ECE 445

Senior Design Laboratory

Final Design

Antweight Battlebot

Team number 15
Daniel Rodriguez
Carlos Carretero

5/7/2025
TA: John Li
Professor: Viktor Gruev

Abstract

This project involved designing and building an Antweight BattleBot for Professor Gruev's Battlebot competition. The bot was composed of five main subsystems: Control, Drivetrain, Power, Chassis, and Weapon. It used an ESP32 for wireless control, L298N motor drivers with N20 brushed DC motors for movement, and a top-mounted spinning blade as its weapon. The chassis and components were 3D printed in PLA to meet the 2-pound weight limit. The final design achieved wireless functionality and basic mobility, though drivetrain performance was slightly below target specifications.

Introduction

The goal of our project was to design and build an Antweight BattleBot that meets the requirements of Professor Gruev's Battlebot competition. Our robot was intended to be compact, fully wireless, and agile, capable of navigating the arena and engaging with other bots using an active weapon system. Figure 1 shows a high-level block diagram of the five main subsystems that define our design: Control, Drivetrain, Power, Chassis, and Weapon. Each subsystem was developed to meet specific performance benchmarks, including a total weight under 2 pounds, a WiFi signal strength between -50 dBm and -60 dBm, a forward velocity of 4.5 ft/s, and a wheel rotation speed of 900 RPM.

Our project was guided by a strict set of rules and requirements outlined by the Antweight BattleBot competition. These included a maximum total weight of 2 pounds and the exclusive use of 3D-printed parts for both the chassis and weapon, limited to materials such as PET, PETG, ABS, or PLA/PLA+. The robot had to be wirelessly controlled via a Bluetooth or Wi-Fi-enabled microcontroller and include a visible LED indicator showing power status. The battery voltage was not allowed to exceed 16 volts, and a manual disconnect was required for safety.

Originally, we planned to use the MCF8316A brushless motor driver to control more efficient and powerful BLDC motors. However, due to complications in our custom PCB design, we were unable to integrate the MCF8316A and pivoted to using L298N drivers with N20 brushed DC motors instead. This change reduced drivetrain performance, resulting in a measured top speed of 3.3 ft/s and a maximum wheel RPM of about 469. The Control subsystem used an ESP32 microcontroller for real-time wireless operation, while dual 7.4V 500mAh batteries powered both motors and electronics. A voltage regulator provided 3.3V to the ESP32. The chassis and weapon components were 3D printed in PLA to reduce weight and allow for accurate modeling. Although we encountered design limitations, particularly with our PCB and motor selection, we successfully implemented a functioning battlebot that satisfied key safety, communication, and modular design requirements.

High-level requirements list

- The battle bot must not exceed 2 lbs with all required components for proper functionality. This includes motors, wheels, chassis, weapons, and onboard electronics.
- A strong wifi connection between -50 dBm to -60 dBm must be established, allowing for consistent and responsive movements from the input device.
- Competent movement with
    - Velocity of 4.5 feet/s
    - Wheel RPM of 900 revolutions per minute
    - Independent bidirectional wheel movement

