

Centro de Investigación en Matemáticas A.C.
Maestría en Ciencias de la Computación y Matemáticas
Industriales

Métodos Numéricos (2019)

Proyecto final. Solución Numérica al Problema de la Cinemática Inversa

Marco Antonio Esquivel Basaldua

El problema de la cinemática inversa es uno de los temas con mayor relevancia en el campo de la robótica. Este permite obtener el valor de las articulaciones necesarias, dado un punto en el espacio de trabajo del robot manipulador. El uso de métodos numéricos iterativos permiten la solución del problema de una forma más generalizada. En este reporte se explica el uso del iterativo Newton-Raphson para la búsqueda de raíces en funciones no lineales.

Introducción

Consideramos un brazo robótico clásico como una serie de barras rígidas (eslabones) unidas entre sí a través de articulaciones con un grado de libertad, ya sea de tipo rotacional (revolución) o prismática (traslación), que permiten un movimiento relativo entre cada dos eslabones consecutivos. Un brazo robótico se puede considerar entonces como una cadena cinemática abierta en la que el extremo libre es conocido como efector final.

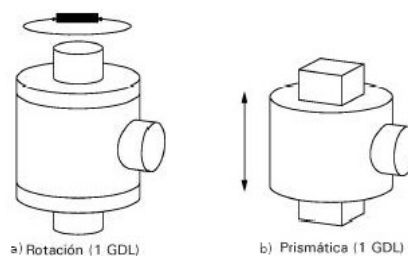


Figura 1. a) Articulación rotacional. b) Articulación prismática.

De acuerdo a la disposición de los eslabones y al tipo de articulación entre dos eslabones consecutivos las configuraciones más usuales para los brazos robóticos son: Robot cartesiano, esférico o polar, robot articulado, cilíndrico, y robot SCARA.

Descripción de posición y orientación

Una de las principales preocupaciones en robótica es la ubicación de los eslabones, las piezas y herramientas con las que trabaja el robot. Estos objetos se describen mediante dos atributos: posición y orientación. Para describir estos atributos de un cuerpo en el espacio se adjunta rígidamente un sistema de coordenadas, o trama, al objeto. En seguida se describe la posición y orientación de esta trama con respecto a algún sistema de coordenadas de referencia (figura 2.), o también llamada trama base ya que el resto de las tramas están descritas con respecto a ella. La trama base a menudo es colocada en la base del brazo robótico.

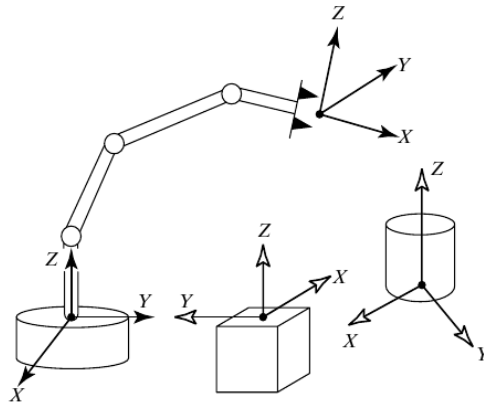


Figura 2. El sistema de coordenadas o "trama" se adjunta a los manipuladores y objetos en el ambiente.

Una vez establecido un sistema de coordenadas se puede ubicar cualquier punto en el espacio con un **vector de posición, P** , de orden 3×1 . Para describir la orientación, se adjunta un sistema de coordenadas al cuerpo y en seguida se da una descripción a este sistema relativo al sistema de referencia con los vectores unitarios de los tres ejes principales, $\hat{X} \hat{Y} \hat{Z}$, en términos de la trama base. Es conveniente apilar estos tres vectores como columnas de una matriz de 3×3 a la que se llamará **matriz de rotación, R** .

Una forma de sintetizar el vector de posición y la matriz de rotación es mediante una matriz de **transformación homogénea, T** , la cual es una matriz de orden 4×4 que consta de la matriz **R** y el vector **P** puestos de la siguiente manera:

$$T = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix}$$

Nótese que se agregan tres ceros y un uno en la última fila de la matriz **T** .

Cinemática directa

La cinemática es la ciencia que trata el movimiento sin considerar las fuerzas que lo ocasionan. Dentro de la cinemática se estudian la posición, velocidad, aceleración y todas las derivadas de mayor orden de las variables de posición.

