (n.d.). DJI.
<https://www.dji.com/support/product/om-4-se>
- [11] Holybro Pixhawk 4 | PX4 Guide (main). (n.d.).
https://docs.px4.io/main/en/flight_controller/pixhawk4.html
- [12] Durandal. (n.d.). Holybro Store. <https://holybro.com/products/durandal>

- [13] DROTEK SAS. (2023, July 8). SIRIUS RTK GNSS ROVER (F9P) - DROTEK ELECTRONICS. <https://store-drotek.com/911-sirius-rtk-gnss-rover-f9p.html>
- [14] MPU-9250 | TDK InvenSense. (2023, November 3). TDK InvenSense. <https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/>
- [15] ADS-B IN Carrier Board | CubePilot. (n.d.). <https://docs.cubepilot.org/user-guides/carrier-boards/ads-b-carrier-board>
- [16] Emlid Systems. (2024, March 7). Reach M2 RTK GNSS Receiver & GPS Module | EMLID. Emlid. <https://emlid.com/reach/>
- [17] EVK-F9P. (2023, November 8). U-blox. <https://www.u-blox.com/en/product/evk-f9p?legacy=Current#Documentation-&-resources>
- [18] ArduSimple. (2024, June 14). simpleRTK2B Budget (u-blox ZED-F9P) multiband RTK GPS board. <https://www.ardusimple.com/product/simplertk2b/>
- [19] Tersus GNSS. (n.d.). BX306 GNSS OEM Board | Tersus GNSS. <https://www.tersus-gnss.com/product/bx306-oem-board>
- [20] IR-LOCK. (n.d.). Here+ RTK Base (M8P). <https://irlock.com/collections/pixhawk2/products/here-base-m8p>
- [21] IR-LOCK. (n.d.-b). Here3 GNSS (M8P) with iStand. <https://irlock.com/collections/pixhawk2/products/here3-gnss-m8p>
- [22] Laser safety facts <https://www.lasersafetyfacts.com/>
- [23] Cube Orange+ and PX4 Power Consumption https://docs.px4.io/main/en/flight_controller/cubepilot_cube_orangeplus.html#
- [24] Aurelia X6 Standard Specs <https://aurelia-aerospace.com/product/aurelia-x6-standard/>
- [25] Gremsey T3V3 Specs <https://gremsy.com/gremsy-t3v3-spec>
- [26] Laser Diodes Specs <https://www.mouser.com/ProductDetail/ams-OSRAM/SFH-4258S?qs=3W28jC%2BmlM3MO3EHwLOW8Q%3D%3D>
- [27] Here 4 Manual <https://docs.cubepilot.org/user-guides/here-4/here-4-manual#overview>

- [28] Orange+ Manual
<https://docs.cubepilot.org/user-guides/autopilot/the-cube-module-overview>
- [29] Ewald, W. (Wolle). (2024, June 13). wollewald/BH1750_WE. GitHub.
https://github.com/wollewald/BH1750_WE
- [30] RobTillaart. (n.d.). *Robtillaart/TSL235R: Arduino Library for the TSL235R light to Frequency Convertor*. GitHub. <https://github.com/RobTillaart/TSL235R>
- [31] RobTillaart. (n.d.). *Robtillaart/TSL235R: Arduino Library for the TSL235R light to Frequency Convertor*. GitHub. <https://github.com/RobTillaart/TSL235R>
- [32] *Configuring a telemetry radio using mission planner*¶. Configuring a Telemetry Radio using Mission Planner - Copter documentation. (n.d.).
<https://ardupilot.org/copter/docs/common-configuring-a-telemetry-radio-using-mission-planner.html#common-configuring-a-telemetry-radio-using-mission-planner>
- [33] Docs.arduino.cc. (n.d.). <https://docs.arduino.cc/learn/microcontrollers/analog-output/>
- [34] OphirBlog. (2018, October 16). *How to measure pulsed laser beams with a photodiode sensor*. Ophir Photonics.
<https://blog.ophiropt.com/how-to-measure-pulsed-laser-beams-with-a-photodiode-sensor/>
- [35] Demodulation for communications simulation - MATLAB. (n.d.).
<https://www.mathworks.com/help/signal/ref/demod.html>
- [36] An-1173 quadrant photodiode circuitry for high ... (n.d.-a).
<https://www.renesas.com/us/en/document/apn/1173-quadrant-photodiode-circuitry-high-precision-displacement-measurement>
- [37] ISO standards
<https://webstore.ansi.org/>
- [38] Saleh, B.E.A. and Teich, M.C., "*Fundamentals of Photonics*," 2nd Edition, Wiley-Interscience, 2007
- [39] ams OSRAM SFH 4258S data sheet
https://www.mouser.com/datasheet/2/588/prd_pim_datasheet_2220224_EN_pdf-3419442.pdf
- [40] Siegman, A.E., "*Lasers*," University Science Books, 1986
- [41] Output divergence of a single mode fiber.
<https://support.photonengr.com/article/235-single-mode-fiber-output-coherent-source>

