ECE 445

Spring 2025
Final Report

Antweight Combat Robot

Team 11
Ryan Middendorf
Teodor Tchalakov

TA: Micheal Molter

Abstract

This final report presents the design and implementation of a competitive ant-weight battle robot that overcomes several complex engineering challenges across mechanical, electrical, and control domains. A key innovation is the use of a six-legged walking system, designed for stability and traction under high-impact conditions. Each leg features dual-servo actuation, coordinated via a custom PCB leveraging a PCA9685 servo driver and ESP32-WROOM-32E microcontroller. A weapon system consists of a high-speed 360-degree shell spinner driven by a brushless motor. Initial custom ESC designs failed under load due to current overshoot and component failure, so an off-the-shelf ESC was adopted to meet performance goals, achieving a 113 MPH tip speed with sub-10 second recovery post-impact. The robot also features robust wireless control via Bluetooth using a PS4 controller, enabling reliable, low-latency maneuverability. System-level integration was validated through rigorous subsystem testing and combat-readiness trials.

1. Introduction

1.1 Problem

The Ant-weight 3D printed Battlebots competition hosted by Professor Gruev puts custom built remote controlled robots to battle each other in an enclosed arena till either robot is disabled making the other the winner of that round. The competition's rules follow the official National Robotics Competition guidelines [1]. Robots must weigh less than two pounds, be composed of 3D-printed PETG, ABS, or PLA(+) materials, and be remotely controlled using either Bluetooth or Wi-Fi. They must feature a combat-effective weapon, provide both manual and automatic shutdown options, and survive intense 2-minute matches in a competitive arena. Creating a winning robot requires balancing offense and defense to enable the robot to execute power physical blows while also not becoming disabled from the blows of others. We are also challenged with building a custom PCB and control solution to survive the 2 minute matches and win the competition by disabling the other robot. With many mechanical, electrical, and software choices for making a winning robot [2], our design needs to carefully assess the tradeoffs in design and implementation.

1.2 Solution

To address this challenge, we developed a robot with a unique walking locomotion system and a robust shell spinner weapon. Instead of traditional wheels, our design uses a scuttling gait with six legs, each powered by two servos to provide rotation and vertical translation. This design improves stability under impact and enhances grip on the arena surface. The shell spinner weapon, inspired by previous successful designs, rotates around the entire robot offering 360-degree attacks and protection. All extra available weight up to 2 pounds will be allocated to thickening up the defensive structure of the robot. The walking mechanism works by having 6 legs with 2 joints, one horizontally rotating joint followed by a translational joint that translates vertically. To control these legs, we will create a custom PCB that uses the PCA9685PW servo driver to control all 12 servos, an ESP32-S3-WROOM-1-N16 to communicate with a PlayStation 4 controller over Bluetooth to command the robot and control the servo driver over I2C. The ESP32 also interfaces over PWM with the electronic speed controller (ESC) circuit containing 3 IRS2004 half driver chips to turn the weapon motor.

Figure 1 Robot CAD on left and actual robot on right

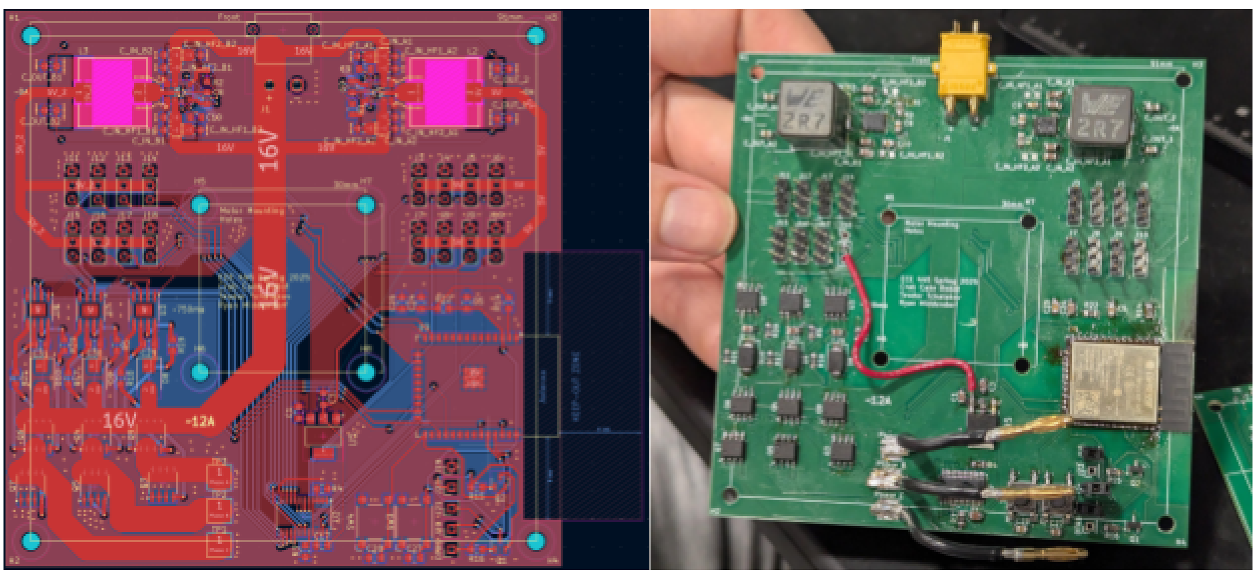

Figure 2 PCB layout on left and assembly right

1.3 High Level Requirements

1. The robot upon powering on should be able to pair with the controller in under 15 seconds.
2. The weapon should have a maximum tip speed of at least 100 mph and should be able to recover to that speed within 10 seconds after a collision.
3. The robot must demonstrate controlled motion for the entire 2-minute match and be capable of crossing the arena in under 30 seconds.

