A01735217 - Diego García Rueda

A01734225 - Jonathan Josafat Vázquez Suárez

A01734193 - Jhonatan Yael Martinez Vargas

DESARROLLO DE LA ACTIVIDAD:

Se usan comandos de limpieza de *workspace*

```
clear all
close all
clc
```

Se genera el comando *tic* para inicializar el tiempo de ejecución del programa

```
tic
```

Se crean variables simbolicas que se usarán en todo el programa.

```
syms th1(t) t cero
```

Se crean las variables simbolicas de las masas y las matrices de inercia. (h es la altura del soporte)

```
syms m1 Ixx1 Iyy1 Izz1 g h
```

Se crean las variables simbolicas que representan la longitud de cada junta y la distancia de cada junta al centro de masa.

```
syms l1 lc1
```

Se establece la configuración del robot, en este caso las 3 articulaciones son prismaticas.

```
RP=[0];
```

Se crea un vector de coordenadas particulares. Posteriormente este vector es derivado para obtener un vector de velocidades articulares y aceleraciones articulares.

```
Q = [th1];
syms th1p(t)
Qp = [th1p];
syms th1pp(t)
Qpp = [th1pp];
```

Se establece el número de los Grados De Libertad (GDL) tanto como valor numerico como dato de tipo *String*

```
GDL= size(RP,2);
GDL_str= num2str(GDL);
```

ARTICULACIÓN 1

Se crea el vector de traslación de la junta 1. Se crea también la matriz de rotación respecto al origen.(-90° en el eje X) Ambos valores se guardan en la página 1.

```
P(:,:,1)= [l1*cos(th1); l1*sin(th1); 0];
R(:,:,1)= [cos(th1) -sin(th1) 0;
           sin(th1) cos(th1)  0;
              0        0      1];
```

CREACIÓN DE MATRICES GLOBALES Y LOCALES

Se crea un vector de ceros, Este vector se usa para "completar" las matrices homogéneas locales y globales, este vector permite que esta matriz sea una matriz cuadrada.

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

Se inicializan las matrices de transformacion locales y global usando los vectores de traslación y las matrices de rotación. Para realizar las matrices cuadradas se agrega el vector de ceros.

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

Se inicializan los valores de los vectores de posición y de rotación (vistos desde el marco de referencia inercial).

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

Se realiza un ciclo *for* en el que usando los Grados De LIbertad (GDL) como iterador. Usando este iterador se van creando las matrices locales de cada junta.

Posteriormente se vn creando las matrices de transformación globales. Aqui existen 2 casos en los que se pueden generar estas matrices:

- Caso #1: Solo hay 1 GDL por lo que la matriz global es igual a la matriz de transformación local de esa junta.
- Caso #2: Hay más de 1 GDL, en eset caso, la matriz global se genera al multiplicar la matriz global anterior (es decir, de la articulacion anterior) por la matriz local actual (de la articulación actual) generando asi la nueva matriz de tranformación global.

Obtenemos la matriz de rotación (RO) y el vector de traslación (PO) a partir de la Matriz de Transformación Homogénea Global.

```
for i = 1:GDL
    i_str= num2str(i);
    %Matrices Locales
    A(:,:,i)=simplify([R(:,:,i) P(:,:,i); Vector_Zeros 1]);

    %Matrices Globales
```

```
    try
       T(:,:,i)= T(:,:,i-1)*A(:,:,i);
    catch
       T(:,:,i)= A(:,:,i);  % GDL == 1
    end

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

    %Da la matriz de rotacion de la matriz global
    RO(:,:,i)= T(1:3,1:3,i);
    %Vector de traslación de la matriz global
    PO(:,:,i)= T(1:3,4,i);

end
```

CALCULO DE JACOBIANOS

Método de diferenciación:

Primero obtenemos el Jacobiano Lineal usando el método de diferenciación. Se realizan derivadas parciales respecto a x, y, z obteniendo asi 3 derivadas parciales. Para esto se usan los vectores de posición

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

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

Jv31= functionalDerivative(PO(3,1,GDL), th1);
```

Se crea la matriz del Jacobiano lineal.

```
jv_d=simplify([Jv11; Jv21; Jv31]);
```

Método Análitico:

Se inicializan los Jacobianos Análiticos (Lineal y Angular). Para esto se utiliza una funcion llamada *calculo_analitico_Jacobianos* quedando de la siguiente manera:

```
[Jv_a, Jw_a] = calculo_analitico_Jacobianos(PO, RO, RP, GDL);
```

En la sección de *try* se encuentran las formulas del Jacobiano Lineal Analiticoo y del Jacobiano Angular Analitico respectivamente.

En la sección de *catch* se encuentran las formulas para los Jacobianos en caso de que solo haya 1 GDL, debido a que no tenemos articulaciones previas usando asi una matriz identidad.

Se simplifican los Jacobianos obtenidos:

```
Jv_a= simplify (Jv_a);
Jw_a= simplify (Jw_a);
```

Se mandan a imprimir las velocidades lineal y angular obtenidas usando sus Jacobianos correspondientes diferenciandolos.

