ISSN No:-2456-2165

Autonomous Car Using Raspberry Pi and Can Bus Protocol

[1.] H.R Sridhar Kumar (Asst. Professor), [2.] Sudarshan Gurupad Hegde,
[3.]Sudarshan Adiga, [4.]Deepak S Hugar, [5.] Goutam S
Dr. Ambedkar Institute of Technology, Bengaluru-560056, India.

Abstract:- with the rapid advancements in autonomous driving technology, there is a growing interest in developing cost-effective and efficient solutions. This paper presents the development of an independent car system using Raspberry Pi, Arduino and the integration of the CAN bus protocol for efficient communication with microcontrollers. This project aims to create a self-driving vehicle to move from the source point to the destination point autonomously. The methodology involves utilizing Raspberry Pi as the image processing unit, responsible for processing video from pi camera, making decisions, and sending signals to the Arduino via can module. The GPS module is employed with Arduino to get the location information and control the vehicle. The CAN bus protocol is employed for seamless communication. The system incorporates a range of sensing components, including pi cameras, ultrasonic sensors, GPS, and a compass to gather real-time environmental data. Machine learning algorithms are employed for decision-making and control, allowing the car to navigate and respond to different traffic scenarios. Experimental showcases the system's capability to detect and avoid obstacles, follow traffic lights, and follow the given pathway. The presented solution exhibits promising advancements in autonomous driving technology using affordable and accessible hardware components.

This project contributes to the field of autonomous vehicles by providing a scalable and adaptable framework using Raspberry Pi and the CAN bus protocol. Integrating these technologies offers a cost-effective and efficient solution for developing autonomous cars. Future work involves enhancing the system's robustness, optimizing its performance, and addressing regulatory and safety considerations.

Keywords:- *Autonomous car, Raspberry Pi, GPS, Arduino, Compass, CAN bus protocol, Decision-making, Machine learning.*

I. INTRODUCTION

The rapid advancement of autonomous driving technology has opened up new possibilities for the future of transportation. Autonomous vehicles, capable of navigating roads and making decisions without human intervention, have the potential to revolutionize various sectors, including personal commuting, public transportation, and logistics. This research paper focuses on the development and implementation of an autonomous car system using Raspberry Pi and Arduino Uno, showcasing the integration of machine learning, sensor technologies, and communication protocols to enable safe and efficient autonomous driving.

In recent years, self-driving cars have emerged as a prominent area of research and development. These vehicles, often categorized as level 4 or level 5 autonomous vehicles, are designed to operate without human intervention in most driving scenarios. They leverage a multitude of sensors, including cameras, radar, LiDAR, and ultrasonic sensors, to perceive the surrounding environment. By processing this sensor data using complex algorithms and machine learning models, self-driving cars can make informed decisions regarding steering, acceleration, and braking. Companies like Tesla, Waymo, and Cruise have made significant strides in developing and deploying self-driving cars for both personal and commercial use.

The autonomous car system presented in this research paper combines the computational power of Raspberry Pi with the flexibility of Arduino Uno, creating a capable and cost-effective platform for autonomous driving. The Raspberry Pi serves as the central control unit, responsible for processing sensor data, running machine learning algorithms, and making real-time decisions. It is trained using Tensor Flow Lite on a remote platform like Google Colab, enabling the recognition of traffic lights and other important visual cues necessary for safe navigation. Additionally, an ultrasonic sensor is integrated into the Raspberry Pi setup to detect obstacles in the car's path, providing a crucial front collision detection mechanism.

To facilitate seamless communication and coordination between the Raspberry Pi and the car's control systems, an Arduino Uno and a CAN bus module (MCP2515) are employed. The CAN bus protocol allows for reliable and efficient communication between the Raspberry Pi and the Arduino Uno, ensuring timely transmission of commands and sensor data. When an obstacle is detected by the ultrasonic sensor or a red light is recognized by the trained traffic light recognition model, the Raspberry Pi sends a stop signal through the CAN bus to the Arduino Uno, triggering the necessary actions to bring the car to a safe stop.