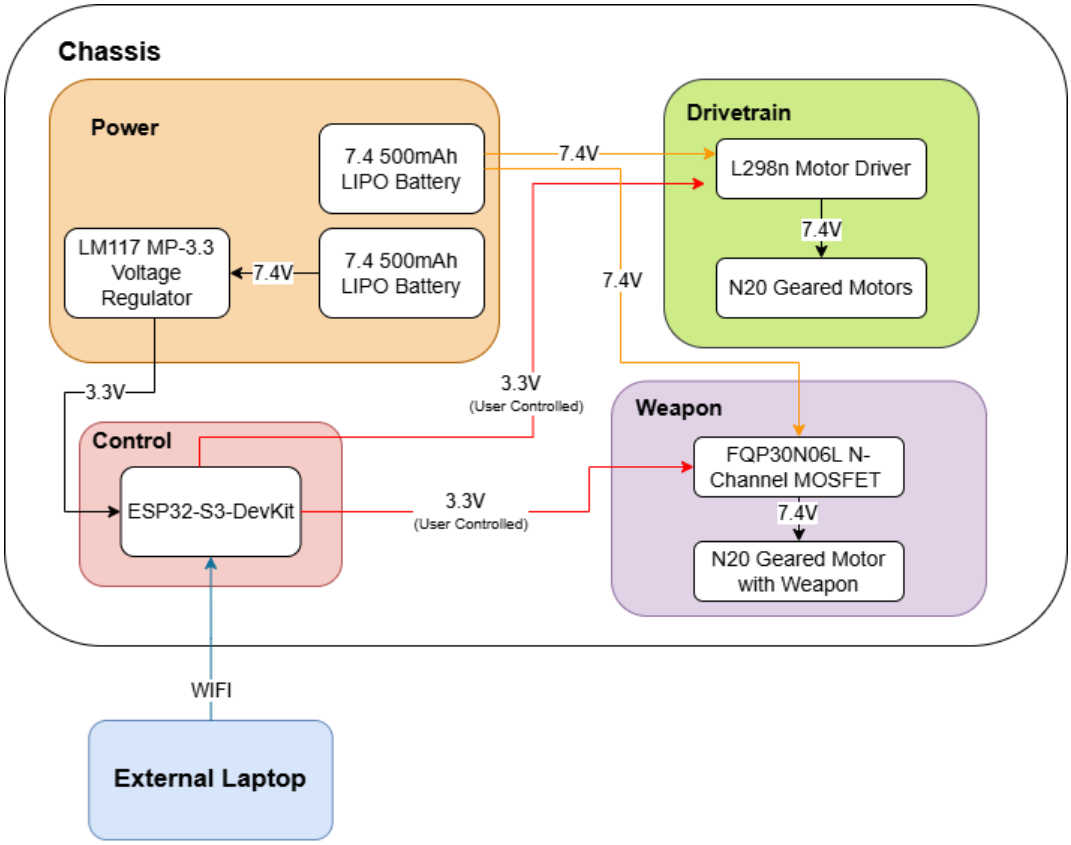

Figure 1: Battlebot Diagram

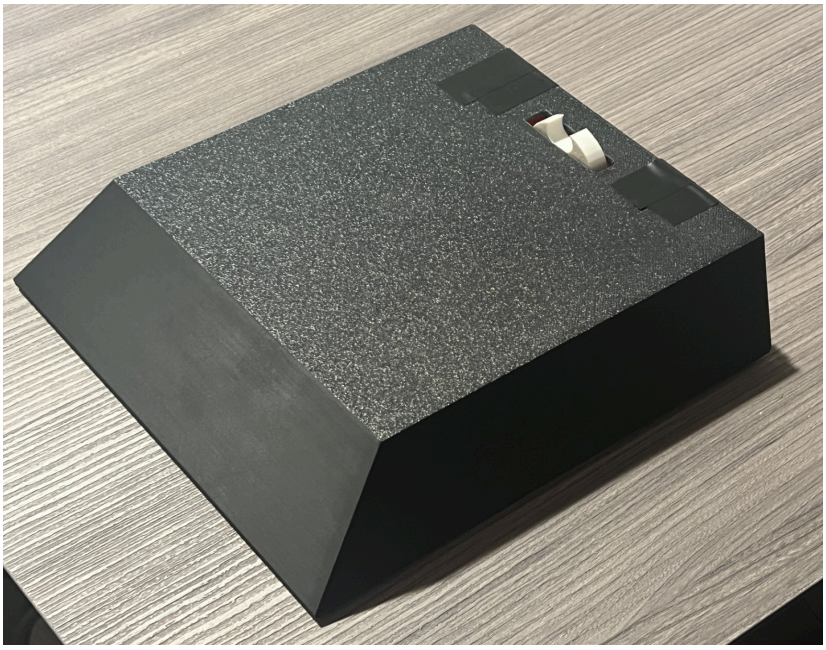

Figure 2: Battlebot final design

Design

For our battle bot to be competent in competition, creating a strong defense would be more important than having a glass cannon. Therefore, for our design, we decided upon a wedge-shaped bot that is as low to the ground as possible to ensure it can't be flipped over. For our weapon, we designed a spinning blade on the back of the chassis that will damage anything that gets on top of it. The battlebot is controlled by a PC that communicates with the ESP32 over wifi. We designed the chassis, wheels, and axles using 3D modeling software and 3D printed them using PLA filament. To move our battle bot, a pair of brushed DC motors is driven using a half-bridge L298n Motor driver, which is controlled by the ESP32. All of this is powered by 2 7.4V LiPO batteries, with one of them stepped down to 3.3V using an LDO voltage regulator for the control subsystem. The LDO was soldered onto our custom PCB, and this is also where both manual disconnects for the batteries are located.

Chassis

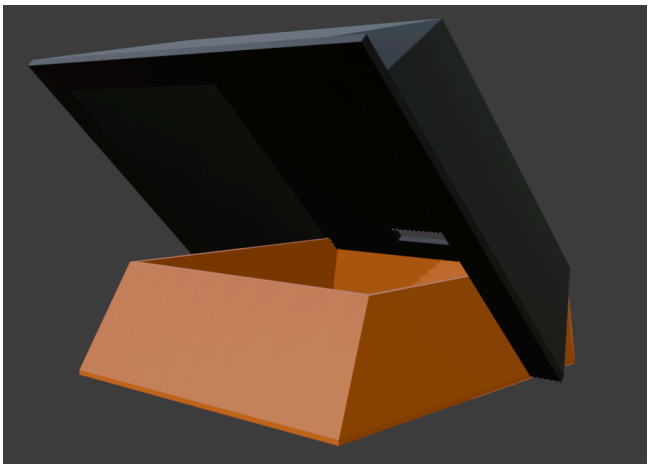

Figure 3: Battlebot 3D Model

Design Procedure

Our chassis had to be 3D printed as it is required by the Antweight BattleBot competition rules. Among several filament options like PETG, ABS, and PLA+, we selected PLA due to its availability in the lab, low cost, high dimensional accuracy, and sufficient strength for prototyping. It was also ideal for our need to create precise fits for mounting internal components. Our primary goals for the chassis were to make it strong enough to survive collisions, ensure it could house all electrical components, and meet the 2-pound weight limit. One key design feature we originally aimed for was a 30-degree front wedge, intended to allow

our bot to deflect or slide under opponents. The model was created using Blender, a free and open-source 3D modeling tool, and was used to render the designs, as shown in Figure 3. Additionally, we had to account for the 225 mm × 225 mm × 225 mm build volume of the Bambu Lab X1 printer, which limited the maximum size of each printed part. This constraint influenced how we split the chassis into multiple pieces.

