Instituto Tecnológico y de Estudios Superiores de Monterrey

TE3001B.101

Fundamentación de robótica (Gpo 101)

Semestre: Febrero - Junio 2022

Actividad 5 (Torque y Matriz de Inercia)

Alumnos:

Frida Lizett Zavala Pérez	A01275226
José Jezarel Sánchez Mijares	A01735226
Antonio Silva Martínez	A01173663

Marzo del 2023

Actividad 5 (Torque y Matriz de Inercia)

La mecánica de Lagrange, emplea energías cinéticas y potenciales que son cantidades escalares. La aplicación de la mecánica de Lagrange da lugar a ***n*** ecuaciones diferenciales correspondientes a ***n*** coordenadas generalizadas.

Suponiendo que la energía potencial está representada por T y la energía cinética por V, entonces el lagrangiano es igual a $L = T - V$.

La ecuación de Lagrange correspondiente a la coordenada generalizada q_i es:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0 \quad i = 1,2,3...$$

El torque puede entenderse como el momento de fuerza o momento dinámico Es la fuerza que requiere ser aplicada para hacer que algo gire.

Una matriz de inercia es una matriz simétrica utilizada en física para representar la distribución de masa de un objeto en rotación alrededor de un eje. Esta matriz describe cómo el objeto resiste los cambios en su movimiento rotacional y se utiliza en la mecánica para calcular las fuerzas y momentos necesarios para mover el objeto en rotación.

A continuación se muestra el código en Matlab **modelo del toque de cada articulación** y la **matriz de inercia** para cada una de las siguientes configuraciones de robots manipuladores

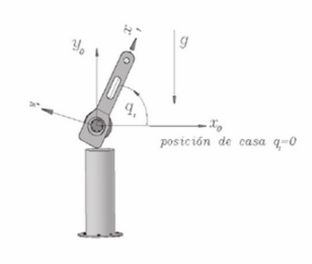
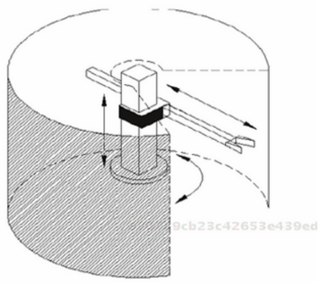
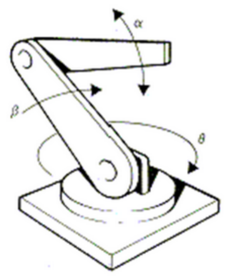


Figura 4.10 Péndulo robot.

Robot Péndulo (1gdl)



Robot Cilíndrico (3gdl)



Robot Angular (3gdl)

Robot Péndulo

Primero se realiza la iniciallización de variables, la inicialización de los vectores y la asignación de cada variable en los vectores que se usarán.

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

tic
%Declaración de variables simbólicas
syms th1(t) t  %Angulos de cada articulación
syms th1p(t) %Velocidades de cada articulación
syms th1pp(t) %Aceleraciones de cada articulación

syms m1 Ixx1 Iyy1 Izz1  %Masas y matrices de Inercia
syms t1 %Tiempos
syms l1 lc1  %l=longitud de eslabones y lc=distancia al centro de
masa de cada eslabón
syms pi g

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

 %Creamos el vector de velocidades articulares
  Qp= [th1p];
 %disp('Velocidades generalizadas');
 %pretty (Qp);
 %Creamos el vector de aceleraciones articulares
  Qpp= [th1pp];
 %disp('Aceleraciones generalizadas');
 %pretty (Qpp);

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

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

Se usan las matrices de rotación para posteriormenteguardar los valores dentro de las matrices de transformación que se inicializan en esta parte.

```
%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);
```

El ciclo for guardará los valores obtenidos de la matriz en una matriz local y una global.

```
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 calculan nos Jacobianos a travéz de la derivación respecto a los ánguolos theta que se usan en la configuración, creando una matriz de jacobianos para calcular el jacobiano líneal.

