KLE Technological University | Creating Value, Leveraging Knowledge

Dr. M. S. Sheshgiri Campus, Belagavi

2024-2025

Department of Electronics and Communication Engineering

Minor Project Report

on

Self Driving Car Using Lane and Object Detection

By:

1. **Amogh Gadad** USN: 02FE22BEC006
2. **Amruta Biradarpatil** USN: 02FE22BEC008
3. **Ashtami Hosapeti** USN: 02FE22BEC012
4. **Naman Timmapur** USN: 02FE22BEC039

Semester: VI, 2024-2025

Under the Guidance of

Prof. Anita Shirol

KLE Technological University
Dr. M. S. Sheshgiri College of Engineering and Technology, Belagavi - 590 008

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"Self Driving Car Using Lane and Object Detection"** is a bonafide work carried out by the student team of

Amogh Gadad (02FE22BEC006), Amruta Biradarpatil (02FE22BEC008), Ashtami Hosapeti(02FE22BEC012), Naman Timmapur (02FE22BEC039).

The project report has been approved as it satisfies the requirements with respect to the minor project work prescribed by the university curriculum for B.E. (VI Semester) in the Department of Electronics and Communication Engineering of KLE Technological University, Dr. M. S. Sheshgiri College of Engineering and Technology, Belagavi campus for the academic year 2024–2025.

Prof. Anita Shirol	**Dr. Dattaprasad.Torse**	**Dr. S. F. Patil**
Guide	Head of Department	Principal

External Viva:

Name of Examiners	**Signature with Date**
1.	
2.	

Acknowledgment

We would like to express our sincere gratitude to our respected guide, **Prof. Anita Shirol**, for her invaluable guidance, continuous encouragement, and steadfast support throughout the duration of our project, **"Self-Driving Car Using Lane and Object Detection."** Her expert insights, constructive suggestions, and consistent mentorship played a pivotal role in steering our efforts toward meaningful outcomes.

We also extend our deep appreciation to **Prof. Vidyadar Dodamani** and **Prof. Shivanand Uppari** for their significant contributions and technical expertise in the field of Automotive Electronics, which greatly enriched the quality of our work. Their detailed feedback and scholarly advice were instrumental in overcoming various challenges and ensuring the project's successful implementation.

The collective support and guidance of our mentors have been fundamental to the completion of this project, and we remain truly grateful for their mentorship and commitment to academic excellence.

-The Project Team

Abstract

This project presents a self-driving car prototype developed using a Raspberry Pi 4, capable of performing lane detection and object detection for autonomous navigation. A webcam captures real-time road images, which are processed using OpenCV to identify lane boundaries. Lightweight object detection models, such as YOLOv5n, enable the system to detect and respond to obstacles effectively.

For collision avoidance, ultrasonic sensors are placed at the front, left, and right to measure distances from nearby objects. Vehicle movement is managed by DC motors controlled via an L298N motor driver. Additionally, a VPN-based webpage enables live video streaming for remote monitoring. This prototype offers a cost-effective solution for implementing basic autonomous driving features.

TABLE OF CONTENTS

Contents

Chapter 1

Introduction

The advancement of autonomous vehicle technology is transforming the future of transportation. Self-driving cars are designed to navigate and make decisions with minimal human intervention, using a combination of sensors, cameras, and intelligent algorithms. This project aims to create a small-scale, cost-effective self-driving car prototype using Raspberry Pi 4. The system integrates computer vision for lane detection, object detection for obstacle recognition, and ultrasonic sensors for distance measurement. By using affordable hardware such as a webcam, DC motors, and ultrasonic sensors, the project demonstrates a simplified yet functional approach to autonomous driving. A VPN-based webpage is also developed to stream the car's real-time view, enabling remote monitoring and control.

