

UNIVERSIDAD POLITÉCNICA DE MADRID

ROBÓTICA
MANIPULADORES

Estudio del manipulador *μArm*

Autores:

Javier Alonso Silva - javier.asilva@alumnos.upm.es
Roberto Álvarez Garrido - roberto.alvarezg@alumnos.upm.es
José Alejandro Moya Blanco - alejandro.moya.blanco@alumnos.upm.es

Última modificación: 17 de noviembre de 2019



Conocimientos previos

Antes de ponernos a hablar sobre los resultados obtenidos en la práctica, antes vamos a hablar sobre algunas características básicas del brazo robótico e introducirlo brevemente. El manipulador robótico μ Arm es un dispositivo creado por la empresa [UFACTORY](#) el cual cuenta con cuatro grados de libertad. De dichos grados de libertad, tres son usados para mover el brazo robótico hasta ciertas posiciones y, el último, para mantener el extremo del mismo paralelo al suelo.

El manipulador es controlado mediante cuatro motores:

El **motor de la base** el cual permite la rotación del manipulador.

En el brazo, el **motor que está a la derecha** (ver la figura 1), coordina el movimiento de la parte inferior del brazo (*Lower Arm* en la figura) con la parte superior del mismo (*Upper Arm* en la figura).

En esta parte del manipulador, el movimiento es como el de un flexo: la parte superior del flexo está supeditada a la parte inferior, de manera que se mantiene de forma constante la altura a la que está el extremo final del mismo.

El otro motor, localizado a la **izquierda del brazo**, se encarga de mantener la orientación del extremo del manipulador. De esta manera, dicho extremo permanecerá paralelo al suelo. Teniendo en cuenta esto, podríamos decir que el robot en verdad solo tiene tres grados de libertad en tanto a que no se controla directamente el movimiento del último grado, ya que al final se mueve para permanecer paralelo al suelo.

El **motor localizado en el extremo**, con el cual se puede actuar sobre el elemento que esté colocado allí. Por ejemplo, cuando se coloca una ventosa permite rotarla o, cuando se coloca la pinza, el movimiento del motor permite abrirla o cerrarla.

Para este estudio, este último motor se descartará, ya que no afecta a las posiciones accesibles por el robot.

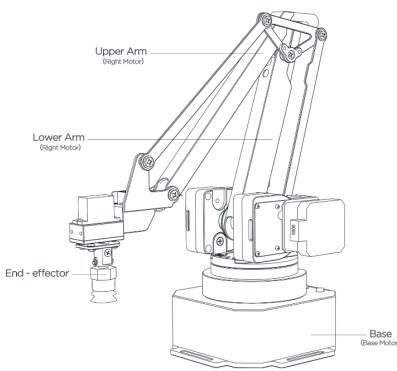


Figura 1: zonas de actuación de los motores en el brazo [1]

Motor	Rango de trabajo
Base	0° ~ 180°
Derecho	0° ~ 130°
Izquierdo	0° ~ 106°
Extremo	0° ~ 180°

Cuadro 1: ángulo de giro de los motores [2]

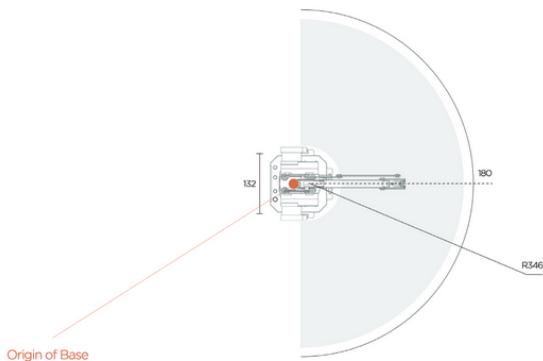


Figura 2: rango del manipulador μ Arm [3]

Toda la información relativa al desarrollo del proyecto puede ser encontrada en [GitHub - UPM Robotics \[4\]](#). Allí están detallados los distintos hitos a conseguir así como más información sobre el robot.

Además, se encuentra disponible la siguiente bibliografía:

- [Manual de usuario](#)
- [Especificaciones](#)
- [Guía del desarrollador](#)
- [Modelo en 3D](#)
- [Web de UFACTORY](#)
- [Soporte de UFACTORY](#)

1. Configuración geométrica

En esta sección vamos a describir la configuración geométrica del brazo robótico. La configuración que obtuvimos fue la siguiente:

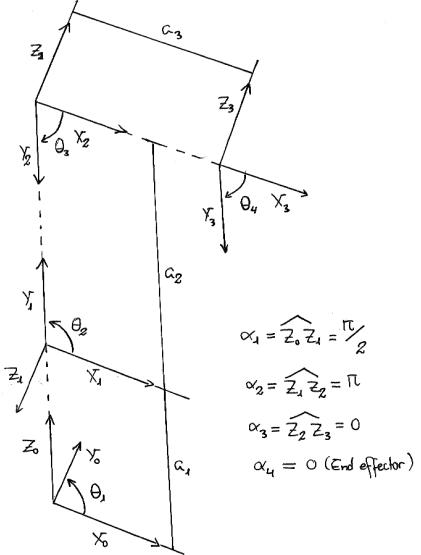


Figura 3: configuración geométrica del robot

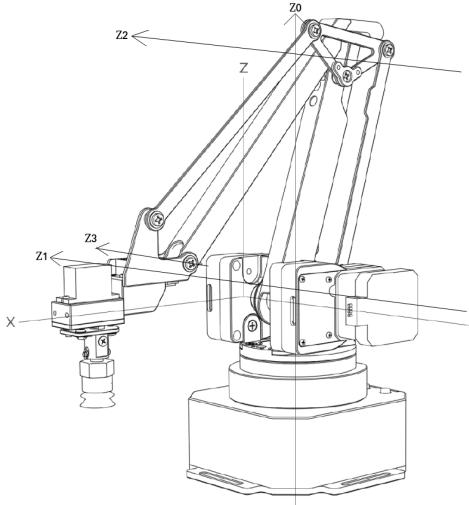


Figura 4: los grados de libertad del brazo, representados por los diferentes Z_i

Usando los datos que están presentes en la documentación al desarrollador [1], pudimos obtener los siguientes datos para los a_i (distancia entre ejes) del manipulador; además, descubrimos que hay una pequeña desviación d_i entre las articulaciones {1, 2}:

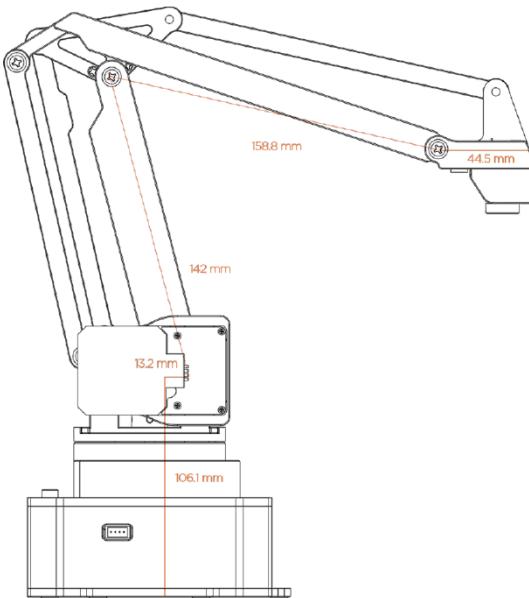


Figura 5: longitudes del brazo robótico [1]

i	a_i (mm.)	d_i (mm.)
1	106,1	13,2
2	142	0
3	158,8	0
4	44,5	0

Cuadro 2: longitudes y desviaciones del manipulador

De esta manera, con los datos obtenidos, generamos una primera matriz de *Denavit-Hartenberg*:

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	13,2	106,1	$\frac{\pi}{2}$
2	θ_2	0	142	π
3	θ_3	0	158,8	0
4	θ_4	0	44,5	0

Cuadro 3: primera tabla de *Denavit-Hartenberg*

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	d_1	a_1	$\frac{\pi}{2}$
2	θ_2	0	a_2	π
3	θ_3	0	a_3	0
4	θ_4	0	a_4	0

Cuadro 4: primera tabla de *Denavit-Hartenberg* parametrizada

La cuestión es que, al hacer los distintos modelos, descubrimos que debido a la orientación del brazo robótico, la d_i está en la posición del equivalente a_i , en particular d_1 y a_1 . Esto es debido a que, como se puede ver en la figura 4 junto con las figuras 3 y 5, el plano sobre el que está d_i es el XZ , lo cual nos obliga a cambiar las posiciones en las que existen a la vez un a_i y un d_i , siempre y cuando ambos sean constantes, que en este caso lo son. De esta forma, la tabla de *Denavit-Hartenberg* quedaría:

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	106,1	13,2	$\frac{\pi}{2}$
2	θ_2	0	142	π
3	θ_3	0	158,8	0
4	θ_4	0	44,5	0

Cuadro 5: segunda tabla de *Denavit-Hartenberg*

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	a_1	d_1	$\frac{\pi}{2}$
2	θ_2	0	a_2	π
3	θ_3	0	a_3	0
4	θ_4	0	a_4	0

Cuadro 6: segunda tabla de *Denavit-Hartenberg* parametrizada

Finalmente, el *end-effector* (véase la figura 1) siempre está perpendicular al plano del suelo, es decir, $\phi_e = \pi$. Por eso mismo, el parámetro $i = 4$ en verdad está supeditado siempre al movimiento que se realice según los ángulos θ_2 y θ_3 , así que no es necesario contemplarlo en la tabla de *Denavit-Hartenberg*:

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	106,1	13,2	$\frac{\pi}{2}$
2	θ_2	0	142	π
3	θ_3	0	158,8	0

Cuadro 7: tabla de *Denavit-Hartenberg*

i	θ_i	d_i (mm.)	a_i (mm.)	α_i
1	θ_1	a_1	d_1	$\frac{\pi}{2}$
2	θ_2	0	a_2	π
3	θ_3	0	a_3	0

Cuadro 8: tabla de *Denavit-Hartenberg* parametrizada

2. Cinemática directa

A continuación, con los valores obtenidos en el apartado anterior, vamos a obtener las distintas matrices de transformación de referenciales de manipuladores. Para ello, partimos de la siguiente matriz:

$$A_{i-1}^i = \begin{pmatrix} \cos \theta_i & -\cos \alpha_i \sin \theta_i & \sin \alpha_i \sin \theta_i & a_i \cos \theta_i \\ \sin \theta_i & \cos \alpha_i \cos \theta_i & -\sin \alpha_i \cos \theta_i & a_i \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Aplicando los distintos pasos, obtenemos:

$$A_0^1 = \begin{pmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & d_1 \cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & d_1 \sin(\theta_1) \\ 0 & 1 & 0 & a_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_1^2 = \begin{pmatrix} \cos(\theta_2) & \sin(\theta_2) & 0 & a_2 \cos(\theta_2) \\ \sin(\theta_2) & -\cos(\theta_2) & 0 & a_2 \sin(\theta_2) \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_2^3 = \begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & a_3 \cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & a_3 \sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Con todas las matrices ya obtenidas, podemos calcular la matriz de transformación directa del manipulador (son 4 columnas, sin embargo no cabe la matriz al completo y por eso la última columna está debajo):

$$A_0^2 = \begin{pmatrix} \cos(\theta_1) \cos(\theta_2) & \sin(\theta_2) \cos(\theta_1) & -\sin(\theta_1) & (a_2 \cos(\theta_2) + d_1) \cos(\theta_1) \\ \sin(\theta_1) \cos(\theta_2) & \sin(\theta_1) \sin(\theta_2) & \cos(\theta_1) & (a_2 \cos(\theta_2) + d_1) \sin(\theta_1) \\ \sin(\theta_2) & -\cos(\theta_2) & 0 & a_1 + a_2 \sin(\theta_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_0^3 = \begin{pmatrix} \cos(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_2 - \theta_3) \cos(\theta_1) & -\sin(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_1) \sin(\theta_2 - \theta_3) & \cos(\theta_1) & \cos(\theta_1) \\ \sin(\theta_2 - \theta_3) & -\cos(\theta_2 - \theta_3) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \cos(\theta_1) & (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) & a_1 + a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3) & 1 \end{pmatrix}$$

