Guía número 4: Control de robots manipuladores.

Unidad de Aprendizaje 4:
Planificación y programación de trayectorias

Aprendizaje Esperado
4.1.-Planifica trayectorias de robots manipuladores, mediante herramientas de software. (Integrada Innovación).

Actividades

1. Simulación control dinámico.

I. Presentación

La presente guía presenta la planificación y posterior simulación de posicionamiento de un robot manipulador de dos grados de libertad.

Criterios de evaluación:

Las actividades consideran los siguientes criterios de evaluación.

- 4.1.1.- Aplicando métodos de cálculo de trayectorias articulares, a partir de una trayectoria espacial deseada.
- 4.1.2.- Detectando singularidades producidas por las trayectorias diseñadas, mediante herramientas de simulación.
- 4.1.3.- Programando trayectoria indirecta, en consola de programación del robot de manipulación.
- 4.1.4.- Priorizando la información, criterios, enfoques y las variables para abordar las situaciones.

II. Instrucciones

1. En el equipo de trabajo, desarrollar las actividades, usando herramientas matemáticas, software de simulación y editor de ecuaciones de MS Office o similar.

III. Ejemplo de aplicación

2. Simulación de un control de posición de un robot.

Paso 2.1: Control dinámico.

2.1.1. Utilizando el algoritmo Newton-Euler, determine la dinámica del robot manipulador, el script se adjunta a continuación:

```
clear all
clc
%% EJEMPLO NEWTON EULER ROBOT RR PLANAR
%% NE-1:Algoritmo DH del ROBOT
syms th1 th2 L1 L2 g th1p th2p th1pp th2pp m1 m2
%% NE-2: Condiciones iniciales
w00=[0;0;0]; %Vector tipo columna
wp00=[0;0;0];
v00=[0;0;0];
vp00=[0;g;0];
z0=[0;0;1];
p1=[L1;0;0]; %Error en el libro de robótica debe ser traspuesto
p2=[L2;0;0];
s1=[-L1/2;0;0];
s2=[-L2/2;0;0];
I1=0;
I2=0;
%% NE-3 Obtenga matrices de rotación y sus inversas
%% Matriz R de 0 a 1
% Formato matrizR(thi,di,ai,alfi)
R01=simplify(matrizR(th1,0,L1,0));
%% Matriz R de 1 a 2
R12=simplify(matrizR(th2,0,L2,0));
R10=simplify(inv(R01));
R21=simplify(inv(R12));
%% Hasta este punto se denomina "Primer parte" (variables iniciales y matrices
inversas)
%% Para los pasos de NE-4 a NE-7 "Segunda parte" (definición de ecuaciones
hacia
adelante)
% las ecuaciones se plantean desde la base hasta el efector final, es
% decir, de 1 hasta n, y el proceso inverso, desde el efector final hasta la
% base, es decir, de n hasta 1.
%% NE-4 obtener la velocidad angular de Si
% eslabón 1 rotacional
w11=simplify(R10*(w00 +z0*th1p));
w22=simplify(R21*(w11 +z0*th2p));
```

```
%% NE-5 Obtenga la aceleración angular del sistema Si
wp11=R10*(wp00 +z0*th1pp) +cross(w00,z0*th1p);
wp22=R21*(wp11 +z0*th2pp) +cross(w11,z0*th2p);
wp11=simplify(wp11);
wp22=simplify(wp22);
%% NE-6 Obtenga la aceleración lineal del sistema i
vp11=cross(wp11,p1) +cross(w11,cross(w11,p1))+R10*vp00;
vp11=simplify(vp11);
vp22=cross(wp22,p2)+cross(w22,cross(w22,p2))+R21*vp11;
vp22=simplify(vp22);
%% NE-7 obtenga la aceleración lineal del sistema i
a11=cross(wp11,s1)+cross(w11,cross(w11,s1))+vp11;
a11=simplify(a11);
a22=cross(wp22,s2)+cross(w22,cross(w22,s2))+vp22;
a22=simplify(a22);
%% NE-8 Obtenga la fuerza ejercida sobre el eslabón i
% utilice i=n.....1
% En la primera iteración, no existe la relación de R23, es cero
f22=m2*a22;
f22=simplify(f22);
f11=R12*f22+m1*a11;
f11=simplify(f11);
%% NE-9 Obtenga el par ejercido sobre el eslabón i
n22=cross((p2+s2),m2*a22)+I2*w22+cross(w22,I2*w22);
n22=simplify(n22)
n11=R12*(n22+cross(R21*p1,f22))+cross((p1+s1),m1*a11)+I1*w11+cross(w11,I1*w11)
;
n11=simplify(n11)
%% NE-10 Obtenga el par o fuerza aplicado a la articulación i
tau2=(transpose(n22))*R21*z0;
tau2=simplify(tau2)
tau1=(transpose(n11))*R10*z0;
tau1=simplify(tau1)
%% Uso de comando coeffs para obtener los coeficientes de las matrices
% https://la.mathworks.com/help/symbolic/coeffs.html
%% Matriz de inercia
M11=coeffs(tau1,th1pp);
M11=M11(2);
M12=coeffs(tau1,th2pp);
M12=M12(2);
M21=coeffs(tau2,th1pp);
M21=M21(2);
M22=coeffs(tau2,th2pp);
M22=M22(2);
%% Matriz M
M=[M11 M12;
M21 M22];
%% Matriz de gravedad
G1=coeffs(tau1,g);
G1=G1(2)
G2=coeffs(tau2,g);
G2=G2(2)
%% Matriz G
G=[G1;
G2];
%% Matriz de coriolis 1
C1=tau1-M11*th1pp-M12*th2pp-G1*g;
```

```
C1=simplify(C1)
%% Matriz de coriolis 2
C2=tau2-M21*th1pp-M22*th2pp-G2*g;
C2=simplify(C2)
%% Matriz C
C=[C1;
C2];
```

