Robótica Industrial - Práctica 1

1. Selección de un robot de 6 GDL

Para la realización de esta práctica se ha seleccionado el robot **LR Mate 200iD** de **Fanuc** por ser el modelo estándar de la serie LR Mate de Robots Articulados.

Como se puede observar en la siguiente figura, se trata de un robot de seis grados de libertad de configuración RRR-RRR, lo que implica que se puede obtener hasta 32 soluciones para llegar a un punto determinado (sin tener en cuenta obstáculos).

Ilustración 1 – Robot LR Mate 200iD de Fanuc

Robot			Controlador							Max. capacidad de carga en la Muñeca (kg)	Alcance (mm)	Ejes	Repetibilidad (mm)	Peso mecánico (kg)	Rango de Movimiento (°)						Velocidad Máxima (°/s)*12						J4 Momento/ Inercia (Nm/kgm²)	J5 Momento/ Inercia (Nm/kgm²)	J6 Momento/ Inercia (Nm/kgm²)	Consumo medio (kW)	Protección	
Serie	Versión	Tipo	Versión: R-30iB	R-30iB Plus	Tipo de Armario: Compact	Open air	Mate	A	B						J1	J2	J3	J4	J5	J6	J1	J2	J3	J4	J5	J6					Cuerpo estándar/ Opcional	Muñeca y brazo J3 estándar/ Opcional
LR Mate 200	iD	4SH	●	●	-	o	●	-	-	4	550	5	± 0.013**	19	360	230	402	240	720	-	460	460	520	560	1500	-	8.86/0.2	4.0/0.046 (5.5/0.083)	-	0.5	IP67	IP67
LR Mate 200	iD	4S	●	●	-	o	●	-	-	4	550	6	± 0.01**	20	360	230	402	380	240	720	460	460	520	560	560	900	8.86/0.2	8.86/0.2	4.9/0.067	0.5	IP67	IP67
LR Mate 200	iD	4SC	●	●	-	o	●	-	-	4	550	6	± 0.013**	20	360	230	402	380	236	720	460	460	520	560	560	900	8.86/0.2	8.86/0.2	4.9/0.067	0.5	IP67	IP67
LR Mate 200	iD	7H	●	●	-	o	●	-	-	7	717	5	± 0.018**	24	360	245	420	250	720	-	450	380	520	545	1500	-	16.6/0.47	4.0/0.046 (5.5/0.15)	-	0.5	IP67/IP69K	IP67/IP69K
LR Mate 200	iD	7C	●	●	-	o	●	-	-	7	717	6	± 0.018**	25	360	245	420	380	250	720	450	380	520	550	545	1000	16.6/0.47	16.6/0.47	9.4/0.15	0.5	IP67	IP67
LR Mate 200	iD	7WP	●	●	-	-	●	-	-	7	717	6	± 0.018**	25	360	245	420	380	250	720	450	380	520	550	545	1000	16.6/0.47	16.6/0.47	9.4/0.15	0.5	IP67/IP69K	IP67/IP69K
LR Mate 200	iD		●	●	-	o	●	-	-	7	717	6	± 0.01**	25	360	245	420	380	250	720	450	380	520	550	545	1000	16.6/0.47	16.6/0.47	9.4/0.15	0.5	IP67/IP69K	IP67/IP69K
LR Mate 200	iD	7L	●	●	-	o	●	-	-	7	911	6	± 0.01**	27	360	245	430	380	250	720	370	310	410	550	545	1000	16.6/0.47	16.6/0.47	9.4/0.15	0.5	IP67/IP69K	IP67/IP69K
LR Mate 200	iD	7LC	●	●	-	o	●	-	-	7	911	6	± 0.018**	27	360	245	430	380	250	720	370	310	410	550	545	1000	16.6/0.47	16.6/0.47	9.4/0.15	0.5	IP67	IP67
LR Mate 200	iD	14L		●	-	O	●	-	-	14	911	6	± 0.01**	27	360	245	430	380	250	720	120	61	58	400	240	400	31.0/0.66	31.0/0.66	13.4/0.30	0.5	IP67/IP69K	IP67/IP69K

Ilustración 2 - Tabla de especificaciones del robot (en morado).

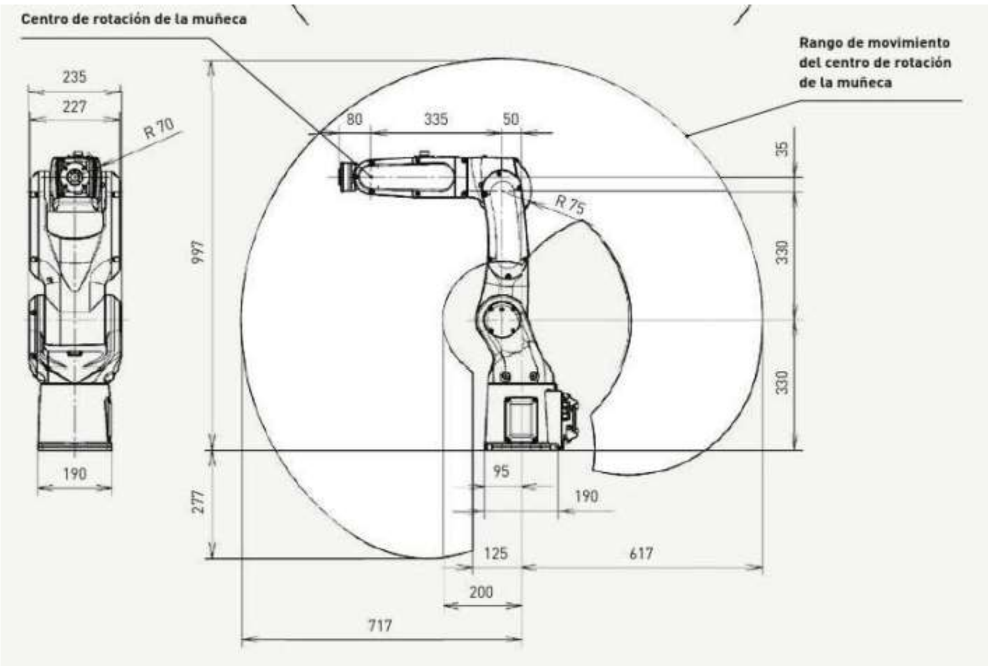


Ilustración 3 - Rango de movimiento (distancias en mm)

Para la realización de la práctica **se pide**:

- El dibujo esquemático del robot, indicando los sistemas de referencia asociados a cada articulación según las reglas de Denavit-Hartenberg sobre el robot.
- Rellenar la tabla de parámetros de Denavit-Hartenberg en base a los parámetros del robot seleccionado y la ubicación de los sistemas de referencia.

