Final presentation

Team 3: Follow-Me Cart: App-controlled smart assistant

Shi Qiao, Zixuan Huang, Jiaming Gu
12/8/2025

Team Members

- Jiaming Gu
  - Senior CE student
  - In charge of Bluetooth/Wi-Fi pairing, Android apps, and algorithms for avoiding obstacle detection and finding ways.
- Zixuan Huang
  - Senior CE student
  - In charge of CV pipeline (user detection and tracking), sensor fusion on Raspberry Pi, dataset/test video capture.
- Shi Qiao
  - Senior EE student
  - In charge of ESP32 firmware, motor drive interface, Power supply, and PCB design

Problem Statement

People are often forced to multitask when shopping for groceries

- Comparing prices/price matching
- Taking care of children
- Wayfinding/searching for a specific item

Things gets even more complicated when you have to push a cart loaded with groceries…

How can we make grocery shopping more convenient?

Proposed solution

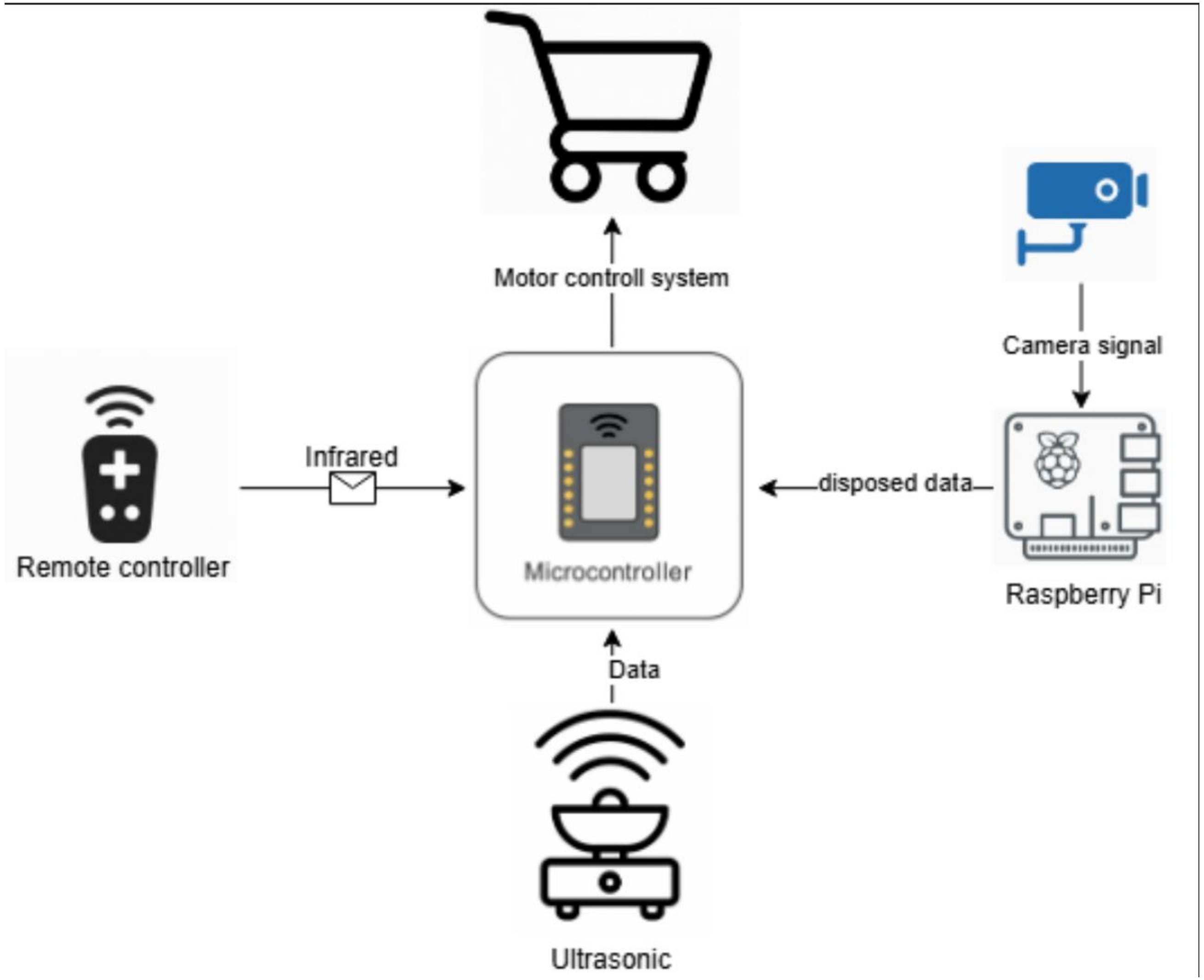
An automated shopping cart that can:

- Self-driven
- Follows the user on its own
- Avoids collision

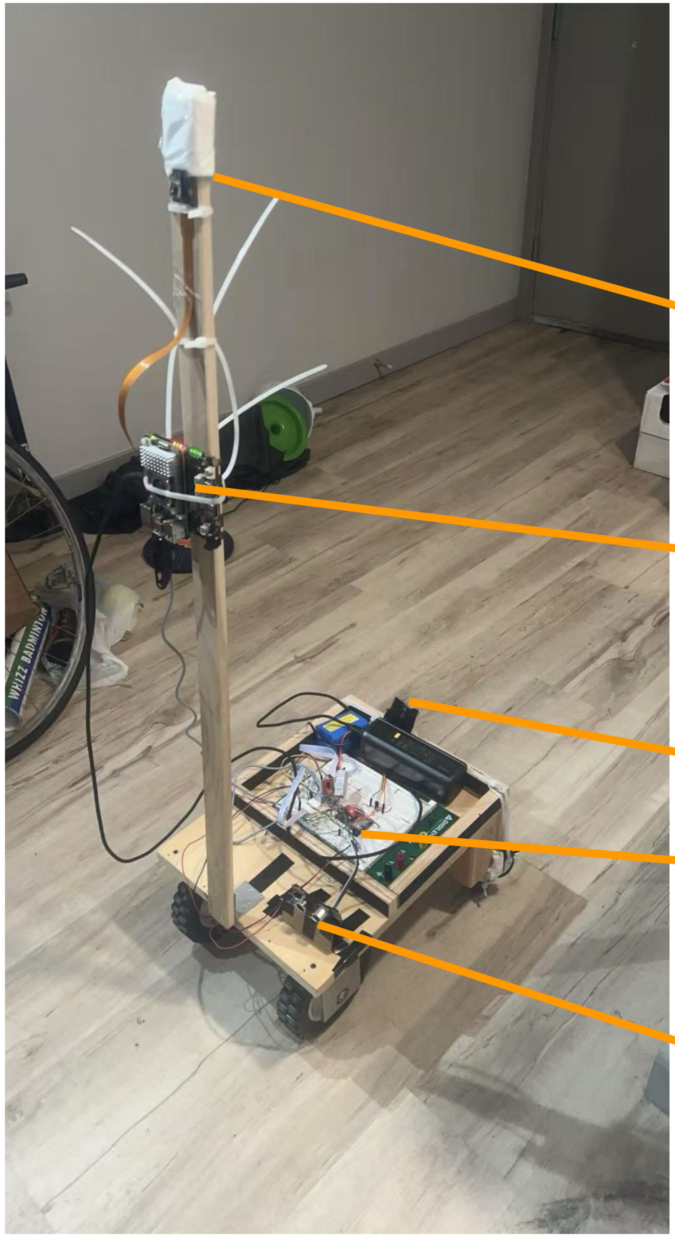
High-Level Requirements

- Have a 90% success rate in avoiding at-fault collisions
  - Obviously we don’t want to hurt anyone…
- Operate for 40-60 minutes without charging
  - According to USDA, average Americans uses 40-48 min per trip
- Follow the customer within 1-2 meters
  - A distance where the customer can easily reach the cart
  - Provides enough distance for the cart to brake

