EMBER BOT

Jonathan Chen, Kevin Rivera, Nancy Ramirez Castillo, Yuwen Zheng

CONCEPT OF OPERATIONS

REVISION – 3
24 April 2025

CONCEPT OF OPERATIONS
FOR
EMBER BOT

PREPARED BY:

TEAM #30

APPROVED BY:

_____
Jonathan Chen 4/24/25

_____
Prof. Kalafatis 4/24/25

_____
Swarnabha Roy 4/24/25

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/5/2025	Jonathan Chen		Draft Release
2	4/24/2025	Jonathan Chen		Minor Edits for Final Report

Table of Contents

List of Tables

No tables in this document.

List of Figures

1. Executive Summary

This project's purpose is to create a solution regarding the rising concerns of fires in dangerous or hard-to-reach locations where first responders would be unavailable. Ember Bot, a mobile-controlled fire-fighting vehicle, is an innovative solution designed to enhance the safety and efficiency of firefighters by addressing challenges posed by hazardous environments. This remotely operated robot uses wireless technology to enable real-time control, allowing it to perform tasks in areas too dangerous or inaccessible for human responders. Equipped with fire-extinguishing mechanisms and fire detection, the vehicle can combat fires in high-risk scenarios. Its compact and agile design enables it to navigate through narrow pathways, debris-filled areas, or uneven terrains, providing a versatile tool for emergency response. The onboard sensors and camera allow it to monitor environmental conditions and provide critical data to operators. By minimizing the exposure of human firefighters to extreme heat, toxic fumes, or unstable structures, the controlled fire-fighting vehicle significantly reduces the risk of injuries and fatalities. Its ability to operate remotely in life-threatening conditions makes it a valuable tool for disaster management and emergency response teams. Additionally, this design scope can be adapted and improved for varying first responder scenarios, from urban environments to industrial facilities.

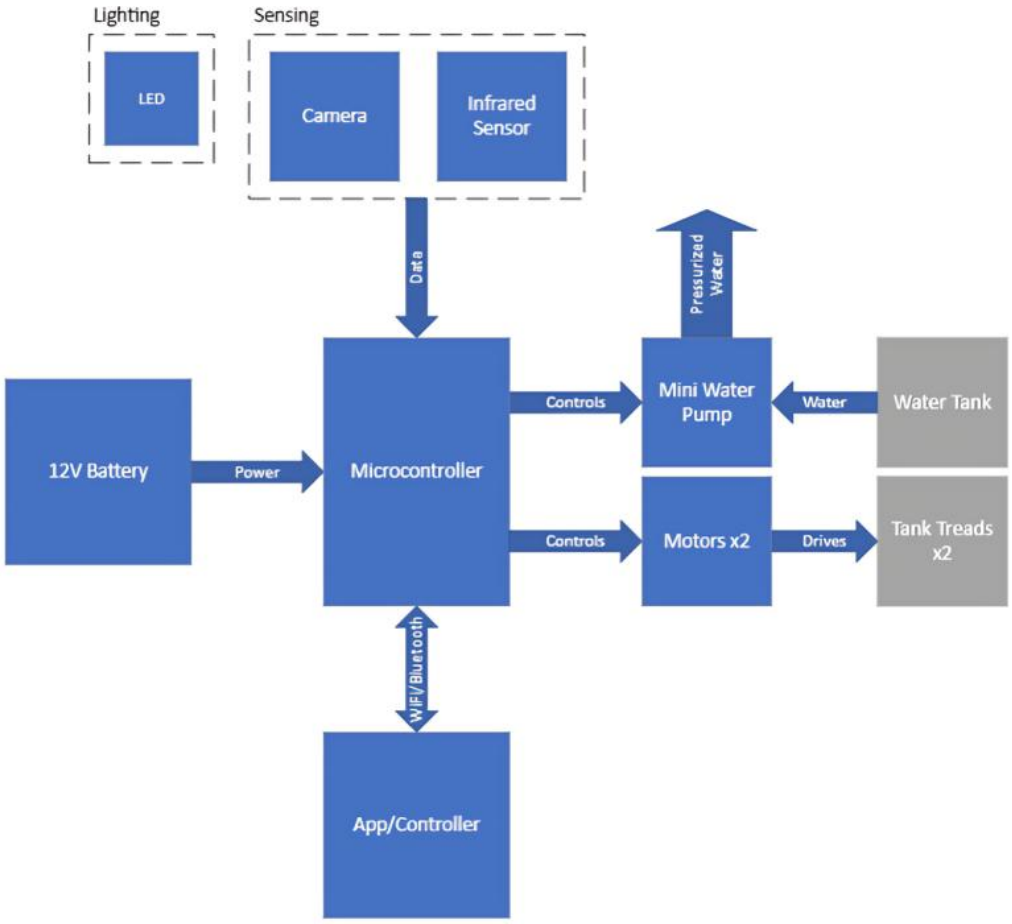


Figure 1. System Diagram of Ember Bot

2. Introduction

In fire emergencies, rapid response and safe intervention are critical. This document introduces Ember Bot, a firefighting robotic vehicle that aims to provide a remotely operated solution for firefighting in unfavorable environments. This robotic vehicle is designed to detect and extinguish fires while minimizing risks to human firefighters.

2.1. Background

Firefighting is a critical operation that often requires personnel to enter dangerous environments, exposing them to extreme heat, toxic smoke, and potential structural collapses. Traditional firefighting methods rely heavily on human intervention, with firefighters manually operating hoses and suppression systems to combat fires. However, these conventional approaches come with significant limitations, including restricted reach, possible loss of personnel, and potential delays in response time, especially in hazardous or hard-to-access locations.

Ember Bot is designed to serve as a versatile and efficient tool for fire suppression in a variety of environments. It can be deployed in residential, commercial, and industrial settings where fire hazards pose significant risks [1]. The ability to remotely control the vehicle makes it especially useful in non-traditional areas, where human intervention may be dangerous or impractical. Beyond emergency response, the system can also be integrated into fire prevention strategies, serving as a preemptive guard against fire-prone locations to detect and respond to potential hazards before they escalate. The adaptability of this robotic vehicle allows it to be customized for different situations, making it a valuable addition to firefighting teams, industrial safety protocols, and other operations.

A key component of our vehicle is introducing remote-controlled operation via a mobile application. This feature allows operators to navigate the robot toward fire-prone areas from a safe distance, ensuring that human exposure to extreme conditions is minimized. The use of Bluetooth and/or WIFI communication provides a stable and reliable control mechanism, unaffected by obstructions or interference that might compromise other wireless technologies in emergency scenarios, such as smoke and connection issues [2].

Additionally, other systems specifically designed include the mobility of our robotic vehicle via threads and the use of a pressurized pump for manual water dispersal. These systems allow it to access areas that may be difficult or dangerous for firefighters to reach. The combination of real-time maneuverability, remote operation, and a controlled suppression system makes our vehicle a valuable enhancement to existing firefighting infrastructure.

2.2. Overview

Ember Bot will consist of a mobile-controlled robotic platform vehicle equipped with fire suppression components. The system will integrate wireless communication, allowing the user to navigate the vehicle and activate the fire suppression mechanism through a mobile application. The robot will receive movement and operation commands through a mobile app, which will process the signals and control the vehicle's mobility and firefighting functions. The onboard pressurized pump will enable manual water dispersal, providing precise control over the firefighting process. Real-time data from the onboard infrared sensors can assist in identifying fire sources and optimizing suppression strategies. The system will improve fire response capabilities by allowing users to safely operate the vehicle from a distance, reducing risks associated with direct human intervention. Over time, the system's usage data and performance metrics can be analyzed to refine fire suppression techniques, improve operational efficiency, and inform future design enhancements. The diagram below illustrates the flow of control and communication within the system.

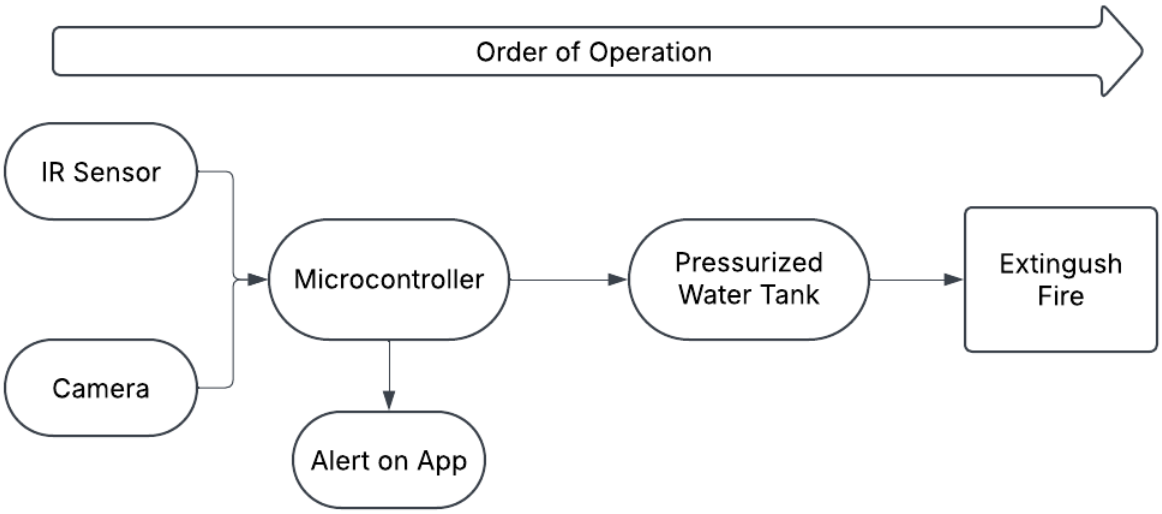


Figure 2. Order of Operation of Fire Suppression

Additional functionality may be considered depending on the goals of the vehicle.

2.3. Referenced Documents and Standards

[1] Lattimer, B. Y. (n.d.). "*Robotics in Firefighting"*. SFPE. https://www.sfpe.org/publications/fpemagazine/fpeextra/etarchives3/fpeetissue100

[2] "*Where there's smoke, there's a signal"*: Homeland Security. U.S. Department of Homeland Security. (n.d.). https://www.dhs.gov/archive/where-theres-smoke-theres-signal

3. Operating Concept

3.1. Scope

For the scope of this project, a list of subsystems is being designed that will be combined at the end so that the Fire Fighting Robotic Vehicle can function as needed to achieve our primary goals for this project. The exact subsystems for this project are as follows:

- Microcontroller Subsystem
- Vehicle Operations & Water Pump Subsystem
- Power Subsystem
- Mobile App Subsystem

Documentation for the design, calculation, programming, and construction of different subsystems will be provided for all parts of the project.

3.2. Operational Description and Constraints

Ember Bot will be used as an efficient tool for firefighters in different scenarios. If a fire is in areas that are too dangerous for firefighters to access, Ember Bot will be enabled and perform tasks in these dangerous areas. The vehicle can also be used when the space is too narrow for a firefighter to get into. It can be controlled with a real-time mobile app to put out fires in these high-risk areas. Scenarios of the possible usage of Ember Bot include but are not limited to:

- Burning buildings that are likely to collapse
- Buring industries with hazardous materials
- Tunnel fire
- Shipboard fire

Ember Bot is not designed to be used in regions where the controller is too far away from the fire or complex regions that will limit the movement of the Robotic Vehicle.

3.3. System Description

Ember Bot is combined with four subsystems including the Microcontroller Subsystem, Vehicle Operations & Water Pump Subsystem, Power Subsystem, and the Mobile App Subsystem. The components the four subsystems use are described below:

Microcontroller: The microcontroller serves as the central processing unit of Ember Bot where it will receive commands from the mobile application and control the motors, light source, IR sensors, camera, and pressurized pump accordingly. The microcontroller processes sensor data to assist in fire detection, pressure output, and navigation, ensuring smooth and efficient operation.

Figure 3. ESP-32 Microcontroller

IR Sensor: The Infrared Sensor is used for detecting fire and obstacles by sensing infrared radiation emitted from heat sources. It helps identify the presence of a fire by detecting temperature variations and can assist in guiding the robot toward the fire.

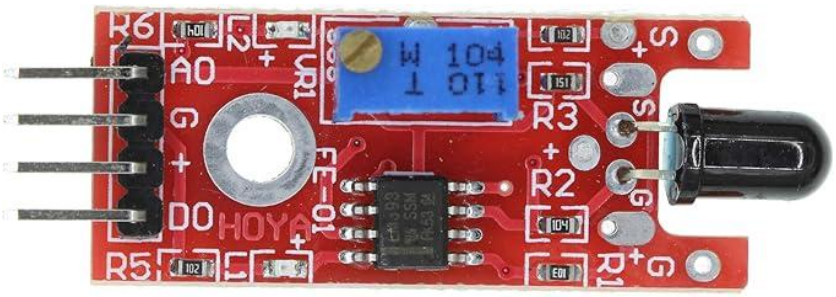


Figure 4. KY-026 IR Sensor

Camera: The camera module provides real-time visual feedback to the operator, enabling remote monitoring and precise control of the robot. The live video feed helps in navigating through hazardous areas and accurately aiming the fire suppression system. It also enhances situational awareness by allowing the operator to manually assess the fire's intensity and surroundings.

Figure 5. Camera Tool

Light Source: The light source is an essential component of Ember Bot, primarily used for visibility enhancement in low-light or smoke-filled environments. It helps the onboard camera capture clearer images for the operator, improving navigation and fire suppression accuracy.

Figure 6. 5V LED Light Source

Vehicle and Pump: The vehicle itself is what will be holding all our robotic vehicles' components and the water tank. Tank treads with motors will be used for mobility in possibly rough terrain. The water tank will hold half a gallon of water and have a cap for it to be refilled. Electronics will also be isolated to ensure no damage during use. A water pump will be used to shoot pressurized water out of a nozzle to extinguish the fires. The nozzle will be able to be aimed at using servo motors to ensure accuracy in hitting its target.

Figure 7. Tank Tread Motor & Water Pump

Power: The entire system is powered by a 12V battery providing a reliable and efficient energy source for all components. The system will use a rechargeable battery to ensure continuous operation without the need for constant replacement and extra cost.

Figure 8. Rechargeable 12V 7A Battery

App and Controller: The mobile app is designed to fulfill all the needs of controlling the robotic vehicle for better task completion. The mobile app will provide real-time recordings for users to capture the fire. The user, a designated person who can easily access their device, can control the robot to move around and put out the fire on this App. The app will be developed using Visual Studio and Dart Programming Language. The app will be able to function on iOS and Android devices.

3.4. Modes of Operations

Ember Bot shall use a primary mode of operation where the vehicle will provide real-time recordings of surrounding areas to users. The user can drive the robotic vehicle around until the fire is captured. Then the user can control the pump on the vehicle to put out the fire. The user can also simply tap on the location of the fire shown on the app. The robotic vehicle can aim automatically and put out the fire in that direction.

Ember Bot will also have warning notifications sent to the app when the battery is running low, or the robot is too far away from the user. Such notifications can lower the cost of maintenance of the robotic vehicle.

3.5. Users

Ember Bot is primarily designed as an efficient tool for designated firefighters when they have high-risk tasks or when they meet inaccessible areas. However, this robotic vehicle can also be used by adults of different ages due to its simple operation and reasonable price as an alternative safety measure.

3.6. Support

Manuals will be provided to users and there will also be instructions in the app so the user can access it immediately when needed. The manual will guide users on how to connect their mobile phone to the robotic vehicle and how to operate the robotic vehicle. The manual will also include possible questions and answers about the robotic vehicle.

4. Scenario(s)

4.1. Assistance in Extinguishing Flames

Ember Bot will be able to assist firefighters in reaching hard-to-reach fires. Firefighters can deploy this robot to put out fires present behind tight openings or in hazardous areas such as areas with a risk of falling debris or potential explosions. Using a camera and infrared sensor, users can navigate the robot and use pressurized water stored in the robot to locate and extinguish the fire.

4.2. Search and Rescue

Using the built-in camera and wireless control, first responders can use Ember Bot to assist in Search and Rescue operations. The small size and tank tread wheels give operators the ability to go into hard-to-reach areas to find potentially trapped individuals. Operators can view the live stream on their phones and use an LED to light their way.

5. Analysis

5.1. Summary of Proposed Improvements

- Ember Bot will allow first responders to combat fires that source from narrow pathways, debris-filled areas, or uneven terrains minimizing human exposure to hazardous environments.
- By responding quickly to fires in inaccessible areas for first responders, the system can help prevent fire escalation and reduce property damage.
- Being equipped with cameras and sensors, the system provides live data and situational awareness to operators while the system is adaptable to various environments, and using wireless technology ensures stable control, even in conditions where other wireless signals may be disrupted.

5.2. Disadvantages and Limitations

- Since the robot is restricted by wireless communication range, this means that it cannot be in scenarios that are too far from the operator.
- Although we will be using tank treads to facilitate movement compared to standard wheels, complex terrains or heavily obstructed environments may limit the robot’s movement and effectiveness.
- The water supply on the robot is finite, requiring refilling for prolonged operations. In situations where fire cannot be extinguished or maintained, having to refill the water tank will allow the fire to spread rapidly in the time it takes the vehicle to return to the operator, fill up, and return to the fire area.
- The system relies on human input through an app for navigation and fire suppression which makes it prone to human error or malfunctioning. Programming the vehicle to be fully autonomous operating will minimize human error and improve efficiency.