Un problema básico en el estudio de la robótica es conocido como **cinemática directa**, que es el problema geométrico estático de calcular la posición y orientación del efector final del manipulador. Dado un conjunto de **ángulos articulares** (desplazamientos de las articulaciones relativos entre dos eslabones consecutivos), el problema de la cinemática directa es calcular la orientación y posición del efector final relativo a la trama base (figura 3).

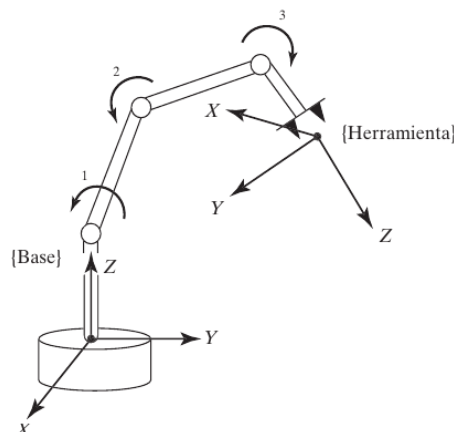


Figura 3. La posición y orientación de la herramienta, o efector final, como función de los valores de los ángulos articulares.

Convención Denavit-Hartenberg

La convención Denavit-Hartenberg es una herramienta muy utilizada en el problema de la cinemática directa. Una vez establecidas las tramas en cada una de las articulaciones, esta convención hace uso de cuatro parámetros para describir la matriz de transformación entre la trama actual y la trama anterior, siendo la trama base la trama cero. Los cuatro parámetros antes mencionados son:

- a_i : es la distancia entre \hat{Z}_i y \hat{Z}_{i-1} a lo largo de \hat{X}_i .
- α_i : es el ángulo entre \hat{Z}_i y \hat{Z}_{i-1} con respecto a \hat{X}_i
- d_i : es la distancia entre \hat{X}_i y \hat{X}_{i-1} a lo largo de \hat{Z}_{i-1}
- θ_i : es el ángulo entre \hat{X}_i y \hat{X}_{i-1} con respecto a \hat{Z}_{i-1}

donde $i = 1, 2, \dots, n$.

Para describir la matriz de transformación desde la trama $i - 1$ a la trama i se utiliza:

$$T_{i-1,i} = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

donde:

$$C\theta_i = \cos(\theta_i)$$

$$S\theta_i = \sin(\theta_i)$$

$$C\alpha_i = \cos(\alpha_i)$$

$$S\alpha_i = \sin(\alpha_i)$$

Para expresar la matriz de transformación desde la trama cero a la trama n , se multiplican las matrices generadas entre cada par de tramas desde la trama base hasta la trama n .

$$T_{0,n} = T_{0,1} T_{1,2} T_{2,3} \dots T_{n-1,n}$$

Cinemática Inversa

El problema de la cinemática inversa, dada la posición y orientación del efector final del manipulador, consiste en calcular todos los conjuntos de posibles ángulos articulares que podrían utilizarse para obtener esta posición y orientación dadas. Se puede pensar en este problema como en una asignación de ubicaciones en el espacio cartesiano 3D, a ubicaciones en el espacio de articulaciones del robot.

Este problema de cinemática inversa no es tan simple como el de la cinemática directa. Debido a que las ecuaciones cinemáticas no son lineales, su solución no siempre es sencilla, o posible, en una forma cerrada, además de que puede existir más de un conjunto de ángulos articulados para llegar a la posición y orientación solicitadas (figura 4).

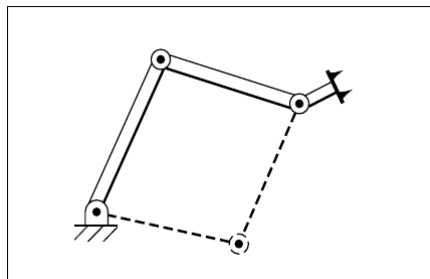


Figura 4. Más de un conjunto de ángulos articulares pueden dar solución al problema de la cinemática inversa.

La existencia o inexistencia de una solución define el espacio de trabajo de un manipulador. La falta de una solución significa que el manipulador no puede obtener la posición y orientación deseadas, ya que se encuentran fuera del espacio de trabajo del manipulador.

