LINE CONTROLLER ROBOT

A comprehensive study & CNN model implementation

MAJOR PROJECT

Submitted in the Partial Fulfilment for the Requirement for the Award of the

Degree of Bachelor of Technology

in

Electronics Engineering (VLSI Design and Technology) (EVDT)

Submitted by:

Darsh Kuchhal(25/A16/031)

Anuj Pratap Singh (25/A15/059)

Prince Kumar (25/A18/019)

Under the supervision of

Ms Soni Rajput

PhD

Department of Electronis Engineering

Delhi Technological University

(Formally Delhi College of Engineering)

Bawana Road, Delhi-110042

November 2025

Project overview

A line-controller robot follows a high-contrast line (usually black on white) using an array of IR sensors, a microcontroller that reads sensors and computes error, and a motor driver that controls two DC motors.

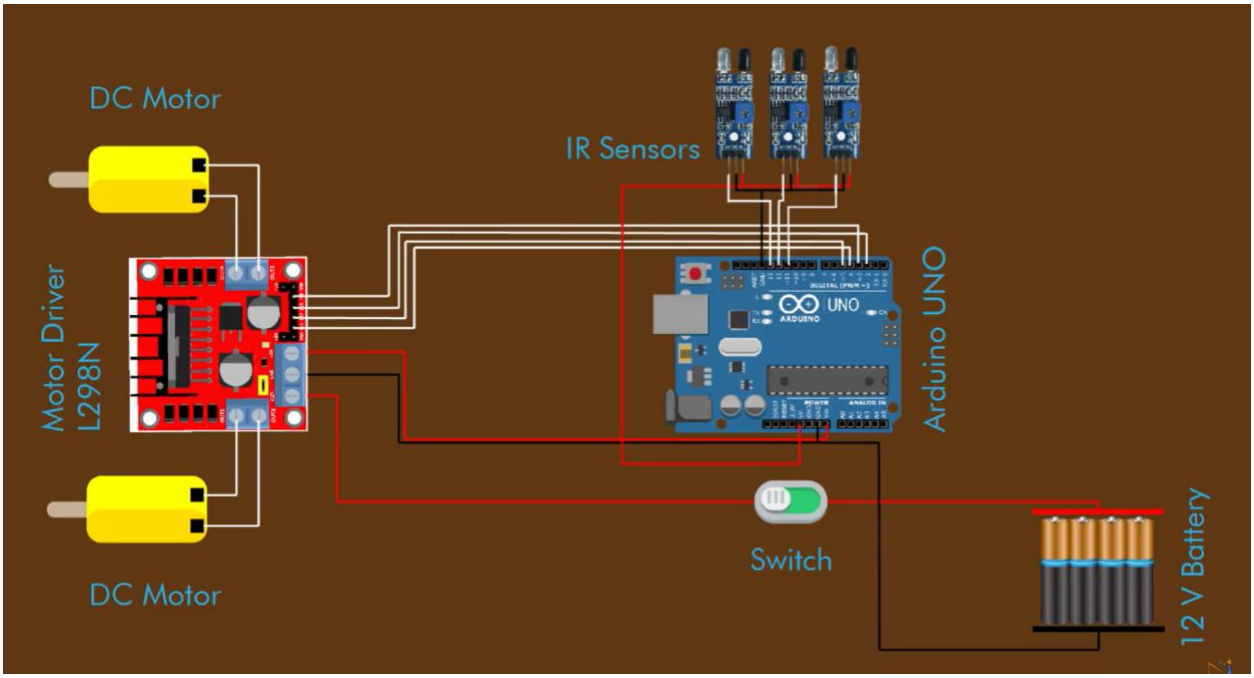
Aim:

- Detect and follow a line robustly at low and moderate speeds.
- Implement closed-loop steering
- Demonstrate performance metrics (response time, cross-track error, steady-state error).

Hardware components

- Microcontroller: **Arduino Uno**
- Sensors: **3 reflectance sensors** (TCRT5000)
- Motors: **2 DC geared motors**
- Motor driver: **L293D** (simple)
- Chassis: small robot chassis kit with mounting plate and wheels.
- Wheels & caster: 2 drive wheels + 1 castor.
- Power: LiPo battery 11.1V/3S.
- Misc: jumper wires, screws, switch, headers.
- Tools: soldering iron, multimeter, small screwdriver, hot glue.