1.1 Motivation

Designing a low-cost self-driving car system offers students and developers a hands-on opportunity to explore real-world applications of embedded systems, computer vision, and robotics. Commercial autonomous vehicles are highly sophisticated and expensive, making it difficult to experiment with their technology in academic settings. By leveraging Raspberry Pi 4 and affordable sensors, this project makes it possible to replicate essential features of autonomous driving, such as lane following and obstacle detection. The motivation lies in creating a compact, scalable platform that not only demonstrates the core principles of self-driving technology but also encourages innovation and learning through practical implementation.

1.2 Objectives

- **To design and construct a working prototype of a self-driving car using Raspberry Pi 4.**
  This prototype will serve as a platform for testing autonomous driving concepts in real-time.

- **To implement lane detection using a webcam and OpenCV image processing techniques.**
  The system will recognize lane boundaries and help the vehicle maintain its course.

- **To integrate object detection to identify and classify obstacles in the car's path.**
  This ensures the vehicle can make decisions to avoid collisions with dynamic or static objects.

- **To use three ultrasonic sensors for front, left, and right distance measurement.**
  These sensors will enable precise obstacle detection and safe navigation in tight spaces.

- **To control the movement of the car using DC motors driven by an L298N motor driver.**
  This includes enabling forward, reverse, stop, and turning based on sensor and camera input.

- **To develop a web interface using VPN and Flask for real-time video streaming.**
  This allows users to monitor the car's camera feed remotely from any internet-enabled device.

- **To provide a cost-effective and educational model for learning autonomous vehicle systems.**
  The project promotes hands-on experience in embedded systems, computer vision, and IoT.

1.3 Problem Statement

Develop a low-cost self-driving car prototype using Raspberry Pi 4 that can autonomously detect lanes, identify and avoid obstacles, and navigate safely. The system should use a webcam for lane tracking, ultrasonic sensors for distance measurement, object detection for obstacle recognition, and DC motors for movement. It must also support real-time monitoring via a VPN-enabled web interface.

1.4 Application in Societal Context

This self-driving car project using Raspberry Pi integrates lane and object detection with obstacle avoidance to demonstrate the social relevance of autonomous systems. It supports:

- **Road Safety:** Reduces accidents through real-time detection and smart navigation.
- **Mobility for the Differently Abled:** Offers independence via assistive automation.
- **Smart Traffic Management:** Improves urban mobility and reduces congestion.
- **Energy Efficiency:** Promotes eco-friendly transport through optimized driving.
- **Education and Research:** Serves as a practical tool for learning AI and embedded systems.
- **Remote Monitoring:** Enables live video streaming for surveillance and control.

This project contributes to sustainable, accessible, and intelligent transportation solutions.

1.5 Literature Survey

Literature Survey

The literature survey across the selected papers reflects a growing focus on vision-based technologies, particularly **deep learning** and **computer vision**, for autonomous vehicle applications such as **lane detection**, **object detection**, and **obstacle avoidance**.

The study *"Vision-Based Robust Lane Detection and Tracking System"* employs **YOLOv5**, a powerful real-time object detection model, enhancing lane detection even under poor visibility and weak road marking conditions. It uses post-processing techniques and temporal information to ensure robust lane tracking.

Complementing this, the work titled *"A Self-Driving Car Platform Using Raspberry Pi and Arduino"* integrates lane, traffic sign, and obstacle detection modules through algorithms like **Canny Edge Detection** and **HAAR cascades**. It uses Raspberry Pi for real-time control and a **Convolutional Neural Network (CNN)** for traffic sign classification, achieving high accuracy in practical environments.

Similarly, the paper *"Object Detection and Lane Changing for Self-Driving Car Using CNN"* emphasizes the use of **CNN** and **OpenCV** for both lane and object detection, offering a reliable deep learning approach to support dynamic navigation decisions such as safe lane changes in response to detected obstacles.

The study *"Self-Automated Car with Obstacles Detection"* builds on these ideas by combining **Raspberry Pi**, **Arduino**, and **HAAR classifiers** for real-time autonomous control. It also focuses on **collision avoidance** and incorporates path control logic including **U-turns** and **direction shifts**.

