Radio Frequency Triangulation

Joshua Broyles
Kathleen Hutchinson
Jack Parkinson
Brandon Stokes

404 FINAL REPORT

CONCEPT OF OPERATIONS

Table of Contents

List of Tables

List of Figures

1. Executive Summary

Radio Frequency (RF) Triangulation enables users to track a transmitter broadcasting at a known frequency in a small (100m) area. Valuable information can be obtained from the precise location of objects. This concept can be used in applications ranging from use in warfare tracking of enemies all the way to tracking animals in the wild. The RF Triangulation begins by turning the transmitter on and placing it away from the receivers. Once the transmitter is activated the receivers can receive the start command from the graphical user interface (GUI) through wifi hotspot and begin scanning the area, starting at north, in ninety degree increments, looking for the strongest signal strength value (RSSI). After the strongest signal direction is determined, a more in depth scan is performed around that pole to find the angle from north the signal is coming from. This angle is then sent to the GUI which takes the gps coordinates of the receivers and their angle that they think the signal is coming from. The GUI can then use this information to triangulate the position of the transmitter.

2. Introduction

Biologists have been looking for a way to monitor wildlife migration patterns without getting directly or indirectly involved during observation. Radio frequency triangulation will allow researchers to monitor animals who have had these transmitters placed on them. With the push of a button researchers can track animals from great distances with the right antenna and time. With the creation of this product, tracking with ease will benefit researchers everywhere.

2.1. Background

This type of direction finding (DF) system was first developed shortly after World War I with the Watson-Watt Technique. This system was made using five nodes, a North, South, East, West, and at the center an Omnidirectional. From this, there were a total of three outputs of north minus south, east minus west, and omnidirectional. The math computations were done on the difference in time received on each of the nodes. This technique was especially useful when finding large signals from great distances. We are enhancing this idea by making a three antenna system that finds the signal strength angle from north a mobile unit that can be placed where needed. With this the antennas can be arranged in a number of ways and places to cover different areas as needed.

2.2. Overview

The system contains four total antennas, one transmitter and three receivers. While each

antenna has its own microcontroller, the transmitter only transmits a set signal. By using a step motor, the receivers will sweep three hundred sixty degrees, searching for the strongest signal from the transmitter. Once found, the angle from a set, true north will be sent to the database through the Pycom device through a WiFi hotspot. Along with the angle, the GPS coordinates of the receiver will be sent as well. The three angles received from the database to the base station will calculate, through triangulation, the approximate location of the transmitter. This location will be compared to the GPS coordinates to give an error percentage, and the GUI will display a top down map of the system for the user. Figure 1 shows the flow of data in this system.

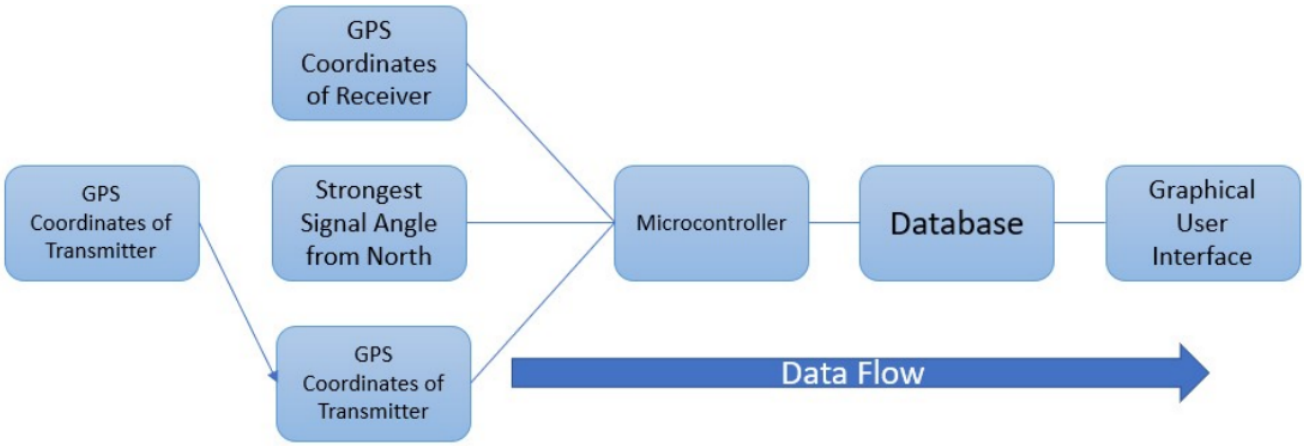


Figure 1. System Overview of Data

2.3. Referenced Documents and Standards

B. G. Stewart, A. Nesbitt and L. Hall, “Triangulation and 3D location estimation of RFI and Partial Discharge sources within a 400kV substation,” 2009 IEEE Electrical Insulation Conference, 2009, oo.164-168, doi: 10.1109/EIC.2009.5166337.

P. Balh and V.N. Padmanabhan, “RADAR: an in-building RF-based user location and tracking system,” Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064), 2000, pp.775-784 vol.2, doi: 10.1109/INFCOM.2000.832252.

Y.Wang, Xu Yang, Yutian Zhao, Yue Liu and L. Cuthbert, “Bluetooth positioning using RSSI and triangulation methods,”2013 IEEE 10th Consumer Communications and Networking Conference (CCNC), 2013, pp. 837-842, doi: 10.1109/CCNC.2013.6488558

3. Operating Concept

3.1. Scope

RF Triangulation enables users to track an object with a known frequency. There is an extreme value in having information on the precise location of items. While some companies may use this data to track a cell phone or other smaller items, the military frequently uses this application to search for enemy warcraft. A huge turning point in war is knowing where your enemy is to precisely strike first. Once the start button is hit, the receivers will go to work sweeping the area and triangulate the coordinates of the transmitter. This system will be able to pinpoint and send the information back to the base station. With time, this system could map out entire paths of an object over a certain time period.

3.2. Operational Description and Constraints

At the start of the process, the 3 antennas will need to be oriented to true North. Once this is done, a start signal will be sent out and then the process of having each of the motors rotate at the same speed taking in the radio signals through the antenna begins. The stepper motor will turn the directional antenna; sweep and find the strongest signal; and send the strongest signal from North to the MCU. The computational part will be done at the user's laptop from where they sent the start signal. Through WiFi, the database is updated with the data gathered from the XBEE, the stepper motor, and each receiver's own set of GPS coordinates. The angles from all individual receivers are triangulated and displayed on GUI. To error-correct this prototype, the signal sent from the transmitter XBEE to the receiver XBEE includes the transmitter's hardcoded GPS location.

Constraints for this system include:
Area: The operational area is 2D, meaning we will not be able to compute a height difference.
Obstructions: The receivers require a line of sight to the transmitter, with limited noise.
Housing: The receivers may experience unideal weather, the vital and electronic parts of the project should be protected.
Maneuverability: Requires a laptop connected to the base station so it is not fully remote.
Accuracy: Requires a smaller area due to radio module limitations.
Economic: The budget will be small for this project, requiring us to settle on the quality of the components.
Setup: The receivers will need to be synced in the same direction, otherwise, it will not be able to compute the strongest signal angle from true North.
Duration: The sweep can take a longer amount of time for sweeps because there must be a pause for a correctly updated RSSI value.

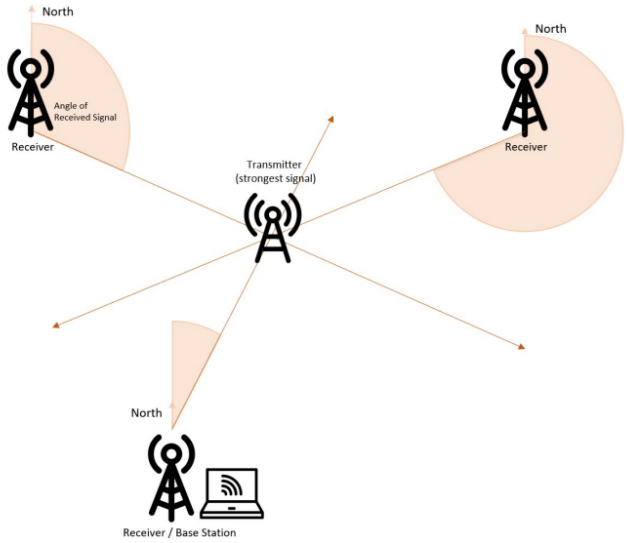


Figure 2: Conceptual Sketch

3.3. System Description

Transmitter: The system will have an omnidirectional antenna constantly transmitting its GPS coordinates on a known frequency. This is the object the receivers will search for.
Receiver system: The system requires three receivers to correctly triangulate the transmitter. There is a directional antenna connected to the radio module, a stepper motor to sweep the area until the strongest signal is found, an MCU to gather all data, and the Pycom to use WiFi to send data to the database.
WiFi: Receivers use WiFi to connect to the database and send the required information that was gathered from the process of the motors finding the strongest signal and receiving the signal that the transmitter was broadcasting.
Computational process: The system has a computational process within the base station where we will have a processor to take in the streams of data from the receivers. From there we will undergo computations that will give a probability of certainty of the location, and then we will display that information into a graphical interface.
Graphic Interface: A graphical interface will display the receivers and the lines given from the strongest signal, and then calculate the intersection of the 3 lines that form an area. We then can calculate the error between the calculated and the expected location.

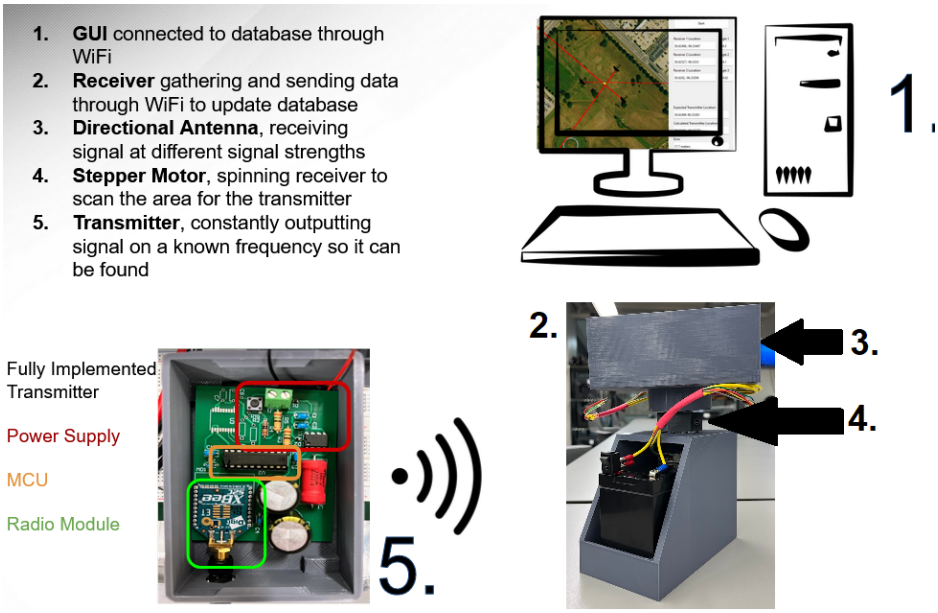


Figure 3: Overview of System

3.4. Modes of Operations

RF Triangulation has three different portions of a full sweep and triangulation: stand-by, sweeping, transmission, and display:

Stand-by: During stand-by mode, the receivers wait for the start signal sent from the user's laptop, until this start button is pressed there is no movement.

Sweep: During a sweep, the receiver is trying to locate the strongest signal angle. This requires a receiver to step completely through North, East, South, and West poles. After one quarter of the compass is determined to have the strongest signal, the receivers will sweep again in between those poles. This step is slowed down to find the strongest signal angle.

Transmission: At the point of strongest signal strength angle being found, the receiver sends the following data: GPS receiver coordinates, GPS transmitter coordinates, and angle from North. All of this data is transmitted through the Pycom, over WiFi, to the database where the triangulation calculations are done.

Display: All three receivers GPS locations are displayed with their respective found angles. Then, the triangulated area is displayed in yellow, along with error percentage found from the GPS transmitter coordinates.

3.5. Users

RF Triangulation benefits those who require a tracking system in a limited area. The basics of triangulation do not require a high level of training considering the project will be almost entirely automated, apart from starting the transmitting and receiving of signals. Users need a base station

with an accessible user interface to calculate and locate the strongest signal angle. There are built-in errors to make sure that angles intersect, that no overlapping data points are used, and that the area triangulated is logical.

For less technical users, that may mean finding a tag that is attached to a wallet or having a mobile cleaning robot map their space by storing the points where the transmitter was found. The military uses triangulation to seek and find enemy ships or aircraft during times of war. Specifically, our project was focused on the biologist's desire to track an animal in a set area so they would not interfere with their lives.

3.6. Support

RF Triangulation would be provided support in the form of a user manual providing information on the designed user interface, system installation, and possible common errors. The section on the user interface includes a detailed explanation of how to start, stop, and export the information depending on the user's needs. This manual will provide information on the accuracy of the calculation and signal processing, along with an example calculation.If needed, support could be provided in the form of frequently asked questions and their solutions or technicians to stay on call or provide in-person assistance.

4. Scenarios

4.1. Radio Tracking

With a strong enough signal, directional antennas can be fashioned to point toward the signal source. By using multiple receivers this signal can be triangulated to give a precise coordinate within the space between the receivers.

4.2. Animal Tracking

Much like radio tracking, animal tracking can be implemented to track certain species, as long as the area does not block or distort the radio signal being transmitted from the animal.

5. Analysis

5.1. Summary of Proposed Improvements

Compared to some radio frequency tracking, our system has some improvements. All antennas will be turned on within the same second and orientated in the same direction to synchronize for data collection thanks to the Pycom implementation and WiFi used. The system will not operate until a user manually turns on the base station and asks for it to find a signal, thus the antennas

will not be searching constantly. This means that the sweeping antennas and base station calculation of transmitted signal direction will be performed automatically. Also, the transmitter will have GPS coordinates in addition to the triangulation for error correction.

5.2. Disadvantages and Limitations

RF Triangulation has limitations including:

- The system will not have a z-axis; thus, the set area will be primarily flat.
- If there are objects in the way of the transmitted signal, it will impede signal strength, giving lower accuracy.
- If using the antennas in an outdoor space, the rain will negatively affect the results of accuracy in positioning.
- This system can only be used in a smaller area due to radio module constraints.
- The system batteries will need to be charged or replaced because this is an outdoor system.

5.3. Alternatives

There are many alternate tracking methods, as there is no predominant indoor positioning system. Possibilities include ultrasonic sensing, infrared beacons, cellular signal triangulation, GPS, WiFi hotspot triangulation, and cell tower triangulation. Each of these methods is severely inhibited by crowded, noisy, indoor environments. GPS would be the ideal solution, but it is not accurate indoors. Similarly, cell tower triangulation has limited accuracy in buildings. We estimate there would be at least a 15 ft. margin of error using both of these methods. All forms of RF triangulation will require several beacons/receivers spaced adequately from one another. IR detection experiences these same issues, and is more reliant on line-of-sight communication. Ultrasound as well is subject to noise and reflections.

5.4 Impact

RF Triangulation potentially has the ability to track any object with the transmitter attached. Because of this, there is a concern about general tracking and mapping of an entire area. If used in more advanced systems, such as an automated miniature vacuum that maps an entire house, the data could possibly be sold. However, since our project is primarily focused on a set area with basic signal transmitting and receiving, there are not many concerns.

FUNCTIONAL SYSTEM REQUIREMENTS

6. Introduction to Functional Requirements

6.1. Purpose and Scope

For a long time, biologists have struggled to effectively track animals such as endangered species without getting overly involved in their lives. If the biologists are too involved it can skew data and keep them from being able to accurately study the creatures. This project is intended to prove that RF Triangulation is a potential solution to this problem. This solution is executed by inserting a transmitter that transmits a known message at a known frequency under the skin of an animal. Then at least three receivers will be placed out into the wilderness that will scan the environment for data and relay it back to a database. From there the user can read data from the database through our GUI. For our system we built our own transmitter that operates at 2.4Ghz. It transmits the signal omni directionally. Our 3 receivers sweep a full 360 degrees with the use of a stepper motor for accurate and precise movements. The receivers read in the signal at each step with a directional antenna and determine the direction in which the received signal strength is the strongest with at least 5 degree accuracy. Each receiver is powered with a 12V battery. From there, they each send their direction to the database which will process the information of all receivers and the results get displayed on the GUI.

6.2 Responsibility and Change Authority

The team leader responsible for checking and verifying that all requirements of the subsystems and overall project are met is Jack Parkinson. Changes in the requirements can only be changed with approval by Dr. Nowka, Max Lesser, and the team leader.

Subsystem	Responsibility
Radio / Transmitting Signal	Josh Broyles
Antenna / Receiving Signal	Jack Parkinson
Sending and Receiving Data to MCU	Kathleen Hutchinson
GUI and Database	Brandon Stokes

Table 1: Subsystems and Leads

7. Applicable and Reference Documents

7.1. Applicable Documents

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

Document Number	Revision/Release Date	Document Title
IEEE 802.15.4	2011	Standard for Low-Rate Personal Area Networks
IEEE 802.11	1997	Standard for Wireless Local Area Networks
IPC A-610E	Revision E – 4/1/2010	Acceptability of Electronic Assemblies
MIL-STD-461	Revision E – 8/20/1999	Requirements for the Control of Electromagnetic Interface Characteristics of Subsystems and Equipment
MIL-STD-464	Revision C – 2/1/2010	Electromagnetic Environmental Effects Requirements for Systems

7.2. Reference Documents

Refer to section 2.3 of Concept of Operations document

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Document Number	Revision/Release Date	Document Title
IEEE C95.*1*	02/08/2019	Standard for Safety Levels With Respect to Human Exposure to Radio Frequency Electromagnetic Fields

7.3. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable document are considered to be for guidance and information only, with the exception of ICDs that have their applicable documents considered to be incorporated as cited.