Visual Aid

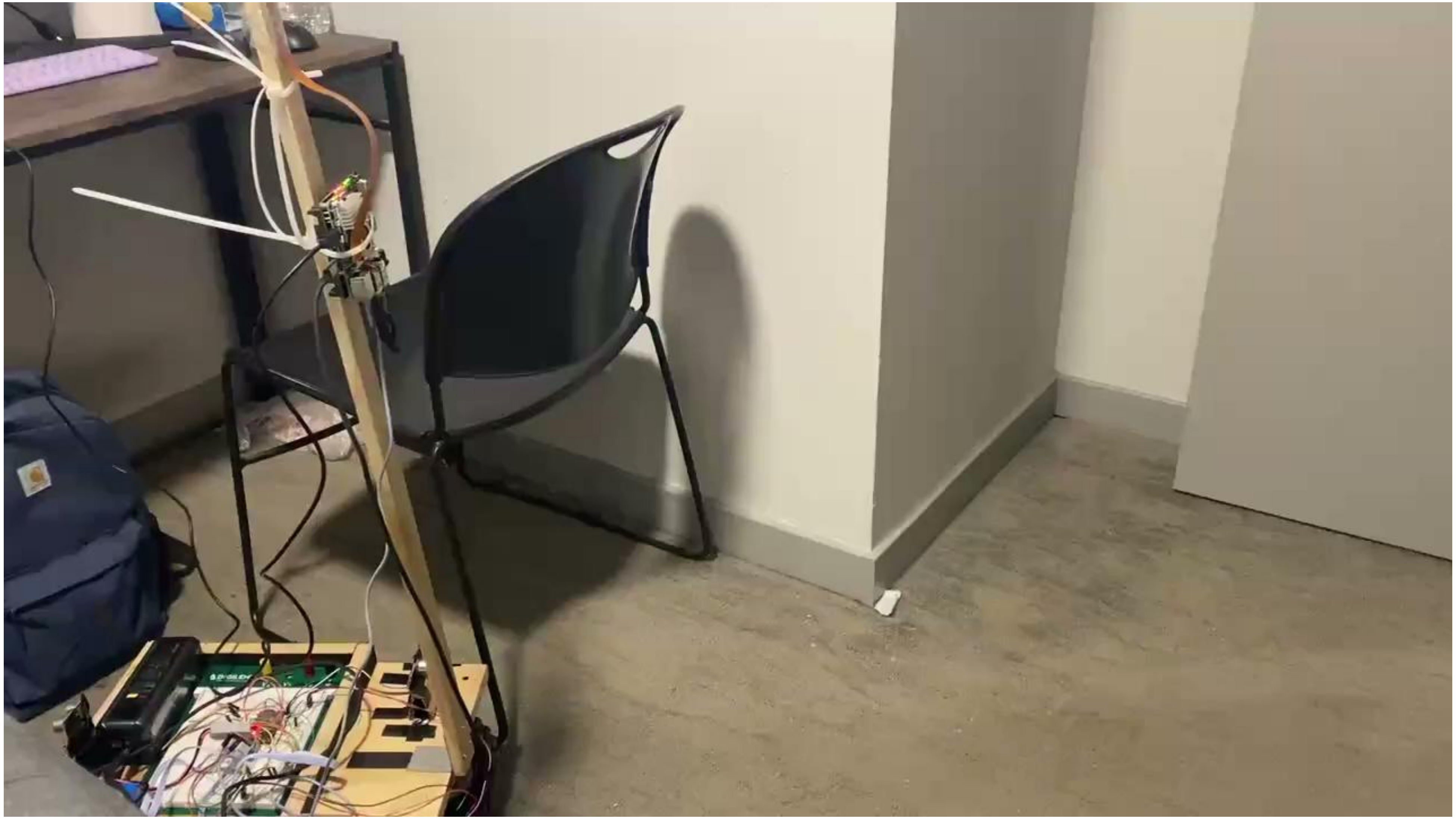

Motor controll system
Camera signal
Infrared
disposed data
Remote controller
Microcontroller
Raspberry Pi
Data
Ultrasonic