DIBUJO ESQUEMÁTICO

Las dos primeras imágenes muestran la numeración de los elementos junto a la posición de los ejes en ausencia de los sistemas de referencia desde un punto de vista isométrico y lateral.

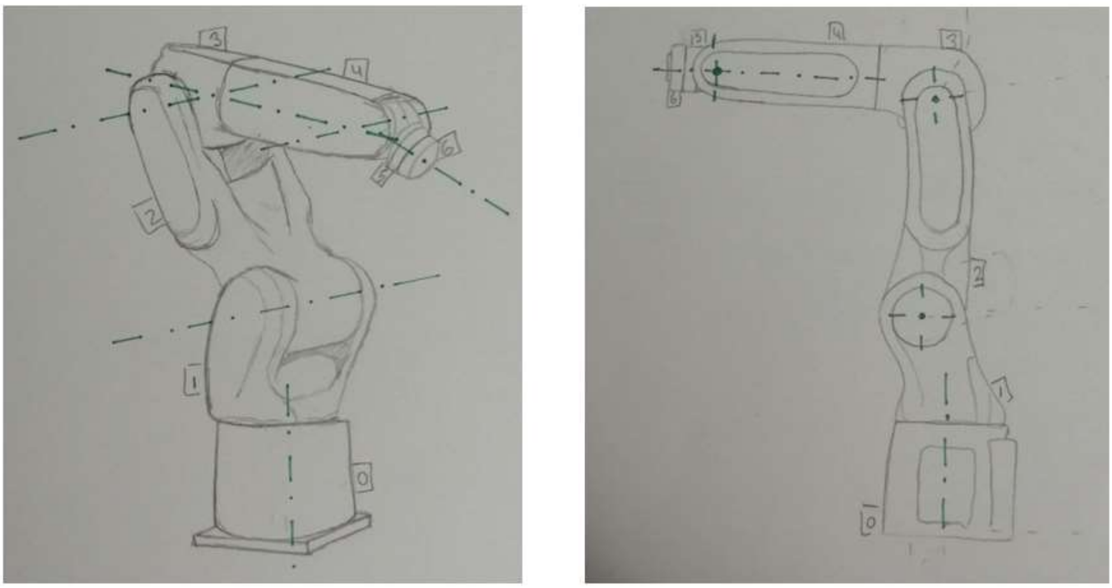


Ilustración 4 - Vista Isométrica y Lateral. Posicionamiento de los ejes.

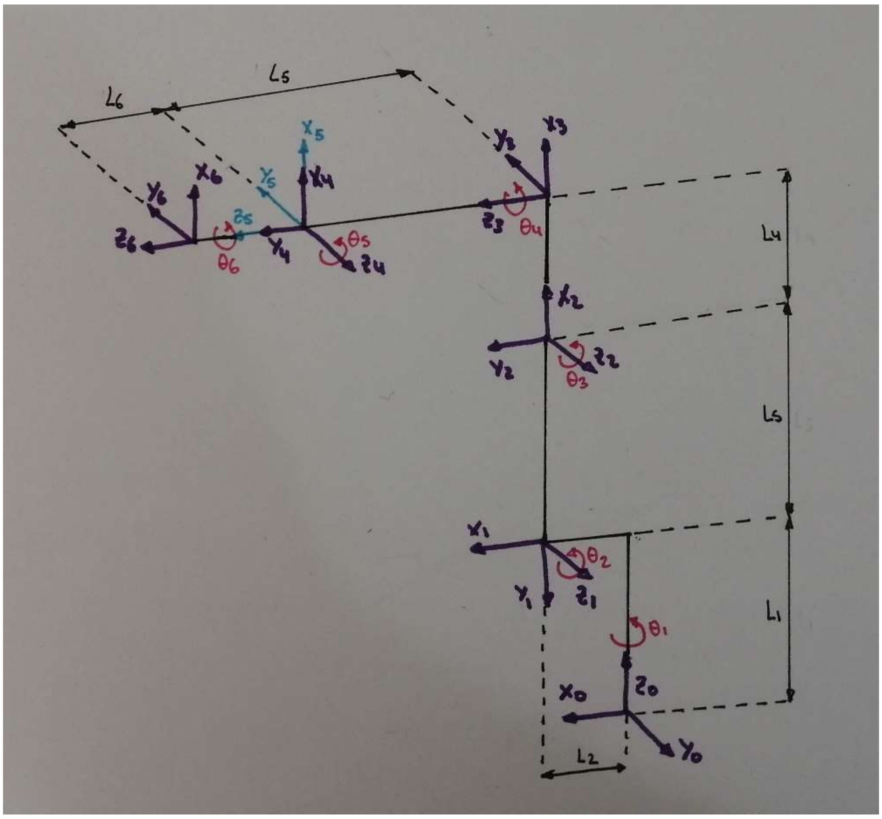


Ilustración 5 - Sistemas de referencia del robot.

TALBLA DE DENAVIT-HARTENBERG

La siguiente tabla se forma a partir de los sistemas de referencia calculados anteriormente (ilustración X), y observando las distancias entre ellos en la figura 3.

Nº Elemento i	$\boldsymbol{\theta_i}$	$\boldsymbol{d_i}$	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$
1	θ_1	L1 = 0.330m	L2 = 0.050m	-90
2	$\theta_2 - 90$	0	L3 = 0.330m	0
3	θ_3	0	L4 = 0.035m	-90
4	θ_4	L5 = 0.335m	0	90
5	θ_5	0	0	-90
6	θ_6	L6 = 0.080m	0	0

Tabla 1 – Parámetros de Denavit-Hartenberg para el robot LR Mate 200iD.

2. Estudio de la aplicación EduBot con el robot PUMA

Tal y como se dice en el enunciado de la práctica, se realizará el estudio de robot PUMA (figura inferior) para familiarizarse con el entorno EduBoT.

Ilustración 6 - Robot PUMA.

