KATHMANDU UNIVERSITY

SCHOOL OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MINI PROJECT REPORT ON

OBSTACLE AVOIDING CAR

Submitted by:

Aakriti Banjara (05)

Sabin Bhattarai (09)

Sudip Subedi (52)

SUBMITTED TO:

Aayush Bista

Department of Electrical Engineering

4th Jan , 2024

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

The development of autonomous systems has garnered immense attention due to their potential applications across various industries. In this project, an obstacle-avoiding car has been designed and implemented using Arduino Uno, servo motor, motor driver, ultrasonic sensor, LiIon battery, and switches. The aim of this project is to create a prototype of an autonomous vehicle capable of navigating through an environment while avoiding obstacles in its path. The integration of hardware components and the utilization of software algorithms in this project form the basis for an intelligent control system. The implementation of sensors, motor controls, and decision-making algorithms allows the car to perceive its surroundings and respond accordingly to ensure safe navigation without human intervention.

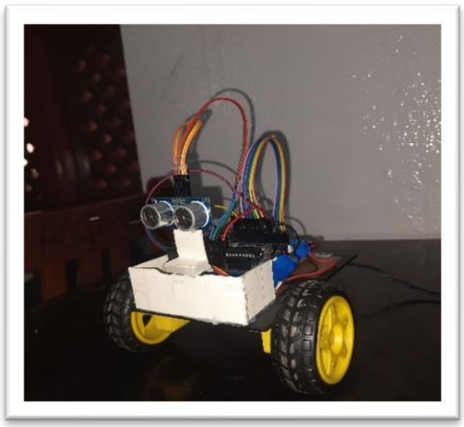


Figure 1 Diagram of the Obstacle avoiding car

Components:

- Arduino Uno: The central control unit responsible for processing sensor data and controlling the motors.
- Ultrasonic Sensor: Used to detect obstacles and measure distances.
- Servo Motor: Controls the steering mechanism of the car.

- Motor Driver: Enables the movement of the car by controlling the motors.
- Li-Ion Battery: Powers the entire system.

Wiring Connections:

Ultrasonic Sensor to Arduino Uno:

- VCC (Sensor) -> 5V pin (Arduino Uno)
- GND (Sensor) -> GND pin (Arduino Uno)
- Trig (Sensor) -> Digital pin (for sending trigger signal)
- Echo (Sensor) -> Digital pin (for receiving echo signal)

Servo Motor to Arduino Uno:

- VCC (Servo Motor) -> 5V pin (Arduino Uno)
- GND (Servo Motor) -> GND pin (Arduino Uno)
- Signal (Servo Motor) -> Digital pin (for controlling servo movement)

Motor Driver to Arduino Uno:

- VCC (Motor Driver) -> 5V pin (Arduino Uno)
- GND (Motor Driver) -> GND pin (Arduino Uno)
- Motor Inputs (Motor Driver) -> Connected to the respective motor terminals.
- Control Pins (Motor Driver) -> Connected to Arduino pins (for motor control)

Li-Ion Battery to Components:

- Positive terminal (Battery) -> VCC of the components (Arduino Uno, sensors, motor driver) • Negative terminal (Battery) -> GND of the components (Arduino Uno, sensors, motor driver)

