Automated Salt Spreader Robot

Design Document

sdmay26-07
Dr. Alexander Stoytchev / Advisor
Isaac Blandin / Project Manager & Software Team Lead
Dylan Bunting / Electrical Team Lead
Hoyt Dzwoniarski / Electrical Team
Thomas Huedepohl / Electrical Team
Mikhail McDade / Software Team
Shobhit Singh / Scribe & Software Team
sdmay26-07@iastate.edu
https://sdmay26-07.sd.ece.iastate.edu
Revised: 05/04/2026

Executive Summary

The issue that our solution addresses is the danger and inconvenience of de-icing driveways and sidewalks during the Winter. This is normally only accomplished through doing it yourself or through expensive, paid services. In our context, this particularly concerns those who are mobility limited, such as the elderly, as this could be very dangerous for them to do themselves or working adults which would benefit from the convenience and time savings.

The salt spreading robot needs to be able to spread and contain enough salt to spread up to 2000 sq. ft, have an accuracy of +/- 5 cm at any given point during the operations, detect if it does not have enough salt or batter life to complete the task, operate in temperatures as low as 5 ℉, while traversing icy conditions and navigating slopes as well as shielding from the writers elements such as water or corrosion.

In terms of design, this solution takes the form of a small, mobile robot solution consisting of a drivetrain, salt spreading mechanism, collision avoidance and positioning sensors, and a control center. These subsystems utilize a range of mechanical components, GPS, computer vision, electrical distribution, and software. We are developing this design through a staged approach of high level design, subsystem proof of concepts, detailed design, software testing platform prototype, and then MVP prototype.

The current state of this project at the close of this semester was Prototype 2.0 aka “SALT-E” which is a prototype which physically supports all of our user and functional requirements with all sensors, processors, actuators, and electrical systems implemented. However, to truly be completed, there is some work to be done for full system integration between our subsystems.

Software-wise, this prototype has given us the ability to test all subsystems in isolation, with most of the planned integration testing completed. However, the software subsystems have not been fully integrated together for full system testing. The software is capable of detecting obstacles using computer vision with a high rate of accuracy, drive from its current location to a given GPS coordinate, actuate our salt spreading system, and generate a salt spreading path with an intuitive geometry-based map UI..

The first steps, if this project were to continue, would be to complete the integration needed to perform full system testing. The goal would be to have drive control, collision avoidance, and path finding working as one. Directly following, the next steps would be to move forward with requirements we didn’t implement from the final design, such as waterproofing the electrical system and adding a salt level detection sensor to the final design. We would also need to modify the robot to meet usability standards and further explore the needs of navigating in wintery environments.

Learning Summary

Development Standards & Practices Used

- IEEE 7009:2024 – Fail-safe Mechanisms for Autonomous Systems
- ISO 13482:2014 – Safety Standards for Service Robots
- FCC Part 15 – Regulations on Wi-Fi and Bluetooth Communications

Summary of Requirements

- Carry enough salt for 2000 sq-ft
- Localization accuracy +/- 5 cm
- Monitor salt levels remotely
- Resistant to corrosion and water
- Develop a working prototype
- Total cost should be less than $1000
- Ensure Robot has emergency shutoffs

Applicable Courses from Iowa State University Curriculum

- COMS 5750: Computational Perception
- CpRE 2880: Embedded Systems I
- CpRE 3880: Embedded Systems II
- EE 2300: Electronic Circuits and Systems
- EE3300: Integrated Electronics
- EE3330: Electronic System Design

New Skills/Knowledge acquired that was not taught in courses

Throughout our project team members had to develop skills that were not areas covered in our courses.

- Mechanical Design & CAD Modeling
- 3D Printing-Centric Design
- Market Analysis
- Designing for Winter/Cold Environments
- Autonomous Navigation
- Pathfinding / Path Planning
- GPS-Based Localization
- Rapid Iteration & Prototyping

Table of Contents

1. Introduction 6
1.1. Problem Statement 6
1.2. Intended Users 6
2. Requirements, Constraints, And Standards 8
2.1. Requirements & Constraints 8
2.2. Engineering Standards 8
3 Project Plan 10
3.1 Project Management/Tracking Procedures 10
3.2 Task Decomposition 10
3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria 11
3.4 Project Timeline/Schedule 11
3.5 Risks and Risk Management/Mitigation 12
3.6 Personnel Effort Requirements 13
3.7 Other Resource Requirements 13
4 Design 14
4.1 Design Context 14
4.1.1 Broader Context 14
4.1.2 Prior Work/Solutions 14
4.1.3 Technical Complexity 16
4.2 Design Exploration 17
4.2.1 Design Decisions 17
4.2.2 Ideation 18
4.2.3 Decision-Making and Trade-Off 18
4.3 Final Design 19
4.3.1 Overview 19
4.3.2 Detailed Design and Visual(s) 21
4.3.3 Functionality 34
4.3.4 Areas of Challenge 34
4.4 Technology Considerations 34
5 Testing 36
5.1 Unit Testing 36
5.2 Interface Testing 36
5.3 Integration Testing 36
5.4 System Testing 37
5.5 Regression Testing 37
5.6 Acceptance Testing 37
5.7 User Testing 37
5.8 Results 38
6 Implementation 43
6.1 Current State Of Implementation 43

6.2. Design Analysis 53
7 Ethics and Professional Responsibility 54
7.1 Areas of Professional Responsibility/Codes of Ethics 54
7.2 Four Principles 55
Individual Users - Beneficence 56
Environmental - Nonmaleficence 56
7.3 Virtues 56
8 Conclusions 58
8.1 Summary of Progress 58
8.2 Value Provided 58
8.3 Next Steps 58
9 References 59
10 Appendices 60
10.1 Appendix 1 - Operation Manual 60
10.2 Appendix 2 - Alternative/initial version of design 69
10.3. Appendix 3 – Code 70
10.5 Appendix 4 -Team Control 70
10.5 Appendix 5 Team Contract 72

List of figures/tables/symbols/definitions:

(Numbered as “Figure {Major Section}-{Order in Section}”)

Figure 4-1: Weighted Decision Matrix for Positioning Sensor Selection

Figure 4-2: Prototype 2.0 CAD Design Overview

Figure 4-3: Simple Overall Design Block Diagram

Figure 4-4: System / Subsystem Organization Diagram

Figure 4-5: Control Subsystem Communication Diagram

Figure 4-6: Control Flow Diagram of Proposed Path Planning Algorithm

Figure 4-7: Drive Control Design Code Flow Chart

Figure 4-8: GNSS Positioning Communication Diagram

Figure 4-9: Computer Vision Training and Detection Flow Charts

Figure 4-10: Servo Controlled Gate Linkage

Figure 4-11: Spinner Motor Driver Circuit

Figure 4-12: Power Distribution Diagram

Figure 4-13: Power Distribution PCB Schematic

Figure 4-14: Drivetrain CAD Models

Figure 4-15: Mobile Companion App High Level Architecture Diagram

Figure 5-1: GPS Module Start Up Behavior Test Results

Figure 5-2: Static GPS Testing Extended Test Results

Table 5-3: Equirectangular Approximization Testing Results Categories 1 & 2

Table 5-4: Equirectangular Approximization Testing Results Categories 3 & 4

Table 5-5: CV Obstacle Detection Testing Results

Figure 6-1: Image of Prototype 2.0

Figure 6-2: Created Path From Path Finding Algorithm

Figure 6-3: GPS Antenna and Magnetometer Placement

Figure 6-4: collision detection sensors mounting locations

Figure 6-5: Salt Spreader Gate Implementation

Figure 6-6: Servo Driver Circuit

1. Introduction

1.1. Problem Statement

In regions that experience freezing temperatures during the Winter, homeowners, businesses, and cities alike have to deal with keeping their sidewalks and driveways clean of snow and ice in order to provide a safe environment for pedestrians and vehicles.

While snow removal is a major component of this process, it has a large amount of dedicated equipment and technology that makes this process a breeze. However, de-icing remains a largely manual process, especially for homeowners. The task of de-icing via salt spreading requires time, labor, and greatly benefits from specific timings, which may or may not be convenient for the user (middle of the night, during work, etc). Additionally, if the de-icing is required in conditions such as packed snow or after freezing rain, the task is not just inconvenient; it can be dangerous.

We believe that this task can be performed safely, without the user present, and at any time of the day, by employing an autonomous salt-spreading mobile robot solution that intelligently spreads salt or other ice-combatting solutions across sidewalks and driveways before and after ice may present itself.

1.2. Intended Users

In our analysis of intended/potential users for our proposed solution, we identified three primary groups and developed personas to represent those groups for use-case analysis.

1. Middle to High-Income, Working Adults
    a. Characteristics: This group is busy, on a schedule, highly values their time, has disposable income, and may often be exhausted from their work.
    b. Need Statement: High-earning, working adults need a way to save time and energy de-icing their property because it takes up valuable time out of their already busy, draining schedules.
    c. Value Creation: Our solution will serve these users by performing the task of spreading salt before and after ice formation. This frees up time in their day by completely clearing their driveway during light snowfall or ice, or only requiring basic levels of shoveling or hired snow removal after heavier snowfall. Due to its autonomous function, without the need for user presence, this can provide value to their life, even while they are sleeping or at work.

2. Elderly, Disabled, or Other Mobility-Limited Individuals
    a. Characteristics: This type of user is largely defined by either a lack of mobility or a general increased fall risk. They have larger risks associated with navigating around ice in the Winter. The potential consequences of falling on the ice are much greater for these individuals.
    b. Need Statement: The elderly and other mobility-limited individuals need a readily available way to have their sidewalks and driveways cleared of ice without putting themselves in danger.

   c. Value Creation: Our solution allows this group to have their property de-iced at the press of a button, without relying on the schedules of others, such as relatives or hired companies, or putting themselves in danger. This grants them the freedom to clear and navigate their sidewalks and driveways without the risk of falling, which could have major consequences for this demographic group.
3. Cities and Universities
   a. Characteristics: Universities and certain cities have a large area of sidewalks and walking paths that experience a large amount of foot traffic as compared to other areas. They have to employ a large amount of resources and potentially expensive labor in order to clear ice from these areas.
   b. Need Statement: Universities and cities need cheaper, more efficient ways to de-ice their expansive sidewalks and walking paths when winter conditions cause full ice coverage.
   c. Value Creation: Universities and dense cities could use fleets of these autonomous salt spreading robots to both pre-treat and post-treat their sidewalks and walking paths without the need for large amounts of labor to either complete the task manually or operate equipment. This could reduce labor costs, reduce the need for temporary/on-call labor, and improve working hours for the existing workforce, as this could reduce the need for certain employees to work during the late hours of the night in anticipation of a night or morning ice storm.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

Functional Requirements:

1. The mobile robot must be able to spread salt autonomously over a pre-defined area up to 2000 sq. ft. (constraint).
2. The tracking system must be able to use a combination of sensors, absolute and relative, for localization with positional accuracy within ∓5 cm at any given point during operation (constraint).
3. The system must be able to detect when it does not have enough salt/ice melt on board to complete a task.
4. The system must be able to operate in temperatures as low as 5 °F, or the minimum effective temperature of most ice melt solutions (constraint).

Physical Requirements:

1. The design must be able to carry enough salt to cover the majority of residential driveways and sidewalks without being refilled (constraint).
2. The drivetrain must be able to traverse icy conditions and navigate within typical sizes and slopes of sidewalks and driveways.
3. The electronic systems must be shielded from moisture and unaffected by wet environments and temperature cycling.
4. Materials must resist corrosion, even when exposed to moisture or salt.

Environmental Requirements:

1. The system must not use too much salt or misplace salt in such a way that would produce excess runoff or significantly damage vegetation.

UI/UX Requirements:

1. The salt reservoir must be easily refilled by the user.
2. The user interface must be simple. It must be intuitive, even to those who are not familiar or overly competent with technology, such as the elderly.

2.2. Engineering Standards

By utilizing engineering standards from various organizations, we are able to guide the creation of our requirements such that our design will be safe, user-friendly, ethically sound, and avoid interference with the ever-growing world of technology. Below, we have listed three of the standards that have the most impact on our design choices.

1. IEEE 7009-2024
    a. This set of standards provides methodologies and tools for developers to design and implement fail-safe mechanisms for autonomous and semi-autonomous systems that are efficient, robust, and transparent. Being that our solution is a

fully autonomous mobile robot, we will need to implement these standards into our physical and software designs to allow for effective fail-safes to maintain safety wherever our system is in operation.

2. ISO 13482:2014
    a. This document covers a wide range of safety standards in relation to service robots (as opposed to industrial robots, which our system would not be considered). It covers general safety requirements and methodologies for developing and testing safety requirements unique to its context. We will need to refer to this to responsibly develop our own requirements for systems such as navigation and create a sound testing framework.
3. FCC Part 15
    a. This regulation governs how devices utilizing Wi-Fi and Bluetooth communicate to ensure that there is no unnecessary or harmful interference to other radio services. We will need to follow these guidelines to ensure that our systems, whether communication with the robot or differential positioning systems, do not interfere with nearby systems that also rely on these technologies.

3 Project Plan

3.1 Project Management/Tracking Procedures

Our team uses an **Agile-Scrum workflow** managed through **Atlassian Jira and Confluence**. We run **weekly sprints**, each beginning with a planning session where tasks are broken into Jira issues under their respective subsystem epics (Platform/Mechanics, Power & Electronics, Sensors/Navigation, Software/Compute, and Salt System). Each issue includes an assignee, description, and acceptance criteria.

- **Sprint master:** Isaac Blandin (rotates weekly)
- **Progress tracking:** Each member updates Jira daily; Confluence is used for documentation, design logs, and meeting minutes.
- **Meetings:** The team meets with the faculty advisor (Dr. Stoytchev) every **Thursday at 4:30 PM** for progress review and feedback. A second **team-only weekly check-in** reviews individual blockers, assigns next-week tasks, and ensures balanced workload.
- **Version control:** Code, CAD, and documents are stored in a private **GitHub repository** linked to Jira for traceability.

This structure keeps the project adaptable for parallel electrical and computer subdivisions that will form later in the semester.

3.2 Task Decomposition

Subsystem	Major Tasks	Sub-Tasks	Dependencies
Platform & Mechanics	Base chassis and enclosure design	CAD modeling, ground-clearance verification, waterproofing, gearbox evaluation	Required before power integration
Power & Electronics	Battery selection and power distribution	Temperature tests, wiring harness, motor driver integration	Requires chassis dimensions
Sensors & Navigation	Localization and perception	RTK-GNSS testing, IMU + camera fusion, MicroPython/CircuitPython driver validation	Requires compute interface
Software & Compute	Embedded control and ROS 2 setup	Motor PWM control, sensor data parsing, path logic, telemetry dashboard	Requires functional sensors
Salt System	Dispensing mechanism	Hopper design, flow-rate calibration, motor feedback, clog testing	Depends on platform frame

Subsystem	Major Tasks	Sub-Tasks	Dependencies
Integration & Testing	System bring-up	Bench tests, interface validation, field test with sand media	Cross-subsystem
Documentation & Reporting	Weekly report and Confluence updates	Sprint summary, advisor notes, QA review	Ongoing

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone	Metric / Evaluation Criterion	Target Date
Sensor library validation	Each sensor communicates successfully via MicroPython or CircuitPython	Oct 25 2025
Component procurement complete	100 % parts ordered & delivered	Oct 31 2025
Mechanical prototype (MVP) ready	Robot frame assembled & motors operational	Dec 1 2025
Navigation + drive test	Path accuracy ≤ 15 cm error	Feb 2026
Dispensing calibration	Flow rate 50–250 g/min ± 15 % uniformity	Mar 2026
Full system demo	Autonomous run ≥ 1 km continuous operation	Apr 2026

3.4 Project Timeline/Schedule

Fall 2025

- Weeks 1–6: Sensor library testing on personal devices; complete SIC 3D-printing training.
- Weeks 7–9: Finalize CAD chassis design, order components (before Halloween).
- Weeks 10–15: Assemble and test first MVP prototype; log results in Confluence.

Spring 2026

- Weeks 1–8: Integrate navigation and dispensing subsystems; conduct bench and field tests.
- Weeks 9–13: Refine software and fusion algorithms; complete Beta field demo.
- Weeks 14–16: Finalize documentation and present final demo.

3.5 Risks and Risk Management/Mitigation

Risk	Probability	Impact	Mitigation
Battery performance drops below 15 °F	0.5	High	Thermal insulation and battery chemistry tests before integration
Motor torque insufficient on ice	0.4	High	Add gearbox or select higher-torque motors after bench tests
Sensor fusion or firmware incompatibility	0.5	High	Parallel testing of MicroPython vs CircuitPython drivers
3D printer training delays (SIC)	0.6	Medium	Schedule training early; outsource print if delayed
Component shipping delays	0.4	Medium	Order before Halloween deadline with backup vendors
Software bugs affect motion control	0.3	Medium	Version control with branch testing and peer review

The primary risks that we ended up running into were not the ones that we initially planned to experience. Much of our risks that were time drains were related to integration strain between subsystems, whether software to software or software to electro-mechanical. This was addressed by temporarily restructuring from software and electrical teams into smaller “integration teams” where they could work together to integrate their parts.

Unfortunately, a risk that we did not account for was mechanical failure of components. These failures, especially the failure of gears in our drivetrain created major pauses in our testing and overall progress. As this occurred too late, we were not able to mitigate this moving forward other than opting to use off-the-shelf, more expensive components for critical structure later on.

3.6 Personnel Effort Requirements

Team Member	Role / Focus	Weekly Hours	Semester Total (≈ 30 weeks)	Actual Total
Isaac Blandin	Sprint master, mechanical CAD, gearbox evaluation	6	180	188.5
Dylan Bunting	Power systems, motor drivers, battery integration	6	180	173.5
Hoyt Dzwoniarski	Sensor testing (MicroPython/CircuitPython), navigation prototype	6	180	150
Thomas Huedepohl	Chassis design, 3D printing coordination, assembly	6	180	102
Mikhail McDade	Software architecture, ROS 2 planning, telemetry	6	180	168
Shobhit Singh	Documentation, QA tracking, risk register updates	6	180	172

Total Effort: ~1080 person-hour

Actual Total Effort: ~954 person-hour

The difference in expected total hours versus actual total stemmed from waiting for parts, mechanical issues, and programming issues that arose later in the project, slowing progress. Certain parts of the robot also came along quicker than expected in some situations, like assembly of the robot and circuit integration.

3.7 Other Resource Requirements

Facilities: Student Innovation Center (SIC) 3D printing lab, ETG filament supply, campus outdoor walkway testing site.

Hardware: Lead acid batteries, motor drivers, RTK-GNSS module, IMU, camera sensor, Arduino UNO, magnetometer, ESP32

Software: Jira and Confluence (Atlassian), GitHub repo, Arduino IDE, assorted Arduino Libraries.

Miscellaneous: Safety gear (E-stop switches, fuses), weatherproof materials, adhesives, Loctite, vibration resistant fasteners, non-ferrous fasteners (for near the magnetometer).

4 Design

4.1 Design Context

4.1.1 Broader Context

Our design problem exists as a service robot that can improve the safety and well-being of users by allowing them to avoid the potentially dangerous task of spreading salt after ice formation, along with saving time and energy for users as well. However, there are considerations that we must make while designing this system, as if poorly implemented, our solution could produce negative effects simultaneously. Users and others in the area of our solution's deployment could be at safety risk if our system were not designed with collision avoidance in mind. Additionally, misuse of salt or other ice-melting solutions can often be hazardous to people and animals if it were to come in direct contact with their skin or are ingested (however, this remains an issue with most other substitutes). Our product also has the potential to displace or eliminate current employment opportunities if it becomes a viable substitute for hired ice-removal services. Ultimately, we believe that our solution, if implemented with these considerations in mind, will provide societal and customer value through improved safety, comfort, and time savings for individual customers and communities that far outweigh the potential downsides.

We believe that our solution does not have a significant impact on the global, cultural, or social aspects of society, other than the indirect potential of reducing the market share of traditional snow removal service providers, which are often small or family-owned businesses. We believe that these negative impacts can also be offset by our solution if we make it a viable economic option for these businesses to implement, reducing their costs in the process.

In terms of environmental context, we again have some considerations that we must make in the process of our designs. Due to the ability for over-salting to cause harm to vegetation and water sources in the nearby area, we must use due diligence in developing our salt-spreading solutions to ensure environmentally sustainable amounts are distributed, along with implementing fail-safes against accidental dispersal.

Within an economic context, our most important consideration is that of being economically viable, and at that, being able to compete with the main substitute good of snow-removal services. We must, then, design our solution to be cost-friendly enough to compete with this substitute or pose ourselves as a complementary good instead of a substitute, via designing for commercial usage.

