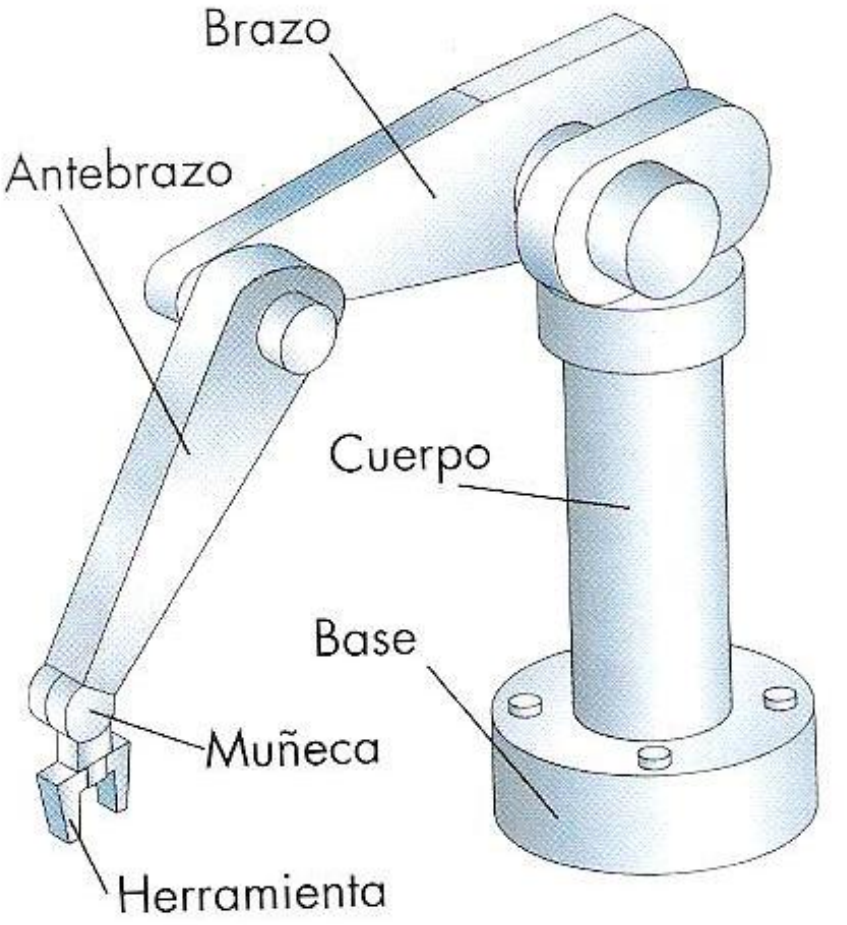
Antonio Silva Martínez - Frida Lizett Zavala Pérez - José Jezarel Sánchez Mijares

Actividad 1 (Velocidades Lineales y angulares)

25/02/2023

Los robots son dispositivos programables y automatizados que pueden realizar tareas específicas de manera autónoma o bajo la supervisión humana. En particular, los robots de 3GDL tienen tres articulaciones que les permiten moverse en tres direcciones diferentes: rotación en torno a un eje, inclinación hacia arriba y hacia abajo, y movimiento hacia adelante y hacia atrás.

Para controlar el movimiento de un robot de 3GDL, se utilizan matrices de transformación homogéneas. Estas matrices son herramientas matemáticas que permiten describir la posición y orientación de un objeto en el espacio tridimensional. Cada matriz representa una transformación que lleva el robot desde una posición inicial hasta una posición final en el espacio. Al multiplicar varias matrices juntas, se puede describir el movimiento completo del robot en términos de las tres articulaciones.



Este código inicializa algunas variables simbólicas, calcula algunas cantidades básicas para un robot de 3 grados de libertad y realiza la creación de algunos vectores.

```
%Limpieza de pantalla
clear all
close all
clc

%Declaración de variables simbólicas
syms th1(t) th2(t) th3(t) t a1 a2 a3

%Configuración del robot, 0 para junta rotacional, 1 para junta prismática
RP=[0 0 0];

%creación de vector de cordenadas articulares
Q= [th1, th2, th3];
%disp('cordenadas articulares');
%pretty(Q);

%creación del vector de velocidades angulares
Qp= diff(Q, t);
%disp('velocidades articulares');
%pretty (Qp);

%Creamos el vector de grado de libertad del robot
GDL= size(RP,2);%Se coloca el numero 2 por que indica la dimencion de las columnas
GDL_str= num2str(GDL);
```

