Actividad 5 | Robot 1 (3 GDL)

Oscar Ortiz Torres A01769292

```
clc
clear all
close all
```

Para encontrar la cinemática directa, se establecen los diferentes sistemas de coordenadas para cada articulación, así como las variables para los ángulos de giros y de las distancias que pueden recorrer los eslabones.

```
syms th1(t) l0 l1(t) l2(t)  t

% Configuración del robot, de cada GDL
RP = [0 1 1];

% Creación del vector de coordenadas articulares
Q = [th1 l1 l2];

% Creación del vector de velocidades generalizadas
Qp = diff(Q, t);

% Número de grados de libertad
GDL = size(RP, 2);
GDL_str = num2str(GDL);
```

Posteriormente, se determinan los sistemas de coordenadas, haciendo que, en el caso de las rotacionales, el eje Z siempre concuerde con el eje de rotación de la junta, y en el caso de las prismáticas, el eje Z corresponda con el eje en el que se traslada la junta.

De esta forma, se analizan las rotaciones necesarias para pasar de un sistema de coordenadas a otro de manera secuencial, siempre considerando cada articulación como independiente y siguiendo la regla de la mano derecha.

Eslabón 1

Se interpretó que el sistema 0 es la base del robot, por la naturaleza de la junta, el valor entre este sistema y el de la primera junta prismática es constante en Z en su vector de posición (o en caso de no haber distancia entre sistemas de coordenadas, se puede interpretar como 0)

```
% Posición de la articulación 1 respecto a 0
P(:,:,1) = [
            0;
            0;
            l0
           ];
```

Se definió igual el sistema de coordenadas para el 0 y el 1, por lo que no es necesario aplicar rotaciones.

```
% Matriz de rotación de la junta 0 a 1
R(:,:,1) = [
            cos(th1)   -sin(th1)     0;
            sin(th1)    cos(th1)     0;
            0           0            1
            ];
```

Eslabón 2

Esta junta prismática se traslada sobre l1, por lo que permanece constante en Z dentro de su vector de posición

```
% Posición de la articulación 2 respecto a 1
P(:,:,2) = [
            0;
            0;
            l1
           ];
```

Para pasar del sistema de coordenadas 1 al 2, se debe aplicar una rotación de -90° en el eje X. El signo debido a la regla de la mano derecha.

```
% Matriz de rotación de la junta 1 a 2
R(:,:,2) = rotX(-90);
```

Eslabón 3

Igual que la junta anterior, por ser del mismo tipo, se traslada sobre l2, por lo que permanece constante en Z dentro de su vector de posición

```
% Posición de la articulación 3 respecto a 2
P(:,:,3) = [
            0;
            0;
            l2
           ];
```

Al no haber otro sistema de referencia al cual llegar, no es necesario aplicar rotaciones en esta matriz, y como naturalmente no rota este tipo de juntas, se obtiene una matriz identidad.

```
% Matriz de rotación de la junta 2
R(:,:,3) = rotZ(0);
```

Matrices

Una vez definidas los vectores de posición y las matrices de rotación de cada articulación, se arman las 3 matrices de tranformación homogenea locales y, al multiplicarlas entre ellas, se obtiene la matriz de tranformación homogenea global.

```
% Creación de vector de ceros
```

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

% Inicialización de las matrices de Transformación Homogenea locales
A(:, :, GDL) = simplify([ ...
                        R(:,:,GDL)      P(:,:,GDL); ...
                        vector_zeros    1 ...
                        ]);

% Inicialización de las matrices de tranformación Homogenea globales
T(:,:,GDL) = simplify([ ...
                        R(:,:,GDL)      P(:,:,GDL); ...
                        vector_zeros    1 ...
                        ]);

% Inicialización de los vectores de posición vistos desde el marco de
% referencia inercial
PO(:,:,GDL) = P(:,:,GDL);

% Inicialización de las matrices de rotación vistas desde el marco de
% referencia inercial
RO(:,:,GDL) = R(:,:,GDL);

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 ...
                        ]); A(:,:,i)

    % Globales
    try
        T(:,:,i) = T(:,:,i-1)*A(:,:,i);
    catch
        T(:,:,i) = A(:,:,i); % Caso específico cuando i=1 nos marcaría error en try
    end

    %disp(strcat('Matruz de Transformación global T', i_str));
    T(:,:,i) = simplify(T(:,:,i));

    % Obtención de la matriz de rotación "RO" y el vector de traslación PO
    % de la matriz de transformación homogenea global T(:,:, GDL)
    RO(:,:,i) = T(1:3,1:3,i);
    PO(:,:,i) = T(1:3,4,i);
end
```