- The robot's operation time relies on battery capacity, if not recharging properly will make the unit inefficient for emergencies.

5.3. Alternatives

Several alternative approaches exist for fire extinguishing in hazardous environments. One alternative is drones with fire suppression that can access elevated areas and be deployed faster. However, their payload is limited, restricting the amount of fire-extinguishing material they can carry. There are also traditional firefighting methods like fire hoses and sprinkler systems, which are widely used, however, these methods rely on human interaction and are inefficient in hard-to-reach places. Another option is fully autonomous firefighting robots that utilize AI-based detection. Although this system requires little to no human interaction, they are significantly more expensive and complex to build and maintain. Ember Bot combines the advantages of remote operation with affordability, its ability to easily navigate through different terrains while being controlled in real-time makes it a practical alternative for first responders.

5.4. Impact

Ember Bot has environmental, societal, and ethical implications. Starting with environmental impact, the system minimizes water waste by targeting the base of the fire to extinguish it faster. By enabling rapid response to fires, it will also help reduce the environmental damage from uncontrolled fires. Now from a societal perspective, the vehicle enhances firefighter safety by reducing exposure to life-threatening conditions. It is essential to ensure the vehicle's functions are working efficiently in critical situations to prevent jeopardizing emergency response efforts, which poses an ethical concern. To emphasize, this system is not designed to replace firefighters but rather to assist them.

EMBER BOT

Jonathan Chen, Kevin Rivera, Nancy Ramirez Castillo, Yuwen Zheng

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – 3
24 April 2025

FUNCTIONAL SYSTEM REQUIREMENTS
FOR
Ember Bot

PREPARED BY:

TEAM #30

APPROVED BY:

_____
Jonathan Chen 4/24/25

_____
John Lusher, P.E. 4/24/25

_____
Swarnabha Roy 4/24/25

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/20/2025	Jonathan Chen		Draft Release
2	4/12/2025	Jonathan Chen		Additional Performance Requirements Added
3	4/24/2025	Jonathan Chen		Final Report

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Purpose and Scope

This document defines the detailed technical requirements for Ember Bot. The purpose of this project is to create a specific targeted solution to the dangers of fires located in hard-to-reach areas or when first responders are unavailable. Designed to be used by fire-fighters primarily, Ember Bot provides any user with the ability to deal with small-scale fires that are unable to be dealt with by normal means. It aims to improve firefighter safety and efficiency by overcoming challenges posed by hazardous situations. Figure 1 demonstrates the functionality of Ember Bot, as integrated within the proposed CONOPS. The verification requirements for Ember Bot are continued in a separate Verification and Validation Plan.

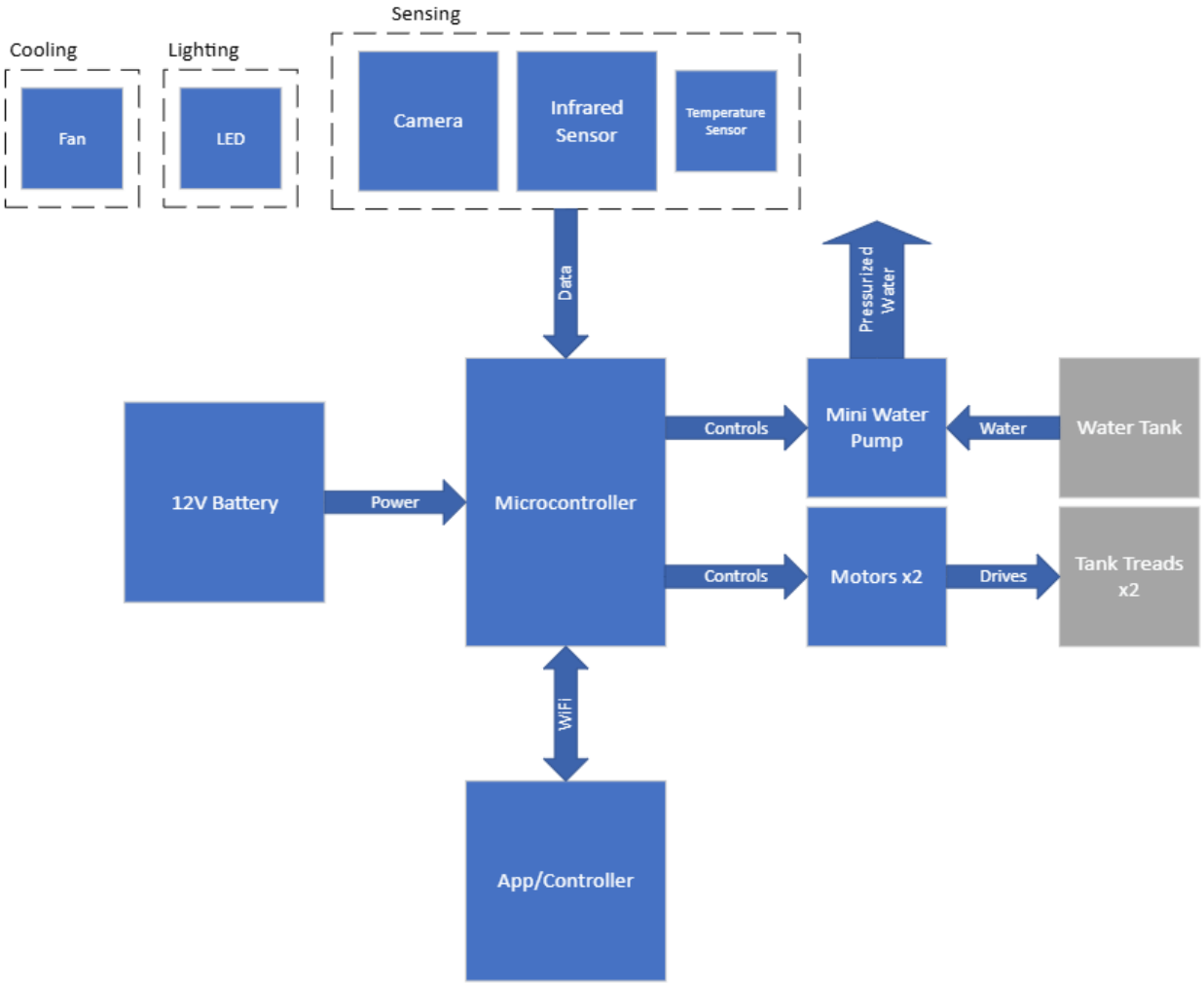


Figure 1. Functional System Diagram of Ember Bot

1.2. Responsibility and Change Authority

The team leader, Jonathan Chen, is responsible for ensuring that all specifications and requirements of Ember Bot are met. Any changes to the team's deliverables of the project must be approved by the team leader, Jonathan Chen, the sponsor, Zian Wang, and our designated TA, Swarnabha Roy. Subsystem owners are also accountable for ensuring their subsystem meets all requirements. The subsystem distribution is as follows:

- Jonathan Chen: Microcontroller and Fire Detection (IR & Camera)
- Yuwen Zheng: Application and Controller Design
- Nancy Ramirez Castillo: Power System
- Kevin Rivera: Car Design and Pump System

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 1. Applicable Documents

Document Number	Revision/Release Date	Document Title
IEC 61508-1:2010	4/1/2010	Functional Safety of Electrical/Electronic Systems
C2-2017	2017	National Electrical Safety Code (R)
ISO 13849-1:2015	11/1/2015	Safety of Machinery – Safety-Related Parts of Control Systems
ISO 10218-1:2011	2011	Robots and Robotic Devices – Safety Requirements for Industrial Robots
IEC 61508-5:2010	2010	Functional safety of electrical/electronic/programmable electronic safety-related systems
NFPA 70:2023	2023	National Electrical Code (NEC)

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 2. Reference Documents

Document Number	Revision/Release Date	Document Title
IEEE Standard 802.15.4-2015	2015	Standard for Low-Rate Wireless Personal Wireless Network
IRJET-V612417	February 2019	Design & Implementation of RF-based Fire Fighting Robot

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

3. Requirements

Ember Bot, the fire-fighting robotic vehicle, is a proof of concept for additional assistance that could be deployed in the field by firefighter responders. In the following section, “fire detection” will refer to the system responsible for gathering the information to relay to the user. This includes infrared sensors and the camera. The term “vehicle operations” will refer to the system responsible for physical components within the vehicle such as movement, lighting, water nozzle control, and a fan and temperature sensor for cooling. The term, “app interface” will refer to the software program that users would use to control robotic vehicle operations.

3.1. System Definition

The Ember Bot is made of four subsystems, which are shown in Figure 2. The power subsystem has parts in both vehicle operations and fire detection. The detection and communication subsystem are exclusive to the fire detection portion. The lighting and cooling, water pump, and motor subsystems are also exclusive to the vehicle operations portion.

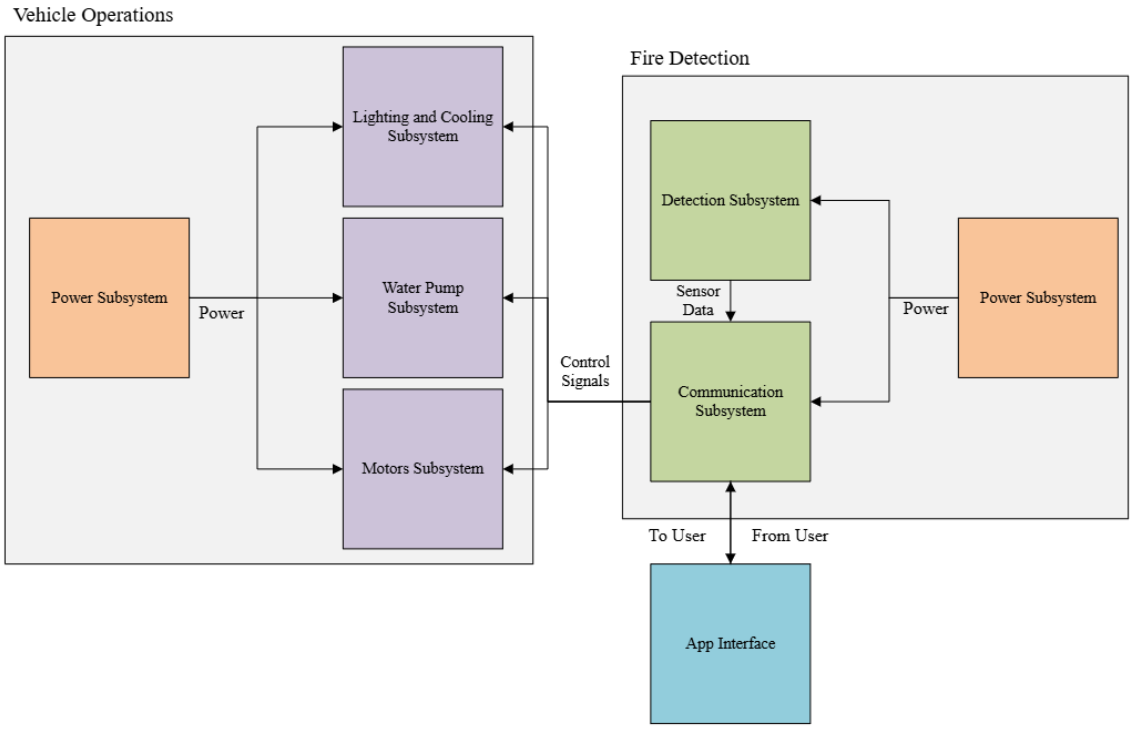


Figure 2. Block Diagram of System

The power subsystem consists of two 11.1V/2.2Ah lithium-ion batteries connected in series to provide enough power to the DC motors. A buck-converter steps down the voltage as needed for the sensors and ESP32. Additionally, the power subsystem will include a

standard wall plug-in for the recharging of batteries, and PCB design to distribute power efficiently and integrate all components.

The motors subsystem contains two 24V DC motors and a dual-channel motor driver. There is one motor on each side of the robotic vehicle, driving a wheel connected to tank treads. The tank treads will give the robot the versatility of traversing potentially rough terrains. The DC motors will be driven by a dual-channel motor driver receiving signals from the communication subsystem.

The detection subsystem in Ember Bot gathers visual data and transmits it to the communication subsystem. It utilizes infrared sensors and a camera to detect environmental conditions and obstacles. The infrared sensors provide data on temperature variations and heat signatures, while the camera captures visual information for further processing. This data is then relayed to the communication subsystem, enabling real-time monitoring and decision-making.

The communication subsystem will collect sensor data and transmit it to the mobile app via Bluetooth. It will also receive commands from the app to control the water pump and motor subsystems, ensuring seamless operation by relaying instructions to the appropriate components and the desired subsystem.

For the water pump subsystem, there is a DC water pump and two servos to move the water nozzle. Both would receive signals from the communication subsystem. The water pump can be turned on and off to control the water output. The two servos can move the water nozzle to aim it wherever the user desires.

The lighting and cooling subsystem contains an LED strip and a cooling fan. The LED strip would be lighting the way in front of the robot to assist in driving the robot. A cooling fan will be on during operation to ensure all electronics are kept within the operating temperature range.

The mobile app subsystem displays real-time video recordings and data gathered from Ember Bot. Users can send the commands via a mobile app so that Ember Bot can locate and put out fires. The mobile app subsystem will contain user manuals and help pages to provide additional assistance.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1. Communication Requirements

The Ember Bot should continuously send real-time video streams and data to the mobile app. It should also be able to continuously receive user commands sent from the mobile

app. The Ember Bot should be able to operate and function within a maximum distance of 200ft (indoor) to 300 ft (outdoor).

Rationale: The maximum communication distance is limited to the maximum operation distance of Wi-Fi.

3.2.1.2. Mobile App Warning Requirements

Under any circumstances, the mobile app should generate a warning within 30s to inform the user of any issues such as battery running low or other emergency incidents Ember Bot may face. Warnings can provide users with information about how to troubleshoot different problems and take the next steps.

Rationale: Thirty seconds allows the Ember Bot systems to predict and analyze different incidents.

3.2.1.3. Power Requirements

Ember Bot should be able to operate reliably on a rechargeable battery system, with the capability to supply continuous power to the microcontroller, motors, pump system, sensors, and communication modules when powered on. The system requires stable voltage rails of 24V, 12V, and 5V with sufficient current capacity to support simultaneous operation of all subsystems during peak load. Battery charging must be efficient and safe, ensuring minimal downtime between operations.

Rationale: Stable power delivery is needed to ensure uninterrupted operations of critical functions, and is essential for maintaining mobility, communication, and fire suppression capabilities.

3.2.1.4. Vehicle Operation Requirements

Ember Bot should be able to travel in potentially dangerous environments without facing issues in travel or damage to the electronics inside. Ember Bot should have a maximum speed of 6.4ft per second to ensure quick response. The frame must also be waterproof to ensure that the electronics housed inside cannot be damaged in the event of a water tank leak.

Rationale: Under the max duty cycle for the PWM signal that controls the motor speeds, Ember Bot can travel up to 6.4ft per second.

3.2.1.5. Water Pump Requirements

Ember Bot should be able to accurately spray water at a maximum of 6ft away. This allows Ember Bot to stay a safe distance from the flame. The water nozzle movements provided

by the servo motors must also be able to step in 1.5-inch intervals with an accuracy of ±0.5 inches. An increase in accuracy will decrease water usage in the event of putting out the flame.

Rationale: The water pump and high-pressure water hose nozzle begin losing overall water concentration in areas past 6ft. With this distance, 1.5-inch intervals can provide accurate coverage while keeping the servos from overshooting the angle.

3.2.2. Physical Characteristics

3.2.2.1. Mass

The mass of the Ember Bot shall be less than or equal to 21 lbs.

Rationale: This would allow users to be able to easily carry and deploy it in the field.

3.2.2.2. Volume Envelope

The volume envelope of the Ember Bot shall be less than or equal to 9 inches in height, 14 inches in width, and 15 inches in length.

Rationale: A lower design would allow users to be able to drive the robot under debris or collapsed areas. The wide design will also allow the robot to carry more water inside.

3.2.2.3. Mounting

The mounting information for the Ember Bot subsystems shall be captured in the Ember Bot ICD.

Rationale: As the Ember Bot subsystems mount to the chassis, the interface between the two includes mechanical, electrical, and thermal details.

3.2.3. Electrical Characteristics

3.2.3.1. Inputs

a. The presence or absence of any combination of the input signals under ICD specifications applied in any sequence shall not damage the Ember Bot, 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 Ember Bot, reduce its life expectancy, or cause any malfunction.

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

3.2.3.2. Power Consumption

a. The maximum peak power of the system shall not exceed 48.84 watts.

Rationale: This is a requirement to ensure continuous operation for motors, but this power should not directly go to smaller components in the power circuit.