Lastly, *"Road Lane Line Detection for Autonomous Cars"* employs traditional image processing tools like **OpenCV**, **Sobel edge detection**, and **perspective transforms** to deliver a straightforward yet effective lane detection pipeline. This approach is particularly suitable for scenarios where **low-computation** solutions are preferred.

Collectively, these works underline the trend towards leveraging **low-cost embedded systems** with advanced computer vision and deep learning algorithms to achieve reliable and cost-effective autonomous driving systems adaptable to real-world conditions.

Recent studies further highlight the growing importance of **real-time, adaptable systems** for autonomous driving, capable of handling diverse conditions such as urban traffic and poor weather. By combining **deep learning** with **low-cost platforms** like Raspberry Pi and Arduino, these systems achieve scalable and affordable deployment.

Techniques such as **bird's-eye view transformation**, **Kalman filtering**, and **data augmentation** enhance lane detection accuracy. Approaches using **monocular vision** and **behavior cloning** enable intelligent steering decisions, while **multi-functional modules** manage lane, object, and traffic sign detection. This integrated approach supports **safer and more reliable autonomous navigation** in real-world environments.

Chapter 2

Project Planing

Project planning involves outlining the essential stages necessary to complete the mini project effectively within the academic timeline. This includes defining specific tasks such as conducting a literature review, designing schematics, implementing and testing simulations, analyzing results, and integrating components. A well-structured plan ensures efficient time management, resource allocation, and smooth coordination among team members throughout the development process.

2.1 Gantt Chart

A Gantt chart is utilized to plan and schedule project tasks effectively. It helps in visualizing the timeline and dependencies between tasks. Tracking progress ensures timely completion of each phase. It also aids in balancing the workload among team members. Overall, it supports systematic and organized project execution.

PHASES	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10
LITERATURE SURVEY										
DESIGN										
IMPLEMENTATION										
TEST AND DEBUG										
FIXING THE BUGS										

Figure 2.1: Gantt chart

2.2 Work Breakdown Structure

The Work Breakdown Structure (WBS) is a hierarchical decomposition of the total project work into smaller, well-defined and manageable components. It plays a key role in organizing the project workflow and assigning responsibilities across various phases. In our self-driving car project, the WBS includes stages such as hardware setup, circuit design, algorithm development, simulation, performance analysis, and documentation. This structured approach ensures better clarity of tasks, facilitates effective coordination among team members, and helps in monitoring progress efficiently. By dividing the work into logical segments, WBS supports timely execution and enhances the overall quality and management of the project.

