

TP N°1

Implementación de una matriz cinemática en DSP

Materia: Robótica

División: R6055

Tema: Implementación de una matriz cinemática en DSP

Profesor: Ing. Hernán Gianetta

JTP: Ing. Damian Granzella

Alumnos: Emiliano Statello

Leandro Arcusin

**UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES**

INDICE

Introducción

3

1.1 CINEMATICA

3

1.2 APLICACIÓN DE LA CINEMATICA EN NUESTRO MODELO

5

Programación

8

2.1 PROGRAMACIÓN CON EL DSP56800/E

8

2.2 GRAFICOS EN MATLAB

10

Conclusión

13

1. INTRODUCCIÓN

1.1 CINEMATICA

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia sin considerar las fuerzas que intervienen. Así, la cinemática se interesa por la descripción analítica del movimiento espacial del robot como una función del tiempo, y en particular por las relaciones entre la posición y la orientación del extremo final del robot con los valores que toman sus coordenadas articulares.

Existen dos problemas fundamentales a resolver en la cinemática del robot; el primero de ellos se conoce como el problema cinemático directo, y consiste en determinar cuál es la posición y orientación del extremo final del robot, con respecto a un sistema de coordenadas que se toma como referencia, conocidos los valores de las articulaciones y los parámetros geométricos de los elementos del robot; el segundo, denominado problema cinemático inverso, resuelve la configuración que debe adoptar el robot para una posición y orientación del extremo conocidas.

En este trabajo práctico, vamos a utilizar la cinemática directa para encontrar la posición en cada instante del extremo manipulador del robot.

Para la resolución del problema cinemático directo mediante matrices de transformación homogénea se utiliza fundamentalmente el álgebra vectorial y matricial para representar y describir la localización de un objeto en el espacio tridimensional con respecto a un sistema de referencia fijo. Dado que un robot se puede considerar como una cadena cinemática formada por objetos rígidos o eslabones unidos entre sí mediante articulaciones, se puede establecer un sistema de referencia fijo situado en la base del robot y describir la localización de cada uno de los eslabones con respecto a dicho sistema de referencia. De esta forma, el problema cinemático directo se reduce a encontrar una matriz de transformación homogénea T que relacione la posición y orientación del extremo del robot respecto del sistema de referencia fijo situado en la base del mismo. Esta matriz T será función de las coordenadas articulares.

Para nuestro proyecto modelizaremos el robot del departamento de electrónica M5 donado por Skaymec de 5 grados de libertad.

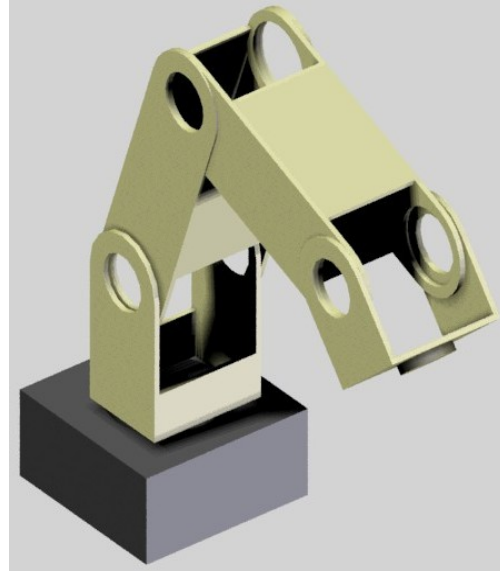
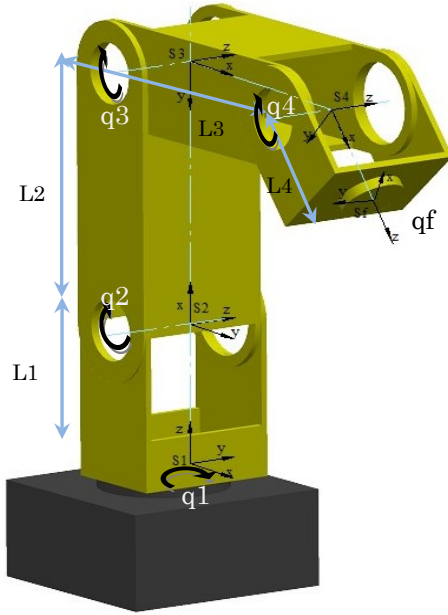
Con el modelo y los sistemas de referencia de cada una de las articulaciones planteados estamos en condiciones de obtener los parámetros de Denavit-Hartenberg que nos permitirán obtener las matrices transferencia parciales de cada articulación.