3.2.3.3. Input Voltage Level

The input voltage level for the Ember Bot shall be 22.2V.

Rationale: The highest voltage of 25.2V is only for the two DC motors; all other components will have a step-down voltage of 12V, 3.3V, or less.

3.2.3.4. Connectors

The Ember Bot will have a wall plug input connector that will recharge the batteries when not in use.

3.2.3.5. External Commands

The Ember Bot 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.

3.2.4. Outputs

3.2.4.1. Data Output

The Ember Bot will be able to show the user the strength of the Wi-Fi signal in real time so that Ember Bot does not travel outside the maximum distance for Wi-Fi access.

Rationale: The signal information shows directly on the Mobile App Control page to make sure users do not lose control of Ember Bot.

3.2.4.2. Diagnostic Output

The users will be able to see warnings on the Mobile App when there are any problems with the Ember Bot. The warnings will show information to help identify and troubleshoot different problems.

Rationale: The user can get immediate help and guidance on both operation and troubleshooting Ember Bot.

3.2.4.3. Raw Video Output

The Mobile App will show real-time recordings from the Ember Bot for detecting fire and controlling the Ember Bot to put out the fire.

Rationale: Users can directly locate the fire and take action.

3.2.5. Environmental Requirements

3.2.5.1. Pressure (Altitude)

The Ember Bot is designed to operate under air pressure at an altitude from sea level (0 ft) to 8751 ft. All the sensors used in the design of Ember Bot will not be affected by altitude.

Rationale: The highest natural point in Texas is 8751 ft (The Guadalupe Peak).

3.2.5.2. Thermal

Ember Bot, as a proof of concept, shall be designed to withstand and operate in temperatures of -25°F to 185°F. Depending on the purpose of Ember Bot, varying materials and components can be swapped out to withstand much higher temperatures.

Rationale: As the robot is designed to be deployed in environments experiencing fires, it must be able to operate in high-temperature conditions, including but not limited to a small house fire. Ember Bot can be scaled with better materials and components to deal with varying types of fires. A cooling fan will be used to keep the temperature in this range.

3.2.5.3. External Contamination

The Ember Bot is designed to operate in all conditions of weather and contamination. Dust, pollen, and insects will not interfere with the operation of Ember Bot. However, weather and contamination such as haze and fog may affect the visibility of Ember Bot. Icy roads may cause Ember Bot to slip.

Rationale: As the Ember Bot is designed to put out fires, it will be able to operate in extreme weather and conditions. However, it is outside the scope of this project for Ember Bot to fight against extreme natural forces such as tornados and hail.

3.2.5.4. Rain

The Ember Bot is designed to put up a fire using a pressurized water pump. All the subsystems of Ember Bot are required to be protected from water and rain. However, flooding may prevent Ember Bot from moving and thus can destroy the interconnection of each subsystem.

Rationale: It is unlikely that Ember Bot is operated in rain or flood due to fire hazards that may not exist under these conditions.

3.2.5.5. Humidity

The Ember Bot is designed to operate in a relative humidity range from 5% to 70%.

Rationale: This range includes considerations like temperature and humidity levels in cities like Texas. This range also considers sea water since salt will accelerate the rust of steel. Designing a system outside this humidity range is outside the scope of this project.

3.2.6 Failure Propagation

The Ember Bot is designed to complete tasks within the control of the users. Each subsystem is designed to prevent failure propagation.

3.2.6.1 Power Supply Failure

Battery depletion, voltage drops, or buck converter failure will lead to the ESP32 shutting down, the water pump burning out, motors stalling, and sensors failing making Ember Bot inoperable in critical scenarios.

3.2.6.2 Connectivity Failure

Ember Bot will be controlled on a phone via Bluetooth that can range to a maximum of 70 meters. If the app malfunctions or loses connections, the bot will not receive commands leading to an inability to navigate or activate the water pump if near fire.

3.2.6.3 No Fire Response

Fire detection comes from the user viewing the IR sensor and camera on the app, if no fire is detected it will lead to delayed or no response which can leave the robotic vehicle in a vulnerable position to overheating or damage from an undetected fire.

3.2.6.4. Ineffective Fire Suppression

The motor failure or pump malfunctions will disable Ember Bot from reaching the fire or extinguishing it, reducing its fire-fighting capability.

4. Support Requirements

4.1.1 Mobile Device with Wi-Fi Access

To control the Ember Bot and use the system as designed, the user must have a mobile device with access to Wi-Fi. The mobile device can be either an Android or IOS system. The user will be able to view the application and control page in the app.

Rationale: Other forms of accessing the mobile app are outside the scope of this Project.

4.2.2 User Distance

The user must stay within a distance range that is less than 200ft(indoor) to 300ft(outdoor) away from the Ember Bot so that the Ember Bot can be connected to the Mobile Device through Wi-Fi.

Rationale: This will be the largest distance that the Ember Bot can operate under Wi-Fi.

4.2.3. Maintenance

The user must maintain and test the Ember Bot at least once every two months to ensure no parts are damaged and the system is working.

Rationale: The system is designed so that users will be able to address any issues using the help page and user manual.

Appendix A: Acronyms and Abbreviations

A	Amperes
BIT	Built-In Test
FOV	Field of View
ICD	Interface Control Document
LED	Light-Emitting Diode
MHz	Mega Hertz (1,000,000 Hz)
PCB	Printed Circuit Board
RF	Radio Frequency
W	Watt
USB	Universal Serial Bus
Lb	Pound
Ah	Amp-hour

Appendix B: Definition of Terms

Fire Detection	Sensors and cameras relay data to the communication subsystem. Information is then sent to the app interface for the user.
Vehicle Operations	Components serve different components for running robotic vehicles. Receiving controls from the communication subsystem.

EMBER BOT

Jonathan Chen, Kevin Rivera, Nancy Ramirez Castillo, Yuwen Zheng

INTERFACE CONTROL DOCUMENT

REVISION – 2
24 April 2025

INTERFACE CONTROL DOCUMENT
FOR
Ember Bot

PREPARED BY:

TEAM #30

APPROVED BY:

_____

Jonathan Chen 4/24/25

_____

John Lusher, P.E. 4/24/25

_____

Swarnabha Roy 4/24/25

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/20/2025	Jonathan Chen		Draft Release
2	4/24/2025	Jonathan Chen		Minor Edits for Final Report

Table of Contents

List of Tables

List of Figures

No table of figures entries was found.

1. Overview

The Interface Control Document (ICD) for Ember Bot will provide detailed explanations of how the multiple components/subsystems in the Concept of Operations and the Functional System Requirements will be executed. The ICD will include the physical descriptions of the various components and the technical requirements of Ember Bot. Additionally; this document will outline how each subsystem will integrate to meet the goals and requirements specified in the Concept of Operations and the Functional System Requirements documents.

2. References and Definitions

2.1. References

IRJET-V612417
Design & Implementation of RF-based Fire Fighting Robot
February 2019

IEEE Standard 802.15.4-2015
Standard for Low-Rate Wireless Personal Wireless Network
2015 Revision

2.2. Definitions

A	Amperes
BIT	Built-In Test
ICD	Interface Control Document
LED	Light-Emitting Diode
MHz	Megahertz (1,000,000 Hz)
PCB	Printed Circuit Board
RF	Radio Frequency
W	Watt
Ah	Amp-hour
lb	Pound
mA	Milliamp

3. Physical Interface

3.1. Weight

The Ember Bot will weigh less than 21lbs. This allows for an average person to be able to lift it and deploy it. The water tank will also be empty when first deployed, with it being filled only when the robot is used.

Table 1. Ember Bot Weight Specifications

Component	Weight
Vehicle shell	Est. 11.24 lbs
Water Pump	1.6 lbs
DC Motors	0.97 lbs
Cooling Fan	0.11 lbs
Temperature Sensor	0.0198 lbs
Water Tank with Water	Est. 4.17 lbs
Infrared Sensor	0.0198 lbs
Camera Module	0.044 lbs
Servo Motors	0.599 lbs
Batteries	0.78 lbs
Motor Driver	0.136 lbs
Microcontroller	0.044 lbs
Buck Converter	0.071 lbs
Total	19.82 lbs

3.2. Dimensions

3.2.1 Dimension of Vehicle Operations

Table 2. Vehicle Operations Dimension Specifications

Component	Length	Width	Height
LED Strip	Est. 6”	Est. 0.5”	N/A
Cooling Fan	2.35”	0.98”	2.38”
Water Pump	6.5”	4”	2.2”
Servo Motor	1.57”	0.79”	1.41"
Temperature Sensor	1.26”	0.55”	N/A
DC Motors	5.5”	1.5”	1.25”
Motor Driver	3.33”	2.44”	0.6”
Buck Converter	0.55”	0.83”	0.55”
Solar Panel	16”	13.1”	0.8”

3.2.2 Dimension of Fire Detection

Table 3. Fire Detection Dimension Specifications

Component	Length	Width	Height
Camera Module	1.06”	1.57”	N/A
Infrared Sensor	1.42”	0.63”	N/A
Microcontroller	1”	1.9”	N/A

3.2.3 Dimension of Vehicle Frame

The dimensions of the robot are an estimated length of 15”, a width of 13.5”, and a height of 9”. The dimensions could be changed to accommodate the components inside the robot and water holder.

3.3 Mounting Locations

3.3.1. Usage of Ember Bot

The robotic vehicle can be placed in a flat unobstructed area. Once placed on the ground, the water tank inside of the robot can be filled to a maximum of half a gallon of water, designed for small fires. The water tank is also optimally designed to reduce space taken as well as limit weight on the overall vehicle. If needed, external water tanks can be used, or Ember Bot can be scaled to deal with other fires. Once filled to the desired amount, it can be turned on and operated via the user.

3.3.2. Mounting of Sensors

The sensors on Ember Bot will be positioned to ensure optimal data collection. Infrared sensors will be placed at appropriate heights and angles to maximize detection accuracy, while the camera module will be mounted to provide a clear, unobstructed field of view. Mounting options will be selected based on stability and environmental conditions to ensure reliable sensor performance.

3.3.3. Mounting of Battery & Power Converter

The batteries will be mounted on each side wall of the chassis to optimize space for all components. The PCB and power converter will be placed in the middle of the chassis having equal distance between sensors and other equipment.

3.3.4. Mounting of Microcontroller

The microcontroller will be housed in a waterproof, dust-resistant enclosure within the chassis, ensuring protection from environmental factors such as moisture, dirt, and debris. All cabling will be routed through sealed cable glands or waterproof connectors to maintain enclosure integrity while allowing efficient data transmission and power distribution. This setup will enhance the durability, reliability, and longevity of the electronics subsystem in various operating conditions.

4. Thermal Interface

4.1. Cooling of Battery

The battery will be air-cooled using a fan that activates when the internal temperature exceeds a set threshold, as detected by a temperature sensor. The heat generated by the battery will be dissipated through a vent. Additional thermal protection can be provided given the material of the robotic vehicle as well as internal heat sinks.

5. Electrical Interface

5.1. Primary Input Power

Power for Ember Bot will be supplied via an 11.1-volt battery alongside additional support 2V batteries, which will be used to supply power to all appropriate subsystems when needed. The voltage will be regulated, with buck converters used to step down 12V to appropriate levels for components that require lower operating voltages.

5.2. Signal Interfaces

The KY-026 flame sensors will communicate with the computing subsystem by outputting an analog voltage signal once within the specified wavelength (760nm – 1100nm), which will interpret flame intensity and temperature readings.

5.3. Voltage and Current Levels

The full voltage available will be used to power both 24V motors before being stepped down to 12V to power the water pump and then to the lowest voltage of 3.3V which will be used for sensors, microcontroller, LED light, ESP32, and camera module. The current will be no more than 2.2A which will be used mostly by the motors and then decreased to milliamps for all other components, reaching the lowest to 1mA.

5.4. User Control Interface

The only user interface for Ember Bot's application will be an application-based GUI, allowing users to monitor the sensor data collected and control Ember Bot in real time. Additional usage of the pump and other additional settings are configured into the application. No additional user interface is required for this system.

6. Communications / Device Interface Protocols

6.1. Wireless Communications (Wi-Fi) & Bluetooth

All communication between Ember Bot and communication with the controller will be managed through Wi-Fi. This implementation adheres to the IEEE 802.15.4-2015 standard for wireless connectivity.

6.2. Microcontroller Input & Output

Ember Bot's ESP32 microcontroller features a range of digital and analog I/O pins, capable of reading and transmitting signals within a 0V-3.3V range. This voltage level is compatible with all the sensor communication protocols used in the system, ensuring reliable data and control.

6.3. Video Interface

The video interface is managed by an ESP32-CAM connected to the ESP32 microcontroller via GPIO pins. The camera streams video to the web application over Wi-Fi, enabling remote access and control. This meets the requirements of all the protocols mentioned above.

EMBER BOT

Jonathan Chen, Kevin Rivera, Nancy Ramirez Castillo, Yuwen Zheng

MILESTONE & VALIDATION PLAN

REVISION – 2
24 April 2025

MILESTONE & VALIDATION PLAN
FOR
Ember Bot

PREPARED BY:

TEAM #30

APPROVED BY:

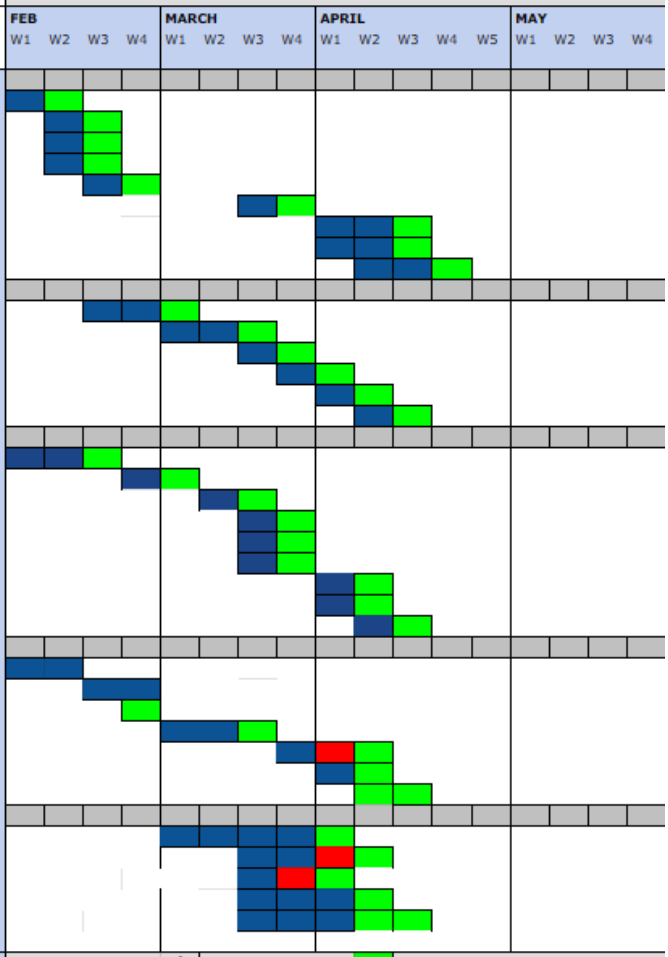
_____
Jonathan Chen 4/24/25

_____
John Lusher, P.E. 4/24/25

_____
Swarnabha Roy 4/24/25

Ember Bot Project Schedule

Deliverable/Task	Owner	Duration	Note
Overall Deliverables			
ConOps	All	2w	Due 2/5
FSR + ICD	All	2w	Due 2/20
Validation Plan	All	2w	Due 2/20
Milestone Plane	All	2w	Due 2/20
Midterm Presentation	All	2w	Due 2/24
Status Update Presentation	All	2w	Due 4/2
Final Presentation	All	3w	Due 4/16
Final Demo	All	3w	Due 4/21
Final Report	All	3w	Due 4/27
Microcontroller & Fire Detection			
ESP32 Installation + Start-Up	Jonathan	3w	
ESP32 Framework	Jonathan	3w	
Camera Module	Jonathan	2w	
IR Sensor Detector	Jonathan	2w	
Light Source	Jonathan	2w	
Microcontroller Connection w/ Systems	Jonathan	2w	
App Development			
Create wireframe and overall structure	Yuwen	3w	
Create clickable prototype	Yuwen	2w	
Create User Interface Design	Yuwen	2w	
Test the app with basic functionality	Yuwen	2w	
Create User Mannual within app	Yuwen	2w	
Create User Help page within app	Yuwen	2w	
Add Wifi Functionality	Yuwen	2w	
Add video streaming	Yuwen	2w	
Finalize App Design	Yuwen	2w	
Power System			
Design circuit schematic	Nancy	2w	
Breadboard Assemble and Testing	Nancy	2w	
Power Efficiency and Load Testing	Nancy	1w	
PCB Design	Nancy	2w	
PCB Testing	Nancy	3w	
Thermal and Safety Analysis	Nancy	2w	
Validate Recharging	Nancy	2w	
Pump System + Car			
Verify Vehicle Movement	Kevin	2w	
Set up Lighting and Cooling	Kevin	2w	
Servo System	Kevin	2w	
Water Pump System	Kevin	3w	
Creating Vehicle Frame	Kevin	2w	



