Line Follower Robot

Aim:

To make a line follower robot using a 3-way IR infrared sensor module.

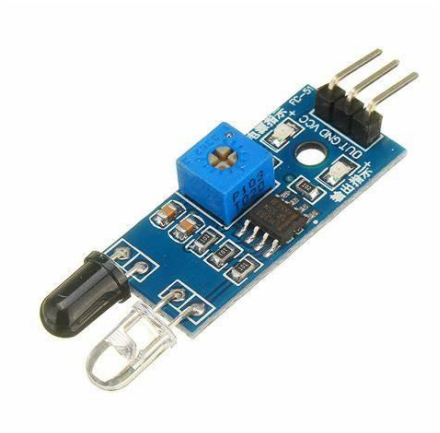
Description:

A line follower robot is a type of autonomous robot that is designed to follow a line or track, typically a dark-colored line on a light-colored surface. To achieve this, the robot is equipped with sensors, often in the form of infrared (IR) sensors, which can detect the contrast between the line and the surrounding surface and make control decisions based on this information.

Components Required:

Arduino UNO:

IR Sensors:



Motor Driver:

Battery:

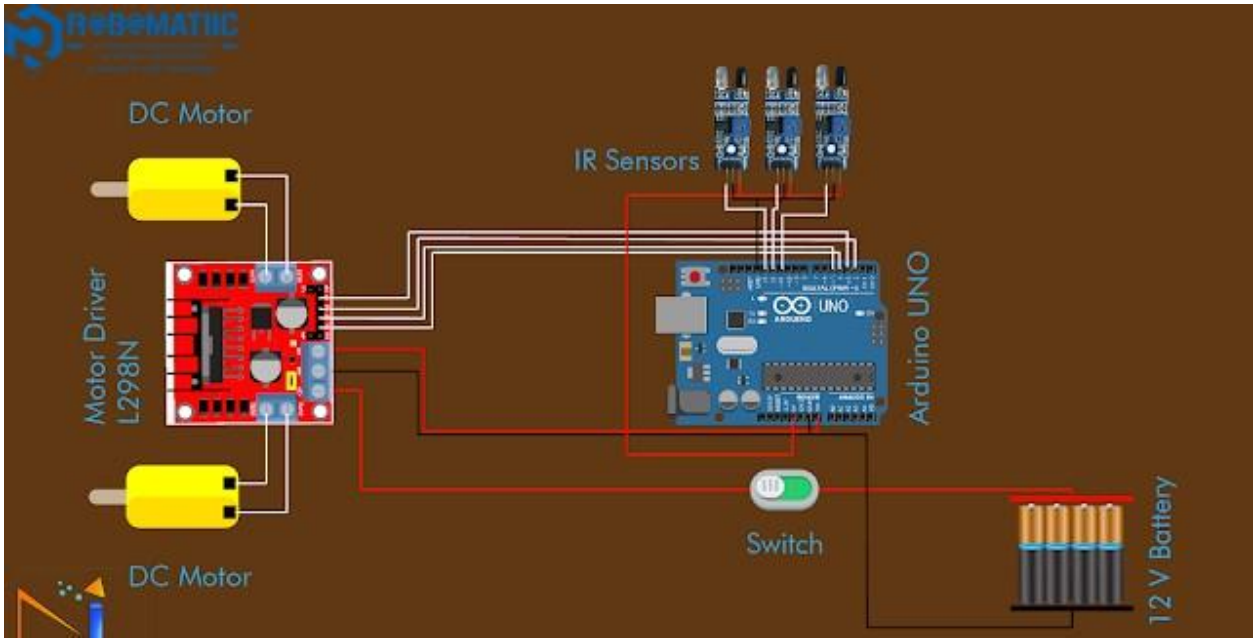
Switch:

Jumper Wires:

DC Motor:

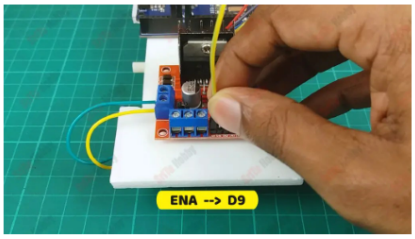
Robot chassis :

