Line-Guided Maze Solver Robot

Introduction

A **maze solver robot** is a smart robot that can move on its own to find the way out of a maze. It follows a black or white line using special sensors that can "see" the path. The robot's brain (microcontroller) reads the sensor signals and decides whether to go straight, turn left, or turn right until it reaches the end.

Why maze following is important:

- **Robotics Competitions** – Used in many school and college robot contests.
- **Rescue Missions** – Can help robots find safe paths in disaster areas.
- **Industry** – Similar technology is used in warehouse robots to carry goods.
- **Education** – Great for learning about sensors, coding, and electronics.

This simple idea is the base for many advanced navigation systems used in real robots today.

Kit Contents

- Arduino Nano
- Digital IR sensor (x5)
- Potentiometer (3362)
- Switch button (MPB04)
- Motor Driver (L293D) ICs
- DC Gear Motors (x4)
- Capacitors and Diodes
- Power supply
- Mounted Pcb
- LM358IC
- 78M05IC

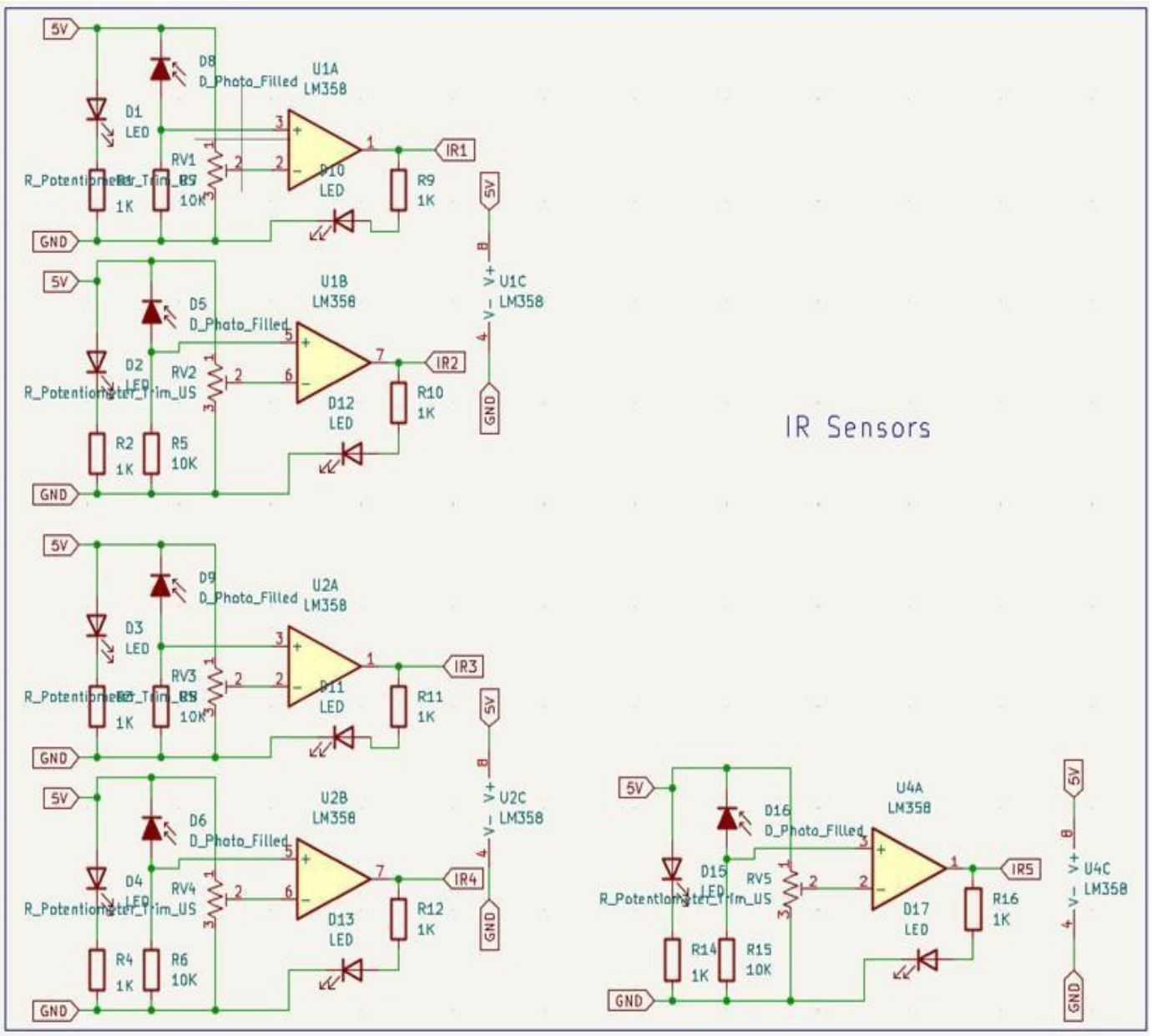
Requirements

- Got it. Here's your updated list in simple format:
- USB Cable (Type A to B)
- Screwdriver Set
- Computer with Arduino IDE installed
- CH340 Driver

