Desarrollo Laboratorio 2 - Robótica 2021-II

Edgar Alejandro Ruíz Velasco

Jesus Daniel Caballero Colina

Jhohan David Contreras Aragón

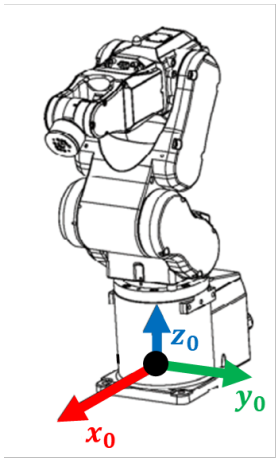
<u>Para este equipo de trabajo se asignó el robot industrial *Epson C4*</u>

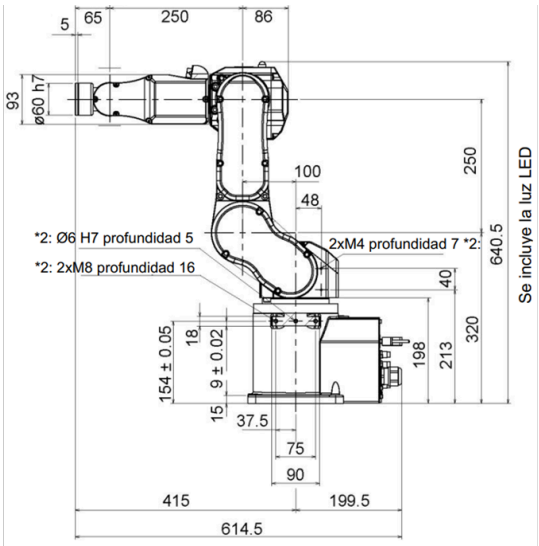
<u>Punto 1.</u> Determine el modelo geométrico inverso del robot asignado haciendo uso de la metodología explicada en clase. Haga una descripción detallada del proceso haciendo uso de imágenes y dibujos que ayuden visualizar de dónde provienen las ecuaciones encontradas.

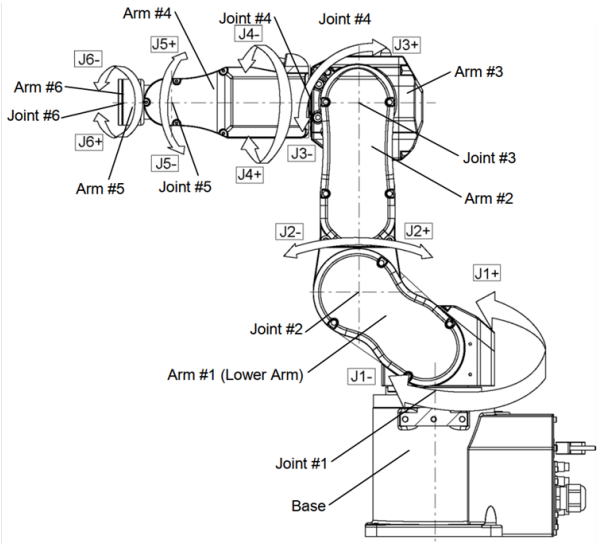
(Ver punto 2)

<u>Punto 2.</u> En las soluciones de la inversa incluya consideraciones respecto a multiplicidad de soluciones.

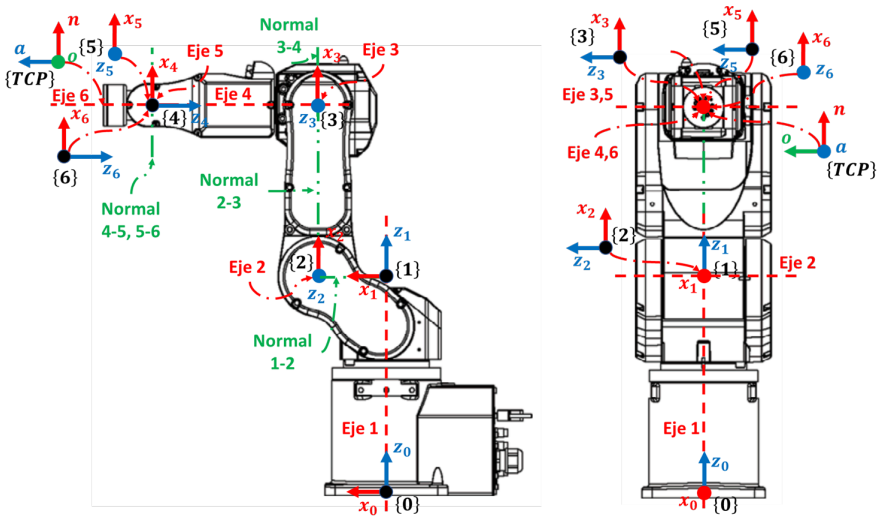
Para el robot asignado se tiene las siguientes medidas y movimientos:







Con las gráficas anterior se realiza la ubicación de MTH y determinación de parámetros DH:



i	α_{i-1}	a_{i-1}	d_i	θ_i	Offset
1	0	0	320	J_1	0
2	$\pi/2$	100	0	J_2	$\pi/2$
3	0	250	0	J_3	0
4	$-\pi/2$	0	-250	J_4	0
5	$\pi/2$	0	0	J_5	0
6	$-\pi/2$	0	0	J_6	0

Se observa que las 3 últimas articulaciones (4, 5 y 6) se encuentran en el mismo punto. Aunque la combinación de esas 3 articulaciones no se considere como una junta esférica (y no se pueda aplicar el método Pieper completo) si se puede realizar el método de desacoplamiento cinemático hasta cierto punto. El metodo a usar es:

1. Hallar la posición del punto p (posición de los sistemas de referencia 4, 5 y 6) mediante las ecuaciones de MTH
2. Por método gráfico, hallar los valores de las 3 primeras articulaciones.
3. Por método analítico, hallar el valor de las últimas articulaciones

Paso 1:

La ubicación de punto P se halla sabiendo que la pose del TCP está ya determinada. Teniendo que la ubicación de la herramienta es $X_{\text{TCP}} = [x \quad y \quad z \quad \psi \quad \theta \quad \phi]^T$ (con $[\theta \quad \psi \quad \phi]$ ángulos rpy), se halla la matriz de rotación del sistema TCP R_{TCP} :

$$R_{\text{TCP}} = \begin{bmatrix} C\phi.C\theta & C\phi.S\theta.S\psi - S\phi.C\psi & C\phi.S\theta.C\psi + S\phi.S\psi \\ S\phi.C\theta & S\phi.S\theta.S\psi - C\phi.C\psi & S\phi.S\theta.C\psi - C\phi.S\psi \\ -S\theta & C\theta.S\psi & C\theta.C\psi \end{bmatrix}$$

Además se sabe que la posición de p cumple la ecuación:

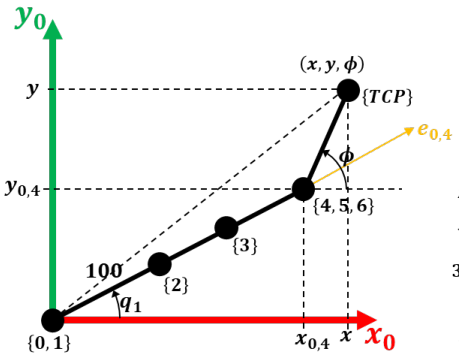
$$X_P = X_{\text{TCP}} - R_{\text{TCP}} \begin{bmatrix} 0 \\ 0 \\ 65 \end{bmatrix}$$

$$X_P = \begin{bmatrix} x_{0,4} \\ y_{0,4} \\ z_{0,4} \end{bmatrix} = \begin{bmatrix} x - 65.(C\phi.S\theta.C\psi + S\phi.S\psi) \\ y - 65.(S\phi.S\theta.C\psi - C\phi.S\psi) \\ z - 65.(C\theta.C\psi) \end{bmatrix}$$

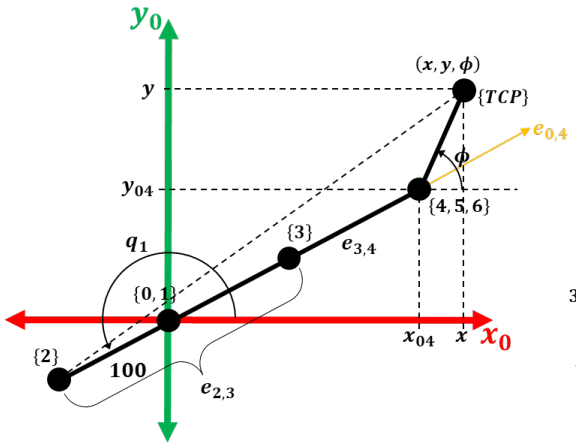
Paso 2:

Los esquemas simplificados que ilustran la ubicación de cada articulación con relación a sus valores son:

Vista aérea

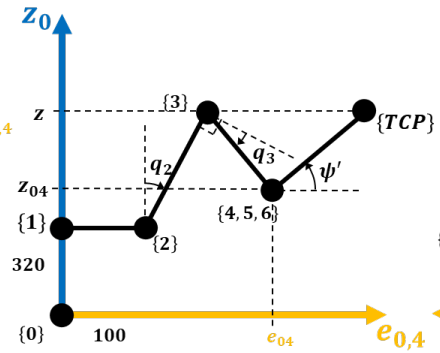


Articulación 1 - Solución 1

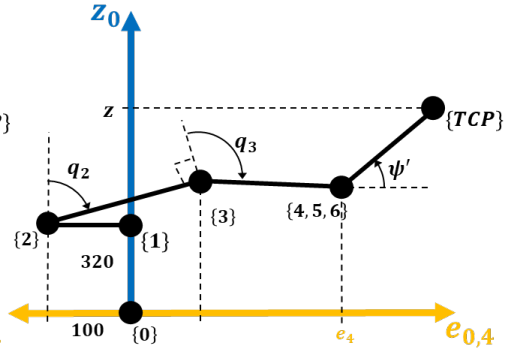


Articulación 1 - Solución 2

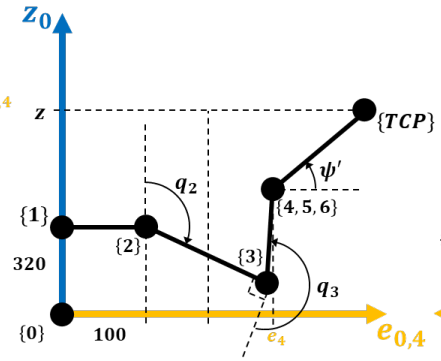
Vista oblicua (eje e)



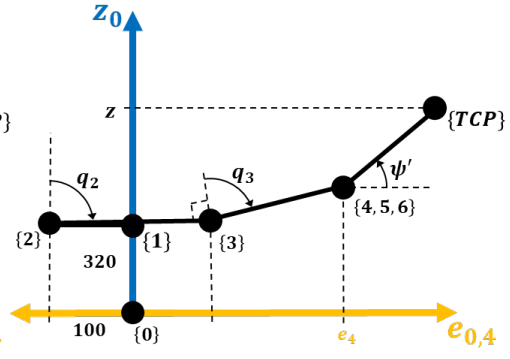
Articulación 2,3 - Solución 1
(Sol. 1 θ_1 Codo arriba)



Articulación 2,3 - Solución 1
(Sol. 2 θ_1 Codo arriba)



Articulación 2,3 - Solución 1
(Sol. 1 θ_1 Codo abajo)

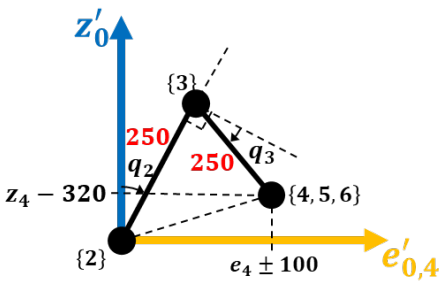


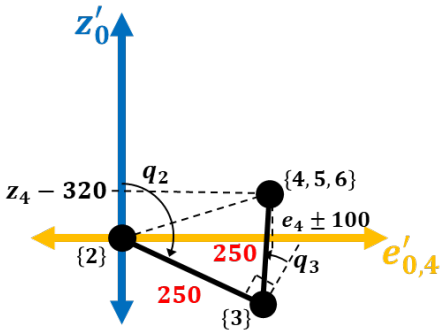
Articulación 2,3 - Solución 1
(Sol. 2 θ_1 Codo abajo)

Con la vista aérea se tiene que:

$$q_{1;i} = \text{atan2}(y_{0,4}, x_{0,4}) \text{ y } q_{1;\text{ii}} = \text{atan2}(y_{0,4}, x_{0,4}) + \pi$$

Con la vista oblicua se tiene siguiente esquema:





Donde:

$$e_4 = \sqrt{x_4^2 + y_4^2}$$

Por teorema de cosenos (y teniendo en cuenta las dos soluciones de θ_1):

$$(e_4 \pm_1 100)^2 + (z_4 - 320)^2 = 2.250^2 - 2.250^2.\cos(180 - (90 + q_3)) = 2.250^2(1 + \cos(90 + q_3))$$

$$(e_4 \pm_1 100)^2 + (z_4 - 320)^2 = 2.250^2(1 - \sin(q_3))$$

$$\sin(q_3) = 1 - \frac{(e_4 - 100)^2 + (z_4 - 320)^2}{2.250^2} = D_1$$

$$\sin(q_3) = 1 - \frac{(e_4 + 100)^2 + (z_4 - 320)^2}{2.250^2} = D_2$$

Es decir que

$$\cos(q_3) = \sqrt{1 - \sin^2(q_3)} = \pm_2 \sqrt{1 - D_1^2} \text{ o } \pm_2 \sqrt{1 - D_2^2}$$

$$q_{3;i,\text{ii}} = \text{atan2}\left(D_1, \pm_2 \sqrt{1 - D_1^2}\right) \text{o } q_{3;\text{iii,iv}} = \text{atan2}\left(D_1, \pm_2 \sqrt{1 - D_2^2}\right)$$