Design Details

The final chassis consists of two separate PLA-printed pieces: a top shell and a bottom shell that are connected at one edge with a hinge mechanism to allow internal access. The outer dimensions of the chassis are 22 cm × 21.5 cm × 7.5 cm, sized to comfortably fit all major components, including an ESP32 Dev Kit, L298N motor driver, two 7.4V 500 mAh LiPo batteries, and our custom PCB power distribution board. It also accommodates the drive and weapon motors. The design includes a steep front wedge, which was originally planned to be 30 degrees, but due to geometry constraints in the final version, the actual angle became 57.43 degrees, as verified through measurement. The rear wheels and axles were printed as separate components and attached using hot glue to allow free spinning. The wheels were chosen to be 41mm in diameter as we needed room for 4 wheels and did not want the wheels to have too much mass. The axles passed through pre-modeled slots in the chassis base to ensure alignment. All fit tolerances were maintained within ±0.5 mm to ensure proper assembly without post-processing. The selection of PLA and the design's segmented structure provided the durability needed to withstand battle impacts while staying under the competition’s weight and size constraints.

Drivetrain

Design Procedure

The drivetrain subsystem was designed to meet four main performance criteria: achieve 900 RPM motor speed, avoid voltage spikes from the power source, bidirectional wheel movement, allow the motors to brake and change direction effectively, be able to do a minimum of 4.5 ft/s and be initializable via I2C using SCL and SDA signals. Initially, we planned to use brushless motors controlled by the MCF8316A driver, as these offered higher efficiency, smoother rotation, and I2C compatibility. However, due to challenges in designing a reliable custom PCB for this driver, we switched to a simpler and more robust solution. We chose the L298N motor driver, shown in Figure 4, because it is capable of driving two DC motors bidirectionally, supports up to 2A output current, and requires only digital input signals, avoiding the need for complex I2C initialization. This decision prioritized ease of integration, reliability, and compatibility with our limited motor power requirements.

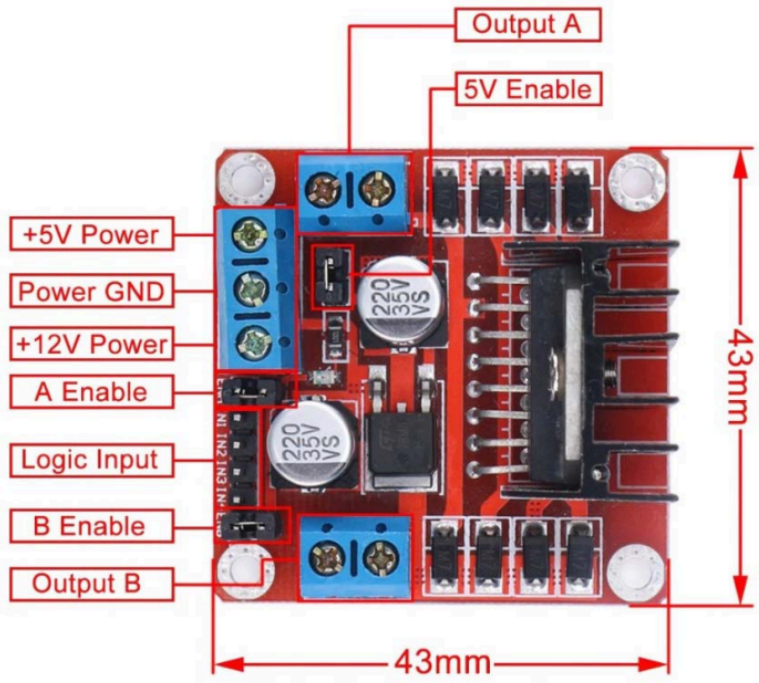

Figure 4: Motor Driver Circuit Diagram

Design Details

The final drivetrain configuration used two N20 brushed DC motors rated for 6V and 1000 RPM with no load. Estimating that the motors would spin at a reasonable 800 rpm with a load, it was calculated, as shown in Figure 5, that the 41mm wheels would allow the car to move at a speed of 5.633 feet per second. This would be 1.133 feet per second above our high-level requirement of 4.5 feet per second, and therefore made the N20 motors appear like a viable option for this project. The motors were controlled by a single L298N dual H-bridge motor driver. We chose this motor driver because it could be activated with a minimum of 2.3V, which was less than the 3.3V the ESP32 could output. Each motor was connected to its respective output channel, and digital input pins from the ESP32 controlled direction and speed via pulse-width modulation (PWM). We originally intended to use I2C to initialize the motor driver, but this requirement became irrelevant due to the L298N using standard logic-level inputs.

$$Wheel\ Circumference = \pi * Wheel\ Diameter = \pi * 41mm = 0.1288m$$

$$Estimated\ Car\ Speed = RPM * Wheel\ Circumference * \frac{3.28}{60} = 5.633\ feet/s$$

Figure 5: Estimated Car Speed

Power

Design Procedure

The power subsystem was conceived at the very beginning of our design choices. Due to the bot having to be completely wireless, we had to be able to power every other subsystem on board. Because of this, we had to be able to meet all the current and voltage requirements of the components of our other subsystems. To achieve a complete wireless design, we designed a PCB that pinned out the batteries onto 3 rails. For 2 out of the 3 rails, we simply kept the voltage at 7.4V. The third rail had a voltage of 3.3V with the help of an LM1117MP-3.3 voltage regulator. The control subsystem, which is made up of the ESP32, needed the 3.3V found on this rail to power on. The rest of our subsystems were powered using the other two rails at the full 7.4V. These subsystems were able to run at 7.4V as they were using motors that were rated for 6V. Due to voltage drops across the motor drivers and MOSFETs used to control these motors, the slightly higher input voltage was beneficial for running these motors at their rated voltages.