2. Design

2.1 Block Diagram

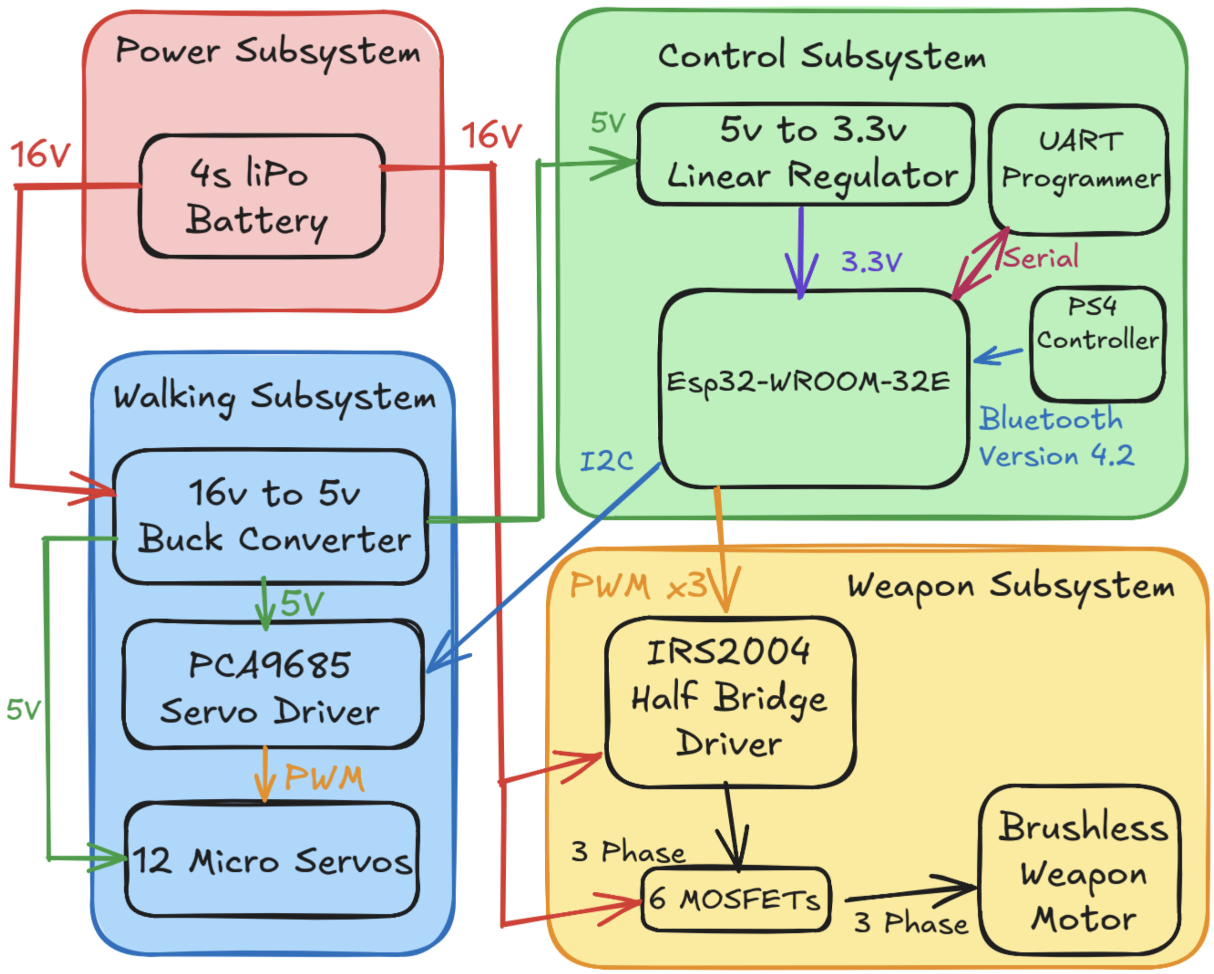

Figure 3 Block diagram of robot system containing power, walking, control, and weapon subsystems. Outlines connections wired and wireless for data transmission and power.

The robot consists of 4 main subsystems, power, control, walking, and weapon. The power subsystem provides power to all other subsystems, and it uses a LiPo battery and voltage regulators to step down the battery voltage. The controller subsystem is responsible for taking user input and controlling the motion of the robot. This consists of an ESP32-WROOM-32E module and a PS4 controller to provide user input. The walking subsystem consists of a PCA9685 servo driver chip and 12 micro servos, all controlled by the ESP32 over I2C and powered by the power subsystem. The weapon subsystem contains the ESC circuit, brushless weapon motor, and the 3D printed shell spinner.

2.2 Design Procedure

The design process began with a survey of common locomotion and weapon systems used in antweight robots [2]. While wheeled robots offer simplicity and speed, we opted for a legged approach to enhance stability and traction, especially under high-impact conditions. This also allows us to compete at a weight of 2-lbs in a standard 1-lb combat robot competition due to the 100% weight bonus given to true walking robots. After reviewing mechanisms like crank-rocker linkages and scissor lifts, we chose a simplified two-joint leg system, providing forward and backward translation and vertical lift using standard 9g servos. By having 6 legs, we can alternate between two tripods that support and translate the robot [3].

For the weapon system, we evaluated vertical and horizontal spinner designs, ultimately selecting a full shell spinner for its 360-degree coverage and protection of the weaker walking subsystem. On the electronics side, we selected the ESP32-WROOM-32E for wireless control due to its low power draw and native Bluetooth Classic support required for a direct connection to the PS4 controller, and paired it with a PCA9685 servo controller for the walking subsystem. The IRS2004 gate driver was chosen for driving the brushless weapon motor due to its phase driving output and compact form factor.

2.3 Design Details

2.3.2 Power Subsystem

The power subsystem uses a 4S 850mAh 75C Tattu LiPo battery capable of delivering 63.75A peak current. This subsystem provides 16 volts for the weapon motor and gate driver, 5V for the servos via an LM61495Q5RPHRQ1 buck converter, and 3.3 V for the ESP32 via an NCP1117LPST33T3G linear regulator. The 5 V power comes from the 16 V line and the 3.3 V power comes from the 5V line. The XT30PW connector and a screw switch allow for reliable connections and safe manual circuit disconnects for shutdown.

Each servo driven at 5 V has a max current draw of 800 ma, which means that our worst case current draw is .8 * 12 = 9.6 A. With a 10% safety margin this is too close to the maximum 10 A output of this buck converter, so we made two copies of the circuit so that half of the servos have dedicated power regulators.

