Satellite Antenna Tracking Mount

Hunter Britton
Michael Raabe
David Santos

Concept of Operations

REVISION – 2
25 April 2023

Concept of Operations
for
Satellite Antenna Tracking Mount

Team 35

Approved by:

_____
Hunter Britton Date

_____
Prof. Kalafatis Date

_____
Pranav Dhulipala Date

Change Record

Rev.	Date	Originator	Approvals	Description
1.0	2/7/2023	David Santos		Draft Release
2.0	4/28/2023	David Santos		Added background importance Updated Block Diagram Added System Diagram Added deliverables to scope Added user interfacing to operation description Added User Flow Diagram Added modes of operation description Added improvements from traditional systems Added more impacts

Table of Contents

List of Figures 4

1. Executive Summary **5**

2. Introduction **5**
- 2.1. Background 5
- 2.2. Overview 6
- 2.3. Referenced Documents and Standards 7

3. Operating Concept **7**
- 3.1. Scope 7
- 3.2. Operational Description and Constraints 7
- 3.3. System Description 8
- 3.4. Modes of Operations 8
- 3.5. Users 8
- 3.6. Support 9

4. Scenario(s) **9**
- 4.1. Emergency 9
- 4.2. Recreation 9
- 4.3. Competition 9

5. Analysis **9**
- 5.1. Summary of Proposed Improvements 9
- 5.2. Disadvantages and Limitations 9
- 5.3. Alternatives 10
- 5.4. Impact 10

List of Figures

Figure 1. System Block Diagram 6
Figure 2. Graphical System Diagram 7
Figure 3. Functional User Flow Diagram 8

1. Executive Summary

Amateur radio systems require specialized equipment and software to operate. Rather than manually adjusting the antenna with dials or computer software, our objective is to provide remote operation that will control the antenna rotator and display data for communication. The Satellite Antenna Tracking Mount will address this issue with a web application and microcontroller that will interface with the mount over the wireless internet. SATM will also provide secure user authentication via hashing account data. This will allow for low-cost remote operation and web browser accessibility.

2. Introduction

The purpose of this document is to provide an overview of the Satellite Antenna Tracking Mount (SATM) system. The SATM is designed to accurately target and track satellites as they orbit the Earth. The user interface allows for the selection of the desired satellite for communication, after which the SATM will perform automated tracking to ensure the antenna is correctly pointed at the selected satellite as it passes overhead at any azimuth or elevation. This feature enables the user to concentrate solely on their communication without having to manually track the satellite.

2.1. Background

In amateur radio, signal strength is of great importance as it determines ability to effectively communicate with other radio operators. They use their antenna and software defined radio (SDR) setups to exchange messages non-commercially for recreation,contesting, and emergencies. The majority of legacy systems for moving satellite antennas are semi-manually controlled, with a user having to press a switch to move the antennas in different directions or input data for software. This outdated system tracks satellites across the sky, but requires the operator to be physically present with a laptop. Few strides have been taken to provide control wirelessly and even fewer allow access through web browsers. The SATM intends to replace this manual control system scheme with an automated one, where a Satellite can be selected out of a set catalog, and the tracking mount seamlessly follows it in the sky. This automatic system would allow for more flexibility when operating the antenna, as well as include less error since the data would be instantly updated.

2.2. Overview

Figure 1: Satellite Antenna Tracking Mount Block Diagram

The SATM system is designed to automate the process of tracking satellites as they traverse the overhead region, eliminating the requirement for manual tracking by the operator using azimuth and elevation controls. The location data of the targeted satellite will be obtained from a pre-existing database that will be continuously updated to ensure accurate and persistent tracking of the satellite. The system is equipped with a web-based user interface, allowing the operator to input the desired satellite to be tracked. This information will be transmitted to the microcontroller, which will drive the motors within the antenna mount, enabling continuous tracking until the satellite descends below the horizon.

2.3. Referenced Documents and Standards

- IEEE Wi-Fi communication standards: IEEE 802.11
- C95.1-2005 - IEEE Standard for Safety Levels with Respect to Human Exposure to Radio Frequency Electromagnetic Fields, 3 kHz to 300 GHz
- CCSDS 502.0-B-2 - Orbit Data System Standards
- FCC 97.213 - Telecommand of an amateur station
- CFR P97T47 - Amateur Radio Service

3. Operating Concept

Figure 2: Graphical System Diagram

3.1. Scope

The SATM will provide wireless connectivity to amateur radio operators that seek to enhance their current manual setups. The final product will include a small piece of additional hardware that includes a microcontroller and motor drivers and a browser-accessible web application. They will initially configure the device to provide the necessary information to the web application such as the model of rotator they are using and its coordinates. When the operator accesses the website on any browser, they will login and select a satellite. The next thing they will see is the relative location of the satellite along with the orientation of the antenna in degrees. The microcontroller will handle the intermediate processes that are usually completed by the user or computer software. These include calculating orbital paths, as well as setting the motor to the appropriate motion sequence. The SATM will immediately benefit operators, but may also be used for other IOT projects with slight alterations such as telescope mounts or weaponry.

3.2. Operational Description and Constraints

The Satellite Antenna Tracking Mount (SATM) is intended to be used by amateur radio enthusiasts to be able to communicate with satellites without having to manually control the azimuth and elevation controls to track the satellite. The SATM system will take the information from a pre-existing database of satellite locations and use it to track a satellite as it passes overhead. The basic user flow is shown below:

Figure 3: Functional User Flow Diagram

The resulting constraints are from this operational description are as follows:

- The SATM system must have access to a WIFi connection.
- The SATM system should be placed in an area without obstructions to the sky. This will allow the antenna array to communicate with the satellite.
- The system should be placed on fairly level ground to allow the rotator to track the satellite fully.
- The budget for the whole project is $300 so that limits the components that we will be able to use in the system.
- The external system components must be strong enough to operate in most weather conditions.

3.3. System Description

- Web Application: The user-friendly web application enables the operator to select the desired satellite for communication. The application can be accessed remotely with an internet connection, and it will retrieve satellite location data from a constantly updated database. This information will then be transmitted to the microcontroller.
- Microcontroller: The microcontroller calculates the relative position of the satellite based on the data obtained from the database and drives the motors within the antenna mount to point the antenna towards the satellite.
- Antenna Mount: The antenna mount is equipped with two motors that regulate the azimuth and elevation angles, allowing precise orientation towards the target satellite. The mount is commercially available and will be controlled by the SATM system through the microcontroller. An antenna is mounted on the mount, making it ready for satellite communication.

3.4. Modes of Operations

There will be two main modes of operations: automatic mode and manual mode.

- In the automatic mode, the user will first select a satellite from the database. Then the user will open the control settings and toggle “autotrack”. Once the “start tracking” button is pressed, the device will automatically retrieve the orbital data and send it to the mcu so that it can perform the propagation calculations and control the motors.
- In the manual mode, the user will toggle “autotrack” to the off position. They will then manually enter the azimuth and elevation. The application will then send the data over HTTPS to the MCU where it will control the motors to point the rotator.

3.5. Users

The main users of the system would be David Gent, W5QZ, and the Aggie Radio Club, W5AC. This system should only be used by those who have a ham radio license and call sign due to the fact that this system would be interfacing with antennas and will be transmitting and receiving information that is regulated by the Federal Communications Commission. The level of training for this system would be minimal since the use is mainly for those who have a ham radio license, but for those that are not licensed the basic knowledge of antennas, radio waves, and satellite movements are necessary.

3.6. Support

The support for the SATM will be provided by a user’s manual that includes all modes of operation, as well as schematics and diagrams for the internals of the device. The source code for the device would be provided as well in order for the users to debug any issues that happen in corner cases. The user manual would describe how to operate the SATM, as well as walk the user through the main modes of operation.

4. Scenario(s)

4.1. Emergency

Radios are utilized in a lot of different emergency situations since they are not as limited in their scope and broadcast range as mobile phones. The SATM could potentially be utilized in an emergency situation where the only way to communicate with someone would be via a satellite radio.

A big part of using radio communication is that many areas have a lot of amateur radio stations that are fail resistant, leading to no choke points or overloading from use.

4.2. Recreation

Operators typically use their setups to simply meet other people and form clubs. The remote operation system will increase access for people with limited mobility or insufficient geography conditions like inclement weather or high rf interference. Users will be able to access the system from anywhere as long as they have an internet connection.

4.3. Competition

Amateur radio contesting or radiosport, is an organized activity where operators try to have the most two-way interactions as possible. The SATM will potentially allow anyone to participate in these activities without physically being there.

5. Analysis

5.1. Summary of Proposed Improvements

This system will take a legacy system that is already built that has manual controls, and convert it to a system that can be remote controlled and automatically points the antenna in the direction of the chosen satellite. This allows for a higher level of precision and a better connection to the target over manual controls.