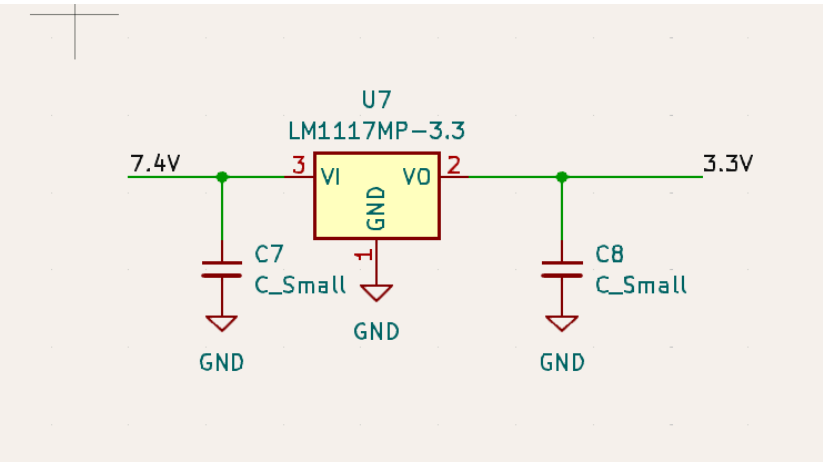

Figure 6: Voltage Regulator Circuit Diagram

Design Details

To power all the motors and the L298n motor driver, we are using a 7.4V 2s LiPO battery with a capacity of 500mAh. Our ESP32-WROOM-1 microcontroller requires a voltage of 3.3V and a minimum operating current of 80mA to 90mA. We use a second 7.4V 2S LiPo battery, which needs to be stepped down to 3.3V. Therefore, the voltage regulator we chose was the LM1117MP-3.3, as it has a max input voltage of 15V, which is almost twice our battery voltage to allow some overhead when the battery is at full charge. We also chose it as it can output a max current of 800mA, which is much higher than required for ESP32 operation.

$$Max\ Current\ Per\ Battery = C_{rate} * Capacity = 30C * 500mAh = 15A$$

Figure 7: Maximum Current Per Battery

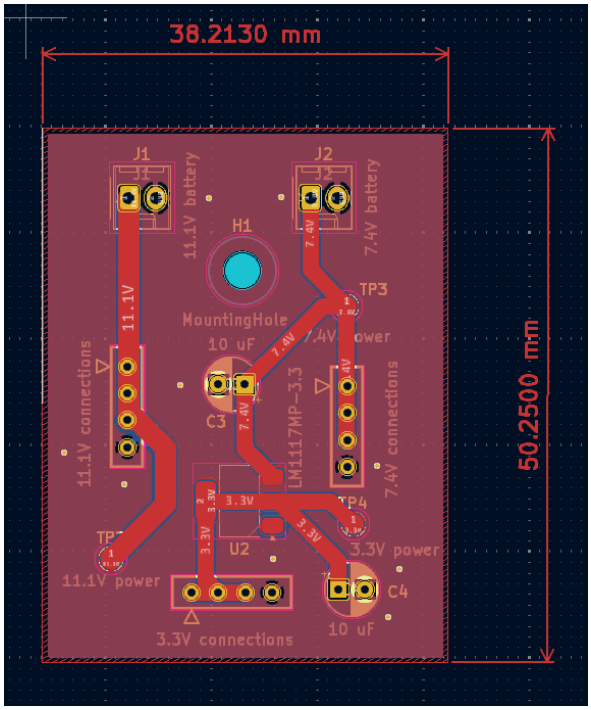

Figure 8: Final PCB Design

Control

Design Procedure

For this subsystem, our main concern was ensuring that the connection strength between our microcontroller and the laptop sending movement inputs was stable. Our high-level Requirement of Wifi connection strength between -50 dBm and -60 dBm was the main influence for this subsystem. Initially, we were going to use an ESP32 S3 standalone microcontroller that would be soldered onto our custom PCB. Due to the issues we ran into with our onboard programming circuit, we had to ultimately use an ESP32-S3-DevKitC-1-N8R8 Development Board to handle our control subsystem. This dev kit was powered by the 3.3V rail from our power subsystem. The other half of this subsystem was software-based and split into two parts: one being the Arduino code flashed onto the dev kit and the other being the Python code to communicate with the microcontroller over the same wifi network. This code established a shared wifi network that the PC and microcontroller could communicate across, and also sent the key inputs from our keyboard to the microcontroller. Each key sets a different pair of pins from the microcontroller to be 3.3V to control which motor should spin and in which direction. The weapon subsystem was also controlled through this code.

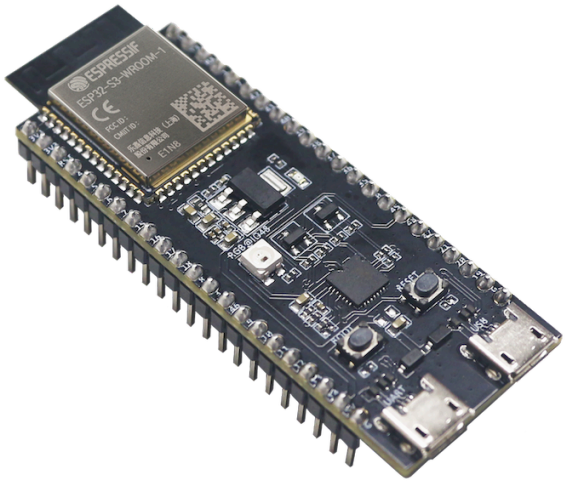

Figure 9: ESP32-S3-DevKitC-1

Design Details

Due to the control subsystem using an ESP32-S3-DevKitC-1, most of the design details are determined by the specifications of the manufacturer of this dev kit. This dev board, like any ESP32, still required 3.3V to power on and was provided by our power subsystem. To flash this device, the onboard USB to UART programming chip allowed us to connect this to a computer over micro USB and write the used Arduino code to its memory cache. To control the motors for our drivetrain subsystem and the weapon subsystem, wires were connected to the pins on the dev kit and then to the correct location in each of the two subsystems. The pins we used were chosen in our code, but in total, 5 pins were used. Two were used to control our left wheel, and two more were used to control the right wheel in both directions. The final pin was tied to the signal needed to activate our weapon using the gate of the MOSFET in that subsystem. Finally, the onboard antenna found on the specific ESP32-S3-WROOM-1 that is on this dev kit allowed us to communicate over wifi seamlessly.

