Course No : CSE 3104

Course Title : Peripherals and Interfacing Laboratory

Project Title : Obstacle Avoiding Car

Submitted By

Imtiag Hossin Joy

1907092

Irhamul Islam

1907093

Objectives

- Design and build a robot car that can avoid obstacles.
- Evaluate the accuracy and effectiveness of the obstacle recognition system
- Test the car's ability to navigate and interact with its surroundings.
- Compare different obstacle avoiding algorithms in terms of accuracy.
- Identify limitations and potential improvements for the obstacle avoiding system.
- Provide recommendations for enhancing the obstacle avoiding system and the car's functionality based on the experiment's results.

Introduction

An obstacle-avoiding car is a robotic vehicle designed to navigate its environment while detecting and avoiding obstacles in its path. The goal is to create a vehicle that can autonomously move around and avoid collisions with objects or barriers, ensuring safe and efficient navigation.

The car typically incorporates various sensors, such as ultrasonic sensors, which emit signals and measure the time it takes for the signals to bounce back. By analyzing the return signals, the car can determine the distance to nearby objects.

Using this distance information, the car employs an algorithm to make decisions on how to maneuver and avoid obstacles. The algorithm might involve techniques such as path planning, collision avoidance, or reactive control. The car can adjust its direction, speed, or both to navigate around obstacles and continue its path.

The obstacle-avoiding car project combines elements of electronics, programming, and mechanical design. It allows you to learn and apply concepts from these domains while developing a system that demonstrates autonomous behavior in a mobile robotic platform.

Theory

Arduino Uno is based on the ATmega328P microcontroller and offers a wide range of input/output pins and features, making it suitable for beginners as well as experienced enthusiasts.

The Arduino Uno board comes with 14 digital input/output pins, where 6 can be used as
PWM (Pulse Width Modulation) outputs. It also has 6 analog input pins, a USB connection for programming and communication with a computer, and a power jack for external power supply. Additionally, it has a reset button for restarting the program execution.

One of the key advantages of Arduino Uno is its simplicity and ease of use. It has a user-friendly programming environment that allows you to write and upload code to the board using the Arduino Integrated Development Environment (IDE). The IDE supports a C/C++-based programming language, making it accessible to beginners and experienced programmers alike.

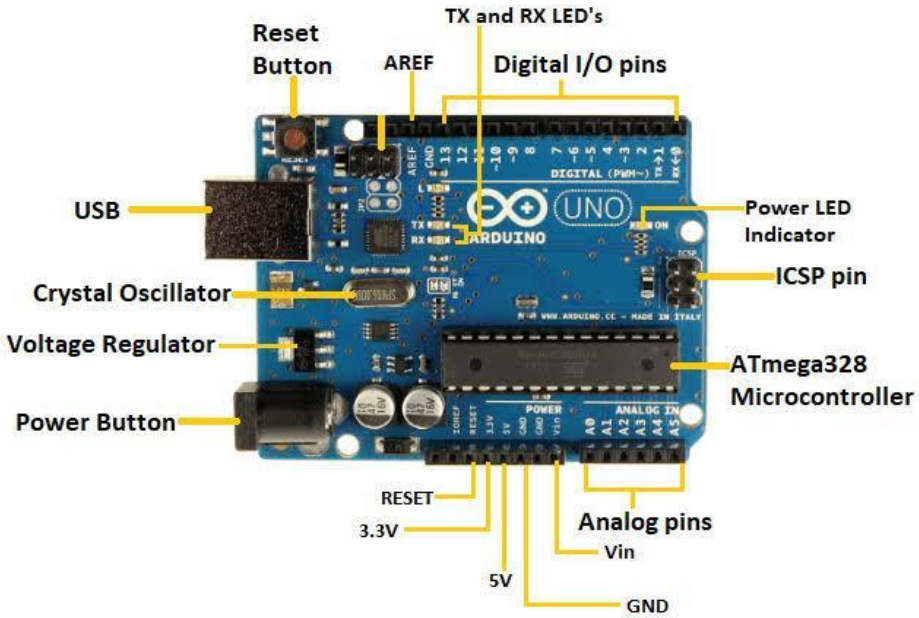


Figure 1: Arduino Uno

The L298 motor driver is a integrated circuit (IC) commonly used to control and drive DC motors or stepper motors in electronic projects. It provides a convenient and efficient solution for motor control, making it widely used in robotics, automation, and other applications that require precise motor control.