- More flexible than non-remote systems due to browser interface
- Less expensive than other remote systems with open-source hardware
- More Customizable
- Better User Interface

5.2. Disadvantages and Limitations

One major limitation of the system is that it requires you to be connected to wifi to operate, as well as the device being connected to wifi to pull the data for the satellite tracking.

Another disadvantage of the device is that this is specifically designed for the legacy system that is already in place, and cannot be used for different rotators without some modification.

5.3. Alternatives

The other alternatives to this project are to manually control the system by hand, or to find an off the shelf remote controlled system. Both of these alternatives have their own disadvantages. The manual control loses the remote control aspect, and sacrifices the precision from the computer controlled system. The off the shelf component does not directly mount on the legacy system and can cost money, as well as require a license or very specific equipment.

5.4. Impact

The Satellite Antenna Tracking System will allow users to have a better signal strength and overall a better radio operation experience. In regards to the environment, this project could potentially increase the amount of radio signals that are active in the community. The main ethical concern can be related to increasing the amount of signals in the environment, and staying within the IEEE 95 standards.

Satellite Antenna Tracking Mount

Hunter Britton
Michael Raabe
David Santos

INTERFACE CONTROL DOCUMENT

REVISION – 2.0
4 April 2023

Interface Control Document
for
Satellite Antenna Tracking Mount

Prepared by:

_____
Team 35 Date

Approved by:

_____
Hunter Britton Date

_____
Prof. S. Kalafatis Date

_____
Pranav Dhulipala Date

Change Record

Rev.	Date	Originator	Approvals	Description
1.0	1/21/2023	David Santos		Draft Release
2.0	4/29/2023	David Santos		Formatted tables Updated Figure 1 Updated Electrical Interface Diagram

Table of Contents

1. Overview **6**

2. References and Definitions **6**
2.1. References 6
2.2. Definitions 6

3. Physical Interface **7**
3.1. Weight 7
3.1.1. Main Control Unit 7
3.1.2. Power Supply Unit 7
3.2. Dimensions 7
3.2.1. Dimensions of MCU Subsystem 7
3.2.2. Dimensions of Power Supply Subsystem 8
3.3. Mounting Locations 8

4. Thermal Interface **9**

5. Electrical Interface **10**
5.1. Primary Input Power 10
5.1.1. Main Control Unit 10
5.1.2. L298N Motor Driver 10
5.2. Voltage and Current Levels 10
5.2.1. Maximum Values 10

6. Communications / Device Interface Protocols **11**
6.1. Wireless Communications (WiFi) 11
6.2. Device Peripheral Interface 11

List of Tables

Table 1. Main Control Unit Weight 7
Table 2. Power Supply Unit Weight 7
Table 3. MCU Dimensions 7
Table 4. Power Supply Dimensions 8
Table 5. Maximum Electrical Values 10

List of Figures

Figure 1. System Graphical Diagram 8
Figure 2. Electrical Interface Diagram 9

1. Overview

This document will show how the web application, power supply, and MCU subsystems will interface with each other. It will list all possible inputs, outputs, and how the system will manage each. First, an explanation of the inputs from the web application and how it is transferred to the MCU will be detailed. A description of how the power supply will be implemented, and how the MCU interfaces with the power supply subsystem will follow.

2. References and Definitions

2.1. References

Refer to section 2.2 of the Functional Systems Requirement document.

2.2. Definitions

W	Watt
A	Amp
mA	Milliamp
mW	Milliwatt
TBD	To Be Determined
g	Grams

3. Physical Interface

3.1. Weight

3.1.1 Main Control Unit

Table 1 : Main Control Unit Weight

Component	Weight	Number of Items	Total Weight
Arduino Uno Board	25 g	1	25 g
ESP8266	1.72 g	1	1.72 g
NEO-6M	22 g	1	22 g
4x20 LCD Display	76 g	1	76 g

3.1.2 Power Supply Unit

Table 2: Power Supply Weight

Component	Weight	Number of Items	Total Weight
Power Supply Board	TBD	1	TBD
AC-DC Converter	TBD	1	TBD
DC-DC Boost Converter	TBD	1	TBD
Output Port	TBD	2	TBD

3.2. Dimensions

Dimensions are in millimeters.

3.2.1. Dimensions of MCU Subsystem

Table 3: MCU Dimensions

Component	Length	Width	Height
Arduino UNO Board	68.6 mm	53.4 mm	24.9 mm
LCD Display	20 mm	4 mm	N/A

3.2.2 Dimensions of Power Supply Subsystem

Table 4: Power Supply Dimensions			
Component	Length	Width	Height
Power Supply Board	TBD	TBD	TBD

3.3. Mounting Locations

The Satellite Antenna Tracking Mount (SATM) should be universally available, but will most likely be deployed in various locations around College Station, Texas. The MCU will be attached by wires to the antenna mount, but the MCU box itself will be mounted directly below the antenna mount. The power supply will be plugged into a wall outlet (most likely with an extension cord), and then the outputs of the power supply will plug directly into the MCU box.

Figure 1: Satellite Antenna Tracking Mount System

4. Thermal Interface

The L298N motor driver has a built-in heat sink. There is also a built-in heat sink for the MCU (Arduino UNO). Also, to prevent overheating of the antenna mount, the gearbox will need to be lubricated.

Figure 2: Electrical Interface Diagram

5. Electrical Interface

5.1. Primary Input Power

5.1.1 Main Control Unit

The MCU will be powered by the power supply board and will take 9V at 50 mA. It will consume a maximum of 1.8 watts of power.

5.1.2 L298N Motor Driver

The motor driver will be powered by the power supply board and will take 24V at 2A. The two motors together will consume a maximum of 48 watts of power.

5.2. Voltage and Current Levels

5.2.1 Maximum Values

Table 5: Maximum Electrical Values

Component	Voltage	Current	Power
Arduino UNO	9 V	≤ 200 mA	1.8 W
L298N Motor Driver	24 V	≤ 2 A	48 W

The values in table 5 allow for gauging how much power the system will use, per second, at peak power consumption. It is estimated that the system will be at peak power most of the time because the motors will be running constantly to track the satellite.

6. Communications / Device Interface Protocols

6.1. Wireless Communications (WiFi)

Using IEEE 802.11ac the Satellite Antenna Tracking Mount will connect to a user's device and network to upload and download data.

6.2. Device Peripheral Interface

The device will have custom connectors to send a digital signal to the customers existing system in order to control the motors on the existing system.

Satellite Antenna Tracking Mount

Hunter Britton
Michael Raabe
David Santos

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – 2.0
28 April 2023

FUNCTIONAL SYSTEM REQUIREMENTS
FOR
Satellite Antenna Tracking Mount

APPROVED BY:

_____
Hunter Britton Date

_____
Prof. S. Kalafatis Date

_____
Pranav Dhulipala Date

Change Record

Rev.	Date	Originator	Approvals	Description
1.0	2/15/2023	David Santos		Draft Release
2.0	4/28/2023	David Santos		Formatted Tables Updated Figure 1 Updated Figure 2 Added microcontroller requirements Added voltage tolerances Updated acronyms list

Table of Contents

1. Introduction **7**
1.1. Purpose and Scope 7
1.2. Responsibility and Change Authority 8

2. Applicable and Reference Documents **9**
2.1. Applicable Documents 9
2.2. Reference Documents 9
2.3. Order of Precedence 9

3. Requirements **10**
3.1. System Definition 10
3.1.1 Web App Hosting 11
3.1.2 Web App Framework 12
3.1.4 Microcontroller 12
3.1.5 Power Supply 12
3.2. Characteristics 12
3.2.1. Functional / Performance Requirements 12
3.2.1.1 Signal Strength and Quality 12
3.2.1.2 Azimuth and Elevation Accuracy 12
3.2.1.3 Tracking Speed 12
3.2.2. Physical Characteristics 13
3.2.2.1 Form Factor 13
3.2.2.2 Weather Resistance 13
3.2.3. Web App Requirements 13
3.2.3.1 Interactive Map 13
3.2.3.2 Satellite Selection 13
3.2.3.3 Rotator Connection Port 13
3.2.4. Electrical Characteristics 13
3.2.4.1 Inputs 13
3.2.4.2 Power Consumption 14
3.2.4.3 Input Voltage Level 14
3.2.4.4 Input Noise and Ripple 14
3.2.4.5 External Commands 14
3.2.5 Outputs 15
3.2.5.1 Data Output 15
3.2.5.2 Diagnostic Output 15
3.2.5.3 Connectors 15
3.2.5.4 Wiring 15
3.2.6 Failure Propagation 16

3.2.6.1 Failure Detection, Isolation, and Recovery (FDIR) 16
3.2.6.2 Built In Test (BIT) 16
3.2.6.3 BIT Critical Fault Detection 16
3.2.6.4 BIT Log 16

4. Support Requirements 17

Appendix A: Acronyms and Abbreviations 18

Appendix B: Definition of Terms 19

List of Tables

Table 1. Subsystem Responsibility 8
Table 2. Standards 9
Table 3. Reference Documents 9

List of Figures

Figure 1. Browser-Based Satellite Antenna Tracking System 7
Figure 2. System Block Diagram 11

1. Introduction

1.1. Purpose and Scope

This document provided detailed system requirements for an amateur radio remote operation system. Amateur radio systems require specialized equipment and software to operate automatically. Typically a controller is used in conjunction with computer software to control the antenna rotator, but our objective is to provide a remote operation system that will allow a user to control the antenna from a web browser. The Satellite Antenna Tracking Mount (SATM) will be formed from a legacy rotator with two motors, additional control and communication hardware, and a web application. SATM will also display pertinent data and provide secure user authentication via hashing account data, allowing for low-cost remote operation. The improved system will provide users greater flexibility and ease-of-use through web accessibility.