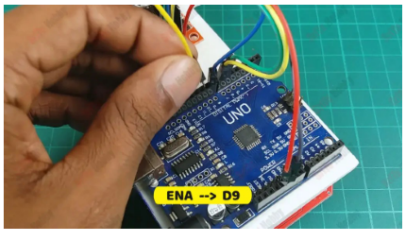
Circuit Diagram of Line Follower:

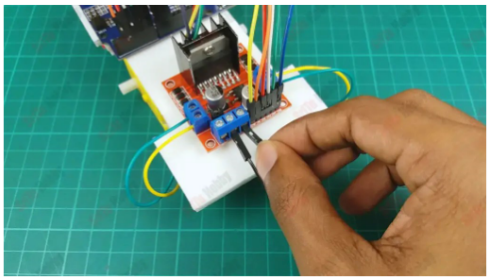


Procedure for Line Follower Robot:

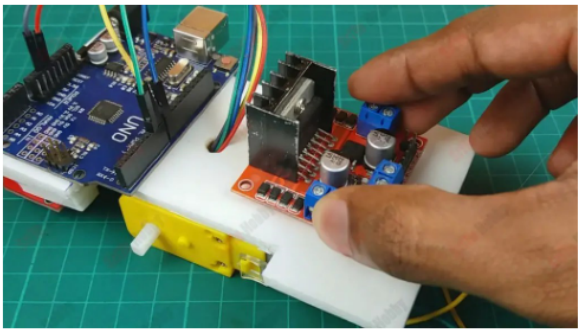
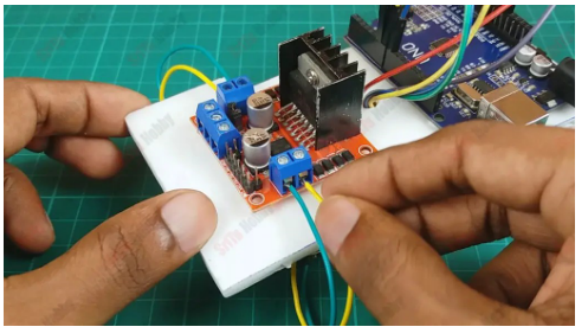
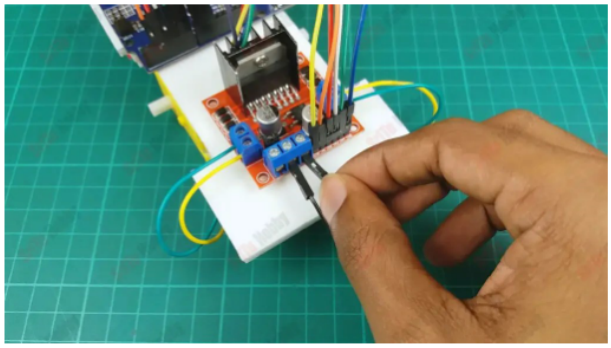
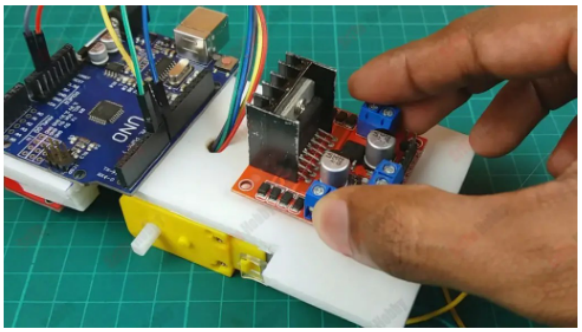
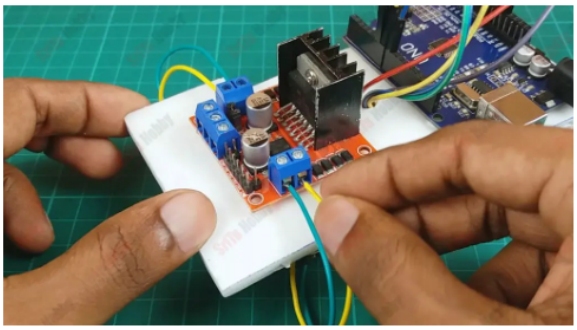
1. Gather the Components: Collect all the required components, including the robot chassis, wheels, motors, IR sensors, motor driver, microcontroller board (e.g., Arduino), power supply, and breadboard.
Motor Driver Connections



