UNIVERSIDAD TECNOLÓGICA NACIONAL - FRBA

ROBOTICA

TRABAJO PRACTICO DE LABORATORIO Nº 1

Implementación de una Matriz Cinemática en DSP

ALUMNOS: Perilli, Hernán
Granzella, Damián

CODIGO: 95-0482

CURSO: R6055

DOCENTE: Ing. Hernán Giannetta

AÑO: 2009

Cinemática del Robot

Introducción Teórica

La cinemática del robot estudia el movimiento del mismo respecto a un sistema de referencia. Este estudio 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.
El problema cinemático directo del robot tal como es abordado en el presente trabajo, consiste en determinar 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.

Denavit y Hartenberg propusieron un método sistemático para describir y presentar la geometría espacial de los elementos de una escala cinemática, y en particular de un robot, con respecto a un sistema de referencia fijo. Este método utiliza una matriz de transformación homogénea para describir la relación espacial entre dos elementos rígidos adyacentes, reduciéndose el problema cinemático directo a encontrar una matriz de transformación homogénea de 4x4 que relacione la localización espacial del extremo del robot con respecto al sistema de coordenadas de su base.

Por otro lado, la cinemática del robot trata también de encontrar las relaciones entre las velocidades del movimiento de las articulaciones y las del extremo. Esta relación viene dada por el modelo diferencial expresado mediante la matriz Jacobiana. Este estudio no será desarrollado en el presente trabajo.

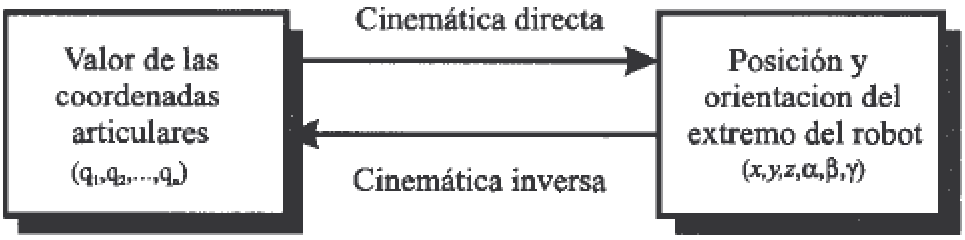
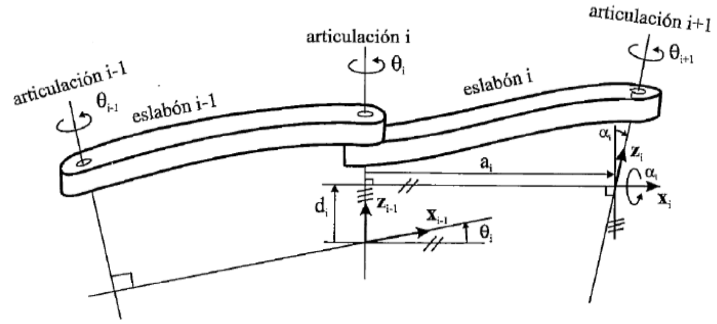


Diagrama de relación entre cinemática directa e indirecta.

Resolución del problema cinemático directo mediante matrices de transformación homogénea

Como ya se ha descripto anteriormente, 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 ó 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 homogénea de transformación 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.
Entonces, la resolución del problema cinemático directo consistirá en encontrar las relaciones que permiten conocer la localización espacial del extremo del robot a partir de los valores de sus coordenadas articulares.

Se analiza el siguiente ejemplo:



$$^{0}\mathbf{A}_3 = {}^{0}\mathbf{A}_1\, {}^{1}\mathbf{A}_2\, {}^{2}\mathbf{A}_3$$

Según la representación de D-H, escogiendo adecuadamente los sistemas de coordenadas asociados a cada eslabón, será posible pasar de uno al siguiente mediante cuatro transformaciones básicas que dependen exclusivamente de las características geométricas del eslabón.
Estas transformaciones básicas consisten en una sucesión de rotaciones y traslaciones que permiten relacionar el sistema de referencia del elemento 'i' con el sistema del elemento 'i-1'. Las transformaciones en cuestión son las siguientes:

1. Rotación alrededor del eje Z_{i-1} un ángulo Φ_i.
2. Traslación a lo largo de Z_{i-1} una distancia d_i; vector $d_i(0,0,d_i)$.
3. Traslación a lo largo de X_i una distancia a_i; vector $a_i(0,0,a_i)$.
4. Rotación alrededor del eje X_i un ángulo α_i.