[42] Smith, W.J., *"Modern Optical Engineering,"* 4th Edition, McGraw-Hill, 2007

□

Appendix B: LLM Prompts and Responses

Prompt 1: I would like to construct an optical beacon tracker using two drones. One will operate as a beacon with an array of laser diodes and the other a tracker outfitted with a quadrant photodiode to follow the beacon using a 2-axis gimbal.

[GPT] Constructing an optical beacon tracker using two drones involves several key components and steps. Here's an overview of the process:

Components Needed

1. **Beacon Drone:**
 - Laser Diodes: An array of laser diodes to serve as the optical beacon.
 - Power Supply: Adequate power source to run the laser diodes.
2. **Tracker Drone:**
 - Quadrant Photodiode: To detect the position of the laser beam.
 - 2-Axis Gimbal: To adjust the orientation of the photodiode to follow the beacon.
 - Microcontroller or Processor: To process the photodiode signals and control the gimbal.
 - Power Supply: To run the electronics and gimbal.
3. **Communication Modules:**
 - Wireless communication modules (e.g., RF modules) to transmit data between the two drones if necessary.
4. **Software:**
 - Control algorithms for tracking and gimbal control.
 - Signal processing algorithms to interpret the photodiode data.

Steps to Construct the System

1. **Setup the Beacon Drone:**
 - Arrange the laser diodes in a pattern that ensures visibility from various angles.
 - Connect the diodes to a power source and a control circuit if you need to modulate the light (e.g., for encoding information).
2. **Setup the Tracker Drone:**
 - Mount the quadrant photodiode on the 2-axis gimbal.
 - Connect the photodiode to a signal processing circuit that can convert the analog signals into digital signals for the microcontroller.
 - Mount the gimbal on the drone and ensure it has a stable power source.
3. **Programming the Tracker:**

- Write a control algorithm for the microcontroller to interpret the signals from the quadrant photodiode. The algorithm should calculate the position of the laser beam on the photodiode and adjust the gimbal to center the beam.
 - Implement PID (Proportional-Integral-Derivative) control or another suitable control method to ensure smooth and accurate tracking.
 - If needed, program the drones to communicate wirelessly for additional coordination.
4. **Integration and Testing:**
 - Assemble the hardware and ensure all connections are secure.
 - Test the beacon and tracker systems separately to ensure they function correctly.
 - Conduct field tests with both drones to fine-tune the tracking algorithm and ensure reliable performance.
 5. **Calibration and Optimization:**
 - Calibrate the photodiode to ensure accurate position detection.
 - Adjust the sensitivity and response time of the gimbal control to optimize tracking performance.
 - Conduct tests in various lighting conditions to ensure the system performs well in different environments.

Considerations

- **Safety:** Ensure the laser diodes are eye-safe or implement safety measures to avoid direct eye exposure.
- **Environmental Factors:** Consider how environmental conditions (e.g., sunlight, obstacles) may affect the tracking performance and plan accordingly.
- **Redundancy:** Implement fail-safes and redundancy in critical components to enhance reliability.

