LINE FOLLOWING ROBOT REPORT

Higher Diploma in Software Engineer

22.1

Robotic Application Development

Assignment 3

Supervised by Mr. Supun Asanga

GAHDSE212F-018 - Dinal Randika

GAHDSE212F-019 - Piyumi Rasanjali

GAHDSE212F-020 - Sugandika H R

Declaration

"I certify that this project does not incorporate without acknowledgement, any material previously submitted for a Diploma in any institution and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my project report, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and summary to be made available to outside organizations."

…….....

signature

Abstract

This project is aimed at developing a simple line following robot. It is a kind of automatic robot which follows black or white line until the line exits. This robot follows the line using Infra-Red Ray (IR) sensors. These sensors read the line and send it to Arduino and then control the robot moment. Also use a PID control loop to keep the balance of the robot. This robot has a simple structure.

Keywords :
Line Follower, Arduino Uno, QTR sensor, L293D Motor Driver, DC power adapter

Acknowledgement

We would like to thank Mr. Supun Asanga for encouraging and providing us with the project idea. Also, who guided us with the information needed for the project. Also, we offer our sincere thanks to everyone who supported us throughout the completion of this project to make it a success. This project has been a wonderful experience where we have learnt and experienced many beneficial things.

Table Of Contents

CHAPTER 1 : INTRODUCTION

1.1 LINE FOLLOWING ROBOT

Line following robot is basically designed to follow black path in white background. This robot reads lines using a QTR sensor which is installed underneath the front path of the body.

Using these reading data robots control it automatically. Good motor quality and good sensing quality will increase the robot movement performance.

The main component of this robot are,

1. Arduino uno board
2. QTR sensor
3. DC motor

1.2 OBJECTIVE OF ROBOT

The objectives of projects are,

- The robot must be capable of following a line.
- It moves through a dark line on a light background.
- It is capable to Taking various degrees of turns.
- The robot Insensitive to environmental factors such as lighting and noise.

1.3 Methodology

The methodology for building a line-following robot can be broken down into the following steps:

- Research:

Start by researching different types of line-following robots and their components. Look for tutorials, guides, and videos online to understand the basic principles of how they work and the common components used in building them.

- Select Components:

Based on your research, select the components you will need for your robot, including a microcontroller, motor drivers, sensors, and a power source.

- Design:

Draw a schematic or create a circuit diagram of your robot's design. Decide on the placement of the various components, the wiring, and how you will power and program the robot.

- Assemble:

Once you have a design, start assembling the robot. Begin with the chassis and then add the motors, sensors, and microcontroller.

- Program:

Write the code for your robot. Program the microcontroller to read the sensor data and control the motors. You will need to use a programming language, such as C or Python, to do this. We are using the C language for our project.

- Test:

Once the robot is assembled and programmed, test it to see if it follows the line correctly. Adjust the code as needed and test it again until the robot performs as desire

1.4 Limitations

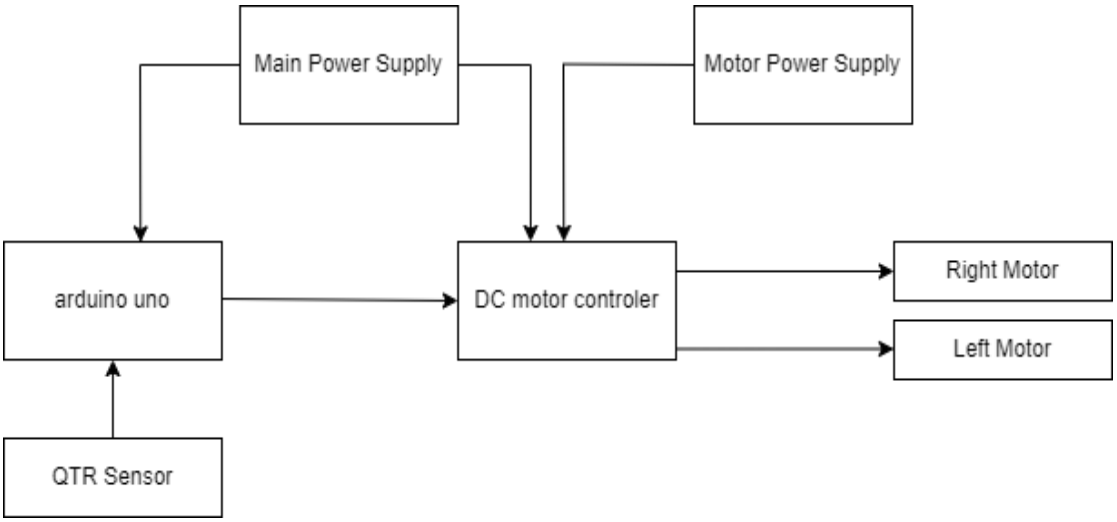
- Line following robots are designed to follow a specific path and may not be capable of performing tasks outside of this narrow scope. They can only operate on a surface with a predefined path.