Cart

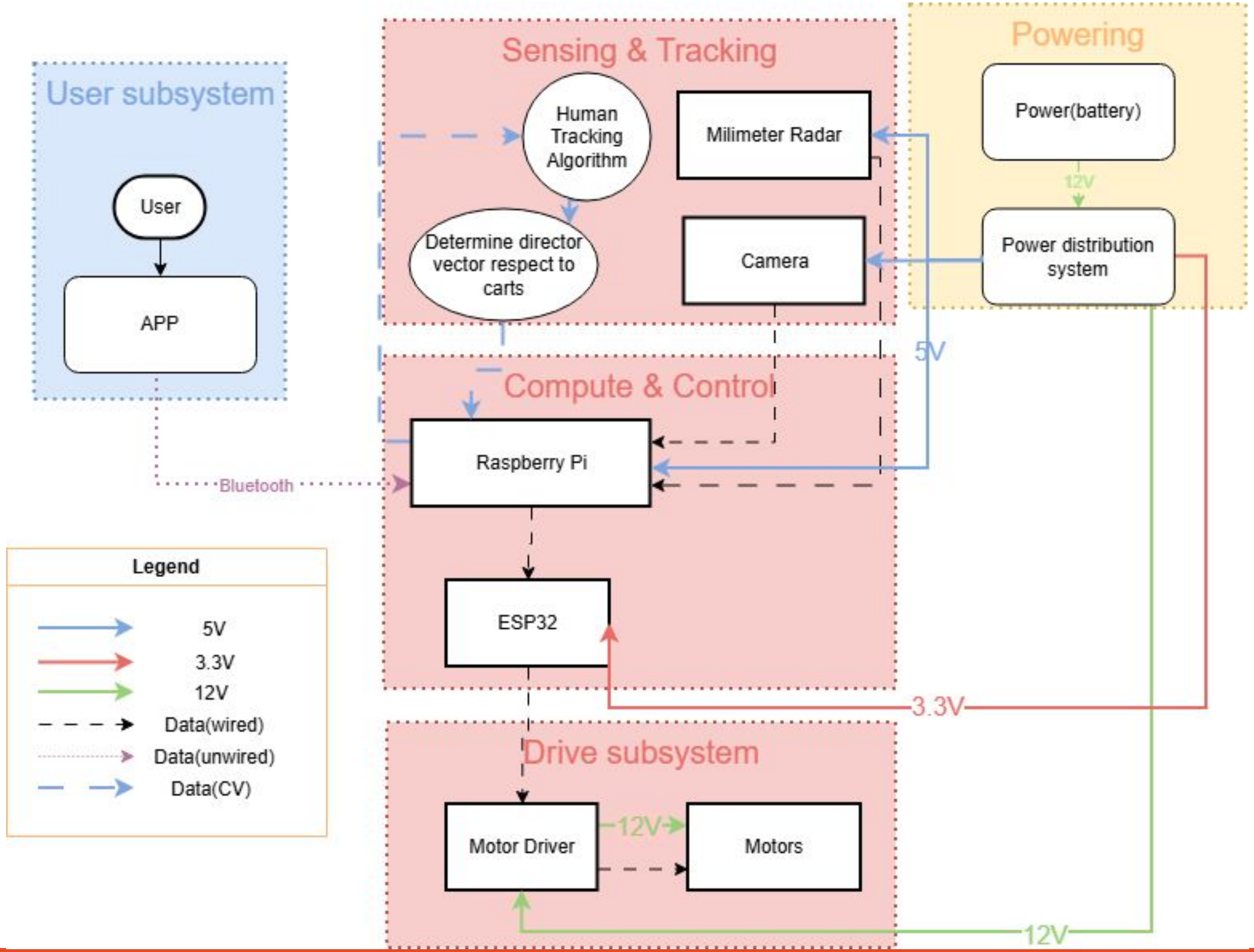


camera
Raspberry Pi
Back Ultrasonic
ESP32
Front Ultrasonic

Video



Block Diagram

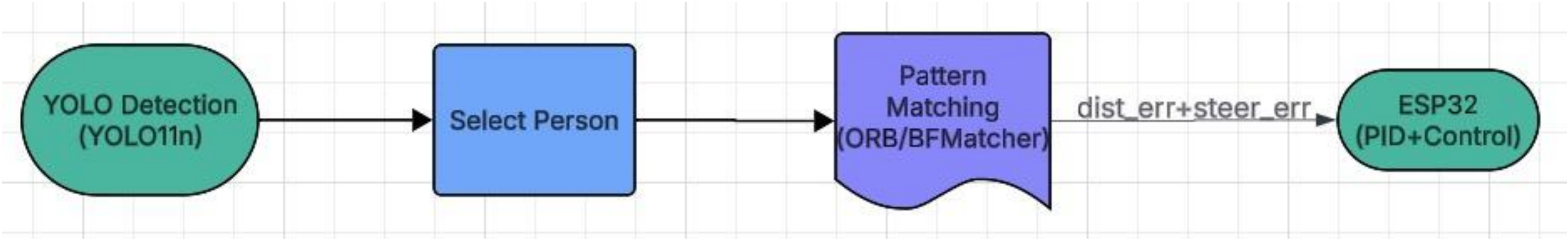


System Architecture

Subsystems:

- Sensing & Tracking
- Compute & Control
- Drive System
- Power System
- User System

Sensing and Tracking



Compute and control

Proportional control on ESP32:

PWM control logic of the motor driver:

PWM speed = Kp*distance_error

Steer change = Ki*steer_error

Left_speed = PWM speed + Steer change

Right_speed = PWM speed - Steer change

if (d_ultrasound >= 0 && d_ultrasound < 50){ l = 0;r = 0;}

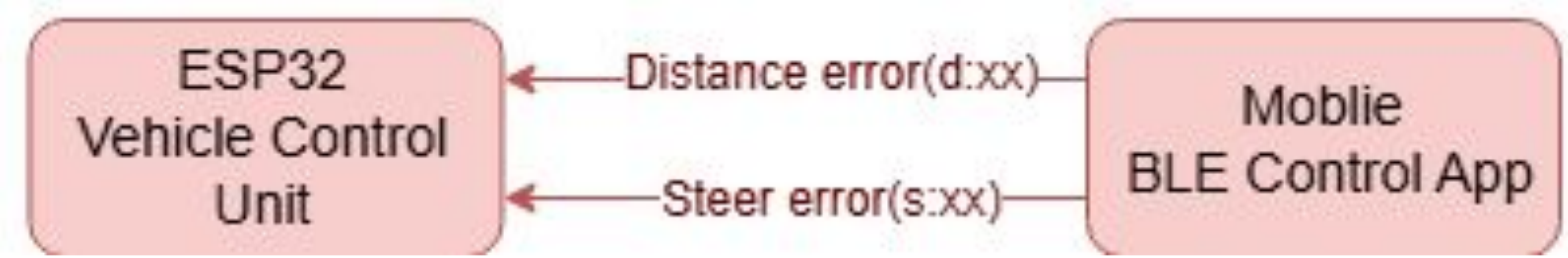
if (b_ultrasound >= 0 && b_ultrasound < 60){l = 0;r = 0;}

PWM speed: Integer from 0 to 255

Communication between controllers

ESP32 has ability of reading information from the serial port, which is provided by physics connection between ESP32 and the Raspberry Pi:

1. Distance error(format: “d:xx”)
2. Steer error(format: “s:xx”)



User communication

Connection to ESP32 with BLE, able of controlling by app

Mobile

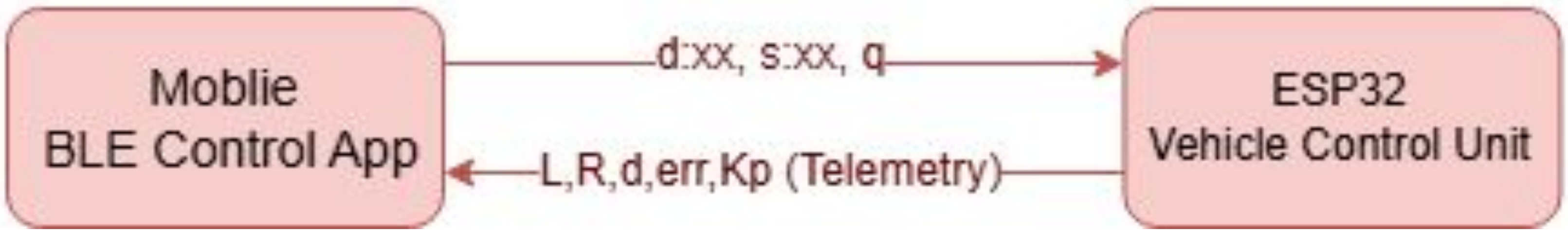
BLE Control App

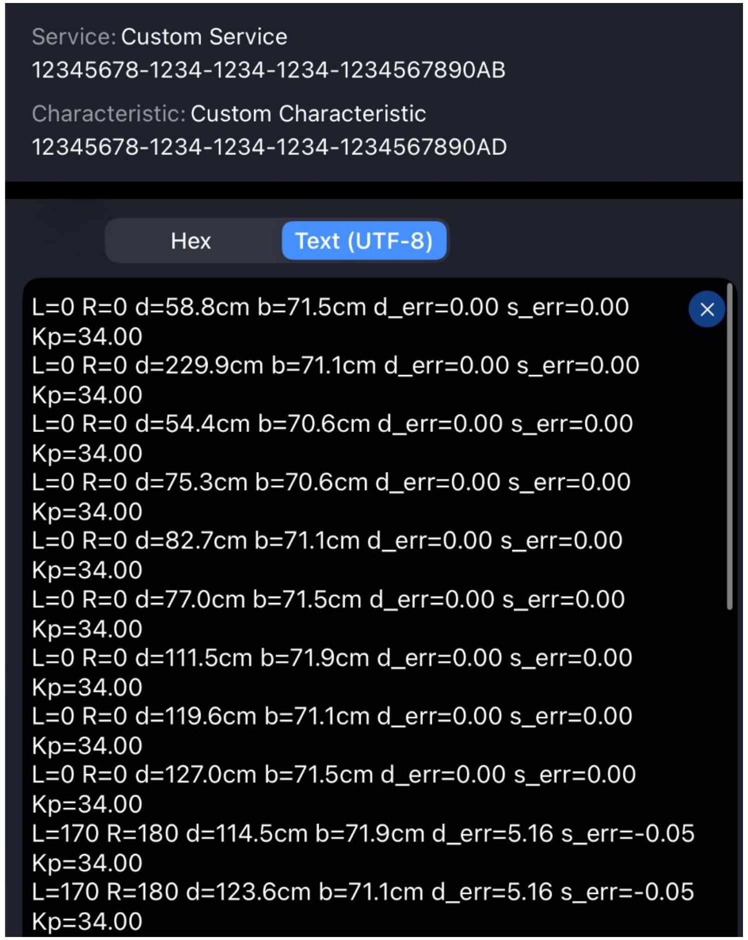
- User Input
- UI Buttons
- Real-time Display

ESP32

Vehicle Control Unit

- Motor Driver (TB6612)
- Dual Motors
- Front / Rear Ultrasonic

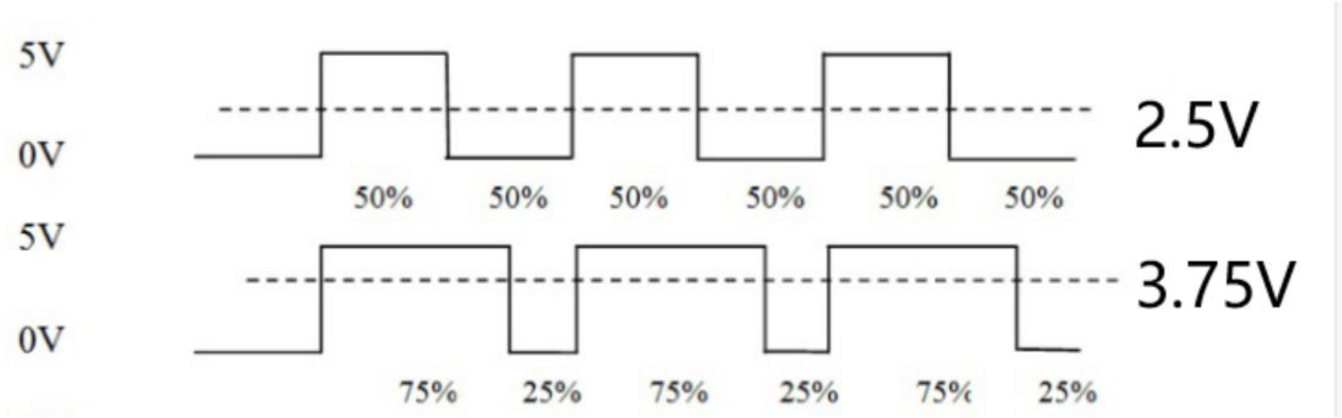


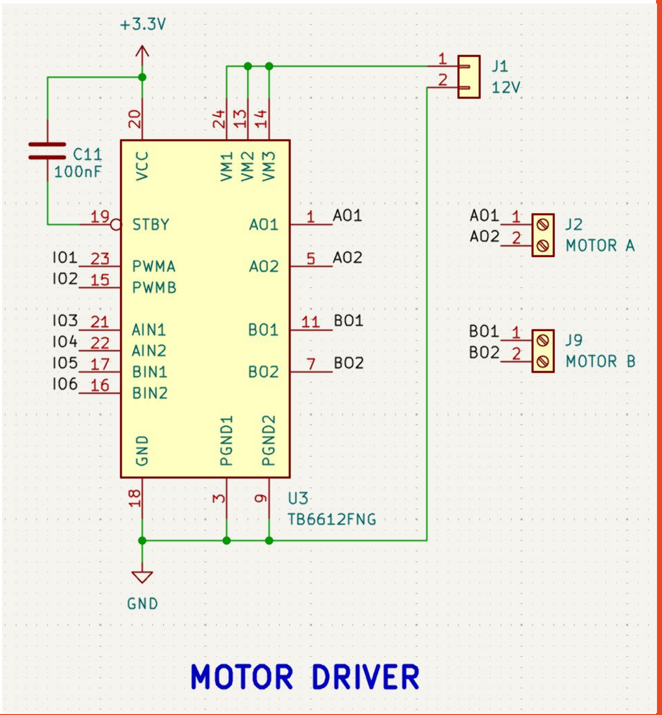


Driving system

Using a TB6612 motor driver to drive the system with PWM signal

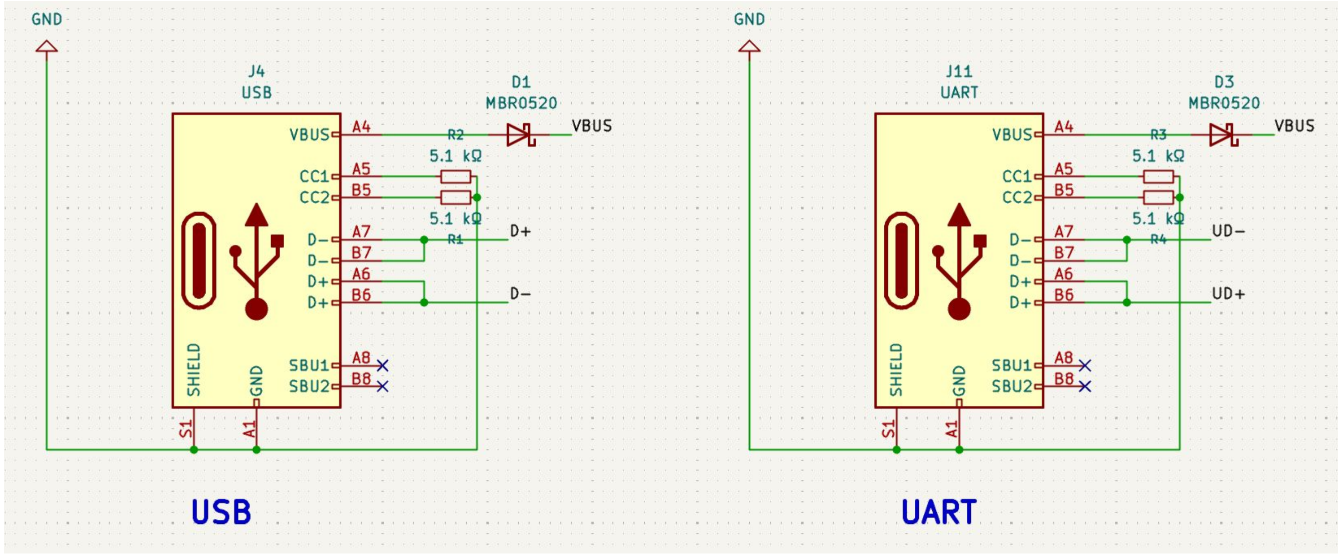
- Think of a light switch keeps flipping on and off
  - The more the switch is on, the higher voltage (brighter/faster) it will be
- Two wheels controlled parallelly

