

TESIS FINAL

Materia: Robótica

Integrantes:

Febles, Matías
Titolo, Hernán
Gasulla, Juan Manuel

Profesor: Mas. Ing. Giannetta Hernan

JTP: Ing. Granzella Damian

Año: 2012

Cinemática del Robot

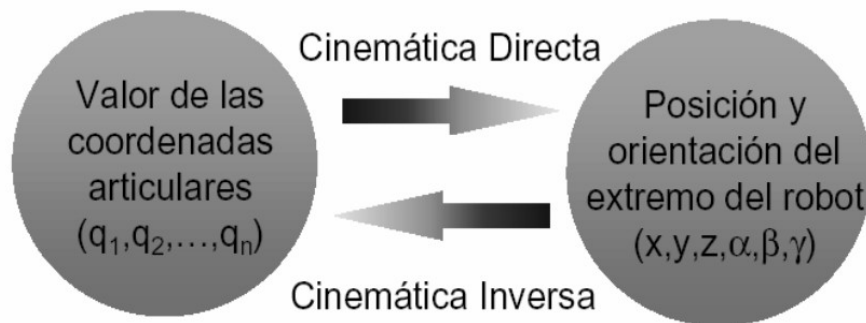
Introducción Teórica

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia. 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 de la herramienta del robot con los valores que toman sus coordenadas de sus articulaciones.

Existen dos problemas fundamentales a resolver con respecto a la cinemática del robot:

A) Cinemática Directa. Consiste en determinar la posición y orientación del extremo final del robot con respecto al sistema de la base del robot a partir de conocer los valores de las articulaciones y los parámetros geométricos.

B) Cinemática Inversa. Resuelve la configuración que debe adoptar el robot para una posición y orientación conocidas del extremo.

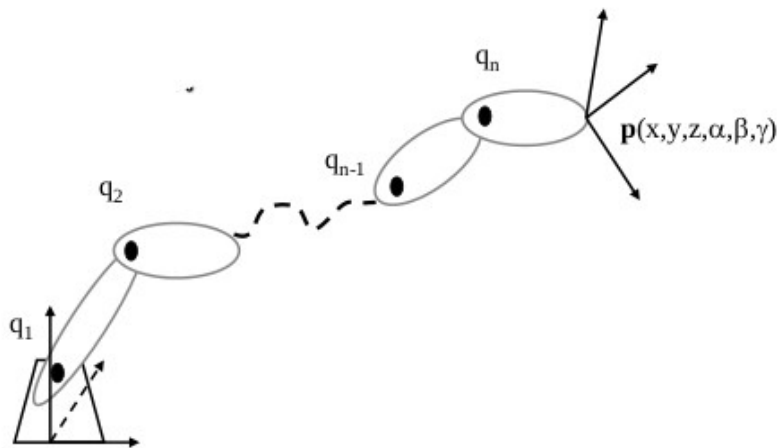


El desarrollo del presente TP se basará en resolver el problema de Cinemática Directa de nuestro manipulador.

El objetivo de la cinemática directa es encontrar una matriz de transformación homogénea **T** que relacione posición y orientación del extremo del robot con respecto a un sistema de referencia fijo y situado, por ejemplo en la base.

$$\begin{aligned}
 x &= f_x(q_1, q_2, q_3, q_4, q_5, q_6) \\
 y &= f_y(q_1, q_2, q_3, q_4, q_5, q_6) \\
 z &= f_z(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \alpha &= f_\alpha(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \beta &= f_\beta(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \gamma &= f_\gamma(q_1, q_2, q_3, q_4, q_5, q_6)
 \end{aligned}$$

Un robot de n grados de libertad está formado por n eslabones unidos por n articulaciones, de forma que cada par articulación-eslabón constituye un grado de libertad.



A cada eslabón se le puede asociar un sistema de referencias solidario a él y, utilizando

las transformaciones homogéneas, es posible representar las rotaciones y traslaciones

relativas entre los distintos eslabones que componen al robot.

La matriz que representa la posición y orientación relativa entre los sistemas asociados a dos eslabones consecutivos del robot suele denominarse matriz ${}^{i-1}\mathbf{A}_i$.

Cuando se consideran todos los grados de libertad mediante cada una de las matrices, la matriz resultante se la denomina matriz de transformación \mathbf{T} .

Algoritmo de Denavit-Hartenberg

En 1955 **Denavit y Hartenberg** propusieron un método matricial que permite establecer de manera sistemática un sistema de coordenadas. La representación de **Denavit-Hartenberg (D-H)** establece que seleccionándose adecuadamente los sistemas de coordenadas asociados a cada eslabón, será posible pasar de uno al siguiente mediante 4 transformaciones básicas que dependen exclusivamente de las características geométricas del eslabón.

Reduciéndose al siguiente patrón de transformaciones que permiten relacionar el sistema de referencia del elemento i con respecto al sistema del elemento $i-1$:

1 – Rotación alrededor del eje z_{i-1} un ángulo θ_i

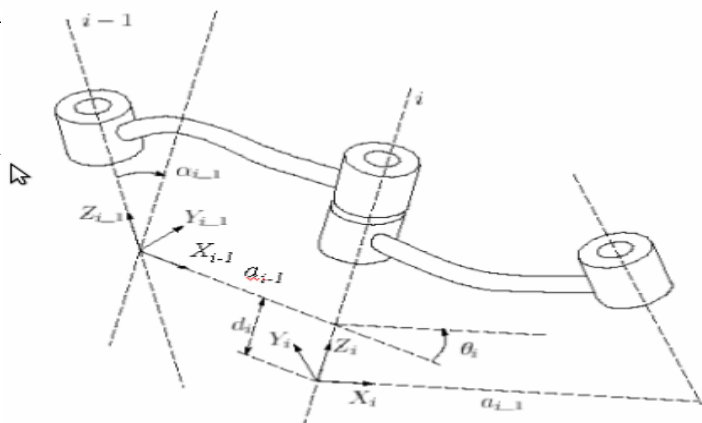
2 – Traslación a lo largo de Z_{i-1} una distancia d_i ; vector d_i (0,0, d_i)

3 - Traslación a lo largo de X_{i-1} una distancia a_i ; vector d_i (0,0, a_i)

4 – Rotación alrededor del eje X_{i-1} un ángulo α_i

Donde θ_i - α_i - a_i - d_i son los parámetros D-H del eslabón.

- **θ_i :** Es el ángulo de x_{i-1} a x_i medida sobre z_i (utilizando la regla de la mano derecha).
- **d_i :** Es la distancia de x_{i-1} a x_i medida a lo largo de z_i
- **a_i :** Es la distancia de z_i a z_{i+1} medida a lo largo de x_i
- **α_i :** Es el ángulo de z_i a z_{i+1} medida sobre x_i (utilizando la regla de la mano derecha).



$${}^{i-1}A_i = T(x, \alpha_{i-1}) T(a_{i-1}, 0, 0) T(z, \theta_i) T(0, 0, d_i)$$

$${}^{i-1}A_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_{i-1} & -s\alpha_{i-1} & 0 \\ 0 & s\alpha_{i-1} & c\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta_i & -s\theta_i & 0 & 0 \\ s\theta_i & c\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ c\alpha_{i-1}s\theta_i & c\alpha_{i-1}c\theta_i & -s\alpha_{i-1} & -d_is\alpha_{i-1} \\ s\alpha_{i-1}s\theta_i & s\alpha_{i-1}c\theta_i & c\alpha_{i-1} & d_ic\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Para que la matriz ${}^{i-1}A_i$ relacione los sistemas coordenados O_i y O_{i-1} es necesario que los sistemas coordenados se determinen mediante los siguientes pasos:

D-H 1. Numerar los eslabones comenzando con 1 (primer eslabón móvil de la cadena) y acabando con n (último eslabón móvil). Se numerara como eslabón 0 a la base fija del robot.

D-H 2. Numerar cada articulación comenzando por 1 (la correspondiente al primer grado de libertad) y acabando en n.

D-H 3. 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.

D-H 4. Para i de 0 a n-1 situar el eje z_i sobre el eje de la articulación i + 1.

D-H 5. 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 situaran de modo que formen un sistema dextrógiro con z_0 .

D-H 6. 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 $i + 1$.

D-H 7. Situar x_i en la línea normal común a z_{i-1} y z_i .

D-H 8. Situar y_i de modo que forme un sistema dextrógiro con x_i y z_i .

D-H 9. 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 .

D-H 10. 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.

D-H 11. 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.

DH 12. 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 $\{S_i\}$.

DH 13. Obtener α_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 $\{S_i\}$.

DH 14. Obtener las matrices de transformación $i-1A_i$ definidas anteriormente.

DH 15. Obtener la matriz de transformación que relaciona el sistema de la base con el del extremo del robot $T = {}^0A_1, {}^1A_2 \dots {}^{n-1}A_n$

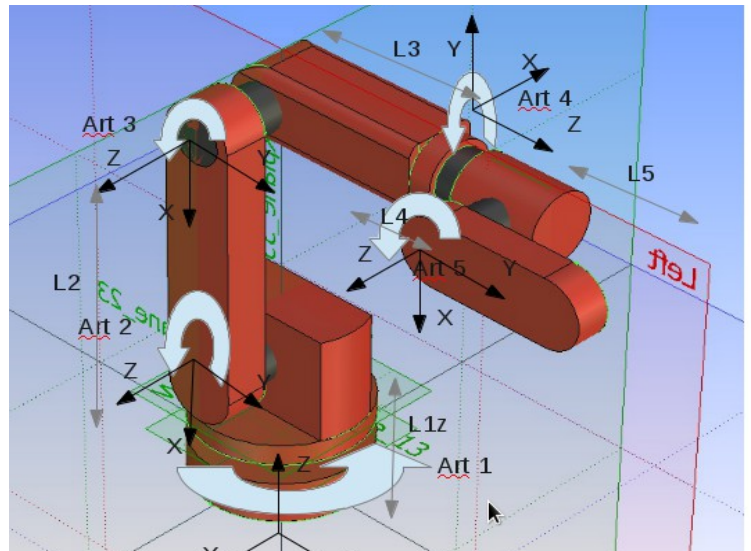
DH 16. La matriz T define la orientación (submatriz de rotación) y posición (submatriz de traslación) del extremo referido a la base en función de las n coordenadas articulares.

Modelo Cinemático de nuestro Robot:

El manipulador que analizaremos es el que tenemos debajo.

Como primera medida debemos hallar la matriz de transformación T de toda la cadena cinemática, aplicando el método matricial de Denavit Hartenberg (D.H.)

Por lo tanto como primera medida lo que haremos es esquematizar el robot indicando las articulaciones y los sistemas de referencia convenientes.

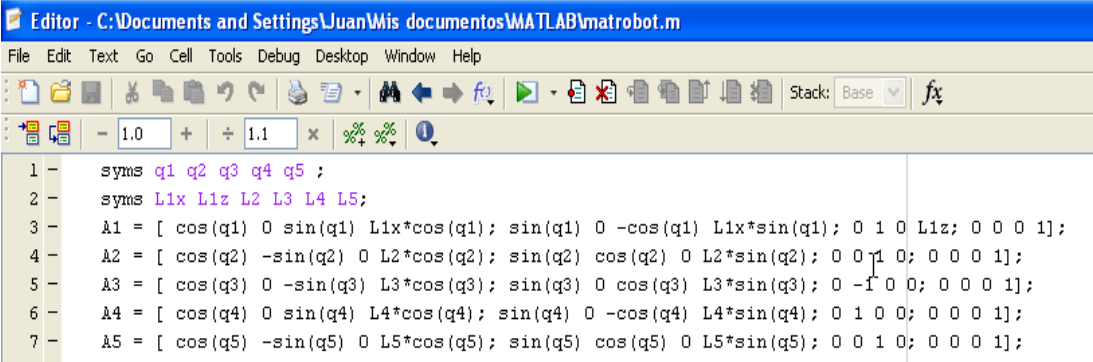


Entonces ahora procedemos a determinar los parámetros θ, d, a, α

	θ	d	a	α
Articulación 1	q1	L1z	0	90
Articulación 2	q2	0	L2	0
Articulación 3	q3	0	L3	90
Articulación 4	q4	0	L4	-90
Articulación 5	q5	0	L5	0

Con esta información procedimos a realizar un script en matlab para poder determinar la matriz de transformación homogénea T.

En primera medida definimos las matrices de transformación entre articulación y articulación.



```

1 - syms q1 q2 q3 q4 q5 ;
2 - syms L1x L1z L2 L3 L4 L5;
3 - A1 = [ cos(q1) 0 sin(q1) L1x*cos(q1); sin(q1) 0 -cos(q1) L1x*sin(q1); 0 1 0 L1z; 0 0 0 1];
4 - A2 = [ cos(q2) -sin(q2) 0 L2*cos(q2); sin(q2) cos(q2) 0 L2*sin(q2); 0 0 1 0; 0 0 0 1];
5 - A3 = [ cos(q3) 0 -sin(q3) L3*cos(q3); sin(q3) 0 cos(q3) L3*sin(q3); 0 -1 0 0; 0 0 0 1];
6 - A4 = [ cos(q4) 0 sin(q4) L4*cos(q4); sin(q4) 0 -cos(q4) L4*sin(q4); 0 1 0 0; 0 0 0 1];
7 - A5 = [ cos(q5) -sin(q5) 0 L5*cos(q5); sin(q5) cos(q5) 0 L5*sin(q5); 0 0 1 0; 0 0 0 1];

```

Luego las multiplico para hallar la matriz de transformación homogénea T

$$[T] = [{}^0A_5] = [{}^0A_1] * [{}^1A_2] * [{}^2A_3] * [{}^3A_4] * [{}^4A_5]$$

Obtenida la Matriz T realizamos la siguiente operación:

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [T] * \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

Y hacemos u=0, v=0, w=0 (en el origen) tendremos las coordenadas (x, y, z) en función de las coordenadas articulares q1, q2, q3, q4 y las dimensiones L1x, L2, L3, L4 L5 de nuestro robot.

Obteniendo en matlab los siguientes resultados:

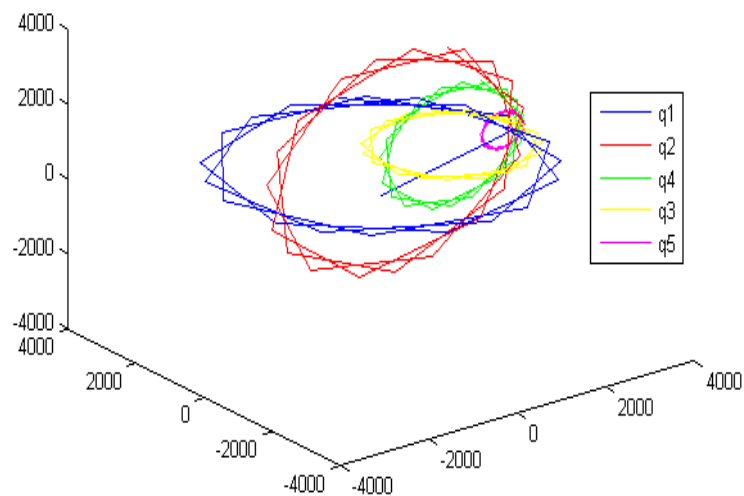
$$x = \cos(q1) * (L1x + L3 * \cos(q2 + q3) + L2 * \cos(q2)) - L4 * \sin(q1) * \sin(q4) - L5 * \cos(q5) * (\sin(q1) * \sin(q4) - \cos(q2 + q3) * \cos(q1) * \cos(q4)) - L5 * \sin(q2 + q3) * \cos(q1) * \sin(q5) + L4 * \cos(q2 + q3) * \cos(q1) * \cos(q4)$$

$$y = \sin(q1) * (L1x + L3 * \cos(q2 + q3) + L2 * \cos(q2)) + L4 * \cos(q1) * \sin(q4) + L5 * \cos(q5) * (\cos(q1) * \sin(q4) + \cos(q2 + q3) * \cos(q4) * \sin(q1)) + L4 * \cos(q2 + q3) * \cos(q4) * \sin(q1) - L5 * \sin(q2 + q3) * \sin(q1) * \sin(q5)$$

$$z = L1z + L3 * \sin(q2 + q3) + L2 * \sin(q2) + L4 * \sin(q2 + q3) * \cos(q4) + L5 * \cos(q2 + q3) * \sin(q5) + L5 * \sin(q2 + q3) * \cos(q4) * \cos(q5)$$

A partir de estos resultados realicé en **matlab** con la función **plot3d** una gráfica que muestra los distintos valores x, y, z evaluando solo q1, q2, q3, q4 y q5 por separado.

Posibles posiciones del manipulador



Implementación del modelo cinemático directo del Robot M5 a través del CodeWarrior.

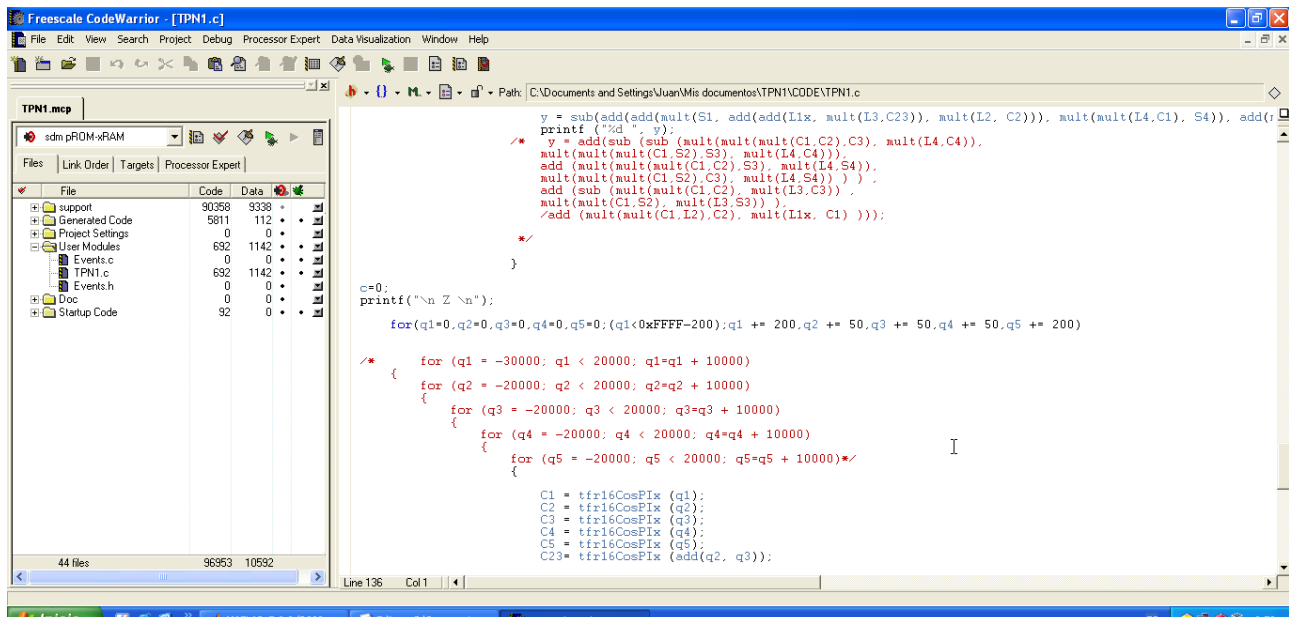
Obtenido el modelo cinemático anterior, ahora procederemos a implementarlo en el elemento que luego controlará a nuestro robot. Como observamos, la tarea de controlar un manipulador requiere de un gran procesamiento matemático y además que ese procesamiento sea muy veloz, es por ello que generalmente se utilizan DSP para tal tarea.