2.1.2. Para obtener la dinámica inversa del manipulador, se utiliza el siguiente script de función embebida:

```
function qpp=dindirRR_planar(tau,qp,q)
%% Vector torque 1 y 2
tau1=tau(1)
tau2=tau(2)
%% Vector ángulos de posicionamiento
th1=q(1)
th2=q(2)
%% Vector velocidad angular
th1p=qp(1)
th2p=qp(1)
%% Parámetros físicos
L1=1;
L2=1;
m1=0.1;
m2=0.1;
g=9.8;
b1=0.1;
b2=0.1;
%% Matriz M obtenida de Newton Euler
M=[(L1^2*m1)/4 + L1^2*m2 + (L2^2*m2)/4 + L1*L2*m2*cos(th2),(m2*L2^2)/4 +
(L1*m2*cos(th2)*L2)/2;
    (L2*m2*(L2/2 + L1*cos(th2)))/2,(L2^2*m2)/4];
%% Matriz G obtenida de Newton Euler
G=[(L2*m2*cos(th1 + th2))/2 + (L1*m1*cos(th1))/2 + L1*m2*cos(th1);
                (L2*m2*(cos(th1)*cos(th2) - sin(th1)*sin(th2)))/2];
%% Matriz de coriolis
C=[-(L1*L2*m2*th2p*sin(th2)*(2*th1p + th2p))/2;
                  (L1*L2*m2*th1p^2*sin(th2))/2];
%% Matriz de fuerza
F=[b1*th1p;
   b2*th2p];
%% Vector Tau
Tau=[tau1;
     tau2];
%% dinámica inversa
qpp=inv(M)*(Tau-G*g-C-F);
```

El lazo de control asociado es el siguiente:

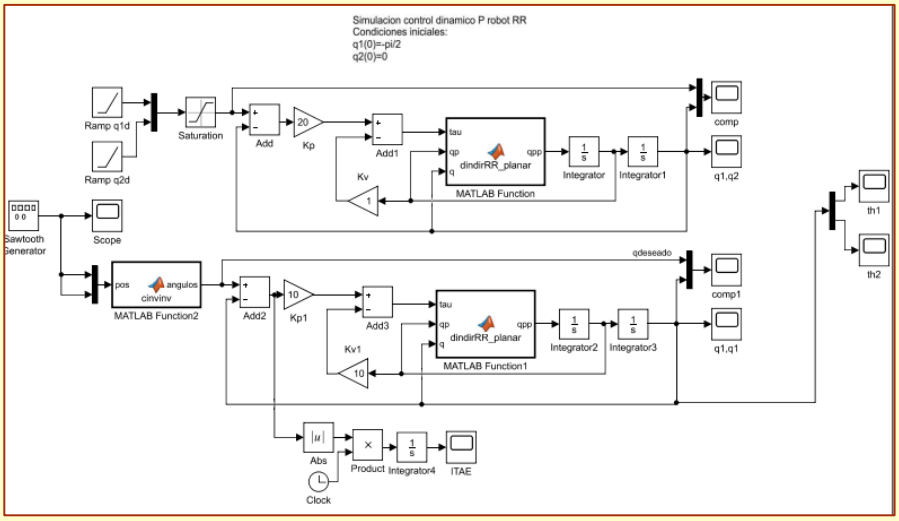

Figura 1: Diagrama de bloques control robot RR

2.1.3. Posteriormente validar el controlador con el indicador de desempeño ITAE.

IV. Ejercicio de aplicación.

Determine el modelo dinámico de un robot RRR, posteriormente realice un control de posicionamiento tipo P, valide el controlador con el indicador de desempeño ITAE.