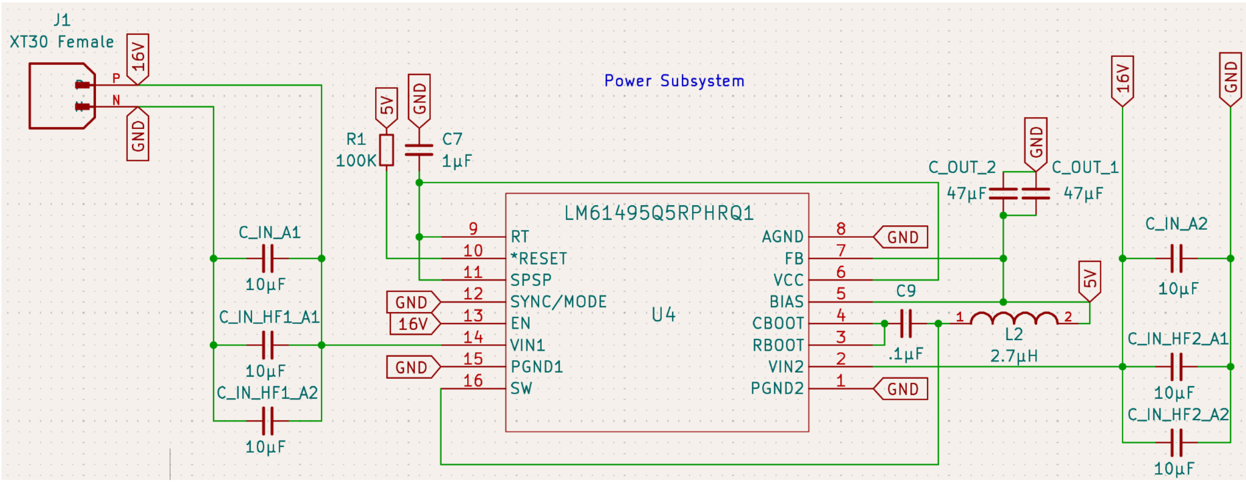

Figure 4 Power subsystem schematic of the 5V regulator circuit using the LM61495Q5RPHRQ1.

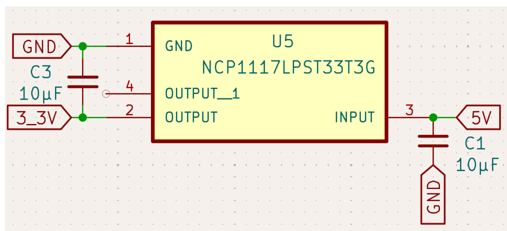

Figure 5 Power subsystem schematic of the 3.3V regulator circuit using the NCP1117LPST33T3G.

2.3.3 Walking Subsystem

The walking mechanism consists of six legs arranged radially, each with two degrees of freedom: rotation and lift. Each leg is controlled by two micro servos, one for horizontal rotation and one for vertical translation. The PCA9685 controller issues PWM signals to individually set the positions of the 12 servos. The ESP32 sends position values over I2C to the servo driver chip. A0-5 are set low to make the I2C address be 0x40.

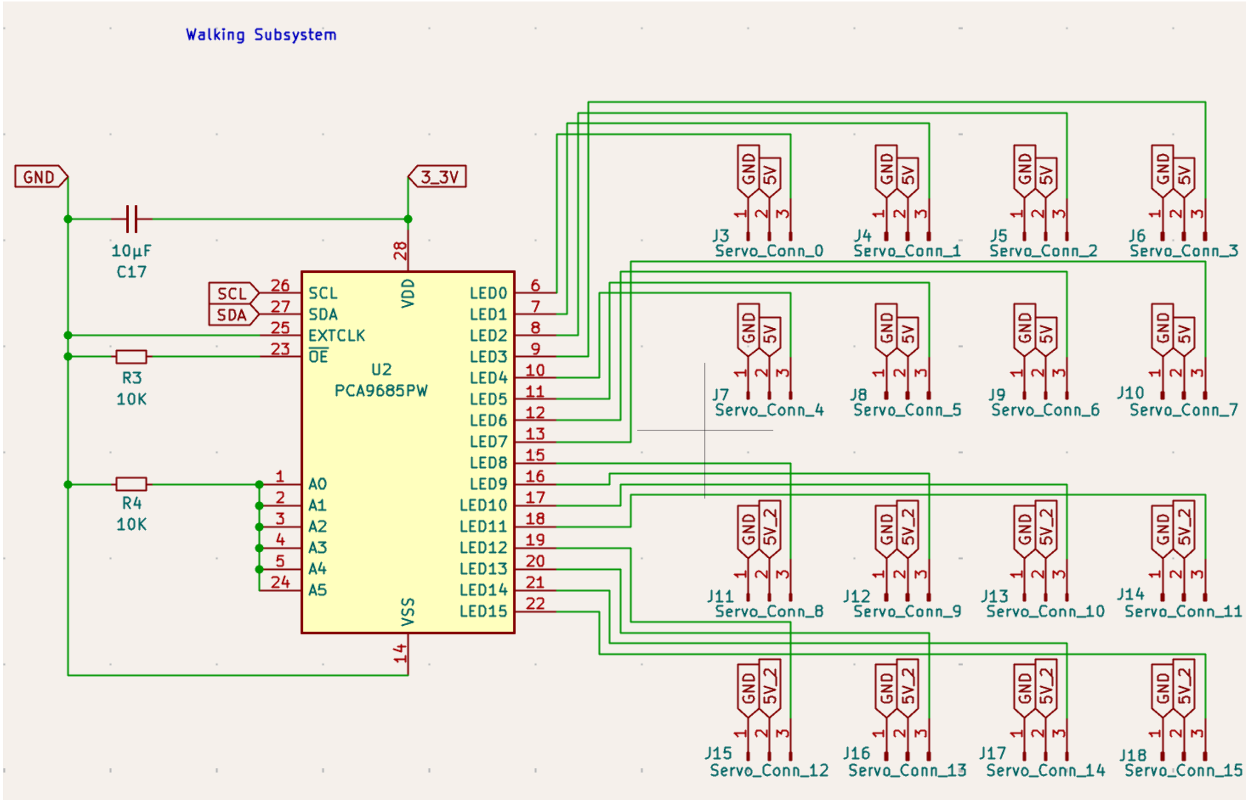

Figure 6 Walking subsystem servo driver with 16 available channels. 12 channels are used for the servos and 1 for the external backup ESC.