Example Implementation

1. **Beacon Drone:**
 - Use multiple laser diodes in a circular arrangement.
 - Control the diodes with a microcontroller to turn them on/off as needed.
2. **Tracker Drone:**
 - Mount the quadrant photodiode on a gimbal.
 - Connect the photodiode to an ADC (Analog-to-Digital Converter) and then to a microcontroller.
 - Implement a control loop in the microcontroller to adjust the gimbal based on the photodiode readings.
3. **Communication:**
 - Use an RF module to transmit the beacon's status to the tracker if needed.

By following these steps, you can construct an optical beacon tracker using two drones. Ensure to test and iterate on your design to achieve the best performance.

[GEM] Your concept for an optical beacon tracker using two drones is a very interesting approach! Here's a breakdown of the components and some considerations for building it:

Drone 1: Beacon

- **Laser Diode Array:** Choose a laser diode array with a specific wavelength suitable for your photodiode and with enough power to be detectable at the desired range. Visible or near-infrared (NIR) lasers are common choices.
- **Collimator:** A collimator lens will be needed to focus the diverging laser beams from the array into a more collimated beam for better detection range.
- **Control System:** A microcontroller can be used to control the laser array, potentially pulsing or modulating the laser light for easier tracking by the receiver drone.

Drone 2: Tracker

- **Quadrant Photodiode:** This photodiode has four sections that allow you to determine the direction of the incoming light based on the intensity difference between the sections.
- **2-Axis Gimbal:** A gimbal will be crucial for precisely pointing the photodiode towards the beacon. The control system will adjust the gimbal based on the photodiode's output.
- **Control System:** This system will process the signal from the quadrant photodiode and determine the direction error. It will then send control signals to the gimbal motors to keep the beacon centered within the photodiode's field of view.

Additional Considerations:

- **Calibration:** Carefully calibrate the photodiode and its alignment with the gimbal's axes.
- **Software:** Develop software algorithms for the control systems on both drones. The beacon drone's software might simply control the laser pulsing. The tracker drone's software will need to interpret the quadrant photodiode signal, calculate direction error, and translate it into gimbal control commands.
- **Safety:** Eye safety is crucial when dealing with lasers. Choose a laser wavelength and power level that complies with safety regulations. Never point the laser beam towards people or aircraft.
- **Communication:** While not essential, consider a communication link between the drones. This could allow for data exchange or provide a backup for tracking in case the photodiode signal is lost momentarily.

Resources:

There's limited information readily available for building a specific optical beacon tracker with drones. However, you can find resources related to the individual components:

- **Laser Diodes and Drivers:** [Online retailers specializing in electronic components]

- Collimators: [Online retailers specializing in optics]
- Quadrant Photodiodes: Same as Laser Diodes
- Gimbal Control Systems: Several companies offer ready-made gimbal control systems designed for drone applications.

Building this project will require knowledge of electronics, optics, control systems, and potentially some software development. If you're new to these areas, consider breaking it down into smaller, more manageable projects to gain the necessary skills before tackling the entire system.

Prompt 2: I would like to implement a GPS module and IMU to transmit location and orientation of the beacon to the tracker to preemptively align the gimbal with the beacon before the quadrant diode has read a signal from the laser diodes. How can I do this?

[GPT] Integrating a GPS module and an IMU (Inertial Measurement Unit) into your beacon and tracker drones can significantly enhance the tracking system by providing location and orientation data. This allows the tracker to preemptively align its gimbal towards the beacon, improving responsiveness and accuracy. Here's how you can implement this:

Components Needed

1. Beacon Drone:

- GPS Module: To provide the beacon's location.
- IMU: To provide orientation data (pitch, roll, yaw).
- Communication Module: To transmit GPS and IMU data to the tracker drone.

2. Tracker Drone:

- GPS Module: To provide the tracker's location.
- IMU: To provide the tracker's orientation.
- Communication Module: To receive GPS and IMU data from the beacon drone.
- Quadrant Photodiode: To fine-tune the tracking once the beacon is within range.
- 2-Axis Gimbal: To orient the photodiode.
- Microcontroller or Processor: To process the GPS, IMU, and photodiode data.

