Autonomous Lawn Mower

Final Engineering Report

By: James Boumalhab, Tamrah Hughes,
Ryan Murray, and George Renzo

Electrical Engineering Senior Design

2024/2025

OUTLINE

1. Executive Summary

This engineering report details the design, implementation, and testing of an autonomous lawn mower intended to solve the problem of time-consuming and labor-intensive lawn maintenance. Maintaining a lawn can be physically demanding and even hazardous, particularly for elderly individuals or those with large properties. While automation has transformed many household tasks, lawn care has remained largely manual. Our solution brings the convenience of indoor robotic systems, like vacuums, to outdoor maintenance by leveraging modern sensors and control systems for hands-free mowing, complemented by smartphone-based manual control.

2. Introduction

As time goes on, automation becomes more and more common. On a quick inspection, it is easy to see why: it is often very convenient. Automation has been brought to all manner of industries and applications, one popular household example is with vacuums. However, some appliances have not taken as quickly to automation.

One of these appliances is lawn mowers. Despite acting in a similar manner to a vacuum, autonomous lawn mowers have not seen the same popularity. Lawn maintenance remains a recurring task in residential and commercial settings, often demanding significant time, physical effort, and attention to detail. For elderly individuals, people with disabilities, or owners of large

properties, mowing the lawn can be particularly burdensome and even pose safety risks. Despite the rise of home automation technologies, such as robotic vacuum cleaners and smart home assistants, automated outdoor maintenance tools have lagged in accessibility and adoption.
Both an automated lawn mower and automated vacuum primarily solve a similar problem, one of convenience. As a general theme, convenience is a major reason for many examples of automation.
This project seeks to bring the same convenience to lawn mowers as automation has brought to other tasks. The proposed system plans to incorporate several systems with the overall goal of being versatile and durable. The versatility and durability of the system will establish it as a practical and logical introduction to future lawn maintenance.

This project attempted to address that gap through the development of an autonomous lawn mower capable of efficiently maintaining a lawn seeks with minimal human intervention. By combining sensor integration, motor control, and wireless communication, our system provides both autonomous operation and manual control via a smartphone for flexibility and reliability. The mower can navigate a predefined path, avoid obstacles, and relay status information, while still allowing the user to take control in more complex scenarios.

To accomplish these goals, the mower used GPS, ultrasonic sensors, and an inertial measurement unit (IMU). The original intention was that GPS would locate the mower on the lawn, using the ESP32 to record and keep track of the points, the ultrasonic sensors would detect immediate obstacles around the mower, and the IMU would detect if the lawnmower was at a dangerous tilt.

The goal of this project, primarily considering the time and monetary constraints, was to show

more of a proof of concept rather than a market-ready product. This goal means that we planned to create a basic physical build capable of supporting the algorithm which mows the lawn autonomously. The algorithm itself would rely on widely available electronic systems for detection and sensing rather than specialized equipment.

Despite the rudimentary expectations of the project, creating the algorithm presented many difficulties and required significant adaptations, the main issue being the accuracy of the GPS module. At the end of the project, the lawn mower had a baseline for an autonomous system rather than a fully integrated and graceful product. The system was basic but demonstrated the capability of the equipment to mow the lawn with minimal human intervention. The final design of the autonomous system differed significantly from original plans for navigation. Rather than relying primarily on GPS, the final demonstrated product mainly relied on the IMU and timing of the motors to gauge distance and direction. There were also routines created which integrated the inaccurate GPS with the ultrasonic sensors to positively identify location by saving locations near permanent obstacles.

The other aspects of the project largely met expectations set from the beginning of the project, not requiring complete redesigns. Of the notable changes, the size of the physical build was reduced since it made turning easier and the benefits of a larger mower became irrelevant as the issue of inaccurate GPS was discovered to be unsolvable with our resources. The BLE subsystem for a remote control option began with MIT App Inventor and MATLAB for the app, but the app was changed to an iOS app, coding in Swift through XCode, as it gave the complexity and availability needed by the ESP32 for communication.

Overall, the autonomous mower project experienced many changes from the original design that we envisioned. With these changes, this project accomplished a basic system which shows the capability of autonomous lawn mowing using these available sensors.

3. System Requirements

3.1. Requirement Overview

Table 1 outlines the initial system requirements and corresponding subsystems necessary to achieve the desired functionality for the autonomous lawn mower. These requirements address various subsystems, including power management, Bluetooth communication, navigation, and safety systems.

Table 1. List of Initial Requirements

Subsystem	Requirement
Power	Power lasts for 40 minutes
Power	Power ESP32, sensors, and motors via battery
Bluetooth App	Enable control of navigation via an app
Bluetooth App	Receive status updates via an app
Bluetooth App	Range of 50 meters
Navigation	Follow predetermined route to mow lawn
Navigation	Detect and obstacle 1 meter away and avoid it
Navigation	GPS accurate to .25 meters
Safety	Have emergency stop button for blade motor

3.2. Power Requirements

Beginning with power, the final version of the project was envisioned to have all the power be sourced from a rechargeable battery. This battery was to be 12 V or 24 V. Having one power source simplified the design and would make power management for the user easier, having only to recharge one battery rather than multiple. Another battery would have been for the smaller components, such as the sensors and ESP32 while a larger battery would power the motors. In this design, the project would use one battery for all systems.

The requirement for lasting 40 minutes came from an estimation of how long it should take to mow a small to medium sized lawn with a regular existing lawn mower. The lawns around South Bend gave a good reference for the lawn the project planned on hypothetically serving. These lawns are small, relatively flat, and have simpler geometries. With the autonomous lawn mower lasting 40 minutes, the target user could have their lawn mowed on one charge. With this requirement as a reality, if a user was to have a larger lawn, they would have to anticipate recharging.

3.3. Bluetooth Requirements

The bluetooth app aspect of this project was included primarily in order to give users the ability to use the mower without the autonomous program running. This option would allow the user to

use the mower in situations where the autonomous algorithm may fail, perhaps encountering an unexpected scenario. Without this function, users would need to use a regular lawn mower or physically manipulate the mower themselves, in either case possibly making having an autonomous mower too troublesome if the user has to use a regular mower anyway.

The second requirement for the bluetooth subsystem was to receive status updates from the lawn mower. This function would allow the user to keep track of the progress of the lawn mower without having to watch it from beginning to end to understand its progress and condition. Important information to include in the status update was the mower's movement and position. While this aspect of the app was secondary to control of the mower, this information available on the app could also be useful for debugging on the user's end.

The requirement for the BLE communication to reach 50 meters was again based on the model of a lawn in the South Bend area. The distance of 50 meters was roughly the furthest point of the lawn from the house, making it so that the bluetooth app would reach all over the lawn of the user.

3.4. Navigation Requirements

The navigation and obstacle detection were the most important requirements of the project, otherwise the project would not fit the description of an autonomous lawn mower. These navigation requirements were made to fit the navigation and obstacle avoidance that was originally conceptualized.

