Line Following Robot

[USING ARDUINO]

A G M S Gunawardhana |AA1345| Praxis 2 | March 4, 2021

Table of Contents

List of Figures & Images

Introduction

Praxis II is a module that is based on designing, testing and developing an automation project. In the module, the task that we have to complete is to make a grid solving line following the robot. The robot should travel through the special 5 points on the grid and reach the destination. Developing the robot expected to learn basic software tools needed for electronic design and manufacturing, equipment used for manufacturing PCBs and casings, documenting, presenting and demonstrating the project and solving problems while doing a project.

The task was given at starting of the semester and the project should be completed within the semester.

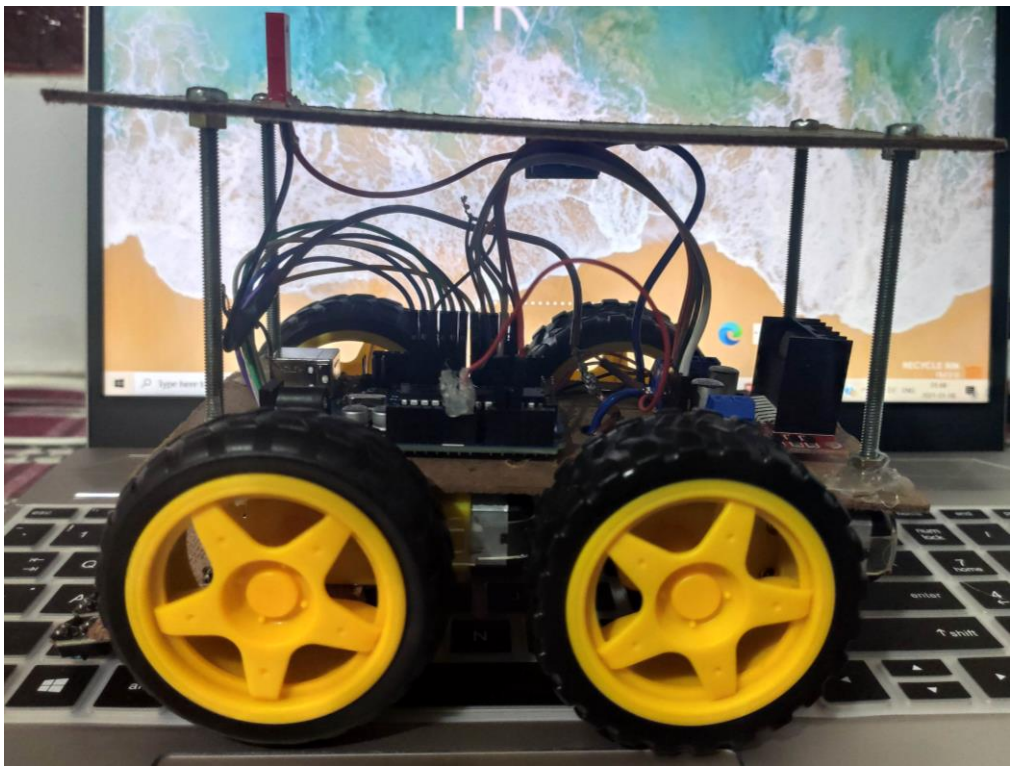

Image:1 – side view of robot

Objectives

The main task is to follow the black line and go through the special 5 points on the provided grid with the shortest path and stop at the black square which situated on the other side of the grid.

To do that task, the robot must move right and left to go smoothly and get various turns too. Finally, when it will detect the destination, it should go a little bit further and stop.