Entonces, de acuerdo a la definición de estas últimas normas expuestas, la matriz de transformación homogénea para pasar del sistema S_{i-1} al S_i quedaría:

$$^{i-1}\mathbf{A}_i = \mathbf{T}(\mathbf{z},\theta_i)\, \mathbf{T}(0,0,d_i)\, \mathbf{T}(a_i,0,0)\, \mathbf{T}(\mathbf{x},\alpha_i)$$

$$^{i-1}\mathbf{A}_i = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Donde Φ_i, a_i, d_i, α_i, son los parámetros D-H del eslabón i. De este modo, basta con identificar los parámetros Φ_i, a_i, d_i, α_i para obtener las matrices A y relacionar así todos y cada uno de los eslabones del robot.

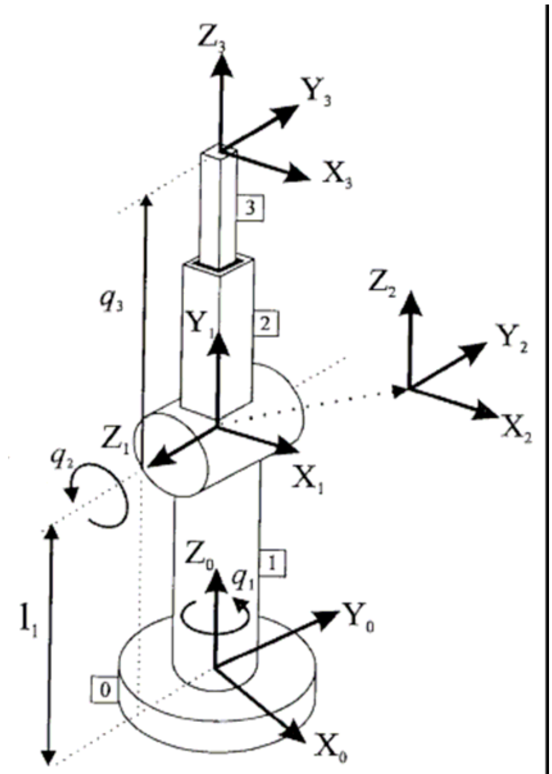
Luego de esto, restaría resolver cada una de las matrices $^{i-1}A_i$ correspondientes a cada una de las tres articulaciones y por último obtener la matriz de transformación homogénea T.
Luego de obtenida T se podrá conocer la posición de extremo del robot situado en el origen del sistema cartesiano S_3, en coordenadas cartesianas del sistema S_0 situado en la base del robot, que se adoptó como sistema de referencia total.

$$T = {}_0A_1 . {}_1A_2 . {}_2A_3$$

Desarrollo de la Práctica

El desarrollo de la práctica se basa en implementar en código C en CodeWarrior para DSP 56800/E de la cadena cinemática directa del siguiente manipulador, utilizando la matriz homogénea, y usando como setpoints, una trayectoria lineal en cada eje. Definir los límites y área de trabajo del manipulador.

Figura del manipulador



Articulaciones y movimientos

Articulación	θ	d	a	α
1	q1	L1	0	90
2	q2	0	0	-90
3	0	q3	0	0

Desarrollo y obtención de la matriz transferencia por el método de Denavit-Hartenberg

A partir de los datos anteriormente dados, se obtienen las matrices A y transferencia T

$$
{}^{0}\mathbf{A}_1 = \begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\qquad
{}^{1}\mathbf{A}_2 = \begin{bmatrix} C_2 & 0 & -S_2 & 0 \\ S_2 & 0 & C_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{2}\mathbf{A}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{0}\mathbf{A}_2 = \begin{bmatrix} C_1C_2 & -S_1 & -C_1S_2 & 0 \\ S_1C_2 & C_1 & -S_1S_2 & 0 \\ S_2 & 0 & C_2 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Finalmente la matriz T se obtiene

$$
\mathbf{T} = {}^{0}\mathbf{A}_1 \, {}^{1}\mathbf{A}_2 \, {}^{2}\mathbf{A}_3
$$

$$
\mathbf{T} = {}^{0}\mathbf{A}_3 = \begin{bmatrix} C_1C_2 & -S_1 & -C_1S_2 & -q_3\,C_1S_2 \\ S_1C_2 & C_1 & -S_1S_2 & -q_3\,S_1S_2 \\ S_2 & 0 & C_2 & q_3\,C_2 + l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

