UR3 Modelo Cinemático Inverso

Juan Pablo Romero Camacho, David Paez Ramirez, José Guillermo Guarnizo Marín.

Bogotá D.C

Universidad Santo Tomas

Grupo de Estudio y Desarrollo en Robótica GED

Facultad de Ingeniería Electrónica

juanpreomerc@usantotomas.edu.co, david.paezr@usantotomas.edu.co, jose.guarnizo@usantotomas.edu.co

Resumen- En el siguiente documento se presenta la investigación y el cálculo del modelo cinemático inverso del manipulador industrial UR3 mediante un método geométrico, posteriormente se realizan simulación mediante una comunicación entre Python y CoppeliaSim, de forma que se pueda comprobar que mediante el modelo cinemático directo e inverso se puede llegar a encontrar las posiciones finales o los valores de ángulos respectivamente.

Palabras clave: Cinemática inversa, manipulador industrial, Vrep, Simulación.

I. INTRODUCCIÓN

El modelo Cinemático directo sirve para encontrar la relación existente entre la posición y la rotación final con respecto a los ángulos de cada una de las uniones, ahora para el caso del modelo cinemático inverso, se quiere encontrar los ángulos que hacen al manipulador estar en una posición y rotación determinada con respecto al origen. De esta forma se puede interpretar.

$$(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = (\alpha, \beta, \gamma, x, y, z) \quad (1)$$

$$(\alpha, \beta, \gamma, x, y, z) = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \quad (2)$$

$$\begin{aligned} {}^0T(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \\ = {}^0T(\theta_1){}_1^2T(\theta_2){}_2^3T(\theta_3){}_3^4T(\theta_4){}_4^5T(\theta_5){}_5^6T(\theta_6) \end{aligned} \quad (3)$$

Para determinar el modelo cinemático inverso del manipulador, es necesario tener en cuenta la construcción de las matrices de transformación de este, la matriz más importante para la construcción del modelo es la que relaciona la posición final con respecto al origen, en este caso como el manipulador es de 6 grados de libertad se obtiene la matriz 0T .

La matriz 0T tendrá dos elementos importantes para el cálculo de la cinemática inversa del manipulador, la primera es la relación que existe de la rotación con respecto al origen y los valores de posición final.

$${}^{i-1}_iT \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$${}^0T \begin{bmatrix} {}^0X_x & {}^0Y_x & {}^0Z_x & {}^0P_x \\ {}^0X_y & {}^0Y_y & {}^0Z_y & {}^0P_y \\ {}^0X_z & {}^0Y_z & {}^0Z_z & {}^0P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

Para poder construir las matrices consecuentes del manipulador es necesario tener los parámetros de Denavit Hartenberg, en este caso, se plantea el siguiente modelo del manipulador, y sus correspondientes parámetros de acuerdo con la orientación de sus ejes.

Figura 1: Ejes de Coordenadas

Respetando las reglas del autor Craig [1], los ejes de coordenadas se mantienen con la rotación orientada en el eje z como aparece en la figura 1.

Tabla 1: Parámetros Denavit – Hartenberg 1

i	α_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	θ_1	d_1
2	90	0	θ_2	0
3	0	$-a_2$	θ_3	0
4	0	$-a_3$	θ_4	d_4
5	90	0	θ_5	d_5
6	-90	0	θ_6	d_6

Tabla 2: Parámetros Denavit – Hartenberg 2

i	α_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	θ_1	d_1
2	90	0	θ_2	0
3	0	a_2	θ_3	0
4	0	a_3	θ_4	d_4
5	90	0	θ_5	d_5
6	-90	0	θ_6	d_6

Como se puede observar el modelo es diferente con respecto al modelo inicial planteado, básicamente es una representación en espejo del modelo anterior, esto es debido a que en la literatura se utiliza un método geométrico basado en un robot UR5 [2], por lo que las distancias, deben de ser vistas desde el mismo punto para que dicho modelo funcione, teniendo en cuenta la orientación inicial del robot para su desarrollo, la tabla 1 muestra las distancias y correspondientes rotaciones del manipulador en cada una de sus uniones.

El cambio en los ejes solamente produce el cambio de signo de las distancias a_2 y a_3 de resto en la tabla 2, tanto en sus posiciones de sus ejes como en relación de distancias, permanecen igual.