Para ello se han seguido los siguientes pasos, los cuales se encuentran de forma detallada en el enunciado de la práctica:

1. **Carga del robot.**
2. **Problema cinemático directo del robot.**
3. **Problema cinemático inverso del robot usando el procedimiento genérico recursivo.**
4. **Problema cinemático inverso del robot usando una solución analítica.**
5. **Coste computacional del modelo cinemático.**

A continuación, se muestra el *scriptlive*, "EduBotPUMA", de Matlab utilizado, el cual combina la práctica con el código utilizado.

Practica 1 - Robot PUMA

NOTA: Se han eliminado los acentos y algunos signos en los textos debido a a que la impresion no reconocia los simbolos.

NOTA: Como para cualquier comando de MATLAB se ha utilizado **help** para entender el funcionamiento de los comandos dados en el enunciado de la practica para su desarrollo.

Table of Contents

1. Carga del robot

Tras iniciar Matlab y cargar EduBot ejecutamos:

```
puma560
```

Se ha creado la variable p560, que contiene los datos del modelo del PUMA 560. Estos incluyen los parametos de la tabla D-H y 3 vectores de dimension 6 asociados a diferentes configuraciones aticulares.

- **qr:** posicion de reposo. El brazo del robot se encuentra estirado en vertical.
- **qz:** posicion de cero. Cada articulacion se encuentra en su origen.
- **qstrech:** posicion encogida. El brazo del robot se encuentra estirado en horizontal, con la primera articulacion a 90 grados.

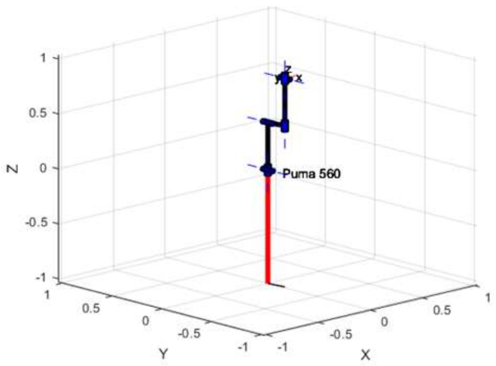
Usando **plot** se pueden visualizar diferentes configuraciones del robot.

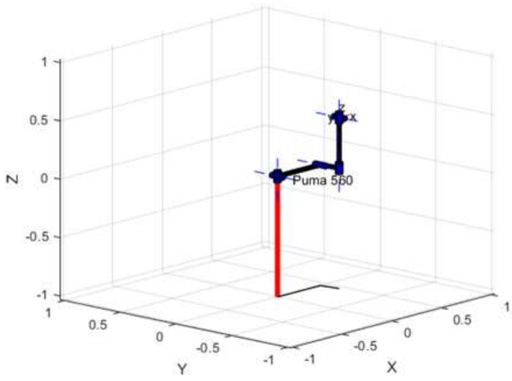
Al utilizar este "livescript" a la derecha se mostraran los resultados de ejecucion de cada codigo. Para los plots, sera necesario abrir en una nueva ventana (si seleccionas arriba a la derecha al pinchar en la imagen) la opcion se "sacar la imagen". A partir de ahi, se puede manipular la imagen 3D para posicionar los ejes de forma que se aprecie mejor las posiciones de las articulaciones y elementos del robot.

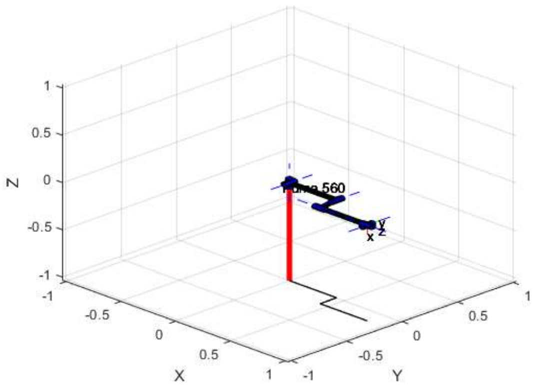
NOTA: Las figuras se solapan, para poder visualizarlas descomentar la que se quiera visualizar o comentar la que este descomentada. Ademas de esta forma no aparecen las lineas de codigo amarillo (warnings) en la impresion.

```
% plot(p560,qr)

% plot(p560,qz)

% plot(p560,qstretch)
```

A continuacion se muestran las imagenes obtenidas al ejecutar cada plot respectivamente.







2. Problema cinematico directo del robot

Mediante **fkine** se obtiene la matriz de transformacion T_1 asociada a la configuracion articular q_1

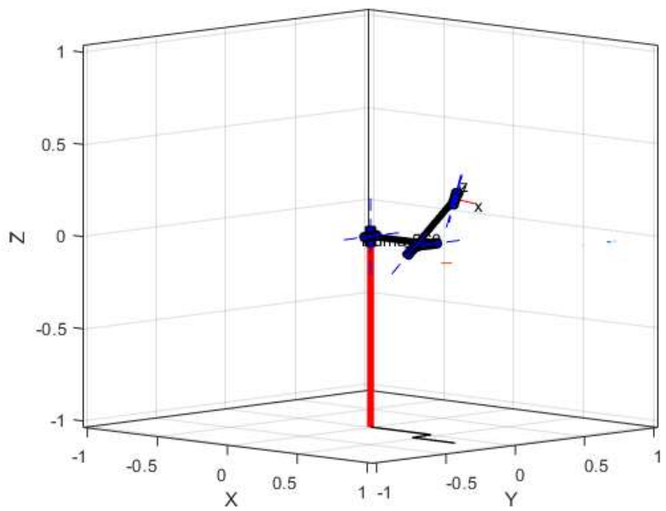
```
q1 = [0 0 -pi/4 pi/4 pi/10 0]; % (rad)
fkine(p560,q1);
```

Se pide la visualizacion del robot en dicha posicion y extraer de la matriz de transformacion T_1 la posicion TCP (punto de la herramienta) en el espacio cartesiano y la orientacion del sistema de referencia movil usando la notacion Roll/Pitch/Yaw.

```
T1 = fkine(p560,q1)
```

```
T1 = 4�4
    0.6940   -0.5000    0.5180    0.7515
    0.6725    0.7071   -0.2185   -0.1500
   -0.2570    0.5000    0.8270    0.2910
         0         0         0    1.0000
```

```
% plot(p560,q1)
```



