# A PROJECT REPORT ON ARDUINO BASE OBSTACLE AVOIDING CAR

**Crimson Technical College**

**Butwal-11, Rupandehi, Nepal**



**Diploma in Computer Engineering 2078 Batch**

**DEPARTMENT OF COMPUTER ENGINEERING**

This Major Project Report is submitted in partial fulfillment of the requirements for the Diploma in Computer Engineering awarded by CTEVT.

## SUBMITED BY D-COM 6th SEMETER STUDENTS

|  |  |
| --- | --- |
| **MEMBERS** | **REGISTRATION NO** |
| AKASH THARU | DCOM-00940-021 |
| AKLESH Ku. YADAV | DCOM-00308-021 |
| PARAS KHADKA | DCOM-00321-021 |
| MAUSAM SHREES | DCOM-00317-021 |
| BIBEK B.K. | DCOM-00310-021 |

Date: 2081 / 08 / 05

Crimson Technical College

Butwal-11, Rupandehi, Nepal

.…................... ..….…….........….

Er. Rajiv Aryal **External Examiner**

**Project Supervisor**

# ACKNOWLEDGEMENT

I would like to sincerely thank everyone who helped and supported me in completing the "**Obstacle Avoiding Car**" project.

First, I want to thank my project guide, Er. Rajiv Aryal, for his constant support, guidance, and valuable advice. His expertise and suggestions helped me overcome challenges and improve the project.

I am also grateful to my teammates for their hard work, patience, and encouragement throughout the project. We worked together as a team, and their contributions were essential to completing this project.

I would also like to thank the online communities and open-source platforms that provided useful information and solutions. These resources helped me solve technical problems and improve the project.

Finally, I appreciate everyone’s help and support in making this project a success.

- **Project Leader**

**Akash Tharu**

# ABSTRACT

|  |  |
| --- | --- |
| **Project Leader** | Akash Tharu |
| **Title** | Obstacle Avoiding Car |
| **Year** | 2081 |
| **Language** | English |
| **Pages** | 25 |
| **Supervisor** | Er. Rajiv Aryal |

This project, titled "Obstacle Avoiding Car," is about building a small robot car that can move around on its own while avoiding obstacles in its path. The car uses an Arduino Uno microcontroller to control its actions, along with ultrasonic sensors to detect objects nearby. When the car senses an obstacle, it automatically changes direction to avoid crashing.

The main goal of this project is to create an affordable and easy-to-build car that can be used in different situations, like navigating in a home or small workspace. The car’s ability to detect and avoid obstacles on its own makes it a useful tool for tasks where automatic movement is needed.

This project shows how simple technology, like sensors and a microcontroller, can be used to make a car that moves on its own. It can be easily improved or customized for different uses, making it a great learning project for robotics and engineering

# TABLE OF CONTENTS

1. **INTRODUCTION:**

**1.1 Overview of Home Automation 1.2 Project Objectives 1.3 Scope of the Project**

1. **SYSTEM DESIGN:**

## 2.1 Block Diagram of Home Automation

**2.2 Components Overview:**

**2.2.1 Arduino Uno (1x) 2.2.2 Motor Driver Shield (1x) 2.2.3 Servo Motor (1x) 2.2.4 Ultrasonic Sensor (1x) 2.2.5 DC Gear Motor (4x) 2.2.6 Wheels (4x) 2.2.7 Li-on Battery (2x) with Battery Holder**

**2.2.8 DC Power Switch**

**2.2.9 Card Board**

**2.2.10 Wire**

1. **SOFTWARE DESIGN:** 
   1. **Arduino IDE and Programming 3.2 System Algorithms**
2. **IMPLEMENTATION:** 
   1. **Hardware Assembly 4.2 Code Integration and Testing 4.3 Troubleshooting and Debugging**
3. **TESTING AND RESULTS:** 
   1. **Test Cases and Methodology 5.2 Performance Evaluation 5.3 Results and Observations**
4. **CHALLENGES AND SOLUTIONS:**