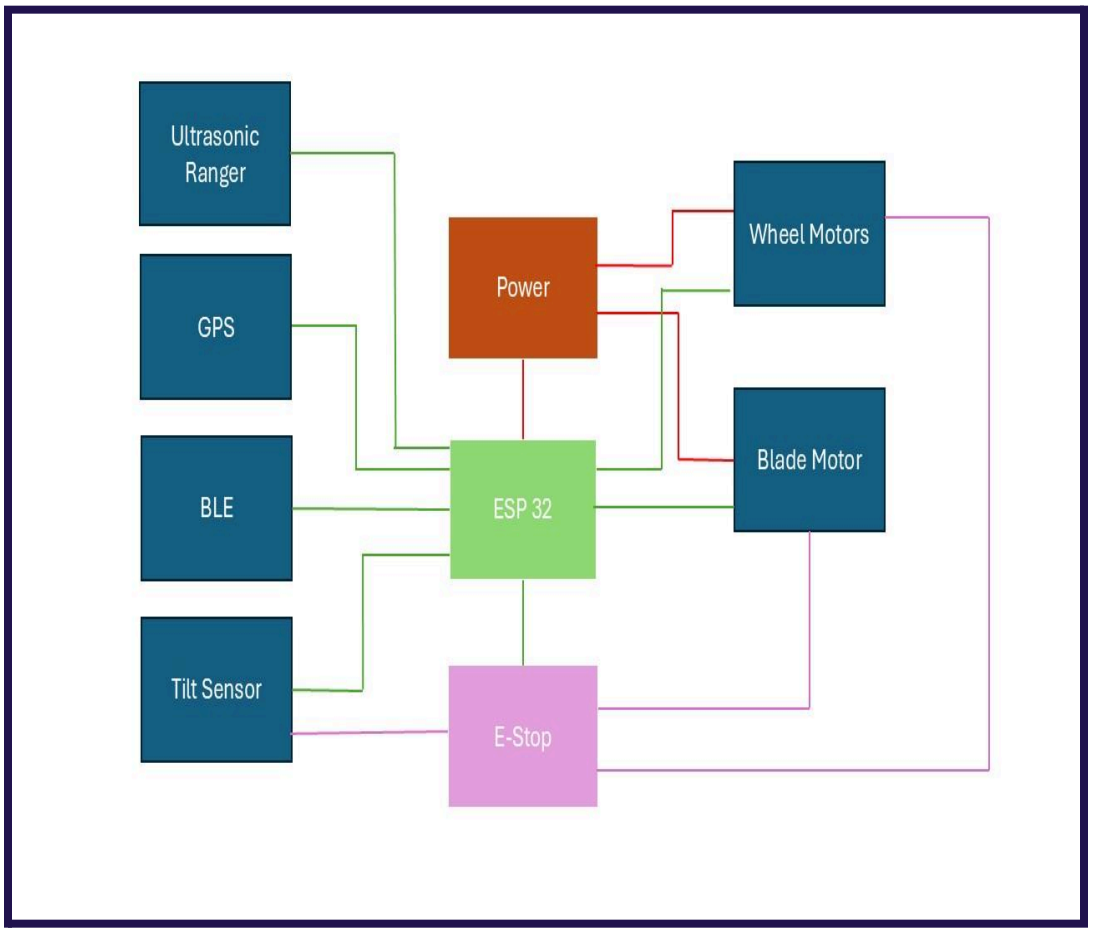
The requirement to follow a predetermined route to autonomously mow the lawn would be the

primary requirement of the project. This requirement would be accomplished through the use of sensors on the mower, originally using GPS.

More toward the obstacle avoidance aspect of the subsystem, there is the requirement to see an object from 1 meter away and navigate around it. The distance of 1 meter was chosen since it was a fair distance for a lawn mower the size of a standard push mower to be moved around some obstacle such as a tree without having to complete some complicated movement. In this way, the lawn mower would have space to turn while moving forward instead of having to back up before continuing.

The other requirement within the navigation subsystem was having a GPS accurate to .25 meters. This distance would allow the lawn mower to know its location based on GPS well enough to not miss strips of grass while completing the autonomous routine. This requirement was made when the navigation system was conceptualized as relying almost entirely on GPS. As the nature of issues related to the accuracy of the GPS modules the project had access to became clear, the navigation system was adapted in a way which made this requirement obsolete.

4. System Overview



The autonomous lawn mower system consists of multiple subsystems working in unison to achieve its goal of autonomous lawn care. Below is a description of each subsystem and its role within the system:

4.1 Power Management Subsystem

Purpose:

The power management subsystem ensures that the mower remains operational for the desired

period, providing stable power to all components, including the microcontroller, sensors, and motors.

Key Components:

- 12V 18Ah Battery
- Step-down converters
- Power management network

Function:

The system can be powered either by a 12V 18Ah or a 24V 18Ah battery, depending on the required operation time. If about 20 minutes of operation is sufficient, the 12V 18Ah battery will suffice. However, if longer operation is needed, up to 40 minutes, the 24V 18Ah battery would be more suitable. We opted for the 18Ah capacity for multiple reasons, two of the most important being cost and current consumption. Given that we have three motors, two of which can draw a maximum current of approximately 12 amps each, along with the suite of sensors, the 18Ah battery size made the most sense in terms of meeting power requirements while staying within our budget.

Below, you can find a picture of the type of battery we decided on.

Figure 1: Image of the Selected Battery Type for the System

The system also utilizes two voltage regulators to manage power distribution for different components. One regulator is an LDO that outputs 1.8V to power the on-board IMU, which provides motion and tilt data essential for navigation and safety. The second regulator, a step-down converter, provides 3.3V to power the ESP32 microcontroller and other devices, ensuring stable operation of the control system and sensors. These regulators help optimize power efficiency, reduce battery strain, and ensure each component receives the appropriate voltage for reliable performance.

Down below, you can find pictures of the two voltage regulators used in our system: one LDO regulator that outputs 1.8V to power the on-board IMU, and a step-down converter that provides 3.3V for the ESP32 microcontroller and other devices.

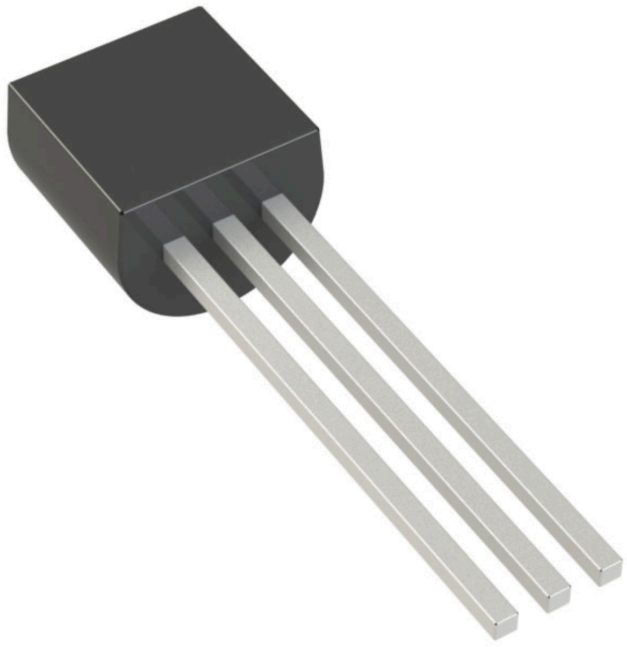


Figure 2: MCP1700-1802E/TO (LDO 1.8V Output)

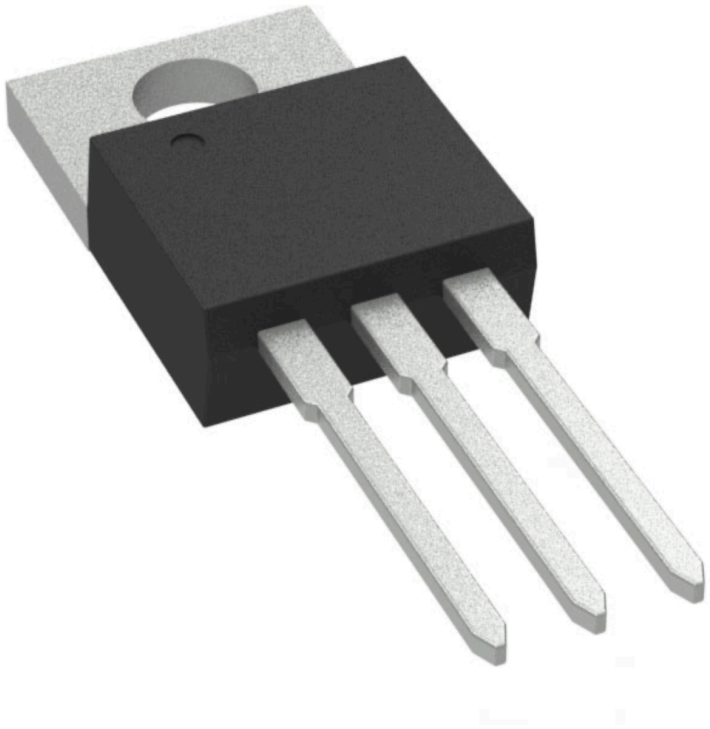


Figure 3: LM1085-3.3 (Regulator 3.3V Output)

The power management network ensures efficient power flow to each component, preventing power losses and ensuring that the mower operates within its designated power requirements.

