Line Following Robot

Awais Asghar
School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology Scholars Ave, H-12, Islamabad
aasghar.bee22seecs@seecs.edu.pk

Abstract*— The report basically represents the making of a Line Following Robot, an automatically controlled vehicle designed to follows the dark black line embedded on the floor with the help of IR sensors feedback. The project comprises of the design and integration of hardware components as well as software algorithms, and mechanical structures. Testing demonstrates successful movement of vehicle with help of Arduino UNO.

Keywords—Arduino UNO, IR Sensors, and Line Following Robot.

I. INTRODUCTION

Line following robots are automatic vehicles capable of recognizing and following a black or white line drawn on the ground. They have many applications in Different Sectors:

- Line-following robots can be used in industries. They are used to transport materials. They are replacing the needs of Traditional conveyor belts.
- Line-following robots can be used navigating purposes in agriculture sector. Robots can navigate rows of crops in greenhouses.
- They can perform different tasks as monitoring plants health, applying fertilizers.
- By following predefined paths, they reduce the need for Labour work and improve efficiency.
- Line-following robots can be found in shopping malls as well as other entertainment venues.
- This report details the design, development, implementation, and testing of a line following robot. The First objective is to create a robot that can accurately follow a white line on a Surface using different Logics used in our Arduino code and with help of IR Sensors. [1]

II. LITERATURE REVIEW

Line following robots have been a staple in robotics competitions and practical applications. Various methods have been employed, from simple light sensors to complex vision systems. Common techniques include the use of infrared sensors to detect line contrast and employing algorithms like proportional control or PID (Proportional-Integral-Derivative) control to maintain line tracking. Despite advancements, challenges like sensor noise and variable line widths persist, necessitating robust design and calibration.

A. Components Used and their Cost

- Arduino UNO: Rs 1600
- L298N Motor Driver IC: Rs 300
- Robot Car Chassis Set: Rs 1500
- Two IR Sensors (each): 100
- Black Tape (Electrical Insulation Tape): 50
- Connecting Wires: 150
- Three Lithium Batteries 4.2V (each): 300

III. HARDWARE DESIGN

A. Arduino UNO

It is basically used to Process data from IR Sensor and Gives Instruction the Motors.

Figure 1: Arduino UNO.

B. IR Sensor

IR Sensor stands for Infrared Sensor. An IR sensor emits or detects IR radiation. It can measure the heat emitted by an object and detect motion. Basically, there are two led on IR sensor: White and black. White Led act as transmitter and black Led act as Receiver. IR sensors basically detect whether there is any object or not of sensitivity. We can also change the sensitivity of Led to detect only our required colour.

Figure 2: IR Sensor.

C. Car Chassis Set

It includes tyres, frame (to place and support sensors and Arduino), cell casing for cells and motors which are Requires to increase torque and limit the rotational speed. And to move the Rubber wheels.

Figure 3:Car Chassis Set.

D. Lithium Cells

To power L298N Motor Driver.

Figure 4:Lithium Battery Cell each of 4.2V.

E. L298N Motor Driver IC:

L298N module consists of an L298 motor driver IC and a 78M05 5V regulator. Basically, a motor Driver module that can control 4 DC motors at a time. Then has Vout of 12V to drive motors and Arduino.

Figure 5: L298N Motor Diver.

F. Mini Breadboard:

We have it for purpose of grounding.as well as for supplying 5V from Arduino UNO to power IR Sensors.

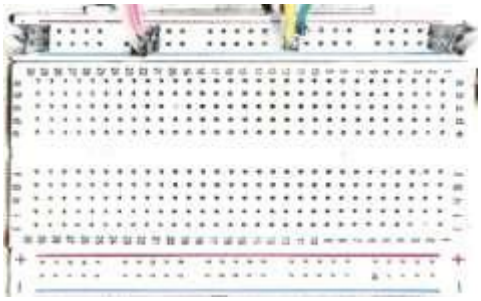

Figure 6:Mini Breadboard.

The IR sensors are positioned at the front of the robot to maximize line detection accuracy. Detailed circuit diagrams illustrate the connections between sensors, microcontroller, and motors.