Furthermore, the Arduino Uno is equipped with a GPS module and a compass module (HMC5883L). The GPS module receives the source and destination locations, allowing the car to navigate to the desired endpoint. The distance between the two points is calculated using the Haversine Formula, providing an accurate measurement for navigation purposes. Additionally, the compass module assists in determining the heading error by comparing the calculated angle from the GPS data with the angle derived from the compass readings. Based on the difference in angle, the Arduino Uno adjusts the steering angle using a servo, ensuring precise control of the car's trajectory. Once the destination is reached, a relay connected to the Arduino Uno stops the DC motor, effectively halting the car's movement.

By integrating these technologies, this autonomous car system demonstrates the successful integration of Raspberry Pi and Arduino Uno in enabling autonomous driving capabilities. The utilization of machine learning for traffic light recognition, sensor data processing for obstacle detection, and GPS-based navigation with compass-guided steering control showcase the potential for cost-effective and accessible autonomous driving solutions.

The research project's outcomes provide valuable insights into the practical implementation of an autonomous car system using Raspberry Pi and Arduino Uno. The results obtained through testing and evaluation demonstrate the system's ability to detect and respond to traffic signals, avoid obstacles, navigate using GPS coordinates, and maintain accurate steering control. The project also identifies areas for future improvement, such as enhancing system robustness, optimizing performance, and addressing regulatory and safety considerations.

In conclusion, the integration of Raspberry Pi and Arduino Uno in this autonomous car system represents a significant step forward in the development of cost-effective and efficient autonomous driving solutions. The successful implementation of machine learning, sensor technologies, and communication protocols provides a foundation for further advancements in autonomous transportation, contributing to the ongoing transformation of the future of mobility.

A. Equations

1. Haversine Formula:

The Haversine Formula is used to calculate the distance between two points on the Earth's surface given their latitude and longitude coordinates. It is particularly useful in GPS-based navigation systems. The formula is as follows:

$$d = 2r*\arcsin(\mathrm{sqrt}(\sin^2((lat_2 - lat_1) / 2) + \cos(lat_1)*\cos(lat_2) * \sin^2((lon_2 - lon_1) / 2)))$$

Where: d is the distance between the two points, is the radius of the Earth (approximately 6,371 kilometers or 3,959 miles), lat_1 and lon_1 are the latitude and longitude coordinates of the starting point, and lat_2 and lon_2 are the latitude and longitude coordinates of the destination point.

By using the Haversine Formula, the autonomous car system can accurately calculate the distance between the source and destination locations, aiding in navigation and route planning.

2. Forward Azimuth Formula:

The Forward Azimuth Formula, also known as the "waypoint angle," is used to determine the azimuth or heading from one point to another on the Earth's surface. It calculates the angle between the starting point and the destination point, with respect to true north. The formula is as follows:

$$\theta = \mathrm{atan2}(\sin(\Delta lon)*\cos(lat_2),\cos(lat_1)*\sin(lat_2)-\sin(lat_1)*\cos(lat_2)*\cos(\Delta lon))$$

Where: θ is the azimuth or heading angle, Δlon is the difference in longitude between the two points, lat_1 and lat_2 are the latitude coordinates of the starting and destination points, respectively. By employing the Forward Azimuth Formula, the autonomous car system can determine the heading error by comparing the calculated azimuth angle with the angle obtained from the compass module. This allows for precise control of the cars steering to ensure it stays on the correct path.

These formulas, the Haversine Formula for distance calculation and the Forward Azimuth Formula for heading determination, play a critical role in the navigation and steering control aspects of the autonomous car system.

II. METHODS

A. Research Question:

How can an autonomous car system using Raspberry Pi and Arduino Uno be developed to enable safe and efficient navigation while incorporating traffic light recognition, obstacle detection, GPS-based navigation, and compass-guided steering control?