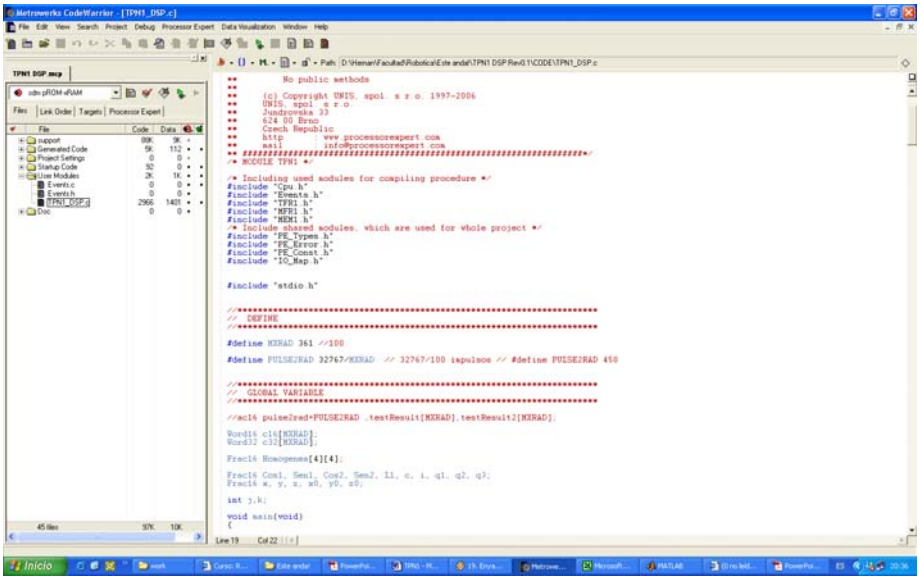
Luego queda:

$$
\begin{bmatrix} x \\ y \\ z \end{bmatrix} = T. \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix}
$$

Y desarrollando la multiplicación de matrices se obtienen las posiciones espaciales x, y, z.

Desarrollo de la matriz T en código C para CodeWarrior

La finalidad del desarrollo en el DSP, es implementar la matriz obtenida en una matriz de 4x4, e ir variando y alternando los valores de q1, q2 y q3, para un valor fijo de L1, y obtener, para casa juego de puntos, las correspondientes coordenadas espaciales x; y; z. La finalidad de implementar el desarrollo en DSP y no en microcontrolador, es optimizar la velocidad de procesamiento y así el resultado final, ya que e microcontrolador ejecuta de a una instrucción por vez, no así el DSP que tiene procesamientos en paralelo. Para ello se utilizaré el DSP56800/E y el desarrollo se hará en CodeWarrior de Freescale.



Para este caso se tomarán x0 = y0 =z0 = 0, por lo tanto (haciendo las multiplicaciones de matrices) no se observarán rotaciones, sino solamente traslaciones.