A continuación detallamos las 16 reglas que posee el algoritmo para que sea más clara la comprensión del resultado:

- **DH1.** Numerar los eslabones comenzando con 1 (primer eslabón móvil de la cadena) y acabando con n (último eslabón móvil). Se numerará como eslabón 0 a la base fija del robot.
- **DH2.** Numerar cada articulación comenzando por 1 (la correspondiente al primer grado de libertad y acabando en n).
- **DH3.** Localizar el eje de cada articulación. Si esta es rotativa, el eje será su propio eje de giro. Si es prismática, será el eje a lo largo del cual se produce el desplazamiento.
- **DH4.** Para i de 0 a $n-1$, situar el eje Z_i , sobre el eje de la articulación $i+1$.
- **DH5.** Situar el origen del sistema de la base (S_0) en cualquier punto del eje Z_0 . Los ejes X_0 e Y_0 se situarán de modo que formen un sistema dextrógiro con Z_0 .
- **DH6.** Para i de 1 a $n-1$, situar el sistema (S_i) (solidario al eslabón i) en la intersección del eje Z_i con la línea normal común a Z_{i-1} y Z_i . Si ambos ejes se cortasen se situaría (S_i) en el punto de corte. Si fuesen paralelos (S_i) se situaría en la articulación $i+1$.
- **DH7.** Situar X_i en la línea normal común a Z_{i-1} y Z_i .
- **DH8.** Situar Y_i de modo que forme un sistema dextrógiro con X_i y Z_i .
- **DH9.** Situar el sistema (S_n) en el extremo del robot de modo que Z_n coincida con la dirección de Z_{n-1} y X_n sea normal a Z_{n-1} y Z_n .
- **DH10.** Obtener θ_i como el ángulo que hay que girar en torno a Z_{i-1} para que X_{i-1} y X_i queden paralelos.
- **DH11.** Obtener D_i como la distancia, medida a lo largo de Z_{i-1} , que habría que desplazar (S_{i-1}) para que X_i y X_{i-1} quedasen alineados.
- **DH12.** Obtener A_i como la distancia medida a lo largo de X_i (que ahora coincidiría con X_{i-1}) que habría que desplazar el nuevo (S_{i-1}) para que su origen coincidiese con (S_i).
- **DH13.** Obtener a_i como el ángulo que habría que girar entorno a X_i (que ahora coincidiría con X_{i-1}), para que el nuevo (S_{i-1}) coincidiese totalmente con (S_i).
- **DH14.** Obtener las matrices de transformación $i-1A_i$.
- **DH15.** Obtener la matriz de transformación que relaciona el sistema de la base con el del extremo del robot $T = {}^0A_1, {}^1A_2 \dots {}^{n-1}A_n$.
- **DH16.** La matriz T define la orientación (submatriz de rotación) y posición (submatriz de traslación) del extremo referido a la base en función de las n coordenadas articulares.

1.2 APLICACIÓN DE LA CINEMATICA EN NUESTRO MODELO

Aplicando el algoritmo de Denavit-Hartenberg ubicamos los sistemas de referencia de todos los grados de libertad que posee el modelo del M5.



Articulación	θ	d	a	α
1	q_1	L_1	0	90°
2	q_2	0	L_2	0
3	q_3	0	L_3	0
4	$q_4 + 90$	0	L_4	90°

Armando las matrices y recordando que la forma general de las matrices A es:

$${}^{i-1}A_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \cdot \sin\theta_i & \sin\alpha_i \cdot \sin\theta_i & a_i \cdot \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cdot \cos\theta_i & -\sin\alpha_i \cdot \cos\theta_i & a_i \cdot \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Aplicando la misma para 0A_1 , 1A_2 , 2A_3 , 3A_4

$${}^0A_1 = \begin{bmatrix} \cos(q_1) & 0 & \sin(q_1) & 0 \\ \sin(q_1) & 0 & -\cos(q_1) & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1A_2 = \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 & L_2 \cdot \cos(q_2) \\ \sin(q_2) & \cos(q_2) & 0 & L_2 \cdot \sin(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2A_3 = \begin{bmatrix} \cos(q_3) & -\sin(q_3) & 0 & L_3 \cdot \cos(q_3) \\ \sin(q_3) & \cos(q_3) & 0 & L_3 \cdot \sin(q_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^3A_4 = \begin{bmatrix} \cos(q_4 + 90) & -\sin(q_4 + 90) & 0 & L_4 \cdot \cos(q_4 + 90) \\ \sin(q_4 + 90) & \cos(q_4 + 90) & 0 & L_4 \cdot \sin(q_4 + 90) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ahora para calcular la matriz homogénea debemos multiplicar todas la matrices tal que