Weapon

Design Procedure

Since our weapon only needed to spin in one direction, the main influence on this subsystem's design was ensuring that we could turn the motor on and off with a signal from the control subsystem. We used another N20 brushed motor to spin our weapon; however, this motor was only rated for 600 RPM, which we chose on purpose as the lower RPM rating ensured a higher torque. The way we ensured that the motor was getting powered only when an on signal was sent from the ESP32 was by using an N-channel MOSFET. This MOSFET would allow the 7.4V

battery to directly power the motor only when the gate received a signal from the ESP32, as the 3.3V signal was higher than the required 2.5V gate voltage. The weapon was designed to be mounted to the top of our car and intended to damage opponents that found themselves victims of the sloped sides of our BattleBot.

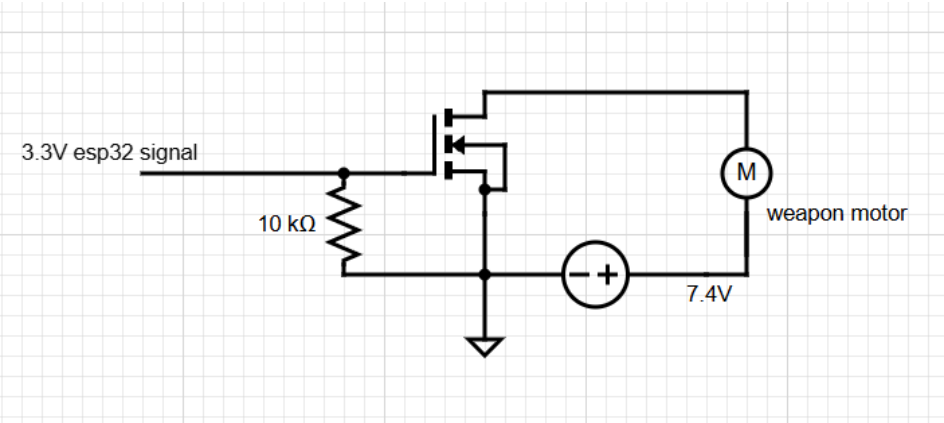

Figure 10: Weapon Wiring Diagram

Design Details
We used a FQP30N06L N-Channel MOSFET due to its low 2.5V gate turn-on voltage as well as its 60V max voltage and 32A max current. We allowed a lot of overhead for this subsystem component as we were connecting the N20 motor directly to the battery, apart from the 10K pull-down resistor between the gate and the source. We also chose this specific MOSFET as it had a fast switching time, and also a drain-source diode that protects the ESP32 from back current spikes caused by the motor. The spinning blade was designed using Blender and 3D printed using PLA plastic. The blade spins at roughly 400 RPM with a diameter of 38mm.

Verification

Chassis

The chassis subsystem was tested against four key requirements: a total weight under 2 pounds, structural durability, compatibility with the custom PCB, and inclusion of a front wedge with at least a 30-degree slope. Weight verification was performed using a digital scale. The fully assembled chassis, including wheels and axles, weighed 472.95 grams, leaving sufficient headroom within the 2-pound (907g) total limit to accommodate the remaining subsystems. To evaluate structural durability, we performed a repeated drop test from a height of 0.5 meters onto a hard surface. After ten consecutive drops, the chassis showed no fractures, cracks, or loosening of structural connections. All mounting points remained aligned, and the hinge mechanism

continued to operate without issue. These results indicate that the chassis was structurally sound and able to withstand impacts typical in battle conditions. Fit verification with the PCB was conducted by placing the custom board inside the chassis. All standoff holes aligned correctly with the board, and there was adequate clearance around components and wiring. This confirmed that the chassis met the requirement of being able to securely house the PCB without modification.

The only requirement not fulfilled was for the front wedge of the robot to have a minimum slope of 30 degrees. As shown in Figure 11, the final design resulted in a calculated slope of 57.43 degrees. This exceeded the requirement due to the front wheels in our design being pushed to be more forward. This, however, did not limit the battlebot's movement and still allowed for cars to potentially get about ours to interact with our top-mounted weapon.

$$Front\ Slope\ Angle = arctan(\frac{height}{slope\ protrusion}) = arctan(\frac{72mm}{46mm}) = 57.4259^{o}$$

Figure 11: Front Slope Angle Calculation

Drivetrain

The drivetrain subsystem was evaluated based on motor speed, voltage stability, braking performance, direction control, and motor driver communication. Some aspects of the system functioned as intended and were successfully verified. The L298N motor driver operated reliably with the 7.4V battery. During extended operation, neither the battery nor the driver experienced significant heating, and no voltage spikes were observed. This confirmed that the motor driver maintained voltage stability and did not draw damaging transient currents. The robot also demonstrated consistent forward motion. When measured over a 4.5-foot distance, the robot achieved a velocity of 3.3 feet per second. Based on the wheel diameter of 41 millimeters, the calculated motor speed was approximately 469 revolutions per minute, as calculated in Figure 12. While this was below the original goal of 900 revolutions per minute, it was sufficient to confirm forward movement and basic functionality.