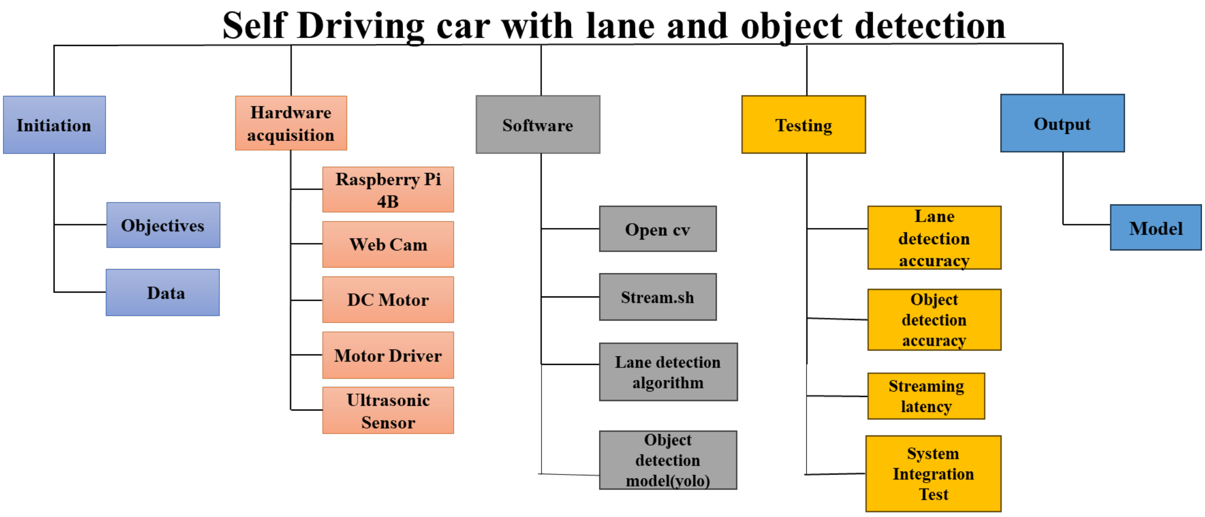


Figure 2.2: WBS

Chapter 3

Design Specification

3.1 Input Specification

- **Supply Voltage (VDD):** 5V / 3.3V Powers the Raspberry Pi, sensors, and motor driver with regulated voltages.
- **Ground (GND):** 0V Common electrical reference for all components.
- **Camera Input (Webcam):** USB Interface Provides live video feed for lane and object detection.
- **Ultrasonic Sensor Inputs:** Digital Echo/Trigger Pins Measures distances (2–400 cm) for obstacle detection using GPIO signals.
- **Web Interface Input:** HTTP-Based Browser Controls Allows users to control the vehicle and view the live feed via a Flask web dashboard.
- **Motor Control Outputs:** PWM Signals via GPIO Controls motor speed and direction using PWM signals from the Raspberry Pi.
- **Object Detection Output:** YOLOv5 Bounding Boxes Provides labeled boxes around detected objects in real time.
- **Logic Levels: Logic '0'** = 0V, **Logic '1'** = 3.3V Standard voltage levels for Raspberry Pi GPIO communication.

3.2 Output Specification

- **Motor Output Signals:** Directional PWM signals Controls motor speed and direction via the **L298N driver** for forward, reverse, left, and right movement.
- **Video Stream Output:** Live MJPEG/H.264 feed Streams real-time video from the **camera** to a Flask-based web interface.
- **Object Detection Output:** YOLOv5 bounding boxes Detects and labels objects (e.g., person, car) in the video stream for navigation decisions.
- **Lane Detection Output:** Lane overlay using OpenCV Detects and overlays **lane lines** on the live video to guide road alignment.

- **System Response:** Movement and stop commands Generates navigation decisions (e.g., **forward, stop, turn**) based on sensor and detection data.
- **Output Logic Levels: Logic '0'** = 0V, **Logic '1'** = 3.3V Follows Raspberry Pi GPIO standards for digital output signals.

3.3 Technology and Tools Used

The project is built around the **Raspberry Pi 4 (4GB)**, which serves as the central processing unit, handling all computational tasks including image processing, sensor interfacing, and decision-making. A **USB webcam** is used to capture real-time video that serves as input for both lane detection and object recognition. For obstacle detection and distance measurement, the system uses three **HC-SR04 ultrasonic sensors** strategically placed at the front and sides of the vehicle. Movement is controlled using **DC motors** driven by an **L298N motor driver module**, which receives PWM signals from the Raspberry Pi's GPIO pins to enable forward, reverse, and directional turns.

OpenCV, an open-source computer vision library, is utilized for processing video frames and detecting lane markings in real time. For object detection, the project uses **YOLOv5**, a state-of-the-art deep learning model that identifies and labels objects such as pedestrians, vehicles, and obstacles. The live video stream and basic user interface are managed through the **Flask web framework**, which allows remote monitoring and control via a browser. A **VPN (Virtual Private Network)** is employed to securely access the web interface from remote locations, enabling real-time monitoring and control over the internet. The entire software stack is developed using **Python 3**, which provides robust libraries and community support for interfacing hardware, implementing AI models, and building web services.

