International Islamic University Chittagong

Combines Quality with Morality

| Project Report |

Project Title :

Obstacle - Avoiding ROBOT

Course Code: CSE-2324
Course Title: Digital Logic Design Lab
Section: 3BF Semester: Autumn- 2024
Submission Date: 22/12/2024

<u>Submitted To:</u>

Mrs. Taniya Sultana
Lecturer
Department: Computer Science and Engineering
International Islamic University Chittagong

<u>Team Details:</u>

Shakila Jahan Saima **(C233442)**
Sadia Zannat **(C233443)**
Tahasina Tasnim Afra **(C233456)**
Maimuna Tabasum **(C233460)**
Tasmia Tabassum Kotha **(C233461)**

Report Overview:

Abstract:

The obstacle-avoiding robot is an autonomous system designed to navigate its environment by detecting and avoiding obstacles in real time. This project combines principles of robotics, sensor integration, and programming to achieve self-sufficient motion control. The robot is equipped with ultrasonic sensors for obstacle detection and uses a microcontroller to process data and make decisions on navigation. The control algorithm enables the robot to maneuver effectively by identifying obstacles and adjusting its path accordingly.

The project emphasizes modularity in design, ensuring easy scalability and integration of additional features like line following or remote control. Testing and iterative refinement have been conducted to ensure reliable operation under varying conditions, demonstrating the robot's capability to adapt dynamically to its environment.

Introduction:

This study is about designing and implementing an autonomous mobile robot capable of navigating dynamic environments by avoiding moving obstacles. The robot employs stereo vision (using two webcams as sensors) to perceive its surroundings and detect obstacles, particularly pedestrians. Pedestrians are chosen as obstacles to simulate a realistic working environment where the robot interacts with humans.

Our Objective:

The primary objective is to develop a moving **obstacle avoidance technique** by integrating the **Neuro-Fuzzy algorithm**[1] as the robot's control system. This algorithm allows the robot to make flexible decisions in real-time, enabling it to adjust its path dynamically while maintaining progress toward its target destination.

The robot is built as a four-wheeled omnidirectional platform, providing enhanced mobility and flexibility for navigating complex environments. The behavior of the robot is programmed to:

1. Identify the target destination.
2. Detect and track moving obstacles.
3. Make real-time decisions to avoid obstacles while continuing toward the goal.

By utilizing stereo vision for environmental sensing and the Neuro-Fuzzy algorithm for intelligent decision-making, the robot demonstrates adaptability in dealing with dynamic, unpredictable surroundings, making it suitable for real-world applications where mobility and obstacle avoidance are crucial.

[1] A hybrid system that combines the learning capabilities of neural networks with the reasoning approach of fuzzy logic. It is used for solving problems where data is uncertain or imprecise.

Literature Survey:

Obstacle-avoiding robots have been a focus of research in robotics due to their applications in automation, autonomous vehicles, and service robots. Researchers have explored techniques to enhance their accuracy, efficiency, and adaptability.

1. **Sensor Technology**
   Obstacle detection relies heavily on sensor technology. Ultrasonic sensors are widely used for their accuracy and cost-effectiveness in short-range obstacle detection.

2. **Neuro Fuzzy Algorithms**
   The Neuro-Fuzzy algorithm helps the robot by interpreting data from stereo vision sensors, by recognizing moving obstacles like pedestrians and make real-time decisions to avoid obstacles while navigating toward a target destination. This combination ensures the robot operates smoothly and efficiently in dynamic, real-world environments.

3. **Microcontroller Integration**
   Microcontrollers like Arduino UNO are widely adopted for their affordability and ease of programming. Their seamless integration with sensors and motors facilitates real-time obstacle avoidance.

4. **Applications and Enhancements**
   Obstacle-avoiding robots are used in various domains, including warehouse automation and healthcare assistance. Enhancements such as multi-sensor fusion, energy-efficient motors, and wireless communication modules have expanded their functionality.

5. **Challenges and Limitations**
   Challenges such as sensor noise, power consumption, and navigation in dynamic environments persist. Research is ongoing to address these issues through hardware and software innovations.

In summary, advancements in sensor technology, control algorithms, and microcontroller applications have propelled the development of obstacle-avoiding robots, though challenges remain for broader real-world applications.