```
Matriz de Transformación local A1
ans =
```

$$\begin{pmatrix} \cos(\mathrm{th}_1(t)) & -\sin(\mathrm{th}_1(t)) & 0 & 0 \\ \sin(\mathrm{th}_1(t)) & \cos(\mathrm{th}_1(t)) & 0 & 0 \\ 0 & 0 & 1 & l_0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
Matriz de Transformación local A2
ans =
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & l_1(t) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
Matriz de Transformación local A3
ans =
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_2(t) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

La matriz de transformación homogenea global contiene las tranformaciones de rotación y traslación necesarias para poder encontrar exactamente donde se encuentra el efector final respecto al origen del robot, y así también poder manipularlo de manera exacta y eficiente.

Se observa que en el eje Z de la matriz de traslación es la suma de la distancia l1 y l0, siendo la primera dependiente del tiempo y la segunda una constante, esto por la naturaleza de la junta a la que pertenecen estas variables.

```
disp('Matriz de Transformación global'); T(:,:,GDL)
```

```
Matriz de Transformación global
ans =
```

$$\begin{pmatrix} 0 & \cos(\mathrm{th}_1(t)) & -\sin(\mathrm{th}_1(t)) & -\sin(\mathrm{th}_1(t))\, l_2(t) \\ 0 & \sin(\mathrm{th}_1(t)) & \cos(\mathrm{th}_1(t)) & \cos(\mathrm{th}_1(t))\, l_2(t) \\ 1 & 0 & 0 & l_0 + l_1(t) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Para poder calcular los vectores de velocidad linear y angular, es necesario calcula el Jacobiano, en este caso se determina de forma analitica.

```
% Inicialización de jacobianos analíticos (lineal y angular)
Jv_a(:,GDL) = PO(:,:,GDL);
Jw_a(:,GDL) = PO(:,:,GDL);

for k = 1:GDL
    if(RP(k) == 0)
        % Para las articulaciones rotacionales
        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)); % Matriz de rotación de 0 con
respecto a 0 es la Matriz Identidad, la posición previa tambien será 0
            Jw_a(:,k) = [0 0 1]; % Si no hay matriz de rotación previa se obtiene
la matriz identidad
        end
    elseif(RP(k) == 1)
        % Para las articulaciones prismáticas
        try
            Jv_a(:,k) = RO(:,3,k-1);
        catch
            Jv_a(:,k) = [0,0,1]; % Si no hay matriz de rotación previa se obtiene
la matriz identidad
        end
            Jw_a(:,k) = [0 0 0];
    end
end
```

Despliegue

El vector de velocidad lineal representa el movimiento existente en cada uno de los ejes respecto al sistema de coordenadas original de la base del robot.

Se observa una descomposición de componentes en los ejes X y Y, esto debido a que el robot gira desde su base, por lo que la posición de la segunda junta prismáticas está en el plano XY.

Por otro lado, en Z solo se desplaza la primera junta prismátiva

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

```
Velocidad lineal obtenida mediante el Jacobiano lineal
V(t) =
```

$$\begin{pmatrix} -\overline{\frac{\partial}{\partial t}\, l_2(t)}\,\sin(\mathrm{th}_1(t)) - \overline{\frac{\partial}{\partial t}\,\mathrm{th}_1(t)}\,\cos(\mathrm{th}_1(t))\,l_2(t) \\ \overline{\frac{\partial}{\partial t}\, l_2(t)}\,\cos(\mathrm{th}_1(t)) - \overline{\frac{\partial}{\partial t}\,\mathrm{th}_1(t)}\,\sin(\mathrm{th}_1(t))\,l_2(t) \\ \overline{\frac{\partial}{\partial t}\, l_1(t)} \end{pmatrix}$$

El vector de velocidad angular representa el único giro que existe en este sistema, como se mencionó anteriormente, existe solo en la base del robot en el eje Z.

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

```
Velocidad angular obtenida mediante el Jacobiano angular
W(t) =
```

$$\begin{pmatrix} 0 \\ 0 \\ \frac{\partial}{\partial t}\,\mathrm{th}_1(t) \end{pmatrix}$$

Funciones de rotacion

```
function r_x = rotX(th)
    r_x = [1 0 0; 0 cosd(th) -sind(th); 0 sind(th) cosd(th)];
end

function r_y = rotY(th)
    r_y = [cosd(th) 0 sind(th); 0 1 0; -sind(th) 0 cosd(th)];
end

function r_z = rotZ(th)
    r_z = [cosd(th) -sind(th) 0; sind(th) cosd(th) 0; 0 0 1];
end
```