Ya teniendo q_3 se puede determinar q_2 por suma de ángulos:

$$q_{2;i,\text{ii}} = 90 - \text{atan2}(z_4 - 320, e_4 - 100) - \left(45 + \frac{q_{3;i,\text{ii}}}{2}\right) \text{ o } q_{2;\text{iii,iv}} = 90 - \text{atan2}(z_4 - 320, e_4 + 100) + \left(45 + \frac{q_{3;\text{iii,iv}}}{2}\right)$$

$$q_{2;i,\text{ii}} = 45 - \text{atan2}(z_4 - 320, e_4 - 100) - \frac{q_{3;i,\text{ii}}}{2} \text{ o } q_{2;\text{iii,iv}} = 135 - \text{atan2}(z_4 - 320, e_4 + 100) + \frac{q_{3;\text{iii,iv}}}{2}$$

Es decir que las soluciones quedarían:

$$\begin{bmatrix} \text{Sol.} \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \text{Sol. 1} & \text{Sol. 2} & \text{Sol. 3} & \text{Sol. 4} \\ q_{1,i} & q_{1,i} & q_{1,\text{ii}} & q_{1,\text{ii}} \\ q_{2,i} & q_{2,\text{ii}} & q_{2,\text{iii}} & q_{2,\text{iv}} \\ q_{3,i} & q_{3,\text{ii}} & q_{3,\text{iii}} & q_{3,\text{iv}} \end{bmatrix}$$

Paso 3:

Como ya se obtuvo los valores de las 3 primeras articulaciones (de una determinada solución k) entonces es conocido el valor de ${}^0R_3$:

$${}^0R_3 = {}^0R_1(q_{1,k}).\,{}^1R_2(q_{2,k}).\,{}^2R_3(q_{3,k}) = \begin{bmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} C\left(q_2 + \frac{\pi}{2}\right) & -S\left(q_2 + \frac{\pi}{2}\right) & 0 \\ 0 & 0 & -1 \\ S\left(q_2 + \frac{\pi}{2}\right) & C\left(q_2 + \frac{\pi}{2}\right) & 0 \end{bmatrix}.\begin{bmatrix} C_3 & -S_3 & 0 \\ S_3 & C_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$${}^0R_3 = \begin{bmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} -S_2 & -C_2 & 0 \\ 0 & 0 & -1 \\ C_2 & -S_2 & 0 \end{bmatrix}.\begin{bmatrix} C_3 & -S_3 & 0 \\ S_3 & C_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$ {}^{0}R_{3} = \begin{bmatrix} -C_1C_2S_3 - C_1S_2C_3 & C_1S_2S_3 - C_1C_2C_3 & S_1 \\ -S_1C_2S_3 - S_1S_2C_3 & S_1S_2S_3 - S_1C_2C_3 & -C_1 \\ C_2C_3 - S_2S_3 & -C_2S_3 - S_2C_3 & 0 \end{bmatrix} $$

Y por la orientación de la disposición de las 3 últimas articulaciones se sabe que la matriz de rotación desde el sistema 3 al sistema TCP (colocandolos en un mismo origen) ${}^{3}R_{\mathrm{TCP}}$ tendría un valor de:

$$ {}^{3}R_{\mathrm{TCP}} = R_{\mathrm{eul,ZYZ}}(-q_4, -q_5, -q_6).R_x(\pi) = \begin{bmatrix} -S_4S_6 + C_4C_5C_6 & S_4C_6 + C_4C_5S_6 & -C_4S_5 \\ -C_4S_6 - S_4C_5C_6 & C_4C_6 - S_4C_5S_6 & S_4S_5 \\ S_5C_6 & S_5S_6 & C_5 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} $$

$$ {}^{3}R_{\mathrm{TCP}} = \begin{bmatrix} -S_4S_6 + C_4C_5C_6 & -S_4C_6 - C_4C_5S_6 & C_4S_5 \\ C_4S_6 + S_4C_5C_6 & C_4C_6 - S_4C_5S_6 & S_4S_5 \\ -S_5C_6 & S_5S_6 & C_5 \end{bmatrix} $$

Es decir que ${}^{3}R_{\mathrm{TCP}}$ deseado en termino de X_{TCP} y las 3 primeras articulaciones es:

$$ {}^{3}R^{*}_{\mathrm{TCP}} = {}^{0}R_{3}^{T}.{}^{0}R_{\mathrm{TCP}} = \begin{bmatrix} -C_1C_2S_3 - C_1S_2C_3 & C_1S_2S_3 - C_1C_2C_3 & S_1 \\ -S_1C_2S_3 - S_1S_2C_3 & S_1S_2S_3 - S_1C_2C_3 & -C_1 \\ C_2C_3 - S_2S_3 & -C_2S_3 - S_2C_3 & 0 \end{bmatrix}^{T} . \begin{bmatrix} C\phi.C\theta & C\phi.S\theta.S\psi - S\phi.C\psi & C\phi.S\theta.C\psi + S\phi.S\psi \\ S\phi.C\theta & S\phi.S\theta.S\psi - C\phi.C\psi & S\phi.S\theta.C\psi - C\phi.S\psi \\ -S\theta & C\theta.S\psi & C\theta.C\psi \end{bmatrix} $$

$$ {}^{3}R^{*}_{\mathrm{TCP}} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} $$

E igualando ${}^{3}R^{*}_{\mathrm{TCP}}$ y ${}^{3}R_{\mathrm{TCP}}$, se puede obtener las siguientes igualdades:

$$ q_4 = \mathrm{atan2}(a_{23}, a_{13}) $$

$$ q_5 = \mathrm{atan2}\left(a_{31}^2 + a_{32}^2, a_{33}\right) $$

$$ q_6 = \mathrm{atan2}(a_{32}, -a_{31}) $$

Estas ecuaciones se implementaron en el archivo EpsonC4_IC.m

Punto 3. Haga uso de las funciones del RVC para hallar la cinemática inversa de su robot asignado y compruebe los resultados del punto anterior. Ya que existen varias funciones en el *Toolbox* explique:

¿Cuál es la diferencia entre estas funciones?

En la libreria RVC se encuentran concretamente siete funciones que pueden ser usaadas para calculos de cinematica inversa. La funcion **ikcon** indica una de las configuraciones posibles de valores de las articulaciones que permite obtener la pose deseada, esto mediante una solucion numerica con limites en las articulaciones. Las funciones **ikine** y **ikunc** tambien utilizan soluciones numericas pero sin restricciones en las articulaciones. La funcion **ikine3** permite obtener a partir de una entrada que especifique las pose de el

efector final una solucion analitica para robots de tres ejes sin muñeca como puede ser el caso de robots desacoplados cinematicamente. La funcion **ikine6s** mediante solucion analitica calcula a partir de una pose una configuracion de posiciones de las articulaciones para robots de seis ejes con muñeca esferica. La funcion **ikine_sym** permite calcular mediante variables simbolicas las diferentes posibles configuraciones siendo la herramienta mas general para la obtencion de soluciones analiticas algebraicas. Finalmente la funcion **ikinem** permite obtener una solucion numerica a partir de una pose utilizando el minimizador Levenberg-Marquad.