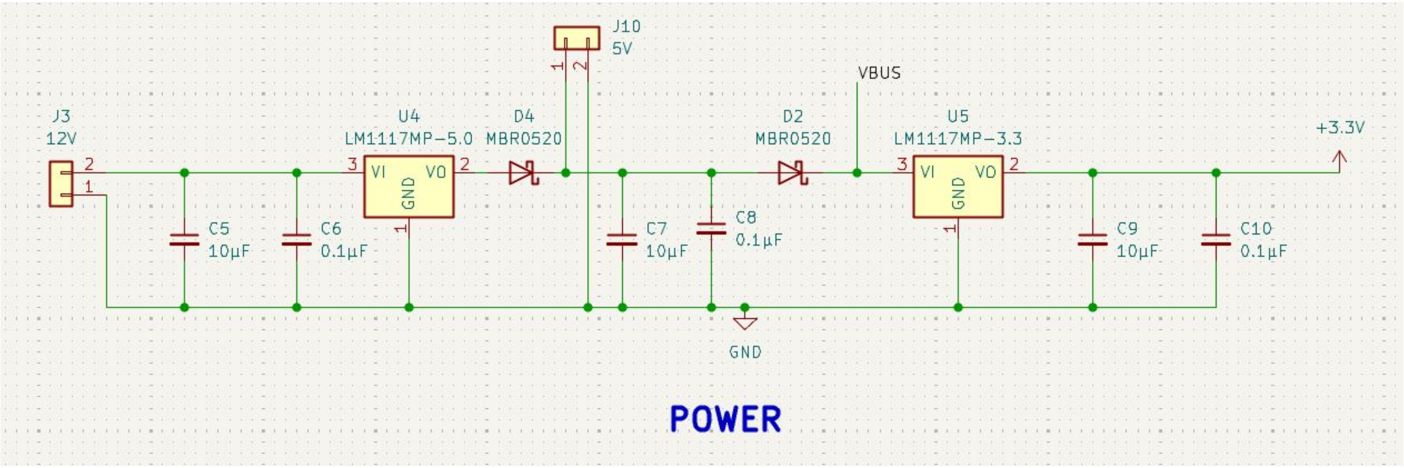


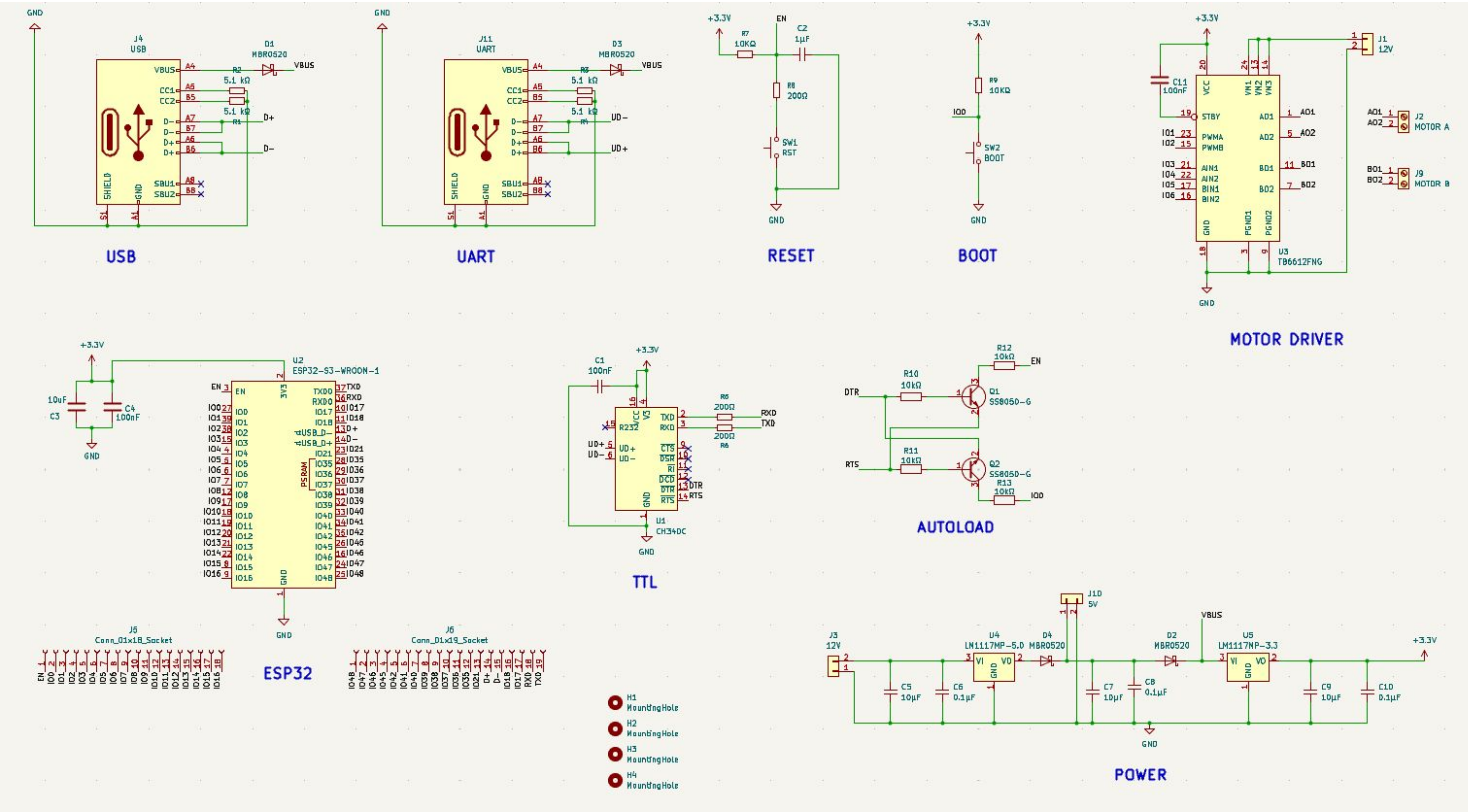


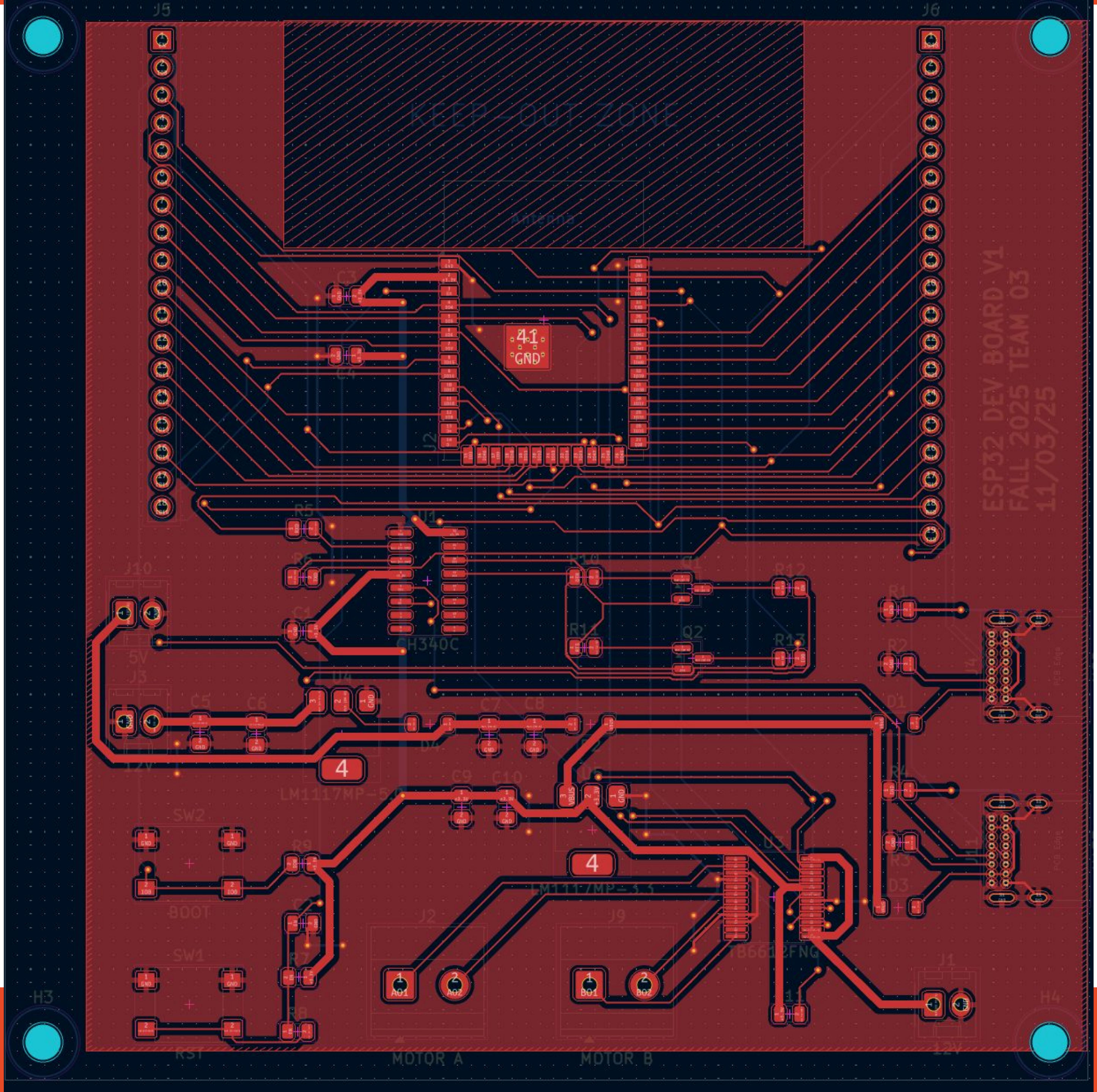
Power System

- Normal Power supply with linear regulators
- USB as an additional power supply and testing serial port
- Two USBs allowing the computer and Pi to be plugged in at the same time






USB
J4 USB
D1 MBR0520
VBUS
UART
J11 UART
D3 MBR0520
RESET
SW1 RST
BOOT
SW2 BOOT
MOTOR DRIVER
U3 TB6612FNG
