LINE FOLLOWING ROBOT USING ROS

PROJECT REPORT
MHC4372

Submitted by

KOLLIPARA MAHIDHAR (21130036)

In partial fulfilment for the award of the degree
Of

BACHELOR OF TECHNOLOGY

in

MECHATRONICS

DEPARTMENT OF MECHATRONICS ENGINEERING

HINDUSTAN INSTITUTE OF TECHNOLOGY AND SCIENCE

PADUR, CHENNAI - 603 103

MAY 2024

BONAFIDE CERTIFICATE

This is to certify that this project report **"LINE FOLLOWING ROBOT "** is the bonafide work of **"MR. KOLLIPARA MAHIDHAR(21130036) "** who carried out the project work as a part of the subject **"MHC4372 – Robot Operating System"** under my supervision during the academic year 2023-2024.

SIGNATURE

Ms. N Seenu

Asst Professor

Dept. of Mechatronics,

Hindustan Institute of Technology & Science,
No. 1, Rajiv Gandhi Salai (OMR),
Padur,
Chennai - 603103

INTERNAL EXAMINER	**EXTERNAL EXAMINER**
Name: ______________________	Name: ______________________

ACKNOWLEDGMENT

We are pleased to acknowledge our sincere thanks to Board of Management of Hindustan Institute of Science and Technology for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We would like to express our sincere and deep sense of gratitude to our Project Guide Ms. N Seenu for her valuable guidance, suggestions and constant encouragement which paved the way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Mechatronics Engineering who were helpful in many ways for the completion of the project.

KOLLIPARA MAHIDHAR (21130036)

TABLE OF CONTENTS

ABSTRACT

An autonomous robot that can follow a black line drawn on a surface with a contrasting colour is called a line-following robot. It is intended to follow the line and move automatically. The robot recognises the line using a variety of optical sensors, so helping the robot maintain its course. Its two sensor array allows for precise and adaptable movement. DC gear motors are used to regulate the robot's wheel movement. The motor driven circuit powers the DC gear motor. The goal of this project is to put the algorithm into practice, regulate the robot's movement by correctly adjusting the control settings, and improve performance. It can be utilised for tour guides in museums, small-scale home applications, industrial automated equipment haulers, and other related uses.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

A line follower robot is an autonomous robot that is programmed to follow a visible line on the ground, typically black on white or vice versa. It detects the position of the line using sensors such as infrared or light sensors, and then adjusts its movement via a control system. Motors power the robot, allowing it to stay on track by adjusting its speed and direction based on sensor inputs. Line follower robots are widely employed in education to teach robotics fundamentals, in competitions to negotiate difficult courses, and in industry to automate material movement. Their simplicity and adaptability make them perfect for learning, experimentation, and a variety of applications.

1.1 WHAT IS LINE FOLLOWER ROBOT

A line follower robot is an autonomous robot that uses sensors, a control system, and motors to walk along a defined path, which is commonly black on a white surface or vice versa. These robots use infrared (IR) or light sensors to determine the line's position and alter their movement accordingly. The control system interprets sensor data to keep the robot aligned, while motors provide power for speed and direction adjustments. Line follower robots are commonly employed in educational settings to teach about sensors and control systems, as well as in robotics competitions where they must navigate complex courses involving turns and intersections.

In industrial environments, they can automate processes such as material transfer along predetermined courses in warehouses or factories.Their flexibility and simplicity make them suitable for both novice and advanced users, allowing for customisation of sensor settings, motor setups, and control algorithms.

1.2 USES OF LINE FOLLOWER ROBOT

Line follower robots have a wide range of applications, including teaching and industrial automation, making them useful tools in a variety of settings. In educational contexts, they are used to teach robotics, electronics, and programming by giving students hands-on experience with sensors, motors, and control systems. Robotics competitions sometimes include line follower events in which robots negotiate intricate courses, requiring participants to build robots that are fast, accurate, and reliable. Line follower robots are used in industrial environments to automate material movement by following specified paths in warehouses and factories. They also contribute to research and development by serving as a testbed for novel sensor technologies, control algorithms, and navigation approaches.

.

CHAPTER 2
SYSTEM DESIGN

2.1 CIRCUIT DESIGN

LINE FOLLOWER

The main components of this project are Arduino UNO, 5 Channel IR Array Sensor module ,L298N motor driver, Bo Motor. The Arduino can draw the power limit of 3.3v and the maximum voltage that can be applied to Vin pins is 12v .