8. Requirements

The Radio Frequency (RF) Triangulation System requires the ability to have reliable predictability of a signal transmitter’s particular location. This system allows users to determine the location of each of the receivers and display both the calculated point and the GPS location of the transmitter calculated through the receiver's angle of the strongest signal from true North through triangulation. There must be a reasonably small percentage of error of each sweep. The three subsystems of this project are: the transmitting radio; the receiving antennas; and the GUI (Graphical User Interface) calculations and database that stores the relevant information.

8.1. System Definition

The RF Triangulation System consists of a reliable location detection system. There are four major subsystems described in 6.2 of the FSR. The project consists of the triangulation of a single known signal from the transmitter to allow the user to find the source of the signal which will be displayed on a map on the GUI. The antennas allow for an automated search system to find the transmitter, so the user only needs to manually turn the system on, wait for the sweep to be completed, and then view both the location plus an error percentage from the GPS coordinates received. The main subsystems are Radio Transmission, Antenna and Motor, Sending and Receiving Data, and GUI with a connected database. All subsystems, except for database and GUI, were in charge of handling their own power subsystems. The transmitter required a power supply of at least 3.3V. However, receivers required 12 V for the stepper motor, 5 V for the WiFi module, and 3.3 V for MCU and all other modules.

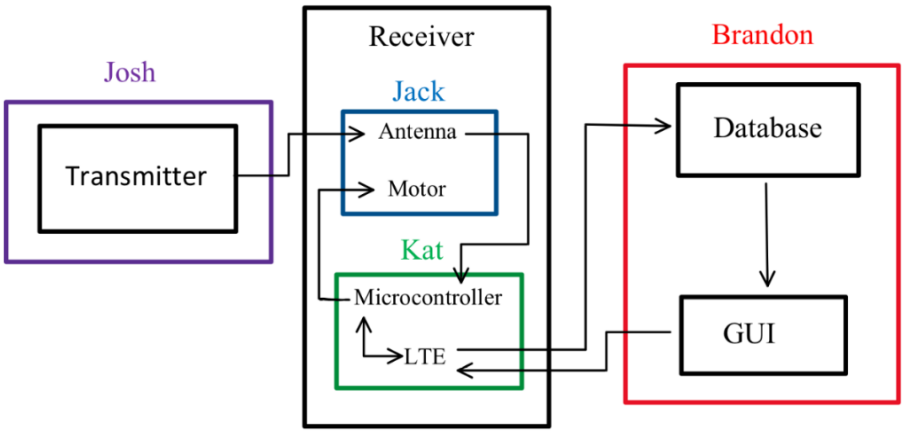


Figure 4: Functional Block Diagram

The basic system involves four total antennas, one transmitting the signal and three receiving, and a base station for the user. These antennas are all running on a power supply from a battery in order to allow general transportation to any area for testing, from fields to parking lots. The antennas are arranged in an open space where the transmitter is placed in the middle of the triangulated area of the receivers. However, this does not mean the transmitter must be placed directly in the middle of the receivers, it must be within line of sight and 150 meters at least from the receivers.

The transmitting analog signal communicates its GPS coordinates to the receivers. The receivers will first check each pole with a stepper motor. The motor moves to find the strongest signal from a specific pole, it will then go to that pole and do a sweep of that area and find the resulting strongest signal. The following data is then collected: GPS coordinates of the transmitter, GPS coordinates of the receiver, and angle of strongest signal from true North. The collection of this signal data is processed in the microcontroller where the WiFi module passes it to the database.

All of this information is processed correctly and sent to the database where it is calculated and displayed on a map in an user interface. If it is not processed correctly the system will start again hoping to find the desired results. The map will display both the calculated point along with the GPS coordinate of the transmitter along with the error percentage. Overall, the system can display the transmitter’s location along with the receiver's location with the strongest signal from each.

8.2. Characteristics

8.2.1. Functional / Performance Requirements

8.2.1.1. Battery Operating Time

The antennas shall be able to operate for at least 30 minutes on battery power. For receivers, this includes an operating step motor and the microcontroller. For the transmitter, this includes the ability to continuously operate the radio and the microcontroller.

> *Rationale: The RF Triangulation system needs to operate for a long enough duration for accurate data collection and display of transmitter location. An hour, with sweeps and collection about every 10 - 30 seconds, should give plenty of data and location approximation.*

8.2.1.2. RF Transmission Time

The receiving, processing, and output of signals to give an approximate location shall not exceed 10 minutes.

> *Rationale: The system's usefulness depends solely on the transmission and collection of the data from the antennas. The speed of reading RSSI and determining the approximate angle triangulation within a short time is crucial for a productive system.*

8.2.1.3. Operational Search Range

The RF Triangulation System shall support a search range of at least 100 meters.

> *Rationale: The RF Triangulation system has a 2.4 GHz radio transmitting and there are 3 receivers each with a 360° step motor with a Yagi antenna attached to it. The data will be transmitted over WiFi so the search range is limited by the range in which the signal will not be corrupted*

8.3.1.4. Time to Load GUI

The RF Triangulation system will take <15 seconds to fully load the interface

Rationale: The RF Triangulation system will take around 30 seconds to fully grab all the required information so taking around 15 seconds to load should give it plenty of time to adjust to the final product

8.3.1.5. Receiver Data Usage

The WiFi transmission will not use more than 100Mb of data within the search time

Rationale: The transmission is only sending a string of data and a connection a the database, less than 100 Mb is very large amount of data and it will not exceed this amount

8.3.1.6. Database Duplicates

The database will not allow duplicate data of multiple receivers inside the database, it has checks within it such as primary keys to only allow a unique value for each receiver.

Rationale: The database should not be handling other values such as a NULL value in one of the locations or a receiver being in the database multiple times.

8.2.1.7. Time to Load Satellite Image

The RF Triangulation interface should be able to take in the receivers location and grab a reasonable box from the web to get the background map in less than 10 seconds

Rationale: The RF Triangulation interface will at first load a grid and then when it gets a reasonable location it will be able to go to the web and pull an image of a location based on satellite images

8.2.1.8. Time for Initial Sweep at Startup

The 360 degree sweep at the start of the locating process should take a maximum of 5 minute seconds to complete.

Rationale: The initial sweep covers all 360 degrees and therefore needs more time to complete its sweep. However, this can't take too long or the system will be inefficient.

8.2.1.9. Time for Regular Sweeps

The sweeps following the initial sweep should take a maximum of 5 minutes to complete.

Rationale: The sweeps following the initial sweep will take less time because it will be covering a smaller angular range. However, the motor will cycle slower for more accurate measurements.

8.2.2. Physical Characteristics

8.2.2.1. Antenna Movement Degrees

The signal receiving antennas of the RF Triangulation system will operate with the ability to search 360 degrees.

Rationale: This is a requirement specified because each antenna must be able to search the entire area, not a specific designated corner of the triangulated area.

8.2.2.2. Weight

The weight of the receiver's contents should not exceed 15 pounds.

Rationale: This maximum weight should satisfy the constraints of the motors, which will have performance issues attempting to move too much weight.

8.2.2.3. Mounting

The mounting information for the RF Triangulation System shall be captured in the RF Triangulation System ICD.

Rationale: As the RF Triangulation System mounts to a stationary platform, mounting requires a specified range and flat field of view.

8.2.3. Electrical Characteristics

8.2.3.1. Inputs

a. The presence or absence of any combination of the input signals in accordance with ICD specifications applied in any sequence shall not damage the RF Triangulation System, reduce its life expectancy, or cause any malfunction, either when the unit is powered or when it is not.

b. No sequence of command shall damage the RF Triangulation System, reduce its life expectancy, or cause any malfunction.
c. There should be an input into the system to start the receivers that is communicated from the GUI

Rationale: By design, should limit the chance of damage or malfunction by user/technician error.

8.2.3.1.1 Input Voltage Level

The input voltage level for the Transmitter shall be +8 VDC to +10 VDC

Rationale: Transmitter needs to have a long battery life in order to be effective since unlike the receivers, it can't be serviced.

The input voltage level for each Receiver shall be +11 VDC to +14 VDC

Rationale: The stepper motor with 28 oz-in holding torque requires a minimum of 12V input voltage. This will be driven by the 12V supply through a linear regulator into the stepper motor driver IC. This supply also provides power to the rest of the board

8.2.3.2. Outputs

8.2.3.2.1 Speed

The time it takes for the system to be started and receive an approximate location shall range from 1 - 5 minutes depending on how accurate the user wants the system to be.

Rationale: By checking more points and doing larger sweeps, the system becomes more precise but the sweeps will take longer. Therefore the time will vary depending on the needs of the user. A sweep with up to 2 degrees of accuracy will only take 5 minutes.

8.2.3.2.2 Power Consumption

The average power for the receiver shall not exceed 6 watts.

Rationale: The 12V7Ah battery shall have an average current draw of at most 0.5A so that the receivers can last at least 14 hours of sweeping without needing to be serviced. This will allow for at least 168 sweeps.

8.2.4. Environmental Requirements

The RF Triangulation System shall be designed to withstand and operate in the environments specified in the following section.

Rationale: RF Triangulation will need to operate in an outdoor environment

similar to College Station, Texas with clear weather conditions.

8.2.4.1. Altitude

The RF Triangulation System shall be designed to withstand and function properly at altitudes from sea level to 400 feet above sea level., the elevation of College Station, TX

8.2.4.2. Thermal

The RF Triangulation System shall be designed to withstand and function properly at temperatures from 50℉ to 110℉.

Rationale: RF Triangulation will need to operate in an outdoor environment similar to College Station, Texas with clear weather conditions. Clear weather conditions assume no rain or snow, but heat can reach to 106℉.

8.2.4.3. Humidity

The RF Triangulation System shall be designed to withstand and function properly at humidity from 0% to 50%.

Rationale: RF Triangulation will need to operate in an outdoor environment similar to College Station, Texas with clear weather conditions.

8.2.5. Failure Propagation

The RF Triangulation System shall not allow propagation of faults beyond the RF Triangulation System interface. Any problems will be caught by the internal tests of the database.

8.2.5.1. Failure Detection, Isolation, and Recover (FDIR)

8.2.5.1.1 Built-In Test (BIT)

The RF Triangulation System shall have an internal subsystem that will generate test signals and evaluate the RF Triangulation System responses and determine if there is a failure.

8.2.5.1.1.1 BIT Critical Fault Detection

The BIT inside the GUI shall be able to detect a critical fault in the RF Triangulation System 100 percent of the time.

Rationale: This allows the system to test for critical faults to inform the user.

8.2.5.1.1.2 BIT False Alarms

The BIT shall have a false alarm rate of less than 5 percent.

Rationale: This limits the number of false alarms sent to the user.

8.2.5.1.2 BIT Log

The BIT shall save the results of each test to a log that shall be stored in the host device of the RF Triangulation System where the GUI is based. Afterwards this will be uploaded to the database

Rationale: This allows the user to monitor test results and show if there are any patterns to system failures.

8.2.5.1.3 Isolation and Recovery

The RF Triangulation System host device should provide for isolation of faults and the ability for the system to be reset.

Rationale: A full reset may be required for extreme system errors through the GUI.

9. Support Requirements

9.1. Customer Requirements

9.1.1. Computer Connected to the Internet

Access to the provided user interface requires a laptop that has a connection to the internet. Without this, you can not access the data you are retrieving from the receivers.

Rationale: The user interface uses the internet to scrape data from the database and find satellite imaging on the area of interest.

9.1.2. WiFi Connection

The Receivers require a WiFi connection to transmit data to the database. Can handle WiFi hotspots, the connection will have to be on 2.4 GHz as the Pycom device does not support 5 GHz WiFi connections.

Rationale: If the receivers have spotty connection or no connection to cellular service, they will not be able to communicate with the database or the user interface.

9.2. Provided System Components

9.2.1. Hardware

The system provided to the customer comes with 3 receivers. This includes all the hardware pre installed and tested as well as a battery and attached antenna.

9.2.2. Software

The system provided to the customer comes with a GUI and database ready to receive and relay data. This GUI is easy to use and only requires internet connection to run.

9.3. Maintenance

9.3.1. Battery Replacement

Each receiver will require battery replacement or charging in order to ensure continued functionality over long periods of time.

> *Rationale: Each receiver is an individual unit and requires an independent power source. Without continued checking of the power supply, a receiver will die and become useless.*

Appendix A Acronyms and Abbreviations

BIT	Built-In Test
Hz	Hertz
ICD	Interface Control Document
mA	Milliamp
mW	Milliwatt
MCU	Microcontroller Unit
ft.	Foot
mph	Miles Per Hour
rpm	Revolutions Per Minute
VDC	Volts Direct Current
V	Volts
FDIR	Failure Detection, Isolation, and Recover
Ah	Amp hours
PWM	Pulse Width Modulation

Appendix B: Definition of Terms

Triangulation	the measurement of a network of triangles to determine the relative position of a point through the calculation of three angles from a particular axis

APPENDIX B - INTERFACE CONTROL DOCUMENT

10. Interface Control Document Overview

This Interface Control Document (ICD) will cover the Radio Frequency (RF) Triangulation System and provide details on the various subsystems. The Concept of Operations (ConOps) and Functional System Requirements (FSR) stated these subsystems previously, and this document will expand on how each subsystem interacts with each other. This includes the power, data communication, placement, and other physical attributes of these subsystems. Overall, it will explain how the subsystems work together to result in the requirements mentioned in the ConOps and FSR documents.

11. References and Definitions

11.1. References

Refer to section 2.3 of Concept of Operations document

11.2. Definitions

SPI	Serial Peripheral Interface
I2C	Inter-Integrated Circuit
UART	Universal Asynchronous Receiver/Transmitter
CSI	Camera Serial Interface
V	Volt
LED	Light Emitting Diode
in-lbs	inch-pounds
rpm	revolutions per minute
A	Ampere
W	Watt
CPU	Central Processing Unit
RAM	Random Access Memory
WLAN	Wireless Local Area Network
ADC	Analog-Digital Converter
dB	Decibel
kHz	Kilohertz
GHz	Gigahertz
Ah	Amp-hour
nm	nanometer
GB	Gigabyte
PWM	Pulse Width Modulation

12. Physical Interface

12.1. Weight

12.1.1. Transmitter Antenna

Transmitter antenna will weigh less than 1 pound (lbs). This allows an average person to lift and move the transmitter to a different point to change coordinates.

Component	Weight (lb)	Number of Items
GW.26.0112	0.00882	1
XBee Module	0.028	1
MCU	0.012	1
Buck Converter	0.0010729898	1
Housing	0.185	1
Power Block	0.00625	1
Antenna	0.01875	1
Battery pack	0.4	1
Total Weight	0.66	

Table 2: Transmitting Antenna Weight Specifications

12.1.2. Receiver Antenna(s)

Receiver antennas will weigh less than 10 pounds (lbs) each. This allows an average person to lift receiving antennas and install them where necessary without any heavy equipment.

Component	Weight (lb)	Number of Items (Per Receiver)
PCB board and parts	0.315	1
12V Battery	5.27	1
Directional Antenna	TBD, 0.1 - 1	1
Stepper Motor	0.441	1

Housing	0.6	1
Tripod	1lb	1
Total Weight	TBD, 7.73 - 8.63	

Table 3: Receiving Antenna Weight Specifications

12.2. Dimensions

In order to stay within reasonable bounds, the antennas can only be about 5 feet tall, maximum. The receiver antennas were created to be about a foot tall, and the transmitter was around a quarter foot tall.

12.4. Mounting Locations

The transmitter will be required to sit on a flat surface, much like the receivers.

In addition, the microcontroller, motors, and battery will all be inside the body of the receiver antennas. The printed circuit board (PCB) will be placed on top of the stepper motor through a clamp designed by 3-D printing. The directional antenna will be allowed to poke out of the box in order to reduce obstructions in signal searching.

13. Electrical Interface

13.1 Stepper Motor

The stepper motors used for the receivers have to be powerful enough to spin the receiver PCB and antenna with precision and speed. Each motor will weigh 0.441 lbs and will be mounted on top of the battery, below the PCB and antenna. The motor is driven by the 12V battery and controlled by a stepper motor driver IC on the PCB. It has a holding torque of 28 in-oz and draws on average about 0.215A from the power supply with a load of 1.5 lbs.

13.3 Primary Input Power

13.3.1. Transmitter

A replaceable battery will be the primary input power for the transmitter. To provide around 40 hours of constant usage with one charge, a 9V battery will be used.

13.3.2. Receiver

A rechargeable battery will be the primary input power for receivers. To provide around 30 minutes of constant usage with one charge, a 12V battery with 7Ah capacity will be used for the power system.

13.4 Thermal Interface

There were two heat sinks that needed to be added to the PCB for 2 of the buck converters. The 12V to 3.3V needed to have a heat sink as well as the 12V to 12V buck converter needed a heat sink.

14 Communications / Device Interface Protocols

14.1 Wireless Communications

Communication will be done through WiFi on each Pycom GPy 1.0 and these will follow IEEE 802.11 standards.

