Práctica 3

Antonio Jesús Heredia Castillo

27 de mayo de 2020

Ejercicio 1

Para implementar la función tenemos que hacer uso de las deducciones que se realizan en la memoria.

Aunque tenemos algunos coeficientes que le pongo directamente el valor a 0, esto se debe a que como sabemos que la aceleración y velocidad al inicio y al final son nulos. Estos coeficientes con valor a 0 son c_{12}, c_{11},c_{32} y c_{31}. Otros coeficientes tendrán el valor de las articulaciones, estos son $c_{10} = qI$, $c_{20} = qD$ y $c_{30} = qF$.

La matriz que tenemos que obtener su inverso tendrá el valor de:

$$M = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{3}{t_1} & \frac{4}{t_1} & -\frac{1}{t_2} & 0 & 0 & 0 & 0 \\ \frac{6}{t_1^2} & \frac{12}{t_1^2} & 0 & -\frac{2}{t_2^2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & \frac{1}{t_2} & \frac{2}{t_2} & \frac{3}{t_2} & -\frac{3}{t_3} & \frac{4}{t_3} \\ 0 & 0 & 0 & \frac{2}{t_2^2} & \frac{6}{t_2^2} & \frac{6}{t_3^2} & -\frac{12}{t_3^2} \end{bmatrix}^{-1}$$

Y la matriz columna:

$$V = \begin{bmatrix} qD - qI \\ 0 \\ 0 \\ qA - qD \\ qA - qF \\ 0 \\ 0 \end{bmatrix}$$

Como podemos ver he quitado algunos valores de la matriz columna original. Esto se debe a que debido a la aceleración y velocidad igual a 0, desaparecían.

Solo tendremos que multiplicar $M \times V$. Y los valores de los coeficientes que nos faltaban por obtener los tendremos ahí.

Ejercicio 2

Gracias a lo realizado en practicas anteriores podemos realizar este ejercicio rápidamente. Realizaremos los siguientes pasos:

1. Tenemos que hacer es uso de la cinemática inversa para conseguir el valor de ambas articulaciones en los puntos intermedios.
2. Obtendremos los coeficientes de ambas articulaciones en la trayectoria 4-3-4 haciendo uso de la función implementada en el ejercicio anterior.
3. Evaluaremos los tres polinomios (uno por cada segmento) de las dos articulaciones en el rango que nos indica el guión de practicas. Cada "tramo" tendrá un tiempo de un segundo y avanzaremos de 0,05.
4. Por ultimo concatenamos los tres segmentos que recorre cada articulación y podremos ver el recorrido de la trayectoria.
5. Para ver el recorrido de la trayectoria usaremos funciones implementadas en practicas anteriores.

Como resultado tendremos:

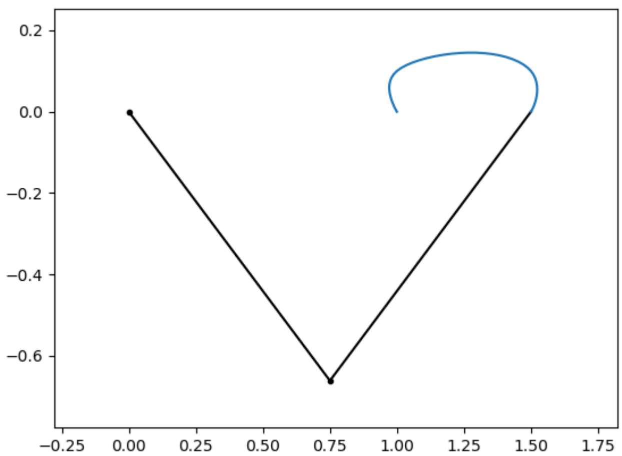

Figura 1: Trayectoria realizada

Archivos de código

ejercicio1-2.py

```
#!/usr/bin/env python3
# −*− coding: utf−8 −*−
"""
@author: Antonio Jesús Heredia Castillo
"""
import matplotlib.pyplot as plt
import numpy as np
from numpy.polynomial.polynomial import polyval
def pcd ( q1 , q2 , l1 , l2 ):
    y=l1*np.sin(q1)+l2*np.sin(q1+q2)
    x=l1*np.cos(q1)+l2*np.cos(q1+q2)
    return (x,y)
def dibujar_robot ( q1 , q2 , l1 , l2 ):
    x0 , y0 = 0 , 0; # Pos. de la articulaci ón 1
    x1 , y1 = pcd ( q1 , 0 , l1 , 0); # Pos. de la articulaci ón 2
    x2 , y2 = pcd ( q1 , q2 , l1 , l2 ); # Pos. del extremo del robot
    x = [ x0 , x1 , x2 ]; y = [ y0 , y1 , y2 ]; # Coordenadas de la trayec .
    plt . plot (x , y , 'k'); # Traza la trayectoria
    plt . plot ( x0 , y0 , 'k.'); # Dibuja la articulaci ón 1
    plt . plot ( x1 , y1 , 'k.'); # Dibuja la articulaci ón 2

def pci(x,y,l1,l2):
    q2 = np.arccos((x*x+y*y−l1*l1−l2*l2)/(2*l1*l2))
    #Descomentar segun se quiera usar arctan o arctan2
    #q1 = np.arctan((y*(l1+l2*np.cos(q2))−x*l2*np.sin(q2))/
                   #(x*(l1+l2*np.cos(q2))+y*l2*np.sin(q2)))
    q1 = np.arctan2((y*(l1+l2*np.cos(q2))−x*l2*np.sin(q2)),(
                    x*(l1+l2*np.cos(q2))+y*l2*np.sin(q2)))
    return q1,q2



def dibujar_trayectoria_pcd (q1s, q2s , l1 , l2 ):
    xs = []
    ys  = []
    for i,q1 in enumerate(q1s):
        x,y = pcd(q1,q2s[i],l1, l2)
        xs.append(x)
```

```
        ys.append(y)
    dibujar_robot(q1,q2s[i],l1,l2)
    plt.plot(xs,  ys)

def animacion_trayectoria_pcd ( q1s , q2s , l1 , l2, titulo ):
    n = min(len( q1s ) , len( q2s ))
    fig1 = plt.figure (titulo)
    for i in range (1 , n ):
        plt . clf ()
        dibujar_trayectoria_pcd ( q1s [0: i ] , q2s [0: i ] , l1 , l2 )
        plt.axis([-5,5,-5,5])
        plt . pause (0.001)

def trayectoria434(qI, qD, qA, qF, t1, t2, t3):

    matrix = np.array(
                [[1,1,0,0,0,0,0],
                [3/t1,4/t1,-1/t2,0,0,0,0],
                [6/(t1*t1),12/(t1*t1),0,-2/(t2*t2),0,0,0],
                [0,0,1,1,1,0,0],
                [0,0,0,0,0,-1,1],
                [0,0,1/t2,2/t2,3/t2,-3/t3,4/t3],
                [0,0,0,2/(t2*t2),6/(t2*t2),6/(t3*t3),-12/(t3*t3)]]
                )
    vector = np.array(
            [[qD-qI],
             [0],
             [0],
             [qA-qD],
             [qA-qF],
             [0],
             [0]]
        )
    matrix = np.linalg.inv(matrix)
    resultado = matrix.dot(vector)
    print(resultado)

    c14 = resultado[1]
    c13 = resultado[0]
    c12 = 0
    c11 = 0
```

```
    c10 = qI
    f1 = (c14,c13,c12,c11,c10)

    c23 = resultado[4]
    c22 = resultado[3]
    c21 = resultado[2]
    c20 = qD
    f2 = (c23,c22,c21,c20)

    c34 = resultado[6]
    c33 = resultado[5]
    c32 = 0
    c31 = 0
    c30 = qF
    f3 = (c34,c33,c32,c31,c30)

    return (f1 , f2 , f3)



qI=pci(1,0,1,1)
qD=pci(1,0.1,1,1)
qA=pci(1.5,0.1,1,1)
qF=pci(1.5,0,1,1)



f11,f12,f13 = trayectoria434(qI[0],qD[0],qA[0],qF[0],1,1,1)
evaluacion11 = np.polyval(f11, np.arange(0, 1, 0.05))
evaluacion12 = np.polyval(f12, np.arange(0, 1, 0.05))
evaluacion13 = np.polyval(f13, np.arange(0, 1, 0.05)-1)


f21,f22,f23 = trayectoria434(qI[1],qD[1],qA[1],qF[1],1,1,1)
evaluacion21 = np.polyval(f21, np.arange(0, 1, 0.05))
evaluacion22 = np.polyval(f22, np.arange(0, 1, 0.05))
evaluacion23 = np.polyval(f23, np.arange(0, 1, 0.05)-1)

q1s = np.concatenate((evaluacion11, evaluacion12, evaluacion13))
q2s = np.concatenate((evaluacion21, evaluacion22, evaluacion23))
```

```
animacion_trayectoria_pcd ( q1s , q2s , 1 , 1, "animación" )
```