METHODOLOGY

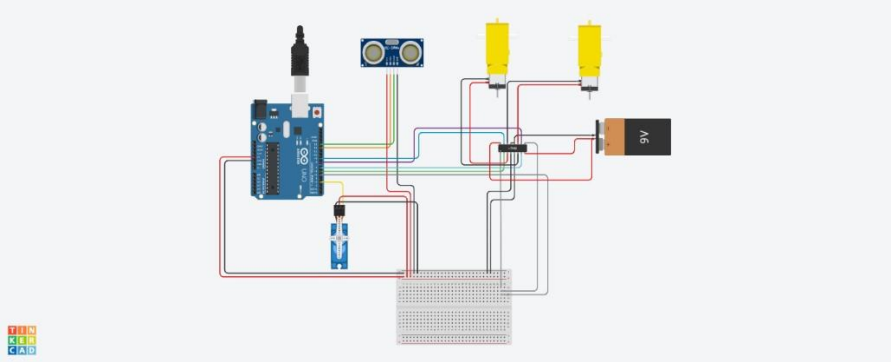


Figure 2 Circuit diagram

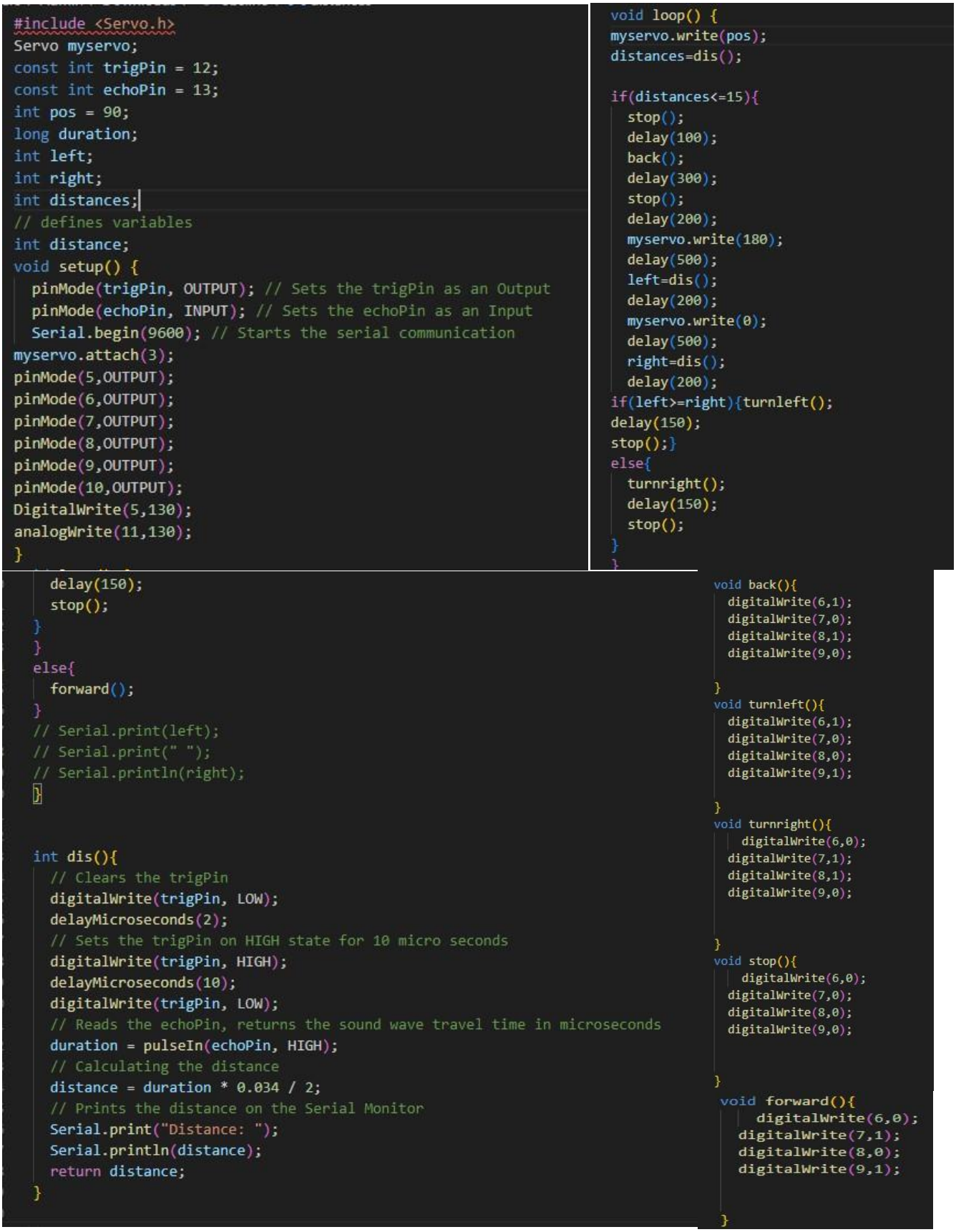
Hardware Assembly: The Arduino Uno serves as the central control unit. The servo motor and motor driver are employed for controlling the movement of the car, while an ultrasonic sensor is utilized for detecting obstacles. These components are interconnected and powered by a Li-Ion battery to ensure mobility and functionality.

Sensor Integration: The ultrasonic sensor is placed strategically to provide accurate distance measurements from obstacles. This data is crucial for the decision-making process of the car's navigation.

Control Algorithm: A control algorithm is programmed in the Arduino Uno to interpret data from the ultrasonic sensor. When an obstacle is detected within a certain range, the algorithm triggers the servo motor to steer the car away from the obstacle's path.

Testing and Calibration: The system undergoes rigorous testing to ensure its functionality and accuracy in obstacle detection and avoidance. Calibration of the sensor and fine-tuning of the control algorithm are performed to enhance the car's responsiveness and reliability.