II. PROCEDIMIENTO

1. Encontrando Theta 1 (θ_1)

Un análisis geométrico del manipulador permite visualizar la relación que existe entre cada una de sus uniones [3], en un principio se analiza el manipulador para determinar la posición del eje 5 con respecto al origen, este se puede determinar trasladando el eje de coordenadas de la posición final restándole la distancia de d_6 la cual corresponde a la distancia que existe entre el eje de coordenadas 5 y la posición final del manipulador.

Figura 2: Ejes de Coordenadas

El robot es analizado en orientación similar, pero en espejo, por lo que algunas distancias se encuentran reflejadas, el nuevo sistema de coordenadas de la figura 2, esta descrito por los siguientes parámetros.

Figura 3: Análisis Theta 1

la traslación del eje final al eje 5 descrita en la figura 3, puede ser descrita de la siguiente manera.

$${}^0_5P = {}^0_6P - d_6 * {}^0_6Z = {}^0_6T \begin{bmatrix} 0 \\ 0 \\ -d_6 \\ 1 \end{bmatrix} \quad (6)$$

Figura 4; Análisis Geométrico 0P5

Como se puede observar en la figura 4, la rotación de θ_1 es analizada con respecto a la posición del eje de coordenadas 5 en donde la distancia que existe entre el origen y la posición 5 está dada en términos del vector de posición de la matriz de transformación correspondiente a 5 con respecto a 0, la hipotenusa de la distancia en x y z (pues son los ejes en los que existe esa distancia) forma los ángulos φ_1 y φ_2

El ángulo de φ_1 se encuentra con la distancia existente entre la posición del eje de coordenadas 5 con respecto al origen, por lo que se puede escribir de la siguiente manera.

$$\varphi_1 = \text{atan2}({}^0_5P_y, {}^0_5P_x) \quad (7)$$

El ángulo φ_2 puede ser igualmente analizado con las distancias hasta el eje 5 con respecto al origen y la distancia d_4 como se muestra en la figura anterior, de forma que el ángulo de φ_2

$$\cos(\varphi_2) = \frac{d_4}{|{}^0_5P_{xy}|} \quad (8)$$

$$\varphi_2 = \pm \text{acos}\left(\frac{d_4}{|{}^0_5P_{xy}|}\right) \quad (9)$$

$$\varphi_2 = \pm \text{acos}\left(\frac{d_4}{\sqrt{({}^0_5P_x)^2 + ({}^0_5P_y)^2}}\right) \quad (10)$$

El ángulo de θ_1 puede ser entonces fácilmente encontrado con la relación entre los ángulos φ_1 y φ_2

$$\theta_1 = \varphi_1 + \varphi_2 + 90 \quad (11)$$

Como se puede observar debido a que el ángulo φ_2 es una función *acos*, el sistema del ángulo θ_1 posee dos respuestas posibles de solución, estas respuestas corresponden concretamente al manipulador orientado hacia la “izquierda” o la “derecha” dependiendo de la posición en la que este se encuentre.

$$\begin{aligned} \theta_1 &= \text{atan2}({}^0_5P_y, {}^0_5P_x) \\ &\pm \text{acos}\left(\frac{d_4}{\sqrt{({}^0_5P_x)^2 + ({}^0_5P_y)^2}}\right) + 90 \end{aligned} \quad (12)$$

2. Encontrando Theta 5(θ_5)

En el ángulo anterior se hizo uso de la posición del eje de coordenadas 5 con respecto al origen, por lo que esas posiciones ya están calculadas y servirán para determinar el ángulo de θ_5 , el cual, pues es la rotación asociada a esa unión, por lo que se plantea.

Figura 5; Análisis Geométrico 0P6

En la figura 5, el componente en y de la posición 6 con respecto a 1 solamente depende de θ_5 , la distancia existente entre los ejes se relaciona con la expresión.

$$-{}^1_6P_y = d_4 + d_6 \cos(\theta_5) \quad (13)$$

Y la posición con de 6 con respecto a 1 puede ser determinada con la matriz de rotación de 0 con respecto a 1 y la distancia de 6 con respecto a 1.

$${}^1P_y = {}^0P_x \cdot (-\sin \theta_1) + {}^0P_y \cdot \cos \theta_1 \quad (14)$$

$$-d_4 - d_6 \cos \theta_5 = {}^0P_x(-\sin \theta_1) + {}^0P_y \cos \theta_1 \quad (15)$$

$$\cos \theta_5 = \frac{{}^0P_x \sin \theta_1 - {}^0P_y \cos \theta_1 - d_4}{d_6} \quad (16)$$

$$\theta_5 = \pm \arccos \frac{{}^0P_x \sin(\theta_1) - {}^0P_y \cos(\theta_1) - d_4}{d_6} \quad (17)$$

Como se puede observar el ángulo tiene dos soluciones, de forma que el manipulador puede llegar a la posición deseada con el codo arriba o abajo.