El código C implementado es el siguiente:

```
#include "Cpu.h"
#include "Events.h"
#include "TFR1.h"
#include "MFR1.h"
#include "MEM1.h"
/* Include 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
//***********************************************************************

#define MXRAD 361 //100

#define PULSE2RAD 32767/MXRAD  // 32767/100 impulsos // #define PULSE2RAD 450


//***********************************************************************
//          GLOBAL VARIABLE
//***********************************************************************

Frac16 Homogenea[4][4];

Frac16 Cos1, Sen1, Cos2, Sen2, L1, c, i, q1, q2, q3;
Frac16 x, y, z, x0, y0, z0;

int j,k;

void main(void)
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  PE_low_level_init();
  /*** End of Processor Expert internal initialization.                    ***/

  /* Write your code here */

                    c=0;
                    q1 = 0;
                    q2 = 0;
                    q3 = 0;
                    L1 = 2000;
                    x0 = 0;
                    y0 = 0;
                    z0 = 0;


                    /*Cos1 = tfr16CosPIx (q1);
                    Cos2 = tfr16CosPIx (q2);
                    Sen1 = tfr16SinPIx (q1);
                    Sen2 = tfr16SinPIx (q2);

                    Homogenea[0][0] = mult(Cos1, Cos2);
```

```
		Homogenea[0][1] = mult(Sen1, Cos2);
		Homogenea[0][2] = Sen2;
		Homogenea[0][3] = 0;
		Homogenea[1][0] = negate(Sen1);
		Homogenea[1][1] = Cos1;
		Homogenea[1][2] = 0;
		Homogenea[1][3] = 0;
		Homogenea[2][0] = negate(mult(Cos1, Sen2));
		Homogenea[2][1] = negate(mult(Sen1, Sen2));
		Homogenea[2][2] = Cos2;
		Homogenea[2][3] = 0;
		Homogenea[3][0] = negate(mult(mult(Cos1, Sen2),q3));
		Homogenea[3][1] = negate(mult(mult(Sen1, Sen2),q3));
		Homogenea[3][2] = add(L1,mult(Cos2,q3));
		Homogenea[3][3] = 1;*/


		printf ("\nValores para L1 = %d", L1);
		printf ("\n\nx\ty\tz");

		printf ("\n\n!!!!!!!!!! VARIANDO q1 SOLAMENTE!!!!!!!!!!!!\n\n");
		for(q1=0, q2=0, q3=0; q1<30000; q1 = q1+500)
		{
			Cos1 = tfr16CosPIx (q1);
			Cos2 = tfr16CosPIx (q2);
			Sen1 = tfr16SinPIx (q1);
			Sen2 = tfr16SinPIx (q2);
			Homogenea[0][0] = mult(Cos1, Cos2);
			Homogenea[0][1] = mult(Sen1, Cos2);
			Homogenea[0][2] = Sen2;
			Homogenea[0][3] = 0;
			Homogenea[1][0] = negate(Sen1);
			Homogenea[1][1] = Cos1;
			Homogenea[1][2] = 0;
			Homogenea[1][3] = 0;
			Homogenea[2][0] = negate(mult(Cos1, Sen2));
			Homogenea[2][1] = negate(mult(Sen1, Sen2));
			Homogenea[2][2] = Cos2;
			Homogenea[2][3] = 0;
			Homogenea[3][0] = negate(mult(mult(Cos1, Sen2),q3));
			Homogenea[3][1] = negate(mult(mult(Sen1, Sen2),q3));
			Homogenea[3][2] = add(L1,mult(Cos2,q3));
			Homogenea[3][3] = 1;

			x = add(add(add(mult(Homogenea [0][0], x0), mult(Homogenea [1][0], y0)),
mult(Homogenea [2][0], z0)), Homogenea [3][0]);
			y = add(add(add(mult(Homogenea [0][1], x0), mult(Homogenea [1][1], y0)),
mult(Homogenea [2][1], z0)), Homogenea [3][1]);
			z = add(add(add(mult(Homogenea [0][2], x0), mult(Homogenea [1][2], y0)),
mult(Homogenea [2][2], z0)), Homogenea [3][2]);

	/*		for(j=0; j<4; j++)
			{

				printf ("%d \t %d \t %d \t %d \t \n",
Homogenea[0][j],Homogenea[1][j],Homogenea[2][j],Homogenea[3][j]);


			}*/
			printf ("\n%d\t%d\t%d", x, y, z);
		}
 }
```

Como se observa, lo que se hace a través de un lazo for, es ir armando la matriz transferencia para cada valor de q1, q2 y q3, dado un valor fijo de L1. Cabe hacer mención también que las operaciones matemáticas, como las multiplicaciones, sumas y operaciones trigonométricas, no se hace de la manera tradicional en lenguaje C, sino por medio de funciones y librerías especiales, debido a la implementación de éstas por medio del DSP y no del microcontrolador.

Trayectorias obtenidas

A continuación se detallarán distintas posibilidades y alternativas en base a las rotaciones de las distintas articulaciones, y la implementación en Matlab de los resultados obtenidos para la visualización de la trayectoria, por medio de la la función plot3 (x, y, z). Se graficarán distintas alternativas en base diferentes combinaciones posibles de movimientos de articulaciones.