$$T = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 \cdot {}^3A_4 \cdot {}^4A_5$$

$${}^0A_2 = {}^0A_1 \cdot {}^1A_2 = \begin{bmatrix} \cos(q_1) & 0 & \sin(q_1) & 0 \\ \sin(q_1) & 0 & -\cos(q_1) & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 & L_2 \cdot \cos(q_2) \\ \sin(q_2) & \cos(q_2) & 0 & L_2 \cdot \sin(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^0A_2 = \begin{bmatrix} \cos(q_1) \cdot \cos(q_2) & -\cos(q_1) \sin(q_2) & \sin(q_1) & L_2 \cdot \cos(q_2) \cdot \cos(q_1) \\ \sin(q_1) \cdot \cos(q_2) & -\sin(q_1) \sin(q_2) & -\cos(q_1) & L_2 \cdot \sin(q_2) \cdot \cos(q_1) \\ \sin(q_2) & \cos(q_2) & 0 & L_2 \cdot \sin(q_2) + L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2A_3 = \begin{bmatrix} \cos(q_3) & -\sin(q_3) & 0 & L_3 \cdot \cos(q_3) \\ \sin(q_3) & \cos(q_3) & 0 & L_3 \cdot \sin(q_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^0A_3 = \begin{bmatrix} \cos(q_1) \cdot \cos(q_2) \cos(q_3) - \cos(q_1) \sin(q_2) \sin(q_3) & -\cos(q_1) \sin(q_2) \sin(q_3) - \cos(q_1) \sin(q_2) \cos(q_3) & \sin(q_1) & \cos(q_1) \cdot \cos(q_2) L_3 \cdot \cos(q_3) - \cos(q_1) \sin(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \cos(q_2) \cdot \cos(q_1) \\ \sin(q_1) \cdot \cos(q_2) \cos(q_3) - \sin(q_1) \sin(q_2) \sin(q_3) & -\sin(q_1) \sin(q_2) \sin(q_3) - \sin(q_1) \sin(q_2) \cos(q_3) & -\cos(q_1) & \sin(q_1) \cdot \cos(q_2) L_3 \cdot \cos(q_3) - \sin(q_1) \sin(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \sin(q_2) \cdot \cos(q_1) \\ \sin(q_2) \cos(q_3) + \cos(q_2) \sin(q_3) & -\sin(q_3) \sin(q_2) + \cos(q_2) \cos(q_3) & 0 & \sin(q_2) L_3 \cdot \cos(q_3) + \cos(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \sin(q_2) + L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^3A_4 = \begin{bmatrix} \cos(q_4 + 90) & -\sin(q_4 + 90) & 0 & L_4 \cdot \cos(q_4 + 90) \\ \sin(q_4 + 90) & \cos(q_4 + 90) & 0 & L_4 \cdot \sin(q_4 + 90) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = {}^0A_3 \cdot {}^3A_4$$

$$T = \begin{bmatrix} nx & ox & ax & px \\ ny & oy & ay & py \\ nz & oz & 0 & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$nx = (\cos(q_1) \cdot \cos(q_2) \cos(q_3) - \cos(q_1) \sin(q_2) \sin(q_3)) \cdot \cos(q_4 + 90) - (\cos(q_1) \sin(q_2) \sin(q_3) - \cos(q_1) \sin(q_2) \cos(q_3)) \cdot \sin(q_4 + 90)$$

$$ny = (\sin(q_1) \cdot \cos(q_2) \cos(q_3) - \sin(q_1) \sin(q_2) \sin(q_3)) \cdot \cos(q_4 + 90) - (\sin(q_1) \sin(q_2) \sin(q_3) - \sin(q_1) \sin(q_2) \cos(q_3)) \cdot \sin(q_4 + 90)$$

$$nz = (\sin(q_2) \cos(q_3) + \cos(q_2) \sin(q_3)) \cdot \cos(q_4 + 90) - (\sin(q_3) \sin(q_2) + \cos(q_2) \cos(q_3)) \cdot \sin(q_4 + 90)$$