¿Cuál debe usar para su robot y por que? (Revise la documentación del *Toolbox*)

Para éste caso se opta por el uso de funciones que calculen analiticamente debido a que las soluciones numericas son genralmente mas lentas que las cerradas, usaremos **ikine_sym** para calcular una solucion mediante variables simbolicas y posteriormente se pensó en usar la funcion **ikine6s** para casos puntuales dado que en su construccion esta pensada para nuestro caso especifico, es decir un robot de seis ejes con muñeca de articulacion esferica, sin embargo solo funciona para modelos construidos mediante la convención Denavit-Hartenberg estandar, por este motivo se opto por el uso de una solucion numerica, concretamente **ikcon** la cual permite tener en cuenta los limites de las articulaciones.

Para el uso de la funcion **ikine_sym** se desglosa el robot en dos partes con tres grados de libertad cada una determinando orientacion y posicion del efector final de forma independiente y así obtener para las primeras tres articulaciones las dos soluciones analiticas de cada articulacion para la determinacion de una posicion arbitraria y luego para las ultimas tres que conforman una articulacion esferica sus respectivas soluciones para la determinacion de la orientacion del efector final.

```
T6b = trotx(pi)*transl([0 0 6.5]);
T6b=[1 0 0 0; 0 -1 0 0; 0 0 -1 -6.5; 0 0 0 1];

%     MDHm(thetai,      di,    ai-1,  alpha-1,   sigma, offset)
Links(1) = Link('revolute','alpha',0,    'a',0, 'd',32, 'offset',0,'modified', 'qlim',[-17*pi/1
Links(2) = Link('revolute','alpha',pi/2, 'a',10,'d',0,  'offset',pi/2,'modified', 'qlim',[-8*pi
Links(3) = Link('revolute','alpha',0,    'a',25,'d',0,  'offset',0,   'modified', 'qlim',[-17*p
Links(4) = Link('revolute','alpha',-pi/2,'a',0, 'd',-25,'offset',0,   'modified', 'qlim',[-10*p
Links(5) = Link('revolute','alpha',pi/2, 'a',0, 'd',0,  'offset',0,   'modified', 'qlim',[-3*pi
Links(6) = Link('revolute','alpha',-pi/2,'a',0, 'd',0,  'offset',0,   'modified', 'qlim',[-2*pi

Epson_C4 = SerialLink(Links,'name','Epson C4','tool',T6b)
```

```
Epson_C4 =

Epson C4 (6 axis, RRRRRR, modDH, fastRNE)

+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|         32|          0|          0|          0|
|  2|         q2|          0|         10|      1.571|      1.571|
|  3|         q3|          0|         25|          0|          0|
|  4|         q4|        -25|          0|     -1.571|          0|
|  5|         q5|          0|          0|      1.571|          0|
|  6|         q6|          0|          0|     -1.571|          0|
+---+-----------+-----------+-----------+-----------+-----------+

grav =    0  base = 1  0  0  0   tool =  1          0          0          0
          0         0  1  0  0           0         -1          0          0
```

```
      9.81        0  0  1  0          0          0         -1       -6.5
                  0  0  0  1          0          0          0          1
```

```
for i=1:3
    L_P(i)=Links(i);
    L_O(i)=Links(i+3);
end

TV1=trotx(-pi/2)*transl([0 0 -25]);

Epson_C4_T = SerialLink(L_P,'name','Epson C4 T','tool',TV1)
```

```
Epson_C4_T =

Epson C4 T (3 axis, RRR, modDH, fastRNE)

+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|         32|          0|          0|          0|
|  2|         q2|          0|         10|      1.571|      1.571|
|  3|         q3|          0|         25|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+

grav =    0  base = 1  0  0  0   tool =  1           0           0           0
          0         0  1  0  0           0           0           1         -25
       9.81         0  0  1  0           0          -1           0 -1.53081e-15
                    0  0  0  1           0           0           0           1
```

```
Epson_C4_O = SerialLink(L_O,'name','Epson C4 O','tool',T6b)
```

```
Epson_C4_O =

Epson C4 O (3 axis, RRR, modDH, fastRNE)

+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|        -25|          0|     -1.571|          0|
|  2|         q2|          0|          0|      1.571|          0|
|  3|         q3|          0|          0|     -1.571|          0|
+---+-----------+-----------+-----------+-----------+-----------+

grav =    0  base = 1  0  0  0   tool =  1          0          0          0
          0         0  1  0  0           0         -1          0          0
       9.81         0  0  1  0           0          0         -1       -6.5
                    0  0  0  1           0          0          0          1
```

A partir de aca el procedimiento se describe mediante imagenes, ésto debido a que la funcion **ikine_sym** usa la funcion findsym de la libreria Symbolic Math Toolbox la cual no está disponible en las versiones actuales de MAtlab por lo que el procedimiento debio ser llevado a cabo en la version 2017b de software y se muestra a continuacion.

```
SolT=Epson_C4_T.ikine_sym(3);
```

```
Sq1=SolT{1};
q1_1=Sq1(1)
```

q1_1 = $\operatorname{atan2}(-25\,ay - ty, -25\,ax - tx)$

```
q1_2=Sq1(2)
```

q1_2 = $\operatorname{atan2}(25\,ay + ty, 25\,ax + tx)$

```
Sq2=SolT{2};
q2_1=Sq2(1)
```

q2_1 =

$$\operatorname{atan2}\left(25, -\sqrt{625\,C_1^2\,ax^2 + 50\,C_1^2\,ax\,tx - 625\,C_1^2\,ay^2 - 50\,C_1^2\,ay\,ty + C_1^2\,tx^2 - C_1^2\,ty^2 + 1250\,S_1\,C_1\,ax\,ay + 50\,S_1\,C_1\,ax\,ty - 500\,C_1\,ax\right.$$

$$\left.+ 50\,ay\,ty - 500\,S_1\,ay + 625\,az^2 + 50\,az\,tz - 1600\,az + ty^2 - 20\,S_1\,ty + tz^2 - 64\,tz + 499}\right) - \operatorname{angle}(25\,az\,i + tz\,i - 25\,C_1\,ax - 25$$

```
q2_2=Sq2(2)
```

q2_2 =

$$-\operatorname{angle}(25\,az\,i + tz\,i - 25\,C_1\,ax - 25\,S_1\,ay - C_1\,tx - S_1\,ty + 10 - 32\,i) + \operatorname{atan2}\left(25, \sqrt{625\,C_1^2\,ax^2 + 50\,C_1^2\,ax\,tx - 625\,C_1^2\,ay^2 -\right.$$