```
%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);
%Derivadas parciales de y respecto a th1 y th2
Jv21= functionalDerivative(PO(2,1,GDL), th1);
%Derivadas parciales de z respecto a th1 y th2
Jv31= functionalDerivative(PO(3,1,GDL), th1);

%Creamos la matríz del Jacobiano lineal
jv_d=simplify([Jv11;
           Jv21;
           Jv31]);
%pretty(jv_d);

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

A partir de los jacobianos obtenidos se obtienen las submatrices de los jacobianos lineales y angulares y con ello sacar las velocidades lineal y angular.

```
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 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
     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

%Obtenemos SubMatrices de Jacobianos
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);

%Matriz de Jacobiano Completa
%disp('Matriz de Jacobiano');
Jac= [Jv_a;
      Jw_a];
Jacobiano= simplify(Jac);
% pretty(Jacobiano);

%Obtenemos vectores de Velocidades Lineales y Angulares
% 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);
```

Enseguida se obtiene la Energía Cinética considerando la distancia de los eslabones a través de los vectores y la matriz de inercia de cada eslabón, y debido a que solo se cuenta con una articulación únicamente se cuenta con una matriz de inercia para el eslabón, posteriormente se extrae la velocidad lineal de cada eje y se obtienen las velocidades angulares de euler.

```
%Energía Cinética
```

```
%Distancia del origen del eslabón a su centro de masa
%Vectores de posición respecto al centro de masa
 P01=subs(P(:,:,1)/2, l1, lc1);

%Creamos matrices de inercia para cada eslabón

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

%Función de energía cinética

%Extraemos las velocidades lineales en cada eje
V=V(t);
Vx= V(1,1);
Vy= V(2,1);
Vz= V(3,1);

%Extraemos las velocidades angular en cada ángulo de Euler
W=W(t);
W_pitch= W(1,1);
W_roll= W(2,1);
W_yaw= W(3,1);
```

Se calculan las velocidades de cada eslabón y se construye la matriz jacobiana de cada articulación, para obtener los vectores de las velocidades angular y lineal, finalmente se hace una suma de ellas para obtener una velocidad neta.

```
%Calculamos las velocidades para cada eslabón

%Matriz de Jacobiano Completa
%disp('Matriz de Jacobiano');
Jac1= [Jv_a;
     Jw_a];
Jacobiano1= simplify(Jac1);
% pretty(Jacobiano);

%Obtenemos vectores de Velocidades Lineales y Angulares
 %disp('Velocidad lineal obtenida mediante el Jacobiano lineal del
Eslabón 1');
```

```
Qp=Qp(t);
V1=simplify (Jv_a*Qp(1));
 %pretty(V1);
 % disp('Velocidad angular obtenida mediante el Jacobiano angular
del Eslabón 1');
W1=simplify (Jw_a*Qp(1));
 % pretty(W1);

%Eslabón 1
V1_Total= V1+cross(W1,P01);
K1= (1/2*m1*(V1_Total))'*(1/2*m1*(V1_Total)) + (1/2*W1)'*(I1*W1);
%disp('Energía Cinética en el Eslabón 1');
K1= simplify (K1);
%pretty (K1);


K_Total= simplify (K1);
%pretty(K_Total)
```

Por último para completar el modelo de energía se calcula la energía potencial por cada uno de los eslabones, para obtener el lagrangiano se hace la resta de la energía cinética total, menos la energía potencial total. Mientras que el modelo de energía es la suma de las energías.

```
%Calculamos la energía potencial para cada uno de los eslabones

%Obtenemos las alturas respecto a la gravedad
 h1= P01(2); %Tomo la altura paralela al eje y

 U1=m1*g*h1;

 %Calculamos la energía potencial total
 U_Total= U1

  %Obtenemos el Lagrangiano
 Lagrangiano= simplify (K_Total-U_Total);
 pretty (Lagrangiano);

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

Añadiendo a la actividad anterior, para obtener el torque y la matriz de inercia, se usa el código siguiente.

