

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FEELT – FACULDADE DE ENGENHARIA ELÉTRICA
ENGENHARIA DE COMPUTAÇÃO

LUCAS ALBINO MARTINS
12011ECP022

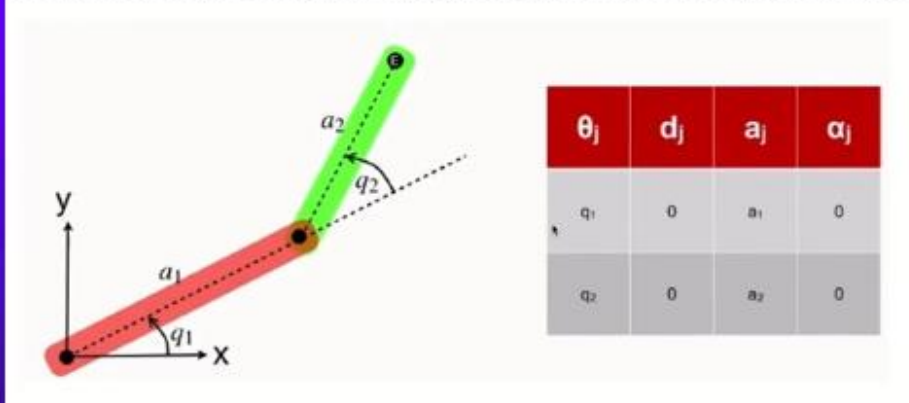
ROBÓTICA: TRABALHO 09 – Cinemática Direta

UBERLÂNDIA
2021

Trabalho 09 : considere o robô de dois elos mostrado abaixo.

Escreva um programa que recebe os parâmetros de Denavit Hartenberg e apresenta as coordenadas (x,y) do seu TCP.

Parâmetros de Denavit-Hartenberg para um robô de dois elos



Programa feito em Python para calcular a tabela de Denavit-Hartenberg de um robô de dois elos.

A captura de tela mostra o editor de código Visual Studio Code com o arquivo `denavit_hartenberg.py` aberto. O código define os parâmetros do robô e calcula as transformações homogêneas para cada elo. O terminal exibe a saída do programa, mostrando as matrizes de transformação homogênea para cada elo e a posição final do TCP.

```
1 import numpy as np
2
3 # Trabalho: Código usando tabelas de Denavit-Hartenberg com Python
4 # Transformação para frame 0 ate o frame 3.
5
6 # Comprimento do link em centímetros
7
8 # Comprimento do link em centímetros
9 a1 = 1 # Comprimento do primeiro link
10 a2 = 1 # Comprimento do segundo link
11 a3 = 1 # Comprimento do terceiro link
12
13 # Iniciando valores para os deslocamentos
14 d1 = 1 # Deslocamento para o primeiro Link
15 d2 = 1 # Deslocamento para o segundo Link
16 d3 = 1 # Deslocamento para o terceiro Link
17
18
19 # Calculando as transformações homogêneas para cada elo
20 T01 = np.array([[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, d1], [0, 0, 0, 1]])
21 T12 = np.array([[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, d2], [0, 0, 0, 1]])
22 T23 = np.array([[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, d3], [0, 0, 0, 1]])
23
24 # Calculando a transformação homogênea total
25 T03 = T01 * T12 * T23
26
27 # Posição final do TCP
28 x = T03[0, 3]
29 y = T03[1, 3]
```

Lucas@lucas-ubuntu:~/Documentos/Robotica\$ python3 denavit_hartenberg.py

Frame 0 para Frame 1:

```
[[ 6.12323400e-17 -6.12323400e-17  1.00000000e+00  0.00000000e+00]
 [ 1.00000000e+00  3.74939946e-33 -6.12323400e-17  0.00000000e+00]
 [ 0.00000000e+00  1.00000000e+00  6.12323400e-17  2.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]
```

Frame 1 para Frame 2:

```
[[ 6.12323400e-17 -6.12323400e-17 -1.00000000e+00  0.00000000e+00]
 [ 1.00000000e+00  3.74939946e-33  6.12323400e-17  0.00000000e+00]
```