Circuit Diagram

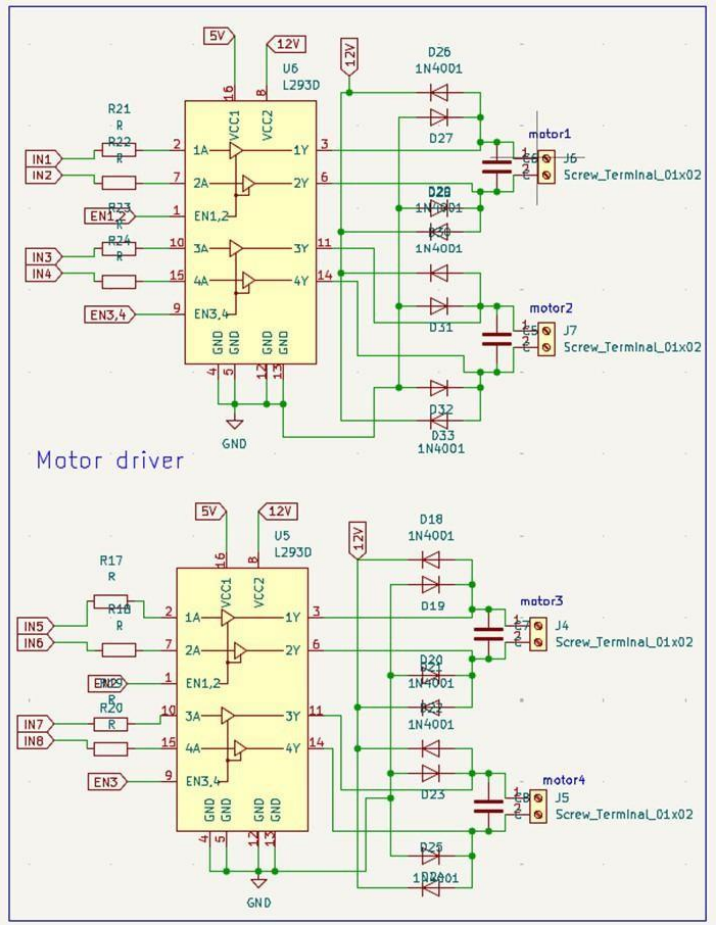
Schematic

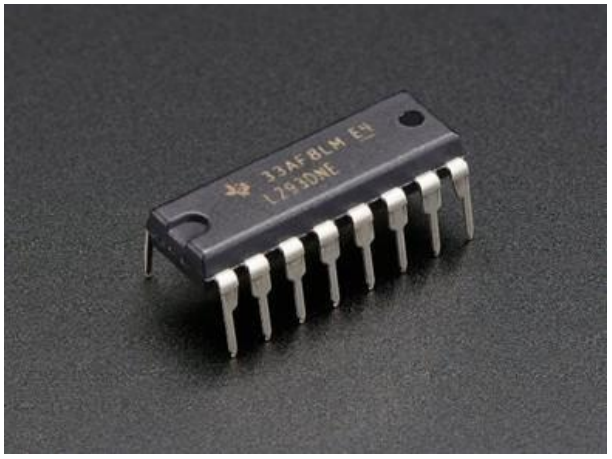
IR Sensors



The IR sensor modules detect lines and obstacles using an IR LED and a photodiode. The signal is processed by an LM358 amplifier, and sensitivity is adjusted with a potentiometer. Each module outputs a digital signal (IR1–IR5): low for a black surface/obstacle, high for a white surface/no obstacle.