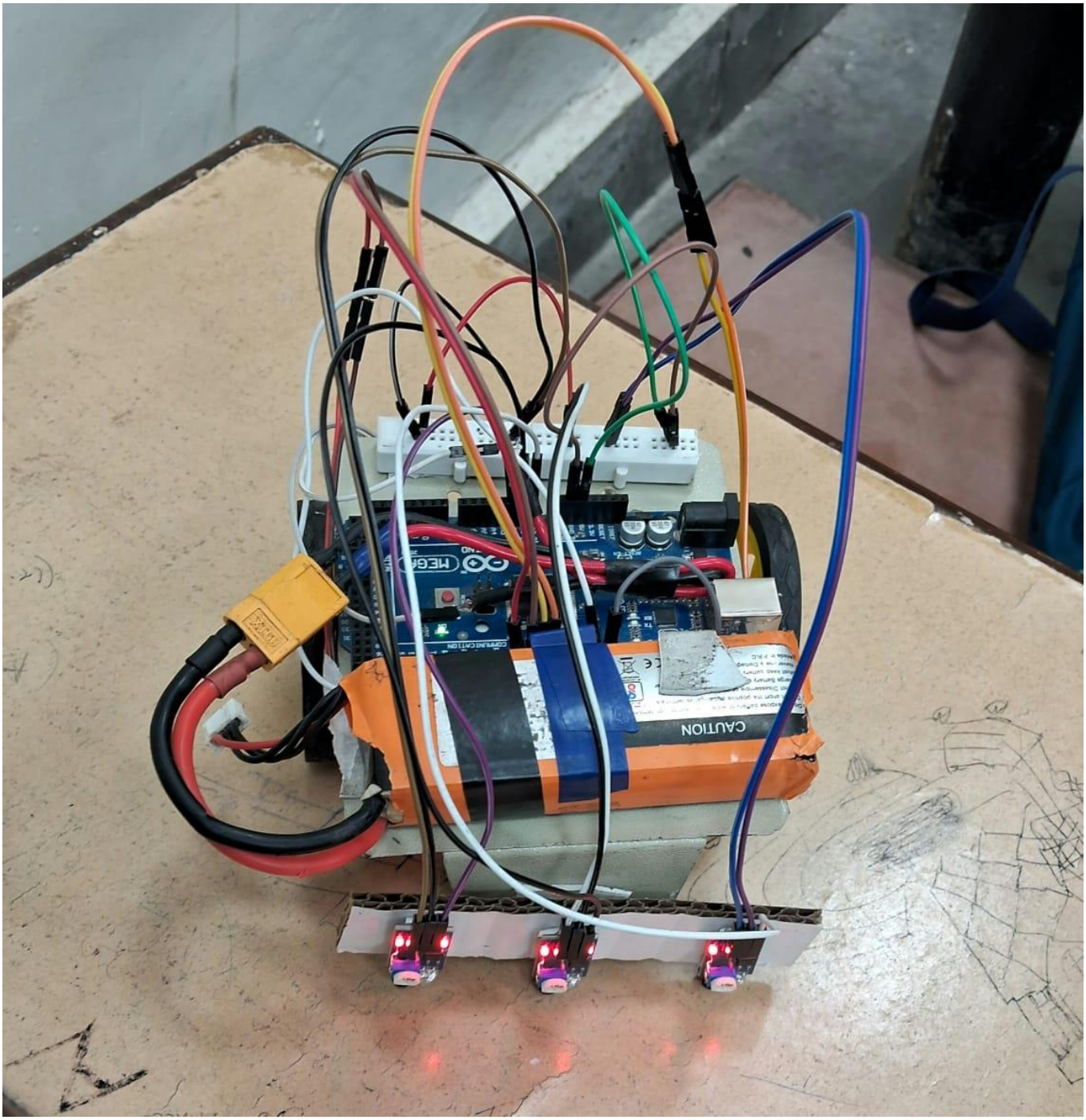
Circuit



Procedure

- IR sensors: each sensor output → Arduino digital input
- Motors: Arduino direction pins → motor driver dir pins. Connect motor driver Vcc to battery, logic Vcc to 5V regulator.
- Battery negative → common ground with Arduino and motor driver.
- Optional: encoder signals → interrupt pins on Arduino.

  Pin mapping example;
- A0, A1, A2 = sensors (for 3 sensors)
- D2/D3 = left motor direction pins, D4/D5 = right motor direction pins
- Place sensors ~5–15 mm above surface.
- Keep sensors rigidly mounted to reduce noise caused by wobble.
- Low center of gravity helps stability.
- Calibrate raw sensor readings on white and black to find thresholds.