Las matrices están parametrizadas. Numéricamente, la matriz de transformación directa A_0^3 quedaría:

$$A_0^3 = \begin{pmatrix} \cos(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_2 - \theta_3) \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_1) \sin(\theta_2 - \theta_3) & \cos(\theta_1) \\ \sin(\theta_2 - \theta_3) & -\cos(\theta_2 - \theta_3) & 0 \\ 0 & 0 & 0 \\ (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \cos(\theta_1) & (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \sin(\theta_1) & 142,0 \sin(\theta_2) + 158,9 \sin(\theta_2 - \theta_3) + 106,1 \\ (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \sin(\theta_1) & 142,0 \sin(\theta_2) + 158,9 \sin(\theta_2 - \theta_3) + 106,1 & 1 \end{pmatrix}$$

Finalmente, hay que añadir una traslación¹ en el eje Z (debido a la posición del *end-effector*) y en el eje X , ya que después de la articulación θ_3 hay una extensión de 44,5 mm. (ver figura 5), para obtener así la posición final del robot ($X_e Y_e Z_e$):

$$A_0^3 = \begin{pmatrix} \cos(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_2 - \theta_3) \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_1) \sin(\theta_2 - \theta_3) & \cos(\theta_1) \\ \sin(\theta_2 - \theta_3) & -\cos(\theta_2 - \theta_3) & 0 \\ 0 & 0 & 0 \\ (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \cos(\theta_1) + T_X & (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) & a_1 + a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3) - T_Z \\ (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) & a_1 + a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3) - T_Z & 1 \end{pmatrix} \quad (1)$$

$$A_0^3 = \begin{pmatrix} \cos(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_2 - \theta_3) \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) \cos(\theta_2 - \theta_3) & \sin(\theta_1) \sin(\theta_2 - \theta_3) & \cos(\theta_1) \\ \sin(\theta_2 - \theta_3) & -\cos(\theta_2 - \theta_3) & 0 \\ 0 & 0 & 0 \\ (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \cos(\theta_1) + 44,5 & (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \sin(\theta_1) & 142,0 \sin(\theta_2) + 158,9 \sin(\theta_2 - \theta_3) + 106,1 - T_Z \\ (142,0 \cos(\theta_2) + 158,9 \cos(\theta_2 - \theta_3) + 13,2) \sin(\theta_1) & 142,0 \sin(\theta_2) + 158,9 \sin(\theta_2 - \theta_3) + 106,1 - T_Z & 1 \end{pmatrix} \quad (2)$$

De esta forma, obtendríamos las siguientes ecuaciones para ($X_e Y_e Z_e$):

$$\left. \begin{array}{l} X_e = (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \cos(\theta_1) + T_X \\ Y_e = (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) \\ Z_e = a_1 + a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3) - T_Z \end{array} \right\} \quad (3)$$

Además, debido al diseño geométrico de este brazo robótico, la orientación del *end-effector* es siempre perpendicular al suelo ($\phi_e = \frac{\pi}{2}$), pero esto no aporta información útil al robot, ya que mantiene siempre esta orientación (véase las figuras 3 y 4). Por este motivo, durante el desarrollo posterior se asumirá que:

$$\phi_e = \phi_{23} = \theta_2 - \theta_3 \quad (4)$$

y que, por consiguiente, los puntos útiles en este brazo robótico son:

$$P = (X_e, Y_e, Z_e, \phi_{23}) \quad (5)$$

¹ T_X es la traslación en X , mientras que T_Z es la traslación en Z

3. Cinemática inversa

Una vez tenemos las ecuaciones correspondientes a $(X_e \ Y_e \ Z_e)$, podemos calcular la cinemática inversa del μ Arm. Con dicha cinemática, se pueden obtener los ángulos $(\theta_1, \theta_2, \theta_3)$ del brazo robótico.

Antes de realizar los cálculos, se planteó el modelo geométrico del robot, para poder deducir los distintos ángulos que se podían formar y cómo estaba distribuido el brazo:

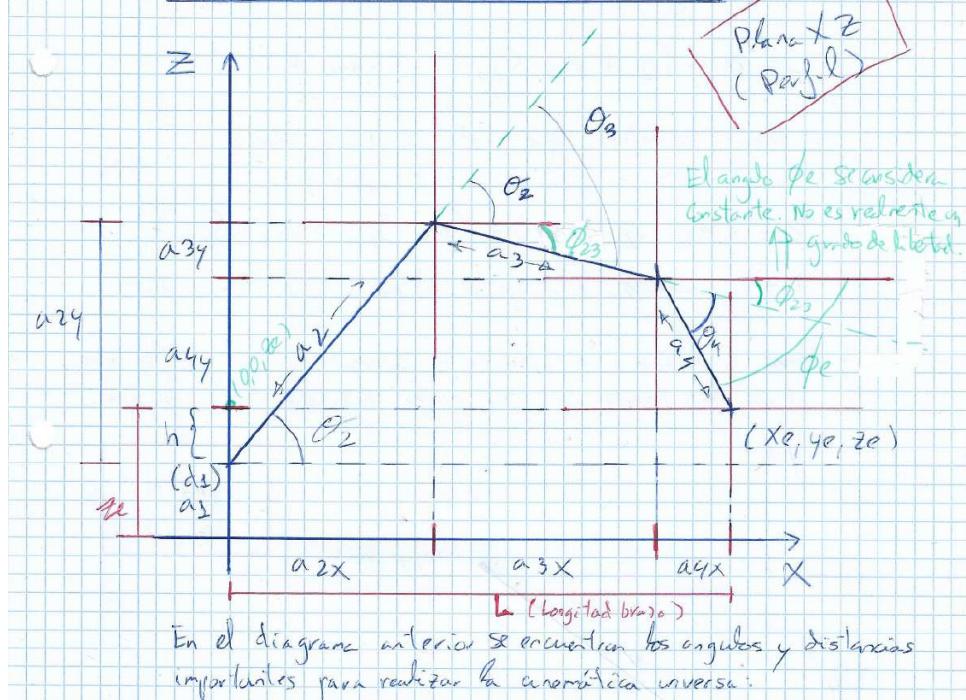


Figura 6: distribución geométrica del brazo

En la figura 6 se puede ver cómo, en el plano XZ (de perfil), se encuentra presente una altura inicial a_1 correspondiente a la base del brazo, así como una pequeña desviación en los ejes d_1 la cual afecta a la posición en Z . A continuación, cada extremidad del brazo está definida junto con la correspondiente articulación θ_i . Como se puede observar, el ángulo θ_2 es positivo pero, debido a cómo está configurado el robot geométricamente, los ángulos θ_3 y θ_4 han de ser negativos².