4.1.2 Prior Work/Solutions

During our team's research on prior work, we discovered that we would actually be first-movers in the specific niche that our product is targeted to compete within, meaning that we did not have direct, similar products to research. As a result, we looked at current solutions in other industries or niches that would serve as reference materials for different subsystems or problems that we expect to encounter. In this search, we found three main products: a large commercial snow-clearing robot with an optional salt-spreading attachment, a personal snow-blowing service robot, and autonomous lawn-mowing robots. Our analysis of these solutions are as follows:

Left Hand Robotics Snowbot Pro [1]:

Description:

- This robot is a large, autonomous snow-removal robot designed for commercial or municipality usage, being around the size of a ride-on lawn mower. It has a variety of attachments, including one designed for spreading salt.

How Ours Stacks Up:

- Pros:
    - Our solution is targeted at coming at a much smaller cost (<$5000 vs $35000)
    - More capable of handling smaller or more congested spaces
    - Capable of more residential or personal usage
- Cons:
    - Our solution would not be viable for the commercial snow-clearing operations that the Snowbot Pro is designed for.
    - Our robot does not possess larger snow-clearing capabilities, such as plowing or brushing.
- Conclusion:
    - The Snowbot Pro and our solution, while accomplishing some of the same tasks, are not designed to fill the same niche. Ours is a personal/residential service robot, while theirs is an industrial robot. However, we are still able to use this solution as a reference while we tackle design challenges such as salt-spreading mechanisms, navigation, and handling winter conditions.

Yarbo Snow Blower [2]:

Description:

- The Yarbo Snow Blower is a personal service robot that autonomously clears residential driveways and sidewalks of snow. It is one of the options for attachments that is built on the Yarbo Core platform, which is a multi-usage autonomous robot that can serve multiple needs across all seasons.

How Ours Stacks Up:

- Pros:
    - Our design is capable of de-icing driveways
- Cons:
    - Our design is reliant on the snow being removed before we are able to spread salt effectively.
    - Not built on a platform that is as versatile or useful for all seasons.
- Conclusion:
    - One of the main takeaways from our analysis of this product is that we would likely actually be positioned as a complementary good to a product such as the Yarbo Snow Blower, allowing one to implement both solutions for a fully autonomous snow removal and de-icing system. Additionally, we were able to take a lot of

inspiration from this system as it was a proven system for a small service robot to be deployed in the Winter conditions.

Husqvarna Automower [3]:

Description:

- The Husqvarna Autonmower is an autonomous service robot that mows lawns and is positioned for personal, residential usage. It implements various navigational systems, including underground boundary wires.

How Ours Stacks Up:

- Due to serving a significantly different purpose and task, we did not feel the need to draw direct pros and cons against the product. We researched this due to it being one of the cheapest and most mass-produced outdoor service robots that we found on the market. We found that this would be a good reference for how they were able to bring down costs, along with the general knowledge from their navigation and other mobile robot-related functions.

Prior Work Research References:

[1] T. Cozzens, "Autonomous Snowbot Pro hits the sidewalk," *GPS WORLD*, May 13, 2019. [Online]. Available: https://www.gpsworld.com/autonomous-snowbot-pro-hits-the-sidewalk/. [Accessed Oct. 28, 2025].

[2] "Yarbo Snow Blower," *yarbo.com*. [Online]. Available: https://www.yarbo.com/products/snow-blower-robot?srsltid=AfmBOopBsykaTng7mL1GXIdoekc9fMoDQVTO1UuS27EhjqroopcLOo2d. [Accessed Oct. 28, 2025].

[3] "Automower 450X," *husqvarna.com*, [Online]. Available: https://www.husqvarna.com/us/robotic-lawn-mowers/automower-450x/. [Accessed Oct. 28, 2025].

4.1.3 Technical Complexity

As our project is a mobile robot solution, our design includes a large set of complex subsystems ranging from areas including physical, software, and electrical, we believe that there is an adequate amount of technical complexity in both breadth and depth. The subsystems that are currently planned are as follows:

- Control
  - As the brain of our solution, our control subsystem must be able to combine sensor, navigation, and collision-detection data in order to effectively command our drive and salt spreading subsystems to accomplish the task.
- Navigation/Positioning
  - Using both absolute and relative position sources, such as GPS, correction data, and local information, this subsystem needs to be able to feed reliable and accurate positional data to be used in path-finding and following algorithms.

- Collision Detection
    - This subsystem combines computer vision, bump sensors, and time-of-flight sensors to detect potential collisions before the robot causes any damage to itself or its surroundings. This subsystem uses this to make navigation decisions or trigger fail-safes.
- Salt Spreading
    - This subsystem is required to both physically spread the salt through motor control and to control and monitor the amount of salt being spread at any given time.
- Power
    - This subsystem maintains and monitors the on-board battery to detect when the system needs to return. It is also responsible for distributing and supplying the correct voltage that each subsystem or electronic component requires.
- Drivetrain
    - This system is primarily made of a few parts: motors, motor control, and the physical method of moving, considering the needs of navigating in winter conditions.
- Mobile App
    - This app is used to monitor or deploy our system through a user interface that is both sufficient in detail and intuitive enough for all of our target users.

We believe that this project meets the requirements of sufficient technical complexity, as the range of subsystems is able to provide a challenge for each of our distinct specializations, and the integration of all subsystems will be a challenge in itself. As our project is exploratory in nature, we will also need to navigate through many unforeseen challenges and adapt to them in stride.

4.2 Design Exploration

4.2.1 Design Decisions

Though we have had a large number of decisions due to the complexity of a mobile robot, there were three primary design decisions that we felt were both complex and crucial to our design as it is today. These decisions are as follows:

1. Drivetrain: Tracks vs Wheels
    a. This was one of the first decisions that we made as a team, since it would form the base for all of the other designs around it, along with being a large enough investment that we could not effectively change our decision down the line due to budget constraints. This decision was ultimately made through a combination of benefit comparison, along with referencing other winter-oriented mobile robot solutions. We were able to determine that tracks experience a higher efficiency with lower slip, which is important in winter's low-friction conditions. Other winter-oriented service robots, such as the Yarbo Snow Blower (see 4.1.2), also use this drive system and have seen success.
    b. Later on, this decision was reversed due to economic constraints. Tracks proved to not be economically viable under our development budget, and thus, we had to pivot. Additionally, the time-to-implementation of a tracked system would have consumed much more time than our final, differential-drive wheeled drive system,

which was an important consideration in a project which already experienced much scope creep.

2. Positioning Sensors
    a. Since our solution will never be successful without accurate and reliable positioning data, this was and still is one of the most crucial design decisions to ensure the viability of our project. This required a deep cost-benefit analysis of each solution and combinations/permutations of each solution via sensor fusion.
    b. We eventually settled on using GPS plus dead-reckoning along with a magnetometer for heading. This tradeoff can be seen in 4.2.2 and 4.2.3
3. Power System
    a. As a fully-electric robot solution, our power system is a critical subsystem of our robot that must be done correctly for us to create a viable solution, along with being able to integrate all of our subsystems into a single cohesive system. This required a large number of factors to be considered such as price, density, weight, safety, and charge/discharge ability.

    b. Our team decided to use sealed lead-acid batteries, which provide a very cheap power to capacity ratio and most importantly, is a very safe option, being a much lower fire risk than other options. However, we have had to sacrifice density, weight, and discharge rates. These tradeoffs had effects on the mechanical design later on due to the size and weight forcing both physical dimensions and torque needed to move the system.

4.2.2 Ideation

One decision that we had to identify multiple outcomes for and had a large weight placed on making the correct decision for in our design was the set of sensors that we would use for positioning data to drive our navigation system. This decision was made even harder by the large number of sensor option candidates that existed in this space and the potential for a combination of sensors via sensor fusion. Our top-five candidates are listed below:

1. Differential GPS
    a. Positional data only from differential GPS for all of our positioning.
2. Differential GPS + Dead-Reckoning
    a. Sensor-fusion of differential GPS with dead-reckoning, depending on the confidence in the GPS signal.
3. Dead Reckoning Only
    a. Usage of dead-reckoning only via both wheel encoders and an IMU.
4. IR/RF Beacons
    a. Usage of stationary IR or RF beacons, in which the rover determines its relative position based on differential timing analysis.
5. Computer Vision Image Targets
    a. Usage of computer vision targets that can be used to determine orientation and relative position from each target.

4.2.3 Decision-Making and Trade-Off

The following figure is a weighted decision matrix that we used to select the positioning sensor option for our project. It weighs cost, accuracy, uptime, and ease of implementation.

	Affordability	Accuracy	Up-Time	Ease-of-Implementation	Total
Criteria Rating	3	5	4	2	
DGPS	4	4	3	3	
Weighted	12	20	12	6	50
DGPS + Dead Reckoning	3	5	5	2	
Weighted	9	25	20	4	58
Dead Reckoning	5	2	5	5	
Weighted	15	10	20	10	55
IR/RF Beacons	2	5	4	1	
Weighted	6	25	16	2	49
CV Image Targets	4	3	4	3	
Weighted	12	15	16	6	49

Figure 4-1: Weighted Decision Matrix for Positioning Sensor Selection

Using this matrix, we were able to determine to use a combination of differential GPS and dead-reckoning through sensor fusion as this scored the highest through our matrix, primarily through its high accuracy and outstanding uptime.

4.3 Final Design

4.3.1 Overview

At its core, our design strives to be a near final stage prototype of a residential salt-spreading robot. Physically, it is a mobile robot solution using a large number of 3D printed components to allow for rapid iteration and prototyping that is capable of driving and spreading salt with additional physical components for sensor mounting and collision detection.

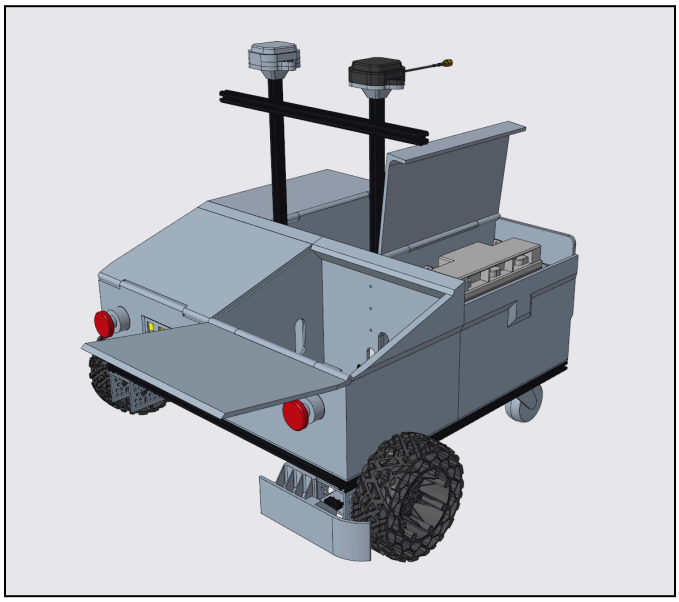


Figure 4-2: Prototype 2.0 CAD Design Overview

However, the true design work went into the integration of a large number of subsystems and functions into a single mobile robot application. This involves the integration of electrical, software, and physical components. On the electrical side, as a mobile solution, we had to choose

and design a battery-powered power distribution system that can not only provide enough power capacity for all functions but also provide the correct voltages to power all of our different subsystems from motors to low-power sensors. The software side of the design involves a large number of subsystems including collision detection and avoidance to ensure safe operation, sensor suites to provide environmental details and physical location for navigation, and our navigation must be able to provide efficient path planning for a variety of residential driveway and sidewalk settings. Physically, the robot must be able to support the previously stated systems while providing physical motion through a drivetrain and actually spread salt to accomplish the goal of the project.

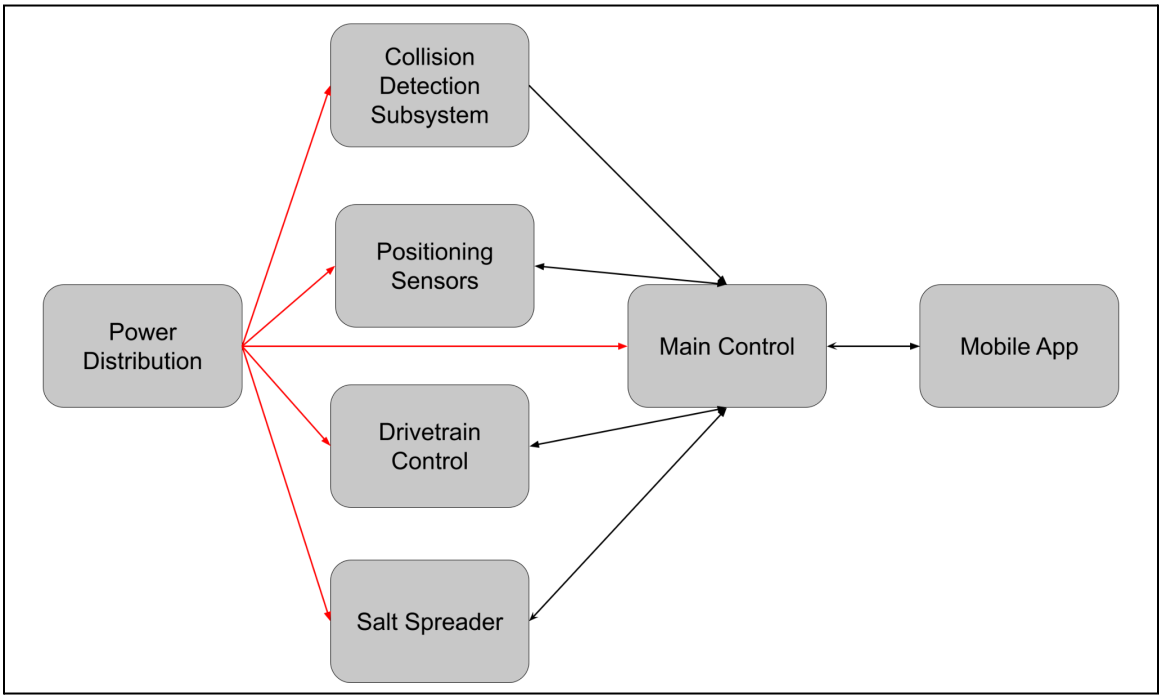


Figure 4-3: Simple Overall Design Block Diagram

4.3.2 Detailed Design and Visual(s)

Overall System Organization

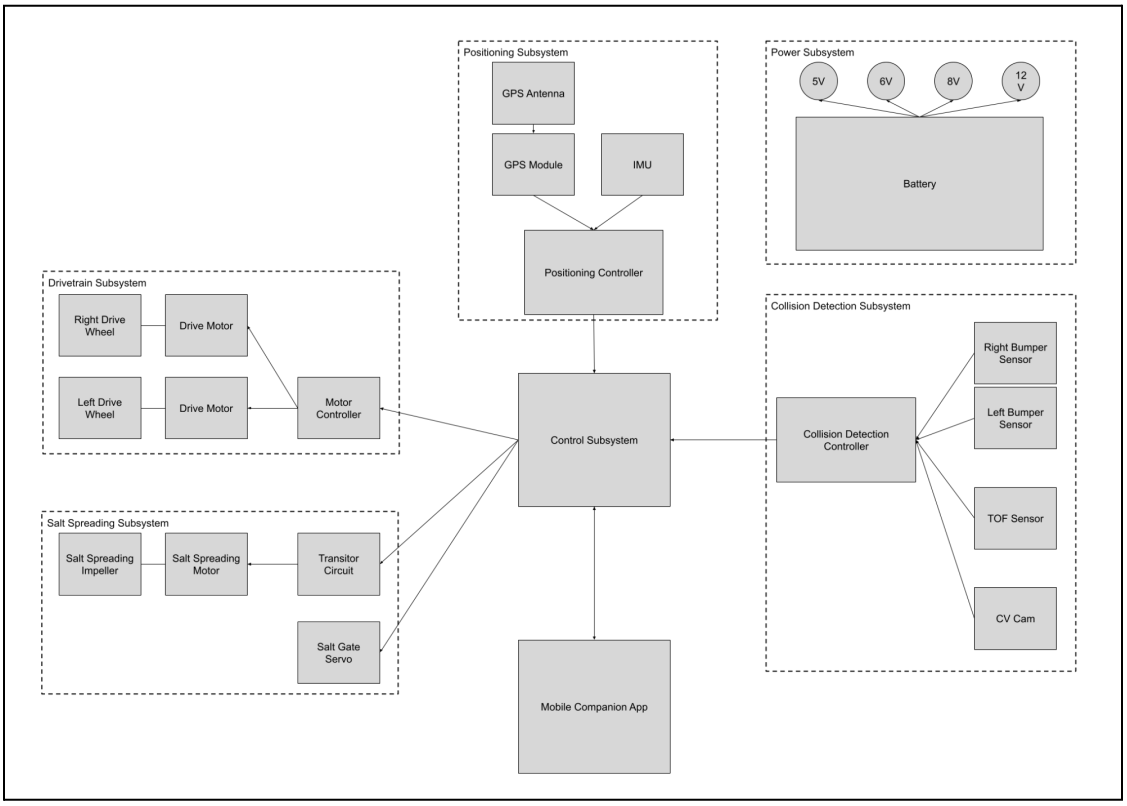


Figure 4-4: System / Subsystem Organization Diagram

The overall application is designed to be a distributed system of sensors, actuators, processors, and co-processors which will allow us to avoid the usual pitfalls of lower-resource processors. While it increases the technical complexity and complexity of integration, it will prevent us from using too much memory or slowing loop times on our main controller.

The main controller will be communicating with the following other subsystems or components:

- Mobile Companion App: user inputs
- Salt Spreading Subsystem: electro-mechanical part of salt distribution
- Drivetrain Motor Control: Velocity commands & feedback
- Positioning: GPS data, real-time corrections, and magnetometer heading
- Power Distribution: monitor batteries and use given power
- Collision Detection / Avoidance: respond to detected obstacles

Control Subsystem - Communications

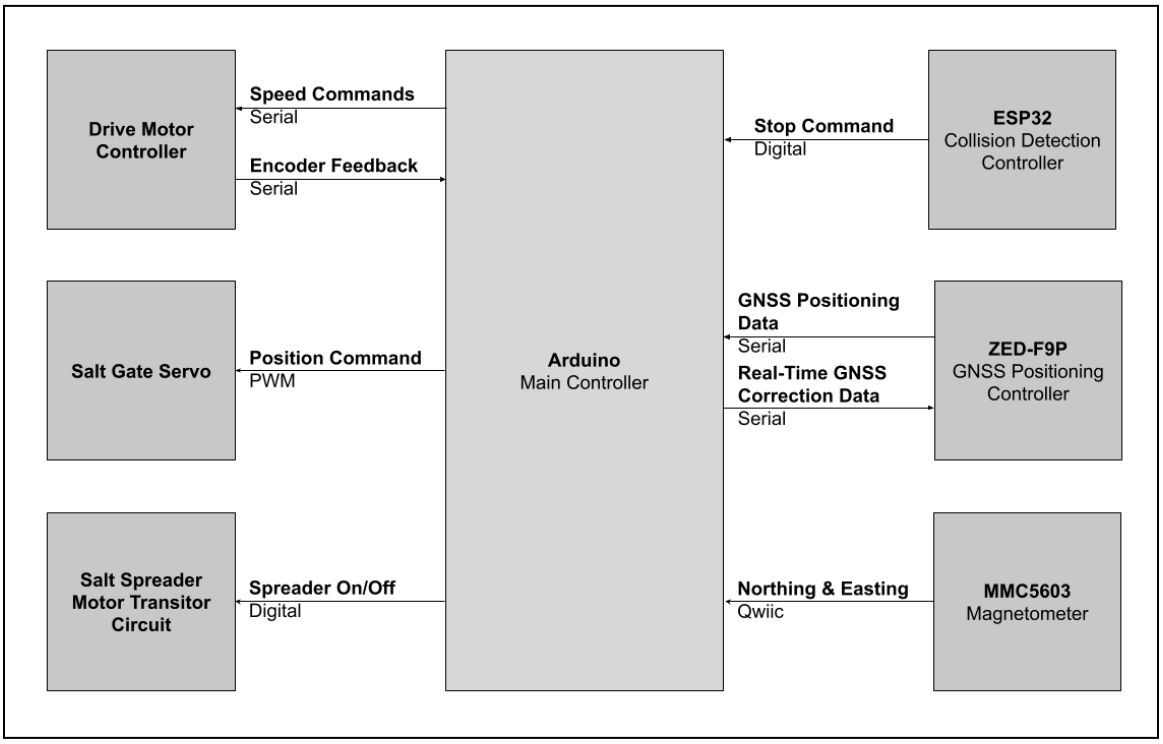


Figure 4-5: Control Subsystem Communication Diagram

For our control subsystem our final design involves a large number of planned interactions between the subsystems and our distributed suite of sensors and co-processors, all being conducted by our main controller.

