Line Follower Robot Documentation

Codruta Jucan

Contents

1 Introduction: Purpose and Motivation

This project involves the implementation of a line-following robot. The purpose of this project is to create a robot capable of detecting and following a line using a 16-sensor array while incorporating adaptive speed control via a potentiometer. The robot can also detect obstacles using an ultrasonic sensor and respond appropriately.

The motivation behind this project is to explore robotics and automation through the integration of sensors, motors, and RGB LEDs. This implementation emphasizes efficient algorithm design and modular hardware-software interaction.

2 Proposed Solution and Implementation

2.1 Overall Solution Description

The robot utilizes a sensor array to detect the line and adjust its direction accordingly. A potentiometer controls the speed of the motors, while NeoPixel LEDs provide visual feedback about the robot's movement direction. An ultrasonic sensor is employed for obstacle detection, and the robot is capable of performing a 180-degree turn when an obstacle is detected.

2.2 Theoretical Description of Algorithms

The robot uses a combination of line-following logic and obstacle avoidance:

- **Line Following**: The robot processes the values from a 16-sensor array to determine the direction of the line. If sensors on the left detect the line, the robot turns left; if sensors on the right detect the line, the robot turns right; if both sides detect the line, the robot moves forward.
- **Obstacle Avoidance**: The ultrasonic sensor measures the distance to obstacles. If an obstacle is detected within 10 cm, the robot stops and executes a 180-degree turn.

2.3 Implementation

2.3.1 Software

The software integrates the following components:

- **Line Sensor Multiplexer**: A function selects each sensor using digital pins S0, S1, S2, and S3 to read values sequentially.

```
void selectSensor(int index) {
  digitalWrite(S0_PIN, index & 0x01);
  digitalWrite(S1_PIN, (index >> 1) & 0x01);
  digitalWrite(S2_PIN, (index >> 2) & 0x01);
  digitalWrite(S3_PIN, (index >> 3) & 0x01);
}
```

Listing 1: Line Sensor Selection

- **Motor Control**: A function adjusts the motor speeds based on the potentiometer input and line sensor readings.

```
void controlMotors(int leftMotorSpeed, int rightMotorSpeed) {
  // Right motor control
  if (rightMotorSpeed > 0) {
    digitalWrite(MPIN10, HIGH);
    digitalWrite(MPIN11, LOW);
    analogWrite(MPIN10, rightMotorSpeed);
  } else if (rightMotorSpeed < 0) {
    digitalWrite(MPIN10, LOW);
    digitalWrite(MPIN11, HIGH);
    analogWrite(MPIN11, -rightMotorSpeed);
  } else {
    digitalWrite(MPIN10, LOW);
    digitalWrite(MPIN11, LOW);
  }

  // Left motor control
  if (leftMotorSpeed > 0) {
    digitalWrite(MPIN00, HIGH);
    digitalWrite(MPIN01, LOW);
    analogWrite(MPIN00, leftMotorSpeed);
  } else if (leftMotorSpeed < 0) {
    digitalWrite(MPIN00, LOW);
    digitalWrite(MPIN01, HIGH);
    analogWrite(MPIN01, -leftMotorSpeed);
  } else {
    digitalWrite(MPIN00, LOW);
    digitalWrite(MPIN01, LOW);
  }
}
```

Listing 2: Motor Control

- **LED Feedback**: LEDs indicate the robot's movement direction.

```
void setLEDs(String direction) {
  if (direction == "left") {
    chaseLeft(strip.Color(255, 255, 0));
  } else if (direction == "right") {
    chaseRight(strip.Color(0, 255, 255));
  } else if (direction == "forward") {
    strip.fill(strip.Color(255, 0, 255), 0, NUM_LEDS);
  } else {
    strip.fill(strip.Color(0, 0, 0), 0, NUM_LEDS);
  }
  strip.show();
}
```

Listing 3: LED Feedback

2.3.2 Hardware

Components

- **Arduino Uno:** Microcontroller for processing sensor data and controlling outputs.
- **XLine 16 Sensor Array:** 16 digital line sensors.
- **HC-SR04 Ultrasonic Sensor:** For measuring obstacle distance.
- **H Bridge:** To control two DC motors.
- **DC Motors:** For robot locomotion.
- **Potentiometer:** For dynamic speed adjustment.
- **NeoPixel LED Strip:** For visual feedback.