Posicion TCP

```
posicionT1 = T1(1:3,4)
```

```
posicionT1 = 3�1
    0.7515
   -0.1500
    0.2910
```

Orientacion del sistema movil usando la notacion Roll/Pitch/Yaw (rad)

```
R = atan (T1(1,3)/T1(2,3))
```

```
R = -1.1716
```

```
P = atan (sqrt((T1(1,3))^2+(T1(2,3))^2)/T1(3,3))
```

```
P = 0.5970
```

```
Y = atan (T1(3,1)/T1(3,2))
```

```
Y = -0.4748
```

3. Problema cinematico inverso del robot usando el procedimiento generico recursivo.

Mediante el comando **ikine,** determinar la condiguracion articular q_2 asociada a al matriz de transformacion T_2

```
T2 = [ 0   0   0   0.8636;
       0   1   0  -0.1501;
      -1   0   0  -0.0203;
       0   0   0   1]
```

```
T2 = 4�4
         0         0         0    0.8636
         0    1.0000         0   -0.1501
   -1.0000         0         0   -0.0203
         0         0         0    1.0000
```

a) Dado que **ikine** utiliza un metodo numerico general para su resolucion, es necesario utilizar una semilla o configuracion q_0 de partida para el calculo ue este suficientemente cerca del punto final.

Con el fin de ver el efecto de la semilla inicial, se calculara la configuracion articular que permite que el robot presetne una localizacion definida por T_2, en vase a dos posible semillas de partida

- La semilla $q_{0a} = [0\ 0\ 0\ 0\ 0\ 0]$ (rad), que es la que **ikine** usa por defecto si se omite la semilla en el comando

```
q0a = [0 0 0 0 0 0]
```

```
q0a = 1�6
     0     0     0     0     0     0
```

```
q2a = ikine(p560,T2)
```

```
q2a = 1�6
   -0.0001   -0.0466   -1.4777    0.0012   -0.0465   -0.0012
```

```
% que es equivalente a
q2a = ikine(p560,T2,q0a)
```

```
q2a = 1�6
   -0.0001   -0.0466   -1.4777    0.0012   -0.0465   -0.0012
```

- La semilla $q_{0b} = [0\ 0\ 0\ -\pi\ \ 0\ \ \pi]$ (rad)

```
q0b = [0 0 0 -pi 0 pi]
```

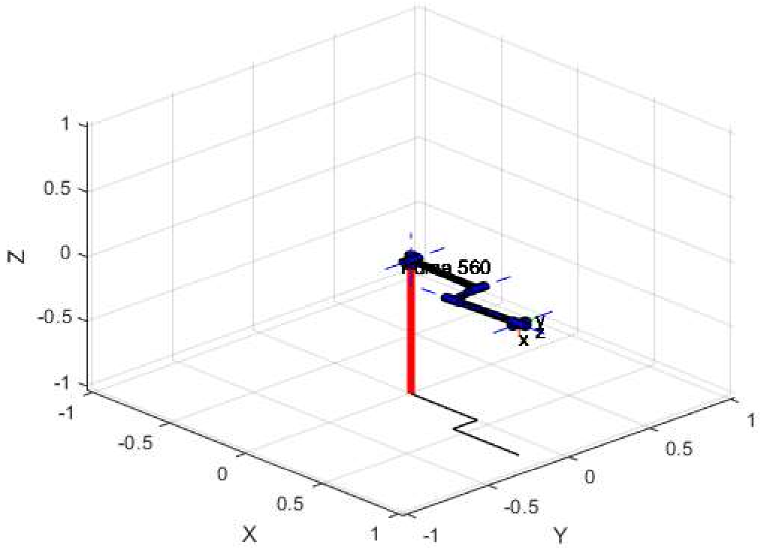
```
q0b = 1�6
         0         0         0   -3.1416         0    3.1416
```

```
q2b = ikine(p560,T2,q0b)
```

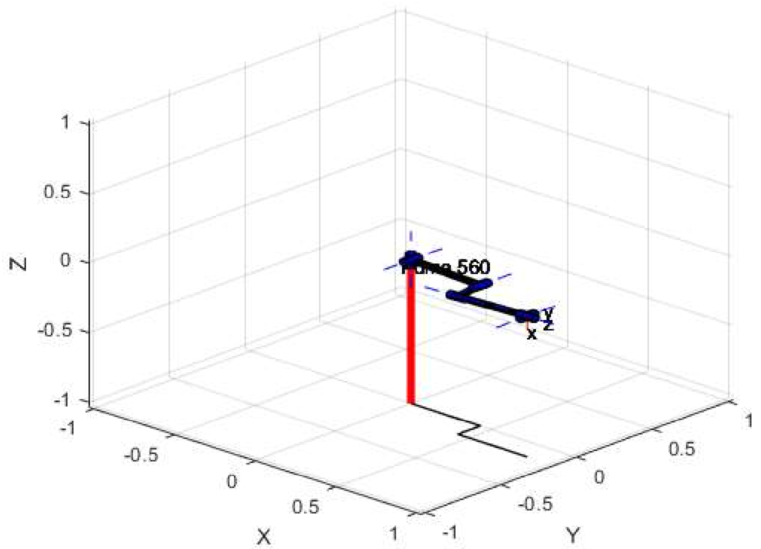
```
q2b = 1�6
   -0.0001   -0.0466   -1.4777   -3.1403    0.0465    3.1403
```

b) Represente las configuraciones q_{2a} y q_{2b} mediante el comando **plot.** �Que observa?

```
%    plot(p560,q2a)
```




```
% plot(p560,q2b)
```



Observacion

Se puede observar que presentan la misma configuracion. Esto se debe a que 0 rad equivale a -3,14 rad, y para la diferencia de signo entre la coordenada articular 5 es la misma posicion pero en sentido contrario.

c) Use **fkine** para determinar la localizacion del robt asociadas a las configuraciones q_{2a} y q_{2b}, definidas por las matrices de transformacion T_{2a} y T_{2b}. �Que conclusion extrae?

```
T2a = fkine(p560,q2a)
```

```
T2a = 4�4
    0.0000   -0.0000    1.0000    0.8636
         0    1.0000    0.0000   -0.1501
   -1.0000    0.0000    0.0000   -0.0203
         0         0         0    1.0000
```

```
T2b = fkine(p560,q2b)
```

```
T2b = 4�4
    0.0000   -0.0000    1.0000    0.8636
    0.0000    1.0000    0.0000   -0.1501
   -1.0000    0.0000    0.0000   -0.0203
         0         0         0    1.0000
```

Conclusion

Al final con el problema cinematico inverso pueden haber mas de una solucion. Se ha partido de dos semillas distintas, peroal final la posicion de la punta es la misma.