3. Encontrando Theta 6 (θ_6)

Para determinar θ_6 se examina y_1 visto desde el eje 6, este ángulo es descrito en coordenadas esféricas, donde el azimut es $-\theta_6$ y el ángulo polar corresponde a θ_5 , como se puede observar es paralelo a los ejes 2,3 y 4 en el eje z y esto solo cambiara con el movimiento de θ_5 .

Figura 6: Coordenadas Esféricas θ_5 y θ_6

En este caso se puede determinar el ángulo correspondiente, visto en la figura 6, gracias a asumir el sistema en coordenadas esféricas de modo tal que 6Y queda expresado.

$${}^6Y = \begin{bmatrix} \sin(\theta_5) \cos(-\theta_6) \\ \sin(\theta_5) \sin(-\theta_6) \\ \cos(\theta_5) \end{bmatrix} \quad (18)$$

$${}^6Y = \begin{bmatrix} -\sin(\theta_5) \cos(\theta_6) \\ \sin(\theta_5) \sin(\theta_6) \\ -\cos(\theta_5) \end{bmatrix} \quad (19)$$

$${}^6Y = {}^0X * -\sin(\theta_1) + {}^0Y * \cos(\theta_1) \quad (20)$$

De esta forma es bastante práctico, puesto que la matriz de transformación ya es conocida desde un comienzo, además que el ángulo θ_1 ya es conocido, y realizando la igualación en el sistema de coordenadas esféricas descritas.

$$\begin{bmatrix} -\sin(\theta_5) \cos(\theta_6) \\ \sin(\theta_5) \sin(\theta_6) \\ -\cos(\theta_5) \end{bmatrix} = {}^0X * -\sin(\theta_1) + {}^0Y * \cos(\theta_1) \quad (21)$$

$$-\sin(\theta_5) \cos(\theta_6) = {}^0X_x * -\sin(\theta_1) + {}^0Y_x * \cos(\theta_1) \quad (22)$$

$$\sin(\theta_5) \sin(\theta_6) = {}^0X_y * -\sin(\theta_1) + {}^0Y_y * \cos(\theta_1) \quad (23)$$

$$\cos(\theta_6) = \frac{{}^0X_x * -\sin(\theta_1) + {}^0Y_x * \cos(\theta_1)}{-\sin(\theta_5)} \quad (24)$$

$$\sin(\theta_6) = \frac{{}^0X_y * -\sin(\theta_1) + {}^0Y_y * \cos(\theta_1)}{\sin(\theta_5)} \quad (25)$$

$$\theta_6 = \text{atan2}(\cos(\theta_6), \sin(\theta_6)) \quad (26)$$

4. Encontrando Theta 3 (θ_3)

Ahora se analizarán las tres uniones restantes del manipulador, se puede observar que cada una de estas uniones se encuentran en paralelo una con respecto a la otra, esto es más notable en el sistema de coordenadas del manipulador presentes en la imagen al comienzo del documento, lo que hace que esta parte del manipulador pueda considerarse como un sistema de 3 grados de libertad planar.

Figura 7: Análisis Geométrico θ_3

Se puede comenzar por determinar la matriz de transformación de 4 con respecto a 1, ya que para este punto se conocen 0T_1 , 4T_5 y 5T_6 , a parte de esto es claro que la distancia medida en x y z entre 4 y 1 dependen únicamente de θ_3 , en la figura 7 se muestra el ángulo que determina la distancia por lo que ya se tiene una relación directa.