The L298 motor driver IC can handle two motors independently or a single motor with higher current requirements. It can control the direction of rotation (forward and reverse) as well as the speed of the motor. The IC uses H-bridge configuration, allowing it to drive the motor in both directions by controlling the polarity of the applied voltage.

One of the key features of the L298 motor driver is its ability to handle high currents. It can provide a maximum output current of around 2A per channel (depending on the specific variant) and a peak current of up to 3A. This capability makes it suitable for driving motors with higher power requirements.

The L298 motor driver IC requires external power supply connections, typically from a separate power source. It also requires control signals from a microcontroller or other digital devices to operate. These control signals determine the motor speed, direction, and braking.

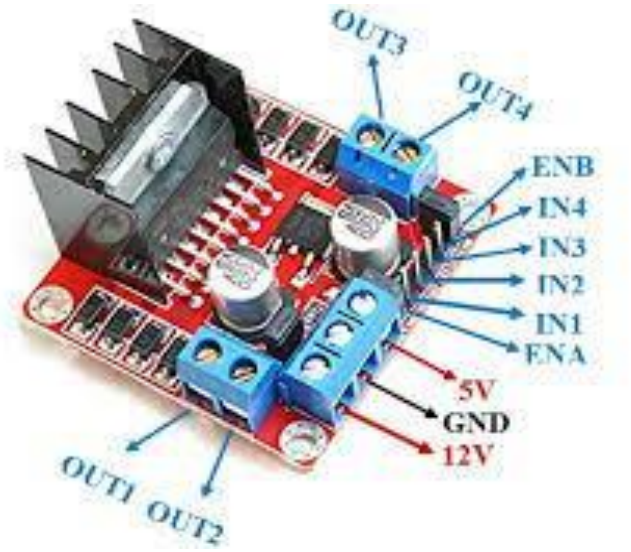


Figure 2: L298 Motor Driver

An ultrasonic sensor is a device that uses sound waves of high frequency (typically above 20 kHz) to detect and measure distances or detect the presence of objects. It emits ultrasonic waves and then listens for the echo produced when the waves bounce back after hitting an object. By measuring the time it takes for the echo to return, the sensor can calculate the distance to the object.

