



**Escuela Superior
de Ingeniería y Tecnología**
Universidad de La Laguna

Informe de Prácticas

Robótica Computacional

Grado en Ingeniería Informática

José Ramón Morera Campos

La Laguna, 29 de diciembre de 2024

Resumen

XXXXX

Palabras clave: XXXXXX, XXXX, XXXX

Índice general

1. Introducción	1
2. Cinemática Directa	2
2.1. Código Implementado	2
2.2. Ejemplo	3
2.2.1. Código	3
2.2.2. Ejecución	3
2.3. Mejoras	3
2.4. Conclusiones	4
3. Cinemática Inversa	5
3.1. Código Implementado	5
3.2. Ejercicios	6
3.2.1. Manipulador 2	6
3.2.2. Manipulador 3	6
3.2.3. Manipulador 4	6
3.3. Mejoras	6
3.3.1. Apartado Uno	6
3.4. Conclusiones	7
4. Localización	8
4.1. Sección Uno	8
4.1.1. Apartado Uno	8
4.2. Sección Dos	8
4.3. Sección Tres	8
4.4. Sección Cuatro	8
5. Filtro de Partículas	10
6. Conclusiones y líneas futuras	11

Índice de Figuras

2.1. Porción del código que se debe modificar para cada manipulador	3
3.1. Otra figura	6
4.1. Ejemplo de figura	9

Índice de Tablas

Capítulo 1

Introducción

En este informe se analizan las cuatro prácticas realizadas a lo largo de la asignatura de Robótica Computacional.

Se estudia el comportamiento de los programas realizados, mostrando ejemplos con diversas configuraciones de parámetros. Adicionalmente, se explican las modificaciones realizadas, así como otras propuestas de mejora.

Finalmente, para cada práctica se exponen unas breves conclusiones sobre los resultados obtenidos. Dichas conclusiones se exponen en inglés, siguiendo los requisitos del informe.

Capítulo 2

Cinemática Directa

En esta práctica se calcula la cinemática directa para manipuladores tridimensionales. Se modifica la morfología del robot indicando la longitud de los elementos rígidos, así como la existencia de articulaciones de revolución o prismáticas. Al ejecutar el script se indican los valores de las variables articulares.

2.1. Código Implementado

El script proporcionado contiene todo el código necesario para calcular la cinemática directa y visualizar los manipuladores. Solamente se deben modificar ciertos parámetros para adaptar el script a la morfología del robot:

- `nvar` Número de variables que se introduzcan al ejecutar el programa, correspondientes a las variables articulares.
- Parámetros de Denavit-Hartenberg son 4 vectores, uno para cada parámetro. El tamaño del vector se corresponde al número de articulaciones.
- Orígenes para cada articulación un vector con las coordenadas homogéneas de cada articulación.
- Matrices `T` son matrices que permiten realizar la transformación de un sistema de coordenadas a otro. Las matrices entre sistemas consecutivos son triviales, y se construyen con sus parámetros de Denavit-Hartenberg. Las matrices que describen la transformación entre sistemas no consecutivos se obtienen multiplicando las matrices de transformación de los sistemas intermedios. Por ejemplo, la matriz `T02`, que transforma el sistema de coordenadas 0 al 2, se obtiene realizando el producto de las matrices `T01` y `T12`.
- Coordenadas de cada articulación se calcula el origen de coordenadas del sistema de cada articulación mediante el producto punto a punto de la matriz de transformación `T0X` y el origen de coordenadas `oXX`.
- Visualización del robot se deben modificar las instrucciones de visualización para adaptarlas a la morfología del robot, incluyendo los orígenes de coordenadas calculados que se quieren visualizar.

En la figura 2.1 se muestra el fragmento del script que se debe modificar.

```

95 # Introducción de los valores de las articulaciones
96 nvar=2 # Número de variables
97 if len(sys.argv) != nvar+1:
98     sys.exit('El número de articulaciones no es el correcto ('+str(nvar)+')')
99 p=[float(i) for i in sys.argv[1:nvar+1]]
100
101 # Parámetros D-H:
102 #      1      2
103 d = [ 0, 0]
104 th = [p[0],p[1]]
105 a = [ 10, 5]
106 al = [ 0, 0]
107
108 # Orígenes para cada articulación
109 o00=[0,0,0,1]
110 o11=[0,0,0,1]
111 o22=[0,0,0,1]
112
113 # Cálculo matrices transformación
114 T01=matriz_T(d[0],th[0],a[0],al[0])
115 T12=matriz_T(d[1],th[1],a[1],al[1])
116 T02=np.dot(T01,T12)
117
118 # Transformación de cada articulación
119 o10 =np.dot(T01, o11).tolist()
120 o20 =np.dot(T02, o22).tolist()
121
122 # Mostrar resultado de la cinemática directa
123 muestra_origenes([o00,o10,o20])
124 muestra_robot ([o00,o10,o20])
125 input()
126

```

Figura 2.1: Porción del código que se debe modificar para cada manipulador

2.2. Ejemplo

Se emplea el Manipulador 3 de entre los propuestos para visualizar la cinemática directa. Se ha elegido puesto que tiene suficiente complejidad para ilustrar correctamente la práctica, pero sin ser excesiva.