```
  %%%%%%%%%%%%%%%%%%%%%%Ecuaciones de
Movimiento%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 %Lagrangiano derivado con respecto a la primera coordenada
generalizada de
 %velocidad

 %Definimos un vector columna de derivadas con respecto al tiempo
 %En este vector agrego las velocidades y aceleraciones
 Qd=[th1p(t); th1pp(t);];

 %Obtenemos las derivadas de la velocidad en la primera coordenada
 %generalizada
 dQ1=[diff(diff(Lagrangiano,th1p), th1),... %Derivamos con
respecto a la primera velocidad generalizada th1p para las 3
posiciones articulaciones
      diff(diff(Lagrangiano,th1p))];%Derivamos con respecto a la
primera velocidad generalizada th1p para las 3 velocidades
articulaciones

 %Definimos el torque 1
 t1= dQ1*Qd- diff(Lagrangiano, th1);
```

Se genera el modelo dinámico matricial a partir de los resultados anteriores.

```
 %Generación del Modelo Dinámico en forma matricial

%Matriz de Inercia

%Extraemos coeficientes de aceleraciones

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

simplify(M);
pretty(M)
toc
```

Resultados

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

                                                                                        2                                   2                                              2
Izz1 |th1p(t)| #5       Izz1 #4 #2      Izz1 |th1p(t)| #4 #2   |th1p(t)| cos(#3) |m1|  #5 #1 (2 l1 + lc1)   cos(#3) |m1|  #4 #1 #2 (2 l1 + lc1)   |th1p(t)| cos(#3) |m1|  #4 #1 #2 (2 l1 + lc1)
---------------- + ---------------- - -------------------- + ------------------------------------------ + ------------------------------------ - ---------------------------------------------
    2 sqrt(#8)         _______                  3/2                            #6                                        _______                                              3/2
                   4 th1p(t) th1p(t)        4 #8                                                          32 l1 lc1 th1p(t) th1p(t)                          32 l1 lc1 #8

                                                / _________           \
                         2                      |  d            d      |
     |th1p(t)| sin(#3) |m1|  #1 #2 (2 l1 + lc1) | -- th1(t) - -- th1(t) |
                                                \ dt           dt      /
   - ---------------------------------------------------------------------
                                      #6

where

                   2              2
   #1 == l1 |lc1|  + 2 lc1 |l1|

         _______
   #2 == th1p(t) + th1p(t)

         ______
   #3 == th1(t) - th1(t)

         _______  d
   #4 == th1p(t) -- th1p(t) + #7 th1p(t)
                 dt

          d
   #5 == -- th1p(t) + #7
         dt

   #6 == 16 l1 lc1 sqrt(#8)

         __________
          d
   #7 == -- th1p(t)
         dt

         _______
   #8 == th1p(t) th1p(t)
```

Robot Cilíndrico

```
%%%%%%%%%%%%%%%%%%%%%%Ecuaciones de
Movimiento%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Lagrangiano derivado con respecto a la primera coordenada
generalizada de
%velocidad

%Definimos un vector columna de derivadas con respecto al tiempo
%En este vector agrego las velocidades y aceleraciones
Qd=[th1p(t); l2p(t); l3p(t); th1pp(t); l2pp(t); l3pp(t)];

%Obtenemos las derivadas de la velocidad en la primera coordenada
%generalizada
dQ1=[diff(diff(Lagrangiano,th1p),
th1),diff(diff(Lagrangiano,th1p), l2),diff(diff(Lagrangiano,th1p),
l3),... %Derivamos con respecto a la primera velocidad
generalizada th1p para las 3 posiciones articulaciones
     diff(diff(Lagrangiano,th1p),
th1p),diff(diff(Lagrangiano,th1p),
l2p),diff(diff(Lagrangiano,th1p), l3p)];%Derivamos con respecto a
la primera velocidad generalizada th1p para las 3 velocidades
articulaciones
%Definimos el torque 1
```