$$\left.- 500\,C_1\,ax + 50\,S_1\,C_1\,ay\,tx + 2\,S_1\,C_1\,tx\,ty - 20\,C_1\,tx + 625\,ay^2 + 50\,ay\,ty - 500\,S_1\,ay + 625\,az^2 + 50\,az\,tz - 1600\,az + ty^2 - 20\right.$$

```
Sq3=SolT{3};
q3_1=Sq3(1)
```

q3_1 =

$$\operatorname{atan}(25) + \operatorname{atan2}\left(32\cos(q_2+q_3) - 10\sin(q_2+q_3) + \sin(q_3) - 25\,az\cos(q_2+q_3) - tz\cos(q_2+q_3) + 25\,C_1\,ax\sin(q_2+q_3) +\right.$$

$$\left.- \sqrt{800\,az + 32\,tz - 10\,\sigma_1 - 32\,\sigma_2 + \frac{\cos(2\,q_3)}{2} - 462\cos(\sigma_3) + 320\sin(\sigma_3) + 10\cos(q_2) + 32\sin(q_2) + 250\,C_1\,ax + 250\,S_1\,ay +\right.$$

$$+\frac{625\,ay^2\cos(\sigma_3)}{2}-\frac{625\,az^2\cos(\sigma_3)}{2}+\frac{ty^2\cos(\sigma_3)}{2}-\frac{tz^2\cos(\sigma_3)}{2}-25\,az\sin(q_2)-tz\sin(q_2)-\frac{625\,ay^2}{2}-\frac{625\,az^2}{2}-\frac{625\,C_1^2\,ax^2}{2}+\frac{62}{}$$

$$+S_1\,ty\,\sigma_1+\frac{C_1^2\,tx^2\cos(\sigma_3)}{2}-\frac{C_1^2\,ty^2\cos(\sigma_3)}{2}-250\,C_1\,ax\cos(\sigma_3)-800\,C_1\,ax\sin(\sigma_3)-250\,S_1\,ay\cos(\sigma_3)-10\,C_1\,tx\cos(\sigma_3)-800$$

$$-25\,C_1\,ax\cos(q_2)-25\,S_1\,ay\cos(q_2)-C_1\,tx\cos(q_2)-S_1\,ty\cos(q_2)+S_1\,ty\,tz\sin(\sigma_3)+25\,C_1^2\,ax\,tx\cos(\sigma_3)-25\,C_1^2\,ay\,ty\cos(\sigma_3$$

$$+25\,C_1\,az\,tx\sin(\sigma_3)+25\,S_1\,ay\,tz\sin(\sigma_3)+25\,S_1\,az\,ty\sin(\sigma_3)+C_1\,tx\,tz\sin(\sigma_3)+625\,C_1\,S_1\,ax\,ay\cos(\sigma_3)+25\,C_1\,S_1\,ax\,ty\cos(\sigma_3)$$

where

$$\sigma_1=\cos(q_2+2\,q_3)$$

$$\sigma_2=\sin(q_2+2\,q_3)$$

$$\sigma_3=2\,q_2+2\,q_3$$

```
q3_2=Sq3(2)
```

```
q3_2 =
```

$$\operatorname{atan}(25)+\operatorname{atan2}\left(32\cos(q_2+q_3)-10\sin(q_2+q_3)+\sin(q_3)-25\,az\cos(q_2+q_3)-tz\cos(q_2+q_3)+25\,C_1\,ax\sin(q_2+q_3)+\right.$$

$$\left.\sqrt{800\,az+32\,tz-10\,\sigma_1-32\,\sigma_2+\frac{\cos(2\,q_3)}{2}-462\cos(\sigma_3)+320\sin(\sigma_3)+10\cos(q_2)+32\sin(q_2)+250\,C_1\,ax+250\,S_1\,ay+10}\right.$$

$$+\frac{625\,ay^2\cos(\sigma_3)}{2}-\frac{625\,az^2\cos(\sigma_3)}{2}+\frac{ty^2\cos(\sigma_3)}{2}-\frac{tz^2\cos(\sigma_3)}{2}-25\,az\sin(q_2)-tz\sin(q_2)-\frac{625\,ay^2}{2}-\frac{625\,az^2}{2}-\frac{625\,C_1^2\,ax^2}{2}+\frac{62}{}$$

$$+\frac{625\,C_1^2\,ax^2\cos(\sigma_3)}{2}-\frac{625\,C_1^2\,ay^2\cos(\sigma_3)}{2}+S_1\,ty\,\sigma_1+\frac{C_1^2\,tx^2\cos(\sigma_3)}{2}-\frac{C_1^2\,ty^2\cos(\sigma_3)}{2}-250\,C_1\,ax\cos(\sigma_3)-800\,C_1\,ax\sin(\sigma_3)$$

$$-25\,C_1\,\mathrm{ax}\cos(q_2)-25\,S_1\,\mathrm{ay}\cos(q_2)-C_1\,\mathrm{tx}\cos(q_2)-S_1\,\mathrm{ty}\cos(q_2)+S_1\,\mathrm{ty}\,\mathrm{tz}\sin(\sigma_3)+25\,C_1^{\,2}\,\mathrm{ax}\,\mathrm{tx}\cos(\sigma_3)-25\,C_1^{\,2}\,\mathrm{ay}\,\mathrm{ty}\cos(\sigma_3$$

$$+25\,C_1\,\mathrm{az}\,\mathrm{tx}\sin(\sigma_3)+25\,S_1\,\mathrm{ay}\,\mathrm{tz}\sin(\sigma_3)+25\,S_1\,\mathrm{az}\,\mathrm{ty}\sin(\sigma_3)+C_1\,\mathrm{tx}\,\mathrm{tz}\sin(\sigma_3)+625\,C_1\,S_1\,\mathrm{ax}\,\mathrm{ay}\cos(\sigma_3)+25\,C_1\,S_1\,\mathrm{ax}\,\mathrm{ty}\cos(\sigma_3)$$

```
Sol0=Epson_C4_0.ikine_sym(3);
```

```
Sq4=Sol0{1};
q4_1=Sq4(1)
```

q4_1 =

$$\operatorname{atan2}\left(\frac{13\,ax}{2}-tx,\frac{13\,az}{2}-tz\right)$$

```
q4_2=Sq4(2)
```

q4_2 =

$$\operatorname{atan2}\left(tx-\frac{13\,ax}{2},tz-\frac{13\,az}{2}\right)$$

```
Sq5=Sol0{2};
q5_1=Sq5(1)
```

q5_1 =

$$\operatorname{atan2}\left(\frac{13\,C_1\,ax}{2}-\frac{13\,S_1\,az}{2}-C_1\,tx+S_1\,tz,ty-\frac{13\,ay}{2}+25\right)$$

```
q5_2=Sq5(2)
```

q5_2 =

$$\operatorname{atan2}\left(\frac{13\,S_1\,az}{2}-\frac{13\,C_1\,ax}{2}+C_1\,tx-S_1\,tz,\frac{13\,ay}{2}-ty-25\right)$$

```
Sq6=Sol0{3};
q6_1=Sq6(1)
```

q6_1 = $\operatorname{atan2}(13\,S_2\,ay-50\,S_2-2\,S_2\,ty+13\,C_1\,C_2\,ax-13\,C_2\,S_1\,az-2\,C_1\,C_2\,tx+2\,C_2\,S_1\,tz,13\,C_1\,az+13\,S_1\,ax-2\,C_1\,tz-2\,S_1$

```
q6_2=Sq6(2)
```

q6_2 = $\operatorname{atan2}(50\,S_2-13\,S_2\,ay+2\,S_2\,ty-13\,C_1\,C_2\,ax+13\,C_2\,S_1\,az+2\,C_1\,C_2\,tx-2\,C_2\,S_1\,tz,2\,C_1\,tz-13\,S_1\,ax-13\,C_1\,az+2\,S_1$

