World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Design of a Mi-Autonomous Vehicle for Smart Mobility Solutions

Fega Eric Ojefia [1,*], Abraham Arigbanla Aanuoluwapo [2], Sunday Ikechukwu [2], Elijah Oluwadara Adetunji [3], Osahon Michael Enodiana [1] and Felix Tochukwu Ekwonu [4]

[1] *Department of Civil Engineering, Faculty of Engineering, University of Benin, Edo state, Nigeria.*
[2] *Department of Mechatronics Engineering, Faculty of Engineering, University of Benin, Edo state, Nigeria.*
[3] *Department of Civil Engineering, Faculty of Engineering, Federal University of Technology Akure, Ondo state, Nigeria.*
[4] *Department of Mechanical Engineering, Faculty of Engineering, University of Benin, Edo State, Nigeria.*

World Journal of Advanced Research and Reviews, 2025, 28(01), 434-448

Publication history: Received on 30 August 2025; revised on 04 October 2025; accepted on 06 October 2025

Article DOI: https://doi.org/10.30574/wjarr.2025.28.1.3443

Abstract

Road accidents account for significant fatalities and injuries, underscoring the critical importance of transportation safety as many victims are left with life-long disabilities. A significant proportion of these road accidents are attributed to human errors and negligence, especially those caused by drunk drivers. Autonomous Vehicles (AV) technology also know has self-driving cars has potential to significantly improve the efficiency and safety of the transportation and logistics industry thereby reducing road accidents and decongesting roads by improving traffic flow. This project proposes the design, development, and evaluation of an Autonomous Vehicle (AV) prototype aimed at demonstrating the feasibility of low-cost, compact automation for indoor and controlled outdoor environments. The vehicle uses an ultrasonic sensor for real-time obstacle detection, with a microcontroller (ESP32)-based control system to execute navigation and collision avoidance maneuvers. A modular architecture was implemented, incorporating sensor data processing, path planning, and control algorithms to ensure responsive and adaptive vehicle behavior. Both simulation and field tests were conducted to validate system performance, with results indicating reliable obstacle detection, effective trajectory tracking, and robust control response under various operating conditions. The successful realization of this mini autonomous vehicle project not only underscores the potential of accessible, cost-effective autonomous systems but also provides a solid foundation for future enhancements and applications in small-scale robotics.

Keywords: Autonomous Vehicles (Avs); Self-Driving Cars; Ultrasonic Sensors; Intelligent Transportation Systems; Robotics; Transportation Safety

1. Introduction

Road transportation remains the most prevalent medium of mobility for the movement of people and goods due to its widespread availability and flexibility. However, it is also associated with significant challenges, including traffic jams, accidents, and poor road infrastructure (Alex et al., 2023; Barberi et al., 2022; Pappalardo et al., 2022). Every year, approximately 1.19 million deaths and a staggering 20-50 million more people suffer non-fatal injuries with many incurring a disability as a result of road traffic crashes. Several of these road accidents are attributed to human errors and negligence including fatigue, distraction, and impaired driving, and especially by drunkenness (World Health Organization, 2023). These challenges underscore the urgent need for a safer and more efficient transportation systems.

One promising innovation that address these issues is the autonomous vehicle (AV) technology. Autonomous vehicles also known as self-driving cars are equipped with technologies that make them capable of sensing their environment, making decisions, and navigating without human intervention. This is a transformative technology with the potential to revolutionize transportation and improve road safety through substantial reduction in the number and severity of

* Corresponding author: Fega Eric Ojefia

accidents, reduced congestion, increased mobility for people with disabilities and the elderly (Zhang et al., 2019), reduced congestion, emissions, and efficient use of the infrastructure (Fagnant et al, 2015). AVs do not get distracted, fatigued, or impaired, which greatly reduces the likelihood of accidents compared to conventional vehicles. Modern AVs employ a range of technologies, including ultrasonic sensors, cameras, radar, and artificial intelligence algorithms, to detect obstacles, interpret road conditions, and make real-time driving decisions.

Despite AV technologies potential to revolutionize the transportation sector, it still faces issues relating to safety and sustainability (Kalra and Paddock, 2016; Greenwald and Kornhauser, 2019). Nevertheless, ongoing research emphasizes the importance of developing and testing scaled prototypes to refine navigation, sensing, and control strategies. Many existing studies focus on large-scale AVs intended for deployment in real-world conditions. For research and educational purposes, scaled prototypes and miniaturized low-cost AVs provide a practical platform to study, demonstrate, and test the fundamental principles of autonomy without the prohibitive costs of full-scale systems. Such miniaturized AVs serve as experimental testbeds for obstacle avoidance, navigation, and system integration, making them valuable in both academic and industrial contexts.

In this project, we present the design and development of a mini autonomous vehicle (AV) prototype that uses ultrasonic sensors for obstacle detection and avoidance. The project aims to demonstrate the feasibility of building a cost-effective AV system that can sense its environment, make simple navigational decisions, and operate with limited human input. The findings from this work highlight how scaled-down AVs can contribute to research, education, and future development in intelligent transportation systems.

2. Materials and methods

2.1. Hardware Implementation

Several essential parts that work together to enable sensing, control, navigation, and mobility are integrated into the hardware implementation of the small autonomous vehicle. The primary components of the hardware include:

- **Microcontroller:** Two microcontrollers were employed in this project: the ESP32 and the Arduino Uno R3. The ESP32 is a low-cost, low-power microcontroller with built-in Wi-Fi and Bluetooth capabilities, giving it internet connectivity capabilities, which is essential for this project. This connectivity enables the ESP32 to receive and process location data from both the mobile application and the GPS module. It is the main controller controlling the servo motor for ultrasonic scanning, and making navigation decisions such as obstacle avoidance and path correction. It then sends motor control instructions to the Arduino Uno, which serves as a secondary unit. The Uno, together with the L293D motor driver shield mounted on it, executes low-level motor actuation to drive the DC motors according to the ESP32's commands.