Como se vio anteriormente en la sección *Conocimientos previos*, en particular la figura 3, se suprimió el último parámetro de la tabla de Denavit–Hartenberg ya que dicho ángulo siempre es perpendicular al plano del suelo y, por ende, está supeditado a los ángulos anteriores θ_2 y θ_3 . En su lugar, se sustituye por una traslación en el eje Z : T_Z (véase las matrices 1 y 2 y la ecuación 3).

De esta forma, se obtuvieron las siguientes ecuaciones para el plano XZ :

$$L = \begin{cases} a_{2X} = a_2 \cdot \cos(\theta_2) \\ a_{3X} = a_3 \cdot \cos(\theta_2 + \theta_3) \\ a_{4X} = a_4 \cdot \cos(\phi_e) \end{cases} \quad (6)$$

$$H = \begin{cases} a_{2Z} = a_2 \cdot \sin(\theta_2) \\ a_{3Z} = a_3 \cdot \sin(\theta_2 + \theta_3) \\ a_{4Z} = a_4 \cdot \sin(\phi_e) \end{cases} \quad (7)$$

²véase la figura 5 para más información sobre la configuración del brazo robótico

Finalmente, se evaluó el plano XY del modelo geométrico del brazo robótico:

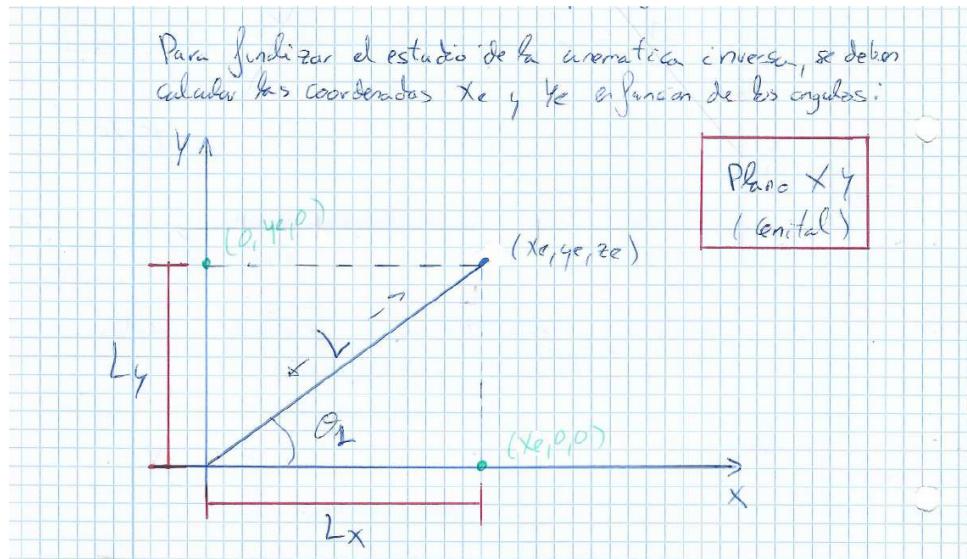


Figura 7: plano XY del brazo robótico

De esta manera, se obtuvieron las siguientes ecuaciones para L_X y L_Y :

$$\begin{cases} L_X = L \cdot \cos(\theta_1) = X_e \\ L_Y = L \cdot \sin(\theta_1) = Y_e \end{cases} \quad (8)$$

La problemática de plantear el modelo de esta forma se mostró a la hora de intentar resolver las ecuaciones de X_e , Y_e y Z_e : al sustituir en la ecuación 3, el sistema a resolver era demasiado complejo, resultando imposible la obtención de alguno de los ángulos, con lo que el sistema no tenía solución.

Como ya se comentó anteriormente, en la tabla de Denavit-Hartenberg se eliminó el último parámetro: pese a que sea un grado de libertad, al estar supeditado a los dos ángulos anteriores y ser siempre perpendicular al plano del suelo, no es necesario tenerlo en cuenta para la obtención de la inversa del robot.

Esto produjo que el último punto efectivo del robot resultara ser el extremo del brazo de la articulación θ_3 :

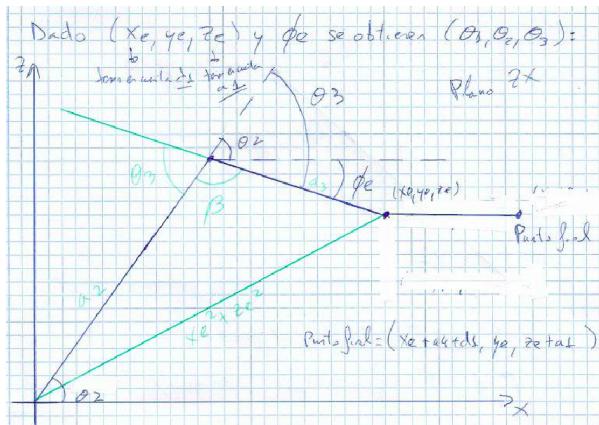


Figura 8: aproximación geométrica del μ Arm

De esta forma, la complejidad se reduce bastante, ya que ahora en el plano XZ solo se cuentan con dos grados de libertad: θ_2 y θ_3 , además de la traslación en X que se comentó anteriormente (T_X) y que se puede ver en la figura 8.