$${}^4T_6 = {}^4T_5 * {}^5T_6 \quad (27)$$

$${}^6T_4 = {}^4T_6^{-1} \quad (28)$$

$${}^1T_0 = {}^0T_1^{-1} \quad (29)$$

$${}^1T_6 = {}^1T_0 * {}^0T_6 \quad (30)$$

$${}^1T_4 = {}^1T_6 * {}^6T_4 \quad (31)$$

$$\theta_3 = \pm \arccos^2(|{}^1P_{xz}|^2 - (a_2)^2 - (a_3)^2) / (2a_2a_3) \quad (32)$$

5. Encontrando Theta 2(θ_2)

El ángulo de θ_2 puede ser encontrado con la relación de $\phi_1 - \phi_2$ cada uno de los ángulos puede ser determinado de la imagen correspondiente al análisis geométrico de θ_3 .

$$\phi_1 = \text{atan2}(-{}^1P_z, -{}^1P_x) \quad (33)$$

$$\frac{\sin(\phi_2)}{-a_3} = \frac{\sin(\phi_3)}{|{}^1P_{xz}|} \quad (34)$$

$$\phi_2 = \text{asin}\left(\frac{-a_3 \sin(\phi_3)}{|{}^1P_{xz}|}\right) \quad (35)$$

ϕ_3 puede ser reemplazado con θ_3 teniendo en cuenta que $\sin \phi_3 = \sin(180^\circ - \theta_3) = \sin(\theta_3)$ combinando las ecuaciones se obtiene.

$$\theta_2 = \phi_1 - \phi_2 = \text{atan2}(-{}^1P_z, -{}^1P_x) - \text{asin}\left(\frac{-a_3 \sin(\phi_3)}{|{}^1P_{xz}|}\right) \quad (36)$$

6. Encontrando Theta 4(θ_4)

El Angulo restante θ_4 se define según los parámetros Denavit Hartenver, como el ángulo visto entre x_3 y x_4 visto a través de z_4 , lo cual puede ser fácilmente obtenido de la matriz de transformación 3T_4 , al igual que en ángulos anteriores, esta

matriz puede ser determinada con las demás que han sido calculadas.

$${}^1T_3 = {}^1T_2 * {}^2T_3 \quad (37)$$

$${}^3T_1 = {}^1T_3^{-1} \quad (38)$$

$${}^3T_4 = {}^3T_1 * {}^1T_4 \quad (39)$$

De esta forma obtenida la matriz de transición de 3 con respecto a 4 solamente es necesario tomar los componentes con respecto a x y hacer $\tan^{-1}(x_y, x_x)$.

$$\theta_4 = \text{atan2}({}^3X_y, {}^3X_x) \quad (40)$$

III. SIMULACIONES

Una vez resuelto el modelo cinemático inverso, este se programa en Python y se realiza una conexión de Python y CoppeliaSim para mover el manipulador de acuerdo con los valores de posición y rotación que ingrese el usuario.

El modelo cinemático directo también se encuentra dentro del programa para verificar que los resultados obtenidos concuerden con los valores de ángulos dados al modelo cinemático directo.

Primero se declaran las distancias y ángulos correspondientes a los parámetros de Denavit Hartenberg con los que se calculan las matrices de transformación

```
Dist = [0.1519,-0.24365,-
        0.21325,0.11235,0.08535,0.0819]
```

```
Alpha = [0,90,0,0,90,-90]
```

Para el modelo cinemático directo las matrices de transformación son calculadas mediante la función de "Rot", dentro de la función "Movimiento".

```
T_01 = Rot(Theta[0],Alpha[0],[0,0,Dist[0]])
```

```
T_12 = Rot(Theta[1],Alpha[1],[0,0,0])
```

```
T_23 = Rot(Theta[2],Alpha[2],[Dist[1],0,0])
```

```
T_34 = Rot(Theta[3],Alpha[3],[Dist[2],0,Dist[3]])
```

```
T_45 = Rot(Theta[4],Alpha[4],[0,Dist[4],0])
```

```
T_56 = Rot(Theta[5],Alpha[5],[0,Dist[5],0])
```

La función “Rot” es la matriz con la que se calculan las respectivas matrices de transformación, por lo que como argumento de la función se debe colocar el ángulo correspondiente de la unión, el Alpha dado anteriormente por los parámetros de Denavit Hartenberg y las distancias también dadas en los parámetros de Denavit Hartenberg.

Si se quiere calcular el modelo cinemático directo se hace uso de la función movimiento que tiene como parámetro la lista con los 6 ángulos del manipulador.

```
def Movimiento(G):
    return T_06
```

La función regresa la matriz de transformación correspondiente al extremo del manipulador con respecto al origen.