Saída do programa.

```

lucas@lucas-ubuntu:~/Documentos/Robotica$ python3 denavit_hartenberg.py
Frame 0 para Frame 1:
[[ 6.12323400e-17 -6.12323400e-17  1.00000000e+00  0.00000000e+00]
 [ 1.00000000e+00  3.74939946e-33 -6.12323400e-17  0.00000000e+00]
 [ 0.00000000e+00  1.00000000e+00  6.12323400e-17  2.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]

Frame 1 para Frame 2:
[[ 6.12323400e-17 -6.12323400e-17 -1.00000000e+00  0.00000000e+00]
 [ 1.00000000e+00  3.74939946e-33  6.12323400e-17  0.00000000e+00]
 [ 0.00000000e+00 -1.00000000e+00  6.12323400e-17  2.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]

Frame 2 para Frame 3:
[[ 1. -0.  0.  0.]
 [ 0.  1. -0.  0.]
 [ 0.  0.  1.  2.]
 [ 0.  0.  0.  1.]]

Frame 0 para Frame 3:
[[-6.123234e-17 -1.000000e+00  0.000000e+00  2.000000e+00]
 [ 6.123234e-17  0.000000e+00 -1.000000e+00 -2.000000e+00]
 [ 1.000000e+00 -6.123234e-17  6.123234e-17  2.000000e+00]
 [ 0.000000e+00  0.000000e+00  0.000000e+00  1.000000e+00]]

lucas@lucas-ubuntu:~/Documentos/Robotica$ █

```

Código:

```
import numpy as np
```

```
# Trabalho: Codigo usando tabelas de Denavit-Hartenberg com Python
```

```
# Transformação para frame 0 ate o frame 3.
```

```
# Comprimento do link em centímetros
```

```
# Comprimento do link em centímetros
```

```
a1 = 1 # Comprimento do primeiro link
```

```
a2 = 1 # Comprimento do segundo link
```

```
a3 = 1 # Comprimento do terceiro link
```

```
# Iniciando valores para os deslocamentos
```

```
d1 = 1 # Deslocamento para o primeiro Link
```

d2 = 1 # Deslocamento para o segundo Link

d3 = 1 # Deslocamento para o terceiro Link

Declarando a tabela do Denavit-Hartenberg.

São quatro colunas para representar:

theta, alpha, r, e d

Convertendo os angulos em radianos.

```
d_h_table = np.array([[np.deg2rad(90), np.deg2rad(90), 0, a1 + d1],  
                      [np.deg2rad(90), np.deg2rad(-90), 0, a2 + d2],  
                      [0, 0, 0, a3 + d3]])
```

Criando as matrizes de transformação homogênea

#

Matriz homogênea do frame frame 0 para o frame 1

i = 0

```
homgen_0_1 = np.array([[np.cos(d_h_table[i,0]), -np.sin(d_h_table[i,0]) *  
np.cos(d_h_table[i,1]), np.sin(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] *  
np.cos(d_h_table[i,0]),  
                      [np.sin(d_h_table[i,0]), np.cos(d_h_table[i,0]) * np.cos(d_h_table[i,1]), -  
np.cos(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] * np.sin(d_h_table[i,0]),  
                      [0, np.sin(d_h_table[i,1]), np.cos(d_h_table[i,1]), d_h_table[i,3]],  
                      [0, 0, 0, 1]])
```

Matriz homogênea do frame frame 1 para o frame 2

i = 1

```
homgen_1_2 = np.array([[np.cos(d_h_table[i,0]), -np.sin(d_h_table[i,0]) *  
np.cos(d_h_table[i,1]), np.sin(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] *  
np.cos(d_h_table[i,0]),  
                      [np.sin(d_h_table[i,0]), np.cos(d_h_table[i,0]) * np.cos(d_h_table[i,1]), -  
np.cos(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] * np.sin(d_h_table[i,0]),  
                      [0, np.sin(d_h_table[i,1]), np.cos(d_h_table[i,1]), d_h_table[i,3]],  
                      [0, 0, 0, 1]])
```

Matriz homogênea do frame frame 2 para o frame 3

i = 2

```
homgen_2_3 = np.array([[np.cos(d_h_table[i,0]), -np.sin(d_h_table[i,0]) *  
np.cos(d_h_table[i,1]), np.sin(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] *  
np.cos(d_h_table[i,0]),  
np.sin(d_h_table[i,0]), np.cos(d_h_table[i,0]) * np.cos(d_h_table[i,1]), -  
np.cos(d_h_table[i,0]) * np.sin(d_h_table[i,1]), d_h_table[i,2] * np.sin(d_h_table[i,0]),  
0, np.sin(d_h_table[i,1]), np.cos(d_h_table[i,1]), d_h_table[i,3]],  
[0, 0, 0, 1]])
```

homgen_0_3 = homgen_0_1 @ homgen_1_2 @ homgen_2_3

Imprimindo as matrizes de transformação homogêneas para o frame 1 ate frame 0.

print("Frame 0 para Frame 1:")

print(homgen_0_1)

print()

print("Frame 1 para Frame 2:")

print(homgen_1_2)

print()

print("Frame 2 para Frame 3:")

print(homgen_2_3)

print()

print("Frame 0 para Frame 3:")

print(homgen_0_3)

print()