## 6.1 Technical Challenges 6.2 Software-Related Issues

**6.3 Hardware-Related Issues**

1. **GANNT CHART:**
2. **CONCLUSION:**
3. **INTRODUCTION:**
   1. **Overview of Home Automation:**

“Obstacle Avoiding Car” is a simple robot designed to navigate its surroundings without colliding with objects. It relies on a sensor, such as an ultrasonic sensor, to detect obstacles in its path and adjust its direction to avoid them. In this project, an obstacle-avoiding car is built using an Arduino as the central control unit, along with an ultrasonic sensor for detection, a servo motor for directional control, and DC motors for movement. The primary goal is to create an affordable, efficient, and reliable robot that can operate in various environments, such as warehouses, homes, or outdoor areas. This project demonstrates the effective integration of hardware and software, showcasing how a microcontroller processes sensor data and controls actuators to perform tasks autonomously. The design also allows for scalability, enabling future enhancements like wireless control, improved sensors, or AI-based navigation, further broadening its real-world applications.

* 1. **Project Objectives:**

**Design and Build**: Create an Arduino-based obstacle avoiding car using sensors, motors, and other components for smooth operation.

**Obstacle Detection**: Use ultrasonic sensors to detect objects in the car's path and guide it to avoid collisions.

**Efficient Movement:** Ensure the car moves efficiently by adjusting its speed and direction based on the detected obstacles.

**Cost-Effective Solution**: Build a simple and affordable robot that is easy to assemble and maintain.

**Flexibility and Upgrades**: Design the system so that it can be improved or expanded by adding more features in the future.

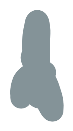
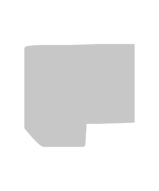
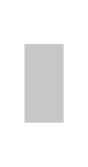
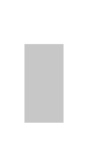
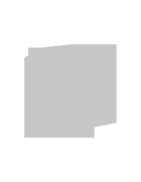
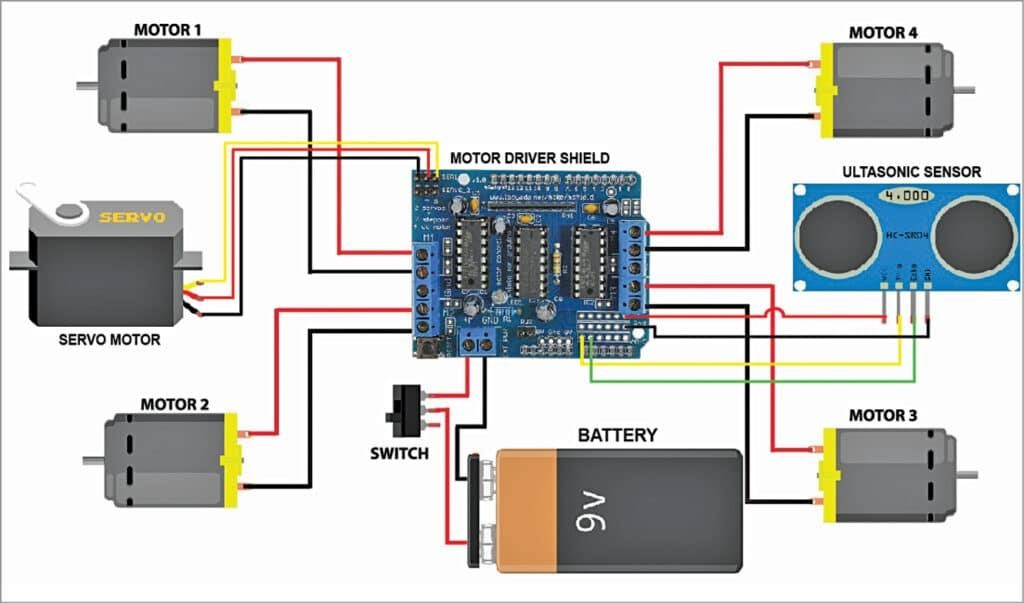
**Testing and Optimization**: Test the car in different conditions to make sure it works reliably and smoothly in real-life situations.