CAUTION

Arduino code (3 IR sensors, analog)

```
// ===== Pin Definitions =====
#define LEFT_SENSOR   A0
#define MIDDLE_SENSOR A1
#define RIGHT_SENSOR  A2

// Motor control pins (L293D)
#define LM1 2
#define LM2 3
#define RM1 4
#define RM2 5

// Enable pins (PWM)
#define ENL 9
#define ENR 10

// Threshold to detect black line (tune this!)
int threshold = 500;

// Base motor speeds
int baseSpeed = 90;     // Forward speed (reduced)
int turnSpeed = 70;     // Slightly slower for turns
int fineTurnSpeed = 60; // Even slower for minor corrections
int stopSpeed = 0;

void setup() {
  Serial.begin(9600);
```

```
  pinMode(LM1, OUTPUT);
  pinMode(LM2, OUTPUT);
  pinMode(RM1, OUTPUT);
  pinMode(RM2, OUTPUT);
  pinMode(ENL, OUTPUT);
  pinMode(ENR, OUTPUT);
}

void loop() {
  int leftValue = analogRead(LEFT_SENSOR);
  int middleValue = analogRead(MIDDLE_SENSOR);
  int rightValue = analogRead(RIGHT_SENSOR);

  Serial.print("L: ");
  Serial.print(leftValue);
  Serial.print("  M: ");
  Serial.print(middleValue);
  Serial.print("  R: ");
  Serial.println(rightValue);

  int L = (leftValue < threshold) ? 1 : 0;
  int M = (middleValue < threshold) ? 1 : 0;
  int R = (rightValue < threshold) ? 1 : 0;

  // ===== Movement Logic =====
  if (M == 0 && L == 1 && R == 1) {
    Serial.println(" Forward");
```

```
  moveForward(baseSpeed);
}
else if (L == 0 && M == 1 && R == 1) {
  Serial.println(" Left");
  turnLeft(turnSpeed);
}
else if (R == 0 && M == 1 && L == 1) {
  Serial.println(" Right");
  turnRight(turnSpeed);
}
else if (L == 0 && M == 0 && R == 1) {
  Serial.println(" Hard Left");
  turnLeft(fineTurnSpeed); // slower sharper correction
}
else if (R == 0 && M == 0 && L == 1) {
  Serial.println(" Hard Right");
  turnRight(fineTurnSpeed); // slower sharper correction
}
else if (L == 1 && M == 1 && R == 1) {
  Serial.println(" Stop");
  stopMotors();
}
else {
  // If lost, slow down and stop to regain track
  Serial.println(" Lost Line - Stop");
  stopMotors();
}
```

```
  delay(20);  // small delay helps stabilize sensor response
}

// ===== Motor Control Functions =====
void moveForward(int speed) {
  analogWrite(ENL, speed);
  analogWrite(ENR, speed);
  digitalWrite(LM1, HIGH);
  digitalWrite(LM2, LOW);
  digitalWrite(RM1, HIGH);
  digitalWrite(RM2, LOW);
}

void turnLeft(int speed) {
  analogWrite(ENL, speed * 0.6);  // left motor slower
  analogWrite(ENR, speed);       // right motor faster
  digitalWrite(LM1, LOW);
  digitalWrite(LM2, HIGH);
  digitalWrite(RM1, HIGH);
  digitalWrite(RM2, LOW);
}

void turnRight(int speed) {
  analogWrite(ENL, speed);
  analogWrite(ENR, speed * 0.6);
  digitalWrite(LM1, HIGH);
  digitalWrite(LM2, LOW);
  digitalWrite(RM1, LOW);
```

```
  digitalWrite(RM2, HIGH);
}

void stopMotors() {
  analogWrite(ENL, stopSpeed);
  analogWrite(ENR, stopSpeed);
  digitalWrite(LM1, LOW);
  digitalWrite(LM2, LOW);
  digitalWrite(RM1, LOW);
  digitalWrite(RM2, LOW);
}
```

Result

https://drive.google.com/file/d/135AqaJQyBBdCF8iLVFIVVvWFeq_K7S5_/view?usp=sharing

Precautions

- Use a regulated 9–12V supply and never exceed component voltage ratings.
- Always connect all grounds (GND) of Arduino, sensors, and motor driver together.
- Calibrate IR sensors properly—maintain 5-15mm height and avoid strong light.
- Keep wiring firm and tidy to prevent loose or tangled connections.
- Disconnect motor power while uploading code to avoid accidental movement.