Steps to Implement the System

1. Setup GPS and IMU on Beacon Drone:

- Connect the GPS and IMU to the microcontroller.
- Ensure the GPS module has a clear view of the sky for accurate positioning.
- Calibrate the IMU for accurate orientation data.

2. Setup GPS and IMU on Tracker Drone:

- Connect the GPS and IMU to the microcontroller.

- Calibrate the IMU on the tracker drone as well.
- 3. **Establish Communication:**
 - Use RF modules (e.g., XBee, LoRa) or any other suitable wireless communication module to transmit the beacon's GPS and IMU data to the tracker.
 - Ensure reliable and low-latency communication between the two drones.
- 4. **Processing GPS and IMU Data:**
 - On the tracker drone, use the received GPS data to calculate the relative position of the beacon drone.
 - Use the IMU data from both drones to determine the relative orientation.
 - Implement an algorithm in the microcontroller to convert the relative position and orientation data into gimbal angles.
- 5. **Preemptive Gimbal Alignment:**
 - Before the quadrant photodiode detects the laser signal, use the calculated gimbal angles to orient the gimbal towards the predicted position of the beacon.
 - Once the quadrant photodiode detects the laser signal, switch to fine-tuning mode to accurately track the beacon.
- 6. **Integration and Testing:**
 - Integrate all components and ensure secure connections.
 - Test the system in a controlled environment to ensure that the GPS and IMU data correctly pre-align the gimbal.
 - Conduct field tests to fine-tune the algorithm and ensure reliable performance.

Example Implementation

1. Beacon Drone:

- GPS Module: e.g., u-blox NEO-M8N.
- IMU: e.g., MPU-9250.
- Microcontroller: e.g., Arduino, STM32.
- Communication Module: e.g., XBee.

```
// Pseudocode for beacon drone
void setup() {
  initGPS();
  initIMU();
  initCommunicationModule();
}
void loop() {
  GPSTData gpsData = readGPS();
  IMUData imuData = readIMU();
  sendDataToTracker(gpsData, imuData);
  delay(100); // Adjust based on required update rate
}
```

2. Tracker Drone:

- GPS Module: e.g., u-blox NEO-M8N.
- IMU: e.g., MPU-9250.
- Microcontroller: e.g., Arduino, STM32.
- Communication Module: e.g., XBee.
- Quadrant Photodiode: e.g., Hamamatsu S5981.
- 2-Axis Gimbal: Servo motors or brushless gimbal.

** Pseudocode for tracker drones also provided.

Considerations

- **Latency:** Ensure that the communication between the drones is fast enough to make real-time adjustments.
- **Synchronization:** Synchronize the clocks of both drones to ensure accurate time-stamping of GPS and IMU data.
- **Error Handling:** Implement error handling for scenarios where GPS or IMU data is unavailable or unreliable.

By following these steps, you can enhance your optical beacon tracker with preemptive alignment using GPS and IMU data, ensuring more responsive and accurate tracking.

[GEM] That's a great improvement to the system! Integrating a GPS module and IMU (Inertial Measurement Unit) on the beacon drone can significantly enhance the overall tracking performance. Here's how it can work:

Beacon Drone with GPS/IMU:

- In addition to the previously mentioned laser diode array and collimator, you'll equip the beacon drone with a GPS module and an IMU.
- The GPS module will provide the beacon drone's location (latitude, longitude, altitude).
- The IMU, typically consisting of an accelerometer, gyroscope, and magnetometer, will provide the beacon drone's orientation data (pitch, roll, yaw).

Data Transmission:

- Choose a telemetry system to transmit the GPS location and IMU data from the beacon drone to the tracker drone. Options include radio modules (e.g., UHF/WiFi) or cellular depending on your range and data rate requirements.

Tracker Drone with Gimbal Control System:

- The tracker drone will receive the GPS location and orientation data of the beacon drone through the telemetry link.
- The tracker drone's control system will use this information to pre-emptively aim the gimbal in the general direction of the beacon. This can significantly reduce the time it takes for the quadrant photodiode to acquire the beacon's signal and improve tracking accuracy.