Side View

Top View

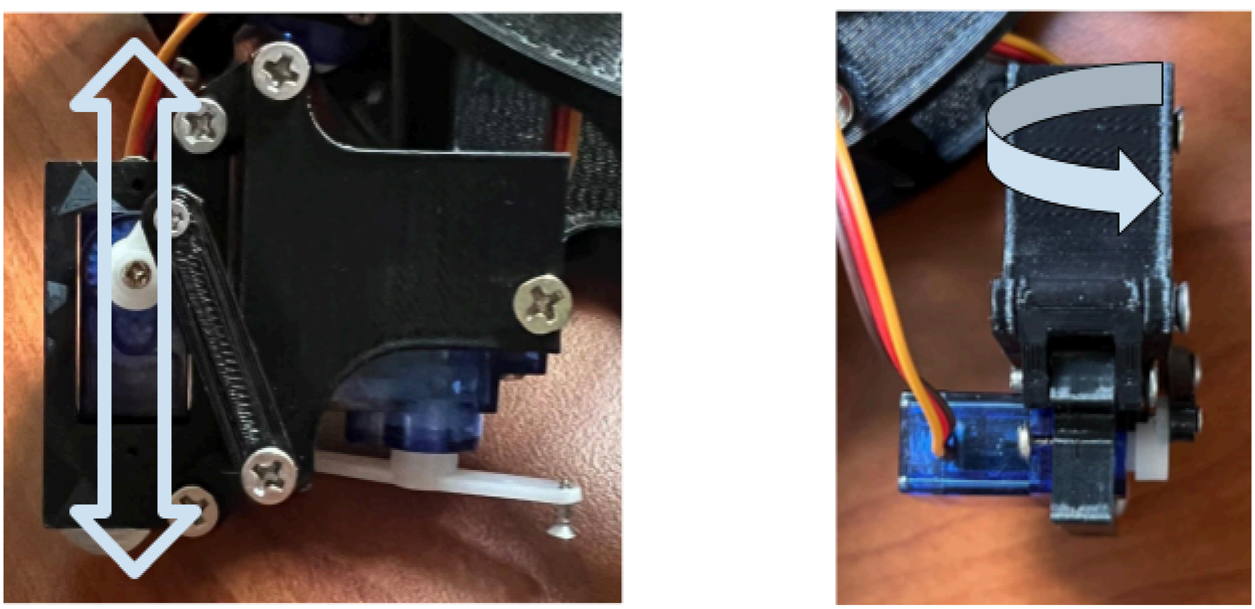

Figure 7 Top and side profiles of the servo driver mechanisms

The code is broken down into walking forward/backwards and turning left/right. To walk forwards, three legs must be in contact with the ground to push the robot forward while the other three legs are getting in a new position in the opposite direction of the first group of three legs. At any given time, there are three legs on the ground. To turn in place, all legs lift the robot and robot counterclockwise or clockwise and then put the robot down on its bottom, allowing the legs to clear the floor and get back into the starting position to pick up and turn the robot again.

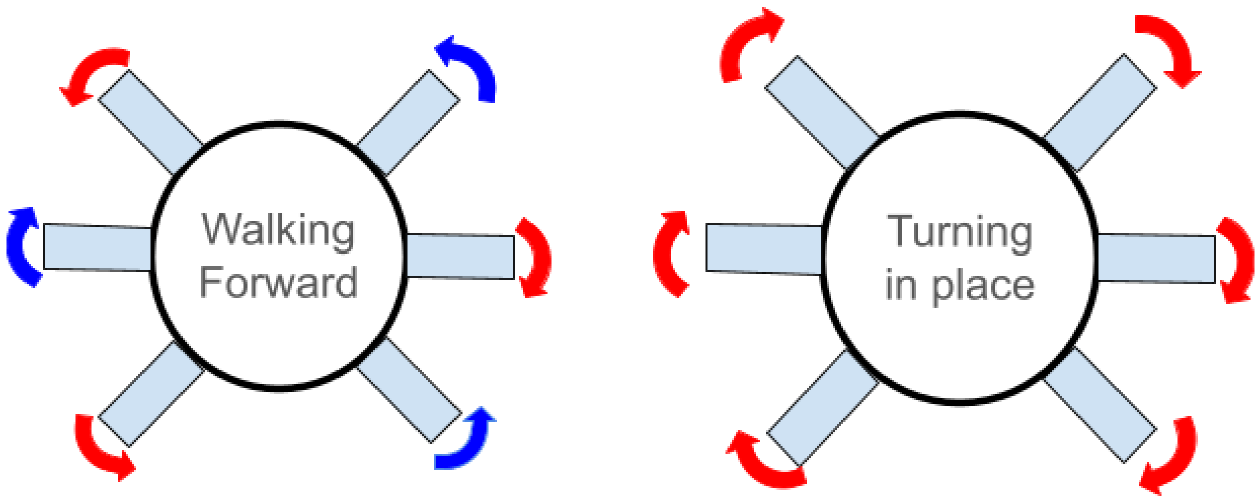

Figure 8 States of forward walking and turning in place motion. Same color means the up and down position are synced together while different color means they are opposite states.

2.3.4 Weapon Subsystem

The shell spinner is turned by a Flash Hobby D3536 750KV brushless motor connected to a custom ESC circuit using 3 IRS2204 chips and 6 external MOSFETs. The motor drives a 3D printed shell designed to maximize moment of inertia and have a small protrusion to cause large impacts at high speeds.

