



# Tecnológico de Monterrey

**Campus Puebla**

**Materia**

Fundamentación de Robótica TE3001B

**Tema**

**Examen**

**Alumno**

Antonio Silva Martínez A01173663

**Fecha**

Mayo 16 2023

Instrucciones:

1. **Crear** un nuevo repositorio nuevo con el nombre: **Evaluación 7.1 (Trayectorias en lazo abierto)**
2. **Implementar** el código requerido para generar las siguientes trayectorias a partir del tiempo y de las velocidades angulares y lineales en un plano 2D, según corresponda. La altura de cada letra debe ser de **3m.**, el ancho puede ser ajustable **a criterio propio** y la separación entre cada letra debe ser de **0.5m.**
3. **Responder** las siguientes preguntas en base al procedimiento empleado para obtener las trayectorias propuestas:
4. **a)** ¿Cuál fué el o los parámetros que se modifican para obtener una trayectoria recta? ¿Porqué?
5. **b)** ¿Cuál fué el o los parámetros que se modifican para obtener una trayectoria curva? ¿Porqué?
6. **c)** ¿Cuál fué el o los parámetros que se modifican para obtener un giro? ¿Porqué?
7. **d)** ¿Qué papel desempeña el vector del tiempo en la generación de la trayectoria?
8. **e)** ¿Cuáles fueron los parámetros que se ajustaron para obtener las dimensiones de las trayectorias deseadas?
9. **4. Generar** un reporte de los pasos para poder obtener el recorrido de las trayectorias e incluir las respuestas de las preguntas descritas en el punto 3. Incluir el **Código en MATLAB**
10. **Subir** el link del repositorio en CANVAS para **“Evaluación”**

Preguntas:

a) ¿Cuál fué el o los parámetros que se modifican para obtener una trayectoria recta? ¿Porqué?

Los parámetros establecidos recaen en los puntos los cuales queremos tomar para hacer la recta, siendo que para poder realizar la recta es necesario que no exista velocidad angular y por ende tenemos que definir como 0 esa velocidad, también se tiene que modificar el tiempo de duración del desplazamiento ya que dependiendo el tiempo que se le dé a dicha trayectoria dependerá de la longitud de esta y también tenemos que tomar en cuenta el tiempo de muestreo ya que si cambiáramos el tiempo de muestreo el resultado quedaría muy diferente siendo que si aumentamos dicho número, el resultado que tenemos sería muy pequeño, pero si lo disminuimos, este sería más grande

b) ¿Cuál fué el o los parámetros que se modifican para obtener una trayectoria curva?  
¿Porqué?

Para obtener una trayectoria curva primero tenemos que debemos de definir el tiempo de muestreo para ver cuantas muestras contará nuestro modelo, posteriormente tomaremos en cuenta el ángulo de la curva para que de ahí pueda partir de ahí, por último tenemos que agregar un desplazamiento lineal ya que sin este, no habría ningún recorrido.

c) ¿Cuál fué el o los parámetros que se modifican para obtener un giro? ¿Porqué?

Para obtener un giro en mi modelo lo que se modificó fueron las velocidades angulares, debido a que el movimiento de la velocidad angular sin contar la velocidad lineal permite que el robot gire en su propio eje y no afecte al desplazamiento en la trayectoria

d) ¿Qué papel desempeña el vector del tiempo en la generación de la trayectoria?

El vector de tiempo toma un papel dentro del muestreo ya que nos permite el poder desplazarnos con un cierto tiempo de muestreo en determinado tiempo, siendo que estos dos van de la mano. El tiempo nos dirá cuánto tiempo se desplazará nuestro robot y en este caso si aumentamos el tiempo de muestreo la figura resultante quedará con otra forma.

e) ¿Cuáles fueron los parámetros que se ajustaron para obtener las dimensiones de las trayectorias deseadas?

Primero definimos un tiempo de muestreo de 0.1 y se tomaron 60 segundos para la simulación, posteriormente se declararon en forma de vectores las velocidades lineales y angulares para el modelo, siendo que en este caso las muestras y las velocidades van de la mano para el tiempo de desplazamiento en cada parte, por último se ajustaron los ejes y los pasos en el modelo para ajustar el "tiempo real" que tomaría la simulación en procesarse

Explicación del código:

El código utilizado se basa en el utilizado para la actividad de zigzag para manchester robotics, en donde se colocaron las velocidades angulares y lineales en un vector para posteriormente realizar una trayectoria deseada.

Los parámetros que se modificaron son el tiempo de duración del desplazamiento, el tiempo de muestreo, el tiempo de la velocidad angular, estos parámetros son esenciales para la

definición de la trayectoria del robot, debido a que estás permitirán el desplazamiento en un tiempo determinado

