FINAL REPORT

NAME- ASHUTOSH JAWANPURIA

ROLL – BTECH/10505/20

BRANCH- ECE (B)

POJECT ASSIGNED- LINE FOLLOWER BOT USING PID

Before proceeding, I would like to extend my sincere gratitude to the ROBOLUTION CLUB for providing me the opportunity to work upon this project.

I would also like to mention Arpita Paul Di, who had been assigned as my mentor.

She had been a great support over the course of this project.

PREREQUISITES REQUIRED:

- Basics of Arduino UNO
- Knowledge of Arduino IDE (Similar to C/C++)
- Knowledge of PID controller
- Knowledge of L298N motor driver

Tasks Accomplished:

- The bot successfully detects black lines via an infrared sensing module which sends a low signal when it receives the IR ray reflected back by a surface is present and a high signal when it does not. Black lines being non reflecting send a high signal.
- There are three sensors present at the front of the bot for left right and the center directions.
- The bot can successfully follow the black lines following the PID control system according to the input given by the sensors.
- The bot stops on detecting a line perpendicular to the line drawn which it was following.

Currently Working On:

- Though the bot follows a line successfully, I am working on a finer PID tuning so as to make the turnings smoother.
- I am also working on making the bot take sharp turns at a greater speed.

Problems Faced:

- The main problems that I faced were hardware related, as the jumper wires turned out to be faulty and the motors and the L298N were not behaving as it should. I was not able to debug the problem initially but finally got to know the problem after manually connecting the wires.
- The positioning and the tuning of the IR sensors needs to be accurate and needed to be corrected frequently.
- The motors had a high RPM and consequently it needed to be driven at a PWM value of 65 , which reduced the range and thus the span of speed over which it can be driven.

Tasks Still to be taken up

- A bot with a greater accuracy can be made using 5 sensors instead of 3 .

Link Of The Video -

Line Follower Bot Using PID

Code Used

```
// IR Sensors

int R_S = A0;     // Right  sensor

int C_S = A1;// center sensor

int L_S= A2;// Left  sensor



// Initial Values of Sensors


// Motor Variables

int ENA = 5;

int MA1 = 8;

int MA2 = 9;

int MB1 = 10;

int MB2 = 11;

int ENB = 6;


//Initial Speed of Motor

int initial_motor_speed = 55;




// PID Constants

float Kp = 30;

float Ki = 0;

float Kd = 15;


float error = 0, P = 0, I = 0, D = 0, PID_value = 0;

float previous_error = 0, previous_I = 0;


int flag = 0;
```

```
void setup()
{
  pinMode(R_S, INPUT);
  pinMode(C_S, INPUT);
  pinMode(L_S, INPUT);


  pinMode(MA1, OUTPUT);
  pinMode(MA2, OUTPUT);
  pinMode(MB1, OUTPUT);
  pinMode(MB2, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);

  Serial.begin(9600);                    //setting serial monitor at a default baund rate of 9600
//  delay(500);
//  Serial.println("Started !!");
//  delay(1000);
}
void loop()
{
  read_ERROR();
  Serial.print("error is");
  Serial.println(error);
  if(error==69)
  {Serial.println("stopping bot");//stopping the bot if all the sensors return high
    stop_bot();
  }
  else if(error==420)
  {Serial.println("slowing bot");//slowing the bot if all the sensors return low
```

```
    analogWrite(ENA,35);
    analogWrite(ENB,34);
   mover();
  }

else
   { calculate_pid();
    motor_control();}

}

void read_ERROR()
{
int LS=digitalRead(L_S);
int CS=digitalRead(C_S);
int RS=digitalRead(R_S);
   if (LS==1&&CS==1&&RS==0)
    error = 2;
  else if ((LS==1)&&(CS==0)&&(RS==0))
    error = 1;
  else if ((LS==0)&&(CS==1)&&(RS==0))
    error = 0;
  else if ((LS==0)&&(CS==0)&&(RS==1))
    error = -1;
  else if ((LS==0)&&(CS==1)&&(RS==1))
    error = -2;
     else if ((LS==1)&&(CS==1)&&(RS==1))
    error = 69;
    else if((LS==0)&&(CS==0)&&(RS==0))
    {
      error=420;
    }

}

void calculate_pid()
{
  P = error;
  I = I + previous_I;
  D = error - previous_error;

  PID_value = (Kp * P) + (Ki * I) + (Kd * D);

  previous_I = I;
  previous_error = error;
}

void motor_control()
{
  // Calculating the effective motor speed:
  Serial.println(PID_value);
  int right_motor_speed  = initial_motor_speed - PID_value;
  int left_motor_speed = initial_motor_speed + PID_value;
//  Serial.println(left_motor_speed);
//Serial.println(right_motor_speed);
//delay(2000);


  right_motor_speed  = constrain(right_motor_speed, 0, 50);
 left_motor_speed = constrain( left_motor_speed, 0,50);
// Serial.println(left_motor_speed);
//Serial.println(right_motor_speed);
```

```
//delay(2000);



  analogWrite(ENB,  right_motor_speed); //Right Motor Speed
  analogWrite(ENA, left_motor_speed ); //Left Motor Speed

  mover();
}

void mover()// this makes the bot go in forward direction with the speed calculated by the pid control system by turning the motors in anticlockwise direction by applying the correct polarity
{

  digitalWrite(MA1, LOW);
  digitalWrite(MA2, HIGH);
  digitalWrite(MB1, LOW);
  digitalWrite(MB2, HIGH);
}
void reverse()// this makes the bot go in reverse direction by turning the motors in anticlockwise direction by applying the correct polarity
{

  digitalWrite(MA1, HIGH);
  digitalWrite(MA2,LOW);
  digitalWrite(MB1, HIGH);
  digitalWrite(MB2, LOW);
}


void stop_bot()// this function is used to stop the bot by turning all the pins to low
{
  digitalWrite(MA1, LOW);

  digitalWrite(MA2, LOW);

  digitalWrite(MB1, LOW);

  digitalWrite(MB2, LOW);

}
```