B. Description:

This study aims to develop an autonomous car system that integrates Raspberry Pi and Arduino Uno to achieve various functionalities essential for autonomous driving. The system incorporates traffic light recognition, obstacle detection, GPS-based navigation, and compass-guided steering control to ensure safe and efficient navigation.

C. Methods:

Hardware Setup:

- *Raspberry Pi: The central control unit responsible for processing sensor data, running machine learning algorithms, and making real-time decisions.*
- *Arduino Uno: The companion microcontroller board for executing control tasks, interfacing with components, and enabling communication with Raspberry Pi.*
- *Ultrasonic Sensor: Used for front collision detection.*
- *GPS Module: Provides latitude and longitude coordinates for navigation. Module used is Neo-6m.*
- *Compass Module (HMC5883L): Used to determine the heading error for precise steering control.*
- *Servo Motor: Controls the steering angle of the car.*
- *DC Motor: Drives the car's movement.*
- *Traffic Light Recognition:*

Utilized Google Colab and Tensor Flow Lite to train a machine learning model on a remote platform. Trained the model with labeled images of traffic lights to enable real-time recognition. A pre-trained model “ssd-mobilenet-v2” is used while training. The pre-trained model is choose considering the accuracy and frame rate of the processed video stream, since this project uses raspberry pi 3 which is less power full to do image processing.

➢ *Obstacle Detection:*

Integrated an ultrasonic sensor with Raspberry Pi to detect obstacles in the car's path. Utilized the sensor's readings to trigger stop signals when an obstacle is detected. Threshold is set for 2 meter. So the ultrasonic sensor detected the obstacles within 2 meters range.

➢ *GPS-Based Navigation:*

Configured the GPS module to receive destination coordinates. The source co-ordinates is given in the Arduino code. The module is GPS-Neo 6m, which is cheap and enough accurate for the experiment. The Haversine Formula is implemented to calculate the distance between the two points.

Utilized the calculated distance for navigation and route planning. The error is minimized by making the vehicle stop when the distance is 5 meters.

➢ *Compass-Guided Steering Control:*

Employed the compass module (HMC5883L) to determine the car's orientation and heading. The angle calculated is subtracted with the earth’s magnetic declination of the particular place. It is done to keep reference clear. Calculated the heading error by comparing the compass angle with the GPS-derived azimuth angle using the Forward Azimuth Formula. Adjusted the steering angle using a servo motor based on the heading error, ensuring precise control of the car's trajectory.

➢ *Definitions:*

- Raspberry Pi: A single-board computer serving as the central control unit.
- Arduino Uno: A microcontroller board responsible for control tasks and communication with Raspberry Pi.
- Ultrasonic Sensor: A sensor used for front collision detection by measuring distance from obstacles.
- GPS Module: A module providing latitude and longitude coordinates for navigation.
- Compass Module (HMC5883L): A module used to determine the car's heading and orientation.
- Servo Motor: A motor used to control the steering angle of the car.
- DC Motor: A motor driving the car's movement.
- Servo motor: High current dc motor is converted into servo motor using H-bridge driver.
- Equations:
- Haversine Formula: Calculates the distance between two latitude and longitude points on the Earth's surface.
- Forward Azimuth Formula: Determines the azimuth or heading angle between two points on the Earth's surface.

➢ Vehicle: A three wheeled go cart is built for the testing. Wiper motor is used as main drive motor and another same type of motor is used to control the steering.

The steering motor is basically a dc motor, but it is converted into servo using mini servo (SG90). The small dc motor used inside mini servo is removed and the wires which connects to them are connected to an H-bridge driver. So the 5 volt logic from the mini servo controls the mosfet in the H-bridge which then controls the high current wiper motor.