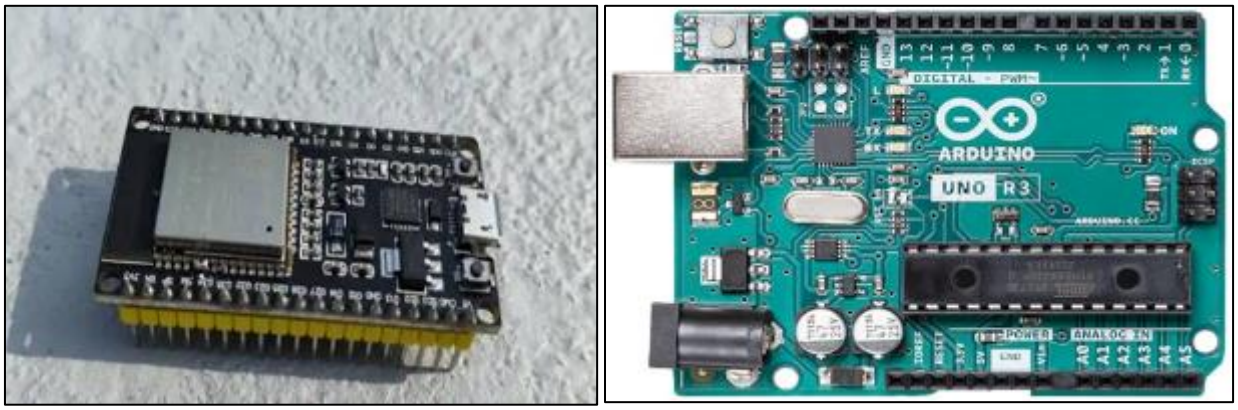

Figure 1 Image of an ESP32 (left) and Arduino uno R3 (right) microcontrollers

- **Motor Driver (L293D):** The motor driver is important for controlling and managing how the motor operates. For this project, L293D motor driver was used. It is a 16-pin Motor Driver IC which can be used to control 4 DC motors simultaneously in any direction. For this project, it was used for controlling the speed and direction of the 4 DC motors used for driving the wheels of the vehicle.

Figure 2 L293D Motor Driver (adapted from Az-Delivery, 2025)

- **Motors (DC Motors):** The vehicle employs DC motors to provide motion. These motors are lightweight, efficient, and well-suited for small-scale robotic applications. For this project, 4 DC motors were used to control the drive and to control the steering of the vehicle.

Figure 3 DC Motor (adapted from botland.store, 2025)

- **GPS Module:** The GPS module is used to select the desired destination for the vehicle, and also to track the location its location. The NEO-6M GPS module was used in this project; it is a GPS receiver compatible with most microcontroller boards. It can get data about location, speed, altitude, and time. It has an external EEPROM, a tiny backup battery, and an LED signal indicator that will blink when it receives a position fix (Dejan, 2022). The NEO-6M GPS Module was connected using the ESP32 UART2 pins (TX/RX), the pin connection can be seen in the circuit design section below. The NEO-6M GPS Module communicates with the ESP32 microcontroller using serial communication protocol

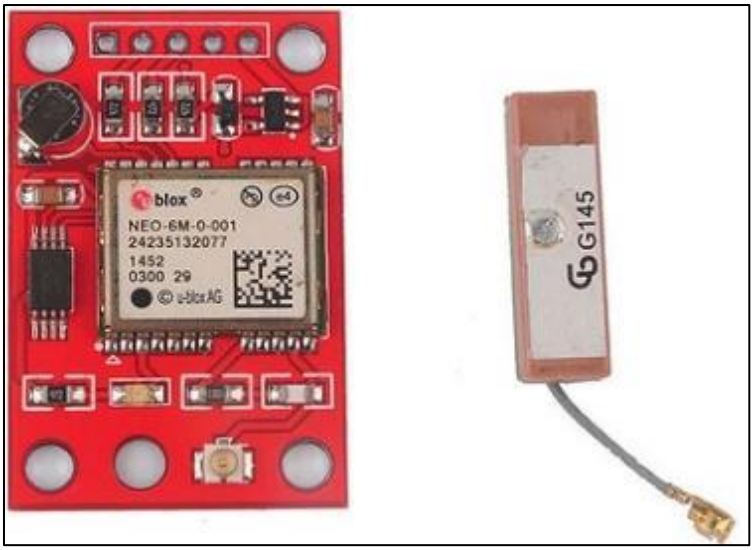

Figure 4 NEO-6M GPS module (adapted from Arduino Forum, 2025)

- **Ultrasonic Sensor:** Ultrasonic sensors emit short, high-frequency sound pulses at regular intervals, which move through the atmosphere at the speed of sound. The sensor uses the time difference between sending out the signal and receiving the echo to calculate the target's distance if the pulses strike an object and are reflected back to it as echo signals. With a range of up to 400cm (about an inch to 13 feet), for this project, the HC-SR04 ultrasonic sensor was used. Two ultrasonic transducers make up the sensor. Ultrasonic sound pulses are produced by a transmitter, and reflected waves are detected by a receiver. The sensor has four pins: VCC and GND connect to the L293D 5V and GND pins, while Trig and Echo connect to ESP32's GPIO pin. The ultrasonic wave is sent from the transmitter using the Trig pin, and the reflected signal is listened for using the Echo pin. The distance is determined by taking into account the travel time and the speed of the sound generated by the module, which emits an ultrasound at 40,000 Hz that travels through the air and bounces back to the module if it encounters any obstacles or objects. Considering the travel time and the speed of the sound we can calculate the distance. By knowing the distance between the vehicle and an object, the microcontroller will be able to process this input and subsequently send a command to the motor driver in order to avoid collision with any obstacle.