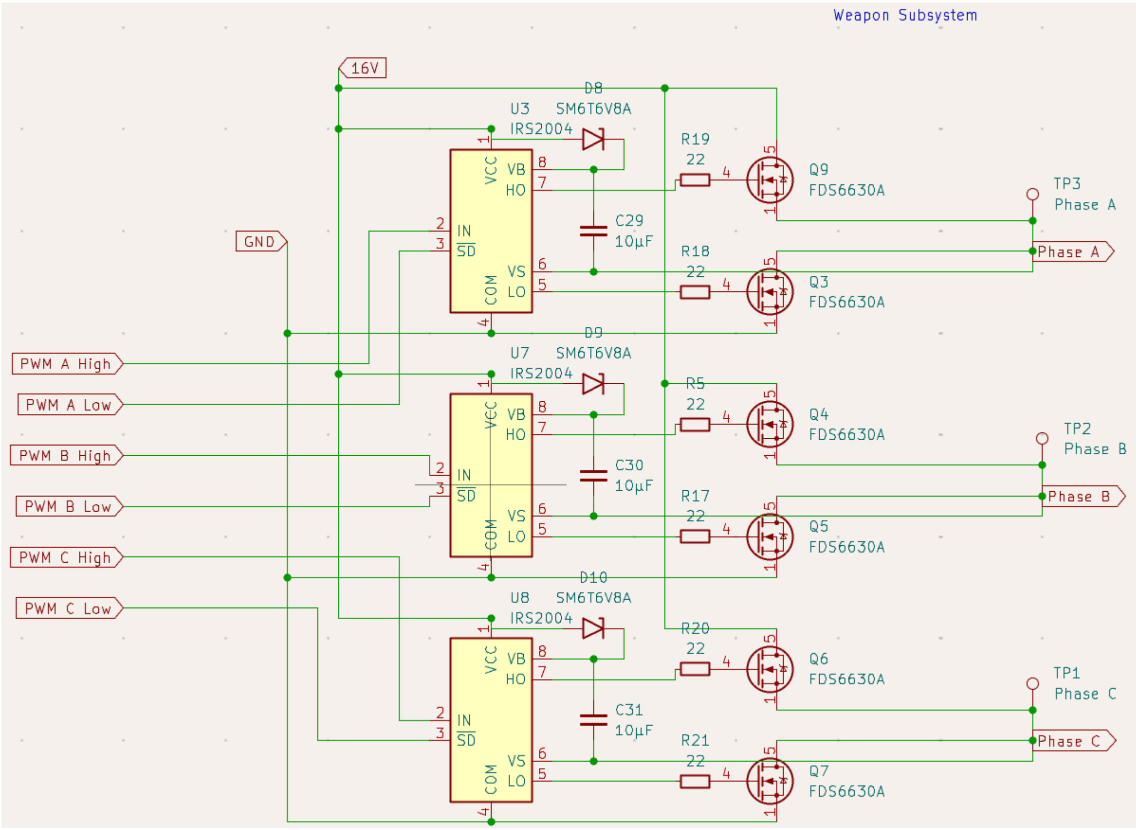

Figure 9 H-bridge circuit driven by three phase offset PWM signals from the ESP32

2.3.5 Control Subsystem

The ESP32 microcontroller communicates via Bluetooth 4.2 with a PlayStation 4 controller and sends I2C commands to the PCA9685 and PWM signals to the IRS2004. Safety features include manual shutdown via controller button and automatic shutdown upon Bluetooth disconnection. The ESP32 is powered independently on a 3.3V line. We developed our code in C++ using the Arduino Framework within the Platformio IDE [6].

The 3 phase PWM signals we use are generated using the ESP32's built in Motor Control PWM (MCPWM) hardware which can be specially configured to generate the necessary input signal. This approach, although difficult to implement, offloads all PWM related computation to the MCPWM hardware. The low signals used are either high to enable the circuit or low to disable it. The ESP32 communicates with the servo driver using I2C.

To connect the PS4 controller directly to the ESP32, we use the DS4 library [4] and we get the Bluetooth MAC address of the PS4 controller we want to use by running the command `system_profiler SPBluetoothDataType` on within MacOS while the PS4 controller is connected to it. Providing this to the DS4 library, the ESP32 will try to connect to that MAC address and it will pair with the device if the device is in pairing mode. Having a direct connection, enhances the reliability of our system compared to having an intermediate computer.

To program the ESP32, we connect the TX and RX pins of the HiLetgo FT232RL Mini USB to TTL Serial Converter Adapter Module [5] to the RX and TX pins respectively on the ESP32. The safety shutoff logic for a manual and automatic shutoff is simple. We configure a button as a toggle button on the PS4 for manually shutdown by the user and we check each iteration of the internal loop if the PS4 controller is connected to decide if we need to automatically shutdown. To re-enable the robot, the PS4 controller must be connected to it again.

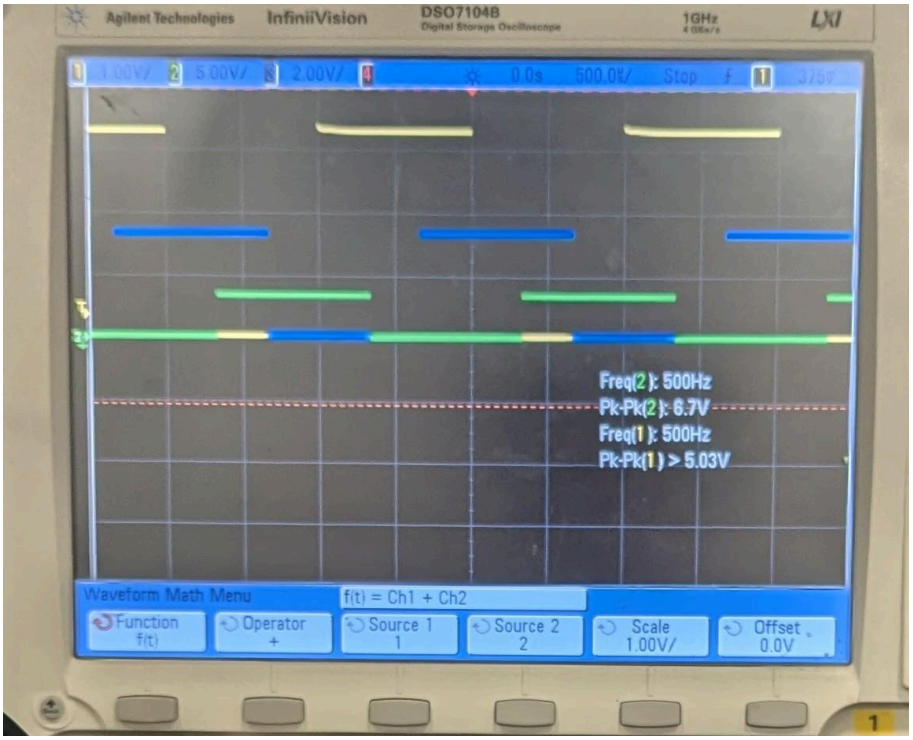

Figure 10 3 Phase 120 degree offset PWM signal using MCPWM ESP32 hardware.

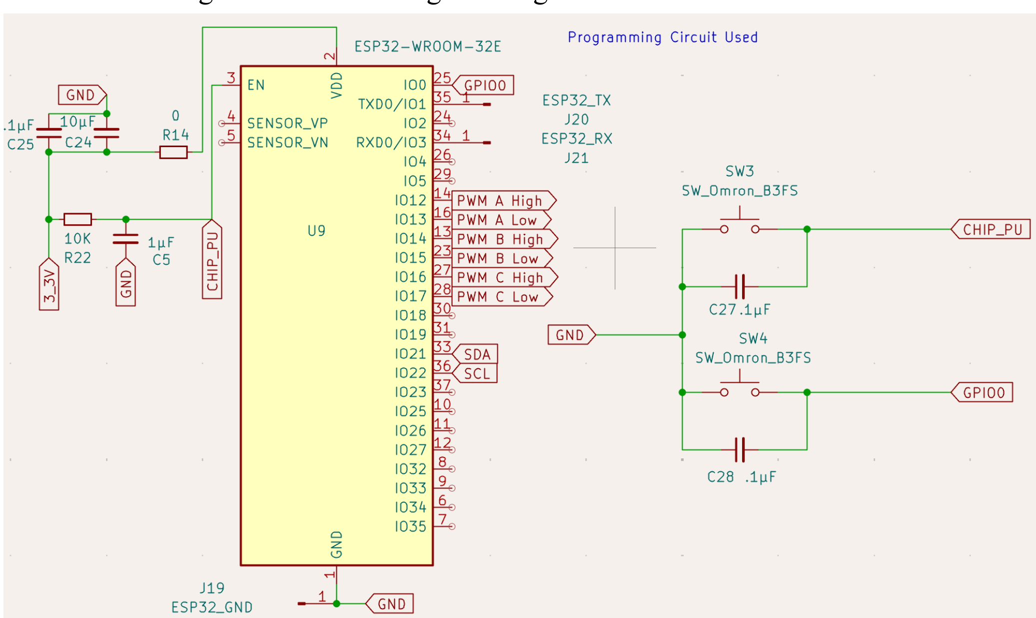

Figure 11 ESP32-WROOM-32E schematic with buttons for chip enable and reset.