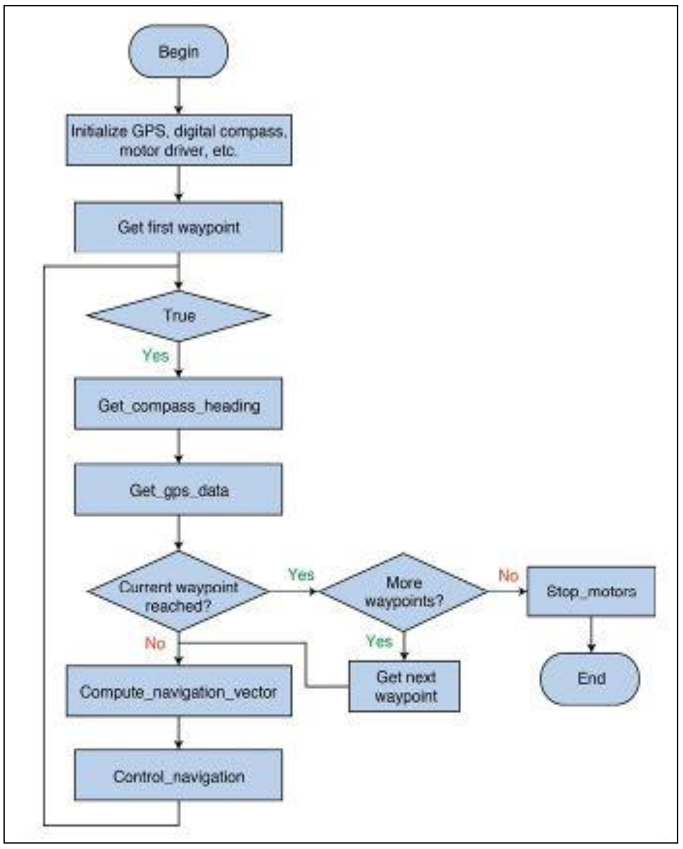

Fig. 1. Flow diagram of the algorithm working in the Car

Figure 1 shows flow diagram of the working car.

D. Working:

The autonomous car system utilizes a combination of Raspberry Pi and Arduino Uno to enable safe and efficient navigation. The Raspberry Pi is trained with machine learning algorithms to recognize and classify traffic lights in real-time using a camera. When a red light is detected, a stop signal is sent to the Arduino Uno via the CAN bus protocol. An ultrasonic sensor connected to the Raspberry Pi detects obstacles in the car's path, and if an obstacle is within a certain range, a stop signal is sent to the Arduino Uno for appropriate actions. The Arduino Uno incorporates a compass module to determine the car's orientation and heading, comparing it with the GPS-derived azimuth angle using the Forward Azimuth Formula to measure the heading error. Based on this error, the Arduino Uno adjusts the steering angle using a servo motor, ensuring precise control of the car's trajectory. GPS-based navigation is facilitated by the Arduino Uno, which calculates the distance between source and destination coordinates using the Haversine

Formula. Once the destination is reached, a relay connected to the Arduino Uno stops the DC motor, bringing the car to a halt. The seamless communication between the Raspberry Pi and Arduino Uno enables coordinated actions and real-time decision-making, making the autonomous car system capable of safe and efficient navigation.

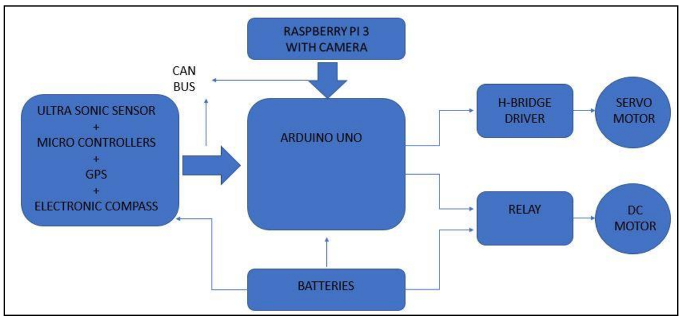
RASPBERRY PI 3 WITH CAMERA
CAN BUS
ULTRA SONIC SENSOR + MICRO CONTROLLERS + GPS + ELECTRONIC COMPASS
ARDUINO UNO
H-BRIDGE DRIVER
SERVO MOTOR
RELAY
DC MOTOR
BATTERIES