En nuestro ejemplo utilizamos el procesador **56F8367** de la familia 56800E de **Freescall**.

Para programarlo utilizamos el entorno de desarrollo que otorga el fabricante que es el **CodeWarrior**.

A continuación se muestra una imagen de cómo se ve el entorno de desarrollo del **CodeWarrior**.

Debajo de la imagen se presenta el código en C que implementamos en el software.



```

/** #####
**
**  Filename : TPN1.C
**
**  Project  : TPN1
**
**  Processor : 56F8367
**
**  Version   : Driver 01.12
**
**  Compiler  : Metrowerks DSP C Compiler
**
**  Date/Time : 10/05/2009, 19.19
**
**  Abstract  :
**
**      Main module.
**
**      Here is to be placed user's code.
**
**  Settings  :
**
**  Contents  :
**
**      No public methods
**
**
**  (c) Copyright UNIS, spol. s r.o. 1997-2006
**
**  UNIS, spol. s r.o.
**
**  Jundrovská 33
**
**  624 00 Brno
**
**  Czech Republic
**
**  http      : www.processorexpert.com
**
**  mail      : info@processorexpert.com
**
** #####*/
*/
MODULE TPN1 */

```

```
/* Including used modules for compiling procedure */

#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

#define a1 10

#define a2 4

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

Frac16 Homogenea[4][4];

Frac16 C1, S1, C2, S2, C3, S3, C4, S4, C5, S5, C6, S6, C23, S23, c, i, q1, q2, q3, q4, q5, L1x, L1z, L2, L3, L4, L5;
Frac16 x, y, z;

int j,k;

void main(void)
{
    for(;;)
    {
        c=0;
        q1 = 0;
        q2 = 0;
        q3 = 0;
```

```

q4 = 0;
q5 = 0;
L1x = 320;
L1z = 780;
L2 = 1280;
L3 = 0;
L4 = 1142;
L5 = 500;

```

```
printf("\n X \n");
```

```
for(q1=0,q2=0,q3=0,q4=0,q5=0;(q1<0xFFFF-200);q1 += 200,q2 += 50, q3 += 50,q4 += 50,q5 += 200)
```

```
{
```

```

C1 = tfr16CosPlx (q1);
C2 = tfr16CosPlx (q2);
C3 = tfr16CosPlx (q3);
C4 = tfr16CosPlx (q4);
C5 = tfr16CosPlx (q5);
C23= tfr16CosPlx (add(q2, q3));
S1 = tfr16SinPlx (q1);
S2 = tfr16SinPlx (q2);
S3 = tfr16SinPlx (q3);
S4 = tfr16SinPlx (q4);
S5 = tfr16SinPlx (q5);
S23 = tfr16SinPlx (add(q2, q3));

```

```

x=sub(add(mult(C1,add(add(L1x,mult(L3,C23)),mult(L2,C2))),mult(mult(L4,C23),mult(C1,C4))),
add(add(mult(mult(L4,S1), S4), mult(L5, mult(C5, sub(mult(S1, S4), mult(mult(C23, C1), C4))))), mult(mult(L5,
S23), mult(C1, S5))));
printf ("%d ", x);

```

```
}
```

```

c=0;
q1 = 0;
q2 = 0;
q3 = 0;
q4 = 0;
q5 = 0;

```

```

L1x = 320;
L1z = 780;
L2 = 1280;
L3 = 1142;
L4 = 200;
L5 = 0;
printf("\n Y \n");

```

```
for(q1=0,q2=0,q3=0,q4=0,q5=0;(q1<0xFFFF-200);q1 += 200,q2 += 50,q3 += 50,q4 += 50,q5 += 200)
```

```
{
```

```

C1 = tfr16CosPlx (q1);
C2 = tfr16CosPlx (q2);
C3 = tfr16CosPlx (q3);
C4 = tfr16CosPlx (q4);
C5 = tfr16CosPlx (q5);
C23= tfr16CosPlx (add(q2, q3));

S1 = tfr16SinPlx (q1);
S2 = tfr16SinPlx (q2);
S3 = tfr16SinPlx (q3);
S4 = tfr16SinPlx (q4);
S5 = tfr16SinPlx (q5);
S23 = tfr16SinPlx (add(q2, q3));

```

```

y = sub(add(add(mult(S1, add(add(L1x, mult(L3,C23)), mult(L2, C2))), mult(mult(L4,C1), S4)), add(mult(L5,
mult(C5, add(mult(C1, S4), mult(mult(C23, C4), S4)))), mult(mult(L4, C23), mult(C4, S1))), mult(mult(L5, S23),

```

```

        mult(S1, S4));
        printf ("%d ", y);

    }
    c=0;
    printf("\n Z \n");

for(q1=0,q2=0,q3=0,q4=0,q5=0;(q1<0xFFFF-200);q1 += 200,q2 += 50,q3 += 50,q4 += 50,q5 += 200)
{
    C1 = tfr16CosPlx (q1);
    C2 = tfr16CosPlx (q2);
    C3 = tfr16CosPlx (q3);
    C4 = tfr16CosPlx (q4);
    C5 = tfr16CosPlx (q5);
    C23= tfr16CosPlx (add(q2, q3));
    S1 = tfr16SinPlx (q1);
    S2 = tfr16SinPlx (q2);
    S3 = tfr16SinPlx (q3);
    S4 = tfr16SinPlx (q4);
    S5 = tfr16SinPlx (q5);
    S23 = tfr16SinPlx (add(q2, q3));

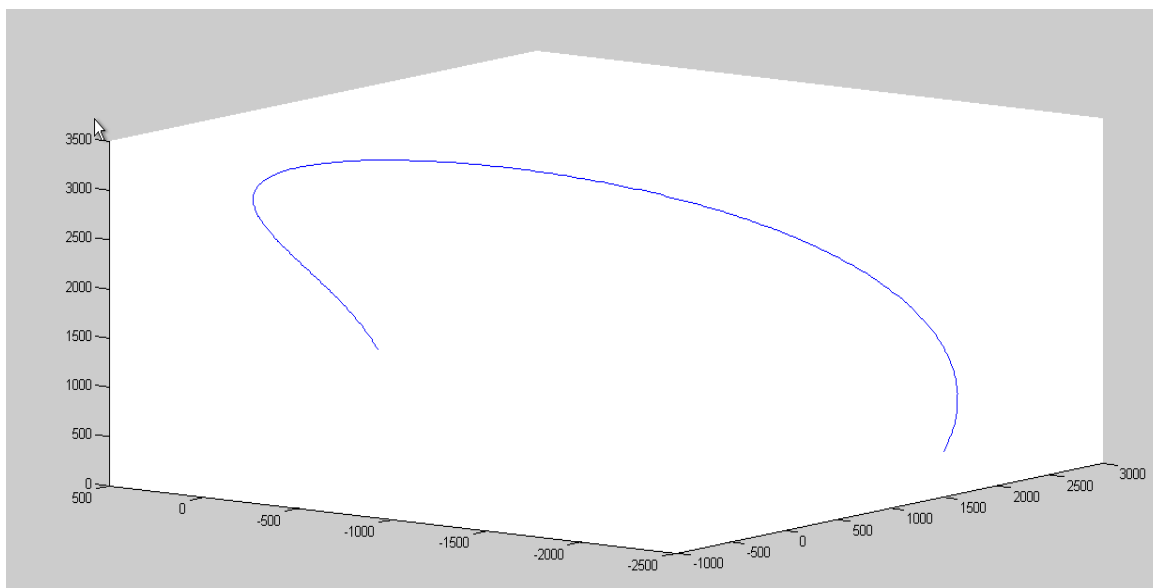
    z = add(add (add (L1z, mult(L3,S23)), mult(L2,S2)), add(add(mult(mult(L4, S23), C4),
    mult(mult(L5, C23), S5)),mult(mult(L5, S23), mult(C4, C5))));
    printf ("%d ", z);

}
}
}

```

Como podemos observar en el código utilizamos la función printf, que gracias a las herramientas de DEBUG del software podemos ver la trayectoria que realiza nuestro robot según se modifican los valores de q1, q2, q3, q4 y q5.

A continuación le pasamos esos parámetros al Matlab que con la función de plot3d los grafica.



Simulación en Matlab a través del Robotics Toolbox

Para llevar a cabo la siguiente simulación es necesario disponer de la herramienta llamada Robotics Toolbox for Matlab (release 8) que se debe agregar a nuestro programa Matlab; este paquete puede descargarse de la página www.petercorke.com.

A continuación se muestra el script para la simulación con el toolbox.

Para tal fin se muestra nuevamente la tabla que habíamos con realizado gracias al algoritmo de Denavit Hartenberg.

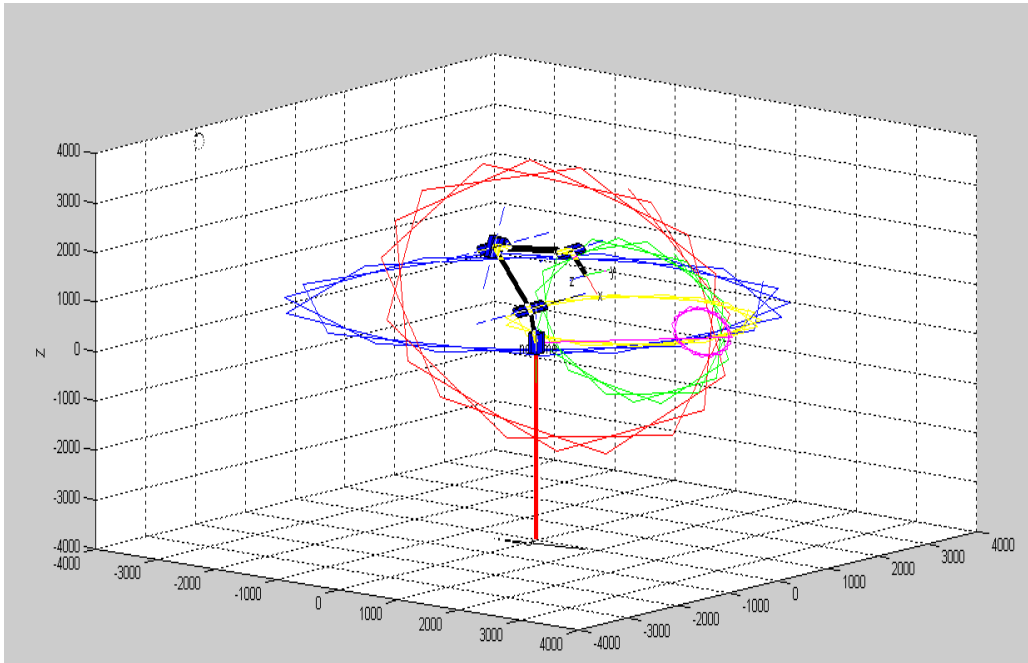
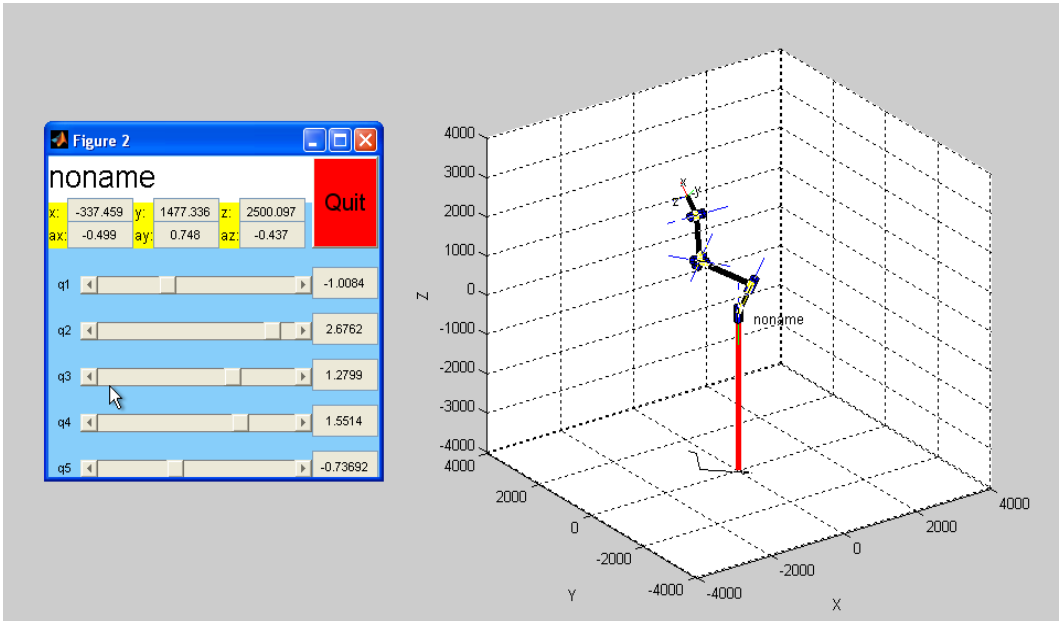
	θ	d	a	α
Articulación 1	q1	L1x	0	90
Articulación 2	q2	0	L2	0
Articulación 3	q3	0	L3	90
Articulación 4	q4	0	L4	-90
Articulación 5	q5	0	L5	0

```

133 - art1 = link([90, L1x, 0, L1z]);
134 - art2 = link([0, L2, 0, 0]);
135 - art3 = link([-90, L3, 0, 0]);
136 - art4 = link([90, L4, 0, 0]);
137 - art5 = link([0, L5, 0, 0]);
138
139
140 - mi_robot = robot({art1, art2, art3, art4, art5});
141
142 - subplot(1,2,2)
143 - plot (mi_robot,[0 0 0 0 0])
144 - drivebot(mi_robot);
145

```

Se obtiene la siguientes simulaciones:



Dinámica del Robot

Marco teórico

Introducción a la dinámica de un robot

El modelo dinámico de un robot nos permite conocer la relación existente entre el movimiento y las fuerzas actuantes que provocan dicho movimiento.

A diferencia del modelo cinemático que solo representa el movimiento con respecto a un sistema de referencia, el modelo dinámico tiene en cuenta las fuerzas y pares aplicadas en las articulaciones y además los parámetros dimensionales del robot como la longitud, masa e inercia de sus elementos. Desde un punto de vista matemático, se relaciona la locación del robot (que está definida por sus variables articulares o por las coordenadas de localización de su extremo), y sus derivadas: velocidad y aceleración.

Un punto a tener en cuenta, es que la complejidad del modelo dinámico se acrecienta en gran medida con el aumento de grados de libertad (GDL). El estudio dinámico del robot tiene como resultado un conjunto de ecuaciones matemáticas que describen la conducta dinámica del manipulador.

Tales ecuaciones de movimiento son útiles para simulación por PC, diseño de ecuaciones de control apropiadas para el robot y la evaluación del diseño y estructura cinemática del robot. Debido al ya comentado aumento de complejidad a medida que crece el número de GDL no siempre es posible hallar una solución cerrada, es decir, un conjunto de ecuaciones diferenciales que al ser integradas nos den como resultado las fuerzas y torques a ser aplicadas para obtener un determinado desplazamiento con una cierta velocidad y aceleración, es decir, un único resultado. Así, el modelo dinámico se debe resolver mediante cálculo numérico e iterativo (por Ej. Uso de series convergentes). Cada eslabón de la cadena se trata como rígido y de masa concentrada, se usa el centro de gravedad. Existen 2 formas básicas de encarar el problema del modelo dinámico:

Modelo Dinámico Directo: Expresa la evolución temporal de las coordenadas articulares del robot en función de las fuerzas y pares que intervienen.

Modelo Dinámico Inverso: Expresa las fuerzas y pares que intervienen en función de la evolución de las coordenadas articulares y sus derivadas.

Al hacer el análisis, se debe tener en cuenta las siguientes fuerzas intervinientes: Inercia, gravedad, coriolis y centrípeta.

Parámetros importantes en el análisis dinámico

Momento Angular

El momento angular es una magnitud física importante porque en muchos sistemas

físicos constituye una magnitud conservada, a la cual bajo ciertas condiciones sobre las fuerzas es posible asociarle una ley de conservación. Se define como:

$$\mathbf{L}_O = \int_V \rho(\mathbf{r}_O \times \mathbf{v}_O) dV$$

Momento de Inercia

Es una medida de la inercia rotacional. Es análogo a la masa de un cuerpo en un sistema traslacional y permite relacionar el momento aplicado a dicho cuerpo con la aceleración angular que va a desarrollar.

$$\mathbf{L} = \mathbf{r} \times \mathbf{p} = \mathbf{r} \times m\mathbf{v}$$

Tensor de Inercia

Es una entidad que caracteriza la inercia rotacional de un sólido rígido. A partir del tensor de inercia, cuyas componentes se calculan:

$$\begin{cases} I_{xx} = \int_M d_x^2 dm = \int_V \rho(y^2 + z^2) dx dy dz \\ I_{yy} = \int_M d_y^2 dm = \int_V \rho(z^2 + x^2) dx dy dz \\ I_{zz} = \int_M d_z^2 dm = \int_V \rho(x^2 + y^2) dx dy dz \end{cases}$$

se puede calcular la energía de rotación:

$$E_{rot} = \frac{1}{2} (\Omega_x \quad \Omega_y \quad \Omega_z) \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix} \begin{pmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} = \frac{1}{2} \sum_j \sum_k I_{jk} \Omega_j \Omega_k$$

Fuerza Centrípeta

La fuerza centrípeta aparece para producir una aceleración a un cuerpo que está girando. Su expresión es la siguiente:

$$\mathbf{a} = -\frac{v^2}{r} \left(\frac{\mathbf{r}}{r} \right) = -\frac{v^2}{r} \hat{\mathbf{u}}_r = -\omega^2 \mathbf{r}$$

Fuerza de Coriolis

La fuerza de Coriolis es una fuerza ficticia que aparece cuando un cuerpo está en movimiento con respecto a un sistema en rotación y se describe su movimiento en ese

referencial. La fuerza de Coriolis es diferente de la fuerza centrífuga. La fuerza de Coriolis siempre es perpendicular a la dirección del eje de rotación del sistema y a la dirección del movimiento del cuerpo vista desde el sistema en rotación.

$$\vec{F}_c = 2m (\vec{v} \times \vec{\omega})$$

Ecuación del Torque

Esta ecuación es la más importante, y es la que permite relacionar todo lo anterior. Y su resolución la haremos a través del toolbox Corke que calcula los valores a través de un método llamado Recursive Newton-Euler.

$$\tau = D\ddot{q} + H + C$$

Existen distintos métodos para resolver el problema:

- Lagrange-Euler
- Newton-Euler
- Variables de estado
- Kane

Método de Newton – Euler

La formulación de Newton-Euler utiliza las ecuaciones de equilibrio de fuerzas y torques (pares):