Chapter 4

Methodology

4.1 Methodology

1. **Define Project Scope and Objectives:** The main objective is to design an autonomous self-driving car capable of real-time lane detection and object avoidance using a Raspberry Pi 4. Key goals include stable lane following, obstacle detection, and live remote monitoring through a web interface.

2. **Component Selection and Integration:** Hardware components such as Raspberry Pi 4, USB webcam, DC motors with L298N motor driver, and HC-SR04 ultrasonic sensors are selected based on performance and compatibility. These are integrated on a chassis to form the robotic vehicle.

3. **Video Capture and Streaming:** A USB webcam is connected to the Raspberry Pi for capturing the live road view. The captured frames are streamed using Flask to a web interface and also processed for lane and object detection.

4. **Lane Detection using OpenCV:** OpenCV is used to process video frames to detect lane lines using edge detection, color filtering, and region of interest masking. The detected lanes are overlaid on the video feed to help guide the vehicle's path.

5. **Object Detection using YOLOv5:** YOLOv5, a pre-trained deep learning model, is implemented on the Raspberry Pi to detect and classify objects like pedestrians, vehicles, and obstacles. Detected objects are marked with labeled bounding boxes on the video stream.

6. **Obstacle Avoidance using Ultrasonic Sensors:** Three ultrasonic sensors are placed at the front and sides to measure distances from nearby objects. These readings help in collision avoidance decisions during movement.

7. **Control Algorithm for Navigation:** A Python-based control algorithm processes inputs from lane detection, object detection, and ultrasonic sensors to determine appropriate actions—such as move forward, stop, or turn. Corresponding PWM signals are sent to the motor driver.

8. **Motor Driver Control (L298N):** The Raspberry Pi sends GPIO-based PWM signals to the L298N driver module, which controls the direction and speed of two DC motors to facilitate motion based on control decisions.

9. **Flask Web Interface and VPN Access:** A Flask-based web interface allows users to monitor the live video stream and control the car remotely via HTTP requests. VPN ensures secure access to this interface from remote locations.

10. **Testing and Optimization:** The system is tested in real-world conditions to evaluate lane tracking accuracy, object detection reliability, obstacle avoidance response, and streaming performance. Based on observations, tuning and improvements are applied for better performance and robustness.