**1.3 Scope of the Project:**

The scope of the “**Obstacle Avoiding Car**" project focuses on designing and building a robot that can detect and avoid obstacles using an Arduino Uno microcontroller. The car will use ultrasonic sensors to detect objects in its path and adjust its movement using motors and wheels to avoid collisions. The project will include assembling the hardware, programming the microcontroller, and testing the system to ensure it works reliably in different environments. This robot is designed to be simple, efficient, and expandable, allowing additional features to be added in the future for enhanced functionality.

The system will be tested in a variety of settings, including indoors and outdoors, to ensure it can perform well in different conditions such as varying light or surfaces. The robot will be able to autonomously navigate spaces without the need for human intervention, making it ideal for practical applications in environments where human presence is limited. Additionally, future updates could include adding wireless control capabilities, improving obstacle detection accuracy, or incorporating machine learning algorithms for more intelligent navigation, making the robot more versatile and adaptive to complex environments.

1. **SYSTEM DESIGN:**
   1. **Block Diagram of the System:**



**Fig: Circuit Diagram**

* 1. **Components Overview:**

The list of components used in our project are briefed below:

* + 1. **Arduino Uno (1x):**

The Arduino Uno is a popular microcontroller board based on the ATmega328P chip. It is widely used in electronics projects for its simplicity, versatility, and ease of programming. The board features 14 digital input/output pins, 6 analog input pins, a 16 MHz quartz crystal, a USB connection for programming, a power jack, and a reset button. The Arduino Uno can be powered via USB or an external power supply, making it ideal for beginners and hobbyists to develop interactive projects such as sensors, motors, and home automation systems.

* + 1. **Moter Driver Shield (1x):**

The Motor Driver Shield is used to control the DC motors' speed and direction. It acts as an interface between the Arduino Uno and the DC motors, allowing the microcontroller to send signals that control motor functions. The motor driver ensures that the motors receive the correct voltage and current for smooth and efficient operation, including forward, reverse, and turning movements.

* + 1. **Servo Motor (1x):**

The Servo Motor controls the steering mechanism of the obstacle-avoiding car. It allows precise movement within a range of 0 to 180 degrees, which helps the robot navigate by turning its front wheels to avoid obstacles. The servo is powered and controlled by the Arduino through PWM (Pulse Width Modulation) signals, providing accurate angle adjustments for the car's steering.

* + 1. **Ultrasonic Sensor (1x):**

The Ultrasonic Sensor is used for obstacle detection. It sends out high-frequency sound waves and measures the time taken for the waves to bounce back after hitting an object. This sensor helps the robot determine the distance to objects in its path and triggers the system to alter its movement when an obstacle is detected. It is crucial for autonomous navigation.

* + 1. **DC Gear Motors (4x):**

The DC Gear Motors provide the driving force to move the obstacle-avoiding car. These motors are equipped with gears that offer high torque at low speeds, making them ideal for driving the car’s wheels. The four motors are mounted on the car's chassis, allowing for forward, backward, and turning movements when controlled by the motor driver shield.

* + 1. **Wheels (4x):**

The wheels are mounted on the DC gear motors to provide movement to the robot. These wheels are designed to be durable and allow the car to travel smoothly on different surfaces. The four wheels give the robot stability and control over its motion, enabling it to navigate around obstacles effectively.

* + 1. **Li-ion Battery 3V (2x) & Battery Holder:**

The Li-ion Batteries provide the necessary power to run the Arduino, motors, servo, and other components. These rechargeable batteries are lightweight and offer a good energy density, making them ideal for mobile robotics projects. The battery holder ensures that the batteries are securely positioned and easily accessible for recharging.

* + 1. **DC Power Switch (1x):**

The DC Power Switch allows the user to control the power supply to the entire robot. It acts as a simple on/off switch to turn the system on and off, ensuring that the robot only consumes power when needed. This component is crucial for managing energy usage and ensuring the system is safe to operate. The DC power switch also helps prevent unnecessary battery drain when the robot is not in use, extending the life of the power source and providing a convenient way to power down the system during storage or transportation.

