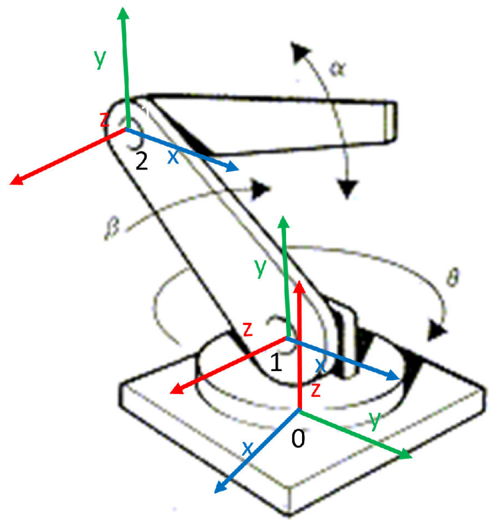
# Actividad 4 | Robot Angular 3 GDL

Oscar Ortiz Torres A01769292




Se declaran las distancias y ángulos de las 3 articulaciones rotacionales

```
clear all
close all
clc

syms theta(t) beta(t) alpha(t) l0 l1 l2 t

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

% Creación del vector de coordenadas articulares
Q = [theta beta alpha];

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

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

## Articulación 0

Al girar unicamente sobre la base, la distancia que exista entre la articulación 0 y 1 se queda como constante en Z de su vector de posición.

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

```
            l0
        ];
```

Para pasar del sistema de coordenadas 0 a 1 es necesario realizar 2 transformaciones de rotación, una de +90° en el eje X y otra de +90° en Y, aplicandolas a la matriz de rotación en Z propia del movimiento de la articulación. Los signos de las transformaciones son positivos por la regla de la mano derecha.

```
% Matriz de rotación de la junta 0 a 1
R(:,:,1) = [
             cos(theta)   -sin(theta)     0;
             sin(theta)    cos(theta)     0;
             0            0             1
             ]*rotX(90)*rotY(90);
```

## Articulación 1

Después de la transformación anterior, la rotación en Z afecta directamente a la posición final del eslabón, por lo que se necesita desomponer en su componente en X y su componente en Y en su vector de posición.

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

Al no existir un nuevo sistema de coordenadas para manejar el siguiente eslabón, se mantiene la matriz de rotación en Z propia del movimiento de la articulación.

```
% Matriz de rotación de la junta 1 a 2
R(:,:,2) = [
             cos(beta)   -sin(beta)     0;
             sin(beta)    cos(beta)     0;
             0            0             1
             ];
```

## Articulación 2

Similar a la articulación anterior, la rotación en Z afecta directamente a la posición del efector final, por lo que se necesita desomponer en su componente en X y su componente en Y en su vector de posición.

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

Al no existir un nuevo eslabón, se mantiene la matriz de rotación en Z propia del movimiento de la articulación.

```
% Matriz de rotación de la junta 2 a 3
R(:,:,3) = [
             cos(alpha)   -sin(alpha)     0;
             sin(alpha)    cos(alpha)     0;
             0           0            1
             ];
```

## Matrices

```
% 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 =
```

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

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

$$\begin{pmatrix} \cos(\beta(t)) & -\sin(\beta(t)) & 0 & l_1 \cos(\beta(t)) \\ \sin(\beta(t)) & \cos(\beta(t)) & 0 & l_1 \sin(\beta(t)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

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

$$\begin{pmatrix} \cos(\alpha(t)) & -\sin(\alpha(t)) & 0 & l_2 \cos(\alpha(t)) \\ \sin(\alpha(t)) & \cos(\alpha(t)) & 0 & l_2 \sin(\alpha(t)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

En la matriz de transformación global del sistema, en la matriz de rotación se observa la propagación de las transformación que se realizaron para pasar del sistema 0 al 1, por otro lado, el vector de posisión involucra en en sus diferentes ejes a los 3 angulos del sistema, mientras que en X y Y solo involucra a las distancias del eslabon 1 y 2, pues son las que rotan en el mismo plano.

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

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

$$\begin{pmatrix} -\sin(\theta(t))\cos(\sigma_2) & \sin(\theta(t))\sin(\sigma_2) & \cos(\theta(t)) & -\sin(\theta(t))\,\sigma_1 \\ \cos(\theta(t))\cos(\sigma_2) & -\cos(\theta(t))\sin(\sigma_2) & \sin(\theta(t)) & \cos(\theta(t))\,\sigma_1 \\ \sin(\sigma_2) & \cos(\sigma_2) & 0 & l_0 + l_1\sin(\beta(t)) + l_2\sin(\sigma_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = l_1\cos(\beta(t)) + l_2\cos(\sigma_2)$$

$$\sigma_2 = \alpha(t) + \beta(t)$$

```
% 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

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

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

$$\begin{pmatrix} \sigma_1 \sin(\theta(t))\,\sigma_5 - \sigma_3 \cos(\theta(t))\,\sigma_4 + l_2\,\sigma_2 \sin(\theta(t)) \sin(\sigma_6) \\ -\sigma_1 \cos(\theta(t))\,\sigma_5 - \sigma_3 \sin(\theta(t))\,\sigma_4 - l_2\,\sigma_2 \cos(\theta(t)) \sin(\sigma_6) \\ l_1\,\sigma_1 \cos(\beta(t)) + l_2\,\sigma_2 \cos(\sigma_6) + l_2\,\sigma_1 \cos(\sigma_6) \end{pmatrix}$$

where

$$\sigma_1 = \overline{\frac{\partial}{\partial t}\,\beta(t)}$$

$$\sigma_2 = \overline{\frac{\partial}{\partial t}\,\alpha(t)}$$

$$\sigma_3 = \overline{\frac{\partial}{\partial t}\,\theta(t)}$$

$$\sigma_4 = l_1 \cos(\beta(t)) + l_2 \cos(\sigma_6)$$

$$\sigma_5 = l_1 \sin(\beta(t)) + l_2 \sin(\sigma_6)$$

$$\sigma_6 = \alpha(t) + \beta(t)$$

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

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

$$\begin{pmatrix} \cos(\theta(t)) \left(\overline{\frac{\partial}{\partial t}\,\alpha(t)} + \overline{\frac{\partial}{\partial t}\,\beta(t)}\right) \\ \sin(\theta(t)) \left(\overline{\frac{\partial}{\partial t}\,\alpha(t)} + \overline{\frac{\partial}{\partial t}\,\beta(t)}\right) \\ \overline{\frac{\partial}{\partial t}\,\theta(t)} \end{pmatrix}$$

## 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
```