Control System Improvements:

- The tracker drone's software will need to be modified to incorporate the received GPS and IMU data. It will calculate the relative position and orientation between the two drones based on this data.
- Based on the calculated relative position, the software will determine the necessary adjustments for the gimbal to point towards the beacon's location.

Benefits:

- **Faster Acquisition:** Pre-positioning the gimbal based on GPS/IMU data significantly reduces the time it takes for the quadrant photodiode to acquire the beacon's signal, especially during initial lock or after temporary signal loss.
- **Improved Accuracy:** The initial pre-alignment by the gimbal improves the overall tracking accuracy, particularly when dealing with smaller beacon signals or longer tracking distances.

Additional Considerations:

- **Data Latency:** Factor in the latency introduced by the telemetry link between the transmission from the beacon and reception by the tracker. Account for this delay when calculating the gimbal pre-positioning.
- **Sensor Accuracy:** The accuracy of the pre-alignment will depend on the accuracy of the GPS and IMU data. GPS accuracy can be affected by factors like satellite visibility and atmospheric conditions. IMU readings might experience drift over time, requiring calibration procedures.

Overall, incorporating a GPS/IMU system on the beacon drone is a valuable addition to your optical beacon tracker project. It improves tracking performance by allowing the tracker to anticipate the beacon's location and pre-position the gimbal for faster and more accurate acquisition.

Prompt 3: When driving the laser diode array should power be routed through the MCU circuit board to allow modulation or is it possible to modulate the laser diodes being fed from a 22.8V, 23 Ah battery?

[GPT] When driving a laser diode array, there are a few considerations to ensure proper modulation and power management. Here's an outline of how you can approach this:

Direct Modulation of Laser Diodes

1. **Microcontroller (MCU) Control:**
 - The MCU can directly modulate the laser diodes by switching them on and off or varying their intensity.
 - This requires routing the power through the MCU circuit board, which allows precise control over the diodes.
2. **Power Considerations:**

- Laser diodes typically require a constant current source to operate reliably. They are sensitive to current variations and can be damaged if overdriven.
 - The MCU will likely need to control a driver circuit (such as a constant current driver) rather than powering the diodes directly, especially if the power requirements exceed the capabilities of the MCU's GPIO pins.
3. **Switching Components:**
- Use switching components like MOSFETs or transistors controlled by the MCU to handle higher power levels. The MCU will send a control signal to the gate/base of the MOSFET/transistor, which in turn will modulate the laser diode current.
 - Ensure the switching components can handle the voltage and current requirements of the laser diodes.

Using a High-Voltage Battery

1. **Voltage Regulation:**
- The 22.8V, 23 Ah battery is a high voltage source. Laser diodes typically require lower voltages (e.g., 3V to 5V). You'll need a voltage regulator or DC-DC converter to step down the voltage to the appropriate level for the laser diodes.
 - A buck converter can be used to efficiently step down the voltage from 22.8V to the required voltage for the laser diodes.
2. **Current Control:**
- Implement a constant current driver to ensure the laser diodes receive a stable current. This is crucial for the longevity and performance of the diodes.
 - The current driver can be modulated by the MCU using PWM (Pulse Width Modulation) or a digital control signal.

Example Circuit Design

1. **Power Supply and Regulation:**
- Use a buck converter to step down 22.8V to the required voltage (e.g., 5V).
 - Ensure the buck converter can handle the power requirements of the entire laser diode array.
2. **Laser Diode Driver:**
- Use a constant current driver circuit to supply a stable current to the laser diodes.
 - Control the current driver with a MOSFET or transistor, which is modulated by the MCU.
3. **MCU Modulation:**
- The MCU outputs a PWM signal or a digital on/off signal.
 - This signal drives the gate of a MOSFET, which in turn controls the current driver supplying the laser diodes.

Example Schematic

1. **Buck Converter:**
 - Input: 22.8V from the battery.
 - Output: 5V (or suitable voltage for laser diodes).
2. **Constant Current Driver:**
 - Input: 5V from the buck converter.
 - Output: Constant current to the laser diodes.
3. **MCU Control:**
 - PWM output from the MCU to the gate of an N-channel MOSFET.
 - Source of the MOSFET connected to the ground.
 - Drain of the MOSFET connected to the current driver.