The communications from the frame of the main controller are as follows:

- Drive Motor Controller
    - Main controller sends velocity commands
    - Motor controller sends encoder feedback
- Salt Spreader
    - Main controller sends PWM angle commands to salt gate servo
    - Main controller sends digital start/stop commands to the spreader spinner transistor circuit
- Collision Detection Controller
    - Collision detection controller sends stop commands and context such as vision zones
- GNSS Positioning Sensor
    - GNSS receiver sends raw / corrected position data
    - Main controller sends real-time correction data
- Magnetometer
    - Main controller receives raw heading data over I2C

Control Subsystem - Pathfinding

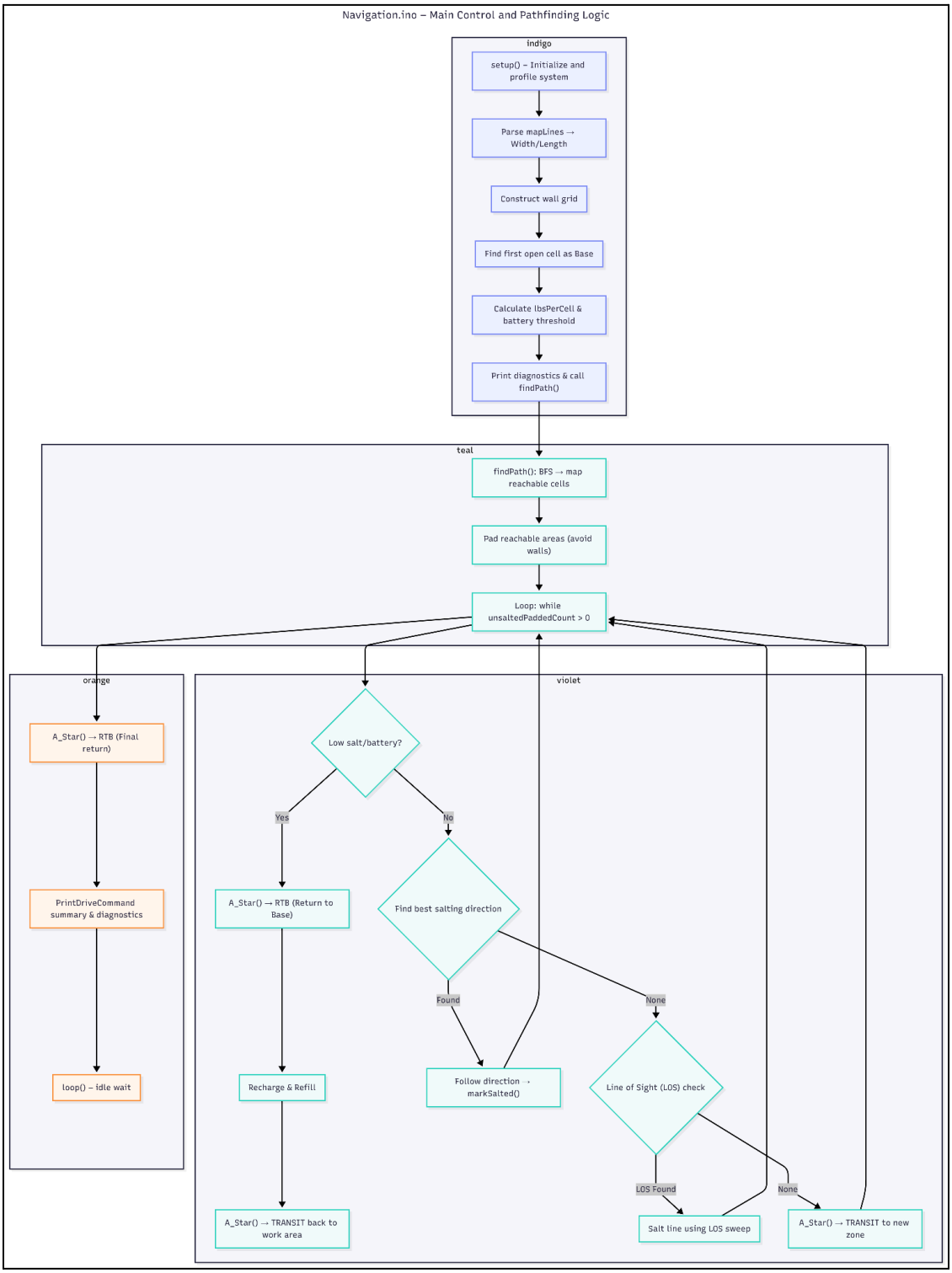


Figure 4-6: Control Flow Diagram of Proposed Path Planning Algorithm

For our actual path planning of our robot, we needed to run an algorithm that could be given an area on a map that needs to be salted.

For memory-constrant reasons this has been designed to use a local grid rather than much more complicated spline-based systems. This is to be implemented using a “home” coordinate and then propagating the coordinates for the other points on the grid.

This is to use an A* algorithm to find home and graph exploration to find long, continuous, unsalted paths for the robot. It will consider factors like not unnecessarily turning, which areas are salted or not, salt capacity, and battery capacity.

Control Subsystem - Drive Control

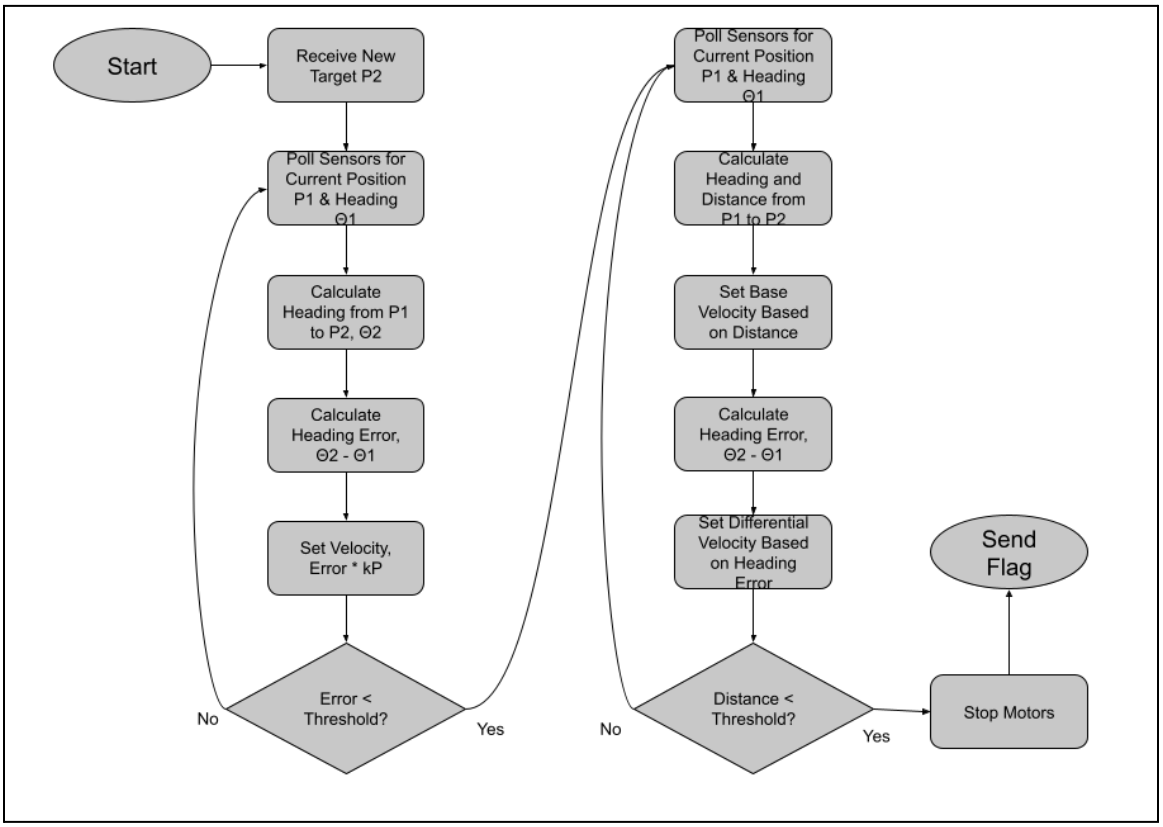


Figure 4-7: Drive Control Design Code Flow Chart

In order to actually drive our robot from its current position to the next position calculated by our path planning algorithm, we had to create a drive control system that determines robot motion based on GPS coordinates using our positioning sensors.

This system uses two distinct stages of motion for each given path. First, the robot should make an in-place turn using full differential velocity to "zero the heading" or get it initially pointing towards the target point. Next, the robot actually begins its distanced-based PID control loop in which we use the remaining distance to target to create an initial straight-line velocity, but then we use heading error to create a differential correction on error to maintain heading correct for any sort of slippage.

Positioning

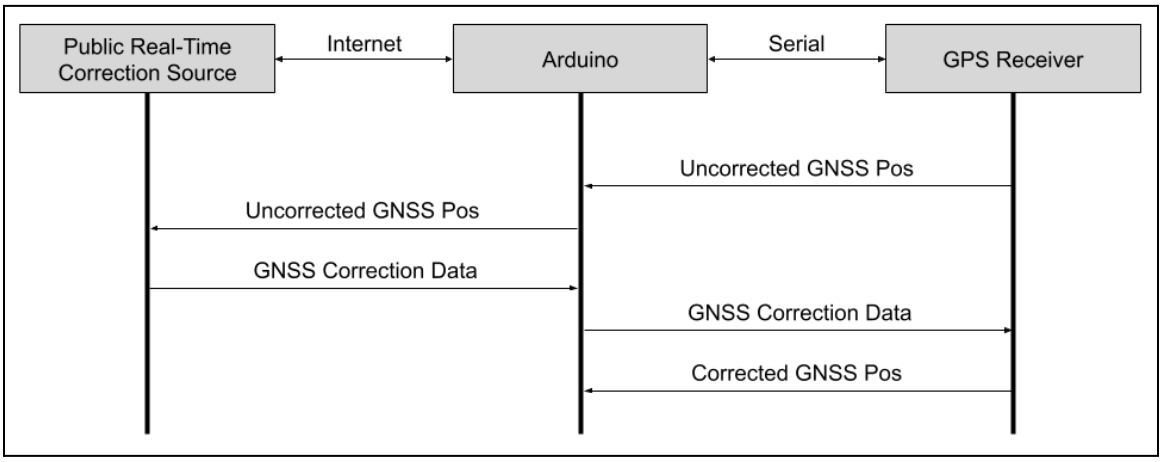


Figure 4-8: GNSS Positioning Communication Diagram

For our positioning needs, we designed a system to use GPS for localization. However, due to the small precision and accuracy required for navigating residential driveways and neighborhoods, we had a requirement for +/- 10 cm localization. As a result, we created a plan to use RTK-based differential GNSS, which requires us to use a public GPS correction source (in Iowa, we have the luxury of the DOT-provided Iowa Real-Time Network) in order to get centimeter-level accuracy. To make this happen, we need to send positioning data from the GPS receiver to our main controller such that it can create a request for correction data over the internet. We can then use the response to correct our GPS position for factors such as atmospheric and weather-related distortion.

Collision Detection/Avoidance

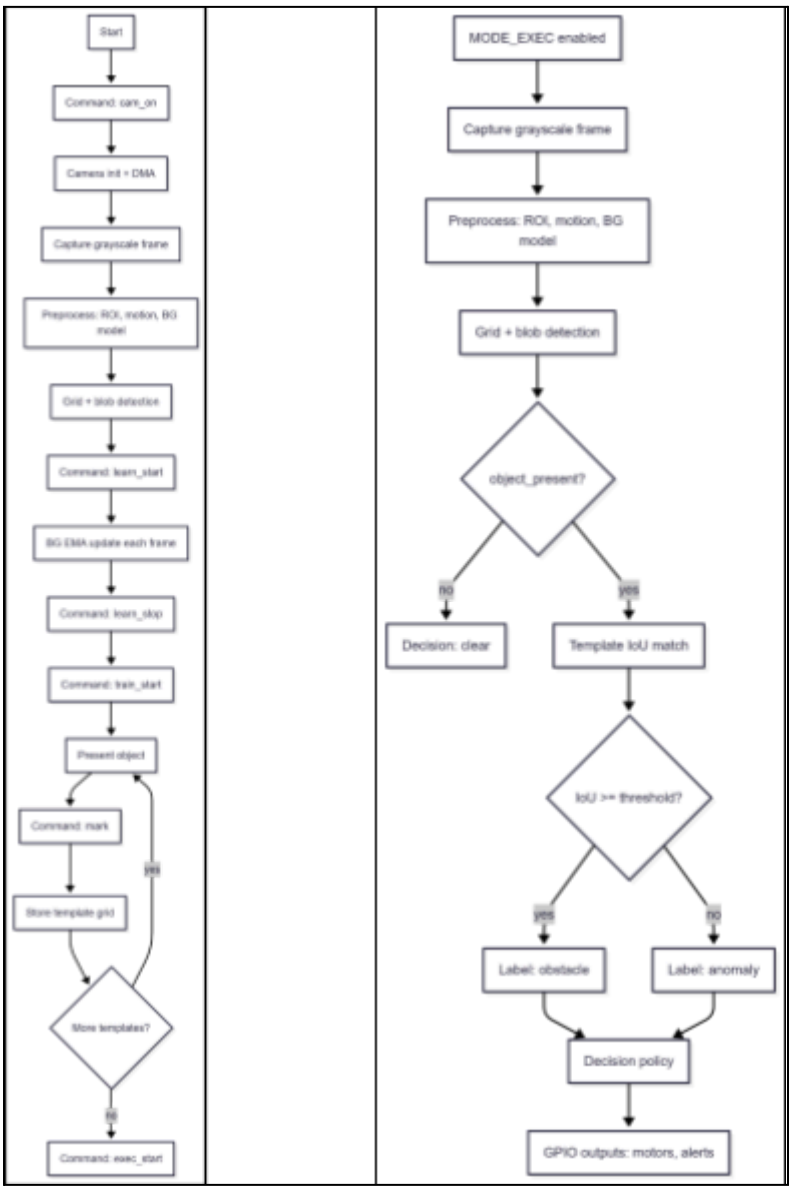


Figure 4-9: Computer Vision Training and Detection Flow Charts

Training starts by enabling the camera and running the standard preprocessing stack (ROI crop, motion energy, and optional background model) to build a binary grid representation of the scene. Background learning is intentionally separated so the EMA model converges on an empty scene before any templates are stored. Once `MODE_TRAIN` is set, each `mark` captures the current grid (post-threshold and blob detection) as a template; these templates are stored at grid resolution, so they represent coarse shape and position rather than pixel-accurate appearance. Repeating marks while alternating objects increases intra-class coverage and reduces temporal bias in the template set. The training phase ends by switching to `MODE_EXEC`, where the template buffer becomes the reference set for IoU matching.

Hopper Gate

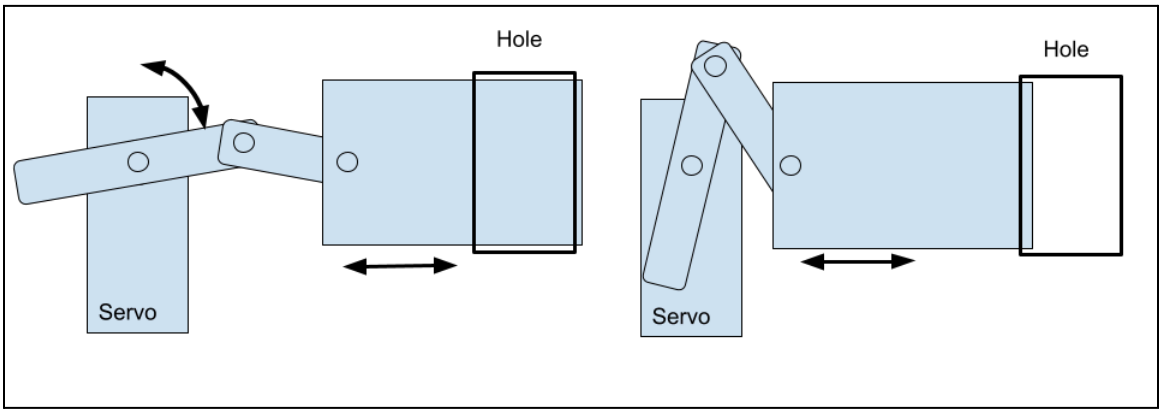


Figure 4-10: Servo Controlled Gate Linkage

In order to control when our robot spreads salt, we need to be able to control when the hopper gate is open and pours salt onto the spinner. We were modifying a Scotts Wizz Spreader, which originally actuated the hopper gate manually through a trigger. The trigger was attached to a bendable piece of plastic, directed through a track, that would open and close the gate. This meant that the gate was activated via linear motion, and was not electronically controlled.

We chose to implement a linkage controlled by a servo motor, to control the gate as seen in Figure 4-10. We chose a Smart Servo V2 for our design out of a desire to prevent the necessity of future replacements, as this servo has steel gears. The servo is controlled directly by the Arduino. The pieces for the linkage were then 3D printed and mounted to the servo horn. We used the original hole on the Scotts Wizz Spreader, and fabricated our design around these dimensions.

Spreader Spinner

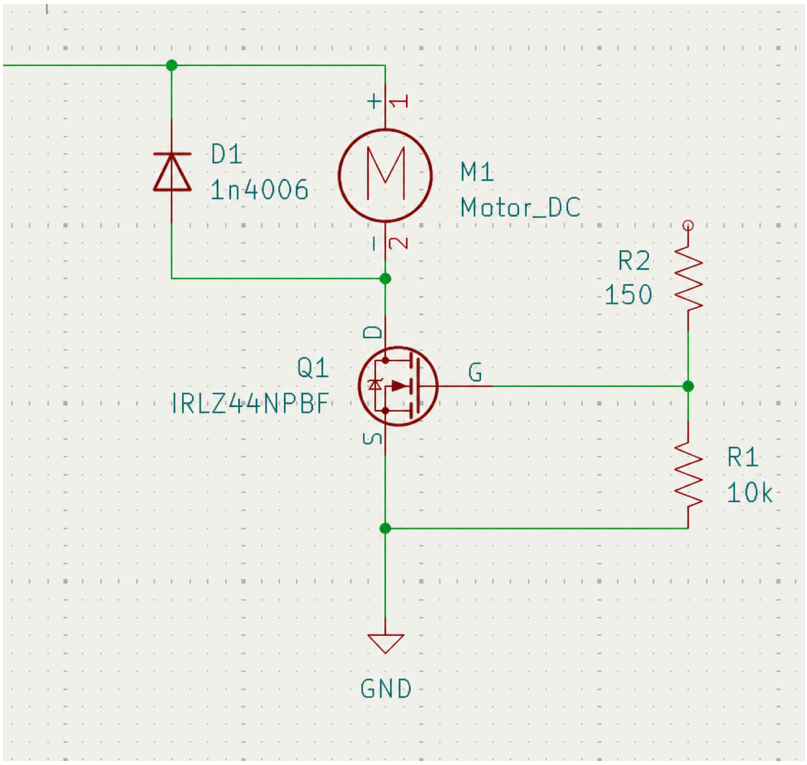


Figure 4-11: Spinner Motor Driver Circuit

Our salt spreading mechanism includes a spinner that is used to spread salt by ejecting it from the robot. The Scotts Wizz Spreader featured a battery operated 6V DC Motor to control the spinner. The motor was actuated by the trigger mechanism, which would cause the motor-battery circuit to be complete, and would allow power to flow to the motor.

Our design consisted of an N-channel MOSFET based driver circuit, to allow the Arduino to control the motor. The IRLZ44N was chosen specifically because it is a logic-level MOSFET; it can be fully opened by the ~5 V signal from an Arduino pin without needing a separate gate driver. A flyback diode D1 (1N4006) clamps the inductive voltage spike when the motor turns off.