```
disp('Velocidad lineal obtenida mediante el Jacobiano lineal (Última junta)');
```

```
Velocidad lineal obtenida mediante el Jacobiano lineal (Última junta)
```

```
V=simplify (Jv_a*Qp');
pretty(V);
```

```
/      _______             \
| -l1 th1p(t) sin(th1(t)) |
|                          |
|      _______             |
|  l1 th1p(t) cos(th1(t)) |
|                          |
\             0            /
```

```


disp('Velocidad angular obtenida mediante el Jacobiano angular (Última junta)');
```

```
Velocidad angular obtenida mediante el Jacobiano angular (Última junta)
```

```
W=simplify (Jw_a*Qp');
pretty(W);
```

```
/    0    \
|         |
|    0    |
|         |
|  _______ |
\ th1p(t) /
```

ENERGÍA CINÉTICA

Se crean los vectores de posición (distancia del origen de la junta a su centro de masa).

```
P01=subs(P(:,:,1)/2, l1, lc1);
```

Se crean las matrices de inercia de cada junta.

```
I1=[Ixx1 0 0;
    0 Iyy1 0;
    0 0 Izz1];
```

Se extraen las velocidades lineales y angulares de cada eje.

```
%Velocidades lineales
V=V(t);
Vx= V(1,1);
Vy= V(2,1);
Vz= V(3,1);

%Velocidades angulares
W=W(t);
```

```
W_pitch= W(1,1);
W_roll= W(2,1);
W_yaw= W(3,1);
```

CALCULO ENERGÍA CINÉTICA DE CADA JUNTA

ESLABON 1:

Usando las formulas de energía cinetica

```
V1_Total = V+cross(W,P01);
pretty(V1_Total*m1)
```

```
/     /                              _______          \ \
|     |    _______                lc1 th1p(t) sin(th1(t)) | |
| -m1 | l1 th1p(t) sin(th1(t)) + ----------------------- | |
|     \                                    2            / |
|                                                         |
|     /                              _______          \ |
|     |    _______                lc1 th1p(t) cos(th1(t)) | |
|  m1 | l1 th1p(t) cos(th1(t)) + ----------------------- | |
|     \                                    2            / |
|                                                         |
\                              0                         /
```

```
pretty((1/2*m1*(V1_Total))')
```

```
/      /                              ______  ___          \     /                              ______  ___
|   __ |       ______  __              sin(th1(t)) lc1 th1p(t) |  __ |       ______  __              cos(th1(t)) lc1
|   m1 | sin(th1(t)) l1 th1p(t) + ----------------------- |  m1 | cos(th1(t)) l1 th1p(t) + ---------------
|      \                                    2            /     \                                    2
| - -----------------------------------------------------, ---------------------------------------------
\                              2                                                        2
```

```
%K1 = (1/2*m1*(V1_Total.^2))+(1/2*I1*(W.^2))
K1 = (1/2*m1*(V1_Total))'*(1/2*m1*(V1_Total)) + (1/2*W)'*(I1*W);
K1 = simplify (K1);
disp('Energía Cinética de la Junta 1:')
```

```
Energía Cinética de la Junta 1:
```

```
pretty(K1)
```

```
              2             2     ______                     2          2             2
Izz1 |th1p(t)|    |th1p(t)|  cos(th1(t) - th1(t)) |m1|  (l1 |lc1|  + 2 lc1 |l1| ) (2 l1 + lc1)
--------------- + ---------------------------------------------------------------------------
       2                                          16 l1 lc1
```

SUMA ENERGÍAS CINETICAS

Se suman las energías cineticas de todos los eslabones quedando de la siguiente manera:

```
K_Total= simplify (K1);
disp('Energía Cinética Total (todas las juntas)')
```

```
Energía Cinética Total (todas las juntas)
```

```
pretty(K_Total)
```

```
              2            2      ______                2          2             2
Izz1 |th1p(t)|    |th1p(t)|  cos(th1(t) - th1(t)) |m1|  (l1 |lc1|  + 2 lc1 |l1| ) (2 l1 + lc1)
--------------- + --------------------------------------------------------------------------
       2                                            16 l1 lc1
```

CALCULO ENERGÍA POTENCIAL TOTAL:

Primero se obtienen las alturas respecto a la gravedad de cada una de las juntas.

