// ====== Line Follower Robot with 4 IR sensors (PID Control) ======

// Components: Arduino UNO + L298N + 2 DC Motors + 4 IR Sensors

// Motor Driver Pins (L298N)

const int ENA = 5; // Left motor speed (PWM)

const int ENB = 6; // Right motor speed (PWM)

const int IN1 = 7; // Left motor IN1

const int IN2 = 8; // Left motor IN2

const int IN3 = 9; // Right motor IN3

const int IN4 = 10; // Right motor IN4

// IR Sensors

const int S0 = 2; // Left-most sensor

const int S1 = 3; // Left-center sensor

const int S2 = 4; // Right-center sensor

const int S3 = 11; // Right-most sensor

// PID constants (tune these values for best performance)

float Kp = 25; // Proportional

float Ki = 0.0; // Integral (often small or zero)

float Kd = 15; // Derivative

// Variables for PID

float error = 0, previousError = 0;

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

float PIDvalue = 0;

// Motor speed settings

int baseSpeed = 120; // Base speed (0-255)

int maxSpeed = 200; // Maximum speed (0-255)

void setup() {

// Motor pins

pinMode(ENA, OUTPUT); pinMode(ENB, OUTPUT);

pinMode(IN1, OUTPUT); pinMode(IN2, OUTPUT);

pinMode(IN3, OUTPUT); pinMode(IN4, OUTPUT);

// Sensor pins

pinMode(S0, INPUT); pinMode(S1, INPUT);

pinMode(S2, INPUT); pinMode(S3, INPUT);

Serial.begin(9600); // For debugging

}

void loop() {

// Read sensors (LOW = black line, HIGH = white surface)

int L2 = !digitalRead(S0);

int L1 = !digitalRead(S1);

int R1 = !digitalRead(S2);

int R2 = !digitalRead(S3);

// Calculate position (weighted average method)

// Assign weights: -3, -1, +1, +3

int sensorSum = L2 + L1 + R1 + R2;

int weightedSum = (-3 \* L2) + (-1 \* L1) + (1 \* R1) + (3 \* R2);

if (sensorSum != 0) {

error = (float)weightedSum / sensorSum; // Normalize error

} else {

// No line detected: keep previous error (avoid random stop)

error = previousError;

}

// PID calculations

P = error;

I += error;

D = error - previousError;

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

previousError = error;

// Calculate motor speeds

int leftMotorSpeed = baseSpeed - PIDvalue;

int rightMotorSpeed = baseSpeed + PIDvalue;

// Constrain speeds to safe range

leftMotorSpeed = constrain(leftMotorSpeed, 0, maxSpeed);

rightMotorSpeed = constrain(rightMotorSpeed, 0, maxSpeed);

// Run motors forward

moveMotors(leftMotorSpeed, rightMotorSpeed);

delay(10); // Small delay for stability

}

// Function: Move motors with given speeds

void moveMotors(int leftSpeed, int rightSpeed) {

// Left motor forward

digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);

// Right motor forward

digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);

analogWrite(ENA, leftSpeed);

analogWrite(ENB, rightSpeed);

}