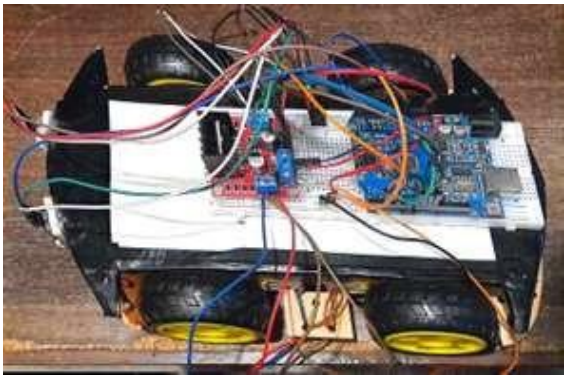

Figure 7:Circuit Diagram of Line Following Robot.

IV. 3.4 Software Design

A. Programming Language and Environment

Arduino IDE is used for coding. It's a user-friendly app. Our code is basically written in C language. C language is quite easy to understand the logics. We need to first verify our code then the code needed to be feed into the Arduino by connecting the cable with the Arduino and your laptop.

B. Simple Proportional Control Algorithm

IR sensors are responsible for the movement of our line following Robot. Like when both IR sensors detect white surface, then it will move straight with a Controlled Forward Speed given code. Similarly, if left sensor detects white Surface only, then it will move in left direction and vice versa for right movement.

C. Pseudocode:

Our main program reads the IR sensor values whether Both are on or one of them is on, or both are off. Then processes these values accordingly by Arduino Uno (by the logic made by us in code) in order to determine Line position's and also adjust motor speed.

V. 4. Implementation

A. Hardware Implementation

All batteries need to be attached at the ends of Rubber wheels. IR Sensors are properly fixed at the front with help of double tape. Breadboard is attached on the top chassis with help of glue gun. Arduino UNO as well as Motor diver is mounted on the chassis. Wiring is done according to the circuit diagram, ensuring proper connections between sensors, motors, and Arduino Uno and L298N Motor Driver. [2]

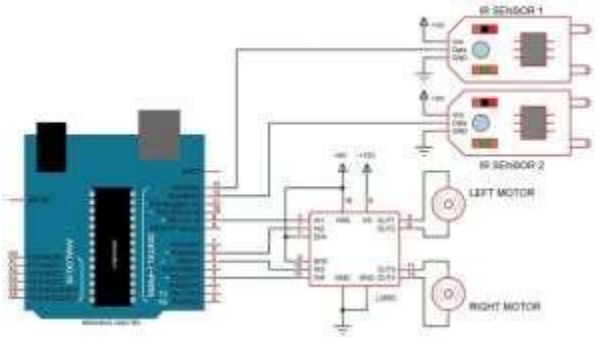

Figure 8: Circuit Diagram of Line Following Robot.

Soldering is done properly. In our case, we have also used Glue gun for permanent connections. This will ensure proper and reliable connections.

B. Software Implementation

Our code basically is modular, with different functions for sensing as well as motor control and decision making for movement. The main loop continuously reads sensor data and adjusts motor speeds. [3]

Then we define some of the pins according to our requirement as:

- *#define IR_LEFT_PIN A1:* Left IR sensor is connected to A1 analog Pin of Arduino.
- *#define IR_RIGHT_PIN A0:* Right IR sensor is connected to A1 analog Pin of Arduino.
- *#define ENA_PIN 5:* Enable Pin of motor driver is connected to PWM Pin 5 of Arduino.
- *#define ENB_PIN 3:* Enable Pin of motor driver is connected to PWM Pin 3 of Arduino
- *#define MOTOR_LEFT_PIN1 11:* Motor left pin is connected to digital pin 11 of Arduino.
- *#define MOTOR_LEFT_PIN2 10:* Motor left pin is connected to digital pin 10 of Arduino.
- *#define MOTOR_RIGHT_PIN1 9:* Motor left pin is connected to digital pin 9 of Arduino.
- *#define MOTOR_RIGHT_PIN2 8:* Motor right pin is connected to digital pin 8 of Arduino.
- PWM Pins are required for the control of motor speed.
- *void moveForward():* This function is used for the forward movement of the forward movement of the Robot.

Figure 9: Forward Motion.