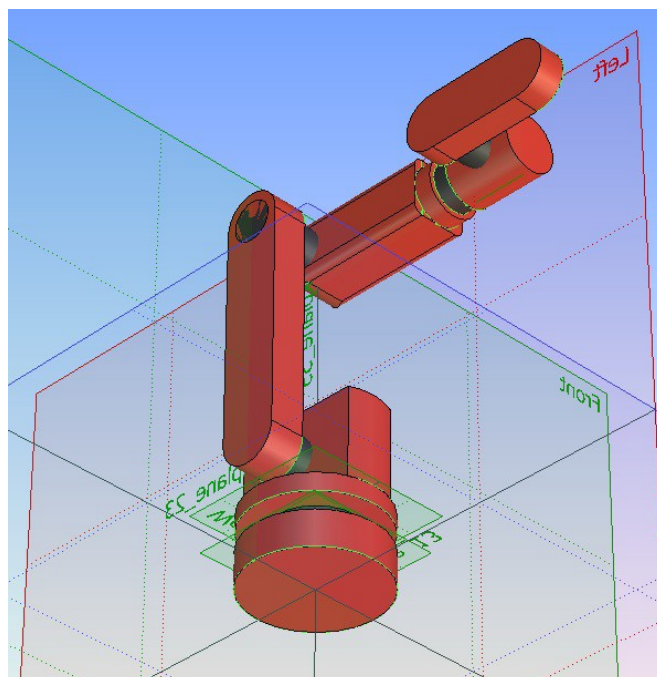
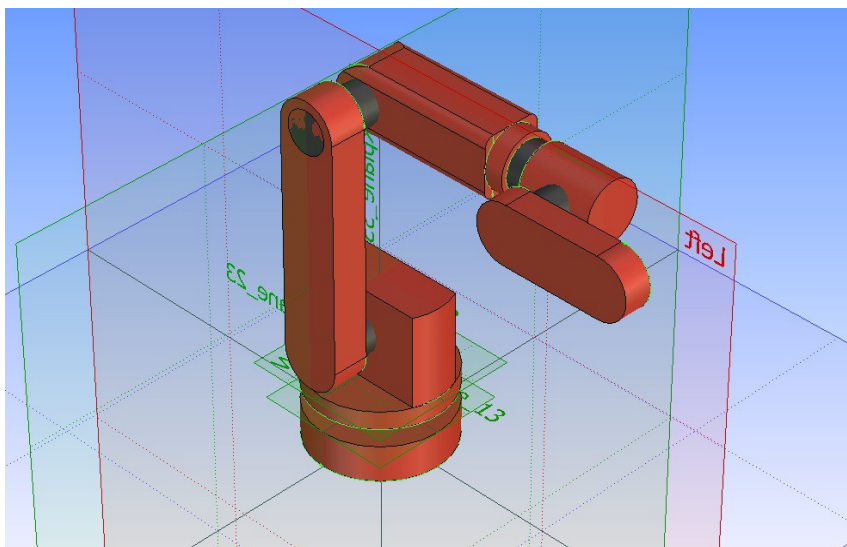
$$\sum F = m \dot{v}$$

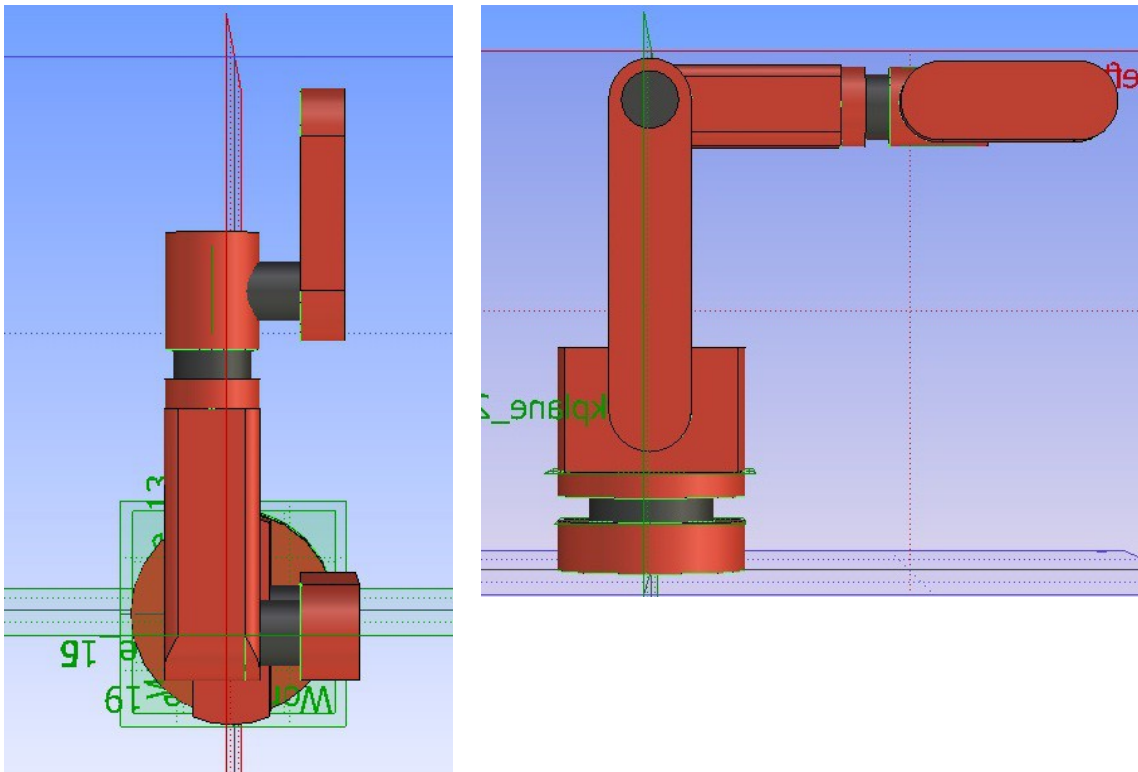
$$\sum T = I \dot{\omega} + \omega \times (I \omega)$$

Obtención de Parámetros utilizando el programa T-FLEX

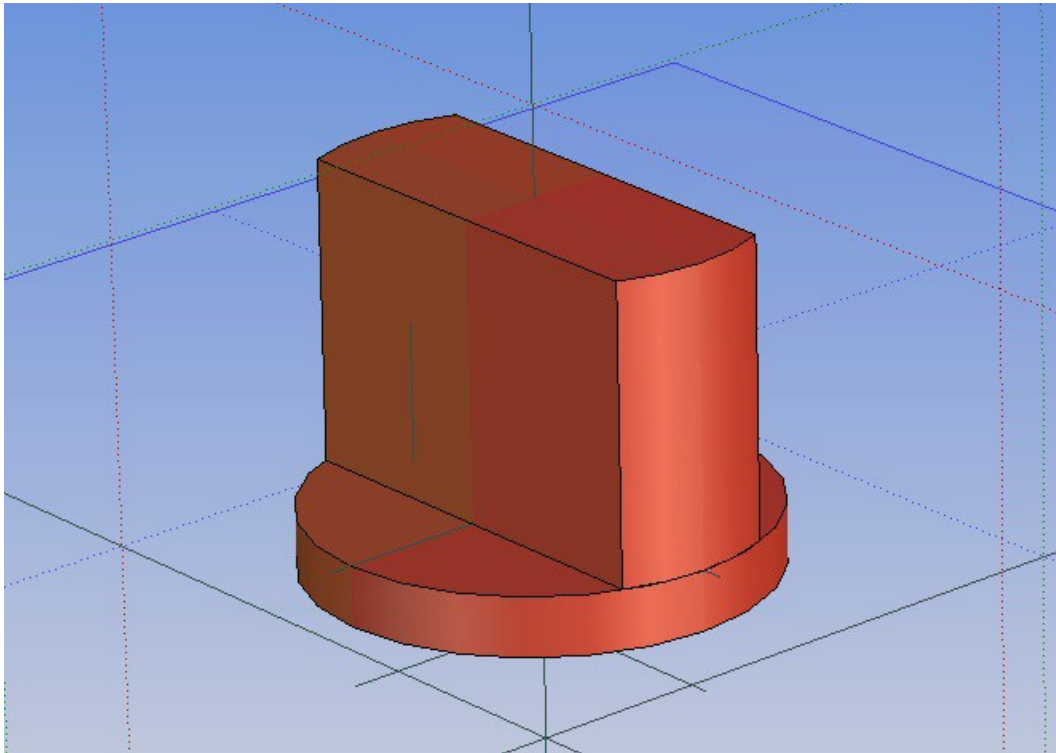
El **T-FLEX** Parametric CAD es un programa que permite hacer un modelado de nuestro robot, este software une funcionalidades de modelado paramétrico 3D con el conjunto de herramientas de producción y dibujo paramétrico. Gracias a este software vamos a obtener lo que nos interesa que es la masa de las articulaciones del robot.

Como primer paso se realizó el diseño del robot muy aproximado y simplificado a modo de tener solamente un concepto general de todas las partes. Dicho robot cuenta con 5 grados de libertad como se puede ver en las siguientes imágenes.

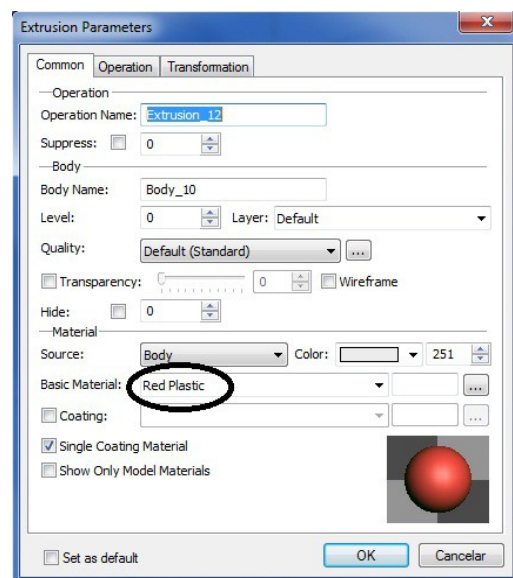
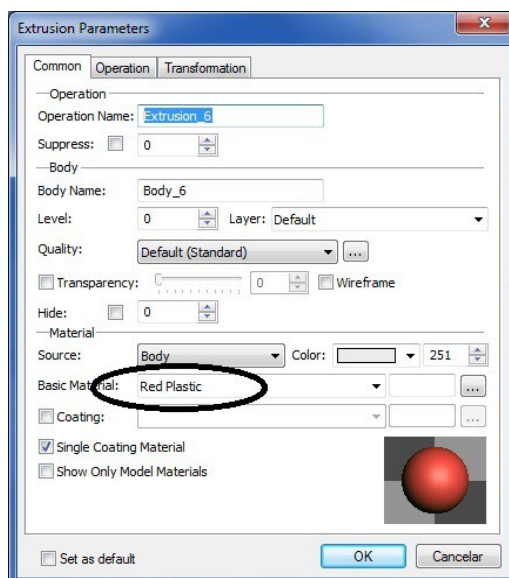


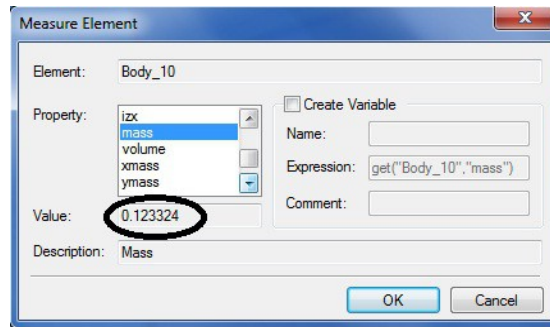
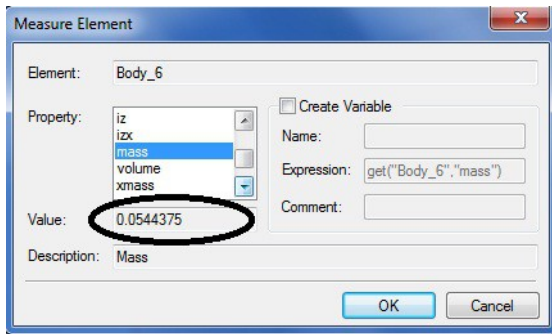


En segunda instancia pasamos a medir la masa de las articulaciones. Dicha masa va a depender de las dimensiones y del tipo de material, para nuestro caso usamos plástico. En las siguientes imágenes podemos ver las articulaciones, las ventanas BODY PROPERTIES con el material usado y las ventanas MEASURE ELEMENT donde podemos observar la masa en Kg.

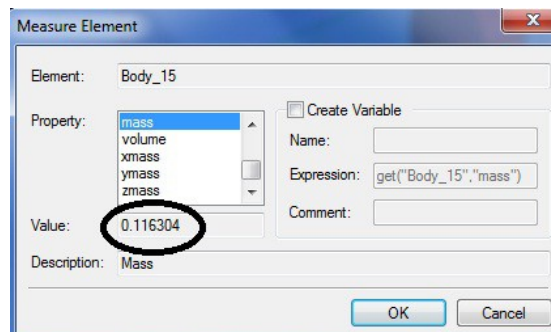
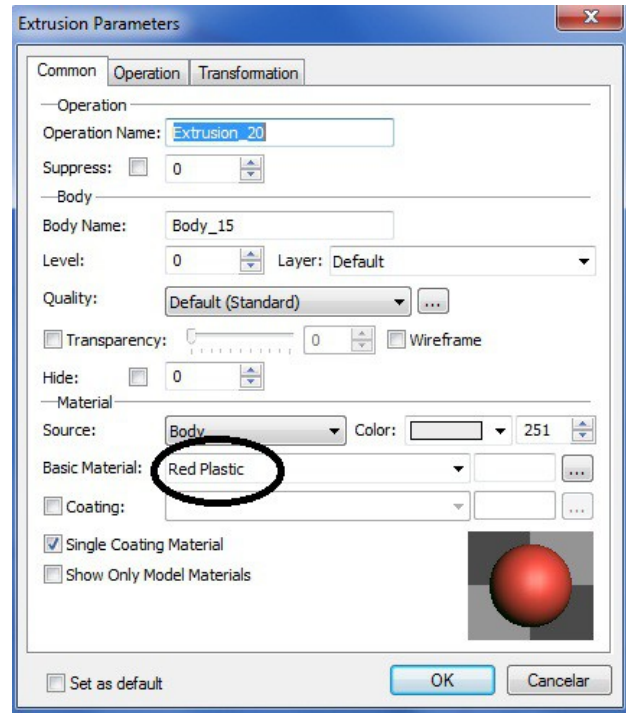
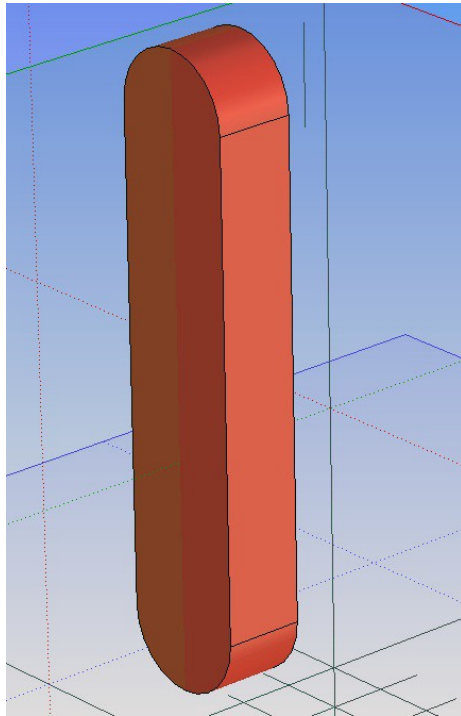
Articulación 1:**Articulación 1**

Como esta articulación se construyo con dos piezas tenemos 2 ventanas de BODY PROPERTIES y 2 de MEASURE ELEMENT.

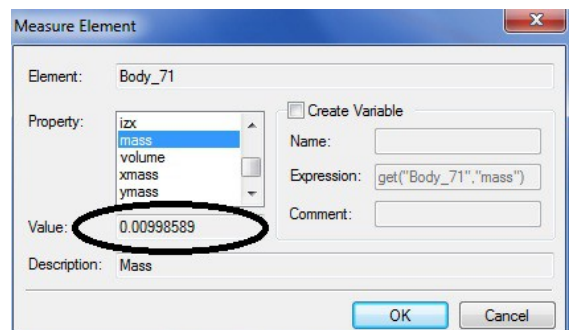
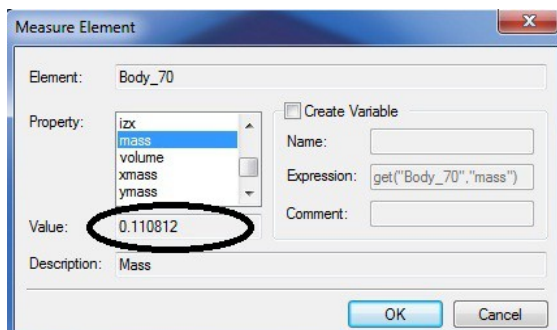
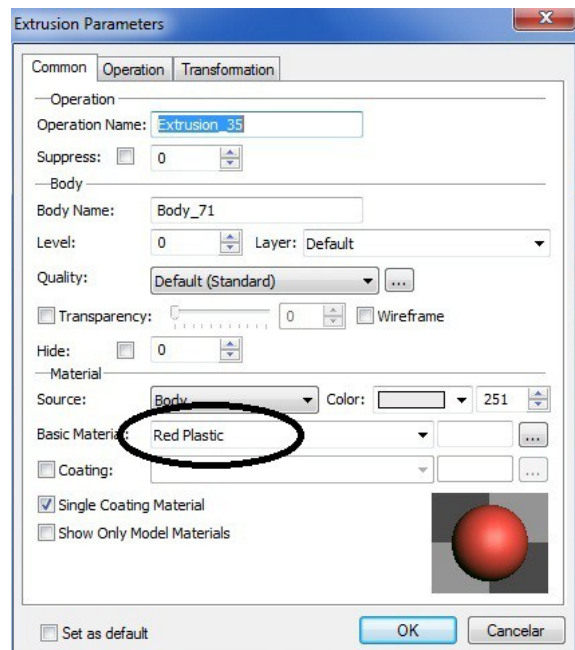
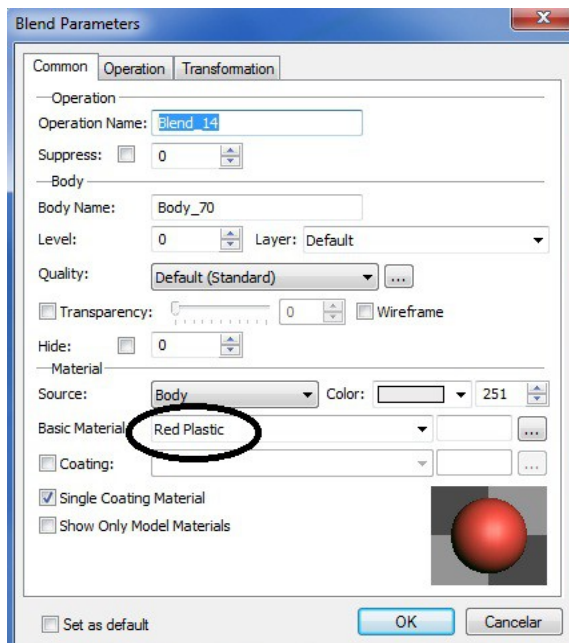
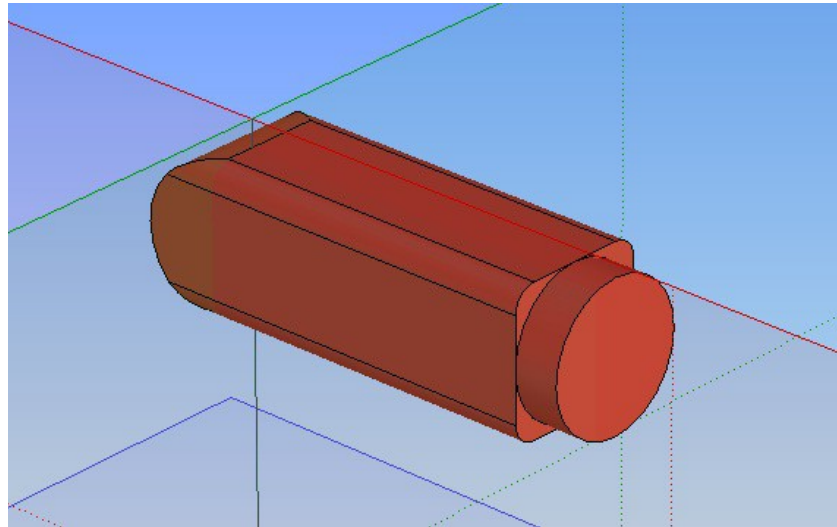