Hardware:

The hardwares used for this project:

- Microcontroller Arduino Uno
- SG90 HC-SR04 Ultrasonic Sensor
- L293D Motor Driver Shield for Arduino
- Servo motor
- Robot car chassis set (acrylic base, 4 mini gear motors, 4 wheels, 4 fasteners)
- Jumper wires (Female to male)
- 2 Lithium-ion batteries (7.4v)
- Soldering iron

Microcontroller Arduino Uno:

Arduino Uno is a **microcontroller board**, developed by Arduino.cc, based on the Atmega328 microcontroller and is marked as the first Arduino board developed(UNO means "one" in Italian). The software used for writing, compiling & uploading code to Arduino boards is called **Arduino IDE** (Integrated Development Environment).

It is an **open-source platform** where anyone can modify and optimize the board based on the number of instructions and tasks they want to achieve.

The Arduino Board comes with **14 digital pins (D0 to D13)** and **6 analog pins (A0 to A5)**. ON-chip ADC is used to sample these pins. A 16 MHz frequency **crystal oscillator** is equipped on the board.It also has **1 Reset Pin**, which is used to reset the board programmatically. In order to reset the board, we need to make this pin LOW. Arduino UNO has a **maximum current rating of 40mA**, so the load shouldn't exceed this current rating or you may harm the board.

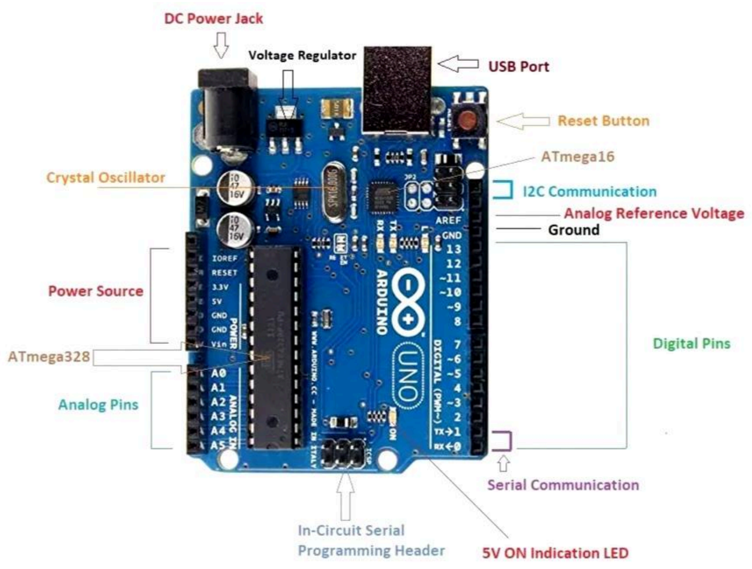


Figure- Labeled Image of Arduino UNO

SG90 HC-SR04 Ultrasonic Sensor:

The HC-SR04 is a type of ultrasonic sensor which **uses sonar to find out the distance of the object from the sensor**. It provides an outstanding range of non-contact detection with high accuracy & stable readings. It includes two modules like ultrasonic transmitter & receiver. The Ultrasonic sensors HC-SR04 includes 4 pins -

- VCC pin- needs to be connected to **VCC** (5V)
- GND pin- needs to be connected to **GND** (0V)
- TRIG pin- this pin receives the control signal (pulse) from Arduino.
- ECHO pin- this pin sends a signal (pulse) to Arduino. Arduino measures the duration of pulse to calculate distance.

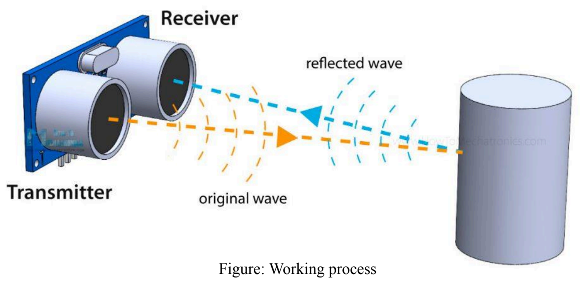
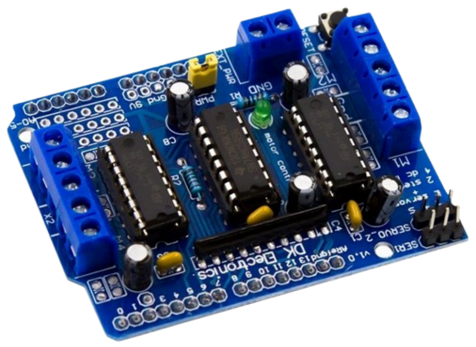