Annotations:

Critical	1	Completed	
External dependency	2	Expected Completion	
Postponed	3	In Progress	
		Behind Schedule	

Validation Plan for Ember Bot

Test Name	Success Criteria	Methodology	Status	Engineer Responsible
Fire Detection	Flame Sensor delivers correct output resulting from Flame/IR	Measure Sensor Output with IR Source at Varying Distances	TESTED	Jonathan Chen
Camera Streaming	Steady Visual Output from Camera Stream	Set up ESP32-CAM to transmit live video. Evaluate frame rate, transmission stability, and stream quality.	TESTED	Jonathan Chen
ESP32 Wi-Fi Access Point	The system connects with the Microcontroller Remotely	ESP32-CAM and Cellular Devices can connect to ESP32 via Wi-Fi. Measure functionality at a distance.	TESTED	Jonathan Chen
LED Output	LED functions as directed	Light Source tested at OFF/ON State	TESTED	Jonathan Chen
Communication Range	All subsystem components can be remotely accessed across the room	System Communication Test is tested across the FEDC room, performed at 5 Feet Intervals till no response.	TESTED	Jonathan Chen
Vehicle Movement	Both motors can be controlled independently via the motor driver	Input control signals can control each motor independently to go forward, backward, turn left, turn right, and turn 180 degrees.	TESTED	Kevin Rivera
Electronics Isolation	There is no risk of water damaging the PCB, motors, and other electronics in the bottom layer of the robot.	Seal and connect the two pieces of the bottom frame and pour 1 gallon of water over it for a short time, and once done check below the frame to see if water leaked.	TESTED	Kevin Rivera
Vehicle Chassis	Vehicle frames can hold all electronics and water tank safely	Place the bottom level of the frame with the water tank attached on top to ensure even weight distribution while moving.	TESTED	Kevin Rivera
Servo Movements	Servos can successfully move and point to a certain point	Attach a laser on the end of the servo configuration to point to an x-y grid. Input x and y coordinates and verify that the laser is pointing to the correct point.	TESTED	Kevin Rivera
Maximum Water Range	Check the maximum range of the water pump system	Turn on the water pump with the nozzle on and measure the distance in which the flow of water goes before beginning to drop.	TESTED	Kevin Rivera

Water Nozzle Movement	The water pump system successfully attached to the servo system	Connect the servos with the water nozzle and verify that the range fits with the coordinate system verified in the servos movement test	TESTED	Kevin Rivera
Cooling Fan Implementation	The cooling fan can turn on and cool the system	Attach the fan to the fan vents and turn it on while running the motors. Measure the temperature difference when the fan is on and off.	TESTED	Kevin Rivera
App Launch on emulator	The app can run and be tested on emulator	The app can obtain all the functions designed on emulator.	TESTED	Yuwen Zheng
App Launch on the mobile device	The app can be installed on mobile devices and can be opened	The app can function on both emulator and a physical mobile device.	TESTED	Yuwen Zheng
Left/Right wheel button	Both buttons can be dragged vertically, showing the axis position on the screen.	Drag each button on-screen and drag both buttons at the same time.	TESTED	Yuwen Zheng
Nozzle Button	The button can be dragged both vertically & Horizontally	Drag the nozzle button on the screen.	TESTED	Yuwen Zheng
User Manual Page Button	The button can direct the user to a new page with a user manual	Click on the button and check if the user is directed to a new page with the user manual.	TESTED	Yuwen Zheng
Help page Button	The button can direct the user to a new page with a help guide on Ember Bot	click on the button and check if the user is directed to a new page with a help page.	TESTED	Yuwen Zheng
Wifi Functionality	The app can establish communication with ESP32 through a Wi-Fi connection	Write a test application running on a laptop that listens for incoming packets, and lets the app send control signals to the laptop's IP.	TESTED	Yuwen Zheng
Video Streaming	The app can receive and display video signals in the mobile app	Test if the video can be received and displayed on the screen of the mobile device.	TESTED	Yuwen Zheng
1.1 Buck (24V-12V) Simulation	Steady-state waveform and outputting a max of 12.6V	Using LTSpice/Altium to select components and verify power rail. Using a higher voltage to compensate for real-world applications will decrease the output voltage.	TESTED	Nancy Ramirez Castillo
1.2 Buck (24V-12V)	Keeps operating under high temperatures and running time	Use the DC Power Supply to provide 24V and measure the output voltage using the Oscilloscope ensuring it runs smoothly for 10 minutes without overheating.	TESTED	Nancy Ramirez Castillo
1.3 Buck (24V-12V)	Smooth step-down voltage from boost converter	Test the flow of power, voltage, and current with the boost and buck converter connected. Outputs at both converters should be minimal compared to when running alone.	TESTED	Nancy Ramirez Castillo

2.1 Buck (12V-5V) Simulation	Steady-state waveform and outputting a max of 5.5V	Using LTSpice/Altium to select components and verify power rail. Using a higher voltage to compensate for real-world applications will decrease the output voltage.	TESTED	Nancy Ramirez Castillo
2.2 Buck (12V-5V)	Low voltage output and current of min. 1.5A	Use the DC Power Supply to provide 12V and measure the output with an Oscilloscope ensuring minimal ripple and runs for 10 minutes without overheating.	TESTED	Nancy Ramirez Castillo
2.3 Buck (12V-5V)	Smooth step-down voltage from the first buck converter	Test the flow of power, voltage, and current with both buck converters connected. Outputs at both converters should be minimal compared to when running alone.	TESTED	Nancy Ramirez Castillo
3.1 Boost (22.2V-24V)	Outputs 23.5V-24.5V and 2.2A current with minimal voltage ripple	Use the DC Power Supply to provide the 22.2V that will be provided by the Li-Po batteries and measure the output voltage using the Oscilloscope ensuring it runs smoothly for 5 mins.	TESTED	Nancy Ramirez Castillo
Full System Load	No overheating under max. load conditions	Power the entire system and let it run under full operation for 15 minutes. Monitor every converter's output with a load connected to simulate next semester's integration.	TESTED	Nancy Ramirez Castillo

Ember Bot

Jonathan Chen, Kevin Rivera, Nancy Ramirez Castillo, Yuwen Zheng

SUBSYSTEM REPORTS

REVISION – 1
24 April 2025

SUBSYSTEMS REPORT
FOR
EMBER BOT

PREPARED BY:

TEAM #30

APPROVED BY:

_____

Jonathan Chen 4/27/2025

_____

Prof. S. Kalafatis 4/27/2025

_____

Swarnabha Roy 4/27/2025

Change Record

Rev.	Date	Originator	Approvals	Description
1	04/24/2025	Jonathan Chen		Original Release

Table of Contents

List of Tables

List of Figures

1. Introduction

Ember Bot is a remotely operated vehicle designed to autonomously detect and extinguish small fires in various environments. The system integrates environmental sensing, wireless communication, real-time video streaming, and targeted water delivery to support firefighting efforts. Ember Bot is divided into the microcontroller subsystem, vehicle operations and water pump subsystem, power subsystem, and mobile app subsystem. Ember Bot is built to offer an effective, responsive solution for rapid fire detection and suppression in both indoor and outdoor settings. Since each subsystem was validated to meet its performance requirements, we look to start integrating these components into a fully operational system as outlined in the ConOps, FSR, and ICD.

Ember Bot

Jonathan Chen

SUBSYSTEM REPORT

REVISION – 1
24 April 2025

2. Microcontroller Subsystem Report

2.1. Subsystem Introduction

The purpose of Ember Bot is to detect nearby fires and extinguish fires, Ember Bot must be able to detect flames as well as communicate with the rest of the bot and the user, which the microcontroller subsystem commands. For this reason, it is crucial to interface the sensors to the microcontroller, which then transfers the data to the appropriate channels.

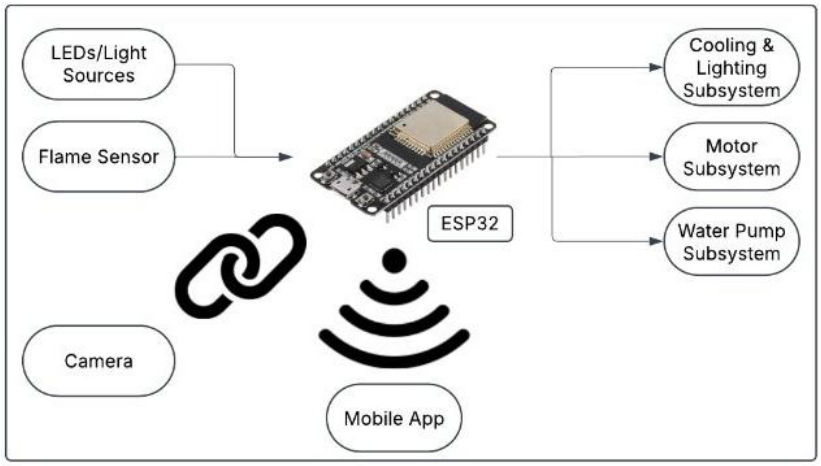


Figure 1. Microcontroller Subsystem Organization

2.2. Fire Detection Sensor

2.2.1. Operation

For the Fire Detection Sensor, a KY-026 Flame Sensor Module was used, which detects infrared light emitted by fire. The sensor uses a 5mm infrared receiver LED to detect the wavelengths emitted by fire around, as well as a potentiometer to adjust the sensitivity of the sensor.

2.2.2. Validation

The Fire Sensor was validated by testing both the analog and digital outputs of the sensor, with varying IR sources and distances. The fire sensor was validated with both a flashlight and a small fire source, due to constraints of the lab.

```
-----------------------------------------------------------------
Analog Voltage Value: 14.2229 V;         Digital Threshold Value: Flame Not Detected
-----------------------------------------------------------------
Analog Voltage Value: 14.2473 V;         Digital Threshold Value: Flame Not Detected
-----------------------------------------------------------------
Analog Voltage Value: 6.6276 V;          Digital Threshold Value: Flame Detected
-----------------------------------------------------------------
Analog Voltage Value: 5.6403 V;          Digital Threshold Value: Flame Detected
-----------------------------------------------------------------
```

Figure 2. Validation of Fire Detection Sensor

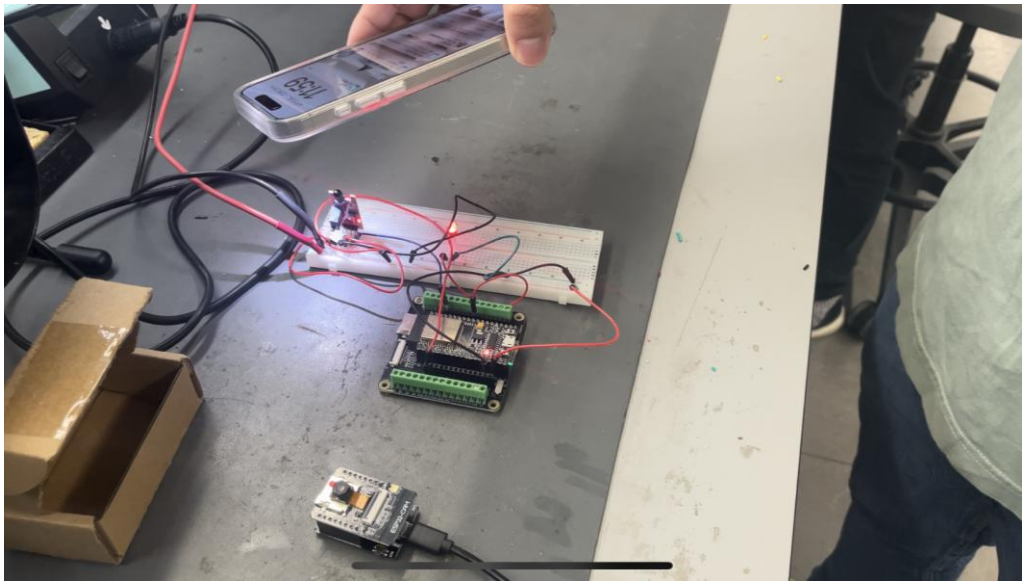


Figure 3. Fire Detection Set-Up

The recorded performance of the KY-026 shows a change between if there is an IR source and if there is not. The distance of the sensor can be changed via the potentiometer.

2.3. Camera

2.3.1. Operation

An ESP32-CAM module is used to continuously stream a live video feed of Ember Bot. The camera provides real-time monitoring through Wi-Fi, enabling users to remotely view the robot's environment. It is configured using the ESP-IDF framework and streams MJPEG frames over a local network.

2.3.2. Validation

The ESP32-CAM was validated by testing its live-streaming capability at various distances and network environments. The camera module was moved at various speeds and conditions to assess image clarity and stability. It was tested to confirm consistent video output.

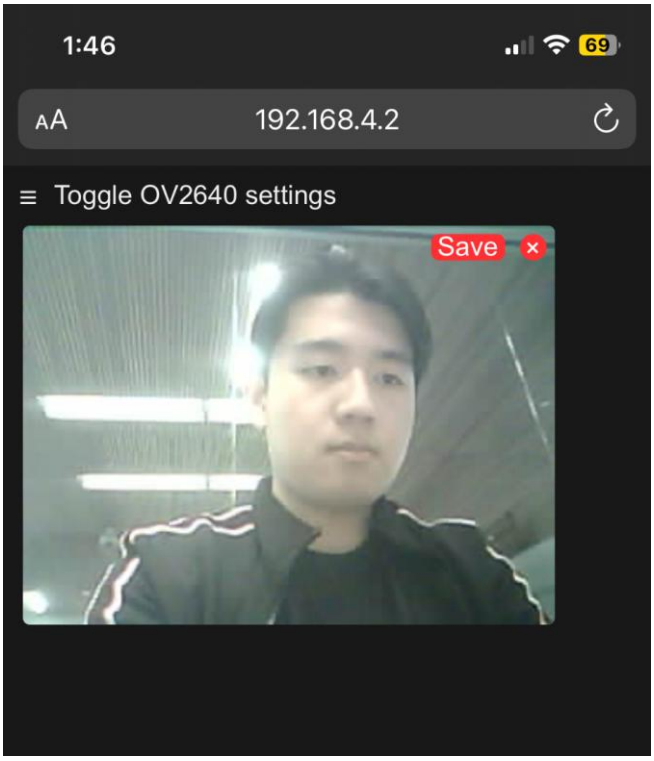


Figure 4. Camera Stream Interface

```
Camera Initialized Successfully!
WiFi connecting....
WiFi connected
Camera Ready! Use 'http://192.168.4.2' to connect
```

Figure 5. Camera Initialization Configuration

The camera can stream the recorded performance of the ESP32-CAM shows stable video transmission and has been tested at varying distances, all resulting in a steady FPS.

2.4. Wi-Fi Access Point

2.4.1. Operation

An ESP32 module is configured as a Wi-Fi Access Point to enable direct wireless communication with Ember Bot. Devices can connect to the ESP32's local network without needing external Wi-Fi infrastructure. This setup provides a reliable connection for video streaming, control signals, and sensor data transfer.

2.4.2. Validation

The ESP32 Access Point was validated by testing device connectivity, signal strength, and data transmission stability at various distances. Connection tests were performed using a smartphone and a laptop. The range and responsiveness of the AP were evaluated in both open spaces and obstructed indoor environments.

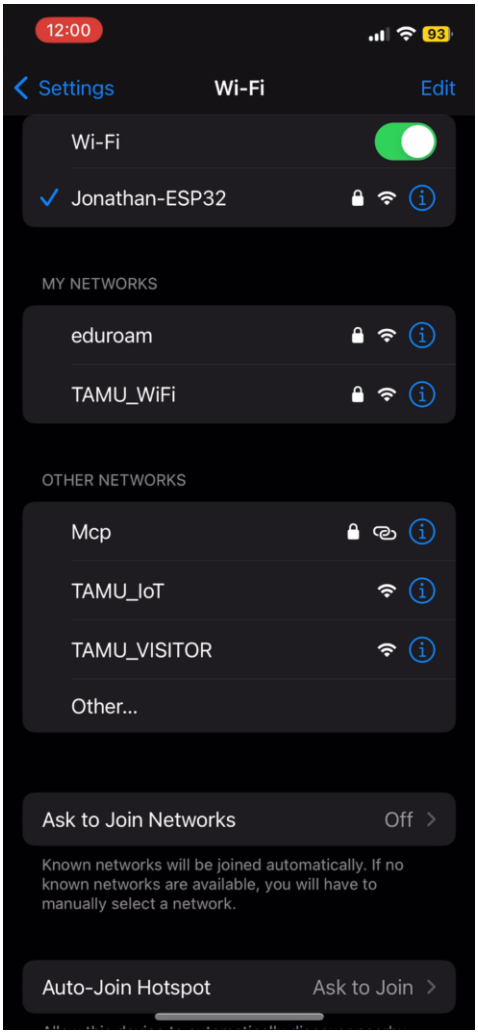


Figure 6. ESP32 Wi-Fi Source

```
Setting AP (Access Point)…AP IP address: 192.168.4.1
KY-026 Flame Detection:
Analog Voltage Value: 14.3939 V;         Digital Threshold Value: Flame Not Detected
-------------------------------------------------------------
Analog Voltage Value: 14.3597 V;         Digital Threshold Value: Flame Not Detected
-------------------------------------------------------------
Analog Voltage Value: 14.2278 V;         Digital Threshold Value: Flame Not Detected
```

Figure 7. Access Point IP Configuration

The recorded performance of the ESP32 Access Point shows stable wireless connections within an approximate range of 20–30 meters indoors and outdoors, with minimal connection drops or latency during operation. However, when there are obstructions, the signal tends to drop in connectivity.