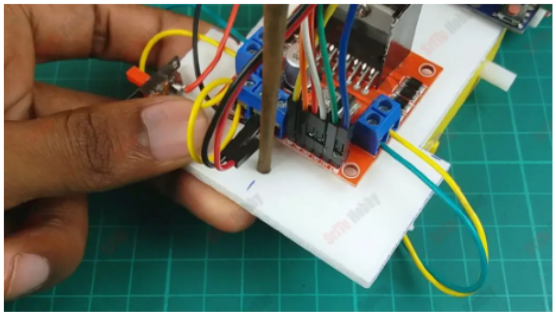


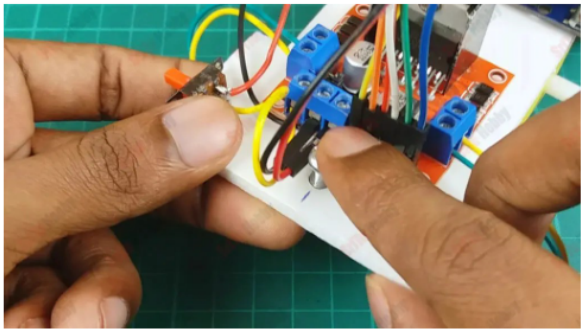


2. Assemble the Hardware: Assemble the robot chassis, attach the motors and wheels, and position the IR sensors on the front of the robot at appropriate distances from each other.

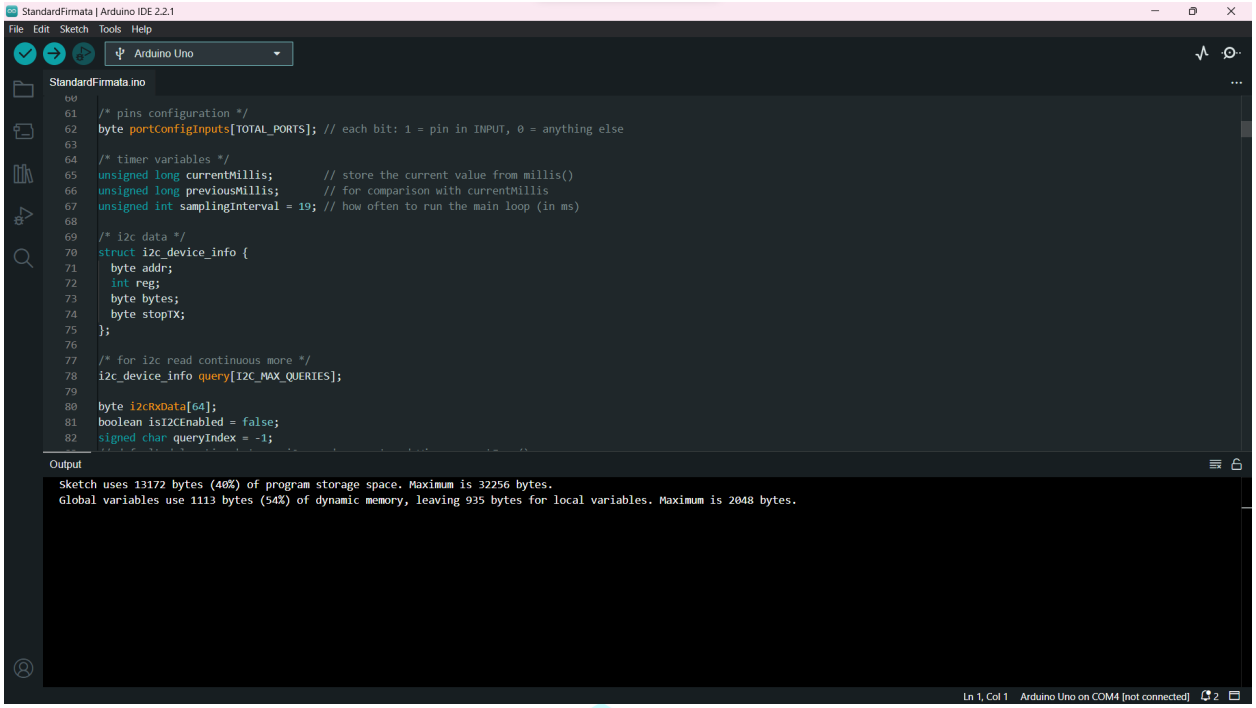






3. Make Necessary Electrical Connections: Connect the IR sensors and motors to the microcontroller using jumper wires, and make the connections as per your code. In your code, you've defined pin numbers for left, center, and right sensors, as well as motor control pins.