3. Verification

3.1 Verification Overview

Each high-level requirement was validated using targeted subsystem tests and full system integration trials. We tested connectivity and response time for the Bluetooth control system, monitored tip speed and spin-up recovery for the weapon, and assessed walking efficiency through timed traversal experiments. We prioritized metrics that reflected combat-readiness, such as startup time, locomotion control accuracy, and energy consumption under load.

3.2 Verification Details

3.2.1 Power Subsystem

The power subsystem was very robust and functioned as intended after moving the 3.3V linear regulator from the 16V line to the 5V line. The heat generated by such a large voltage drop was too much for the regulator to handle. We tested the power subsystem using a multimeter as well as using a stopwatch to perform calculations. These validations can be viewed in Appendix A.

Table 1 Power Subsystem Requirements and Verification

Requirements	Verification
Provide 25 A continuous current	Operating at or under 30C discharge, 25.5 A for our battery, will guarantee it lasts the full 2 minutes of a match. To test this we can run our bot on a full battery and see how long it takes to fully discharge. If it's more than 2 minutes it passes.
Provide 60 A peak current	To test this we can hook a battery up to a load like a resistor and check the current with a multimeter. If the measured current is greater than 60 A (should be 63.75) it passes.
Provide a stable 5V for the servos and 3.3V for the esp32 chip	To test this we can run our servos and esp32 chip and check the voltages supplied to them with a multimeter. If it reads the correct voltage without fluctuating it passes.

3.2.2 Walking Subsystem

The walking subsystem worked very reliably and exceeded our expectations when it came to its travel speed. We tested using a stopwatch to time it as it crossed a 1 meter distance, used a ruler to measure the distance between the chassis and the ground when it is lifted to its max height, and visually inspected the servos as we commanded them through a set of movements. These validations can be viewed in Appendix B.

Table 2 Walking Subsystem Requirements and Verification

Requirements	Verification
The robot can walk forward and backwards at a speed of at least 5 cm/s	Once the robot is assembled, run the walking code, command the robot to walk 1 meter, and time it with a stopwatch. If the bot covers this distance in under 10 seconds it passes.
Must be able to lift the entire 2lb robot 3mm	Once the robot is assembled, move 3 of the legs to their lifting position and measure the gap under the robot with calipers or a ruler. If the gap is larger than 3mm it passes.
The servo driver must be able to correctly move each servo individually	Once all the servos are connected to the servo driver which is getting commands from the ESP32 chip, have each servo move a set distance sequentially. If all the servos move the correct distance in the correct order it passes.

3.2.3 Weapon Subsystem

The weapon subsystem unfortunately did not work as intended. The brushless motor and shell both functioned well and inflicted a lot of damage to opposing bots, but the ESC circuit had a few issues. We did get a smaller motor spinning, but we really struggled to tune the values we were sending the motor and couldn't prevent the motor from fighting itself as it tried to rotate. We also were unable to drive our larger weapon motor as it drew too much current and would blow the MOSFETS as soon as it was powered on for testing. In our final design we used an off the shelf ESC controlled by a PWM signal sent from the ESP32. Using this ESC we were able to successfully verify all our requirements using a tachometer and some simple calculations. These validations can be viewed in Appendix C.

Table 3 Weapon Subsystem Requirements and Verification

Requirements	Verification
Must be able to reach a tip speed of 100 mph	With the shell on the robot and in a safe testing location, spin the shell to its top speed and use a tachometer to measure its rpm. From this the tip speed can be easily calculated and if it's over 100 mph it passes.
Must recover from impacts by returning to its original speed in under 10 seconds	With the shell on the robot and in a safe testing location, spin the shell to an intermediate speed, roughly 50 mph tip speed, and have it hit a test object. Measure it with a tachometer the whole time and if it returns to its original speed before the impact within 10 seconds it passes.
The esc must be able to provide the motor with enough current that it doesn't burn out during operation	With the shell on the robot and in a safe testing location, spin the shell to its top speed and then stop it and make sure the esc still works and was able to handle the current running through it and the back emf as the motor slowed down. It passes if the esc is still in its original condition

3.2.4 Control Subsystem

The control subsystem was very successful and controlled all 12 micro servos for the walking subsystem individually as well as the ESC for the weapon subsystem. It reliably connected to the PS4 controller in under 5 seconds and when connection was lost the ESP32 correctly disabled the robot. We verified our requirements with a stopwatch and visual inspection of the robot while connecting and disconnecting the controller, and during the tachometer testing we did for the weapon subsystem. These validations can be viewed in Appendix C and D.

Table 4 Control Subsystem Requirements and Verification

Requirements	Verification
Robot will automatically disable all motors when the Bluetooth connection is lost.	Connect the PlayStation controller to Bluetooth. Hold the joystick forward to walk forward, enable the weapon motor to spin, and press the PlayStation button to turn off the controller. With the last command sent being to walk forward, the test passes if the robot stops moving.
Will pair with the bluetooth controller in under 15 seconds	Turn the robot on and start a stopwatch at the same time. If the controller connects in under 15 seconds it passes.
Sends the correct PWM signal to the weapon subsystem allowing precise control of the weapon speed	With the robot in a safe testing location, slowly increase the weapon output on the controller while measuring the shell's speed with a tachometer. If the values on the tachometer climb synchronously with the commands being sent to the ESP32 it passes.

4.Cost Analysis

4.1 Parts

Table 5 Cost analysis for every part on PCB and in robot.