Documentation and Report: Comprehensive documentation of the project, including circuit diagrams, code snippets, and test results, is compiled for submission. This report encapsulates the entire process from design to implementation.



```
#include <Servo.h>
Servo myservo;
const int trigPin = 12;
const int echoPin = 13;
int pos = 90;
long duration;
int left;
int right;
int distances;
// defines variables
int distance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
myservo.attach(3);
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);
pinMode(9,OUTPUT);
pinMode(10,OUTPUT);
DigitalWrite(5,130);
analogWrite(11,130);
}
```

```
void loop() {
myservo.write(pos);
distances=dis();

if(distances<=15){
  stop();
  delay(100);
  back();
  delay(300);
  stop();
  delay(200);
  myservo.write(180);
  delay(500);
  left=dis();
  delay(200);
  myservo.write(0);
  delay(500);
  right=dis();
  delay(200);
if(left>=right){turnleft();
delay(150);
stop();}
else{
  turnright();
  delay(150);
  stop();
}
}
```

```
    delay(150);
    stop();
  }
  }
  else{
    forward();
  }
  // Serial.print(left);
  // Serial.print(" ");
  // Serial.println(right);
  }


  int dis(){
    // Clears the trigPin
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);
    // Calculating the distance
    distance = duration * 0.034 / 2;
    // Prints the distance on the Serial Monitor
    Serial.print("Distance: ");
    Serial.println(distance);
    return distance;
  }
```

```
void back(){
  digitalWrite(6,1);
  digitalWrite(7,0);
  digitalWrite(8,1);
  digitalWrite(9,0);

}
void turnleft(){
  digitalWrite(6,1);
  digitalWrite(7,0);
  digitalWrite(8,0);
  digitalWrite(9,1);

}
void turnright(){
    digitalWrite(6,0);
  digitalWrite(7,1);
  digitalWrite(8,1);
  digitalWrite(9,0);


}
void stop(){
    digitalWrite(6,0);
  digitalWrite(7,0);
  digitalWrite(8,0);
  digitalWrite(9,0);


}
 void forward(){
    digitalWrite(6,0);
   digitalWrite(7,1);
   digitalWrite(8,0);
   digitalWrite(9,1);


 }
```

Figure 3 Arduino code of the obstacle avoiding car

Explanation of the code

Libraries Used:

- #include<Servo.h>: This line includes the Servo library, allowing control of servo motors.

Global Variables:

- Servo myservo; Declares a servo object named myservo.
- const int trigPin = 12; and const int echoPin = 13;: Define the pins for the ultrasonic sensor's trigger and echo respectively.
- int pos = 90; Initializes a variable for servo position.
- long duration; Variable to store the duration of the ultrasonic wave.
- int left; and int right; Variables to store distances obtained from the left and right directions.
- int distances; Variable to store the distance obtained from the front.

Setup Function:

- pinMode() commands set the pinMode for the trigger and echo pins of the ultrasonic sensor as well as for the motor control pins.
- Serial.begin(9600); initiates serial communication at a baud rate of 9600.
- myservo.attach(3);: Attaches the servo to pin 3.
- DigitalWrite(5,130); and analogWrite(11,130);: Incorrect capitalization (DigitalWrite instead of digitalWrite) and a typo in analogWrite (it should be analogWrite, not AnalogWrite). These lines attempt to set a digital pin (5) to a value (130) but may not be functional due to the incorrect syntax.

Loop Function:

- myservo.write(pos);: Sets the servo motor to the initial position (90 degrees).
- distances = dis();: Retrieves the distance measured by the ultrasonic sensor.
- Conditional statements:
- If an obstacle is detected within 15 units:
- The car stops (stop()), moves backward (back()), then turns the servo to check distances on the left and right sides.
- Determines whether to turn left or right based on the measured distances.

- If no obstacle is detected:
- Moves forward (forward()).

dis() Function:

- Triggers the ultrasonic sensor to measure distance.
- Calculates the distance based on the time taken for the ultrasonic wave to return.
- Prints the distance measured by the sensor.
- Returns the distance value.

Motor Control Functions:

- back(), turnleft(), turnright(), stop(), and forward(): These functions control the motor's movements by setting specific pins to HIGH or LOW to achieve forward, backward, left, right, and stop motions.

CONCLUSION

In conclusion, the development and implementation of the obstacle-avoiding car project using Arduino Uno, servo motors, ultrasonic sensors, and motor drivers have showcased the practical integration of hardware components and software algorithms to create an autonomous navigation system. The project aimed to design a prototype capable of perceiving its environment, detecting obstacles, and maneuvering autonomously to avoid collisions.

Through the utilization of an ultrasonic sensor, the car successfully detected obstacles within its path, triggering appropriate responses to navigate around them. The control algorithm, albeit needing refinement in some aspects, demonstrated the ability to make decisions based on the distance measurements obtained from the sensor.

The project's successful aspects included the integration of sensor data with motor control, enabling the car to move forward, backward, turn left, and turn right, effectively avoiding obstacles within its range.