* + 1. **Cardboard (1x):**

Cardboard is used to create the chassis and structure of the obstacle-avoiding car. It is lightweight and easy to work with, making it a practical choice for prototyping the robot. The cardboard chassis holds the various components together, providing a stable base for the motors, sensors, and Arduino board.

* + 1. **Wires:**

Wires are essential for connecting all the components of the robot, including the Arduino Uno, motor driver, motors, servo, sensors, and power supply. They provide the electrical connections needed to transfer signals and power between the components. Jumper wires of various lengths and types (male-to-male, male-tofemale) are used for these connections, ensuring that the circuit is properly assembled.

**SOFTWARE DESIGN:**

* 1. **Arduino IDE and Programming:**

The Arduino IDE (Integrated Development Environment) is used for writing, compiling, and uploading code to the Arduino Uno microcontroller for the "Obstacle Avoiding Car" project. The IDE supports programming in C/C++ and offers built-in functions to control various components such as sensors, motors, and servos. It simplifies the development process by allowing easy integration of libraries, such as those for ultrasonic sensors and servo motors. For this project, the Arduino is programmed to interpret data from the ultrasonic sensors, calculate the distance to obstacles, and then control the motors to adjust the car's direction. The

Arduino IDE’s simplicity and flexibility make it perfect for developing and testing the functionality of the robot.

* 1. **System Algorithms:**

The system algorithms for the "Obstacle Avoiding Car" project are designed to enable the robot to autonomously navigate its environment without colliding with objects. The main algorithm continuously monitors the data from the ultrasonic sensors to detect obstacles in front of the car. When an obstacle is detected, the car adjusts its movement by controlling the servo motor for steering and the DC motors for movement. The algorithm ensures smooth, real-time navigation by processing sensor data, detecting objects at varying distances, and making quick decisions to avoid collisions. This system aims to optimize the car's pathfinding capabilities while minimizing errors or delays in movement.

* 1. **Sensor Data Processing:**

In the "Obstacle Avoiding Car" project, sensor data from the ultrasonic sensors is used to detect objects in the robot’s path. The sensor sends a signal, which bounces off nearby objects and returns to the sensor. Based on the time it takes for the signal to return, the Arduino calculates the distance to the obstacle. When an object is detected within a predefined threshold distance, the car’s movement is adjusted by controlling the DC motors and servo motor to either stop, move backward, or change direction. The data is processed in real-time to ensure immediate response and smooth navigation.

* 1. **Moter Control Algorithm:**

The motor control algorithm in this project is responsible for adjusting the robot’s movements based on sensor input. When the car is free of obstacles, it moves forward. If an obstacle is detected within a certain range, the algorithm initiates a turn or reverse movement. The servo motor is used for steering, adjusting the car’s direction, while the DC motors control forward and backward motion. The algorithm ensures that the robot can autonomously make decisions, such as stopping, reversing, or turning, to navigate around objects. This method allows the car to avoid obstacles efficiently, making it a fully functional autonomous robot.

1. **IMPLEMENTATION:**
   1. **Hardware Assembly:**

The hardware assembly for the "Obstacle Avoiding Car" project involves connecting and integrating several components to create a fully functional autonomous robot. The Arduino Uno serves as the central controller, interfacing with the ultrasonic sensors for obstacle detection, the DC motors for movement, and the servo motor for steering. The motor driver shield is used to control the motors, providing the necessary power and control signals to ensure the car moves efficiently. Components such as the motors, sensors, and power supply are connected through jumper wires on a breadboard, enabling smooth data transfer and control signals. Power is supplied via the Li-on batteries and a DC power switch, which allows the user to control the car's power. The assembly ensures that all hardware components work together to enable the robot to autonomously navigate around obstacles.

* 1. **Code Integration and Testing:**

The code integration and testing phase for the "Obstacle Avoiding Car" project involves combining the different code modules responsible for controlling the sensors, motors, and servo. Once the code is integrated, it is uploaded to the Arduino Uno, which manages tasks like reading data from the ultrasonic sensors, calculating distances, and controlling the motors to avoid obstacles. The integrated code ensures the car's movement is continuously adjusted based on sensor input to avoid collisions. After integration, the system undergoes extensive testing to ensure all components function correctly. Testing includes verifying sensor accuracy, motor responsiveness, and the overall stability of the system. The car’s ability to avoid obstacles in different conditions is thoroughly tested to ensure smooth operation.