Description	Manufacturer	Quantity	Unit Price	Order Price	Seller Link
ESP32-WROOM-32E-N8 Module	Espressif Systems	1	$5.28	$5.28	Link
XT30PW	Amass	1	$0.69	$0.69	Link
68001-203HLF Header Pins	Amphenol ICC	16	$0.21	$3.36	Link
PCA9685PW, 118 Servo Driver	NXP USA Inc.	1	$2.76	$2.76	Link
LM61495Q5RPHRQ1 Buck	Texas	2	$6.41	$12.82	Link

Converter 16V to 5V	Instruments				
NCP1117LPST33T3G Linear Regulator 5V to 3.3V	Onsemi	1	$0.82	$0.82	Link
IRS2004	Infineon Tech	3	$1.18	$3.54	Link
CAP CER 47UF 6.3V X5R 0805 C0805C476M9PACTU	Kemet	4	$0.93	$3.72	Link
RES 0 OHM JUMPER 1/8W 0805 RMCF0805ZT0R00	Stackpole Electronics Inc	1	$0.10	$0.10	Link
RES SMD 22 OHM 1% 1/8W 0805 RT0805FRE0722RL	Stackpole Electronics Inc	6	$0.10	$0.60	Link
RES 100K OHM 5% 1/8W 0805 RMCF0805JG10K0	Stackpole Electronics Inc	2	$0.10	$0.20	Link
TVS DIODE 5.8VWM 10.5VC SMB SM6T6V8A	STMicroelectro nics	3	$0.35	$1.05	Link
FIXED IND 2.7UH 16.2A 4.7MOHM SM 7843320270	Würth Elektronik	2	$3.13	$6.26	Link
SWITCH TACTILE SPST-NO 0.05A 12V TS04-66-55-BK-100-SMT	Same Sky	2	$0.18	$0.36	Link
FingerTech Mini Power Switch	FingerTech Robotics	1	$8.43	$8.43	Link
CAP CER 0.1UF 25V Y5V 0805 CL21F104ZAANNNC	Samsung Electro-Mechan ics	5	$0.10	$0.50	Link
CAP CER 1UF 50V X7R 0805 CL21B105KBFNNNG	Samsung Electro-Mechan ics	3	$0.11	$0.33	Link
CAP CER 10UF 50V X5R 0805	Samsung Electro-Mechan ics	17	$0.71	$12.07	Link
RES 10K OHM 5% 1/8W 0805 RMCF0805JG10K0	Stackpole Electronics Inc	2	$0.10	$0.20	Link
TRANS NPN 25V 1.5A SOT23-3 SS8050-G	Comchip Technology	2	$0.24	$0.48	Link
MOSFET N-CH 30V 12.5A 8SOIC FDS6680A	Onsemi	6	$0.94	$5.64	Link
Tattu 14.8V 4S 850mAh LiPo Battery	Tattu	1	$19.49	$19.49	Link
D3536 750KV Brushless Motor	Flash Hobby	1	$19.99	$19.99	Link
DURAMIC 3D PLA+	Duramic 3D	1	$19.99	$19.99	Link

Filament 1.75mm 1Kg					

The total parts cost is $169, with only $32.87 for components that are soldered to the PCB.

4.2 Labor

Our project has 2 partners that we will count as getting paid for their labor. For high skill engineering labor at 40$/hour person with about 6 hours of work per person per week left with 7 weeks left gives 40 * 2.5 * 8 * 7 = $5600 per person.

5. Conclusion

5.2 Accomplishments

Our robot successfully met all our high-level requirements. We achieved reliable Bluetooth pairing within the target 15 seconds, and verified that the shell spinner reached over 100 mph tip speed and recovered in under 10 seconds after an impact. The walking system enabled traversal across the arena in less than 30 seconds, and maintained control for the full 2-minute match duration. All subsystems were successfully integrated on a custom PCB which proved robust and efficient throughout testing and competition.

5.3 Uncertainties and Future Work

While the robot performed well overall, there are areas that could be improved. Mainly the ESC circuit needs a redesign in order to work properly. At a minimum it needs more powerful mosfets that can handle the current draw of our large brushless motor, and ideally would include a simplified control circuit as tuning the IRS2004 drivers was very difficult. Additionally, even though our walking mechanism met all of our requirements, the speed could be increased by optimizing the travel paths of the legs. Finally, the overall size and weight could be reduced so more weight could be put into the shell and other armor making the bot more resilient to the impacts it will take throughout future competitions.

5.4 Ethical considerations

Our design followed the IEEE Code of Ethics, prioritizing user safety by including manual and automatic shutdown features. All team members shared responsibilities equitably and documented work honestly. Our power and mechanical systems were tested rigorously to ensure that no user or competitor could be harmed due to unintended failure and we implemented shutoffs into the code in case the bluetooth disconnected or anything else went awry.

5.5 Safety

Safety was addressed at both the hardware and software level. A screw switch allows for rapid manual power cutoff, while loss of Bluetooth connection automatically disables all motors. The weapon subsystem was tested exclusively in enclosed environments, and a physical weapon stop prevents accidental activation outside the arena.

References

[1] "The World's First Robotics Competition," National Robotics Challenge, https://www.thenrc.org/contest-manual (accessed Mar. 6, 2025).

[2] "Antweight Robots," BattleBots Wiki, https://battlebots.fandom.com/wiki/Category:Antweight_Robots (accessed Mar. 6, 2025).

[3] "How Scuttle Walks," dapperrogue, https://www.youtube.com/watch?v=U8zYyy_4mZI (accessed Feb. 12, 2025).

[4] "DS4-32," styxnanda, https://github.com/styxnanda/DS4-32 (accessed Mar. 6, 2025)

[5] "Amazon.com: HiLetgo FT232RL Mini USB to TTL Serial Converter Adapter Module 3.3V/5.5V : Electronics," Amazon.com, 2025. https://www.amazon.com/HiLetgo-FT232RL-Converter-Adapter-Breakout/dp/B00IJXZQ7C (accessed Apr. 02, 2025).

[6] "Professional collaborative platform for embedded development — PlatformIO latest documentation," docs.platformio.org. https://docs.platformio.org/en/latest/ (accessed Mar. 6, 2025)

Appendix A: Power Subsystem

A.1 Voltage Regulation

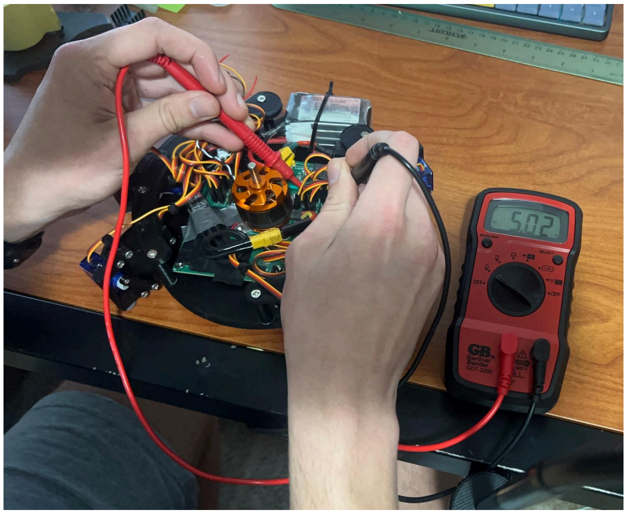

Figure 12 5V output from buck converter

Figure 13 3.3V output from linear regulator

A.2 Battery Calculations

The battery needs to be able to supply 47.5A if everything draws their max current simultaneously

