PRÁCTICA 1: Posición y orientación espacial y cinemática directa

1. Rotaciones en el espacio

Considere 2 tramas en el espacio, $\{A\}$ y $\{B\}$, que inicialmente coinciden. Considere además una serie de puntos, $(p_1, p_2, ..., p_{35})$, como los mostrados en la Figura 1. Las coordenadas p_{i_x}, p_{i_y} y p_{i_z} de cada punto, expresadas respecto a la trama $\{B\}$, están descritas por los vectores p_{x_B}, p_{y_B} y p_{z_B} (donde el elemento en la i-ésima posición se corresponde con p_i) que se muestran a continuación:

```
import numpy as np
pxB = np.array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 10,
                10, 10, 10, 10, 10, 10, 10, 10, 10, 11, 12, 13,
                14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14])
pyB = np.array([ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,
                 2,  3,  4,  5,  6,  7,  8,  9, 10, 10, 10, 10,
                10,  9,  8,  7,  6,  5,  4,  3,  2,  1,  0])
pzB = np.array([ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0])
pB = np.array([pxB, pyB, pzB])
```

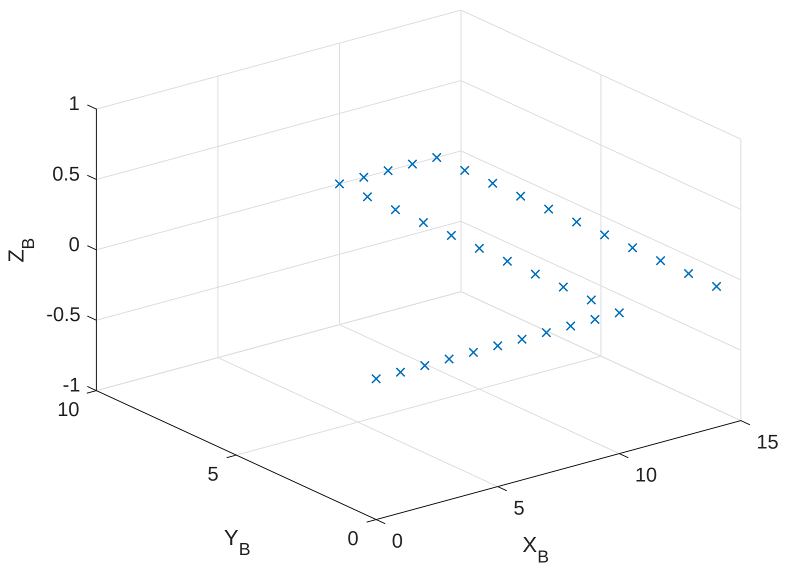

Figura 1: Representación en 3D de los puntos del ejercicio 1

Ejercicio 1

Represente las posiciones de los puntos respecto de la trama $\{A\}$ cuando:

a. La trama $\{B\}$ se gira un ángulo $\alpha = 90^{\underline{o}}$ alrededor del eje X_A.

b. La trama $\{B\}$ se gira un ángulo $\alpha = 90^{\underline{o}}$ alrededor del eje Y_A.

c. La trama $\{B\}$ se gira un ángulo $\alpha = 90^{\underline{o}}$ alrededor del eje Z_A.

Interprete el resultado.

Ejercicio 2

Represente las posiciones de los puntos respecto de la trama $\{A\}$ cuando la trama $\{B\}$ del ejercicio anterior se rota desde la posición original (superpuesta a la trama $\{A\}$) en torno al eje X_B un ángulo $\gamma = 60^{\underline{o}}$, a continuación se gira en torno al eje Y_B un ángulo $\beta = 90^{\underline{o}}$, y después se gira en torno al eje Z_B un ángulo $\alpha = 30^{\underline{o}}$.
Compruebe como afecta el orden de las rotaciones en las coordenadas respecto de la trama $\{A\}$. Comente el resultado.

2. Algoritmo de Denavit-Hartenberg

El algoritmo de Denavit-Hartenberg nos indica cómo debemos colocar los sistemas de coordenadas solidarios a cada eslabón de un brazo robótico. Una vez hecho esto, debemos calcular una tabla de parámetros, que tendrá 4 columnas y tantas filas como articulaciones tenga el robot que estamos analizando. A partir de dicha tabla podemos construir fácilmente la matriz de transformación homogénea que nos permite pasar del sistema de coordenadas de la base al del extremo del brazo robótico.

Ejercicio 3

Diseñe e implemente una función en Python que reciba como entrada una matriz de dimensión $n \times 4$ con los parámetros de Denavit-Hartenberg de un manipulador cualquiera, y devuelva como salida la matriz de transformación homogénea (como un array de dimensión 4×4) que relaciona el sistema de coordenadas de la base y el sistema de coordenadas del extremo del robot. Contemple la posibilidad de que los ángulos de rotación y los desplazamientos sean variables simbólicas (use el paquete `sympy`). Compruebe el correcto funcionamiento de la función con algunos ejemplos.