Figure 1. Browser-Based Satellite Antenna Tracking System

The following definitions differentiate between requirements and other statements.

Shall:	This is the only verb used for the binding requirements.
Should/May:	These verbs are used for stating non-mandatory goals.
Will:	This verb is used for stating facts or declaration of purpose.

1.2. Responsibility and Change Authority

The team leader, Hunter Britton, will be responsible for verifying all requirements are satisfied and the verification plan. Performance requirements are only to be changed by David Gent or by the consensus of the team members with the approval of David Gent. Each owner is responsible for the fulfillment of the functional requirements for their subsystem shown below.

Table 1. Subsystem Responsibility

Subsystem	Owner
MCU / IO	Hunter Britton
Power Supply / Motor Driver	Michael Raabe
Web Application	David Santos

2. Applicable and Reference Documents

2.1. Applicable Documents

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

Table 2. Standards		
Document Number	**Revision/ Date**	**Document Title**
IPC A-610E	Revision E – 4.1.2010	Acceptability of Electronic Assemblies
IEEE 802.11ba	Revision ba - 2021	Wake-Up Radio Operation
CCSDS 502.0-P-2.1	Revision 2.1 - 10.21	Orbit Data Messages
FCC 97.213	Revision 213 - 6.20.1989	Telecommand of an Amateur Station

2.2. Reference Documents

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.

Table 3. Reference Documents		
Document Number	**Revision/ Date**	**Document Title**
CFR P97T47	6.20.1989	Amatuer Radio Service
C95.6	10.23.2022	IEEE Standard for Safety Levels With Respect to Human Exposure to Electromagnetic Fields

2.3. 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 report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

3. Requirements

This section defines the minimum requirements that the development item(s) must meet. The requirements and constraints that apply to performance, design, interoperability, reliability, etc., of the system, are covered.

3.1. System Definition

The SATM system is divided into three subsystems to manage complexity: microcontroller and peripherals, power supply and motor driver, and web application interface. The microcontroller is responsible for taking data from the rotator motors, GPS, and web application settings and using it to control the LED display, antenna positioning, and relaying that back to the web application with a WiFi module. The power supply subsystem will convert power from the standard wall outlet 120v 60Hz to power the motor driver and microcontroller. Finally the web application will be used to interface with the rotator through the Firebase platform. The web application will feature search capabilities for the rotator and satellite, data visualization, and a map visualizer.

A user will first connect their rotator via Wifi. The web app will then request GPS location, as well as the current position of the antenna from the microcontroller to determine where to place the marker on the map. Once a satellite is selected, the web app will access the NASA satellite catalog to relay the orbital elements. The microprocessor can then calculate the path of the satellite as well as the motor controls. Alternatively, the user can choose to manually enter the elevation and azimuth positioning.

Figure 2. System Block Diagram

The block diagram in Figure 2 illustrates the subsystem added to legacy systems as well and their sub components. The MCU acts as an intermediate controller between the web application and rotator system. This PCB along with the power supply and driver will be mounted onto the upper part of the rotator with enough slack that it can move freely.

3.1.1 Web Application Hosting

Firebase is a collection of web services developed by Google and will serve as the web application hosting platform. Similar to Amazon Web Services (AWS), it provides functions typically handled by the server, such as real-time databases and user authentication, through the cloud. It was designed to enable rapid prototyping and scalability, making it perfect for IoT applications that use web hosting. Firebase works on multi-tier payments to cater to the diverse users, ranging from hobbyists to professionals.The “Spark Plan” is the free version and provides all the necessary features for this project with the option to upgrade. Firebase incorporates the REST API, which is a software architecture that uses HTTP requests, such as “GET”, “POST”, “PUT”, and “DELETE”, to interact with resources on the server. HTTP can use URLs to access resources with a variety of formats like JSON or XML. The lightweight, user-friendly interface of the Google Cloud Platform will allow for easy web development.

3.1.2 Web App Framework

The front-end will use Vue.js because it is lightweight and versatile. This will enable an implementation of a satellite search bar, a connection port for the microcontroller, and an interactive map with the Esri JavaScript API. The satellite catalog will be made available through REST API.

3.1.4 Microcontroller

The microcontroller will use an Arduino Mega because it contains a fast and powerful processor that can handle the computations required for predicting the orbital path of a satellite. The Arduino Mega will take the inputs from a Beitian BN-280 GPS system and use those coordinates to compute the direction that the antenna needs to point. The output of the Arduino will be fed to an LCD screen that will output the coordinates of the satellite, the azimuth and elevation, as well as transfer those coordinates to the web app.

3.1.5 Power Supply

There will be a power supply board, separate from the Arduino UNO board, that will take in a 120 V, 60 Hz input from a standard wall outlet, and have two outputs. One output will go directly to the Arduino UNO board, which requires 7-12 VDC and a current not exceeding 200 mA. The other output from the power supply board will go to a separate input on the Arduino UNO board for the L298N motor driver. The L298N requires 24 VDC and a current not exceeding 2.5 A.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1 Signal Strength and Quality

Antenna shall have a Signal-to-Noise Ratio (SNR) >= 30dB and a RST Report of R5S5T9 or better indicating perfect readability, good signals, and rectified AC signals but strongly ripple-modulated.

Rationale: Together, signal strength and quality provide a good indication of how intelligible the audio signals are in amateur radio communication. A strong and high-quality signal can ensure that the audio is clear and audible, even under challenging conditions such as interference, noise, or path loss. Conversely, a weak or poor-quality signal may result in distorted, garbled, or unintelligible audio, making it difficult to communicate effectively.

3.2.1.2 Azimuth and Elevation Accuracy

Antenna shall point to the selected satellite within 4 percent, the error of the motor gearbox.

Rationale: The directional nature of satellite transmitters and antennas can have implications for satellite tracking and communication. To establish a reliable communication link with a satellite, the ground station antenna must be pointed accurately towards the satellite's position and must use an antenna that is capable of receiving the specific frequency and direction of the satellite's signal. Additionally, changes in the satellite's position or orientation can affect the strength and direction of the signal, requiring adjustments to the tracking system and communication parameters.

3.2.1.3 Tracking Speed

The antenna should have a tracking speed of at least 2 degrees per second to ensure that the center of the antenna is within 2 degrees for every second of movement of the satellite.

Rationale: As a satellite moves across the sky, its position changes rapidly, and a tracking system must be able to keep up with this movement to maintain a stable signal. If the tracking speed is too slow, the antenna may fall out of alignment with the satellite, resulting in a loss of signal or poor signal quality. This can be especially problematic for applications that require a continuous or high-quality signal, such as satellite communication or Earth observation.

3.2.2. Physical Characteristics

3.2.2.1 Form Factor

The hardware added to the rotator shall be minimized to 5 kg and a total volume of 0.75 m^3

Rationale: If the hardware is too large, it may interfere with the antennas or the rotators, limiting the functionality.

3.2.2.2 Weather Resistance

The hardware enclosure should withstand non-freezing temperatures and reasonable precipitation.

Rationale: The additional hardware should be as durable as the rotator and antenna

3.2.3. Web App Requirements

3.2.3.1 Interactive Map

The Web Application will have an interactive map of earth as a mercator projection. Day/Night Terminator, Ground Track, Labels, and a grid may also be added to the map.

Rationale: The map allows the user to easily see where the rotator is in relation to the satellite.

3.2.3.2 Satellite Selection

The Web Application shall have a search bar to select a satellite to track

Rationale: The satellite to be tracked needs to be defined or the web app does not know what orbital elements to request

3.2.3.3 Rotator Connection Port

The web application should have a connection method for the microcontroller. It should allow the user to disconnect, reboot, search, or reconnect.