4. Problema cinematico inverso del robot usando una solucion analitica

Determinar, usando el comando **ikine560,** la configuracion articular q_2 asociada a la matriz de transformacion T_3

```
    T3 = [ 0   0   1   0.7580;
           0   1   0  -0.1500;
          -1   0   0  -0.0176;
           0   0   0   1]
```

```
T3 = 4�4
         0         0    1.0000    0.7580
         0    1.0000         0   -0.1500
   -1.0000         0         0   -0.0176
         0         0         0    1.0000
```

Este comando, **ikine560,** permite obtener la solucion analitica espec�fica del Puma 560. Mediante el se especifica la configuracion particular deseada.

Se pide cacular las soluciones para alcanzar la localizacion marcada por una matriz de transformacion T_3 con las siguientes configuraciones especificas:

- q_{3a} : Hombro izquierda, codo arriba, flip.
- q_{3b} : Hombro izquierda, codo abajo, flip.
- q_{3c} : Hombro derecha, codo abajo, noflip.

```
q3a = ikine560(p560,T3,'luf')
```

```
q3a = 1�6
    2.7508   -3.6189   -0.5233    2.4896    0.6788    0.5361
```

```
q3b = ikine560(p560,T3,'ldf')
```

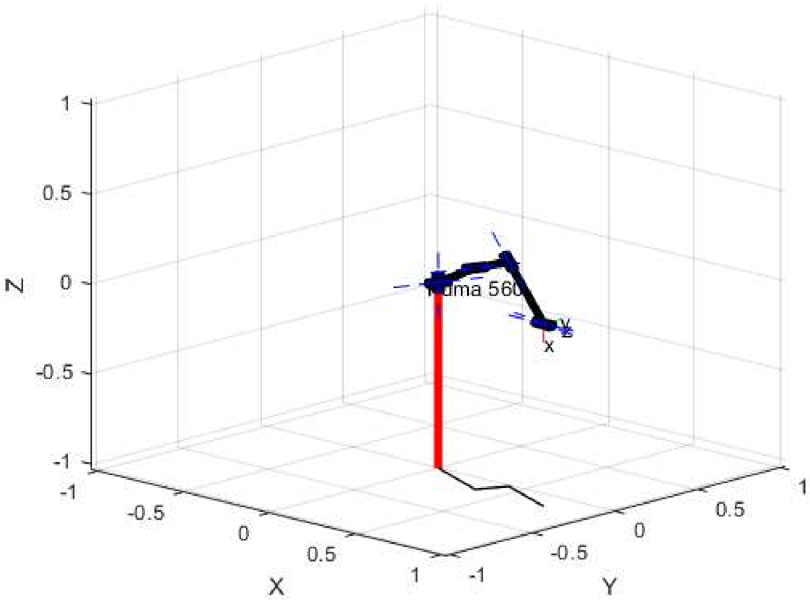
```
q3b = 1�6
    2.7508   -2.6178   -2.5243    0.7798    0.5725    2.4482
```

```
q3c = ikine560(p560,T3,'rdn')
```

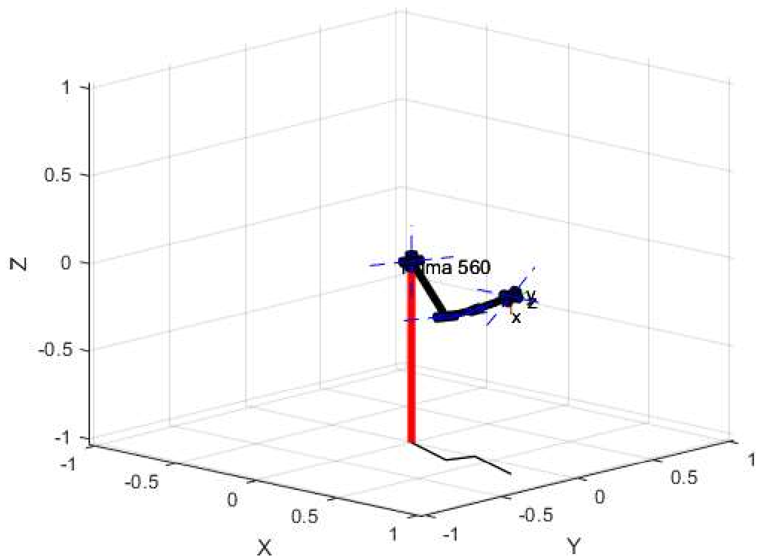
```
q3c = 1�6
    0.0001   -0.5238   -0.5233   -0.0001   -0.5237    0.0001
```

Mediante el comando **plot** se visualizaran las tres configuraciones. Posteriormente se analizaran las diferencias.

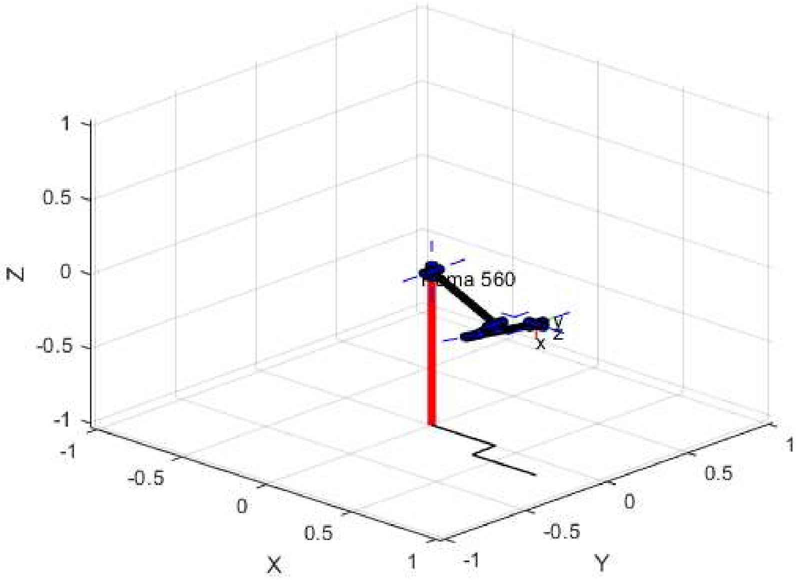
```
% plot(p560,q3a)
```




```
% plot(p560,q3b)
```




```
% plot(p560,q3c)
```



Analisis

Se observa como la configuracion espeficiada en el codigo modifica la configuracion de las articulaciones, pero que la posicion final de la punta es la misma para los tres casos.