Figure: Working process

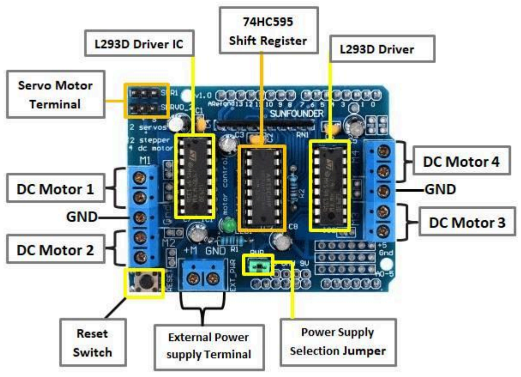
L293D Motor Driver Shield For Arduino:

The L293D is a quad H-bridge driver IC, designed to control two DC motors simultaneously. It offers a convenient way to drive the motors in both forward and reverse directions, as well as adjust their speed. Key features of the L293D include:



- **Quad H-bridge:** Provides four independent H-bridge circuits for controlling two motors and each H-bridge can deliver up to 0.6A to the motor.
- **Bidirectional control:** Allows for both forward and reverse rotation of the motors.
- **PWM control:** Enables precise speed regulation using pulse-width modulation[2].
- **Current limiting:** Protects the motors and driver IC from excessive current draw.
- **Thermal shutdown:** Ensures safe operation by shutting down the driver if it overheats.

The shield also comes with a **74HC595 shift register**[3] that extends 4 digital pins of the Arduino to the 8 direction control pins of two L293D chips. Before using the L293D motor driver shield with Arduino IDE, we need to install the **AFMotor.h** library which contains the commands to control DC, stepper and servo motors.



This motor shield can drive DC motors having voltages between 4.5 to 25V. The motor is connected to any of M1, M2, M3 or M4 motor terminals. In this case, Arduino pin11 for M1, pin3 for M2, pin5 for M3, pin6 for M4 and pins 4, 7, 8 and 12 are all in use.

Servo Motor:

Servo motor is a component that can rotate its handle from 0 to 180 degree angle. It is used to control the angular position of the object. The servo motor includes 3 pins- (**VCC**, **GND**, **Signal pin**) . After connecting VCC pin and GND pin to 5V and 0V respectively, we can control the servo motor by generating proper PWM (pulse-width modulation) signal to signal pin.

[2] **Pulse-Width Modulation (PWM)**: A technique used to control the amount of power delivered to an electronic device by varying the width of the voltage pulses in a periodic signal.

[3] **Shift Register:** A digital circuit used to store and move data bits sequentially, either to the left or right, often used for data storage or transfer in microcontrollers.

Robot Car Chassis Set:

This is a set of **four-wheeled robot chassis, 4 DC motors, 4 wheels and 4 fasteners .**

- **DC motors-** DC motors use direct currents with the current flowing in a single direction and convert the energy into mechanical energy.
- **Wheels-** Rubber wheels of diameter 65mm and width 27mm.
- **Acrylic Base-** The transparent base holds the car body including microcontroller and motor shield.

