Trabajo Práctico Nº6

Control de trayectorias

Facultad de Ingeniería, UNCuyo

Robótica II

Fabiancic, Federico

Fragapane, Adriel

Gargiulo, Pascual

Gattás, Samir

Torres, Rodrigo

Introducción

Para generar las trayectorias tanto horizontal como vertical se emplea el software Matlab donde se desarrollan la cinemática directa e inversa del robot, las cuales serán posteriormente empleadas para poder desarrollar las trayectorias en conjunto con funciones de interpolación, ya sea, en el espacio articular “jtraj” como en el espacio cartesiano con la función “intep1”.

Para recibir los valores articulares asociado a cada uno de los motores del robot, se realiza una comunicación por puerto serie entre el arduino y los valores enviados por Matlab desde la pc. Para establecer dicha comunicación se emplea el siguiente programa desarrollado en el entorno arduino:

```
#include <Servo.h>
 Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
int posDeseada[5];
char bufferRecep[50];
int indCoor;
int indBuf;
void setup() {
Serial.begin(115200);
servo1.attach(3);
servo2.attach(4);
servo3.attach(5);
servo4.attach(6);
servo5.attach(7);
posDeseada[0]=(int)(90);
posDeseada[1]=(int)((float)90*(float)((float)70/(float)90));
posDeseada[2]=((int)((float)90*(float)((float)85/(float)90)))+60;
posDeseada[3]=0+85;
posDeseada[4]=90;
indCoor = 0;
indBuf = 0;
}
```

```
void loop() {
        servo1.write(posDeseada[0]);
        servo2.write(posDeseada[1]);
        servo3.write(posDeseada[2]);
        servo4.write(posDeseada[3]);
        servo5.write(posDeseada[4]);
}
void serialEvent() {
  while(Serial.available()) {
   char inChar = (char)Serial.read();
   switch(inChar){
    case ':':
     indBuf=0;
    break;
    case '\r':
    bufferRecep[indBuf] = 0;
   cargarValores();
    break;
    default:
      bufferRecep[indBuf++]= inChar;
    break;
   }
 }
}
void cargarValores(){
  int i=1;
  int j=0;
  char token[10];
  indCoor = 0;
  if (bufferRecep[0]=! 0){
  while(bufferRecep[i] != 0){
```

```
      if(bufferRecep[i] != ','){
        token[j++] = bufferRecep[i];
        if(bufferRecep[i+1] == 0){
          token[j] = '\0';
          posDeseada[indCoor++] = atoi(&token[0]);
          j = 0;
           }
      }
      else{
        token[j] = '\0';
        posDeseada[indCoor++] = atoi(&token[0]);
        j = 0;
      }
      i++;
    }
  posDeseada[1]=(int)((float)posDeseada[1]*(float)((float)70/(float)90));
  posDeseada[2]=((int)((float)posDeseada[2]*(float)((float)85/(float)90)))+60;
  posDeseada[3]=posDeseada[3]+85;
 }
}
```

El código de Matlab empleado fue el siguiente:

```
%practico de control dinámico

 clc
close all
clear all
instrreset

longHombro = 97;
longBrazo = 142;
longAntBrazo = 77.5;
longMuneca = 71.5;
cabeceo = 0;

% Cinematica inversa
%implementada en funcion CinInversa
```

```
%% CINEMATICA DIRECTA
L2 = Link([0 longHombro 0 pi/2 0]);
L3 = Link([0 0 longBrazo 0 0]);
L4 = Link([0 0 longAntBrazo 0 0]);
L5 = Link([0 0 0 pi/2 0]);
L6 = Link([0 longMuneca 0 0 0]);

bot = SerialLink([L2 L3 L4 L5 L6], 'name', 'paletizador');


%% FUNCION DE CINEMATICA INVERSA (CinInversa)
function q = CinInversa(pX,pY,pZ,ox,oy)

if nargin<5
    error('Faltan datos')
end

longHombro = 97;
longBrazo = 142;
longAntBrazo = 77.5;
longMuneca = 71.5;
cabeceo = 0;

q2 = atan2(pY,pX);
Z = pZ-longHombro;
r=(pX^2+(pY)^2)^(0.5);
Afr = sin(cabeceo)*longMuneca;
ladoB = r-Afr;
Afz = cos(cabeceo)*longMuneca;
ladoA = Z+Afz;
hipotenusa = sqrt((ladoA^2)+(ladoB^2));
alfa = atan2(ladoA,ladoB);
beta = acos((longBrazo^2 + hipotenusa^2 - longAntBrazo^2)
/(2*longBrazo*hipotenusa));
angBrazo = alfa + beta;
q3=angBrazo;
gamma = acos((longBrazo^2+longAntBrazo^2-
hipotenusa^2)/(2*longBrazo*longAntBrazo));
angAntBrazo = -(pi-gamma);
q4=angAntBrazo;
angMuneca = cabeceo-angBrazo-angAntBrazo;
q5=angMuneca;
q6=acos(-sin(q2)*ox+cos(q2)*oy);

q = [q2,q3,q4,q5,q6];
end


%% FUNCION PARA ENVIAR DATOS DESDE MATLAB A ARDUINO

    arduin=serial('COM6','BaudRate',115200);
    fopen(arduin);

    for i= 1 : 84

    art1=q(i,1);
    art2=q(i,2);
    art3=q(i,3);
    art4=q(i,4);
    art5=q(i,5);
```

```
msg=[':',num2str(round((art1*180/pi))),',',num2str(round((art2*180/pi))),',',n
um2str(round((((art3*180/pi))+90))),',',num2str(round(((art4*180/pi)-90)*-
1)),',',num2str(round((art5*180/pi)))];
    fprintf(arduin,'%s\r',msg);
    fprintf('%s\r',msg);
    pause(0.5);
end
```

Ejercicio_1

Generación de la trayectoria horizontal

```
q0 = interp1([1 0],[-100 100],t);

for i=1:101
 Q0(i,:)=CinInversa(q0(i),100,50,1,0);
end


q=Q0;

bot.plot(q);
```

Ejercicio_2

Generación de la trayectoria vertical

```
q2 = CinInversa(0,230,200,0,1);
q3 = CinInversa(0,230,100,0,1);
Q1 = jtraj(q2,q3,t)


q=Q1;

bot.plot(q);
```

Ejercicio_3

Realizando las trayectorias aplicando una carga al efector final, se pudo corroborar que esta carga dinámica afectó mayormente a la trayectoria vertical que a la horizontal. En este caso, desarrollando las mismas trayectorias que en los puntos anteriores pudimos observar lo siguiente:

- El robot comenzaba a vibrar solo saliendo de su estado de equilibro para ciertas posiciones desfavorables, donde se pudo comprobar que el par motor disponible no era suficiente para satisfacer las condiciones dinámicas a las cuales lo estábamos sometiendo.
- En algunos pruebas donde el brazo sí ese encontraba en una posición de equilibrio, al aplicarle un golpe (torque perturbador) el robot comenzaba a vibrar y en la mayoría de los casos probados no lograba retornar nuevamente a su estado de equilibrio, lo cual si lo consiguió cuando se realizó la misma prueba pero quitando la carga adicional aplicada al efector final ya que para este caso el torque requerido era menor.

- Las articulaciones que más falla presentaron (par disponible inferior al que requerían las condiciones dinámicas a las cuales se lo sometió) fueron principalmente la 3 y la 4.