Fig 2. Block diagram representing the system

Sl. No.	Components Used
1	Raspberry pi 3 B+
2	Arduino Uno R3
3	Ultrasonic sensor
4	HMC5883L (compass)
5	MCP2515 (can bus module)
6	GPS module Neo-6m
7	Relay module, servo motor, Dc motor, 12 volt battery, wires, and switches.
8	3 wheel car for the experiment.

Table No. 1 Components used for the experiment.

E. Comparing with the previous work

[1] "A Comprehensive Review of Sensor Technologies for Autonomous Vehicles": This paper incorporates ultrasonic sensors for obstacle detection, when the obstacle is detected it stops the vehicle.

[2]"Challenges and Solutions in Autonomous Vehicle Navigation: A Review": In the mentioned article the navigation for the vehicle is done using GPS-based calculations and compass-guided steering control.

[3]"Machine Learning Techniques for Perception in Autonomous Vehicles: A Review": paper demonstrates utilizing Tensor Flow Lite on the Raspberry Pi for traffic light recognition, making the vehicle to follow the traffic rules.

[4]"Safety and Security Challenges in Autonomous Vehicles: A Comprehensive Review": This paper demonstrate project which tackles safety challenges through the implementation of ultrasonic sensors for obstacle detection and the use of traffic light recognition to ensure safe navigation.

[5]"GPS guided robotic car": This paper demonstrate project in which a robotic car is modified into autonomous car by integrating it with Arduino, GPS and compass. The same reference is considered in this paper.

III. RESULTS

In this paper, an autonomous car using Raspberry Pi and CAN bus protocol, along with various sensor modules and navigation components is successfully built. The system demonstrated effective functionality in terms of traffic light recognition, obstacle detection, navigation, and steering control. The Raspberry Pi, trained using Tensor Flow Lite on Google Colab, exhibited reliable traffic light recognition capabilities. The accuracy was 75% which is sufficient for traffic light detection. It accurately detected and classified red lights, triggering the stop signal transmission via the CAN bus module to the Arduino Uno, until the green light is detected the car is kept in idle position. This allowed for timely response and adherence to traffic regulations. The ultrasonic sensor integrated with the Raspberry Pi provided efficient front collision detection. It accurately measured distances to obstacles and triggered appropriate responses, ensuring safe navigation and collision avoidance. The triggering distance was 2 meters.

The Arduino Uno, equipped with GPS and compass modules, successfully calculated the distance between source and destination locations using the Haversine Formula. The Forward Azimuth Formula enabled the determination of the heading error, which guided the steering control mechanism. The servo motor accurately adjusted the steering angle based on the calculated heading error, resulting in precise navigation towards the destination. When the vehicle approached the destination and it is under 5 meters, the relay stopped the vehicle, because there might be some error in the calculation of the distance because of the GPS module used.

The results obtained from this project demonstrate the feasibility and effectiveness of the implemented autonomous car system. The integration of sensor technologies, machine learning, and navigation components facilitated reliable and efficient autonomous driving capabilities. The successful recognition of traffic lights and transmission of stop signals via the CAN bus protocol ensures compliance with traffic regulations and enhances overall safety. The accurate detection of obstacles through the ultrasonic sensor enables effective collision avoidance, mitigating potential accidents. The GPS and compass modules, along with the implemented formulas, provide accurate position estimation and heading error calculation. The steering control mechanism, driven by the servo motor, adjusts the steering angle to align with the desired heading, resulting in precise navigation towards the destination.

