// === Pin Definitions ===

#define ENA 9

#define IN1 10

#define IN2 11

#define ENB 6

#define IN3 12

#define IN4 13

#define S1 2

#define S2 3

#define S3 4

#define S4 5

#define S5 8

int lastSeen = 0; // -1 = left, 1 = right, 0 = center

int lostCounter = 0;

// PID control variables

float Kp = 25, Ki = 0, Kd = 20;

int baseSpeedLeft = 90;

int baseSpeedRight = 90;

int sharpSpeed = 60;

int error = 0, lastError = 0;

float integral = 0;

void setup() {

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

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

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

pinMode(S1, INPUT); pinMode(S2, INPUT);

pinMode(S3, INPUT); pinMode(S4, INPUT); pinMode(S5, INPUT);

Serial.begin(9600);

}

void setMotorSpeed(int rightSpeed, int leftSpeed) {

analogWrite(ENA, constrain(rightSpeed, 0, 255));

analogWrite(ENB, constrain(leftSpeed, 0, 255));

}

void forward() {

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

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

}

void rotateLeft() {

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

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

}

void rotateRight() {

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

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

}

void loop() {

int s1 = !digitalRead(S1);

int s2 = !digitalRead(S2);

int s3 = !digitalRead(S3);

int s4 = !digitalRead(S4);

int s5 = !digitalRead(S5);

int pattern = (s1 << 4) | (s2 << 3) | (s3 << 2) | (s4 << 1) | s5;

Serial.print("Pattern: ");

Serial.println(pattern, BIN);

// Sharp or Obtuse Turns

if (pattern == 0b10000 || pattern == 0b11000) {

setMotorSpeed(sharpSpeed, baseSpeedLeft + 20);

rotateLeft();

lastSeen = -1;

delay(150);

return;

}

if (pattern == 0b00001 || pattern == 0b00011) {

setMotorSpeed(baseSpeedRight + 20, sharpSpeed);

rotateRight();

lastSeen = 1;

delay(150);

return;

}

// Line Lost

if (pattern == 0b00000) {

lostCounter += 5;

if (lostCounter > 500) {

if (lastSeen == -1) {

setMotorSpeed(60, 100);

rotateLeft();

} else if (lastSeen == 1) {

setMotorSpeed(100, 60);

rotateRight();

} else {

setMotorSpeed(70, 70);

rotateRight();

}

} else {

if (lastSeen == -1) {

setMotorSpeed(sharpSpeed, baseSpeedLeft + 20);

rotateLeft();

} else if (lastSeen == 1) {

setMotorSpeed(baseSpeedRight + 20, sharpSpeed);

rotateRight();

} else {

setMotorSpeed(70, 70);

rotateRight();

}

}

return;

}

// Line Found: PID

lostCounter = 0;

int weights[5] = {-2, -1, 0, 1, 2};

int sensors[5] = {s1, s2, s3, s4, s5};

int activeCount = 0;

int weightedSum = 0;

for (int i = 0; i < 5; i++) {

weightedSum += sensors[i] \* weights[i];

activeCount += sensors[i];

}

if (activeCount == 0) return;

error = weightedSum;

integral += error;

float derivative = error - lastError;

float correction = Kp \* error + Ki \* integral + Kd \* derivative;

int leftSpeed = baseSpeedLeft + correction;

int rightSpeed = baseSpeedRight - correction;

setMotorSpeed(rightSpeed, leftSpeed);

forward();

lastError = error;

if (error < 0) lastSeen = -1;

else if (error > 0) lastSeen = 1;

else lastSeen = 0;

delay(5);

}