Figure 5 HC-SR04 Ultrasonic Sensor (adapted from Embe Tronicx, 2025)

- **Power Source:** The system is powered by a rechargeable battery, which guarantees portability and uninterrupted operation. To meet the voltage needs of each component, a regulated power distribution was put in place. For this project, three 3.7v lithium-ion rechargeable batteries were used. This provides the power needed by the vehicle to operate properly.

Figure 6 3.7v lithium-ion batteries (adapted from PKNERGY, 2025)

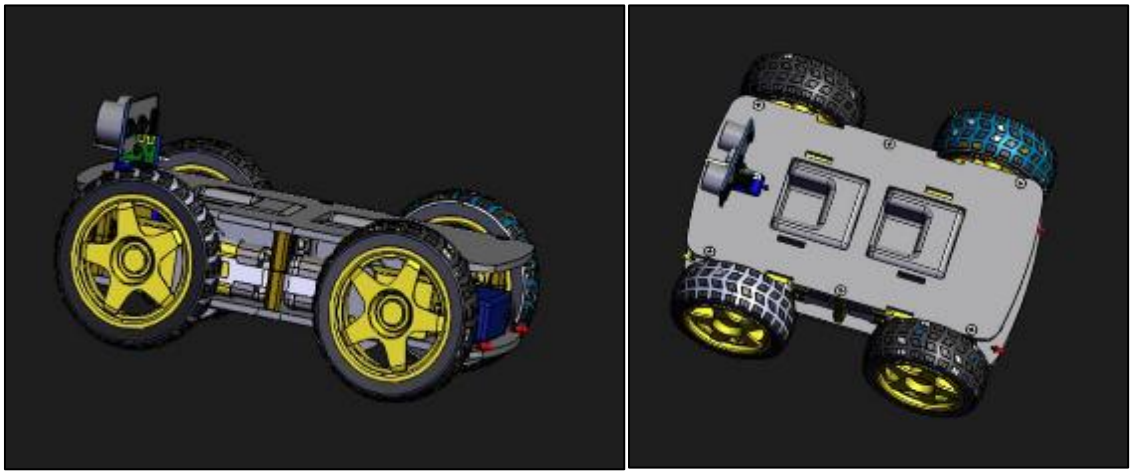

Figure 7 SolidWorks design of the 4-wheel chassis used for the project

- **Chassis**: This is the base frame of the car, which contains the wheels (4 wheels in this case), the other components/parts of the car are mounted on the chassis. A 4-wheel drive chassis was selected for improved stability and traction, with caster wheel for balance. Fig. 7 represents two images showing the design of a 4-wheel chassis we made using solid works software.
- **Server Motor:** The servo motor in this project was used to enable object detection by providing rotational movement for the ultrasonic sensor (HC-SR04). The sensor was mounted on the servo, allowing it to rotate within a specified angular range and scan the environment in multiple directions. This setup improved obstacle detection and navigation accuracy by expanding the sensor's field of view. The servo was controlled using pulse-width modulation (PWM) signals from the microcontroller to adjust its position precisely during scanning.

Figure 8 Sg90 Servo Motor (adapted from ElectronicsWings, 2025)

- **Wires and Connectors (Jumper Wires and Breadboard):** These provide the necessary electrical connections between components. A breadboard was used during prototyping for flexible circuit testing and integration.
- **Other Components:** Additional supporting components such as resistors, switches, and voltage regulators were included to ensure safe and reliable operation of the system.