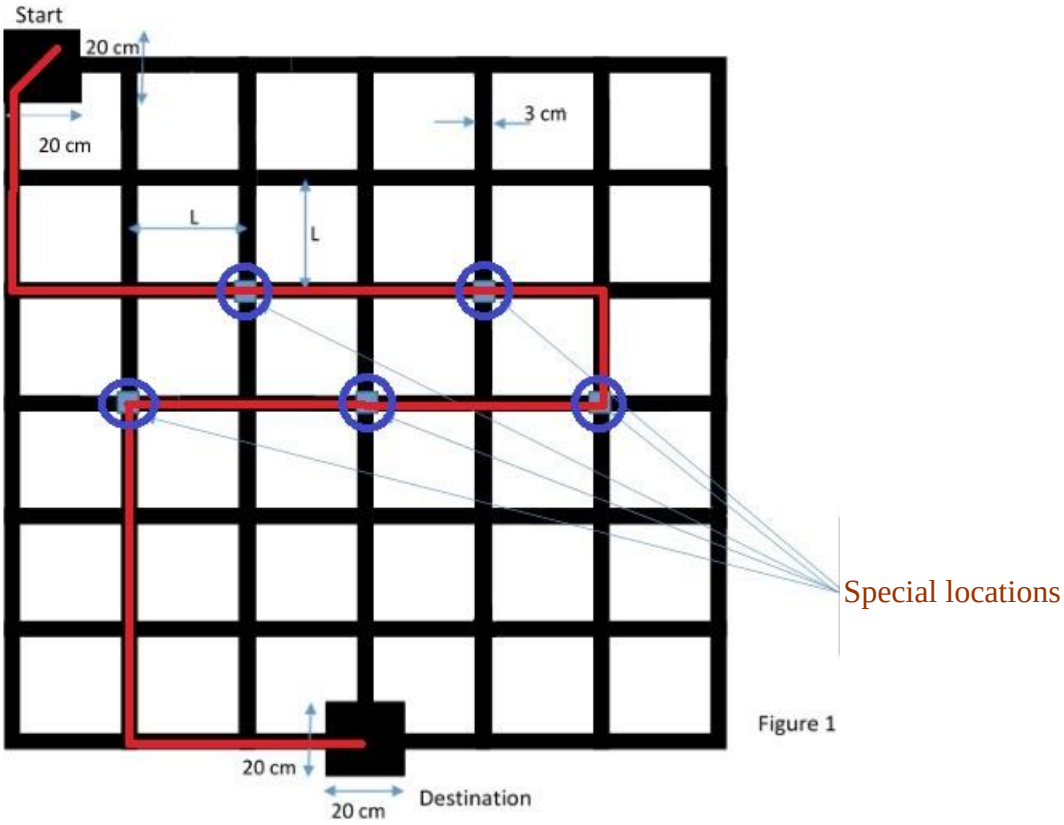

Figure 01: Path of robot

In that path, there are 15 T-junctions and 3 L-junctions. To reach the destination, several 90˚ turns should get by the robot on the path. The path is as the figures -01. (The special locations are highlighted with blue circles. L=20cm)

Methodology

1.Components:

The line following robot was built with the following components,

1. Arduino Uno board
2. Tcrt5000 sensors (6 of them)
3. L298N Motor driver shield
4. DC Motors (4 of them)
5. Wheels (4 of them)
6. Normal 18650 batteries (3 x 3.7V) with holder
7. Switch
8. DC wires
9.Hardboard

2.Circuit design:

The circuit design is as follows.

