World Robot Olympiad 2025

Future Engineers Category

Team Name: KCST Team

Institution: Kuwait College of Science and Technology

Team Members: Abbas Faisal Ali Muzaffar, Sara Hazem Salman, Qutiba Almanshad

Coach: Zainab Abualhassan

Date: May 14, 2025

Table of Contents

1 Introduction

The WRO 2025 Future Engineers category presents a challenging task where teams must design an autonomous robot capable of navigating a dynamic racetrack while avoiding obstacles and adhering to specific instructions. The robot must follow the track using color-coded traffic signs (pillars), where the **red pillar** instructs the robot to stay on the **right side** of the lane, and the **green pillar** instructs the robot to stay on the **left side**.

This project aims to develop an autonomous robot equipped with sensors and AI models to detect these color-coded pillars and navigate the track accordingly. The robot uses YOLOv11 for object detection, which allows it to identify and respond to these traffic signs in real time. Additionally, the robot must handle obstacles and perform autonomous parking once it reaches the parking zone.

This document provides a detailed overview of the components used in the design of the robot, the integration of these components, and the difficulties faced during development, including time constraints and academic responsibilities. A group photo of the team is shown in Figure 1.

Figure 1: Group photo annotated with team member names from left to right: Sara, Zainab, Abbas, Eng. Zainab (coach)

2 Robot Components

This section describes the key hardware components used in the autonomous robot. Each component plays a vital role in the functionality of the robot, from detecting the red and green pillars to controlling the motors for movement.

2.1 Raspberry Pi 4 Model B

The Raspberry Pi 4 Model B is the heart of the autonomous robot. It serves as the central processing unit (CPU), responsible for managing all functions, including running the YOLOv11 model for object detection, processing sensor data, and controlling the robot's motors. The Raspberry Pi communicates with the AI camera, receives sensor data, and sends commands to the motors based on the detected pillars, as shown in Figure 2.

Key Features:

- 4GB RAM for multitasking and running computationally intensive models.
- Quad-core ARM Cortex-A72 processor, capable of running deep learning models and handling real-time decision-making.
- HDMI and USB ports for additional peripherals.
- Built-in Wi-Fi and Bluetooth connectivity for remote access and data transfer.
- GPIO pins for interfacing with various sensors and motors.

Figure 2: Raspberry Pi 4

2.2 Motor Driver Board

The motor driver board is used to control the robot's motors based on the commands received from the Raspberry Pi. It acts as an intermediary, providing the necessary power to drive the motors and execute movement commands such as forward, backward, and steering, as shown in Figure 3.

Key Features:

- Controls up to 2 DC motors for movement and steering.
- Operates at voltages between 5V and 12V, supplying the necessary current for the motors.
- Enables smooth control of motor speed and direction.
- Ensures that the robot responds promptly to the Raspberry Pi's commands.

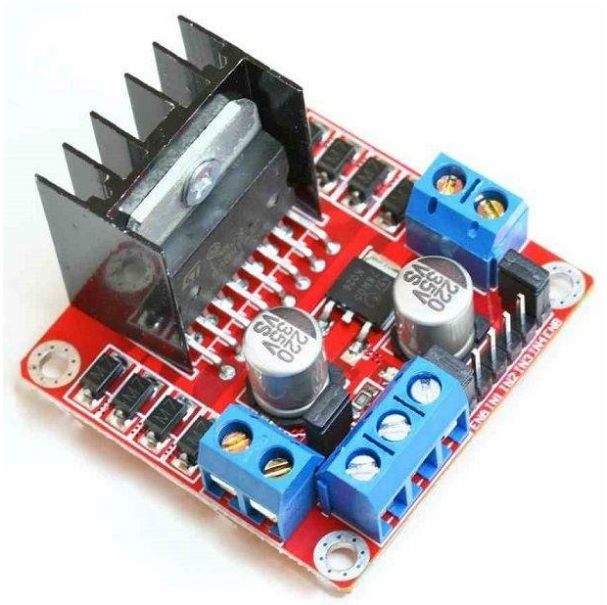

Figure 3: Motor Driver Board

2.3 Ultrasonic Sensors

Ultrasonic sensors are essential for obstacle detection. These sensors emit sound waves and measure the time it takes for the waves to bounce back after hitting an obstacle, allowing the robot to calculate the distance to objects in its path. Positioned at the front of the robot, the sensors help it avoid collisions while navigating, as shown in Figure 22.

Key Features:

- Measures distances from 2 cm to 400 cm with high accuracy.
- Provides real-time data to help avoid obstacles.
- Low-cost and efficient method for obstacle avoidance.

Figure 4: Ultrasonic Sensor

2.4 AI Camera (Sony IMX500)

The AI camera (Sony IMX500) is used to capture real-time images of the track. The camera feeds the images to the Raspberry Pi, where they are processed by the YOLOv11 object detection model to identify the red and green pillars. These color-coded pillars guide the robot's movement along the track. The camera is also used to detect obstacles and aid in parking, as shown in Figure 5.

Key Features:

- High-resolution imaging sensor for accurate object detection.
- Supports real-time object detection using YOLOv11.
- Direct AI processing on the camera, reducing computational load on the Raspberry Pi.
- Easily integrates with Raspberry Pi for control.

Figure 5: AI Camera (Sony IMX500)

2.5 Power Supply

The power supply is a crucial component that ensures the robot remains operational during the competition. The rechargeable lithium-ion battery provides the necessary power for the Raspberry Pi, motors, and sensors. The battery's capacity ensures that the robot can operate for an extended period without requiring a recharge, as shown in Figure 6.

