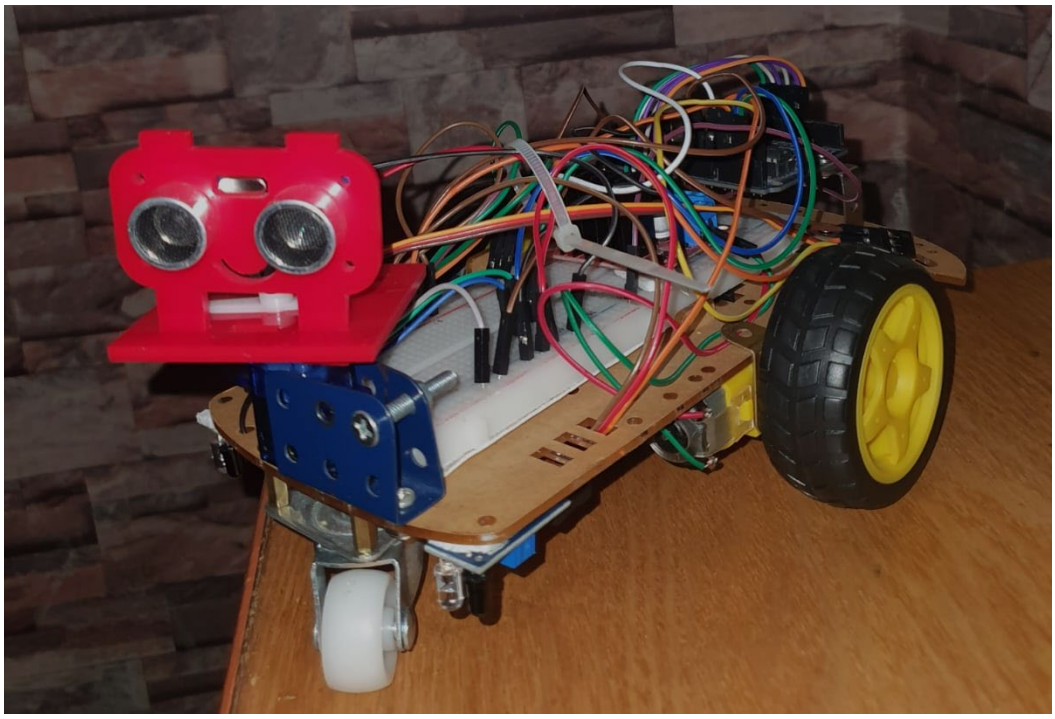
Line Follower with Obstacle Avoiding Robot

Project Name: Maze Master (Mido)

Team 2

Name	ID
Ahmed Medhat Mahfouz	20201701703
Mina George Raouf	20201701741
Ali Ahmed Hamed Shaker	20201701725
Mohamed Ibrahim Mazroa	20201701732
Mohamed Reda Abu Bakr	20201701733
Kevin Cherif	20201701729
Jana Tarek Elsherif	20201701711
Anan Mohamed Ali	20201701727
Ali Rafeeq Amer	20201701705

Content:



1. Introduction:

This project focuses on the design and implementation of a circuit for an autonomous car capable of navigating along a predetermined path while intelligently avoiding obstacles. It involves the integration of advanced technologies such as sensor arrays, microcontrollers, and sophisticated algorithms to replicate human-like cognitive abilities in vehicular automation. The challenge lies in achieving a delicate balance between speed, accuracy, and safety, contributing to the evolving field of autonomous vehicles. The successful development of such a system has the potential to redefine the future of transportation by providing safer and more intelligent mobility solutions.