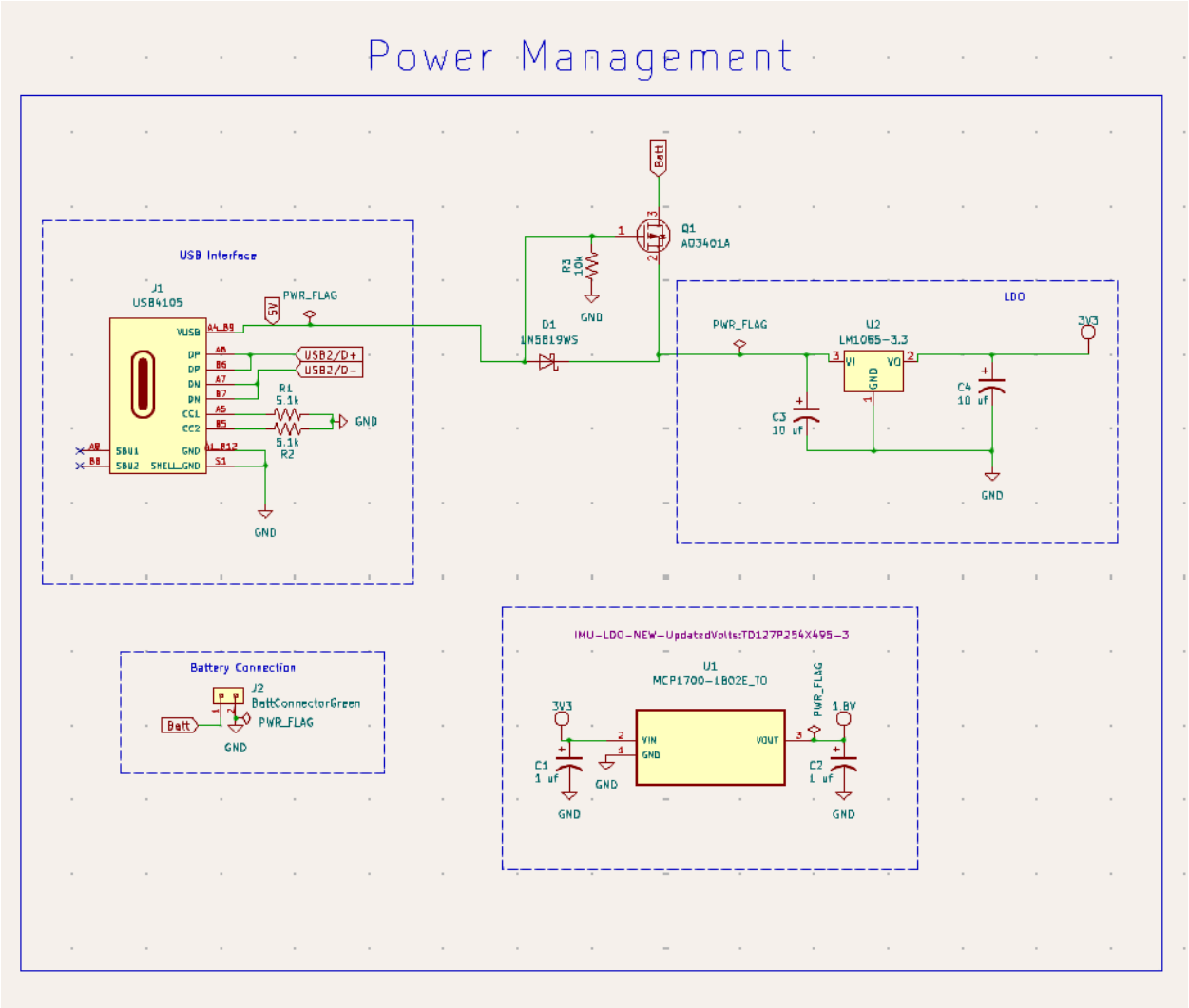


Figure 4: Power Management Schematic

4.2 Navigation and Control Subsystem

Purpose:

The navigation and control subsystem enables the mower to autonomously follow a predetermined path while avoiding obstacles in real-time.

Key Components:

- GPS module (Adafruit Ultimate GPS Breakout)
- Ultrasonic sensors (DFRobot URM09)
- IMU (SparkFun 9DoF IMU Breakout)

Function:

The Adafruit Ultimate GPS Breakout is ideal for our autonomous lawn mower due to its high sensitivity, low power consumption, and reliable performance. It tracks 22 satellites with 66 channels and has a 10 Hz update rate for accurate location tracking. With a low current draw of 20mA, it supports the mower's 40-minute operation time. Its compatibility with 3.3V Inputs and data-logging capability ensure efficient power management and reliable operation, making it a great fit for the system's navigation needs. Thus, the GPS module allows the mower to track its location and follow a predefined path across the lawn.

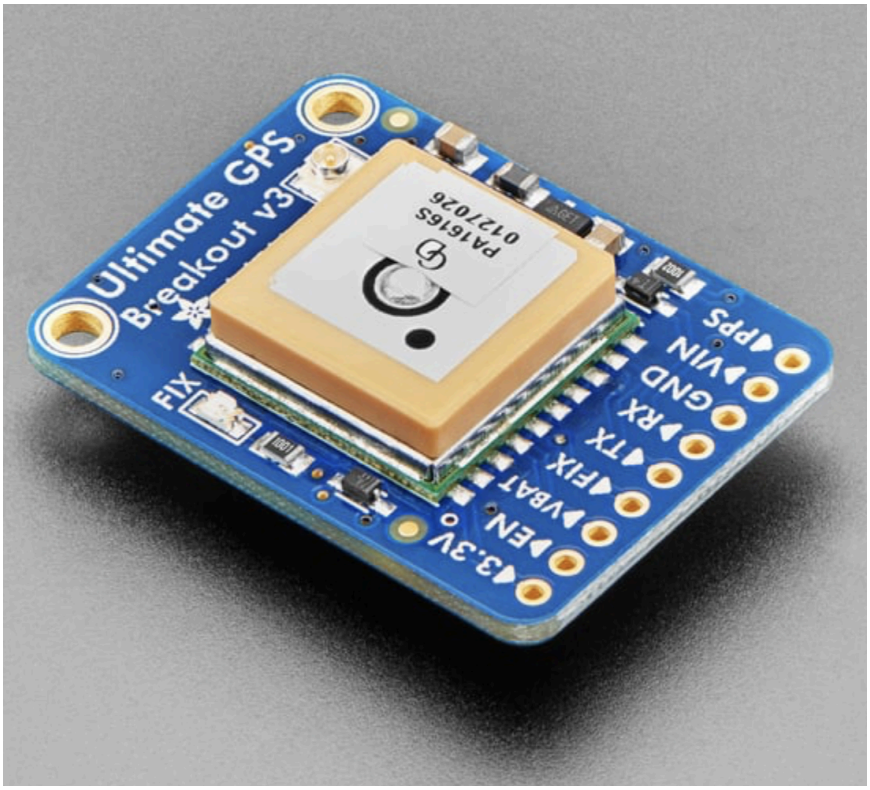


Figure 5: Ultimate GPS Breakout V3 Module

The Gravity URM09 Ultrasonic Distance Sensor was selected for its high accuracy and fast measurement frequency, capable of detecting obstacles with a resolution of 1 cm and a maximum frequency of 50Hz. Its sensitivity is crucial for the mower's ability to react quickly to changes in the environment, which is essential for safe navigation. The low current draw (20mA) and I2C communication also make it power-efficient, enabling longer operation time on the mower's battery.

Figure 6: The Gravity URM09 Ultrasonic Distance Sensor

The SparkFun 9DoF IMU Breakout (ICM-20948) was chosen to provide precise orientation and motion sensing for the autonomous lawn mower. This inertial measurement unit combines a 3-axis gyroscope, 3-axis accelerometer, and 3-axis magnetometer in a compact, low-power package. The IMU enables the mower to determine its real-time heading, acceleration, and angular velocity, critical inputs for accurate navigation and control. Accurate orientation data from the ICM-20948 supports path-following algorithms, ensuring the mower can follow predefined trajectories with minimal drift. In addition, the accelerometer enables a tilt-based safety shutoff feature, which immediately stops the blade motor if the mower tips beyond a safe

angle of 35 degrees—preventing accidents during operation on uneven terrain. The IMU communicates via I2C, which simplifies integration with the ESP32 microcontroller and reduces wiring complexity. Overall, the ICM-20948 enhances both the safety and navigation capabilities of the system.