$$ox = -(\cos(q_1) \cdot \cos(q_2) \cos(q_3) - \cos(q_1) \sin(q_2) \sin(q_3)) \cdot \sin(q_4 + 90) - (\cos(q_1) \sin(q_2) \sin(q_3) - \cos(q_1) \sin(q_2) \cos(q_3)) \cdot \cos(q_4 + 90) + \sin(q_1)$$

$$oy = (\sin(q_1) \cdot \cos(q_2) \cos(q_3) - \sin(q_1) \sin(q_2) \sin(q_3)) \cdot \sin(q_4 + 90) - (\sin(q_1) \sin(q_2) \sin(q_3) - \sin(q_1) \sin(q_2) \cos(q_3)) \cdot \cos(q_4 + 90) - \cos(q_1)$$

$$oz = (\sin(q_2) \cos(q_3) + \cos(q_2) \sin(q_3)) \cdot \sin(q_4 + 90) - (\sin(q_3) \sin(q_2) + \cos(q_2) \cos(q_3)) \cdot \cos(q_4 + 90)$$

$$ax = 0 \quad ay = 0$$

$$px = -(\cos(q_1) \cdot \cos(q_2) \cos(q_3) - \cos(q_1) \sin(q_2) \sin(q_3)) \cdot L_4 \cdot \cos(q_4 + 90) - (\cos(q_1) \sin(q_2) \sin(q_3) - \cos(q_1) \sin(q_2) \cos(q_3)) \cdot L_4 \cdot \sin(q_4 + 90) + \cos(q_1) \cdot \cos(q_2) L_3 \cdot \cos(q_3) - \cos(q_1) \sin(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \cos(q_2) \cdot \cos(q_1)$$

$$py = (\sin(q_1) \cdot \cos(q_2) \cos(q_3) - \sin(q_1) \sin(q_2) \sin(q_3)) \cdot L_4 \cdot \cos(q_4 + 90) - (\sin(q_1) \sin(q_2) \sin(q_3) - \sin(q_1) \sin(q_2) \cos(q_3)) \cdot L_4 \cdot \sin(q_4 + 90) + \sin(q_1) \cdot \cos(q_2) L_3 \cdot \cos(q_3) - \sin(q_1) \sin(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \sin(q_2) \cdot \sin(q_1)$$

$$pz = (\sin(q_2) \cos(q_3) + \cos(q_2) \sin(q_3)) \cdot L_4 \cdot \cos(q_4 + 90) - (\sin(q_3) \sin(q_2) + \cos(q_2) \cos(q_3)) \cdot L_4 \cdot \sin(q_4 + 90) + \sin(q_2) L_3 \cdot \cos(q_3) + \cos(q_2) L_3 \cdot \sin(q_3) + L_2 \cdot \sin(q_2) + L_1$$

Ahora hallamos las posiciones respecto de la posición de referencia que en este caso es el origen del sistema de referencia (x0, y0, z0):

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{aligned} x &= -(Cos(q_1).Cos(q_2)Cos(q_3) - Cos(q_1)Sen(q_2)Sen(q_3)).L_4.Cos(q_4 + 90) - (Cos(q_1)Sen(q_2)Sen(q_3) - Cos(q_1)Sen(q_2)Cos(q_3)).L_4.Sen(q_4 + 90) \\ &\quad + Cos(q_1).Cos(q_2)L_3.Cos(q_3) - Cos(q_1)Sen(q_2)L_3.Sen(q_3) + L_2.Cos(q_2).Cos(q_1) \\ y &= (Sen(q_1).Cos(q_2)Cos(q_3) - Sen(q_1).Sen(q_2)Sen(q_3)).L_4.Cos(q_4 + 90) - (Sen(q_1).Sen(q_2)Sen(q_3) - Sen(q_1).Sen(q_2)Cos(q_3)).L_4.Sen(q_4 + 90) \\ &\quad + Sen(q_1).Cos(q_2)L_3.Cos(q_3) - Sen(q_1).Sen(q_2)L_3.Sen(q_3) + L_2.Sen(q_2).Sen(q_1) \\ z &= (Sen(q_2)Cos(q_3) + Cos(q_2)Sen(q_3)).L_4.Cos(q_4 + 90) - (Sen(q_3)Sen(q_2) + Cos(q_2)Cos(q_3)).L_4.Sen(q_4 + 90) + Sen(q_2)L_3.Cos(q_3) \\ &\quad + Cos(q_2)L_3.Sen(q_3) + L_2.Sen(q_2) + L_1 \end{aligned}$$

2. PROGRAMACIÓN

2.1 PROGRAMACIÓN CON EL DSP56800/E

Para poder realizar la prueba, generamos un programa en C utilizando el CodeWarrior, el cual nos ofrece una plataforma de trabajo sobre el DSP56800/E. En dicho programa, fuimos variando los parámetros de las articulaciones para generar todos los casos posibles y analizar el comportamiento del robot.