2.5. Subsystem Conclusion

The microcontroller subsystem allows Ember Bot to sense its environment, communicate with the user, and respond effectively to fire hazards. The ESP32 operates as a Wi-Fi Access Point, creating a direct and reliable wireless connection to the user's device without the need for external infrastructure. The ESP32-CAM streams a live video feed, providing continuous visual monitoring for remote operation. The integrated fire sensor detects potential fire sources by sensing infrared radiation, supporting both autonomous and user-controlled firefighting actions. Together, these components enable quick, accurate responses while maintaining stable communication and environmental awareness in a variety of operating conditions.

Ember Bot

Kevin Rivera

SUBSYSTEM REPORT

REVISION – 1
24 April 2025

3. Vehicle Operations & Water Pump Subsystem Report

3.1. Subsystem Introduction

Ember Bot must be able to operate in areas potentially isolated from the user, such as inside collapsed buildings. To do this, it must have its own water tank and motors that are to be controlled by user inputs via the mobile application. To allow for complete directional control of Ember Bot, each motor can be controlled bidirectionally and independently from one another. The user will also have the capability of controlling where they would like to shoot out the water through the app. With the water tank being placed on top of the robot, all electronics must be isolated and protected from possible tank leakage. An initial cover was designed with a fan attached to it for additional system cooling.

3.2. Vehicle Operations

3.2.1. Operation

To control the two 24V DC motors, a Cytron Dual Channel motor drive is connected to them. The motor driver can receive PWM signals to control the speed and direction signals to dictate the direction of each motor independently. The method of speed control via the PWM signals depended on the duty cycle of said PWM signals. A 100% duty cycle would run the motors at full speed, a 50% duty cycle could run the motors at half speed, a 25% duty cycle would run the motors at quarter speed, etc. If the direction ports on the motor control are not receiving any signal, both motors would run in the forward direction. If there is a signal higher than 3V, then the motors would run in reverse direction. These signals will be sent out by our central ESP32.

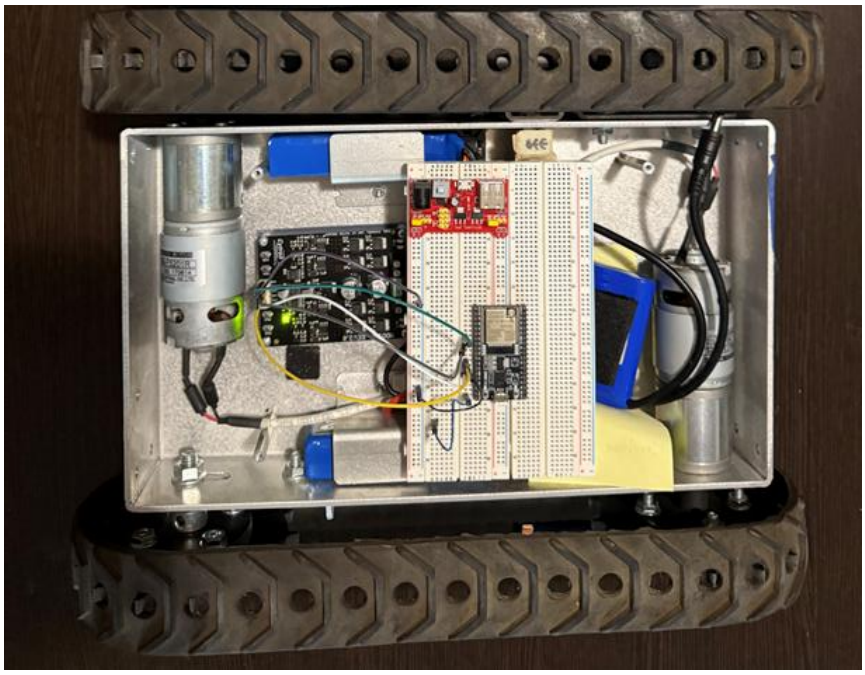


Figure 8. Inside Ember Bot

The frame was designed and 3D printed with PLA as two parts. The back piece had fan holes and vents designed into it to allow for proper air circulation to cool the motor driver and PCBs that will be placed inside. A 12V fan was installed on the frame. Hot glue and water-resistant tape were used to connect both pieces.

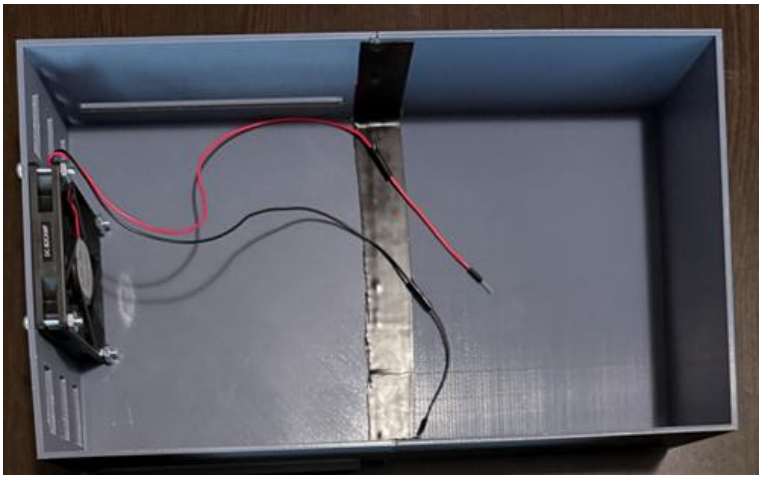


Figure 9. Inside the Frame

Figure 10. Frame Place on Ember Bot

In the case of entering a dark environment in search and rescue scenarios, an LED strip will be attached to the front of Ember Bot. This is to provide additional lighting for the camera, which will be streaming to the mobile application.

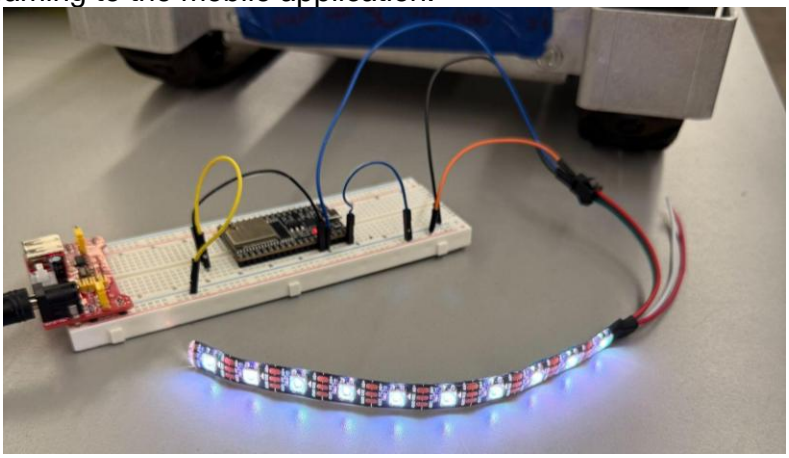


Figure 11. LED Strip

3.2.2. Validation

An ESP32 was programmed to loop a code that would have the motors run at full speed for 3 seconds, half speed for 2 seconds, quarter speed for 1 second, and zero speed

for 4 seconds. This is a total of 10 seconds of operation. The PWM signals can be seen in Figure 8.

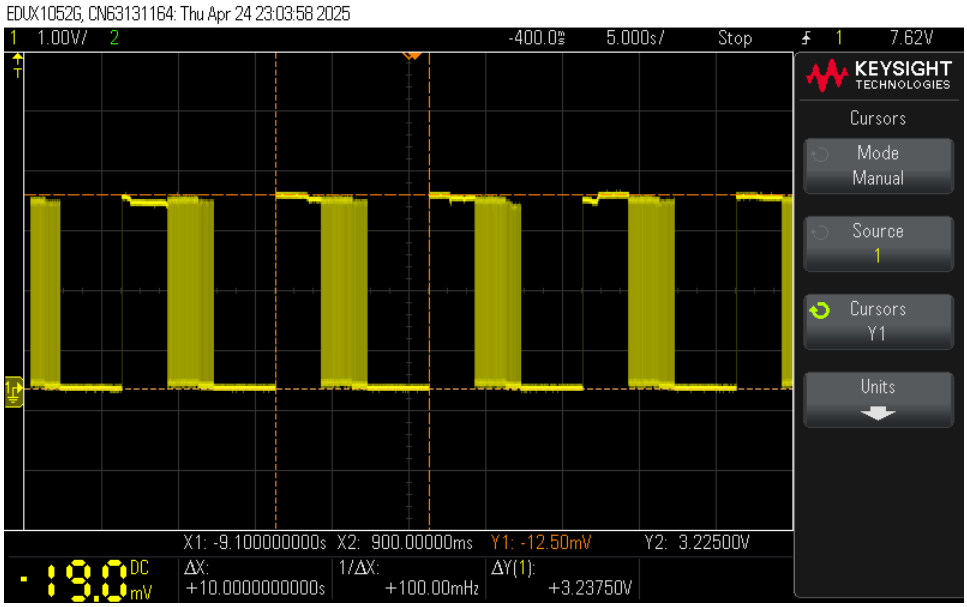


Figure 12. PWM Signals for Motor Speed Control

In a single loop of this code, each PWM and its duty cycle can be seen. The section in the cursors in Figure 9 is the PWM signal for full speed. This can be seen as there are no dips in the duty cycle.

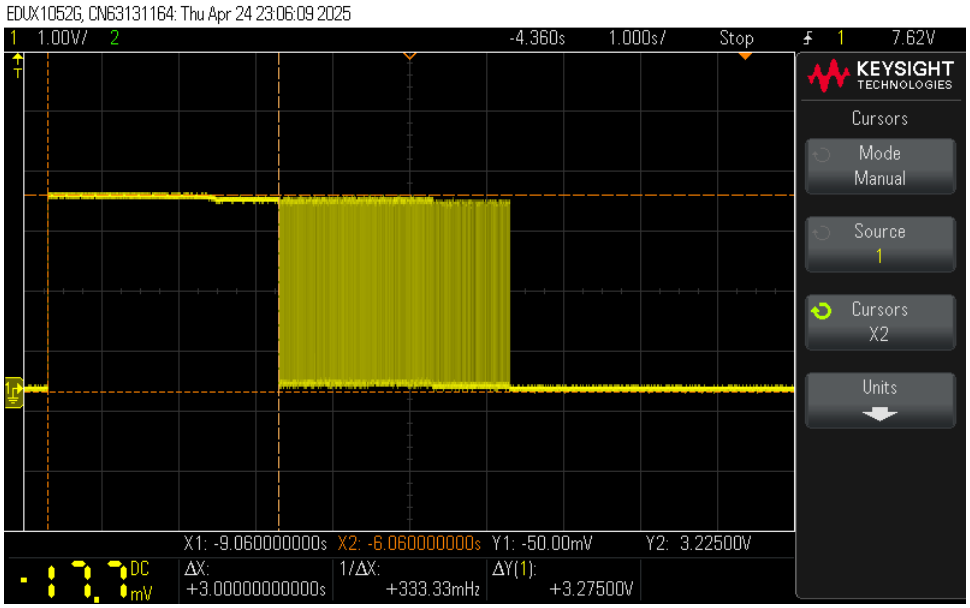


Figure 13. PWM Signal from One Code Cycle

In Figure 10, both the half-speed and quarter PWM can be observed. The half-speed PWM duty cycle has less spacings than the quarter-speed PWM duty cycle.

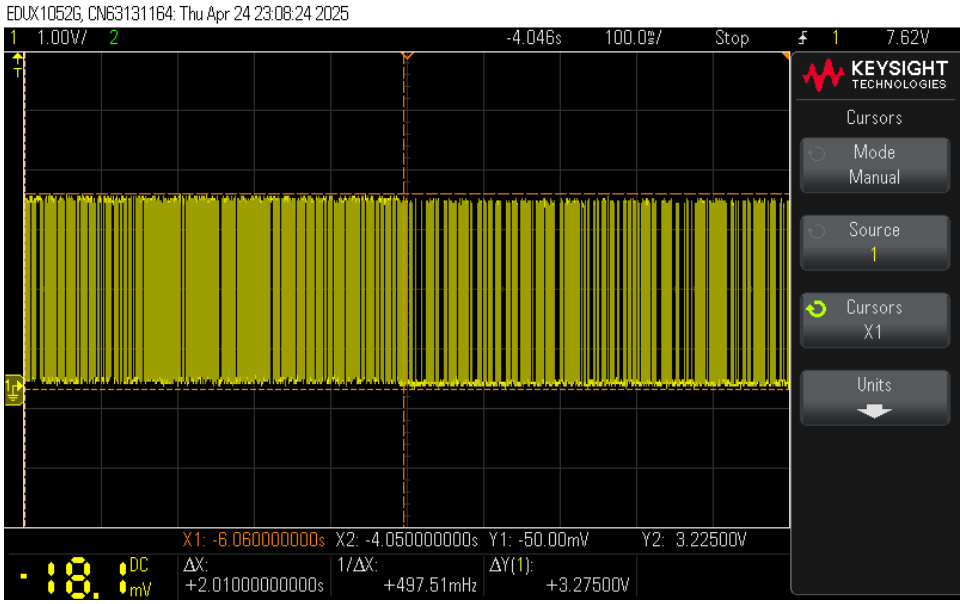


Figure 14. 50% And 25% Speed PWM Signals

The program also looped both motors to go forward for 3 seconds, backward for 2 seconds, and the right motor to go forward while the left motor went backward for 5 seconds. The right motor direction signal is the green waveforms, and the left motor direction signal is the yellow waveforms.

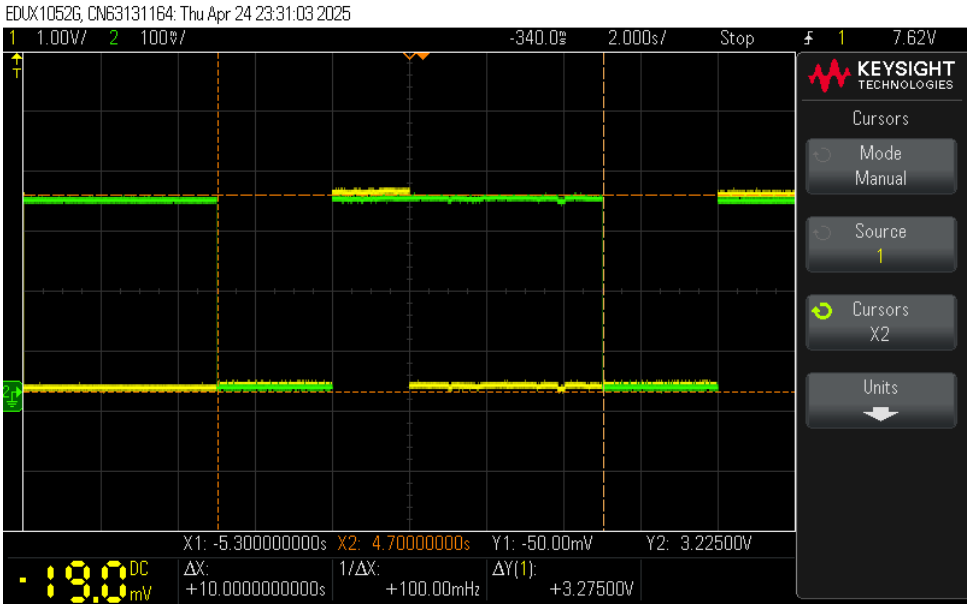


Figure 15. Motor Direction Signals

To observe the effectiveness of the cooling fan, three experiments were conducted. Each experiment had the motors run forward at half speed for 2 seconds, half speed forward for 2 seconds, half speed backward for 2 seconds, and full speed backward for 2 seconds. This was looped for 5 minutes with the temperature of the motor drive recorded every 30 seconds. After the 5-minute loop, the batteries were turned off and again the temperature was recorded

every 30 seconds for 6 minutes. This was done for three cases, one with the frame completely off, one with the frame attached and without the fan on, and a last one with the frame attached and the fan on 5V.