2.2.1. Código

2.2.2. Ejecución

2.3. Mejoras

En esta práctica no se han implementado mejoras, pero se proponen algunas que podrían ser interesantes.

- Lectura de JSON se podría implementar la lectura de la morfología del robot desde un fichero JSON, evitando tener que modificar el script para cada manipulador.
- Modificación de los parámetros en tiempo real Sería de gran utilidad poder modificar las variables articulares mientras se visualiza el manipulador, para poder estudiar fácilmente cómo lo afectan.

2.4. Conclusions

We have studied the forward kinematics of a 3D manipulator. We have seen that we can characterize any manipulator by its Denavit-Hartenberg parameters, and that it is easy to calculate the coordinates of each joint. We have also seen that the visualization of the manipulator is very useful to understand its behavior and to check that the calculations made to obtain the parameters are correct.

Although the script provided is very useful, it has room for improvements. Also, we still have to manually calculate the Denavit-Hartenberg parameters, which can be a tedious task for complex manipulators.

Capítulo 3

Cinemática Inversa

En esta práctica se calcula la cinemática directa para manipuladores tridimensionales. Se modifica la morfología del robot indicando la longitud de los elementos rígidos, así como la existencia de articulaciones de revolución o prismáticas. Al ejecutar el script se indican los valores de las variables articulares.

3.1. Código Implementado

El script proporcionado contiene todo el código necesario para calcular la cinemática directa y visualizar los manipuladores. Solamente se deben modificar ciertos parámetros para adaptar el script a la morfología del robot:

- `nvar` Número de variables que se introduzcan al ejecutar el programa, correspondientes a las variables articulares.
- Parámetros de Denavit-Hartenberg son 4 vectores, uno para cada parámetro. El tamaño del vector se corresponde al número de articulaciones.
- Orígenes para cada articulación un vector con las coordenadas homogéneas de cada articulación.
- Matrices `T` son matrices que permiten realizar la transformación de un sistema de coordenadas a otro. Las matrices entre sistemas consecutivos son triviales, y se construyen con sus parámetros de Denavit-Hartenberg. Las matrices que describen la transformación entre sistemas no consecutivos se obtienen multiplicando las matrices de transformación de los sistemas intermedios. Por ejemplo, la matriz `T02`, que transforma el sistema de coordenadas 0 al 2, se obtiene realizando el producto de las matrices `T01` y `T12`.
- Coordenadas de cada articulación se calcula el origen de coordenadas del sistema de cada articulación mediante el producto punto a punto de la matriz de transformación `T0X` y el origen de coordenadas `oXX`.
- Visualización del robot se deben modificar las instrucciones de visualización para adaptarlas a la morfología del robot, incluyendo los orígenes de coordenadas calculados que se quieren visualizar.

En la figura 2.1 se muestra el fragmento del script que se debe modificar.

```

95 # Introducción de los valores de las articulaciones
96 nvar=2 # Número de variables
97 if len(sys.argv) != nvar+1:
98     sys.exit('El número de articulaciones no es el correcto ('+str(nvar)+')')
99 p=[float(i) for i in sys.argv[1:nvar+1]]
100
101 # Parámetros D-H:
102 #      1      2
103 d = [ 0, 0]
104 th = [p[0],p[1]]
105 a = [ 10, 5]
106 al = [ 0, 0]
107
108 # Orígenes para cada articulación
109 o00=[0,0,0,1]
110 o11=[0,0,0,1]
111 o22=[0,0,0,1]
112
113 # Cálculo matrices transformación
114 T01=matriz_T(d[0],th[0],a[0],al[0])
115 T12=matriz_T(d[1],th[1],a[1],al[1])
116 T02=np.dot(T01,T12)
117
118 # Transformación de cada articulación
119 o10 =np.dot(T01, o11).tolist()
120 o20 =np.dot(T02, o22).tolist()
121
122 # Mostrar resultado de la cinemática directa
123 muestra_origenes([o00,o10,o20])
124 muestra_robot ([o00,o10,o20])
125 input()
126

```

Figura 3.1: Otra figura

3.2. Ejercicios

Se proponen 4 manipuladores para los que se debe calcular la cinemática directa. Como se proporciona la solución del primero a modo de ejemplo, en este apartado se incluyen las soluciones de los ejercicios 2, 3 y 4.