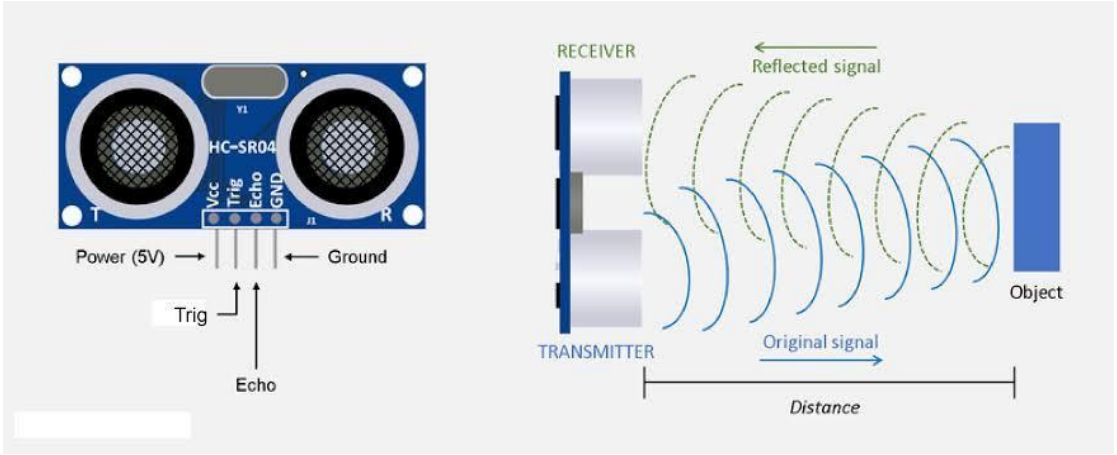


Figure 3: Ultrasonic Sensor

Ultrasonic sensors consist of a transmitter and a receiver. The transmitter emits ultrasonic waves, while the receiver detects the reflected waves. The sensor operates based on the principle of time-of-flight, where the time taken for the sound wave to travel to the object and back is measured.

If we need to measure the specific distance from your sensor, this can be calculated based on this formula:
Distance = ½ T x C
(T = Time and C = the speed of sound)
At 20°C (68°F), the speed of sound is 343 meters/second (1125 feet/second), but this varies depending on temperature and humidity.

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision.

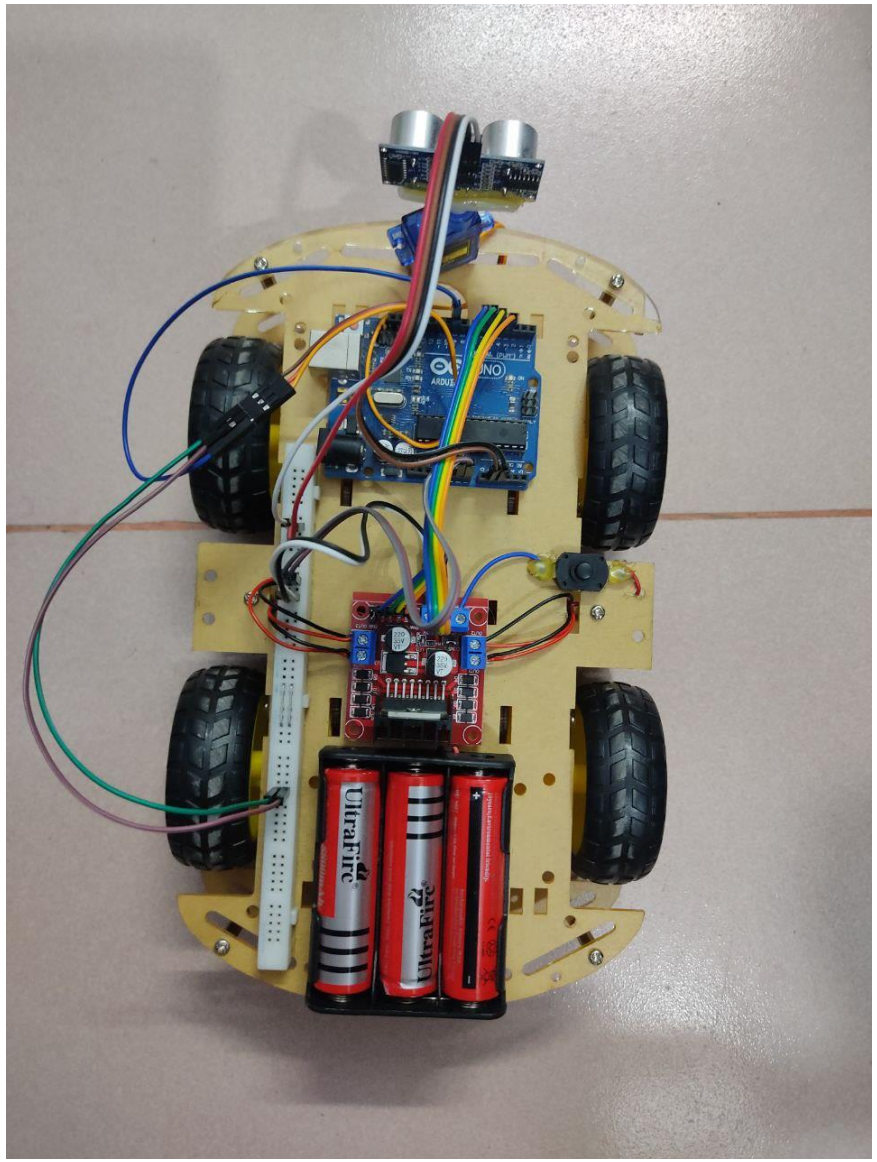
If there is a need to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism.