Mediante el comando **help ikine560** se puede observar en detalle las disntas condifuraciones posibles.

```
help ikine560
```

```
 ikine560 Inverse kinematics for Puma 560

        Q = ikine560(ROBOT, T, CONFIG)

  Solve the inverse kinematics of the Puma-like (spherical wristed)
  ROBOT whose end-effector pose is given by T.

  The optional third argument specifies the configuration of the arm
  the form of a string containing one or more of the configuration c
        'l' or 'r'      lefty/righty
        'u' or 'd'      elbow
        'n' or 'f'      wrist flip or noflip.

  The default configuration is 'lun'.

  REFERENCE:

  Inverse kinematics for a PUMA 560 based on the equations by Paul a
  From The International Journal of Robotics Research
  Vol. 5, No. 2, Summer 1986, p. 32-44


  AUTHOR:
  Robert Biro           gt2231a@prism.gatech.edu
  with Gary Von McMurray
```

5. Coste computacional del modilo cinematico

Se procede a comparar el coste computacional del calculo del modelo cinematico inverso usando el metodo numerico general y analitico. Para ello se utilizara la matriz de transformacion T_3.

```
fprintf("Metodo numerico general")
```

```
Metodo numerico general
```

```
tic
ikine(p560,T3)
```

```
ans = 1�6
    0.0001   -0.5238   -0.5233   -0.0001   -0.5237    0.0001
```

```
toc
```

```
Elapsed time is 0.030633 seconds.
```

```
fprintf("Metodo numerico analitico")
```

```
Metodo numerico analitico
```

```
tic
ikine560(p560,T3,'lun')
```

```
ans = 1�6
    2.7508   -3.6189   -0.5233   -0.6520   -0.6788   -2.6055
```

```
toc
```

```
Elapsed time is 0.013803 seconds.
```

Observarciones

Es mas rapida la solucion analitica. Esto se debe a que el programa tiene que realizar menos calculos, posiblemente porque es una funcion especifica para este robot.

3. Robot serie personal

En esta sección se generará un objeto robot personal en base a los datos geométricos del robot industrial seleccionado, en este caso el **LR Mate 200iD**, que se han calculado en el primer apartado.

Se han seguido los siguientes pasos, los cuales se encuentran de forma detallada en el enunciado de la práctica:

1. **Generación del objeto robot.**
2. **Problema cinemático directo e inverso utilizando EduBot**
3. **Programación del problema cinemático directo.**

A continuación, se muestra el *scriptlive* utilizado para la realización de este apartado, “EduBotPersonal”. En él, al igual que en aparado dos, se combina tanto la realización de la práctica como el código utilizado.

Practica 1 - Robot Serie Personal

NOTA: Se han elimiado los acentos debido a que la impresion pdf no los reconocia.

Table of Contents

1. Generacion del objeto robot

A partir del trabajo previo realizado, mediante la tabla de Denavit-Hartenberg se ha creado un objeto robot para poder ser usado y simulado en Edubot.

Nº Elemento i	θ_i	d_i	a_i	α_i
1	θ_1	L1 = 0.330m	L2 = 0.050m	-90
2	$\theta_2 - 90$	0	L3 = 0.330m	0
3	θ_3	0	L4 = 0.035m	-90
4	θ_4	L5 = 0.335m	0	90
5	θ_5	0	0	-90
6	θ_6	L6 = 0.080m	0	0

El comando **link** permite definir cada uno delos elementos del robot en base a la fila asociada de la tabla D-H.

L{i} = link ([$\alpha_i \quad a_i \quad \theta_i \quad d_i$])

Nota: Las dimensiones longitdinales se han de implementar en metos, y las angulares en radianes.

```
L{1} = link([-pi/2  0.050  0      0.330]);      % Elemento 1
L{2} = link([0      0.330  -pi/2  0.000]);      % Elemento 2
L{3} = link([-pi/2  0.035  0      0.000]);      % Elemento 3
L{4} = link([pi/2   0.000  0      0.335]);      % Elemento 4
L{5} = link([-pi/2  0.000  0      0.000]);      % Elemento 5
L{6} = link([0      0.000  0      0.080]);      % Elemento 6
```

Mediante el comando **robot** se ha generado el objeto robot para ser usado por **Edubot.**

```
mirobot = robot(L,'LRMate200iD')
```

```
mirobot =

LRMate200iD (6 axis, RRRRRR)
                grav = [0.00 0.00 9.81]         standard D&H parameters

        alpha    A       theta    D       R/P
        -1.5708 0.05     0        0.33     R        std
        0       0.33     -1.5708 0        R        std
        -1.5708 0.035    0        0        R        std
        1.5708  0        0        0.335    R        std
        -1.5708 0        0        0        R        std
        0       0        0        0.08     R        std
```

2. Problema cinematico directo e inverso utilizando Edubot

a) Partiendo de la congiguracion artiqular q_4, calcular la localizacion TCP del robot T_4 usando el comando **fkine** y visualizarla mediante **plot**

```
q4 = [0 -pi/6 -pi/6 0 -pi/6 pi]      % (rad)
```