Esta nueva configuración permite aplicar el teorema del coseno³:

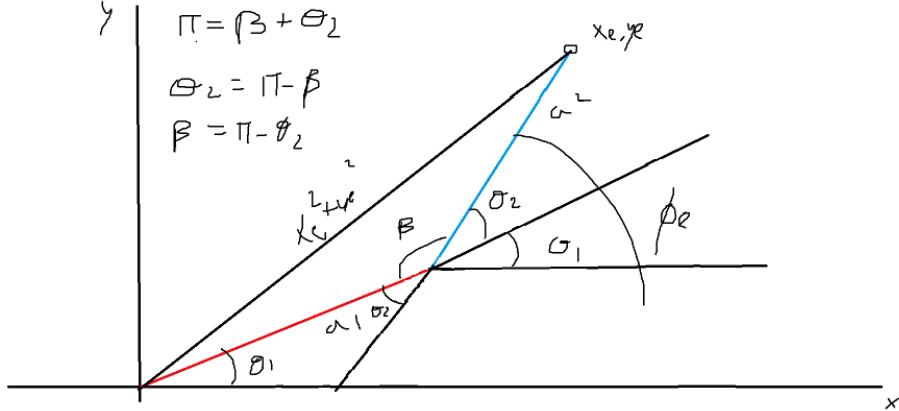


Figura 9: configuración del brazo robótico donde se puede aplicar el teorema del coseno

Así, se pueden extraer θ_2 y θ_3 :

$$\pi = \beta + \theta_3 \implies \beta = \pi - \theta_3 \quad (9)$$

$$\phi_{23} = \theta_2 - \theta_3 \quad (10)$$

En este caso, ϕ_e siempre va a ser $\frac{\pi}{2}$, por lo que no podemos obtener directamente el valor de θ_2 , ya que no se puede obtener directamente ϕ_{23} . Por este motivo, como se comentó anteriormente, el valor de $\phi_e = \phi_{23}$ (véase las ecuaciones 4 y 5).

Usando la ecuación 3, podemos despejar:

$$\begin{aligned} X_e^2 + Z_e^2 &= a_2^2 + a_3^2 - 2a_2a_3 \cos(\beta) = \\ &= a_2^2 + a_3^2 - 2a_2a_3 \cos(\pi - \theta_3) = \\ &= a_2^2 + a_3^2 + 2a_2a_3 \cos(\theta_3) \end{aligned}$$

$$\cos(\theta_3) = \frac{X_e^2 + Z_e^2 - a_2^2 - a_3^2}{2a_2a_3} \quad (11)$$

$$\sin(\theta_3) = \sqrt{1 - \cos^2(\theta_3)} \quad (12)$$

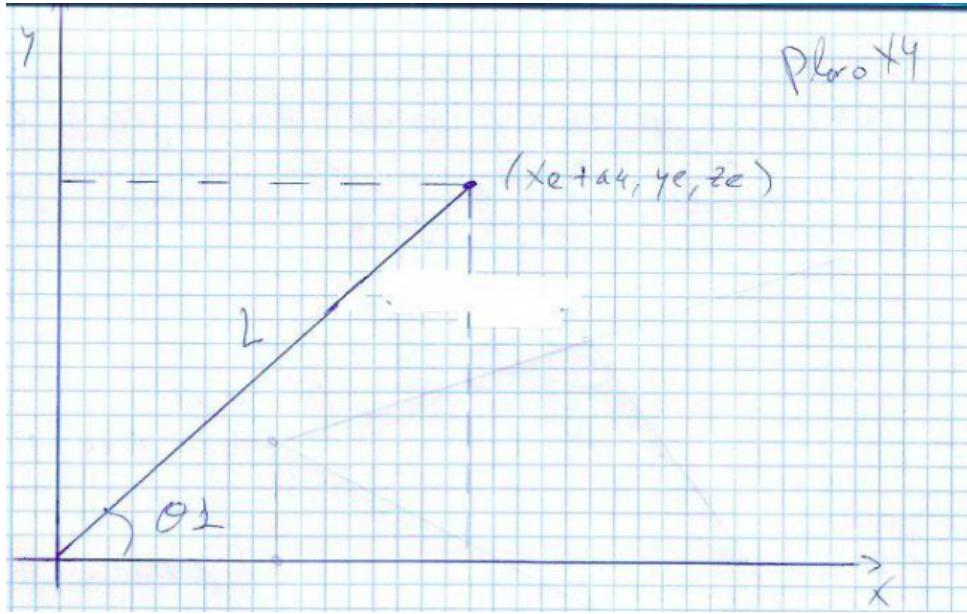
$$\implies \theta_3 = \arctan\left(\frac{\sin(\theta_3)}{\cos(\theta_3)}\right) \quad (13)$$

$$\implies \theta_2 = \phi_{23} + \theta_3 \quad (14)$$

$$(15)$$

Finalmente, en el plano XY tenemos la siguiente configuración:

³Teorema del coseno: $c^2 = a^2 + b^2 - 2ab \cos \beta$

Figura 10: visión del μ Arm en el plano XY

De este modo, aplicando el teorema de Pitágoras⁴, se obtiene:

$$\begin{aligned} L &= \pm \sqrt{(X_e + T_X)^2 + Y_e^2} \\ \cos(\theta_1) &= \frac{X_e + T_X}{\sqrt{(X_e + T_X)^2 + Y_e^2 + d_1}} \\ \sin(\theta_1) &= \frac{Y_e}{\sqrt{(X_e + T_X)^2 + Y_e^2 + d_1}} \\ \Rightarrow \tan(\theta_1) &= \frac{\sin(\theta_1)}{\cos(\theta_1)} \end{aligned}$$

$$\theta_1 = \arctan \left(\frac{\frac{Y_e}{\sqrt{(X_e + T_X)^2 + Y_e^2 + d_1}}}{\frac{X_e + T_X}{\sqrt{(X_e + T_X)^2 + Y_e^2 + d_1}}} \right) \quad (16)$$

$$= \arctan \left(\frac{Y_e}{X_e + T_X + d_1} \right) \quad (17)$$

(18)

De esta forma, un algoritmo de obtención de la inversa sería:

- Obtener θ_1 a partir del punto $(X_e + T_X, Y_e, Z_e - T_Z)$, usando la ecuación 17.
- Obtener (X_e, Y_e, Z_e) quitando las traslaciones T_X y T_Z : $X_e = X_e - T_X$; $Z_e = Z_e + T_Z$.
- Obtener $\cos(\theta_3)$ con la ecuación 11 y el $\sin(\theta_3)$ con la ecuación 12.