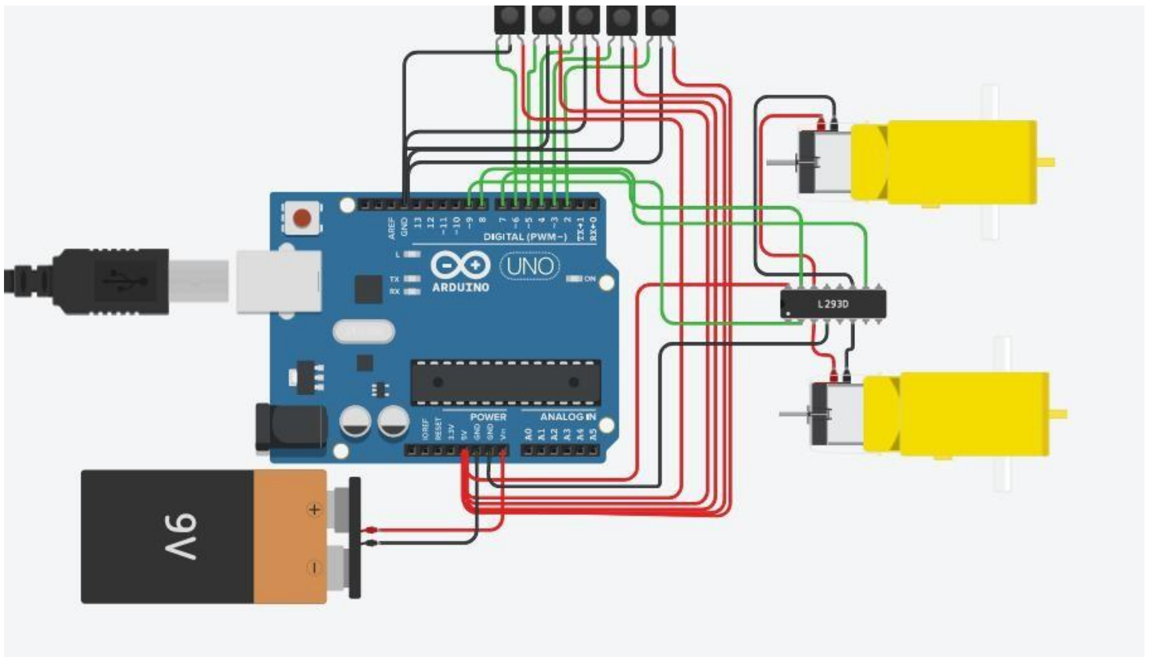

Fig 2.1 Circuit diagram

CHAPTER 3

PROJECT REQUIREMENTS

3.1 HARDWARE COMPONENTS

Arduino UNO

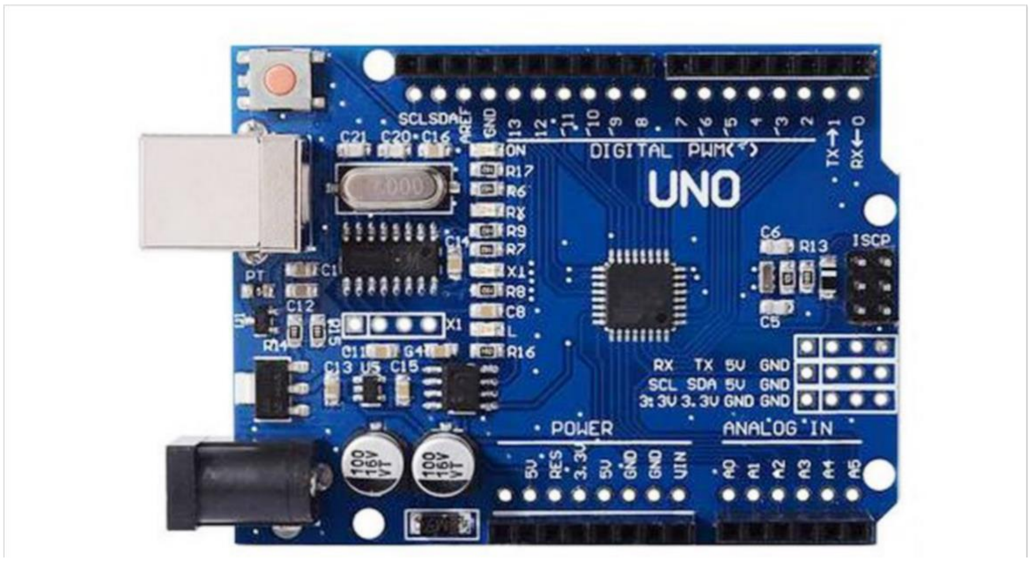

Fig 3.1 Arduino UNO

The Arduino Uno microcontroller is an 8-bit ATmega328P device. It makes use of parts such a voltage regulator, crystal oscillator, serial communication, etc. to assist the microcontroller. There are 14 digital I/O pins on it (six of which can be utilised for PWM). It features an ICSP header, a reset button, a power barrel jack, a USB port, and six distinct analogue input pins. Using a type B USB connector, you may programme this board using the Arduino IDE (Integrated Development Environment) platform. This board can be powered by an external voltage between 7 and 20 volts or by a USB cable.

5 Channel IR Array Sensor

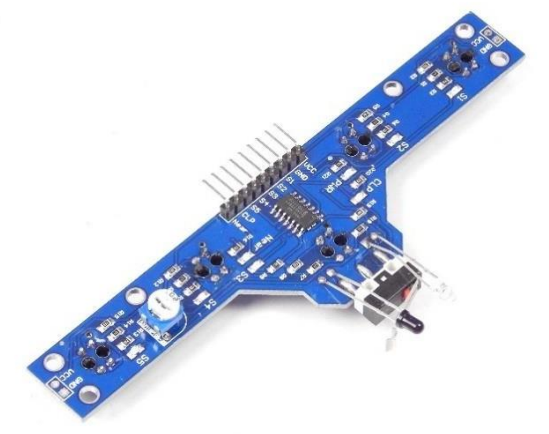

Fig 3.2 IR Array Sensor

A 5-channel infrared array sensor is a type of sensing module that is commonly utilised in robotics and automation applications, such as edge detection and line following. The 5-channel infrared array sensor can distinguish variations in light reflection between surfaces because it is made to detect infrared light. Its five separate infrared (IR) sensor modules are assembled into one, usually in the form of a linear array. Infrared light can be detected and emitted by each sensor unit thanks to the inclusion of an IR LED and a photodiode or phototransistor.

L298N Motor Driver