Caso 1: Varía solamente q1 (q2 = 0; q3 = 0)

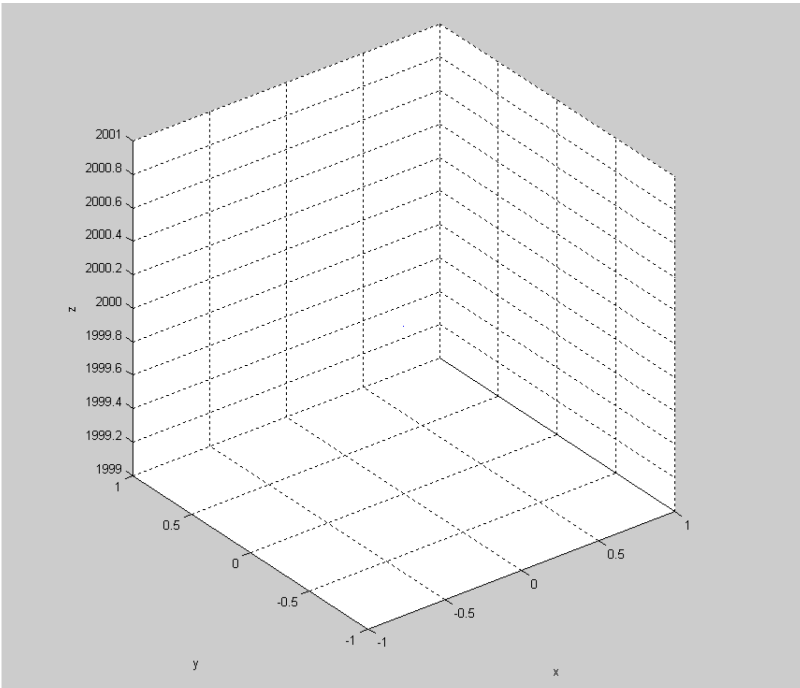


Fig. 1

Caso 2: Variado solamente q2 (q1 = 0; q3 = 0)

El gráfico obtenido es similar a la Fig. 1

Caso 3: Variado en forma simultánea q1 y q2 (q3 = 0)

El gráfico obtenido es similar a la Fig. 1

Caso 4: Variado solamente q3 (q1 = 0; q2 = 0)

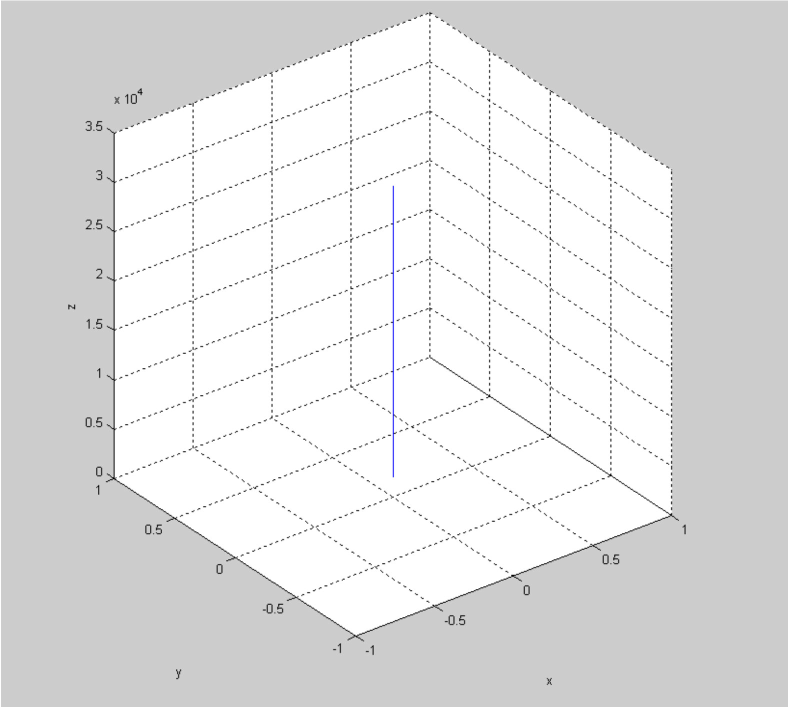


Fig. 2

Caso 5: Variado en forma simultánea q1 y q3 (q2 = 0)

El gráfico obtenido es similar a la Fig. 2

Caso 6: Variado en forma simultánea q2 y q3 (q1 = 0)