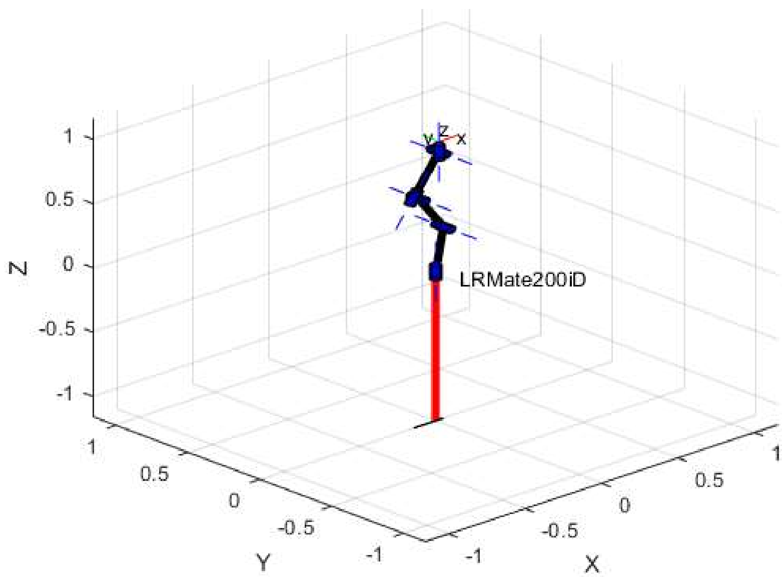
```
q4 = 1�6
         0   -0.5236   -0.5236         0   -0.5236    3.1416
```

```
T4 = fkine(mirobot,q4)
```

```
T4 = 4�4
    1.0000    0.0000    0.0000    0.0222
   -0.0000    1.0000    0.0000   -0.0000
   -0.0000   -0.0000    1.0000    1.0034
         0         0         0    1.0000
```

VIsualizacion

```
% plot(mirobot,q4)
```



b) Partiendo de T_4, usar la resolucion numerica general del problema cinematico inverso **ikine** para determinar la configuracion articular. �Se obtiene la misma confguracion de partida? �Es siempre asi? Probar al menos con dos configuraciones y semillas diferentes.

```
q4a = ikine(mirobot,T4)      % Utiliza la semilla q0a por defecto
```

```
q4a = 1�6
    6.2832    6.7131  -14.9762    6.2832    6.6922   -9.4248
```

Utilizamos la semilla creada para el PUMA, $q_{0b} = [0\ 0\ 0\ -\pi\ 0\ \pi]$ (rad)

```
q0b = [0 0 0 -pi 0 pi]       % semilla q0b
```

```
q0b = 1�6
         0         0         0   -3.1416         0    3.1416
```

```
q4b = ikine(mirobot,T4, q0b)
```

```
q4b = 1�6
   -3.1416    6.6263  -14.9513   -3.1416   -6.7542    3.1416
```

Nueva semilla $q_{0c} = [\pi \ \ 0 \ \ -\pi \ \ \ 0 \ \ \ 0 \ \ \ 0]$ (rad)

```
q0c = [pi 0 -pi 0 0 0]       % semilla q0c
```

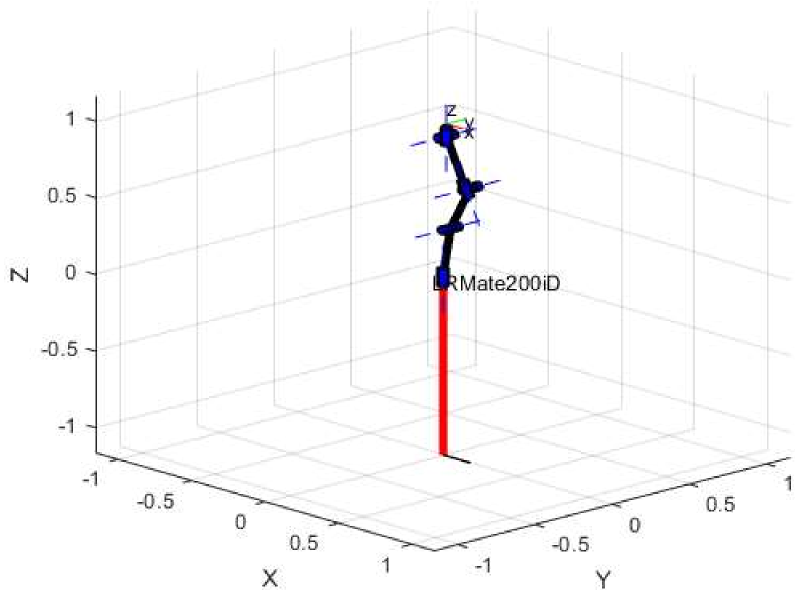
```
q0c = 1�6
    3.1416         0   -3.1416         0         0         0
```

```
q4c = ikine(mirobot,T4,q0c)
```

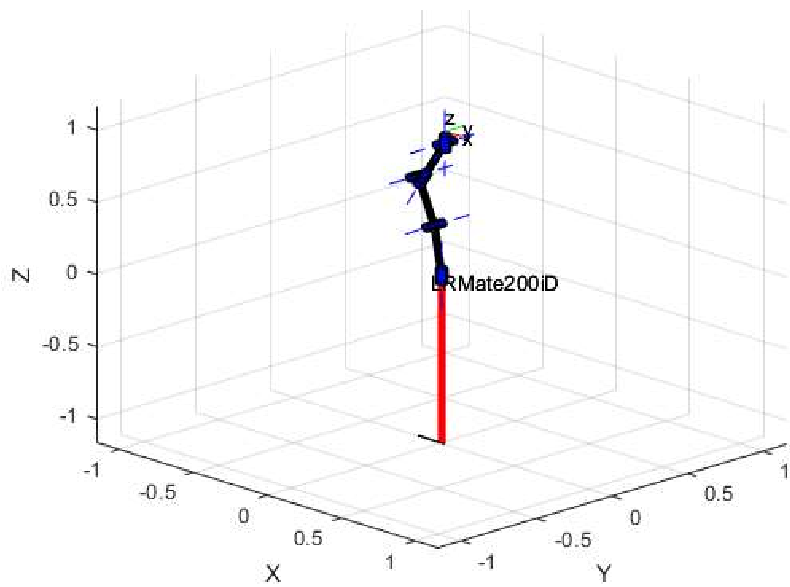
```
q4c = 1�6
    3.1416    0.3431   -2.3849   -0.0000    0.4710    0.0000
```