Para simplificar el análisis, decidimos usar para los valores de q_1 , q_2 y q_3 el tipo de dato `frac16`, la cual es la notación utilizada para los puntos flotantes en este tipo de microprocesadores.

Por simplicidad, a continuación transcribimos el código genérico utilizado. Se debe tener en cuenta las restricciones de variabilidad de los parámetros para cada caso.

```
/*#####  
**  Filename : TP1.C  
**  Project  : TP1  
**  Processor : 56F8367  
**  Version   : Driver 01.13  
**  Compiler  : Metrowerks DSP C Compiler  
**  Date/Time : 24/05/2011, 10:38  
#####*/  
/* MODULE TP1 */
```

```
/* Including needed modules to compile this module/procedure */  
#include "Cpu.h"  
#include "Events.h"  
#include "TFR1.h"  
#include "MFR1.h"  
#include "MEM1.h"  
/* Including shared modules, which are used for whole project */  
#include "PE_Types.h"  
#include "PE_Error.h"  
#include "PE_Const.h"  
#include "IO_Map.h"  
#include "stdio.h"  
#define MXRAD 90  
#define PULSE2RAD 32767/MXRAD  
#define L1 75  
#define L2 125  
#define L3 126  
#define L4 45  
Frac16 c,i;;  
Frac16 pulse2rad=PULSE2RAD,testResult[MXRAD],testResult2[MXRAD];
```



```

/* Ángulo q1 */ Frac16 q1[MXRAD];
/* Ángulo q2 */ Frac16 q2[MXRAD];
/* Ángulo q3 */ Frac16 q3[MXRAD];
/* Ángulo q4 */ Frac16 q4[MXRAD];

/*Matriz Homogenea*/
Frac16 Total[4][4];
Frac16 x,y,z,aux,aux1,aux2,aux3,aux4;

Word16 c16[MXRAD]; Word32
c32[MXRAD];

void main(void)
{
    int j;

    PE_low_level_init();
    /** End of Processor Expert internal initialization.      ***/

    /* Write your code here */

for(i=0;i<MXRAD;i++)
{
    /*Valorizacion de los argumentos*/
    C32[i]=(L_mult(pulse2rad,i));
    q1[i]=extract_l(c32[i]);
    q2[i]=extract_l(c32[i]);
    q[i]=extract_l(c32[i]);
    q4[i]=extract_l(c32[i]);

    /*Elemento Fila 1 Columna 4 (x)*/
    aux= mult(TFR1_tfr16CosPlx(q2[i]), TFR1_tfr16CosPlx(q3[i]));
    aux2=mult(TFR1_tfr16SenPlx(q2[i]), TFR1_tfr16SenPlx(q3[i]));
    aux1=add(aux,negate(aux2));
    /*1° termino*/
    aux3=mult(aux1, L4*TFR1_tfr16SenPlx(q4[i]));
    /*2° termino*/
    aux4=mult(add(aux2,negate(mult(TFR1_tfr16SenPlx(q2[i]), TFR1_tfr16CosPlx(q3[i])))),L4*TFR1_tfr16CosPlx(q4[i]));
    aux5=add(aux3,add(aux4,aux1*L3));
    Total[0][3]=mult(TFR1_tfr16CosPlx(q1[i]),add(aux5,L2* TFR1_tfr16CosPlx(q2[i])));

    /*Elemento Fila 2 Columna 4 (y) */
    aux5=add(negate(aux3),add(aux4,aux1*L3));
    Total[1][3]=mult(TFR1_tfr16SenPlx(q1[i]),add(aux5,L2* TFR1_tfr16SenPlx(q2[i])));

    /*Elemento Fila 3 Columna 4 (z)*/