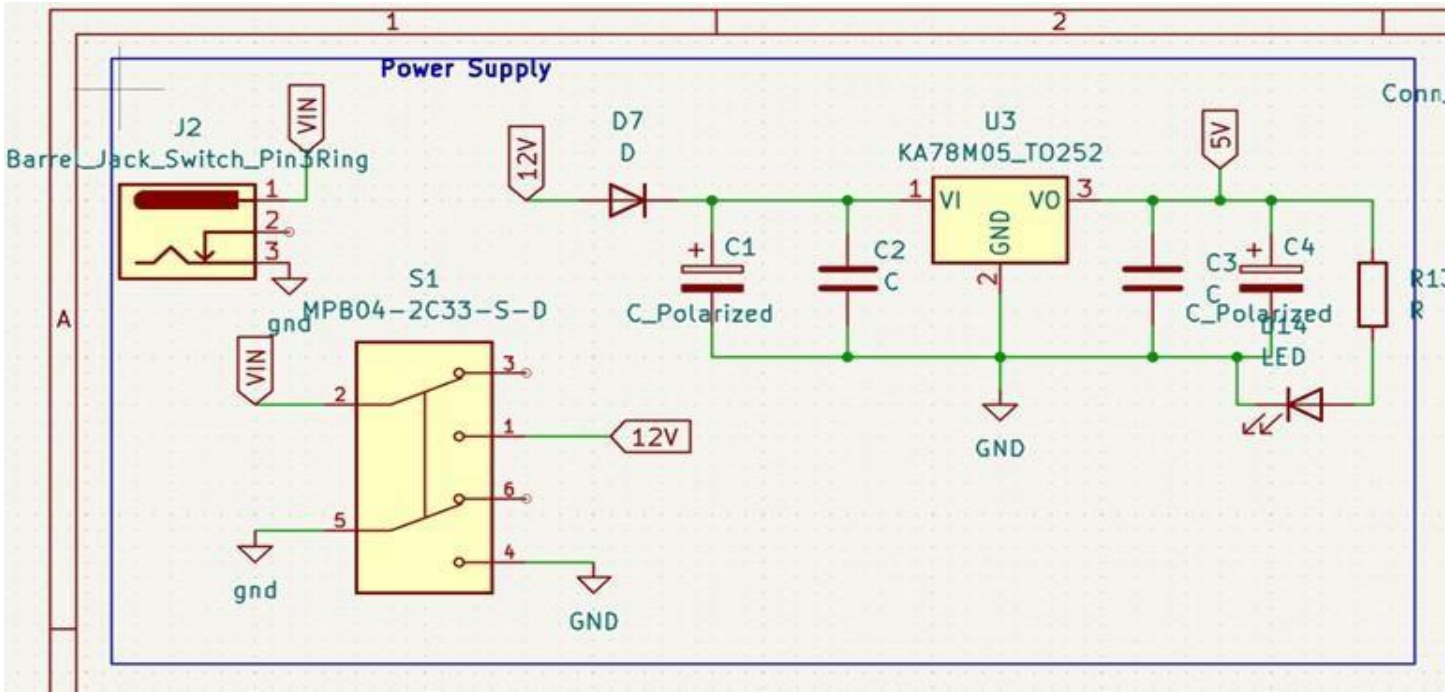
✇*Motor Driver:*

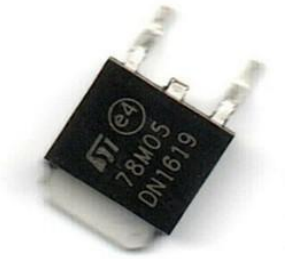




The motor driver circuit uses two L293D ICs to control four DC motors. Each IC drives two motors in an H-bridge configuration, allowing forward and reverse motion. Input pins take signals from the microcontroller, enable pins activate outputs, and flyback diodes protect against back.