Key Features:

- Lithium-ion rechargeable battery for long operational time.
- Provides stable voltage (5V and 12V) to power all components.
- Efficient power management to maximize battery life during competition.

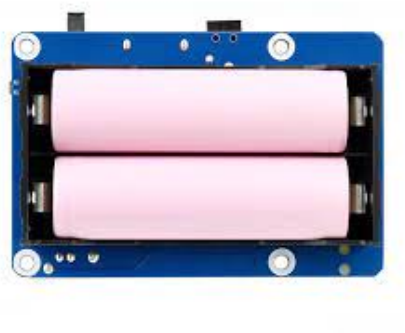

Figure 6: Power Supply

3 Assembly Process

The assembly process for building the robot involved several key steps, each requiring careful planning and coordination. Below is a step-by-step breakdown of how the robot was assembled:

3.1 Car Base Setup

We started by preparing both the lower and upper car bases. These serve as the foundation for attaching all other components, as shown in Figure 7.

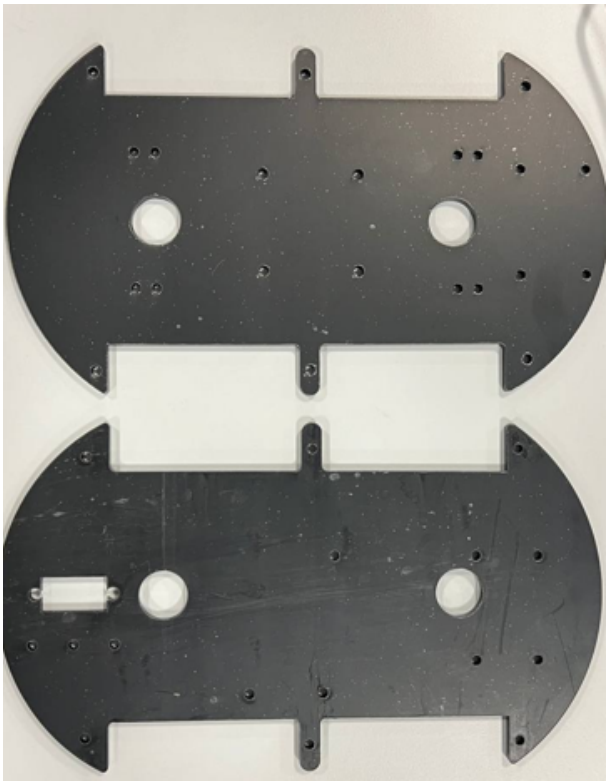

Figure 7: Lower and upper base preparation

3.2 Installing the Motors

Two motors were attached to the lower base using nails. Yellow nails were placed at the edges to help connect the lower and upper bases later on, as seen in Figure 8.

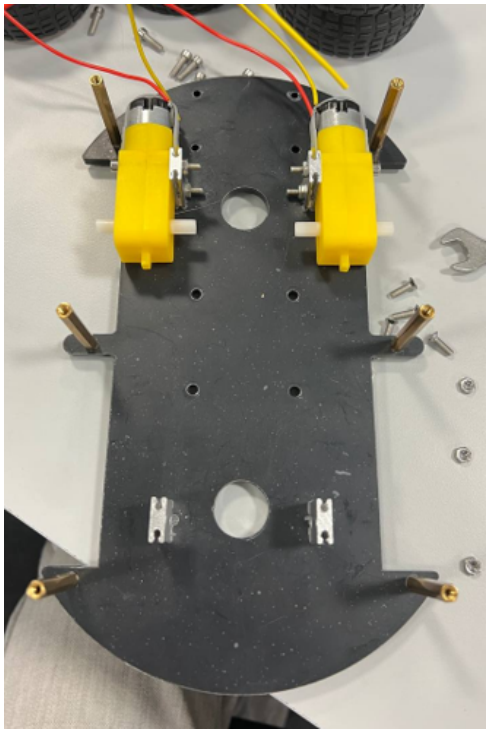

Figure 8: Motors installed with nails

3.3 Mounting the Wheels

Each motor was connected to one wheel, enabling movement on both sides (Figure 9).

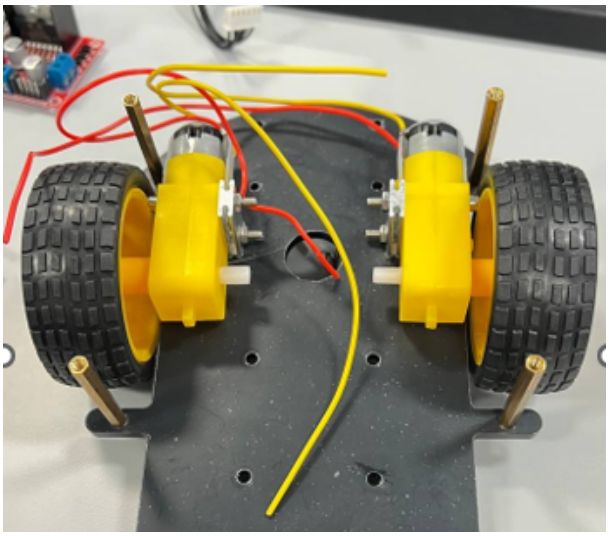

Figure 9: Wheels mounted on motors

3.4 Connecting the Motor Driver

The motors were wired to an L298N motor driver using the OUT1, OUT2, OUT3, and OUT4 terminals, as shown in Figure 10.