Se puede ver al problema de la cinemática directa como determinar la posición y orientación del efector final dados los ángulos, y al problema de la cinemática inversa como determinar los ángulos dada la posición y orientación del efector final.

Definición del problema

En este proyecto se da solución al problema de la cinemática inversa a un brazo robótico articulado de tres grados de libertad (figura 5). El problema se centra en dar solución a los ángulos necesarios dadas las posiciones de los puntos por los cuales debe pasar el efector final (el final de la cadena cinemática).

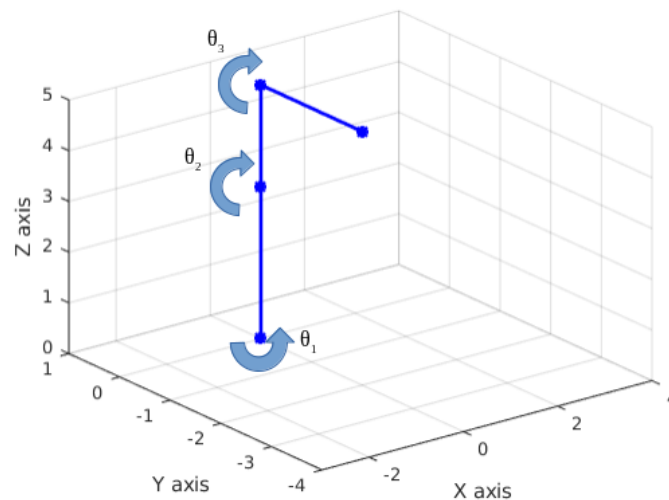


Figura 5. Representación del robot articulado de tres grados de libertad.

Las longitudes de cada uno de los eslabones pueden ser determinadas por el usuario especificándolas en el archivo de texto "*lengths.txt*" (cada valor debe estar separado por un salto de línea). Los puntos, en espacio tridimensional se deben especificar en el archivo "*Path.txt*". Como primer valor en este archivo se debe especificar el número de puntos que debe recorrer el efector final seguido de las coordenadas cartesianas de cada uno de ellos en líneas separadas, las coordenadas deben estar ordenadas en la coordenadas x y z (separadas por un espacio).

Utilizando la convención Denavit-Hartenberg para la configuración del robot dado se tienen los parámetros

- $a_1 = 0, a_2 = l_2, a_3 = l_3$
- $\alpha_1 = \pi/2, \alpha_2 = 0, \alpha_3 = 0$
- $d_1 = l_1, d_2 = 0, d_3 = 0$
- Los valores de los ángulos θ_i serán determinados por el programa.

Donde l_1, l_2, l_3 son las longitudes de los eslabones especificadas por el usuario.

Aplicando la definición de la convención Denavit-Hartenberg y con la ayuda del software MATLAB para la multiplicación de las tres matrices de transformación generadas se obtienen las ecuaciones que determinan la posición, \mathbf{P} , del efector final como una función de los ángulos $\theta_1, \theta_2, \theta_3$.

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$P = \begin{bmatrix} l_3 C\theta_3 (C\theta_1 C\theta_2 - C\alpha_1 S\theta_1 S\theta_2) - l_3 S\theta_3 (C\theta_1 S\theta_2 + C\alpha_1 C\theta_2 S\theta_1) + l_2 C\theta_1 C\theta_2 - l_2 C\alpha_1 S\theta_1 S\theta_2 \\ l_3 C\theta_3 (C\theta_2 S\theta_1 + C\alpha_1 C\theta_1 S\theta_2) - l_3 S\theta_3 (S\theta_1 S\theta_2 - C\alpha_1 C\theta_1 C\theta_2) + l_2 C\theta_2 S\theta_1 + l_2 C\alpha_1 C\theta_1 S\theta_2 \\ l_1 + l_2 S\alpha_1 S\theta_2 + l_3 S\alpha_1 C\theta_2 S\theta_3 + l_3 S\alpha_1 C\theta_3 S\theta_2 \end{bmatrix}$$