$$RPM = \frac{Velocity}{Wheel\ Circumference} * 60 = \frac{1.006m/s}{0.1288m} * 60 = 468.634\ revolutions\ per\ minute$$

Figure 12: Final Wheel RPM Calculations

However, not all drivetrain requirements were met. The N20 brushed DC motors, which each drew approximately 35 milliamps under normal conditions, did not produce enough torque to support effective turning or directional changes. The robot was unable to pivot or rotate while

stationary and required forward momentum to turn. This significantly limited maneuverability and made it difficult to control the bot during matches. Additionally, the braking requirement, which specified stopping within 1 foot when moving at 4 feet per second, was not achieved. Without active braking, the bot coasted to a stop over a longer distance. These limitations indicate the need for more powerful motors and enhanced control strategies in future designs. The original design also included a requirement for motor driver initialization using I2C via SCL and SDA signals. This requirement became irrelevant after switching from the initially planned brushless motor drivers to the L298N, which uses basic digital inputs instead of I2C communication. As a result, this requirement did not apply to the final implementation.

Power

The power subsystem was designed to provide stable and sufficient voltage and current to the ESP32 microcontroller, drivetrain motors, and weapon motor, while also meeting the safety requirements of the competition. Each battery and voltage regulation component was tested against performance specifications to verify functionality.

The LM1117MP-3.3 voltage regulator successfully stepped down the 7.4V input from Battery 1 to a stable 3.3V output, satisfying the requirement for a constant regulated voltage to power the ESP32. The regulator was rated to supply up to 800 milliamps, and our measurements showed the ESP32 only drew a maximum of approximately 120 milliamps during peak wireless activity, as reflected in Figure 13 in the section labeled "ESP32 + WiFi + Signals." Therefore, the first three power requirements, constant 3.3V output, voltage step-down, and up to 0.8A supply, were fully satisfied.

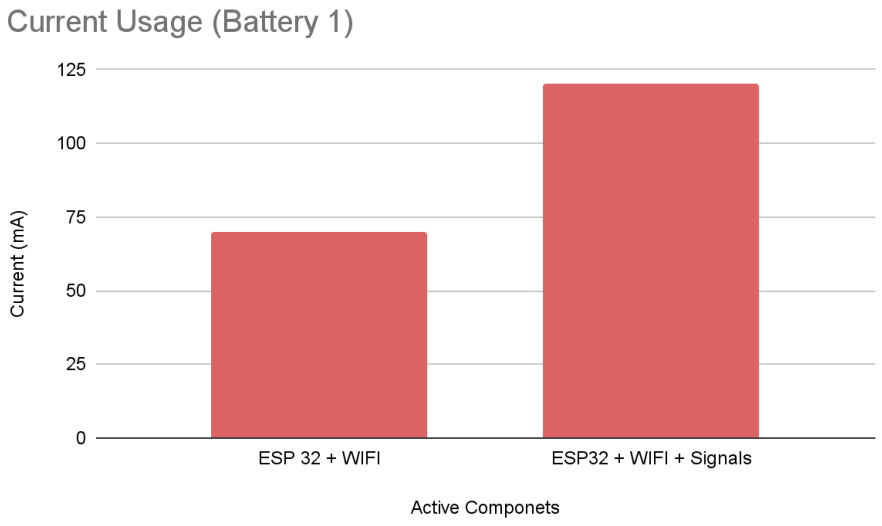

Figure 13: Current Usage in Battery 1

The maximum current draw per battery was calculated in Figure 7 as 15A, confirming that each battery was capable of supporting significantly more current than was required. However, our final configuration did not come close to this draw. As shown in Figure 14, the drive motors drew 90 mA sustained, spiking to 111 mA, while the weapon motor drew 32 mA sustained, occasionally spiking to 60 mA. Combined, the maximum current draw was 122 mA sustained and 171 mA at peak. This is well below the L298N motor driver's 2A capacity and far below the battery's 15A discharge capability.

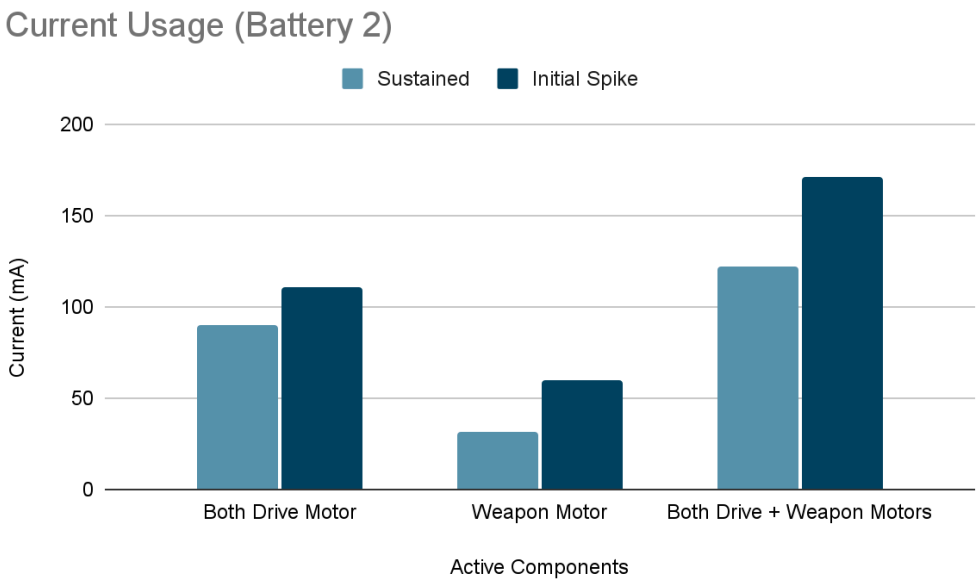

Figure 14: Current Usage in Battery 2

Using the actual current values, we also calculated the maximum run time for each battery. For Battery 1 powering the ESP32, the expected runtime was 2.882 hours, calculated in Figure 15, and for Battery 2 powering both the drivetrain and weapon, the runtime was 4.141 hours, calculated in Figure 16. Given that matches in the competition can only last up to 3 minutes, these results confirmed that power availability was more than sufficient, and the system was capable of running for extended durations without the need for recharging. This further highlighted that our motor selection was overly conservative, and that stronger motors could have been used without exceeding power constraints.