- 35A for weapon motor
- 12A for all 12 servos
- 0.5A for the ESP32 and PCA9685 chips

$$850mAh * 75C = 63.75A \qquad (1)$$

Equation 1 Peak current draw calculation

The battery only needs to supply 18.5A during normal operation

- 15A for the weapon motor
- 3A for the servos
- 0.5A for the ESP32 and PCA9685 chips

$$\frac{60\frac{C}{min}}{2min} = 30C \qquad 850mAh * 30C = 25.5A \qquad (2)$$

Equation 2 Average current battery can supply for 2 mins

Appendix B: Walking Subsystem

B.1 Walking Speed

Table 6 Trials to cover 2 meters

Trial Number	1	2	3	4	5
Time (s)	27.8	27.3	28.5	28.4	27.7

Average time is 27.94 seconds, which gives a speed of 200 / 27.94 = 7.158 cm/s.

B.2 Robot Vertical Clearance

Robot successfully clears the 3mm distance.

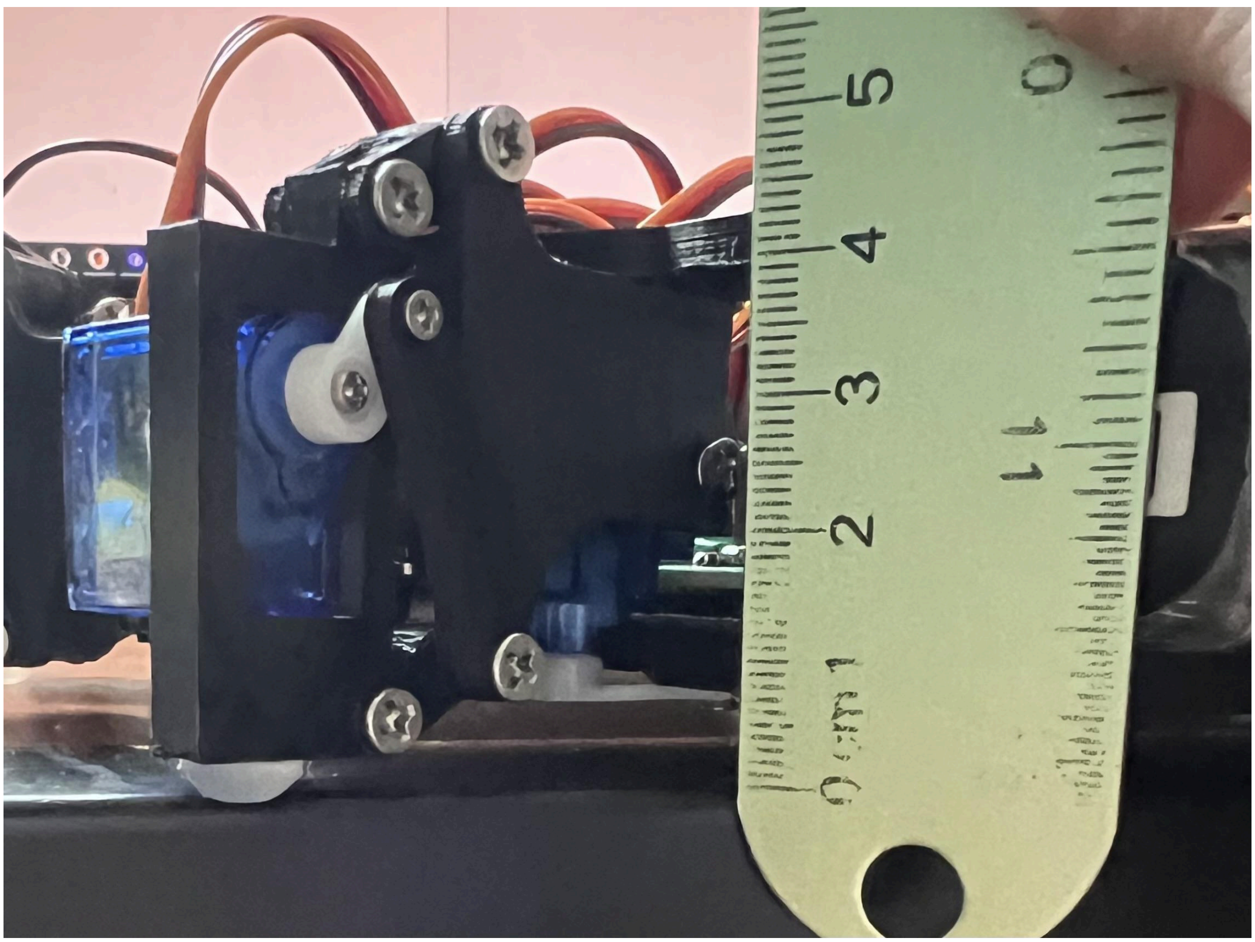

Figure 14 Robot using servo strength to lift the robot above ground. Lifted robot a distance of 6mm.

Appendix C: Weapon Subsystem

C.1 Tip Speed Testing and Calculations

Figure 15 RPM value measured using a tachometer

In a test box our shell reached 3172 RPM in 6.52 seconds and the shell has a diameter of 12 inches. By plugging these values into a simple equation we can see that the robot has a tip speed of 113 MPH and since it spun up from a rest in under 10 seconds it must be able to recover after impacts as well.

$$12\ in * \pi * 3172\ RPM = 119581.6\ \frac{in}{min} * \frac{60\ \frac{min}{hour}}{63360\ \frac{in}{mile}} = 113\ MPH \tag{3}$$

Equation 3 Tip speed calculation

Appendix D: Control Subsystem

D.1 Automatic Safety Disconnect

First turning on the weapon and walking the robot forward, then turning off the PlayStation controller, the robot successfully immediately stops moving and spinning the weapon.

D.2 Robot Bluetooth Reconnect Speed

Table 7 Bluetooth connection speed trials

Trial Number	1	2	3	4	5
Time (s)	5.4	4.8	5.3	4.5	5.1

The average time it takes to connect with the robot from turning on the Playstation controller is 5.02 seconds.