3.2.1. Manipulador 2

3.2.2. Manipulador 3

3.2.3. Manipulador 4

3.3. Mejoras

En primer lugar, se abordan las mejoras implementadas.

3.3.1. Apartado Uno

A continuación, se describen las mejoras propuestas, que no se han llegado a implementar.

3.4. Conclusions

Capítulo 4

Localización

Los capítulos intermedios servirán para cubrir los siguientes aspectos: antecedentes, problemática o estado del arte, objetivos, fases y desarrollo del proyecto.

4.1. Sección Uno

4.1.1. Apartado Uno

Texto del apartado uno

- Item 1
- Item 2
- Item 3
- Item 4

4.2. Sección Dos

- Item 1
- Item 2
- Item 3

4.3. Sección Tres

Bla, bla, bla.

4.4. Sección Cuatro

Bla, bla, bla.

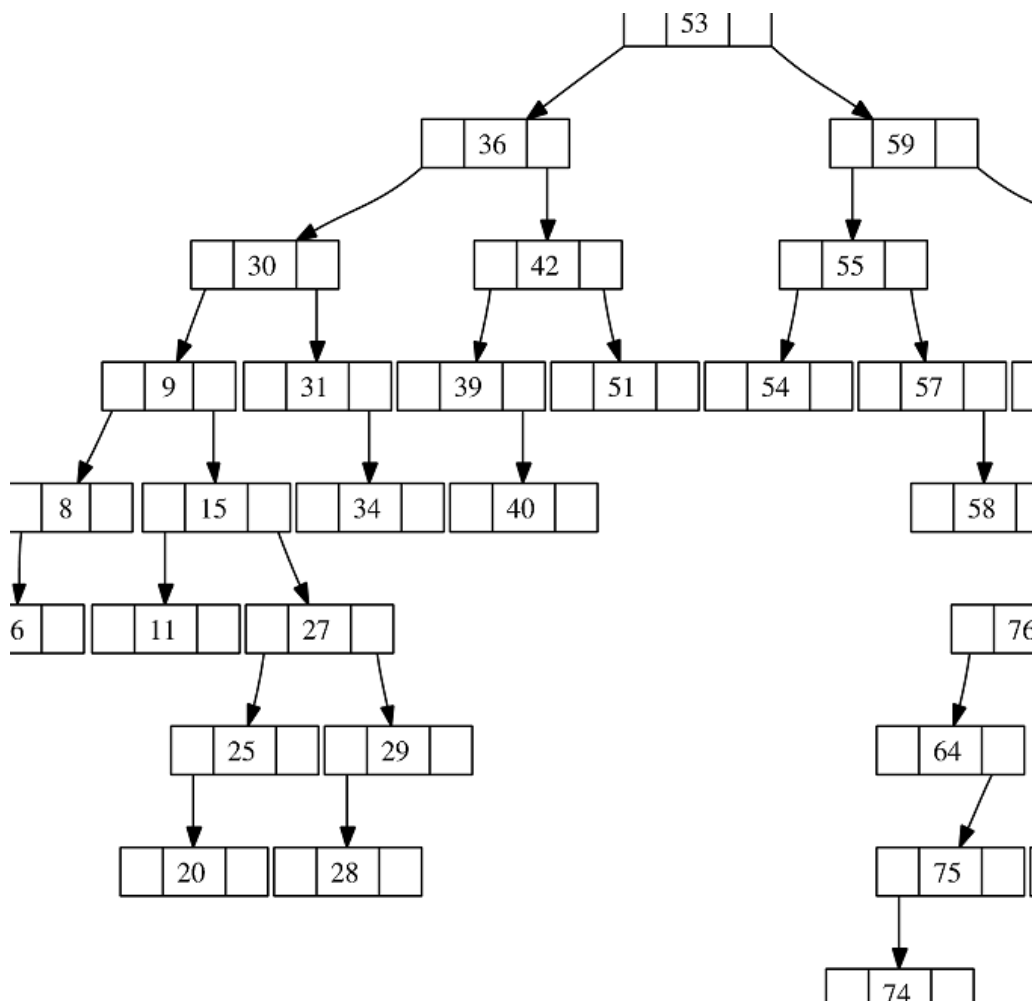


Figura 4.1: Ejemplo de figura

Capítulo 5

Filtro de Partículas

Este capítulo es obligatorio. Toda memoria de Trabajo de Fin de Grado debe incluir unas conclusiones y unas líneas de trabajo futuro

Capítulo 6

Conclusiones y líneas futuras

This chapter is compulsory. The memory should include an extended summary and conclusions in english.

Bibliografía