Power Distribution

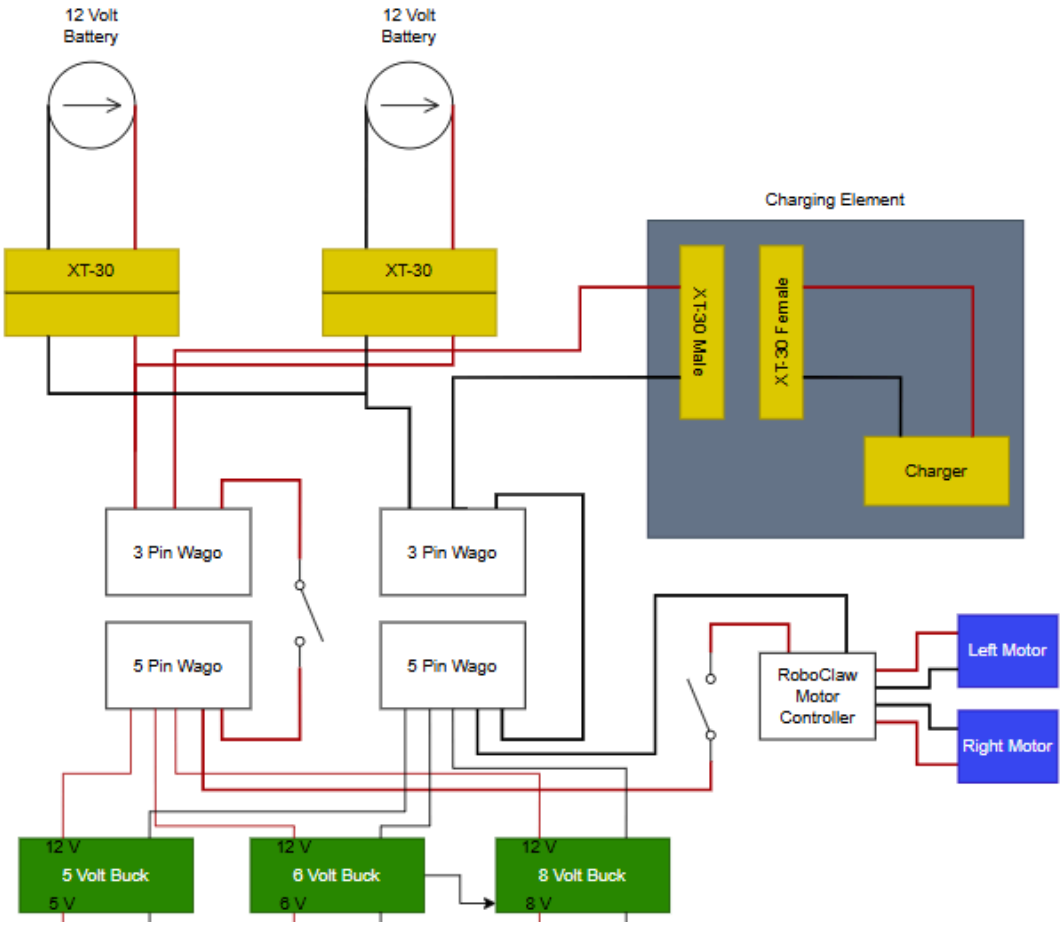


Figure 4-12: Power Distribution Diagram

The design uses two 12 volt lead-acid batteries in series to power the system. The 12 volt rail is stepped down through three buck converters, to give 5 volt, 6 volt, 8 volt, and 12 volt rails to power the robot. The 5 volt rail powers the GPS module and ESP32. The 6 volt rail powers the salt spreaders motor. The 8 volt powers the Arduino and the 12 volt powers the motors. The 12 volt rail also goes through a voltage divider down to 4.5 volts max to act as an ADC translation of the total battery charge for the Arduino.

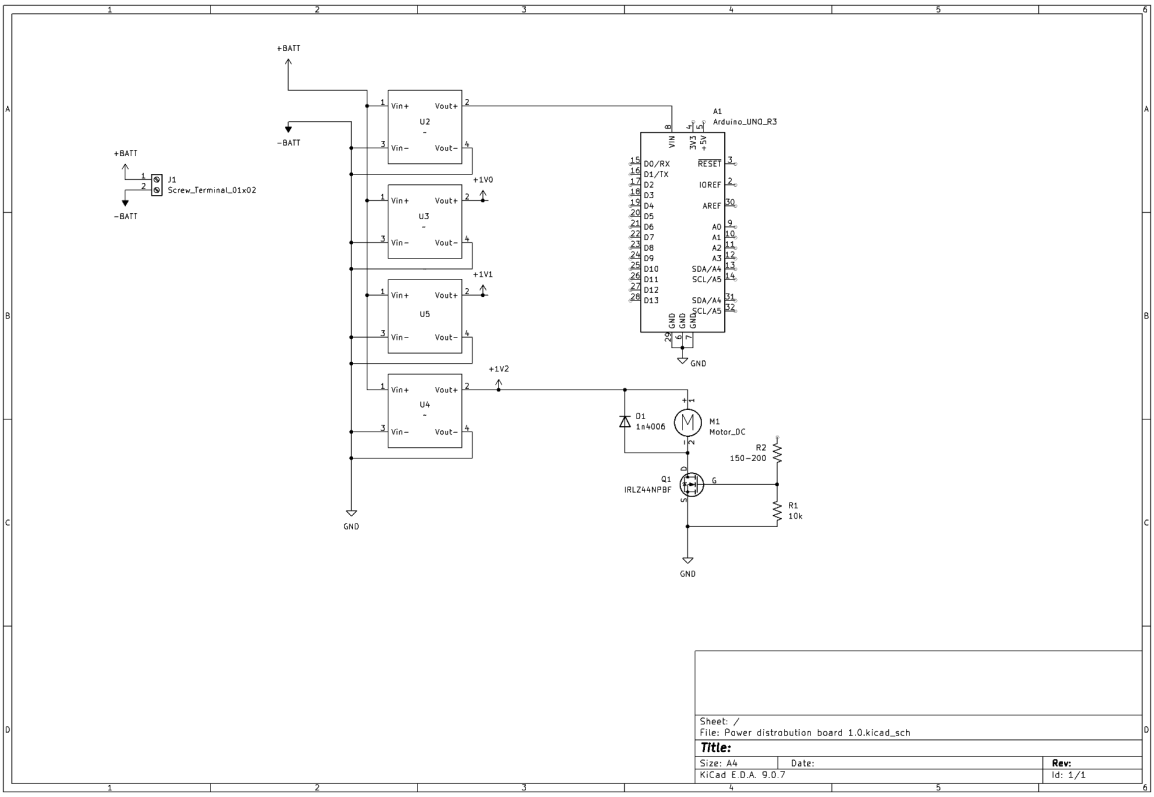


Figure 4-13: Power Distribution PCB Schematic

The preceding figure is our schematic for a power distribution PCB that would include spots for our bucks and daughter boards to connect, as well as our spreader spinner circuit built into the PCB. This would help our design and circuitry to be a lot cleaner.

Drivetrain

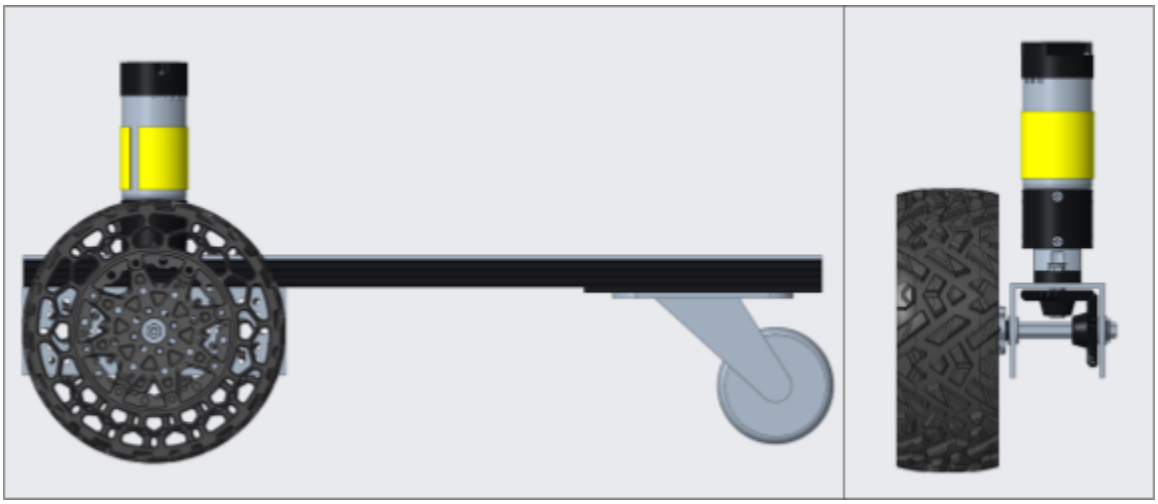


Figure 4-14: Drivetrain CAD Models

To meet our requirements we needed to design a drivetrain that could turn well, and had enough torque to drive the weight of the robot, even with large capacity batteries on board.

The drivetrain is run by two goBILDA Yellow Jacket brushed DC motors rated at 312 RPM driving two 144 MM tires. The back tires are caster wheels, allowing for better mobility. The design of two bevel gears and the vertical motor was decided to allow the robot to be more narrow, and give a way to enclose the motors without losing ground clearance.

Mobile Companion App

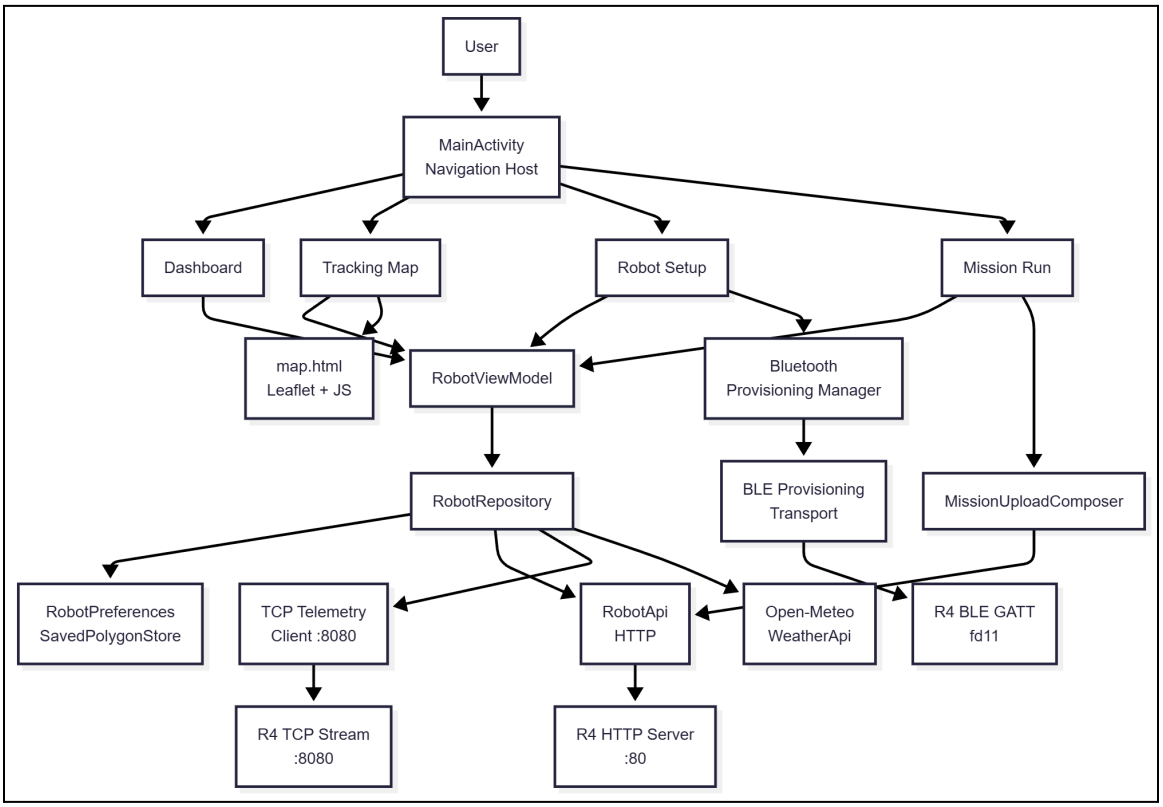


Figure 4-15: Mobile Companion App High Level Architecture Diagram

The preceding figure shows the designed high-level architecture of the mobile companion app. The app is to connect via bluetooth such that it can set up the WiFi connection for the robot and then start communicating with the webserver hosted on the Arduino. Using the webserver it must be able to upload robot preferences, establish a salting area, send high level commands, and display telemetry data received from the robot.

4.3.3 Functionality

The intended functionality of our solution can be described by a set of steps that are either accomplished by the user or the robot itself:

1. If not charged enough for a full run, charge the robot.
2. If not filled with salt to an adequate level as indicated by the app, add more salt to the reservoir.
3. Once there has been ice accumulation and most of the snow has been cleared, if needed, use the app to trigger deployment of the robot.
4. The robot will then exit its resting place and go to its path's starting point.
5. The robot will open its salt gate and salt spreading impeller.
6. The robot will continue along its path, avoiding obstacles or warning the user through the app if it cannot resolve the issue itself.
7. Upon completion of the path, it will return to its resting place.
8. The robot will notify the user if it needs to be charged or refilled with salt before its next run.

4.3.4 Areas of Challenge

There were several challenges encountered throughout the development of our project. The mechanical implementation of the robot grew to become an issue, with the large amount of mechanical complexity leading to long-increasing time-to-implementation and a large time drain on the project as a whole with our members not having as much mechanical design experience as they have in software development and electrical design. One major hurdle stemming from this was the strength of some of our prints. The 3D printed bevel gears were breaking at the layer line or shearing their teeth, often causing large pauses in software testing and slowing down general progress in the late semester. This led to a high amount of integration strain, stemming from the delays, but also from the large scope we set out to complete. Some other features from our design we didn't get to fully implement were waterproofing the system and adding a salt level detection sensor that would tell the user, app and robot that it was running low on the amount of salt it was carrying.

One of the greatest challenges we faced throughout this design process was the proper integration of software as the technical complexity and level of distribution (both physically across controllers and work division among team members) grew as the size of our code base grew. While team communication and more frequent peer-coding sessions alleviated some of this strain, this was an issue through the end of our time working on the project and was a key cause for much of our currently unmet requirements and features.

4.4 Technology Considerations

While most of the technologies we are using in our design are well established and have been proven to be functional, the primary technology tradeoffs that we have been having to consider are the cost-benefit tradeoffs due to our budget constraints. While there are highly accurate positioning systems available to us, we have had to reduce our expected accuracy as the best systems are far from affordable. This also applies to a lot of our collision avoidance needs; the high precision sensors or sensors such as advanced computer vision or Lidar have not come down in cost enough for them to be economically viable for us.

We have been combating this issue through smart implementation of lower-cost solutions through methods such as sensor fusion and lower-computational-need grid and graph-based algorithms.

Additionally, the usage of a distributed control network allows us to reduce the strain on any individual processor. This has its own challenges as it increases the integration strain and increases the amount of communication planning required.

The usage of 3D printing as our main construction method has produced its own set of strengths and weaknesses. While it benefits from low cost, no shipping times, and ease of design iteration, it has weaknesses in the form of physical strength, long print times for some parts, and the time required for designing our own parts.

5 Testing

5.1 Unit Testing

Due to the multifaceted nature of our project, we utilized a wide range of tools to test each component or subsystem that comprises the mobile robot application.

Hardware-wise, unit tests will be performed using physical tests such as the drivetrain's ability to drive (likely using a simplified version of the code during early development) and testing the power distribution system using a multimeter. Each power rail will have identified test points and expected voltage values and tolerances.

Software-wise, the unit tests will take many forms. Sensors with their own computational abilities, such as the GPS module, will be tested for accuracy and reliability independently of the whole system using the OEM's software tools. Similar tests will be conducted to evaluate the ESPCam's ability to detect approaching obstacles. Less complicated functions, such as the ability for the bump sensors to interrupt the code, will be tested in conjunction with our Arduino using the ArduinoUnit library through more traditional unit test structures (asserts and the Serial Monitor). All core logic (calculations and data processing) will be structured such that it can be tested without hardware and integrated into a set of unit tests.

5.2 Interface Testing

Our interface testing will be primarily designed around the communication between sensors and controllers on the robot. For each protocol/connection, data will be transmitted from point to point. When possible, sample data will be sent, and an assertion will verify that it was transmitted and received correctly. When sample data is not possible (e.g., from the GPS unit) data will be verified to be the same as reported to the PC-based software manually. PWM for motor control will have to be tested during integration testing.

Two wireless tests are to be conducted: a test to ensure the main controller can connect to an RTCM server and a test to verify that two-way communication can be established with the mobile app.

5.3 Integration Testing

Our system involves multiple integrated systems, especially in our control system. Communication tests will verify whether each sensor or distributed processor can report its data to the main controller and produce the correct trigger or interrupt within the main control loop. All of these communications from the sensors will be run through their respective data processing, if needed, and assertions will check the validity of the pre- and post-processing data. All of these communication tests will need to be verified through monitoring/logging and manual checks, as the use of real sensor data would not allow for the expected values. However, assertions will be implemented to check for signals outside of acceptable ranges.

Tests will also verify that the main controller produces the correct physical response from each physical subsystem (drivetrain or salt spreading mechanism). These will have to be tested through visual inspection.

The electrical system is to be tested under load after integration of the other systems to verify the appropriate power draw and to assess voltage regulation. This will be tested using a multimeter under simulated loads on the robot, as measuring voltages and currents on a moving robot would be challenging.

5.4 System Testing

System testing will involve unit testing of each component according to requirements (e.g. does the GPS unit report <5cm accuracy), along with each integration test once the full system is created. These tests will be run concurrently whenever possible to stress-test our computational and power needs. Once these conditions are met, we will run tests on the main functionality of the mobile app, specifically using it to trigger a de-icing job at a known test site. We will monitor the combination of sensors, navigation, driving, obstacle avoidance, and salt distribution.

5.5 Regression Testing

To ensure software quality does not regress, we will set up a testing pipeline for all of our unit tests that can be automated.

For tests that cannot be automated, such as those involving physical components or real sensor data, we will rerun related tests according to a dependency map*. For example, if a more powerful controller were added, we would retest the power system. Similarly, if we changed our GPS data, we would retest the GPS unit and all downstream integration tests.

*Our dependency map would be a visual representation of how each piece of software or hardware relates to each other, as shown in a condensed, text-based form below.

GPS module -> 5V rail -> power distribution sub-system (test current draw)

GPS module -> I2C -> positioning code -> navigation code

5.6 Acceptance Testing

Our acceptance testing will primarily take the form of continuous testing throughout our design process, with an emphasis on ensuring that each requirement is addressed by at least one test. All relevant testing results will be reported to our advisor, and milestone tests will be presented live to our advisor. System testing, according to requirements, will be our ultimate acceptance test.

5.7 User Testing

For user testing, we plan to implement two main components. First, we will test the user interface of the mobile app by conducting an unguided user interface test with a sample of users of varying technology skill levels and ages. Without prior knowledge of the app, we would record the time, the number of touches required, and the accuracy of each test subject in accomplishing a task or locate a piece of information.

Next, we would run demonstrations of the full functionality and have independent observers grade the quality of salt spreading.

5.8 Results

GPS Accuracy Testing

In order to test whether we met our requirements for GPS accuracy, we ran a set of tests to see both the startup and long-term performance of our RTK GNSS localization system.

These tests were mostly run in overcast conditions (typically worse for GPS accuracy) and were run in locations where we could ensure at least a 30 degree vertical line of sight to the sky. These were also run less than 2 miles away from the Iowa DOT base station that is used for the Iowa Real-Time Network which we use to get GNSS correction data.

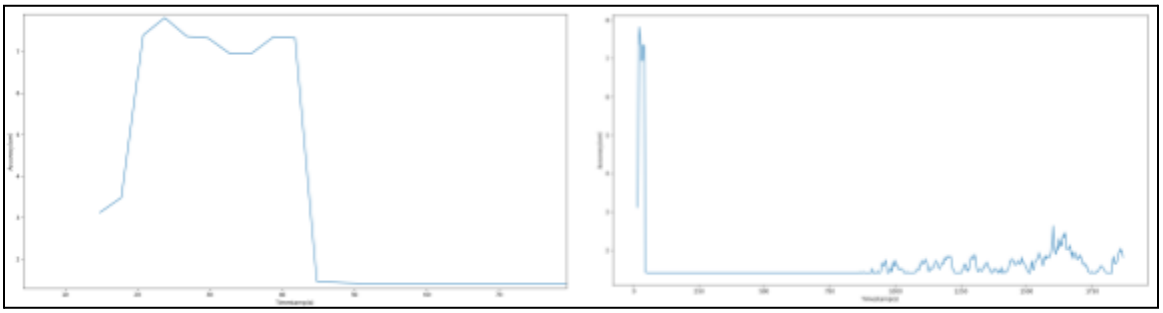


Figure 5-1: GPS Module Start Up Behavior Test Results

Our startup testing, shown in the figure above, was run to determine how long it takes our system to achieve an RTK fix and thus have stable, accurate positioning data. In our tests, we determined that we would reasonably expect it to require a minute before we can trust our positioning data, as our testing yielded a typical time of 40-50 seconds before achieving sub-inch accuracy.