Rationale: The web application has to initially contact the microcontroller for the GPS Location, and Motor Position data.

3.2.4. Electrical Characteristics

3.2.4.1 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 Satellite Antenna Tracking Mount, 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 Satellite Antenna Tracking Mount, reduce its life expectancy, or cause any malfunction.

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

3.2.4.2 Power Consumption

a. The maximum peak power of the system shall not exceed 50 watts.

Rationale: This is because the motors will consume a maximum of 48 watts. The Arduino board will consume a maximum of 1.8 watts.

3.2.4.3 Input Voltage Level

The input voltage level for the Satellite Antenna Tracking Mount will be 9 +-2 VDC for the Arduino UNO board and 24 +-4 VDC for the L298N motor driver input.

Rationale: The Arduino UNO board has an input voltage operating range of 7-12 VDC and the L298N motor driver has an input voltage operating range of 7-35 VDC. The motors themselves require 24 VDC in order to operate.

3.2.4.4 Input Noise and Ripple

The input noise and ripple for the Satellite Antenna Tracking Mount shall operate while in the presence of a 1.5 Volt RMS ripple superimposed on the steady-state voltage over the frequency range of 0 Hz to AC.

Rationale: Aircraft bus specification compatibility, MIL-STD-704F

3.2.4.5 External Commands

The Satellite Antenna Tracking Mount shall document all external commands in the appropriate ICD.

Rationale: The ICD will capture all interface details from the low level electrical to the high-level packet format.

3.2.5 Outputs

3.2.5.1 Data Output

The Satellite Antenna Tracking Mount outputs the azimuth and elevation of the antenna to the web application, as well as the GPS location of the system.

Rationale: The Satellite Antenna Tracking Mount information passes directly to the customer's system.

3.2.5.2 Diagnostic Output

The Satellite Antenna Tracking Mount outputs the azimuth and elevation information in order to show any errors in the tracking output.

Rationale: Provides the customer with the information to adjust the antenna's mounting location.

3.2.5.3 Connectors

The Satellite Antenna Tracking Mount will use external connectors to interface with the customers Yaesu-G5500.

Rationale: This is a requirement specified by our customer due to constraints of their system in which the Satellite Antenna Tracking Mount is integrating into.

3.2.5.4 Wiring

The Satellite Antenna Tracking Mount shall follow the guidelines outlined in MIL-HDBK-5400 paragraph 4.3.35 Wire and cable.

Rationale: Conform to aircraft standard.

3.2.6 Failure Propagation

The Satellite Antenna Tracking Mount System shall not allow propagation of faults beyond the Satellite Antenna Tracking Mount System interface.

3.2.6.1 Failure Detection, Isolation, and Recovery (FDIR)

3.2.6.2 Built In Test (BIT)

The Satellite Antenna Tracking Mount will have a heartbeat signal talking to the web application so verification can be made if any packets are dropped in communication.

3.2.6.3 BIT Critical Fault Detection

The BIT shall be able to detect a critical fault in the Satellite Antenna Tracking Mount 95 percent of the time.

Rationale: This is a requirement specified by our customer due to constraints of their system in which the Satellite Antenna Tracking Mount is integrating.

3.2.6.4 BIT Log

The BIT shall save the results of each run into a log so the customer can view previous data points and keep a track of the satellites they have tracked.

Rationale: This is a requirement specified by our customer due to constraints of their system in which the Satellite Antenna Tracking Mount is integrating.

4. Support Requirements

4.1 Web Browser

Modern web browser with support for JavaScript, CSS, HTML5, and web graphics.

4.2 Wifi Connection

Minimum Download Speed: 10Mbps

Minimum Upload Speed: 5Mbps

Latency < 100ms

4.3 120V 60Hz Power Outlet

Standard NEMA 5-15 outlet with 15 A capacity and maximum power delivery of 1800 watts. GFCI protection for outlets near water sources. Electrical standards and codes can vary by state or local jurisdiction, so it is important to check the specific regulations in the area where the wall outlet will be installed.

Appendix A: Acronyms and Abbreviations

SATM	Satellite Antenna Tracking Mount
MCU	Micro Control Unit
IO	Input Output
IPC	Institute of Printed Circuits
IEEE	Institute of electrical and electronic engineers
CCSDS	Consultative Committee for Space Data Systems
FCC	Federal Communication Committee
CFR	Code of Federal Regulations
ICD	Interface Control Document
GPS	Global Positioning System
LED	Light-emitting Diode
NASA	National Aeronautics and Space Administration
PCB	Printed Circuit Board
AWS	Amazon Web Services
Iot	Internet of Things
HTTP	Hypertext Transfer Protocol
URL	Uniform Resource Locator
REST	REpresentational State Transfer
API	Application Programming Interface
JSON	JavaScript Object Notation
XML	Extensible Markup Language
LCD	Liquid Crystal Display
VDC	Voltage Direct-Current
SNR	Signal Noise Ratio
kg	kilogram
VDC	Voltage Direct Current
BIT	Built-In Test
CSS	Cascading Style Sheet
Mbps	Megabits per second
ms	millisecond
GFCI	Ground Fault Circuit Interrupter
NEMA	National Electrical Manufacturers Association

Satellite Antenna Tracking Mount

Hunter Britton
Michael Raabe
David Santos

SCHEDULE AND VALIDATION

REVISION – Final
7 February 2023

Change Record

Rev.	Date	Originator	Approvals	Description
1.0	2/7/2023	David Santos		Draft Release
2.0	4/28/2023	David Santos		Formatted Tables Added method column to tests

System Execution Timeline				
Task	Owner	Status	Start Date	End Date
Concept of Operations	All	Completed	02/13/2023	02/13/2023
Sponsor Web App Verification	David	Completed	02/15/2023	02/15/2023
Functional System Requirements	All	Completed	02/22/2023	02/22/2023
Interface Control Document	All	Completed	02/22/2023	02/22/2023
Finalize wireframe	David	Completed	02/22/2023	02/26/2023
Toolbar	David	Completed	02/27/2023	03/10/2023
Midterm Presentation	All	Completed	03/01/2023	03/01/2023
Sponsor Specification Approval	All	Completed	03/01/2023	03/01/2023
First Milestone - Parts Order	All	Completed	03/03/2023	03/03/2023
Subsystem Circuit Design	All	Completed	03/05/2023	03/05/2023
Learn Arduino IDE	Hunter	Completed	03/06/2023	03/06/2023
Setup MCU Communication	David	Completed	03/14/2023	03/14/2023
GPS Location Services working	Hunter	Completed	03/15/2023	03/15/2023
Interactive Map	David	Completed	03/18/2023	04/07/2023
Second Milestone	All	Completed	03/20/2023	03/20/2023
Status Update	All	Completed	03/29/2023	03/29/2023
Completed circuit	Michael	Completed	03/30/2023	03/30/2023

drawing for power supply				
Microcontroller/GPS location to satellite Math	Hunter	Completed	03/30/2023	03/30/2023
Power Supply PCB Assembled	Michael	ComCompCompleted	03/31/2023	03/31/2023
Validate Power Supply Output	Michael	Completed	04/01/2023	04/01/2023
Validate Microcontroller can send output signals	Hunter	Completed	04/03/2023	04/03/2023
Third Milestone	All	Completed	04/03/2023	04/03/2023
LED display done	Hunter	Completed	04/05/2023	04/05/2023
Microcontroller PCB Done	Hunter	Completed	04/05/2023	04/05/2023
Web App Validation	David	Completed	04/08/2023	04/18/2023
Microcontroller PCB Assembled	Hunter	Completed	04/12/2023	04/12/2023
Finish final presentation preparation	All	Completed	04/17/2023	04/17/2023
Work on Subsystem presentations	All	Completed	04/17/2023	04/17/2-23
Final Presentation	All	Completed	04/17/2023	04/17/2023
Finish Subsystem presentations	All	Completed	04/23/2023	04/23/2023
Subsystem Presentation	All	Completed	04/23/2023	04/25/2023
Final Report	All	Completed	04/20/2023	04/29/2023

Schedule and Validation
Satellite Antenna Tracking Mount

Revision -2

Completed
In Progress

Task	Owner	FEB	MARCH				APRIL			
		Feb 27	March 6	March 13	March 20	March 27	April 3	April 10	April 17	April 24
React Components	David S.	Toolbar		Orbital Data Fetching		Mercator Map		Validation		
Interactive Map	David S.			ArcGis Instance		Map Settings	Graphics	Validation		
Firebase Hosting	David S.						Add to Project			
MCU/Rotator Communication	David S. Hunter B.		Order ESP8266		HTTPS Server		Rotator Controls	Validation		
TLE Propagation	Hunter B.	Research Orbital Dynamics			Arduino Implementation			Calc/Improve Accuracy		
MCU IO (GPS/LED)	Hunter B.		GPS Code				LED Display	Validation		
Power Supply Unit	Michael R.	Define Specifications		Order Parts		Simulations/Assembly		PCB Layout		
Motor Drivers	Michael R.	Define Specifications		Order Parts		Test Parts				