⁴Teorema de Pitágoras: $h^2 = c_1^2 + c_2^2$

- Calcular θ_3 usando la ecuación 13.
- Calcular θ_2 usando la ecuación 14.

4. Matriz Jacobiana

La matriz Jacobiana nos permite obtener el movimiento \vec{x} , correspondiente a las velocidades en el extremo del robot dadas unas velocidades de las articulaciones \vec{q} [5].

La matriz Jacobiana es una matriz de derivadas, esto es, varía según el tiempo. De esta forma, se puede definir la matriz Jacobiana de un manipulador como:

$$\vec{x} = J \cdot \vec{q} \begin{cases} \vec{q} = (\theta_1, \dots, \theta_n, d_1, \dots, d_n) \\ \vec{x} = (X_e, Y_e, Z_e, \phi_e) \\ J \equiv \text{matriz Jacobiana} \end{cases} \quad (19)$$

$$J_1(\vec{q}) = \begin{pmatrix} \frac{\partial X_e}{\partial q_0} & \frac{\partial X_e}{\partial q_1} & \dots & \frac{\partial X_e}{\partial q_n} & \frac{\partial X_e}{\partial d_1} & \dots & \frac{\partial X_e}{\partial d_n} \\ \frac{\partial Y_e}{\partial q_0} & \frac{\partial Y_e}{\partial q_1} & \dots & \frac{\partial Y_e}{\partial q_n} & \frac{\partial Y_e}{\partial d_1} & \dots & \frac{\partial Y_e}{\partial d_n} \\ \frac{\partial Z_e}{\partial q_0} & \frac{\partial Z_e}{\partial q_1} & \dots & \frac{\partial Z_e}{\partial q_n} & \frac{\partial Z_e}{\partial d_1} & \dots & \frac{\partial Z_e}{\partial d_n} \end{pmatrix} \quad (20)$$

$$J_2(\vec{q}) = \begin{pmatrix} \frac{\partial \phi_X}{\partial q_0} & \frac{\partial \phi_X}{\partial q_1} & \dots & \frac{\partial \phi_X}{\partial q_n} & \frac{\partial \phi_X}{\partial d_1} & \dots & \frac{\partial \phi_X}{\partial d_n} \\ \frac{\partial \phi_Y}{\partial q_0} & \frac{\partial \phi_Y}{\partial q_1} & \dots & \frac{\partial \phi_Y}{\partial q_n} & \frac{\partial \phi_Y}{\partial d_1} & \dots & \frac{\partial \phi_Y}{\partial d_n} \\ \frac{\partial \phi_Z}{\partial q_0} & \frac{\partial \phi_Z}{\partial q_1} & \dots & \frac{\partial \phi_Z}{\partial q_n} & \frac{\partial \phi_Z}{\partial d_1} & \dots & \frac{\partial \phi_Z}{\partial d_n} \end{pmatrix} \quad (21)$$

$$J(\vec{q}) = \begin{pmatrix} J_1(\vec{q}) \\ J_2(\vec{q}) \end{pmatrix} \quad (22)$$

En nuestro caso particular, obtenemos la siguiente matriz Jacobiana:

$$J_1(\vec{q}) = \begin{pmatrix} -(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) & (-a_2 \sin(\theta_2) - a_3 \sin(\theta_2 - \theta_3)) \cos(\theta_1) & a_3 \sin(\theta_2 - \theta_3) \cos(\theta_1) \\ (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \cos(\theta_1) & (-a_2 \sin(\theta_2) - a_3 \sin(\theta_2 - \theta_3)) \sin(\theta_1) & a_3 \sin(\theta_1) \sin(\theta_2 - \theta_3) \\ 0 & a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) & -a_3 \cos(\theta_2 - \theta_3) \end{pmatrix} \quad (23)$$

$$J_2(\vec{q}) = \begin{pmatrix} 0 & 1 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad (24)$$

$$J(\vec{q}) = \begin{pmatrix} -(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_1) & (-a_2 \sin(\theta_2) - a_3 \sin(\theta_2 - \theta_3)) \cos(\theta_1) & a_3 \sin(\theta_2 - \theta_3) \cos(\theta_1) \\ (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \cos(\theta_1) & (-a_2 \sin(\theta_2) - a_3 \sin(\theta_2 - \theta_3)) \sin(\theta_1) & a_3 \sin(\theta_1) \sin(\theta_2 - \theta_3) \\ 0 & a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) & -a_3 \cos(\theta_2 - \theta_3) \\ 0 & 1 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad (25)$$

Entre otras utilidades de la matriz Jacobiana, podemos obtener diferentes ecuaciones que van relacionando el movimiento del brazo con el trabajo que hace, la fuerza que es necesaria y la potencia:

$$W = \int F^T \cdot v \, dt \quad (26)$$

$$P = \frac{W}{t} = \dots = F^T \cdot v \quad (27)$$

Debido a la equivalencia energética, el trabajo se realiza a la misma velocidad, independientemente del sistema [5], por lo que se puede escribir la potencia en términos del espacio del *end-effector*:

$$P = F_x^T \cdot \vec{x} \quad (28)$$

$$P = F_q^T \cdot \vec{q} \quad (29)$$

donde F_x es la fuerza aplicada al brazo robótico y \vec{x} la velocidad del mismo; F_q es el torque de las articulaciones y \vec{q} es la velocidad angular de las mismas.

De esta información se deducen las matrices que conforman la Jacobiana. La ecuación 20 describe la velocidad lineal del *end-effector*, y puede ser escrita como $J_w(\vec{q})$; mientras, la ecuación 21 describe la velocidad angular del *end-effector*, que puede ser escrita como $J_\omega(\vec{q})$.