```

```

aux1=add(aux,aux2);
/*2° termino*/
aux3=mult(aux1, (L4*TFR1_tfr16CosPlx(q4[i])));
aux= mult(TFR1_tfr16CosPlx(q2[i]), TFR1_tfr16SenPlx(q3[i]));
aux2=mult(TFR1_tfr16SenPlx(q2[i]), TFR1_tfr16CosPlx(q3[i]));
aux1=add(aux,(aux2));
/*1° termino*/
aux4=mult(aux1,(L4* TFR1_tfr16SenPlx(q4[i]));
aux5= add(negate(aux4),negate(aux3));
Total[2][3]=add(aux4,add((aux1*L3),add((TFR1_tfr16SenPlx(q2[i])*L2),L1)));

/* Calculamos las coordenadas cartesianas del punto extremo del robot ( X Y Z)*/

x=Total[0][3];
y=Total[1][3];
z=Total[2][3];

/*Imprimo X,Y,Z y los argumentos q1,q2,l3*/
printf ("%d %d %d\n",x,y,z);
Print("Q1= %d\t Q2=%d\t Q3= %d\t Q4=%d\n"q1[i],q2[i],q3[i],q4[i]);

    }
}

/* END TPN1 */

```

2.2 GRAFICOS EN MATLAB

En primera instancia con los toolbox de robotica de matlab chequeamos que la matriz este correcta .

Escribiendo el siguiente código:

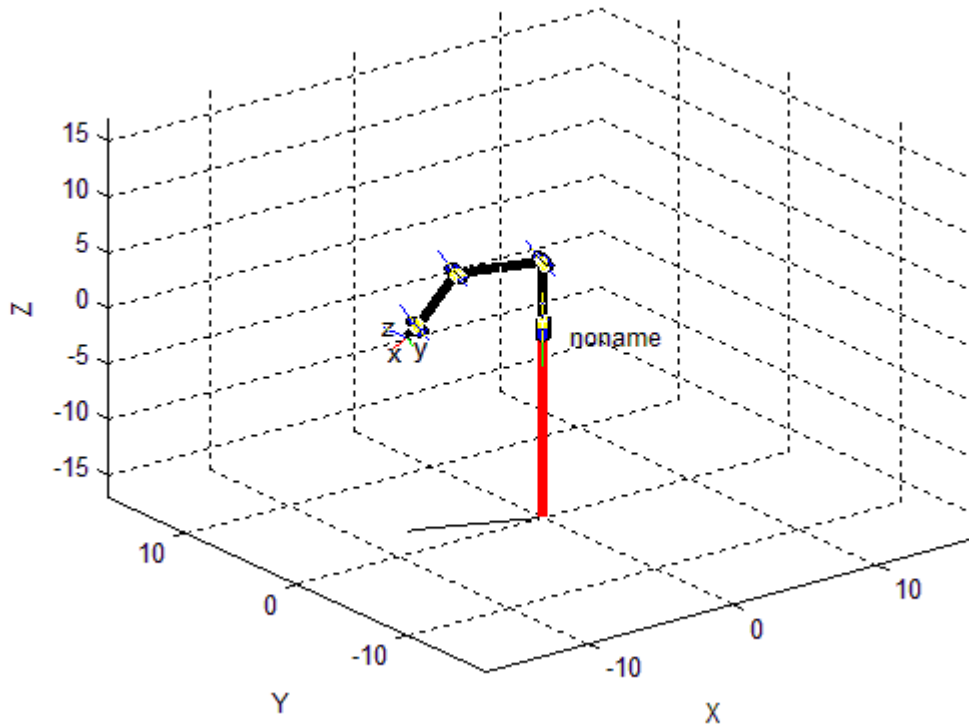
```

O1=0;
O2=0;
O3=0;
O4=0;
L1=link([(pi/2) 0 O1 6 0], 'standard')
L2=link([0 5 O2 0 0], 'standard')
L3=link([0 5 O3 0 0], 'standard')
L4=link([(pi/2) 1 (O4+pi/2) 0 0], 'standard')
r=robot({L1 L2 L3 L4})
syms q1 q2 q3 q4
plot(r,[0 0 0 0])
drivebot(r)

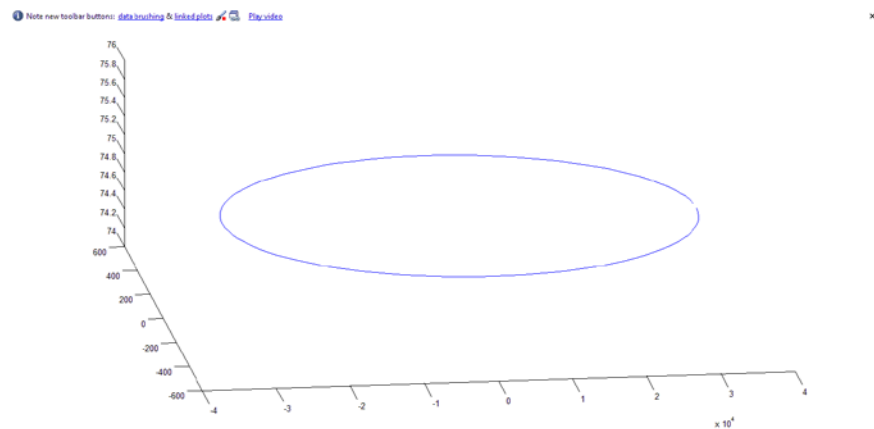
```