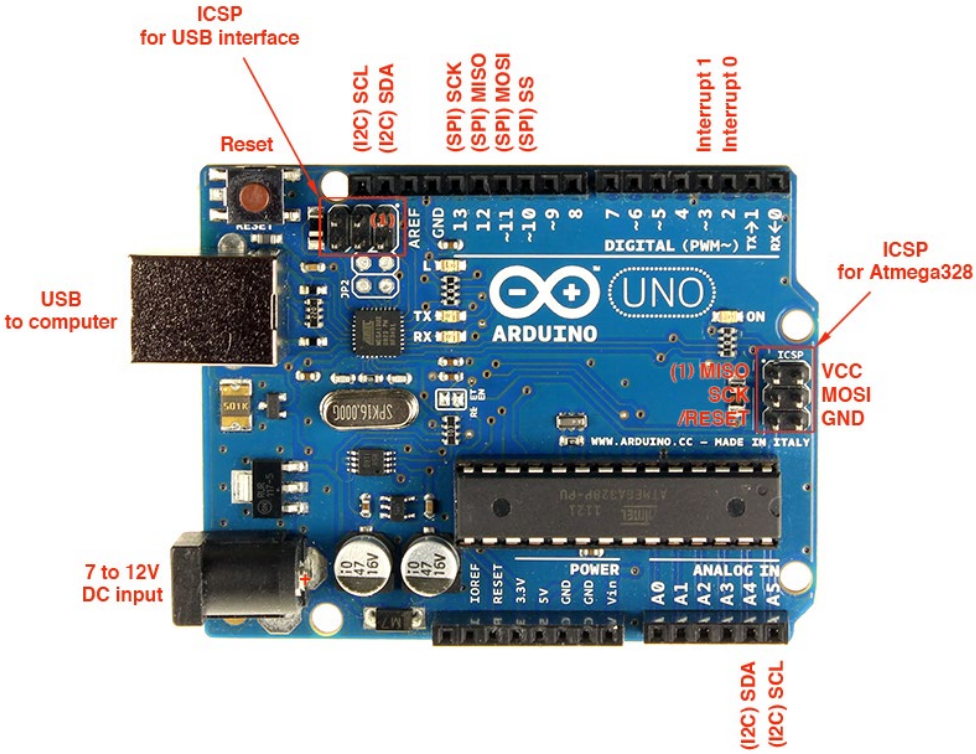
2. Components:

- Arduino Uno
- Ultrasonic Sensor
- Infrared Sensor
- H Bridge Driver (L298N)
- Servo Motor
- DC Motor
- Wheels
- Batteries
- Battery Holder
- Jumpers Wires
- Switch
- Car Chassis
- Bread board

1- Arduino Uno:

Description: The Arduino UNO is a microcontroller board based on the ATmega328P chip. It includes digital and analogue input/output pins, a USB interface for programming and power, a 16 MHz crystal oscillator, a power jack, an ICSP header, and a reset button.

Usage: The Arduino UNO is widely used for prototyping and building electronic projects. It serves as the brain of the project, executing code and controlling various components. It is particularly popular in the maker community for its ease of use and versatility in interfacing with sensors, actuators, and other electronic modules.

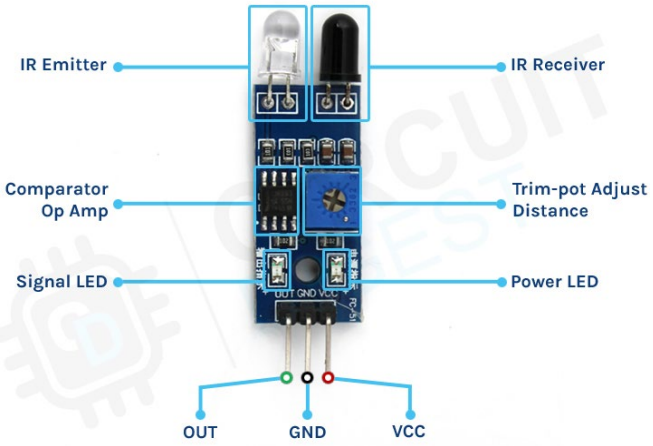


2- Ultrasonic Sensor:

Description: An ultrasonic sensor measures distance by sending out ultrasonic sound waves and calculating the time it takes for the waves to bounce back after hitting an object.

Usage: Commonly used for distance measurement and obstacle detection. It's often employed in robotics, automation, and security systems.