J1 12V
J2 MOTOR A
J9 MOTOR B
U2 ESP32-S3-WROON-1
ESP32
J5 Conn_01x18_Socket
J6 Conn_01x19_Socket
U1 CH340C
TTL
Q1 SS8050-G
Q2 SS8050-G
AUTOLOAD
H1 MountingHole
H2 MountingHole
H3 MountingHole
H4 MountingHole
J3 12V
U4 LM1117MP-5.0
U5 LM1117MP-3.3
J10 5V
POWER

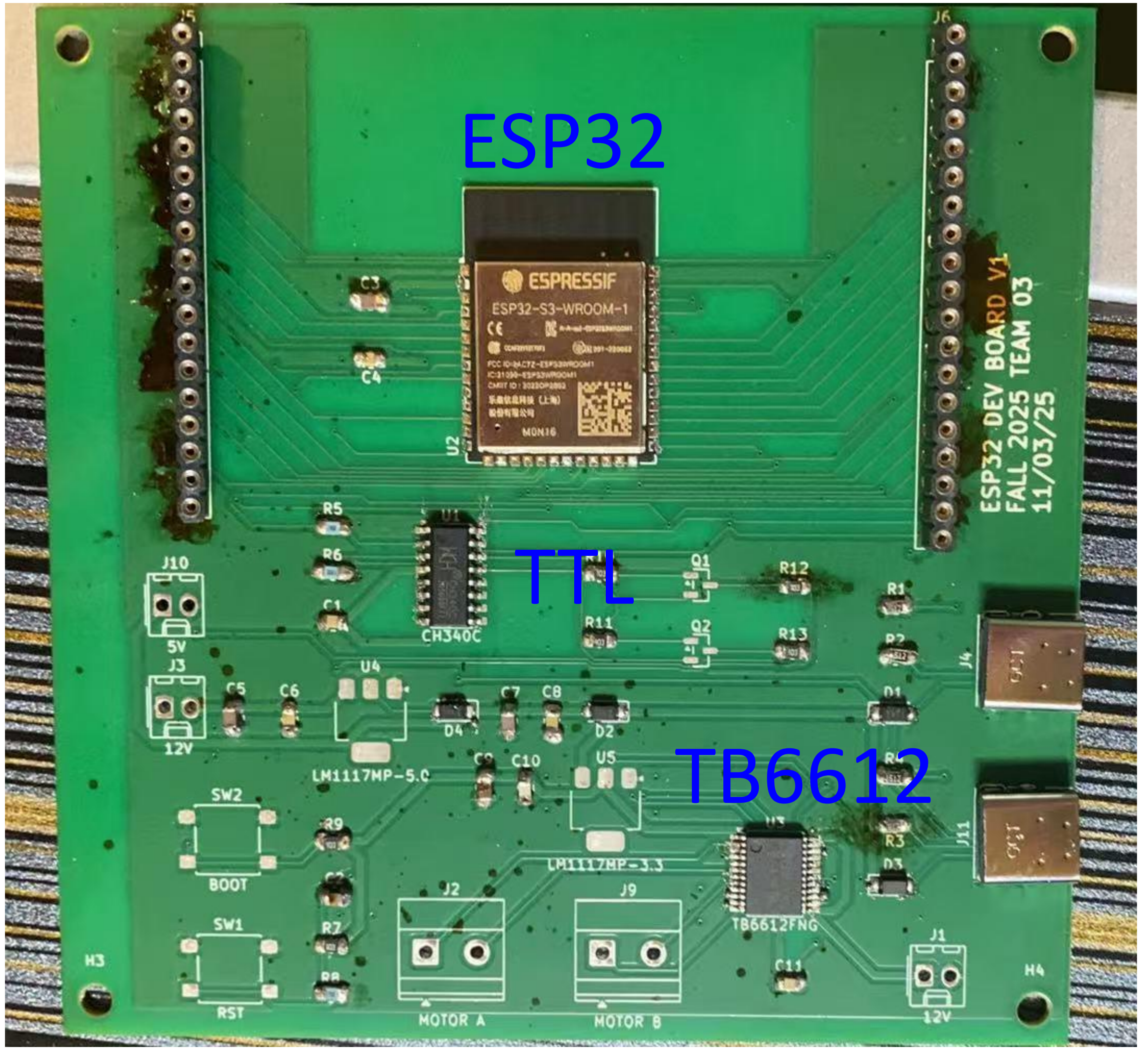

KEEP-OUT ZONE
ESP32 DEV BOARD V1
FALL 2025 TEAM 03
11/03/25
CH340C
LM1117MP-5.0
LM1117MP-3.3
TB6612FNG
BOOT
RST
MOTOR A
MOTOR B
12V
5V