Si se desea calcular el modelo cinemático directo, lo que recibe la función es la matriz de transformación de 6 con respecto a 0 y a partir de ello realiza el cálculo de los ángulos.

```
def Cinematica_Inversa(M):
    return Theta1, Theta2, Theta3, Theta4, Theta5, Theta6
```

En este caso la función retorna los valores de ángulos a los cuales debe desplazarse el manipulador, de esta forma ya se tiene la base para indicar los movimientos en CoppeliaSim, por lo que se procede a realizar la conexión.

```
vrep.simxFinish(-1)

clientID=vrep.simxStart('127.0.0.1',19999,True,5000,5) # Connect to Coppelia Sim
if clientID!=-1:
    print ('Conectado a Vrep')
else:
    print ('Fallo')
    sys.exit("No se pudo conectar")
time.sleep(1)

Joints = []
```

```
for i in range(0,7):
    errorcode, Robot_Joint = vrep.simxGetObjectHandle(clientID,'UR3_joint'+str(i),vrep.simx_opmode_blocking)
    Joints.append(Robot_Joint)
```

Donde “Joints” es una lista con las direcciones de cada una de las uniones del manipulador en CoppeliaSim, de esta forma cada se pueden almacenar para posteriormente moverlas.

La función del movimiento de las uniones esta descrita como “Move_Robot” y como parámetro recibe los ángulos calculados mediante el modelo cinemático inverso para desplazar al robot.

```
def Move_Robot(Target):
    for Joint in range(6,-1,-1):
        errorcode = vrep.simxSetJointTargetPosition(clientID,Joints[Joint],Target[Joint]*math.pi/180,vrep.simx_opmode_blocking)
        time.sleep(0.05)
```

cómo se puede ver se empieza a mover cada una de las uniones desde la punta hasta la base, para que el movimiento no sea tan brusco, de igual forma que se aplica un atraso (Delay).

Finalmente, la interfaz gráfica necesaria para la interacción remota del manipulador desde un laboratorio con el usuario [4], está diseñada para ingresar los 6 valores correspondientes a rotación y posición en el manipulador además de un botón de trayectoria, el cual hace que el robot llegue a la misma posición, pero con diferentes, rotaciones.

Figura 8: Interfaz de Usuario - Modelo Cinemático Inverso

Para construir la matriz 0T con valores de rotación y posición se, toman los ángulos de rotación con respecto a cada uno de los ejes (x, y, z) y con los tres ángulos se calcula la rotación del eje mediante los elementos de entrada de texto de la interfaz mostrada en la figura 8, los elementos son utilizados para la rotación y traslación mediante las siguientes matrices.

$$M_x \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(R_x) & -\sin(R_x) & 0 \\ 0 & \sin(R_x) & \cos(R_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (41)$$

$$M_y \begin{bmatrix} \cos(R_y) & 0 & \sin(R_y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(R_y) & 0 & \cos(R_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (42)$$

$$M_z \begin{bmatrix} \cos(R_z) & -\sin(R_z) & 0 & 0 \\ \sin(R_z) & \cos(R_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (43)$$

$${}^0T = M_x * M_y * M_z * \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \quad (44)$$

Con la matriz de transformación creada se puede pasar directamente a la función de cinemática inversa, en el caso que no exista solución o la solución este en un punto de singularidad, el manipulador no se moverá.

En este caso se realiza la simulación del manipulador para una trayectoria en donde se desplaza al mismo punto, pero con 30 ángulos distintos.

```
for i in range (0,30):
    Rx = 10*i
    Ry = 300 - (10*i)
    Rz = 0
    Px = 0.25
    Py = 0.01
    Pz = 0.015
```

Para lo cual se obtiene el siguiente resultado en el movimiento del robot, desde la simulación en CoppeliaSim.

Figura 9a: Trayectoria - Robot en Estado Inicial

Figura 9b: Trayectoria - Robot (i = 0)

Figura 9c: Trayectoria - Robot (i=15)

Figura 9d: Trayectoria - Robot (i=29)

El programa realizado en Python permite la manipulación del robot, más sin embargo un trabajo a futuro es la manipulación de las velocidades, como se puede observar en las figuras 9a, 9b, 9c y 9d, el manipulador se traslada efectivamente a las posiciones, pero puede ser bastante brusco debido a que se está moviendo la unión sin algún tipo de control.