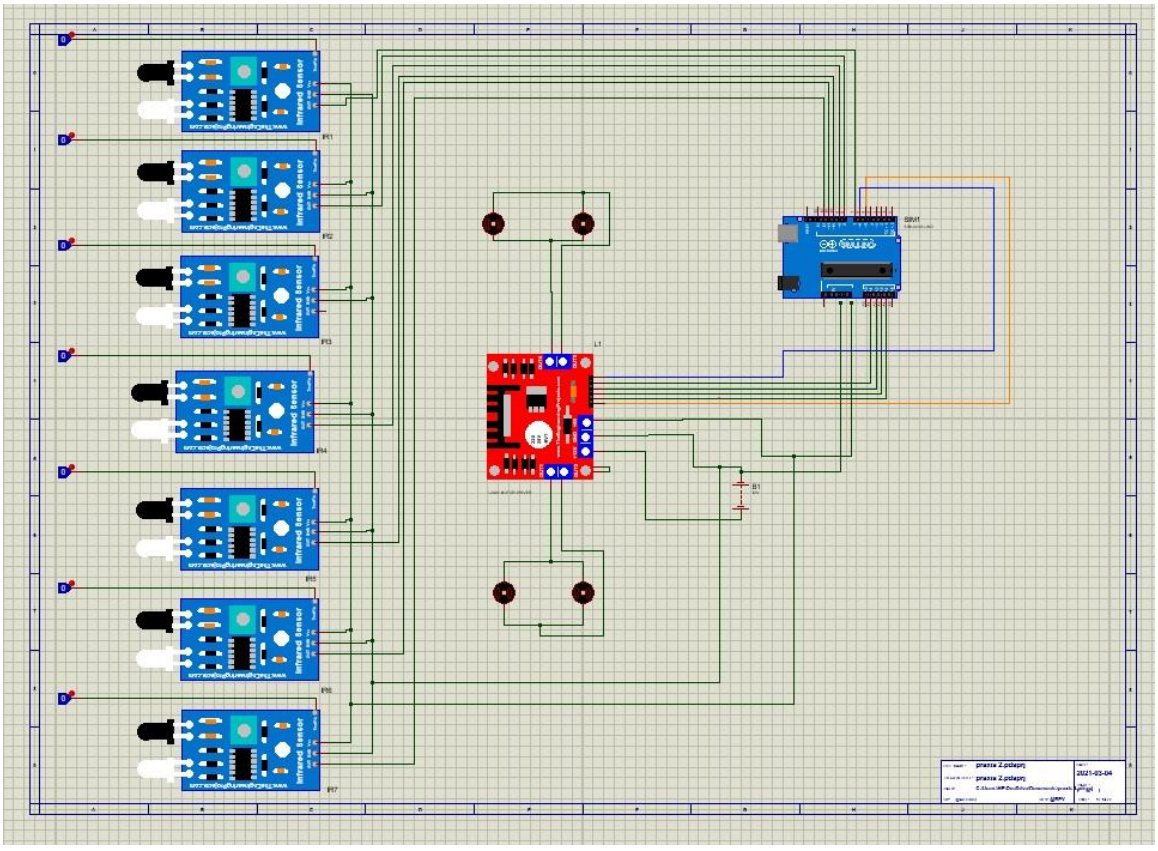

Figure 2: Circuit design

3.Prototype Design:

Design a prototype is the most important part because If our design is not very good, we will face some stability and balancing problems. The design was done in the following image.

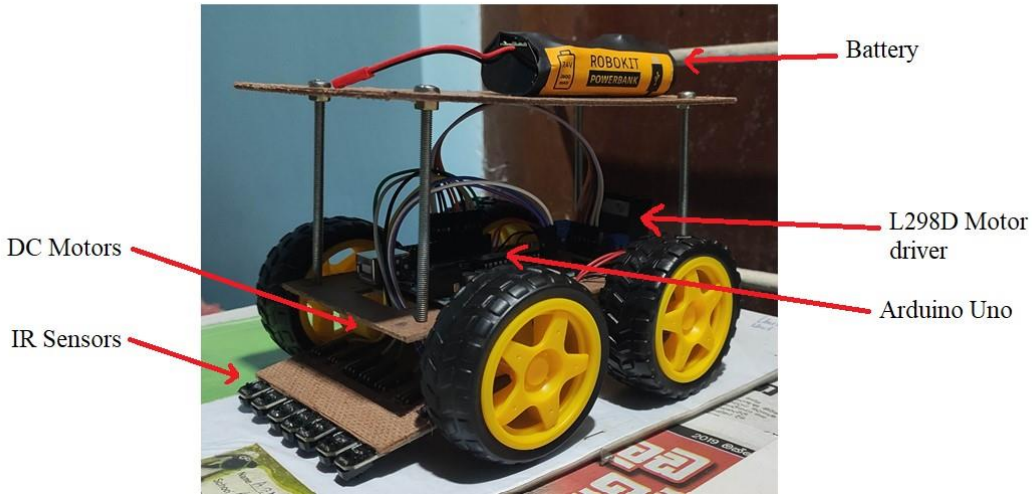

Image:2 – side view of

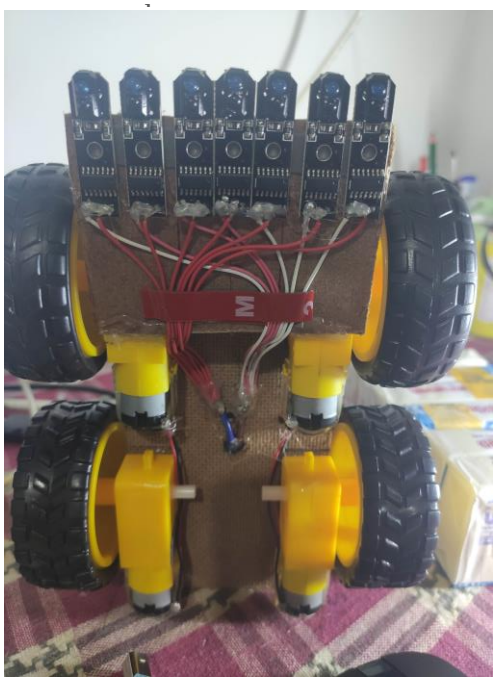

Image:3 – under view

4.Coding and Troubleshooting:

Coding was done with two parts.