14.2 Signal Interfaces

14.2.1. XBee Module

The XBee Module will output a signal from the transmitter that will be received by the antennas on the receiver's XBee Module through universal asynchronous receiver-transmitter (UART).

14.2.3 Pycom GPy 1.0

The Pycom GPy 1.0 will be used on the receivers as an WiFi Module to connect to the database through UART connection.

14.3 User Control Interface

The GUI will be located on the user's laptop, and it will both communicate a signal to the antennas to sweep and will receive information from the receivers through the Pycom module.

14.4. Microcontroller Pin Interface

There are two separate types of microcontrollers for the transmitter and receiver respectively. The standardized interface for the microcontroller on the receiver, the ESP32, is shown in Table 5. This highlights the most important reserved GPIOs for the corresponding modules.

ESP32-WROOM-32D GPIO	Use
16	Pycom UART, RX
17	Pycom UART, TX
18	XBEE UART, RX
19	XBEE UART, TX
25	Stepper Motor: Step
26	Stepper Motor: Direction
27	Stepper Motor: Reset
32	Stepper Motor: Homen
33	Stepper Motor: USM0
35	Stepper Motor: USM1

Table 4: Pin Interface for ESP32-WROOM-32D Microcontroller for Receivers

14.5 Host Device

The host device should follow the standard protocol of the manufacturer of the host device, in our case will be Dell’s standards.

14.6 Graphical User Interface

The GUI should be displayed on the host device's screen through HDMI or built in screen which follows HDMI protocols or manufacturers standards for the screen.

14.7 Transmitter XBEE Radio

The transmitter will use a XBee radio operating at 2.4GHz that uses IEEE 802.11b standard. This will be used to transmit the signal containing the transmitter's GPS location.

APPENDIX C – SUBSYSTEM REPORTS

15. Introduction

The Radio Frequency Triangulation system will begin when the GUI presses start, it then begins searching for the strongest signal by stepping through each pole and then performing a sweep around the pole that has the strongest signal to find the exact angle. It will then determine the location of the transmitted signal from the triangulation of all three receiver's strongest angle lines. The system's receivers gather data from the modules and push all of the data to the database hosted on AWS. This data is then used in the calculations for the triangulation of three receivers, with the GPS coordinates of the transmitter used only as error correction. The system is broken down into the transmitter and its power, receiver modules and their power, computing, communication, and information storage subsystems.

16. Power Subsystem Report

16.1 Subsystem Introduction

The power subsystem consists of four separate converters. On the transmitter side, there is only one converter; however, the receiver side has three converters required for powering and regulating the system. Each power subsystem is designed to supply regular power to all the components of the system that will be placed in the field. The subsystem is powered exclusively by batteries. The power subsystem was tested to confirm its stability, capacity, and consistency. Each test helped verify that the power subsystem will support the components of the system that will be powered by it.

16.2 Subsystem Details

For the transmitter, a constant supply of 3.3 V was required to power the MCU and the XBEE on the board.

For the receiver, the stepper motor driver required the highest amount of voltage at 12 V. The second highest was 5 V for the GPy 1.0, or the Pycom, which had its own voltage regulator built in. Lastly, the ESP32 and XBEE required 3.0 to 3.3 V constantly. The primary challenge with the power subsystem for the receivers was finding the correct buck converters, these buck converters are different from the first revision of the project due to current draw.

16.3 Subsystem Validation

16.3.1. Transmitter Power Supply

The Transmitter power supply is a simple buck converter that works to turn the 9 V battery

input into a smooth 3.3 V output for the different modules on the pcb. We ended up not implementing the ublox module into the final board so the system draws a very small amount of current to operate. With the calculated current draw hovering around 0.03 A but averaging around 0.022 A.

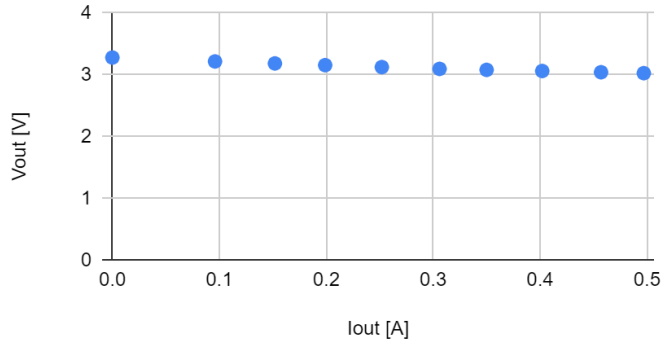


Figure 5:Transmitter Load Regulation

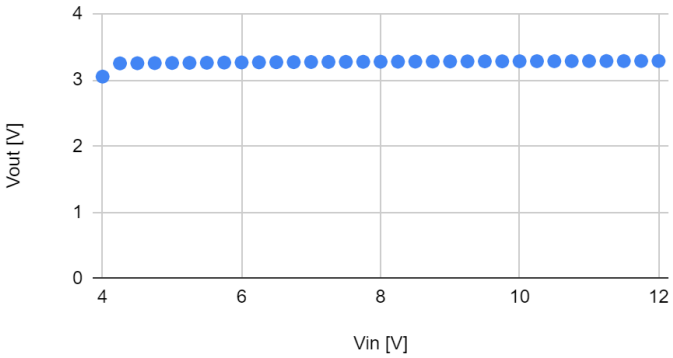


Figure 6:Transmitter Line Regulation

Above we can see Figure 5, at constant 9V, performs in the ranges needed for the transmitter (0.02 - 0.04A). For Figure 6, at constant 0.018A, can stably keep the voltage at 3.3V until 4V is reached, then the buck converter loses efficiency.

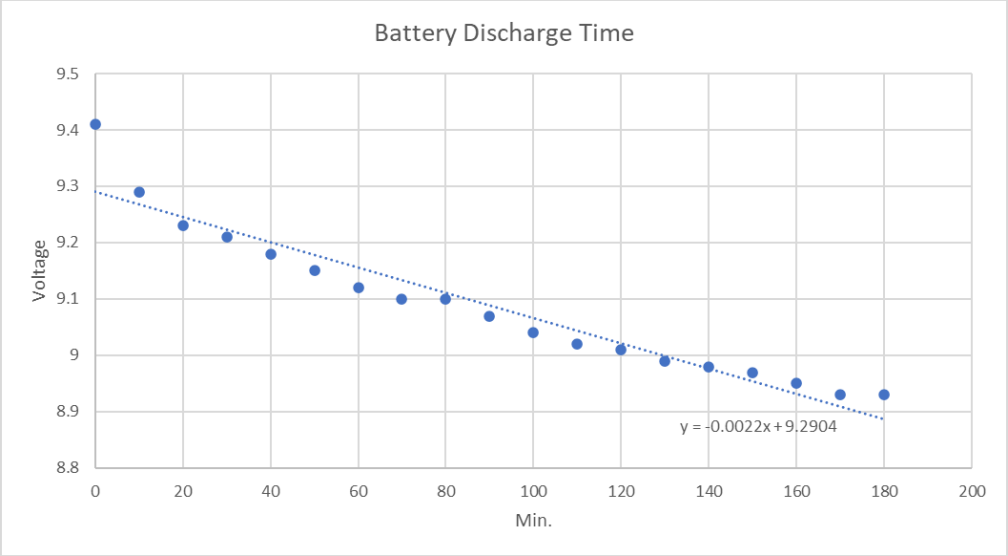


Figure 7: Battery Discharge Time

In Figure 7 we can see that the 9 V battery pack provides input voltage ranging above what the expected value is. From Figure 6 we know that the buck converter can handle voltages such as this spike. Using the trendline as a general guide we know that the transmitter can last for about forty hours without falling below the four volt threshold.

16.3.2. Receiver Power Supply

The receiver power supply consisted of three separate buck converters. First, the 12 V to 3.3 V buck converter was used for supplying power to the ESP32 and XBEE. Originally, the current draw was calculated with left off modules, the UBLOX and magnetometer; the table below displays the current required for the subsystem including these parts, in case a future development is made with the GPS and magnetometer.

Description	Max Current Draw [A]
ESP32	0.5
XBEE	0.033
UBLOX	0.067
Magnetometer	0.0027
Total	0.6027

Table 5: Receiver Current

As the calculated current for the modules was around 0.6 A, the system went with a 12 to

3.3 V with 1.0 A, found in the buck converter LM2575T-3.3NOPB. This system worked well with two capacitors and the inductor specified by the manufacturer in the datasheet. However, possibly due to its smaller footprint, there were some heating issues, so the project decided to use heat sinks. Below are the load and line regulations conducted on this buck converter.

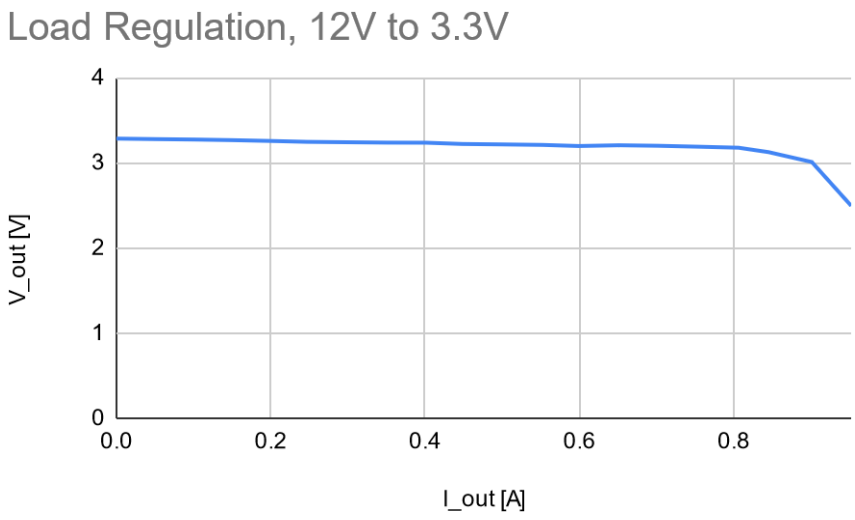


Figure 8: Load Regulation of 12V to 3.3V Buck

As the graph displays, using a voltage supply of 12V like the battery used, the buck converter stays consistent in a stable 3.3 V output when current draw is changed. For line regulation below, the current draw was set to 0.55 A, which is an acceptable current draw considering the max was calculated with UBLOX and magnetometer in mind, with only maxes being considered of each part. Most likely a 0.55 A draw would be more expected.

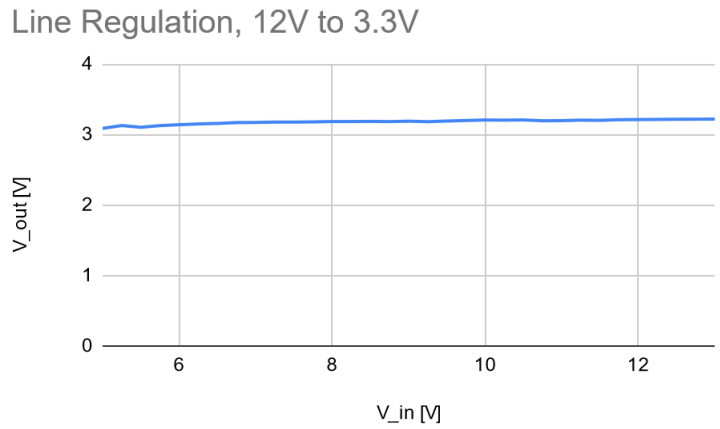


Figure 9: Line Regulation of 12V to 3.3V Buck

This concludes this particular buck converter's validation. The next part is dealing with the GPy1.0 which requires 3.5 V as a minimum, with a 0.15 A max current draw. Thus, an appropriate 12 V to 5 V buck converter, LM2594N-5.0/NOPB, with 0.5 A output was decided on. Below, with a voltage of 12V, the load regulation displays the consistency of this buck converter.

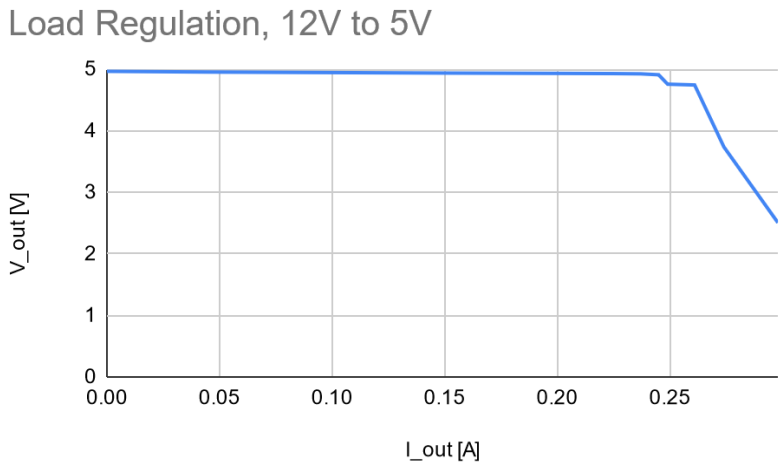


Figure 10: Load Regulation of 12V to 5.0V Buck

This is an acceptable test, and there were no problems with testing using the GPy1.0's full system. Below, the line regulation had a given load current of 0.2 A.

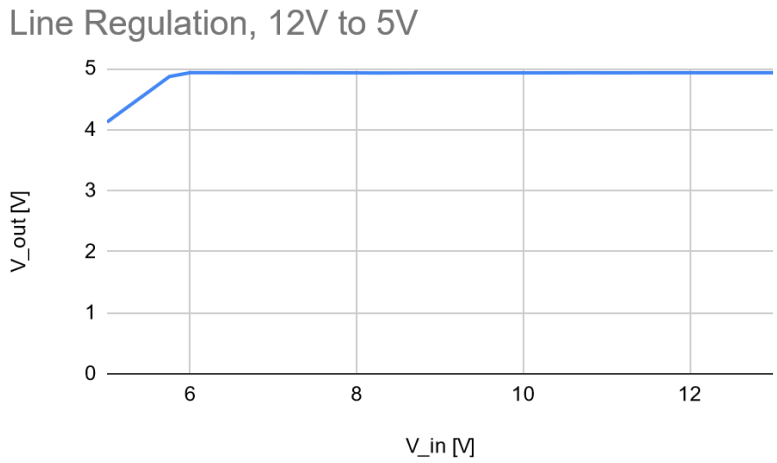


Figure 11: Line Regulation of 12V to 5.0V Buck

For this test, it stayed extremely constant and was even better when supplied with the voltage input the project would supply. Thus concludes the validation for the 12 V to 5 V converter.

Finally, the stepper motor required a regulated 12 V from the already 12 V battery supply. The regulator decided to change from the original because the input voltage on the original revision was too high. The regulator, MIC29150-12WT, validated with the load and line regulations as well. The load regulation was constant until around 1.0 A with a steep drop, but stepper motor IC uses a max current of 0.9 A; when doing full system tests the receiver's highest current draw (with MCU, Stepper Motor IC, Radio Modules, and WiFi working in sync) was only around 1.2 A, so the stepper motor IC most probably never actually got to its highest current. However, another heat sink was required because it had a similar footprint to the 12V to 3.3V converter.

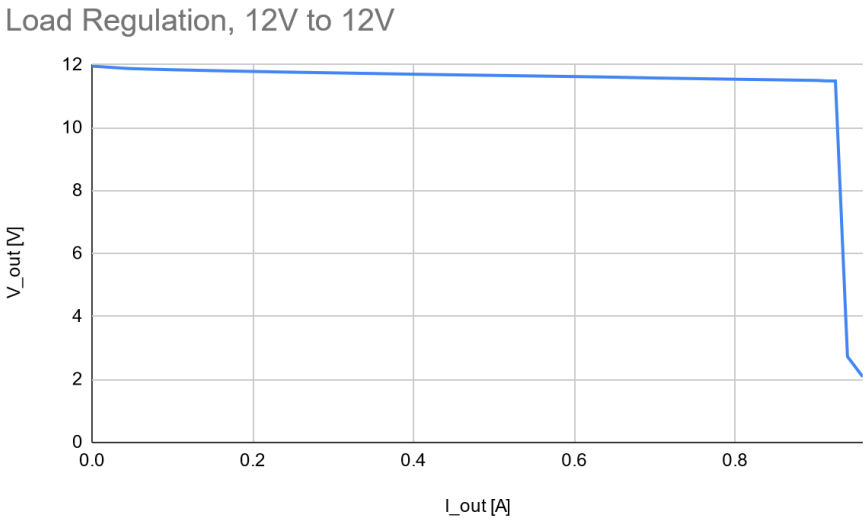


Figure 12: Load Regulation of 12V to 12V Buck

The line regulation also found this buck converter to work well. There was a load current of 0.599 A supplied.