Con la matriz Jacobiana ya obtenida se pueden obtener los puntos críticos. Dichos puntos permiten conocer posiciones inalcanzables por el brazo robótico, ya que el determinante en dicho punto es 0 y, por tanto, no existe la inversa.

Para este brazo robótico, el determinante es:

$$|J(\vec{q})| = -a_2 a_3 (a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1) \sin(\theta_3) \quad (30)$$

Analíticamente, los puntos críticos del μArm son: $\theta_3 = 0$ y $\theta_3 = \pi$. Pero, al observar las tablas con los ángulos de giro de los motores (ver tabla 1), comprobamos que el brazo robótico no es capaz de llegar a los 180° y, por la configuración geométrica del mismo, tampoco puede tener 0° ⁵, lo cual nos conduce a que no hay puntos críticos en este brazo robótico:

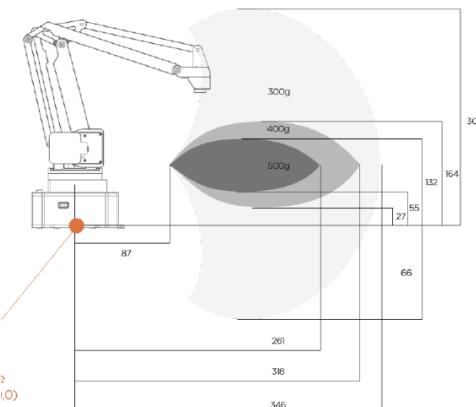


Figura 11: área de trabajo del μArm [1]

⁵el motor tiene esos grados de libertad, pero la articulación no, y está limitada por la estructura

Igualmente, cuando se dan estos casos, se ha de hacer uso de o bien la matriz pseudo-inversa (J^+) de la matriz Jacobiana (si J^{-1} no existe) o bien de la matriz transpuesta (J^T):

$$J^+ = J^T \cdot (J \cdot J^T)^{-1} \quad (31)$$

Una característica de la pseudo-inversa es que, en principio, siempre existe. Además, si la inversa de la Jacobiana J^{-1} existe, entonces la pseudo-inversa será igual a la matriz inversa:

$$J^+ = J^{-1} \Leftrightarrow \exists J^{-1} \quad (32)$$

Esto se explica con mayor detalle en el punto 4.1.

4.1. Matriz Jacobiana inversa

La matriz Jacobiana inversa permite saber qué velocidad hay que aplicar en las articulaciones \vec{q} para obtener una velocidad en el *end-effector* \vec{x} .

En el caso del μArm , la Jacobiana inversa resulta:

$$J^{-1} = \begin{pmatrix} -\frac{\sin(\theta_1)}{a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1} & \frac{\cos(\theta_1)}{a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1} & 0 \\ -\frac{\cos(\theta_1) \cos(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} & -\frac{\sin(\theta_1) \cos(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} & -\frac{\sin(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} \\ -\frac{(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3)) \cos(\theta_1)}{a_2 a_3 \sin(\theta_3)} & -\frac{(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3)) \sin(\theta_1)}{a_2 a_3 \sin(\theta_3)} & -\frac{a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3)}{a_2 a_3 \sin(\theta_3)} \end{pmatrix} \quad (33)$$

Como se observa en la ecuación 30, el determinante del brazo robótico se hace cero cuando el ángulo $\theta_3 = 0$ o bien cuando el mismo ángulo es π . Pero, debido a la configuración geométrica del robot (ver figura 11), dichas configuraciones son inalcanzables.

Igualmente, para tratar configuraciones singulares, se utiliza la pseudo-inversa de *Moore-Penrose*. Dicha matriz existe y es única por cada matriz A de entrada, y se define según las ecuaciones 31 y 32.

Para el caso particular del μArm , la pseudo-inversa es:

$$J^+ = \begin{pmatrix} -\frac{\sin(\theta_1)}{a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1} & \frac{\cos(\theta_1)}{a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3) + d_1} & 0 \\ -\frac{\cos(\theta_1) \cos(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} & -\frac{\sin(\theta_1) \cos(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} & -\frac{\sin(\theta_2 - \theta_3)}{a_2 \sin(\theta_3)} \\ -\frac{(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3)) \cos(\theta_1)}{a_2 a_3 \sin(\theta_3)} & -\frac{(a_2 \cos(\theta_2) + a_3 \cos(\theta_2 - \theta_3)) \sin(\theta_1)}{a_2 a_3 \sin(\theta_3)} & -\frac{a_2 \sin(\theta_2) + a_3 \sin(\theta_2 - \theta_3)}{a_2 a_3 \sin(\theta_3)} \end{pmatrix} \quad (34)$$

Como se puede apreciar, la pseudo-inversa es exactamente igual que la inversa. Esto es debido a que, como se muestra en la ecuación 32, la pseudo-inversa es igual a la inversa siempre y cuando la inversa exista:

$$J^+ = J^{-1} \Leftrightarrow \exists J^{-1}$$

Esto se cumple ya que es una de las propiedades de la inversa de *Moore-Penrose*:

- "if A is invertible, its pseudo-inverse is its inverse. That is, $A^+ = A^{-1}$ " [6].

5. Planificación y descripción de una trayectoria

Una trayectoria es el recorrido que debe describir un manipulador para cambiar su posición de un punto a otro. La trayectoria que describe un manipulador está formada por un conjunto puntos cartesianos en el espacio, es decir, puntos con tres coordenadas espaciales.

Para este caso, la trayectoria que va a describir el manipulador se define como:

$$T = \{P_1, P_2, \dots, P_n\} \setminus P_k \in \mathbb{R}^3, \forall k \in \{0, \dots, n\} \quad (35)$$

Un caso de interés a la hora de definir una trayectoria es en el que los puntos del recorrido están definidos por una función matemática $f(x)$ de dos dimensiones. Dado que esta trayectoria se define en un plano, una de las coordenadas se fija a cero. En el caso de usar $y = f_y(x)$, el valor de la coordenada z se fijaría a cero y por lo tanto la trayectoria estaría contenida en el plano XY . De igual forma, en el caso de usar $z = f_z(x)$, el valor de la coordenada y se fijaría a cero y por lo tanto la trayectoria estará contenida en el plano XZ .

