Embedded system:

Department of Electrical Engineering

Namal University Mianwali

Name	**Hassan zeb | Badar Hafeez**
Roll No	**NUM-BSEE-2022-23 | NUM-BSEE-2022-25**
Submitted to	**MS. NAUREEN SHAUKAT & MS. MISBAH BATOOL**

Title: LINE FOLLOWER ROBOT

Contents

Abstract:

This project involves creating a line follower robot, a small vehicle that autonomously follows a path marked by a black line on a white surface. Using sensors to detect the line, a microcontroller to process data, and motors to drive the robot, we developed a system that smoothly navigates curves and intersections. Despite challenges in sensor calibration and movement stability, we successfully built a robot that accurately follows a test track. This project showcases how electronics and programming can combine to solve practical problems in robotics and automation.

Introduction:

A line follower robot is an autonomous robot that follows a path marked by a black line on a white surface. In our project, we built such a robot using simple electronic components and a microcontroller. The robot has three sensors that detect the line and send information to the microcontroller, which then adjusts the robot's direction by controlling the motors. If the middle sensor detects the line, the robot moves forward; if the left sensor detects the line, the robot turns left; and if the right sensor detects the line, it turns right. This setup allows the robot to navigate the path smoothly, even around curves. Building this robot helped us learn how to combine electronics, coding, and mechanical parts to solve a real-world problem.

Objective:

- **Design and Build:** To design and construct a functional line follower robot using basic electronic components, including sensors, a microcontroller, and motors.
- **Sensor Integration**: To integrate three sensors (left, right, and middle) that can accurately detect the line and provide real-time data to the microcontroller.

- **Programming:** To develop and implement a control algorithm that processes sensor inputs and controls the motors to ensure the robot follows the line smoothly.
- **Navigation:** To enable the robot to navigate various path configurations, including straight lines, curves, and intersections, with high accuracy and stability.
- **Performance Testing:** To test the robot's performance under different conditions and refine the sensor calibration and motor control for optimal functionality.
- **Educational Value:** To enhance understanding of robotics, automation, and the integration of hardware and software in a practical application.
- **Problem Solving:** To identify and address challenges encountered during the design and implementation phases, such as sensor inaccuracies and movement stability issues.

Equipment required :

Software:

- **Arduino Idle**

Hardware:

- **Arduino**
- **Motor Driver**
- **2 DC Motors**
- **1 Rotating Wheel**
- **2 Wheels**
- **Wires (Male-to-Male and Male-to-Female)**
- **3 IR Sensors**

Methodology or Procedure:

Component Setup: Gather all necessary components: Arduino, motor driver, two DC motors, one rotating wheel, wires (male-to-male and male-to-female), and three IR sensors.

Circuit Assembly:

- Connect the Arduino to the motor driver. Use digital pins 2, 4, 7, and 8 on the Arduino to control the IN1, IN2, IN3, and IN4 pins on the motor driver, respectively.
- Connect the ENA and ENB pins on the motor driver to PWM pins 3 and 5 on the Arduino to control motor speed.
- Connect the DC motors to the motor driver's output terminals (OUT1, OUT2, OUT3, OUT4).
- Attach the IR sensors to the analog pins on the Arduino: the right sensor to A0, the left sensor to A1, and the middle sensor to A2.
- Use male-to-male and male-to-female wires to connect the components as shown in the provided circuit diagram.

Power Supply: Connect the power supply to the motor driver and the Arduino. Use two 3.7V lithium batteries to provide the necessary power.