**Here’s a simplified version of the source code for our project:**

#include <AFMotor.h> // Library to control motors with Adafruit Motor Shield

#include <NewPing.h> // Library for ultrasonic sensor (distance measurement)

#include <Servo.h> // Library for controlling servo motors

|  |
| --- |
| #define TRIG\_PIN A0 // Pin for the TRIG of the ultrasonic sensor  #define ECHO\_PIN A1 // Pin for the ECHO of the ultrasonic sensor  #define MAX\_DISTANCE 200 // Maximum distance (in cm) the ultrasonic sensor can measure  #define MAX\_SPEED 140 // Maximum speed for DC motors  #define MAX\_SPEED\_OFFSET 15 // Offset for maximum speed to reduce load    // Initialize NewPing object for the ultrasonic sensor  NewPing sonar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);    // Initialize four DC motors connected to the Adafruit Motor Shield  AF\_DCMotor motor1(1, MOTOR12\_1KHZ);  AF\_DCMotor motor2(2, MOTOR12\_1KHZ);  AF\_DCMotor motor3(3, MOTOR34\_1KHZ);  AF\_DCMotor motor4(4, MOTOR34\_1KHZ);    // Initialize servo for turning the ultrasonic sensor  Servo myservo;  boolean goesForward = false; // Flag to check if the car is moving forward int distance = 100; // Initial distance for obstacle detection int speedSet = 0; // Speed setting variable for motors    void setup() { myservo.attach(10); // Attach the servo motor to pin 10 myservo.write(115); // Set the servo to the neutral position delay(2000); // Wait for 2 seconds for setup    // Read the initial distance from the ultrasonic sensor distance = readPing(); delay(100);  distance = readPing(); delay(100);  distance = readPing(); delay(100);  distance = readPing(); delay(100);  }    void loop() {  int distanceR = 0; // Variable for the distance to the right int distanceL = 0; // Variable for the distance to the left delay(40); // Small delay to prevent sensor overload    // If an obstacle is detected within 50 cm, stop and avoid it if(distance <= 50) {  moveStop(); // Stop the car delay(100); // Wait for 100ms moveBackward(); // Move the car backward  delay(300); // Delay to allow the car to move backward moveStop(); // Stop the car delay(200); // Wait for 200ms    // Look to the right and left for better path  distanceR = lookRight(); delay(200);  distanceL = lookLeft(); delay(200);    // Choose the direction with more space to turn if(distanceR >= distanceL) { |