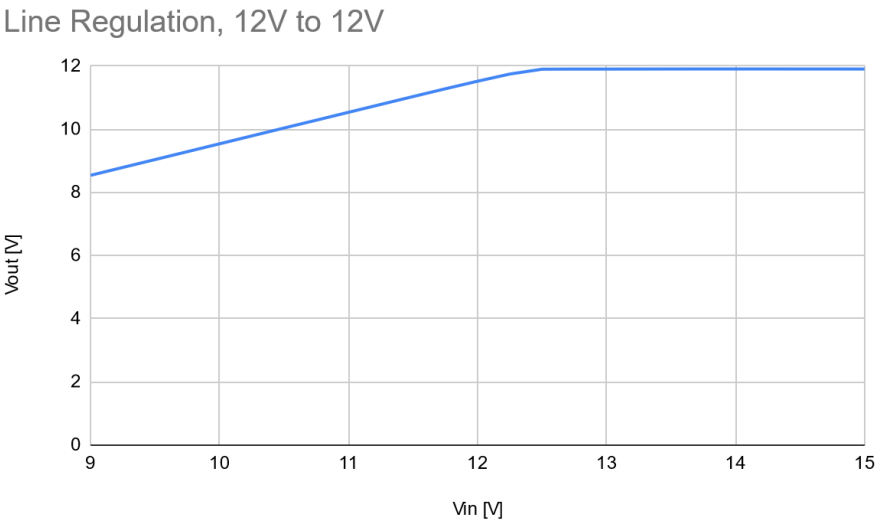


Figure 13: Line Regulation of 12V to 12V Buck

This concluded the individual tests for each converter and regulator in the receiver subsystem. Finally, to validate the continuous battery life for 30 minutes, the system was placed in a basic operation mode and had the voltage of the battery marked every 5 minutes. The constant current was around 0.301 A.

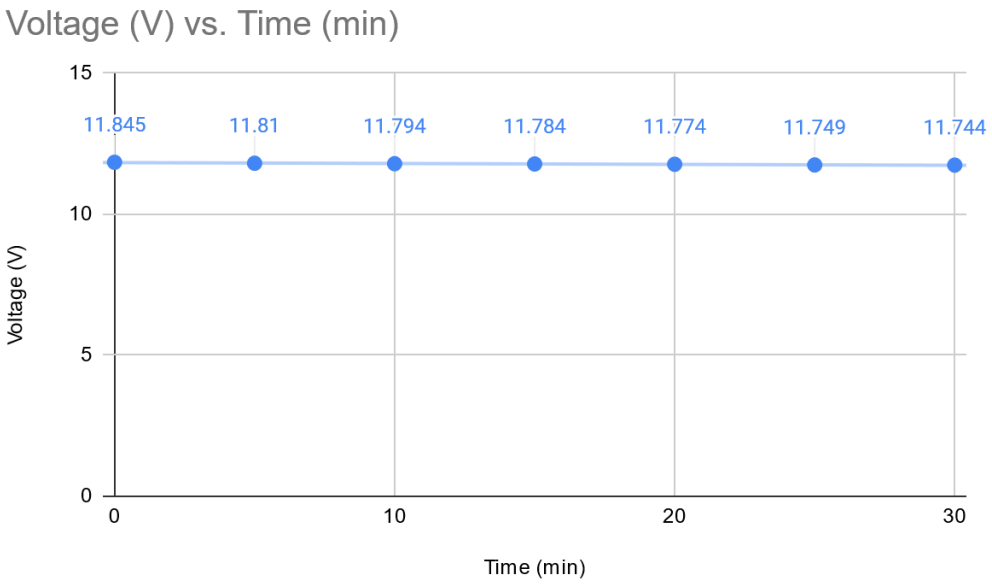


Figure 14: Battery Consumption of Receiver Subsystem

There was an average less than 0.1 V per five minutes, meaning the system could easily operate for more than the minimum 30 minutes specified for validation.

16.4 Subsystem Conclusion

Each part of the subsystem was shown to work correctly. The power subsystem effectively generates power for each of the components. Both the transmitter and receivers are also more than able to operate for 30 minutes as required.

17. Sensor Subsystem Report

17.1 Subsystem Introduction

This subsystem deals with the two major modules in this project, tied to the MCU. The XBEE is the radio module, consisting of transmitting and receiving signals between antennas. The GPy1.0, or the Pycom, is used as a WiFi module. It will collect the data from the ESP32 and send it to the database over WiFi.

17.2 XBEE RF Module

17.2.1. Operation

The XBEE RF module in this system works as the signal connection between the transmitter and the receiver, using a consistent 2.4 GHz and 9600 baud rate for radio modules.

The transmitter is set as the “coordinator” with a specific address. The XBEE from the transmitter sends out on a set frequency on a unique personal area network (PAN) identifier (ID). This PAN ID will be required on all receivers to join the network, this ensures that there are no outlying signals that the receiver will try and listen to.

Each receiver is configured as an “end device”, this means that the receiver can only receive the signal and cannot transmit back to the transmitter. As it listens, it will only look for the transmitter's unique address.

In this system, the transmitter will be sending its GPS coordinates for the receiver to take in for the database’s error correction. This is unnecessary for a real world system because the point is to find an unknown location through triangulation; however, our system is based on an improvement of error for most triangulation systems. Thus, the hardcoded GPS coordinates are what the receiver receives and sends to the MCU over UART.

17.2.2. Testing, RSSI

The XBEE RF module was used for both communication from the transmitter and to the receiver, but more importantly it allowed the receiver to read in the signal strength to help find the transmitter’s location. When trying to diagnose our system's lack of directionality on the radio module, the project members assumed it was due to the lack of strength in the antennas purchased. However, the diagnosis turned to XBEE’s as the root of the problem. Figure 15 depicts a common graph trend in all antennas that were tested with the XBEE’s XCTU software.

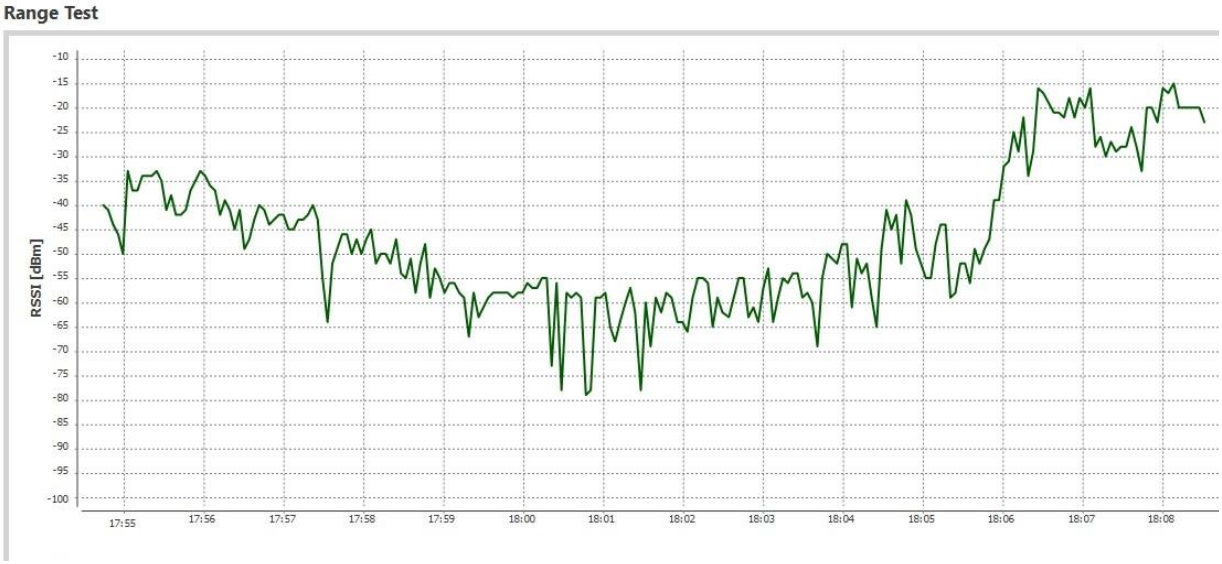


Figure 15: RSSI Range Test Example

While doing RSSI range tests, the system tested three separate antennas, one omnidirectional and two directional. The omni directional had a good amount of package retention from far away. However, when switching to a basic directional antenna the RSSI test performed from an echo test from the ESP32 on the XCTU application displayed a lack

of directional gain. When moving far away, there was an increasing loss of both directionality and in packages. After switching to a yagi antenna with less broad read-in, the XBEEs were concluded to be the problem. The application showed that the RSSI when pointed at the transmitter ended up being more than pointing 90 degrees away from the transmitter. Overall, the curve in Figure 15 shows no major changes in RSSI even when pointed away, it just curves slightly. If this was implemented into our system, the antennas may never hope to triangulate the area correctly.

17.2.3. Testing, ESP32 / Proof of Concept

Originally, the RSSI value would be read in from the XBEE to the MCU over UART. However, this became extremely complicated extremely quickly and was unable to be read in due to the complications of XBEE's on this subject and from the fact that the RSSI was not directional enough to help in this project.

First, the XBEEs have two versions: AT and API. This subsystem has the transmitter in AT mode, which reads in basically like a wired (but wireless) connection. For example, the transmitter sends a string, it receives as a string; However, API mode splits the packet into an actual packet sent in HEX with: a start delimiter, length bytes, API identifier, API frame ID, destination address low, option bye, the actual data packet (transmitter GPS coordinates), and the checksum. To get this, the XBEE's arduino library by Andrew Rapp was going to be used, but the problem arose from the actual library.

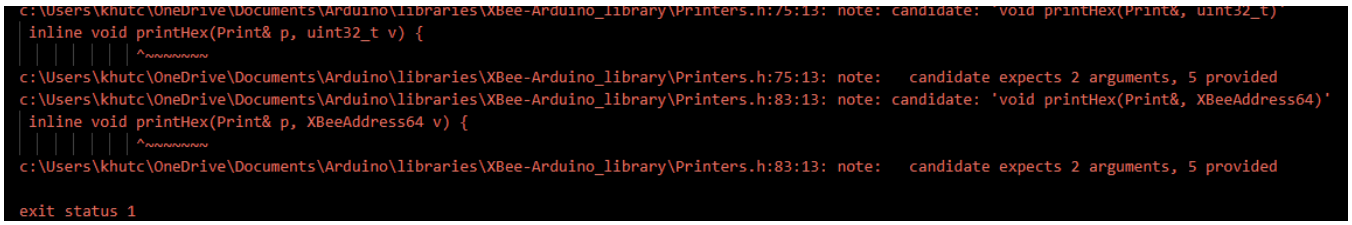


Figure 16: Error in XBEE Library in Arduino

The code that would be used is separating the packet and reading in only the RSSI value as long as a packet was received.

Since the project was nearing demo time, it was preferable to try again in AT mode, instead of API mode. When researching API mode and XBEE S2C radio modules, it should be noted that most threads were extremely negative about the accuracy of XBEE's RSSI.

Switching back to AT mode, it would be possible for two different ways to read in the RSSI. The first would be to read in a PWM signal from pin 6 on the XBEE from the "pulsein" function on the ESP32. The width of this signal would be hypothetically the length of the packet. However, when hooked up to the oscilloscope to diagnose, the PWM signal was both incredibly small and fast. After looking at the signal, the output of the code, which was only zeroes, made more sense. Using the code below, the RSSI should have been able to be read in.

```
#define rssi_pin 15
#define period 200 //us
void loop() {
  // put your main code here, to run repeatedly:
  unsigned long rssi;
  rssi = pulseIn(rssi_pin, low, period);
  Serial.println(rssi);
}
```

Figure 17: RSSI Command for PWM Read-In

The setup is skipped from this code because the serial baud rate is always 9600 for all XBEE's. Unfortunately, as previously mentioned, the output was only 0 with no directionality. Even if the transmitting XBEE was taken around a corner in a large building, the RSSI would only print out as 0. Thus, this was not a useful value to read in for the stepper motor.

The next option in AT mode is to send out an AT command to print out the RSSI value. This requires writing "+++", waiting for max 5 seconds (at this point an ACK of "OK" would be sent), and then writing "AT" plus the command. This also ended up failing, the XBEE would not consistently go into command mode and also print out the RSSI of the last packet.

```
void loop() {
  // put your main code here, to run repeatedly:
  if(!xbeecmd.enter()){
    while(1){
      Serial.println("XBEE not at 9600");
      delay(1000);
    }
  }
  else{
    Serial.println("send rssi command");
    xbeecmd.rssi();
    xbee_serial.end();
    delay(1000);
    xbee_serial.begin(9600);
  }
  if(xbeecmd.enter()){
    Serial.println("XBEE succeeded")
  }
  else{
    Serial.println("XBEE failed");
  }
  xbeecmd.exit();
}
```

Figure 18: RSSI Command for AT Command

Above is the code implemented, where the rssi() command would print out "LAST PACKET RSSI: " and use the "ATDB" command that returns the last packet's RSSI value.

There was a very genuine attempt to read in the RSSI value from the XBEE although it was not particularly good when it came to actual application of directionality. This value would only be used to find the angle of the transmitter and thankfully did not change the database's calculations.

17.2.4. Validation, ESP32

Since the RSSI value was not able to be printed out, the receiver's XBEEs stayed in AT mode and sent the transmitter's hardcoded GPS coordinates to the ESP32 over UART. This was completed and validated correctly. These coordinates would be sent from the ESP32 to Pycom which would send the data to the database for error correction.

The entire collection of the GPS coordinate string to ESP32 to Pycom to database took around a second to complete once the stepper motor successfully sweeped; this validation data will be seen in section 17.3.2. The validation of a completed transmission is seen below, where the "esp32 check " string is from the Pycom starting the sweep.

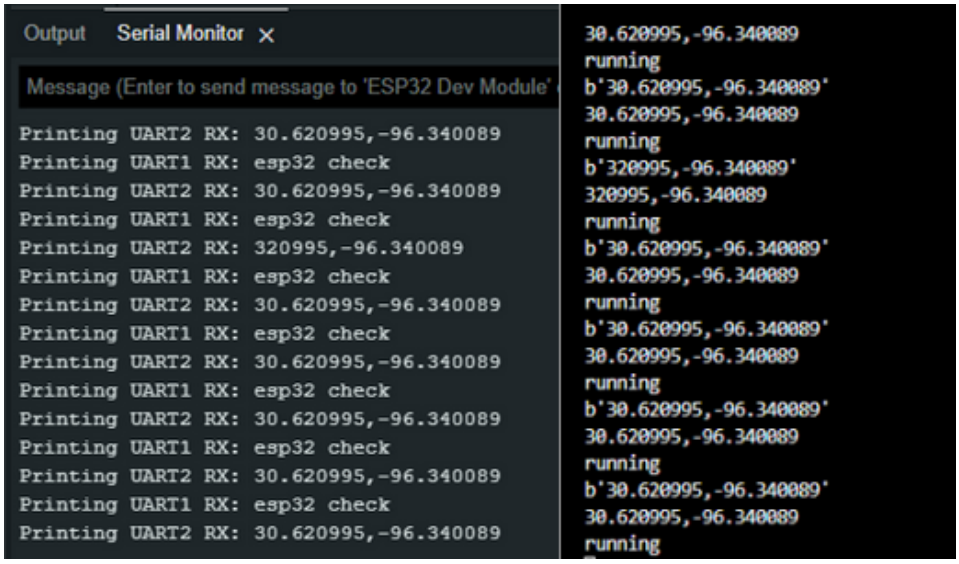


Figure 19: Transmission of GPS Coordinates

This figure shows the left as the serial monitor from the ESP32 sending out to Pycom, on the right, the GPS coordinates from the transmitter.

17.2.5. Conclusion

Unfortunately, due to hardware limitations, the XBEE served only as a way to transmit GPS coordinates from transmitter to receiver successfully. The XBEE S2C is not directionally sensitive enough for this project, even if the quality of the antenna improved. Additionally, there was a significant package drop outside of the set range for this project as well. While the package drop rate is not as significant as the RSSI value, it should be noted that the range is not as large as expected for this system.

In the future, a system like this would benefit from a different frequency than 2.4 GHz, and a more directional-sensitive radio module. An update in the hardware would also make it possible for a better RSSI implementation on the software side. More reliable outputs may make it easier to debug the output of a PWM signal or packet. Unfortunately, the XBEE's seem unreliable in the aspect of RSSI.

17.3 GPy 1.0

17.3.1. Operation

To connect to the database the Pycom GPy 1.0 was found as a method to do this. However, this project will not be using it as the main microcontroller, but only as a way to transmit data to the database.

This board has a built-in antenna to connect to a WiFi LAN network that will allow for the ESP32 to send over UART the data collected from the receiver's XBee module.

17.3.2. Validation

There were three validations for this system of it being able to stay connected for 5 minutes and that it would not drop more than 1 time per 5 minutes and would reconnect within 20s. The last one was confirming that it was able to send an update to the database when given the required data.

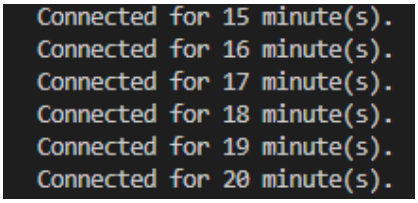


Figure 20: Validation for being connected for 20 minutes

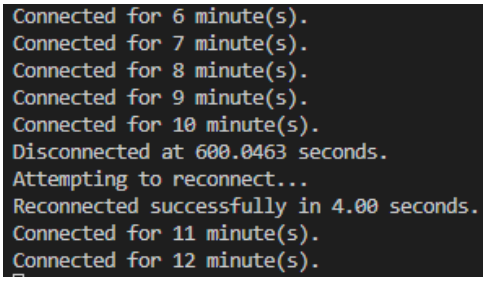


Figure 21: Validation for reconnection within 20 second

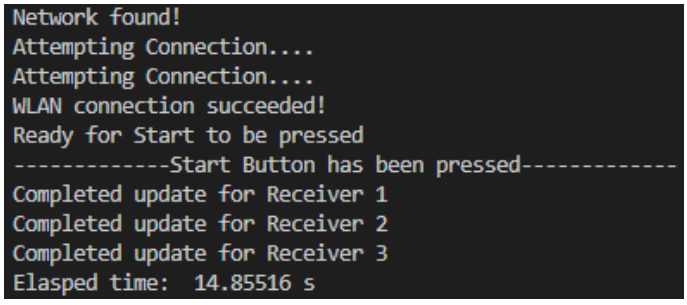


Figure 22: Validation for being able to send data to database

17.3.3. Conclusion

Overall, the GPy1.0 performed well after changing from LTE to WiFi. The GPy1.0 performed the task of sending all required data to the database and was completely validated.

