#include <NewPing.h>

// Connections:

#define motor1 8 // Motor control HIGH = FWD, LOW = BWD

#define motor2 9 // Speed input

#define motor3 10 // Speed input

#define motor4 11 // Motor control HIGH = FWD, LOW = BWD

#define trigPin1 2 // IMPULSE SENT FROM THE RIGHT SENSOR

#define echoPin1 3 // PULSE RECEIVED FROM THE RIGHT SENSOR

#define trigPin2 4 // IMPULSE SENT FROM THE LEFT SENSOR

#define echoPin2 5 // PULSE RECEIVED FROM THE LEFT SENSOR

#define trigPin3 6 // IMPULSE SENT FROM THE FRONT SENSOR

#define echoPin3 7 // PULSE RECEIVED FROM THE FRONT SENSOR

int maxdistance = 200;

int speed;

int ramspeed = 250; // MAX SPEED

int safespeed = 200; // NORMAL SPEED

int stucktimer = 0;

int timer = 0;

int z = 0;

int y = 0;

int ls = 0, fs = 0, rs = 0, lm, rm;

long duration, distance, RightSensor, BackSensor, FrontSensor, LeftSensor;

void setup() {

Serial.begin(9600);

// Motor driver pins to output

pinMode(motor1, OUTPUT); // SIGNAL OUTPUT OF ROTATION OF THE CONTROL

pinMode(motor2, OUTPUT); // SIGNAL OUTPUT OF ROTATION OF THE CONTROL

pinMode(motor3, OUTPUT); // SIGNAL OUTPUT OF ROTATION OF THE CONTROL

pinMode(motor4, OUTPUT); // SIGNAL OUTPUT OF ROTATION OF THE CONTROL

pinMode(trigPin1, OUTPUT); // SIGNAL OUTPUT OF THE RIGHT TRIGGER

pinMode(echoPin1, INPUT); // SIGNAL INPUT OF THE RIGHT ECHO

pinMode(trigPin2, OUTPUT); // SIGNAL OUTPUT OF THE LEFT TRIGGER

pinMode(echoPin2, INPUT); // SIGNAL INPUT OF THE LEFT ECHO

pinMode(trigPin3, OUTPUT); // SIGNAL OUTPUT OF THE FRONT TRIGGER

pinMode(echoPin3, INPUT); // SIGNAL INPUT OF THE FRONT ECHO

pinMode(13, OUTPUT); // LED LIGHT

}

// LOOPING STRUCTURES

void loop() {

Sonar(); // COMPUTING THE DISTANCE OF 3 ULTRASONIC SENSOR

PossibleWallSensor(); // FUNCTION OF THE POSSIBLE MAZE STRUCTURE

// LS = LEFT SENSOR , FS = FRONT SENSOR , RS = RIGHT SENSOR

if ( ls == 1 & fs == 0 & rs == 1 ) // TO FORWARD

{

SAFEDRIVE();

}

if ( ls == 0 & fs == 0 & rs == 0 ) // TO FORWARD

{

SAFEDRIVE();

}

if ( ls == 1 & fs == 1 & rs == 1 ) // TURN TO 180° LEFT ROTATION

{

ROTATE();

STOP();

}

if ( ls == 1 & fs == 1 & rs == 0 ) // TURN TO 90° RIGHT

{

RIGHT\_TURN();

delay(400);

STOP();

}

if ( ls == 0 & fs == 1 & rs == 1 ) // TURN TO 90° LEFT

{

LEFT\_TURN();

delay(400);

STOP();

}

if ( ls == 0 & fs == 1 & rs == 0 ) // TURN TO 90° LEFT

{

LEFT\_TURN();

delay(400);

STOP();

}

if ( lm == 1 )

{

LEFT\_MOVE();

}

if ( rm == 1 )

{

RIGHT\_MOVE();

}

}

// ALL FUNCTIONS

void PossibleWallSensor()

{

if (LeftSensor <= 15)

{

ls = 1;

}

else

{

ls = 0;

}

if (RightSensor <= 15)

{

rs = 1;

}

else

{

rs = 0;

}

if (FrontSensor <= 15)

{

fs = 1;

digitalWrite(13, HIGH);

}

else

{

fs = 0;

digitalWrite(13, LOW);

}

if (LeftSensor <= 5)

{

lm = 1;

}

else

{

lm = 0;

}

if (RightSensor <= 5)

{

rm = 1;

}

else

{

rm = 0;

}

}

void Sonar()

{

SonarSensor(trigPin1, echoPin1);

RightSensor = distance;

SonarSensor(trigPin2, echoPin2);

LeftSensor = distance;

SonarSensor(trigPin3, echoPin3);

FrontSensor = distance;

Serial.print(LeftSensor);

Serial.print(" - ");

Serial.print(FrontSensor);

Serial.print(" - ");

Serial.println(RightSensor);

}

void SonarSensor(int trigPin, int echoPin)

{

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = (duration / 2) / 29.1;

}

void SAFEDRIVE()

{

digitalWrite(motor1, HIGH);

digitalWrite(motor4, HIGH);

analogWrite(motor2, safespeed);

analogWrite(motor3, safespeed);

}

void STOP()

{

digitalWrite(motor1, HIGH);

digitalWrite(motor4, HIGH);

analogWrite(motor2, LOW);

analogWrite(motor3, LOW);

}

void CHARGE()

{

digitalWrite(motor1, HIGH);

analogWrite(motor2, ramspeed);

analogWrite(motor3, ramspeed);

digitalWrite(motor4, HIGH);

}

void ROTATE() // Default reaction after timer runs out

{

LEFT\_TURN();

delay(1000);

}

void LEFT\_TURN()

{

digitalWrite(motor1,HIGH);

digitalWrite(motor4,LOW);

analogWrite(motor2,200);

analogWrite(motor3,200);

}

void RIGHT\_TURN()

{

digitalWrite(motor1,LOW);

digitalWrite(motor4,HIGH);

analogWrite(motor2,200);

analogWrite(motor3,200);

}

void LEFT\_MOVE()

{

digitalWrite(motor1,HIGH);

digitalWrite(motor4,HIGH);

analogWrite(motor2,0);

analogWrite(motor3,200);

}

void RIGHT\_MOVE()

{

digitalWrite(motor1,HIGH);

digitalWrite(motor4,HIGH);

analogWrite(motor2,200);

analogWrite(motor3,0);

}