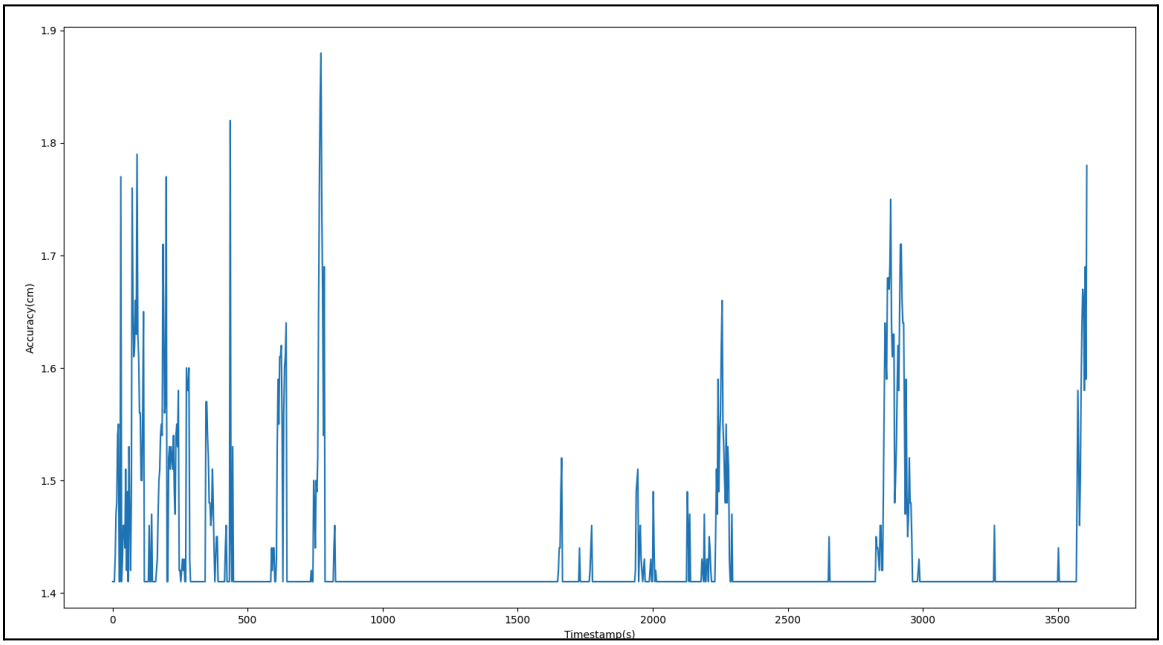


Figure 5-2: Static GPS Testing Extended Test Results

In our post-startup, extended GPS testing, we ran some static position tests of the GNSS positioning accuracy. These yielded very promising results with accuracy falling below 2 cm over the entire duration of our tests.

Our primary conclusions from this testing is that, at least in the conditions of our tests, we fall well below our 5 cm accuracy requirements as set forth in our early stages of the project. However, we do believe that extended testing would be necessary for further productization of the concept. These would include GPS accuracy testing in snowy conditions, testing near buildings or other sources of GPS multi-pathing, and testing while in locations further away from their real-time correction sources.

Equirectangular Approximization Testing

For the purposes of our solution, we had to prioritize runtime over very fine accuracy for our calculations of GPS-coordinate-based point to point distance and heading calculations. As a result, we implemented equirectangular approximation which projects coordinates onto a localized flat earth model, introducing some error into the calculation. Due to this, we ran tests on this algorithm in terms of produced distances and headings along with the run time for the calculations.

We ran these tests with four different distance ranges that would be relevant to our application of residential driveways and sidewalks. These were following a spread of north, east, and diagonal movements at different latitudes since latitude changes are what affect flat earth vs. round earth distance models the most:

P1	P2	Expected Distance (m)	Expected Heading (°)	Calculated Distance (m)	Calculated Heading (°)	Run Time (us)
Category 1: 100 Meter Range (~50m to ~150m)						
0.0000000, 0.0000000	0.0010000, 0.0000000	111.1949266	0.0000000	111.1949266	0.0000000	245
0.0000000, 0.0000000	0.0000000, 0.0010000	111.1949266	90.0000000	111.1949266	90.0000000	330
42.0266000, -93.6465000	42.0276000, -93.6465000	111.1949266	0.0000000	111.1949266	0.0000000	654
42.0266000, -93.6465000	42.0266000, -93.6455000	82.5996452	89.9996614	82.5993829	89.9996653	718
0.0000000, 0.0000000	0.0008000, 0.0008000	125.8000049	44.9998000	125.8026987	45.0000000	744
Category 2: 10 Meter Range (~50 to ~15m)						
0.0000000, 0.0000000	0.0001000, 0.0000000	11.1194927	0.0000000	11.1194927	0.0000000	536
0.0000000, 0.0000000	0.0000000, 0.0001000	11.1194927	90.0000000	11.1194927	90.0000000	325
42.0266000, -93.6465000	42.0267000, -93.6465000	11.1194927	0.0000000	11.1194927	0.0000000	654
42.0266000, -93.6465000	42.0266000, -93.6464000	8.2599645	89.9999661	8.2599383	89.9999665	717
0.0000000, 0.0000000	0.0000800, 0.0000800	12.5800005	44.9999800	12.5802699	45.0000000	748

Table 5-3: Equirectangular Approximization Testing Results Categories 1 & 2

P1	P2	Expected Distance (m)	Expected Heading (°)	Calculated Distance (m)	Calculated Heading (°)	Run Time (us)
Category 3: 1 Meter Range (~0.5m to ~1.5m)						
0.0000000, 0.0000000	0.0000100, 0.0000000	1.1119493	0.0000000	1.1119493	0.0000000	534
0.0000000, 0.0000000	0.0000000, 0.0000100	1.1119493	90.0000000	1.1119493	90.0000000	332
42.0266000, -93.6465000	42.0266100, -93.6465000	1.1119493	0.0000000	1.1119493	0.0000000	650
42.0266000, -93.6465000	42.0266000, -93.6464900	0.8259965	89.9999966	0.8259938	89.9999967	714
0.0000000, 0.0000000	0.0000080, 0.0000080	1.2580000	44.9999980	1.2580270	45.0000000	745
Category 4: 10 Centimeter Range (~5cm to ~15cm)						
0.0000000, 0.0000000	0.0000010, 0.0000000	0.1111949	0.0000000	0.1111949	0.0000000	503
0.0000000, 0.0000000	0.0000000, 0.0000010	0.1111949	90.0000000	0.1111949	90.0000000	287
42.0266000, -93.6465000	42.0266010, -93.6465000	0.1111949	0.0000000	0.1111949	0.0000000	655
42.0266000, -93.6465000	42.0266000, -93.6464990	0.0825996	89.9999997	0.0825994	89.9999998	687
0.0000000, 0.0000000	0.00000008, 0.00000008	0.1258000	44.9999998	0.0125803	45.0000000	524

Table 5-4: Equirectangular Approximization Testing Results Categories 3 & 4

Through this testing, we were able to confirm that in the distances that are relevant for our application, flat-earth projection algorithms are beyond adequate. In the testing, the worst distance error was a little less than 2.7 mm and the worst heading error was 0.72 arcseconds or 0.0002 degrees, which is significantly less than the noise of our sensor readings. Meanwhile, we were able to maintain less than a millisecond for running these calculations in all given cases.

We see no reason to believe that this would cause us any issues for users or our higher level algorithms, no matter the location on Earth.

Computer Vision Obstacle Detection Testing

In order to test our computer vision obstacle detection, we ran a set of tests with different sets of objects and trained models to check for issues such as delayed recognition, missed recognitions, and false positives.

Run ID	Objects trained	Presentation protocol	Immediate recog.	Delayed recog.	Misses	False-positive time	Notes
R-2026-05-04-A	10	3x for ~3s each, alternating	9	1	2	10-15 s / 300 s	10th object required extra time; 2 misses before recognition.
R-SYN-01	12	3x for 2-3s each, alternating	11	1	0	0-3 s / 300 s	Illustrative baseline with stable lighting.
R-SYN-02	10	3x for ~3s each, alternating	8	2	1	5-8 s / 300 s	Illustrative run with background motion.
R-SYN-03	8	4x for ~2s each, alternating	8	0	0	0 s / 240 s	Illustrative best-case after calibration.

Table 5-5: CV Obstacle Detection Testing Results

In our testing, we had four different runs with different training. In the testing, three of our trained models had either misses, delayed recognitions, false positives, or a combination. However, our final run was able to identify all 8 trained objects immediately and had no false positives in the 240 seconds of run time. We identified this as the best-case solution, but we recognize a need to continue improving this and further confirm the effectiveness of the solution.

We concluded from these tests that this subsystem has much room for improvement and that potential improvements for this system would include a lighting system on the robot, more model training, and other quality of life improvements like a stereo camera system which would allow for distance measurements to improve size approximation.

6 Implementation

6.1 Current State Of Implementation

Overview - Prototype 2.0



Figure 6-1: Image of Prototype 2.0

The current state of our project as the semester comes to a close culminates in the form of Prototype 2.0 aka “SALT-E”.

This prototype physically facilitates most of the requirements for the project with a fully functional drivetrain, all planned sensors mounted, a digitally controlled salt spreading system, and all processors and co-processors mounted and wired.

The electrical system, as seen later in this section, provides all needed voltage, current, and capacity needs for our subsystems.

Software-wise, this prototype has given us the ability to test all subsystems in isolation, with most of the planned integration testing completed. However, the software subsystems have not been fully integrated together for full system testing. The software is capable of detecting obstacles using computer vision with a high rate of accuracy, drive from its current location to a given GPS coordinate, actuate our salt spreading system, and generate a salt spreading path with an intuitive geometry-based map UI.

Control



Figure 6-2: Created Path From Path Finding Algorithm

Designed to autonomously create a path to salt for the robot while accounting for capacity and level of available salt and power. Keeping the controllers memory constraints in mind to avoid having the Arduino crash because it ran out of memory or heap fragmentation the following was implemented:

1. Flattening the 2D array grid into a 1D array of 8 bits
2. Using a queue to evaluate nodes without fragmenting the heap
3. Removing sqrt and pow when routing and instead use octile distance calculations when doing integer math

For coverage the following were utilized:

1. Zone sweeping where when it checks for the first or next path it takes the longest continuous unsalted path that's physically reachable that the robot can fit through

2. As the robot traverses through the path it subtracts its current salt and battery capacity from how much that cell costs for salt and battery levels

3. If any of the resources drop below a specific level it will return to base and refill then return to its last position using A star, and the system will terminate if a zone requires more transit energy than the batteries maximum capacity

To prevent the algorithm from generating a path that zig-zagging a filer was utilized:

1. Line-of-Sight (LOS) Verification using Bresenham's Line Algorithm to cast a mathematical ray between distant nodes on the generated path, verifying that the direct vector does not intersect with the hard boundaries of the concrete.

2. Waypoint Minimization, The algorithm scans ahead and identifies the furthest possible safe node, erasing the intermediary zig-zags.

3. Compressing thousands of individual cell calculations into a highly minimal list of corner-only GPS coordinates. The physical robot receives only the critical turning points, allowing it to drive in smooth, continuous, straight lines down angled corridors.

Prior to hardware deployment, the navigation logic was validated using a custom C++ Ground Control Station (GCS) using SFML graphics library. The GCS parses satellite ground-truth maps, overlays the algorithm's abstract grid view, and visually simulates the robot's state machine. Which allowed for testing without the robot or microcontroller.

Positioning

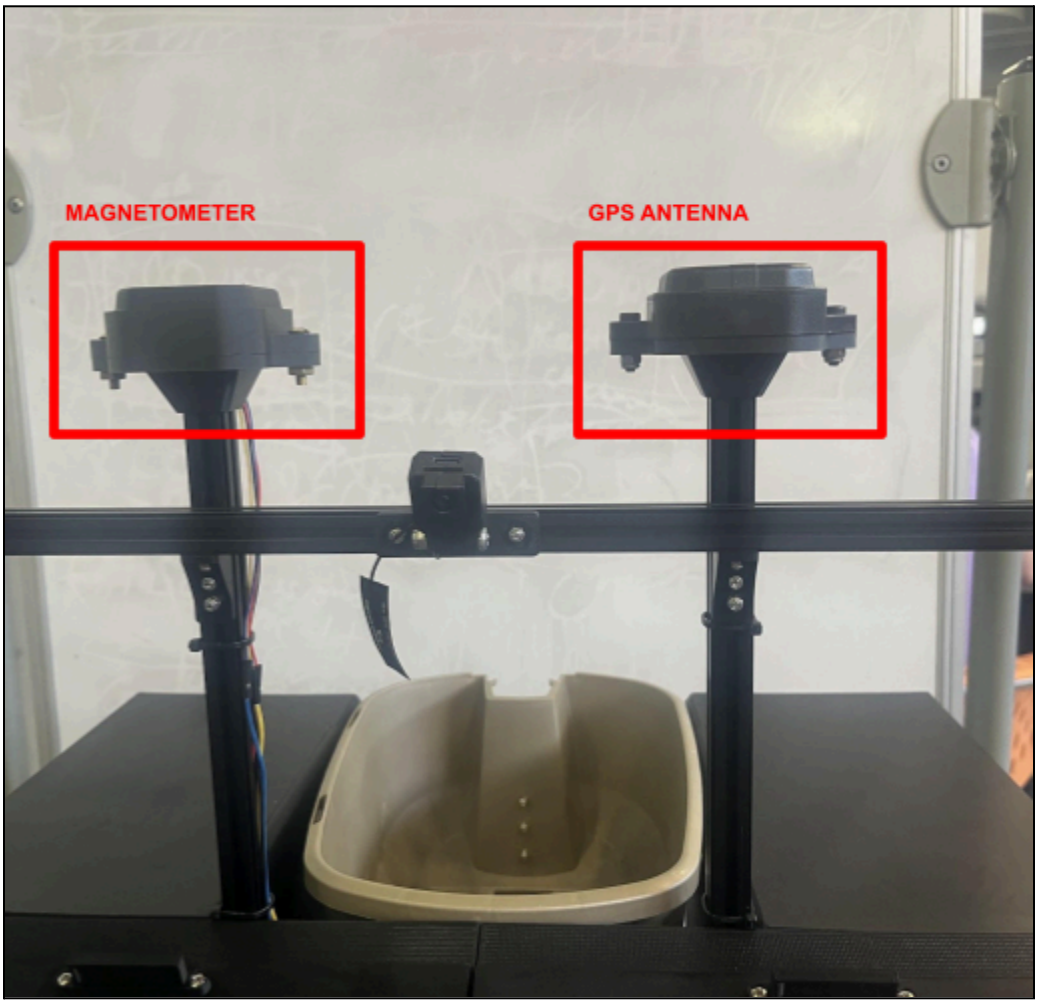


Figure 6-3: GPS Antenna and Magnetometer Placement

Using the main sensor strut, we were able to mount both our GPS and magnetometer in ideal locations on the robot. By mounting the GPS antenna at the top of the robot, we were able to prevent any of the rest of the robot from interfering with signals. The magnetometer benefits from being pulled further away from ferrous components and the magnetic fields produced by the motors. However, extra precautions were also taken later on including using all brass fasteners near it as they are non-ferrous and will not cause hard-iron distortion.

We were able to implement RTK differential GPS localization using public GNSS correction sources to achieve sub inch accuracy levels during ideal conditions.

For the magnetometer, we were at-best able to achieve +/- 3 degree accuracy due to not having the time needed to implement more complex soft-iron calibration techniques, but we were able to get hard-iron calibration.

Collision Detection / Avoidance

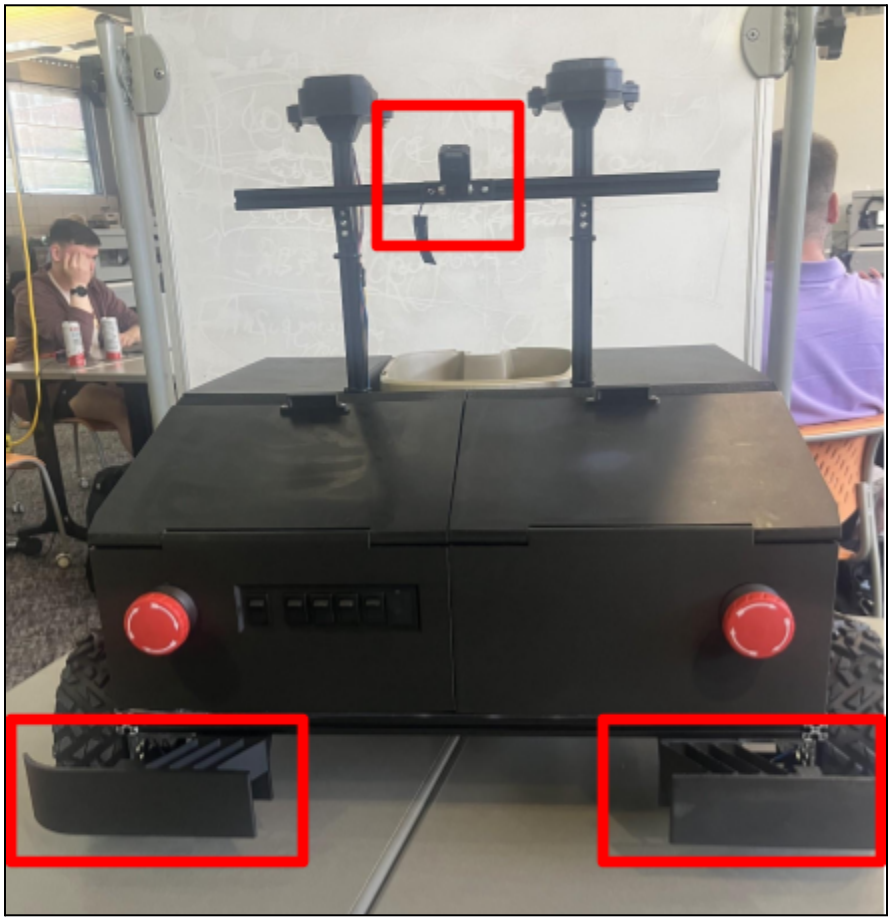


Figure 6-4: collision detection sensors mounting locations

We were able to mount and implement our collision detection and avoidance sensors onto the robot through a mixture of physical and electrical implementation. We were able to use the sensor strut and an adjustable angle camera mount to put our computer vision camera in a position and maximize visibility. The front bumpers were implemented using compliant 3D printed bumpers which actuate microswitches behind it.

In software, our computer vision uses masking to detect whether we have an object in one of three zones (left, center, right) and is able to communicate with digital pins to the main controller of the location of the potential obstacle.

Our two bumper sensors, despite their fancy mechanical design, are implemented using very simple microswitches wired to digital pins on the Arduino.

On the main controller, an object is created that polls each of these digital pins and throws an error that will stop the system in a safe manner.

Salt Spreader Hopper Gate

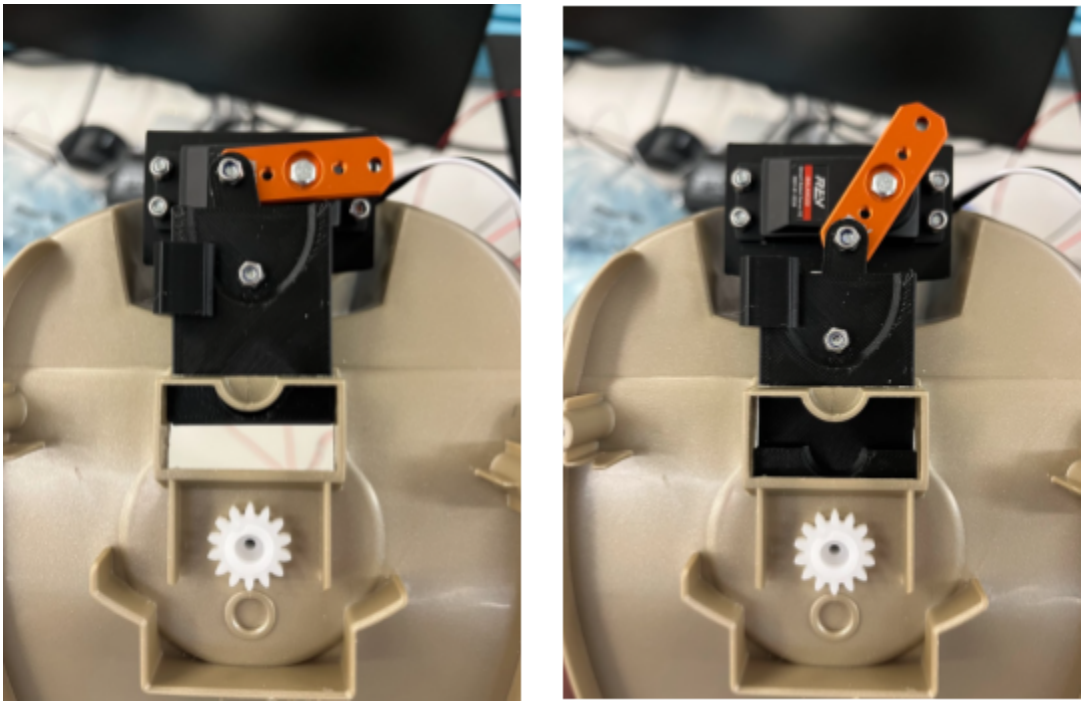


Figure 6-5: Salt Spreader Gate Implementation

We implemented the designed servo-actuated linkage onto our salt hopper. The attachments to the servo horn were 3D printed from PLA. We included a mechanical hard-stop to prevent any software errors from causing the linkage to fall out of the gate channel. This system has been confirmed to produce as much of an opening as the spreader did when using its original, manually operated controls.

This is powered using a dedicated 6V rail from the power distribution system and given PWM control signals from the main controller which has a dedicated salt spreader object created for control abstraction.