Aplicando:

$$C\alpha_1 = \cos(\alpha_1) = \cos(\pi/2) = 0$$

$$S\alpha_1 = \sin(\alpha_1) = \sin(\pi/2) = 1$$

Se simplifican las expresiones de **P**:

$$P = \begin{bmatrix} l_3 C\theta_1 C\theta_2 C\theta_3 - l_3 C\theta_1 S\theta_2 S\theta_3 + l_2 C\theta_1 C\theta_2 \\ l_3 S\theta_1 C\theta_2 C\theta_3 - l_3 S\theta_1 S\theta_2 S\theta_3 + l_2 S\theta_1 C\theta_2 \\ l_1 + l_2 S\theta_2 + l_3 C\theta_2 S\theta_3 + l_3 S\theta_2 C\theta_3 \end{bmatrix}$$

El punto inicial desde el cual el brazo robótico comienza su recorrido es el punto dado por los ángulos $\theta_1 = \theta_2 = \theta_3 = \pi/2$, que dan un valor de **P** de:

$$P_0 = \begin{bmatrix} 0 \\ -l_3 \\ l_1 + l_2 \end{bmatrix}$$

El problema consiste en determinar un set de ángulos $\theta_1, \theta_2, \theta_3$ que hagan posible el paso por diferentes valores de posición **P** de manera optima es decir que logren hacer el recorrido de forma "suave". En la practica, un recorrido suave de un punto a otro en el sistema cartesiano asegura que los motores no sean forzados a velocidades que puedan afectar la estructura mecánica y electrónica del manipulador y eventualmente ocasionar accidentes.

Metodología

Expresando la posición del efector final **P**, al aplicar el concepto de la cinemática directa, como una función de los ángulos $\theta = [\theta_1, \theta_2, \theta_3]$, $P = f(\theta)$, y siendo x_d el vector con las coordenadas deseadas del efector final, se define $g(\theta) = x_d - f(\theta)$. Utilizando el método iterativo de **Newton-Raphson** el objetivo es encontrar el set de ángulos articulares θ_d tal que

$$g(\theta_d) = x_d - f(\theta_d) = 0$$

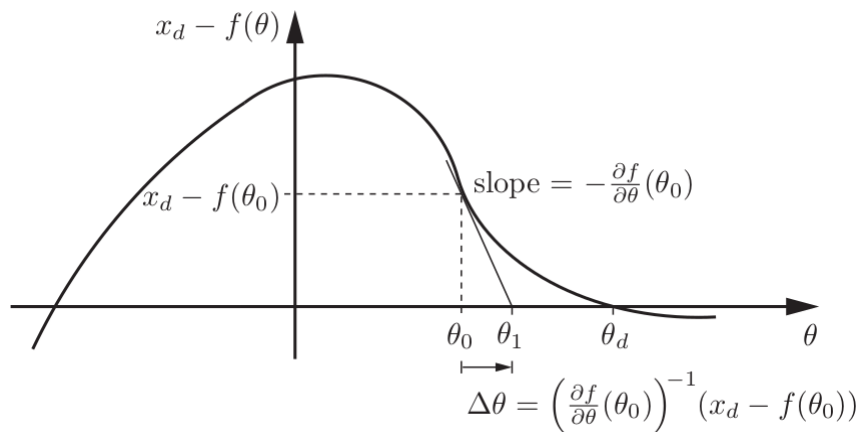


Figura 6. Primer paso en el método de Newton-Raphson para la función $g(\theta)$.

Dado un set de ángulos iniciales θ_0 que sea aproximado a la solución θ_d , la cinemática directa se puede expresar mediante la expansión de Taylor como:

$$x_d = f(\theta_d) = f(\theta_0) + \frac{\partial f}{\partial \theta} \bigg|_{\theta_0} (\theta_d - \theta_0) + h.o.t.$$

donde $\frac{\partial f}{\partial \theta} \bigg|_{\theta_0} = J(\theta_0)$ es la matriz jacobiana de las coordenadas de la posición del final de la cadena cinemática evaluada en θ_0 , y $(\theta_1 - \theta_0) = \Delta\theta$. Truncando la expansión de Taylor en su primer orden se tiene:

$$J(\theta_0)\Delta\theta = x_d - f(\theta_0)$$

Asumiendo que $J(\theta_0)$ es cuadrada, como lo es en nuestro caso, e invertible, se puede resolver $\Delta\theta$ como:

$$\begin{aligned}\Delta\theta &= J^{-1}(\theta_0)(x_d - f(\theta_0)) \\ (\theta_1 - \theta_0) &= J^{-1}(\theta_0)(x_d - f(\theta_0)) \\ \theta_1 &= \theta_0 + J^{-1}(\theta_0)(x_d - f(\theta_0)) \\ \theta_{i+1} &= \theta_i + J^{-1}(\theta_i)(x_d - f(\theta_i))\end{aligned}$$

