Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Puebla

Título del Trabajo: Actividad 1

Alumnos
Antonio Mendez Rodriguez | A01738269

Materia: Fundamentación de Robótica

Clave: TE3001B

Grupo: 101

Fecha de entrega: 18 de Febrero del 2026

Reporte de actividad 1

Objetivo:
Obtener el vector de velocidades lineal y angular del efector final de un robot planar con 3 grados de libertad (3GDL), mediante el uso de matrices de transformación homogénea y el cálculo del Jacobiano.

Sección 1: Declaración de Variables
Se declaran las variables simbólicas que representan las coordenadas articulares del robot (th1, th2, th3), así como las longitudes de los eslabones (l1, l2, l3).
Estas variables permiten realizar el análisis cinemático de forma general sin asignar valores numéricos.

Sección 2: Configuración del Robot
Se define el tipo de articulación de cada grado de libertad mediante el vector:
RP = [0 0 0]
Donde:
- 0 indica una junta rotacional
- 1 indicaría una junta prismática

En este caso, el robot cuenta con tres juntas rotacionales (RRR).

Sección 3: Coordenadas Generalizadas
Se crea el vector de coordenadas articulares:
Q = [th1 th2 th3]

Este vector representa la posición angular de cada una de las articulaciones del robot.

Sección 4: Velocidades Generalizadas
Se obtiene el vector de velocidades articulares derivando respecto al tiempo:
Qp = dQ/dt

Este vector representa la velocidad angular de cada articulación.

Sección 5: Grados de Libertad
Se calcula el número de grados de libertad del robot utilizando la función size, el cual corresponde al número de juntas del manipulador.

Sección 6: Posiciones y Rotaciones
Se definen:
- Los vectores de posición Pi
- Las matrices de rotación Ri

Estas matrices describen la orientación y posición de cada eslabón respecto al anterior.

Sección 7: Inicialización

Se inicializan:

- Las matrices de transformación homogénea locales Ai
- Las matrices globales Ti
- Las matrices de rotación respecto al marco inercial ROi
- Las posiciones del efector final POi

Sección 8: Cinemática Directa
Se calculan las matrices de transformación homogénea locales y globales mediante:
Ti = Ti-1 por Ai

A partir de estas matrices se obtiene:
Po = T(1:3,4)

El cual corresponde al vector de posición del efector final respecto al marco de referencia inercial.

Sección 9: Jacobiano Diferencial
Se calcula el Jacobiano lineal derivando parcialmente la posición del efector final respecto a cada coordenada articular. Este método se basa directamente en la definición matemática del Jacobiano.

Sección 10: Jacobiano Analítico y Velocidades
En esta sección se calcula el Jacobiano de forma analítica considerando el tipo de articulación de cada junta.
Para cada grado de libertad:

Si la articulación es rotacional (RP(k) = 0):
Se calcula la contribución a la velocidad lineal mediante el producto cruz entre: El eje z de la articulación anterior y la diferencia entre la posición del efector final y la posición de la junta anterior

Este producto cruz representa la velocidad tangencial generada por una rotación.
La contribución a la velocidad angular se obtiene directamente del eje de rotación de la articulación anterior, ya que una junta rotacional genera velocidad angular alrededor de su eje.

Resultados:

```
Velocidad lineal obtenida mediante el Jacobiano lineal

/ - #4 (l3 sin(#1) + l2 sin(#2)) - #5 (l1 sin(th1(t)) + l3 sin(#1) + l2 sin(#2)) - l3 #3 sin(#1) \
|                                                                                                  |
|   #4 (l3 cos(#1) + l2 cos(#2)) + #5 (l1 cos(th1(t)) + l3 cos(#1) + l2 cos(#2)) + l3 #3 cos(#1)  |
|                                                                                                  |
\                                                0                                                 /

where

   #1 == th1(t) + th2(t) + th3(t)

   #2 == th1(t) + th2(t)

         _________
          d
   #3 == -- th3(t)
         dt

         _________
          d
   #4 == -- th2(t)
         dt

         _________
          d
   #5 == -- th1(t)
         dt
```

```
Velocidad angular obtenida mediante el Jacobiano angular

/                  0                  \
|                                     |
|                  0                  |
|                                     |
| _________    _________    _________ |
|  d            d            d        |
| -- th1(t) + -- th2(t) + -- th3(t)   |
\ dt           dt           dt        /
```

Código de Matlab:

```
%Limpieza de pantalla

clear all

close all

clc

%SECCIÓN 1

%Declaración de variables simbólicas

syms th1(t) th2(t) th3(t) t l1 l2 l3

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

```
%SECCIÓN 2

%Configuración del robot, 0 para junta rotacional, 1 para junta
prismática

RP=[0 0 0];
```

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

%SECCIÓN 3
%Creamos el vector de coordenadas articulares
Q= [th1, th2 th3];
disp('Coordenadas generalizadas');
pretty (Q);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%SECCIÓN 4
%Creamos el vector de velocidades generalizadas
Qp= diff(Q, t);
disp('Velocidades generalizadas');
pretty (Qp);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%SECCIÓN 5
%Número de grado de libertad del robot
GDL= size(RP,2);
GDL_str= num2str(GDL);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%SECCIÓN 6
%Junta 1
%Posición de la junta 1 respecto a 0
P(:,:,1)= [l1*cos(th1); l1*sin(th1);0];
%Matriz de rotación de la junta 1 respecto a 0
R(:,:,1)= [cos(th1) -sin(th1)  0;
          sin(th1)  cos(th1)  0;
          0         0         1];
```

```
%Junta 2

%Posición de la junta 2 respecto a 1

P(:,:,2)= [l2*cos(th2); l2*sin(th2);0];

%Matriz de rotación de la junta 2 respecto a 1

R(:,:,2)= [cos(th2) -sin(th2)  0;

          sin(th2)  cos(th2)  0;

          0         0         1];
```

```
%Junta 3

%Posición de la junta 3 respecto a 2

P(:,:,3)= [l3*cos(th3); l3*sin(th3);0];

%Matriz de rotación de la junta 2 respecto a 1

R(:,:,31)= [cos(th3) -sin(th3)  0;

          sin(th3)  cos(th3)  0;

          0         0         1];

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