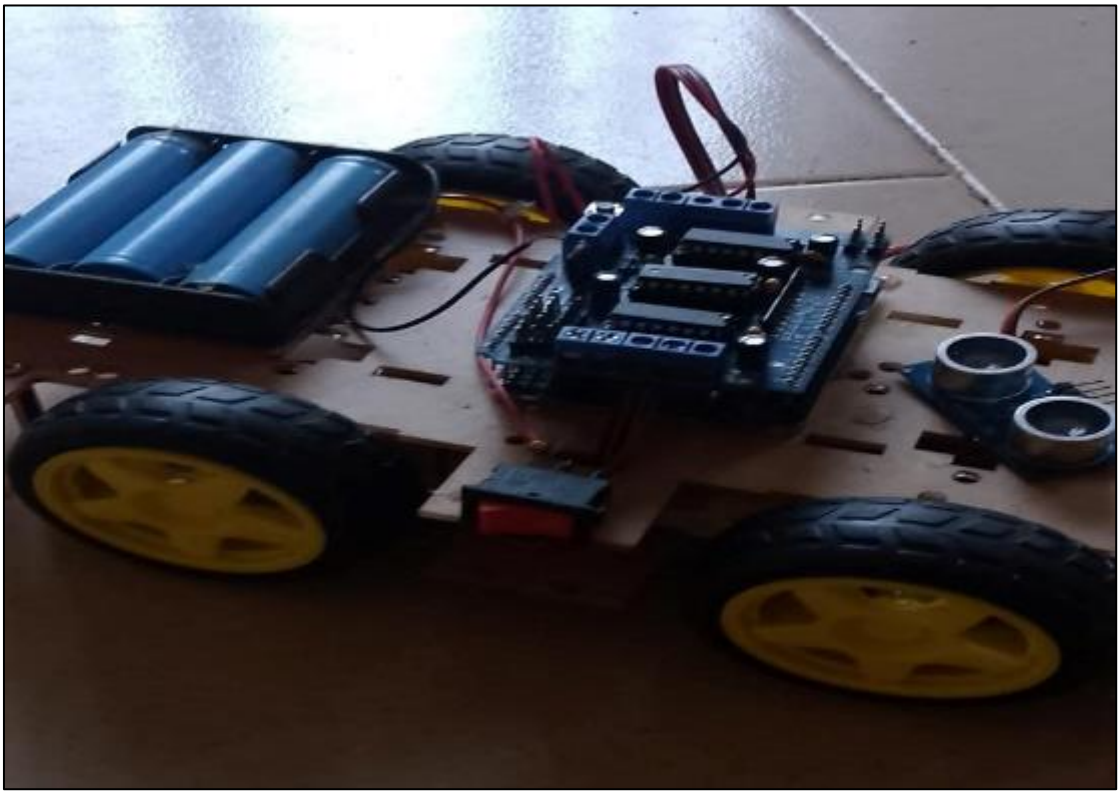

Figure 9 Image of our project (mini autonomous vehicle)

2.2. System Architecture

The overall system architecture of the mini autonomous vehicle is organized into four main functional modules

- **GPS Navigation:** The NEO-6M GPS module is used provide real-time location data in terms of latitude and longitude. This information is used by the ESP32 microcontroller to determine the vehicle's current position and guide it toward the destination.

- **Obstacle Avoidance:** An ultrasonic sensor (HC-SR04), mounted on a servo motor(Sg90), is employed to detect obstacles in the vehicle's path. By scanning the environment, the sensor enables the system to make real-time decisions for collision avoidance.
- **Motor Control (Arduino Uno + L293D Shield):** Because the L293D motor shield is designed to mount directly on the Arduino Uno, the Uno is used as the motor interface. It receives motion commands from the ESP32 via a serial connection and drives the four DC gear motors accordingly.
- **Microcontroller Unit:** The system uses both an ESP32 and an Arduino Uno R3. The ESP32 functions as the central controller, handling GPS navigation, obstacle detection, and decision-making. The Arduino Uno receives navigation commands from the ESP32 after they have been determined. The Uno, equipped with the L293D motor driver shield, acts as the motor control unit, translating the ESP32's commands into speed and direction signals that drive the four DC motors.

2.3. Block Diagram Description

The system's block diagram shows how each module interacts with the others to achieve autonomous navigation. The ESP32 receives real-time location data from the GPS module, while the ultrasonic sensor on a servo motor provides obstacle distance data. Serving as the main controller, the ESP32 interprets this data, makes navigation decisions, and then communicates movement commands to the Arduino Uno. The Uno, integrated with the L293D motor driver shield, executes these commands by controlling the speed and direction of the four DC motors that drive the vehicle. A rechargeable battery supplies power to all components, ensuring uninterrupted and independent operation.

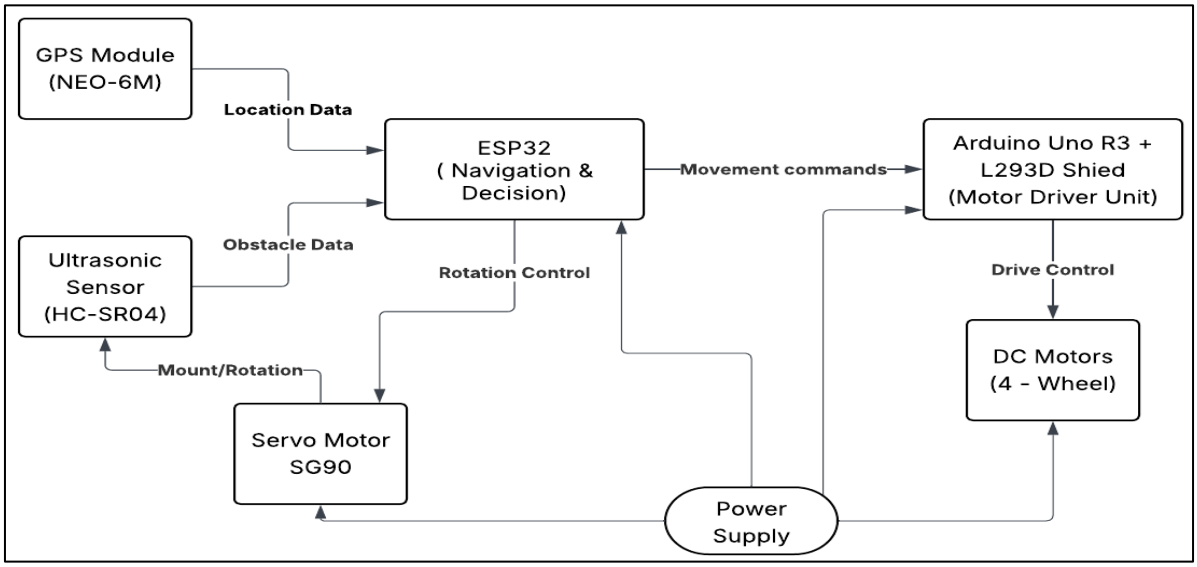

Figure 10 System architecture block diagram

2.4. Circuit Design

2.4.1. Power

- **Battery pack →**
  - Batt + → M+ on L293D shield (EXT_PWR screw terminal)
  - Batt - → GND (EXT_PWR screw terminal)

