UTN – Departamento de Electrónica
Cátedra : Robótica

Trabajo Práctico Nº 1

Implementación de una matriz cinemática en un DSP

INTEGRANTES: BALEA, MIGUEL
MURUAGA, LEONARDO

PROFESOR: Hernán Giannetta

FECHA DE ENTREGA: 08 / 06 / 09

CURSO: R-6055

AÑO: 2009

Índice

Introducción:

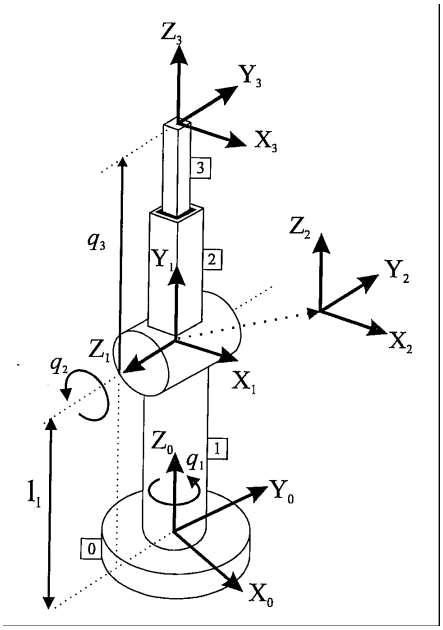
La cinemática es la parte de la mecánica clásica que estudia las leyes del movimiento de los cuerpos sin tener en cuenta las causas que lo producen, limitándose esencialmente, al estudio de la trayectoria en función del tiempo. Para el caso de un robot, se resume al estudio de su movimiento con respecto a un sistema de referencia.

Este estudio puede ser realizado de dos formas:

- Problema cinemático directo: Determinar la posición y orientación del extremo final del robot , con respecto a un sistema de coordenadas de referencia , conocidos los ángulos de las articulaciones y los parámetros geométricos de los elementos del robot.
- Problema cinemático inverso: Determinar la configuración que debe adoptar el robot para alcanzar una posición y orientación del extremo conocidas.

En este trabajo se resolverá el problema cinemático directo. Para ello se recurrirá al método de la matriz homogénea, según la representación de Denavit-Hartenberg, donde se obtiene una matriz de transformación T, que depende de la geometría del robot y que relaciona la representación entre un sistema y el de referencia.

Para el desarrollo de esta práctica se trabajará con el siguiente robot:



Los parámetros necesarios para el cálculo de la matriz homogénea del mismo son:

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

Como se observa en la tabla, este robot cuenta con 3 grados de libertad, definidos por q_1, q_2 y q_3.

Para hallar la matriz T, se procede de la siguiente forma:

$$ {}^0\mathbf{A}_1 = \begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1\mathbf{A}_2 = \begin{bmatrix} C_2 & 0 & -S_2 & 0 \\ S_2 & 0 & C_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^2\mathbf{A}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^0\mathbf{A}_2 = \begin{bmatrix} C_1C_2 & -S_1 & -C_1S_2 & 0 \\ S_1C_2 & C_1 & -S_1S_2 & 0 \\ S_2 & 0 & C_2 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{T}={}^0\mathbf{A}_3 = \begin{bmatrix} C_1C_2 & -S_1 & -C_1S_2 & -q_3C_1S_2 \\ S_1C_2 & C_1 & -S_1S_2 & -q_3S_1S_2 \\ S_2 & 0 & C_2 & q_3C_2 + l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Y finalmente, la posición del extremo del robot en el sistema de referencia será la siguiente:

$$ \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} x_3 \\ y_3 \\ z_3 \\ 1 \end{bmatrix} $$

Implementación:

Como resulta de observar el gráfico del robot, independientemente del movimiento que se realice la posición final en el sistema S3 será:

$$ \begin{bmatrix} x_3 \\ y_3 \\ z_3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} $$

Haciendo la multiplicación:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

se obtiene:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \begin{bmatrix} -q_3 \cdot C_1 \cdot S_2 \\ -q_3 \cdot S_1 \cdot C_2 \\ q_3 \cdot C_2 + L_1 \\ 1 \end{bmatrix}$$

Que es el cálculo que se realizará en el procesador. De esta forma se reducen las cuentas de 12 a 3 por punto calculado. Para aumentar la velocidad un poco más, se realizarán todos los cálculos por medio del DSP que tiene integrado el controlador. Se recurre como controlador al 56F8367 de Motorola.