Programming:

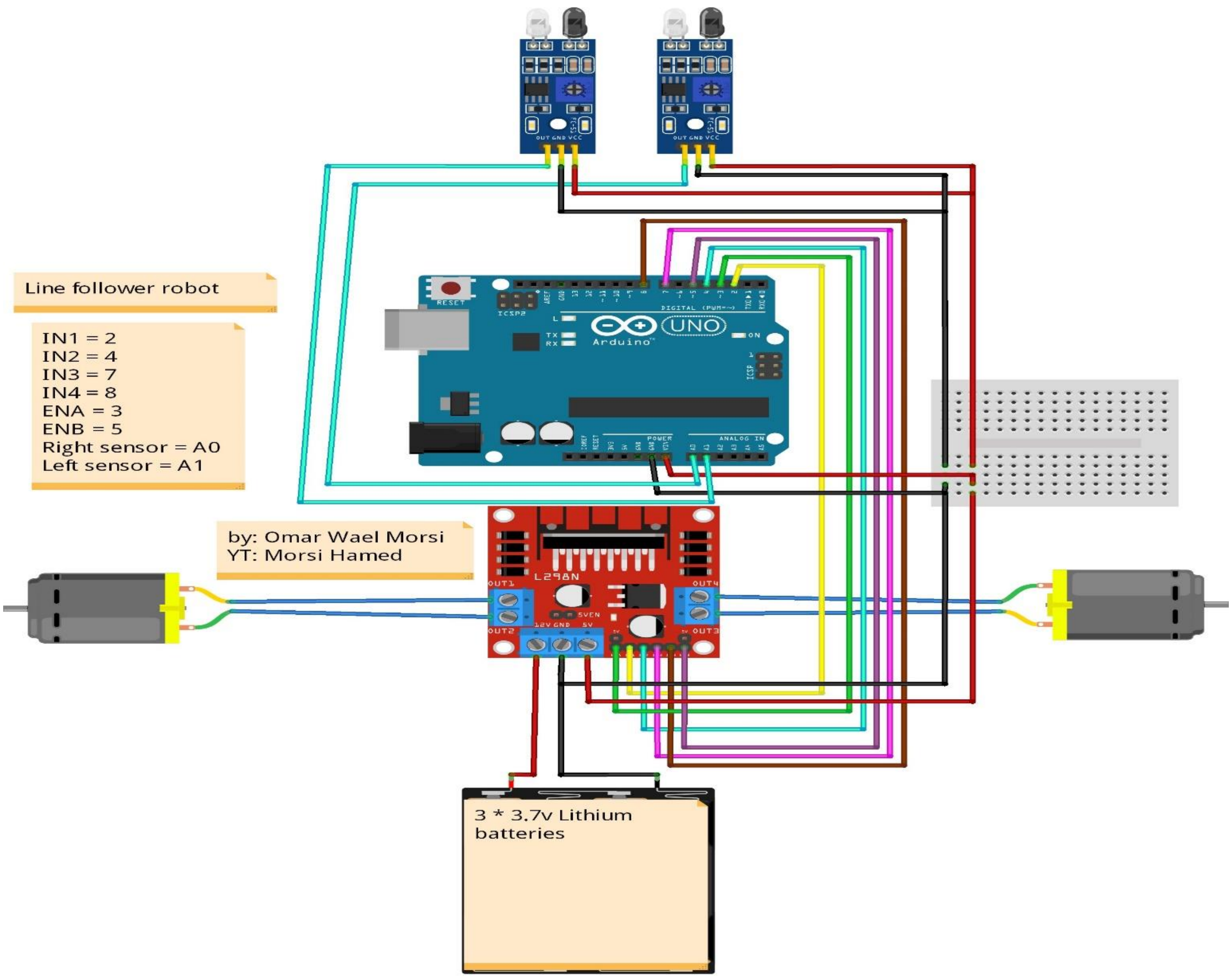
- Write the control code for the Arduino. Initialize the sensor pins (A0, A1, A2) as input and motor control pins (2, 4, 7, 8, 3, 5) as output.
- In the code’s setup function, set the pin modes appropriately and start serial communication for debugging.
- In the loop function, continuously read the values from the IR sensors. Based on the sensor readings, adjust the motor speeds to follow the line. If the middle sensor detects the line, move forward; if the left sensor detects the line, turn left; if the right sensor detects the line, turn right.