18. Receiver Antenna Subsystem Report

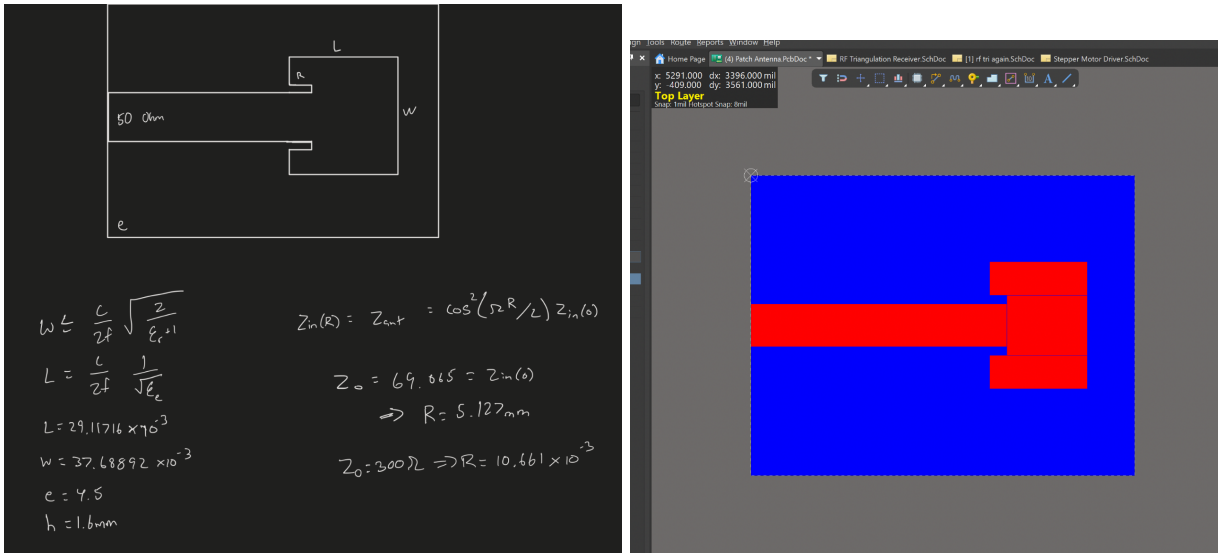
18.1. Subsystem Introduction

The Receiver Antenna Subsystem went through many changes and adaptations. The goal of this subsystem was to create a directional antenna at 2.4Ghz that would be our means of determining the direction of the transmitter relative to the receiver. The antenna designed needed to have high gain in the direction it was pointing and low gain everywhere else. After many iterations of antennas of different sizes and types, 2 different kinds of antennas were bought and tested for the completion of the project.

18.2. Subsystem Details

18.2.1. Patch Antenna

The first antenna design was a microstrip patch antenna. This decision was made because of the patch antenna's ability to be lightweight and easily manufacturable since it would be on a PCB board that could be fabricated through a company.



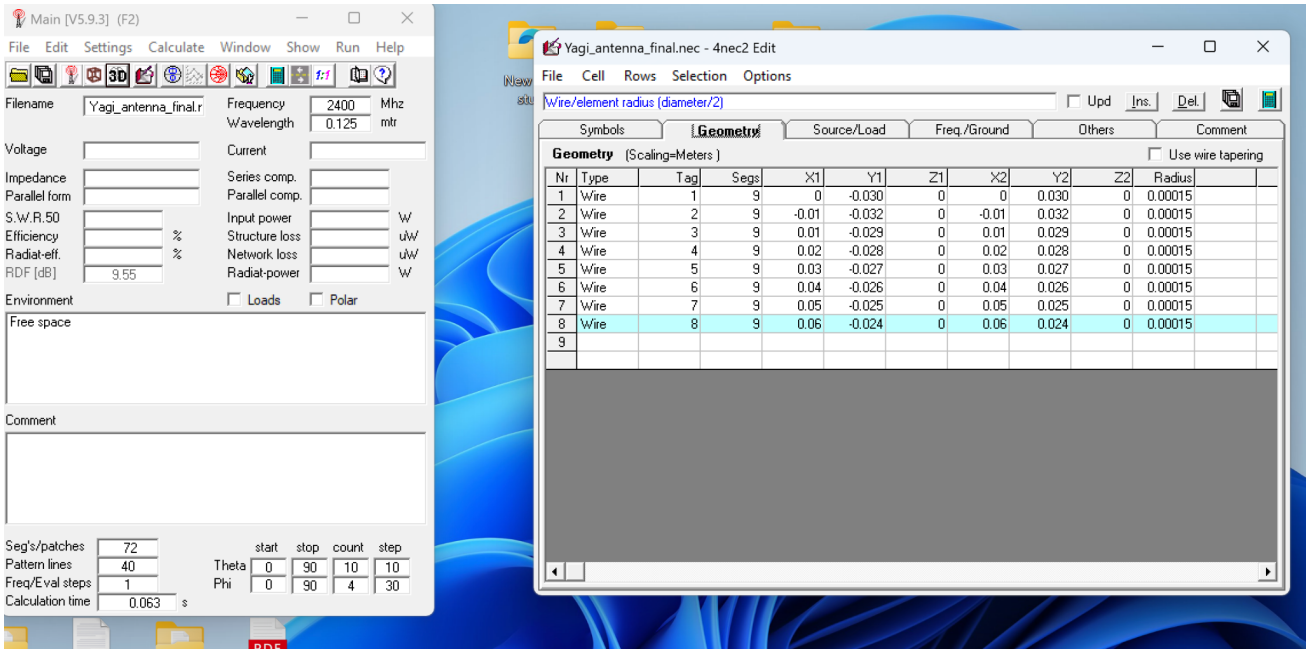
Figures 23 and 24: Patch antenna calculations and PCB design

As seen in Figures 23 and 24 the patch antenna was designed and the PCB design was completed and ready to be ordered. Before ordering though, the design needed to be simulated to verify that it worked properly and the radiation patterns were as needed. Due to an inability to find patch antenna simulation software, this design didn't make it past this

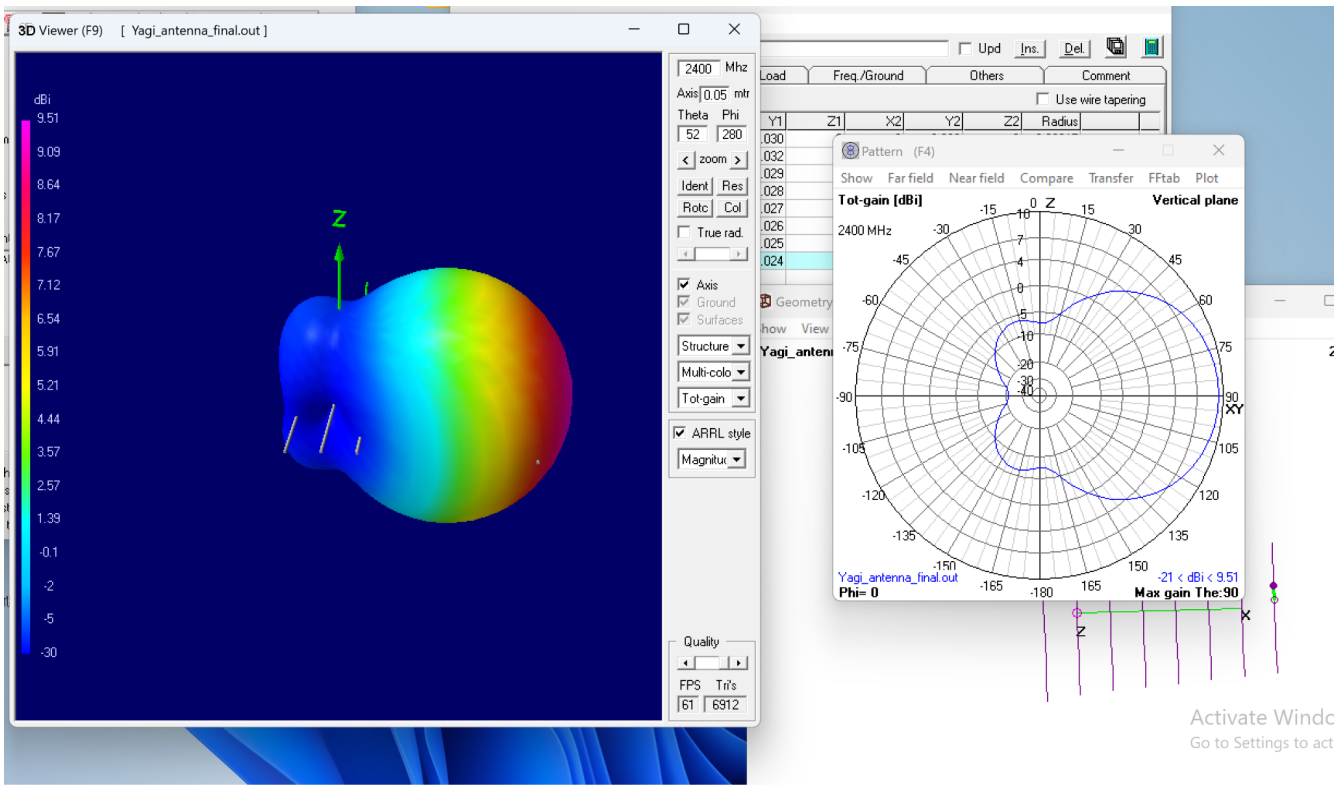
stage.

18.2.3. Yagi Antenna

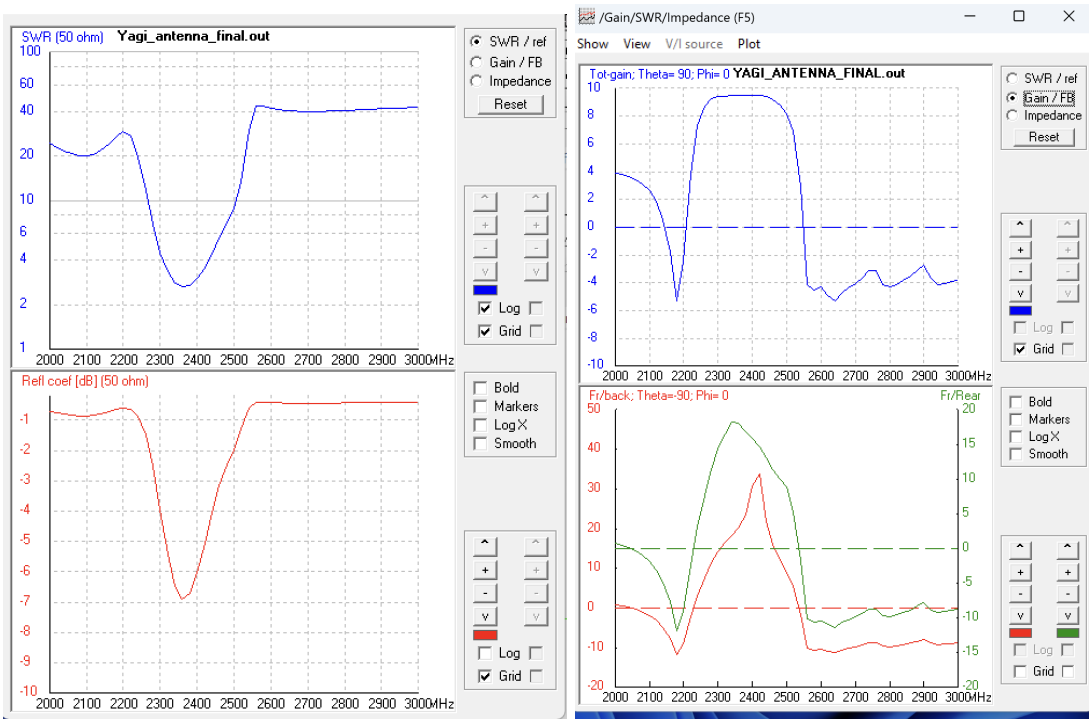
The next design approach was a Yagi-Uda antenna. This design is heavier, harder to manufacture, and optimally used for lower frequencies. However, it is much simpler to design and simulate. Using a yagi antenna calculator, a simple start to the design was created. Using a simulation software called 4NEC2, 12 different yagi designs were simulated with varying numbers of directors and different distances between the elements. A design was chosen that had strong gain, strong front to back ratio, low SWR, and a good radiation pattern. The data on the chosen design is shown in the figures below.



Figures 25: Physical dimensions of Yagi antenna design



Figures 26: Radiation pattern of Yagi antenna design



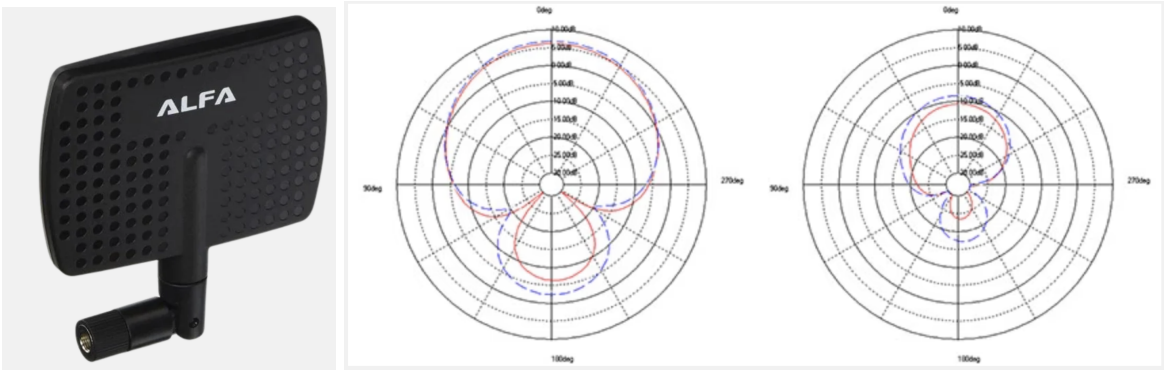
Figures 27 and 28: SWR, Gain, and Fr/back ratio of Yagi antenna design

The simulation data of this yagi antenna was very promising. However, the design was

quite small due to the project's high frequency. This made manufacturing of the antenna quite difficult. Due to the difficulty and a fast approaching deadline, this design was also scrapped.

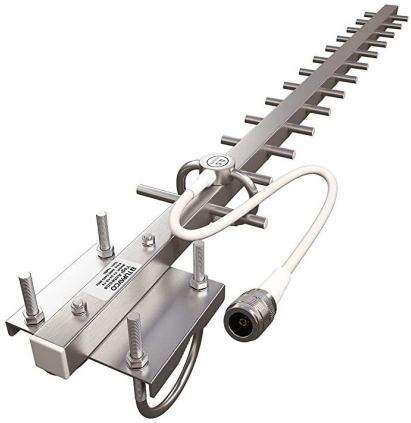
18.2.4. Purchased Antennas

The last resort was to buy an antenna so that the integration of the project could move forward. After some research 2 potential antennas were chosen. After discussing the antennas with the project sponsor, the decision was made to buy one of the cheaper antennas first, test it, and buy the other antenna if needed.



Figures 29: Purchased antenna-1 and its radiation pattern

Above in Figure 29, is antenna-1, the cheap antenna that was purchased. After failed validation with this antenna, a second antenna was purchased. Antenna-2 is shown below.



Figures 30: Purchased antenna-2

- Frequency: 2400-2483 MHz
- Gain: 15 dBi
- VSWR: <1.5:1
- Polarization: Horizontal or Vertical
- Horizontal Beamwidth: 30°
- Vertical Beamwidth: 25°
- F/B Ratio: >16 dB
- Max Input Power: 100 W
- Lightning Protection: DC Ground
- Connector: N Female
- Dimensions: 24.81in
- Weight: 0.93 lb
- Cable Length: 9.45in
- Reflector Material: Aluminum Alloy
- Mast Size: Ø40-Ø50mm
- Rated Wind Velocity: 210km/h
- Operating temperature: -40~+150F

Figures 31: Antenna-2 characteristics and dimensions

18.3. Subsystem Validation

18.3.1. Range Testing

Since we were unable to gain access to a characteristic chamber to test the antennas, we tested them by connecting them to the Xbees and doing a range test. This test is explained in depth in section 17.2.2. In summary, we connected the antennas to the Xbee radios and ran a range test on the XCTU software. This range test performed echoing between two Xbees and displayed the RSSI (received signal strength) of each packet received. It also displayed when a packet was lost. By doing this test, we were able to determine the range and directionality of the antennas by pointing them in different directions at different distances from the transmitter and seeing how it affected the RSSI value.

18.3.2. Antenna-1 Test

In section 17.2.2 you can find Figure 15. This figure displays the validation results from testing antenna-1. The receiver started close to the transmitter, slowly sweeping a 180 degree angle around it. We then continued sweeping while backing away from the transmitter until we started losing packets. Finally, we kept sweeping and got closer to the transmitter again. This test showed us that there was little to no directionality from antenna-1. The distance from the transmitter affected the RSSI as expected, but the direction the antenna pointed did not affect the RSSI. Also, the antenna did not have as much range as expected. Packet loss became an issue when the range reached only 60 meters.