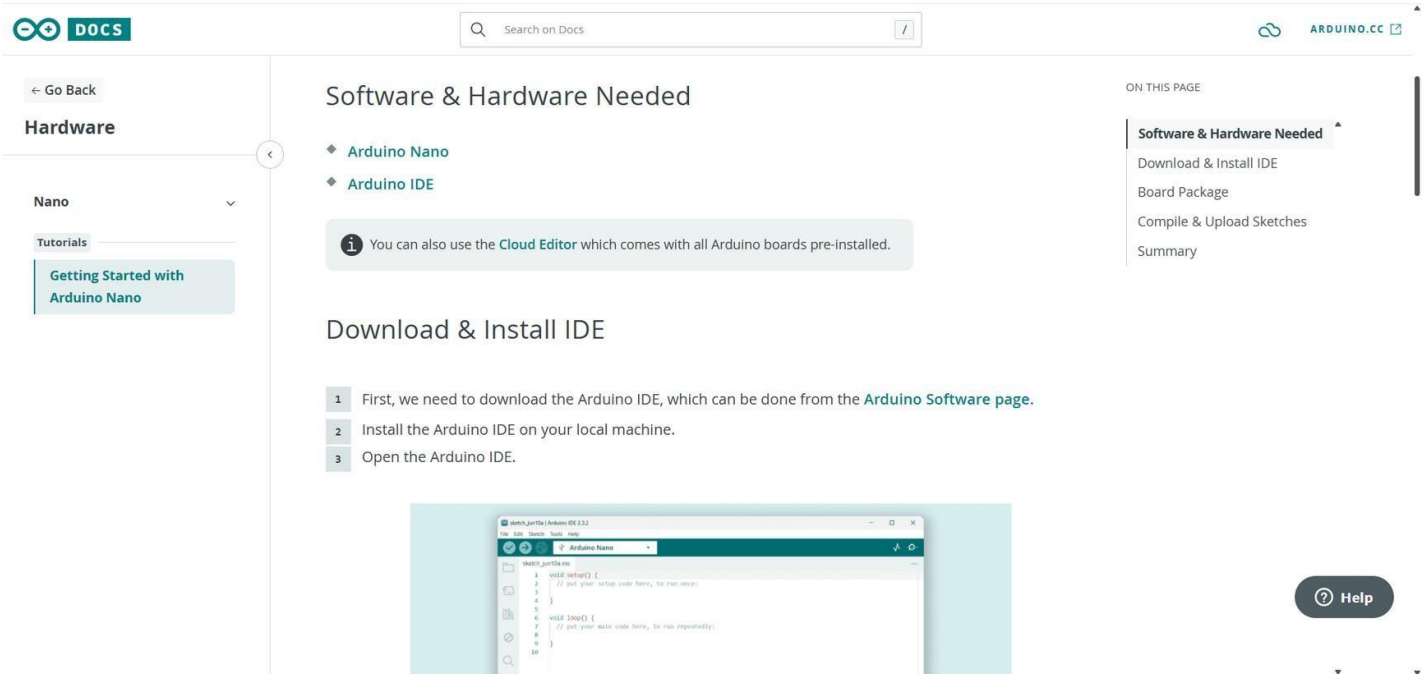
☐ Power supply:





The power supply circuit converts a 12V input, typically from a battery, to a regulated 5V output using a KA78M05 voltage regulator. A barrel jack and switch control power input, while filter capacitors and a protection diode ensure stability and safety. An indicator LED shows when power is available.

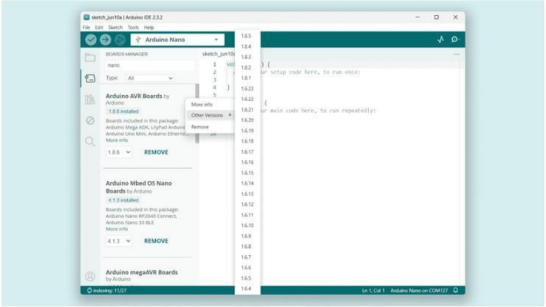
⬇Download Process & Setup



← Go Back
Hardware
Nano
Tutorials
Getting Started with Arduino Nano

Board Package

The **Arduino Core for AVR devices** comes preinstalled with the IDE, so no additional installation is necessary to get started. To use a different version than the latest, open the "Board Manager" from the left-hand menu. Search for Nano and install the version you want to use.



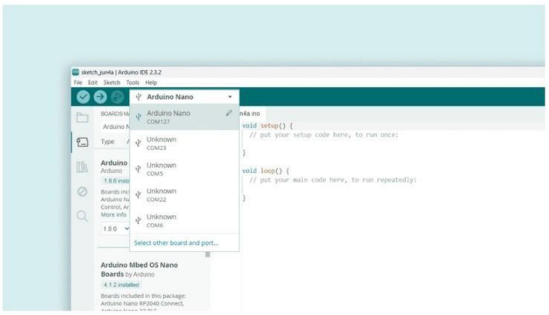
Arduino AVR Board Package

ON THIS PAGE
Software & Hardware Needed
Download & Install IDE
Board Package
Compile & Upload Sketches
Summary

You should now be able to select your board in the board selector. You will need to have your board

Arduino AVR Board Package

You should now be able to select your board in the board selector. You will need to have your board connected to your computer via the USB Mini B connector at this point.



Arduino Nano board found.

ON THIS PAGE
Software & Hardware Needed
Download & Install IDE
Board Package
Compile & Upload Sketches
Summary

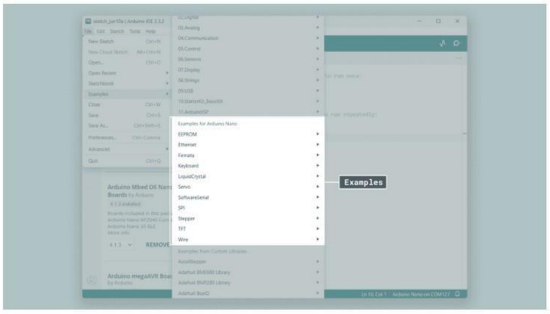
Congratulations, your board is now ready to be used via the Arduino IDE.