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


%Obtenemos las de derivadas de la velocidad en la segunda
coordenada
 %generalizada
 dQ2=[diff(diff(Lagrangiano,l2p), th1),diff(diff(Lagrangiano,l2p),
l2),diff(diff(Lagrangiano,l2p), l3),... %Derivamos con respecto a
la primera velocidad generalizada th1p para las 3 posiciones
articulaciones
      diff(diff(Lagrangiano,l2p), th1p),diff(diff(Lagrangiano,l2p),
l2p),diff(diff(Lagrangiano,l2p), l3p)];%Derivamos con respecto a
la primera velocidad generalizada th1p para las 3 velocidades
articulaciones

 %Definimos el torque 2
 t2= dQ2*Qd- diff(Lagrangiano, l2);


%Obtenemos las de derivadas de la velocidad en la tercera
coordenada
%generalizada
dQ3=[diff(diff(Lagrangiano,l3p), th1),diff(diff(Lagrangiano,l3p),
l2),diff(diff(Lagrangiano,l3p), l3),... %Derivamos con respecto a
la primera velocidad generalizada th1p para las 3 posiciones
articulaciones
      diff(diff(Lagrangiano,l3p), th1p),diff(diff(Lagrangiano,l3p),
l2p),diff(diff(Lagrangiano,l3p), l3p)];%Derivamos con respecto a
la primera velocidad generalizada th1p para las 3 velocidades
articulaciones

%Definimos el torque 3
t3= dQ3*Qd- diff(Lagrangiano, l3);

 %Generación del Modelo Dinámico en forma matricial

%Matriz de Inercia

%Extraemos coeficientes de aceleraciones

M=[diff(t1, th1pp), diff(t1, l2pp), diff(t1, l3pp);...
   diff(t2, th1pp), diff(t2, l2pp), diff(t2, l3pp);...
```

```
   diff(t3, th1pp), diff(t3, l2pp), diff(t3, l3pp)];
rank (M);

M=M(t);

simplify(M);
pretty(M)
toc
```

Robot Angular

```
%%%%%%%%%%%%%%%%%%%%%%Ecuaciones de
Movimiento%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 %Lagrangiano derivado con respecto a la primera coordenada
generalizada de
 %velocidad

 %Definimos un vector columna de derivadas con respecto al tiempo
 %En este vector agrego las velocidades y aceleraciones
 Qd=[th1p(t); th2p(t); th3p(t); th1pp(t); th2pp(t); th3pp(t)];

 %Obtenemos las derivadas de la velocidad en la primera coordenada
 %generalizada
 dQ1=[diff(diff(Lagrangiano,th1p),
th1),diff(diff(Lagrangiano,th1p),
th2),diff(diff(Lagrangiano,th1p), th3),... %Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 posiciones
articulaciones
      diff(diff(Lagrangiano,th1p),
th1p),diff(diff(Lagrangiano,th1p),
th2p),diff(diff(Lagrangiano,th1p), th3p)];%Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 velocidades
articulaciones

 %Definimos el torque 1
 t1= dQ1*Qd- diff(Lagrangiano, th1);


 %Obtenemos las de derivadas de la velocidad en la segunda
coordenada
```

```
 %generalizada
 dQ2=[diff(diff(Lagrangiano,th2p),
th1),diff(diff(Lagrangiano,th2p),
th2),diff(diff(Lagrangiano,th2p), th3),... %Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 posiciones
articulaciones
      diff(diff(Lagrangiano,th2p),
th1p),diff(diff(Lagrangiano,th2p),
th2p),diff(diff(Lagrangiano,th2p), th3p)];%Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 velocidades
articulaciones
```

```
 %Definimos el torque 2
 t2= dQ2*Qd- diff(Lagrangiano, th2);


  %Obtenemos las de derivadas de la velocidad en la tercera
coordenada
 %generalizada
 dQ3=[diff(diff(Lagrangiano,th3p),
th1),diff(diff(Lagrangiano,th3p),
th2),diff(diff(Lagrangiano,th3p), th3),... %Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 posiciones
articulaciones
      diff(diff(Lagrangiano,th3p),
th1p),diff(diff(Lagrangiano,th3p),
th2p),diff(diff(Lagrangiano,th3p), th3p)];%Derivamos con respecto
a la primera velocidad generalizada th1p para las 3 velocidades
articulaciones