4.2 Functional Block diagram

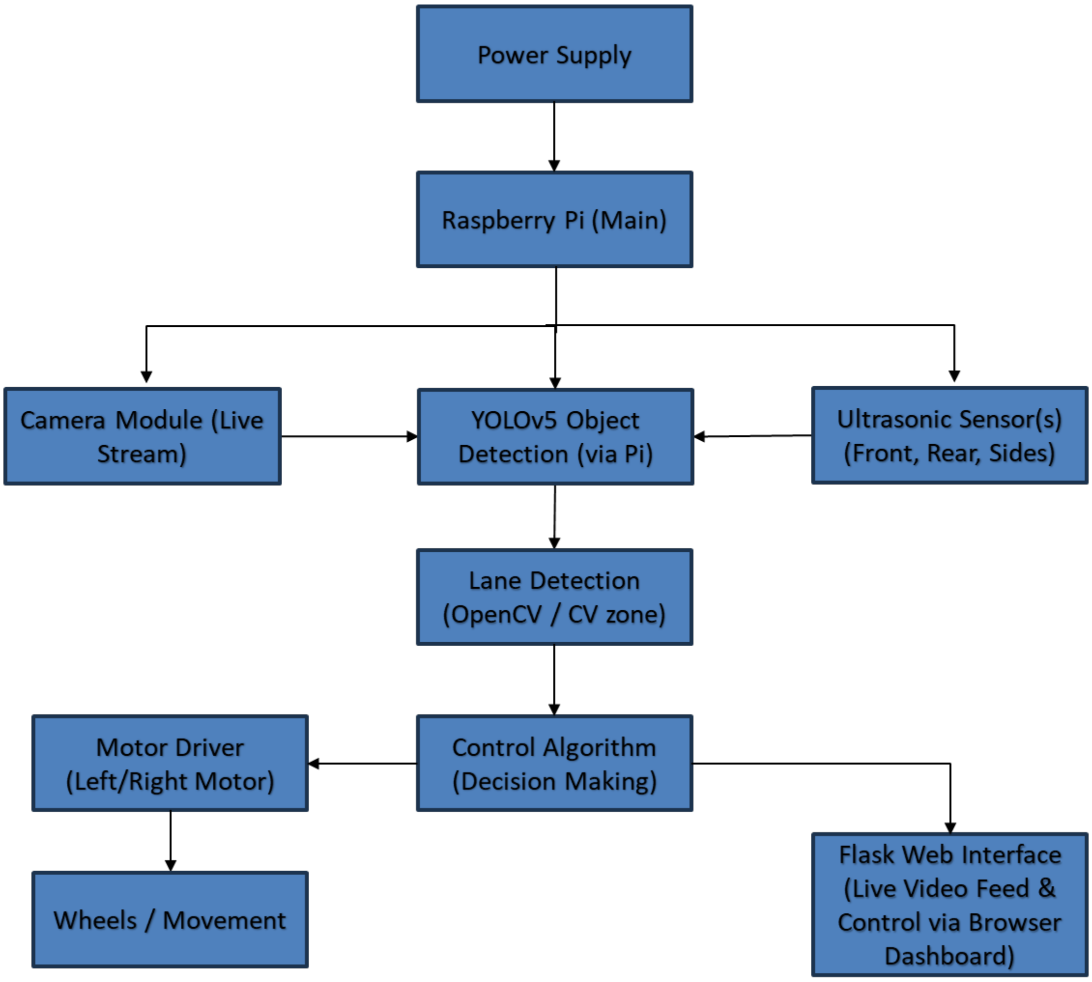
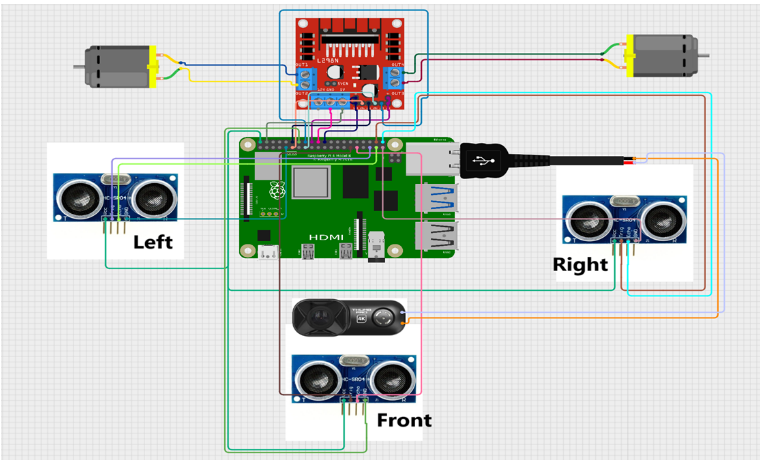


Figure 4.1: Block Diagram

Chapter 5

Implementation

5.1 Simulation Image Explanation



L298N Connections:
IN1, IN2 = 17, 18
IN3, IN4 = 22, 23
ENA, ENB = 25, 24

Ultrasonic Sensor Connections:
TRIG_F, ECHO_F = 5, 6
TRIG_L, ECHO_L = 19, 26
TRIG_R, ECHO_R = 20, 21

Figure 5.1: Simulation layout of the self-driving car.