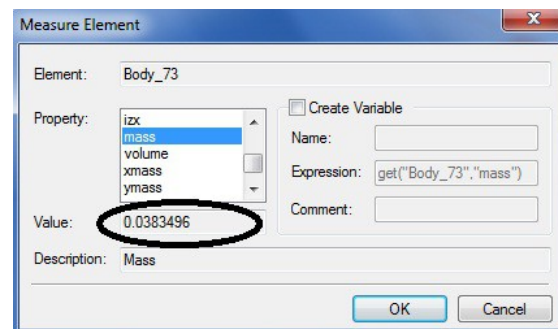
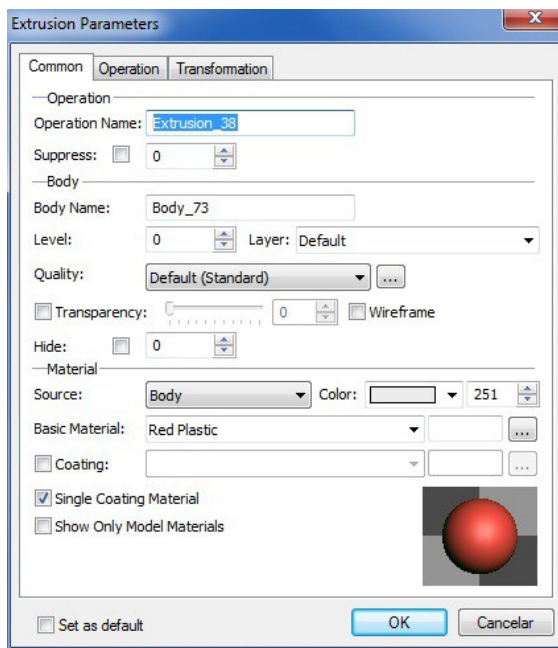
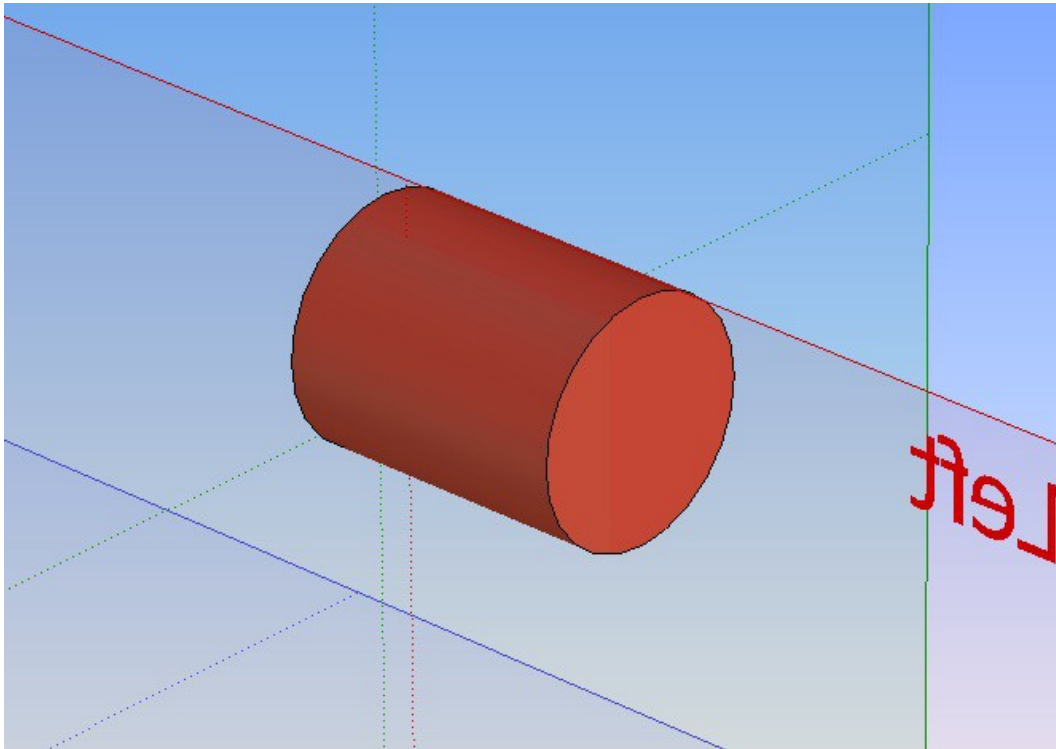
Podemos decir que la masa total de la articulación 1 es **M1= 0.1777 Kg.**

Articulación 2:

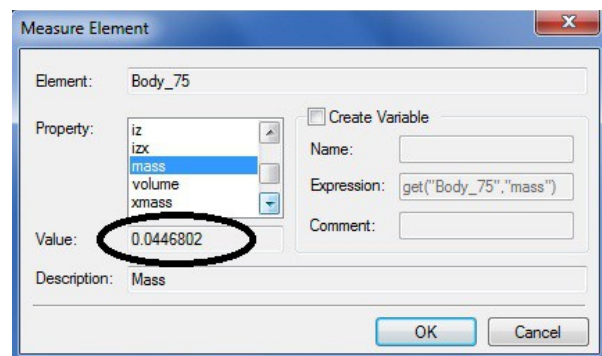
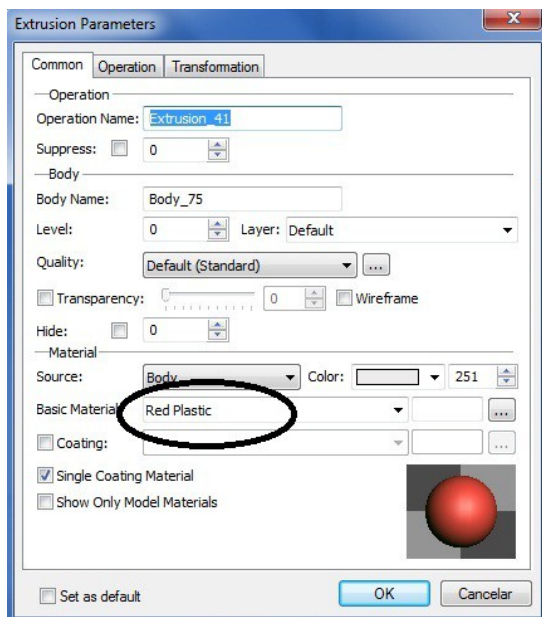
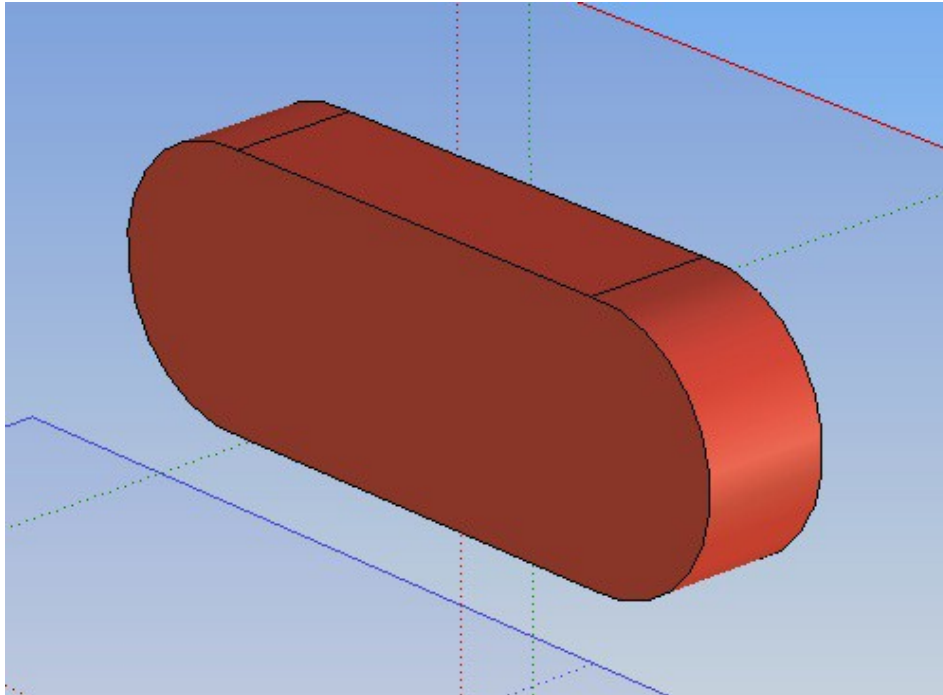
En este caso tenemos que la masa de la articulación 2 es $M_2 = 0.1163 \text{ Kg}$.

Articulación 3:

En este caso la articulación 3, esta formada por 2 piezas y se muestran las propiedades de ambas piezas. Tenemos que la masa de la articulación 3 es $M_3 = 0.1207 \text{ Kg}$.

Articulación 4:

Vemos que la masa de la articulación 4 es $M_4 = 0.0383 \text{ Kg}$.

Articulación 5:

Vemos que la masa de la articulación 5 es $M_5 = 0.0446 \text{ Kg}$.

Obtención del Modelo dinámico

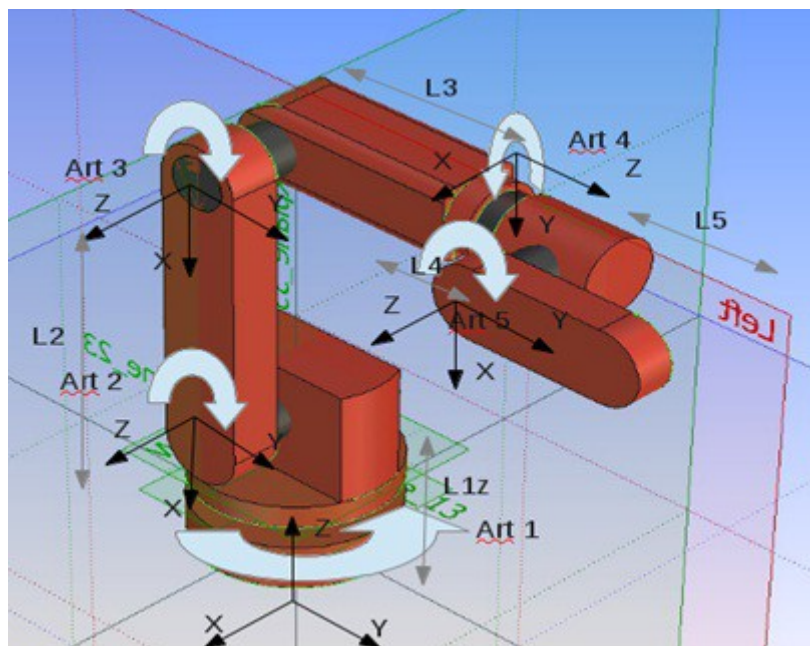
Se obtendrá el modelo dinámico a través del método Newton Euler utilizando el toolbox Hemero desarrollado por el señor Aníbal Ollero Baturone, quien explica su funcionamiento en el libro Robótica: “Manipuladores y robots móviles”.

Utilizando Matlab a través de la función rne permite obtener el modelo dinámico. Para ello simplemente hay que pasarle una matriz de parámetros dyn (dinámica).

Para hallar la matriz dyn necesitamos obtener algunos parámetros como ser las masas de las articulaciones y los parámetros de D.H que se calcularon en el trabajo práctico número 1.

Parámetros Denavit-Hartenberg (D.H)

Estos ya fueron calculados en el trabajo práctico número 1 que ahora pasamos solo a presentarlos. En base al siguiente diagrama se obtiene los siguientes parámetros D.H:



Entonces ahora procedemos a determinar los parámetros θ, d, a, α

	θ	d	a	α
Articulación 1	q1	L1	0	90
Articulación 2	q2	0	L2	0
Articulación	q3	0	L3	90

3				
Articulación 4	q4	0	L4	-90
Articulación 5	q5	0	L5	0

Sigma

Se necesita saber también otro parámetro que no está en la tabla anterior que se denomina sigma(i) este indicará el tipo de articulación (será 0 si es de rotación y 1 si por el contrario es prismática (traslacional) el subíndice i indicara el numero de articulación para nuestro caso el valor de i estará entre 1 a 4 (por ser cuatro grados de libertad) .

	Sigma
Articulación 1	0
Articulación 2	0
Articulación 3	0
Articulación 4	0
Articulación 5	0

Masas

	Masas (Kg)
Articulación 1	0.1777
Articulación 2	0.1163
Articulación 3	0.1207
Articulación 4	0.0383
Articulación 5	0.0446

Longitudes

	Longitudes (cm)
Articulación 1	60.4
Articulación 2	155.3
Articulación 3	102.7
Articulación 4	40
Articulación 5	86.3

Centros de masas

Introducción:

El centro de gravedad o centro de masas de un sistema continuo es el punto geométrico definido como:

$$\vec{C}_{CM} = \frac{\int \vec{r} dm}{\int dm} = \frac{\int \vec{r} dm}{M}$$

En mecánica del sólido rígido, el centro de masa se usa porque tomando un sistema de coordenadas centrado en el, la energía cinética total K puede expresarse como:

$$K = \frac{1}{2} MV^2 + K_{rot}$$

Siendo “m” la masa total del cuerpo, “v” la velocidad de traslación del centro de masas

y Krot la energía de rotación del cuerpo, expresable en términos de la velocidad angular y el tensor de inercia.

Para facilitar la resolución del problema se supondrá que las masas M1, M2, M3, M4 y M5 están concentradas en los extremos de dichas articulaciones de nuestro robot.

$$\mathbf{lcm1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{lcm2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{lcm3} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{lcm4} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{lcm5} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Determinación de la matriz dyn

Para resolver las ecuaciones se utiliza una matriz llamada **dyn**. Dicha matriz posee nx20 elementos, en donde n es la cantidad de articulaciones del robot a estudiar.

Las columnas de dicha matriz **dyn** se conforman con los siguientes datos:

Columna 1: alpha (i-1): Parámetros de Denavit-Hartenberg

Columna 2: a (i-1)

Columna 3: theta (i)

Columna 4: d (i)

Columna 5: sigma (i): Tipo de articulación; 0 si es de rotación y 1 si es prismática

Columna 6: masa: Masa del enlace i

Columna 7: rx: Centro de masas del enlace respecto al cuadro de referencia de dicho enlace

Columna 8: ry

Columna 9: rz

Columna 10: Ixx: Elementos del tensor de inercia referido al centro de masas del enlace

Columna 11: Iyy

Columna 12: Izz

Columna 13: Ixy

Columna 14: Iyz

Columna 15: Ixz

Columna 16: Jm: Inercia de la armadura

Columna 17: G: Velocidad de la articulación / velocidad del enlace

Columna 18: B: Fricción viscosa, referida al motor

Columna 19: Tc+: Fricción de Coulomb (rotación positiva), referida al motor

Columna 20: Tc-: Fricción de Coulomb (rotación negativa), referida al motor

Así pues para un robot con n enlaces, la matriz DYN tendría dimensiones nx20. Los parámetros de la matriz podrán tener las unidades que se deseen, siempre que se sea coherente en el uso de dichas unidades. Es decir que si se introducen las masas en Kg y los centros de masas en metros, al escribir el tensor de inercia se deberá expresar en Kg m². En este caso, utilizaremos Kg y metros. En nuestro caso, la matriz **dyn** de **5X20** correspondiente sería la siguiente:

α	a	Θ	d	σ	$masa$	r_x	r_y	r_z	I_{xx}	I_{yy}	I_{zz}	I_{xy}	I_{yz}	I_{xz}	J_m	G	B	T_{c+}	T_{c-}
90	0	$t1$	60	0	0.177	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1.55	$t2$	0	0	0.116	0	0	0	0	0	0	0	0	0	0	1	0	0	0
90	1.03	$t3$	0	0	0.120	0	0	0	0	0	0	0	0	0	0	1	0	0	0
-90	0.40	$t4$	0	0	0.038	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0.86	$t5$	0	0	0.044	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Simulación en Matlab utilizando el toolbox Hemero:

El objetivo es obtener las ecuaciones de cupla, torque o par necesarios para los cinco motores brushless que operan las cinco articulaciones de nuestro robot (por considerar cinco grados de libertad) a fin de saber que cupla debo aplicarle a cada motor a fin de obtener una respuesta satisfactoria al requerimiento solicitado a nuestro robot.

Obtenidos los parámetros necesarios con el método de D.H y con el programa T-FLEX y ya habiendo calculado la matrix dyn estamos en condiciones de calcular el modelo dinámico a través de la herramienta matemática Matlab utilizado el **toolbox Hemero** nombrado anteriormente. A continuación se muestra el código implementado en Matlab.

```
>> syms t1 t2 t3 t4 t5 real; % Variables articulares tita1, tita2, tita3, tita4, tita5
```

```
>> syms td1 td2 td3 td4 td5 real; % Velocidades articulares tita'1, tita'2, tita'3, tita'4, tita'5
```

```
>> syms tdd1 tdd2 tdd3 tdd4 tdd5 real; % Aceleraciones articulares tita"1, tita"2, tita"3, tita"4, tita"5
```

```
>> syms g real; % Aceleracion de la gravedad
```

```
>> dyn=[90 0 q1 60 0 0.177 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        0 1.55 q2 0 0 0.166 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        90 1.03 q3 0 0 0.120 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        -90 0.40 q4 0 0 0.038 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        0 0.86 q5 0 0 0.044 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0];
```

```
>> q=[t1 t2 t3 t4 t5]; % Vector de variables articulares
```

```
>> qd=[td1 td2 td3 td4 td5]; % Vector de velocidades articulares
```

```
>> qdd=[tdd1 tdd2 tdd3 tdd4 tdd5]; % Vector de aceleraciones articulares
```

```
>> grav=[0 -g 0]; % Vector de aceleracion de la gravedad
```

```
>> tau = rne(dyn, q, qd, qdd, grav)
```

```
>> simple(tau)
```

```
>> % Escribimos la matriz dinamica con los parametros dinamicos del manipulador.
>>
>> syms t1 t2 t3 t4 t5 real; % Variables articulares tita1, tita2, tita3, tita4, tita5
>> syms td1 td2 td3 td4 td5 real; % Velocidades articulares tita'1, tita'2, tita'3, tita'4, tita'5
>> syms tdd1 tdd2 tdd3 tdd4 tdd5 real; % Aceleraciones articulares tita''1, tita''2, tita''3, tita''4, tita''5
>> syms g real; % Aceleracion de la gravedad
>>
>> dyn=[90 0 q1 60 0 0.177 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        0 1.55 q2 0 0 0.166 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        90 1.03 q3 0 0 0.120 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        -90 0.40 q4 0 0 0.038 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0;
        0 0.86 q5 0 0 0.044 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0];
>>
>> q=[t1 t2 t3 t4 t5]; % Vector de variables articulares
>> qd=[td1 td2 td3 td4 td5]; % Vector de velocidades articulares
>> qdd=[tdd1 tdd2 tdd3 tdd4 tdd5]; % Vector de aceleraciones articulares
>>
>> grav=[0 -g 0]; % Vector de aceleracion de la gravedad
>>
>> tau = rne(dyn, q, qd, qdd, grav)

tau =

[ cos(90)*((2*sin(90)*((473*sin(t4)*(sin(90)*sin(t3)*(tdd1 + tdd2) + td3*sin(90)*cos(t3)*(td1 + td2)))/12500 + (11*(
```