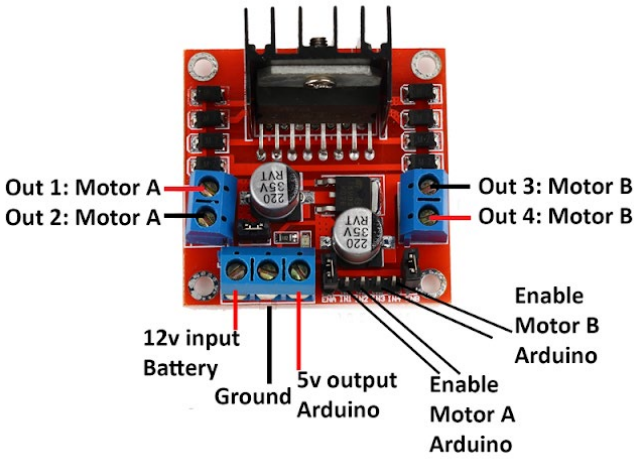
3- IR Sensor (Infrared Sensor):



Description: IR sensors use infrared light for detection. They typically consist of an emitter and a receiver. The emitter sends out infrared light, and the receiver detects the reflection or absence of the emitted light.

Usage: Widely used for object detection, proximity sensing, and line following in robotics, home automation, and electronic devices like TV remote controls.

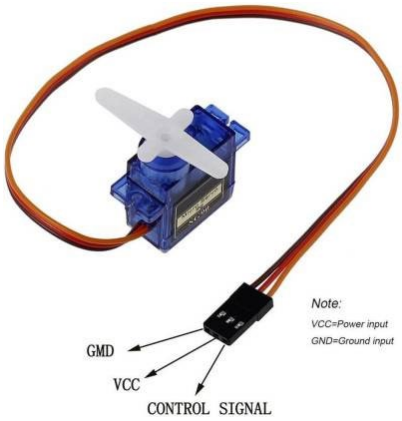
4- H-Bridge (L298N):



Description: An H-bridge is an electronic circuit used in motor control, featuring four switches arranged in an H shape. The load, often a DC motor, connects at the center, and the switches regulate current flow, enabling forward or reverse movement. The H-bridge design provides precise control over rotation, braking, and coasting.

Usage: Crucial in applications like robotics, electric vehicles, and industrial machinery, H-bridges play a vital role in accurate motor control. By selectively opening and closing switches, the motor's direction is managed – closing one pair for forward and the other for reverse. Braking involves briefly closing all switches to create a short circuit. This control versatility makes H-bridges essential for systems requiring dynamic and reversible motor operations.

5- Servo Motor:



Description: A servo motor is a rotary actuator that allows for precise control of angular position. It consists of a motor, gears, and a feedback control system.

Usage: Commonly used in projects that require precise control of the rotational position, such as robotics, automation, and remote-controlled vehicles.

6- DC Motor:

Description: A DC motor is an electro-mechanical device converting electrical energy to mechanical motion based on the Lorentz force principle. It comprises a stator (stationary, with field windings) and a rotor (rotating armature, with coil windings), causing motor shaft rotation through magnetic field interaction. Speed and direction depend on applied voltage polarity and magnitude.

Usage: DC motors, valued for simplicity, reliability, and ease of control, find extensive application in robotics, RC vehicles, automation, and various industrial processes. Their ability to be used for tasks ranging from precise robotic positioning to powering electric vehicle wheels stems from the controlled speed and direction. Further, their versatile application is enhanced by easy interfacing with electronic circuits like motor drivers such as the L293D or H-bridges, making them integral in systems requiring controlled and dynamic motion.

7- Wheels:

Description: Wheels are circular components designed to rotate around an axle, facilitating the movement of a vehicle or object. In the context of a car, wheels provide traction and support, enabling the vehicle to roll smoothly. They come in various sizes and materials, and their design influences factors such as speed, stability, and manoeuvrability.

Usage: Wheels are fundamental in automotive applications, including cars, bicycles, and robotics. In the specific context of a car moving through a line and avoiding obstacles, the type and arrangement of wheels play a crucial role in determining the vehicle's ability to navigate and maintain stability during movement.

8- Battery (3.7V):

Description: A battery is a device that stores electrical energy and releases it as needed through chemical reactions.

Usage: Provides power to electronic devices and circuits. In the context of the mentioned components, it supplies the energy needed to operate motors, sensors, and other electronic components in a circuit.

9- Battery Holder:

Description: A battery holder is a device designed to secure and organize one or more batteries in a specific arrangement. Typically made of plastic or insulating materials, it ensures a secure fit for batteries and provides electrical connections for powering electronic devices.