Para el uso de **ikcon** se debe establecer una pose especifica arbitraria la cual se obtiene aleatoriamente de la siguiente manera.

```
dhparams = [0,   0,    0.32, 0;
            0.10,pi/2, 0     0;
            0.25,0,    0,    0;
            0,   -pi/2,-0.25,0;
            0,   pi/2, 0,    0;
            0,   -pi/2,0 ,   0;
            0,   pi,   0.065,0];
robot = rigidBodyTree;
```

```
body1 = rigidBody('body1');
body2 = rigidBody('body2');
body3 = rigidBody('body3');
body4 = rigidBody('body4');
body5 = rigidBody('body5');
body6 = rigidBody('body6');
body7 = rigidBody('TCP');

jnt1 = rigidBodyJoint('jnt1','revolute');
jnt1.HomePosition = pi/2;
jnt2 = rigidBodyJoint('jnt2','revolute');
jnt2.HomePosition = pi/2;
jnt3 = rigidBodyJoint('jnt3','revolute');
jnt3.HomePosition = 0;
jnt4 = rigidBodyJoint('jnt4','revolute');
jnt4.HomePosition = 0;
jnt5 = rigidBodyJoint('jnt5','revolute');
jnt5.HomePosition = 0;
jnt6 = rigidBodyJoint('jnt6','revolute');
jnt6.HomePosition = 0;
jnt7 = rigidBodyJoint('jntTCP','revolute');
jnt7.HomePosition = 0;

setFixedTransform(jnt1,dhparams(1,:),'mdh');
setFixedTransform(jnt2,dhparams(2,:),'mdh');
setFixedTransform(jnt3,dhparams(3,:),'mdh');
setFixedTransform(jnt4,dhparams(4,:),'mdh');
setFixedTransform(jnt5,dhparams(5,:),'mdh');
setFixedTransform(jnt6,dhparams(6,:),'mdh');
setFixedTransform(jnt7,dhparams(7,:),'mdh');

body1.Joint = jnt1;
body2.Joint = jnt2;
body3.Joint = jnt3;
body4.Joint = jnt4;
body5.Joint = jnt5;
body6.Joint = jnt6;
body7.Joint = jnt7;

addBody(robot,body1,'base')
addBody(robot,body2,'body1')
addBody(robot,body3,'body2')
addBody(robot,body4,'body3')
addBody(robot,body5,'body4')
addBody(robot,body6,'body5')
%addBody(robot,body7,'body6')
config_home = homeConfiguration(robot);
config = homeConfiguration(robot);

aik = analyticalInverseKinematics(robot);
opts = showdetails(aik);
```

```
Robot: (6 bodies)

Index      Base Name   EE Body Name     Type                    Actions
-----      ---------   ------------     ----                    -------
    1           base          body6   RRRSSS   Use this kinematic group
```

```
aik.KinematicGroup = opts(1).KinematicGroup;
disp(aik.KinematicGroup)
```

```
              BaseName: 'base'
    EndEffectorBodyName: 'body6'
```

```
generateIKFunction(aik,'willowRobotIK');
rng(0);
expConfig = randomConfiguration(robot)
```

expConfig = 1×6 struct

Fields	JointName	JointPosition
1	'jnt1'	1.9775
2	'jnt2'	2.5497
3	'jnt3'	-2.3437
4	'jnt4'	2.5973
5	'jnt5'	0.8316
6	'jnt6'	-2.5287

```
eeBodyName = aik.KinematicGroup.EndEffectorBodyName;
baseName = aik.KinematicGroup.BaseName;
expEEPose = getTransform(robot,expConfig,eeBodyName,baseName)
```

```
expEEPose = 4×4
   -0.5367   -0.8283    0.1612    0.0223
    0.7234   -0.3533    0.5932   -0.0517
   -0.4344    0.4349    0.7887    0.2148
         0         0         0    1.0000
```

```
qq=Epson_C4.ikcon(expEEPose)
```

```
qq = 1×6
   -1.8232   -2.5757    0.1192    3.1337    1.5939    2.3660
```

Punto 4. Haga uso del RST para hallar la cinemática inversa de su robot asignado y compruebe los resultados anteriores.

Aprovechando la creacion del modelo del robot en RST realizada en el codigo anterior y usando la funcion **analyticalInverseKinematics** que permite el calculo de una de las soluciones de cinematica inversa de forma analitica.

```
ikConfig = willowRobotIK(expEEPose,false);
eeWorldPose = getTransform(robot,expConfig,eeBodyName);

generatedConfig = repmat(expConfig, size(ikConfig,1), 1)
```

generatedConfig = 8×6 struct

	1	2	3	4	5	6
1	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
2	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
3	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
4	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
5	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
6	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
7	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct
8	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct	1×1 struct

```
SS1=generatedConfig(1,:)
```

SS1 = 1×6 struct

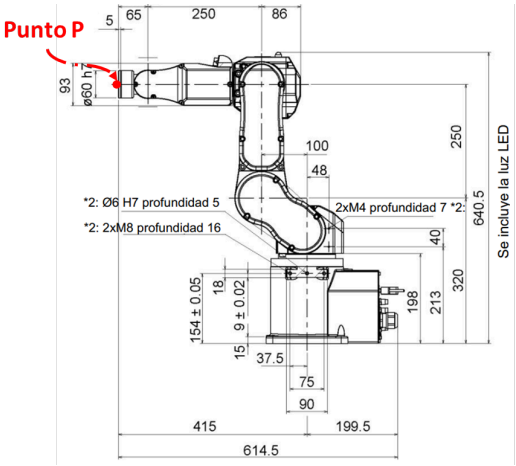
Fields	JointName	JointPosition
1	'jnt1'	1.9775
2	'jnt2'	2.5497
3	'jnt3'	-2.3437
4	'jnt4'	2.5973
5	'jnt5'	0.8316
6	'jnt6'	-2.5287

Punto 5. Compare los métodos.

Para el caso se compara la fincion **ikcon** del Robotics Vision Control con la funcion **analyticalInverseKinematics** del Robotics Sistem Toolbox, la principal diferencia entre éstos el metodo utilizado para realizar el calculo de las soluciones, por un lado el primero utiliza un metodo numerico iterativo que permite la obtencion de solo un conjunto de soluciones, por otro lado el segundo obtiene la solucion de forma analitica calculando todas las soluciones de forma cerrada. Ambas funciones son compatibles con modelos de robot de seis grados de libertad.