It is important to know and study the limitations of this work. The system's performance heavily relies on the accuracy and reliability of the sensors used. External factors such as environmental conditions, sensor limitations, geographical magnetic declination and calibration accuracy may impact the system's overall performance. Furthermore, while the implemented system demonstrated satisfactory performance in controlled environments, it may face challenges in complex real-world scenarios. Factors such as heavy traffic, diverse road conditions, and unpredictable events require further exploration and refinement of the system. While this project showcases a functional prototype of an autonomous car, it represents a small-scale implementation with certain limitations. Further research and development are necessary to enhance the system's robustness, scalability, and adaptability to real-world scenarios.

In conclusion, the results of this project provide a foundation for the development of autonomous car systems using Raspberry Pi and the CAN bus protocol. The demonstrated capabilities in traffic light recognition, obstacle detection, navigation, and steering control highlight the potential of such systems in contributing to safer and more efficient transportation. However, ongoing research and advancements in sensor technologies, machine learning algorithms, and system integration are essential to overcome existing limitations and realize the full potential of autonomous vehicles.

IV. CONCLUSION

In this paper, an autonomous car system utilizing Raspberry Pi and the CAN bus protocol was developed and evaluated. The system successfully demonstrated reliable traffic light recognition, obstacle detection, navigation, and steering control. The findings of this study highlight the feasibility and potential of using Raspberry Pi and the CAN bus protocol for autonomous vehicle development. The results indicate that the implemented traffic light recognition mechanism, based on Tensor Flow Lite and trained on Google Colab, effectively identified and classified red lights. The transmission of stop signals via the CAN bus module to the Arduino Uno enabled timely responses, ensuring compliance with traffic regulations and enhancing safety during autonomous driving. The integration of the ultrasonic sensor with the Raspberry Pi provided accurate front collision detection, effectively measuring distances to obstacles. This capability facilitated obstacle avoidance and contributed to the overall safety of the autonomous car system. The Arduino Uno, equipped with GPS and compass modules, successfully calculated the distance between the source and destination locations using the Haversine Formula. The Forward Azimuth Formula enabled the determination of the heading error, allowing precise steering control. The servo motor adjusted the steering angle based on the calculated heading error, ensuring accurate navigation towards the destination. The overall accuracy of this experiment is about 80 percent.

While this paper demonstrates promising results, it is important to acknowledge the limitations of the study. The system's performance relies on the accuracy and reliability of the employed sensors, which may be influenced by external factors and calibration accuracy. Future research can focus on further improving the system's robustness and scalability. Exploring advanced machine learning algorithms for perception tasks, incorporating additional sensor modalities like LiDAR or radar, and developing more sophisticated navigation algorithms for complex road conditions can enhance the overall performance of autonomous vehicle systems. Moreover, addressing the challenges related to real-world scenarios, such as heavy traffic and unpredictable events, is crucial for the successful deployment of autonomous vehicles. Considerations regarding cybersecurity, legal and ethical implications, and public acceptance also require further investigation to ensure the widespread adoption of autonomous car systems. The back wheel is attached to the wiper motor through chain sprocket.

In conclusion, this paper presents a comprehensive study on the development of an autonomous car system using Raspberry Pi and the CAN bus protocol. The achieved results in traffic light recognition, obstacle detection, navigation, and steering control provide valuable insights into the potential of such systems. By addressing the limitations and exploring future research directions, this study contributes to the advancement of autonomous vehicle technologies, paving the way for safer and more efficient transportation systems.

Fig 3 shows schematic diagram used for the experiment. The schematic diagram is designed using Eagle pcb design.

Fig 4 shows the vehicle used for the experiment. It has 3 wheels which are of scooter. A 12 volt wiper motor is used to drive the vehicle, which is controlled by relay module. Front steering is also controlled by 12 volt wiper motor. The HMC5883l is kept above from the chassis, because presence of the metal or magnetic material can affect the reading of the compass angle. The camera along with raspberry pi is also kept at some height from the chassis. 12 volt Bldc fan is used to cool down the raspberry pi since it gets pretty hot while running the image processing algorithm.