Validation Plan

Attribute	Specification	Method	Result	Owner
Signal Strength	SNR >= 30dB	1. Setup Hardware 2. Choose Satellite 3. Connect Rotator 4. Initial Communication 5. Read SDR measurement	404	All
RST Report	R5S5T9	1. Setup Hardware 2. Choose Satellite 3. Connect Rotator 4. Initial Communication 5. Request RST Report	404	All
Azimuth Accuracy	+- 4%	1. Calculate perfect north 2. Use compass to find Azimuth 3. Compare with angle of rotator	404	Michael/Hunter
Tracking Velocity	2 deg/sec	1. Initial Tracking 2. Use Vernier to record tracking 3. Analyze video to calculate angular velocity	404	Michael/Hunter
Power Supply to Motors	48 Watts +-2.5	1. Measure Output Voltage 2. Measure Output Current 3. Calculate Power	47 Watts	Michael
Power Supply to Microcontroller	1.8 Watts +-0.5	1. Measure Output Voltage 2. Measure Output Current 3. Calculate Power	1.77 Watts	Michael
Microcontroller Motor Control output	Continuous Signal for 6 minutes	1. Run Rotator Continuously for 6 minutes 2. Record behavior and performance	404	Hunter
Microcontroller GPS location	Accurate location within 20 meters	1. Setup GPS 2. Read Sensor Data 3. Compare with Iphone GPS	17m Accuracy	Hunter
Satellite Prediction	Accurate within 3%	1. Compute coordinates and angles. 2. Compare with actual values	1-2.87%	Hunter

Satellite	All active satellite are searchable and orbital data can be retrieved	1. type in known satellite 2. select option 3. show orbital data in console	All satellite able to be retrieved	David
Rotator	User can connect to rotator	1. use IP address to connect to rotator 2. have esp8266 return rotator object data	rotator object data returned on connection	David
Manual control	User can input azimuth and elevation and send it to the mcu	1. click “control menu” 2. input azimuth and elevation 3. click “start tracking” 4. see if server receives data	Server received azimuth and elevation and returned success message	David
Automatic control	User can track satellite without azimuth/elevation data	1. Click “control menu” 2. select “autotrack” 3. click “start tracking” 4. see if server receives command	Server received auto tracking command and orbital data to propagate path	David
Map	User can visualize relation position of satellite and rotator	1. Mount Application. 2. Select Satellite/Rotator 3. Verify Map	Relative position of Map and Satellite visible on map	David
Accessibility	User can access rotator from any browser	1. connect to web app from various places	Able to access web app in all browsers	David
Usability	Interface is visible pleasing and usable	Surveys	Mostly Positive Surveys (see report)	David

Performance on Execution Plan

The execution plan was executed completely. Although some tasks were actually performed somewhat out of order to accommodate the schedules of all the teammates and the people operating the equipment used for the validation of the sensors, each task that was set forth in the execution plan was completed on time. The project was completed in full, with all of the objectives that were outlined at the beginning of the project demonstrated as required by the initial proposed solution.

Performance on Validation Plan

The validation plan was executed and was rigorously reviewed to ensure that all specifications were met. Data was collected from each subsystem to ensure proper performance, as well as the margin of error being within acceptable ranges. Although some difficulties were encountered, with the satellite prediction software being a major component to this, they were overcome to ensure proper performance of the system.

Satellite Antenna Tracking Mount

Hunter Britton
Michael Raabe
David Santos

SUBSYSTEM REPORT

REVISION – 1.0
22 April 2023

SUBSYSTEM REPORT
FOR
Satellite Antenna Tracking Mount

APPROVED BY:

_____
Hunter Britton Date

_____
Prof. S. Kalafatis Date

_____
Pranav Dhulipala Date

Change Record

Rev.	Date	Originator	Approvals	Description
1.0	4/28/2023	David Santos		Draft Release

Table of Contents

1. Introduction **6**
2. Power Supply **6**
2.1 Power Overview 6
2.2 Power Specifications 6
2.3 Power Validation 10
2.4 Power Conclusion 10
3. Microcontroller Unit Subsystem Report **11**
3.1 MCU Overview 11
3.2 MCU Specifications 11
3.3 MCU Validation 12
3.4 MCU Conclusion 14
4. Web Application Subsystem **15**
4.1 Web App Overview 15
4.2 Web App Specifications 15
4.3 Web App Validation 18
4.4 Web App Conclusion 25
Appendix A: Acronyms and Abbreviations **26**

List of Tables

Table 1. Output Voltage Validation 9
Table 2. ISS Test Prediction/Actual 13
Table 3. MCU Pin List 13
Table 4: GPS Module Validation 15
Table 5. Map Helper Unit Tests 20
Table 6. Helper Function Unit Tests 21
Table 7. React UI Tests 22
Table 8. React UI Tests Continued 23
Table 9. UX Design Survey Responses 24
Table 10. UX Performance Survey Responses 24
Table 11. Execution Timeline (Log) 26
Table 12. Execution Timeline (Log) Continued 26

List of Figures

Figure 1. L298N Pinout Diagram 7
Figure 2. Buck Converter Schematic 8
Figure 3. Buck Converter PCB Layout 8
Figure 4. AC Power MultiSim Simulation 9
Figure 5. Azimuth/Elevation Motor Schematic 10
Figure 6. Satellite Prediction (Percent Error) 12
Figure 7. Arduino Mega 14
Figure 8. ESP8266 Module 14
Figure 9 Browser Web App Mockup 17
Figure 10. Control Menu 18
Figure 11. Dialog Box 18
Figure 12. Map Controls 18
Figure 13. Satellite Popup 19
Figure 14. Suggestions Box 19
Figure 15. Observer Popup 19
Figure 16. Web App Latency 25

1. Introduction

The Satellite Antenna Tracking System (SATS) is an intuitive, user-friendly solution to remote operation of amateur radio setups. With SATS, a user can select a satellite, and it will provide a calculated prediction of the orbit path to the user interface via a microcontroller. The system will then provide an input to the legacy antenna to move the antenna in the direction of the satellite according to the calculated azimuth and elevation. The system will take in power from the wall outlet and convert it to the needed voltage and current for the subsystems, as well as the legacy DC motors. Since each of these subsystems have been designed and tested in accordance with the validation plan, there is a clear path to integration as specified in the Concept of Operations (ConOps), Functional System Requirements (FSR) , and Interface Control Document (ICD).

2. Power Supply

2.1 Power Overview

The power subsystem is designed to supply power to the motor drivers and microcontroller from a standard 120V 60Hz wall outlet. The subsystem supplies power to the Arduino, the legacy azimuth rotor, the legacy elevation rotor, and the logic circuits for the two L298N motor drivers.

2.2 Power Specifications

To begin the power conversion from the wall outlet, 120 VAC goes into two 24V, 60W AC-DC converters. Directly from each of these 24V converters, the output goes into two separate inputs. One wire goes straight to one of the L298N motor drivers to power the azimuth rotor. The other output wire will be used to power a 24V to 6V DC-DC buck converter in order to power the logic circuits within the L298N motor drivers. For the other 24V converter, one output wire goes directly to the other L298N motor driver, and the other output wire will be used to power a 24V to V DC-DC buck converter in order to power the Arduino microcontroller. The actual input voltage range for the logic circuits within the L298N motor drivers is 5-7 volts. This is the reason for the design of a buck converter to 6V. That way there is some room for error on either side. The input voltage range for the Arduino microcontroller is 7-12 volts. This is the reason for the design of an 8V buck converter. This also allows for some error either above or below 8V.

Figure 1: L298N (src:MinuteEngineers.com)

Each buck converter utilizes a 555 astable timer circuit connected to a 9V power supply to create a 9V square wave with a ~40 KHz frequency to control the switching in each buck converter. This timer circuit is connected to an IRF510 MOSFET to act like a SPST voltage-controlled switch.

The Arduino normally operates at around 50 mA of current, but can draw a maximum of 200 mA. The azimuth and elevation rotors each draw 2 A of current, and the logic circuits each require 36 mA of current. With the 2.5 A max current output for each of the 24V AC-DC converters, there will be more than enough current to power the entire system.

The L298N motor drivers will be programmed to work with the Arduino microcontroller in the future. For this semester, the goal was to take care of all of the power needs of the system.