Tomando los valores obtenidos por el dsp graficamos los puntos x,y,z

Se visualiza la modelización del robot.



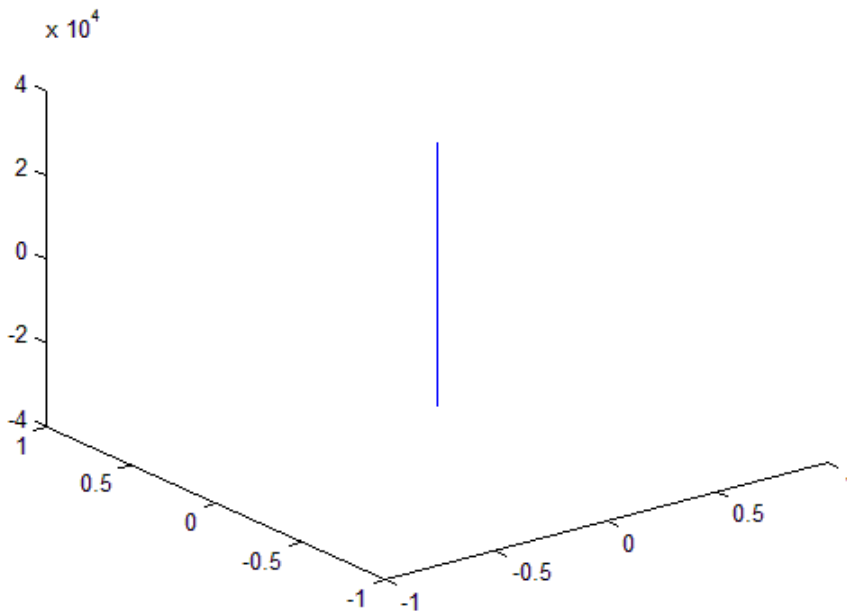
Haciendo variar Q_1 se tiene




Haciendo variar Q2 solamente se tiene:

Note new toolbar buttons: [data brushing](#) & [linked plots](#)   [Play video](#)

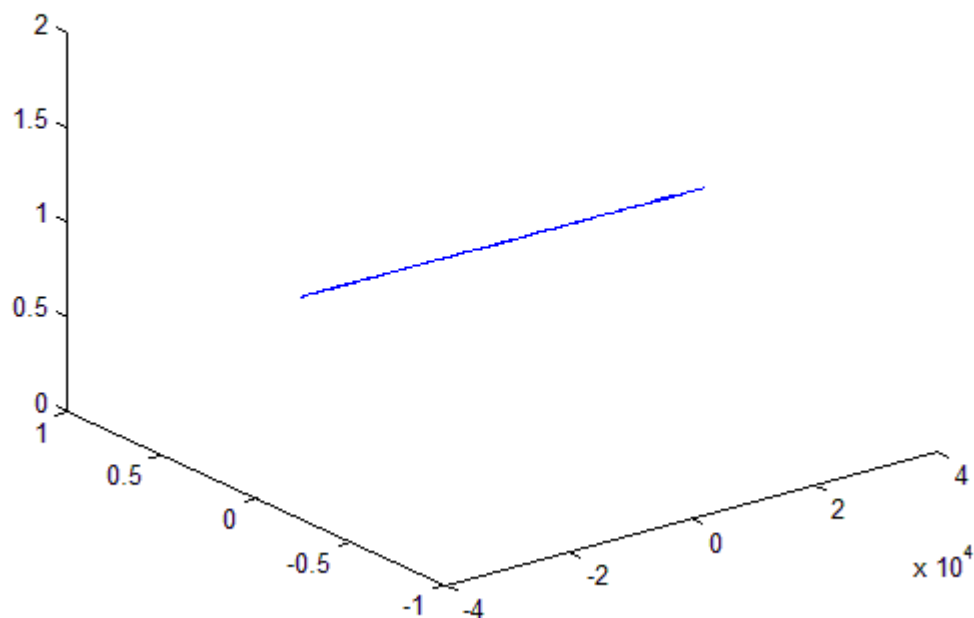
×



Haciendo variar Q4

Note new toolbar buttons: [data brushing](#) & [linked plots](#)   [Play video](#)