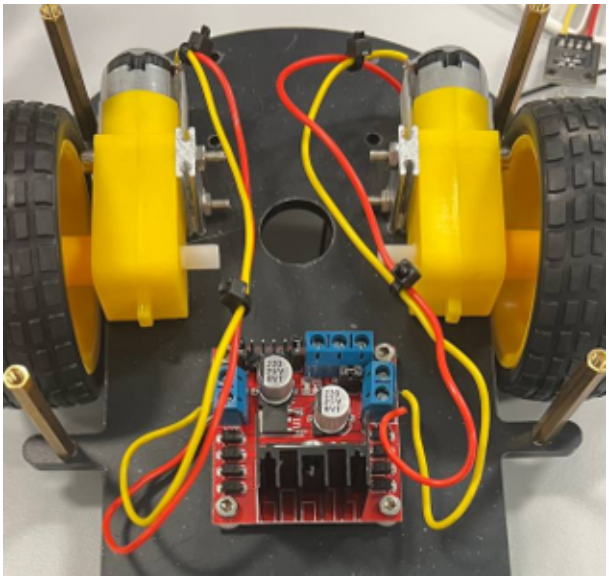

Figure 10: L298N motor driver connections

3.5 Rear Wheel Mechanism

We added an axle with nails to link the two rear wheels, since we were limited to two motors (Figure 11).

Figure 11: Rear axle linkage

3.6 Initial Prototype (Cancelled)

We initially glued the wheels to the axle, but it didn't work well and was discarded, as shown in Figure 12.

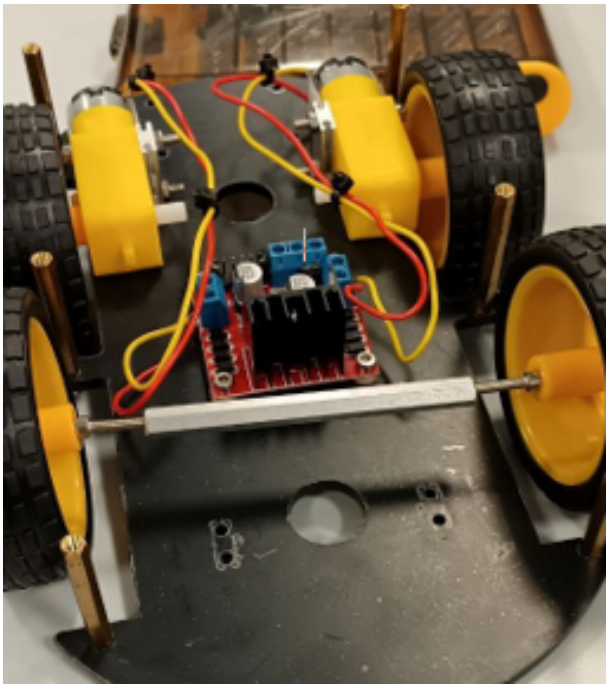

Figure 12: Cancelled prototype version

3.7 Exploring Back Wheel Solutions

We tested extra motor placements to better support the rear wheels (Figure 13).

Figure 13: Rear support testing with extra motors

3.8 Custom Axle Holder

We built a LEGO structure to support the axle and allow smooth rotation, as shown in Figure 14.

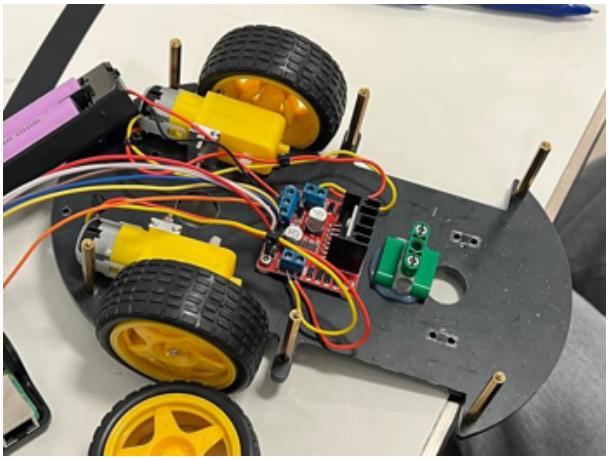

Figure 14: LEGO axle holder

3.9 Final Rear Wheel Setup

The final version included glued wheels securely fixed to the axle (Figure 15).

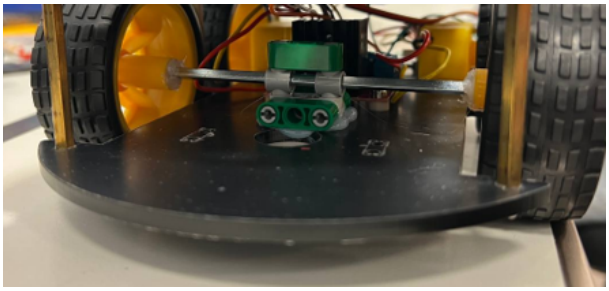

Figure 15: Final rear wheel setup

3.10 Camera Integration (Raspberry Pi)

We installed the Raspberry Pi camera to support AI object detection, as seen in Figure 16.