- Part I: -

The first part was making the robot follow the black line and get 90˚ turns. After a series of coding, the robot was able to successfully go on a straight line and get turns. Some if statements were used for that. The following images are showing that.

```
/*Forward.....................................................*/
if (IR1read ==1 && IR2read ==1 && IR3read == 0 && IR4read == 0 && IR5read == 0 && IR6read == 1 && IR7read == 1 ) {//ok
  forward();
}
```

Figure 3: Forward function

```
if (IR1read ==1 && IR2read ==1 && IR3read == 1 && IR4read == 0 && IR5read == 0 && IR6read == 0 && IR7read == 0) {//ok
  right();
  delay(20);
}
if (IR1read ==1 && IR2read ==1 && IR3read ==1 && IR4read == 1 && IR5read == 0 && IR6read == 0 && IR7read == 0) {//ok
  right();
  delay(25);
}
if (IR1read ==1 && IR2read ==1 && IR3read == 1 && IR4read == 0 && IR5read == 0 && IR6read == 0 && IR7read == 1) {//ok
  right();
  delay(20);
}
if (IR1read ==1 && IR2read ==1 && IR3read == 1 && IR4read == 1 && IR5read == 1 && IR6read == 0 && IR7read == 0) {//ok
  right();
  delay(20);
}
if (IR1read ==1 && IR2read ==1 && IR3read == 1 && IR4read == 1 && IR5read == 1 && IR6read == 1 && IR7read == 0) {//ok
  right();
```

Figure 4: Turn right function

```
if (IR1read ==0 && IR2read ==0 && IR3read == 0 && IR4read == 0 && IR5read == 0 && IR6read ==0 && IR7read == 1) {//ok
  forward();
  delay(20);
}
if (IR1read ==1 && IR2read ==0 && IR3read == 0 && IR4read == 0 && IR5read == 1&& IR6read == 1&& IR7read == 1) {//ok
  left();
  delay(20);
}
if (IR1read ==1 && IR2read ==1 && IR3read == 0 && IR4read == 0 && IR5read == 1 && IR6read == 1&& IR7read == 1) {//ok
  left();
  delay(20);
}
if (IR1read ==0 && IR2read ==0 && IR3read == 0 && IR4read == 1 && IR5read == 1 && IR6read == 1&& IR7read == 1) {//ok
  left();
  delay(20);
}
if (IR1read == 0 && IR2read == 0 && IR3read == 1 && IR4read == 1 && IR5read == 1 && IR6read == 1 && IR7read == 1) {//ok
  left();
  delay(20);
```

Figure 4: Turn left function

➢ Part II: -

The second part was guiding the robot to follow the black lines and do what was instructed while detecting a junction. A method of counting junctions was used to fulfil that task. The count was uploaded to a pre-define array and stored. According to the count, the various task was appointed through the coding process. So, there were 15 T junctions and 3 L junctions in the grid. The following images are showing the code.

```
void CountingXJunctions(){
  int i=0;
  while(i<10) {
  int count =(Xcount[0]);
  count=count+1;
  Xcount[0]=count;

  Serial.println("  ");
  Serial.print("count :");
  Serial.print(count);
  Serial.println("  ");

  if(count == 1){
    leftTX();
    delay(100);
    forward();
    delay(50);
    Serial.println("Take turn & Go Forwad...X1... ");
    }

 else if(count == 2){
    forward();
    delay(100);
    Serial.println("..... Go Forwad.... ");
```

Figure 5: Junctions Counting function

```
if (IR1read ==0 && IR2read ==0 && IR3read == 0 && IR4read == 0 && IR5read == 0 && IR6read == 0 && IR7read == 0) {//ok
     Serial.print("!!!!!!!!!X Junctinon !!!!!!!!!!");
    CountingXJunctions();
 }

  else {
   breake();
   delay(20);
 }
 if (IR1read ==1 && IR2read ==1 && IR3read == 0 && IR4read == 0 && IR5read == 0 && IR6read == 0 && IR7read == 0) {//ok
  // right();
  // delay(475);
   Serial.print("*** T Junctinon ***");
  CountingLeftTjunctions();

 }
```

Figure 6: Calling Junctions Counting function

For correcting and recovery was done by stopping and reversing the robot. So, the robot is going back and detect the path again if something happened. That code is given in the below figure.

```
/*Stoping Or Back.............................................................. */

if (IR1read ==1 && IR2read ==1 && IR3read == 1 && IR4read == 1 && IR5read == 1&& IR6read == 1 && IR7read == 1) {//ok
   int Bcount =(Xcount[0]);
  Serial.print("!!!!!!!WrongTURN - Back!!!!!!!");
  if(Bcount< 15){
    backward();
    delay(50);
    }
  breake();
  delay(20);
 }
```

Figure 7: Stopping and recovery function

Troubleshooting was mainly done by Proetus software. Errors were checked after the code was uploaded to the robot. A serial monitor was used for that.