Si la cinemática inversa es lineal en θ , entonces el valor estimado $\theta_1 = \theta_0 + \Delta\theta$ satisface la ecuación $x_d = f(\theta_1)$ de forma exacta. Si la cinemática directa no es lineal en θ , como es el caso en general, la estimación θ_1 da un valor más aproximado a la solución buscada que θ_0 . Repitiendo el proceso se produce la secuencia $\{\theta_0, \theta_1, \theta_2, \dots\}$ hasta converger a θ_d (figura 6).

Como se muestra en la figura 6, si existen múltiples soluciones al problema de la cinemática inversa, el proceso iterativo converge a la solución más cercana a la estimación inicial θ_0 . Si la estimación inicial no está próxima a la solución el proceso iterativo tal vez no converja. Para solucionar en parte este problema, dados dos puntos consecutivos por los que tiene que pasar el efector final, el recorrido entre ambos es dividido en diez subpuntos por los que se habrá de pasar, sin embargo si la distancia entre dos puntos consecutivos es muy larga se tiene el riesgo que no se converja a una solución.

El algoritmo iterativo de Newton-Raphson para encontrar θ_d es:

1. Dado x_d y la estimación inicial θ_0 , inicializar $i = 0$
2. Definir $e = x_d - f(\theta_i)$. Mientras $\|e\| > \epsilon$, para un valor pequeño de ϵ
 - Definir $\theta_{i+1} = \theta_i + J^{-1}(\theta_i)e$
 - Incrementar i

Para el cálculo de J^{-1} se utiliza la descomposición LU de J usando el método de **Doolittle** y después se da solución a los sistemas de ecuaciones para matrices triangulares.

Resultados

El código desarrollado en el lenguaje de programación C++ se encarga de leer las longitudes (l_1, l_2, l_3) de los tres eslabones existentes en la configuración del brazo robótico a partir del archivo "lengths.txt", y de leer los puntos que el efector final debe recorrer a partir del archivo "Path.txt". Dado como punto inicial el punto:

$$P_0 = \begin{bmatrix} 0 \\ -l_3 \\ l_1 + l_2 \end{bmatrix}$$

se generan los valores de los ángulos articulares necesarios para pasar por cada uno de los puntos solicitados.

Para un ejemplo en el que:

$$l_1 = 3, l_2 = 2, l_3 = 2$$

solicitando que se pase por los puntos:

$$P_1 = \begin{bmatrix} 2 \\ -3 \\ 3 \end{bmatrix}, P_2 = \begin{bmatrix} 0 \\ -2 \\ 2 \end{bmatrix}, P_3 = \begin{bmatrix} 0 \\ -2 \\ 5 \end{bmatrix},$$

Con los valores de las longitudes dadas se tiene un valor P_0 :

$$P_0 = \begin{bmatrix} 0 \\ -2 \\ 5 \end{bmatrix},$$

Nótese que ya que $P_0 = P_3$, lo que se busca es que el efector final regrese a su posición de inicio.

Al seleccionar un error de $\epsilon = 0.1$ se obtiene el siguiente set de 31 resultados para los ángulos necesarios en cada una de las articulaciones.

angles.txt			
1	1.5708	1.5708	1.5708
2	1.6708	1.6208	1.6208
3	1.75614	1.69991	1.63712
4	1.83087	1.79347	1.63065
5	1.89688	1.89866	1.60178
6	1.95508	2.01276	1.55047
7	2.00648	2.13363	1.47668
8	2.05198	2.26014	1.37988
9	2.09239	2.39243	1.25838
10	2.12842	2.53232	1.10829
11	2.16067	2.6843	0.921095
12	2.12793	2.61251	1.1143
13	2.09184	2.56526	1.27329
14	2.05141	2.53194	1.41224
15	2.00586	2.51134	1.53404
16	1.9544	2.50263	1.64088
17	1.89611	2.5055	1.7337
18	1.83002	2.51971	1.81277
19	1.75513	2.54484	1.87781
20	1.67053	2.58009	1.92827
21	1.57562	2.62401	1.96342
22	1.57077	2.46655	2.03598
23	1.5708	2.30154	2.08431
24	1.5708	2.14008	2.10636
25	1.5708	1.98921	2.10189
26	1.5708	1.85616	2.07116
27	1.5708	1.74639	2.01559
28	1.5708	1.6624	1.93728
29	1.5708	1.60421	1.83815
30	1.5708	1.57049	1.71934
31	1.5708	1.55971	1.58079

Figura 7. Archivo de ángulos generado por el programa.