If motor is powered by a DC power supply then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the DC servo motor working.

It consists of three parts:

- Controlled device
- Output sensor
- Feedback system

Figure 4: Servo Motor



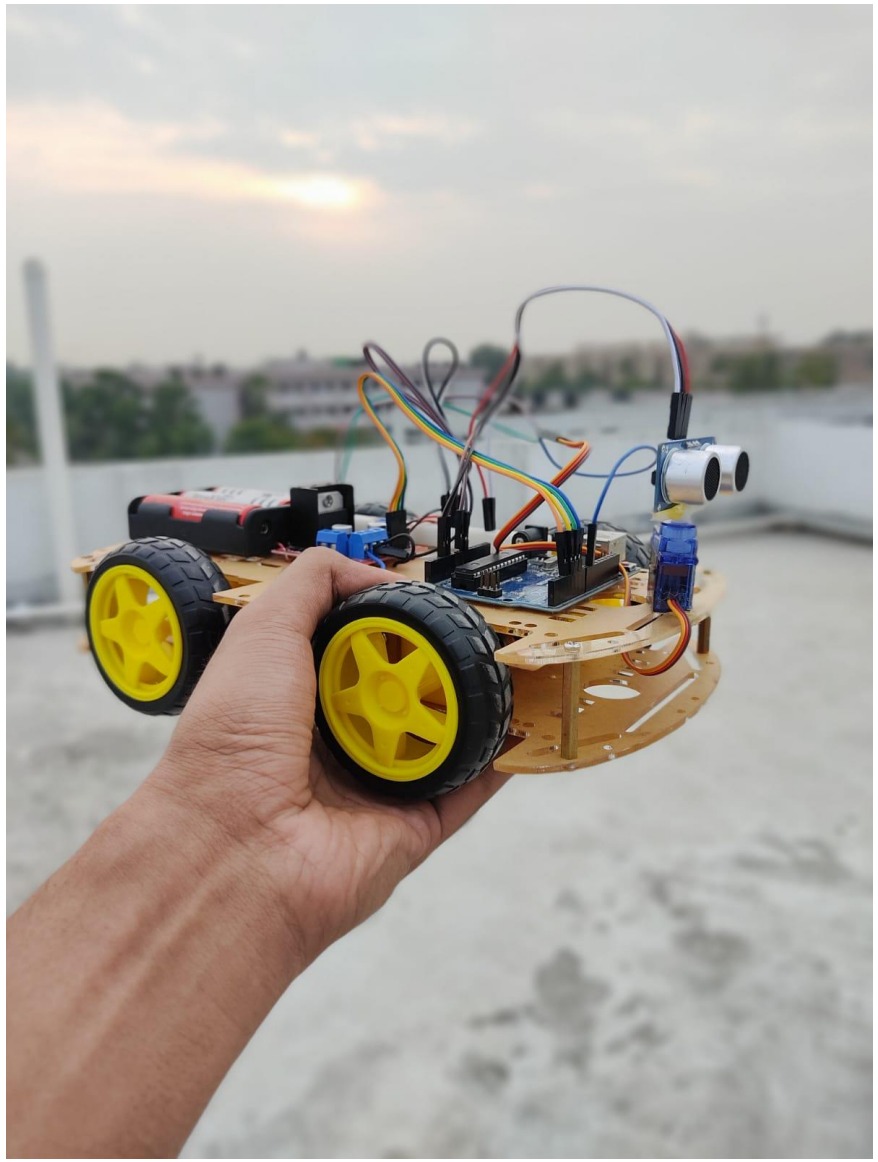


Figure 5: Project view

Connecting Components

Software tools:

- Programming Languages : Arduino.
- Operating Systems : Windows 11
- IDE : Arduino IDE

Hardware tools:

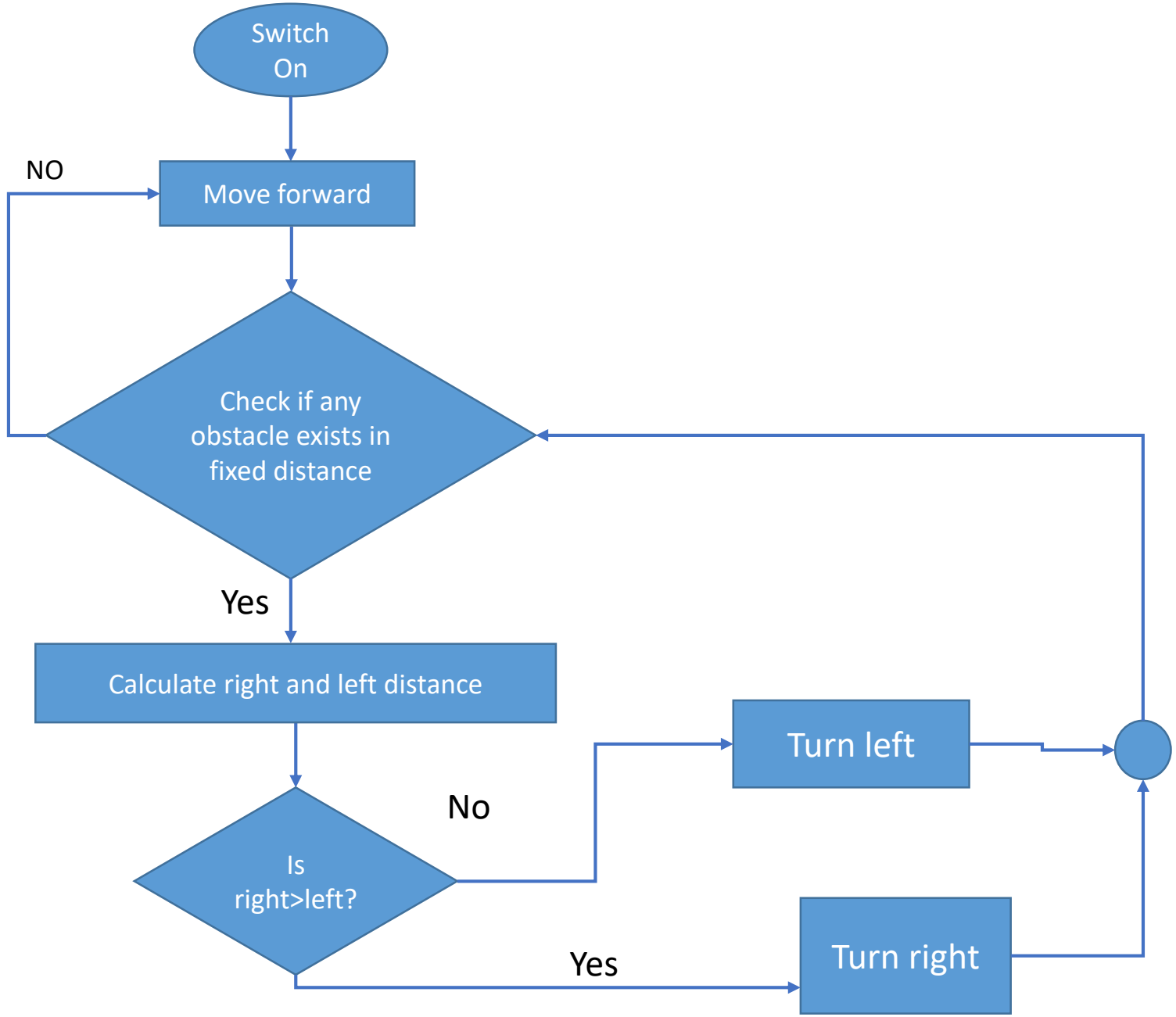
Table 1: Table for Hardware Tools

Serial	Equipment name	Rating	Quantity
01	Arduino Uno	Micro-controller: ATmega328. Operating Voltage: 5V. Input Voltage (recommended):7-12V. Digital I/O Pins: 14 (of which 6 provide PWM output). Analog Input Pins: 6. Weight: 28(without Cable) 54(with cable) Flash Memory: 32 KB	**01**

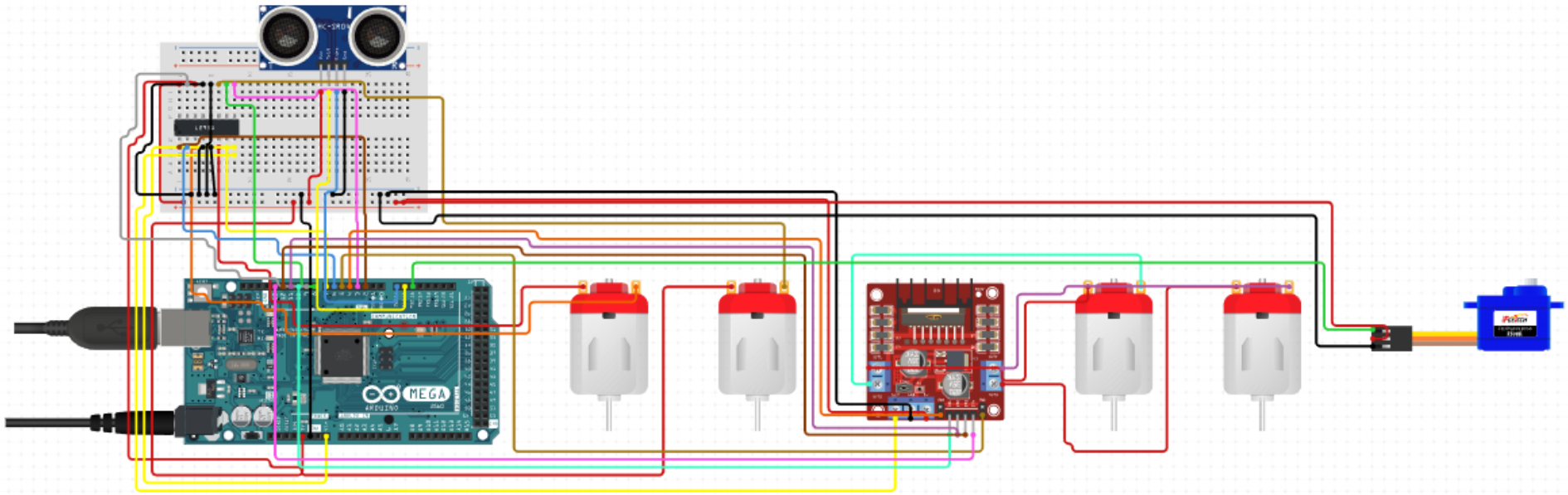
Serial	Equipment Name	Rating	Quantity
02	L298 Motor Driver	Double H bridge Drive Chip: L298N. Operating Voltage(VDC): 5~35 Peak Current (A): 2 Continuous Current (A): 0-36mA No. of Channels: 2	**01**
03	Ultrasonic Sensor	Range : 40 cm to 300 cm. The response time :50ms-200ms The Beam angle: around $5^{0.}$ Operating voltage: 20 - 30 VDC. Preciseness I: ±5% Frequency :120 kHz. Resolution : 1mm	**01**
04	Servo Motor	Rated Output Power: 200 W (1/4 HP) Operating voltage: 4.8~ 6.6v. Motor Frame Size: 60 mm (2.36 in.)	**01**