2.5. Arduino Uno + L293D Shield

- Shield sits directly on the Arduino Uno (no wires needed for motor pins).
- Motor connections:
  - M1 terminals → Top right DC motor
  - M2 terminals → Top left DC motor
  - M3 terminals → Bottom Left DC motor
  - M4 terminals → Bottom Right DC motor

2.6. Ultrasonic Sensor (HC-SR04) → Arduino Uno

- VCC → 5V(L293D)
- GND → GND(L293D)
- TRIG → D13(ESP
- ECHO → D12(ESP)

2.7. GPS Module → ESP32

- VCC → 3.3V (ESP32)
- GND → GND
- TX (GPS) → RX (ESP32 RX2)
- RX (GPS) → TX (ESP32 TX2)

2.8. ESP32 ↔ Arduino Uno Communication

- ESP32 TX0 → Arduino RX (D0)
- ESP32 RX0 → Arduino TX (D1)

2.9. Servo Motor

- 5V → 5V (L293D Servo1 header)
- GND → (L293D Servo1 header)
- PMW → D14(ESP32)

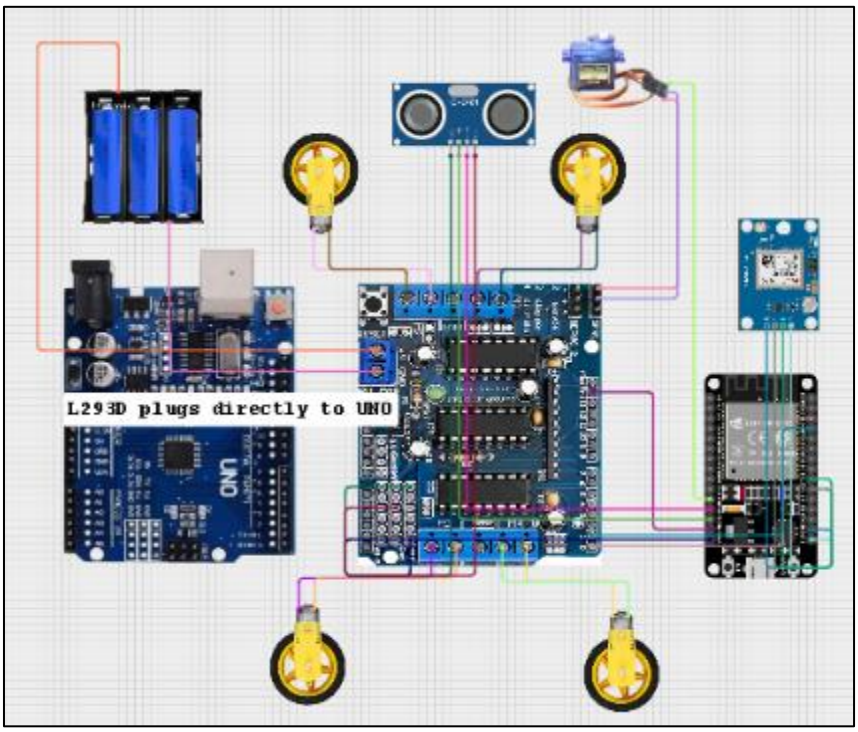

Figure 11 Circuit Design

2.10. Software Implementation

2.10.1. System Requirement

The Arduino IDE Served has the base for software development environment of the project. To ensure the proper functionality of the various models, the following libraries were required

- **Tiny GPS++**: Used with the ESP32 to parse and analyze GPS data from the NEO-6M module.
- **ESP32Servo**: Used to control SG90 servo motor that rotates the ultrasonic sensor for obstacle detection.
- **Wire/I²C (default in Arduino IDE)**: Required for communication between the ESP32 and Arduino Uno.

2.11. Writing the Code

The Arduino IDE was used to program the system. Motor driving, obstacle detection, and GPS parsing were all managed by different modules. The Arduino Uno with the L293D shield oversaw motor execution, while the ESP32 acted as the main controller.

- A simplified code structure is shown in the Appendix
- Basic code outline of the bot(car)
- #include <Tiny GPS++.h>
- #include <HardwareSerial.h>
- Tiny GPSP lus gps;
- Hardware Serial gps Serial (1);
- #define TRIG_PIN 4
- #define ECHO_PIN 5
- #define MOTOR1_IN1 26
- #define MOTOR1_IN2 27
- #define MOTOR2_IN3 14
- #define MOTOR2_IN4 12
- #define ENA 25
- #define ENB 33
- Void setup () {
- Serial. Begin (115200);
- gpsSerial.begin(9600, SERIAL_8N1, 16, 17); // RX, TX
- pin Mode (TRIG_PIN, OUTPUT);
- pin Mode (ECHO_PIN, INPUT);
- pin Mode (MOTOR1_IN1, OUTPUT);
- pin Mode (MOTOR1_IN2, OUTPUT);
- pin Mode (MOTOR2_IN3, OUTPUT);
- pin Mode (MOTOR2_IN4, OUTPUT);
- pin Mode (ENA, OUTPUT);
- pin Mode (ENB, OUTPUT);

Void loop () {

- While (gpsSerial.available() > 0) {
- Gps.encode(gpsSerial.read());
- If (gps.location.is Updated ()) {
- Serial.print("Latitude: ");
- Serial.println(gps.location.lat());
- Serial.print("Longitude: ");
- Serial.println(gps.location.lng());