- *void turnLeft():* This function is used for the forward movement of left movement of the Robot.

Figure 10: Left Direction.

- *void turnRight():* This function is used for the forward movement of right movement of the Robot.

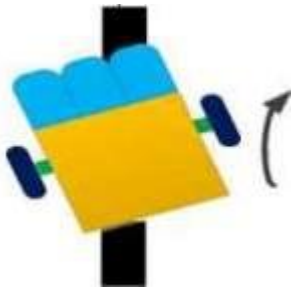

Figure 11: Right Direction.

- *void stopMotors():*This function is used to stop the movement of robot.
- *Decision of Motor actions based on IR Sensor readings:* if (irLeftValue >= IR_THRESHOLD && irRightValue >= IR_THRESHOLD) {
- *moveForward():* In this case, Both sensors see white surface and Robot will move in move forward direction.
- *turnLeft():* In this case, only left sensor see white surface and Robot will move in move left direction
- *turnRight():* In this case, only right sensor see white surface and Robot will move in move right left direction.

C. Integration:

Each hardware component is tested many times before testing. Similarly, Code always has some errors which we need to debug as well as for the proper movement of car along straight path as well as to move left or right, we need to change our code logics many times. When both hardware and software components work properly then we need to test our car movement.

VI. TESTING AND CALIBRATION

A. Test Plan

- *Straight line following:* For straight path, we have tested our car, and it is working properly with the controlled speed as given by us in code.
- *Curved line navigation:* We have also tested it. It is not working properly at first. As our speed is too high at that time. But then we control our speed. Another Hurdle is that our path is too small due to which it becomes difficult to move along curved path properly. Along Long path it can move along curved path.
- Then we create some changes in our code Like we had controlled our turn speed to 70.so that it has enough time to detect surface properly.
- We are quite successful in doing this. As our car moves properly along the curved path as well as straight path.

Figure 12:Line Following Robot Path while doing testing.

- Basically, we have made a line path by ourselves with help of two white sheets attached with each other.
- Then a black tape is used to make curved path for the line following Robot.

B. Calibration

- We have adjusted our IR sensors sensitivity with help of Screwdriver to move the screw attached on IR sensor.
- When led light of IR sensor detects black line, it will glow. In this way we have adjusted its sensitivity.
- We have adjusted sensitivity on both sensors Properly.
- We have also checked this calibration through serial Monitor in Arduinos IDEs.
- Fine-tuning motor speeds for smooth turns.
- Balancing left and right motor speeds to avoid drift.

C. Results

- Accuracy of line following.
- Speed of navigation.
- Response time to changes in line direction.
- Observations and Data Collected:
- Data on sensor readings and motor adjustments during operation.
- Performance under different lighting conditions and track configurations.

VII. RESULTS AND DISCUSSION

The line following robot successfully follows the predefined path with high accuracy. It handles straight lines and curves effectively, though intersections require further optimization. Challenges such as sensor noise and uneven surfaces were mitigated through calibration and algorithm adjustments. The robot's performance meets the initial objectives, demonstrating reliable line tracking and robust operation.

VIII. APPLICATIONS AND FUTURE WORK

- Automated guided vehicles (AGVs) in warehouses and manufacturing.
- Educational tools for teaching robotics and control systems.
- Enhanced navigation systems for autonomous vehicles.
- Integrating advanced sensors like cameras for more complex path recognition.
- Implementing machine learning algorithms for adaptive line following.
- Expanding capabilities to handle more complex environments with obstacles and dynamic paths.

IX. CONCLUSION

This project successfully designed, developed, and tested a line following robot. The robot demonstrates effective line tracking using simple and cost-effective components. Future enhancements can further improve its capabilities and expand its application scope.

References

[1] M. Mehdi Sanaatiyan, Mehran Pakdaman, “Design and Implementation of Line Follower Robot,” in *IEEE*, 2009.

[2] AUTODESK Instructables, [Online]. Available: https://www.instructables.com/Make-a-FAST-Line-Follower-Robot-Using-PID/. [Accessed May 2024].

[3] “ROBU .IN,” [Online]. Available: https://robu.in/how-to-make-a-line-follower-robot-using-arduino-connection-code/. [Accessed May 2024].