Resultados obtenidos con Matlab

Expresión del torque (T1) para la articulación numero 1:

$$\begin{aligned}
 \mathbf{T1} = & (5654821*td1)/5000000 + (1234221*td2)/5000000 + (41*td1*\cos(t3)^2)/3125 + (41*td2*\cos(t3)^2)/3125 \\
 & - (20339*td1*\cos(t4)^2)/625000 - (20339*td2*\cos(t4)^2)/625000 - (322493*td2^2*\sin(t2))/1000000 - \\
 & (48719*td4^2*\sin(t4))/1250000 + (41*td3*\cos(90))/3125 - (12139*td1*\cos(90)^2)/625000 - \\
 & (12139*td2*\cos(90)^2)/625000 + (20339*td1*\cos(90)^4)/625000 + (20339*td2*\cos(90)^4)/625000 + \\
 & (20339*td3*\cos(90)^3)/625000 + (20339*td4*\cos(90)^2)/625000 + (322493*td1*\cos(t2))/500000 + \\
 & (322493*td2*\cos(t2))/1000000 + (4223*td1*\cos(t3))/62500 + (4223*td2*\cos(t3))/62500 + \\
 & (20339*td4*\cos(t3))/625000 + (48719*td4*\cos(t4))/1250000 + (713*g*\cos(90)*\cos(t1))/1250 + \\
 & (20727*td3*\cos(90)*\cos(t3))/312500 + (473*td3*\cos(90)*\cos(t4))/15625 - (322493*td1*td2*\sin(t2))/500000 - \\
 & (4223*td1*td3*\sin(t3))/62500 - (4223*td2*td3*\sin(t3))/62500 - (20339*td3*td4*\sin(t3))/312500 + \\
 & (20339*td1*\cos(t3)^2*\cos(t4)^2)/625000 + (20339*td2*\cos(t3)^2*\cos(t4)^2)/625000 + \\
 & (20339*td1*\cos(90)^2*\cos(t3))/312500 + (20339*td2*\cos(90)^2*\cos(t3))/312500 - \\
 & (20339*td1*\cos(90)^4*\cos(t3))/312500 - (20339*td2*\cos(90)^4*\cos(t3))/312500 - \\
 & (20339*td3*\cos(90)^3*\cos(t3))/625000 - (20339*td4*\cos(90)^2*\cos(t3))/625000 + \\
 & (473*td1*\cos(90)^2*\cos(t4))/15625 + (473*td2*\cos(90)^2*\cos(t4))/15625 + \\
 & (20339*td3*\cos(90)*\cos(t4)^2)/625000 - (29799*td4*\cos(90)^2*\cos(t4))/1250000 - \\
 & (20727*td3^2*\cos(90)*\sin(t3))/312500 + (1271*td1*\cos(t2)*\cos(t3))/12500 + (1271*td2*\cos(t2)*\cos(t3))/25000 + \\
 & (14663*td4*\cos(t2)*\cos(t4))/250000 + (48719*td1*\cos(t3)*\cos(t4))/625000 + (48719*td2*\cos(t3)*\cos(t4))/625000 \\
 & + (473*td4*\cos(t3)*\cos(t4))/31250 - (41*td1*td3*\sin(2*t3))/3125 - (41*td2*td3*\sin(2*t3))/3125 + \\
 & (20339*td1*td4*\sin(2*t4))/625000 + (20339*td2*td4*\sin(2*t4))/625000 - (1271*td3*\sin(t2)*\sin(t3))/25000 - \\
 & (14663*td1*\sin(t2)*\sin(t4))/125000 - (14663*td2*\sin(t2)*\sin(t4))/250000 + (473*td3*\sin(t3)*\sin(t4))/31250 \\
 & + 1271*td1*td2*\cos(t3)*\sin(t2))/12500 - (1271*td1*td3*\cos(t2)*\sin(t3))/12500 - (1271*td2*td3*\cos(t2)*\sin(t3))/12500 - \\
 & (14663*td1*td2*\cos(t2)*\sin(t4))/125000 - (14663*td1*td4*\cos(t4)*\sin(t2))/125000 - \\
 & (14663*td2*td4*\cos(t4)*\sin(t2))/125000 - (48719*td1*td3*\cos(t4)*\sin(t3))/625000 - \\
 & (48719*td1*td4*\cos(t3)*\sin(t4))/625000 - (48719*td2*td3*\cos(t4)*\sin(t3))/625000 - \\
 & (48719*td2*td4*\cos(t3)*\sin(t4))/625000 - (20339*td3^2*\cos(90)^3*\cos(t4)^2*\sin(t3))/625000 - \\
 & (14663*td4*\cos(90)^2*\cos(t2)*\cos(t4))/250000 \\
 & (473*td1*\cos(90)^2*\cos(t3)^2*\cos(t4))/15625 - (20339*td2*\cos(90)^2*\cos(t3)*\cos(t4)^2)/312500 - \\
 & (473*td2*\cos(90)^2*\cos(t4))/15625 + (20339*td1*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + \\
 & (20339*td2*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + (20339*td3*\cos(90)^3*\cos(t3)*\cos(t4)^2)/625000 - \\
 & (14663*td1^2*\cos(90)^2*\cos(t2)*\sin(t4))/250000 + (20339*td3^2*\cos(90)*\cos(t4)^2*\sin(t3))/625000 - \\
 & (67639*td3^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 - (29799*td4^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 + \\
 & (14663*td1^2*\cos(t3)*\cos(t4)*\sin(t2))/250000 + (20339*td3^2*\cos(t3)*\cos(t4)*\sin(t4))/625000 + \\
 & (10403*g*\cos(90)*\cos(t1)*\cos(t2))/50000 + (48719*td3*\cos(90)*\cos(t3)*\cos(t4))/1250000 - \\
 & (20339*td1*td3*\cos(90)^2*\sin(t3))/312500 + (20339*td1*td3*\cos(90)^4*\sin(t3))/312500 - \\
 & (20339*td2*td3*\cos(90)^2*\sin(t3))/312500 + (20339*td1*td4*\cos(90)^3*\sin(t3))/312500 + \\
 & (20339*td2*td3*\cos(90)^4*\sin(t3))/312500 + (20339*td2*td4*\cos(90)^3*\sin(t3))/312500 + \\
 & (20339*td3*td4*\cos(90)^2*\sin(t3))/312500 - (473*td1*td3*\cos(90)^3*\sin(t4))/15625 - \\
 & (473*td1*td4*\cos(90)^2*\sin(t4))/15625 - (473*td2*td3*\cos(90)^3*\sin(t4))/15625 - \\
 & (473*td2*td4*\cos(90)^2*\sin(t4))/15625 - (10403*g*\cos(90)*\sin(t1)*\sin(t2))/50000 + \\
 & (20339*td1*td3*\cos(90)^2*\cos(t4)^2*\sin(t3))/312500 + (946*td1*td3*\cos(90)^3*\cos(t3)^2*\sin(t4))/15625 - \\
 & (20339*td1*td3*\cos(90)^4*\cos(t4)^2*\sin(t3))/312500 + (473*td1*td4*\cos(90)^2*\cos(t3)^2*\sin(t4))/15625 + \\
 & (20339*td2*td3*\cos(90)^2*\cos(t4)^2*\sin(t3))/312500 - (20339*td1*td4*\cos(90)^3*\cos(t4)^2*\sin(t3))/156250 + \\
 & (946*td2*td3*\cos(90)^3*\cos(t3)^2*\sin(t4))/15625 - (20339*td2*td3*\cos(90)^4*\cos(t4)^2*\sin(t3))/312500 + \\
 & (473*td2*td4*\cos(90)^2*\cos(t3)^2*\sin(t4))/15625 - (20339*td2*td4*\cos(90)^3*\cos(t4)^2*\sin(t3))/156250 - \\
 & (20339*td3*td4*\cos(90)^2*\cos(t4)^2*\sin(t3))/312500 + (14663*td1^2*\cos(90)*\cos(t2)*\cos(t4)*\sin(t3))/250000 - \\
 & (41*g*\cos(90)^2*\cos(t1)*\sin(t2)*\sin(t4))/1250 - (41*g*\cos(90)^2*\cos(t2)*\sin(t1)*\sin(t3))/1250 + \\
 & (473*g*\cos(90)^3*\cos(t1)*\sin(t2)*\sin(t4))/12500 + (473*g*\cos(90)^3*\cos(t2)*\sin(t1)*\sin(t4))/12500 - \\
 & (14663*td1*\cos(90)^2*\cos(t3)*\sin(t2)*\sin(t4))/250000 + (473*td1*\cos(90)^3*\cos(t3)*\sin(t3)*\sin(t4))/15625 + \\
 & (473*td2*\cos(90)^3*\cos(t3)*\sin(t3)*\sin(t4))/15625 - (20339*td1*\cos(90)^3*\cos(t4)*\sin(t3)*\sin(t4))/312500 - \\
 & (20339*td2*\cos(90)^3*\cos(t4)*\sin(t3)*\sin(t4))/312500 - (20339*td3*\cos(90)^2*\cos(t4)*\sin(t3)*\sin(t4))/625000 - \\
 & (20339*td1*td3*\cos(t3)*\cos(t4)^2*\sin(t3))/312500 - (20339*td2*td3*\cos(t3)*\cos(t4)^2*\sin(t3))/312500 - \\
 & (20339*td1*td4*\cos(t3)^2*\cos(t4)*\sin(t4))/312500 - (20339*td2*td4*\cos(t3)^2*\cos(t4)*\sin(t4))/312500 \\
 & (473*td2*\cos(90)^2*\cos(t3)^2*\cos(t4))/15625 + (20339*td1*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + \\
 & (20339*td2*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + (20339*td3*\cos(90)^3*\cos(t3)*\cos(t4)^2)/625000 + \\
 & (14663*td2^2*\cos(90)^2*\cos(90)^2*\cos(t2)*\sin(t4))/250000 + (14663*td4^2*\cos(90)^2*\cos(90)^2*\cos(t2)*\sin(t4))/250000 + \\
 & (20339*td3^2*\cos(90)*\cos(t4)^2*\sin(t3))/625000 - (67639*td3^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 - \\
 & (29799*td4^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 - (14663*td2^2*\cos(t3)*\cos(t4)*\sin(t2))/250000 - \\
 & (14663*td3^2*\cos(t3)*\cos(t4)*\sin(t2))/250000 - (14663*td4^2*\cos(t3)*\cos(t4)*\sin(t2))/250000 + \\
 & (20339*td3^2*\cos(t3)*\cos(t4)*\sin(t4))/625000 + (10403*g*\cos(90)*\cos(t1)*\cos(t2))/50000 + \\
 & (1271*td3*\cos(90)*\cos(t2)*\cos(t3))/25000 + (48719*td3*\cos(90)*\cos(t3)*\cos(t4))/1250000 - \\
 & (20339*td1*td3*\cos(90)^2*\sin(t3))/312500 + (20339*td1*td3*\cos(90)^4*\sin(t3))/312500 - \\
 & (20339*td2*td3*\cos(90)^2*\sin(t3))/312500 + (20339*td1*td4*\cos(90)^3*\sin(t3))/312500 + \\
 & (20339*td2*td3*\cos(90)^4*\sin(t3))/312500 + (20339*td2*td4*\cos(90)^3*\sin(t3))/312500 \\
 & (473*td2*\cos(90)^2*\cos(t3)^2*\cos(t4))/15625 + (20339*td1*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + \\
 & (20339*td2*\cos(90)^4*\cos(t3)*\cos(t4)^2)/312500 + (20339*td3*\cos(90)^3*\cos(t3)*\cos(t4)^2)/625000 + \\
 & (14663*td2^2*\cos(90)^2*\cos(t2)*\sin(t4))/250000 + (14663*td4^2*\cos(90)^2*\cos(t2)*\sin(t4))/250000 + \\
 & (20339*td3^2*\cos(90)*\cos(t4)^2*\sin(t3))/625000 - (67639*td3^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 - \\
 & (29799*td4^2*\cos(90)^2*\cos(t3)*\sin(t4))/1250000 - (14663*td2^2*\cos(t3)*\cos(t4)*\sin(t2))/250000 -
 \end{aligned}$$

$$(14663*td3^2*cos(t3)*cos(t4)*sin(t2))/250000 - (14663*td4^2*cos(t3)*cos(t4)*sin(t2))/250000 + (20339*td3^2*cos(t3)*cos(t4)*sin(t4))/625000 + (10403*g*cos(90)*cos(t1)*cos(t2))/50000$$

Expresión del torque (T2) para la articulación numero 2:

$$\begin{aligned} T2 = & (28539*td1*cos(90)^2*cos(t3)^2)/625000 - (28539*td2*cos(90)^2*cos(t3)^2)/625000 + \\ & (20339*td1*cos(90)^4*cos(t3)^2)/625000 + (20339*td2*cos(90)^4*cos(t3)^2)/625000 + \\ & (20339*td1*cos(90)^2*cos(t4)^2)/312500 + (20339*td2*cos(90)^2*cos(t4)^2)/312500 - \\ & (20339*td1*cos(90)^4*cos(t4)^2)/625000 - (20339*td2*cos(90)^4*cos(t4)^2)/625000 - \\ & (20339*td3*cos(90)^3*cos(t4)^2)/625000 + (20339*td3^2*cos(90)^3*sin(t3))/625000 + \\ & (29799*td4^2*cos(90)^2*sin(t4))/1250000 + (473*td1*cos(t3)^2*cos(t4))/15625 + \\ & (473*td2*cos(t3)^2*cos(t4))/15625 - (1271*td2^2*cos(t3)*sin(t2))/25000 - (1271*td3^2*cos(t3)*sin(t2))/25000 - \\ & (14663*td2^2*cos(t2)*sin(t4))/250000 - (14663*td4^2*cos(t2)*sin(t4))/250000 + (473*td3^2*cos(t3)*sin(t4))/31250 - \\ & (473*td4^2*cos(t3)*sin(t4))/31250 - (20339*td1*cos(90)^4*cos(t3)^2*cos(t4)^2)/625000 - \\ & (20339*td2*cos(90)^4*cos(t3)^2*cos(t4)^2)/625000 - (1271*td1*cos(90)*sin(t2)*sin(t3))/12500 - \\ & (1271*td2*cos(90)*sin(t2)*sin(t3))/25000 - (29799*td1*cos(90)*sin(t3)*sin(t4))/625000 - \\ & (29799*td2*cos(90)*sin(t3)*sin(t4))/625000 - (48719*td4*cos(90)*sin(t3)*sin(t4))/1250000 - \\ & (1271*td1*td2*cos(t3)*sin(t2))/12500 - (1271*td1*td3*cos(t2)*sin(t3))/12500 - (1271*td2*td3*cos(t2)*sin(t3))/12500 - \\ & (14663*td1*td2*cos(t2)*sin(t4))/125000 - (14663*td1*td4*cos(t4)*sin(t2))/125000 - \\ & (14663*td2*td4*cos(t4)*sin(t2))/125000 - (48719*td1*td3*cos(t4)*sin(t3))/625000 - \\ & (48719*td1*td4*cos(t3)^2*sin(t4))/15625 - (48719*td2*td3*cos(t4)*sin(t3))/625000 - \\ & (48719*td2*td4*cos(t3)*sin(t4))/625000 - (20339*td3^2*cos(90)^3*cos(t4)^2*sin(t3))/625000 - \\ & (14663*td4*cos(90)^2*cos(t2)*cos(t4))/250000 - (20339*td3*cos(90)*cos(t3)*cos(t4)^2)/625000 + \\ & (29799*td4*cos(90)^2*cos(t3)*cos(t4))/1250000 - (1271*td2^2*cos(90)*cos(t2)*sin(t3))/25000 - \\ & (1271*td3^2*cos(90)*cos(t2)*sin(t3))/25000 - (48719*td3^2*cos(90)*cos(t4)*sin(t3))/1250000 - \\ & (48719*td4^2*cos(90)*cos(t4)*sin(t3))/1250000 + (14663*td1*cos(t2)*cos(t3)*cos(t4))/125000 + \\ & (14663*td2*cos(t2)*cos(t3)*cos(t4))/250000 + (14663*td1*cos(90)^2*sin(t2)*sin(t4))/125000 + \\ & (14663*td2*cos(90)^2*sin(t2)*sin(t4))/250000 - (473*td1*cos(90)^3*sin(t3)*sin(t4))/15625 - \\ & (473*td2*cos(90)^3*sin(t3)*sin(t4))/15625 - (67639*td3*cos(90)^2*sin(t3)*sin(t4))/1250000 - \\ & (473*td1*td4*cos(t3)^2*sin(t4))/15625 - (473*td2*td4*cos(t3)^2*sin(t4))/15625 + \\ & (20339*td3*td4*cos(t4)^2*sin(t3))/312500 - (14663*td3*cos(t4)*sin(t2)*sin(t3))/250000 - \\ & (14663*td4*cos(t3)*sin(t2)*sin(t4))/250000 + (20339*td3*cos(t4)*sin(t3)*sin(t4))/625000 - \\ & (20339*td1*td4*cos(90)*sin(t3))/312500 - (20339*td2*td4*cos(90)*sin(t3))/312500 + \\ & (473*td1*td3*cos(90)*sin(t4))/15625 + (473*td2*td3*cos(90)*sin(t4))/15625 - (473*td3*td4*cos(90)*sin(t4))/15625 - \\ & (20339*td1*cos(90)^2*cos(t3)*cos(t4)^2)/312500 - (473*td1*cos(90)^2*cos(t3)^2*cos(t4))/15625 - \\ & (20339*td2*cos(90)^2*cos(t3)*cos(t4)^2)/312500 - (473*td2*cos(90)^2*cos(t3)^2*cos(t4))/15625 + \\ & (20339*td1*cos(90)^4*cos(t3)*cos(t4)^2)/312500 + (20339*td2*cos(90)^4*cos(t3)*cos(t4)^2)/312500 + \\ & (20339*td3*cos(90)^3*cos(t3)*cos(t4)^2)/625000 + (14663*td2^2*cos(90)^2*cos(t2)*sin(t4))/250000 + \\ & (14663*td4^2*cos(90)^2*cos(t2)*sin(t4))/250000 + (20339*td3^2*cos(90)*cos(t4)^2*sin(t3))/625000 - \\ & (67639*td3^2*cos(90)^2*cos(t3)*sin(t4))/1250000 - (29799*td4^2*cos(90)^2*cos(t3)*sin(t4))/1250000 - \\ & (14663*td2^2*cos(t3)*cos(t4)*sin(t2))/250000 - (14663*td3^2*cos(t3)*cos(t4)*sin(t2))/250000 - \\ & (14663*td4^2*cos(t3)*cos(t4)*sin(t2))/250000 + (20339*td3^2*cos(t3)*cos(t4)*sin(t4))/625000 + \\ & (10403*g*cos(90)*cos(t1)*cos(t2))/50000 + (1271*td3*cos(90)*cos(t2)*cos(t3))/25000 + \\ & (48719*td3*cos(90)*cos(t3)*cos(t4))/1250000 - (20339*td1*td3*cos(90)^2*sin(t3))/312500 + \\ & (20339*td1*td3*cos(90)^4*sin(t3))/312500 - (20339*td2*td3*cos(90)^2*sin(t3))/312500 + \\ & (20339*td1*td4*cos(90)^3*sin(t3))/312500 + (20339*td2*td3*cos(90)^4*sin(t3))/312500 + \\ & (20339*td2*td4*cos(90)^3*sin(t3))/312500 \\ & (20339*td1*td3*cos(90)^4*cos(t3)*sin(t3))/312500 + (28539*td2*td3*cos(90)^2*cos(t3)*sin(t3))/312500 - \\ & (20339*td1*td4*cos(90)^3*cos(t3)*sin(t3))/312500 - (20339*td2*td3*cos(90)^4*cos(t3)*sin(t3))/312500 - \\ & (20339*td2*td4*cos(90)^3*cos(t3)*sin(t3))/312500 - (946*td1*td3*cos(90)*cos(t3)^2*sin(t4))/15625 - \\ & (473*td1*td3*cos(90)^3*cos(t3)*sin(t4))/15625 + (20339*td1*td4*cos(90)*cos(t4)^2*sin(t3))/156250 - \\ & (946*td2*td3*cos(90)*cos(t3)^2*sin(t4))/15625 - (473*td1*td4*cos(90)^3*cos(t4)*sin(t3))/15625 - \\ & (473*td2*td3*cos(90)^3*cos(t3)*sin(t4))/15625 + (20339*td2*td4*cos(90)*cos(t4)^2*sin(t3))/156250 - \\ & (473*td2*td4*cos(90)^3*cos(t4)*sin(t3))/15625 - (48719*td3*td4*cos(90)^2*cos(t4)*sin(t3))/625000 - \\ & (20339*td1*td3*cos(90)^3*cos(t4)*sin(t4))/312500 - (20339*td1*td4*cos(90)^2*cos(t4)*sin(t4))/156250 - \\ & (20339*td2*td3*cos(90)^3*cos(t4)*sin(t4))/312500 + (20339*td1*td4*cos(90)^4*cos(t4)*sin(t4))/312500 - \\ & (20339*td2*td4*cos(90)^2*cos(t4)*sin(t4))/156250 + (20339*td2*td4*cos(90)^4*cos(t4)*sin(t4))/312500 + \\ & (20339*td3*td4*cos(90)^3*cos(t4)*sin(t4))/312500 - (41*g*cos(90)*cos(t3)*sin(t1)*sin(t2))/1250 - \\ & (473*g*cos(90)*cos(t1)*sin(t2)*sin(t4))/12500 - (473*g*cos(90)*cos(t2)*sin(t1)*sin(t4))/12500 - \\ & (14663*td1*cos(90)*cos(t2)*sin(t3)*sin(t4))/250000 - (14663*td1*cos(90)*cos(t4)*sin(t2)*sin(t3))/250000 - \\ & (473*td1*cos(90)*cos(t3)*sin(t4))/15625 + (1271*td1*td3*cos(90)*cos(t3)*sin(t2))/12500 - \\ & (1271*td2*td3*cos(90)*cos(t3)*sin(t2))/12500 + (20339*td1*td4*cos(90)*cos(t3)*sin(t3))/312500 + \\ & (20339*td2*td4*cos(90)*cos(t3)*sin(t3))/312500 - (29799*td1*td3*cos(90)*cos(t3)*sin(t4))/625000 - \\ & (29799*td1*td4*cos(90)*cos(t4)*sin(t3))/625000 - (29799*td2*td3*cos(90)*cos(t3)*sin(t4))/625000 - \\ & (29799*td2*td4*cos(90)*cos(t4)*sin(t3))/625000 - (48719*td3*td4*cos(90)*cos(t3)*sin(t4))/625000 + \\ & (20339*td1*td3*cos(90)*cos(t4)*sin(t4))/312500 + (20339*td2*td3*cos(90)*cos(t4)*sin(t4))/312500 - \\ & (20339*td3*td4*cos(90)*cos(t4)*sin(t4))/312500 - (14663*td2^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - \\ & (14663*td3^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - (14663*td4^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - \\ & (20339*td3^2*cos(90)^2*cos(t3)*cos(t4)*sin(t4))/625000 + (41*g*cos(90)*cos(t1)*cos(t2)*cos(t3))/1250 + \\ & (14663*td3*cos(90)*cos(t2)*cos(t3)*cos(t4))/250000 + (14663*td1*td2*cos(90)^2*cos(t2)*sin(t4))/125000 + \\ & (28539*td1*td3*cos(90)^2*cos(t3)*sin(t3))/312500 - (20339*td1*td3*cos(90)^4*cos(t3)*sin(t3))/312500 + \\ & (14663*td1*td4*cos(90)^2*cos(t4)*sin(t2))/125000 + (1271*td1*td3*cos(90)*cos(t3)*sin(t2))/12500 - \end{aligned}$$