[GEM] Modulating the laser diode array directly with the battery voltage (22.8V) is not recommended. Here's why:

- **Laser Diode Damage:** Laser diodes are very sensitive to current fluctuations. Directly connecting them to a battery could lead to uncontrolled current surges that can damage the diodes.
- **Limited Modulation Capability:** Batteries typically provide a constant voltage level. You wouldn't be able to modulate the laser light (e.g., turning it on and off or varying its intensity) for potential ranging or communication purposes.

Power Routing with MCU for Modulation:

- **Driver Circuit:** A dedicated laser diode driver circuit is necessary. This circuit will take a control signal from the MCU and convert it into the appropriate current profile required by the laser diode array. The MCU can modulate this control signal to achieve the desired laser output.
- **Power Supply:** The driver circuit will likely require its own separate power supply with a lower voltage than the battery (e.g., 5V or 12V) for proper operation. This power supply can be regulated from the main battery voltage.
- **MCU Control:** The MCU can generate a control signal based on a pre-programmed pattern or in response to external commands. This allows for functionalities like turning the laser on and off, pulsing it at specific frequencies, or even transmitting data by modulating the laser intensity.

Benefits of MCU Control:

- **Precise Modulation:** The MCU enables precise control over the laser diode array's output, allowing for features like pulsed operation or data transmission.
- **Safety Features:** The driver circuit can incorporate safety features like current limiting and temperature monitoring to protect the laser diodes.

In conclusion, routing the power through the MCU circuit board with a dedicated driver circuit is the recommended approach for driving your laser diode array. This method

offers precise control, modulation capabilities, and safety features for your laser beacon project.

Prompt 4: What is the best way to relay positional data from one drone to another using a ground control station in-between? We plan on using an orange+ with here4 GPS and telemetry module.

[GPT] Relaying positional data from one drone to another using a ground control station (GCS) involves setting up a communication link between the drones and the GCS, which acts as an intermediary. Here's a step-by-step guide to achieving this with your setup involving the Orange Cube+ flight controller, Here4 GPS, and telemetry modules:

1. Hardware Setup

Drones

- **Flight Controllers:** Both drones should have the Orange Cube+ flight controller installed.
- **GPS Modules:** Use Here4 GPS modules on both drones for accurate positional data.
- **Telemetry Modules:** Equip both drones with compatible telemetry modules (e.g., SiK radio or RFD900).

Ground Control Station (GCS)

- **GCS Software:** Use a robust GCS software like Mission Planner, QGroundControl, or MAVProxy.
- **Telemetry Module:** Use a telemetry module compatible with the ones on the drones to communicate with both drones simultaneously.

2. Communication Setup

Telemetry Link

- **Frequency and Channel:** Ensure that the telemetry modules on both drones and the GCS are set to the same frequency and channel to establish communication.
- **Baud Rate:** Configure the telemetry baud rate to match across all modules (common baud rates are 57600 or 115200).

MAVLink Protocol

- **MAVLink Configuration:** Configure the drones and GCS to use the MAVLink protocol for communication. This is typically the default protocol used by most flight controllers and GCS software.

3. Software Configuration

GCS Configuration

- **Multiple Vehicle Support:** Ensure your GCS software supports multiple vehicles. Both Mission Planner and QGroundControl can handle multiple drones.
- **Vehicle Identification:** Assign unique system IDs to each drone to differentiate between them.

Positional Data Relay

- **Data Request:** Configure the GCS to request positional data from the source drone (Drone A).
- **Data Relay:** Set up the GCS to forward the received positional data to the target drone (Drone B).