18.3.3. Antenna-2 Test

In Figure 32 below is the validation results from testing antenna-2. It was expected that this highly directional antenna would have better results than antenna-1, however the direction the antenna was pointing did not affect the RSSI. As you can see in figure 32 and 33, the distance from the transmitter affected the RSSI but the sweeping did not. When the antenna moved from being pointed directly at the transmitter to 90 degrees from it, the RSSI stayed consistent.

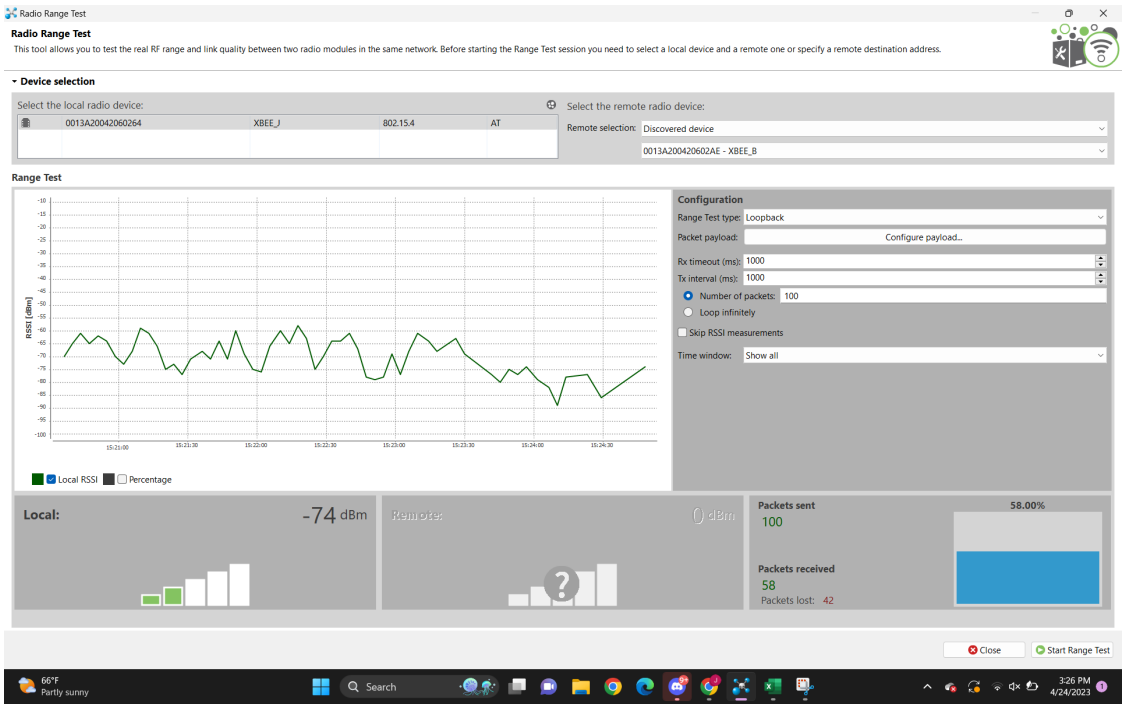


Figure 32: Antenna-2 test at 90m

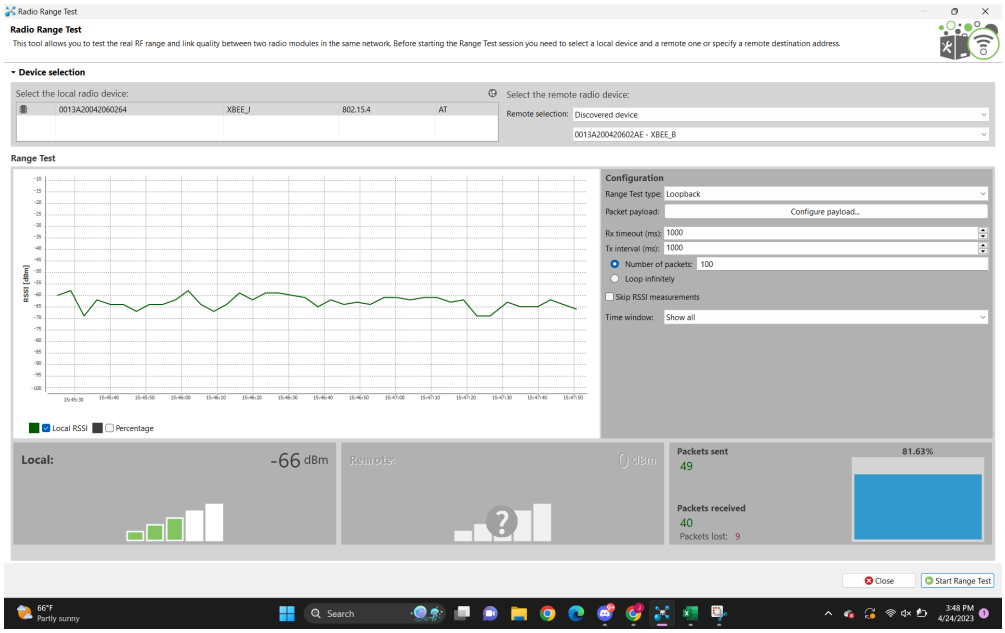


Figure 33: Antenna-2 test at 20m

18.5 Conclusion

In conclusion, the antenna for the system went through multiple design ideas and ended up being purchased. Even though an industry-made antenna was purchased, the subsystem was still unable to perform its duty of changing the RSSI value to maximize when pointed at

the transmitter. Due to our testing, we determined that the issues did not lie with the antenna, but with the Xbee radios.

19. Receiver Stepper Motor Subsystem Report

19.1. Subsystem Introduction

The function of the Stepper Motor Subsystem is to spin the receiver and antenna so that the antenna could sweep the area accurately and quickly. In order to do this, the receiver PCB was placed inside a 3D printed box that sat on top of a 12V stepper motor and was clamped to it using a 3D printed coupler. The stepper motor was driven by a stepper motor driver IC that received control signals from the MCU. The motor would spin around checking different points and then return to its original position.

19.2. Subsystem Details

19.2.1. Motor characteristics

The stepper motor chosen is a hybrid bipolar 12V stepper motor. It has a current rating of 350mA and has 200 steps per revolution before microstepping. This means it has a full step angle of 1.8 degrees. It has a holding torque of 28 oz-in, which should be plenty of torque to rotate up to 2.5 pounds. It is driven by a stepper motor driver IC with up to 1/8th microstepping, allowing for peak accuracy.

19.2.2. PCB design

As mentioned above, the stepper motor was connected to the PCB where it was driven by a stepper motor driver IC. This driver received power to drive the motor through the 12V linear regulator, as well as power for the logic of the IC through a 12V-3.3V buck converter. It was connected to 6 different GPIO pins on the MCU that sent control signals back and forth between the driver and the MCU. This is how the stepper motor was controlled. In Figure 34 below, you can see the stepper motor’s connection to the board, the stepper motor driver, and its connections to the MCU.

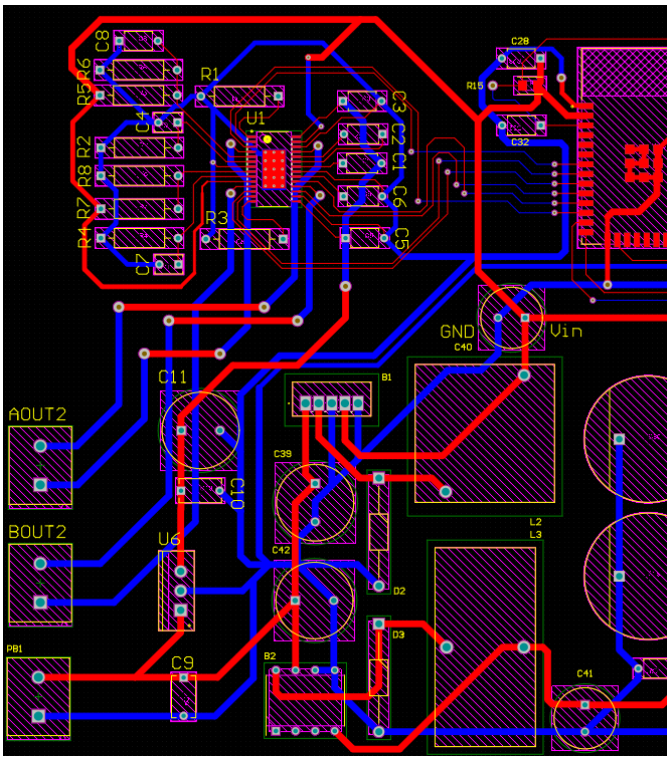


Figure 34: Stepper Motor Subsystem part of the receiver PCB

19.2.3. Stepper Motor Code

The code to control the stepper motor makes up most of the MCU code and is hundreds of lines long. To see the complete code, visit our GitHub. In Figure 35 below, you can find an integral part of the code, the step function. This function is called all throughout the code and is how the motor takes one step. The function takes in a step direction, size, and speed.

```
36  void step(bool step_direction, int step_speed, bool USM0, bool USM1) {
37
38      /* Step function instructions
39
40      direction
41          1 => clockwise
42          0 => counter_clockwise
43
44      speed
45          ms per period
46
47      size
48          00  => full step
49          01  => half step
50          10  => quarter step
51          11  => eighth step
52
53      */
54
55      //Set delay based on period
56      int pulse = step_speed / 2;
57
58      //Set direction
59      gpio_set_level(DIR_GPIO, step_direction);
60
61      //Set step size
62      gpio_set_level(USM0_GPIO, USM0);
63      gpio_set_level(USM1_GPIO, USM1);
64
65      //step once
66
67      //set step high
68      gpio_set_level(STEP_GPIO, 1);
69      //set pulse size (1 = 1ms)
70      vTaskDelay(pulse / portTICK_PERIOD_MS);
71
72      //set step low
73          gpio_set_level(STEP_GPIO, 0);
74      //set pulse size (1000 = 1s)
75      vTaskDelay(pulse / portTICK_PERIOD_MS);
76
77  }
```

Figure 35: Step function inside MCU code

Once the receiver receives a signal from the database telling it to sweep, the first thing the stepper motor will do is check all the poles to determine the pole with the strongest signal. The system starts pointing north so from there, the system waits to get the RSSI value and then moves to the next pole. After checking North, East, South, West, it will rotate counterclockwise back to north in order to keep the power cords untangled. The stepper motor will move to each pole 3 times, waiting for a few seconds to check RSSI, and take the average RSSI at each pole. This is to make sure that the pole that has the strongest signal is chosen. Once a pole has been chosen, the motor will move to that pole and do a sweep across it. The motor then checks 10 different directions starting from 45 degrees left of the pole to 45 degrees right of it. The size of the sweep can be changed within the code for more accuracy, as well as the amount of points it checks in the sweep. Once the system is done gathering data, the motor moves back to north.

19.3. Subsystem Validation

19.3.1. Stepper Motor Precision

In our validation plan, it says that the stepper motor must be able to move a 2.5 lb load with precision of +/- 5 degrees. In order to validate this, a 2.5 pound load and a 17 inch long antenna was placed on top of the stepper motor. It was then spun 360 degrees clockwise followed by 360 counter clockwise. It was then measured how far off it was at different speeds. That test was run twice and then again with a load of 1lb, which is the weight of our actual system. The data from that test is displayed below in Figure 36.

2.5 lbs		
Speed (steps pe	Inches Off	Percent Off
5	0.09375	0.08776927009
10	2	1.872411095
15	2.1875	2.047949635
20	4	3.74482219
5	0.125	0.1170256934
10	2.25	2.106462482
15	2	1.872411095
20	5.5	5.149130512
1lb		
Speed	Inches Off	
5	0.125	0.1170256934
10	0.0625	0.05851284672
15	2.25	2.106462482
20	4.5	4.212924964

Figure 36: Stepper Motor Precision Test

19.3.2. Stepper Motor Speed

In our validation plan, it says the stepper motor must be able to move a 2.5lb load at a speed of 20 steps per second. In order to validate this, a 2.5 pound load was placed on top of the stepper motor. It was then spun 360 degrees clockwise followed by 360 counter clockwise. It was then measured how long it took the motor to complete its spins. That test was again with a load of 1lb, which is the weight of our actual system. The data from that test is displayed below in Figure 37.

Speed	Time is should take	Time it takes	%error
2.5 lbs			
5	80	80.87	1.0875
10	40	40.16	0.4
15	26.67	26.54	0.4874390701
20	20	19.57	2.15
25	16	infinity	100
1lb			
5	80	80.38	0.475
10	40	40.23	0.575
15	26.67	26.82	0.5624296963
20	20	20.09	0.45
25	16	infinity	100

Figure 37: Stepper Motor Speed Test

19.4. Conclusion

The stepper motor worked flawlessly during our full integration testing, despite just barely passing our validation requirements. It moved quickly and precisely just as needed. The code is relatively straightforward and the stepper motor driver IC served its purpose perfectly.

20. Receiver MCU Subsystem Report

20.1. Subsystem Introduction

The receiver's MCU subsystem is basically the data center of the receiver. This is the brain of the project, where all data is collected to be transmitted to the WiFi module to send to the database for calculations. The ESP32 also handles the stepper motor's movement as previously mentioned in section 19.

20.2. Subsystem Details

In this subsystem section, there is the communication of three main modules. Originally, as seen in the figure below, there were two other modules but unfortunately they could not be implemented.

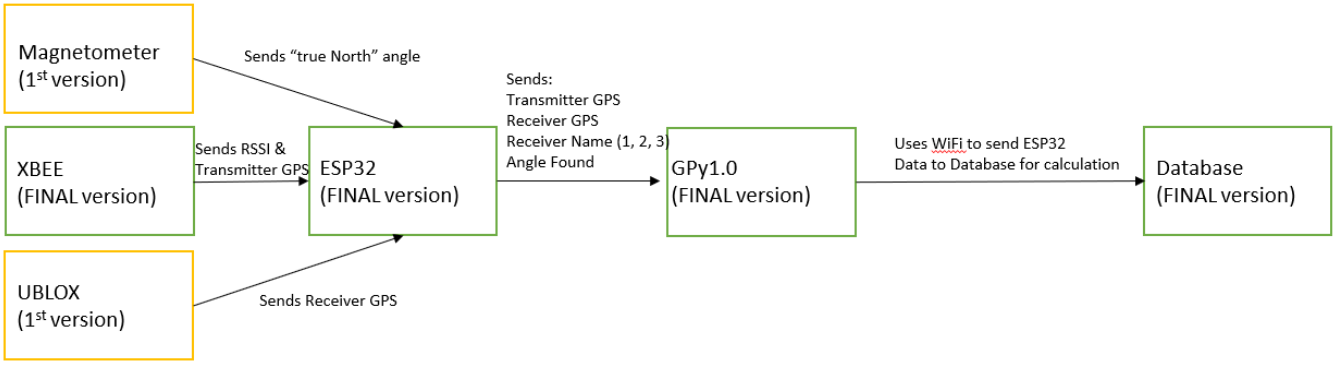
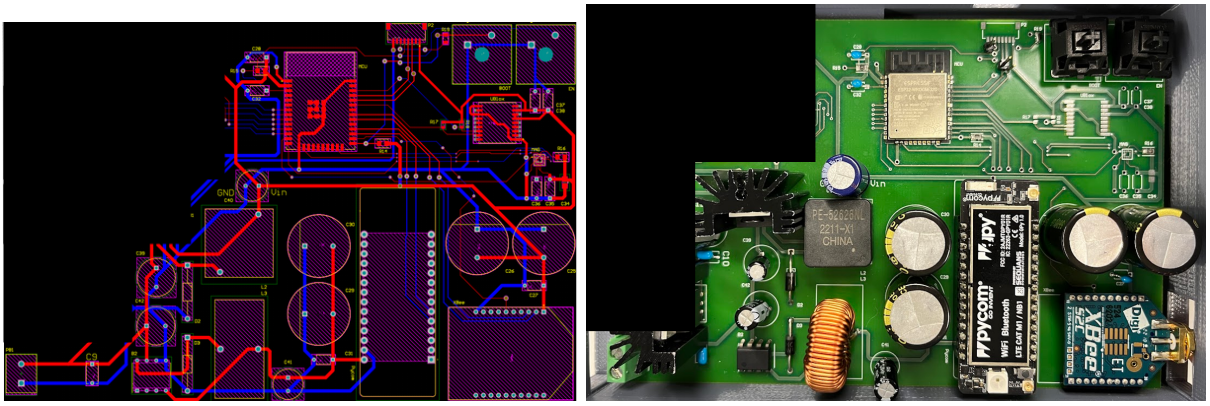


Figure 38: Overview of ESP32 Subsystem

The boxes in green are the current modules on the PCB board. Also, as previously mentioned, all modules are on their own power subsystems. The next figure depicts the PCB layout, with the following picture that has the pieces completely soldered on.



Figures 39: PCB Design & Implementation

The ESP32 connects to the Pycom and XBEE through UART connections. It needs to successfully receive data from the Pycom (start signal), XBEE (the transmitter's GPS coordinates), the stepper motor (strongest signal angle from true North), and it needs additionally to add the receiver's name and GPS coordinates. It sends all this information to the Pycom, excluding the start signal, in a string separated by commas so the database can receive, separate, calculate, and display the data on the GUI successfully.

20.3. Subsystem Validation

Once the subsystem lost the two modules and RSSI was not used due to the XBEE's lack of directionality seen in section 17, the MCU worked correctly. The TX0 and RX0 were able to flash code from the Boot and Enable sequence, and every receiver was able download their required code. Overall the code was largely similar for all three receivers, where the GPS Coordinates for the receiver and its name were changed; this code can be seen in the system's GitHub online.