Se crea el modelado de las articulaciones a travez de vectores y matrices con respecto a un eje, en este caso el eje correspondiente es el eje z. Estas ariculaciones se toman en cuenta una especto a la otra, es decir la articulación 3 se visualiza con respecto a la articulación 2 y así respectivamente

```
%Articulación 1
%Posición de la articulación 1 respecto a 0
%es para los es para la proyeccion en x y en y
P(:,:,1)= [a1*cos(th1);
           a1*sin(th1);
                    0];
%Matriz de rotación de la junta 1 respecto a 0
R(:,:,1)= [cos(th1) -sin(th1)  0;%Analisis de robot péndulo
           sin(th1)  cos(th1)  0;
           0         0         1];

%Articulación 2
%Posición de la articulación 2 respecto a 1
P(:,:,2)= [a2*cos(th2);
           a2*sin(th2);
                    0];
```

```
%Matriz de rotación de la junta 1 respecto a 0
R(:,:,2)= [cos(th2) -sin(th2)  0;%Analisis de robot péndulo
           sin(th2)  cos(th2)  0;
           0         0         1];

%Articulación 3
%Posición de la articulación 3 respecto a 2
P(:,:,3)= [a3*cos(th3);
           a3*sin(th3);
                    0];
%Matriz de rotación de la junta 1 respecto a 0
R(:,:,3)= [cos(th3) -sin(th3)  0;%Analisis de robot péndulo
           sin(th3)  cos(th3)  0;
           0         0         1];
```

Para poder ver el modelado de las matrices necesitamos inicializar las matrices de transfomación, esto se realizará de dos formas, una de forma local y otra de forma global, esto se realiza para poder ver el modelo resultante. Posteriormente se iteran las articulaciones del robot y se crea una cadena de caracteres para poder crear una matriz.

```
%Creación de 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);

for i = 1:GDL
    i_str= num2str(i);
   %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);
    PO(:,:,i)= T(1:3,4,i);
    %pretty(RO(:,:,i));
    %pretty(PO(:,:,i));
end
```

Se calcula el Jacobiano lineal de un robot manipulador de 3 grados de libertad (GDL) de dos formas diferentes: de forma diferencial y de forma analítica.

Primero, se definen variables simbólicas para las coordenadas articulares y se configura el robot mediante la matriz RP, que indica si cada junta es rotacional o prismática. Luego, se crean matrices de transformación local y global para cada junta y se obtienen las matrices de rotación y posición correspondientes.

Después, se calcula el Jacobiano lineal de forma diferencial mediante las derivadas parciales de la posición del extremo del robot con respecto a cada coordenada articular. Se crea la matriz del Jacobiano lineal y se muestra su resultado.

Finalmente, se calcula el Jacobiano lineal de forma analítica para cada junta, considerando si es rotacional o prismática. Para las juntas rotacionales, se usa la relación entre la velocidad angular y lineal, mientras que para las juntas prismáticas solo se considera la velocidad lineal. Los resultados se guardan en las matrices Jv_a y Jw_a.

```
%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), th3);
%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);


%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
       %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));%Matriz de rotación de 0 con respect
            Jw_a(:,k)=[0,0,1];%Si no hay matriz de rotación previa se obtiene la Matriz
         end
     else
%         %Para las juntas prismáticas
        try
            Jv_a(:,k)= RO(:,3,k-1);
        catch
            Jv_a(:,k)=[0,0,1];
        end
            Jw_a(:,k)=[0,0,0];
     end
end
```

Por ultimo se despliegan los resultados obtenidos de las matrices.

```
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);
```

Concluciones:

El análisis de los grados de libertad y el número de articulaciones de un robot es importante para entender su capacidad de movimiento y su flexibilidad para realizar diferentes tareas, mientras que el número de articulaciones determina el número de segmentos móviles del robot.

Por lo tanto, la combinación de ambos factores define la complejidad y versatilidad del robot, así como su capacidad para realizar tareas específicas. Un mayor número de grados de libertad y articulaciones aumentará

la capacidad del robot para realizar movimientos complejos y precisos en diferentes direcciones, mientras que un menor número limitará su capacidad de movimiento y tareas posibles.