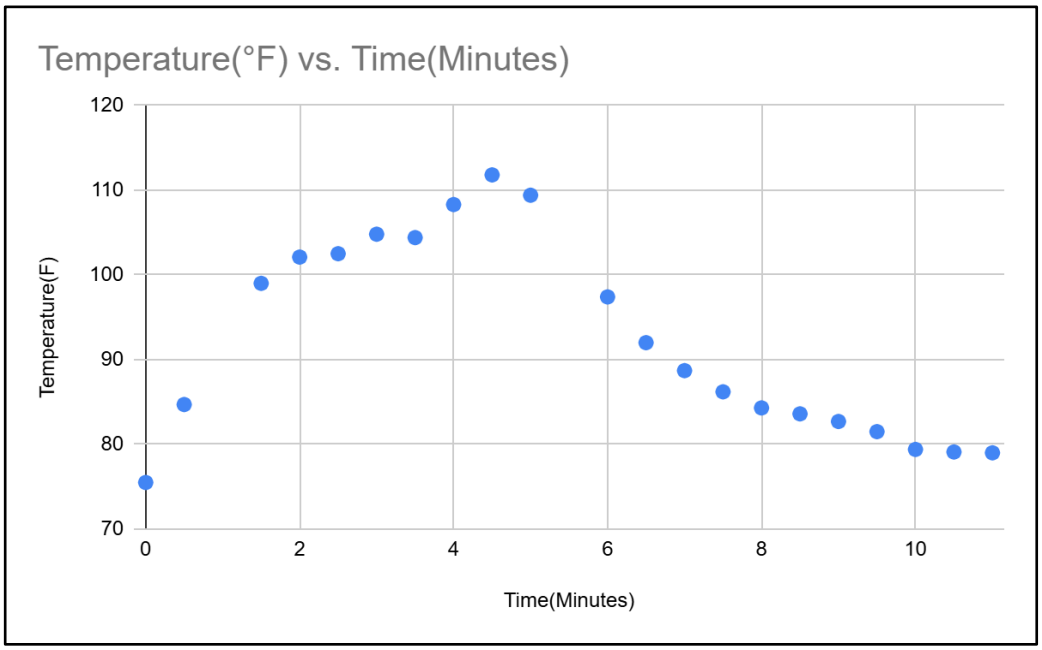


Figure 16. Temperature Reading for Motor Drive Without the Frame

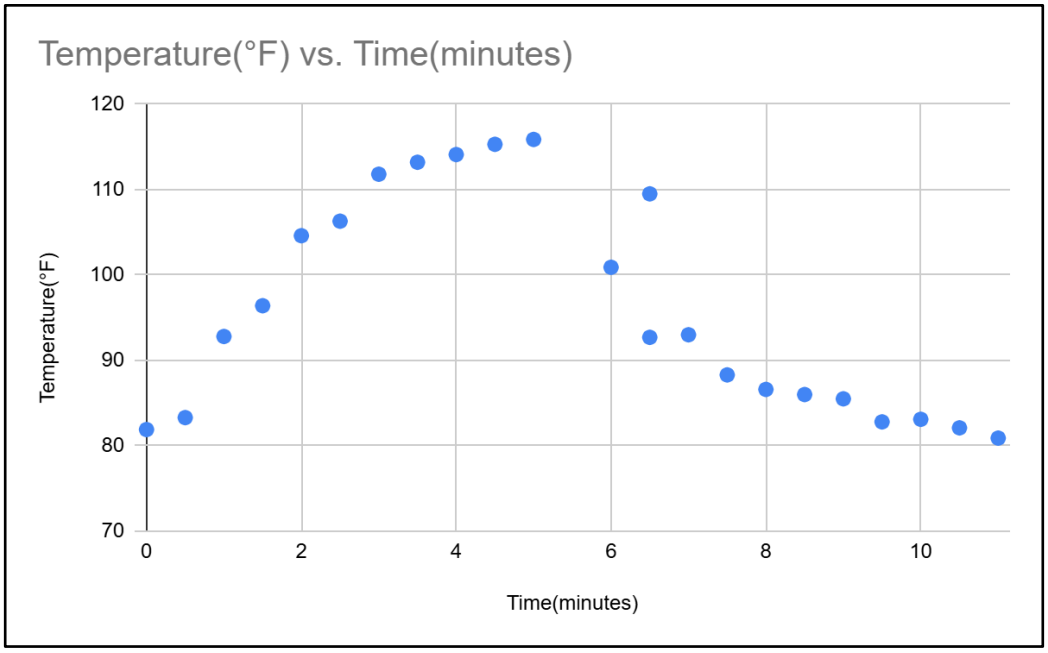


Figure 17. Temperature Reading for Motor Drive With the Frame On

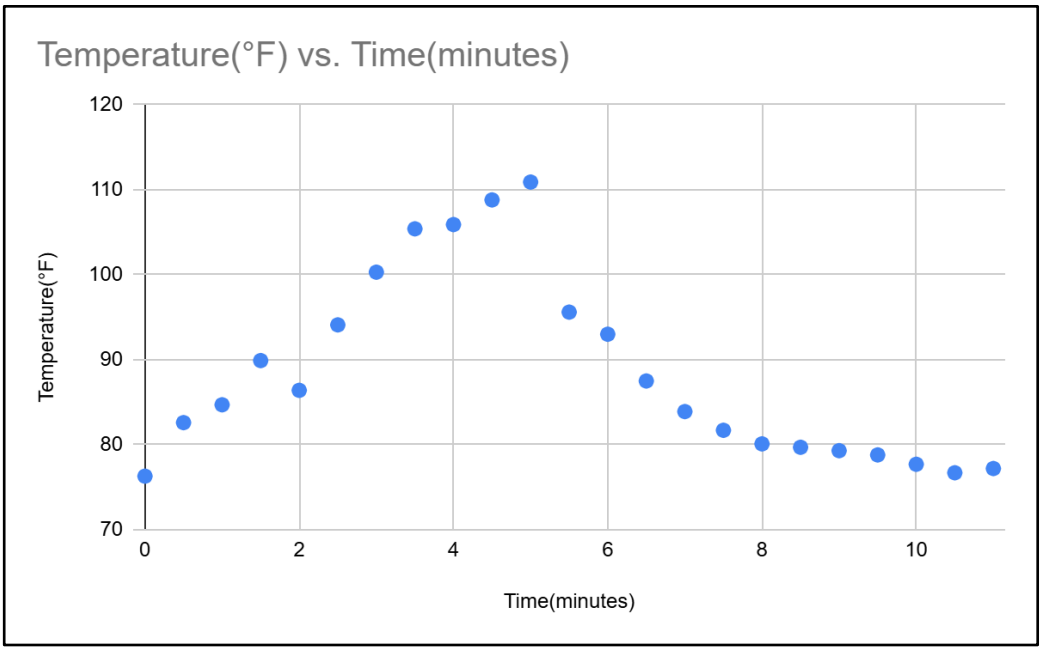


Figure 18. Temperature Reading for Motor Drive With the Frame and Fan On 5V

Table 1. Average Temperatures for All Test Cases

Test Case	Average Temperature (℉)
No frame	92.21
Frame attached, fan off	95.96
Frame attached, fan on 5V	88.54

3.3. Water Pump Subsystem

3.3.1. Operation

To have the water nozzle be able to point to a point the user enters through the mobile application, two HK15138 servos were configured as seen in Figure 15. The bottom servo moves the structure in the x-direction and the servo on its side controls the nozzle slot in the y-direction. With both of these being able to be controlled via degrees of rotation, the nozzle holder can point the end of the nozzle to a desired x and y coordinate on a plane.

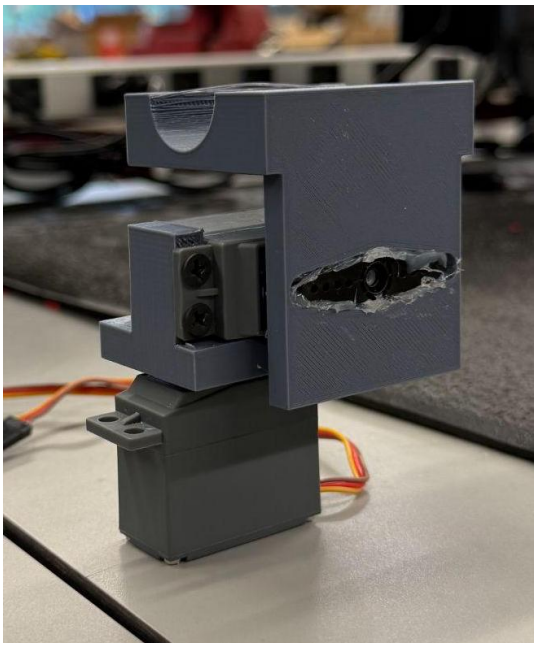


Figure 19. Servo Motor Arrangement for the Water Nozzle Movement

Servo motors receive PWM signals and interpret them as an angle to move to. By understanding the servo reference angle used to point it directly straight, the servo can be pointed in both the clockwise and counterclockwise direction. At a recommended distance from the hose to the target being 5ft, trigonometry can be used to determine the angle at which the angle of rotation must be increased or decreased to have steps of 1.5 inches.

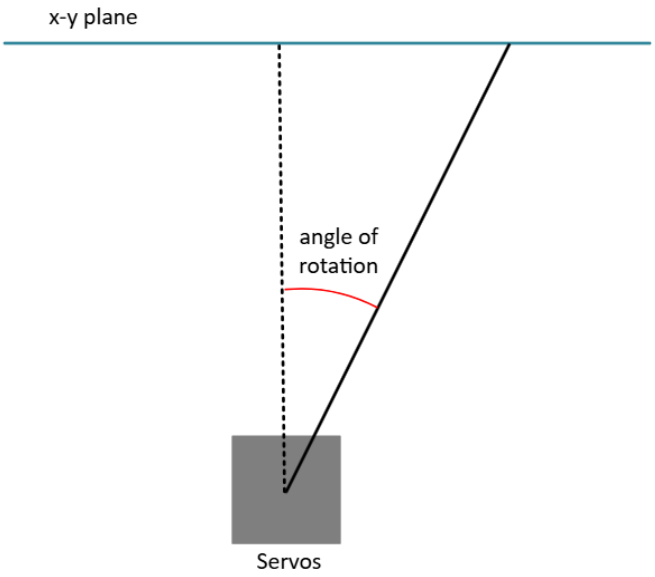


Figure 20. Method Of Using the Servos to Point to a Desired Point

A Bayite 12V DC water transfer pump is connected to a water tank that will pressurize the water to shoot out a powerful stream. The end of the output hose has a high-pressure adjustable spray nozzle to direct the water accurately.

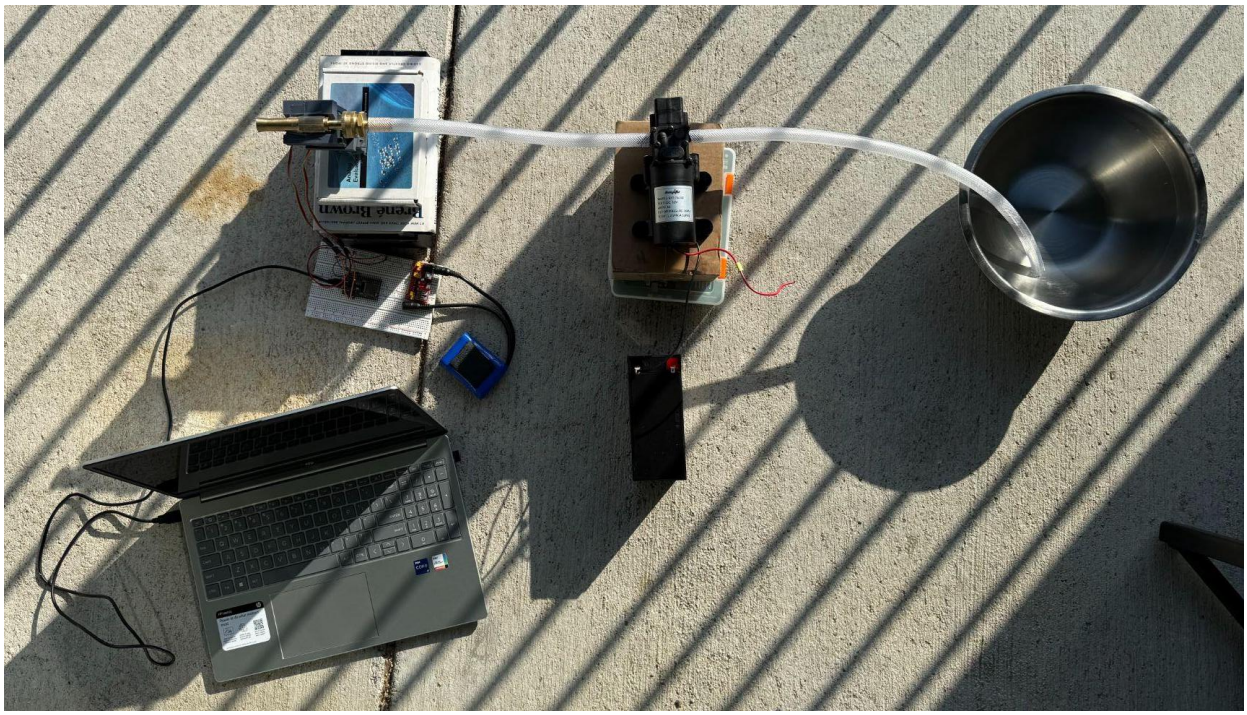


Figure 21. Water Pump System for Testing

3.3.2. Validation

Both servos were connected to an ESP32 with a program capable of translating x and y coordinates to servo positions. A laser pointer was attached to the nozzle holder to showcase the accuracy of the system. Coordinates ranging from -40 to 40 on both the x and y axes were tested. At a distance of 5 ft from the nozzle holder to the plane, this gave a full range of 10 ft in both directions. When coordinates were inputted, the laser had an accuracy of ∓0.6 inches. The servo configuration was also capable of holding the water nozzle and moving it properly as seen in Figure 19.

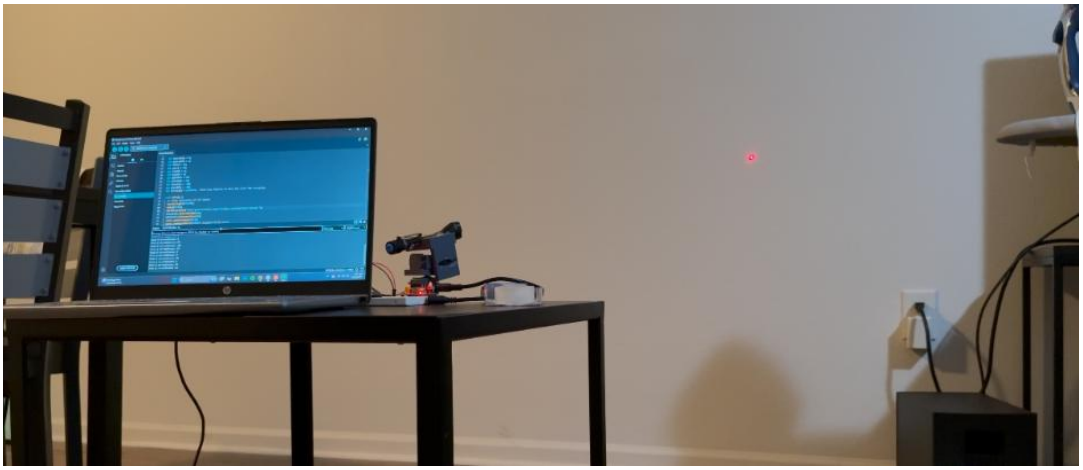


Figure 22. Laser Pointer Attached to Nozzle Holder for Testing

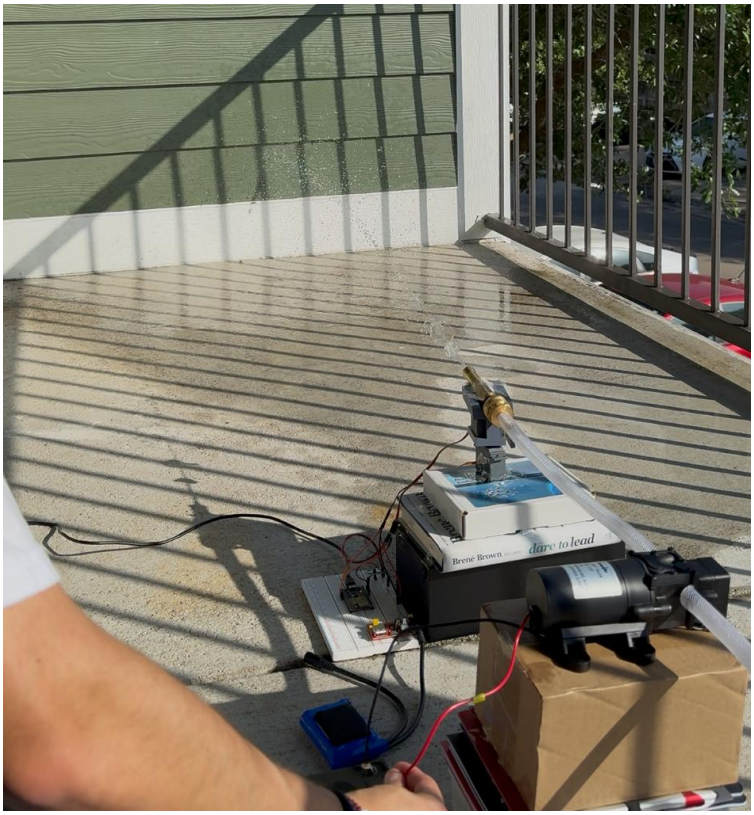


Figure 23. Water Pump System Being Used

3.4. Subsystem Conclusion

The vehicle operations subsystem ensures that Ember Bot can respond accurately to the user's desired inputs while maintaining all electronics in the operating range. The water pump system is vital to the effectiveness of Ember Bot’s ability to extinguish flames. The added feature of a movable nozzle provides the user with additional flexibility in this endeavor.