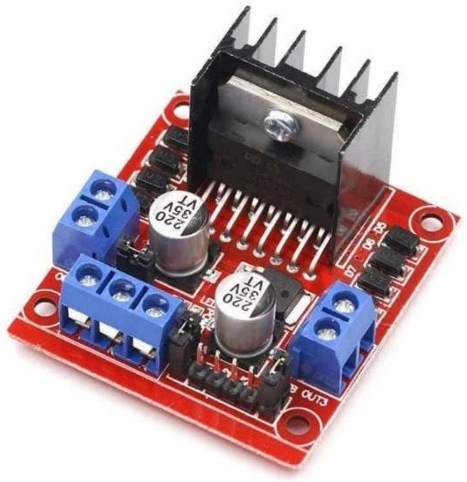

Fig 3.3 L298N Motor Driver

One of the best and simplest methods for controlling DC motors is L298N. The DC motor's spinning direction and speed are managed by the two-channel motor driver. This motor driver module, model number L298N, has a high power output. Motors that are DC and stepper are driven by it. This motor driver is an integrated circuit that includes a 78M05 5V voltage regulator, resistors, capacitors, power LED, and a 5V jumper in addition to an L298N motor driver IC.

Bo Motor

Fig 3.4 Bo motor

3.2 SOFTWARE REQUIREMENTS

ARDUINO IDE for Programming

The Arduino IDE (Integrated Development Environment) is a software application that allows you to programme and upload code to Arduino boards. It is primarily developed to make the development of Arduino-based applications easier.

Robot Operating System(ROS) for Communication and Controlling

ROS is an open-source platform for developing, controlling, and simulating robotic systems. It is not an operating system in the classic sense, but rather a set of software tools, libraries, and protocols that run on top of current operating systems such as Linux. ROS is a modular architecture that supports distributed computing, allowing developers to create complicated robotics applications by dividing them down into smaller, manageable components called nodes. These nodes communicate utilising a publish-subscribe messaging system, which allows for seamless data sharing and coordination.