Testing and Calibration:

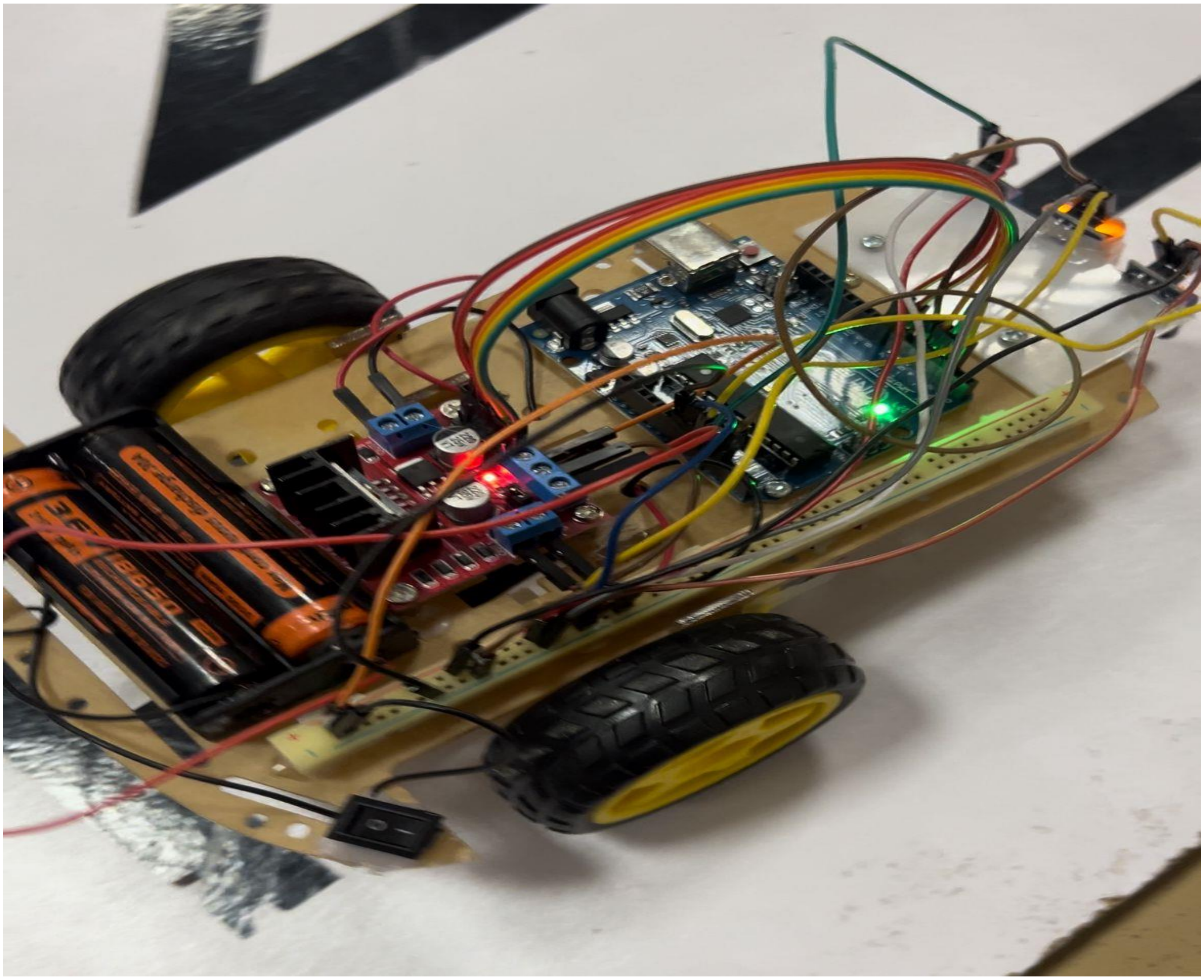
- Upload the code to the Arduino and place the robot on the track.
- Observe the robot’s movement and make necessary adjustments to the sensor calibration and motor speeds in the code to ensure smooth and accurate line following.