$$Battery\ 1\ Runtime = \frac{Energy}{Power} = \frac{7.4V*500mAh}{4.1V*120mA+3.3V*120mAh} = 2.882\ hours$$

Figure 15: Battery 1 Runtime Calculations

$$Battery\ 2\ Runtime = \frac{Energy}{Power} = \frac{7.4V*500mAh}{7.4V*90mA+6.5V*35mA} = 4.141\ hours$$

Figure 16: Battery 2 Runtime Calculations

Both batteries included manual disconnect switches, complying with competition safety rules. Each battery operated at 7.4V, which is well below the 16V maximum competition limit. Additionally, multiple runs showed that disconnecting either battery left only minimal residual current, which safely dissipated without damaging any components or causing unexpected behavior.

Control

The control subsystem was evaluated based on wireless communication reliability, microcontroller stability, signal transmission, input lag, and voltage behavior under load. The ESP32 successfully connected to the laptop over WiFi and maintained communication throughout all trials. The connection remains active during long periods of operation, with no observed disconnections, resets, or reboots. This confirms that the ESP32 did not reset or fail under multitasking conditions, satisfying our requirement. The voltage levels at the output pins of the ESP32 were measured at a consistent 3.3V, which was sufficient to drive both the motor drivers and the MOSFET controlling the weapon subsystem. These outputs remained stable during all tests, confirming that no significant voltage spikes occurred under heavy load.

The laptop was able to communicate wirelessly with the ESP32, receiving data and sending control signals to the motors and weapon. The graph in Figure 17, labeled Connection Strength (dBm) vs. Time (s), shows that the signal strength remained between -50 dBm and -60 dBm, which met our high-level requirement for maintaining strong wireless communication over a typical 10-foot range. These connection numbers were recorded by making the PC print out the dBm signal strength to the terminal every 10 seconds. This range provided a stable connection without delay or signal loss. Input commands from the PC were also processed with minimal latency, and the system consistently responded with less than 0.5 seconds of lag between user input and bot movement. This was confirmed through real-time testing of the bot's driving and weapon activation, which consistently responded immediately to PC input.

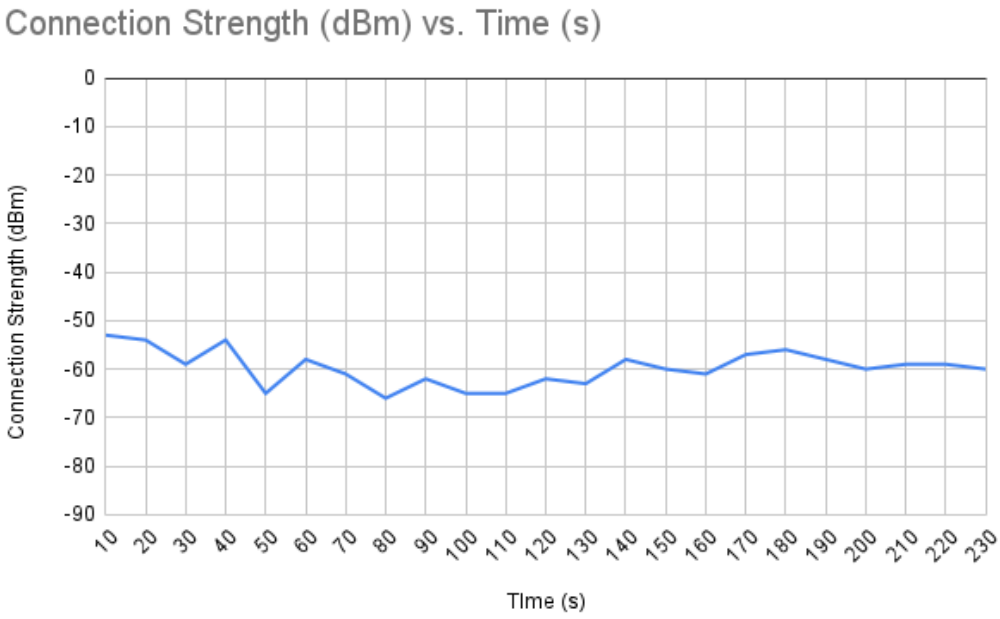

Figure 17: Connection Strength over Time

Weapon

The weapon subsystem was tested to verify activation timing, mechanical alignment, structural integrity, thermal behavior, and safe operating voltage. All electrical and performance-based requirements were satisfied during testing, though one mechanical constraint was altered due to design changes.