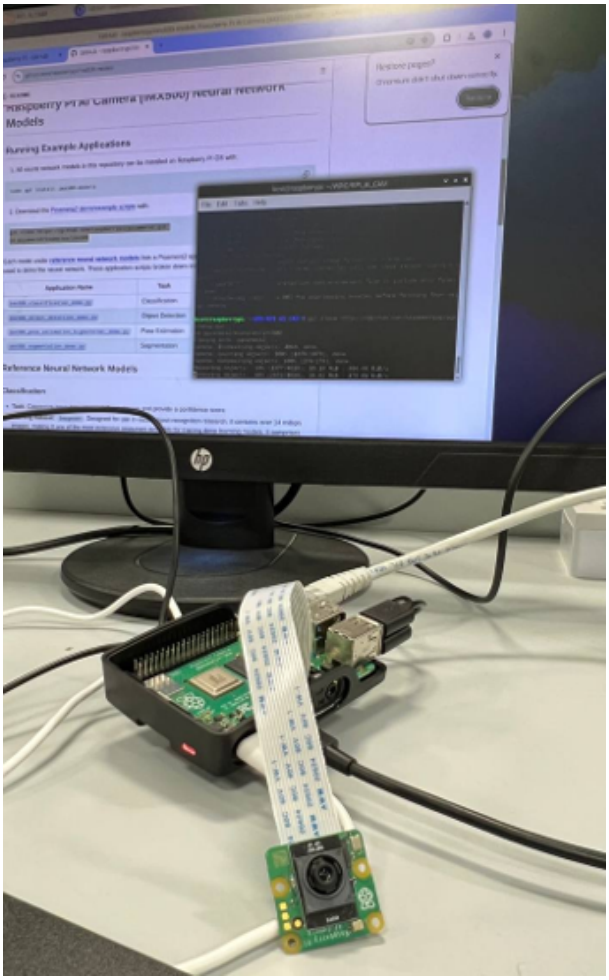

Figure 16: Installing Raspberry Pi camera

3.11 Setting Up the Camera

The camera was connected to the Raspberry Pi system (Figure 17).

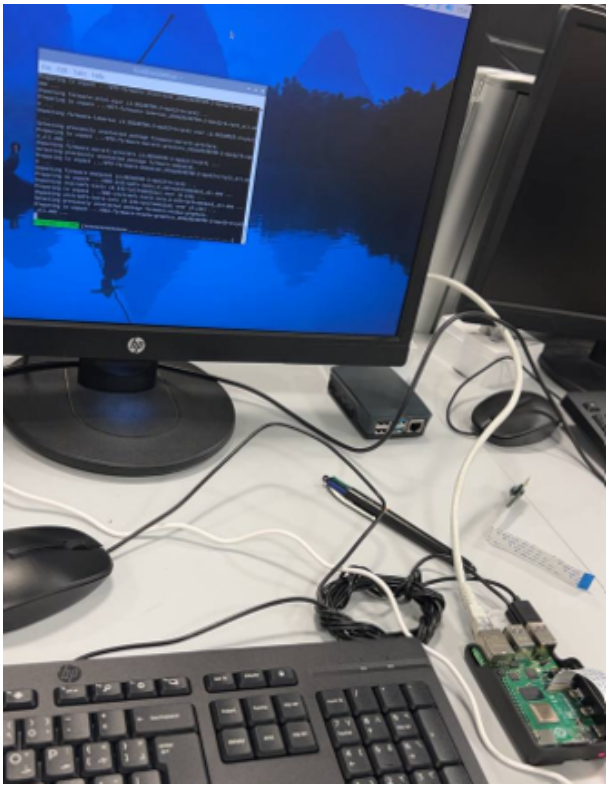

Figure 17: Camera connection complete

3.12 Camera Testing

We programmed and tested the camera. It worked as expected, as shown in Figure 18.

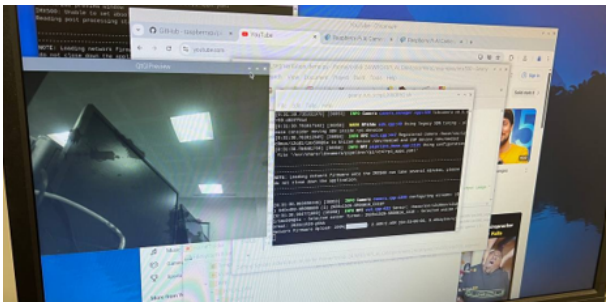

Figure 18: Camera test success

3.13 Camera Calibration and Focus Adjustment

To ensure optimal image clarity and accurate object detection, we manually calibrated the AI camera by adjusting its lens focus. This step was essential to reduce blur and improve detection performance, especially under varying lighting conditions.

Figure 19 shows the calibration process during which the focus ring was carefully turned while monitoring the real-time video feed.

Figure 19: Manual calibration of the AI camera by adjusting the lens focus

3.14 Object Detection Feature

The camera was able to detect objects using AI (Figure 20).

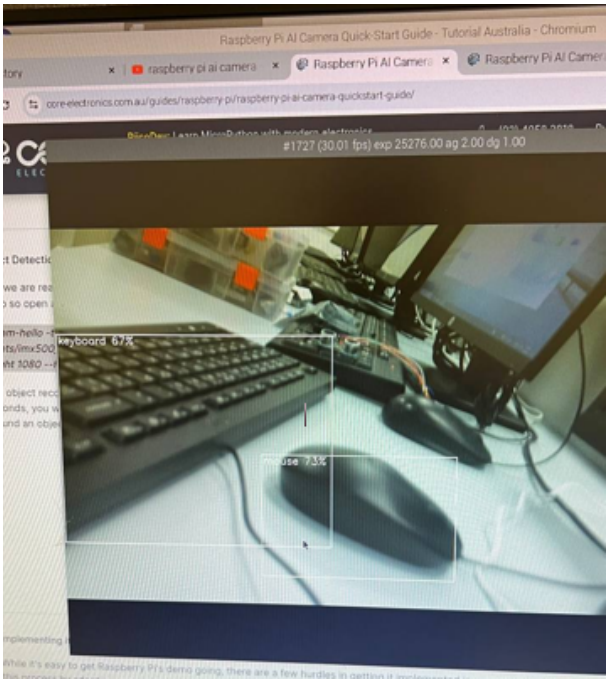

Figure 20: Object detection in action

3.15 Upper Base and Wiring

We installed the upper base and connected the motor driver to the Raspberry Pi, as seen in Figure 21.