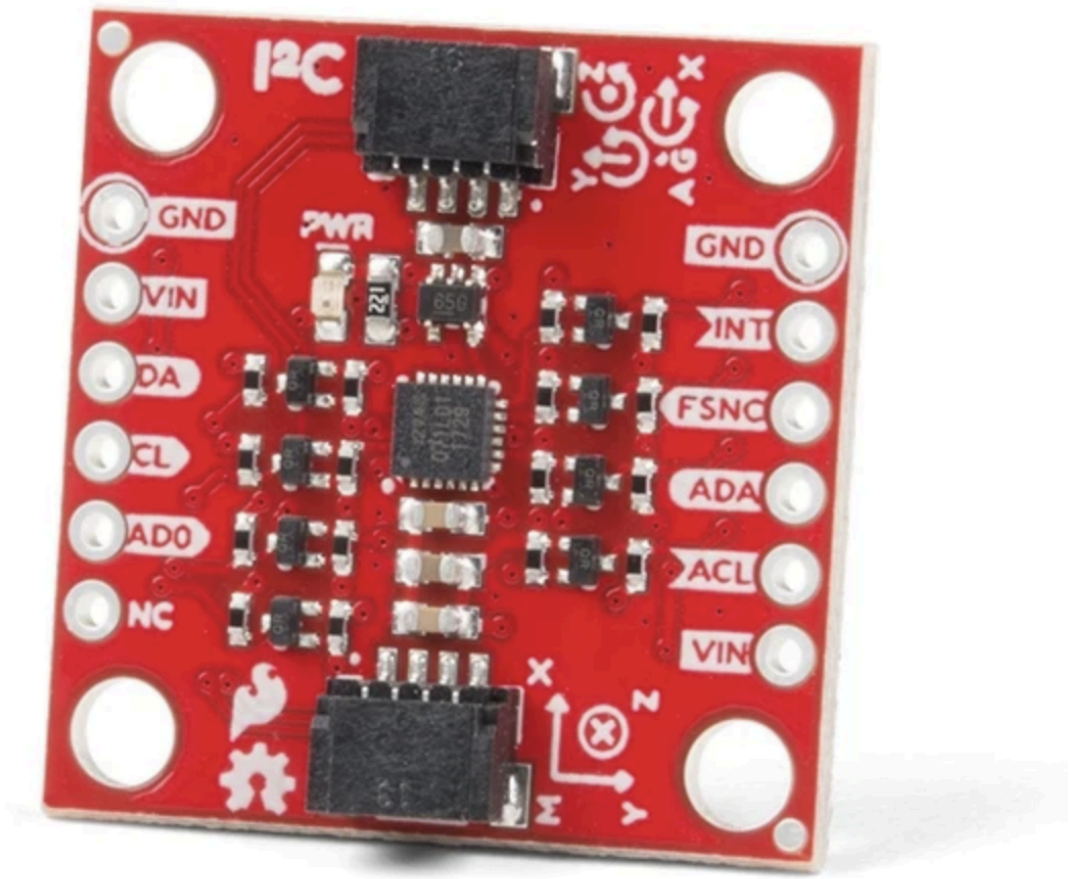


Figure 7: SparkFun 9DoF IMU Breakout (ICM-20948)

4.3 Bluetooth Communication Subsystem

Purpose:

The Bluetooth communication subsystem enables the mower to interface with a smartphone app for remote control and monitoring.

Key Components:

- ESP32-S3 microcontroller with Bluetooth Low Energy (BLE) capabilities
- Smartphone app for control and feedback

Function:

The ESP32 microcontroller enables Bluetooth Low Energy (BLE) communication between the autonomous mower and the user's smartphone. Through a custom mobile application, the user can control the mower's movement, start or stop operations, and receive real-time status updates. The BLE connection offers a reliable communication range of up to 50 meters, allowing users to manage the mower while attending to other tasks nearby. Additionally, the app displays critical information such as GPS location, battery level, and system status, ensuring the user remains informed and in control throughout the mowing process.

4.4 Safety Subsystem

Purpose:

The safety subsystem is designed to ensure the mower operates safely, with features to protect

users and prevent damage to the mower.

Key Components:

- Emergency stop button
- Blade safety mechanisms
- IMU (for tilt detection)

Function:

Safety is paramount for this autonomous mower. An emergency stop button allows the user to immediately halt the mower's operation in case of an emergency.

Figure 8: E-Stop Button

The onboard IMU continuously monitors the mower's tilt angle in real time. If the tilt exceeds 35 degrees—suggesting the mower may be tipping or has encountered unstable terrain—the system automatically shuts off the blade motor to prevent damage or injury. This safety mechanism

ensures the mower operates within safe parameters and can respond rapidly to unexpected conditions, enhancing overall operational reliability.

4.5 Electrical Assembly and Integration Subsystem

Purpose:

This subsystem involves the assembly and integration of all electrical hardware components into a fully functional system.

Key Components:

- PCB (Printed Circuit Board)
- Manual pick-and-place assembly process

Function:

The PCB acts as the central hub for the mower's electronics, interconnecting the ESP32 microcontroller, motor drivers, sensors, and power distribution circuitry. It ensures organized signal routing, compact layout, and reliable electrical connections across the entire system.

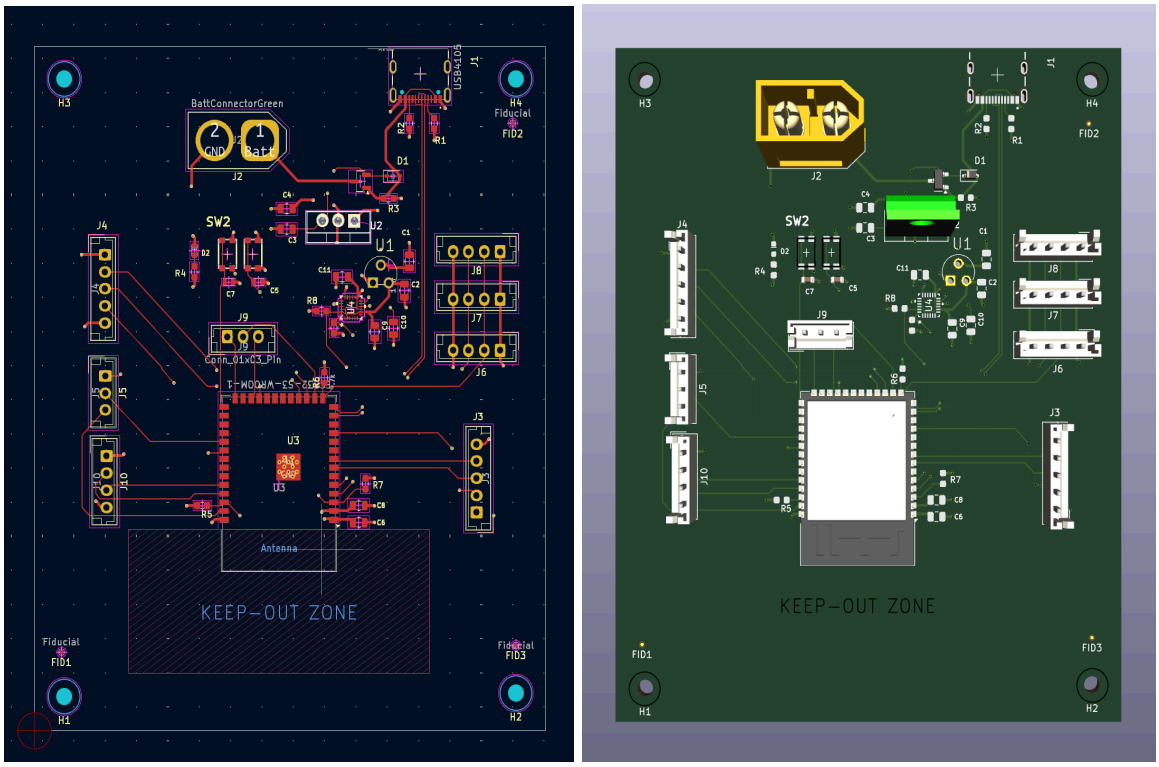


Figure 9: PCB Layout & 3D Viewer

The assembly process involves manual pick-and-place and hand soldering techniques to ensure that all components are properly installed and connected. The assembly process is performed at EIH, a facility with the necessary equipment and expertise to handle the integration of complex electronic systems. The careful integration of all subsystems ensures that the mower operates smoothly and all components are accessible for maintenance or upgrades.

4.6 Motor and Drive Subsystem

Purpose

To enable the mower to move autonomously across outdoor terrain and perform cutting operations effectively, while ensuring safe and controlled motion.

Components

- **Drive Motors:** Two 24V 250W brushed DC gear motors
- **Blade Motor:** 12V DC 775 high-speed motor (10,000 RPM)
- **Motor Drivers:** VNH5019 Motor Driver Carriers (×3)
- **Microcontroller:** ESP32-S3 (controls PWM signals for all motors)

Function

The two 24V gear motors are mounted on the rear wheels to provide propulsion. These motors use gear reduction to increase torque and reduce speed, enabling the mower to move steadily over grass, bumps, and inclines.