|  |
| --- |
| turnRight(); // Turn right if the right side has more space moveStop(); // Stop the car after the turn  } else {  turnLeft(); // Turn left if the left side has more space moveStop(); // Stop the car after the turn }  } else {  moveForward(); // Continue moving forward if no obstacle is detected }    // Update the distance after each loop distance = readPing();  }    // Function to check the right side for obstacles int lookRight() {  myservo.write(50); // Turn the servo to the right delay(500); // Wait for the servo to stabilize int distance = readPing(); // Read the distance from the right delay(100); // Small delay  myservo.write(115); // Reset the servo back to the neutral position return distance; // Return the measured distance }    // Function to check the left side for obstacles int lookLeft() {  myservo.write(170); // Turn the servo to the left delay(500); // Wait for the servo to stabilize int distance = readPing(); // Read the distance from the left delay(100); // Small delay  myservo.write(115); // Reset the servo back to the neutral position return distance; // Return the measured distance }    // Function to read the distance from the ultrasonic sensor int readPing() {  delay(70); // Delay for sensor reading int cm = sonar.ping\_cm(); // Get the distance in cm  if(cm == 0) { // If the sensor returns 0, set it to 250 (out of range) cm = 250;  }  return cm; // Return the measured distance }    // Function to stop the car void moveStop() {  motor1.run(RELEASE); // Stop motor 1 motor2.run(RELEASE); // Stop motor 2 motor3.run(RELEASE); // Stop motor 3 motor4.run(RELEASE); // Stop motor 4 }    // Function to move the car forward  void moveForward() { if(!goesForward) {  goesForward = true; // Set the flag to true motor1.run(FORWARD); // Move motor 1 forward motor2.run(FORWARD); // Move motor 2 forward motor3.run(FORWARD); // Move motor 3 forward motor4.run(FORWARD); // Move motor 4 forward |
| // Gradually increase speed to avoid draining the battery too quickly for(speedSet = 0; speedSet < MAX\_SPEED; speedSet += 2) {  motor1.setSpeed(speedSet); // Set motor 1 speed motor2.setSpeed(speedSet); // Set motor 2 speed motor3.setSpeed(speedSet); // Set motor 3 speed motor4.setSpeed(speedSet); // Set motor 4 speed  delay(5); // Delay to smooth the speed transition }  }  }    // Function to move the car backward void moveBackward() {  goesForward = false; // Set the flag to false (moving backward)  motor1.run(BACKWARD); // Move motor 1 backward motor2.run(BACKWARD); // Move motor 2 backward motor3.run(BACKWARD); // Move motor 3 backward motor4.run(BACKWARD); // Move motor 4 backward    // Gradually increase speed to avoid draining the battery too quickly for(speedSet = 0; speedSet < MAX\_SPEED; speedSet += 2)  { motor1.setSpeed(speedSet); // Set motor 1 speed motor2.setSpeed(speedSet); // Set motor 2 speed motor3.setSpeed(speedSet); // Set motor 3 speed motor4.setSpeed(speedSet); // Set motor 4 speed  delay(5); // Delay to smooth the speed transition }  }    // Function to turn the car to the right  void turnRight() { motor1.run(FORWARD); // Move motor 1 forward motor2.run(FORWARD); // Move motor 2 forward motor3.run(BACKWARD); // Move motor 3 backward motor4.run(BACKWARD); // Move motor 4 backward  delay(500); // Wait for 500ms for the turn to complete motor1.run(FORWARD); // Move motor 1 forward again motor2.run(FORWARD); // Move motor 2 forward again motor3.run(FORWARD); // Move motor 3 forward again motor4.run(FORWARD); // Move motor 4 forward again }    // Function to turn the car to the left void turnLeft() { motor1.run(BACKWARD); // Move motor 1 backward motor2.run(BACKWARD); // Move motor 2 backward motor3.run(FORWARD); // Move motor 3 forward motor4.run(FORWARD); // Move motor 4 forward  delay(500); // Wait for 500ms for the turn to complete motor1.run(FORWARD); // Move motor 1 forward again motor2.run(FORWARD); // Move motor 2 forward again motor3.run(FORWARD); // Move motor 3 forward again motor4.run(FORWARD); // Move motor 4 forward again } | | |

**4.3 Troubleshooting and Debugging:**

Troubleshooting and debugging in the "Obstacle Avoiding Car" project focus on identifying and resolving issues that may occur during hardware setup or code execution. Common challenges include improper motor wiring, faulty ultrasonic sensor readings, or failure in motor driver connections. Debugging starts with checking hardware connections and ensuring proper voltage and wiring. For software debugging, the Arduino IDE’s serial monitor is used to display sensor readings and track the flow of data. If the sensor is not detecting obstacles correctly, the timing or calculation in the code may need adjustment. Once all issues are addressed, the system is thoroughly tested to ensure reliable obstacle detection and smooth navigation. This iterative process ensures the car functions autonomously in real-world conditions, avoiding obstacles efficiently.

1. **TESTING AND RESULTS:**
   1. **Test Cases and Methodology:**

The test cases and methodology for the "Obstacle Avoiding Car" project are designed to verify the robot's ability to navigate autonomously and avoid obstacles in real-world scenarios. Each test case is focused on a specific aspect of the system, such as sensor accuracy, motor responsiveness, and obstacle detection.