Salt Spreader Spinner Driver Circuit

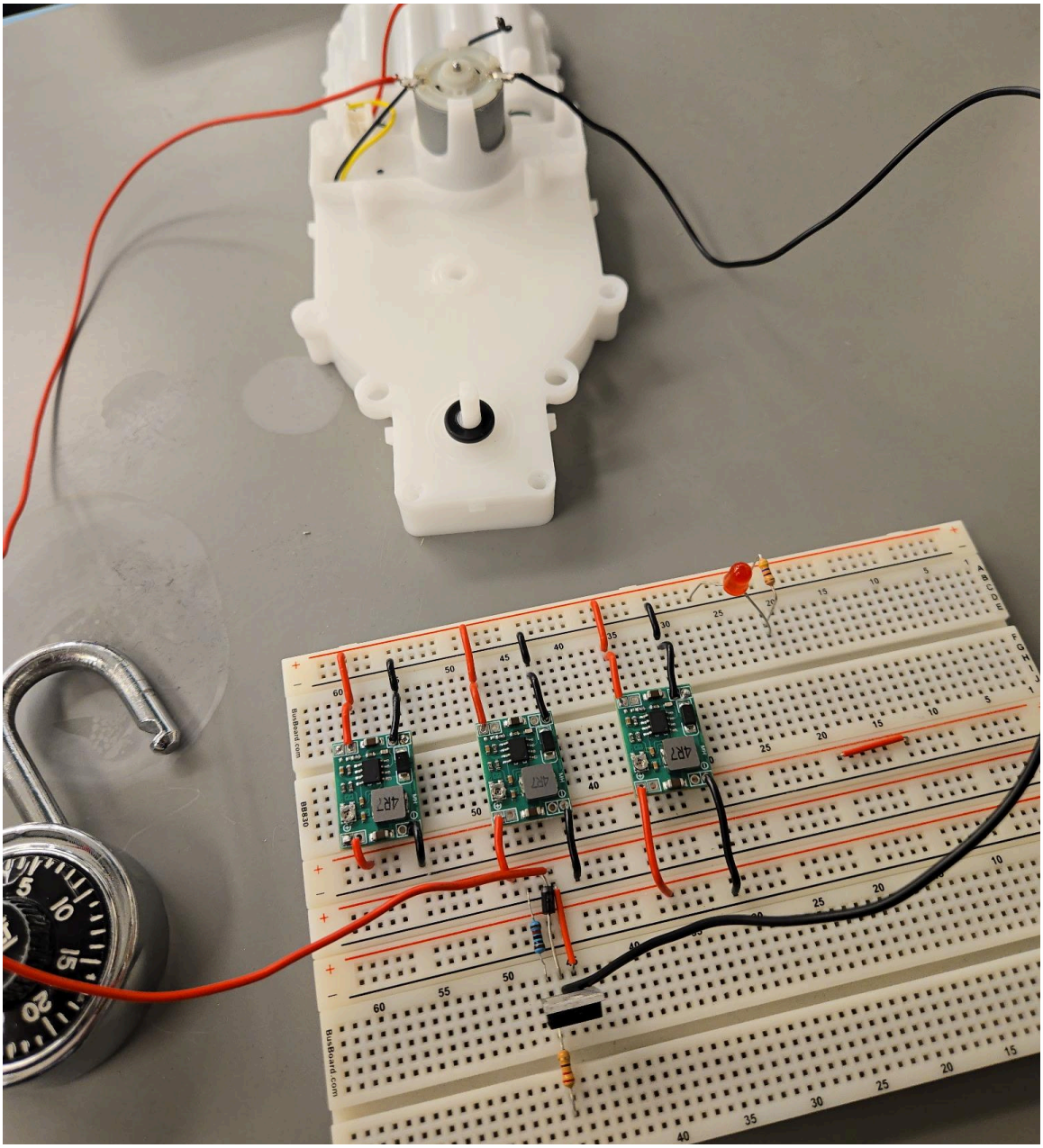


Figure 6-6: Servo Driver Circuit

This is the implementation of the driver circuit that allows for software control of the salt spinner. The circuit allows us to pull current from the 6V rail while giving a command from the 5V logic of Arduino. This is tied into the same salt spreader object as the salt gate in software allowing for coordinated control.

Power Distribution

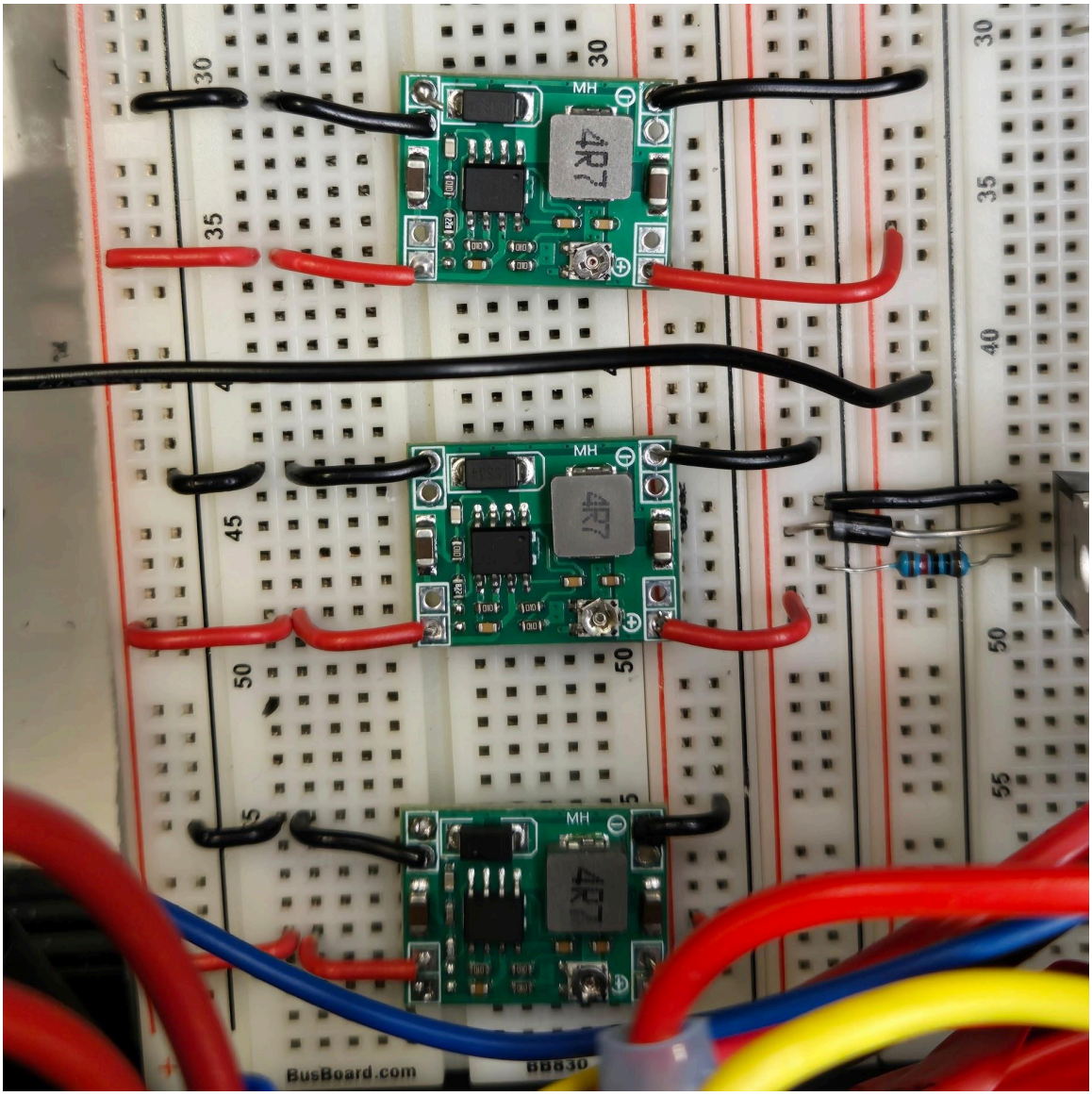


Figure 6-7: Power Distribution Implementation

The figure above shows the three buck converters of the power distribution for the robot. The rails are 5, 6, and 8 volts going down.

Additionally, there is a main power switch and two additional switches that can be used to cut power to the drivetrain and salt spreading subsystems to prevent inadvertent motion during testing or maintenance, keeping the operator safe.

Drivetrain

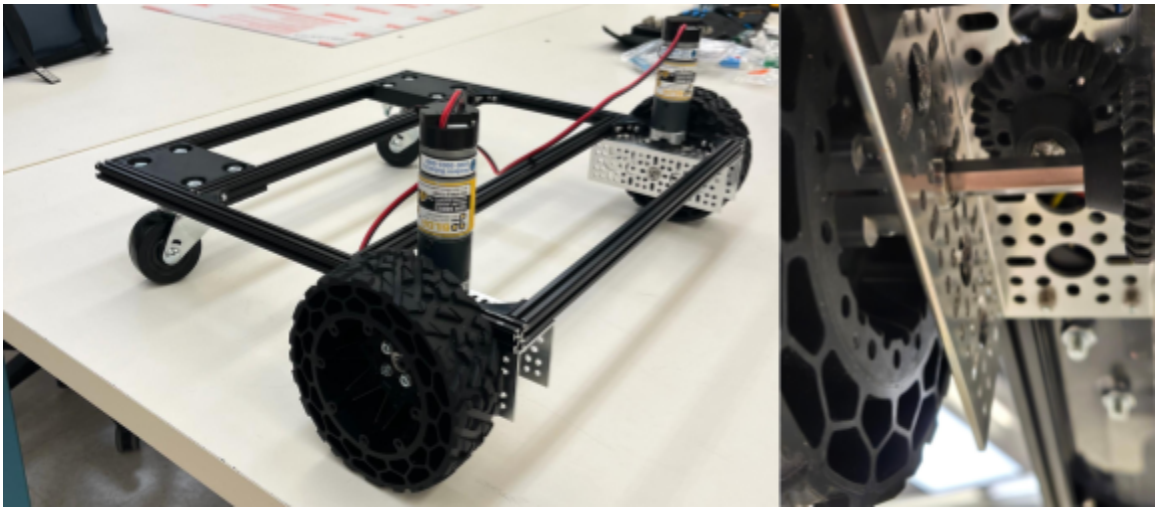


Figure 6-8: Drivetrain Implementation

We constructed our drivetrain using 15 mm aluminum extrusion and 3D printed components. The usage of x-extrusion for the main structure both provided the needed rigidity for the weight of the system and flexible mounting options as we moved forward with construction.

The actual drive system was implemented using off-the-shelf components and 3D printed bevel gears. The drivetrain was wired with an off the shelf motor controller and is using encoder-based closed-loop control for both velocity control and distance tracking when in the absence of accurate GPS-based localization.

Mobile Companion App

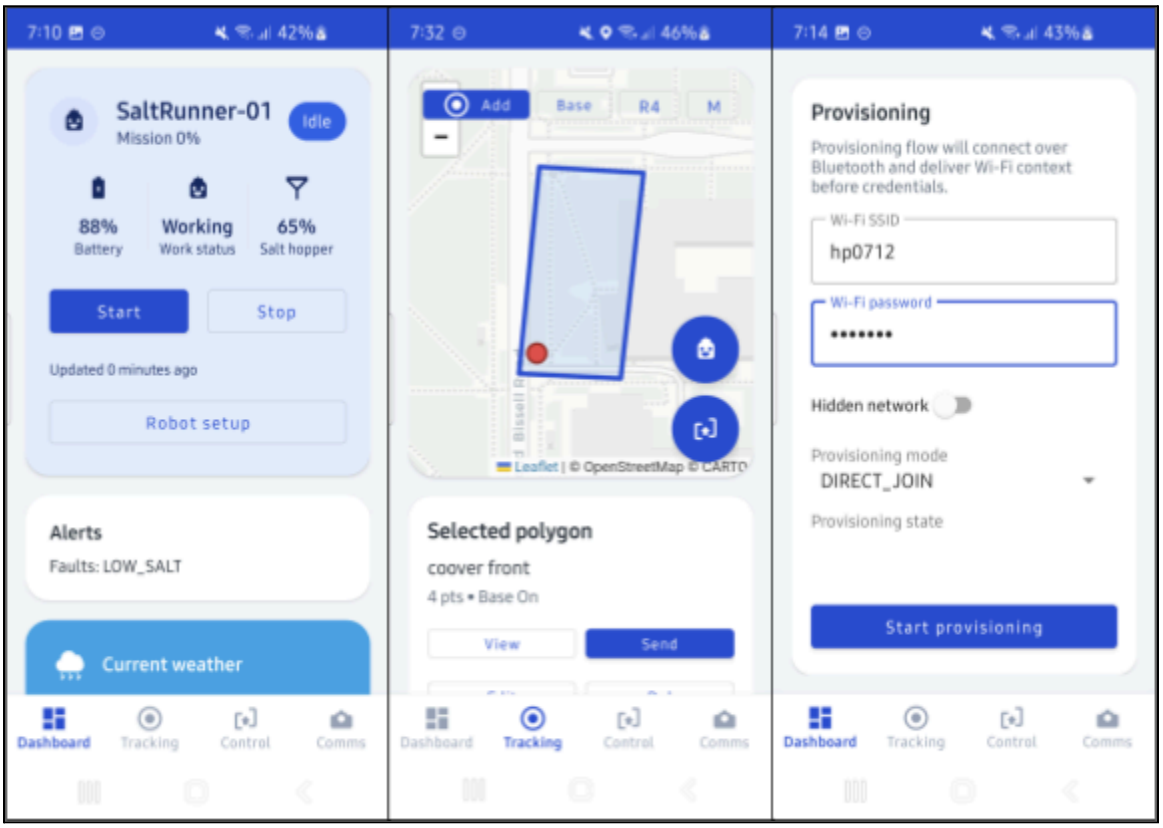


Figure 6-9: Screenshots from Mobile Companion App

The mobile companion app can currently connect via bluetooth to set up wifi communications. Once connected you can draw out a map of the salt area and send it to the bot.

We also developed a developed testing page that we could use to perform simple testing with our physical actuators such as the drive motors and salt spreader gate and spinner.

6.2. Design Analysis

The design works well, driving and spreading salt like it's supposed to. It drives at a moderate speed, and the app works well, showing the camera, allowing control of the robot, and the salt is spread evenly throughout the path. It works well as it can out perform what we originally set out to do in most scenarios, and was able to perform multiple test runs.

Speed is not as optimal as it could be, peaking at .7 m/s, while a good speed, the robot could probably move a little quicker. This is primarily due to the weight of the batteries on the robot.

On the software side, there is much that can be improved from further integration of planned sensors and optimization of our more resource-heavy algorithms like the pathfinding algorithm. A better analysis from the beginning of the resource needs for our planned algorithms would have allowed us to make better hardware choices, as the software complexity ballooned as the semesters progressed. Better scoping of the project would have likely mitigated some of these issues as well.

On the mechanical side, the drivetrain gearing was one of the systems that posed the most trouble after the initial design stage as we had three separate instances of the 3D printed gears being damaged. At first, we were having the gear body separate from the axle-mating body at the printing layer lines. Additionally, we had issues with the gears, as they are held in place on the axle with friction, sliding under large loads. This would shrink the contact patch, either increasing gear backlash, or causing catastrophic failure of the gear.

7 Ethics and Professional Responsibility

The development of our autonomous salt-spreading robot introduces a range of ethical considerations tied to safety, environmental impact, and societal responsibility. As an autonomous system operating in public and residential spaces, our design must prioritize the protection of users and bystanders through reliable collision avoidance, fail-safe mechanisms, and transparent system behavior. Additionally, because improper salt distribution can negatively affect vegetation, water systems, and local ecosystems, our team is committed to minimizing environmental harm through controlled and efficient spreading strategies. Beyond technical concerns, we also recognize broader societal implications, including accessibility for mobility-limited users.

7.1 Areas of Professional Responsibility/Codes of Ethics

Area of Responsibility	Definition	Relevant IEEE Code	How Our Team Has Addressed the Area
Work Competence	Professional and quality work	"Seek and accept honest criticism of technical work..." (Item 5) "To maintain and improve our technical competence." (Item 6)	We are choosing to iterate over many prototypes and validating the systems
Financial Responsibility	Create a financially reasonable product and save costs where applicable	"To be honest and realistic in stating claims or estimates based on available data." (Item 5)	One of our primary decision factors in many choices was economically based.
Communication Honesty	Be honest with our claims, and offer full transparency on our project	(Item 5)	We are openly communicating issues we run into and fixes that may help
Health, Safety and Well-Being	Ensuring the safety of users and their environments	"To hold paramount the safety, health, and welfare of the public... and to disclose promptly factors that might endanger the public or the environment" (Item 1)	We added bump sensors, object based detection, and emergency stop features
Sustainability	Minimize the negative environmental impact	"... or the environment" (Item 1)	We optimized battery usage for time and power usage, and plan to salt the minimum to help the environment
Social Responsibility	The product should benefit society and treat all fairly	(Item 1)	Our product will be useful to home and business owners

We are performing well both in financial and social responsibility. Our team evaluates numerous options, with affordability being one of the key considerations when selecting a part. Without including work hours, we have managed to keep our design more financially plausible.

Our sustainability is an area that needs improvement and re-examination. We are actively working on making a design that works as more of a proof-of-concept, and we haven't devised a more efficient salt distribution method that is eco-friendly.

7.2 Four Principles

Broader Context	Beneficence	Nonmalificence	Respect for Autonomy	Justice
Public Safety and individual users	Our primary purpose is to aid users in de-icing their driveways and sidewalks.	We are using object based detection and bump sensors, as well as a low drive speed to prevent harm	The robot will follow users directions and remain in gps region	Reliable quality salt distribution for owners
Social, global and cultural	We are trying to create a reliable method of de-icing for those without the means.	Avoiding unreliable, expensive designs or anything that may be stigmatized or offensive	Control of the device over the app allows users to customize use, and the basic design is unoffensive	Ensures consistent performance regardless of driveway type, neighborhood or user.
Environmental	By using parts that an owner can order and install, it promotes repairing vs replacing a device. Ability to optimize sale usage reduces run-off	Attempting to mitigate salt use to preserve the environment, but poor tuning could lead to excess use.	Users can choose eco-friendly salt alternatives and adjust minimum-spread settings.	Environmental impact considered across different regions
Economic	Provides a solution for people not capable of using traditional de-icing methods.	Minimizes manufacturing waste where feasible and promotes repairing.	Users can integrate with different regional salt types or standards.	Strives for cost fairness by using standard components and avoiding proprietary lock-ins.

Individual Users - Beneficence

Our robots' main benefit is reducing injury risk and providing a solution for the elderly or physically impaired. To ensure this we focused on safe autonomous navigation, and an even distribution of salt.

Environmental - Nonmaleficence

If improperly set up, our robot could provide increased salt usage and environmental damage.

7.3 Virtues

Team Virtues

Our team has defined the following virtues as important to us.

- **Communication** – Due to our larger team meeting in groups, communication between those groups is important for our team to succeed.
- **Time Management** – Our team has had a shorter schedule then the others, leading us to needing to be more efficient with our time. We have attempted to do so, but will need to put more focusing into this next semester.
- **Perseverance** – Our original project was ended, and we had to change lanes quickly. Through this we attempted to catch up and find stable footing, while also being diligent about the choices we made.

Dylan Bunting

Virtue Demonstrated: A virtue I have demonstrated through my senior design work is commitment to quality. Throughout the project I have been consistent in putting quality into the work I have done, from the design of the power distribution, to the hardware work I did on the robot.

Virtue Not Yet Demonstrated: While displaying the virtue of making clear and thorough documentation towards the end of the semester, during the first semester and early in the second, I was not doing a great job of making this documentation as it came along, instead waiting until it all sort of compounded into quite a bit of labor to catch up. This is important because if anyone else picks up on the project or takes over my section, they need to know what I have done and how to also work on it.

Hoyt Dzwoniarski

Virtue Demonstrated: I have been able to demonstrate the virtue of time management in my work throughout this project. This is crucial to my performance throughout the semester because it allows me to work on a schedule which maximizes productivity. I was able to have consistent and regular time that I would work on the project each week.

Virtue Not Yet Demonstrated: This semester I haven't demonstrated the virtue of consistency. Due to variances in workload, and the state of the project, my work per week has varied throughout the semester. In order to improve on this in the future it will be important for me to be more aware of what my schedule is and how this will affect my availability for upcoming weeks.

Issac Blandin

Virtue Demonstrated: This semester, I have been able to demonstrate general organizational skills and leadership along with building better testing skillsets. I feel that resiliency stood out this semester with working under high pressure due to the overinflated scope of our project and the intensity of the software requirements.

Virtue Not Yet Demonstrated: Some virtues that I struggled with this semester was time management and timeliness at meetings.

Mikhail McDade

Virtue Demonstrated - Collaboration : I was able to effectively help other members and aid in the process of the computational aspects between EE and CprE as well as get work in before the deadline.