Figure 10: 24V 250W Electric Brushed Gear Reduction Electric Motor

The 775 blade motor spins the cutting blade at high speed. Powered by a 12V line, it delivers reliable mowing performance when combined with a sharp manganese steel blade. The blade motor is physically isolated from the drive motors for both safety and modularity.

Figure 11: DC 12-24V 775 Motor with Lawn Mower Set

Each motor is controlled by a VNH5019 motor driver. The two drive motors are managed using a compact Dual VNH5019 carrier, while the blade motor uses a separate single VNH5019 driver. These drivers are capable of handling high current loads and include built-in protection features such as thermal shutdown and overcurrent limiting. Speed and direction are controlled via PWM signals sent from the ESP32 microcontroller.

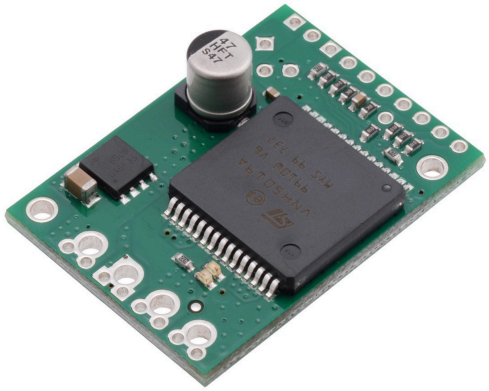


Figure 12: VNH5019 Motor Driver Carrier

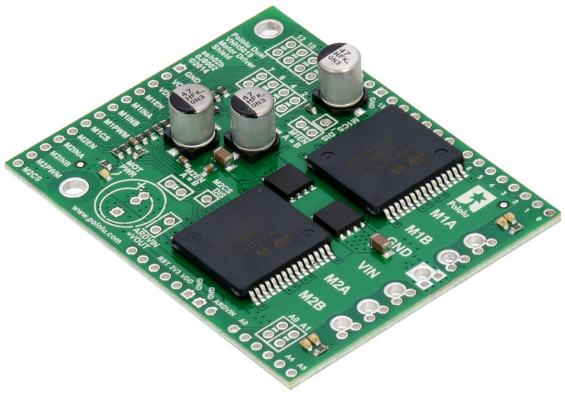


Figure 13: Dual VNH5019 Motor Driver Carrier

5. Physical Design and Assembly

The physical structure of the mower is built on a sturdy wooden chassis that supports all key components, including the motors, electronics, and cutting system. Wood was selected for its affordability, lightweight properties, and ease of cutting and drilling, which made it ideal for rapid prototyping. Just as importantly, Home Depot offered the tools and workspace necessary to make accurate cuts and adjustments on-site, which enabled quick iteration during the design and build process.

5.1. Initial Design and Four-Wheel Layout

In the first design iteration, the mower featured a four-wheel layout: two drive wheels at the rear and two fixed wheels at the front. While this configuration provided stability, testing quickly revealed severe turning limitations. The fixed front wheels restricted the mower's ability to pivot, resulting in a wide turning radius and difficulty navigating tight spaces.

To address this, we replaced the fixed front wheels with 360-degree caster wheels. This modification significantly improved maneuverability, allowing the mower to pivot smoothly and handle tight corners more effectively. Alongside this change, the wooden chassis was also resized, reduced in width and length, to further improve agility and reduce the turning radius, creating a more compact and responsive design.

5.2. Motor Mounting Challenges and Spacer Solution

The rear drive motors 24V 250W brushed DC units were mounted using metal brackets.

However, early in the build, we discovered sizing mismatches: the mounts did not perfectly align with the motors, and gaps between the motor housings and the brackets caused instability. After exploring several mounting options and failing to find brackets with ideal dimensions, we developed a practical solution using custom-cut wooden spacers.

These wooden spacers were placed between the motors and the metal mounts, bridging the gap and allowing for proper alignment. In addition to ensuring a snug fit, the spacers also helped absorb motor vibrations during operation, resulting in a more stable and quieter ride. The ability to hand-cut these spacers using Home Depot's resources proved essential to executing this fix quickly and effectively.

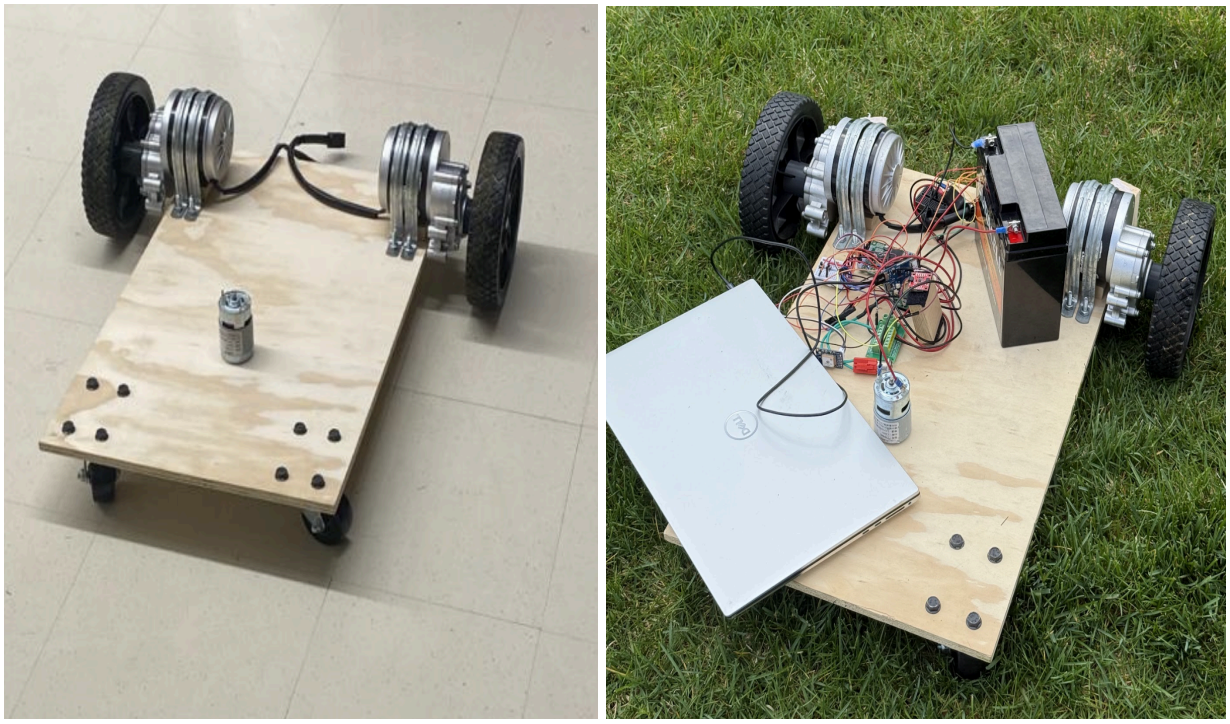


Figure 14: Lawn Mower Physical Design

5.3. Wheel-Motor Interface Adaptation

Another challenge arose when trying to connect the rear wheels directly to the motor shafts. The internal diameter of the wheel hubs did not match the motor shaft size, leading to slippage. To resolve this, we designed and 3D-printed a custom coupling insert that fit tightly between the shaft and the wheel hub.

This adapter acted as a mechanical interface that enabled secure torque transfer and eliminated rotational play. The solution was simple but effective, allowing smooth, consistent motion and maintaining the structural integrity of the drive system.

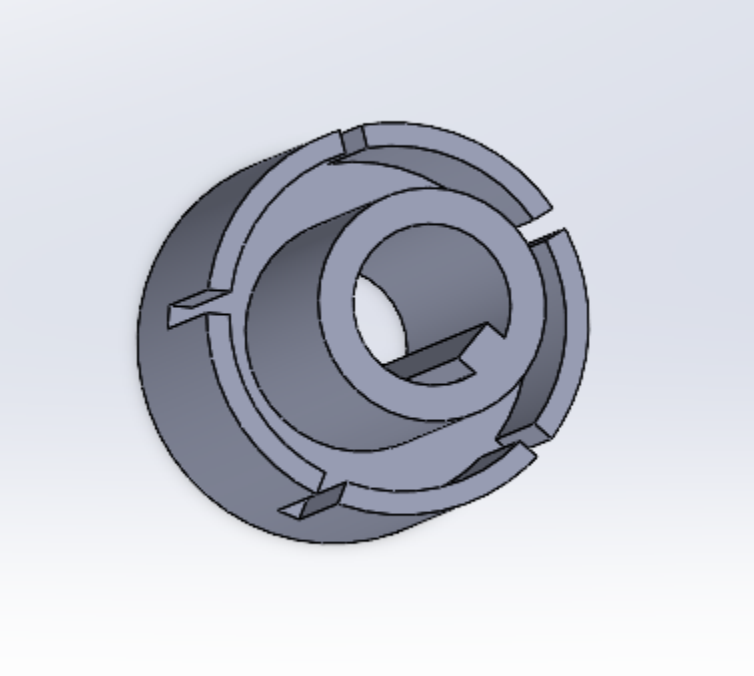


Figure 15: Wheel-Motor Adapter

5.4. Blade Motor Mounting

The blade motor, a high-speed 12V DC 775 unit rated at 10,000 RPM, is centrally mounted on the underside of the chassis. Rather than using a bracket, we secured the motor directly using nuts threaded onto the motor's mounting bolts, which provided a strong and vibration-resistant fit.