CHAPTER 4
IMPLEMENTATION

To build a line follower robot with the Robot Operating System (ROS), you must integrate hardware components, configure ROS, and create a line-following algorithm. Begin by configuring the hardware, which includes a robot platform with motors and wheels, an IR array sensor for detecting lines, motor controllers, and a microcontroller to communicate with ROS. Next, install ROS on a compatible platform, often Ubuntu, and set up ROS nodes for sensor input, motor control, and line-following logic. The sensor integration process entails reading data from the IR array and publishing it to a ROS topic, allowing other nodes to use it. For motor control, configure a node to regulate the robot's speed and direction using the line-following algorithm. The line-following algorithm, which uses sensor data to calculate the robot's position in relation to the line and modifies motor speeds to stay on course, is the central component of the implementation. For smoother control, a proportional-integral-derivative (PID) controller is a popular method. After the robot is working properly, you may test it on a track and adjust the control parameters and sensor positions to improve performance. Safety may be improved by adding more features like obstacle detection, and ROS tools like RViz can help visualise sensor data and facilitate troubleshooting.

CHAPTER 5
RESULT AND ANALYSIS

We finally connected the circuit after many mistakes and issues, and everything functions exactly as we had anticipated.

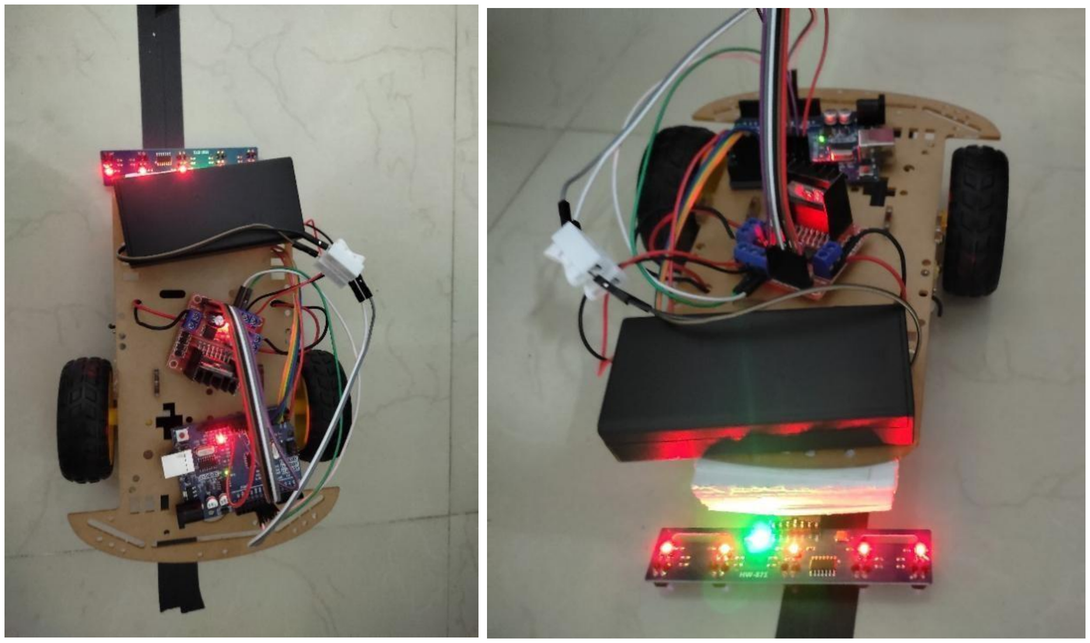

Fig 5.1 Prototype

Throughout testing, the line-following robot performed dependably across a range of line widths and forms. It demonstrated the efficiency of the used algorithm by tracking lines with little variance. Difficulties like abrupt line breaks or different lighting were handled well by means of strong sensor data processing. All things considered, the project succeeded in its goal of autonomously following predetermined routes, creating the framework for future advancements in robotic automation and control.

CHAPTER 6

CONCLUSION

In conclusion, creating a line-following robot with the help of Arduino and ROS has produced encouraging results. The project effectively illustrated the robot's capacity to reliably and precisely move along predetermined routes on its own. Using Arduino for low-level hardware interface and ROS for high-level control, we have developed a flexible platform that can be further improved and customised for a range of robotics and automation applications. In the long run, the project contributes to the development of autonomous robotics technology by providing opportunities for investigating sophisticated algorithms, sensor fusion methods, and integration with other robotic systems.