$$(1271*td2*td3*cos(90)*cos(t3)*sin(t2))/12500 + (20339*td1*td4*cos(90)*cos(t3)*sin(t3))/312500 + (20339*td2*td4*cos(90)*cos(t3)*sin(t3))/312500 - (29799*td1*td3*cos(90)*cos(t3)*sin(t4))/625000 - (29799*td1*td4*cos(90)*cos(t4)*sin(t3))/625000 - (29799*td2*td3*cos(90)*cos(t3)*sin(t4))/625000$$

Expresión del torque (T3) para la articulación numero 3:

$$\begin{aligned} T_3 = & (14663*td1*cos(90)^2*cos(t3)*sin(t2)*sin(t4))/125000 - (14663*td2*cos(90)^2*cos(t3)*sin(t2)*sin(t4))/250000 \\ & - (14663*td3*cos(90)^2*cos(t2)*sin(t3)*sin(t4))/250000 + (473*td1*cos(90)^3*cos(t3)*sin(t3)*sin(t4))/15625 + \\ & (473*td2*cos(90)^3*cos(t3)*sin(t3)*sin(t4))/15625 - (20339*td1*cos(90)^3*cos(t4)*sin(t3)*sin(t4))/312500 - \\ & (20339*td2*cos(90)^3*cos(t4)*sin(t3)*sin(t4))/312500 - (20339*td3*cos(90)^2*cos(t4)*sin(t3)*sin(t4))/625000 - \\ & (20339*td1*td3*cos(t3)*cos(t4)^2*sin(t3))/312500 - (20339*td2*td3*cos(t3)*cos(t4)^2*sin(t3))/312500 - \\ & (20339*td1*td4*cos(t3)^2*cos(t4)*sin(t4))/312500 - (20339*td2*td4*cos(t3)^2*cos(t4)*sin(t4))/312500 + \\ & (14663*td2^2*cos(90)*sin(t2)*sin(t3)*sin(t4))/250000 + (14663*td3^2*cos(90)*sin(t2)*sin(t3)*sin(t4))/250000 + \\ & (14663*td4^2*cos(90)*sin(t2)*sin(t3)*sin(t4))/250000 - (1271*td1*td2*cos(90)*cos(t2)*sin(t3))/12500 - \\ & (1271*td1*td3*cos(90)*cos(t3)*sin(t2))/12500 - (1271*td2*td3*cos(90)*cos(t3)*sin(t2))/12500 + \\ & (20339*td1*td4*cos(90)*cos(t3)*sin(t3))/312500 + (20339*td2*td4*cos(90)*cos(t3)*sin(t3))/312500 - \\ & (29799*td1*td3*cos(90)*cos(t3)*sin(t4))/625000 - (29799*td1*td4*cos(90)*cos(t4)*sin(t3))/625000 - \\ & (29799*td2*td3*cos(90)*cos(t3)*sin(t4))/625000 - (29799*td2*td4*cos(90)*cos(t4)*sin(t3))/625000 - \\ & (48719*td3*td4*cos(90)*cos(t3)*sin(t4))/625000 + (20339*td1*td3*cos(90)*cos(t4)*sin(t4))/312500 + \\ & (20339*td1*td4*cos(90)*cos(t4)*sin(t4))/312500 - (20339*td3*td4*cos(90)*cos(t4)*sin(t4))/312500 - \\ & (14663*td2^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - (14663*td3^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - \\ & (14663*td4^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - (20339*td3^2*cos(90)^2*cos(t3)*cos(t4)*sin(t4))/625000 \\ & + (41*g*cos(90)*cos(t1)*cos(t2)*cos(t3))/1250 + (14663*td3*cos(90)*cos(t2)*cos(t3)*cos(t4))/250000 + \\ & (14663*td1*td2*cos(90)^2*cos(t2)*sin(t4))/125000 + (28539*td1*td3*cos(90)^2*cos(t3)*sin(t3))/312500 - \\ & (20339*td1*td3*cos(90)^4*cos(t3)*sin(t3))/312500 + (14663*td1*td4*cos(90)^2*cos(t4)*sin(t2))/125000 + \\ & (28539*td2*td3*cos(90)^2*cos(t3)*sin(t3))/312500 - (20339*td1*td4*cos(90)^3*cos(t3)*sin(t3))/312500 - \\ & (20339*td2*td3*cos(90)^4*cos(t3)*sin(t3))/312500 + (14663*td2*td4*cos(90)^2*cos(t4)*sin(t2))/125000 - \\ & (20339*td2*td4*cos(90)^3*cos(t3)*sin(t3))/312500 - (946*td1*td3*cos(90)*cos(t3)^2*sin(t4))/15625 - \\ & (473*td1*td3*cos(90)^3*cos(t3)*sin(t4))/15625 + (20339*td1*td4*cos(90)*cos(t4)^2*sin(t3))/156250 - \\ & (946*td2*td3*cos(90)*cos(t3)^2*sin(t4))/15625 - (473*td1*td4*cos(90)^3*cos(t4)*sin(t3))/15625 - \\ & (473*td2*td3*cos(90)^3*cos(t3)*sin(t4))/15625 + (20339*td2*td4*cos(90)*cos(t4)^2*sin(t3))/156250 - \\ & (473*td2*td4*cos(90)^3*cos(t4)*sin(t3))/15625 - (48719*td3*td4*cos(90)^2*cos(t4)*sin(t3))/625000 - \\ & (20339*td1*td3*cos(90)^3*cos(t4)*sin(t4))/312500 - (20339*td1*td4*cos(90)^2*cos(t4)*sin(t4))/156250 - \\ & (20339*td2*td3*cos(90)^3*cos(t4)*sin(t4))/312500 + (20339*td1*td4*cos(90)^4*cos(t4)*sin(t4))/312500 - \\ & (20339*td2*td4*cos(90)^2*cos(t4)*sin(t4))/156250 + (20339*td2*td4*cos(90)^4*cos(t4)*sin(t4))/312500 + \\ & (20339*td3*td4*cos(90)^3*cos(t4)*sin(t4))/312500 - (41*g*cos(90)*cos(t3)*sin(t1)*sin(t2))/1250 - \\ & (473*g*cos(90)*cos(t1)*sin(t2)*sin(t4))/12500 - (473*g*cos(90)*cos(t2)*sin(t1)*sin(t4))/12500 - \\ & (473*td1*cos(90)*cos(t3)*sin(t3)*sin(t4))/15625 - (473*td2*cos(90)*cos(t3)*sin(t3)*sin(t4))/15625 + \\ & (20339*td1*cos(90)*cos(t4)*sin(t3)*sin(t4))/312500 + (20339*td2*cos(90)*cos(t4)*sin(t3)*sin(t4))/312500 - \\ & (14663*td1*td2*cos(t3)*cos(t4)*sin(t2))/125000 - (14663*td1*td3*cos(t2)*cos(t4)*sin(t3))/125000 - \\ & (14663*td1*td4*cos(t2)*cos(t3)*sin(t4))/125000 - (14663*td2*td3*cos(t2)*cos(t4)*sin(t3))/125000 - \\ & (14663*td2*td4*cos(t3)*cos(t4)*sin(t3))/15625 + (14663*td3*td4*sin(t2)*sin(t3)*sin(t4))/125000 \\ & (473*td1*cos(90)^2*cos(t3)^2*cos(t4))/15625 - (20339*td2*cos(90)^2*cos(t3)*cos(t4)^2)/312500 - \\ & (473*td2*cos(90)^2*cos(t3)^2*cos(t4))/15625 + (20339*td1*cos(90)^4*cos(t3)*cos(t4)^2)/312500 + \\ & (20339*td2*cos(90)^4*cos(t3)*cos(t4)^2)/312500 + (20339*td3*cos(90)^3*cos(t3)*cos(t4)^2)/625000 - \\ & (14663*td1^2*cos(90)^2*cos(t2)*sin(t4))/250000 + (20339*td2^2*cos(90)*cos(t4)^2*sin(t3))/312500 - \\ & (67639*td3^2*cos(90)^2*cos(t3)*sin(t4))/1250000 - (29799*td4^2*cos(90)^2*cos(t3)*sin(t4))/1250000 + \\ & (14663*td1^2*cos(t3)*cos(t4)*sin(t2))/250000 + (20339*td3^2*cos(t3)*cos(t4)*sin(t4))/625000 + \\ & (10403*g*cos(90)*cos(t1)*cos(t2))/50000 + (48719*td3*cos(90)*cos(t3)*cos(t4))/1250000 - \\ & (20339*td1*td3*cos(90)^2*cos(t3))/312500 + (20339*td1*td3*cos(90)^4*sin(t3))/312500 - \\ & (20339*td2*td3*cos(90)^2*sin(t3))/312500 + (20339*td1*td4*cos(90)^3*sin(t3))/312500 + \\ & (20339*td2*td3*cos(90)^4*sin(t3))/312500 + (20339*td2*td4*cos(90)^3*sin(t3))/312500 + \\ & (20339*td3*td4*cos(90)^2*sin(t3))/312500 - (473*td1*td3*cos(90)^3*sin(t4))/15625 - \\ & (473*td1*td4*cos(90)^2*sin(t4))/15625 - (473*td2*td3*cos(90)^3*sin(t4))/15625 - \\ & (473*td2*td4*cos(90)^2*sin(t4))/15625 - (10403*g*cos(90)*sin(t1)*sin(t2))/50000 + \\ & (20339*td1*td3*cos(90)^2*cos(t4)^2*sin(t3))/312500 + (946*td1*td3*cos(90)^3*cos(t3)^2*sin(t4))/15625 - \\ & (20339*td1*td3*cos(90)^4*cos(t4)^2*sin(t3))/312500 + (473*td1*td4*cos(90)^2*cos(t3)^2*sin(t4))/15625 + \\ & (20339*td2*td3*cos(90)^2*cos(t4)^2*sin(t3))/312500 - (20339*td1*td4*cos(90)^3*cos(t4)^2*sin(t3))/156250 + \\ & (946*td2*td3*cos(90)^3*cos(t3)^2*sin(t4))/15625 - (20339*td2*td3*cos(90)^4*cos(t4)^2*sin(t3))/312500 + \\ & (473*td2*td4*cos(90)^2*cos(t3)^2*sin(t4))/15625 - (20339*td2*td4*cos(90)^3*cos(t4)^2*sin(t3))/156250 - \\ & (20339*td3*td4*cos(90)^2*cos(t4)^2*sin(t3))/312500 + (14663*td1^2*cos(90)*cos(t2)*cos(t4)*sin(t3))/250000 - \\ & (41*g*cos(90)^2*cos(t1)*sin(t2)*sin(t3))/1250 - (41*g*cos(90)^2*cos(t2)*sin(t1)*sin(t3))/1250 + \\ & (473*g*cos(90)^3*cos(t1)*sin(t2)*sin(t4))/12500 + (473*g*cos(90)^3*cos(t2)*sin(t1)*sin(t4))/12500 \\ & (1271*td1*td3*cos(90)*cos(t3)*sin(t2))/12500 - (1271*td2*td3*cos(90)*cos(t3)*sin(t2))/12500 + \\ & (20339*td1*td4*cos(90)*cos(t3)*sin(t3))/312500 + (20339*td2*td4*cos(90)*cos(t3)*sin(t3))/312500 - \\ & (29799*td1*td3*cos(90)*cos(t3)*sin(t4))/625000 - (29799*td1*td4*cos(90)*cos(t4)*sin(t3))/625000 - \\ & (29799*td2*td3*cos(90)*cos(t3)*sin(t4))/625000 - (29799*td2*td4*cos(90)*cos(t4)*sin(t3))/625000 - \\ & (48719*td3*td4*cos(90)*cos(t3)*sin(t4))/625000 + (20339*td1*td3*cos(90)*cos(t4)*sin(t4))/312500 + \\ & (20339*td2*td3*cos(90)*cos(t4)*sin(t4))/312500 - (20339*td3*td4*cos(90)*cos(t4)*sin(t4))/312500 - \\ & (14663*td2^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - (14663*td3^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - \\ & (14663*td4^2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/250000 - (20339*td3^2*cos(90)^2*cos(t3)*cos(t4)*sin(t4))/625000 \\ & + (41*g*cos(90)*cos(t1)*cos(t2)*cos(t3))/1250 + (14663*td3*cos(90)*cos(t2)*cos(t3)*cos(t4))/250000 + \end{aligned}$$

$$(14663*td1*td2*cos(90)^2*cos(t2)*sin(t4))/125000 + (28539*td1*td3*cos(90)^2*cos(t3)*sin(t3))/312500 - (20339*td1*td3*cos(90)^4*cos(t3)*sin(t3))/312500 + (14663*td1*td4*cos(90)^2*cos(t4)*sin(t2))/125000$$

Expresión del torque (T4) para la articulación numero 4:

$$\begin{aligned} T4 = & (20339*td1*td3*cos(90)^4*cos(t3)*cos(t4)^2*sin(t3))/312500 + \\ & (20339*td1*td4*cos(90)^3*cos(t3)*cos(t4)^2*sin(t3))/156250 + \\ & (20339*td2*td3*cos(90)^4*cos(t3)*cos(t4)^2*sin(t3))/312500 + \\ & (20339*td2*td4*cos(90)^3*cos(t3)*cos(t4)^2*sin(t3))/156250 + \\ & (20339*td1*td3*cos(90)^3*cos(t3)^2*cos(t4)*sin(t4))/156250 + \\ & (20339*td2*td3*cos(90)^3*cos(t3)^2*cos(t4)*sin(t4))/156250 + \\ & (20339*td1*td4*cos(90)^4*cos(t3)^2*cos(t4)*sin(t4))/312500 + \\ & (20339*td2*td4*cos(90)^4*cos(t3)^2*cos(t4)*sin(t4))/312500 - \\ & (473*g*cos(90)^2*cos(t1)*cos(t2)*sin(t3)*sin(t4))/12500 - (473*g*cos(90)^2*cos(t1)*cos(t4)*sin(t2)*sin(t3))/12500 - \\ & (473*g*cos(90)^2*cos(t2)*cos(t4)*sin(t1)*sin(t3))/12500 - (473*g*cos(90)^3*cos(t1)*cos(t3)*sin(t2)*sin(t4))/12500 - \\ & (473*g*cos(90)^3*cos(t2)*cos(t3)*sin(t1)*sin(t4))/12500 + \\ & (20339*td1*td3*cos(90)^3*cos(t3)*cos(t4)*sin(t3)*sin(t4))/312500 + \\ & (20339*td2*td3*cos(90)^3*cos(t3)*cos(t4)*sin(t3)*sin(t4))/312500 + \\ & (473*g*cos(90)^2*sin(t1)*sin(t2)*sin(t3)*sin(t4))/12500 - (14663*td1*td2*cos(90)*cos(t2)*cos(t4)*sin(t3))/125000 - \\ & (14663*td1*td3*cos(90)*cos(t2)*cos(t3)*sin(t4))/125000 - (14663*td1*td3*cos(90)*cos(t3)*cos(t4)*sin(t2))/125000 - \\ & (14663*td1*td4*cos(90)*cos(t2)*cos(t3)*sin(t4))/125000 - (14663*td2*td3*cos(90)*cos(t2)*cos(t3)*sin(t4))/125000 - \\ & (14663*td2*td4*cos(90)*cos(t2)*cos(t4)*sin(t3))/125000 - (14663*td3*td4*cos(90)*cos(t2)*cos(t3)*sin(t4))/125000 - \\ & (14663*td3*td4*cos(90)*cos(t3)*cos(t4)*sin(t2))/125000 - (473*td1*td4*cos(90)*cos(t3)*cos(t4)*sin(t3))/15625 - (473*td2*td4*cos(90)*cos(t3)*cos(t4)*sin(t3))/15625 + \\ & (20339*td1*td3*cos(90)*cos(t3)*cos(t4)*sin(t4))/312500 + (20339*td2*td3*cos(90)*cos(t3)*cos(t4)*sin(t4))/312500 + \\ & (14663*td1*td2*cos(90)*sin(t2)*sin(t3)*sin(t4))/125000 + (14663*td2*td4*cos(90)*sin(t2)*sin(t3)*sin(t4))/125000 + \\ & (473*g*cos(90)*cos(t1)*cos(t2)*cos(t3)*cos(t4))/12500 - (14663*td1*td2*cos(90)^2*cos(t2)*cos(t3)*sin(t4))/125000 - \\ & (14663*td1*td4*cos(90)^2*cos(t3)*cos(t4)*sin(t2))/125000 - (14663*td2*td4*cos(90)^2*cos(t3)*cos(t4)*sin(t2))/125000 - \\ & (14663*td3*td4*cos(90)^2*cos(t2)*cos(t4)*sin(t3))/125000 + (946*td1*td3*cos(90)^2*cos(t3)*cos(t4)*sin(t3))/15625 - \\ & (20339*td1*td4*cos(90)*cos(t3)*cos(t4)^2*sin(t3))/156250 + (946*td2*td3*cos(90)^2*cos(t3)*cos(t4)*sin(t3))/15625 + \\ & (473*td1*td4*cos(90)^3*cos(t3)*cos(t4)*sin(t3))/15625 - (20339*td2*td4*cos(90)*cos(t3)*cos(t4)^2*sin(t3))/156250 + \\ & (473*td2*td4*cos(90)^3*cos(t3)*cos(t4)*sin(t3))/15625 - (20339*td1*td3*cos(90)*cos(t3)^2*cos(t4)*sin(t4))/156250 - \\ & (20339*td1*td3*cos(90)^3*cos(t3)*cos(t4)*sin(t4))/312500 + \\ & (20339*td1*td4*cos(90)^2*cos(t3)*cos(t4)*sin(t4))/156250 - \\ & (20339*td2*td3*cos(90)*cos(t3)^2*cos(t4)*sin(t4))/156250 - \\ & (20339*td2*td3*cos(90)^3*cos(t3)*cos(t4)*sin(t4))/312500 + \\ & (20339*td1*td4*cos(90)^4*cos(t3)*cos(t4)*sin(t4))/156250 + \\ & (20339*td2*td4*cos(90)^2*cos(t3)*cos(t4)*sin(t4))/156250 - \\ & (20339*td2*td4*cos(90)^4*cos(t3)*cos(t4)*sin(t4))/156250 - \\ & (20339*td3*td4*cos(90)^3*cos(t3)*cos(t4)*sin(t4))/312500 - (473*g*cos(90)*cos(t3)*cos(t4)*sin(t1)*sin(t2))/12500 - \\ & (20339*td1*td4*cos(90)*cos(t3)*cos(t4)*sin(t3))/312500 - \\ & (20339*td2*td4*cos(90)*cos(t3)*cos(t4)*sin(t3))/312500 + \\ & (14663*td1*td3*cos(90)^2*sin(t2)*sin(t3)*sin(t4))/125000 + \\ & (14663*td2*td3*cos(90)^2*sin(t2)*sin(t3)*sin(t4))/125000, (1234221*td1)/5000000 + (1234221*td2)/5000000 + \\ & (41*td1*cos(t3)^2)/3125 + (41*td2*cos(t3)^2)/3125 - (20339*td1*cos(t4)^2)/625000 - \\ & (20339*td2*cos(t4)^2)/625000 + (322493*td1^2*sin(t2))/1000000 - (48719*td4^2*sin(t4))/1250000 + \\ & (41*td3*cos(90))/3125 - (12139*td1*cos(90)^2)/625000 - (12139*td2*cos(90)^2)/625000 + \\ & (20339*td1*cos(90)^4)/625000 + (20339*td2*cos(90)^4)/625000 + (20339*td3*cos(90)^3)/625000 + \\ & (20339*td4*cos(90)^2)/625000 + (322493*td1*cos(t2))/1000000 + (4223*td1*cos(t3))/62500 + \\ & (4223*td2*cos(t3))/62500 + (20339*td4*cos(t3))/625000 + (48719*td4*cos(t4))/1250000 + \\ & (20727*td3*cos(90)*cos(t3))/312500 + (473*td3*cos(90)*cos(t4))/15625 - (4223*td1*td3*sin(t3))/62500 - \\ & (4223*td2*td3*sin(t3))/62500 - (20339*td3*td4*sin(t3))/312500 + (20339*td1*cos(t3)^2*cos(t4)^2)/625000 + \\ & (20339*td2*cos(t3)^2*cos(t4)^2)/625000 + (20339*td1*cos(90)^2*cos(t3))/312500 + \\ & (20339*td2*cos(90)^2*cos(t3))/312500 - (20339*td1*cos(90)^4*cos(t3))/312500 - \\ & (20339*td2*cos(90)^4*cos(t3))/312500 - (20339*td3*cos(90)^3*cos(t3))/625000 - \\ & (20339*td4*cos(90)^2*cos(t3))/625000 + (473*td1*cos(90)^2*cos(t4))/15625 + \\ & (473*td2*cos(90)^2*cos(t4))/15625 + (20339*td3*cos(90)*cos(t4)^2)/625000 - \\ & (29799*td4*cos(90)^2*cos(t4))/1250000 - (20727*td3^2*cos(90)*sin(t3))/312500 + \\ & (1271*td1*cos(t2)*cos(t3))/25000 + (48719*td1*cos(t3)*cos(t4))/625000 + (48719*td2*cos(t3)*cos(t4))/625000 + \\ & (473*td4*cos(t3)*cos(t4))/31250 - (41*td1*td3*sin(2*t3))/3125 - (41*td2*td3*sin(2*t3))/3125 + \\ & (20339*td1*td4*sin(2*t4))/625000 + (20339*td2*td4*sin(2*t4))/625000 - (14663*td1*sin(t2)*sin(t4))/250000 + \\ & (473*td3*sin(t3)*sin(t4))/31250 - (28539*td1*cos(90)^2*cos(t3)^2)/625000 - \\ & (28539*td2*cos(90)^2*cos(t3)^2)/625000 + (20339*td1*cos(90)^4*cos(t3)^2)/625000 + \\ & (20339*td2*cos(90)^4*cos(t3)^2)/625000 + (20339*td1*cos(90)^2*cos(t4)^2)/312500 + \\ & (20339*td2*cos(90)^2*cos(t4)^2)/312500 - (20339*td1*cos(90)^4*cos(t4)^2)/625000 - \\ & (20339*td2*cos(90)^4*cos(t4)^2)/625000 - (20339*td3*cos(90)^3*cos(t4)^2)/625000 + \\ & (20339*td3^2*cos(90)^3*sin(t3))/625000 + (29799*td4^2*cos(90)^2*sin(t4))/1250000 + \\ & (473*td1*cos(t3)^2*cos(t4))/15625 + (473*td2*cos(t3)^2*cos(t4))/15625 + (1271*td1^2*cos(t3)*sin(t2))/25000 + \\ & (14663*td1^2*cos(t2)*sin(t4))/250000 + (473*td3^2*cos(t3)*sin(t4))/31250 - (473*td4^2*cos(t3)*sin(t4))/31250 - \\ & (20339*td1*cos(90)^4*cos(t3)^2*cos(t4)^2)/625000 - (20339*td2*cos(90)^4*cos(t3)^2*cos(t4)^2)/625000 - \end{aligned}$$

$$\begin{aligned}
& (1271 * tdd1 * \cos(90) * \sin(t2) * \sin(t3)) / 25000 - (29799 * tdd1 * \cos(90) * \sin(t3) * \sin(t4)) / 625000 - \\
& (29799 * tdd2 * \cos(90) * \sin(t3) * \sin(t4)) / 625000 - (48719 * tdd4 * \cos(90) * \sin(t3) * \sin(t4)) / 1250000 - \\
& (48719 * td1 * td3 * \cos(t4) * \sin(t3)) / 625000 - \\
& (1271 * td1 * td3 * \cos(90) * \cos(t3) * \sin(t2)) / 12500 - (1271 * td2 * td3 * \cos(90) * \cos(t3) * \sin(t2)) / 12500 + \\
& (20339 * td1 * td4 * \cos(90) * \cos(t3) * \sin(t3)) / 312500 + (20339 * td2 * td4 * \cos(90) * \cos(t3) * \sin(t3)) / 312500 - \\
& (29799 * td1 * td3 * \cos(90) * \cos(t3) * \sin(t4)) / 625000 - (29799 * td1 * td4 * \cos(90) * \cos(t4) * \sin(t3)) / 625000 - \\
& (29799 * td2 * td3 * \cos(90) * \cos(t3) * \sin(t4)) / 625000 - (29799 * td2 * td4 * \cos(90) * \cos(t4) * \sin(t3)) / 625000 - \\
& (48719 * td3 * td4 * \cos(90) * \cos(t3) * \sin(t4)) / 625000 + (20339 * td1 * td3 * \cos(90) * \cos(t4) * \sin(t4)) / 312500 + \\
& (20339 * td2 * td3 * \cos(90) * \cos(t4) * \sin(t4)) / 312500 - (20339 * td3 * td4 * \cos(90) * \cos(t4) * \sin(t4)) / 312500 - \\
& (14663 * td2^2 * \cos(90)^2 * \cos(t2) * \cos(t3) * \sin(t4)) / 250000 - (14663 * td3^2 * \cos(90)^2 * \cos(t2) * \cos(t3) * \sin(t4)) / 250000 - \\
& (14663 * td4^2 * \cos(90)^2 * \cos(t2) * \cos(t3) * \sin(t4)) / 250000 - (20339 * td3^2 * \cos(90)^2 * \cos(t3) * \cos(t4) * \sin(t4)) / 625000 \\
& + (41 * g * \cos(90) * \cos(t1) * \cos(t2) * \cos(t3)) / 1250 + (14663 * tdd3 * \cos(90) * \cos(t2) * \cos(t3) * \cos(t4)) / 250000 + \\
& (14663 * td1 * td2 * \cos(90)^2 * \cos(t2) * \sin(t4)) / 125000 + (28539 * td1 * td3 * \cos(90)^2 * \cos(t3) * \sin(t3)) / 312500 - \\
& (14663 * td1 * td4 * \cos(90)^2 * \cos(t4) * \sin(t2)) / 125000 + (20339 * td1 * td3 * \cos(90) * \cos(t3)^2 * \cos(t4) * \sin(t4)) / 156250 - \\
& (20339 * td1 * td3 * \cos(90)^3 * \cos(t3) * \cos(t4) * \sin(t4)) / 312500 + \\
& (20339 * td1 * td4 * \cos(90)^2 * \cos(t3) * \cos(t4) * \sin(t4)) / 156250
\end{aligned}$$

Expresión del torque (T5) para la articulación numero 5:

$$T5 = 0$$

Aclaración: La última articulación tiene el par motor igual a cero, ya que el extremo del robot está sin carga.

Se puede observar la complejidad de las expresiones de torque, esto se debe la cantidad de grados de libertad. Dicha complejidad aumenta al agregar mas articulaciones y por consiguiente grados de libertad.

Finalmente las expresiones de los torque quedan en función de los siguientes parámetros:

t1, t2, t3, t4, t5: Variables articulares tita1, tita2, tita3, tita4

td1, td2, td3, td4, td5: Velocidades articulares tita'1, tita'2, tita'3, tita'4

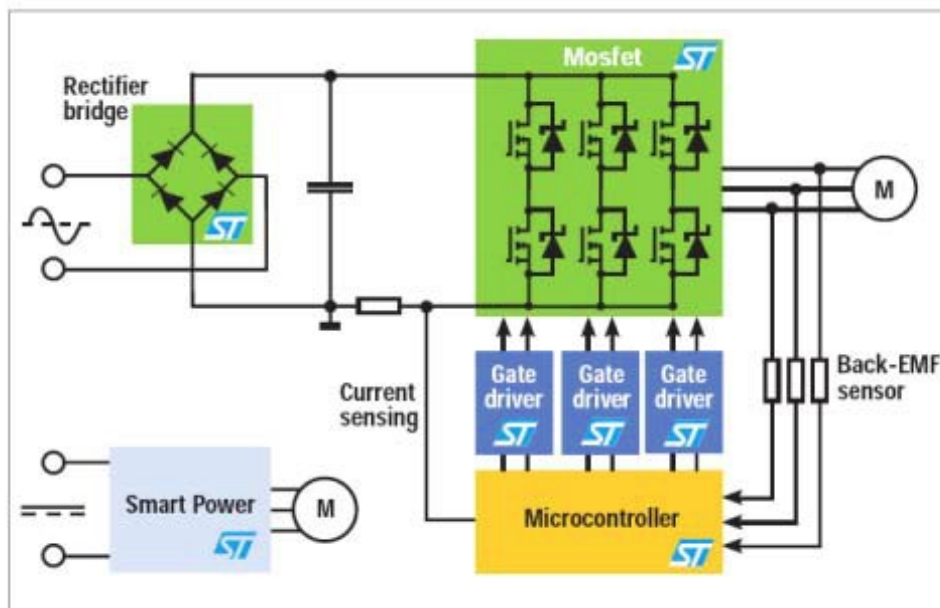
tdd1, tdd2, tdd3, tdd4, tdd5: Aceleraciones articulares tita''1, tita''2, tita''3 y tita''4

g= Aceleración de la gravedad

Sistema de control para motores Brushless

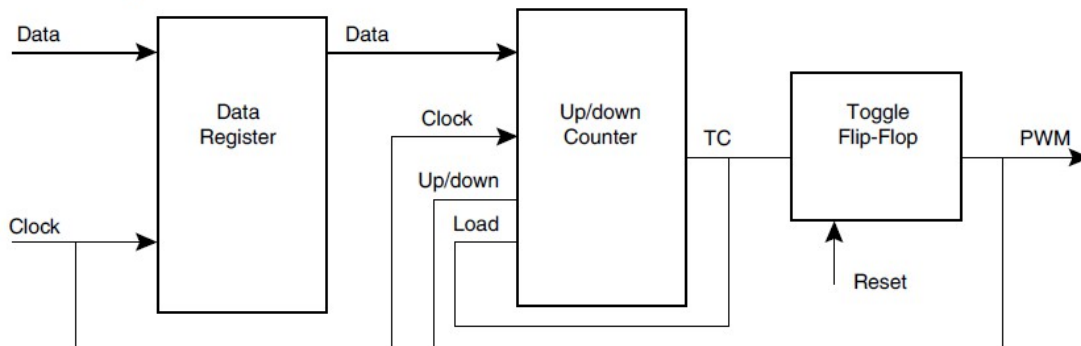
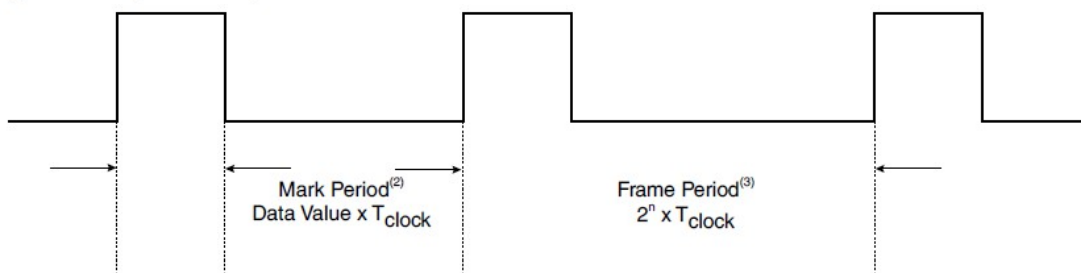
Para crear un sistema de control para un motor Brushless utilizaremos modulación **PWM**, módulos **FPGA** y el lenguaje descriptor de hardware **VHDL**

El diagrama en bloques del drive para dicho motor es el siguiente:



High-frequency three-phase brushless DC motor drive

Implementaremos el PWM usando FPGAs de Atmel

Block Diagram**Figure 1.** Sample PWM Output Waveform⁽¹⁾

- Notes:
1. Duty Cycle is calculated by taking the ratio of Mark Period and Frame Period:
Duty Cycle = Mark Period/Frame Period = Data Value/ 2^n .
 2. Mark Period = Data Value $\times T_{\text{clock}}$.
 3. Frame Period = $T_{\text{clock}} \times 2^n$, where "n" is the binary counter width.

Descripción Funcional

Un registro de Datos es utilizado para almacenar el valor de la cuenta, este valor determina el ancho del pulso.

El up / down de la cuenta es cargado con un nuevo valor del registro de datos cuando el contador alcanza su cuenta final.

Un flip-flop T genera la salida PWM. Cuando se carga la primer cuenta el contador comienza a decrecer hasta 0. Durante este tiempo la señal de salida PWM es igual a 0.

Cuando la cuenta llega a 0 se dispara el Flip-Flop T y pasa al estado alto la señal PWM.

Luego se re-carga el valor del Dato y la cuenta ahora incrementa hasta el máx valor. Nuevamente cuando se llega a este valor máx la señal PWM pasa del estado alto a bajo.

El valor del Dato es recargado y el ciclo se repite.

La dirección del contador se controla por la señal PWM.

El ciclo de trabajo de la señal PWM es controlado por el valor del Dato cargado en el contador.

Cuanto más alto el valor del Dato, más alto el ciclo de trabajo.

Código ejemplo dado por Atmel para generar la señal PWM**pwm_fpga.vhd**

```
library IEEE;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all ;
USE work.user_pkg.all;
ENTITY pwm_fpga IS
PORT ( clock,reset :in STD_LOGIC;
      Data_value :in std_logic_vector(7 downto 0);
      pwm :out STD_LOGIC);
END pwm_fpga;
ARCHITECTURE arch_pwm OF pwm_fpga IS
SIGNAL reg_out : std_logic_vector(7 downto 0);
SIGNAL cnt_out_int : std_logic_vector(7 downto 0);
SIGNAL pwm_int, rco_int : STD_LOGIC;
BEGIN
-- 8 BIT DATA REGISTER TO STORE THE MARKING VALUES .
-- THE MARKING VALUES WILL DETERMINE THE DUTY CYCLE OF PWM OUTPUT
PROCESS(clock,reg_out,reset)
BEGIN
IF (reset ='1') THEN
    reg_out <="00000000";
ELSIF (rising_edge(clock)) THEN
    reg_out <= data_value;
END IF;
END PROCESS;
-- 8 BIT UPDN COUNTER. COUNTS UP OR DOWN BASED ON THE PWM_INT
SIGNAL AND
GENERATES
-- TERMINAL COUNT WHENEVER COUNTER REACHES THE MAXIMUM VALUE OR
WHEN IT
TRANSISTS
-- THROUGH ZERO. THE TERMINAL COUNT WILL BE USED AS INTERRUPT TO
AVR FOR GENERATING
-- THE LOAD SIGNAL.

-- INC and DEC are the two functions which are used for up and down counting.
They are defined in sepearate user_pakge library

PROCESS (clock,cnt_out_int,rco_int,reg_out)
BEGIN
IF (rco_int = '1') THEN
    cnt_out_int <= reg_out;
ELSIF rising_edge(clock) THEN
IF (rco_int = '0' and pwm_int ='1' and cnt_out_int <"11111111") THEN
    cnt_out_int <= INC(cnt_out_int);
ELSE
IF (rco_int ='0' and pwm_int ='0' and cnt_out_int > "00000000") THEN
```

```

        cnt_out_int <= DEC(cnt_out_int);
    END IF;
END IF;
END IF;
END PROCESS;
-- Logic to generate RCO signal

PROCESS(cnt_out_int, rco_int, clock,reset)
BEGIN
    IF (reset = '1') THEN
        rco_int <= '1';
    ELSIF rising_edge(clock) THEN
        IF ((cnt_out_int = "11111111") or (cnt_out_int = "00000000")) THEN
            rco_int <= '1';
        ELSE
            rco_int <= '0';
        END IF;
    END IF;
END IF;
END PROCESS;

-- TOGGLE FLIP FLOP TO GENERATE THE PWM OUTPUT.

PROCESS (clock,rco_int,reset)
BEGIN
    IF (reset = '1') THEN
        pwm_int <= '0';
    ELSIF rising_edge(rco_int) THEN
        pwm_int <= NOT(pwm_int);
    ELSE
        pwm_int <= pwm_int;
    END IF;
END PROCESS;
pwm <= pwm_int;
END arch_pwm;

```

Donde INC y DEC, se implementan en el siguiente código (también, de Atmel):

--user_pkg.vhd

```

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
PACKAGE user_pkg IS
    function INC(X: STD_LOGIC_VECTOR) return STD_LOGIC_VECTOR;
    function DEC(X: STD_LOGIC_VECTOR) return STD_LOGIC_VECTOR;
END user_pkg ;
PACKAGE BODY user_pkg IS
    function INC(X: STD_LOGIC_VECTOR) return STD_LOGIC_VECTOR is
        variable XV: STD_LOGIC_VECTOR(X'LENGTH - 1 downto 0);
    begin
        XV := X;

```



```

        for I in 0 to XV'HIGH LOOP
        if XV(I) = '0' then
        XV(I) := '1';
exit;
        else XV(I) := '0';
end if;
end loop;
return XV;
end INC;
function DEC(X: STD_LOGIC_VECTOR) return STD_LOGIC_VECTOR is
variable XV: STD_LOGIC_VECTOR(X'LENGTH - 1 downto 0);
begin
XV := X;
for I in 0 to XV'HIGH LOOP
    if XV(I) = '1' then
        XV(I) := '0';
        exit;
    else XV(I) := '1';
    end if;
end loop;
return XV;
end DEC;
END user_pkg;

```

Una vez obtenida la generación de PWM en función de los datos de entrada, hay que controlar el puente H.