The power validation was previously discussed in section 16.3.2.; overall, this power subsystem worked correctly and as expected for the MCU and modules connected. The timing for transmission time is seen in section 22.3, where the database is able to receive data within the time required.

```
String sig_angle;
sig_angle = String(transmitter_direction);
/*
Pycom wanted to be sent one long string,
separated by commas. Arduino does not allow for
more than one value sent at a time without
stringing them together with '+'
*/
String send_string;
String comma;
comma = ",";
send_string = ReceiverName + comma + rec2 + comma +  XBeeReceived + comma + sig_angle;
Serial.println("Rssi");
//MySerial1.print(best_rssi);
Serial.println(get_rssi());
Serial.println("receiver number");
Serial.println(send_string);
MySerial1.print(send_string);
```

Figure 40: Print Out of ESP32 Code

Above is a small snippet of code from the ESP32's main loop. As seen, the sequence that the database chose goes from receiver's name (1, 2, or 3), the GPS coordinates of the receiver, the transmitter's coordinates, and the signal angle found. These were sent successfully to the Pycom over WiFi to the database where the GUI displayed the map with calculated error percentage.

20.4. Subsystem Diagnostic and Mitigation / Proof of Concept

Overall this subsystem had some modifications. First, the UBLOX was not implemented. To remedy this, the project agreed that hardcoding the GPS coordinate of the receiver into the main loop would be the best way to complete error correction. Secondly, the magnetometer was not implemented. This module was to find true North so the antennas could hypothetically be oriented at any pole, but still find true North for the strongest signal. To mitigate this, the system decided the best decision was to orientate the receivers the same way and consider this to be true North.

Lastly, for the XBEE, as extensively discussed in section 17, specifically section 17.2.3, the RSSI value was not implemented. The RSSI depends heavily on the module the system read from, the XBEE, however the tests performed showed there was not enough specific directionality from the module to be used in the test. Due to this, for demonstration purposes, the RSSI value was determined through a dictionary implemented in the code. This dictionary would not be used for a real world application; however, in order to show that the system sweeps, finds a correct angle, transmits the data, and updates the GUI, this dictionary was necessary. This can be found in the system's GitHub in the receiver code.

20.5. Subsystem Conclusion

Overall, this subsystem was an overall challenge, with many improvements and revisions made to the circuit and code along the way. Due to time constraints and working with other subsystems to have a complete project, this subsystem became focused on functionality. Functionality depended heavily on the transmission of data to the database, being able to communicate with the transmitter, and stepper motor movement. This subsystem completes these three tasks completely.

However, adding the GPS and magnetometer modules would allow for more automatic processing of data and less manual user input; thus, if this project was worked on more, those modules would definitely increase the satisfaction of the user. The biggest downfall of this subsystem was the radio module. Unfortunately due to the hardware chosen, there are not many fixes that could be made without just updating the module itself to a more reliable radio with a substantial increase in directionality.

21. Transmitter Subsystem Report

21.1. Subsystem Introduction

The Transmitter subsystem was formed to provide a strong and reliable signal for the receivers to look for. The transmitter is not what is actually being sold to customers but is a way for us to show off the system's RF triangulation. Many of the modules that went into the final design of the transmitter were based on the principle of cheap but reliable. The details below explain why this is the case.

21.2. Subsystem Details

The Transmitter was created to provide a signal for the receivers to look for. The message the signal sent was not important but we designed the transmitter to provide its GPS coordinates through this message to the receivers. This would allow the GUI to compare the triangulated coordinates with a true location and generate an error percentage. From there we can figure out how to improve on the design of the receivers to make them more accurate.

21.2.1. Component Selection

The design goal as stated above was to reliably provide a strong signal for a low cost system. Thus, the parts picked must follow these guidelines. The MCU we decided on was the MSP430 which has a low level computing power while providing the connection protocols needed to make the transmitter system work as a whole.

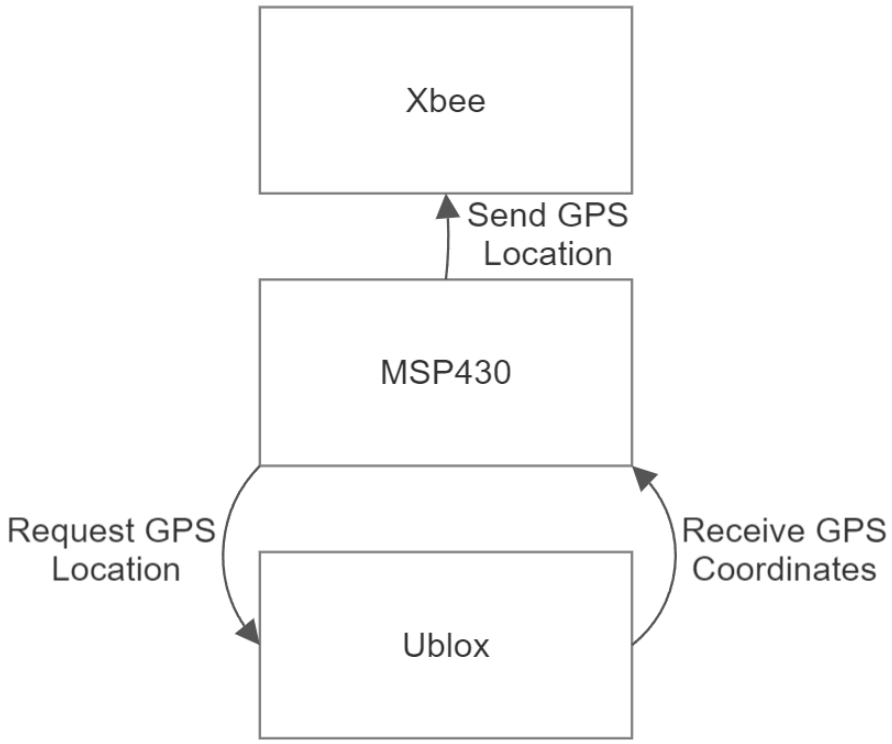


Figure 41: Transmitter Block Diagram First Iteration

As seen above in Figure 41, the MSP430 was originally planned to set up an I2C connection with the Ublox module to get the GPS location. This however proved to be a challenge for the MSP430 which was challenging enough to understand how all the registers can be manipulated. An I2C connection could not be made with the ublox module so we decided to move on and try to finish the system to provide the reliable signal as soon as possible for the receivers.

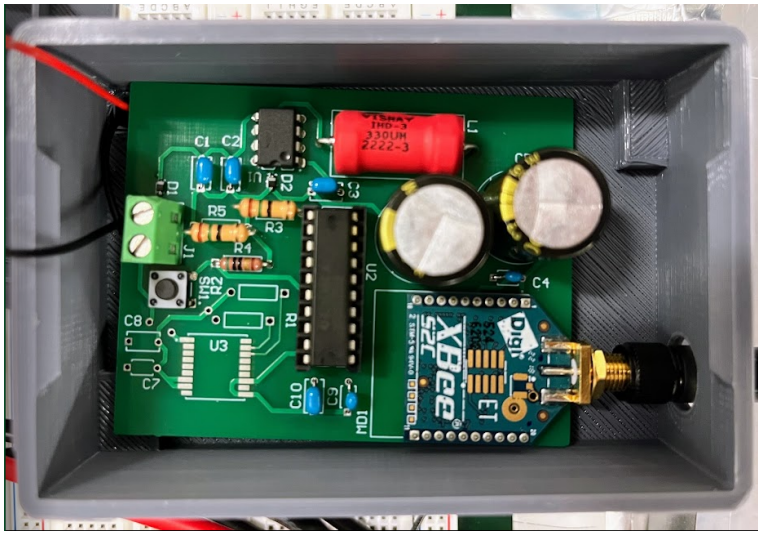


Figure 42: Final Transmitter Design

In theory the GPS module would work with a better MCU in place with better debugging and computational power. With the GPS location hard coded in the MCU we can run the code to send

each character at a 9600 baud rate through an omnidirectional antenna to the receivers.

21.3. Subsystem Validation

As shown in the ESP32 validation Figure 19, the transmitter was able to send a signal with a given message at a known frequency. Looking at the transmitter the only real concern is can this omnidirectional antenna reach the one hundred meter needed for the system. Using the path to the Bonfire Memorial we were able to measure the distance at one hundred meters and found no packet loss. This continued all the way to one hundred fifty meters before we started to lose packets

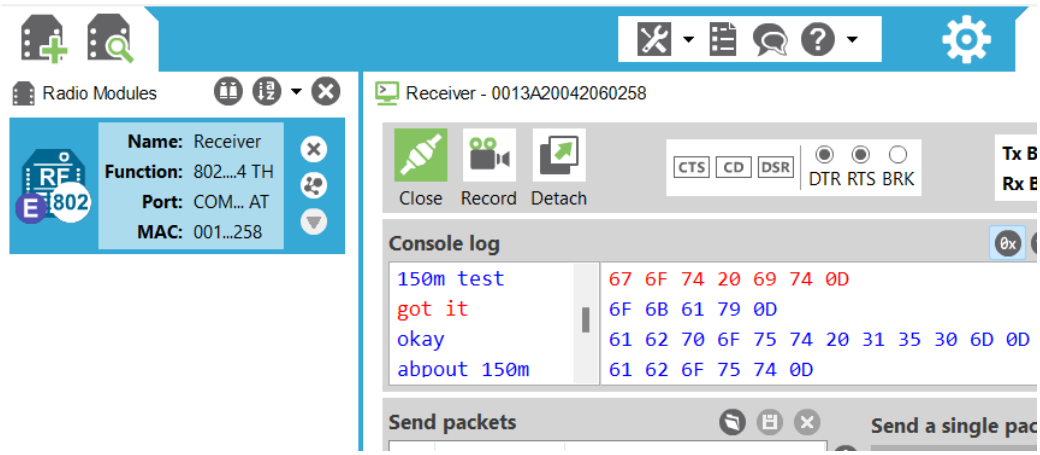


Figure 43: Transmitter Range Test

Using this test combined with the ESP32 validation we can see that the transmitter was able to provide a fast and reliable signal to the receivers to debug their systems with.

21.5. Subsystem Conclusion

Although we didn't use the ublox module to gain a constant updated gps location, we were still able to show the functionality of the receiver system using a hardcoded coordinate system. In the future this transmitter can be improved upon by making it smaller and having a working gps module.

22. GUI and Database Subsystem Report

22.1. Subsystem Introduction

The GUI and Database subsystem was created to allow us to visually see how the receivers were positioned and how the transmitter location was calculated by creating the intersection of the 3 strongest signal lines from the starting point of the receivers. The database allows us to have a WiFi connection from the receivers to the database to upload the data itself. The details and reasoning are included.

22.2. Subsystem Details

The GUI and Database subsystem was created to allow us to visually see how the

receivers were positioned and how the transmitter location was calculated by creating the intersection of the 3 strongest signal lines from the starting point of the receivers. The database allows us to have a WiFi connection from the receivers to the database to upload the data itself. The details and reasoning are included.

22.2.1. Information Storage Infrastructure

The data gathered by the Radio Frequency receivers is stored online in a PostgreSQL database. The PostgreSQL database is hosted on Amazon Web Services (AWS) which provides support for PostgreSQL in its Relational Database Service (RDS). This service is free depending on the services you selected, more features can be added for the paid versions of the database.

The database is managed through the pgAdmin4 and VSCode's Postgres extension, it provides a graphical interface to easily visualize, manipulate database properties, and send queries.

22.2.2. Information Storage Design

For the database design I made individual tables for how I believe the data will be coming in. This will later be changed to be a single table where each run has a unique ID associated with it to allow visualization through a single table. Each table has 7 columns, the columns hold the receiver data such as its latitude and longitude, its name, its strongest signal angle, and the time the data got last updated. Currently I am using the most ideal datatypes of smallint and double precision types

	recname [PK] smallint	rec_lat double precision	rec_long double precision	trans_lat double precision	trans_long double precision	sig_angle double precision	time_last_updated text
1	1	30.62486	-96.33467	30.62496	-96.33383	359.3	(2023, 4, 10, 10, 44, 0, 155130, None)
2	2	30.62527	-96.3333	30.62496	-96.33383	234.1	(2023, 4, 10, 10, 44, 0, 182234, None)
3	3	30.6242	-96.33306	30.62496	-96.33383	139.62	(2023, 4, 10, 10, 44, 0, 139972, None)

Table 2: PostgreSQL Table

The first column is the receiver name that contains a small int consisting of either 1,2,3. These are all unique and can not have the same one in there twice, this also is a "smallint" type that can hold 2 bytes or the range (-32,768 to 32,767) which is the most efficient datatype I can use for this.

All the other columns in the table are stored as double precision or more commonly known as floats, which allow a decimal number to be inserted there. This is needed for the latitude and longitude values since one degree of either is equal to around 69 miles, this data needs to be precise. For the angle I made it a float as well to allow for higher precision. The last column is stored as text and is in a tuple in the form of (year, month, day, hour, minute, second, microsecond, timezone), this was the most effective way to store this information as the time library that Micropython uses, inside the Pycom, returned the data in this format.

22.2.3. Software Infrastructure Stack

The data persists in the database and can be accessed at any point allowing users to quickly access the data gathered by RF Triangulation run easily. The GUI was built using PyQt6, Python Imaging Library, Python's math library, geopy ,an open source satellite imaging library that utilizes maptiler, UTM, and sqlalchemy all hosted locally on my machine.

Python was the primary programming language used in this project since I was most comfortable with the language and it allowed me to use the vast open source libraries that python is built upon.

- SQLalchemy allows connections to the databases to access their data.
- Satellite images using maptiler allowed an input of two coordinates of the smallest coordinate and the largest coordinate of the receivers to build an image that contains a satellite image of where the receivers are located.
- The imaging library allowed the system to convert the satellite image to an image that was used within the GUI, as well as drawing the location of the receivers and the strongest signal line.
- UTM and geopy were used to convert the coordinates of the receivers with a given grid to an XY plane that was very useful in being able to place where the receivers were and the lines.
- Math library was used to calculate the lines from the given angles and then using the line formulas to get the individual lines of each of them and from there using the formula for intersections of lines to get the area formed from 3 lines. With that using the 3 points to find the centroid of the given area to give an approximate for the calculated transmitter location
- Geopy was then used to display the size of the grid containing all of the receivers as well as calculating the error between the expected and calculated transmitter location

22.2.4. Website Design Interface

The GUI homepage does not display any starting information but rather just the setup of how the data will be displayed. Once the start button is pressed a start signal is sent to the microcontrollers to start the motors and begin the sweep to find the strongest signal, after this is complete the data will be displayed in each of the corresponding boxes and the map will be displayed as well showing the location of the receivers and the calculated transmitter location.

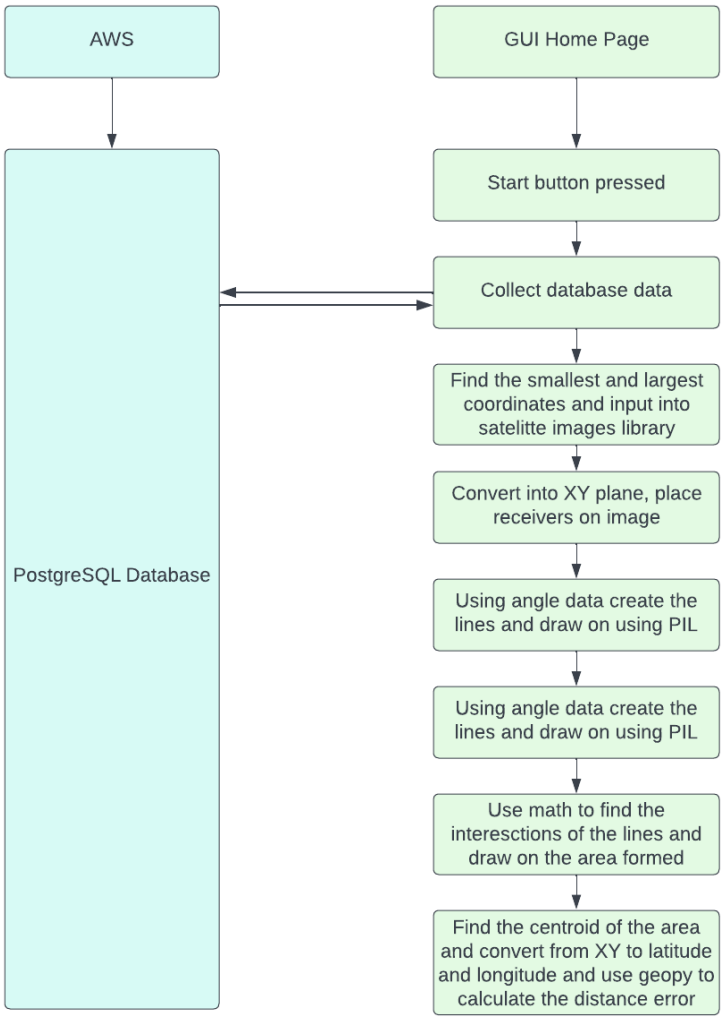


Figure 44: GUI Design Schematic

22.2.5. GUI Design

The following are screenshots of the GUI to help provide a snapshot of the current state of the subsystem. They show what was described in the diagram of the previous figure.