CHAPTER 7

REFERENCE

1. Alisher K, Alexander K, Alexandr B. Control of the mobile robots with ROS in robotics courses. Procedia Engineering. 2015 Jan 1;100:1475-84.
2. Villa MA, Florez DR, Guzmán LS, Toledo NV, Pulido C. ROS-based human leader and robot follower using a Pioneer 3-DX robot. Revista UIS Ingenierías. 2016;15(2):63-71.
3. Zaman S, Slany W, Steinbauer G. ROS-based mapping, localization and autonomous navigation using a Pioneer 3-DX robot and their relevant issues. In2011 Saudi International Electronics, Communications and Photonics Conference (SIECPC) 2011 Apr 24 (pp. 1-5). IEEE.
4. Joseph L. ROS Robotics projects. Packt Publishing Ltd; 2017 Mar 31.
5. Araújo A, Portugal D, Couceiro MS, Rocha RP. Integrating Arduino-based educational mobile robots in ROS. Journal of Intelligent & Robotic Systems. 2015 Feb;77:281-98.

CHAPTER 8

APPENDIX - CODE SNIPPETS

```
#include <ros.h>
#include <std_msgs/String.h>

ros::NodeHandle nh;

const int irPin1 = 2;
const int irPin2 = 3;
const int irPin3 = 4;
const int irPin4 = 5;
const int irPin5 = 6;

const int ledPin = 13;     // LED pin

std_msgs::String str_msg;
ros::Publisher ir_pub("/ir_state", &str_msg);

void movCallback(const std_msgs::String& cmd_msg) {
  String command = cmd_msg.data;

  if (command == "forward") {
    digitalWrite(7, 0);
    digitalWrite(8, 1);
    digitalWrite(9, 0);
    digitalWrite(10, 1);
  } else if (command == "right") {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 1);
  } else if (command == "left") {
    digitalWrite(7, 0);
    digitalWrite(8, 1);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
  } else if (command == "cright") {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 1);
  } else if (command == "cleft") {
    digitalWrite(7, 0);
    digitalWrite(8, 1);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
  } else if (command == "sright") {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 1);
  } else if (command == "sleft") {
    digitalWrite(7, 0);
    digitalWrite(8, 1);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
  } else if (command == "mright") {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 1);
  } else if (command == "mleft") {
    digitalWrite(7, 0);
    digitalWrite(8, 1);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
  } else if (command == "stop") {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
  } else {
    digitalWrite(7, 0);
    digitalWrite(8, 0);
    digitalWrite(9, 0);
    digitalWrite(10, 0);
    digitalWrite(11, 0);

  }
}

ros::Subscriber<std_msgs::String> sub("mov_state", &movCallback);

void setup() {
  pinMode(irPin1,  INPUT);
  pinMode(irPin2,  INPUT);
  pinMode(irPin3,  INPUT);
  pinMode(irPin4,  INPUT);
  pinMode(irPin5, INPUT);

  pinMode(ledPin, OUTPUT);

  pinMode(7,   OUTPUT);
  pinMode(8,   OUTPUT);
  pinMode(9,   OUTPUT);
  pinMode(10, OUTPUT);

  nh.initNode();
  nh.advertise(ir_pub);
  nh.subscribe(sub);
}

void loop() {
  int irState1 = digitalRead(irPin1);
  int irState2 = digitalRead(irPin2);
  int irState3 = digitalRead(irPin3);
  int irState4 = digitalRead(irPin4);
  int irState5 = digitalRead(irPin5);
   if (irState1 && irState2 && !irState3 && irState4 && irState5 ) {
    str_msg.data = "00100";
  } else if (irState1 && irState2 && irState3 && !irState4 && irState5 ) {
    str_msg.data = "00010";
  } else if (irState1 && !irState2 && irState3 && irState4 && irState5 ) {
    str_msg.data = "01000";
  } else if (!irState1 && !irState2 && !irState3 && !irState4 && !irState5 ) {
    str_msg.data = "11111";
  } else if (!irState1 && irState2 && irState3 && irState4 && irState5 ) {
```

```
  str_msg.data = "10000";
} else if (irState1 && irState2 && irState3 && irState4 && !irState5 ) {
  str_msg.data = "00001";
} else if (!irState1 && !irState2 && irState3 && irState4 && irState5 ) {
  str_msg.data = "11000";
} else if (irState1 && irState2 && irState3 && !irState4 && !irState5 ) {
  str_msg.data = "00011";
} else if (irState1 && !irState2 && !irState3 && irState4 && irState5 ) {
  str_msg.data = "01100";
} else if (irState1 && irState2 && !irState3 && !irState4 && irState5 ) {
  str_msg.data = "00110";
} else {
  str_msg.data = "00000";
}
ir_pub.publish(&str_msg);
nh.spinOnce();

// Wait for incoming messages from ROS
delay(100);
```