Virtue Not Yet Demonstrated: Things that I have not demonstrated yet are better time management and keeping up with information.

Shobhit Singh

Virtue Demonstrated: I demonstrated responsibility this semester. I was always able to accomplish my tasks on time and my progress helped to move our group forward.

Virtue Not Yet Demonstrated: There were times this semester where I struggled to communicate all of my progress to the group.

Thomas Huedepohl

Virtue Demonstrated: I demonstrated adaptability by adjusting my approach when project requirements shifted or when initial designs didn't work as expected. I was willing to scrap and rework solutions rather than force something that wasn't fitting the team's needs.

Virtue Not Yet Demonstrated: I didn't always have a strong grasp of what was happening on other parts of the project, particularly the work bridging EE and CprE. To improve, I plan on actively trying to admit when I don't have information I need and finding where to get it.

8 Conclusions

8.1 Summary of Progress

The current state of our project as the semester comes to a close culminates in the form of Prototype 2.0 aka “SALT-E”.

This prototype physically facilitates most of the requirements for the project with a fully functional drivetrain, all planned sensors mounted, a digitally controlled salt spreading system, and all processors and co-processors mounted and wired.

The electrical system, as seen later in this section, provides all needed voltage, current, and capacity needs for our subsystems.

Software-wise, this prototype has given us the ability to test all subsystems in isolation, with most of the planned integration testing completed. However, the software subsystems have not been fully integrated together for full system testing. The software is capable of detecting obstacles using computer vision with a high rate of accuracy, drive from its current location to a given GPS coordinate, actuate our salt spreading system, and generating a salt spreading path with an intuitive geometry-based map UI

8.2 Value Provided

Our design meets many of the requirements stated initially and of our intended users. We have an intuitive user interface that can be used by the elderly and those with mobility impairments. The robot platform is versatile and acts as a robust proof of concept for autonomous vehicle systems. We were able to prove the feasibility of many subsystems such as GPS localization, camera object detection, and power distribution. This project shows that a cost effective and user friendly autonomous salt spreading robotic platform is able to be constructed.

8.3 Next Steps

Our first steps if this project were to continue would be to complete the integration needed for full system testing with the path finding, drive control, and collision avoidance working as one within our custom task scheduling software.

From this point, the next steps of our design would shift towards review and modification of our existing design to meet usability standards. We would move forward with waterproofing of the electrical system, component housings, and other at risk components. We would also move toward redesigning the drivetrain to incorporate tracks, rather than wheels, for a more capable movement system. We would also move forward with a potential lighting system to allow usage of our object detection systems at night. Much of our code base would benefit from optimizations that would allow for better runtimes and more responsiveness on our resource-limited microcontrollers.

9 References

[1] T. Cozzens, "Autonomous Snowbot Pro hits the sidewalk," *GPS WORLD*, May 13, 2019. [Online]. Available: https://www.gpsworld.com/autonomous-snowbot-pro-hits-the-sidewalk/. [Accessed Oct. 28, 2025].

[2] "Yarbo Snow Blower," *yarbo.com*. [Online]. Available: https://www.yarbo.com/products/snow-blower-robot?srsltid=AfmBOopBsykaTng7mL1GXIdoekc9fM0DQVTO1UuS27EhjqroopcLO02d. [Accessed Oct. 28, 2025].

[3] "Automower 450X," *husqvarna.com*, [Online]. Available: https://www.husqvarna.com/us/robotic-lawn-mowers/automower-450x/. [Accessed Oct. 28, 2025].

[4] Uline, "Ice Melt Rock Salt" uline.com [Online]. Available: https://www.uline.com/Product/Detail/S-7125/Grounds-Maintenance/Ice-Melt-50-lb-Bag?pricode=WB0870&gadtype=pla&id=S-7125&gad_source=1&gad_campaignid=21817469039&gbraid=0AAAAAD_uetMwZBmhPKFY5P5RqjBCIX7py&gclid=CjwKCAiAxc_JBhA2EiwAFVs7XCHLeK6D7Jeg16s3kDRDKmSpuHSGglY02537WrWP5SbfShqh6810fx0C_P0QAvD_BwE. [Accessed Nov. 6th 2025]

10 Appendices

10.1 Appendix 1 - Operation Manual

First, open up the application, when nothing is connected, you should see the following screen.

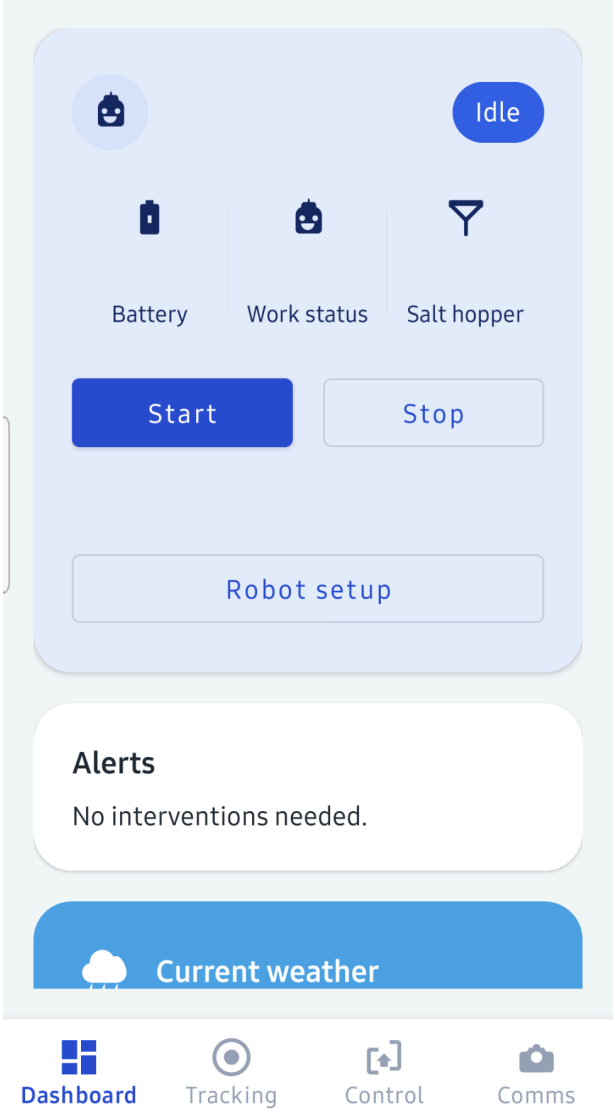


Figure 10.1.1

For the first use, please select “Robot Setup” to begin. This will lead to the following screen, where you will connect via bluetooth to set up the wifi connection.

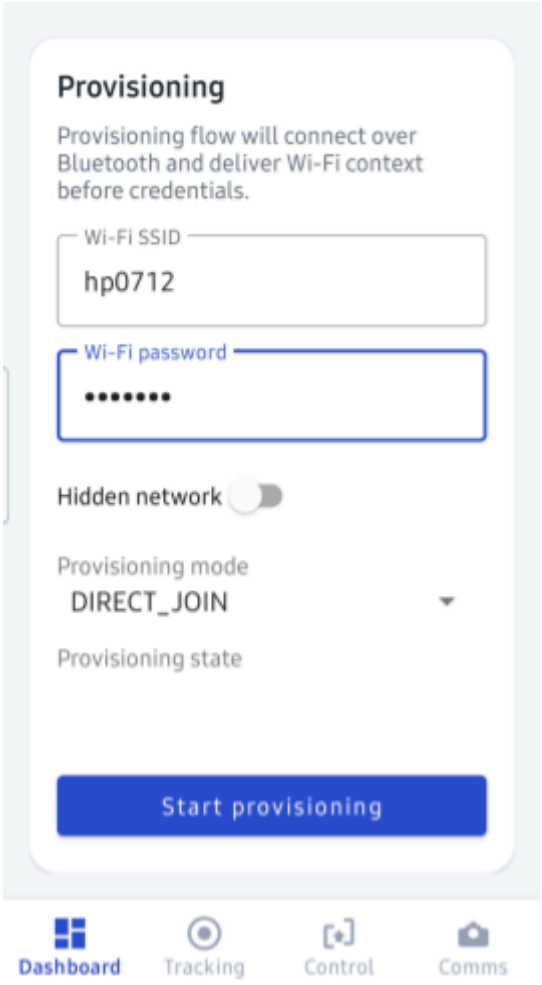


Figure 10.1.2

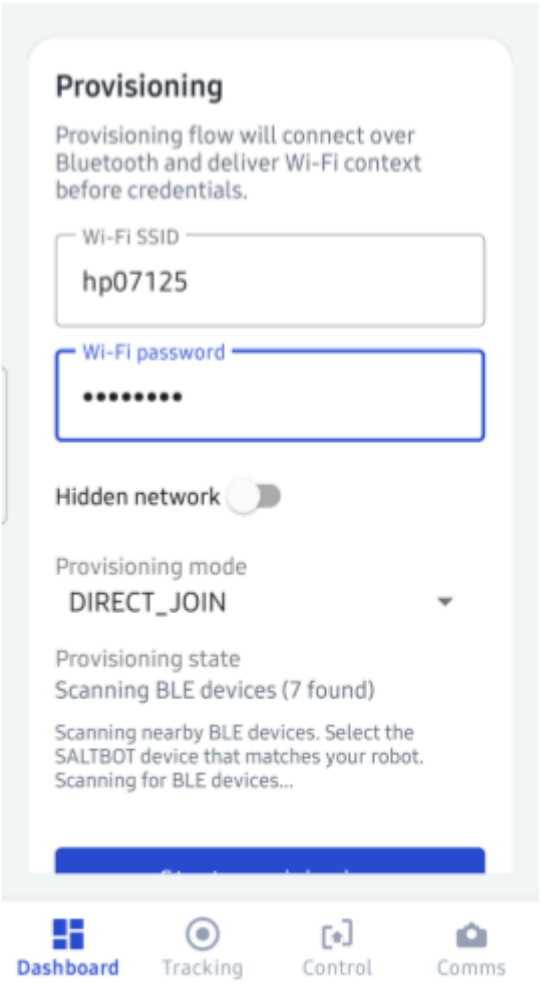


Figure 10.1.3

When you are on the first screen above, enter in your wifi information, enable bluetooth on your device, and then click the blue button that says “Start Provisioning”. The screen will scan and show nearby devices, and you will want to select the device “Salt_Spreader_R4” to connect to the device.

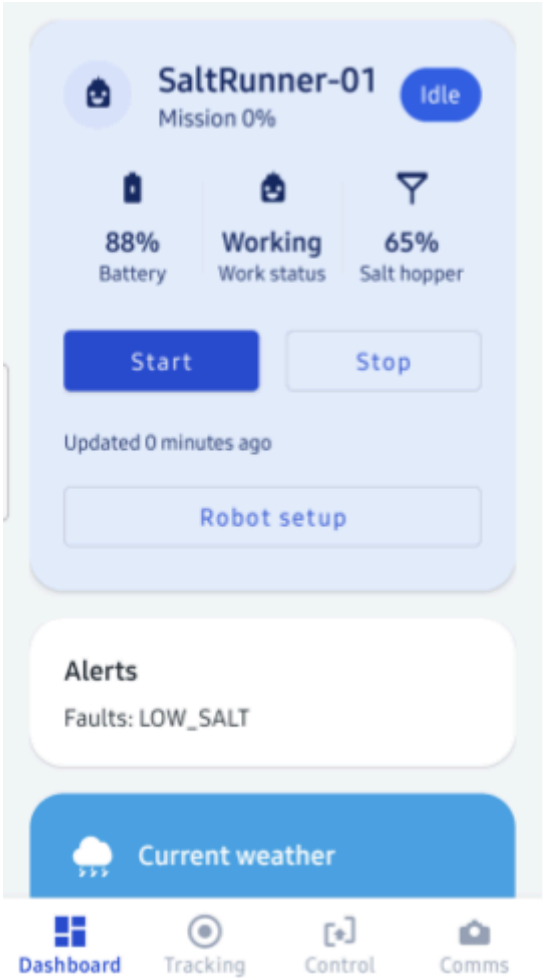


Figure 10.1.4

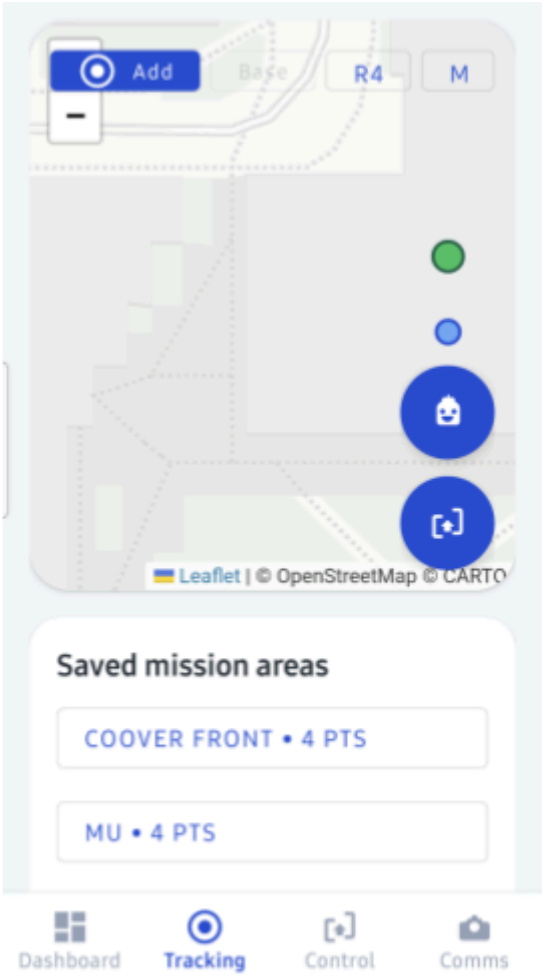


Figure 10.1.5

When the device has been linked, the home screen should change to look like the above picture. To start setting up the salt zone, please click on “Tracking” in the bottom bar, which should lead you to the other screen. The green dot is your user device, and the blue dot is the connected robot. If you have saved zones you can click those on the bottom middle, or you can click the blue “Add” button in the top left corner.

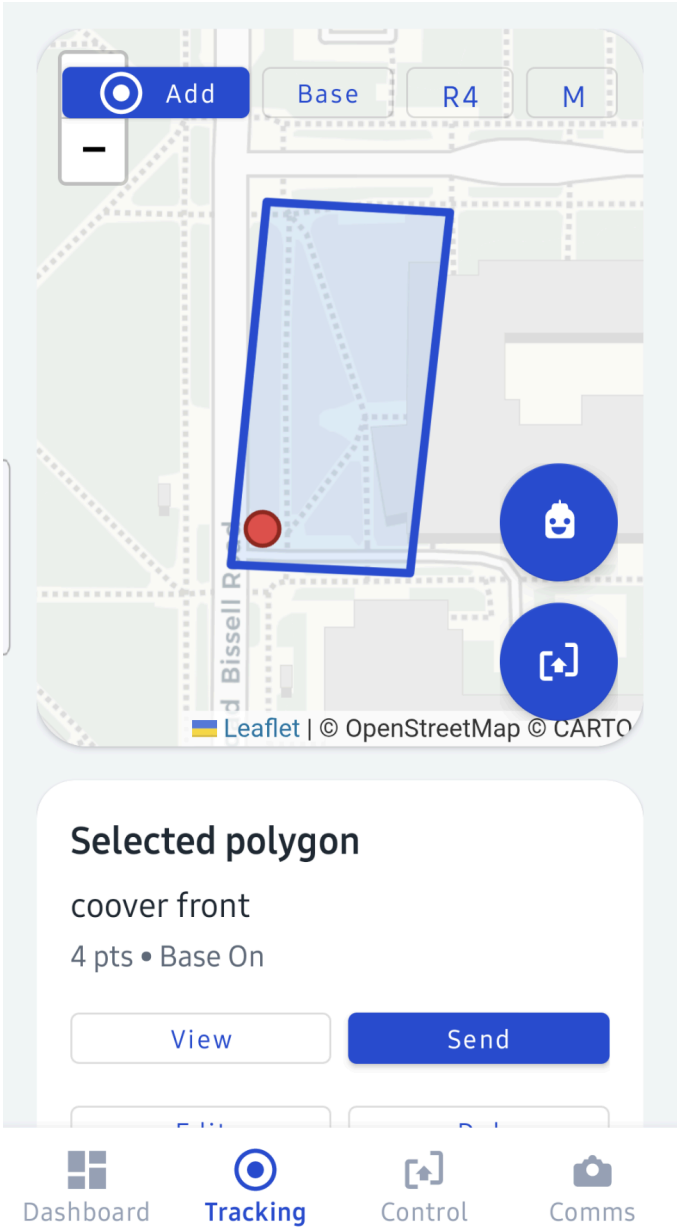


Figure 10.1.6

Pictured above is what the screen will look like if you click on an existing zone, with the red dot being the robots base station, where it will return when complete.

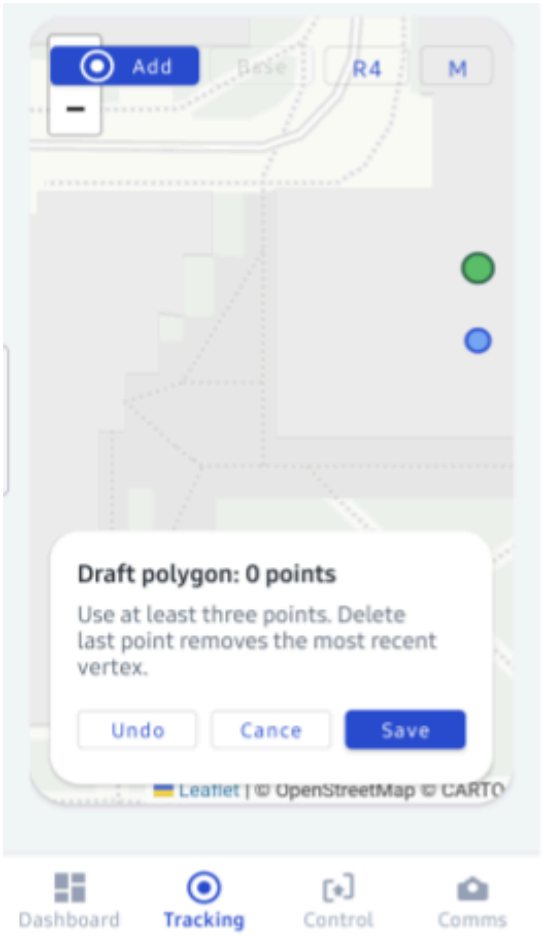


Figure 10.1.7

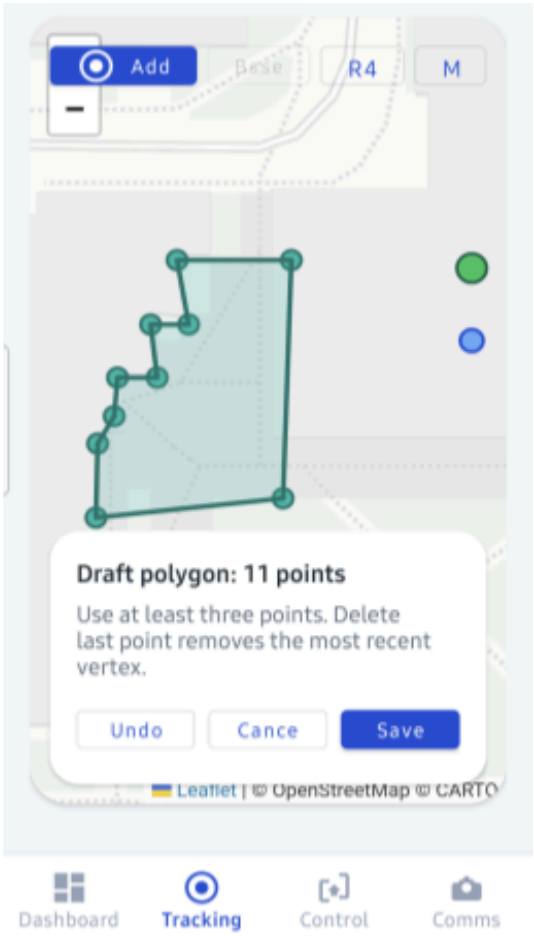


Figure 10.1.8

If adding a new zone you will be presented with the above screen. Clicking on the map will add boundary points, allowing you to trace out what you want salted. Once done, click “Save” to bring up the following screen and save the zone under whatever name you desire.