×

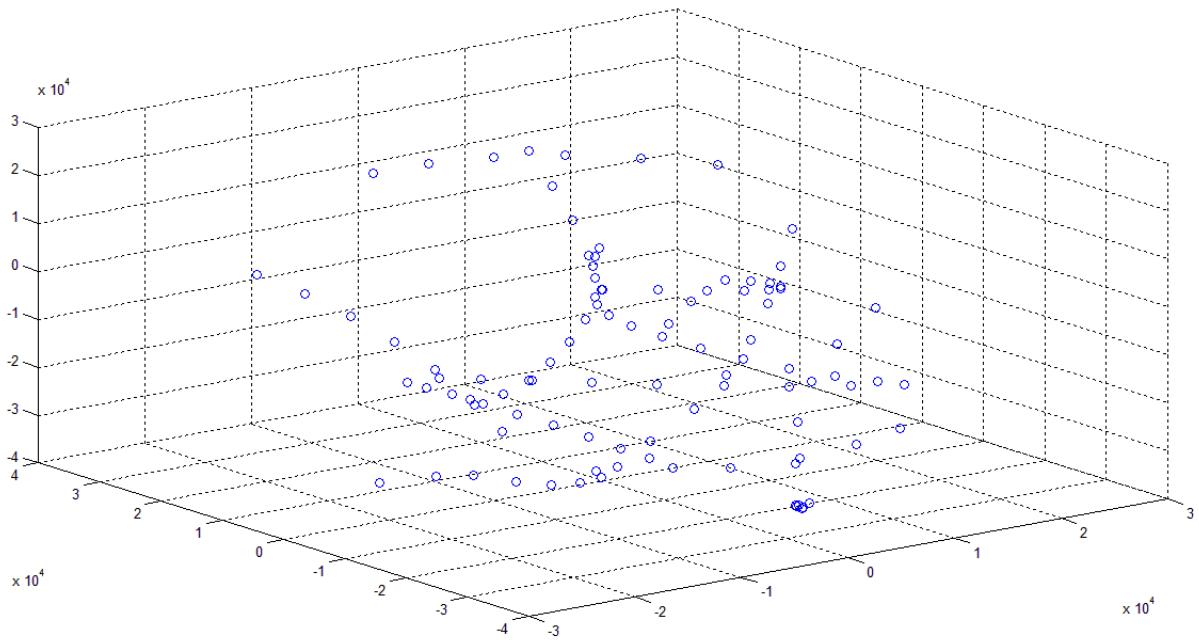


Ahora si hacemos mover todas las articulaciones.

Gráficos de algunos puntos en el espacio:

Note new toolbar buttons: [data brushing](#) & [linked plots](#)  [Play video](#)

x



Si se graficaran todos los puntos se describirían todo el espacio de trabajo. Generando una semi esfera.

3. CONCLUSIÓN

La utilización del CodeWarrior fue una herramienta excelente, ya que nos permite desarrollar y simular virtualmente nuestros programas sobre un DSP. Una de las ventajas también importantes es que viene con un set de funciones ya armadas para la utilización en nuestros programas, lo cual nos permite enfocarnos directamente sobre el problema.

También la utilización del Matlab con el toolbox de robotica para verificar y graficar los resultados, lo cual nos permitió de una forma fácil, observar cómo se iba desplazando el brazo del robot, de acuerdo a los parámetros que iban variando.

Es importante destacar que cuando resolvemos los ejercicios matemáticamente, no tenemos en cuenta los límites físicos que tiene el robot. Gracias a que pudimos graficar el desplazamiento, pudimos ver y tomar precauciones en el rango de variación de los parámetros.

Por último, de acuerdo a todos los gráficos anteriores, podemos observar que no es posible generar un movimiento en línea recta variando los 3 parámetros al mismo tiempo. Para lograr esto, se debería utilizar la teoría de la cinemática inversa, y calcular para una trayectoria dada, los parámetros necesarios de las articulaciones.