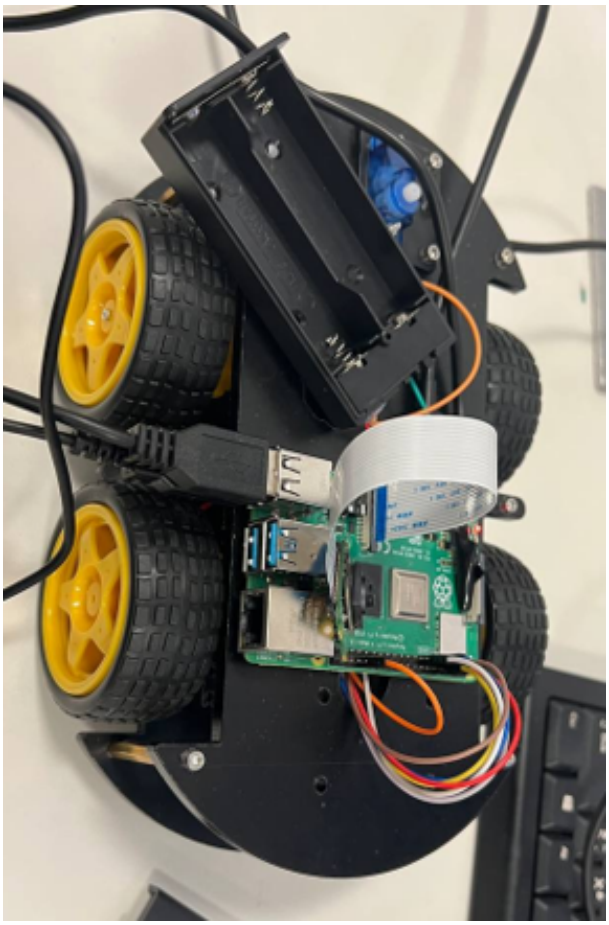

Figure 21: Wiring and upper base in place

3.16 Additional Sensors

An ultrasonic sensor and battery holder were attached at the front of the car (Figure 22).

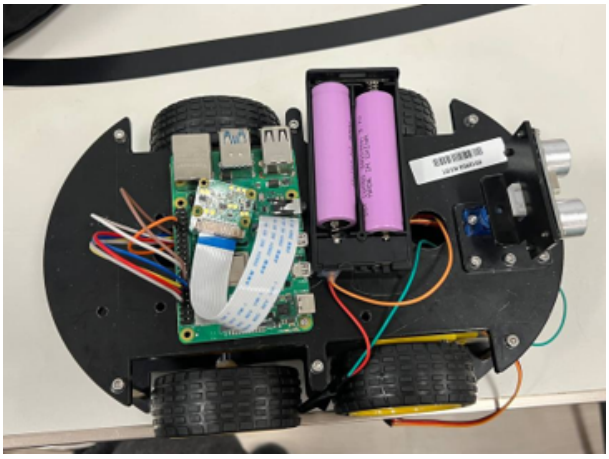

Figure 22: Ultrasonic sensor installed

3.17 Ultrasonic Sensor Programming

The ultrasonic sensor was programmed to detect objects in front of the car, as shown in Figure 23.

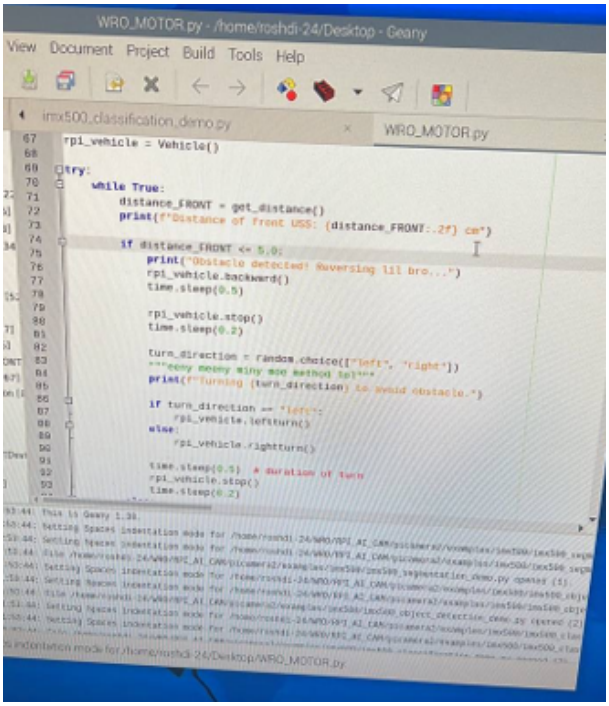

Figure 23: Ultrasonic sensor in action

3.18 Final Design Overview

Final images showing the complete design from different angles are illustrated in Figure 24.