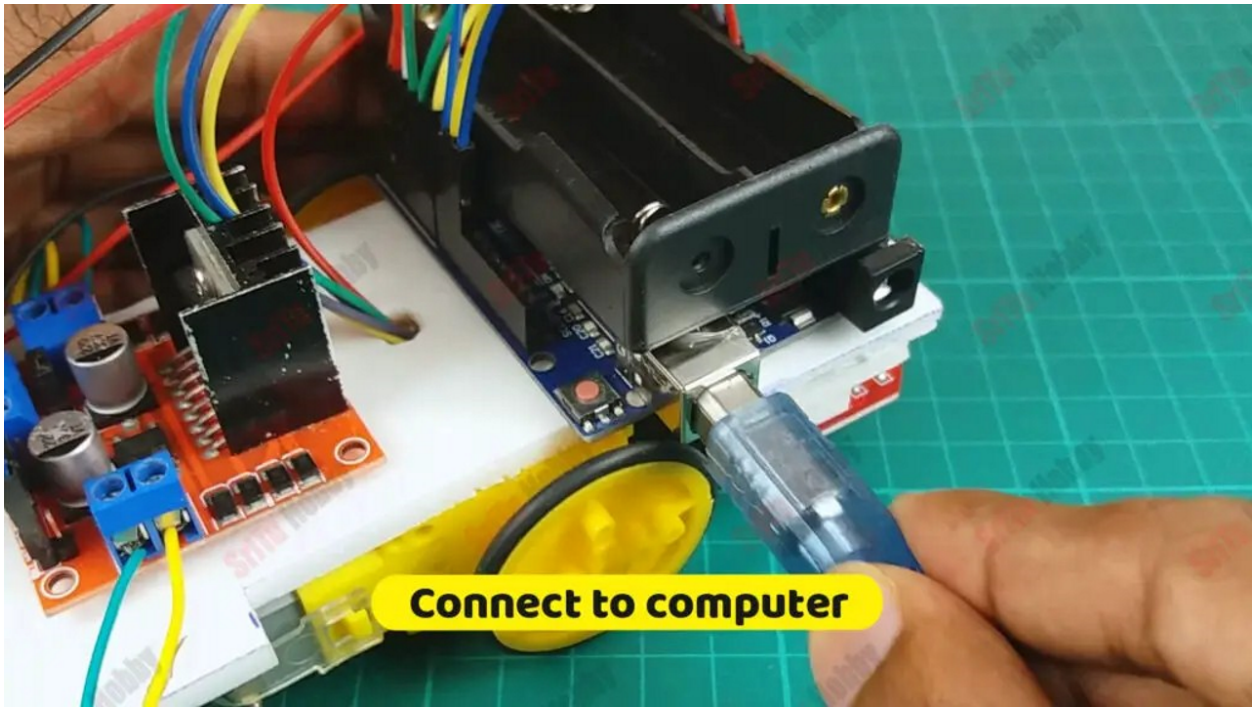




4. Load the Code: Copy and paste your Arduino code into the Arduino IDE or platform of your choice. Upload the code to the microcontroller.



5. Set Up the Power Supply: Connect the power supply (batteries or other suitable source) to the appropriate pins on the microcontroller and motor driver. Ensure that the voltage levels are correct for your components.



6. Adjust the Sensor Placement: Position the robot on a track with a line to follow. Adjust the IR sensor positions so that they can detect the line correctly. You may need to fine-tune the placement.

7. Power On the Robot: Turn on the power supply for your robot and ensure that it's receiving power. The microcontroller should start running your code.

8. Calibrate the Code: If necessary, calibrate your code to match the specific conditions of the track. Adjust the speed and thresholds in your code for optimal line following.