Jumper wires:

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other. Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. **Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. In this project we used male to female jumper wires.**

Lithium-ion Battery:

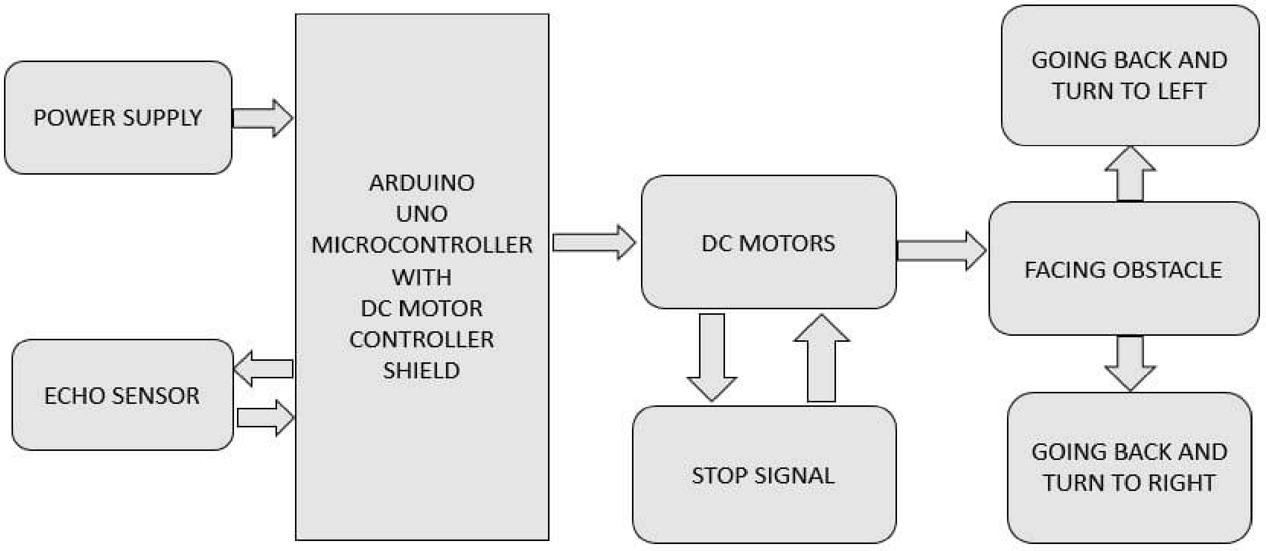
A **lithium-ion battery**, also known as the Li-ion battery, is a type of secondary (rechargeable) battery composed of cells in which lithium ions move from the anode through an electrolyte to the cathode during discharge and back when charging.

Soldering Iron:

A **soldering iron** is a hand tool used in soldering. It supplies heat to melt solder so that it can flow into the joint between two workpieces.A soldering iron is composed of a heated metal tip (the *bit*) and an insulated handle.To solder jumper wires for the project, we followed these steps:

1. Stripping the wire.
2. Tinned the wire and soldering the jumper to the board. Then poked the tinned jumper end through to the board.
3. Clipped the excess.

Block Diagram:


POWER SUPPLY
ARDUINO UNO MICROCONTROLLER WITH DC MOTOR CONTROLLER SHIELD
ECHO SENSOR
DC MOTORS
STOP SIGNAL
GOING BACK AND TURN TO LEFT
FACING OBSTACLE
GOING BACK AND TURN TO RIGHT

Working Principle:

An obstacle avoiding robot uses a proximity sensor module, besides other parts. In this case, this robot uses a proximity sensor developed by ourselves. The robot is controlled by a program that is embedded into a microcontroller (arduino uno). The logics produced by the microcontroller are further processed by an interface module, in this case, also developed by ourselves. The interface module translates microcontroller's logics into voltage and current that can practically drive the four motors. Here is a short description of functionalities of the components which made the implementation possible-

1. **Ultrasonic Sensor**:
    - Sends a sonic burst using the transmitter module, consisting of 8 pulses of 40 kHz, when triggered.
    - Measures the time taken for the sound wave to return after hitting a surface.
    - Continuously calculates the distance between the robot and obstacles.
    - Sends distance data to the Arduino for processing.
2. **Arduino**:
    - Processes the distance information received from the ultrasonic sensor.
    - Decides actions based on the distance:
        - If the distance is less than 25 cm, it stops the robot.

        - Commands the servo motor and ultrasonic sensor to scan left and right for better clearance.
        - Based on the scan results, decides whether the robot should turn left or right.
        - Controls motor directions to avoid obstacles.
3. **Servo Motor**:
    - Rotates the ultrasonic sensor to scan the left and right sides.
    - Helps determine the side with the maximum clearance.
4. **Wheel Motors**:
    - Drive the robot forward under normal conditions.
    - Reverse and turn left or right based on Arduino's decisions:
        - If the left side has more clearance, the left wheel motor reverses for a left turn.
        - If the right side has more clearance, the right wheel motor reverses for a right turn.