Punto 6. Compruebe mediante su modelo geométrico inverso la configuración del robot para las posturas de la herramienta halladas en el anterior laboratorio, tales como puntos de calibración.

El punto de calibración se encuentra en:



J_1	J_2	J_3	J_4	J_5	J_6	$x_P\ [mm]$	$y_P\ [mm]$	$z_P\ [mm]$	ψ	θ	ϕ
0	0	0	0	0	0	0	415	570	0	$\pi/2$	π

```
Configs = EpsonC4_IC([415 0 570 0 pi/2 pi])
```

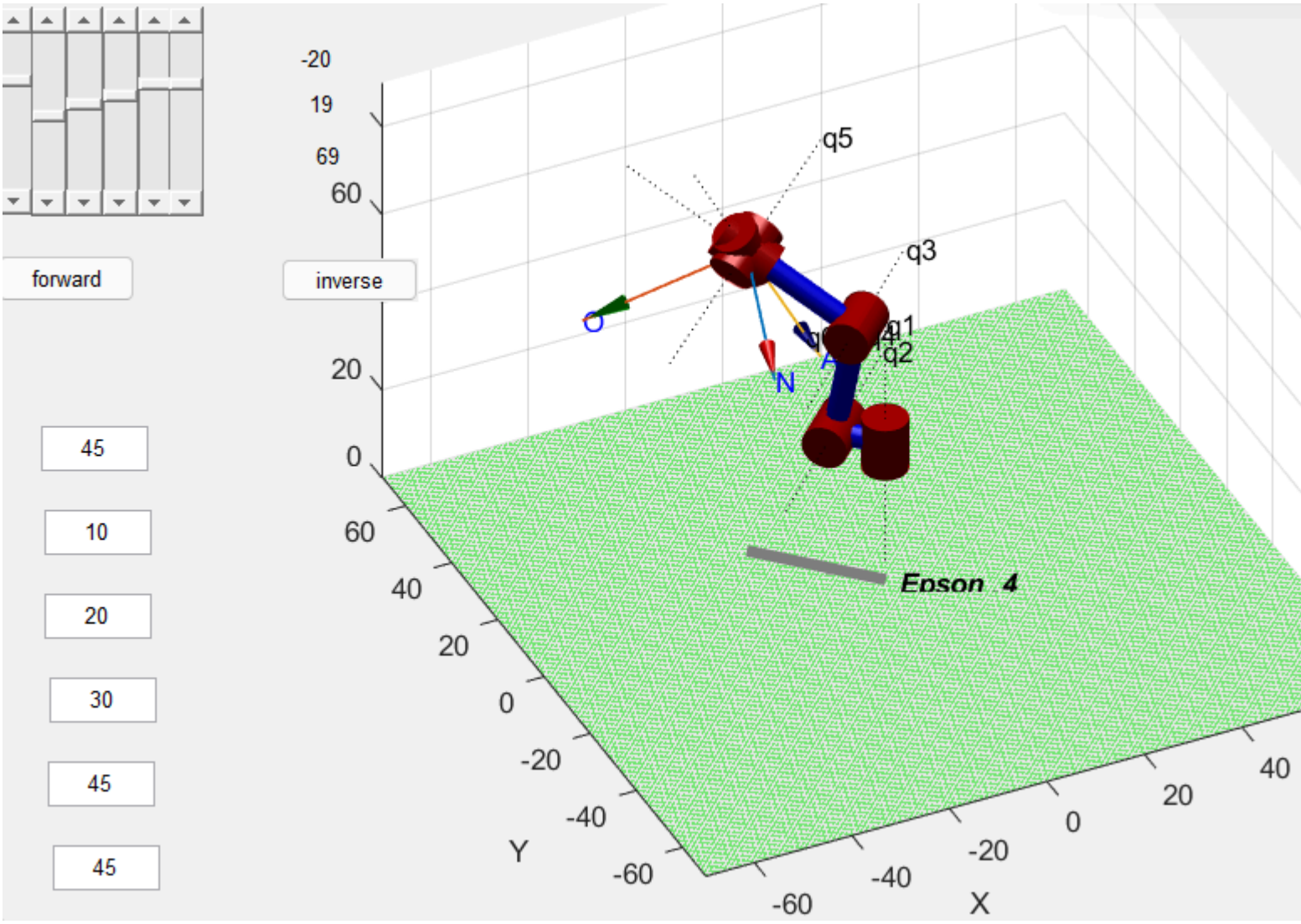
```
Warning: Configuración 3 no valida
Warning: Configuración 4 no valida
Configs = 4×6
         0         0         0         0         0   -0.0000
         0  -90.0000  180.0000         0  -90.0000    0.0000
         0         0         0         0         0         0
         0         0         0         0         0         0
```

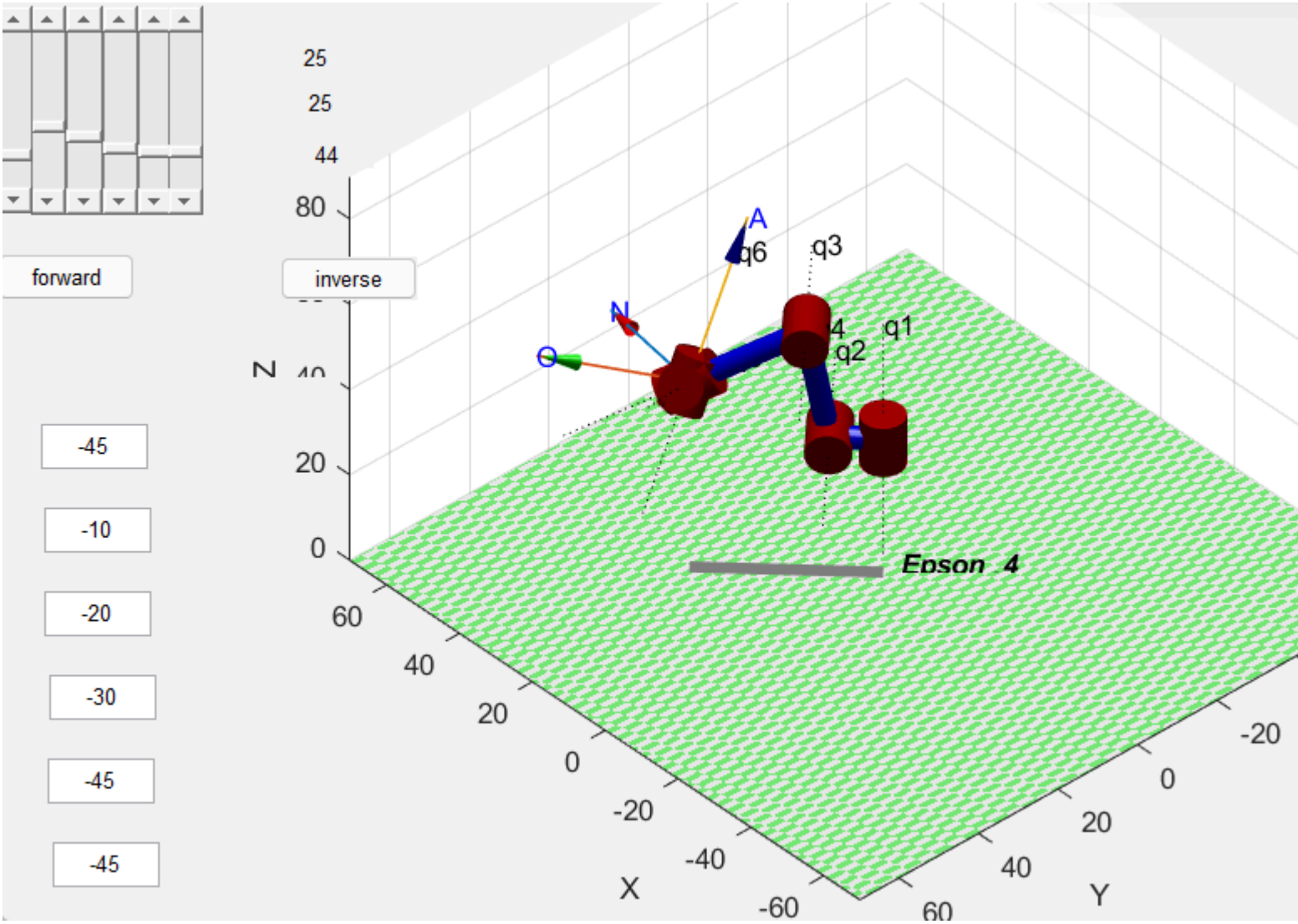
<u>Punto 7.</u> Proponga 4 posturas (*x, y, z, roll, pitch, yaw*)' que estén dentro del espacio de trabajo y determine la configuración del manipulador y complete la tabla 1:

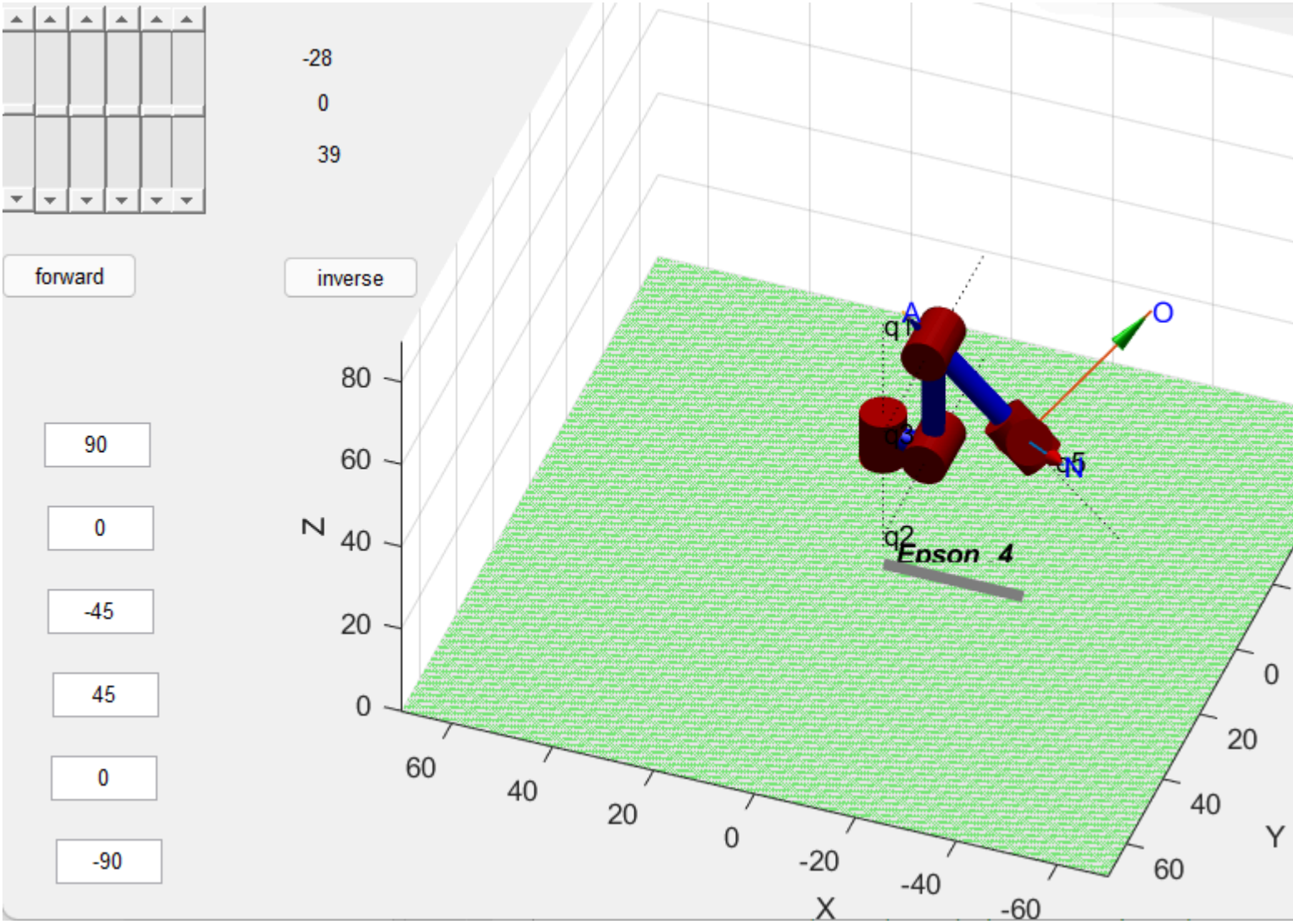
Tabla 1

x	y	z	roll	pitch	yaw	q_1	q_2	q_3	q_4	q_5	q_6
10	20	30	45	10	20	45	10	20	30	45	45
-10	-20	-30	-45	-10	-20	-45	-10	-20	-30	-45	-45
-40	-20	20	35	5	10	90	0	-45	45	0	-90

<u>Punto 8.</u> Haciendo uso de la GUI verifique que con la configuración calculada se obtiene la postura indicada de la herramienta. Haga capturas de pantalla resaltando con cotas los valores *x y z*, e incluya en el informe los resultados


forward
inverse
45
10
20
30
45
45
-20
19
69
60
20
0
60
40
20
0
-20
-40
-60
Y
-60
-40
-20
0
20
40
X
q5
q3
q1
q2
O
N
Epson 4


forward
inverse
-45
-10
-20
-30
-45
-45
25
25
44
80
20
0
Z
60
40
20
0
-20
-40
-60
X
-20
0
20
40
60
Y
A
q6
q3
N
O
4
q2
q1
Epson 4



<u>Punto 9.</u> Actualice la GUI para que tenga la opción de ingresar la posición de la herramienta, obtener la configuración del robot mediante cinemática inversa y observar el robot en la posición. Compruebe su funcionamiento.