**The testing methodology follows these steps.**

* + - 1. **Unit Testing:** Verify Test each component individually (motors, sensors, and servo) for functionality and performance.
      2. **Integration Testing:** Combine components and test the system to ensure that sensors and motors work together as expected.
      3. **Real-World Scenario Testing:** Test the car in various environments (indoor, outdoor, on different surfaces) to simulate different conditions and obstacle types.
      4. **User Testing:** Evaluate the system’s performance in real-world use, ensuring the robot reacts appropriately to obstacles and handles navigation tasks effectively.

**Example test cases include**:

* + 1. **Obstacle Detection Test:** Verify that the ultrasonic sensor detects obstacles at varying distances (e.g., 30 cm, 50 cm) and triggers the correct motor response (stop or reverse).
    2. **Moter Functionality Test:** Ensure that the motors respond correctly to movement commands (forward, backward, left turn, right turn)..
    3. **Sensor Test:** Ensure that the ultrasonic sensor detects obstacles at the correct range and provides accurate distance readings.
    4. **Servo Steering Test:** Check that the servo motor moves the wheels to the appropriate angle when a turn is commanded.

This structured approach ensures that the system is robust, responsive, and capable of operating autonomously under various conditions.

* 1. **Performance Evaluation:**

Performance evaluation of the "Obstacle Avoiding Car" focuses on the robot’s ability to autonomously detect and avoid obstacles, as well as its efficiency in navigating its environment. Key performance metrics include:

* + - **Sensor Accuracy:** The precision of the ultrasonic sensor in detecting obstacles at various distances.
    - **Motor Responsivness:** The time taken for the motors to respond to commands, including turning, reversing, and moving forward.
    - **Obstacle Avoiding Efficiency:** The robot’s ability to navigate around obstacles without collisions, as well as its capacity to adjust movement based on real-time sensor feedback.
    - **Power Consumption:** The battery life of the car, particularly under continuous testing, to evaluate how efficiently power is used during operation.

The evaluation process involves testing the car in different environments, with varying obstacle types and distances, and measuring its ability to maintain smooth and responsive navigation. Additionally, factors like battery life and system stability under continuous movement are also assessed. The results help identify areas for future improvement in sensor accuracy, motor control, and energy efficiency.

* 1. **Results and Observations:**

The results and observations for the "Obstacle Avoiding Car" project highlight the robot’s ability to navigate autonomously and avoid obstacles efficiently. The system successfully detects obstacles using the ultrasonic sensor and responds by adjusting the car’s movement, such as stopping, reversing, or turning. The motors show minimal delay in response to commands, and the servo motor accurately adjusts the steering for turns.

**Key Observations:**

* + - The ultrasonic sensor accurately detects obstacles within the expected range (10 cm to 100 cm), though performance slightly degrades at very close distances (e.g., under 5 cm).
    - The car successfully avoids obstacles and reverts to a pre-defined path without significant delays in movement.
    - The robot operates effectively on flat, clear surfaces, but its performance may be affected on uneven terrains or surfaces with sharp turns.
    - Power consumption is relatively efficient, with the car running for extended periods on a single battery charge.

**However, some limitations were noted:**

* + - The car's responsiveness may be slower in complex environments with many obstacles, requiring further refinement in decision-making algorithms.
    - The car's turning radius could be improved, as it may have difficulty navigating tighter spaces.

Overall, the system meets its objectives of providing autonomous obstacle avoidance, demonstrating reliable sensor integration, and efficient motor control. The results suggest that while the system is functional, there is potential for improvement in terms of sensor range, motor agility, and navigation in more complex environments.

1. **CHALLENGES AND SOLUTIONS:**
   1. **Technical Challenges:**

The "Obstacle Avoiding Car" project encountered several technical challenges during development. One of the major issues was ensuring the ultrasonic sensor consistently detected obstacles with accurate distance readings. The sensor's range could vary depending on factors such as surface material and environmental conditions (e.g., temperature), causing occasional false readings or missed detections. Another challenge was the motor control system. The motors occasionally experienced delays in response to commands, especially when switching between different movement directions or speeds. Additionally, balancing the power requirements of the motors, sensors, and Arduino board presented challenges, especially with the limited battery life under continuous use. These challenges were addressed through iterative testing, sensor calibration, and optimizing the motor control algorithms for faster and more reliable responses.