Ember Bot

Nancy Ramirez Castillo

SUBSYSTEM REPORT

REVISION – 1
24 April 2025

4. Power Subsystem Report

4.1. Subsystem Introduction

The Ember Bot's power subsystem delivers reliable and efficient power to all other subsystems and components, enabling the robot to operate safely and effectively during fire-fighting missions. The system utilizes two 11.1V/2200mAh 3S Li-Po batteries connected in series to provide a nominal 22.2V power source, which is then regulated through a series of DC-DC converters. These converters generate dedicated voltage rails (24V, 12V, and 5V) to meet unique power requirements for components in every subsystem. Additionally, an AC-DC battery charger is implemented to support safe charging and achieve battery longevity.

4.2. Subsystem Details

4.2.1. Li-Po Batteries

The system is equipped with two 11.1V/2200mAh 3S Li-Po batteries connected in series to supply approximately 22.2V.

Figure 24. Turnigy 11.1V Battery

4.2.2. 24V Boost Converter

A 24V boost converter is used to stabilize and increase the 22.2V voltage to a constant power to the two 24V DC motors via a dual-channel H-bridge motor driver. This boost converter was designed with a boost controller, LM5155DSSR, which supports a wide input range (3.5V to 45V), thermal shutdown protection at 347°F, and a maximum junction operating temperature of 257°F. It includes a fast-switching Schottky Diode with low forward voltage drop, an N-channel MOSFET for efficient switching, and high-voltage, high-temperature rated resistors and capacitors to ensure durability under operating conditions.

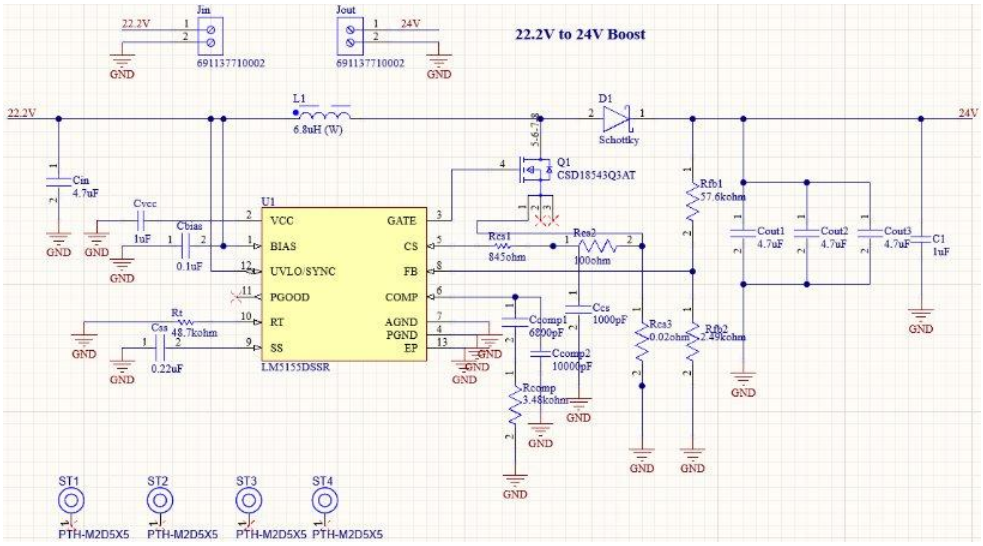


Figure 25. 24V Boost Converter Schematic

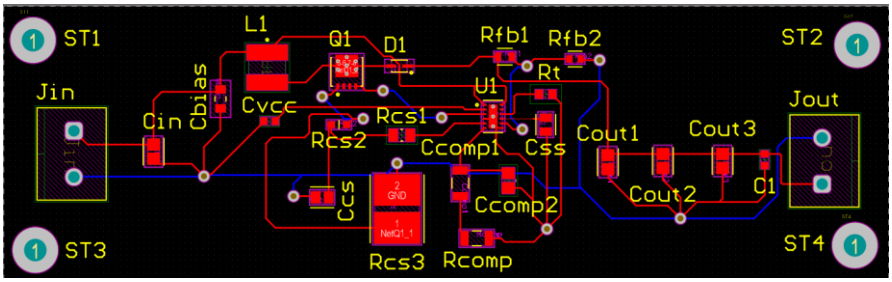


Figure 26. 24V Boost Converter PCB Traces

Figure 27. 24V Boost Converter Assembled PCB

4.2.3. 12V Buck Converter

The 12V buck converter supplies mid-level voltage to the water pump and DC cooling fan. This converter uses the TPS54202DDCR IC to step down the input voltage while maintaining steady output regulation. The design includes an inductor to store and transfer energy between cycles, along with high-temperature rated capacitors and resistors to ensure reliable operation.

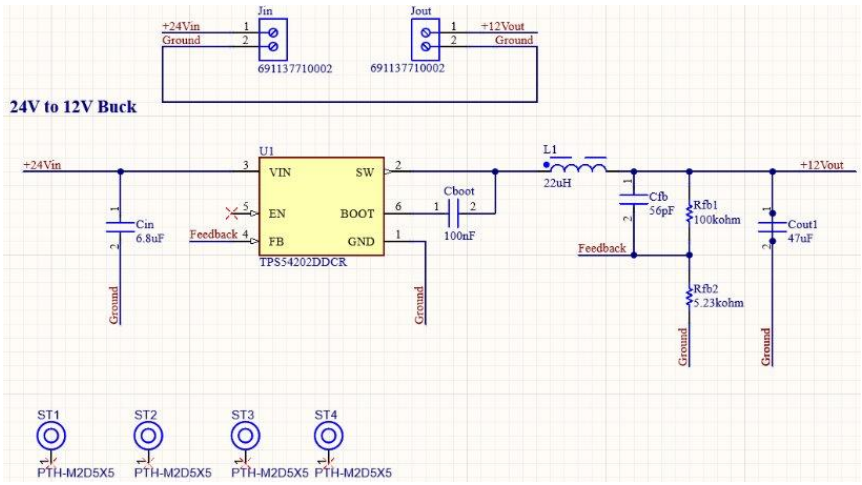


Figure 28. 12V Buck Converter Schematic

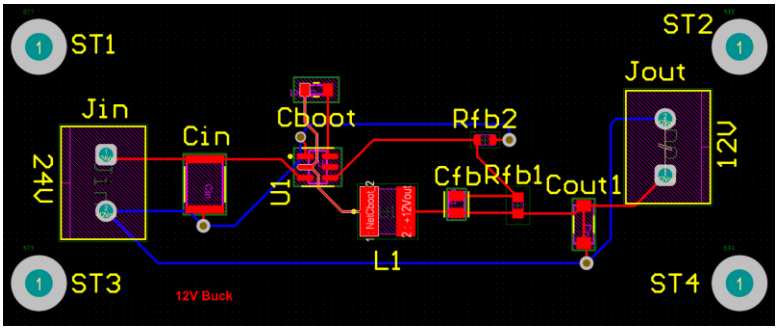


Figure 29. 12V Buck Converter PCB Traces

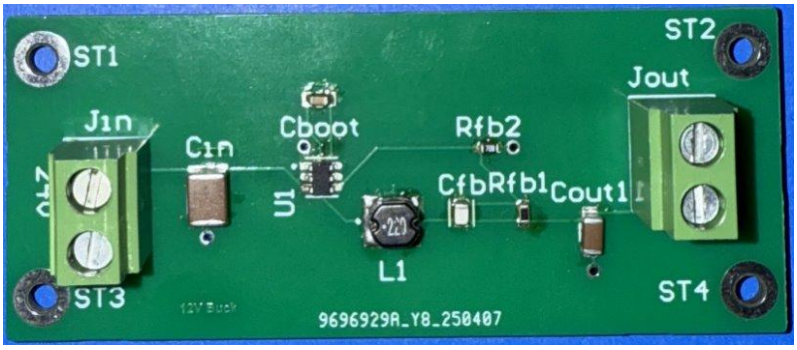


Figure 30. 12V Buck Converter Assembled PCB

4.2.4. 5V Buck Converter

The 5V buck converter powers critical components, including the ESP32 microcontroller, the ESP32-CAM module, the KY-026 Fire Detection Sensor, the HK15138 servo motors, and the LED Strip. The design is based on the TPS56637RPAR IC, which helps step down the input voltage to a stable 5V power rail. Supporting components like high-temperature rated capacitors and resistors are included to ensure reliable performance under operating conditions.

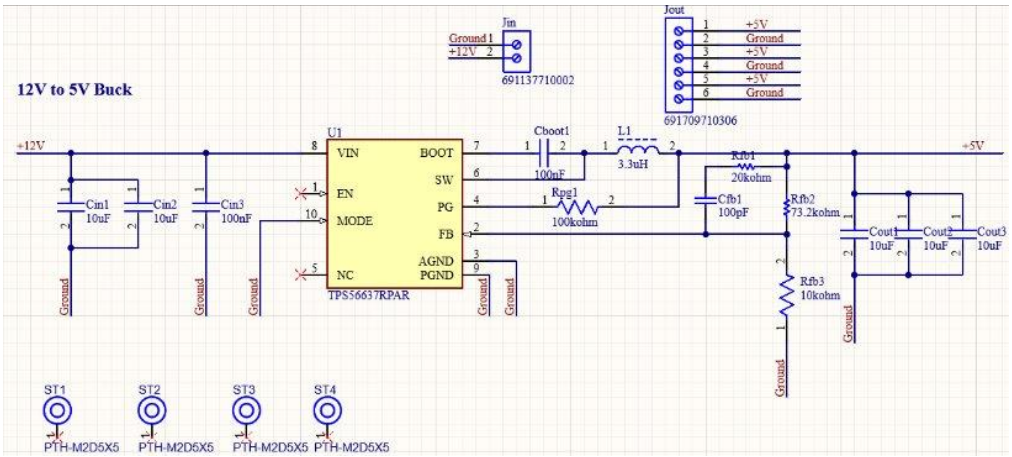


Figure 31. 5V Buck Converter Schematic

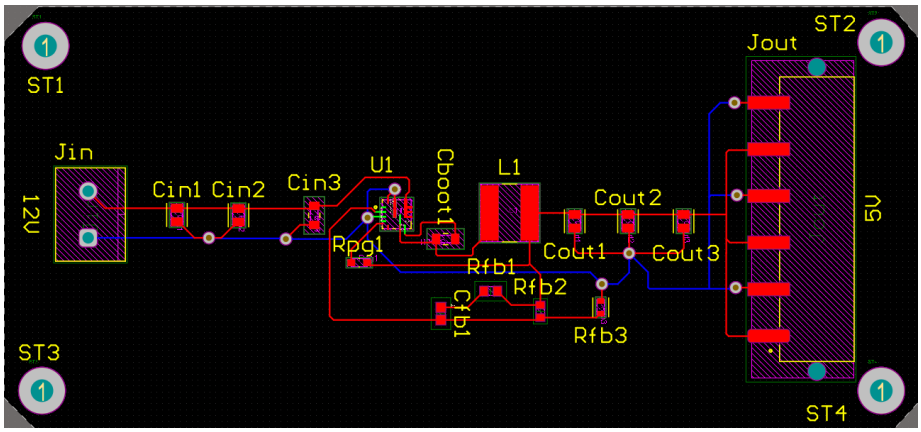


Figure 32. 5V Buck Converter PCB Traces

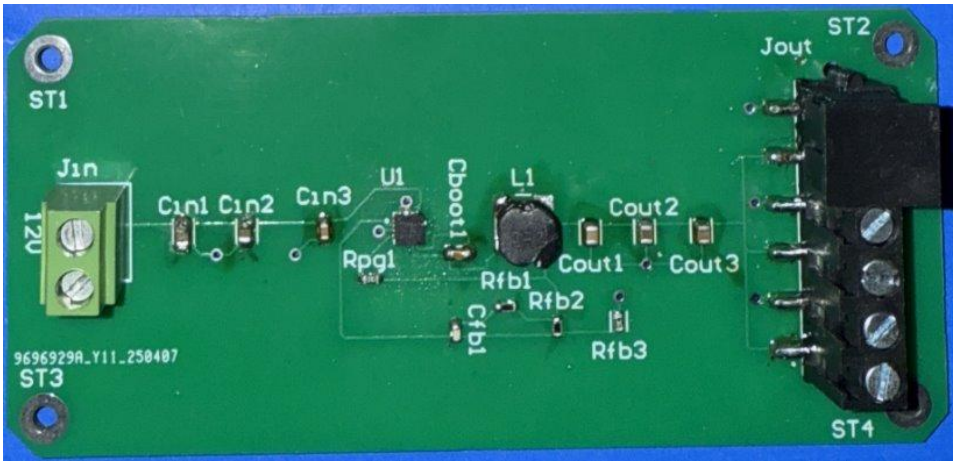


Figure 33. 5V Buck Converter Assembled PCB

4.2.5. AC-DC Charger

To recharge the battery packs without removing them completely from the system, a Supulse Li-Po battery charger is used.

Figure 34. Supulse Li-Po Battery Charger

4.3. Subsystem Validation

4.3.1. Simulation Verification

Validation of the power subsystem began with pre-manufacturing simulations for each PCB design.

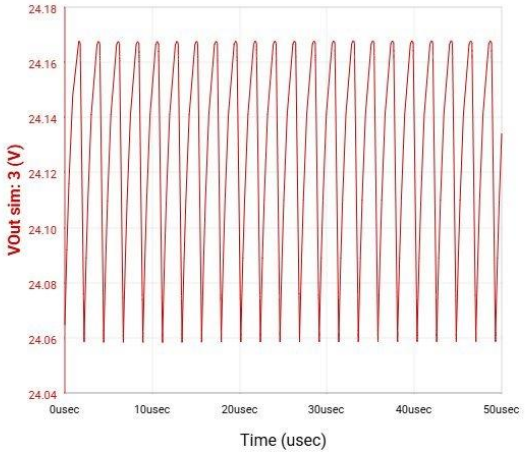


Figure 35. 24V Boost Converter Steady-State Simulation

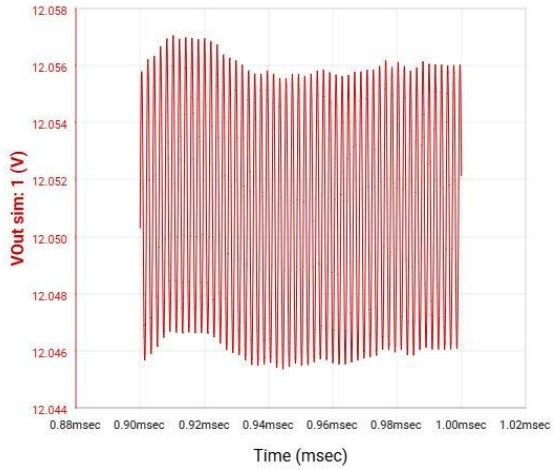


Figure 36. 12V Buck Converter Steady-State Simulation

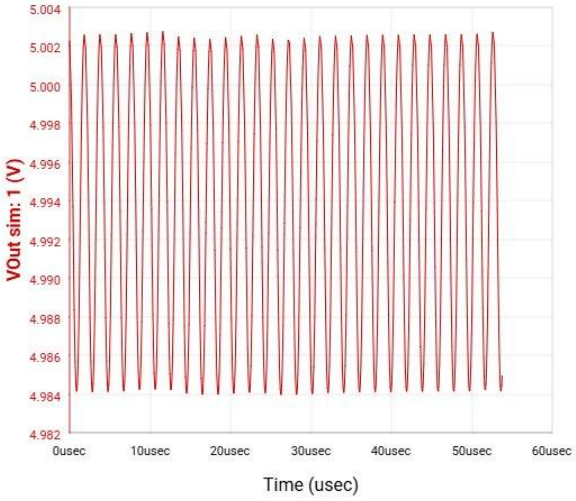


Figure 37. 5V Buck Converter Steady-State Simulation

A summary of simulation results, including the minimum, maximum, and average voltage readings is provided in Table 2.

Table 2. Steady-State Simulation Voltage Averages

	24V Boost	**12V Buck**	**5V Buck**
Average Vout	24.13 V	12.05 V	5.01 V
Maximum Vout	24.55 V	12.55 V	5.07 V
Minimum Vout	23.69 V	11.52 V	4.90 V

4.3.2. Voltage Verification

After PCB manufacturing, testing of each voltage rail was performed under small load conditions using a DC Power Supply and Digital Multimeter. The LM5155-based boost

converter delivered a consistent 22.19V output, close to the battery's combined input voltage, indicating a possible issue with feedback configuration. However, the TPS54202 and TPS56637 buck converters failed to generate stable 5V and 12V outputs, respectively. Multiple debugging attempts were made, including rechecking component values, verifying connections, and measuring voltages across multiple components and transistors, but consistent output regulation could not be achieved. As a precaution, further testing with an electronic load was not performed to avoid risking damage to other components. Due to time constraints, updated PCBs could not be manufactured.