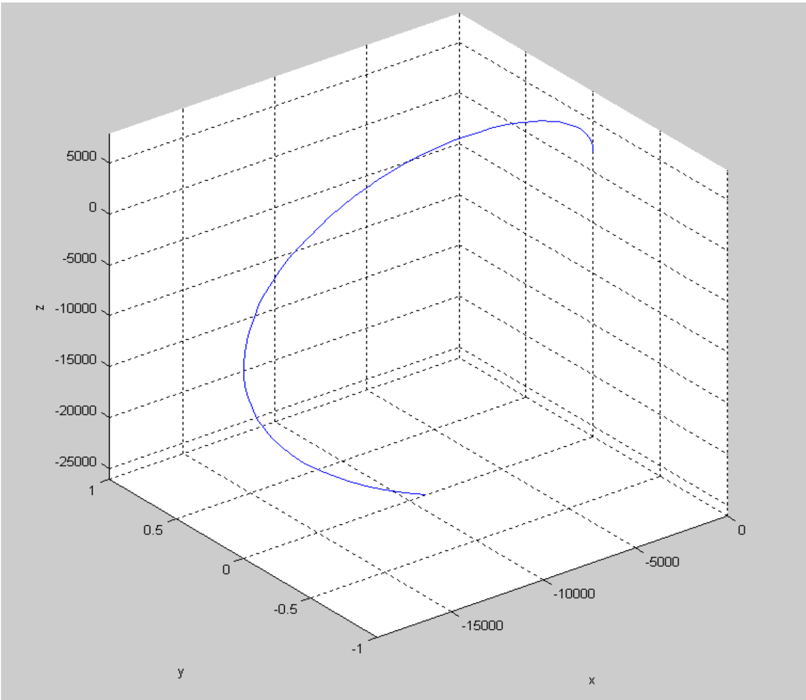


Fig. 3

Caso 7: Variado de igual manera q1, q2 y q3

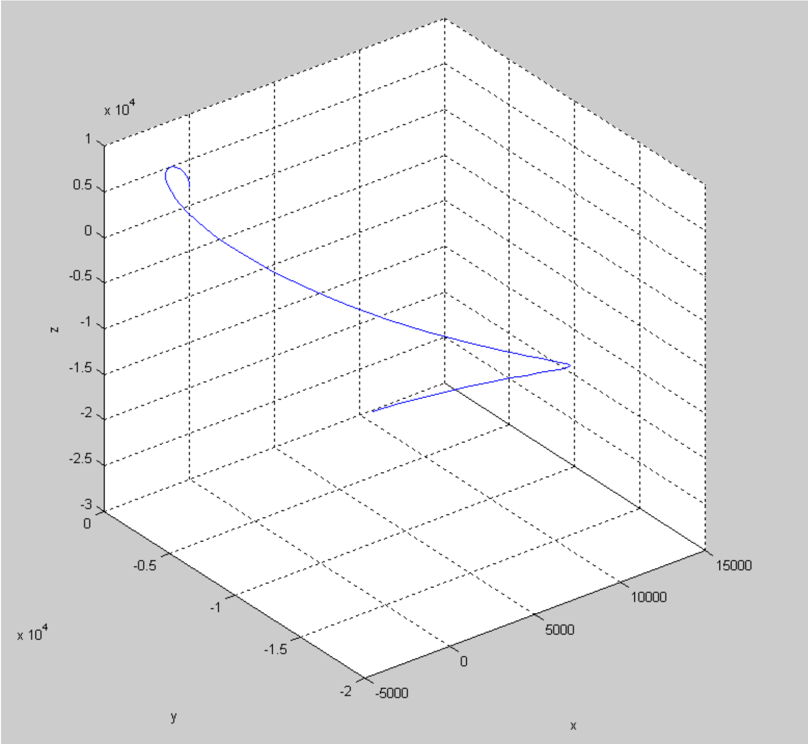


Fig. 4

Caso 8: Variado q1 < q2 < q3

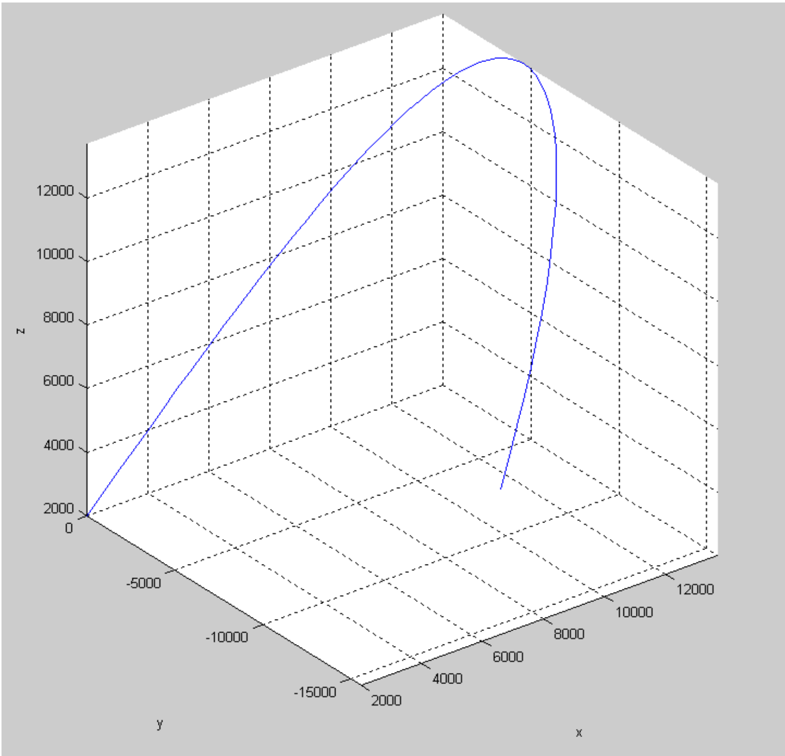


Fig. 5

Caso 9: Variado q1 > q2 > q3

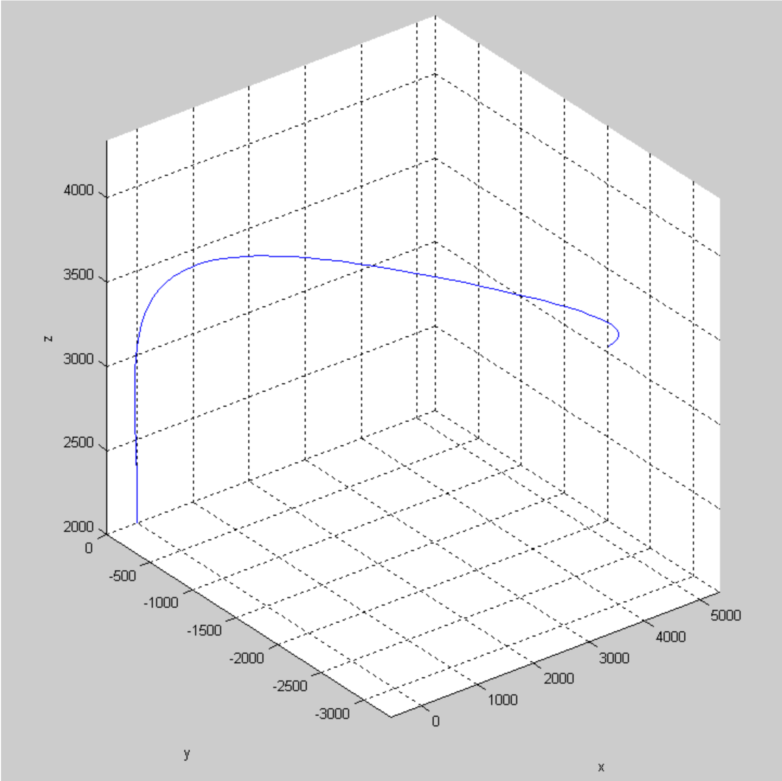


Fig. 6

Caso 10: Variado q2 > q1 > q3

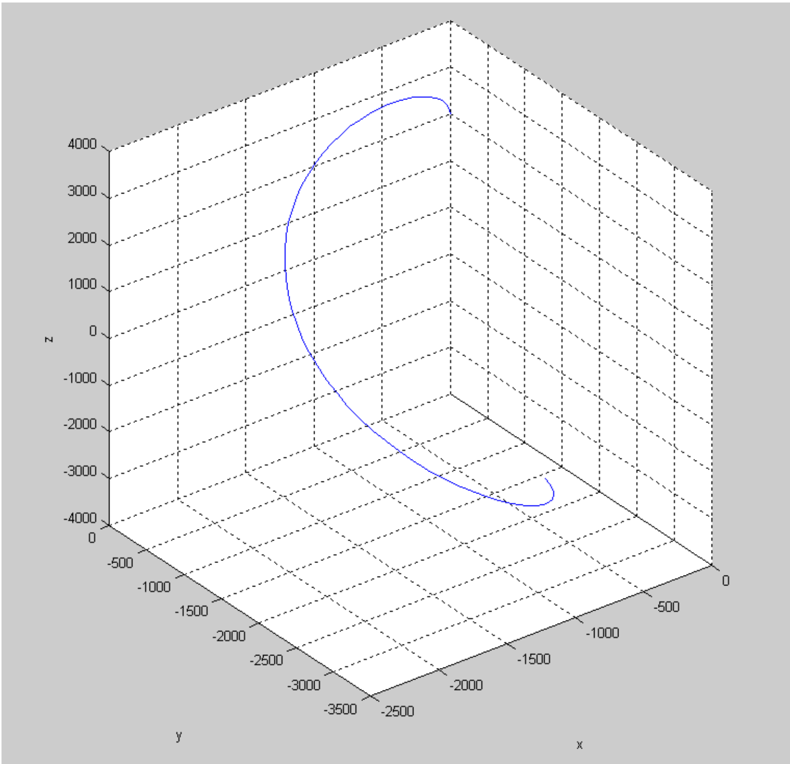


Fig. 7

Caso11: Variado q3 > q1 > q2

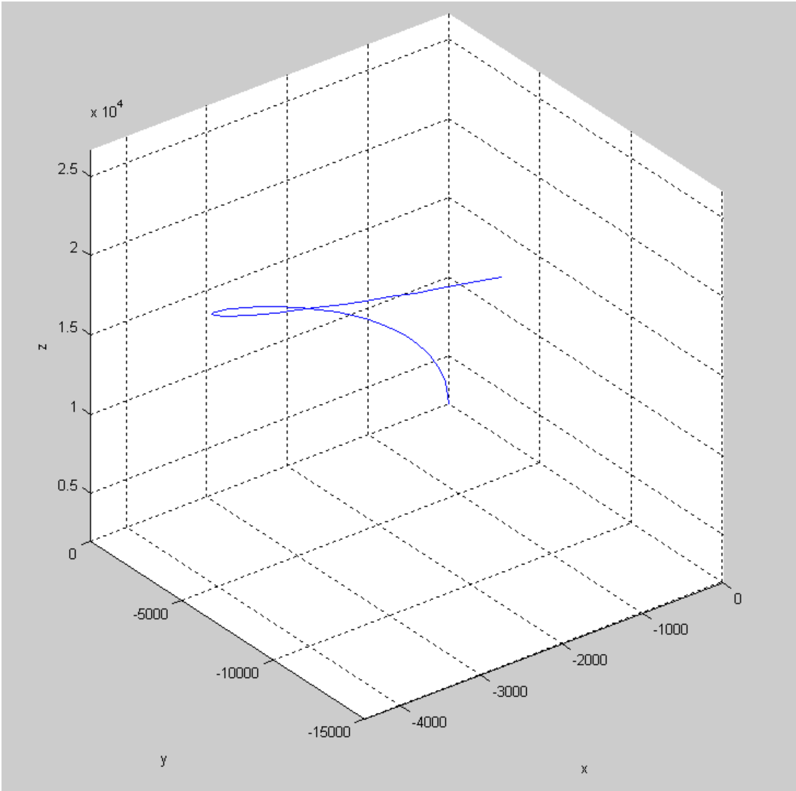


Fig. 8

Conclusiones:

Por medio de las experiencias realizadas se puede observar que a través de las operaciones matriciales, y la implementación de éstas a través de su procesamiento por medio del DSP, se podrá obtener en cada instante el punto final espacial del extremo de manipulador en base a las rotaciones de las articulaciones del mismo. Si incrementando y/o decrementando, o armando distintos juegos de de rotaciones de las correspondientes articulaciones, calculando para caso su punto final y ploteando todos los resultado obtenido en Matlab, se obtendrá al trayectoria del extremo desde el punto inicial hasta el final. Esto servirá para observar por todos los puntos espaciales por los que atravesará el robot y tomar dimensión del lugar y dimensiones por donde se moverá.

Se concluye finalmente que el método de Denavit-Hartenberg implementado en este caso para la determinación de la cinemática directa, con el fin de establecer las posiciones finales del brazo robot, y acumulando distintos puntos con las sucesivas variaciones de las articulaciones dadas, concuerdan con los resultados obtenidos, teniendo coherencia e implementación práctica para su desarrollo.