* 1. **Software-Related Issues:**

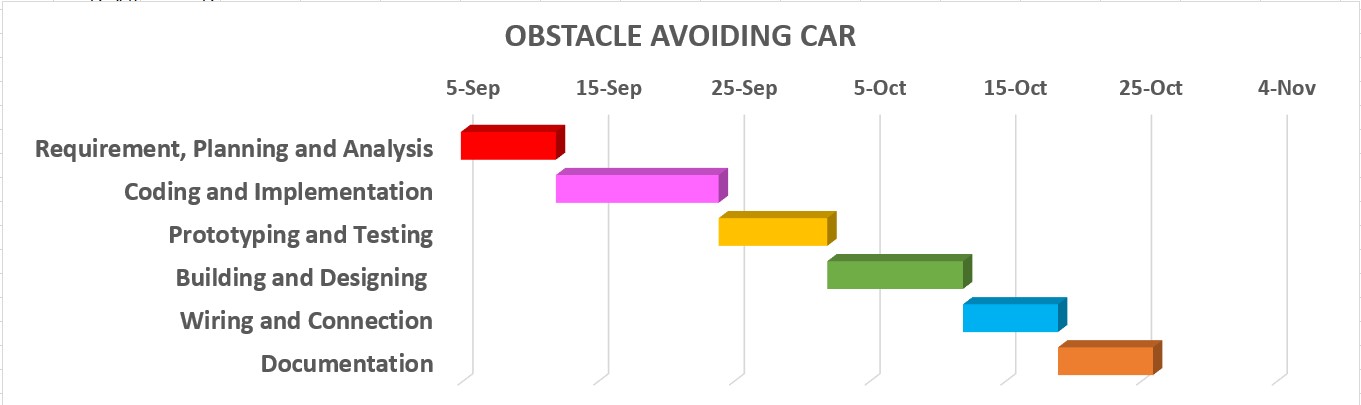
Several software-related issues arose during the development of the "Obstacle Avoiding Car." One significant challenge was fine-tuning the logic for obstacle avoidance. Initially, the car would stop or reverse too abruptly, or its path would be inefficient, requiring further refinement in the decision-making process. The car's navigation algorithm had to be optimized to respond quickly to changes in its environment and to make smooth, efficient turns. Debugging the motor control code was also challenging, as ensuring that both motors worked in sync at all times was crucial for the robot's stability and accurate movement. Additionally, integrating sensor data for real-time decision-making proved difficult, especially with noisy readings from the ultrasonic sensors. These software challenges were overcome by improving sensor filtering techniques and optimizing control flow to handle sensor inputs and motor outputs more efficiently.

* 1. **Hardware-Related Issues:**

\

The "Obstacle Avoiding Car" project faced several hardware-related issues during assembly and testing. One significant concern was ensuring stable connections between components, especially the motors and the Arduino board. Loose connections, especially with the motor drivers, occasionally caused the motors to malfunction or not respond as expected. Another hardware issue arose when selecting an appropriate power source. The initial battery setup was insufficient for the car's motors, which led to power drops and erratic behavior. The car would sometimes stall or fail to move when the battery voltage dropped below a certain threshold. Additionally, issues arose with the ultrasonic sensor’s alignment, which required precise placement on the car to ensure reliable obstacle detection at different angles. These hardware challenges were addressed by improving the wiring connections, optimizing the power supply, and recalibrating the sensor alignment to ensure consistent readings and reliable performance.

**7.GANNT CHART:**



**8.CONCLUSION:**

The “**Obstacle Avoiding Car**” is an autonomous robot that uses an Arduino Uno, ultrasonic sensors, a servo motor, and DC motors to detect and avoid obstacles in its path. The ultrasonic sensor measures distances, and the car makes decisions like stopping, turning, or moving forward to avoid collisions. This project highlights how simple components can work together to create a functional system. Despite challenges like sensor calibration and motor control, the car performed well and achieved its goal. It offers opportunities for future improvements, such as adding more sensors or smarter algorithms, making it a great step into robotics.