9. Observe Robot Behavior: Observe how the robot follows the line. It should follow the track based on the sensor readings and execute the appropriate motor control functions.

10. Troubleshoot and Refine: If the robot doesn't follow the line correctly, make adjustments to the sensor positions, code, or the track itself. Continue to test and refine the system until it successfully follows the line as expected.

To Write a Coding:

Here is a step-by-step procedure for setting up a line-following robot using the code you provided and making the necessary connections:
Firstly,sensor pins and motor driver pins are defined. Later, a variable is created for the speed of this robot.

```
#define left 6
#define center 5
#define right 4
//motor one
#define ENA 7
#define IN1 8
#define IN2 9
//motor two
#define ENB 12
#define IN3 10
#define IN4 11
```

```
int Speed = 120; // speed of this robot
```

Setup Function (`void setup()`):
This function is executed once when the Arduino board is powered on or reset.

```
void setup() {
  Serial.begin(9600);
  pinMode(left, INPUT);
  pinMode(center, INPUT);
  pinMode(right, INPUT);
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENB, OUTPUT);
}
```

Explanation:
It sets up serial communication at a baud rate of 9600, allowing you to communicate with the Arduino via a serial monitor.
It configures the `left`, `center`, and `right` pins as input pins for the sensors.
It configures the motor control pins `ENA`, `IN1`, `IN2`, `IN3`, `IN4`, and `ENB` as output pins.

Loop Function (`void loop()`):
The `loop` function is where the main logic of your program runs in a continuous loop.

```
void loop() {
  bool leftV = digitalRead(left);
  bool centerV = digitalRead(center);
  bool rightV = digitalRead(right);
  Serial.println(rightV);
  if (leftV == 0 && centerV == 1 && rightV == 0) {
    carforward();
    Serial.println("forward");
```

```
  } else if (leftV == 0 && centerV == 0 && rightV == 0) {
    carStop();
    Serial.println("Stop");
  } else if (leftV == 1 && centerV == 1 && rightV == 1) {
    carStop();
    Serial.println("Stop");
  } else if (leftV == 1 && centerV == 1 && rightV == 0) {
    carturnleft();
    Serial.println("left0");
  } else if (leftV == 0 && centerV == 1 && rightV == 1) {
    carturnright();
    Serial.println("right0");
  } else if (leftV == 1 && centerV == 0 && rightV == 0) {
    carturnleft();
    Serial.println("left1");
  } else if (leftV == 0 && centerV == 0 && rightV == 1) {
    carturnright();
    Serial.println("right1");
  }
}
```

Explanation:
It reads the values from the `left`, `center`, and `right` sensors and stores them in the `leftV`, `centerV`, and `rightV` variables.
It prints the value of `rightV` to the serial monitor.
It uses a series of if-else statements to determine the robot's action based on sensor input. The actions include moving forward, stopping, turning left, and turning right.

Motor Control Functions (`carforward`, `carturnleft`, `carturnright`, `carStop`):
These functions control the motors to perform various actions. Each function configures the motor control pins and sets the motor speed.

carforward: Moves the robot forward.
carturnleft: Turns the robot left.
carturnright: Turns the robot right.
carStop: Stops the robot by setting all motor control pins to low.

The analogWrite function is used to control the motor speed by sending a PWM (Pulse Width Modulatin) signal to the enable pins `ENA` and `ENB`.

The digitalWrite function is used to set the direction of the motors by manipulating the input pins `IN1`, `IN2`, `IN3`, and `IN4`.

The Serial.println statements provide feedback on the actions being taken and the sensor values being read.

Arduino Coding:

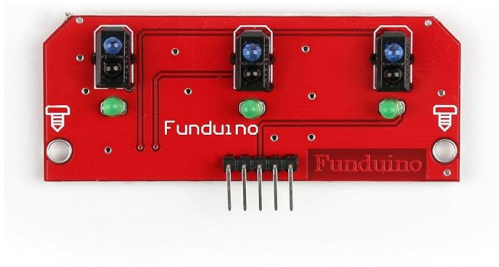
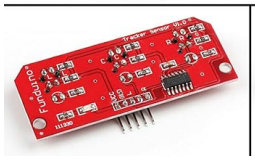
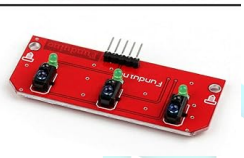
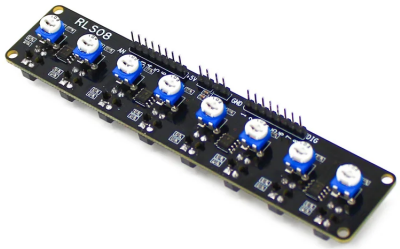
```
#define left 6
#define center 5
#define right 4
//motor one
#define ENA 7
#define IN1 8
#define IN2 9
//motor two
#define ENB 12
#define IN3 10
#define IN4 11
int Speed = 120; // speed of this robot
void setup() {
  Serial.begin(9600);
  pinMode(left, INPUT);
  pinMode(center, INPUT);
  pinMode(right, INPUT);
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENB, OUTPUT);
}
void loop() {
  bool leftV = digitalRead(left);
```

```
  bool centerV = digitalRead(center);
  bool rightV = digitalRead(right);
  Serial.println(rightV);
  if (leftV == 0 && centerV == 1 && rightV == 0) {
    carforward();
    Serial.println("forward");
  } else if (leftV == 0 && centerV == 0 && rightV == 0) {
    carStop();
    Serial.println("Stop");
  } else if (leftV == 1 && centerV == 1 && rightV == 1) {
    carStop();
    Serial.println("Stop");
  } else if (leftV == 1 && centerV == 1 && rightV == 0) {
    carturnleft();
    Serial.println("left0");
  } else if (leftV == 0 && centerV == 1 && rightV == 1) {
    carturnright();
    Serial.println("right0");
  } else if (leftV == 1 && centerV == 0 && rightV == 0) {
    carturnleft();
    Serial.println("left1");
  } else if (leftV == 0 && centerV == 0 && rightV == 1) {
    carturnright();
    Serial.println("right1");
  }
}
void carforward() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
void carturnleft() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
```

```
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
void carturnright() {
  analogWrite(ENA, Speed);
  analogWrite(ENB, Speed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
}
void carStop() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
}
```

What is the Difference between 3 IR sensor Array and 8 IR sensor Array?

Sl.No	3 IR Sensor Array	8 IR Sensor Array
1	A 3 or 5 IR sensor module typically includes three or five infrared sensors on a single board or module.	An 8 IR sensor array, as the name suggests, consists of eight infrared sensors, often arranged in a linear or circular pattern.
2.	These modules are often used in line follower robots to detect lines or tracks on the ground. They are suitable for basic line following applications where the robot needs to stay on a simple path.	These arrays are more versatile and can be used for various applications beyond simple line following. They can provide more data points for detecting multiple lines, obstacles, or objects around the robot.
3.	They are relatively simple to use and are suitable for beginners or educational purposes.	The additional sensors provide the capability to detect and respond to a wider range of track configurations or obstacles.

4.	These modules have limited coverage, so they may not be as effective for more complex tasks or wider tracks.	The larger number of sensors can lead to more precise tracking and decision-making, especially in more complex environments.
5.	3 or 5 IR sensor modules are generally more affordable than larger arrays.	8 IR sensor arrays are generally more expensive than 3 or 5 sensor modules due to the increased number of sensors.
Images	  	