Figure 2: Buck Converter Schematic

Figure 3: Buck Converter PCB Layout

Figure 4: MultiSim AC Power Simulation

2.3 Power Validation

The power subsystem was validated to meet all of the requirements needed. The integration of this subsystem with the other subsystems will be done in the future.

Table 1: Output Voltage Validation

System	Acceptable Range	Output
24V AC-DC Converter	23.5 - 24.5 V	23.8 V
24V AC-DC Converter	23.5 - 24.5 V	23.7 V
8V Buck Converter	7 - 12 V	8.28 V
6V Buck Converter	5 - 7 V	5.77 V

Figure 5: Azimuth/Elevation Yaesu Motor Schematic (src: Yaesu.com)

2.4 Power Conclusion

The power subsystem worked as intended, and will provide the needed power for the entire system from a standard 120 VAC wall outlet. As shown in the table above, each system provides the correct amount of power needed. The subsystem will be integrated with the microcontroller and legacy subsystem in the future. The power subsystem is currently finished completely on a perfboard, and the PCB has been designed and ordered. The power PCB will be assembled and finished over the Summer so that it can be utilized in the final system next semester.

3. Microcontroller Unit Subsystem Report

3.1 MCU Overview

The microcontroller subsystem is designed to predict and calculate the satellite position, as well as provide the longitude, latitude, azimuth, and elevation. The subsystem communicates with the web application to provide these values needed to display the location of the satellite, as well as communicates with the motor driver to position the antenna in the direction that would allow it to communicate with the chosen satellite.

3.2 MCU Specifications

The microcontroller selected is an Arduino Mega 2560. The Arduino was selected due to the number of input and output pins, as well as having the necessary computing power to predict the orbit of a satellite.

The Arduino Mega 2560 has 54 digital pins, with 15 of those pins having Pulse Width Modulation output, and 16 Analog outputs. The Analog outputs have 10 bits of resolution which reads from 0 to 5 Volts, which would allow the current direction of the motors to be read. The Arduino can be powered from 7 to 12 Volts, with the converter supplying 8 Volts. The Arduino is also able to supply 5 Volts to power the GPS and LCD screen.

The main feature of the microcontroller system is to calculate and predict the location of any satellite in orbit using Two Line Elements (TLE's) provided by the Web Application from Celestrak. This calculation was based on the Plan13 algorithm written by James Miller. No matter the satellite selected, the microcontroller can predict the orbit and pathway and provide the output within 1%.

The microcontroller can also output the longitude, latitude, azimuth and elevation information onto an LCD screen for manual debugging and control. These values are outputted after the microcontroller finishes predicting the location of the satellite at the provided time.

The microcontroller also reads data from the Beitian BN-280 GPS module to provide a higher accuracy for the azimuth and elevation, as well as to provide the web application with the current location of the system.

The communication between the motors and the microcontroller will be added in the future for a high level of precision for the direction of the antenna.

3.3 MCU Validation

The microcontroller subsystem was validated to meet all the requirements needed. Since the majority of the functionality of the subsystem is independent of the rest of the subsystems, it was validated by itself. The integration of this subsystem with the power subsystem and web application will be needed to control the legacy system.

The chart below showcases the error of the satellite prediction code.

Figure 6: Percent Error of Satellite Prediction compared to Actual Orbit

The two spikes for the prediction error are when the satellite passes over the Prime Meridian, 0 degrees. This error can be attributed to the prediction software outputting .50 degrees, leading to a large percent error when compared to 0 degrees. Although this spike is large in the longitude, the main component of the calculation is the azimuth and elevation, since those two values showcase what direction to point the antenna to communicate with the satellite, and the azimuth and elevation stay within a small percent error.

The table below shows the prediction of the ISS and compares the values to known quantities measured by Celestrak in orbit, as well as the error between the two values.
To run through an entire 24 hours of prediction for the chosen satellite the code takes around one minute and 30 seconds. This could be sped up by only predicting the satellite location for times in the future, rather than the entire 24 hour day.

Table 2: International Space Station (ISS) Test Case												
Time (UTC)	Predicted				Actual				Error			
	$\widehat{\phi}$	$\widehat{\lambda}$	$\widehat{\alpha}$	$\widehat{\theta}$	α	θ	ϕ	λ	$\Delta\alpha$	$\Delta\theta$	$\Delta\phi$	$\Delta\lambda$
22:42:41	-8.2	-57.3	129.3	-23.4	129.2	-23.5	-8.3	-57.2	0.09	0.47	1.57	0.21
22:43:41	-11.2	-55.2	129.9	-25.5	129.7	-25.6	-11.3	-55.0	0.15	0.47	1.33	0.36
22:44:41	-14.2	-52.9	130.3	-27.6	130.2	-27.7	-14.3	-52.7	0.08	0.43	1.12	0.34

In the table above, latitude (ϕ), longitude(λ), azimuth(α), and elevation (θ), were validated and tested across to ensure that the margin of error was within the accepted amount. Testing was completed with multiple satellites to obtain statistical mean absolute error (MAE).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y - \widehat{y}\right|$$

$$MAE(\phi) \simeq 1.31 \quad MAE(\lambda) \simeq 0.30 \quad MAE(\alpha) \simeq 0.11 \quad MAE(\theta) \simeq 0.43$$

Table 3: MCU Pin List	
Arduino Mega Pin	Label
+5 V	Power the GPS and LCD module
GND	Provide Ground for GPS and LCD Module
10	GPS Module Rx
11	GPS Module Tx
D20	SDA for LCD
D21	SCL for LCD
A0	Motor 1 Output
A1	Motor 2 Output

To allow for user construction and debugging a pin table was designed for use with the subsystem. Barring any modifications, this table will facilitate a full integration with the other subsystems.

Figure 7: Arduino Mega (src: arduino.com)

Figure 8: ESP8266 Module (src: arduino.com)

Table 4: GPS Module Validation						
Location	GPS		Iphone		Error	
	lat	$long$	lat	$long$	lat	$long$
Reed Stadium	30.604097	-96.347002	30.606082	-96.349300	0.006486%	0.002385%
HEB	30.582128	-96.322196	30.582176	-96.322194	0.0002%	0%

3.4 MCU Conclusion

The subsystem works as intended and expected and provides support for the other subsystems. There are also enough pins for the needed functionality, as well as extra pins to allow for the connection to other legacy antenna mounts and systems. Without the processing power of the Arduino Mega, the Satellite Antenna Tracking System would not be able to predict the location of a satellite and communicate with it in real time.

4. Web Application Subsystem

4.1 Web App Overview

The web application is the primary interface for the user to select a satellite or other space objects, retrieve its orbital data, and control the rotator. Remote operation can either be automatic with propagation algorithms or manual by inputting the azimuth and elevation. A mercator map projection provides the user with real-time visualization of the satellite and its relative position to the observer. Communication with the microcontroller unit is supported through HTTPS methods and JSON responses over an internet connection.

4.2 Web App Specifications

The React.js library is used to define the front-end architecture, allowing for the user interface to work seamlessly as distinct components with various handlers, hooks, and state variables. The serverless single-page applications (SSPA) topology was implemented to reduce development time and provide easy back-end hosting with Google Firebase. The bundle.js and index.html are stored there in the cloud and maintain extensive data analytics. Satellite images from Garmin are used along with the ArcGis servers to provide high-quality maps for visualization. Once a satellite is selected, the map updates to show the graphic and the ground track which is the line directly below the satellite.

Figure 9: Web Application Mockup

Click to Open in Sandbox

Figure 10: Control Menu

The control menu has a switch to toggle the auto-tracking feature, two input fields for manual control and a tracking button. Each of the fields have built-in error handle to ensure the azimuth and elevation can be actualized on the chosen rotator.

Figure 11: Dialog Box

The dialog pops up when the “Connect” button is clicked. It has a simple antenna graphic, a input field for the IP address with IPv4 and IPv6 error handling as well as a button to initiate the connection attempt.

Figure 12: Map Controls

The map is generated with custom settings to toggle the various components such as a grid, the day/night terminator, and the satellite ground track. Also built-in are some zoom in/out controls which adds to the usability of the map.

Figure 13: Satellite Popup

When the satellite graphic is clicked, a small table showing the location, altitude and universal time can be seen. The popup can also zoom or be docked in the corner of the map to keep it out of the way.

Figure 14: Suggestions Box

The suggestion box displays the results of the satellite query, returning up to five results for the user to choose from. Since the choice must be from the list, it is not possible to select a satellite for which there is no orbital data.

Figure 15: Observer Popup

When the observer marker graphic is clicked, a small table showing the location, model and local time can be seen. The popup can also zoom or be docked in the corner of the map to keep it out of the way.