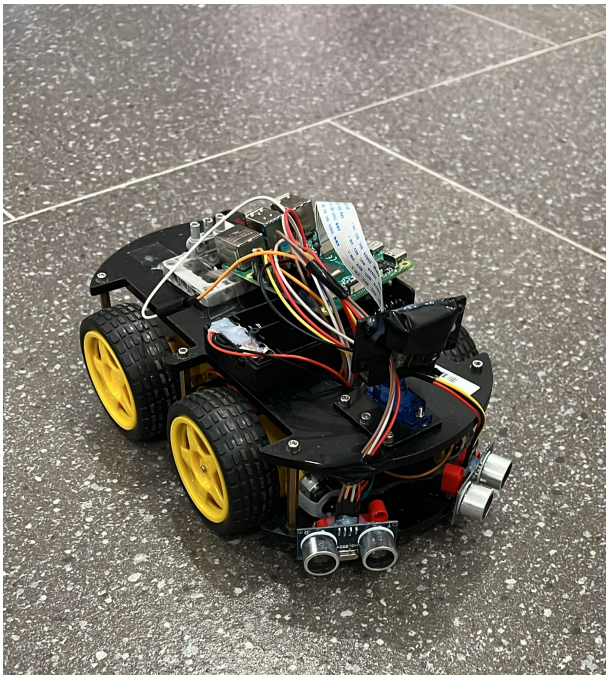

Figure 24: Final smart car design

4 Object Detection Model Training and Deployment

4.1 Dataset Preparation

To enable the robot to detect red and green pillars and parking boxes, a labeled dataset was created using RoboFlow. The dataset consisted of three object classes: `greenbox`, `redbox`, and `xparking`, with a total of over 5,000 annotated images. The distribution of images across the three classes is shown in Figure 25.

The dataset is publicly available at: `https://universe.roboflow.com/your-dataset-link`

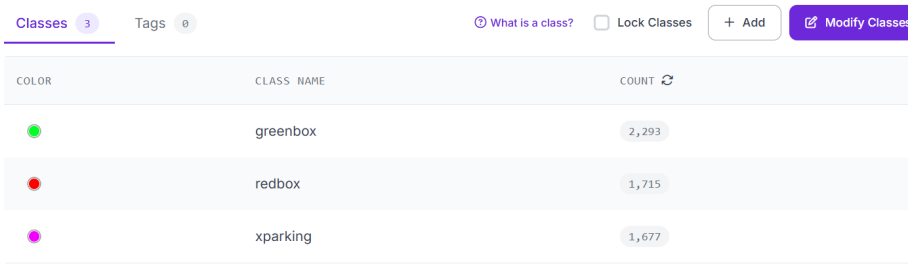

Figure 25: Class distribution and image count for the object detection dataset

The images were manually annotated with bounding boxes around the relevant objects and exported in the YOLO format to ensure compatibility with the YOLOv11 training pipeline. A sample of the labeled dataset is shown in Figure 26.

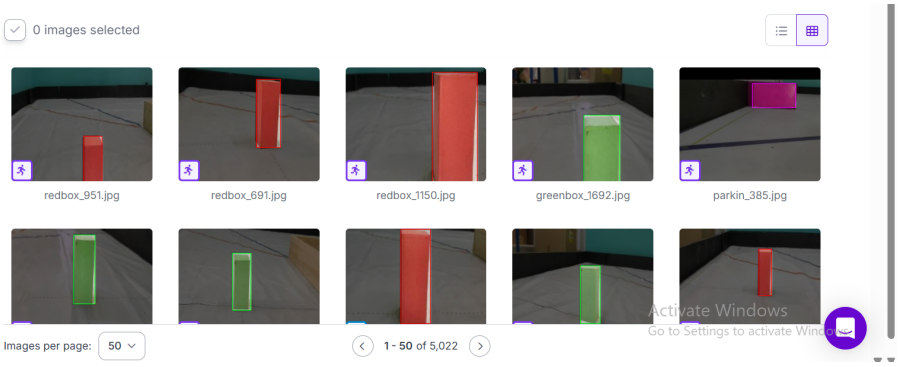

Figure 26: Example images from the labeled RoboFlow dataset prepared for YOLOv11 training

4.2 Model Training

The YOLOv11 model was selected to train on the prepared dataset. Training was performed on a GPU-enabled workstation to ensure faster convergence and better performance. The model was trained following the AI camera's documentation, optimizing it for real-time detection of color-coded signals. The training process is illustrated in Figure 27.

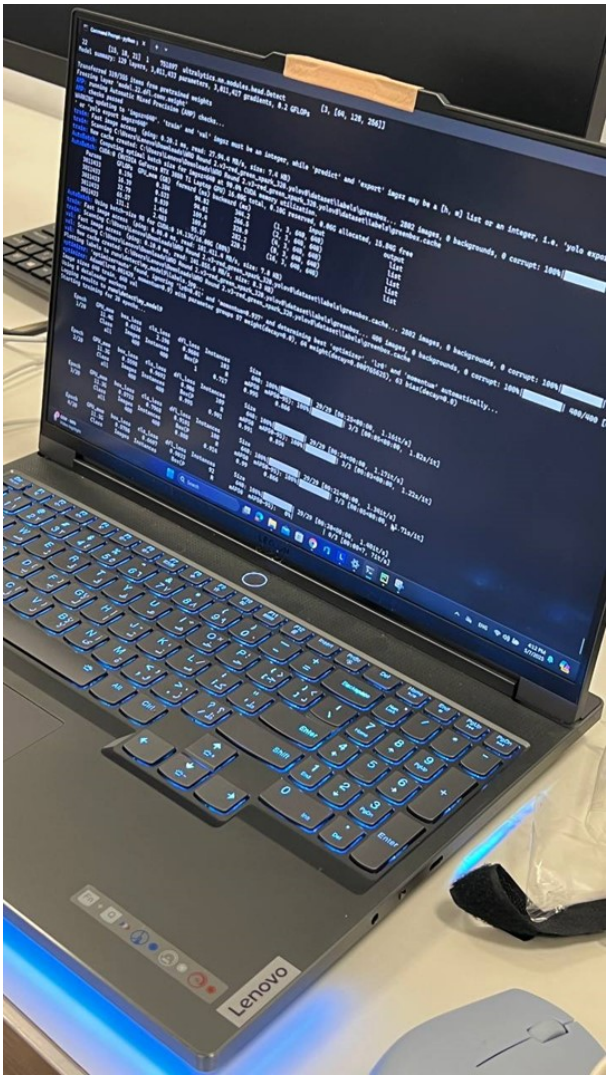

Figure 27: Training the YOLOv11 model on a GPU-enabled computer using the RoboFlow dataset

4.3 Model Testing and Deployment

After deploying the converted model onto the Sony IMX500 AI camera, the robot was tested in real-time conditions to verify its ability to detect and classify red and green pillars and parking boxes.