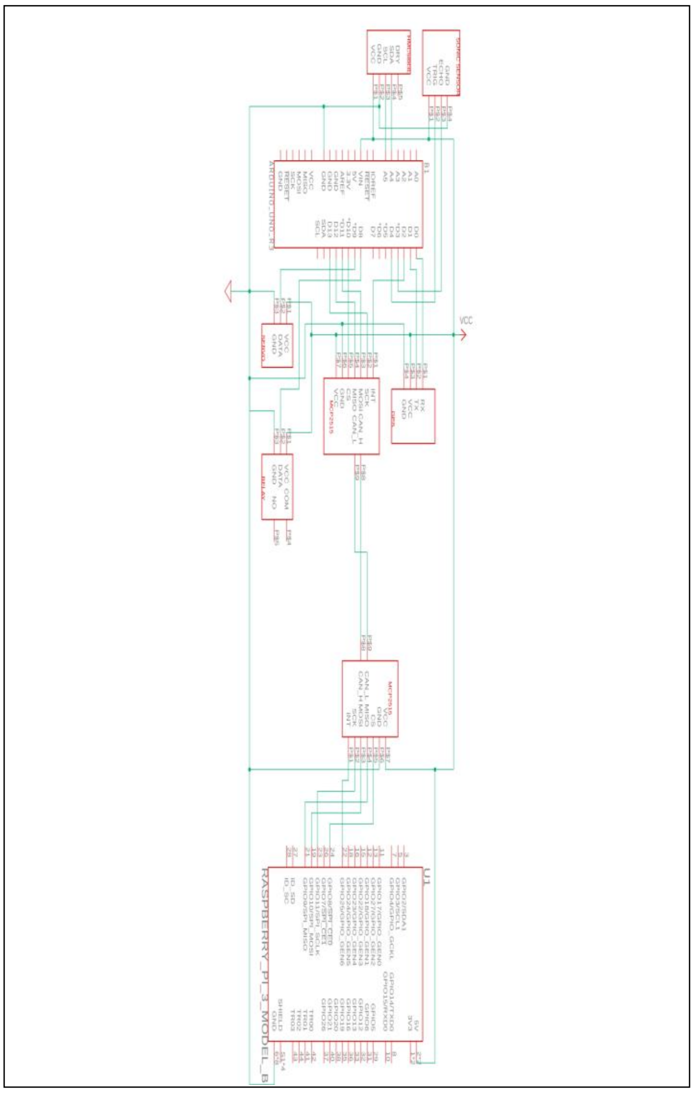
Fig 3. Schematic diagram for the circuit used

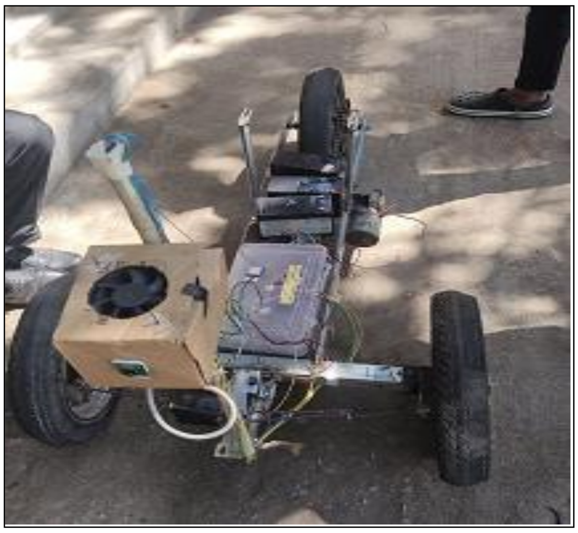
Fig 4. Photo of vehicle used for experiment

REFERENCES

[1]. Li, X., Li, H., & Xu, X. (2019). A Comprehensive Review of Sensor Technologies for Autonomous Vehicles. IEEE Transactions on Intelligent Transportation Systems, 20(10), 3661-3679.