This process operates in a continuous loop, allowing the robot to move and avoid obstacles seamlessly.

Circuit Diagram:

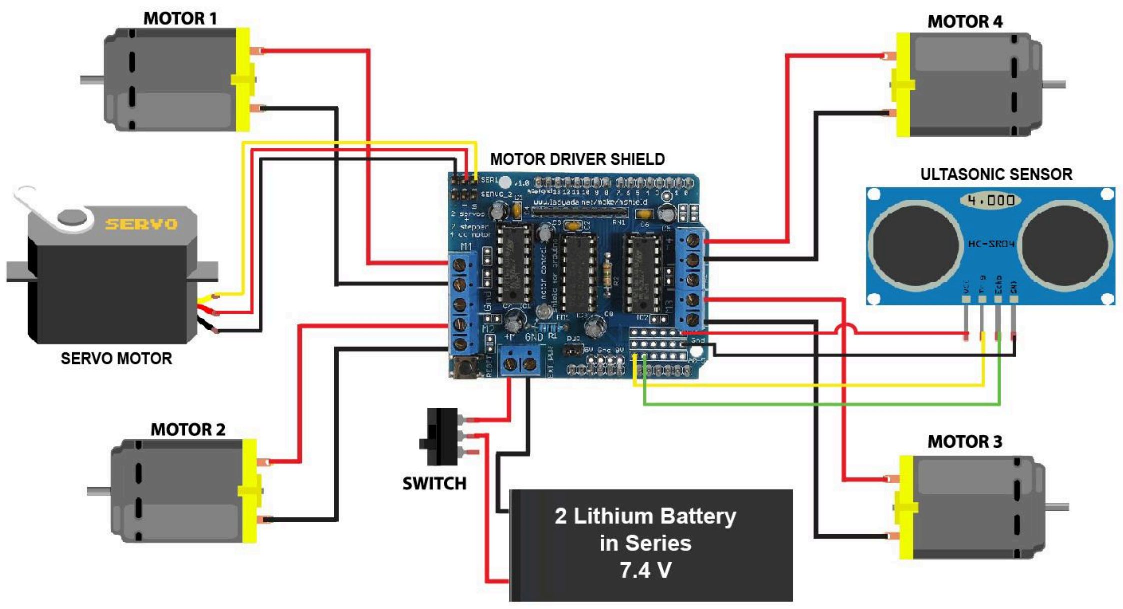


Figure: Circuit diagram of the obstacle avoiding robot.

Applications:

Here are **five different uses** of an obstacle-avoiding robot:

1. **Autonomous Navigation in Warehouses**:
   - These robots can transport goods or materials across warehouses while avoiding obstacles such as shelves, equipment, or workers.
   - They streamline inventory management and reduce the risk of human errors.

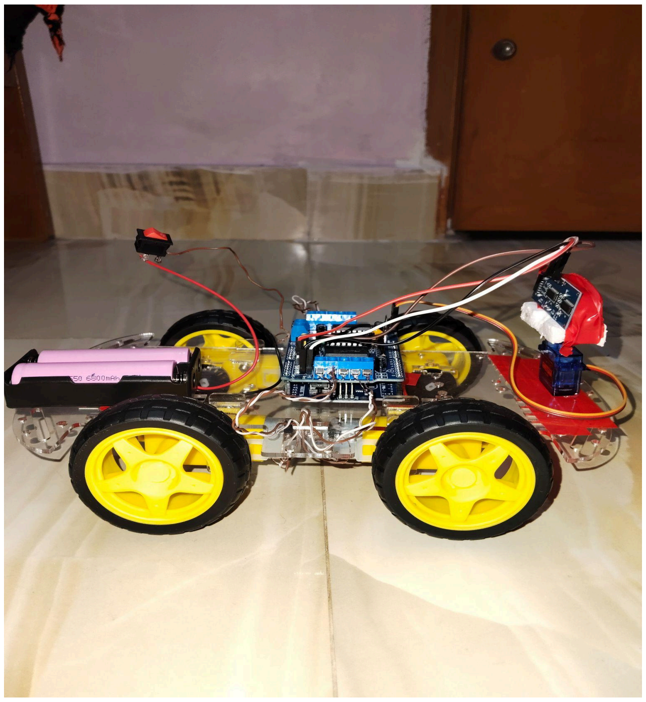
2. **Search and Rescue Operations**:
   - Obstacle-avoiding robots can navigate through rubble or dangerous environments (like collapsed buildings) to search for survivors.
   - They help minimize risks to human rescuers by reaching inaccessible or hazardous areas.