}

}

}

2.12. Code outline for GPS and movement

- #include <Wire.h>
- #include <TinyGPS++.h>
- #include <HardwareSerial.h>
- #include <Servo.h>
- #define trigPin 12
- #define echoPin 14
- #define servoPin 13
- #define motor1A 26

```
#define motor1B 25
#define motor2A 33
#define motor2B 32
#define motor3A 27
#define motor3B 4
#define motor4A 19
#define motor4B 18
Hardware Serial gps Serial (1);
Tiny GPSP lus gps;
Servo my Servo;
Float target Lat = 37.7749, target Lon = -122.4194;
Float kp = 1.5, ki = 0.01, kd = 0.1;
Float error, last Error, integral, derivative;
Float min Dist = 20.0;
Void move Forward () {
Digital Write (motor1A, HIGH);
Digital Write (motor1B, LOW);
Digital Write (motor2A, HIGH);
Digital Write (motor2B, LOW);
Digital Write (motor3A, HIGH);
Digital Write (motor3B, LOW);
Digital Write (motor4A, HIGH);
Digital Write (motor4B, LOW);
}
Void moveBackward() {
digitalWrite(motor1A, LOW);
digitalWrite(motor1B, HIGH);
digitalWrite(motor2A, LOW);
digitalWrite(motor2B, HIGH);
digitalWrite(motor3A, LOW);
digitalWrite(motor3B, HIGH);
digitalWrite(motor4A, LOW);
digitalWrite(motor4B, HIGH);
}
Void turnLeft() {
digitalWrite(motor1A, LOW);
digitalWrite(motor1B, HIGH);
digitalWrite(motor2A, HIGH);
digitalWrite(motor2B, LOW);
digitalWrite(motor3A, LOW);
digitalWrite(motor3B, HIGH);
digitalWrite(motor4A, HIGH);
digitalWrite(motor4B, LOW);
}
Void turnRight() {
digitalWrite(motor1A, HIGH);
digitalWrite(motor1B, LOW);
digitalWrite(motor2A, LOW);
digitalWrite(motor2B, HIGH);
digitalWrite(motor3A, HIGH);
digitalWrite(motor3B, LOW);
digitalWrite(motor4A, LOW);
digitalWrite(motor4B, HIGH);
}
```

```
Void stop Motors () {
Digital Write (motor1A, LOW);
Digital Write (motor1B, LOW);
Digital Write (motor2A, LOW);
Digital Write (motor2B, LOW);
Digital Write (motor3A, LOW);
Digital Write (motor3B, LOW);
Digital Write (motor4A, LOW);
digitalWrite(motor4B, LOW);
Float get Distance () {
Digital Write (trigPin, LOW);
Delay Microseconds (2);
Digital Write (trigPin, HIGH);
Delay Microseconds (10);
Digital Write (trigPin, LOW);
return pulseIn(echoPin, HIGH) * 0.034 / 2;
}
Void controlServo(int angle) {
myServo.write(angle);
delay (500);
}
Float getHeading(float lat1, float lon1, float lat2, float lon2) {
Float dLon = (lon2 – lon1);
Float y = sin(dLon) * cos(lat2);
Float x = cos(lat1) * sin(lat2) – sin(lat1) * cos(lat2) * cos(dLon);
Float heading = atan2(y, x) * 180.0 / PI;
If (heading < 0) heading += 360;
Return heading;
}
Void controlRobot(float currentLat, float currentLon, float heading) {
Float targetHeading = getHeading(currentLat, currentLon, targetLat, targetLon);
Error = targetHeading – heading;
Integral += error;
Derivative = error – lastError;
Float correction = kp * error + ki * integral + kd * derivative;
lastError = error;
if (get Distance() < minDist) {
stop Motors();
control Servo(90);
delay(1000);
control Servo(0);
turn Right();
delay(500);
} else {
If (abs(correction) > 15) {
If (correction > 0) turn Right();
Else turnLeft();
} else {
Move Forward();
}
}
}
```

Void setup () {

```
Serial. begin(115200);
```

- gpsSerial.begin(9600, SERIAL_8N1, 16, 17);
- pin Mode(trigPin, OUTPUT);
- pin Mode(echoPin, INPUT);
- pin Mode(motor1A, OUTPUT);
- pin Mode(motor1B, OUTPUT);
- pin Mode(motor2A, OUTPUT);
- pin Mode(motor2B, OUTPUT);
- pin Mode(motor3A, OUTPUT);
- pin Mode(motor3B, OUTPUT);
- pin Mode(motor4A, OUTPUT);
- pin Mode(motor4B, OUTPUT);
- my Servo.attach(servoPin);
- my Servo.write(0);
- }
- Void loop() {
- While (gpsSerial.available()) {
- Gps.encode(gps Serial.read());
- If (gps.location.is Updated()) {
- Float current Lat = gps.location.lat();
- Float current Lon = gps.location.lng();
- Float heading = gps.course.deg();
- controlRobot(currentLat, currentLon, heading);
- }
- }
- }

2.13. Testing and Calibration

- **GPS Testing:** Verified that the NEO-6M GPS module produced precise latitude and longitude readings.
- **Avoidance of Obstacles:** Ultrasonic sensor distance threshold (≈20 cm) was calibrated for accurate detection.
- **Motor Control:** Modified motor direction logic and confirmed reliable turning, forward, and backward motion.