Compile & Upload Sketches

To compile and upload sketches, you can use the:

- **Checkmark** for compiling code.
- **Right arrow** to upload code.

There are several examples available for the Nano board, which can be accessed directly in the IDE, through **File > Examples**. These examples can be used directly without external libraries.



ON THIS PAGE
Software & Hardware Needed
Download & Install IDE
Board Package
Compile & Upload Sketches
Summary

COM Port Issue Fix

Sometimes, when uploading code to your Arduino, the COM port may not appear in Arduino IDE → Tools → Port. This usually happens if your board uses the CH340/CH341 USB-to-Serial chip and the driver is missing.

How to Fix:

1. Check in Device Manager – If you don't see the COM port in Arduino IDE, open Device Manager and look under Ports (COM & LPT) or Other Devices. The board may show as "USB-SERIAL CH340" or "Unknown Device."
2. **Download CH340 Driver:**
   - https://drive.google.com/file/d/1JNB9n2InM8JCi14wzXzv3ECKrWqyHnfU/view
3. Restart Your Computer – After installation, restart and reconnect the Arduino.
4. Update Driver if Needed – If still not detected, right-click the device in Device Manager → Update Driver → choose Browse my computer for drivers → select the CH340 driver folder.

{/} Arduino Code Overview

```
// === IR SENSOR PIN ===
#define leftSensorPin 6
#define rightSensorPin 2
#define centerSensorPin 4
// === Obstacle sensor pins (added definitions) ===
#define obstacle1Pin 3
#define obstacle2Pin 5
// === MOTOR PIN DEFINITIONS ===
#define motor1A A2
#define motor1B A3
#define EN1 10
#define motor2A A0
#define motor2B A1
#define EN2 11
#define motor3A 12
#define motor3B 13
```

```
#define EN3 7
#define motor4A A4
#define motor4B A5
#define EN4 9
void moveForward();
void turnLeft();
void turnRight();
void stopMotors();
void setup() {
pinMode(motor1A, OUTPUT);
pinMode(motor1B, OUTPUT);
pinMode(EN1, OUTPUT);
pinMode(motor2A, OUTPUT);
pinMode(motor2B, OUTPUT);
pinMode(EN2, OUTPUT);
pinMode(motor3A, OUTPUT);
pinMode(motor3B, OUTPUT);
pinMode(EN3, OUTPUT);
pinMode(motor4A, OUTPUT);
pinMode(motor4B, OUTPUT);
pinMode(EN4, OUTPUT);
// Enable motor drivers
digitalWrite(EN1, HIGH);
digitalWrite(EN2, HIGH);
digitalWrite(EN3, HIGH);
digitalWrite(EN4, HIGH); // corrected EN4 instead of repeating EN3
// IR sensor input pins
pinMode(leftSensorPin, INPUT);
```

```
  pinMode(rightSensorPin, INPUT);
  pinMode(centerSensorPin, INPUT);
  // Obstacle sensor input pins
  pinMode(obstacle1Pin, INPUT);
  pinMode(obstacle2Pin, INPUT);
  stopMotors();
  }
 void loop() {
  int leftSensor = digitalRead(leftSensorPin);
  int rightSensor = digitalRead(rightSensorPin);
  int centerSensor = digitalRead(centerSensorPin);
  int obstacle1 = digitalRead(obstacle1Pin);
  int obstacle2 = digitalRead(obstacle2Pin);
  if (obstacle1 == HIGH && obstacle2 == HIGH) {
  stopMotors();
  }
  else if (leftSensor == HIGH && centerSensor == LOW && rightSensor == HIGH) {
  moveForward();
  }
  else if (leftSensor == LOW && centerSensor == HIGH && rightSensor == HIGH) {
  turnLeft();
  }
  else if (leftSensor == HIGH && centerSensor == HIGH && rightSensor == LOW) {
  turnRight();
  }
  else {
  stopMotors();
  }
```

