

Session 3: Obstacle avoider robot

Robotics Club

Linefollower using Differential drive

```
//-----Global declaration-----//  
//motor  
int lm_pin1=2;           //<-----Changes  
int lm_pin2=3;  
int rm_pin1=4;  
int rm_pin2=5;  
  
//sensor  
int ls_pin=6;  
int rs_pin=7;  
int rs_value;  
int ls_value;  
  
//-----function defination-----//  
void read_sen_value()  
{  
    rs_value = digitalRead(rs_pin);  
    ls_value = digitalRead(ls_pin);  
}
```

```

void check_direction_move()
{
    if(ls_value == 0 && rs_value == 0) //WW
    {
        forward();//forward
    }
    else if(ls_value == 0 && rs_value == 1)//WB
    {
        right();//right
    }
    else if(ls_value == 1 && rs_value == 0)//BW
    {
        left();//left
    }
    else if(ls_value == 1 && rs_value == 1)//BB
    {
        STOP();//STOP
    }
}

void forward() //-----changes in all below functions
{
    digitalWrite(lm_pin1, 1);
    digitalWrite(lm_pin2, 0);
    digitalWrite(rm_pin1, 1);
    digitalWrite(rm_pin2, 0);
}

```

```
void left()
{
    digitalWrite(lm_pin1, 0);
    digitalWrite(lm_pin2, 1);
    digitalWrite(rm_pin1, 1);
    digitalWrite(rm_pin2, 0);
}
void right()
{
    digitalWrite(lm_pin1, 1);
    digitalWrite(lm_pin2, 0);
    digitalWrite(rm_pin1, 0);
    digitalWrite(rm_pin2, 1);
}
void STOP()
{
    digitalWrite(lm_pin1, 0);
    digitalWrite(lm_pin2, 0);
    digitalWrite(rm_pin1, 0);
    digitalWrite(rm_pin2, 0);
}
```

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

  delay(2000);

  pinMode(rm_pin1, OUTPUT); //motor          //<-----Changes
  pinMode(rm_pin2, OUTPUT);
  pinMode(lm_pin1, OUTPUT);
  pinMode(lm_pin2, OUTPUT);

  pinMode(rs_pin, INPUT); //input
  pinMode(ls_pin, INPUT);
}

//-----loop-----//
void loop() {
  read_sen_value();

  check_direction_move();
}
```

Obstacle Avoider using IR Sensor

```
//-----Global declaration-----//
//motor
int lm_pin1=2;
int lm_pin2=3;
int rm_pin1=4;
int rm_pin2=5;

//delay
int turn_delay = 100;
int reverse_delay = 50;

//sensor
int ls_pin=6;
int rs_pin=7;
int rs_value;
int ls_value;

//-----function defination-----//
void read_sen_value()
{
    rs_value = digitalRead(rs_pin);
    ls_value = digitalRead(ls_pin);
}
```



```

void check_direction_move()
{
    if(ls_value == 0 && rs_value == 0) //WW //object on both side
    {
        reverse();
        delay(reverse_delay);
        right();                //<-----Changes
        delay(turn_delay*2);
    }
    else if(ls_value == 0 && rs_value == 1)//WB           //object on left side
    {
        reverse();
        delay(reverse_delay);
        right();//right
        delay(turn_delay);      //<-----Changes
    }
    else if(ls_value == 1 && rs_value == 0)//BW           //object on right side
    {
        reverse();
        delay(reverse_delay);
        left();//left
        delay(turn_delay);      //<-----Changes
    }
    else if(ls_value == 1 && rs_value == 1)//BB           //no object
    {
        forward();              //<-----Changes
    }
}

```

```
void forward()
{
    digitalWrite(lm_pin1, 1);
    digitalWrite(lm_pin2, 0);
    digitalWrite(rm_pin1, 1);
    digitalWrite(rm_pin2, 0);
}
```

```
void reverse()
{
    digitalWrite(lm_pin1, 0);
    digitalWrite(lm_pin2, 1);
    digitalWrite(rm_pin1, 0);
    digitalWrite(rm_pin2, 1);
}
```

```
void left()
{
    digitalWrite(lm_pin1, 0);
    digitalWrite(lm_pin2, 1);
    digitalWrite(rm_pin1, 1);
    digitalWrite(rm_pin2, 0);
}
```

```
void right()
{
    digitalWrite(lm_pin1, 1);
    digitalWrite(lm_pin2, 0);
    digitalWrite(rm_pin1, 0);
    digitalWrite(rm_pin2, 1);
}
```

```
}  
void STOP()  
{  
    digitalWrite(lm_pin1, 0);  
    digitalWrite(lm_pin2, 0);  
    digitalWrite(rm_pin1, 0);  
    digitalWrite(rm_pin2, 0);  
}  
  