El código en C desarrollado es el siguiente:

```
#include "Cpu.h"
#include "Events.h"
#include "TFR1.h"
#include "MFR1.h"
#include "MEM1.h"
#include "XFR1.h"
#include "AFR1.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 l1 32767*.5
#define PASO 32767/180      //Pasos de 1 grado
#define ANG_MAX 32768/2 // Maximo 90 grados


Frac16 T[4][4]=
{
{0,0,0,0},
```

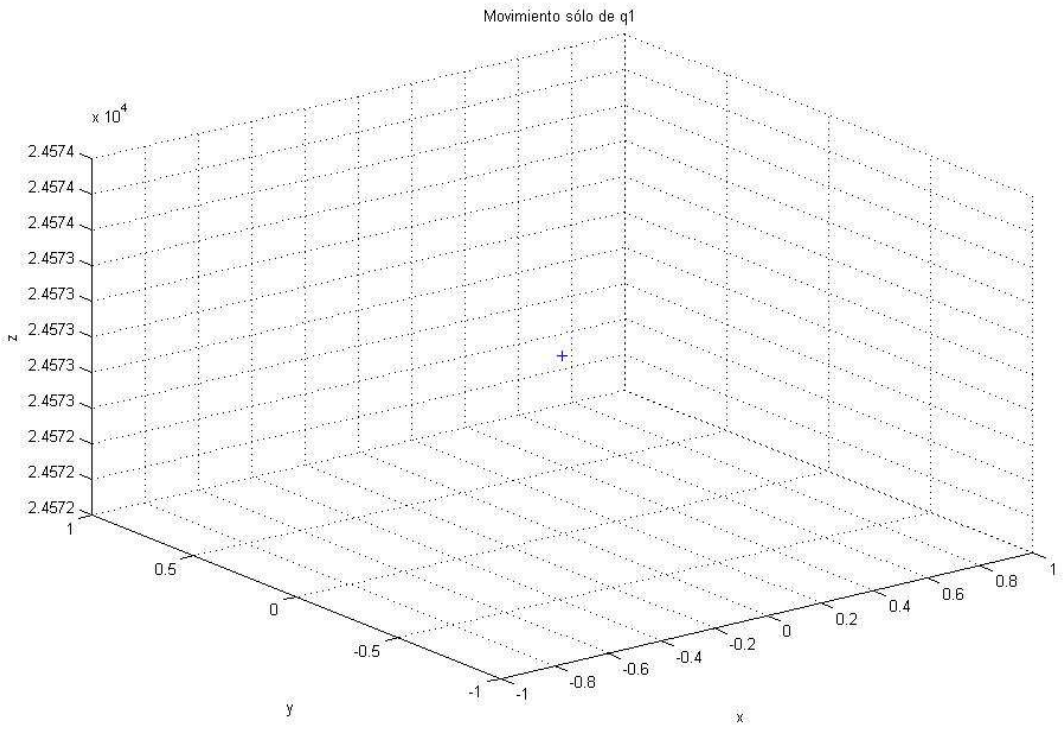
```
{0,0,0,0},
{0,0,0,0},
{0,0,0,1},
};

Frac16 init[4][1]={0,0,0,1};
Frac16 final[4][1]={0,0,0,1};

Frac16 q1, q2, q3;


void main(void)
{

/* Write your local variable definition here */
 Frac16 i;

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

  /* Write your code here */

 for(;;)
  {

          for (i=0; i<=ANG_MAX; i+=PASO)
          {         q1 = i;
                    q2 = i;
                    q3 = l1/4+i;


        T[1][4] = negate(mult(mult(q3,TFR1_tfr16CosPIx(q1)),TFR1_tfr16SinPIx(q2)));
        T[2][4] = negate(mult(mult(q3,TFR1_tfr16SinPIx(q1)),TFR1_tfr16SinPIx(q2)));
        T[3][4] = add(mult(q3,TFR1_tfr16CosPIx(q2)),l1);


          printf("%d \t %d \t %d \t %d \t\n", i, T[1][4],T[2][4],T[3][4]);
    }
  }
}
```

Resultados:

Los datos obtenidos por la simulación fueron graficados mediante Matlab.
Como se ve en el código, se trabajó con pasos de aproximadamente 1º, describiendo un movimiento total de 90º.