4.3 Web App Validation

The web application validation was divided into unit tests, behavioral ui tests, fetch latency, and usability. Unit testing was implemented in the Jest framework to input test cases while the ui tests were manually simulated. Fetch latency was first measured as the time lagging between selecting a satellite and the retrieval of orbital data. The time between the ESP8266 module communications was also measured. Finally a survey was distributed to receive feedback on the overall experience, layout, bugs, and functionality.

Table 5: Map Helper Unit Tests

TEST ID	Method	Description	Test Cases	Expected Results/Behavior	Actual Results	Pass/Fail
MH1 MH2	getGround()	Generates coordinate path for satellite	Longitude =100 Latitude = 45 Longitude = 500 Latitude = -2993	Coordinates of polyline that intersect with Satellite "Out of Range" Error	Coordinates of polyline that intersect with Satellite "Out of Range" Error	Pass Pass
MH3	getGrid()	Generates grid of coordinates	N/A	Coordinates of Latitude/Longitude at 15 degree interval	Coordinates of Latitude/Longitude at 15 degree interval	Pass
MH4	getDayNight()	Calculates Earth's terminator line coordinates	N/A	Coordinates of day/night terminator returned	Coordinates of day/night terminator returned	Pass
MH5	renderGrid()	Renders grid graphic on map	N/A	Grid shows on map	Grid shows on map	Pass
MH6	renderDayNight()	Creates day/night graphic on map	N/A	Terminator shows on map	Terminator shows on map	Pass
MH7 MH8	renderGroundTrack()	Renders ground track graphic	Longitude =100 Latitude = 45 Longitude = 500 Latitude = -2993	Polyline rendered intersecting (100,45) "Out of Range" Error	Polyline rendered intersecting (100,45) "Out of Range" Error	Pass Pass
MH9 MH10	renderObserver()	Renders observer graphic on map	Longitude =100 Latitude = 45 Longitude = 500 Latitude = -2993	Marker rendered at (100,45) "Out of Range" Error	Marker rendered at (100,45) "Out of Range" Error	Pass Pass
MH11 MH12	renderSatellite()	Renders satellite graphic on map	Longitude =100 Latitude = 45 Longitude = 500 Latitude = -2993	Satellite Icon rendered at (100,45) "Out of Range" Error	Satellite Icon rendered at (100,45) "Out of Range" Error	Pass Pass

Table 6: Helper Unit Tests

ID	Method	Description	Test Cases	Expected	Actual	Result
H1 H2	getSatellite()	Retrieves satellite data from Celestrak	Name = ISS (Zarya) Name = aksfhsdf	console.log(Satellite != null) => true console.log(Satellite != null) => false	true false	Pass Pass
H3 H4 H5 H6	getRotator()	Retrieves rotator info from IP	"slakdgo" "102.2.4.2.4" "123.123.123.123" "192.168.50.100"	console.log(Rotator != null) => false console.log(Rotator != null) => false console.log(Rotator != null) => false console.log(Rotator != null) => true	false false false true	Pass Pass Pass Pass
H7 H8	fetchOrbitalData( )	Fetches orbital data from Celestrak	Name = ISS (Zarya) Format = "JSON" Name = aksfhsdf Format = sdfgs	OBJECT_NAME: "ISS (ZARYA)" OBJECT_ID: "1998-067A" EPOCH: "2023-04-25T12:46:22.350432" MEAN_MOTION: 15.50274552 ECCENTRICITY: 0.0005628 ... Error	OBJECT_NAME: "ISS (ZARYA)" OBJECT_ID: "1998-067A" EPOCH: "2023-04-25T12:46:22.350432" MEAN_MOTION: 15.50274552 ECCENTRICITY: 0.0005628 ... Error	Pass Pass
H9 H10 H11 H12	fetchJson()	Fetches JSON data from IP	"slakdgo" "102.2.4.2.4" "123.123.123.123" "192.168.50.100"	"Invalid IP Error" "Invalid IP Error" "IP Timeout Error" Name: "David's Rotator" IP: "192.168.50.100" Call_Sign: "W5QZ" ...	"Invalid IP Error" "Invalid IP Error" "IP Timeout Error" Name: "David's Rotator" IP: "192.168.50.100" Call_Sign: "W5QZ" ...	Pass Pass Pass Pass
H13	getPositionGd()	Calculates geodetic coordinates of satellite	x: 3049.86548, y: -472.53814, z: -6332.4052	longitude: -0.84379 latitude: 0.23418 height: 35856.7679	longitude: -0.84379 latitude: 0.23418 height: 35856.7679	Pass
H14	getSatRec()	Converts TLE string to satellite record	RIGIDSPHERE 2 (LCS 4) 1 05398U 71067E 23115.51147481 .00001221 00000+0 41465-3 0 9997 2 05398 87.6149 296.8478 0064821 76.1852 284.6547 14.34765896708998	{error: 0, satnum: "05398", epochyr: 23, epochdays: 115.51147481, ndot: 0.00001221…}	{error: 0, satnum: "05398", epochyr: 23, epochdays: 115.51147481, ndot: 0.00001221…}	Pass
H15	getLookAngles()	Calculates azimuth, elevation, and range	{error: 0, satnum: "05398", epochyr: 23, epochdays: 115.51147481, ndot: 0.00001221…}	longitude: 2.7705218839347348, latitude: -0.03913181951360097, height: 768.8038068041697}	longitude: 2.7705218839347348, latitude: -0.03913181951360097, height: 768.8038068041697}	Pass
H16 H17 H18 H19	isValid()	Validates IPv4 address	"slakdgo" "102.2.4.2.4" "123.123.123.123" "192.168.50.100"	false false true true	false false true true	Pass Pass Pass Pass
H20	getLocalTime()	Calculates local time from longitude	N/A	Time: "22:04:16"	Time: "22:04:16"	Pass
H21	getTimeUTC()	Calculates UTC from GMST	N/A	12:52:55	12:52:55	Pass

Table 7: React UI Tests

Component	ID	Method	Description	Test Cases	Steps	Expected	Actual	Result
Search Bar	SB1 SB2 SB3	handleSearchTextChange()	Updates search suggestions on input	"ISS" "Cap" "ZZZ"	Type test case in search bar	"ISS (Zarya)" "CAPE-3" ""	"ISS (Zarya)" "CAPE-3" ""	Pass Pass Pass
	SB4 SB5	handleSuggestionClick()	Handles suggestion click event	"ISS"	1. Type in test case in search bar 2. Click A Suggestion	console.log(Satellite != null) => true console.log(SearchText) => ISS (Zarya)	True ISS (Zarya)	Pass Pass
	SB6	handleClickAway()	Clears suggestions and search bar	"ISS"	1. Type in test case in search bar 2. Click away without selecting a satellite	console.log(suggestions) => undefined	undefined	Pass
Rotator Connector	RC1 RC2 RC3 RC4	handleConnect()	Attempts to connect to rotator	"slakdgo" "102.2.4.2.4" "123.123.123.123" "192.168.50.100"	1. Click "Connect a Rotator" 2. setup server to 192.168.50.100 3. enter test case 4. Press "Connect"	"Invalid IP" "Invalid IP" "Timed Out" "Connected Successfully"	"Invalid IP" "Invalid IP" "Timed Out" "Connected Successfully"	Pass Pass Pass Pass
	RC5 RC6	handleCancel()	Clears input field and errors	N/A	1. Click "Connect a Rotator" 2. Press "Cancel"	console.log(openDialog) => false console.log(error) => ""	false ""	Pass Pass
	RC7 RC8 RC9	handleClick()	Opens/closes rotator dialog	isTracking= False connected = False isTracking= False connected = True isTracking= True connected = True	1. reach test case state 2. press "Connect/Disconnect"	Dialog Box Opens Disconnects Rotator Nothing	Dialog Box Opens Disconnects Rotator Nothing	Pass Pass Pass
	RC10	handleInputChange()	Sets IP address state on input	"123.123.123"	1. Press "Connect" 2. type in test case into field	console.log(ipAddress) => "123.123.123"	"123.123.123"	Pass
MapSettings	MS1 MS2	handleClose()	Closes map settings popup	N/A	1. Mount application 2. Click Map Setting Icon (wrench) 3. Click the Icon again to Close	Map Settings pops up to the left of the Icon Popup closes	Map Settings pops up to the left of the Icon Popup closes	Pass Pass
	MS3 MS4	handleShowGridChange()	Toggles visibility of grid layer	N/A	1. Mount application 2. Click Map Setting Icon (wrench) 3. Click "Grid" to show 4. Click "Grid" to Close	grid shows on switch right grid disappears on switch left	grid shows on switch grid disappears on switch	Pass Pass
	MS5 MS6	handleShowDayNightChange()	Toggles visibility of terminator layer	N/A	1. Mount application 2. Click Map Setting Icon (wrench) 3. Click "DayNight" to show 4. Click "DayNight" to Close	terminator shows on switch right terminator disappears on switch left	terminator shows on switch terminator disappears on switch	Pass Pass
	MS7 MS8	handleShowGroundTrackChange()	Toggles visibility of ground track	N/A	1. Mount application 2. Type in known satellite in search bar 3. Click on suggestion 2. Click Map Setting Icon (wrench) 3. Click "Ground Track" to show 4. Click "Grid" to Close	Ground track shows on switch right Ground track disappears on switch left	Ground track shows on switch right Ground track disappears on switch left	Pass Pass