Dependiendo del valor de las entradas de los sensores Hall, hay que elegir que transistores se van a activar, y que transistor va a recibir los pulsos de PWM. La activación/desactivación de cada transistor, se realiza según el siguiente diagrama de estados:

Estado	Q1	Q2	Q3	Q4	Q5	Q6
0	1	0	0	PwmSignal	0	0
1	1	0	0	0	0	PwmSignal
2	0	0	1	0	0	PwmSignal
3	0	PwmSignal	1	0	0	0
4	0	PwmSignal	0	0	1	0
5	0	0	0	PwmSignal	1	0

El código es el siguiente:

--Driver motor fpga

```

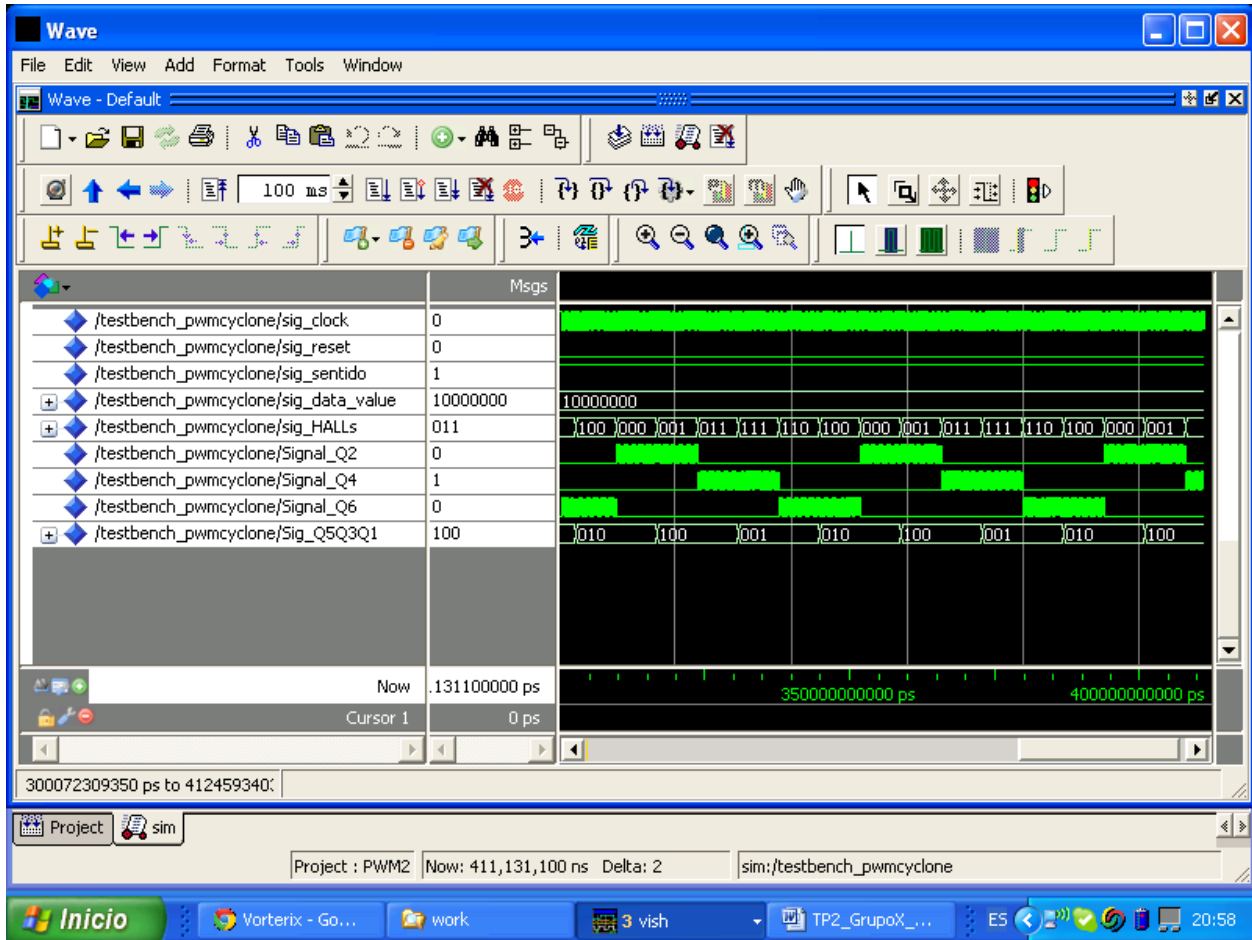
library IEEE;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY driver_motor_fpga IS
PORT ( ClockDriver :in STD_LOGIC;

```

```
ResetDriver :in STD_LOGIC;
SentidoGiro :in STD_LOGIC;
DataDriver :in std_logic_vector(7 downto 0);
SensoresHall :in std_logic_vector(2 downto 0);
    Q5Q3Q1 :out std_logic_vector(2 downto 0);
    Q2 :out STD_LOGIC;
    Q4 :out STD_LOGIC;
    Q6 :out STD_LOGIC
);
END driver_motor_fpga;
ARCHITECTURE Arch_Driver_motor_fpga OF driver_motor_fpga IS
--Este componente, es el definido en pwm_fpga.vhd del ejemplo hecho por
ATMEL
COMPONENT pwm_fpga
PORT (clock: IN STD_LOGIC;
reset: IN STD_LOGIC;
data_value: IN std_logic_vector(7 downto 0);
pwm: OUT STD_LOGIC);
END COMPONENT;
SIGNAL PwmSignal: STD_LOGIC;
BEGIN
-- Conexion del modulo pwm
ModuloPWM: pwm_fpga
PORT MAP(
clock => ClockDriver,
reset => ResetDriver,
data_value => DataDriver,
pwm => PwmSignal
);
PROCESS(SensoresHall,SentidoGiro, PwmSignal)
BEGIN
IF(SentidoGiro = '0') THEN --En caso de que gire en sentido horario
CASE SensoresHall IS
WHEN "001" =>
    Q5Q3Q1 <= "001";
    Q6 <= PwmSignal;
    Q4 <= '0';
    Q2 <= '0';
WHEN "000" =>
    Q5Q3Q1 <= "001";
    Q6 <= '0';
    Q4 <= PwmSignal;
    Q2 <= '0';
WHEN "100" =>
    Q5Q3Q1 <= "100";
    Q6 <= '0';
    Q4 <= PwmSignal;
    Q2 <= '0';
WHEN "110" =>
    Q5Q3Q1 <= "100";
```

```
        Q6 <= '0';
        Q4 <= '0';
        Q2 <= PwmSignal;
    WHEN "111" =>
        Q5Q3Q1 <= "010";
        Q6 <= '0';
        Q4 <= '0';
        Q2 <= PwmSignal;
    WHEN "011" =>
        Q5Q3Q1 <= "010";
        Q6 <= PwmSignal;
        Q4 <= '0';
        Q2 <= '0';
    WHEN OTHERS => NULL;
END CASE;
END IF;
IF(SentidoGiro = '1') THEN --En caso de que gire en sentido antihorario
CASE SensoresHall IS
    WHEN "011" =>
        Q5Q3Q1 <= "100";
        Q6 <= '0';
        Q4 <= PwmSignal;
        Q2 <= '0';
    WHEN "111" =>
        Q5Q3Q1 <= "001";
        Q6 <= '0';
        Q4 <= PwmSignal;
        Q2 <= '0';
    WHEN "110" =>
        Q5Q3Q1 <= "001";
        Q6 <= PwmSignal;
        Q4 <= '0';
        Q2 <= '0';
    WHEN "100" =>
        Q5Q3Q1 <= "010";
        Q6 <= PwmSignal;
        Q4 <= '0';
        Q2 <= '0';
    WHEN "000" =>
        Q5Q3Q1 <= "010";
        Q6 <= '0';
        Q4 <= '0';
        Q2 <= PwmSignal;
    WHEN "001" =>
        Q5Q3Q1 <= "100";
        Q6 <= '0';
        Q4 <= '0';
        Q2 <= PwmSignal;
    WHEN OTHERS => NULL;
END CASE;
```

```
END IF;  
END PROCESS;  
END ARCHITECTURE Arch_Driver_motor_fpga;
```



Compilador

Introducción

Un compilador es un programa que se encarga de buscar y ejecutar instrucciones que correspondan con un lenguaje determinado.

Mediante nuestro compilador, podemos ejecutar instrucciones de movimiento de la plataforma, de una manera más simple para el usuario, con un código de más alto nivel.



PARSER Y SCANNER

Para cumplir con el objetivo de nuestro compilador debemos usar dos programas. Un parser y un scanner.

El parser es un programa que recibe una serie de símbolos o TOKENS y analiza su sintaxis. Además la hace coincidir con una estructura previamente definida y en base a eso ejecuta una serie de comandos.

Escribir un parser es una tarea compleja, pero por suerte no hay que realizarla desde cero.

Para esto utilizamos un software que se llama Bison. Este programa toma un archivo que explicaremos más adelante y genera como salida otro archivo .c que luego compilaremos con el GNU/GCC.

El scanner es un programa que recibe una serie de palabras, y a través del uso de expresiones regulares las convierte en tokens. Por ejemplo:

Entrada	Token
Números del 0 al 9	DIGITO
Palabra “movq”	FUNCIONMOVQ

Su producto de salida es la entrada del parser. Para generar el código del parser utilizamos un software llamado Flex.

Ambos, el Flex y el Bison son programas de código abierto, que se pueden instalar en cualquier distribución de GNU/Linux.

El Bison proviene de un soft anterior llamado Yacc y el Flex de un soft llamado Lex.

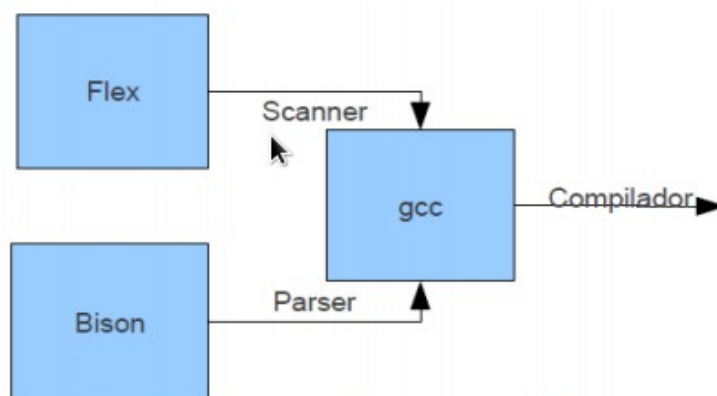


Ilustración 2: Arquitectura del compilador implementado

INSTALACIÓN

En cualquier distribución compatible con Debian, tipeamos en una consola las siguientes

líneas para instalar el Flex y el Bison

```
sudo apt- get install flex
```

```
sudo apt- get install bison
```

ARCHIVOS FLEX

El formato de un archivo de Flex es el siguiente:

Declaraciones en C y del scanner (includes, defines, etc.)

Expresiones regulares y Tokens y los archivos se compilan de la siguiente forma:

```
flex nombearchivo.l
```

Normalmente los archivos de Flex son de extensión .l, pues su formato original se llama Lex.

La salida del flex es, como se dijo anteriormente un archivo .c con su archivo .h asociado.

ARCHIVOS BISON

Los archivos de Bison tienen la siguiente estructura:

- Declaraciones en C y del parser
- Reglas gramáticas y acciones
- Subrutinas en C

Para generar un archivo de salida con el Bison, debemos ejecutar la siguiente línea de comando:

```
bison -d nombearchivo.y
```

y tendremos entonces un archivo .c y su archivo .h asociado.

La extensión .y proviene del formato Yacc.

MAKEFILE

También consideramos apropiado incluir en el trabajo un Makefile, archivo que se utiliza con el comando make de GNU/Linux para generar el compilador.

```
.COMP : clean
```

```
all: flex bison source

source:
    gcc -lm -o robotica *.c

flex: *.l
    flex *.l

bison: *.y
    bison -d *.y

clean :
    rm -f *.c *.h robotica *tab*
```

COMPILADOR

Nuestro compilador sigue el formato de instrucciones del lenguaje C, al que ya estamos acostumbrados.

Implementa 1 función principal:

movq().

Toma los valores de las 5 articulaciones del robot y calcula la posición del actuador final en un sistema cuyo origen de coordenadas se encuentra en la base del robot.

Además implementamos la función delay(ms) que genera una demora de tiempo en milisegundos y la función help que muestra un menú de ayuda.

Mostramos a continuación una captura de pantalla del proceso completo de utilización, desde la compilación al uso


```

juan@juan: ~/Documentos/UTN/Robotica/compilador
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
make
flex *.l
bison -d *.y
gcc -lm -o robotica *.c
juan@juan:~/Documentos/UTN/Robotica/compilador$ ./robotica
help
Funciones soportadas:

    movq(q1,q2,q3,q4,q5);:
        Datos los 4 angulos de las articulaciones, en grados, indica la
        posición X Y Z del robot

    delay(ms):
        Detiene la ejecucion del programa durante un tiempo expresado en
        ms

    exit:
        Sale del compilador
movq(10,30,40,50,5);
q1: 10.0 q2: 30.0 q3: 40.0 q4: 50.0 q5: 5.0=> x: 1504.61 y: 1540.44 z: 2425.58
delay(10);
Sleeping for 10 ms...
exit();
Gracias por utilizar este compilador

```

A continuación listamos el código de los archivos para generar el parser y el scanner:

/robotica.l

```

%{
#include <stdio.h>
#include "robotica.tab.h"
extern YYSTYPE yylval;
}%
digit [0-9]
%%

-?{digit}+  {
    yylval.entero=atoi(yytext);
    return(INTNUM);
}

-?{digit}+".{digit}*  {
    yylval.flotante=atof(yytext);
    return(FLOATNUM);
};

movq return MOVQFUNC;
delay return DELAYFUNC;
help return HELPFUNC;
exit return EXITFUNC;

```

```
(" return OPAR;

)" return CPAR;

"," return COMA;

";" return SEMICOLON;

"\n" { return (yytext[0]);}

[ \t]+ /* ignore whitespace */;

%%
```

/robotica.y

```
{

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#define L1x 320
#define L1z 780
#define L2 1280
#define L3 0
#define L4 1142
#define L5 500
#define PI 3.14

//Variables que uso despues, van aca o en el main?
float q1deg,q2deg,q3deg,q4deg,q5deg;
//float qi1,qi2,qi3,qi4,qf1,qf2,qf3,qf4;
void yyerror(const char *str)
{

    fprintf(stderr,"error: %s\n",str);
}

int yywrap()

{
return 1;
}

main()
{
    yyparse();
}

%}
```

```

%union{
int entero;
float flotante;
}

%token <entero>INTNUM
%token <flotante>FLOATNUM
%token MOVQFUNC DELAYFUNC HELPFUNC EXITFUNC OPAR CPAR SEMICOLON COMA
%%
commands: /* empty */
        | commands command
        ;

command: '\n'
|
movq
|
delay
|
help
|
exit
;

movq:
MOVQFUNC OPAR INTNUM COMA INTNUM COMA INTNUM COMA INTNUM COMA INTNUM CPAR SEMICOLON
{
float x,y,z;
float q1,q2,q3,q4,q5;
q1deg=$3;
q2deg=$5;
q3deg=$7;
q4deg=$9;
q5deg=$11;
q1=q1deg*PI/180;
q2=q2deg*PI/180;
q3=q3deg*PI/180;
q4=q4deg*PI/180;
q5=q5deg*PI/180;

x =
cos(q1)*(L1x  +  L3*cos(q2  +  q3)  +  L2*cos(q2))  -  L4*sin(q1)*sin(q4)  -
L5*cos(q5)*(sin(q1)*sin(q4)  -  cos(q2  +  q3)*cos(q1)*cos(q4))  -  L5*sin(q2  +
q3)*cos(q1)*sin(q5) + L4*cos(q2 + q3)*cos(q1)*cos(q4);

y  =  sin(q1)*(L1x  +  L3*cos(q2  +  q3)  +  L2*cos(q2))  +  L4*cos(q1)*sin(q4)  +
L5*cos(q5)*(cos(q1)*sin(q4)  +  cos(q2  +  q3)*cos(q4)*sin(q1))  +  L4*cos(q2  +
q3)*cos(q4)*sin(q1) - L5*sin(q2 + q3)*sin(q1)*sin(q5);

z = L1z + L3*sin(q2 + q3) + L2*sin(q2) + L4*sin(q2 + q3)*cos(q4) + L5*cos(q2 + q3)*sin(q5)
+ L5*sin(q2 + q3)*cos(q4)*cos(q5);

printf("q1: %.1f q2: %.1f q3: %.1f q4: %.1f q5: %.1f=> x: %.2f y: %.2f z:

```

```
%f\n",q1deg,q2deg,q3deg,q4deg,q5deg,x,y,z);  
};  
  
exit:  
EXITFUNC  
{  
printf("Gracias por utilizar este compilador\n");  
exit(0);  
};  
  
delay:  
DELAYFUNC OPAR INTNUM CPAR SEMICOLON  
{  
int sleepTime=$3;  
printf("Sleeping for %d ms...",sleepTime);  
usleep(sleepTime*1000);  
};  
  
help:  
HELPFUNC  
{  
printf("Funciones soportadas: \n\n\tmovq(q1,q2,q3,q4,q5);\n\n\tDatos los 4 angulos de las articulaciones, en grados, indica la posición X Y Z del robot\n");  
  
printf("\n\tdelay(ms):\n\n\tDetiene la ejecucion del programa durante un tiempo expresado en ms\n");  
  
printf("\n\texit:\n\n\tSale del compilador\n");  
};  
  
%%
```