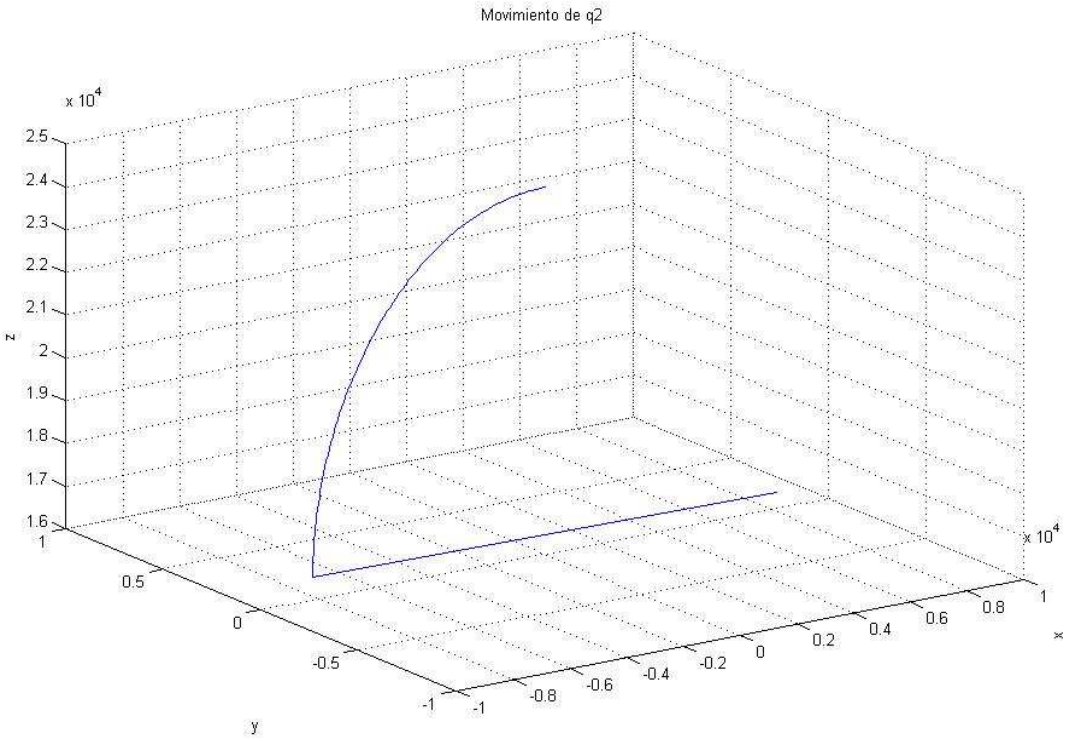
- Para un mejor análisis e interpretación se procedió en primer lugar a graficar el movimiento aislado de cada articulación:

  - Movimiento de q1:



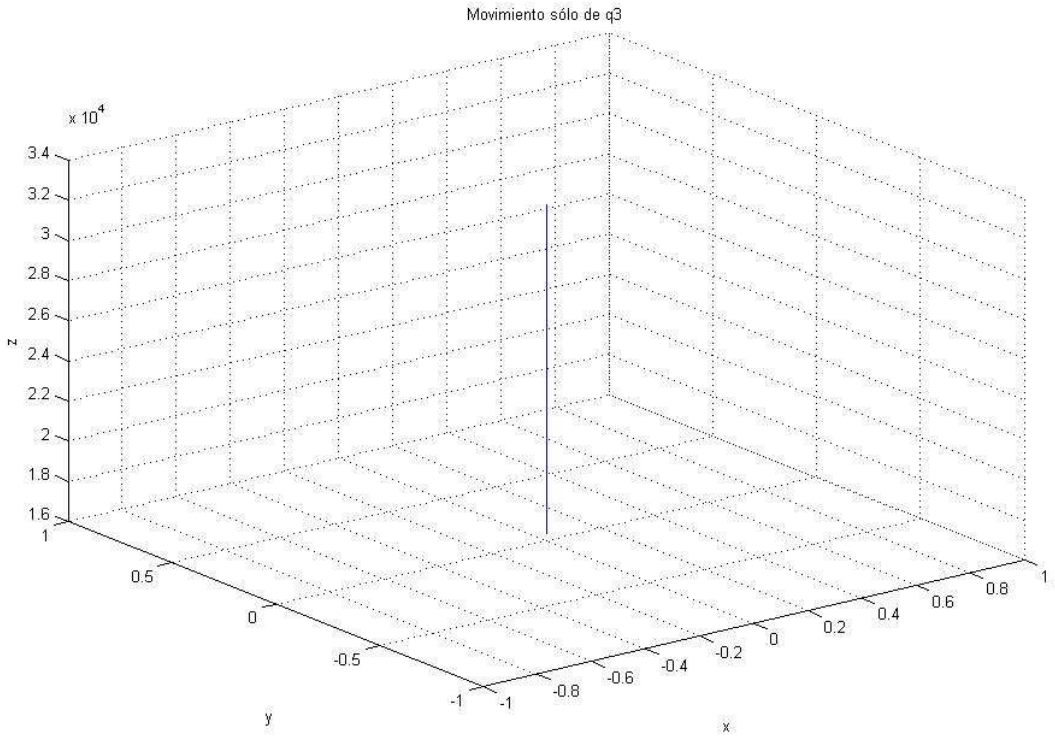
Se ha graficado una cruz para poder visualizar mejor el punto. Este movimiento da como resultado un punto, ya que con el brazo extendido, el extremo se encuentra sobre el eje de rotación del robot y el movimiento sólo lo hace girar sobre sí mismo.

- Movimiento de q2:



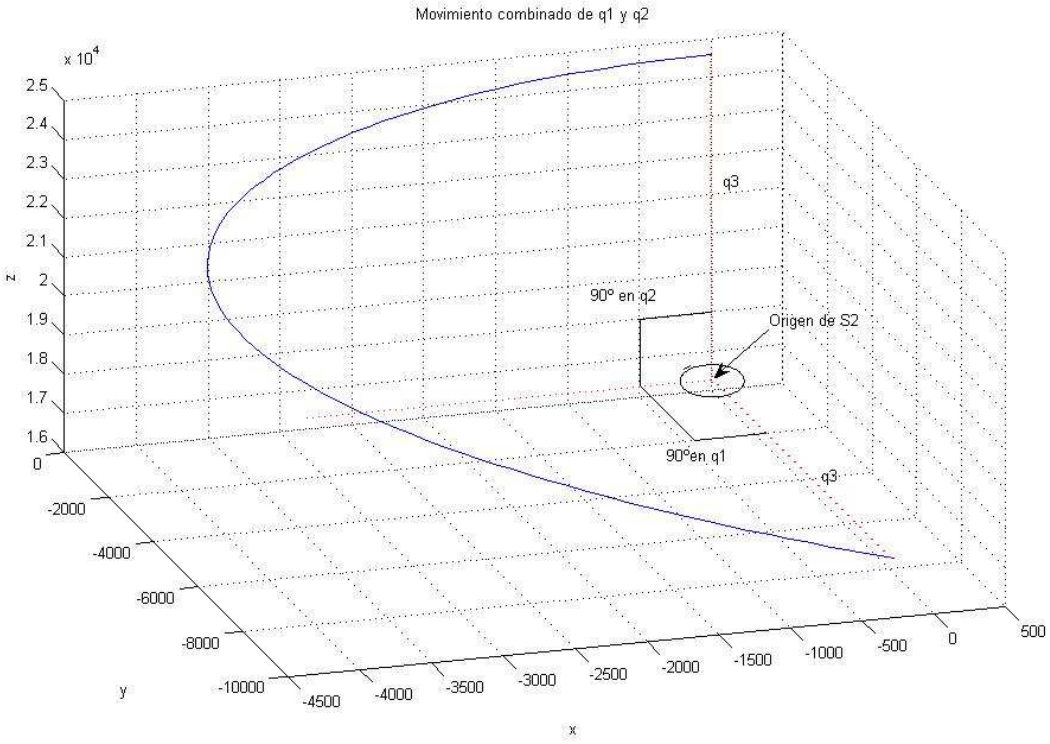
Se ve en la gráfica el desplazamiento de 90º ocasionado por la flexión del brazo. También se ve una zona recta en la misma describiendo un diámetro de la circunferencia a la que la curva pertenece.
Esta zona no debería pertenecer al movimiento del robot y se puede entender como un overflow de la variable utilizada para calcular los puntos. Esto causa que la misma se resetee y siga graficando en el inicio del rango de movimiento (-90º a 90º). Si se siguiera aumentando la variable para graficar se obtendría una semicircunferencia.
La causa más probable de este overflow es el redondeo que se realiza al hacer los define's de l paso y del ángulo máximo.