Estos datos son guardados en el archivo de texto *angles.txt* en el que cada fila representa el valor de los ángulos θ que se deben aplicar a cada una de las articulaciones de manera consecutiva para pasar por los puntos solicitados.

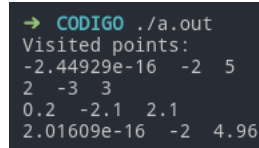
Los puntos por los cuales realmente pasa el efector final son:

```
➔ CODIGO ./a.out
Visited points:
-2.44929e-16 -2 5
1.99318 -2.9777 2.98835
0.208905 -2.08772 2.10641
-2.07174e-16 -1.97703 4.40638
```

Figura 8. Cada fila representa las coordenadas de los puntos realmente visitados por el efector final con un valor $\epsilon = 0.1$.

Se observa que el error más grande es en la coordenada z de P_3 (ultimo valor de la ultima fila) en la que se tiene un error de: $error = 5 - 4.40638 = 0.5936$.

Al aplicar un valor de $\epsilon = 1 \times 10^{-12}$, se obtiene un set de 122 tríos de ángulos y un registro de pasar por los puntos mostrados en la figura 9.

A terminal window with a dark background and light green text. It shows the command 'CODIGO ./a.out' and the output 'Visited points:' followed by four rows of coordinates. The first row is '-2.44929e-16 -2 5', the second is '2 -3 3', the third is '0.2 -2.1 2.1', and the fourth is '2.01609e-16 -2 4.96'.

```
→ CODIGO ./a.out
Visited points:
-2.44929e-16 -2 5
2 -3 3
0.2 -2.1 2.1
2.01609e-16 -2 4.96
```

Figura 9. Cada fila representa las coordenadas de los puntos realmente visitados por el efector final con un valor $\epsilon = 1e-12$.

Como se puede observar el valor del error desciende a 0.1 en el peor de los casos.

Para una mejor visualización de los resultados se generó en el software MATLAB el programa de cinemática directa, *Plot_IK.m*, que, a partir de los ángulos generados por el programa en C++, simula las posiciones obtenidas al aplicar cada uno de los valores de los ángulos en el archivo "*angles.txt*". Estos resultados se pueden observar en el vídeo dentro de la carpeta *PRESENTACION*.

Conclusiones

En la actualidad prácticamente todos los manipuladores existentes en el mercado y en la industria cuentan con un algoritmo para la solución del problema de cinemática inversa. Al aplicar un método iterativo los que se busca es solucionar el problema de una forma más generalizada que por ejemplo al utilizar un método geométrico en el que las dimensiones y configuración de cada manipulador requiere un set ecuaciones distintas para resolver el problema.

La generación de ángulos con este método asegura una transición suave entre los puntos que recorrerá el efector final del brazo robótico sin que se generen fuerzas o velocidades que afecten la estructura mecánica del robot.

El programa generado presenta los siguientes puntos de mejora:

- El programa no se encuentra blindado contra puntos solicitados que se encuentren fuera de su espacio de trabajo. En caso de solicitar una ubicación que el brazo robótico no pueda alcanzar se seguirá con el algoritmo sin que éste llegue a converger.
- Generalizar el problema para un manipulador articulado de n eslabones. Actualmente el algoritmo soluciona el problema para $n = 3$ eslabones.
- Incluir articulaciones prismáticas.
- Incluir una animación en MATLAB (o algún otro software) más fluida y que simule de mejor forma la aplicación de los ángulos generados.

Fuentes

Craig, John J. "*Robótica*". PEARSON EDUCATION, México, 2006.

Kevin M. Lynch and Frank C. Park. "*Modern Robotics: Mechanics, Planning and Control*". 2017

Lukas Barinka, Roman Berka. "*Inverse kinematics - Basic Methods*". Dept. of Computer science and Engineering, Czech Technical University.