Table 8: React UI Tests Continued

Component	ID	Method	Description	Test Cases	Steps	Expected	Actual	Result
Mercator Map	MM1 MM2	handleToggleMapSettings()	Opens or closes map settings	Settings open = false Settings closed = true	1. Mount application 2. Click Map Setting Icon (wrench) 3. Click the Icon again to Close	Map Settings pops up to the left of the Icon Popup closes	Map Settings pops up to the left of the Icon Popup closes	Pass Pass
Dropdown Menu	DM1 DM2 DM3 DM4	handleToggleMenu()	Handles menu toggle event	Satellite = null Rotator = null Satellite != null Rotator = null Satellite = null Rotator != null Satellite != null Rotator != null	1. Reach the test case state by connecting to rotator/selecting satellite 2. Click Control Menu button in the top left corner (Slider Icon)	Warning Message Warning Message Warning Message Control Menu Opens	Warning Message Warning Message Warning Message Control Menu Opens	Pass Pass Pass Pass
	DM5 DM6	handleCloseMenu()	Handles control menu close event	Azimuth = 120 Elevation = 10 Azimuth = 1210 Elevation = 110	1. Connect a rotator and choose a satellite 2. Click Control Menu button in the top left corner (Slider Icon) 3. enter test case into text field 4. click away from control menu	Azimuth = 120 Elevation = 10 Azimuth = "" Elevation = ""	Azimuth = 120 Elevation = 10 Azimuth = "" Elevation = ""	Pass Pass
	DM7 DM8 DM9 DM10	handleCommand()	Sends start/stop tracking command	isTracking = False command = "Start" isTracking = False command = "Stop" isTracking = False command = "" isTracking = True command = "Start"	1. Connect a rotator and choose a satellite 2. Click Control Menu button in the top left corner (Slider Icon) 3. click "autoTrack" 4. reach test case state by using commands	"Rotator has started auto-tracking" Client Request:: StartAuto "Rotator has stopped tracking" Client Request:: Stop "HTTP/1.1 404 Not Found" No Change	"Rotator has started auto-tracking" Client Request:: StartAuto "Rotator has stopped tracking" Client Request:: Stop "HTTP/1.1 404 Not Found" No Change	Pass Pass Pass Pass
	DM11	handleCloseSnack()	Clears snack message popup	N/A	1. Click Control Menu button in the top left corner (Slider Icon)	Snack bar message displays and disappears	Snack bar message displays and disapears	Pass
	DM12 DM13	handleAzimuth()	Validates and sets azimuth input	Azimuth = 120 Elevation = 10 Azimuth = 1210 Elevation = 110	1. Connect a rotator and choose a satellite 2. Click Control Menu button in the top left corner (Slider Icon) 3. enter test case	True False	True False	Pass Pass
	DM14 DM15	handleElevation()	Validates and sets elevation input	Azimuth = 120 Elevation = 10 Azimuth = 1210 Elevation = 11330	1. Connect a rotator and choose a satellite 2. Click Control Menu button in the top left corner (Slider Icon) 3. enter test case	True False	True False	Pass Pass
	DM16 DM17	handleToggleAutoTrack()	Toggles tracking control and resets	isTracking = False Azimuth = 120 Elevation = 10 isTracking = True	1. Connect a rotator and choose a satellite 2. Click Control Menu button in the top left corner (Slider Icon) 3. click Autotrack	AutoTrack Switch Toggles On Azimuth = "" Elevation = "" AutoTrack Switch Toggles Off	AutoTrack Switch Toggles On Azimuth = "" Elevation = "" AutoTrack Switch Toggles Off	Pass Pass

Table 9: UX Design Survey

Question	Were you able to accomplish your tasks efficiently and effectively?	Was the web app easy to navigate and use?	Did you find the design of the web app visually appealing?	Were the layout and organization of the web app intuitive and easy to follow?	Did you encounter any issues with the design or layout that made it difficult to use the web app?
Respondent A	yes	yes	no	yes	yes
Respondent B	yes	yes	yes	yes	no
Respondent C	yes	yes	yes	yes	no
Respondent D	yes	yes	yes	yes	no
Respondent E	yes	yes	no	yes	yes
Respondent F	no	yes	no	yes	no
Respondent G	yes	yes	yes	yes	no
Rate	85.71%	100.00%	57.14%	100.00%	28.57%

Table 10: UX Performance Survey

Question	Did the web app have all the necessary features you needed?	Were the features and functionality easy to understand and use?	Did you encounter any issues or bugs while using the features?	Did the web app load quickly and respond to your interactions in a timely manner?	Did you experience any lag or delays while using the web app?	Did the web app perform as you expected it to?
Respondent A	yes	no	no	yes	Yes	yes
Respondent B	yes	yes	no	yes	no	yes
Respondent C	yes	yes	no	yes	no	yes
Respondent D	yes	yes	no	yes	no	yes
Respondent E	yes	yes	yes	yes	no	yes
Respondent F	yes	no	yes	yes	yes	yes
Respondent G	yes	yes	no	yes	no	yes
Rate	100.00%	71.43%	28.57%	100.00%	28.57%	100.00%

Figure 16: Web App Latency

Table 11 Web App Execution Timeline

Date	Task
Feb 15	Finalized features with sponsor
Feb 26	Finalized wireframe
Feb 27	Begun working on toolbar
Feb 27	Added Search Bar, Control Menu Icon, Connect Button
March 3	Created method for searching for satellites and dropdown suggestions
March 4	Added control menu dropdown
March 4	Added control menu switch/fields
March 4	Added Dialog Box to connect to MCU
March 5	Added IPaddress verification
March 5	Created Method for fetching orbital data
March 5	Styled Toolbar
March 5	Styled Dialog Box
March 10	Created Dialog Box Graphic
March 14	Figured out how to send props to React children
March 14	Built test esp8266 server
March 14	Created fetching for mcu server
March 14	Finished Connect/Disconnect Button

Table 12 Web App Execution Timeline Continued	
Date	**Task**
March 17	finished Satellite helpers (orbital functions)
March 17	created helper for rotator object
March 18	Signed up for Arcgis developer account and began mercator map
March 20	Added Zoom button and stylized base map
March 20	Finished satellite and rotator graphics in adobe illustrator
March 24	Added MapSettings Button/Logic
March 25	Started creating map helper to add/render graphics
March 25	figured out how to create /plot paths on arcgis
April 1	Finished “get” functions for map
April 3	Figured out how to import png files into map
April 7	Added Rotator Icon
April 8	Started Validation Plan for UI
April 10	Started Validation Plan for usability
April 18	Finished Validation

4.4 Web App Conclusion

The subsystem works as intended and can be observed in the demo on page 15. Note that there is no connection to an actual ESP8266 server as that is a feature that will be implemented next semester. Overall the web application works as expected and is ready for integration however there are a few changes I would like to make in order to improve the user experience such as adjusting certain styles and making it more responsive. As for validation, there were 76 unit tests that all passeds successfully that covered error handling. ui behavior, and helper functions. Most of the usability surveys came back positive but they were only yes or no questions so I would like to get some more in depth feedback. Latency was mostly under 100ms which is reasonable, as any more would make the web application less user friendly. The only significant delays were with the virtual private networks (VPNs) which ran up to 188 ms.

Appendix A: Acronyms and Abbreviations

AC	Alternating Current
DC	Direct Current
SPST	Single Pole Single Throw
MOSFET	Metal-Oxide Field-Effect Transistor
FSR	Functional System Requirements
MAE	Mean Absolute Error
kHz	Kilohertz (1,000 Hz)
mA	Milliamp
mW	Milliwatt
PCB	Printed Circuit Board
MCU	Microcontroller Unit
GPS	Global Positioning System
LCD	Liquid Crystal Display
Rx	Receive
Tx	Transmit
SDA	Serial Data Line
SCL	Serial Clock Line
MHz	Megahertz
GND	Ground
JSON	Javascript Object Notation
SSPA	Serverless Single-Page Application
HTML	Hypertext Markup Language
ICD	Interface Control Document
HTTP	Hypertext Transfer Protocol
TLE	Two Line Elements
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
VPN	Virtual Private Network