- Line following robots typically operate within a limited range or area, and may not be suitable for applications that require long-distance travel or exploration.

- Line following robots are not designed to adapt to changes in the environment or to navigate through obstacles.

CHAPTER 2 : TECHNOLOGY

2.1 Block Diagram for line following robot



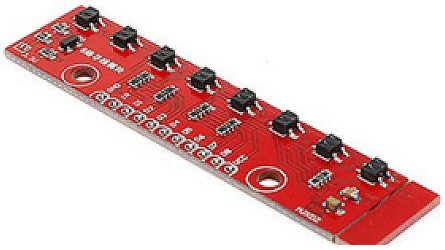
2.2 Hardware Requirement

I. Arduino uno board

Arduino uno is an 8-bit ATmege328P microcontroller.It has mainly 14 digital I/O pins and 6 analog input pins, USB connection.
Arduino IDE is used to write a program and the program inserts to an arduino board using a USB cable which connects the arduino and computer.

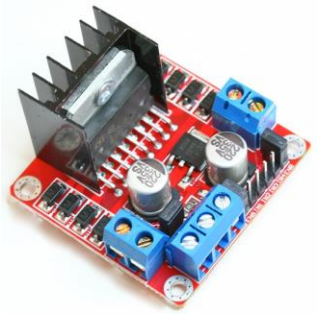
2. QTR sensor

This sensor is used to sense how much infrared light is reflected from a nearby surface. Using this data can move the robot on the line.



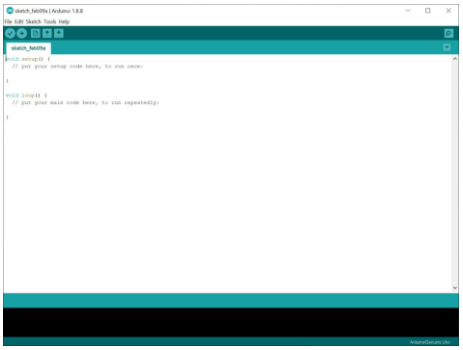
3. Motor Driver

This acts as the interface between arduino and motors which are used to control speed and the movement. A L293D motor driver is used for this robot because it can control 2 DC motors.



2.3Software Required

Arduino 1.8.18 is used for coding and uploading a program to the Arduino Board.



CHAPTER 3 ：ANALYSIS AND DESIGN

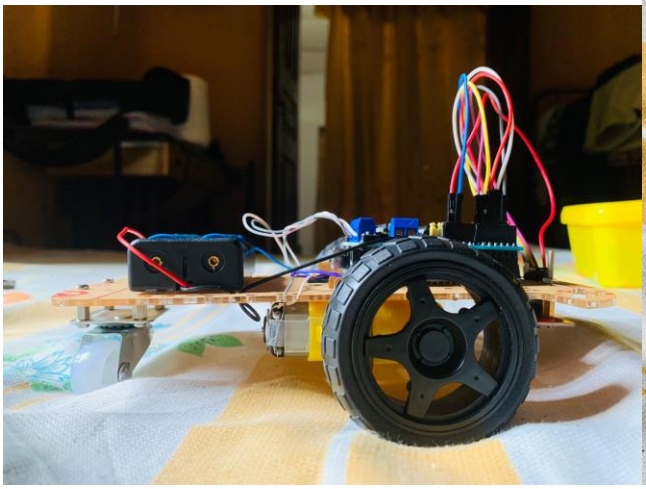
3.1 Process Explain

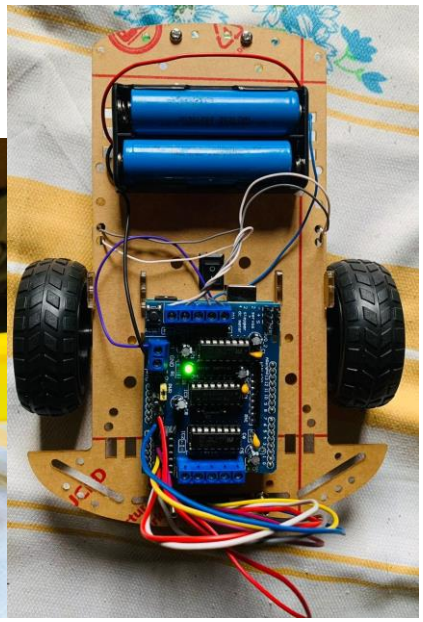
1. Straight Direction
   The robot will move in a straight direction when the input from all qtr sensors is low.