The weapon motor, controlled through a MOSFET switch activated by the ESP32, consistently responded to input in under 0.5 seconds. The ESP32 outputs a stable 3.3V control signal to the MOSFET gate, allowing the weapon motor to turn on and off quickly and reliably. During operation, the weapon system was able to maintain over 400 revolutions per minute, significantly exceeding the original requirement of 100 RPM. It remained stable at this speed for more than 3 continuous minutes, which is longer than any single match duration, confirming its durability. The voltage supplied to the weapon motor was also only 7.4 volts, directly from the second battery. This value matched the rated input for the N20 motor used and did not exceed its voltage tolerance. No signs of motor damage or electrical stress were observed during repeated activation cycles. Thermal performance was also within acceptable limits. As mentioned in the power subsystem verification, the weapon motor only drew a maximum of 60 milliamps during peak operation, which contributed to the battery and MOSFET remaining cool throughout extended use. There was no observed overheating, and no performance degradation was noted during testing.

The mechanical drawback we had was that although the original mechanical requirement specified that the weapon must sit flush with the top of the chassis and extend no more than 0.5 centimeters, a design change introduced a top-mounted spinning blade. As a result, this requirement was no longer applicable in its original form. The new design allows the weapon to extend above the car to optimize offensive capability, and the blade remains securely fastened and balanced during use.

Cost

Below is a cost breakdown of all the components we used for our project. We have also included a breakdown of manufacturing costs for our project.

Parts cost:
1x 2 pack 1000RPM 6V N20 Brushed Motors - $9.99
1x 2 pack 600RPM 6V N20 Brushed Motors - $9.99
1x L298n motor driver module - $6.99
1x ESP32-S3-DevKitC-1-N8R8 Development Board - $15.99
1x LM1117MPX-3.3 - $1.01
6x CONN HEADER VERT 2POS 2.54MM - $0.16
6x WR-PHD 2.54 MM SOCKET HEADER 4 P - $0.40
3D Printing Cost:
Base price: $4.00
Price per gram of PLA: $0.10
Total cost to print for 472.95 = $51.295

Total cost: $102.62

Labor cost:
Average electrical engineer salary per hour in Chicago: $55.02
Estimated time to complete: 10 hours per week, totaling 70 hours
Total cost of labor is $3850.40

Conclusion

Our team successfully designed and implemented a functional Antweight BattleBot that met all critical competition requirements. The final robot was fully wireless, under the 2-pound weight limit, and featured independent mobility and a high-speed weapon system. Despite constraints

related to motor power and PCB complexity, the design demonstrated reliable performance, modularity, and compliance with safety and ethical guidelines.

Throughout the semester, we achieved key milestones: a stable WiFi-controlled system using the ESP32, a durable PLA-printed chassis capable of housing all components, and a spinning weapon that performed consistently above our minimum RPM threshold. Our power system remained cool and efficient during operation, and manual disconnects ensured safe handling of all electrical components. We also confirmed that our weapon and drivetrain subsystems operated reliably, although we recognized that our motor selection limited turning performance and braking ability. Upgrading to higher torque motors in the future would improve agility and directional control. Additionally, a fully functional custom PCB remains an area for future development.

In a broader context, this project demonstrates how accessible engineering tools like 3D printing, open-source software, and low-cost microcontrollers can empower students to build real-world embedded systems. Environmentally, we minimized waste by using PLA, a biodegradable material, and designing a chassis that could be reprinted and reused. Societally, the project encourages teamwork, problem-solving, and hands-on technical skills that directly apply to careers in robotics, automation, and embedded systems development.

Ethics

Our BattleBot project raises several ethical considerations that we have addressed by the IEEE Code of Ethics. We prioritized the safety and well-being of ourselves and those around us by carefully designing a robot that operates safely in an indoor environment while minimizing environmental impact. We remained honest about our technical limitations, especially in areas such as motor control and PCB design, and sought help and conducted research where necessary. Throughout the project, we welcomed constructive feedback from peers and instructors and made design adjustments accordingly. We ensured that every decision we made was grounded in practical testing and clear engineering rationale.

Safety

Safety was a central focus throughout the development of our BattleBot, especially given the risks associated with LiPo batteries, high-speed motors, and rotating weapon systems. We used two 7.4V 500 mAh LiPo batteries, each well below the competition's 16V maximum voltage limit, to ensure safe operating conditions. To comply with safety requirements and minimize the risk of short circuits or overheating, each battery included a manual disconnect switch that allowed for immediate shutdown in case of malfunction. The voltage regulator and motor driver were monitored during extended operation and did not reach unsafe temperatures, in part due to

the low current draw of our motors and weapon system. Our weapon motor, spinning a top-mounted blade at over 600 RPM, was secured to the chassis and tested in a safe, enclosed environment to prevent injury or damage in case of detachment or imbalance. We avoided the use of hazardous chemicals or biological materials entirely, limiting all safety concerns to electrical and mechanical domains. All testing was conducted under supervision, and charging procedures were followed with care to prevent fire hazards, aligning with best practices for LiPo battery safety.

References

[1] Espressif Systems, ESP32 Series Datasheet, Espressif Systems, 2023. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf. [Accessed: 10-Feb-2025].

[2] IEEE, "IEEE SA - IEEE Code of Ethics," IEEE, 2024. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 8-Feb-2025].

[3] Texas Instruments, MCF8316D 4.5-V to 60-V sensorless FOC brushless motor driver with integrated MOSFETs, Texas Instruments, Dec. 2023. [Online]. Available: https://www.ti.com/lit/ds/symlink/mcf8316d.pdf. [Accessed: 9-March-2025].

[4] Handson Technology, L298N Motor Driver Module Datasheet. [Online]. Available: https://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf. [Accessed: 16-March-2025].

[5] Handson Technology, GA12-N20 Micro DC Motor Datasheet. [Online]. Available: https://www.handsontec.com/dataspecs/GA12-N20.pdf. [Accessed: 20-March-2025].