Importantly, the cutting blade itself came with all the necessary fasteners and tools for proper installation, ensuring a secure attachment. This direct mounting approach proved reliable and minimized additional hardware, helping streamline the assembly process while maintaining safety.

Figure 16: Cutting Blade Mounting

6. Navigation Development

6.1. Initial Plans

Initially, the autonomous navigation routine was to rely exclusively on GPS. This system would create a grid of GPS points on the lawn and then navigate to each one to go over every part of the lawn. However, most GPS modules give 2-3 meters of inaccuracy. After getting a working code to receive GPS data, we saw that the inaccuracy occurs with every reading, meaning that each new reading could be about 2 meters off, despite remaining in the same location. Despite seeing this inaccuracy with the GPS module, we continued with designing a system relying on GPS, as there were ways to get data accurate to 2 cm with real time kinematic (RTK) GPS. Our priority was to have a working system for the GPS reliant navigation, and if the system showed success given the inaccuracies, we would continue to integrate RTK via Continuously Operating Reference Stations (CORS). Integrating CORS too early could have been a waste of time if the base GPS program did not work.

The first step in creating the GPS reliant navigation system was receiving GPS data. We collected GPS points in decimal degrees for an easier translation to meters. The first main program developed to test the GPS system was called GPS-S3-1_Expanded. This code tested the GPS on its own as a subsystem for navigation. The program had the user save a set of points, then on entering “search mode” performed trigonometry to find the direction the user needed to be heading, which direction the user was heading, and the distance to the previously saved point. When the user reached the saved point, the code would print that it found the point, then direct

the user to the next point. This program showed basic success of being able to navigate to a point, but also demonstrated some issues that needed to be addressed. A screenshot of the serial monitor with this code running in search mode is shown in figure 18.

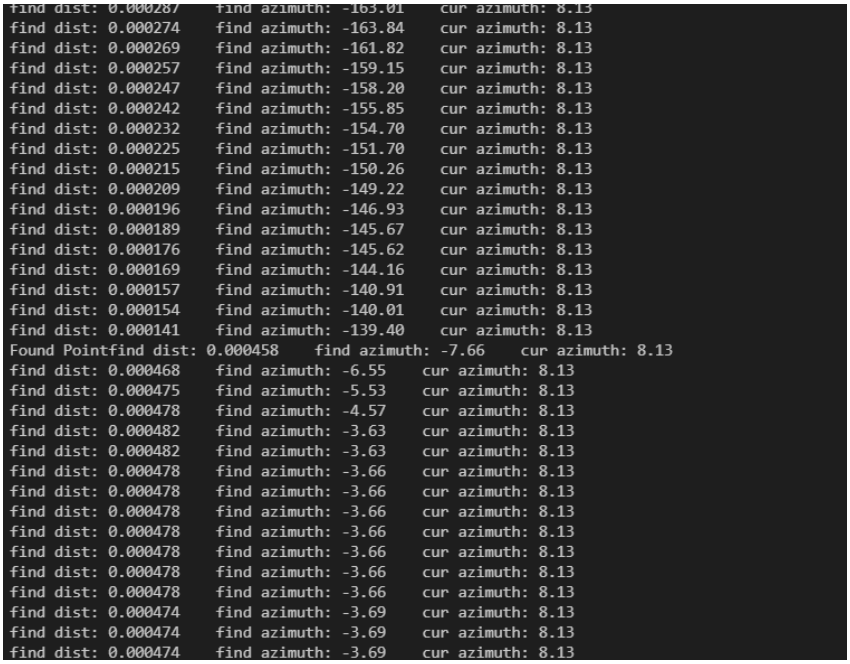


Figure 18. GPS-S3-1_Expanded program serial monitor while searching

In this code, current azimuth is the direction the user is currently headed, calculated by taking the angle of the user's current position and their position 5 seconds ago. Find azimuth is the direction to the target, using the current position and the target point. Distance here is shown in decimal degrees. The problem presented by this code was that current azimuth was unreliable since the 2 meter accuracy of GPS readings could cause significant effects on the calculation of the azimuth. To demonstrate this point, figure 19 shows what could happen with the calculation.

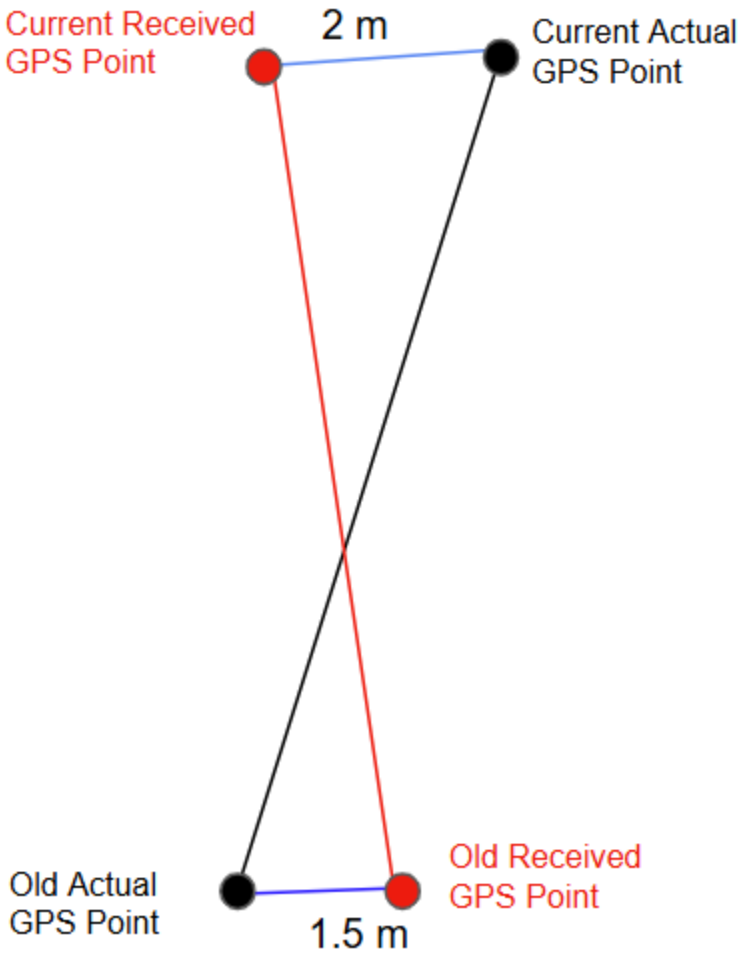


Figure 19. Current azimuth calculation error

As demonstrated in the figure, the GPS reading inaccurate values can produce significantly different angles, leading to current azimuth calculations which were not helpful. Find azimuth values suffered the same effect, but to a smaller effect since the distance to the target point was much more significant, making the 2 m inaccuracy less significant, at least until it got close to the point. At that range, it was more productive to look for when the distance to the target got smaller.

The option to resolve this issue with current azimuth was to lengthen the amount of time between the old point and new point. However, this change would have a negative impact on integration, where the navigation becomes less responsive, needing more time to tell which way it is going, which may have also been impractical for the size of lawns we envisioned.

The next step in the development of this code saw some integration with the driving. The system was tested by having the mower save one point and then drive forward until it found that point. For this testing, the mower was simply pulled straight back so that there was no turning necessary. However, in this testing it became much more obvious that the GPS inaccuracy was too extreme to be practical. When the threshold for saying the point was set so that the mower actually found it and stopped, the mower could be stopped up to 5 meters away. When the threshold was lower, the GPS could miss the point, reading a farther off value when it passed the actual point, and the mower would not stop. At this stage in development, we also discovered that using the CORS network would require buying a new GPS module, costing about $150, which had been out of our budget since early in the project.