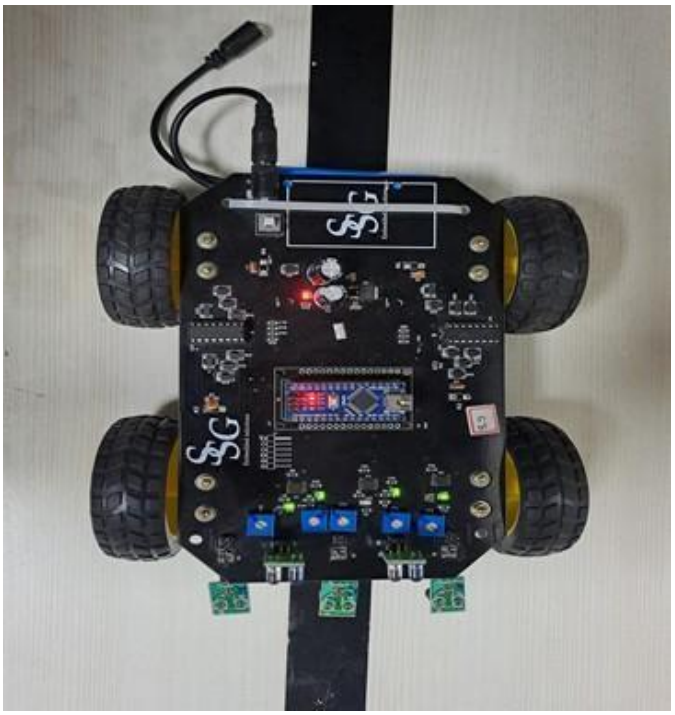
```
void moveForward() {
digitalWrite(motor1A, HIGH);
digitalWrite(motor1B, LOW);
analogWrite(EN1, 255);
digitalWrite(motor2A, HIGH);
digitalWrite(motor2B, LOW);
analogWrite(EN2, 255);
digitalWrite(motor3A, LOW);
digitalWrite(motor3B, HIGH);
analogWrite(EN3, 255);
digitalWrite(motor4A, HIGH);
digitalWrite(motor4B, LOW);
analogWrite(EN4, 255);
}
void turnRight() {
digitalWrite(motor1A, HIGH);
digitalWrite(motor1B, LOW);
analogWrite(EN1, 255);
digitalWrite(motor2A, HIGH);
digitalWrite(motor2B, LOW);
analogWrite(EN2, 255);
digitalWrite(motor3A, HIGH);
digitalWrite(motor3B, LOW);
analogWrite(EN3, 255);
digitalWrite(motor4A, LOW);
digitalWrite(motor4B, HIGH);
analogWrite(EN4, 255);
```

```
void turnLeft() {
digitalWrite(motor1A, LOW);
digitalWrite(motor1B, HIGH);
analogWrite(EN1, 255);
digitalWrite(motor2A, LOW);
digitalWrite(motor2B, HIGH);
analogWrite(EN2, 255);
digitalWrite(motor3A, LOW);
digitalWrite(motor3B, HIGH);
analogWrite(EN3, 255);
digitalWrite(motor4A, HIGH);
digitalWrite(motor4B, LOW);
analogWrite(EN4, 255);
}
void stopMotors() {
digitalWrite(motor1A, LOW);
digitalWrite(motor1B, LOW);
analogWrite(EN1, 0);
digitalWrite(motor2A, LOW);
digitalWrite(motor2B, LOW);
analogWrite(EN2, 0);
digitalWrite(motor3A, LOW);
digitalWrite(motor3B, LOW);
analogWrite(EN3, 0);
digitalWrite(motor4A, LOW);
digitalWrite(motor4B, LOW);
analogWrite(EN4, 0);}
```

Process

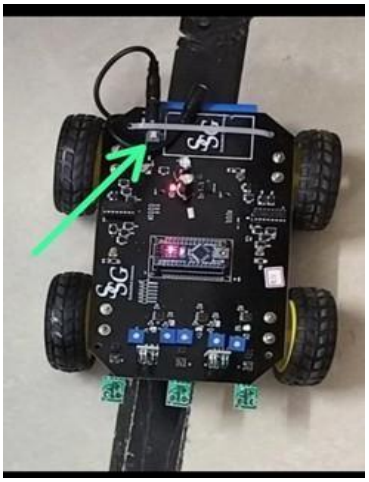
1) **Initial Setup:**

- First, place the robot on the black line (track).