Representaciones

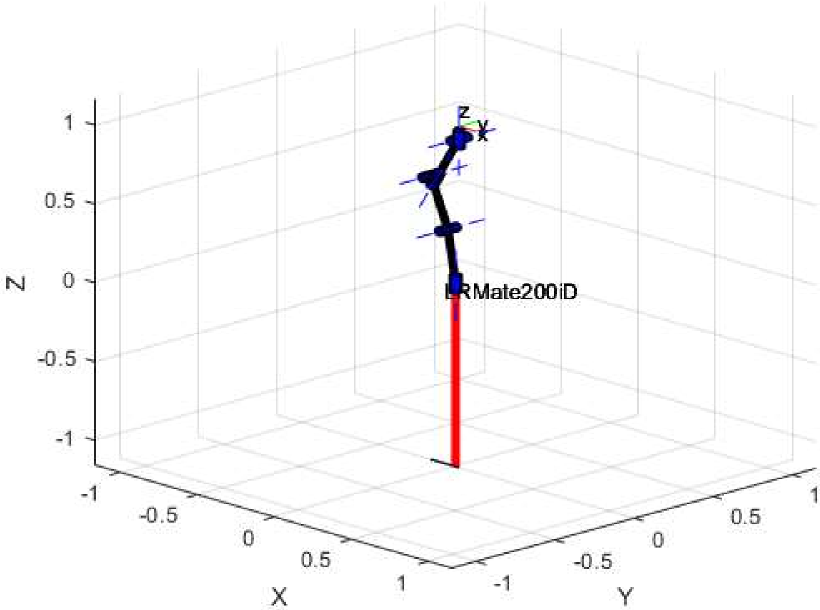
```
% plot(mirobot,q4a)
```




```
% plot(mirobot,q4b)
```




```
% plot(mirobot,q4c)
```



Conclusiones

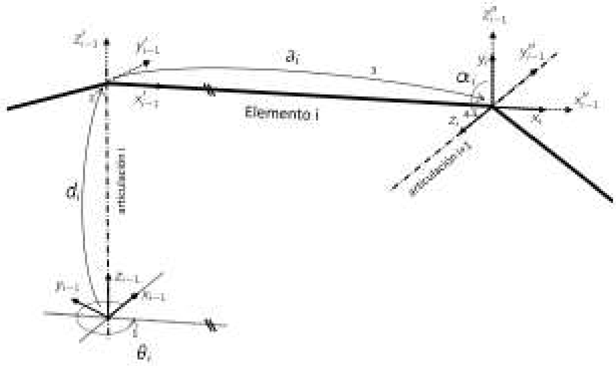
Como se puede observar la configuracion varia, pero la posicion de la punta es la misma en los tres casos.

3. Programacion del problema cinematico directo

En base a los parametros de Denavit-Hartenberg anteriormente calculados, se ha definido una funcion de matlab, **mi_fkine**, que devuelve la localizacion del TCP en forma de matriz de transformacion homogenea **T** ante cualquier configuracionarticular **q.**

```
T=mi_fkine(q);
```

Para definir esta funcion, se ha considerado las matrices de transformacion que conectan los sistemas de refrencia asociados a cada articulacion, segun el metodo de Denavit-Hartenberg.



$$
\begin{aligned}
{}^{i-1}_{\;\;i}\mathbf{T} &= R(z_{i-1},\theta_i)\,T(z_{i-1},d_i)\,T(x_i,a_i)\,R(x_i,\alpha_i)\\
&= \begin{bmatrix}
\cos\theta_i & -\cos\alpha_i\,\sin\theta_i & \sin\alpha_i\,\cos\theta_i & a_i\,\cos\theta_i\\
\sin\theta_i & \cos\alpha_i\,\cos\theta_i & -sin\alpha_i\,\cos\theta_i & a_i\,\sin\theta_i\\
0 & \sin\alpha_i & \cos\alpha_i & d_i\\
0 & 0 & 0 & 1
\end{bmatrix}
\end{aligned}
$$

Nota: Hay un error en la matriz T de la imagen superior, T(1,3) es producto de senos.

Esta matriz puede ser generada de forma sencilla a patir de los parametros de DH usando el siguiente comando

$$Ti=denavit\ (\theta_i,\ d_i,\ a_i,\ \alpha_i);$$

Se ha utilizado al funcion creada para comprobar si los resultados de los comandos **fkine** y el de **mi_fkine** son iguales ante diversas configuraciones articulares

```
q5 = [pi/3 pi 0 pi/2 -pi/2 0];

T5_fkine = fkine(mirobot,q5)
```

```
T5_fkine = 4�4
    0.5000    0.0000    0.8660   -0.0732
    0.8660   -0.0000   -0.5000   -0.2868
   -0.0000    1.0000   -0.0000   -0.0350
         0         0         0    1.0000
```

```
T5_mi_fkine = mi_fkine(q5)
```

```
T5_mi_fkine = 4�4
    0.5000    0.0000    0.8660   -0.0732
    0.8660   -0.0000   -0.5000   -0.2868
   -0.0000    1.0000   -0.0000    0.2600
         0         0         0    1.0000
```

Probamos con otra configuracion

```
q6  = [-pi pi/2 0 pi/6 pi -pi/2]
```

```
q6 = 1�6
   -3.1416    1.5708         0    0.5236    3.1416   -1.5708
```

```


T6_fkine = fkine(mirobot,q6)
```

```
T6_fkine = 4�4
   -0.5000    0.8660    0.0000   -0.4150
   -0.8660   -0.5000   -0.0000   -0.0000
   -0.0000   -0.0000    1.0000    0.0750
         0         0         0    1.0000
```

```
T6_mi_fkin = mi_fkine(q6)
```

```
T6_mi_fkin = 4�4
   -0.5000    0.8660    0.0000   -0.1200
   -0.8660   -0.5000   -0.0000   -0.0000
   -0.0000   -0.0000    1.0000    0.0750
         0         0         0    1.0000
```

Funcion

```
function M = mi_fkine(q)
T01 = denavit(q(1),     0.330,  0.050, -pi/2);
T12 = denavit(q(2)-pi/2,0,      0.035,     0);
T23 = denavit(q(3),     0,      0.035, -pi/2);
T34 = denavit(q(4),     0.335,  0,      pi/2);
T45 = denavit(q(5),     0,      0,     -pi/2);
T56 = denavit(q(6),     0.080,  0,         0);

M = T01*T12*T23*T34*T45*T56;
```

```
end
```

Conclusion

Como se puede observar se obtiene practicamente los mismo resultados. La diferencia en la posicion pz puede deberse a error de calculo numero de MATLAB.