Requirements & Verification Table

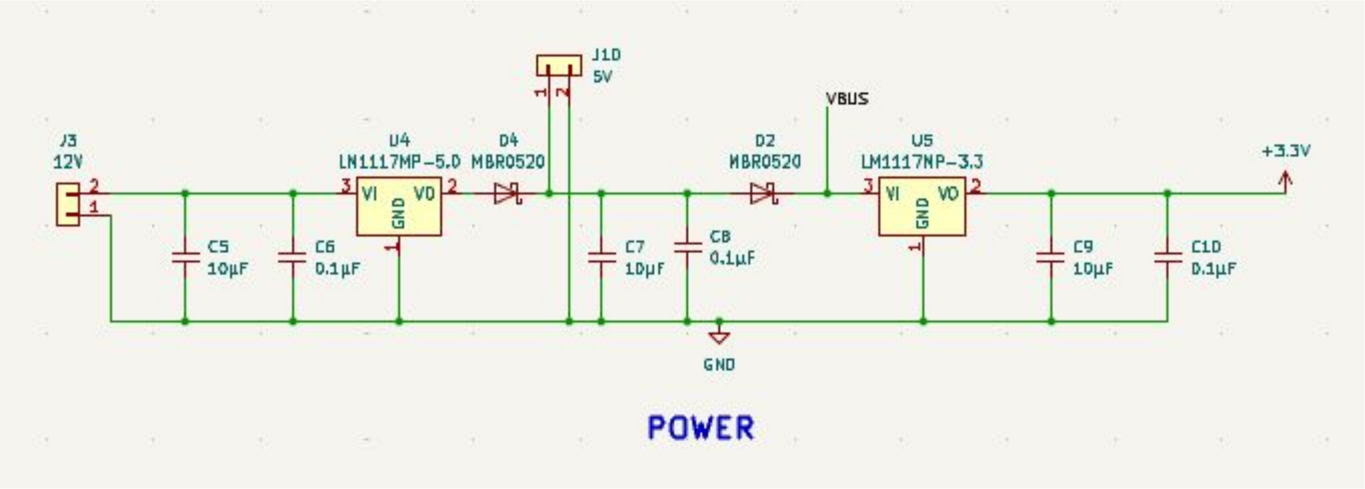
Requirements	Verification	Results
Safety Stop: When any radar sensor detects an obstacle closer than 0.5 m, the braking response must occur within 100 ms.	Conduct tests by let the cart move at various speeds towards an obstacle	Succeed, the ultrasound module will avoid collision when the distance is less than 70 cm
Reliable user lock: tracker keeps the same user ID for >=90% of frames during a 5-minute walk	Record a 2-min run; count ID switches in the log. Checking if overall ID consistency >=90%	Succeed. YOLO is pretrained with specific pattern, ensuring cart will only follow user with this pattern
Direction vector accuracy: bearing error ≤ 5° and distance MAE ≤ 0.25 m from 1–3 m.	Place markers at 1, 2, and 3 m and have the user stand on each marker. Report the mean error.	Partially succeed. YOLO would output the current distance correctly within around 2 m. And output the current angle in radians within tolerable error.
Command work reliably indoors to at least 5 m	Standing 1,3,5,7 m away from the cart, checks if the command really works.	Partly succeeded; the signal is not stable at 5 m but works for about 4 m.

Challenges - PCB

- Problem: The third round PCB arrived later than expected, leaving a relatively short time for debugging.
- Solution: Applied the original breadboard design.



Power

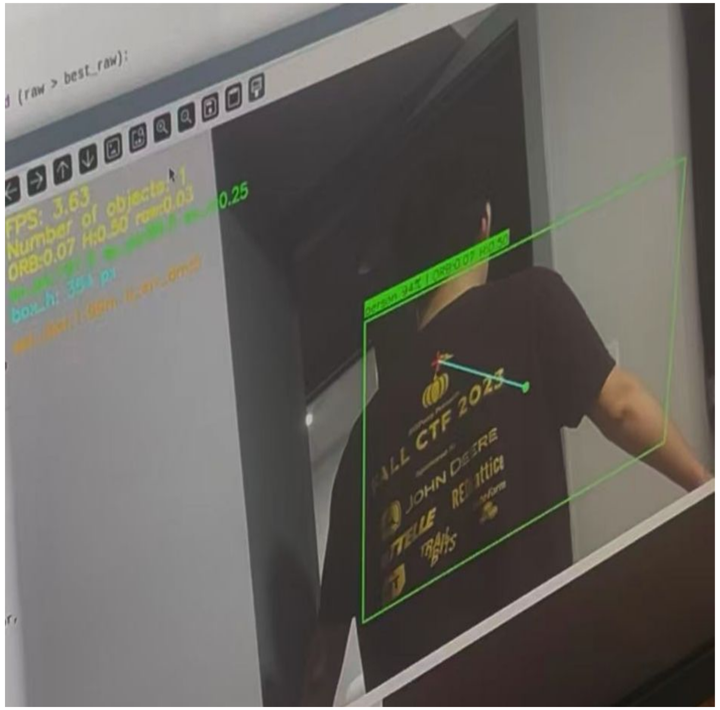


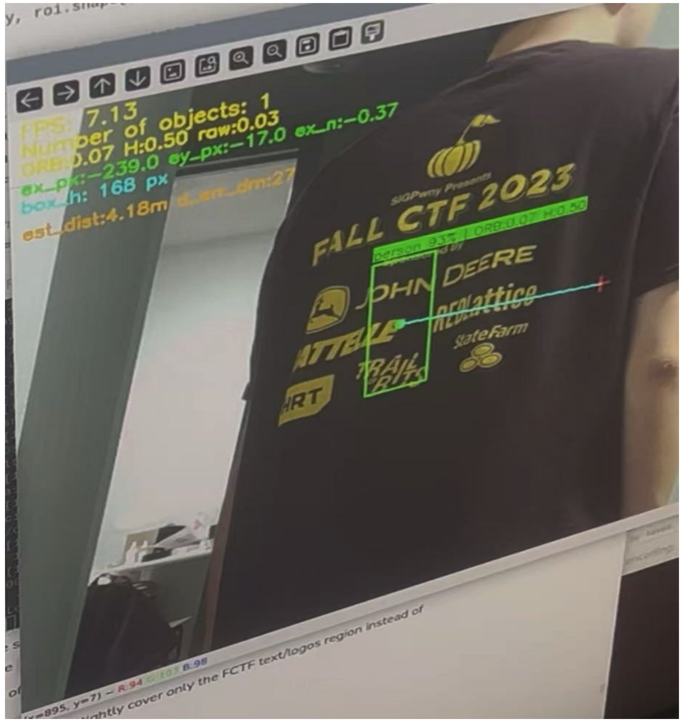
1. Input and Regulation: Accepts 12V and 5V input and generates 5V and 3.3V locally using LM117MP-5.0 linear regulators.