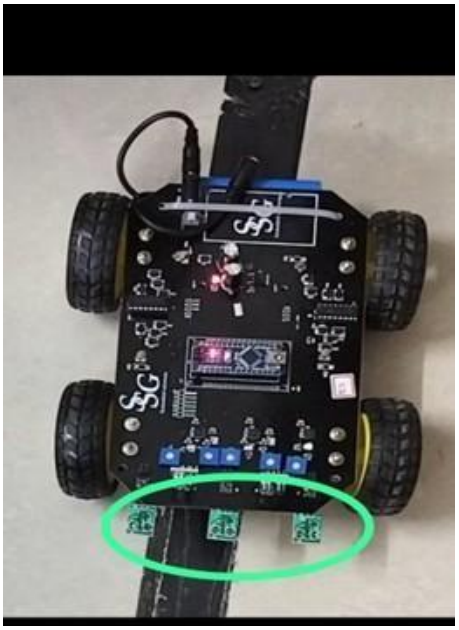
2) **Power On:**

- Turn on the robot using the switch button.



3) **Sensor Configuration:**

• The robot has three IR sensors located at the bottom: left sensor, center sensor, and right sensor.

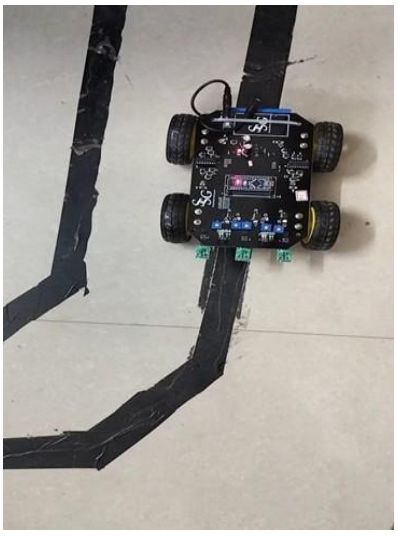


4) **Line Detection Logic:**

a. If the left sensor detects the black line, the robot will turn left.

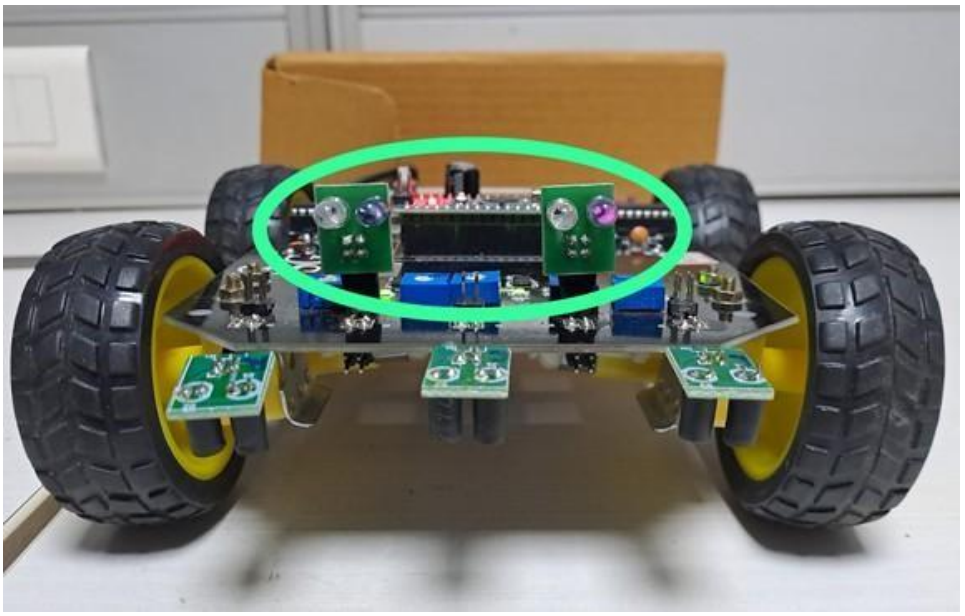
b. If the right sensor detects the black line, the robot will turn right.

c. If the center sensor detects the black line, the robot will move straight forward.