Usage: Crucial in electrical systems, battery holders offer a convenient and organized way to contain and connect batteries. In applications like autonomous cars navigating obstacles, a battery holder acts as the power source for the entire system, ensuring a stable supply of electrical energy to drive motors, sensors, and other components. Its design facilitates easy battery replacement, contributing to overall system efficiency and functionality.

10- Wires :

Description: Wires are conductive strands, often made of copper or aluminium, that transmit electrical signals or power between electronic components. They are crucial conduits for the flow of electricity in circuits, connecting various elements and facilitating communication within a system.

Usage: In automotive systems, wires play an integral role, connecting sensors, actuators, and electronic modules to the control unit. This connectivity enables the exchange of information and power, especially in applications like obstacle detection where wires link sensors to the control circuitry.

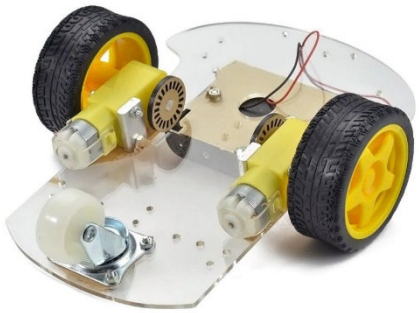
11- Switch:

Description: A switch is a mechanical or electronic device that opens or closes a circuit, interrupting or allowing the flow of electrical current. It provides a means to control the operation of electrical devices by toggling between on and off states.

Usage: Switches are used in various applications within a car's electrical system. In the context of the described project, a switch might be employed to activate or deactivate specific functionalities, such as turning the entire system on or off, initiating obstacle avoidance mode, or controlling other aspects of the car's behaviour.

12- Car Chassis:

Description: The car chassis is the framework that provides structural support for the various components of a vehicle, including the engine, suspension, wheels, and other systems. It forms the foundation upon which the entire car is built.

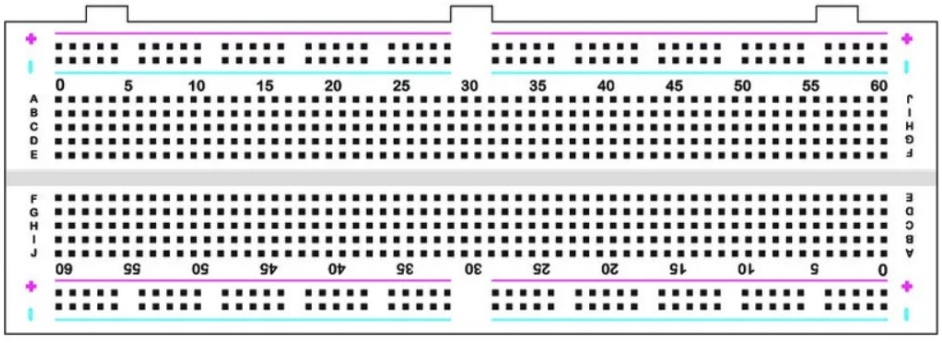


Usage: The car chassis is critical for the structural integrity and overall stability of the vehicle. It determines the car's shape, size, and weight distribution, influencing factors such as handling, performance, and safety. In the context of a car moving along a line and avoiding obstacles, the chassis supports and houses the motor, wheels, sensors, and other essential components, contributing to the overall functionality and performance of the autonomous system.