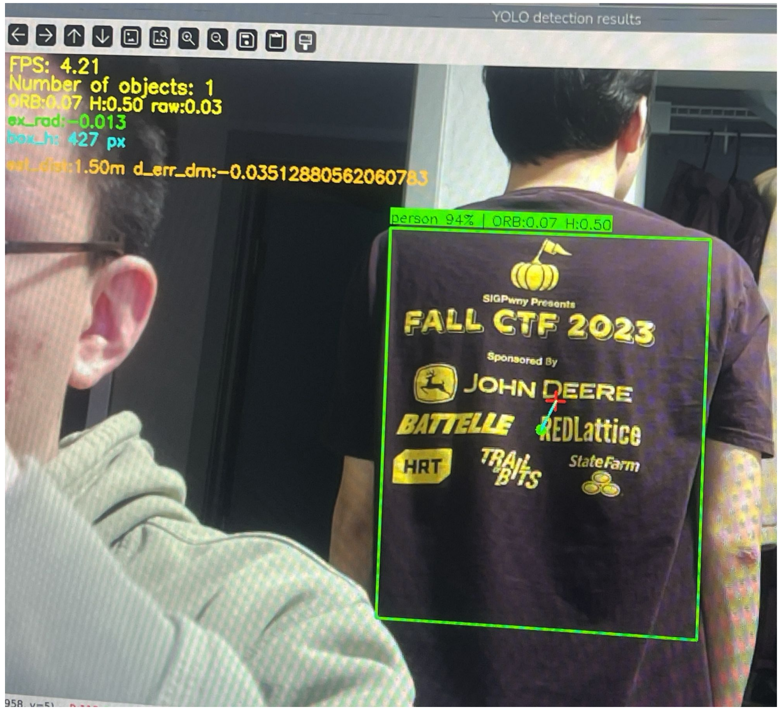
2. Decoupling and Stability: Each regulator uses **10 µF bulk + 0.1 µF ceramic** capacitors at input/output to maintain stability under load steps and cable inductance.

Challenges - YOLO

- Similar Clothing->Combine ORB+HSV histogram
- Unstable bounding boxes-> Develop an algorithm to dynamically change the size of box
- Low FPS on YOLO-> Model and pipeline optimization







Challenges - Cart

Problem: The cart does not drive in the desired direction after inputting the speed command to the microcontroller.

Solution: Securely affix the wheel in place by stretching the active axes in two directions (from inside and outside)

It took us a lot of time to figure out the reason why the cart is not moving in desired direction

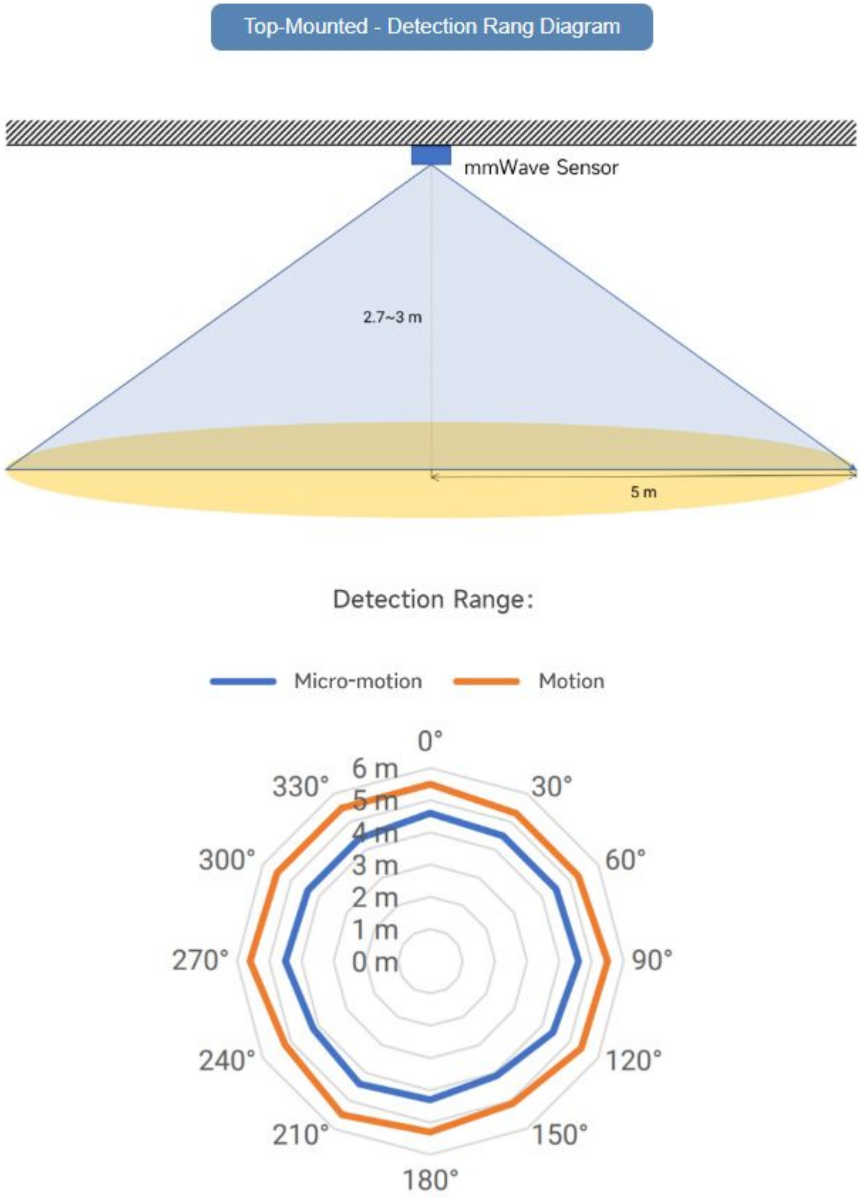
Drawback: Steering function damaged

Challenges - Sensors

Problem: Millimeter wave radars did not behave as expected, which was supposed to be used for obstacle detection:

1. broad and vague detection direction
2. unstable for detecting human body

Solution: Use ultrasonic sensor instead, which provides narrower range and a more reliable distance detection.



Conclusion

1.Developed a multi-sensor tracking system integrating **ultrasonic sensors**, **YOLO object detection**, and **pattern matching** for reliable person identification.

2. Demonstrated successful target-following during tests, even with multiple people in the scene.

3. Achieve real-time performance on Raspberry Pi and ESP32, and sort of support control of the follow-me platform.

Future Work

1. Improve sensor-fusion algorithms to better handle crowded or noisy environments.

2. Enhance pattern matching so that the box drawn around the pattern won’t fluctuate, which has a serious impact on the accuracy when calculating errors.

3. Integrate more efficient or quantized neural models to increase FPS on embedded hardware. So that the cart could follow the person smoothly.

Ethics and Safety

- Prevent Collisions with pedestrians (especially elderly and children)
- User control clarity and misuse prevention (“HOLD” function for disability)
- Safe motor behavior (E-STOP)
- Electrical Safety and Power Protection