En este caso y para simplificar la situación, la trayectoria se va describir en el plano XY , siendo también posible aplicar este razonamiento para los planos XZ o YZ . Para obtener los puntos de la trayectoria, se debe muestrear la función $y = f_y(x)$ para ciertos valores de x y, de esta forma, se obtendrían los puntos $P_i = (x, f_y(x), 0)$.

Dado que el μ Arm tiene una precisión mínima de movimiento de 0,2 mm. (en su *end-effector*), la función $f_y(x)$ se va a muestrear para valores de x siguiendo la expresión $x_i = x_{i-1} + 0,2 \cdot k$ con $k \in \mathbb{N}$. De esta manera, para cada valor de x se obtiene un valor de y y dado que z es siempre nulo, se obtienen todos los puntos de la trayectoria.

Una vez definida, el manipulador debe de moverse de un punto a otro siguiendo todos los puntos de la misma, para así poder describir la función matemática. Para que esto ocurra, se debe determinar las coordenadas articulares de los ejes de giro para todos y cada uno de los puntos y, de esta forma, se podrán controlar los motores adecuadamente.

Para este manipulador, se han obtenido las expresiones explícitas de la cinemática inversa (*IK*)⁶ y, por lo tanto, para cada punto del espacio, se conoce exactamente las coordenadas articulares del mismo. En el caso de no conocer las expresiones de la cinemática inversa, se podrían utilizar otros métodos de planificación que aproximan la cinemática inversa, por ejemplo mediante el uso de la matriz Jacobiana.

En el gráfico siguiente se representa cómo, dados dos puntos en el espacio, se puede calcular la variación de las coordenadas articulares necesaria para que el manipulador se desplace de un punto al otro:

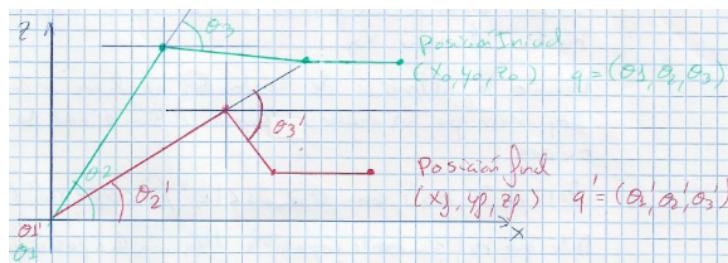


Figura 12: descripción de la trayectoria del brazo robótico

⁶matrices de la cinemática inversa representadas en el punto 3, en particular las ecuaciones 17, 14 y 13

Dado el punto inicial (P_o) y punto final (P_f) de un desplazamiento, si se quiere calcular la variación que hay que aplicar sobre las articulaciones para realizar dicho desplazamiento, se realizan los siguientes cálculos:

- Se aplican las expresiones de cinemática inversa sobre el punto inicial $P_o = (x_o, y_o, z_o)$ y punto final $P_f = (x_f, y_f, z_f)$ del desplazamiento, de forma que se obtienen las coordenadas articulares correspondientes a cada punto $q_o = (\theta_{1o}, \theta_{2o}, \theta_{3o})$ y $q_f = (\theta_{1f}, \theta_{2f}, \theta_{3f})$.
- Para calcular la variación en cada uno de los ángulos, se resta el vector de coordenadas articulares del punto final con las del punto inicial:

$$\Delta q = (\theta_{1f} - \theta_{1o}, \theta_{2f} - \theta_{2o}, \theta_{3f} - \theta_{3o}) \quad (36)$$

- Se aplica dicha variación de coordenadas articulares Δq sobre los motores del manipulador para realizar el desplazamiento.

Una vez se ha descrito el proceso anterior, basta con aplicarlo de forma secuencial sobre todos los puntos de la trayectoria calculados a partir de la función $f_y(x)$, desde el punto inicial x_1 de la trayectoria hasta el punto final x_n ; se consigue pues que las coordenadas articulares vayan variando progresivamente en cada uno de los puntos de la trayectoria y por lo tanto el manipulador se desplaza.

Durante este proceso, se debe comprobar el valor de los ángulos de las coordenadas articulares para que estos no excedan los límites reales de giro de cada una de las articulaciones. Además, dado que este manipulador tiene posiciones singulares, pero no puede alcanzarlas debido al diseño físico, no existe problema al describir la trayectoria.

Referencias

- [1] *uArm Swift Pro_Devloper Guide v1.0.6.pdf*, en, 2019. dirección: http://download.ufactory.cc/docs/en/uArm%20Swift%20Pro_Devloper%20Guide%20v1.0.6.pdf (visitado 02-11-2019).
- [2] *uArm-Swift-Specifications-171012.pdf*, en, 2019. dirección: <http://download.ufactory.cc/docs/en/uArm-Swift-Specifications-171012.pdf> (visitado 02-11-2019).
- [3] *uArm pro User Manual v1.1.0.pdf*, en, 2019. dirección: <http://download.ufactory.cc/docs/en/uArm%20pro%20User%20Manual%20v1.1.0.pdf> (visitado 02-11-2019).
- [4] *UPM-Robotics/uarm*, original-date: 2019-11-01T11:13:54Z, nov. de 2019. dirección: <https://github.com/UPM-Robotics/uarm> (visitado 02-11-2019).
- [5] travisdewolf, *Robot control part 2: Jacobians, velocity, and force*, en, sep. de 2013. dirección: <https://studywolf.wordpress.com/2013/09/02/robot-control-jacobians-velocity-and-force/> (visitado 17-11-2019).
- [6] *Moore-Penrose inverse*, en, Page Version ID: 926414640, nov. de 2019. dirección: https://en.wikipedia.org/w/index.php?title=Moore%20Penrose_inverse&oldid=926414640 (visitado 17-11-2019).

A. Código Python