3. **Home Cleaning Robots**:
   - Used in vacuum cleaners (e.g., robotic vacuums like Roomba) to avoid furniture, walls, and other obstacles while cleaning floors.
   - They improve efficiency and automate household cleaning tasks.

4. **Agricultural Applications**:
   - Robots equipped with obstacle-avoidance systems can autonomously navigate fields for tasks like planting, irrigation, or pest control while avoiding crops and obstacles.
   - This ensures precision farming and reduces labor dependency.

5. **Military and Defense**:
   - Obstacle-avoiding robots can be deployed for reconnaissance or surveillance in dangerous areas (e.g., minefields or enemy zones).
   - They help gather intelligence while avoiding obstacles and minimizing human exposure to risks.

Results:



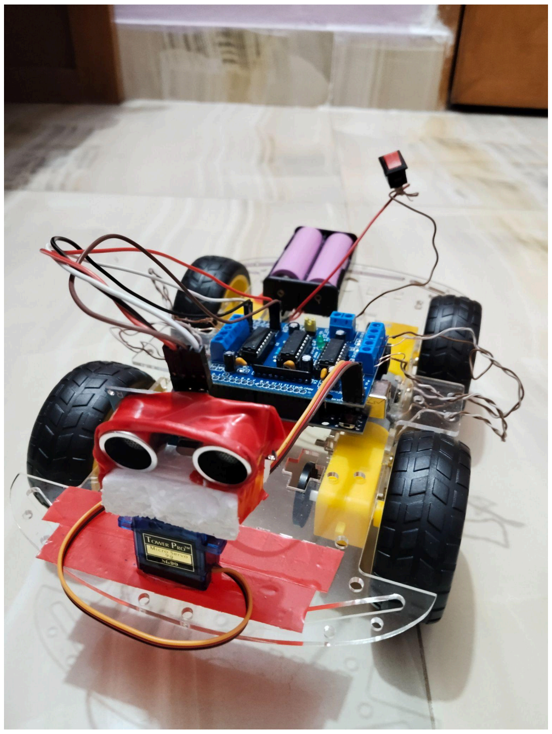


Figure: The complete structural interface of the obstacle avoiding robot.

Program Code:

In order to program our idea we need to install the arduino IDE from the website of arduino.cc. We download the Adafruit Motor Sheild library extension and include the library AFmotor.h and Servo.h in arduino folder for smooth operation. Here is the code programmed for obstacle avoidance and motor operation-

```
#include <AFMotor.h>

#include <Servo.h>



Servo servo;



AF_DCMotor LF(1);

AF_DCMotor LB(2);

AF_DCMotor RF(3);

AF_DCMotor RB(4);



#define trigPin  A0

#define echoPin  A1



void setup() {

  servo.attach(10);  // Attach the servo

  pinMode(trigPin, OUTPUT);  // Declare "trigPin" as "Output Pin".

  pinMode(echoPin, INPUT);   // Declare "echoPin" as "Input Pin".
  Serial.begin(9600);      // Start serial communication
}

void loop() {
  float distance = search();  // Measure distance
  Serial.print("Distance = ");
  Serial.println(distance);

  // Adjust to avoid zero distance readings
  if (distance > 0) {
    if (distance <= 25) {
      RobotStop();
      delay(200);

      if (distance <= 30) {  // If obstacle found within 30 cm
        RobotStop();  // Robot stops
        delay(500);
        servo.write(5);  // Move servo to 5 degrees
        delay(500);

        float RightDistance = search();  // Measure right distance
        delay(100);
        servo.write(180);  // Move servo to 180 degrees
```

```
delay(500);


float LeftDistance = search();  // Measure left distance

delay(100);

servo.write(90);  // Reset servo to center position

delay(300);


// Decide which direction to turn based on distances

if (LeftDistance > RightDistance) {

   Motor_Speed(140);

   RobotBackward();

   delay(300);

   RobotStop();

   delay(100);

   RobotLeft();

   delay(500);

   RobotStop();

   delay(100);

} else {

   Motor_Speed(140);

   RobotBackward();

   delay(300);

   RobotStop();
```

```
          delay(100);

          RobotRight();

          delay(500);

          RobotStop();

          delay(100);

        }

      }

    } else if (distance <= 60) {

      Motor_Speed(170);

      RobotForward();

    } else {

      Motor_Speed(200);

      RobotForward();

    }

  } else {

    Serial.println("Invalid reading");

    RobotStop(); // Stop the robot if distance is invalid

  }


  delay(100);  // Add a small delay before the next loop iteration

}


float search() {
  float duration = 0.00;
  float CM = 0.00;

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH, 30000); // Timeout after 30ms
  CM = (duration / 58.82);  // Convert time to distance in cm

  // Return 0 if distance is invalid
  if (CM > 0 && CM < 400) {  // Range check
    return CM;
  } else {
    return 0;  // Invalid distance reading
  }
}

void RobotForward() {
  LF.run(FORWARD);
  RF.run(FORWARD);
```

```
   LB.run(FORWARD);
   RB.run(FORWARD);
}

void RobotBackward() {
   LF.run(BACKWARD);
   LB.run(BACKWARD);
   RF.run(BACKWARD);
   RB.run(BACKWARD);
}

void RobotLeft() {
   LF.run(BACKWARD);
   LB.run(BACKWARD);
   RF.run(FORWARD);
   RB.run(FORWARD);
}

void RobotRight() {
   LF.run(FORWARD);
   LB.run(FORWARD);
   RF.run(BACKWARD);
   RB.run(BACKWARD);
```

```
}


void RobotStop() {
   LF.run(RELEASE);
   LB.run(RELEASE);
   RF.run(RELEASE);
   RB.run(RELEASE);
}


void Motor_Speed(int mtrspd) {
   LF.setSpeed(mtrspd);
   LB.setSpeed(mtrspd);
   RF.setSpeed(mtrspd);
   RB.setSpeed(mtrspd);
}
```

Conclusion:

The designed small avoiding obstacle robot has been developed successfully in this project and operated nearly as expected. This project demonstrates the seamless interaction between hardware and software to achieve autonomous navigation. The robot's ability to detect and avoid obstacles ensures efficient and safe movement, making it ideal for various applications, from automation in industries and homes to exploration in hazardous environments. This project not only showcases the potential of robotics in solving real-world problems but also serves as a foundation for further advancements in autonomous systems and intelligent robotics.

References:

The necessary articles and youtube links that had been followed for guidance through the whole implementation process of the project -

YouTube:

https://youtu.be/BhrrNtihIe8?si=OicvqSuaKZNWgY6q

https://youtu.be/Uio4A9A9S4Q?si=CZHNSoXvaJOX7kJD

https://youtu.be/3awCkLS7gHI?si=vyBdyY9jpqLTii8G

https://youtube.com/shorts/lYkJzfeTbWs?si=thE1zm4BshaUGAa8

Websites:

https://projecthub.arduino.cc/parth2008/obstacle-avoiding-car-d7450d

https://en.m.wikipedia.org/wiki/Obstacle_avoidance#:~:text=Obstacle%20avoidance%2C%20in%20robotics%2C%20is,to%20interact%20with%20its%20environment.

https://irispublishers.com/ijasc/fulltext/the-several-uses-for-obstacle-avoidance-robots.ID.000503.php

https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-uno.html

Costing:

Components	**Pricing (BDT)**
Arduino Uno R3	800
L293D Arduino Motor Driver Shield	200
SG90 HC-SR04 Ultrasonic Sensor	90
Servo Motor	150
Robot Car Chassis Kit	900
Jumper Wires & Connecting Wires	100
Battery Holder	40
Lithium-ion Batteries & Charger	330
Soldering iron	180
Total	2790/-