```
h1= P01(2) + h;
```

A partir de las alturas se calcula la energía potencial. Utilizando como parametros la masa del eslabon, su altura y la garvedad

```
U1=m1*g*h1;
```

Se calcula la energía potencial total sumando la energia potencial de cada eslabon, quedando de la siguiente manera:

```
U_Total= U1
```

```
U_Total =
```

$$g\,m_1\left(h+\frac{\mathrm{lc}_1\sin(\mathrm{th}_1(t))}{2}\right)$$

Se manda a imprimir la suma de la energía cinética total y de la energía potencial total.

```
H= simplify (K_Total+U_Total);
  pretty (H)
```

```
              2
Izz1 |th1p(t)|         /     lc1 sin(th1(t)) \
--------------- + g m1 | h + --------------- |
       2               \            2        /

              2      ______                2          2             2
     |th1p(t)|  cos(th1(t) - th1(t)) |m1|  (l1 |lc1|  + 2 lc1 |l1| ) (2 l1 + lc1)
   + --------------------------------------------------------------------------
                                    16 l1 lc1
```

CALCULO DEL LANGRAGIANO

Ya teniendo los valores de la energia cinetica total y de la energía potencial total podemos realizar el calculo del Langragiano:

```
Lagrangiano= simplify (K_Total-U_Total);
pretty (Lagrangiano);
```

```
              2
Izz1 |th1p(t)|         /     lc1 sin(th1(t)) \
--------------- - g m1 | h + --------------- |
       2               \            2        /

              2      ______                2          2             2
     |th1p(t)|  cos(th1(t) - th1(t)) |m1|  (l1 |lc1|  + 2 lc1 |l1| ) (2 l1 + lc1)
```

```
+ ---------------------------------------------------------------------------
                                    16 l1 lc1
```

ECUACIONES DE MOVIMIENTO:

Se define un vector columna de derivadas con respecto al tiempo:

```
Qd = [th1p(t); th1pp(t)];
```

TORQUE 1:

Se obtienen las derivadas de la velocidad en la primera coordenada:

```
 dQ1=[diff(diff(Lagrangiano,th1p), th1),...
     diff(diff(Lagrangiano,th1p), th1p)];
```

Definimos el torque 1.

```
t1= dQ1*Qd- diff(Lagrangiano, th1);
```

GENERACIÓN DEL MODELO DINÁMICO DE FORMA MATRICIAL

Primero se hace una matriz de inercia usando los coeficientes de las aceleraciones.

```
M=[diff(t1, th1pp)];
rank (M);

M=M(t);
```

FUERZAS CENTRIPETAS Y DE CORIOLIS

Se realizan derivadas parciales en el tiempo respecto a todas las variables

```
M11=[diff(M(1,1),th1)]*Qp;%Se deriva parcialmente en el tiempo respecto a todas las var

Mp=[M11]
```

Mp(t) = $()$

Se define la energía cinética en su forma matricial:

```
k=1/2*transpose(Qp)*M*Qp;
dk=[diff(k, th1)]
```

dk(t) = $()$

Se calculan las fuerzas centripetas y de Coriolis:

```
C= Mp*Qp-dk;
```

PAR GRAVITACIONAL

Se sustituyen las velocidade y aceleraciones por 0 y se calcula el torque de cada uno de los motores

```
r=cero;
a1=subs(t1, th1p, r)
```

```
a1(t) =
```

$$\mathrm{th1pp}(t)\left(\frac{\mathrm{Izz}_1\,|\mathrm{cero}|}{\sqrt{\mathrm{cero}\,\overline{\mathrm{cero}}}}-\frac{\mathrm{Izz}_1\,|\mathrm{cero}|\,\sigma_3}{4\,\sigma_4}+\frac{\mathrm{Izz}_1\,\sigma_3}{4\,\mathrm{cero}\,\overline{\mathrm{cero}}}+\frac{\sigma_2\,|\mathrm{cero}|\,|m_1|^2\,\sigma_1\,(2\,l_1+\mathrm{lc}_1)}{8\,l_1\,\mathrm{lc}_1\,\sqrt{\mathrm{cero}\,\overline{\mathrm{cero}}}}+\frac{\sigma_2\,|m_1|^2\,\sigma_1\,\sigma_3\,(2\,l_1+\mathrm{l}}{32\,\mathrm{cero}\,l_1\,\mathrm{lc}_1\,\overline{\mathrm{cero}}}\right.$$

where

$$\sigma_1 = 2\,\mathrm{lc}_1\,|l_1|^2 + l_1\,|\mathrm{lc}_1|^2$$

$$\sigma_2 = \cos\left(\overline{\mathrm{th}_1(t)} - \mathrm{th}_1(t)\right)$$

$$\sigma_3 = (\mathrm{cero} + \overline{\mathrm{cero}})^2$$

$$\sigma_4 = (\mathrm{cero}\,\overline{\mathrm{cero}})^{3/2}$$