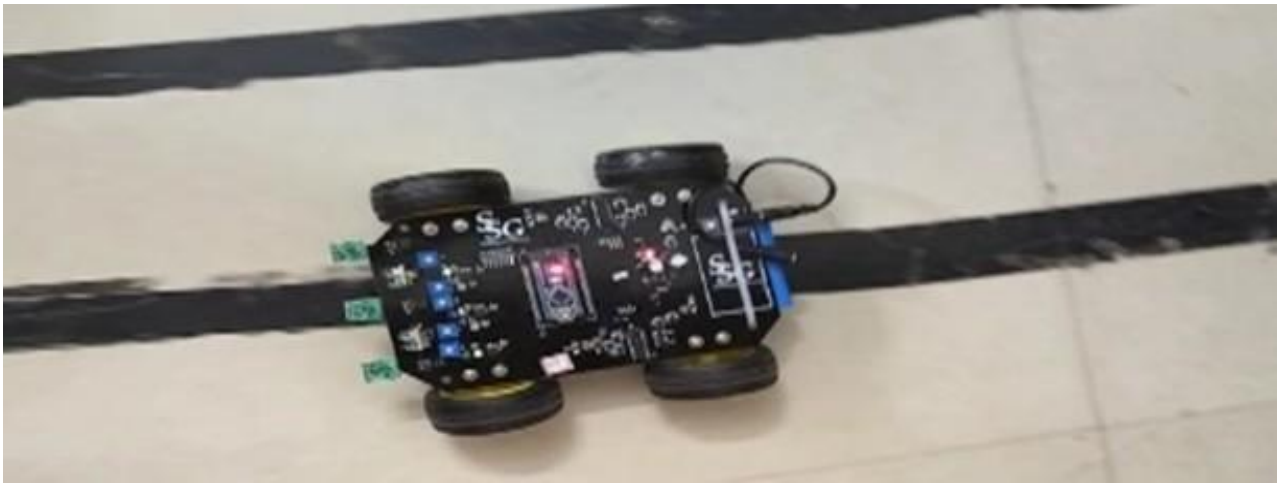
5) **Obstacle Detection Logic**

- If Sensor detects an obstacle ahead, the robot will either stop or move backward depending on the logic implemented in the code.



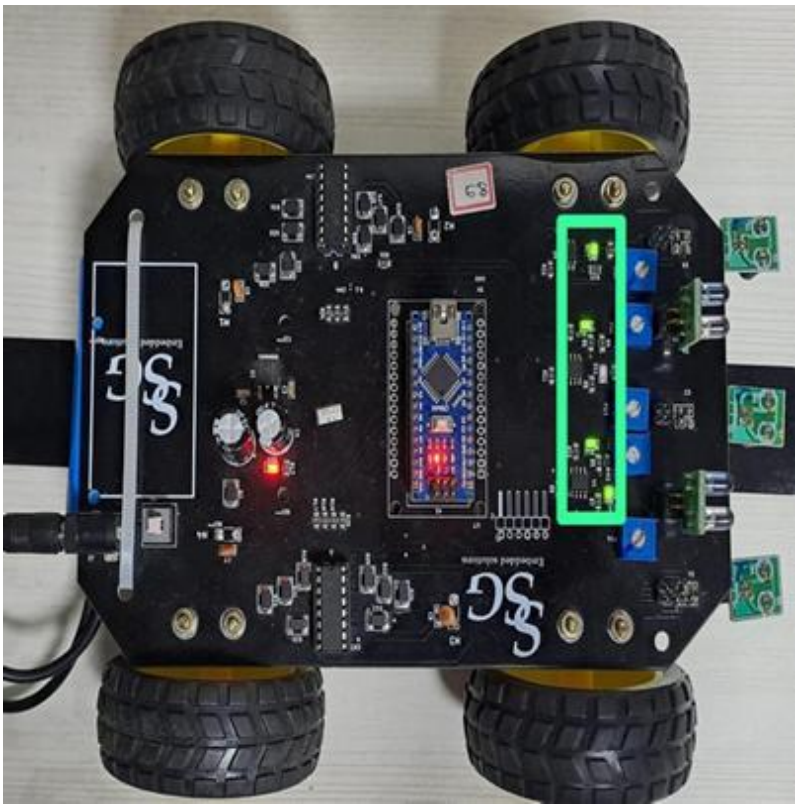
6) **Continuous Operation:**

- Based on these sensor readings, the robot continuously follows the black path.



7) **Led Indicators**

• If the robot is not moving, adjust the potentiometer. Near the potentiometer, there are three LEDs. When all three LEDs are ON, it means the robot is correctly aligned on the black line.



8) **Adjusting Sensitivity**

• Adjusting Sensor Sensitivity (Short Version):

• Right Sensor: Turn the right potentiometer anticlockwise until its LED turns OFF on detecting the black line → robot turns right.

• Center Sensor: Adjust center potentiometer until center LED turns OFF on the black line → robot moves forward.

• Left Sensor: Adjust left potentiometer until left LED turns OFF on the black line → robot turns left.

9) Troubleshooting

• If the robot still does not function properly, fine-tune the potentiometers by rotating them clockwise or anticlockwise as needed.

• ☐ Line Follower Robot Demonstration Video:

• [Click here to view the video](https://drive.google.com/file/d/11wq-rT6hHIoJApXoP8utQfJZ0lptYdIs/view?usp=drivesdk)

Workshops