By following these steps, a functional self-driving robot car was created, capable of GPS-based navigation and basic obstacle avoidance, forming the foundation for more advanced autonomous vehicle systems.

3. Result and analysis

3.1. Obstacle Detection Results

Table 1 summarizes the vehicle's response to objects placed at varying distances.

Table 1 Ultrasonic Sensor Readings and Vehicle Response

Measured Distance (cm)	Remark	Vehicle's Action
10	Obstacle detected	Car stops, while the system scans for alternative route
15	Obstacle detected	Car stops, while the system scans for alternative route
75	Obstacle detected	Car keeps moving forward
25	Obstacle detected	Car keeps moving forward
19	Obstacle detected	Car stops, while the system scans for alternative route

From the above table it can be seen that when the distance between an object/obstacle and the vehicle is less than 20cm, the car stops while the ultrasonic sensor keeps scanning for alternative (obstacle free) routes. The process of scanning and determining a new route takes the system a maximum of 3 seconds.

Table 2 Summary of Key Performance Parameters

Parameter	Observation	Remark
Navigation accuracy	The car successfully followed the predefined path	There was some minor deviations in complex environments
Obstacle detection	Ultrasonic sensor was able to detect obstacles within a range of up to 400cm	The sensor performance can be compromised by environmental factors such as humidity, air pressure, and temperature.
Obstacle avoidance	Successful avoided static obstacles	Struggled with unpredictable moving objects
Decision making speed	Scanning and determining a new route takes approximately 3 seconds	Response time needs optimization for dynamic obstacles

3.2. Analysis

Key findings from the experimental results include

- The mini autonomous vehicle demonstrated a high level of autonomy in structured (static) environments.
- Reliable obstacle detection ensured smooth navigation.
- Efficient power consumption allowed for prolonged operation.
- Performance declined in environments with bad terrain and dynamic obstacles.
- Wireless communication faced occasional interference, leading to minor delays.

4. Conclusion

In this project, a mini autonomous vehicle with GPS navigation and ultrasonic sensors for obstacle avoidance was demonstrated. Although processing constraints and environmental factors occasionally impacted performance, the system successfully combined sensor data with decision-making algorithms to enable safe and autonomous travel. Future improvements such as compass or vision-based modules, enhanced computational capacity, and real-time monitoring could further strengthen reliability and expand functionality. Overall, the work provides a solid foundation for advancing autonomous mobility research and development.

Integration Of V2i And V2x Systems: Opportunities and Future Directions

- Potential for V2I/V2X Integration in Mini AV Prototypes

The mini autonomous vehicle (AV) developed in this study provides a foundation for advancing research into Vehicle-to-Infrastructure (V2I) and Vehicle-to-Everything (V2X) communication. Although the current prototype primarily employs GPS and ultrasonic sensors for navigation and obstacle avoidance, its modular architecture makes it well-suited for expansion into connectivity-based experiments.

Integrating V2I would enable the vehicle to interact with intelligent transportation infrastructure, such as smart traffic signals and roadside units (RSUs), thereby simulating real-world applications like adaptive traffic signal priority, real-time hazard warnings, and congestion management. For example, a prototype can be programmed to adjust its speed upon receiving Signal Phase and Timing (SPaT) information from a connected traffic light, which has been shown to reduce stops and energy consumption in real-world scenarios (Sabir and Amine, 2021). Similarly, V2X connectivity extends these capabilities beyond infrastructure by enabling direct Vehicle-to-Vehicle (V2V) communication, which supports cooperative maneuvers such as platooning. Research on cooperative adaptive cruise control shows that inter-vehicle communications combined with edge computing can smooth traffic flow, reduce delays, and enhance safety (Chang et al., 2019). More recently, decentralized multi-agent reinforcement learning has been applied to V2V platooning, demonstrating scalable solutions for coordinated vehicle behavior in complex environments (Abdelrahman et al., 2023).

Thus, the prototype can serve as a low-cost, experimental testbed for simulating connected vehicle ecosystems, bridging the gap between classroom demonstrations and real-world transportation engineering applications.

Future directions

Building on the prototype presented in this study, several promising directions for future research and development are identified.

- **Integration of Advanced Communication Modules:** While the ESP32 offers Wi-Fi and Bluetooth, future iterations could integrate LTE/5G or dedicated C-V2X modules to support low-latency communication. This would enable the mini-AV to participate in cooperative safety applications such as intersection movement assistance and real-time traffic rerouting.
- **Cooperative Driving and Platooning:** Multiple mini-AVs could be deployed to test cooperative driving strategies. Recent studies using reinforcement learning for altruistic cooperative adaptive cruise control show that mixed traffic environments can still benefit from partial penetration of V2V-equipped vehicles, improving stability and efficiency (Lu et al., 2023).
- **Simulation of Connectivity Loss:** Testing system robustness under intermittent or failed connectivity is essential. Model Predictive Control (MPC)-based studies demonstrate that communication loss significantly affects platoon stability and vehicle safety, emphasizing the need for fault-tolerant communication strategies (Razzaghpour et al., 2021).
- **Smart Intersection Prototyping:** Creating scaled traffic light systems capable of broadcasting SPaT data to the prototype would simulate adaptive signal control and priority routing, which is a critical application of V2I systems.
- **AI and Vision-based Integration:** Incorporating cameras and computer vision algorithms into the prototype would allow traffic sign recognition and pedestrian detection. When combined with V2X data, this creates redundancy and enhances overall reliability.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Alex, F.J., Tan, G., Kyei, S.K. et al. (2023) 'Transmission of viruses and other pathogenic microorganisms via road dust: emissions, characterization, health risks, and mitigation measures', Atmospheric Pollution Research, 14(1), Article 101642.