2. Right Curve
   If the left side sensor gives high input, then the right side motor's speed will slow down. The speed is changed according to how many sensors give high input.

3. Left Curve
   If the right side sensor gives high input, then the left side motor's speed will slow down. The speed is changed according to how many sensors give high input3.2 Arduino working logic table

3.2 Picture of Line Following Robot





Future Implementation

In the process of the line following robot, most of the features are identified.

- Use of color sensors.
- Use of a camera for better recognition and precise tracking the path.
- Implement to identify the shortest path.

GANTT CHART

	1 week	2 week	3 week	4 week	5 week	6 week
Research						
Design						
Assemble						
Problem						
Test						

REFERENCES

[1] S. Sahoo and B. Sahu, "Line Follower Robot: A Review," International Journal of Engineering Research & Technology, vol. 3, no. 8, pp. 2011-2015, 2014.
[2] R. N. Mahapatra and P. K. Mishra, "Line Follower Robot: An Overview," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 5, pp.

APPENDICES

```
#include <AFMotor.h>    //Adafruit Motor Shield Library. First you must download and install AFMotor library
#include <QTRSensors.h> //Pololu QTR Sensor Library. First you must download and install QTRSensors library

AF_DCMotor motor1(1, MOTOR12_1KHZ ); //create motor #1 using M1 output on Motor Drive Shield, set to 1kHz PWM frequency
AF_DCMotor motor2(2, MOTOR12_1KHZ ); //create motor #2 using M2 output on Motor Drive Shield, set to 1kHz PWM frequency

#define KP 2 //experiment to determine this, start by something small that just makes your bot follow the line at a slow speed
#define KD 5 //experiment to determine this, slowly increase the speeds and adjust this value. ( Note: Kp < Kd)
#define M1_minumum_speed 150  //minimum speed of the Motor1
#define M2_minumum_speed 150  //minimum speed of the Motor2
#define M1_maksimum_speed 250 //max. speed of the Motor1
#define M2_maksimum_speed 250 //max. speed of the Motor2
#define MIDDLE_SENSOR 4       //number of middle sensor used
#define NUM_SENSORS 5         //number of sensors used
#define TIMEOUT 2500          //waits for 2500 us for sensor outputs to go low
#define EMITTER_PIN 2         //emitterPin is the Arduino digital pin that controls whether the IR LEDs are on or off. Emitter is controlled by digital pin 2
#define DEBUG 0

//sensors 0 through 5 are connected to analog inputs 0 through 5, respectively
QTRSensorsRC qtrrc((unsigned char[]) { A4,A3,A2,A1,A0} ,NUM_SENSORS, TIMEOUT, EMITTER_PIN);

unsigned int sensorValues[NUM_SENSORS];

void setup()
{
delay(1500);
manual_calibration();
set_motors(0,0);
}

int lastError = 0;
int last_proportional = 0;
int integral = 0;
```

```
void loop()
{
unsigned int sensors[5];
int position = qtrrc.readLine(sensors); //get calibrated readings along with the line position, refer to the QTR Sensors Arduino Library for more details on line position.
int error = position - 2000;

int motorSpeed = KP * error + KD * (error - lastError);
lastError = error;

int leftMotorSpeed = M1_minumum_speed + motorSpeed;
int rightMotorSpeed = M2_minumum_speed - motorSpeed;

// set motor speeds using the two motor speed variables above
set_motors(leftMotorSpeed, rightMotorSpeed);
}

void set_motors(int motor1speed, int motor2speed)
{
if (motor1speed > M1_maksimum_speed ) motor1speed = M1_maksimum_speed;
if (motor2speed > M2_maksimum_speed ) motor2speed = M2_maksimum_speed;
if (motor1speed < 0) motor1speed = 0;
if (motor2speed < 0) motor2speed = 0;
motor1.setSpeed(motor1speed);
motor2.setSpeed(motor2speed);
motor1.run(FORWARD);
motor2.run(FORWARD);
}

//calibrate for sometime by sliding the sensors across the line, or you may use auto-calibration instead
void manual_calibration() {

int i;
for (i = 0; i < 250; i++)
{
qtrrc.calibrate(QTR_EMITTERS_ON);
delay(20);
}

if (DEBUG) {
Serial.begin(9600);
for (int i = 0; i < NUM_SENSORS; i++)
{
```

```
Serial.print(qtrrc.calibratedMinimumOn[i]);
Serial.print(' ');
}
Serial.println();

for (int i = 0; i < NUM_SENSORS; i++)
{
Serial.print(qtrrc.calibratedMaximumOn[i]);
Serial.print(' ');
}
Serial.println();
Serial.println();
}
}
```