//-----setup-----//  
void setup() {  
    Serial.begin(9600);  
  
    delay(2000);  
  
    pinMode(rm_pin1, OUTPUT); //motor  
    pinMode(rm_pin2, OUTPUT);  
    pinMode(lm_pin1, OUTPUT);  
    pinMode(lm_pin2, OUTPUT);  
  
    pinMode(rs_pin, INPUT); //input  
    pinMode(ls_pin, INPUT);  
}  
  
//-----loop-----//  
void loop() {  
    read_sen_value();  
  
    check_direction_move();  
}
```

BASIC OF ULTRASONIC SENSOR

```
int triggerpin = 1;
int echopin = 2;
int distance, t;

void setup(){
  Serial.begin(9600);
  pinMode(triggerpin, OUTPUT); //TO ULTRA SONIC SENSOR
  pinMode(echopin, INPUT);    //FROM ULTRA SONIC SENSOR
}

void loop(){
  digitalWrite(triggerpin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerpin, LOW);

  digitalRead(echopin);

  t = pulseIn(echopin, HIGH);

  distance = t*(0.034)/2;
  Serial.println(distance);
}
```

BASIC OF SERVO MOTOR

basic_servo_sweep

```
#include <Servo.h>
Servo myservo;

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    myservo.write(pos);                // tell servo to go to position in variable 'pos'
    delay(15);                          // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);                // tell servo to go to position in variable 'pos'
    delay(15);                          // waits 15ms for the servo to reach the position
  }
}
```

pot_and_servo

```
#include <Servo.h>
Servo myservo; //servo object

int potpin = 0;
int val;

void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
}

void loop() {
  val = analogRead(potpin); // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180); // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val); // sets the servo position according to the scaled value
  delay(15); // waits for the servo to get there
}
```


Obstacle Avoider using ultrasonic sensor and servo motor

obstacle_avoider_ultrasonic_servo

```
//-----Libraries-----//
#include <Servo.h>          //<-----Changes
Servo myservo;

//-----Global declaration-----//
int lm_pin1=2;              //motor
int lm_pin2=3;
int rm_pin1=4;
int rm_pin2=5;
int turn_delay= 500;
int min_dist = 10;

int servopin = 6;           //servo
int pos = 0; //servo position
int delay_per_degree = 15;

int triggerpin = 7;         //ultrasonic
int echopin = 8;
int distance,t;
int f_dist,l_dist, r_dist;
```

```
//-----function defination-----//  
void forward()  
{  
    digitalWrite(lm_pin1, 1);  
    digitalWrite(lm_pin2, 0);  
    digitalWrite(rm_pin1, 1);  
    digitalWrite(rm_pin2, 0);  
}  
void turn_left()  
{  
    digitalWrite(lm_pin1, 0);  
    digitalWrite(lm_pin2, 1);  
    digitalWrite(rm_pin1, 1);  
    digitalWrite(rm_pin2, 0);  
  
    delay(turn_delay);  
    Stop();  
}  
void turn_right()  
{  
    digitalWrite(lm_pin1, 1);  
    digitalWrite(lm_pin2, 0);  
    digitalWrite(rm_pin1, 0);  
    digitalWrite(rm_pin2, 1);  
  
    delay(turn_delay);  
    Stop();  
}  
}
```

```
}  
void Stop()  
{  
    digitalWrite(lm_pin1, 0);  
    digitalWrite(lm_pin2, 0);  
    digitalWrite(rm_pin1, 0);  
    digitalWrite(rm_pin2, 0);  
}  
//-----Changes-----//  
void move_servo(int angle1,int angle2)  
{  
    int temp_delay = delay_per_degree*abs(angle1-angle2);  
    myservo.write(angle2);  
    delay(temp_delay);  
}
```

```
int find_ultra_distance()
{
    digitalWrite(triggerpin, LOW);
    delayMicroseconds(2);
    digitalWrite(triggerpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerpin, LOW);

    digitalWrite(echopin);

    t = pulseIn(echopin, HIGH);

    distance = t*(0.034)/2;
    return distance;
}
```

```
//-----setup-----  
void setup() {  
  Serial.begin(9600);  
  
  myservo.attach(servopin); //<-----Changes  
  myservo.write(90);  
  delay(2000);  
  
  pinMode(rm_pin1, OUTPUT); //motor  
  pinMode(rm_pin2, OUTPUT);  
  pinMode(lm_pin1, OUTPUT);  
  pinMode(lm_pin2, OUTPUT);  
  
  pinMode(triggerpin, OUTPUT); //TO ULTRA SONIC SENSOR  
  pinMode(echopin, INPUT);    //FROM ULTRA SONIC SENSOR  
}
```

```
//-----loop-----//  
void loop() {  
  f_dist = find_ultra_distance();  
  
  if(f_dist < min_dist)  
  {  
    Stop();  
    move_servo(90,0);  
    l_dist = find_ultra_distance();  
    move_servo(0,180);  
    r_dist = find_ultra_distance();  
    move_servo(180,90);  
    if(l_dist < r_dist)  
    {  
      turn_right();  
    }  
    else  
    {  
      turn_left();  
    }  
  }  
  else  
  {  
    forward();  
  }  
}
```