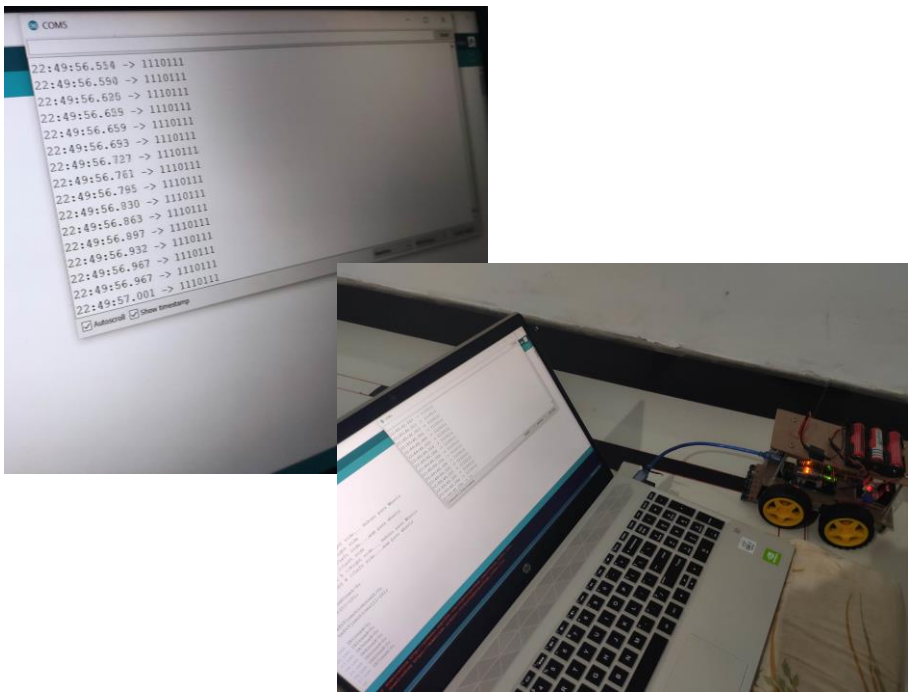

Image:4 – testing path for wrong inputs

Challenges & Risks

While doing the project I met some challenges and roadblocks.

The first one is I met with a problem with the battery. Choosing the correct battery is Challenging because there are various types and features but after a pain full experience, a decide to go with normal 18650 batteries. Those are essay to use and reliable.

While doing the project there is a considerable risk of damaging a motor drive and sensors because if we use the robot for a long time in those parts going to overheat and burn. So that should consider while using the robot.

Making a sensor panel is also challenging because if we make too big a sensor panel it makes it harder to go along the path and also it could reach the limits of robot size. Make a short and compact panel to make it essay to detect junctions and path.

The next big challenge was travelling along the path and make turns at junctions because in every junction sensor panel input are the same. But after a series of coding, I found a way to overcome that problem by using the count. At the same time, I faced a problem and risk with that also. That was counting wrongly, but in the end, changing delays and doing some proper arrangements to the path fix the problems.

Image:5 – Bad battery problem

Further improvement

At this level, I think it’s good to have an LCD screen on the robot for getting outputs like how many junctions pass already and how much to complete, motor speeds and battery life etc.

If we want to go further with prototype design and components, I highly recommend having good motors for the robot becomes current one’s had problems with speeding up and controlling speeds. I think that using a camera and GPS can be upgraded to a commercial level complete product. If we use those things, we can find the path without using IR sensors panel and send those camera footages to a mobile phone or another display. Then we can monitor all the moments form other location too.

Reference

[1]"Robot Research Lab," 24 03 2019. [Online]. Available:

https://www.youtube.com/watch?v=PP4fvBVe3rI&t=2s.

[2] Gberl, "Robot Research Lab," 13 03 2019. [Online]. Available:

http://robotresearchlab.com/2019/03/13/build-a-custom-pid-line-following-robot-from-scratch/.

[3]” How to Make Line Follower Robot Using Arduino,” 13 04 2019. [Online]. Available:

https://www.instructables.com/Line-Follower-Robot-Using-Arduino-2/

[4]’’Arduino-l... Web results Arduino Line Follower Robot - Electronics Hub’’13 04 2019. [Online]. Available:

https://www.electronicshub.org/arduino-line-follower-robot/