- Movimiento de q3:



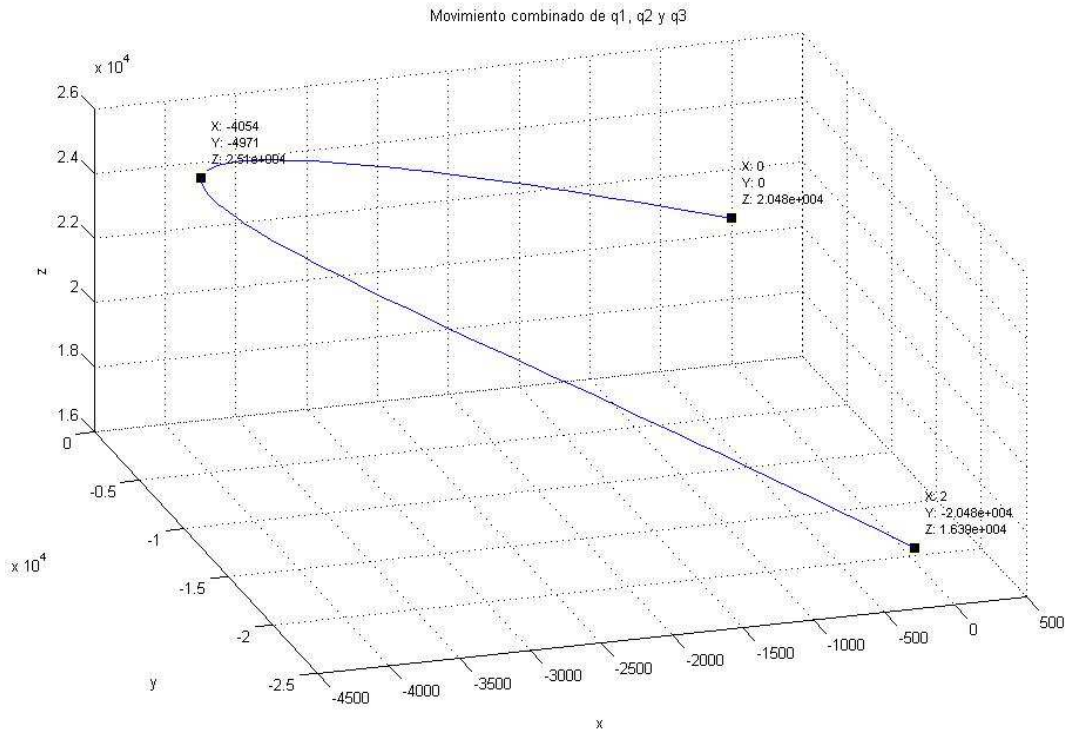
Dado que q_3 realiza un desplazamiento y el brazo está extendido, es lógico esperar que el punto se desplace sobre el eje z solamente.

- Como segundo paso se realizó un movimiento en conjunto de q_1 y q_2, obteniendo el siguiente desplazamiento:



Aquí se ven agregados ciertos elementos para poder interpretar el movimiento. Se puede observar el origen de S2 y cómo la diferencia entre el punto inicial y final corresponden a un desplazamiento de 90º para q_1 y q_2 simultáneamente.

- Finalmente se procede a realizar el movimiento conjunto de q_1, q_2 y q_3:



Se ve que el movimiento es muy similar al anterior, con la diferencia que ahora el punto se va alejando del origen debido al aumento lineal de q_3

Conclusiones:

Por medio de esta práctica se pudo confirmar el funcionamiento del método de la matriz homogénea para la cinemática directa. Además, se pudo fijar los conceptos del tema y aclarar dudas que surgen sólo de la implementación del método.