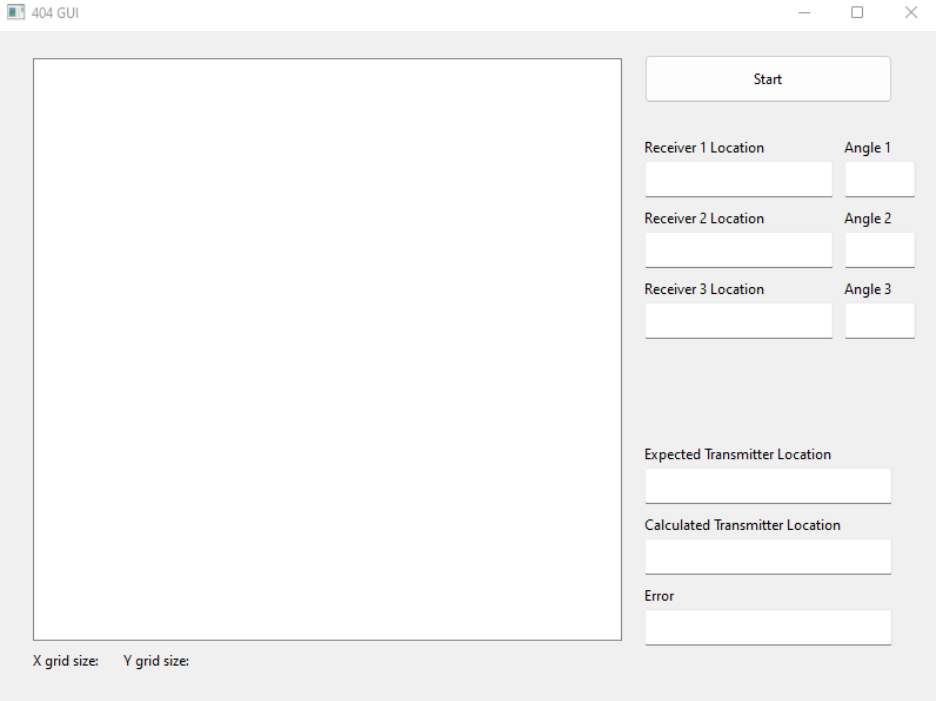


Figure 45: GUI Layout

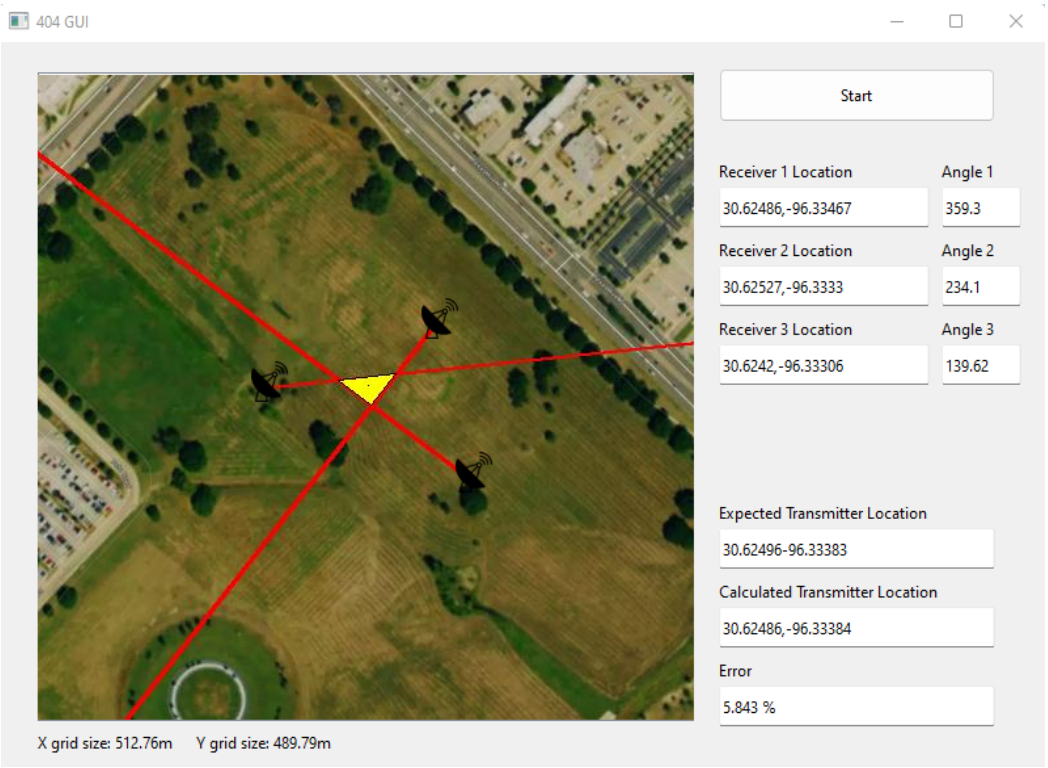


Figure 46: GUI Result

22.3. Subsystem Validation

The PostgreSQL database is hosted on enterprise servers managed by Amazon. Amazon automatically provides redundancy for information security and decreased downtime in the event of a failure. The GUI was validated with user testing and confirming the input data versus the displayed data, this allowed me to see that the math and data was correct. I performed timing as well for a required run to make sure it met the specified requirements.

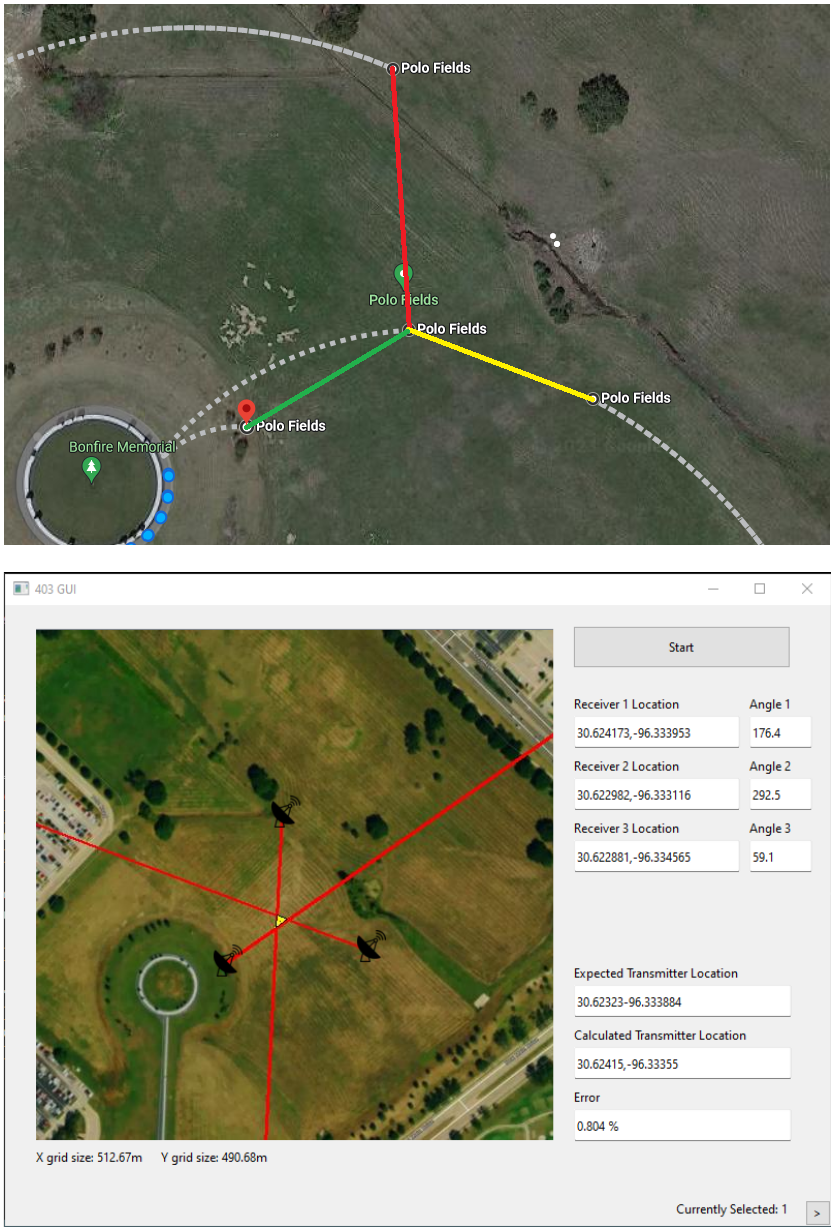


Figure 47: GUI Testing validation

```
Time taken for GUI to load:  2.237
RAM used in process:  105.211  Mb
Memory percent is :  0.644 %
```

```
Time taken for GUI to load:  1.996
RAM used in process:  106.277  Mb
Memory percent is :  0.651 %
```

Figure 48: GUI Timing and memory output

From this we can see that all the required timings are met for being less than 15 seconds for load time.

The error calculations were no more than 10% for the data sets based on the search area and the found triangulation area.

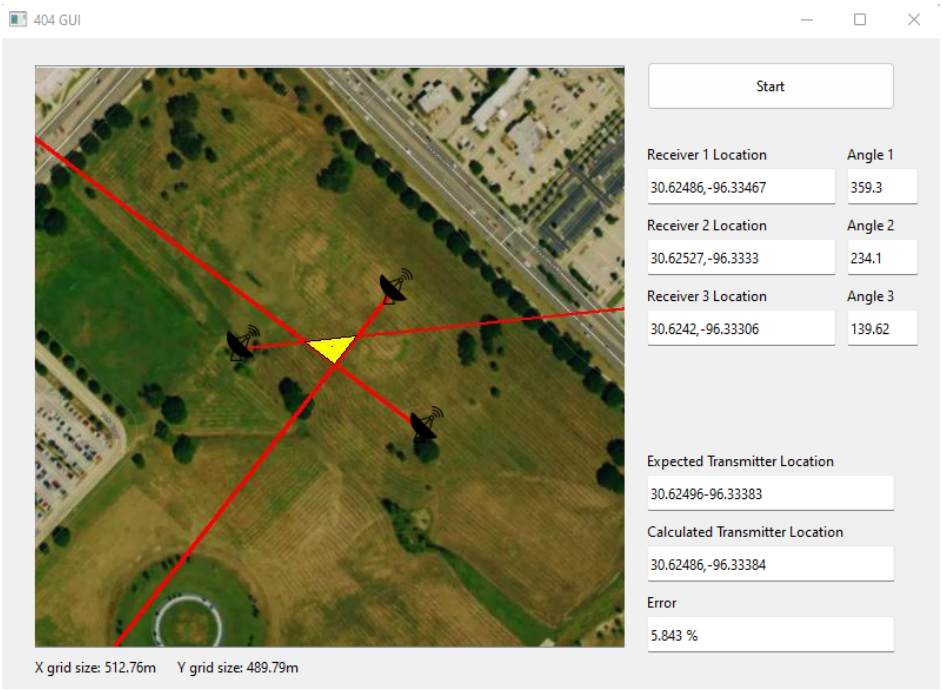


Figure 49: GUI Error Calculations

22.5. Subsystem Conclusion

The GUI and Database subsystem currently contains all of the required functionality and is fully validated with all requirements being met for different scenarios.

23. Conclusion

The RF Triangulation System proved to be a right step in decreasing the error of tracking an object with triangulation. This project could have been put together a bit more quickly to allow for some more exploration into an alternative to RSSI; however, the implementation of a transmitter that successfully transmits a signal with its coordinates, the receivers that sweep an area, and a GUI that displays all information correctly is a difficult task in itself.

23.1. Learnings

This project provided an essential building block for the members involved. The development and the assignment of subsystems allowed for each member to develop their own specific skills.

Teamwork was essential in this project; for most hardware validation, especially on the receiver side, the multiple members were involved at every stage. This project required many long days in the laboratory to develop the project as much as possible with the limited knowledge developed throughout the course of the project. Many members sacrificed personal hours to help other members, while others took on the challenge of deciphering datasheets for components. This project was, through and through, a group effort and could not have been completed without all members of the team.

For technical learning, the project required a strong understanding in reading datasheets. Most problems stemmed from the trickiness of manufacturers with unclear information presented. After this, creating the circuits specified in datasheets in Altium was critical. Designing, assembling, and testing a circuit board is an extremely time consuming process. Testing usually meant problems for this project, which had to be mitigated in some way. Troubleshooting and critical thinking were the major tools for these errors.

On the software side, learning how to code the respective microcontrollers was critical. Along with this, the database needed to be able to take in the data that the MCUs would send on UART. Being able to read in, let alone do calculations and data display took many hours.

Overall, the presentations on the system will be the most significant tool used in future careers. System presentations force engineers to look at the reality of their subsystem and be able to confidently display their knowledge of the working side, the errors, and how they plan to fix for the future. Engineers need to be able to coherently display the purpose of their project, and allow for inputs from both peers and superiors.

24. Validation

Paragraph	Test Name	Success Criteria	Methodology	Status	Responsible Engineer
8.2.1.1.	Battery Operational Time	System operates continuously on battery power for 30 minutes.	System transmitter and receivers are put into default operational state and left to run for 30 minutes.	TESTED	Full Team
8.2.1.3.	Operational Search Range	The receiver shall have an accuracy of 90% on transmitted packages at 100m.	Measure the amount of packets sent from transmitter and the amount received correctly by receiver at 100m in default operational state for 5 minutes.	FAILED	Kathleen Hutchinson
8.2.1.4. 8..2.1.7.	System Connection	The time it takes to connect and transmit data between the GUI, transmitter, and receiver shall be 30s.	System runs start signal from application to receivers and back to application with data and timed using timer function into code.	TESTED	Brandon Stokes Kathleen Hutchinson
8.2.1.5.	Receiver Data Usage	The receiver shall not take up more than 1GB of RAM on user's computer.	Measure the amount of data used by system for 5 minute default transmitting state.	TESTED	Brandon Stokes
8.2.1.6.	Database Duplicates	The Database should not be able to take in duplicate values or values that do not match the expected criteria	Attempt to insert a new row with wrong data and duplicate receiver name to test the primary key	TESTED	Brandon Stokes
8.2.1.8. 8.2.1.9.	Motor Speed	The motor can turn with speed of 20 steps per second while spinning a 2.5 pound load.	After connecting a load of 2.5 pounds (well under the weight of the system), spin the motor different speeds to find max speed.	TESTED	Jack Parkinson
8.2.2.1.	Antenna Movement Precision	The motor can turn with precision of +/- 5 degree of accuracy while spinning a 2.5 pound load.	After connecting a load of 2.5 pounds (well under the weight of the system), spin the motor different amounts of steps and direction and verify that it is at the appropriate angle.	TESTED	Jack Parkinson
8.2.2.2.	Weight	The weight on top of the motor shall be at max 2.5lbs (1134g).	Measure receiver unit with digital scale.	TESTED	Jack Parkinson
8.2.2.3.	Mounting/Housing	Receiver housing shall be durable enough to not fall apart while placed on top of the motor.	Place housing on top of motor and run system test for 3 minutes.	TESTED	Kathleen Hutchinson Jack Parkinson
8.2.3.1.	Input Voltage, Transmitter	The input voltage level for the MSP430 shall be 3.3V	Use E-Load to validate line and load regulation	TESTED	Josh Broyles
8.2.3.1.	Input Voltage, Receiver	The input voltage level for the ESP32 shall be 3.3V	Use E-Load to validate line and load regulation	TESTED	Kathleen Hutchinson
8.2.3.1.	Input Voltage, Pycom	The input voltage level for the GPy shall be 5V.	Use E-Load to validate line and load regulation	TESTED	Kathleen Hutchinson
8.2.3.1.	Input Voltage, Stepper Motor	The input voltage level for the DRV811PWPR (motor driver) shall be at least 10V	Use E-Load to validate line and load regulation	TESTED	Jack Parkinson
9.2.2.	WiFi Stability	The wifi hotspot does not drop more than 1 time per 5 minutes and shall reconnect within 20s.	System is put into default operational state (tracking transmitter) and left to run for 30 minutes while Pycom tracks wifi connection	TESTED	Brandon Stokes Kathleen Hutchinson
N/A	Full System Demo	A user of system is able to accurately track the transmitter in an open space with a positional error of less than <10%.	System runs start signal to receivers which tracktransmitter's strongest signal with the motor, both stationary and moving, which sends data to GUI that outputs an accurate map with an error of <10% calculated through GPS points.	TESTED	Full Team

The information above displays the validation tests ran for certain sections of this document. The only unpassable test was the XBEE's inability to be reliable in 90% packet receiving, along with its RSSI data.

25. Execution Plan

Below depicts the execution plan of this project. See key, many parts ended up being behind schedule.

	1/24/23	1/31/23	2/7/23	2/14/23	2/21/23	2/28/23	3/7/23	3/14/23	3/21/23	3/28/23	4/4/23	4/11/23	4/18/23	4/23/23	4/30/23
Ring out PCB															
Test Radio Distance															
Finish Programing MCU															
Assemble PCB															
Validate PCB															
Validate Messages to Receivers															
Finalize Schematic/PCB Design															
Order/Print PCB															
Program Modules															
Validate PCB															
Finish ESP32															
Connect Antenna															
Finalize Antenna Design															
Order/ Build Antenna															
Test Antenna															
Test Motor Controller															
Database to Single Table															
Rework out of bounds situation															
Finish Pycom															
Add Error checking to data															
Integrate Reciever Modules															
Test Inter-Communication															
Complete System Validation															
Final Demo															
Final Report															

Key
Not Started
In Progress
Completed
Behind Schedule