```
a2=subs(a1, th1pp,r)
```

```
a2(t) =
```

$$\mathrm{cero}\left(\frac{\mathrm{Izz}_1\,|\mathrm{cero}|}{\sqrt{\mathrm{cero}\,\overline{\mathrm{cero}}}}-\frac{\mathrm{Izz}_1\,|\mathrm{cero}|\,\sigma_3}{4\,\sigma_4}+\frac{\mathrm{Izz}_1\,\sigma_3}{4\,\mathrm{cero}\,\overline{\mathrm{cero}}}+\frac{\sigma_2\,|\mathrm{cero}|\,|m_1|^2\,\sigma_1\,(2\,l_1+\mathrm{lc}_1)}{8\,l_1\,\mathrm{lc}_1\,\sqrt{\mathrm{cero}\,\overline{\mathrm{cero}}}}+\frac{\sigma_2\,|m_1|^2\,\sigma_1\,\sigma_3\,(2\,l_1+\mathrm{lc}_1)}{32\,\mathrm{cero}\,l_1\,\mathrm{lc}_1\,\overline{\mathrm{cero}}}-\right.$$

where

$$\sigma_1 = 2\,\mathrm{lc}_1\,|l_1|^2 + l_1\,|\mathrm{lc}_1|^2$$

$$\sigma_2 = \cos\left(\overline{\mathrm{th}_1(t)} - \mathrm{th}_1(t)\right)$$

$$\sigma_3 = (\mathrm{cero} + \overline{\mathrm{cero}})^2$$

$$\sigma_4 = (\mathrm{cero}\,\overline{\mathrm{cero}})^{3/2}$$

```
%Torque gravitacional en el motor 1
G1=a2;
```

```
G1(t) =
```

$$\text{cero}\left(\frac{\text{Izz}_1\,|\text{cero}|}{\sqrt{\text{cero}\,\overline{\text{cero}}}}-\frac{\text{Izz}_1\,|\text{cero}|\,\sigma_3}{4\,\sigma_4}+\frac{\text{Izz}_1\,\sigma_3}{4\,\text{cero}\,\overline{\text{cero}}}+\frac{\sigma_2\,|\text{cero}|\,|m_1|^2\,\sigma_1\,(2\,l_1+\text{lc}_1)}{8\,l_1\,\text{lc}_1\,\sqrt{\text{cero}\,\overline{\text{cero}}}}+\frac{\sigma_2\,|m_1|^2\,\sigma_1\,\sigma_3\,(2\,l_1+\text{lc}_1)}{32\,\text{cero}\,l_1\,\text{lc}_1\,\overline{\text{cero}}}\right.\text{-}$$

where

$$\sigma_1=2\,\text{lc}_1\,|l_1|^2+l_1\,|\text{lc}_1|^2$$

$$\sigma_2=\cos\left(\overline{\text{th}_1(t)}-\text{th}_1(t)\right)$$

$$\sigma_3=(\text{cero}+\overline{\text{cero}})^2$$

$$\sigma_4=(\text{cero}\,\overline{\text{cero}})^{3/2}$$

```
Warning: Unable to find explicit solution. For options, see help.

solution =

Empty sym: 0-by-1

No solution exists.
```

Se crea el vector del par gravitacional.

```
G = [G1]
```

```
G(t) =
```

$$\text{cero}\left(\frac{\text{Izz}_1\,|\text{cero}|}{\sqrt{\text{cero}\,\overline{\text{cero}}}}-\frac{\text{Izz}_1\,|\text{cero}|\,\sigma_3}{4\,\sigma_4}+\frac{\text{Izz}_1\,\sigma_3}{4\,\text{cero}\,\overline{\text{cero}}}+\frac{\sigma_2\,|\text{cero}|\,|m_1|^2\,\sigma_1\,(2\,l_1+\text{lc}_1)}{8\,l_1\,\text{lc}_1\,\sqrt{\text{cero}\,\overline{\text{cero}}}}+\frac{\sigma_2\,|m_1|^2\,\sigma_1\,\sigma_3\,(2\,l_1+\text{lc}_1)}{32\,\text{cero}\,l_1\,\text{lc}_1\,\overline{\text{cero}}}\right.\text{-}$$

where

$$\sigma_1=2\,\text{lc}_1\,|l_1|^2+l_1\,|\text{lc}_1|^2$$

$$\sigma_2=\cos\left(\overline{\text{th}_1(t)}-\text{th}_1(t)\right)$$

$$\sigma_3=(\text{cero}+\overline{\text{cero}})^2$$

$$\sigma_4=(\text{cero}\,\overline{\text{cero}})^{3/2}$$

Se usa el comando *toc* para mostrar el tiempo de ejecución del programa.

```
toc
```

```
Elapsed time is 4.297077 seconds.
```