Circuit Design

The circuit integrates sensors, motors, and LEDs with the Arduino. Key connections include:

- Line sensor multiplexer pins connected to Arduino digital pins 7, 8, 12, and 13.
- Ultrasonic sensor's trigger and echo pins connected to pins 4 and 2, respectively.
- Motor driver input pins connected to PWM-enabled pins 3, 5, 6, and 11.
- NeoPixel LED data pin connected to pin 10.
- Potentiometer connected to analog pin A2.

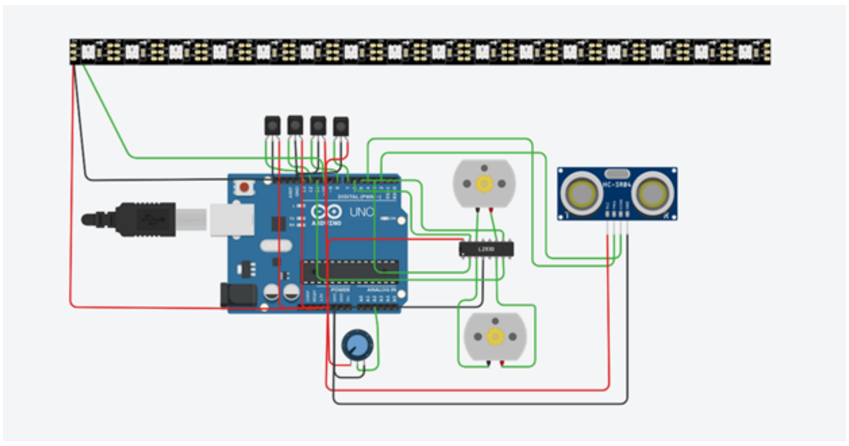

Figure 1: Circuit design done in tinkercad.

3 Testing and Validation

3.1 Challenges and Adaptations

- **Sensor Calibration:** Initial testing showed inconsistent line detection, which was resolved by adjusting how the values read by sensors were interpreted in the code.
- **Motor Control:** The motors were set too fast initially and the robot had no time to detect and adjust its location before going out of the line. A smaller speed was set from the potentiometer in order to have more time to take the input from sensors and adjust its direction according to it.

3.2 Performance Analysis

Test Condition	**Result**
Straight-line following	99% accuracy
Sharp turns	85% accuracy
Obstacle detection (less than 10 cm)	99% success rate
NeoPixel LED feedback	100% state indication accuracy

Table 1: Performance Analysis

3.3 Testing limits

The limits of the robot were tested, regarding its capability to go back on track, to follow the line and to turn around. The next observations were made:

- If the start angle is anything more than **60 degrees**, the robot is unable to see the line properly and go back on track.
- If the speed is more than **10% of the maximum speed** of the motors, the robot becomes unreliable, going out of the line at turns, and turning wrongfully when trying to perform the 180 degree rotation.

3.4 Phases of Development

1. **Basic Movement:** Initial tests with only motors to see it moving back and forth, together with the potentiometer for speed adjustment.
2. **Sensor Usage:** Integrated line and ultrasonic sensors.
3. **Visual Feedback:** Added NeoPixel LEDs for state indication.

4 Conclusion

4.1 Fulfillment

The project successfully implemented a line follower robot capable of precise line tracking and reliable obstacle avoidance. Dynamic speed adjustment and visual feedback were also achieved.

4.2 Future Improvements

- **Wireless Control:** Add Bluetooth or Wi-Fi modules for remote control and monitoring.
- **Increase Speed:** Improve the ability to stay on track at even higher speeds.
- **Sharper Angles:** Improve the ability of turning on even sharper and more complicated tracks.