 %Definimos el torque 3
 t3= dQ3*Qd- diff(Lagrangiano, th3);
```

```
%Generación del Modelo Dinámico en forma matricial

%Matriz de Inercia
```

```
%Extraemos coeficientes de aceleraciones

M=[diff(t1, th1pp), diff(t1, th2pp), diff(t1, th3pp);...
   diff(t2, th1pp), diff(t2, th2pp), diff(t2, th3pp);...
   diff(t3, th1pp), diff(t3, th2pp), diff(t3, th3pp)];
rank (M);

M=M(t);



 %Fuerzas Centrípetas y de Coriolis

 %Definimos Mp

 M11=[diff(M(1,1),th1), diff(M(1,1),th2), diff(M(1,1),th3)]*Qp;%Se
deriva parcialmente en el tiempo respecto a todas las variables
 M12=[diff(M(1,2),th1), diff(M(1,2),th2), diff(M(1,2),th3)]*Qp;
 M13=[diff(M(1,3),th1), diff(M(1,3),th2), diff(M(1,3),th3)]*Qp;

 M21=[diff(M(2,1),th1), diff(M(2,1),th2), diff(M(2,1),th3)]*Qp;
 M22=[diff(M(2,2),th1), diff(M(2,2),th2), diff(M(2,2),th3)]*Qp;
 M23=[diff(M(2,3),th1), diff(M(2,3),th2), diff(M(2,3),th3)]*Qp;

 M31=[diff(M(3,1),th1), diff(M(3,1),th2), diff(M(3,1),th3)]*Qp;
 M32=[diff(M(3,2),th1), diff(M(3,2),th2), diff(M(3,2),th3)]*Qp;
 M33=[diff(M(3,3),th1), diff(M(3,3),th2), diff(M(3,3),th3)]*Qp;

  Mp=[M11, M12, M13;...
      M21, M22, M23;...
      M31, M32, M33];


%Definimos la energía cinética en su forma matricial
k=1/2*transpose(Qp)*M*Qp;

%Definimos dk

dk=[diff(k, th1); diff(k, th2); diff(k, th3)];

%Fuerzas centrípetas y de Coriolis
```

```
 C= Mp*Qp-dk;


 %Par Gravitacional
 %se sustituyen las velocidades y aceleraciones por 0
 r=cero;
 a1=subs(t1, th1p, r);
 a2=subs(a1, th2p, r);
 a3=subs(a2, th3p, r);
 a4=subs(a3, th1pp,r);
 a5=subs(a4, th2pp,r);
 a6=subs(a5, th3pp,r);

 %Torque gravitacional en el motor 1
 G1=a6;


 b1=subs(t2, th1p, r);
 b2=subs(b1, th2p, r);
 b3=subs(b2, th3p, r);
 b4=subs(b3, th1pp,r);
 b5=subs(b4, th2pp,r);
 b6=subs(b5, th3pp,r);

 %Torque gravitacional en el motor 2
 G2=b6;

 c1=subs(t3, th1p, r);
 c2=subs(c1, th2p, r);
 c3=subs(c2, th3p, r);
 c4=subs(c3, th1pp,r);
 c5=subs(c4, th2pp,r);
 c6=subs(c5, th3pp,r);

 %Torque gravitacional en el motor 3
 G3=c6;

% Vector de par gravitacional

G=[G1;G2;G3]

 toc
```

Conclusiones

Debemos tener presente y considerar que en el movimiento de un robot se ven involucrados varios factores además de las transformaciones y las rotaciones, diversas fuerzas del espacio físico se ven involucradas y para poder representar, observar y análizar su comportamiento debemos conocer las ecuaciones de movimiento descritas, así como su precedencia y la mandera en que se están aplicando.