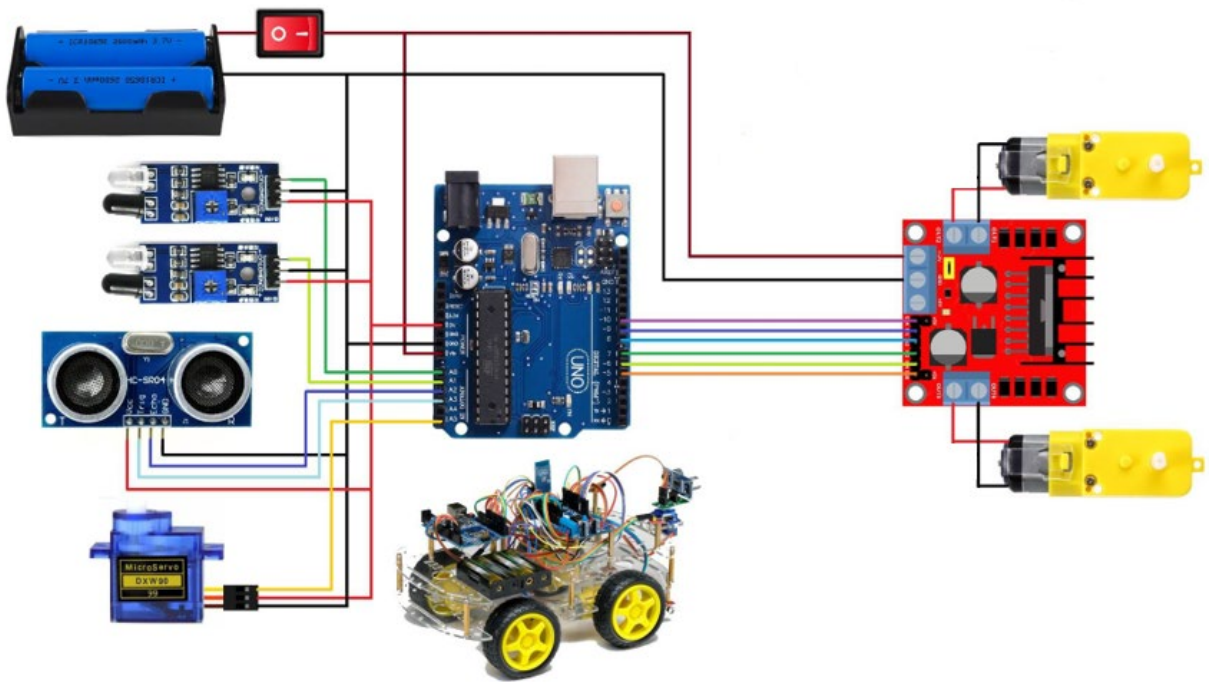
13- Breadboard:

Description: A breadboard is a crucial tool in electronics prototyping, offering a solder-free platform for constructing and testing circuits. It features a rectangular board with a grid of holes and underlying metal strips that connect specific holes in a predetermined pattern. These holes facilitate the easy insertion and connection of electronic components, enabling the creation and modification of circuits during design and testing phases.

Usage: Breadboards find extensive use in electronic projects, particularly in the initial development stages. They function as a temporary assembly platform by allowing components to be plugged into the board and connected through internal metal strips. This facilitates experimentation with various circuit configurations, testing component functionality, and troubleshooting without the permanence of soldered connections. Breadboards are indispensable for students, hobbyists, and professionals, providing a versatile and reusable platform for exploring and refining electronic designs.



3- Circuit Design:



4- Implementation:

```
// L298 (H-Driver) Pins
#define enA 5  // Enable1
#define in1 6  // Motor1
#define in2 7  // Motor1
#define in3 8  // Motor2
#define in4 9  // Motor2
#define enB 10 // Enable2

// IR Sensors Pins
#define L_S A0 // Left
#define R_S A1 // Right

// Ultrasonic Pins
#define echo A2    // Echo
#define trigger A3 // Trigger
```

```
// Servo Motor Pins
#define servo A5 // (Ultrasonic Holder)

int Set=20; // Ultra Sonic Stopping Distance
int distance_L, distance_F, distance_R;  // Variables for
measured distances Left, Forward, Right

void setup(){ // Setup code, which runs once

Serial.begin(9600); // Start serial communication at 9600bps

pinMode(R_S, INPUT); // Declare Right IR sensor as input
pinMode(L_S, INPUT); // Declare Left  IR sensor as input

pinMode(echo, INPUT );    // Declare Ultrasonic sensor
Echo    pin as input
pinMode(trigger, OUTPUT); // Declare Ultrasonic sensor
Trigger pin as Output

pinMode(enA, OUTPUT); // Declare as output for L298 Pin enA
pinMode(in1, OUTPUT); // Declare as output for L298 Pin in1
pinMode(in2, OUTPUT); // Declare as output for L298 Pin in2
pinMode(in3, OUTPUT); // Declare as output for L298 Pin
in3
pinMode(in4, OUTPUT); // Declare as output for L298 Pin in4
pinMode(enB, OUTPUT); // Declare as output for L298 Pin enB

analogWrite(enA, 210); // Write The Duty Cycle 0 to 255
enAble Pin A for Motor1 Speed
analogWrite(enB, 210); // Write The Duty Cycle 0 to 255
enAble Pin B for Motor2 Speed

pinMode(servo, OUTPUT); // Declare Servo Motor pin as output

for (int angle = 70; angle <= 140; angle += 5) { // Sweep
the servo from 70 to 140 degrees in 5-degree increments
  servoPulse(servo, angle);}   // Call the function to set
the servo to the current angle
```

```
for (int angle = 140; angle >= 0; angle -= 5) { // Sweep the
servo back from 140 to 0 degrees in 5-degree decrements
  servoPulse(servo, angle);}   // Call the function to set
the servo to the current angle

for (int angle = 0; angle <= 70; angle += 5) { // Sweep the
servo from 0 to 70 degrees in 5-degree increments
  servoPulse(servo, angle);}   // Call the function to set
the servo to the current angle

distance_F = Ultrasonic_read(); // Read the distance from
the ultrasonic sensor
delay(500); // Introduce a delay of 500 milliseconds
} // End of Setup code


void loop(){  // Main Loop Function
distance_F = Ultrasonic_read();   // Read distance from
Ultrasonic sensor
Serial.print("D F=");Serial.println(distance_F);   // Print
distance value to Serial monitor
// If both right and left sensors are on white color then it
will call forward function
 if((digitalRead(R_S) == 0)&&(digitalRead(L_S) == 0)){ //
Check if both right and left sensors detect an obstacle
    // Check if the front distance is greater than a certain
threshold ('Set')
  if(distance_F > Set){forward();}     // If front distance
is sufficient, move forward
                  else{Check_side();}  // If front distance
is not sufficient, check the side for further action
 }

// Else, If Right Sensor is Black and Left Sensor is White
then it will call turn Right function
else if((digitalRead(R_S) == 1)&&(digitalRead(L_S) ==
0)){turnRight();}
```

```
// Else, If Right Sensor is White and Left Sensor is Black
then it will call turn Left function
else if((digitalRead(R_S) == 0)&&(digitalRead(L_S) ==
1)){turnLeft();}

delay(10);  // Introduce a delay of 10 milliseconds
}

void servoPulse (int pin, int angle){ // Function to
generate a servo pulse with specified angle
int pwm = (angle*11) + 500;      // Convert angle to
microseconds
 digitalWrite(pin, HIGH);   // Activate the servo
 delayMicroseconds(pwm);    // Introduce a microsecond delay
based on the 'pwm' variable
 digitalWrite(pin, LOW);    // Deactivate the servo
 delay(50);   // Introduce a delay for the servo refresh
cycle

}


long Ultrasonic_read(){ // Function to read distance from
Ultrasonic sensor
  // Trigger a pulse to initiate ultrasonic sensor
measurement
  digitalWrite(trigger, LOW); // Set trigger pin to LOW
  delayMicroseconds(2); // Wait for 2 microseconds
  digitalWrite(trigger, HIGH); // Set trigger pin to HIGH to
start the ultrasonic pulse
  delayMicroseconds(10);  // Keep the trigger pin HIGH for
10 microseconds
  // Measure the pulse duration and convert it to distance
in centimeters
  long time = pulseIn (echo, HIGH); // Measure the duration
of the pulse from the echo pin
  return time / 29 / 2; // Convert pulse duration to
distance in centimeters
}
```

```
void compareDistance(){ // Function to compare distances and
adjust the robot's path
    if(distance_L > distance_R){   // If left distance is