6 IR Sensors Array Coding:

```
int ledPin = 6;
void setup() {
Serial.begin(9600);
pinMode (10, OUTPUT);
pinMode (11, OUTPUT);
pinMode (12, OUTPUT);
pinMode (13, OUTPUT);
pinMode (6, OUTPUT);
pinMode (7, OUTPUT);
pinMode (A0, INPUT);
pinMode (A1, INPUT);
pinMode (A2, INPUT);
pinMode (A3, INPUT);
```

```
pinMode (A4, INPUT);
pinMode (A5, INPUT);
pinMode(ledPin, OUTPUT);
}
void loop() {
  int s1 = digitalRead(A0);
  int s2 = digitalRead(A1);
  int s3 = digitalRead(A2);
  int s4 = digitalRead(A3);
  int s5 = digitalRead(A4);
  int s6 = digitalRead(A5);
  Serial.print(s1);
  Serial.print(" -- ");
  Serial.print(s2);
  Serial.print(" -- ");
  Serial.print(s3);
  Serial.print(" -- ");
  Serial.print(s4);
  Serial.print(" -- ");
  Serial.print(s5);
  Serial.print(" -- ");
  Serial.print(s6);
   Serial.println("");
  Serial.println("");
  delay(100);
if ( (s1==1) && (s2==1) && (s3==1) && (s4==1) && (s5==1) && (s6==1) ) {
forward();
}
else if ( (s1==1) && (s2==1) && (s3==0) && (s4==0) && (s5==1) && (s6==1) ) {
forward();// for thicker lines
}
else if ( (s1==0) && (s2==0) && (s3==0) && (s4==0) && (s5==0) && (s6==0) ) {
stop();
}
else if ( (s1==0) && (s2==0) && (s3==0) && (s4==1) && (s5==0) && (s6==0) ) {
 right();
}
else if ( (s1==0) && (s2==0) && (s3==1) && (s4==0) && (s5==0) && (s6==0) ) {
left();
}
else if ( (s1==0) && (s2==1) && (s3==1) && (s4==0) && (s5==0) && (s6==0) ) {
sharpleft();
}
else if ( (s1==0) && (s2==0) && (s3==0) && (s4==1) && (s5==1) && (s6==0) ) {
```

```
sharpright();
}
else if ( (s1==1) && (s2==1) && (s3==1) && (s4==0) && (s5==0) && (s6==0) ) {
sharpleft();
}
else if ( (s1==0) && (s2==0) && (s3==0) && (s4==1) && (s5==1) && (s6==1) ) {
sharpright();
}
else{
forward();
}
}
void forward()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, HIGH);
digitalWrite(11, LOW);
digitalWrite(12, HIGH);
digitalWrite(13, LOW);
digitalWrite(ledPin, HIGH);
Serial.println(" Moving Forward");
}
void backward()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, LOW);
digitalWrite(11, HIGH);
digitalWrite(12, LOW);
digitalWrite(13, HIGH);
digitalWrite(ledPin, HIGH);
Serial.println("Moving Backward");
}
void stop()
{
digitalWrite(6, LOW);
digitalWrite(7, LOW);
digitalWrite(10, LOW);
digitalWrite(11, LOW);
digitalWrite(12, LOW);
digitalWrite(13, LOW);
digitalWrite(ledPin, LOW);
Serial.println("Stop");
```

```
}
void left()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, HIGH);
digitalWrite(11, LOW);
digitalWrite(12, LOW);
digitalWrite(13, LOW);
digitalWrite(ledPin, HIGH);
Serial.println(" Rotate Left");
}
void right()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, LOW);
digitalWrite(11, LOW);
digitalWrite(12, HIGH);
digitalWrite(13, LOW);
digitalWrite(ledPin, HIGH);
Serial.println(" Rotate Right");
}
void sharpleft()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, HIGH);
digitalWrite(11, LOW);
digitalWrite(12, LOW);
digitalWrite(13, HIGH);
digitalWrite(ledPin, HIGH);
Serial.println(" Rotate Left");
}
void sharpright()
{
digitalWrite(6, HIGH);
digitalWrite(7, HIGH);
digitalWrite(10, LOW);
digitalWrite(11, HIGH);
digitalWrite(12, HIGH);
digitalWrite(13, LOW);
digitalWrite(ledPin, HIGH);
Serial.println(" Rotate Right");
```

```
}
```