4. Implementation Steps

Drone Configuration

1. **Assign System IDs:** Ensure that each drone has a unique MAVLink system ID.
 - Connect to each drone via the GCS and set the system ID in the flight controller parameters.
 - Parameter: `SYSID_THISMAV`
2. **Telemetry Configuration:** Verify that telemetry settings are properly configured for both drones.
 - Parameters: `SERIALx_BAUD` (set appropriate baud rate),
`SERIALx_PROTOCOL` (set to MAVLink)

GCS Configuration

1. **Connect to Both Drones:** Establish telemetry links to both drones.
 - Open the GCS software and connect to both drones using their respective telemetry links.
 - Ensure that the GCS can receive and display positional data for both drones.
2. **Script for Data Relay:** Implement a script or use GCS features to relay data from Drone A to Drone B.
 - **Mission Planner:** Use the Mission Planner scripting feature (Python) to write a script that reads the GPS data from Drone A and sends it to Drone B.
 - **QGroundControl:** Use QGroundControl's MAVLink forwarding feature or write a custom MAVLink message handler if needed.

5. Example Python Script for Mission Planner

Here's a basic example of how you might script this in Mission Planner using Python:

```
python
```

Copy code

```
import time

from pymavlink import mavutil


# Connect to the GCS
master = mavutil.mavlink_connection('tcp:127.0.0.1:5760')


# Function to forward GPS data from one drone to another
def relay_gps_data():
    while True:
        # Receive GPS data from Drone A
        msg = master.recv_match(type='GPS_RAW_INT',
                                blocking=True)
        if msg:
            gps_data = msg.to_dict()
            print(f"Received GPS data from Drone A:
{gps_data}")

            # Create a new GPS_RAW_INT message for Drone B
            gps_msg =
mavutil.mavlink.MAVLink_gps_raw_int_message(
                gps_data['time_usec'],
                gps_data['fix_type'],
                gps_data['lat'],
                gps_data['lon'],
```

```

        gps_data['alt'],
        gps_data['eph'],
        gps_data['epv'],
        gps_data['vel'],
        gps_data['cog'],
        gps_data['satellites_visible']
    )

    # Send the GPS data to Drone B
    master.mav.send(gps_msg)

    print(f"Sent GPS data to Drone B: {gps_data}")

    # Sleep for a short period to avoid excessive CPU
usage
    time.sleep(1)

# Start relaying GPS data
relay_gps_data()

```

6. Testing and Validation

- **Test Individual Links:** Test the telemetry link between each drone and the GCS separately to ensure proper communication.
- **Test Data Relay:** Run the script and verify that positional data from Drone A is correctly relayed to Drone B.
- **Field Test:** Perform a field test with both drones to ensure reliable data relay under real-world conditions.

By following these steps, you can successfully relay positional data from one drone to another using a ground control station as an intermediary. This setup ensures accurate and reliable communication between the drones, enabling advanced coordination and functionality.

[GEM] There isn't a direct way for the Cube Orange+ with Here4 GPS and telemetry module to relay positional data between drones itself. The Cube Orange+ is designed for single drone communication with a ground control station.

However, you can achieve your goal of sharing positional data between drones using the ground control station as a relay with some additional software configuration. Here's how:

Software on the Ground Control Station (GCS):

- You'll need software on your GCS that supports MAVLink (Micro Air Vehicle Link) protocol. This is a common drone communication protocol that the Cube Orange+ uses. Popular open-source options include Mission Planner or QGroundControl. <https://mavlink.io/en/>
- The software needs to be configured to receive telemetry data (including position) from each drone via their respective radio links.
- You'll then need to configure MAVLink message forwarding within the GCS software. This will allow the GCS to receive positional data from one drone and re-transmit it to the other drone via their respective radio links.

Additional Considerations:

- This method will introduce some latency (delay) in the data transmission as it needs to travel through the GCS first.
- Ensure the GCS has a reliable communication link with both drones to minimize data dropouts.
- The specific configuration steps will depend on the software you're using on the GCS. Consult the software's documentation for detailed instructions on MAVLink message forwarding.

Alternative (if applicable):

- If your application allows it, some drone autopilots support MAVLink companion computer functionality. This allows a separate computer (like a Raspberry Pi) to be mounted on the drone and communicate with the autopilot using MAVLink. You could potentially configure this companion computer to receive positional data from the autopilot and then transmit it directly to another drone via a separate radio link, bypassing the GCS altogether. However, this solution requires additional hardware and software configuration on the drones themselves.