more than right distance, perform a left turn; otherwise,
perform a right turn
                      // Execute left turn sequence
        turnLeft();     // Turn the robot left
        delay(350);     // Wait for 350 milliseconds
        forward();      // Move the robot forward
        delay(750);     // Wait for 750 milliseconds
        turnRight();    // Turn the robot right
        delay(350);     // Wait for 350 milliseconds
        forward();      // Move the robot forward
        delay(750);     // Wait for 750 milliseconds
        turnRight();    // Turn the robot right
        delay(350);     // Wait for 350 milliseconds
    } else {          // Execute right turn sequence
        turnRight();    // Turn the robot right
        delay(350);     // Wait for 350 milliseconds
        forward();      // Move the robot forward
        delay(750);     // Wait for 750 milliseconds
        turnLeft();     // Turn the robot left
        delay(350);     // Wait for 350 milliseconds
        forward();      // Move the robot forward
        delay(750);     // Wait for 750 milliseconds
        turnLeft();     // Turn the robot left
        delay(350);     // Wait for 350 milliseconds
    }
}

void Check_side(){ // Function to check distances on the
robot's sides
    Stop();   // Stop the robot
    delay(100); // Introduce a delay
 for (int angle = 70; angle <= 140; angle += 5)  {   //
Sweep the servo from 70 to 140 degrees to check the left
side
```

```
   servoPulse(servo, angle);  } // Call the function to set
the servo to the current angle
    delay(300);   // Introduce a delay
    distance_L = Ultrasonic_read(); // Read the distance on
the left side
    Serial.print("D L=");Serial.println(distance_L); //
Print the distance value for the Left sensor
    delay(100);   // Introduce a delay
  for (int angle = 140; angle >= 0; angle -= 5)  {   //
Sweep the servo from 140 to 0 degrees to check the right
side
   servoPulse(servo, angle);  } // Call the function to set
the servo to the current angle
    delay(500);  // Introduce a delay
    distance_R = Ultrasonic_read(); // Read the distance on
the right side
    Serial.print("D R=");Serial.println(distance_R); //
Print the distance value for the Right sensor
    delay(100);   // Introduce a delay
 for (int angle = 0; angle <= 70; angle += 5)  {   // Sweep
the servo from 0 to 70 degrees to reset its position
   servoPulse(servo, angle);  } // Call the function to set
the servo to the current angle
    delay(300);   // Introduce a delay
    compareDistance();   // Compare distances and adjust the
robot's path
}

void forward(){  // Forward Function
digitalWrite(in1, LOW);  // Left Motor backward Pin
digitalWrite(in2, HIGH); // Left Motor forward Pin
digitalWrite(in3, HIGH); // Right Motor forward Pin
digitalWrite(in4, LOW);  // Right Motor backward Pin
}

void backward(){ // Backward Function
digitalWrite(in1, HIGH); // Left Motor backward Pin
digitalWrite(in2, LOW);  // Left Motor forward Pin
digitalWrite(in3, LOW);  // Right Motor forward Pin
```

```
digitalWrite(in4, HIGH); // Right Motor backward Pin
}

void turnRight(){ // Turn Right Function
digitalWrite(in1, LOW);  // Left Motor backward Pin
digitalWrite(in2, HIGH); // Left Motor forward Pin
digitalWrite(in3, LOW);  // Right Motor forward Pin
digitalWrite(in4, HIGH); // Right Motor backward Pin
}

void turnLeft(){ // Turn Left Function
digitalWrite(in1, HIGH); // Left Motor backward Pin
digitalWrite(in2, LOW);  // Left Motor forward Pin
digitalWrite(in3, HIGH); // Right Motor forward Pin
digitalWrite(in4, LOW);  // Right Motor backward Pin
}

void Stop(){ // Stop Function
digitalWrite(in1, LOW); // Left Motor backward Pin
digitalWrite(in2, LOW); // Left Motor forward Pin
digitalWrite(in3, LOW); // Right Motor forward Pin
digitalWrite(in4, LOW); // Right Motor backward Pin
}
```

5- Conclusion:

The Arduino-based autonomous car project presented integrates various components, including an L298 H-bridge motor driver, infrared sensors, ultrasonic sensors, and a servo motor, to achieve intelligent navigation. The vehicle autonomously follows a path while avoiding obstacles, showcasing the practical implementation of electronic and robotic principles. Careful calibration and systematic control logic in the code enable the car to make informed decisions based on sensor inputs. Emphasizing the significance of sensor fusion for effective obstacle avoidance, this project serves as a foundation for advancements in autonomous vehicle technology and robotics by creating a dynamic system responsive to its environment.