```
%SECCIÓN 7

%Creamos un vector de ceros

Vector_Zeros= zeros(1, 3);
```

```
%Inicializamos las matrices de transformación Homogénea locales

A(:,:,GDL)=simplify([R(:,:,GDL) P(:,:,GDL); Vector_Zeros 1]);

%Inicializamos las matrices de transformación Homogénea globales

T(:,:,GDL)=simplify([R(:,:,GDL) P(:,:,GDL); Vector_Zeros 1]);

%Inicializamos las posiciones vistas desde el marco de referencia
inercial

PO(:,:,GDL)= P(:,:,GDL);
```

```
%Inicializamos las matrices de rotación vistas desde el marco de referencia inercial

RO(:,:,GDL)= R(:,:,GDL);

%Inicializamos las INVERSAS de las matrices de rotación vistas desde el marco de referencia inercial

RO_inv(:,:,GDL)= R(:,:,GDL);

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

```
%SECCIÓN 8

for i = 1:GDL

   i_str= num2str(i);

   %Locales

   disp(strcat('Matriz de Transformación local A', i_str));

   A(:,:,i)=simplify([R(:,:,i) P(:,:,i); Vector_Zeros 1]);

   pretty (A(:,:,i));
```

```
  %Globales

   try

      T(:,:,i)= T(:,:,i-1)*A(:,:,i);

   catch

      T(:,:,i)= A(:,:,i);

   end

   disp(strcat('Matriz de Transformación global T', i_str));

   T(:,:,i)= simplify(T(:,:,i));

   pretty(T(:,:,i))
```

```
   RO(:,:,i)= T(1:3,1:3,i);

   RO_inv(:,:,i)= transpose(RO(:,:,i));

   PO(:,:,i)= T(1:3,4,i);

  %pretty(RO(:,:,i));
```

```
  % pretty(RO_inv(:,:,i));

  % pretty(PO(:,:,i));

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Calculamos la matriz de transformación del marco de referencia
inercial

%visto desde el actuador final

% disp(strcat('Matriz de Transformación T', GDL_str,'_O calculada de
forma manual'));

% RF_O=RO_inv(:,:,GDL);

% PF_O=-RF_O*PO(:,:,GDL);

% TF_O= simplify([RF_O PF_O; Vector_Zeros 1]);

% pretty(TF_O);
```

```
%disp(strcat('Matriz de Transformación T', GDL_str,'_O calculada de
forma auntomática'));

%pretty(simplify(inv(T(:,:,GDL))));
```

```
%SECCIÓN 9

%Calculamos el jacobiano lineal de forma diferencial

disp('Jacobiano lineal obtenido de forma diferencial');

%Derivadas parciales de x respecto a th1 y th2

Jv11= functionalDerivative(PO(1,1,GDL), th1);

Jv12= functionalDerivative(PO(1,1,GDL), th2);

Jv13= functionalDerivative(PO(1,1,GDL), th2);
```

```
%Derivadas parciales de y respecto a th1 y th2

Jv21= functionalDerivative(PO(2,1,GDL), th1);

Jv22= functionalDerivative(PO(2,1,GDL), th2);

Jv23= functionalDerivative(PO(2,1,GDL), th3);
```

```
%Derivadas parciales de z respecto a th1 y th2
Jv31= functionalDerivative(PO(3,1,GDL), th1);
Jv32= functionalDerivative(PO(3,1,GDL), th2);
Jv33= functionalDerivative(PO(3,1,GDL), th3);
```

```
%Creamos la matríz del Jacobiano lineal
jv_d=simplify([Jv11 Jv12 Jv13;
              Jv21 Jv22 Jv23;
              Jv31 Jv32 Jv33]);
pretty(jv_d);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%SECCIÓN 10
%Calculamos el jacobiano lineal de forma analítica
Jv_a(:,GDL)=PO(:,:,GDL);
Jw_a(:,GDL)=PO(:,:,GDL);
```

```
for k= 1:GDL
   if RP(k)==0 %Casos: articulación rotacional
      %Para las juntas de revolución
       try
           Jv_a(:,k)= cross(RO(:,3,k-1), PO(:,:,GDL)-PO(:,:,k-1));
           Jw_a(:,k)= RO(:,3,k-1);
       catch
           Jv_a(:,k)= cross([0,0,1], PO(:,:,GDL));
           Jw_a(:,k)=[0,0,1];
       end

       %Para las juntas prismáticas
```

```
    elseif RP(k)==1 %Casos: articulación prismática
%
       try
           Jv_a(:,k)= RO(:,3,k-1);
       catch
           Jv_a(:,k)=[0,0,1];
       end
           Jw_a(:,k)=[0,0,0];
    end
end
```

```
Jv_a= simplify (Jv_a);
Jw_a= simplify (Jw_a);
disp('Jacobiano lineal obtenido de forma analítica');
pretty (Jv_a);
disp('Jacobiano ángular obtenido de forma analítica');
pretty (Jw_a);
```

```
disp('Velocidad lineal obtenida mediante el Jacobiano lineal');
V=simplify (Jv_a*Qp');
pretty(V);
disp('Velocidad angular obtenida mediante el Jacobiano angular');
W=simplify (Jw_a*Qp');
pretty(W);
```

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