Finalmente, para demostrar el funcionamiento se mostrará como seleccionando los valores de origen (ángulos [0,0,0,0,0,0] si se tratara del modelo cinemático directo), es decir la matriz de transformación 0T , para construirla se puede realizar con la imagen del manipulador en su estado inicial, es decir la posición con la que se encontraron sus valores de

Denavit Hartenver, de esta forma se nota que en cuanto a rotación, solamente el eje x roto 90 grados con respecto al origen, y la posición final es $[-0.4569, -0.19245, 0.06655]$, por lo que al ingresar esos valores en la interfaz de usuario debería obtenerse la solución, para acomodar al robot en el origen.

Figura 10: Manipulador, Posición de Origen Encontrada

Como se puede observar en la figura 10, al poseer varias soluciones en 3 de sus ángulos, hace que se llegue al mismo punto con la misma rotación con respecto al eje de origen, más sin embargo no en los mismos ángulos.

Se puede observar el comportamiento del robot, las diferentes acciones que se aplicaron y profundizar en el código desarrollado en Python en el siguiente video de trabajo: <https://youtu.be/8AI9yLs4MRA>.

IV. CONCLUSIONES

El modelo cinemático inverso del manipulador tiene diferentes soluciones en 3 de sus 6 grados de libertad, lo que permite que el manipulador pueda desplazarse con cierta libertad, sin embargo, existen limitaciones de este, en las cuales no puede funcionar correctamente, como lo son los puntos extremos del manipulador, en los cuales no puede llegar además de provocar singularidades en las velocidades del sistema, por lo que tener en cuenta esos factores es algo importante.

En este caso particular la conexión con CoppeliaSim fue construida efectivamente mediante Python, en el software de CoppeliaSim LUA existe una librería especializada en movimiento de manipuladores con velocidades y aceleraciones para que estos tengan movimientos más fluidos, sin embargo esta librería está diseñada para trabajar en LUA y la conexión del sistema Python – CoppeliaSim no permite el uso de esta librería, específicamente por que los retardos en la comunicación no son factibles en el procesamiento que hacen las funciones de la librería en tiempo real, sin embargo el manipulador puede ser desplazado y si se quiere dan un mayor funcionamiento, el modelo cinemático inverso del manipulador puede ser utilizado en CoppeliaSim.

La conexión de Python con CoppeliaSim, permite el uso de algunas herramientas que por si mismo LUA no permite, como lo es el uso de interfaces graficas (tkinter) utilizado en este trabajo de forma muy sutil para la interfaz de usuario en el posicionamiento final del robot.

El modelo UR3 de Universal Robotics, posee versiones de mayor tamaño, UR5 y UR10 las cuales, al ser estructuralmente el mismo robot, pueden tener el mismo modelo cinemático inverso, solo que teniendo en cuenta el cambio en sus distancias.

Como se puede observar se puede llegar a un mismo punto desde muchos ángulos distintos, el modelo cinemático inverso debe tener en cuenta el área de trabajo en la cual se va a implementar el manipulador, para así asegurar la integridad del personal de trabajo y del manipulador mismo.

REFERENCIAS

- [1] J J.-Craig “Robótica”, Pearson Educación, vol. 3, pp. 62-100, 2006
- [2] Andersen, R. S. (2018). Kinematics of a UR5. Aalborg University: Aalborg, Denmark.
- [3] Hawkins, K. P. (2013). Analytic inverse kinematics for the universal robots ur-5/ur-10 arms. Technical report, Georgia Institute of Technology. Available at: https://smartech.gatech.edu/bitstream/handle/1853/50782/ur_kin_tech_report_1.pdf.
- [4] Martinez S.J., Guarnizo J.G., Avendano J. (2021) Design and Implementation of a Remote Virtual Laboratory by Internet Applied to KUKA LBR IIWA 14 R820. In: Cortes Tobar D., Hoang Duy V., Trong Dao T. (eds) AETA 2019 - Recent Advances in Electrical Engineering and Related Sciences: Theory and Application. AETA 2019. Lecture Notes in Electrical Engineering, vol 685. Springer, Cham. https://doi.org/10.1007/978-3-030-53021-1_45