Laboratorio 3
Asignatura: Robótica y Sistemas Autónomos
Tema: Cinemática y Control PID para un robot móvil
Objetivos

- Aprender a comunicar Arduino UNO con Python a través del puerto serial.
- Introducir conceptos de cinemática usando un robot móvil de tipo diferencial.
- Introducir conceptos de control PID

Entrega: 23 de Mayo

Parte 1: Modelo cinemático
Para esto se requiere hacer uso del laboratorio 2 donde deben considerar poner en funcionamiento los motores DC usando el chasis del robot. Los elementos que se requieren son:

(2) Motores DC
(1) Driver para motores, puede ser LN298 o LN911
(1) Arduino UNO
(1) Batería entre 9 a 12V
(1) Cables Dupont
(1) Conector de batería
(1) Regla o cinta métrica

Programa la locomoción del robot móvil asignado usando el IDE de Arduino. En el Anexo1, hay un código de apoyo para dos motores DC.

Preguntas

1- ¿Cuál es el radio (en cms) de las ruedas (R_i y y R_d) la distancia entre las ruedas base del robot móvil asignado?
2- ¿Si se quiere que el robot recorra 1metro desde un punto inicio a un punto destino de manera recta, ¿Cuál es el número de vueltas de las ruedas del robot? (para este punto considere una velocidad máxima)
3- ¿Cuál es la distancia mínima del robot?
4- ¿De acuerdo el centro de masa del robot cual es la distancia a la base del robot (parte delantera)?

Parte 2: Control PID

Vamos a realizar la siguiente comunicación con Arduino y Python a través del puerto serial. Para eso se requiere instalar la librería pyserial de Python:

Puede usar algunos de los comandos:
Opción 1: pip install PySerial
Opción 2: python -m pip install PySerial

1- Abrir el IDE de Arduino y subir el código que captura la respuesta del sensor de obstáculos:

```
#define TRIGGER_PIN  12
#define ECHO_PIN     11


long distancia;

void setup(){
  Serial.begin(9600);
  pinMode(TRIGGER_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

}

void loop(){

  distancia=Ultrasonido(TRIGGER_PIN, ECHO_PIN);
  Serial.println((distancia));

  delay(100);
}

long Ultrasonido(int trigger, int eco){

    long duration; //timepo que demora en llegar el eco
    long distance; //distancia en centimetros

     digitalWrite(trigger,LOW);
     delayMicroseconds(2);
     digitalWrite(trigger,HIGH);
     delayMicroseconds(10);
     digitalWrite(trigger,LOW);

     duration = pulseIn(eco, HIGH); //obtenemos el ancho del pulso
```

```
        distance = (duration*.0343)/2;


        return distance;
    }
```

2- Crear un Script en Python y pegar el siguiente código y probar que se está recibiendo la distancia del sensor desde Arduino UNO.

```
import serial, time
arduino = serial.Serial('COM3', 9600)
while True:
  time.sleep(2)
  data =arduino.readline()
  distancia=int(data)
  print(distancia)


arduino.close()
```

3- Use el controlador PID en el Anexo 2.
4- Configure velocidades de 3 niveles (alta, media y baja) para la locomoción del robot. Haga uso del controlador PID para controlar las velocidades (alta, media, baja) usando el sensor de obstáculos. Con la salida del controlar, controlar los niveles de velocidad.

Preguntas

- Dibuje el diagrama del control de lazo cerrado teniendo como entrada un sensor de obstáculos.
- ¿Cuál fue el rango de velocidades que ha asignado?
- ¿Cuál es el valor máximo (distancia) que logra calcular el sensor SR04?

Evaluación

1- Funcionamiento. Este se muestra el día de la entrega (40 ptos)
2- Respuesta de las preguntas (30 ptos)

Anexo 1: Código Arduino UNO Motores

```
#define  INT1  5
#define  INT2  6
#define  INT3  9
#define  INT4  10

int speedMotor = 255;

void setup() {
  pinMode(INT1,OUTPUT);
  pinMode(INT2,OUTPUT);
  pinMode(INT3,OUTPUT);
  pinMode(INT4,OUTPUT);

}

void loop() {

    motorA('R',128);
    //motorB('R',128);
}


void motorA(char d, int velocity)
{
  if(d =='A'){
    analogWrite(INT1,LOW);
    analogWrite(INT2,velocity);
  }else if (d =='R'){
    analogWrite(INT1,velocity);
    analogWrite(INT2,LOW);
  }else{
    digitalWrite(INT1,LOW);
    digitalWrite(INT2,LOW);
```

```
  }
}

void motorB(char d,int velocity)
{
  if(d =='A'){
    analogWrite(INT3,LOW);
    analogWrite(INT4,velocity);
  }else if (d =='R'){
    analogWrite(INT3,velocity);
    analogWrite(INT4,LOW);
  }else{
    analogWrite(INT3,LOW);
    analogWrite(INT4,LOW);
  }
}
```

Anexo 2: Control PID en Python

```
import time
import random


setPoint=100 #Set point for speed control, using ultrasonic sensor
time_PID = 0
integral = 0
time_prev = -1e-6
e_prev = 0
curSpeed = 0
deltat = 0.1
def PID(Kp, Ki, Kd, setpoint, measurement):
    global time_PID, integral, time_prev, e_prev
    # Value of offset - when the error is equal zero
    offset = 320


    # PID calculations
```

```
    e = setpoint - measurement

    P = Kp*e
    integral = integral + Ki*e*(time_PID - time_prev)
    D = Kd*(e - e_prev)/(time_PID - time_prev)

    # calculate manipulated variable - MV
    MV = offset + P + integral + D

    # update stored data for next iteration
    e_prev = e
    time_prev = time_PID
    return MV

if __name__=="__main__":
    n=100
    for i in range(1, n):
      distance=random.randrange(1,200)
      time_PID= i * deltat
      pid_out=PID(0.6,0.2,0.1,setPoint, distance)
      time_prev = time_PID

      print(pid_out)
      if pid_out>300:
```

```
    curSpeed=255 #velocidad alta

  if 50<pid_out<200:
    curSpeed=100 #velocidad media
  if pid_out<30:
    curSpeed=100 #velocidad baja

  time.sleep(0.5)
```