In terms of working, the Raspberry Pi acts as the central controller, processing video frames from the webcam using OpenCV for lane detection and YOLOv5 for object detection. Simultaneously, the ultrasonic sensor constantly monitors for nearby obstacles and sends distance data to the Raspberry Pi. Based on lane position, detected objects, and obstacle proximity, the Python-based decision-making algorithm determines the appropriate motion commands. These commands are sent via GPIO to the L298N motor driver, which controls the movement of the DC motors (forward, reverse, left, right). The vehicle also supports live video streaming through a Flask-based web interface, accessible remotely via a secured VPN connection. This setup enables the car to operate autonomously in real-time with continuous environmental awareness.

Chapter 6

Result And Discussion

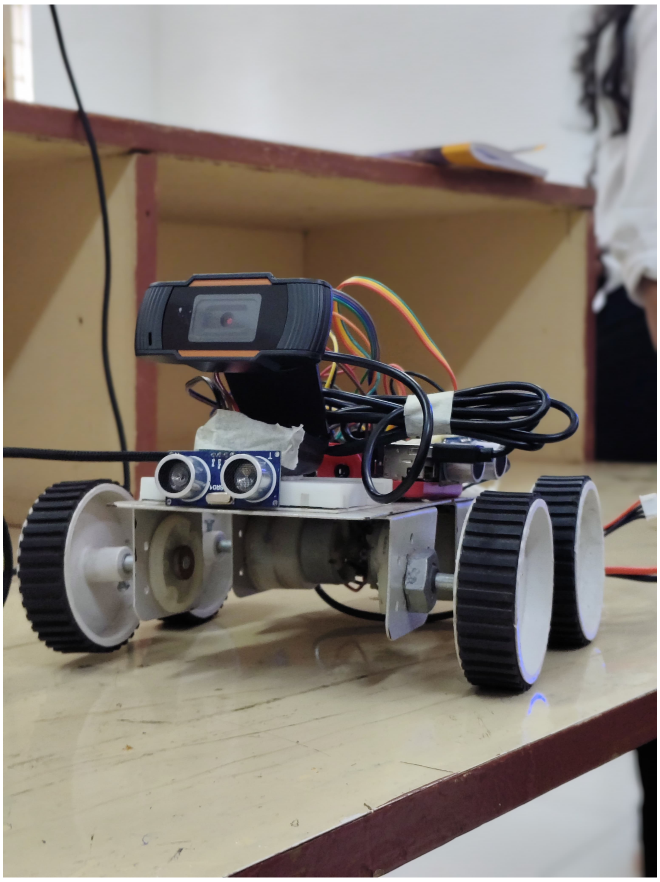


Figure 6.1: Model of the self-driving car.

Chapter 7

Conclusion

7.1 Conclusion

This project successfully demonstrates the development of a low-cost autonomous self-driving car system using Raspberry Pi 4. By integrating computer vision techniques such as lane detection with a webcam and object detection using YOLOv5, the vehicle can navigate roads safely. The use of three ultrasonic sensors provides accurate distance measurements to detect and avoid obstacles, enhancing the system's reliability. The L298N motor driver effectively controls the DC motors to enable smooth and precise vehicle movements. Additionally, live video streaming through a secure VPN web interface allows remote monitoring, increasing the system's usability in real-time environments. Overall, the project showcases the potential of combining affordable hardware and advanced image processing algorithms for autonomous navigation.

7.2 Future Scope

The self-driving car system can be further enhanced by incorporating additional sensors such as LiDAR or infrared sensors to improve obstacle detection accuracy, especially in low-light or complex environments. Integration of GPS modules can enable route planning and navigation over larger areas. Machine learning models could be expanded to include traffic sign recognition and pedestrian detection for more comprehensive autonomous driving capabilities. Improving the system's computational efficiency and deploying edge AI accelerators could enable faster decision-making for real-time applications. Lastly, expanding the remote interface to include control commands and telemetry feedback can make the vehicle suitable for diverse use cases such as delivery, surveillance, or educational platforms.