8 IR Sensors Array Module Code:

```
/* Define motor controll inputs */
const int motorRPin1 = 2; // signal pin 1 for the right motor, connect to IN1
const int motorRPin2 = 3;  // signal pin 2 for the right motor, connect to IN2
const int motorREnable = 5; // enable pin for the right motor (PWM enabled)
const int motorLPin1 = 4; // signal pin 1 for the left motor, connect to IN3
const int motorLPin2 = 7; // signal pin 2 for the left motor, connect to IN4
const int motorLEnable = 6; // enable pin for the left motor (PWM enabled)
const int irPins[8] = {A0, A1, A2, A3, A4, A5 , 12 , 11};
int irSensorDigital[8] = {0,0,0,0,0,0,0,0};
int i,j = 0;
int d = 1000;
int irSensors = B00000000;
int motorLSpeed = 255;
int motorRSpeed = 255;
int error = 140;   // 145 best 200  //  normal 255  // mad 0
void setup() {


  Serial.begin(9600);


  pinMode(motorLPin1,OUTPUT);
  pinMode(motorLPin2,OUTPUT);
  pinMode(motorLEnable,OUTPUT);

  pinMode(motorRPin1,OUTPUT);
  pinMode(motorRPin2,OUTPUT);
  pinMode(motorREnable,OUTPUT);

  /* Set-up IR sensor pins as input */
  for (int i = 0; i <= 7; i++)
  {pinMode(irPins[i], INPUT);}

}
void loop() {
    scanD();
    check();
```

```
}

void check( )
{
    switch (irSensors) {

    case B00000000: // on white paper
    rightS();
    break;

    case B10000000: // leftmost sensor on the line
    rightS();
    break;

    case B01000000:
    rightS();
    break;

    case B00100000:
    rightS();
    break;

    case B00010000:
    rightS();
    break;

    case B00001000:
    leftS();
    break;
    case B00000100:
    leftS();
    break;

    case B00000010:
    leftS();
    break;

    case B00000001:
    leftS();
    break;

    case B11000000:
    rightS();
    break;
```

```
case B01100000:
rightS();
break;
case B00110000:
rightS();
break;

case B00011000:
go();
break;
case B00001100:
leftS();
break;
case B00000110:
leftS();
break;

case B00000011:
leftS();
break;

case B11100000:
rightS();
break;

case B01110000:
rightS();
break;

case B00111000:
rightS();
break;

case B00011100:
leftS();
break;

case B00001110:
leftS();
break;

case B00000111:
leftS();
```

```
break;

case B11110000:
rightS();
break;

case B01111000:
rightS();
break;

case B00111100:
go();
break;

case B00011110:
leftS();
break;

case B00001111:
leftS();
break;

case B11111000:
rightS();
break;

case B01111100:
rightS();
break;

case B00111110:
leftS();
break;

case B00011111:
leftS();
break;

case B11111100:
rightS();
break;

case B01111110:
go();
```

```
    break;

    case B00111111:
    leftS();
    break;


    case B11111110:
    rightS();
    break;

    case B01111111:
    leftS();
    break;

    case B11111111:
     go();

    break;


    default:

    Serial.print("Unhandled case: ");

  }

}




void rightS()
{
    Serial.println("                    right motor forward (spin)");
    analogWrite(motorREnable, motorRSpeed);
    digitalWrite(motorRPin1, HIGH);
    digitalWrite(motorRPin2, LOW);

    analogWrite(motorLEnable, motorLSpeed-error);
    digitalWrite(motorLPin1, LOW);
    digitalWrite(motorLPin2,HIGH);

}
```

```
void leftS()  //turn left
{
    Serial.println("                    left  motor forward (spin)");
    analogWrite(motorREnable, motorRSpeed-error);
    digitalWrite(motorRPin1, LOW);
    digitalWrite(motorRPin2, HIGH);

    analogWrite(motorLEnable, motorLSpeed);
    digitalWrite(motorLPin1, HIGH);
    digitalWrite(motorLPin2, LOW);

}
void go()
{
   Serial.println("                    forward ");
    analogWrite(motorREnable, motorRSpeed);
    digitalWrite(motorRPin1, HIGH);
    digitalWrite(motorRPin2, LOW);

    analogWrite(motorLEnable, motorLSpeed);
    digitalWrite(motorLPin1, HIGH);
    digitalWrite(motorLPin2, LOW);

}
void stopme()
{
    Serial.println("                    stop");
    analogWrite(motorREnable, motorRSpeed);
    digitalWrite(motorRPin1, LOW);
    digitalWrite(motorRPin2, LOW);

    analogWrite(motorLEnable, motorLSpeed);
    digitalWrite(motorLPin1, LOW);
    digitalWrite(motorLPin2, LOW);

}



void scanD()
{
  for ( byte count = 0; count < 8;count++ )
{
```

```
  bitWrite(irSensors, count, !digitalRead( irPins[count] ));
}
}
```