[2] Arduino Forum (2025) NEO-6M GPS Module heating, LED off. Available at: https://forum.arduino.cc/t/neo-6m-gps-module-heating-led-off/852130

[3] Az-Delivery (2025) L293D Motor Driver Shield, 4-Channel driver engine shield. Available at: https://www.az-delivery.de/en/products/4-kanal-l293d-motortreiber-shield-schrittmotortreiber

[4] Barberi, S., Arena, F. and Termine, F. (2022) 'BIM applied to intelligent transport systems', AIP Conference Proceedings, 2611(1), Article 060011.

[5] Botland.store (2025) DC Motor with 1:48 gear 3-6V with double sided shaft 200RPM. Available at: https://botland.store/geared-dc-angle-motors/16016-dc-motor-with-148-gear-3-6v-with-double-sided-shaft-200rpm-5904422344078.html

[6] EmbeTronicx (2025) Ultrasonic Sensor [HC-SR04] basics. Available at: https://embetronicx.com/tutorials/tech_devices/ultrasonic_sensor/

[7] ElectronicsWings (2025) Servo Motor Basics, Working principle and interfacing with Arduino. Available at: https://www.electronicwings.com/sensors-modules/servo-motor

[8] Fagnant, D.J., Kockelman, K.M. and Bansal, P. (2015) 'Operations of shared autonomous vehicle fleet for Austin, Texas, market', Transportation Research Record, 2563, pp. 98–106.

[9] Greenwald, J. and Kornhauser, A. (2019) 'It's up to us: Policies to improve climate outcomes from automated vehicles', Energy Policy, 127, pp. 445–451. https://doi.org/10.1016/j.enpol.2018.12.017

[10] Kalra, N. and Paddock, S.M. (2016) 'Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?', Transportation Research Part A: Policy and Practice, 94, pp. 182–193. https://doi.org/10.1016/j.tra.2016.09.010

[11] Pappalardo, G., Caponetto, R., Varrica, R. et al. (2022) 'Assessing the operational design domain of lane support system for automated vehicles in different weather and road conditions', Journal of Traffic and Transportation Engineering (English Edition), 9(4), pp. 631–644.

[12] PKNERGY (2025) 3.7V Li-ion 21700 5000mAh. Available at: https://www.pknergy.com/3-7v-li-ion-21700-5000mah-product/

[13] Shetty, A., Yu, M., Kurzhanskiy, A., Grembek, O., Tavafoghi, H. and Varaiya, P. (2021) 'Safety challenges for autonomous vehicles in the absence of connectivity', Transportation Research Part C: Emerging Technologies, 128, 103133. https://doi.org/10.1016/j.trc.2021.103133

[14] World Health Organization (2023) Road traffic injuries: Fact sheet, 13 December. Available at: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[15] Zhang, R., Spieser, K., Frazzoli, E. et al. (2015) Models, algorithms, and evaluation for autonomous mobility-on-demand systems. Proceedings of the American Control Conference (ACC), Chicago.

[16] Sabir, M. and Amine, A. (2021) 'PrOMor: A proposed prototype of V2V and V2I for crash prevention in the Moroccan case', Advances in Science, Technology and Engineering Systems Journal, 6(1), pp. 178–185. Available at: https://www.astesj.com/v06/i01/p22/

[17] Chang, B.-J., Tsai, C.-Y. AND Lai, S.-F. (2019) 'Cooperative adaptive driving for platooning autonomous self-driving based on edge computing', International Journal of Applied Mathematics and Computer Science, 29(1), pp. 143–154. https://doi.org/10.2478/amcs-2019-0016

[18] Abdelrahman, A., Gu, Y. AND Mo, Y. (2023) Scalable decentralized cooperative platoon using multi-agent deep reinforcement learning. arXiv preprint. Available at: https://arxiv.org/abs/2312.06858

[19] Hussain, R., Son, J. AND Kim, H. (2019) 'Cybersecurity challenges in vehicular communications: A survey', Vehicular Communications, 21, 100182. https://doi.org/10.1016/j.vehcom.2019.100182

[20] Shah, G., Zaman, M. AND Sattar, A. (2023) 'Scalable cellular V2X solutions: Large-scale deployment challenges of connected vehicle safety networks', IEEE Access, 11, pp. 23450–23462. Available at: https://cs.paperswithcode.com/paper/scalable-cellular-v2x-solutions-large-scale

[21] Zhou, Y., Chen, L. AND Zhang, S. (2024) 'Sampled-data cooperative adaptive cruise control for string-stable vehicle platooning with communication delays: A linear matrix inequality approach', Machines, 12(3), 165. https://doi.org/10.3390/machines12030165

[22] Lu, Z., Yang, C. AND Chen, H. (2023) 'Altruistic cooperative adaptive cruise control of mixed traffic platoon based on deep reinforcement learning', IET Intelligent Transport Systems, 17(12), pp. 1748–1761. https://doi.org/10.1049/itr2.12386

[23] Razzaghpour, M., Kamal, M. AND Mukai, M. (2021) Impact of communication loss on MPC-based cooperative adaptive cruise control and platooning. arXiv preprint. Available at: https://arxiv.org/abs/2106.09094

[24] Wang, Y., Li, X. AND Chen, F. (2023) 'Safety assessment of cooperative platooning in mixed traffic', Proceedings, 36(1), 38. https://doi.org/10.3390/proceedings3610038