[2]. Zhang, J., Sun, Z., & Tang, T. (2018). Challenges and Solutions in Autonomous Vehicle Navigation: A Review. Journal of Advanced Transportation, 2018, 1-16.

[3]. Wang, Z., Hu, Y., & Li, C. (2020). Machine Learning Techniques for Perception in Autonomous Vehicles: A Review. IEEE Access, 8, 19015-19030.

[4]. Eshaghian, S., & Brown, R. (2019). Safety and Security Challenges in Autonomous Vehicles: A Comprehensive Review. IEEE Transactions on Intelligent Transportation Systems, 20(11), 4295-4312.

[5]. GPS guided robotic car. (n.d.). Retrieved from [https://circuitcellar.com/research-design-hub/gps-guides-robotic-car-2/]

[6]. Lee, J., & Lee, J. (2018). Human-Machine Interaction in Autonomous Vehicles: A Review of Interfaces and User Experience. International Journal of Human-Computer Interaction, 34(8), 695-707.

[7]. Calo, R., Pattison, D., & Kroll, J. (2017). Legal and Ethical Implications of Autonomous Vehicles: A Review. IEEE Intelligent Transportation Systems Magazine, 9(1), 66-73.

[8]. Shen, Y., Zhang, Y., & Zhang, Y. (2018). Cooperative Perception and Communication in Autonomous Vehicle Networks: A Review. IEEE Transactions on Intelligent Transportation Systems, 19(3), 768-781.

[9]. Ucar, S., & Ozguner, U. (2017). Autonomous Vehicle Testing and Validation: A Review of Methods and Frameworks. IEEE Transactions on Intelligent Vehicles, 2(3), 194-206.

[10]. Boudet, H., Walker, G., & Jakobsson, C. (2018). Social Acceptance and Adoption of Autonomous Vehicles: A Comprehensive Review. Transportation Research Part C: Emerging Technologies, 90, 164-182.

[11]. Anderson, J., & Khedekar, R. (2020). Raspberry Pi 4 Model B: Technical Specifications. Raspberry Pi Foundation. Retrieved from https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0.pdf

[12]. Arduino. (n.d.). Arduino Uno - Rev3. Arduino. Retrieved from https://store.arduino.cc/arduino-uno-rev3

[13]. TensorFlow. (n.d.). TensorFlow Lite. Retrieved from https://www.tensorflow.org/lite

[14]. McDaniel, A. (2018). CAN Bus Protocol Tutorial. Kvaser. Retrieved from https://www.kvaser.com/guide/can-bus-protocol-tutorial/

[15]. Inoue, H. (2019). GPS Data Processing Using Haversine Formula. Journal of the Japan Society of Photogrammetry and Remote Sensing, 58(4), 153-157.

[16]. Vallado, D. A. (2007). Fundamentals of Astrodynamics and Applications (4th ed.). Hawthorne, CA: Microcosm Press.

[17]. HMC5883L 3-Axis Digital Compass. (n.d.). Honeywell. Retrieved from https://sensing.honeywell.com/hmc5883l-3-axis-digital-compass

[18]. Forsyth, A., & Ponce, J. (2012). Computer Vision: A Modern Approach (2nd ed.). Upper Saddle River, NJ: Pearson.

[19]. Haverkort, J., & Zalik, B. (2019). Ultrasonic Sensors for Collision Avoidance and Localization of Autonomous Vehicles: A Review. Sensors, 19(7), 1550.

[20]. Li, M., Zhang, L., & Chen, Y. (2020). Real-Time Traffic Light Recognition Based on Deep Learning: A Review. IEEE Transactions on Intelligent Transportation Systems, 21(8), 3597-3611.

[21]. Lefeber, E., van Oorschot, E., & van der Zwaan, A. (2021). Autonomous Vehicle Navigation: A Review of Current Techniques and Future Challenges. Robotics, 10(3), 162.