6.2. Second Navigation Plan

With relying on GPS being clearly impractical, the plan shifted to integrating the ultrasonic sensors to find pre existing obstacles as reference points. The ultrasonic sensor could detect up to 5 meters, so the GPS needed to know that it was within 5 meters of the point to positively identify the obstacle as the one it saved. To test this new standard for GPS accuracy, we developed the code Dist_GPS. This code allowed the user to press a button to see how far the

GPS thinks it is from the last point the user saved. This code returned the distance in meters rather than decimal degrees for a more clear sense of distance. This code showed that staying still for about a minute and testing the point had the GPS give an error of less than 1 meter. When saving a point, walking away, then returning to the same point, the GPS thought it was up to 5 meters away from the original point. With this error, the ultrasonic sensor detection could work. From Dist_GPS, we saw that we need to keep the GPS still in one place longer to reduce the error and got a sense of the GPS error.

To test the integration of ultrasonics and GPS to find pre saved points, we made Obs_IVO. The user saves a point near an obstacle, then tries to find it with GPS and the ultrasonics, somewhat like GPS-S3-1_Expanded, but the ultrasonics must also see the object to find the point. This code showed decent success, being able to find the point when brought back to the saved obstacle about 80% of the time.

The code Basic_Ultra_IVO integrated the Obs_IVO code into the driving of the mower, which also showed success.

To be able to navigate to the points the IMU would need to be integrated with its capability to tell direction based on the Earth's magnetic field. Using the IMU for direction data came up with the GPS exclusive navigation, but there was no attempt at integration since the angles would be different, coordinate north and magnetic north being in different directions.

6.3. Third Navigation Plan

The shift from the second to third navigation plan was not due to a failure in the concept of the

second, but rather a shift due to time constraints. The third includes many features that would have been in the second, but this third plan just did not include GPS due to a last minute issue with the module where the external antenna no longer connected. While the module should still have worked without the external antenna, the GPS no longer seemed to get a fix, so GPS was dropped since that problem could not be resolved in time.

The matter of finding distance still needed to be resolved, so to get a reference for distance travelled, had the mower go full speed straight forward. The mower consistently went 10 feet in 8 seconds, so this ratio was used as a reference.

The IMU for directional navigation was necessary in the second and the development of it was intended to be in the second plan until the hardware issues with GPS arose. When the mower drove straight in testing, it would often hit a bump and go off track. To fix this issue, the IMU would be used to turn to the correct direction.

This function was created in Turning_Test. The goal IMU was hard coded into the script and the mower would turn to that angle with the IMU data. One issue that presented itself in the development of this function was that when the motors were turning, the magnetic fields from the coils in the motors disturbed the IMU readings. To combat this problem, the motors stop intermittently so as to allow the IMU to get accurate readings of the direction before continuing to find the goal direction. The impact of this effect is that the turning cannot be continuous and the mower must stop to correct the angle, rather than being able to correct while moving, resulting in more robotic motion. The turning code itself is basic, always turning for a set time, either one second or half a second. The motors turn faster if the angle is more than 90 degrees

away, slower if it is between 30 and 90, and slowest if it is less than 30 and more than 10. Within 10 degrees was chosen as the threshold to say the direction was found. The IMU varied by about 1 degree while still, but 10 degrees was chosen since it was small enough to show the mower going in roughly a straight line without having to wait for an excessively long time for the mower to turn to the goal direction.

This turning function was integrated into the full integrated code called JGIS2_RTOS as a part of the autonomous routine. Instead of saving a GPS point, the navigation code saved an IMU direction. Upon starting search mode, the autonomous routine had the mower go forward for about 5 feet, determined by the timing of the motors, then stop and use the turning function to correct the angle. It then continued forward and checked again. This would continue for a hard coded number of iterations before turning around to the right, by turning right, then right again, placing itself to the previously completed strip. At this point, it would correct the distance again, but now the goal direction was 180 degrees from the original saved direction to face the other way. The mower would repeat the straight and correction movement until it turned around to the left. This routine would produce a box shape of strips going all the way across the box.

6.4. Obstacle Avoidance

The obstacle avoidance subroutine was developed in the code project named Basic_Movement. This used the ultrasonic sensors to detect an obstacle and timing of the motors to get around the obstacle. The routine first had the motor see an obstacle as detected by the ultrasonic sensor in front of the mower. The mower would then stop, turn to the left, then drive forward, then turn left. If the mower still detected an obstacle, the mower would repeat the turn left, forward, then

right to attempt to get around the obstacle. If it was clear, the mower would drive forward and box around the obstacle, returning back on track. Issues that arose with this routine had to do with the placement of the sensors. Originally, the sensors were mounted on the board, level with the rest of wiring and components. However, the ultrasonic sensors kept detecting the grass, making the mower think there was an object in front of it. This was first adjusted by changing the angle of the ultrasonics upward. However, during later testing the grass was longer and the ultrasonics saw the grass again. To resolve this issue, the ultrasonics were placed on blocks in front of the mower to lift them up, seen in figure 20. During this process, the original ultrasonics broke with hot glue being used to secure them. When attempting to readjust the original ultrasonic sensors, the hot glue ripped out a few components, breaking the ultrasonics. To get a working system, we found a HC-SR04 sensor to use for the mower. This routine did not see integration into JGIS2_RTOS due to an unexpected issue with GPIO pins, where setting the ultrasonic pins resulted in an error related to the usb pins.

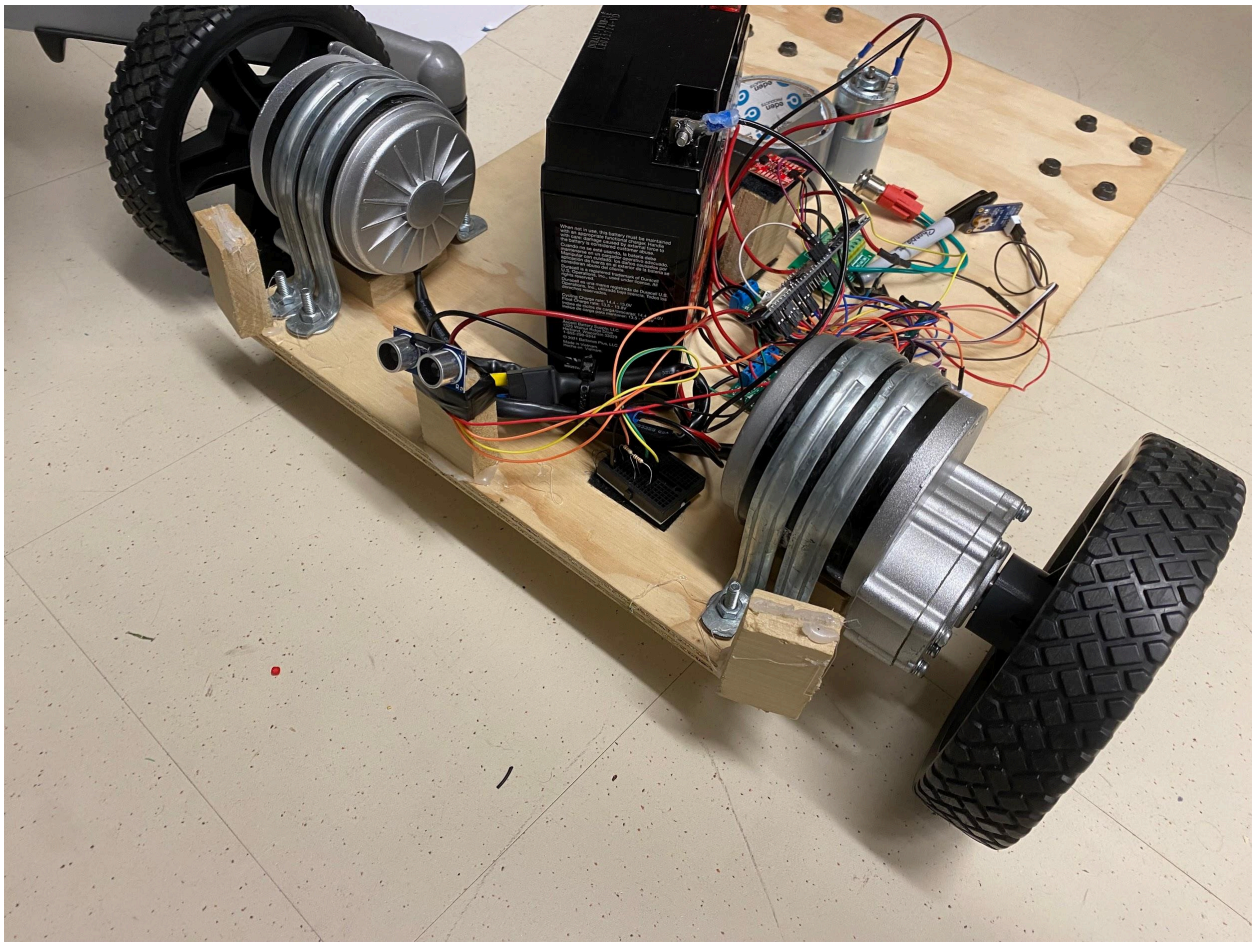


Figure 20. Ultrasonic placement blocks

7. Bluetooth Development

7.1. First BLE Apps

On the ESP32 side, we used the NIMBLE library to have BLE communication from the ESP32. Early codes sent integers to the ESP32 to turn on the on board LED. The transmission of BLE was observed on the app nRF Connect for debugging purposes. The first BLE apps developed for