Table 3. 24V Boost Converter Transistor Voltage Measurements

Pinout	Voltage Reading
2: VCC	22.2 V
3: Gate	5.512 mV
5: Current Sense	0.563 V

Table 4. 12V Buck Converter Transistor Voltage Measurements

Pinout	Voltage Reading
2: Switch Node	4.498 mV
3: VIN	23.998 V
5: Feedback	0.022 V
6: Boot	6.490 V

Table 5. 5V Buck Converter Transistor Voltage Measurements

Pinout	Voltage Reading
2: Feedback	29.528 mV
4: Power Good	0.129 mV
6: Switch Node	44.614 mV
7: Boot	0.228 mV
8: VIN	0.651 mV

4.3.3. Thermal Testing

Nevertheless, validation included individual thermal testing, and integrated testing was performed. Each board was run for 5 minutes with another 5 minutes to cool down.

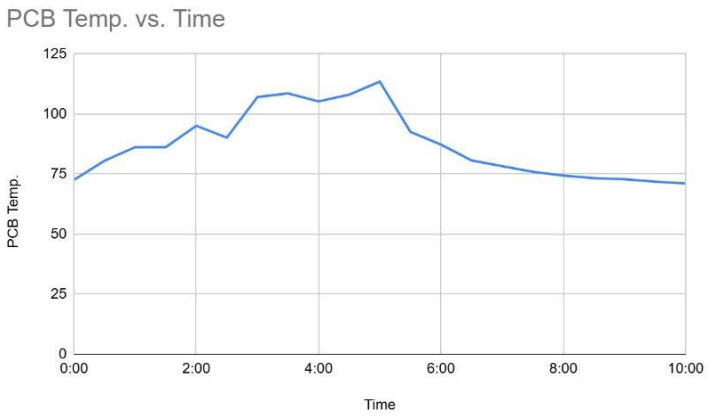


Figure 38. 24V Boost Converter PCB Temperature Over Time Graph

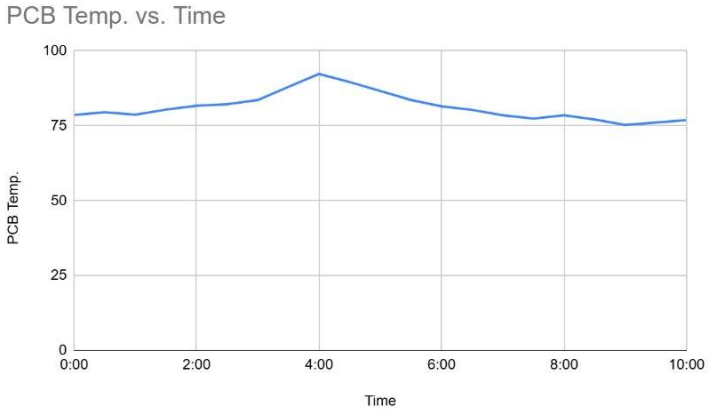


Figure 39. 12V Buck Converter PCB Temperature Over Time Graph

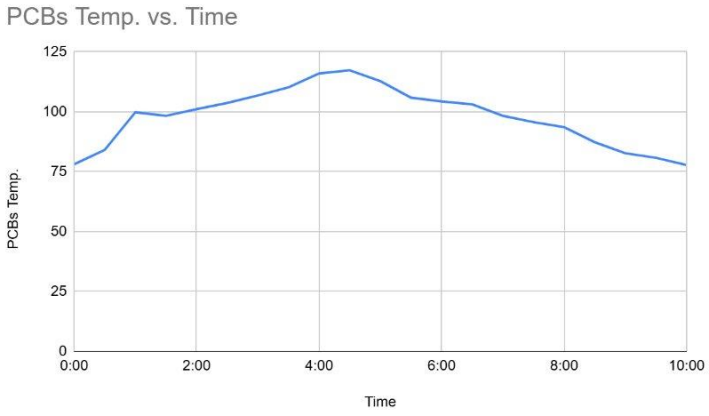


Figure 40. Connected 24V Boost and 12V Buck Converter PCBs Temperature Over Time Graph

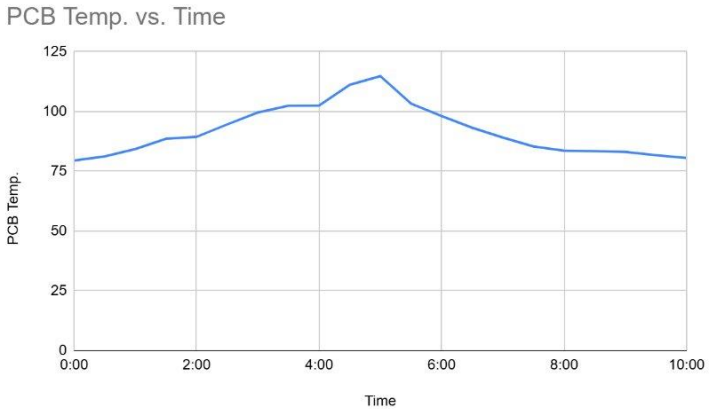


Figure 41. 5V Buck Converter PCB Temperature Over Time Graph

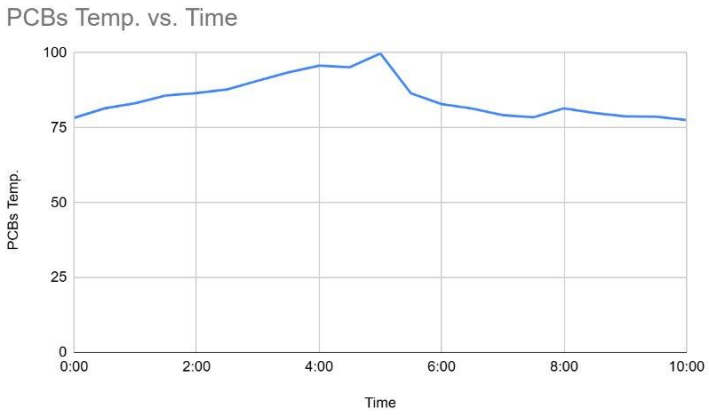


Figure 42. Both Buck Converter Connected PCB Temperature Over Time Graph

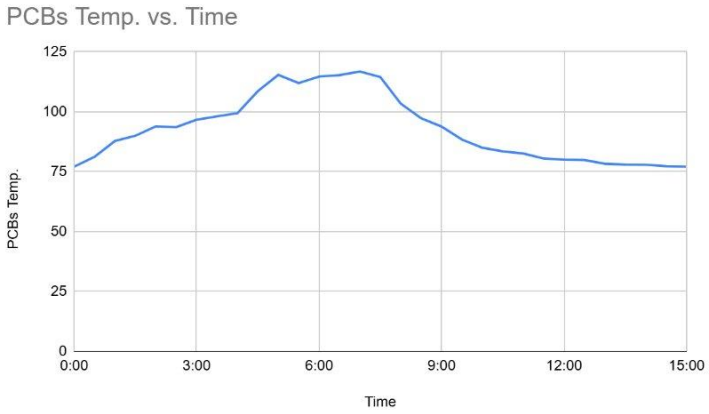


Figure 43. All Boards Connected Temperature Over Time Graph

A summary of average temperature when running, cooling, and overall temperatures during thermal testing is provided in Table 3.

Table 6. Average PBC Temperature Values

	24V Boost	12V Buck	Boost & Buck Connecter	5V Buck	Both Bucks Connected	All Connected
Avg. Temp. When Running	95.77°F	83.75°F	102.5°F	95.28°F	88.92°F	100.04°F
Avg. Temp. Overall	87.24°F	81.26°F	97.85°F	91.87°F	84.91°F	92.84°F
Avg. Temp. When Cooling	81.09°F	79.26°F	97.75°F	90.58°F	82.26°F	87.90°F

4.3.4. Charger Verification

The Li-Po battery charger was validated by monitoring its charging status indicators during operation. When connected to a discharge battery, the charger's red indicator LEDs are turned on, signaling that charging is in progress. Once the batteries reached full charge, the red lights turned off and the green lights lit up. This behavior aligned with the expected functionality of the Supulse charge, demonstrating successful integration and reliable operation for safe battery maintenance.

Figure 44. Battery Charger Red Lights Indicating Charging

Figure 45. Battery Charger Green Lights Indicating Full Charge

4.4. Subsystem Conclusion

Although the power subsystem did not successfully meet the design objectives of providing safe, reliable, and scalable power delivery for Ember Bot, it provided key lessons in converter design, testing, and power system integration. The 24V boost converter achieved partial success, delivering an output close to the target, while the 12V and 5V buck converters failed to operate as intended and will require further debugging and redesigning for future iterations. The Li-Po battery charger performed reliably and added safe charging functionality to the system. Moving forward, improvements will include re-evaluating the layout and component selection for buck converters. Despite the challenges, the work laid a strong foundation for understanding power delivery in embedded systems and highlighted areas for improvement in future designs.

Ember Bot

Yuwen Zheng

SUBSYSTEM REPORTS

REVISION – 1
24 April 2025

5. App Interface Subsystem Report

5.1. Subsystem Introduction

The mobile app subsystem is designed primarily as the controller for the Ember Bot. It will communicate with Ember Bot through the ESP32 microcontroller. The mobile app can receive and send signals from and to the ESP32 microcontroller through a Wi-Fi connection. The mobile app can also receive and display live video signals in the app for user interface. The user will be able to control Ember Bot using the mobile app inside/outside their home.

5.2. Subsystem Details

The app is built with Flutter and Dart programming languages. It can run on both iOS and Android systems. The app includes 6 buttons for now. The most left/right buttons are used to control the left/right track of Ember Bot. The middle button is to control the water nozzle. On the bottom right corner is the video play button that can play and pause video. The two buttons on the app bar are for user settings and user manual. User settings will include features such as LED light, and Ember Bot speed. The user manual will include a guide to operate and quick debugging of Ember Bot. The coordinates are displayed for users and will be sent to ESP32 for calculating nozzle angles. The buttons for the left/right track only have vertical coordinates. The button for the nozzle has both vertical and horizontal coordinates.

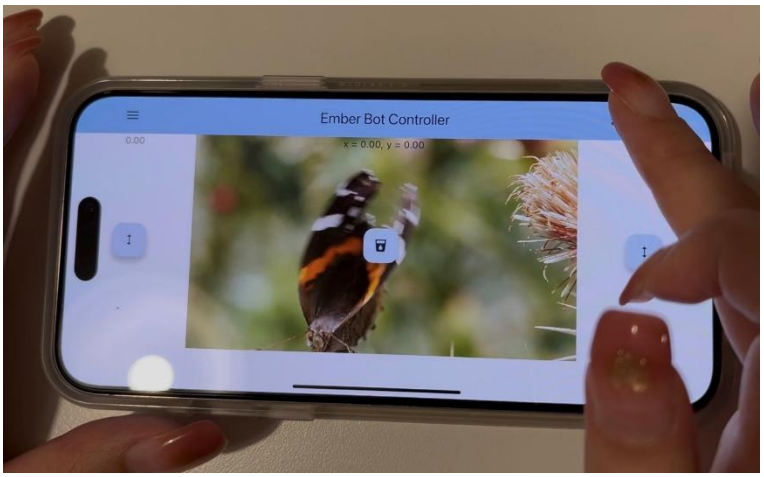


Figure 46. Mobile App Loaded onto iPhone

There are over 500 lines of code written. I will only display a segment of the code. To go through the code, I divide all the code into three parts. The first part of my code is to initiate an app and set up the theme of the app. This code makes sure that the app can start running on both emulator and physical devices. The second part of my code is to implement different buttons needed. To implement draggable buttons, I had to define the different states of each button when they are being taped, dragged, and released. Other buttons implemented involved redirecting to a new page. The third part of the code is to define app states for each widget. Each component in the app is considered a widget. The app is all of the single widgets lying on top of each other. The app states determine the flow of the app when the user presses on different buttons.

```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'package:http/http.dart' as http;
import 'package:video_player/video_player.dart';

Future<ServerResponse> createRequest(double left, double midX, double midY, double right, String ip) async {
  final response = await http.post(
    Uri.parse(ip),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
    },
    body: jsonEncode(<String, double>{
      'left_position': left,
      'mid_x': midX,
      'mid_y': midY,
      'right_position': right}),
  );

  if (response.statusCode == 200) {
    // If the server did return a 201 CREATED response,
    // then parse the JSON.
    return ServerResponse.fromJson(jsonDecode(response.body) as Map<String, dynamic>);
  } else {
    // If the server did not return a 201 CREATED response,
    // then throw an exception.
    print("Connection failed with IP $ip");
    return ServerResponse.fromJson({'null': null});
    // throw Exception('Failed to create request.');
  }
}

class ServerResponse {
  final String time;
  final String response;

  const ServerResponse({required this.time, required this.response});

  factory ServerResponse.fromJson(Map<String, dynamic> json) {
    return switch (json) {
      {'time': String time, 'response': String response} => ServerResponse(time: time, response: response),
      _ => throw const FormatException('Failed to load response.'),
    };
  }
}

void main() {
  runApp(const EmberBotApp());
}
```

Figure 47. Code Segment 1

```
@override
Widget build(BuildContext context) {
  var appState = context.watch<EmberBotAppState>();

  return Scaffold(
    appBar: AppBar(
      backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      centerTitle: true,
      title: Text(widget.title),
      leading:
        IconButton(
          onPressed: () {
            print("Menu button pressed");
          }, icon: Icon(Icons.menu)
        ),
      actions: [
        IconButton(onPressed: () {
          print("Settings button pressed");
          _navigateAndDisplaySettings(appState, context);
          // Navigator.push(
          //   context,
          //   MaterialPageRoute(builder: (context) => const UserSettingPage()),
          // );
        },
        icon: Icon(Icons.settings)),
        IconButton(onPressed: () {
          print("Build button pressed");
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => const UserManualPage()),
          );
        },
        icon: Icon(Icons.build))
      ],
    ),
    body: Center(
      child: Row(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: <Widget>[
          Expanded(
            flex: 1,
            child: Column(
              mainAxisAlignment: MainAxisAlignment.start,
              children: [
                Text(appState.leftPadel.toStringAsFixed(2)),
                SizedBox(height: 114,),
                LeftMovementControlButton(),
              ],
            ),
          ),
          Expanded(
            flex: 3,
```

Figure 48. Code Segment 2

5.3. Subsystem Validation

The design of this subsystem is completely implemented by code. To verify the mobile app subsystem, I first tested all the features on emulators. I successfully loaded the mobile app to my own iPhone. The app can successfully run and function on an iPhone. Due to the

lack of an Android phone, I cannot load the app on an Android system device. However, the app can run on an Android emulator. I also created a mini server just to test if signals were successfully sent from the mobile app. Figure 47 shows all three coordinates that are displayed for each button that are sent to the ESP32 for calculations.

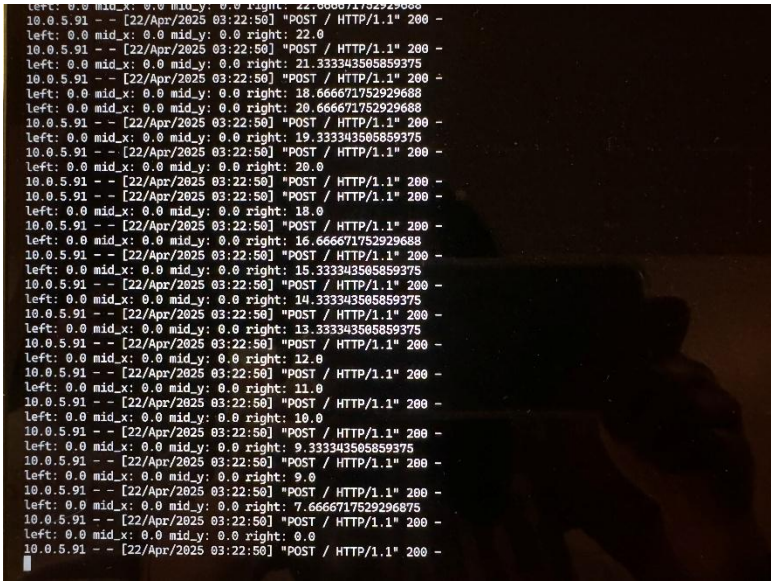


Figure 49. Coordinates Data Sent Back to Server

5.4. Subsystem Conclusion

The mobile app subsystem acts like a tool and controller for Ember Bot. It can easily be used by adults to control Ember Bot when facing fire hazards both indoors and outdoors. The app is successfully designed and developed. It has been both tested on an emulator and loaded onto physical devices. Ember Bot will include extra features and will be upgraded with more buttons if needed.