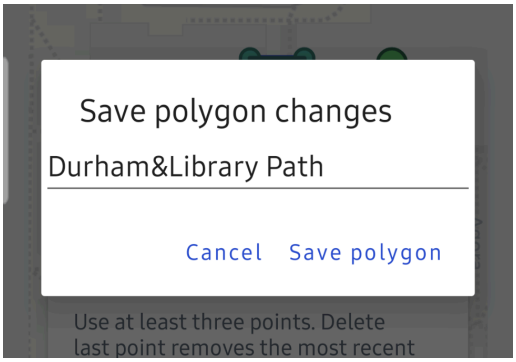


Figure 10.1.9

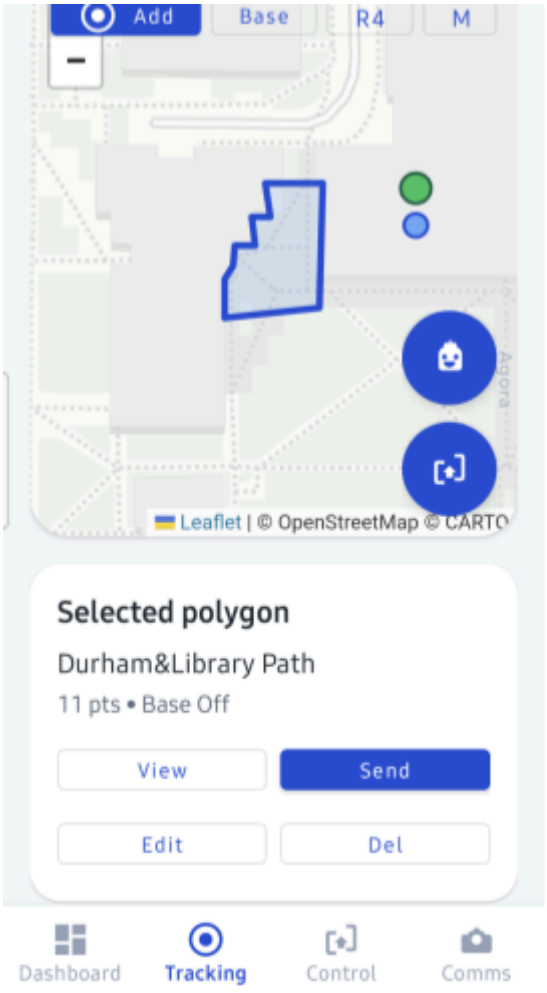


Figure 10.1.10

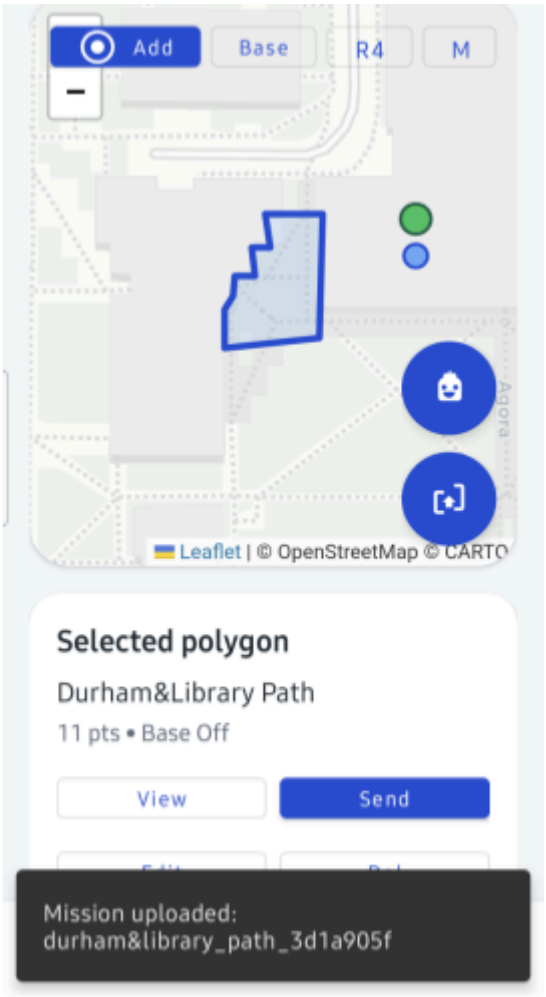


Figure 10.1.11

Once you have saved the zone, or opened a previously saved zone, you will be presented with the above screen. Clicking “View” will show you the points, “Edit” will allow you to change the zone, and “Del” will delete the chosen path. Clicking “Send” will send the path to the robot, and if the “Mission uploaded” banner appears, you have successfully uploaded the path to the robot.



Figure 10.1.12

To turn the robot on to connect via bluetooth, flip the first switch on the left from the pictured position to up. Before plugging in the robot, ensure the power is switched off for the robot. The first switch on the left is power for the whole robot, the second switch is for the drivetrain specifically, and the third is for the salt spreaders electronics.



Figure 10.1.13

Load the salt spreader by adding your salt of choice to the top of the hopper.

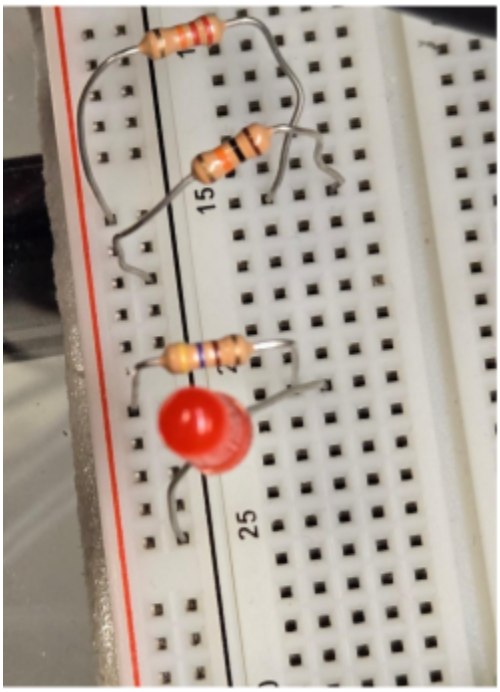


Figure 10.1.14



Figure 10.1.15

Once the robot has completed spreading the salt, to charge it, ensure the red LED pictured above is turned OFF, and then you can plug in the charge to the adapter pictured above in figure 10.1.15. The green LED on the charger base will become solid when the batteries are full.



Figure 10.1.16

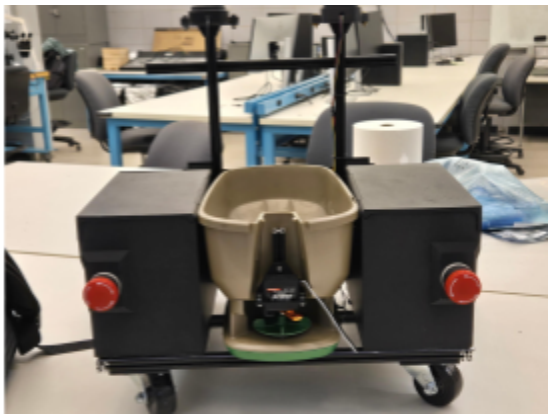


Figure 10.1.17

In-case of emergency, please depress any of the e-stop switches, which will cut power to the motors and prevent the robot from driving.

FAQ

Robot won't run?

Check switches are set to on and e-switches aren't depressed, and the red LED and the LEDs on the Arduino, GPS, and motor controller are all on.

10.2 Appendix 2 - Alternative/initial version of design

Tracked-Drive Concept:

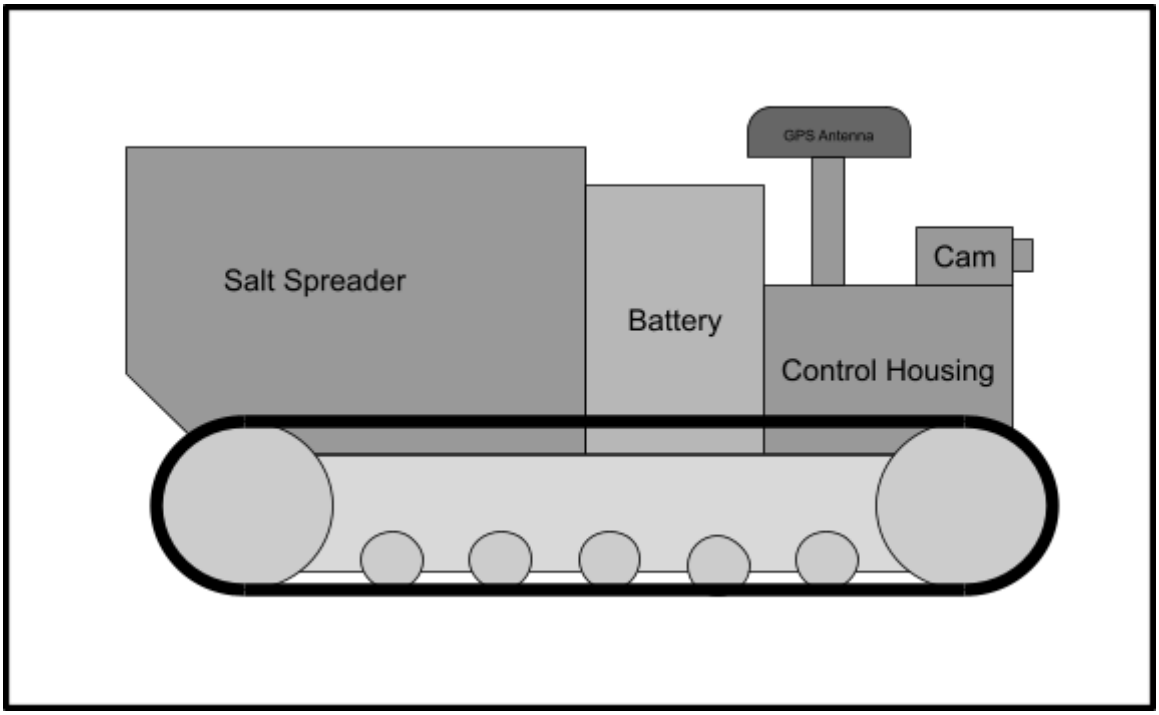


Figure X: Tracked Drive Concept Drawing

Our initial concept for our drive system was to use a tracked drive system in order to increase the contact patch, and therefore traction on the low-friction surfaces common in winter conditions.

However, due to budget concerns and the time-to-implementation for the mechanical design, we scrapped this in order to focus more on the electrical and software aspects of the project.

Prototype 0.5:

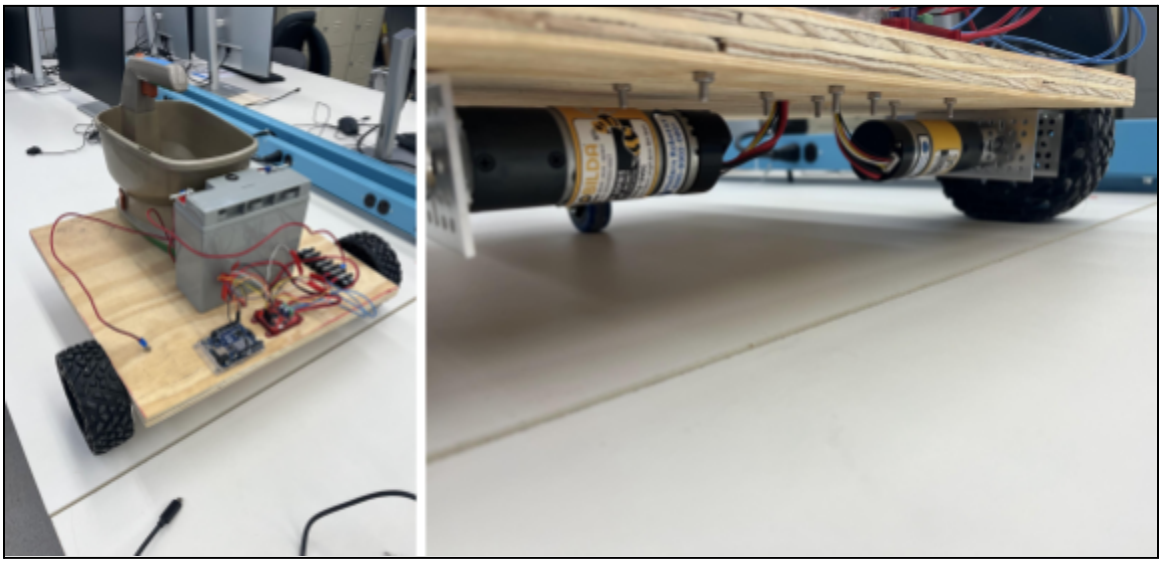


Figure X: Prototype 0.5

Prototype 0.5 was our first iteration of the actual implementation of our design aimed at working out potential mechanical limitations and providing a test bench for early software development.

As a whole, this platform was phased out for prototype 2.0 in order to improve appearance, weather-resistance, weight reduction, and mounting flexibility.

10.3. Appendix 3 – Code

See Git repository for detailed code documentation.

Git Repository Link: https://git.ece.iastate.edu/sd/sdmay26-07#

10.5 Appendix 4 -Team Control

Team Members

- Hoyt Dzwoniarski
- Mikhail McDade
- Dylan Bunting
- Thomas Huedepohl
- Isaac Blandin
- Shobhit Singh

Required Skill Sets for Your Project

- Mobile Robot Design
- Electrical Design
- Software Design
- Sensor Integration

- Mobile App Development
- Mechanical Design
- 3D Printing

Skill Sets covered by the Team

- Mobile Robot Design - Hoyt, Thomas, Isaac
- Electrical Design - Hoyt, Dylan, Thomas
- Software Design - Mikhail, Isaac, Shobhit
- Sensor Integration - Isaac, Shobhit
- Mobile App Development - Mikhail, Isaac, Shobhit
- Mechanical Design - Hoyt, Thomas, Isaac
- 3D Printing - Dylan, Thomas, Isaac

Project Management Style Adopted by the team

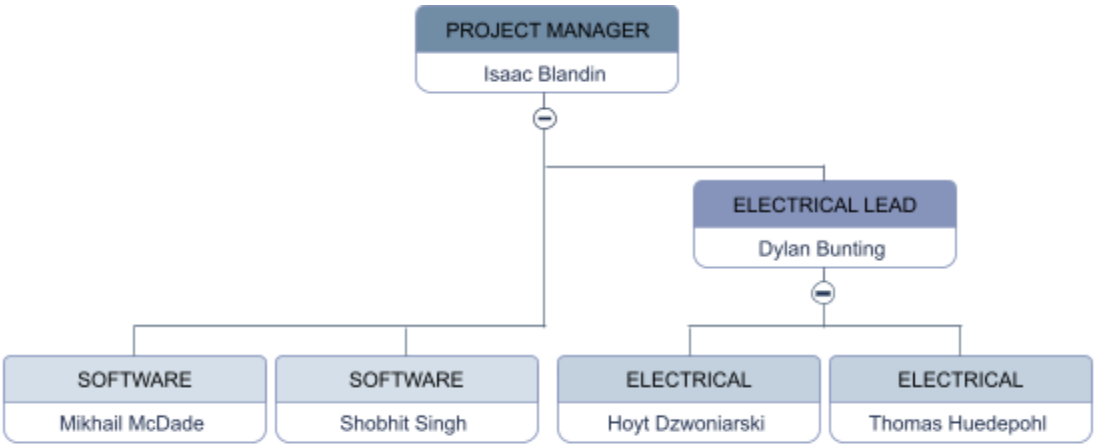
Our team adopted an agile framework as our project management style due to the highly exploratory nature of our project. We expect setbacks and revisions to the design over time, so an agile framework was needed for its flexibility.

Project Management Roles

Isaac Blandin - Project Manager/Software Team Lead

Dylan Bunting - Electrical Team Lead

We are organized as shown in the following organizational chart



10.5 Appendix 5 Team Contract

Team Members:

1) Hoyt Dzwoniarski
2) Thomas Huedepohl
3) Mikhail McDade
4) Isaac Blandin
5) Dylan Bunting
6) Shobhit Singh

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    a. All meetings with clients should take place face-to-face to allow for the fastest feedback and guaranteed attention.
    b. Team meetings will take place virtually for improved flexibility unless face-to-face is considered necessary for any specific meeting.

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. Discord will be used for group communication and communication with the client.
    b. Outlook calendar will be used to track the scheduling of group members and organize meetings efficiently. Members are expected to keep their calendars updated with classes and all other time commitments.

3. Decision-making policy (e.g., consensus, majority vote):
    a. Consensus will be the desired and ideal decision-making policy for group decisions; however, if we are at a standstill, majority vote will be used to break the tie.

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    a. All meeting minutes should be recorded by the team member holding the role of Secretary and placed in the team drive once compiled following each meeting.
    b. Each member shall still be responsible for recording any information important for their work during meetings.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
    a. Team members are expected to attend all team meetings and actively participate in all group discussions unless an excruciating circumstance presents itself.
    b. Members should contact the group if they cannot attend a meeting as far in advance as possible.
    c. If members miss a team meeting for more than two consecutive weeks, we will work together to figure out a solution to allow for a more conducive meeting time.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    a. Team members should each contribute a relatively equal amount of work to the culmination of the final project.

   b. Certain members may have more of a focus on certain aspects of the project (such as client relations, technical design, organization, planning, etc); however, the hours and work contributed should be generally evenly distributed across all team members.

3. Expected level of communication with other team members:
   a. Team members should reply promptly to all team messages and emails. Members should aim to respond within 24 hours to any given message.
   b. When a question is posed to the whole group, a written response may not always be necessary; however, members should at a minimum acknowledge the message (reacting to the message on Discord, etc).

4. Expected level of commitment to team decisions and tasks:
   a. Members will all have an equal opportunity to contribute to team decisions and tasks.
   b. If a member has been infrequent in their contribution to the project recently (missed a meeting, unresponsive to messages, etc) they may forfeit some of their decision-making power in favor of members who have been actively leading and contributing to the project.
   c. Members who fail to properly contribute to the project will be confronted if the rest of the group finds their behavior inadequate.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
   a. **Hoyt:** Creative Design
   b. **Isaac:** Project Manager / Software Team Lead
   c. **Mikhail:** Communications
   d. **Shobhit:** Meeting moderator & Secretary (keeping tabs and minutes)
   e. **Thomas:** CAD Co-lead
   f. **Dylan:** Electrical Team Lead

2. Strategies for supporting and guiding the work of all team members:
   Weekly checkups to see how well each team member feels about the project, and if there is a bigger issue, then we can try to discuss a solution as a team, but any other questions or concerns will probably end up in a Teams or Discord chat.

3. Strategies for recognizing the contributions of all team members:
   By setting specific goals and tasks for each team member, we can monitor progress via weekly checkups and or progress reports via Teams meetings

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
   a. **Hoyt**: EE major with experience in circuit design, digital logic, and a focus on VLSI
   b. **Isaac**: CPRE Major with experience in: microprocessor usage, sensor integration, sensor fusion, low-computational power algorithm creation, parametric modeling, system simulation, 3D printing, CNC manufacturing, rapid prototyping.

c. **Mikhail:** CprE major, with a background in networking, cybersecurity, Circuit design, embedded systems programming, CPU MIPS architecture, and a familiarity with running and monitoring Linux servers
d. **Shobhit**: A CPRE major, with background in cybersecurity protocols, embedded systems programming and an interest in pursuing quantum computing research. Thorough understanding of embedded level programming protocols and principles along with networking models and encryption algorithms. This project is connected to the history of computing which made me interested in it.
e. **Thomas:** EE major with experience in embedded systems, cad, 3d printing, CNC
f. **Dylan**:EE major with a background in digital logic, semiconductors, circuit design, embedded systems, 3D printing, and CAD.

2. Strategies for encouraging and supporting contributions and ideas from all team members:
    a. Assigning tasks based on interests and expertise, along with always keeping check on other team members through an accountability buddy system. Communicating about new ideas as well as facing a block in the Discord, so that no one is ever out of the loop in any situation.
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
    a. Clear communication that one feels that they are not being provided the means to contribute is the first step in this situation. If an inclusion concern is brought up, it shall become the first talking point at the next team meeting.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    a. Improve documentation and communication skills in the context of an engineering project.
    b. Effectively communicate and collaborate on tasks to optimize productivity as a team rather than as an individual.

2. Strategies for planning and assigning individual and team work:
    a. Create a scrum board (or similar) for team tasks, assigning story points to all tasks
    b. Perform workload planning for all members per sprint based on story points.
    c. Consider outside obligations, such as exams, in each member's workload capacity for any given planning cycle. (e.g., Allow for a lighter workload for a member during a week that they have multiple exams without viewing them as contributing less).

3. Strategies for keeping on task:
    a. The assigned Scrum tasks with deadlines provide a focused task that must be worked towards.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    1st time: Respectful conversation with the rest of the team.
    2nd time: Team will discuss consequences.
    3rd time: Consequences implemented.

2. What will your team do if the infractions continue?
   If infractions continue following communication and/or consequences from the team, instructors will be contacted for further action.

****

a) *I participated in formulating the standards, roles, and procedures as stated in this contract.*
b) *I understand that I am obligated to abide by these terms and conditions.*
c) *I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.*

1) Isaac Blandin DATE 09/11/2025
2) Hoyt Dzwoniarski DATE 09/11/2025
3) Thomas Huedepohl DATE 09/11/2025
4) Mikhail McDade DATE 09/11/2025
5) Dylan Bunting DATE 09/16/2025
6) Shobhit Singh DATE 09/16/2025