Figure 28 shows the overall setup, with the robot and AI camera connected to a monitor for visual feedback. In Figure 29, we observe a successful detection of red and green objects directly on the Raspberry Pi interface after training. Finally, Figure 30 demonstrates the real-time generation of the `detections.json` file, which contains the label and coordinates of the detected object. This file is then parsed by the robot's control system to make movement decisions.

Figure 28: Complete testing setup with real-time visual detection of the green pillar

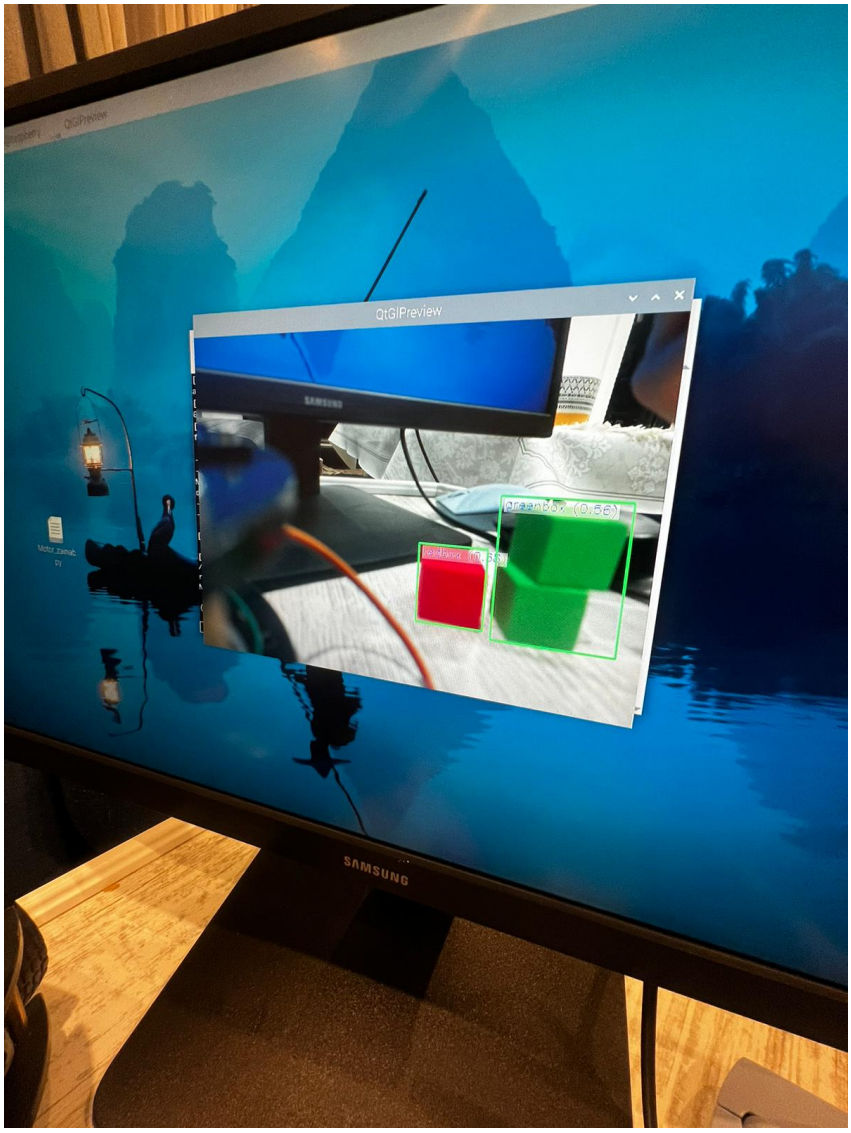

Figure 29: Detection results shown on the Raspberry Pi using the trained model

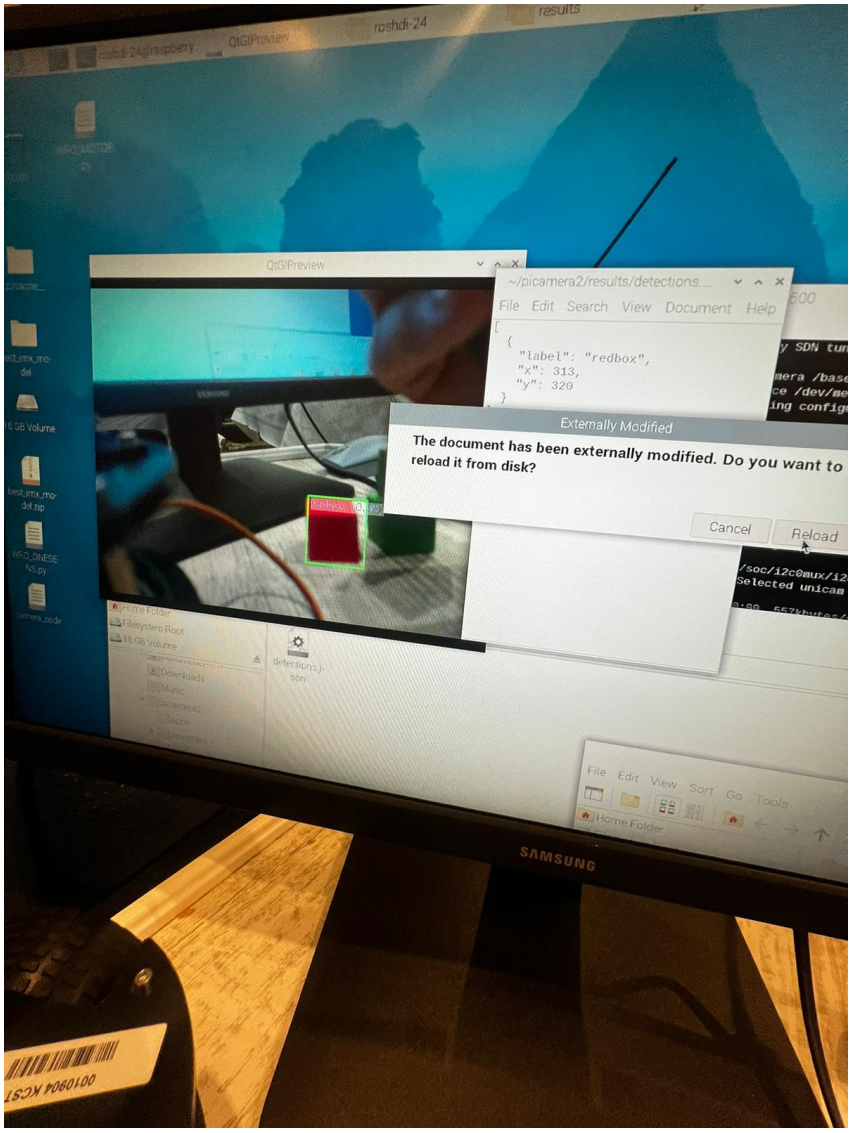

Figure 30: JSON file generated on the Raspberry Pi with detected object label and coordinates

5 Difficulties Faced

Throughout the development of the robot, several challenges were encountered. These included both technical and non-technical difficulties that affected the pace and quality of the project.

5.1 Lack of Time

One of the primary challenges faced was the lack of sufficient time to complete all the tasks. As the competition deadline approached, the complexity of integrating the various components (YOLOv11 for object detection, sensors for obstacle avoidance, motor control, etc.) and fine-tuning the robot's behavior increased. Time constraints significantly impacted the amount of testing and optimization that could be done, which is critical for autonomous systems that rely on precise decision-making.

5.2 University Responsibilities

Alongside the development of the robot, university responsibilities, including teaching and grading duties, took up a significant portion of time. This made it difficult to dedicate sufficient hours to the robot's development and testing. Balancing academic work and the demands of the robot development project proved challenging.

5.3 Final Exams and Deadlines

The timing of the project coincided with my final exams and academic deadlines, which placed additional pressure on the project timeline. The workload from studying for exams, preparing assignments, and fulfilling academic responsibilities delayed some parts of the testing and fine-tuning process. It also added stress to the overall project management, limiting the time available for troubleshooting and adjustments.

6 GitHub Repository (In Progress)

As part of our documentation and collaboration efforts, we are in the process of organizing and uploading all scripts, configuration files, and trained models to a public GitHub repository. This repository will include:

- Python code for camera initialization and detection parsing.
- The trained YOLOv11 model converted to `.imx` format.
- Instructions for running the object detection system on the Raspberry Pi.