Código utilizado:

```
%Limpieza de pantalla
```

```
clear all
```

```
close all
```

```
clc
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% TIEMPO
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
tf=80; % Tiempo de simulación en segundos (s)
```

```
ts=0.1; % Tiempo de muestreo en segundos (s)
```

```
t=0:ts:tf; % Vector de tiempo
```

```
N= length(t); % Muestras
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% CONDICIONES INICIALES
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%Inicializamos las variables que se van a emplear
```

```
x1= zeros (1, N+1); % Posición (X) en el centro del eje que une las ruedas en metros (m)
```

```
y1= zeros (1, N+1); % Posición (Y) en el centro del eje que une las ruedas en metros (m)
```

```
phi= zeros (1, N+1); % Orientación del robot en radiaanes (rad)
```

```
%Damos valores a nuestro punto inicial de posición y orientación
```

```
x1(1)=0; %Posición inicial eje x
```

```
y1(1)=0; %Posición inicial eje y
```

```
phi(1)=0; %Orientación inicial del robot
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PUNTO DE CONTROL
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%Inicializamos el punto de control
```

```
hx= zeros (1, N+1);    % Posición en el eje (X) del punto de control en metros (m)
```

```
hy= zeros (1, N+1);    % Posición en el eje (Y) del punto de control en metros (m)
```

```
%Igualamos el punto de control con las proyecciones X1 y Y1 por su
```

```
%coincidencia
```

```
hx(1)= x1(1);    % Posición del punto de control en el eje (X) metros (m)
```

```
hy(1)= y1(1);    % Posición del punto de control en el eje (Y) metros (m)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% VELOCIDADES DE REFERENCIA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
v = [    0*ones(1,10)    1*ones(1,30)    0*ones(1,10)    1*ones(1,10)    0*ones(1,10)
1*ones(1,20)    0*ones(1,10) 1*ones(1,10)    0*ones(1,10) 1*ones(1,30)    0*ones(1,10)
1*ones(1,15)    0*ones(1,10) 1*ones(1,30)    0*ones(1,10) 1*ones(1,15)    0*ones(1,10)
1*ones(1,30)    0*ones(1,10) 1*ones(1,15)    0*ones(1,10) 1*ones(1,20)    0*ones(1,10)
1*ones(1,30)    0*ones(1,10) 1*ones(1,34)    0*ones(1,10) 1*ones(1,31)    0*ones(1,10)
1*ones(1,5)     0*ones(1,10) 1*ones(1,15)    0*ones(1,10) 1*ones(1,18)    0*ones(1,10)
1*ones(1,18)    0*ones(1,10) 1*ones(1,15)    0*ones(1,1300)]; % Velocidad lineal de
referencia (m/s)
```

```
w = [pi/2*ones(1,10)    0*ones(1,30)  pi/2*ones(1,10)    0*ones(1,10) -pi*ones(1,10)
0*ones(1,20)  pi*ones(1,10) 0*ones(1,10)  pi/2*ones(1,10) 0*ones(1,30) pi/2*ones(1,10)
0*ones(1,15) pi/2*ones(1,10) 0*ones(1,30) -pi/2*ones(1,10) 0*ones(1,15) -pi/2*ones(1,10)
0*ones(1,30) -pi/2*ones(1,10) 0*ones(1,15) -pi*ones(1,10) 0*ones(1,20) pi/2*ones(1,10)
0*ones(1,30) -6*pi/7*ones(1,10) 0*ones(1,34) 6*pi/7*ones(1,10) 0*ones(1,31)
-pi/2*ones(1,10) 0*ones(1,5) -pi/3*ones(1,10) 0*ones(1,15) -pi/6*ones(1,10) 0*ones(1,18)
pi*ones(1,10) 0*ones(1,18) -pi/6*ones(1,10) 0*ones(1,15) 0*ones(1,1300)]; % Velocidad
angular de referencia (rad/s)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% BUCLE DE SIMULACION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
for k=1:N
```

```
    %Aplico la integral a la velocidad angular para obtener el angulo "phi" de la orientación
```

```
    phi(k+1)=phi(k)+w(k)*ts; % Integral numérica (método de Euler)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% MODELO CINEMATICO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
    xp1=v(k)*cos(phi(k));
```

```
    yp1=v(k)*sin(phi(k));
```

```
    %Aplico la integral a la velocidad lineal para obtener las cordenadas
```

```
    %"x1" y "y1" de la posición
```

```
    x1(k+1)=x1(k)+ ts*xp1; % Integral numérica (método de Euler)
```

```
    y1(k+1)=y1(k)+ ts*yp1; % Integral numérica (método de Euler)
```

```
    % Posicion del robot con respecto al punto de control
```

```
    hx(k+1)=x1(k+1);
```

```
    hy(k+1)=y1(k+1);
```

```
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% SIMULACION VIRTUAL 3D
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% a) Configuracion de escena
```

```
scene=figure; % Crear figura (Escena)
```

```
set(scene,'Color','white'); % Color del fondo de la escena
```

```
set(gca,'FontWeight','bold') ;% Negrilla en los ejes y etiquetas
```

```

sizeScreen=get(0,'ScreenSize'); % Retorna el tamaño de la pantalla del computador

set(scene,'position',sizeScreen); % Configurar tamaño de la figura

camlight('headlight'); % Luz para la escena

axis equal; % Establece la relación de aspecto para que las unidades de datos sean las mismas
en todas las direcciones.

grid on; % Mostrar líneas de cuadrícula en los ejes

box on; % Mostrar contorno de ejes

xlabel('x(m)'); ylabel('y(m)'); zlabel('z(m)'); % Etiqueta de los eje

view([-0.1 35]); % Orientacion de la figura

axis([-2 15 -1 5 0 1]); % Ingresar limites minimos y maximos en los ejes x y z [minX maxX
minY maxY minZ maxZ]

% b) Graficar robots en la posicion inicial

scale = 4;

MobileRobot;

H1=MobilePlot(x1(1),y1(1),phi(1),scale);hold on;

% c) Graficar Trayectorias

H2=plot3(hx(1),hy(1),0,'r','lineWidth',2);

% d) Bucle de simulacion de movimiento del robot

step=80; % pasos para simulacion

for k=1:step:N

    delete(H1);

    delete(H2);

```