Circuit configuration:

Software circuit:



Hardware circuit:



Code:

```
int IN1 = 2 ;
int IN2 = 4 ;
int IN3 = 7 ;
int IN4 = 8 ;
int ENA = 3;
int ENB = 5;
int Msensor = A2;
int Lsensor = A1;
int Rsensor = A0;
int RIGHT , LEFT, mid;
int x = 100;
int y = 100;
```

```
int x1=80;
int y1=80;


void setup() {
Serial.begin(9600);
pinMode(IN1 , OUTPUT);
pinMode(IN2 , OUTPUT);
pinMode(IN3 , OUTPUT);
pinMode(IN4 , OUTPUT);
pinMode(ENA , OUTPUT);
pinMode(ENB , OUTPUT);
pinMode(Rsensor , INPUT);
pinMode(Lsensor , INPUT);
pinMode(Msensor , INPUT);
}

void loop() {
  RIGHT = digitalRead(Rsensor);
  LEFT = digitalRead(Lsensor);
  mid = digitalRead(Msensor);
  Serial.print("RIGHT sensor: ");
  Serial.println(RIGHT);
  Serial.print("LEFT sensor: ");
  Serial.println(LEFT);
  Serial.print("Mid sensor: ");
  Serial.println(mid);

  if (RIGHT == 0 && LEFT == 0 && mid == 1) {
    digitalWrite(IN1 , HIGH);
    digitalWrite(IN2 , LOW);
    digitalWrite(IN3 , HIGH);
    digitalWrite(IN4 , LOW);
    analogWrite(ENA , x);
    analogWrite(ENB , y);
    Serial.println("FORWARD");
  }

  else if (RIGHT == 0 && LEFT == 1) {
    digitalWrite(IN1 , HIGH);
    digitalWrite(IN2 , LOW);
    digitalWrite(IN3 , LOW);
    digitalWrite(IN4 , HIGH);
    analogWrite(ENA , x1); // Motor A forward
    analogWrite(ENB , 0); // Motor B stopped
    Serial.println("LEFT");
  }
```

```
  else if (RIGHT == 1 && LEFT == 0) {
   digitalWrite(IN1 , LOW);
   digitalWrite(IN2 , HIGH);
   digitalWrite(IN3 , HIGH);
   digitalWrite(IN4 , LOW);
   analogWrite(ENA , 0); // Motor A stopped
   analogWrite(ENB , y1); // Motor B forward
   Serial.println("RIGHT");
  }

  else if (RIGHT == 1 && LEFT == 1 && mid == 1) {
   digitalWrite(IN1 , HIGH);
   digitalWrite(IN2 , LOW);
   digitalWrite(IN3 , HIGH);
   digitalWrite(IN4 , LOW);
   analogWrite(ENA , x);
   analogWrite(ENB , y);
   Serial.println("FORWARD");
  }

//  else if (RIGHT == 0 && LEFT == 0 && mid==0) {
//
//    digitalWrite(IN1 , LOW);
//    digitalWrite(IN2 , LOW);
//    digitalWrite(IN3 , LOW);
//    digitalWrite(IN4 , LOW);
//    analogWrite(ENA , 0); // Motor A stopped
//    analogWrite(ENB , 0); // Motor B stopped
//    Serial.println("STOP");
//
//  }
}
```

Limitations:

There were a few problems we ran into during testing. First off, the robot didn't work well in sunlight because its sensors couldn't see the line. Also, the sensors we used weren't great at detecting the line accurately, especially in tricky spots. Plus, we had issues with the battery running out too quickly and sometimes the robot got confused when it came across breaks or in loops in the line. So, we need to find better sensors that work in different lighting.

Conclusion:

In the conclusion, this project introduces a fundamental setup for a line-following robot. By interpreting signals from infrared sensors, the robot can detect and stay on track with a designated line. While the current code provides a solid starting point, there's room for growth. Future improvements might include fine-tuning movements for smoother navigation or adding features like obstacle detection for a more versatile robot. Additionally, upgrading to IR array sensors could enhance the robot's performance, offering more precise and reliable line tracking.

References:

https://www.youtube.com/watch?v=h7k4kL8wFCE