this project were made on MIT App Inventor as it was a simple way to get a BLE app. While starting on MIT App Inventor was easy, the block code quickly became tedious and did not offer much functionality and reading from the ESP32 became too complicated with byte management. Seeing that MATLAB had BLE capabilities, we wrote some short code which called the same LED functions and wrote integers to the ESP32, but reading from the ESP32 remained complicated. In addition, using a MATLAB app for the project seemed impractical.

7.2. iOS BLE App

Seeing iOS apps as an option, we switched to coding in Swift on XCode to create iOS apps to interact with the ESP32 over BLE. This option proved to be difficult initially, taking up a significant amount of time to create a basic working app, but did offer the robustness needed for our project. Initial apps were still sending integers to call certain functions within the ESP32 code. The first iOS app that saw integration with other systems was 4Dir, which was a 5 button app which told the mower which way to go. The app interface is shown in figure 21.

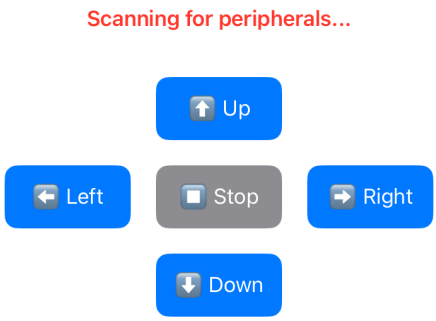


Figure 21. Four direction IOS app

After getting a baseline understanding of sending integers over BLE, it was simple to call functions in the ESP32 code, where a certain integer on a characteristic went to a switch statement to call a certain function. Writing from the ESP32 to the app was not a far step from writing to the ESP32, so we were able to send status updates from the ESP32 to the app in the form of strings. From 4Dir, we developed Joy1, which could control the motion of the mower with a joystick. Further, we created JGIS, which integrated the joystick, buttons to save, turn on searching, toggle the blade, and read status updates. The interface for JGIS is shown in figure 22. This JGIS app saw integration into the JGIS2_RTOS code, allowing the user to control the mower via remote control and use the autonomous routine. After spending time with Swift in XCode, debugging and adding more functionality became easy despite long hours getting it started.

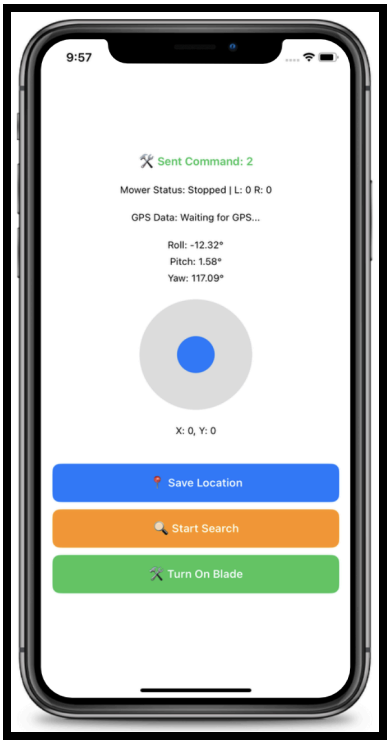


Figure 22. JGIS IOS app

8. User Manual

As the product stands, the user should download and run the folder JGIS_Updated on XCode on a Mac connected to their IOS device. This will install the app JGIS to their phone. The user should upload the file JGIS2_RTOS to the ESP32-S3. For full functionality, the user will open the JGIS app on their phone and if the ESP32 is in range, it will automatically connect.

To start the product's autonomous program, the user should set the mower down in the direction they want it to start. They will then either press the button corresponding to saving the point on the hardware or on the app. This will save the direction the mower is facing. To start the autonomous routine, the user will press the button corresponding to starting the search on the

hardware or on the app. This action will start the autonomous routine. At this point, the mower will start and perform the autonomous routine on its own. The user can see the current status of the mower, such as the speed, with the app above the joystick.

If the user wants to use the remote control feature, they can interact with the joystick on the app. This will turn off search mode if it is on and BLE control will stop if search mode is turned back on.

Issues that may arise will probably have to do with the direction of the mower. This issue can be solved by adjusting the threshold for what is forward within the JGIS2_RTOS code.

9. Changes and Future Development

9.1. Navigation and Obstacle Detection

In terms of navigation, this project saw significant changes without much time to make complete working systems that would be ready for a market setting. Within a short period of time, on the order of weeks and with current equipment, we would see the reintroduction of the second navigation system. The GPS is important for knowing the location of the mower and using that system would give the GPS more functionality for being able to autonomously navigate. Instead of seeing the values for the distance to mow before turning around being hardcoded, they would be based on the GPS readings, giving an approximate distance. Other options for recording the distance is timing how long the mower takes to reach the end or solely using pre existing obstacles for finding the edge of the lawn. Using pre-existing obstacles could also give the option for non-square patterns. Additionally, the obstacle would see integration into autonomous

routine, after resolving the GPIO issue.

With additional resources, it would be useful to acquire a CORS capable module, to integrate the original navigation system. This system would require less complexity in terms of coordination of sensors and give a better sense of where the mower is at all times. For motors, having encoders would be useful, giving some redundancy so that the mower can get accurate distance without relying solely on GPS data.

9.2. Physical Build

The current physical build could see improvement with weather proofing. It has been a prototype-like design for the duration of this project as the focus was on a working autonomous program rather than a perfect physical design. Further, the blade being adjustable would allow for different heights to cut the grass. The wheels could be changed to give better friction on the grass where the current wheels slip without much weight from our mower.

9.3. Bluetooth App

The BLE app serves its purpose in this project but could see improvement in terms of presentation. The app is basic and lacks a good aesthetic, something that could be improved in the future. Instead of everything being on one screen, the app could be improved by allowing the user to swipe between a status page, showing data from the sensors, and a control page, showing the joystick and control buttons for the user to control the mower.

10. Conclusions

The purpose of the autonomous lawn mower project was to demonstrate it as a working concept. This project did not meet the original expectations we had for it. A significant reason for this result was that there were necessary changes that occurred within short periods of time, not allowing for the full development of important subsystems. To have a more complete working product, we needed to start with more redundancy. We planned on the GPS system working, so it was not necessary to have things such as encoders in the motors, but it happened that it would have been useful. Through the changes in the navigation system, we showed that it was possible to have an autonomous routine with the sensors that were not intended to be used for navigation.

On a basic level, this product demonstrated that an autonomous system could work with the available materials and resources. However, time constraints and other difficulties limited progress. With work already done here, it will be possible to continue this system to have a working and productive autonomous lawn mower.

11. Appendices

11.1. References

Horsey, Julian. “DIY Arduino Robot Lawnmower (Video).” *Geeky Gadgets*, 3 Aug. 2024, www.geeky-gadgets.com/diy-arduino-robot-lawnmower-24-08-2017/.

“Arduino Controlled Robot Lawnmower - Ultrasonic and RGB Sensors.” *Electromaker.Io - Unleash Your Creativity with DIY Electronics & Maker Projects*, www.electromaker.io/project/view/arduino-controlled-robot-lawnmower---ultrasonic-and-rgb-sensors. Accessed 7 Feb. 2025.

11.2. Important datasheets

1. IMU Sensor: SparkFun 9DoF IMU Breakout – ICM-20948
2. Motor Drivers:
    - Pololu Dual VNH5019 Motor Driver Carrier (for drive motors)
    - Pololu Single VNH5019 Motor Driver Carrier (for blade motor)
3. GPS Module: Adafruit Ultimate GPS Breakout
4. Ultrasonic Sensor: URM09

11.3. Technical Files

Platformio Code

PCB Files

App Files

11.4. Video Demonstration

Video Demo