Serial	Equipment Name	Rating	Quantity
05	DC Gear Motor	Operating Voltage(VDC): 3~12 Shaft Length (mm): 8.5 Shaft Diameter (mm): 5.5 (Double D-type) No Load Current: 40-180mA. Rated Speed(After Reduction): 180 RPM Rated Torque: 0.35 Kgcm	04
06	Smart robot car Wheels	---	04
07	On/Off switch	----	01
08	Battery Cell Holder	3 cell capacities	01
09	Battery Cell	3.7 volt li-ion 8800 mAh	03
10	Connecting Wires	---	As required

Flow Chart


Switch On
NO
Move forward
Check if any obstacle exists in fixed distance
Yes
Calculate right and left distance
Is right>left?
No
Turn left
Yes
Turn right

Circuit Diagram



Pseudo Code

Import Servo library
Import NewPing library

Declare constants:
    LeftMotorForward = 6
    LeftMotorBackward = 5
    RightMotorForward = 4
    RightMotorBackward = 3
    trig_pin = A1
    echo_pin = A2
    maximum_distance = 200

Declare variables:
    goesForward = false
    distance = 100

Create a NewPing instance named sonar with trig_pin, echo_pin, and maximum_distance

Create a Servo instance named servo_motor

Setup:

Set the pinMode for RightMotorForward, LeftMotorForward, LeftMotorBackward, and RightMotorBackward as OUTPUT

Attach servo_motor to pin 9

Write 60 to servo_motor

Delay 2000 milliseconds

Read distance from the ultrasonic sensor and assign it to distance

Delay 100 milliseconds

Repeat the previous two steps three more times with a delay of 100 milliseconds

Loop:

Declare and initialize distanceRight and distanceLeft as 0

Delay 50 milliseconds

Print distance to the serial monitor

If distance is less than or equal to 25:

Call moveStop function

Delay 300 milliseconds

Call moveBackward function

Delay 400 milliseconds

Call moveStop function

Delay 300 milliseconds

```
 Set distanceRight to the result of calling lookRight function
        Delay 300 milliseconds
        Set distanceLeft to the result of calling lookLeft function
        Delay 300 milliseconds

        If distance is greater than or equal to distanceLeft:
            Call turnRight function
            Call moveStop function
        Else:
            Call turnLeft function
            Call moveStop function
    Else:
        Call moveForward function
    Read distance from the ultrasonic sensor and assign it to distance
lookRight function:
    Write 0 to servo_motor
    Delay 500 milliseconds
    Read distance from the ultrasonic sensor and assign it to distance
    Delay 100 milliseconds
    Write 60 to servo_motor
    Return distance
```

lookLeft function:
    Write 130 to servo_motor
    Delay 500 milliseconds
    Read distance from the ultrasonic sensor and assign it to distance
    Delay 100 milliseconds
    Write 60 to servo_motor
    Return distance
    Delay 100 milliseconds

readPing function:
    Delay 70 milliseconds
    Read distance in centimeters from the ultrasonic sensor and assign it to cm
    If cm is equal to 0, set cm to 250
    Return cm
moveStop function:
    Set RightMotorForward, LeftMotorForward, RightMotorBackward, and LeftMotorBackward to LOW
moveForward function:
    If goesForward is false:
        Set goesForward to true
        Set LeftMotorForward and RightMotorForward to HIGH
        Set LeftMotorBackward and RightMotorBackward to LOW

moveBackward function:

Set goesForward to false

Set LeftMotorBackward and RightMotorBackward to HIGH

Set LeftMotorForward and RightMotorForward to LOW

turnRight function:

Set LeftMotorForward and RightMotorBackward to HIGH

Set LeftMotorBackward and RightMotorForward to LOW

Delay 250 milliseconds

Set LeftMotorForward and RightMotorForward to HIGH

Set LeftMotorBackward and RightMotorBackward to LOW

turnLeft function:

Set LeftMotorBackward and RightMotorForward to HIGH

Set LeftMotorForward and RightMotorBackward to LOW

Delay 250 milliseconds

Set LeftMotorForward and RightMotorForward to HIGH

Set LeftMotorBackward and RightMotorBackward to LOW

Applications:

1. **Robotics Research and Education**: Obstacle-avoiding cars serve as valuable platforms for robotics research and education. They provide hands-on experience in building autonomous systems, developing navigation algorithms, and understanding sensor integration.

2. **Industrial Automation**: Obstacle-avoiding cars can be used in industrial settings to transport materials or perform tasks while avoiding obstacles. They can navigate within warehouses, factories, or assembly lines, enhancing efficiency and safety.

3. **Home Cleaning Robots**: Obstacle-avoiding cars can be utilized in robotic vacuum cleaners or floor-cleaning robots. By equipping them with sensors, such as infrared or ultrasonic sensors, the robots can detect obstacles and navigate around furniture and other objects.

4. **Security and Surveillance**: Obstacle-avoiding cars can be employed in security or surveillance systems. They can patrol designated areas and detect and avoid obstacles, ensuring comprehensive coverage and minimizing collisions.

5.Personal Assistance Robots: Obstacle-avoiding cars can form the basis for personal assistance robots that help individuals with mobility impairments or elderly individuals. These robots can navigate through a home environment, avoiding obstacles and providing support.

6.Agricultural Automation: Obstacle-avoiding cars can be adapted for agricultural purposes. They can autonomously move through fields or greenhouses, avoiding plants and obstacles, and perform tasks such as monitoring crop health, collecting data, or applying treatments.

7.Warehouse and Logistics: Obstacle-avoiding cars can be used in warehouses or logistics centers for material handling and inventory management They can navigate aisles, avoiding obstacles, and transport goods efficiently within the facility.

8.Autonomous Vehicles: The principles and technologies used in obstacle-avoiding cars can be applied to larger-scale autonomous vehicles, such as self-driving cars. The ability to detect and avoid obstacles is crucial for the safe navigation of autonomous vehicles in real-world environments.

Discussion and Conclusion :

The entire project of obstacle avoiding car was exciting to exercise and acquire practical knowledge about Arduino and various hardware components. But during execution, we had to face many challenges. The main challenge of the project was the proper power distribution among the components.

To solve this, we had to use extra breadboard and connecting wires. It was quite toilsome as we had to change our entire circuit design. After ensuring the proper distribution of power, the set up were rigorously checked by test runs.
Now the next issue was the motor driver was not having proper connection with the motors.One motor was not responding properly.Then we troubleshooted the entire process by reassembling the motors with the motor driver.
Then we had to modify our code because of obstacle detecting malfunction. We had to change the delay values and modify some functions.Then after uploading the new code the car was properly functioning according to the alogrithm.
In conclusion, the development of the obstacle avoiding car, gave us a practical knowledge about Arduino UNO as well as providing us a clear and concise insights of hardware devices. The entire project was an exciting endeavor to exhibit our knowledge about Arduino as well as interfacing devices.

References :

- https://docs.arduino.cc
- https://www.quora.com
- https://www.flyrobo.in/blog/obstacleavoidingcar

Under The Guidance Of:

Md Repon Islam

Lecturer
Department of Computer Science and Engineering
Khulna University of Engineering and Technology

Md. Badiuzzaman Shuvo

Lecturer
Department of Computer Science and Engineering
Khulna University of Engineering and Technology