- Shell scripts for auto-running detection and movement control.

The GitHub repository will help other teams or developers replicate our workflow and build similar AI-based autonomous systems. The current version of the setup guide we followed is publicly available at: `https://github.com/Aizu-x/WRO-2025`

7 Future Improvements (In Progress)

The current integration of the AI camera, ultrasonic sensor, and motor control system has yielded promising results. As shown in previous figures, the robot can successfully detect objects and respond accordingly using real-time JSON output from the detection system.

However, to further enhance performance and reliability, we plan to:

- Add two additional ultrasonic sensors on the left and right sides of the robot to improve obstacle detection and side avoidance.
- Optimize the movement control algorithm by adjusting the motor PWM values for smoother and faster navigation.
- Experiment with increasing the robot's overall speed while maintaining safe turning and braking based on object distance.

- Improve dataset quality by adding more diverse scenarios for parking boxes and angled object views.
- Upload and maintain a complete GitHub repository to support ongoing collaboration and testing.

These enhancements are intended to better prepare the robot for the dynamic and unpredictable conditions expected during the WRO 2025 competition.

8 Conclusion

The autonomous robot designed for the WRO 2025 Future Engineers category integrates various advanced components, including the Raspberry Pi, AI camera, motor driver, ultrasonic sensors, and more, to navigate the track and perform autonomous parking. YOLOv11 plays a critical role in detecting the red and green pillars, guiding the robot to stay in the correct lane.

Despite the challenges faced, such as time constraints, academic responsibilities, and final exams, the project was successfully completed. Future improvements will focus on optimizing the power management system, enhancing the object detection accuracy, and refining the robot's ability to adapt to dynamic environments.

B. Template Technical Summary

Team name	KCST Team
Team number	– (To be assigned)
Team members	Abbas Faisal Ali Muzaffar, Sara Hazem Salman, Qutiba Almanshad
Team coach	Zainab Abualhassan
Robotic set	Self-assembled (Custom platform using Raspberry Pi and various sensors)
Size	24 cm (length) × 15 cm (width) × 14 cm (height)
Weight	1.1 kg
Building materials	Acrilic base, some lego components
Controllers	Raspberry Pi 4 Model B
Battery	7.5 V 2200 mAh Lithium-ion battery pack
Sensors	2× Ultrasonic sensors, Sony IMX500 AI camera
Motors	2× DC motors with driver board (L298N)
Pneumatic System	Not used
Programming Environment & Language	Python, YOLOv11, OpenCV
Picture of robot