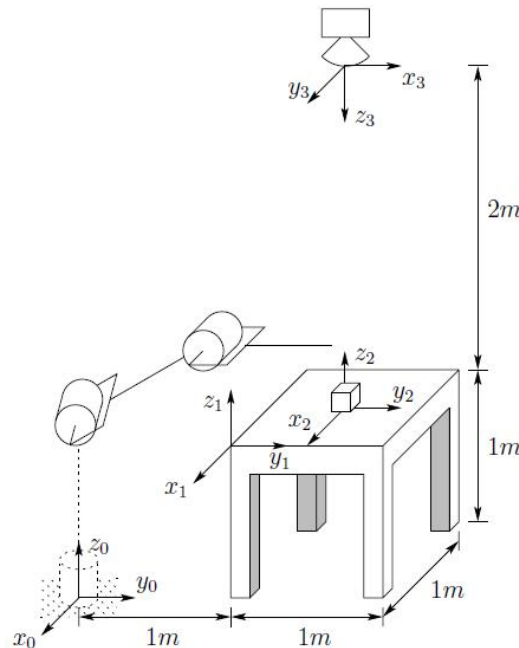


```
In [1]: import numpy as np
```

Questão 1 (peso 0,10):

Considere a figura abaixo. Um robô é colocado a 1 metro de uma mesa cuja duas de suas pernas estão sobre o eixo y_0 , como mostrado. O topo da mesa está a 1 metro de altura da base do robô, e tem 1 metro quadrado. Um sistema de coordenadas **o1x1y1z1** é fixado na intersecção de suas arestas da mesa, como mostrado. Um cubo medindo 20cm de lado é colocado no centro da mesa, com um sistema de coordenadas **o2x2y2z2** estabelecido no seu centro, como mostrado. Uma câmera é posicionada a 2 metros acima do topo da mesa, com um sistema de coordenadas **o3x3y3z3**. Ache as transformações homogêneas relacionando cada um desses sistemas ao sistema da base o **0x0y0z0**. Ache a transformação homogênea relacionando o sistema **o2x2y2z2** ao sistema **o3x3y3z3**.



Resposta

para **0x0y0z0** e **o1x1y1z1** temos somente a translação em y e em z , portanto a matriz de transformação será $R = I$ (identidade) * t

```
In [2]: np.array([
    [1,0,0,0], # translação em x = 0
    [0,1,0,1], # translação em y = 1
    [0,0,1,1], # translação em z = 1
    [0,0,0,1]  # dimensão extra da matriz homogênea
])
```

```
Out[2]: array([[1, 0, 0, 0],
               [0, 1, 0, 1],
               [0, 0, 1, 1],
               [0, 0, 0, 1]])
```

para **0x0y0z0** e **o2x2y2z2**, também não temos rotação, portanto a parte de rotação da matriz seguirá sendo a identidade

```
In [3]: np.array([
    [1,0,0,-0.5], # translação em x = -0.5m
    [0,1,0, 1.5], # translação em y = 1 + 0.5m (distancia ate a mesa + 50cm até o centro da mesa)
    [0,0,1, 1.1], # translação em z = 1m + 0.1m (distancia até a altura da mesa + distancia até o centro do cubo)
    [0,0,0, 1.]  # dimensão extra da matriz homogênea
])
```

```
Out[3]: array([[ 1. ,  0. ,  0. , -0.5],
               [ 0. ,  1. ,  0. ,  1.5],
               [ 0. ,  0. ,  1. ,  1.1],
               [ 0. ,  0. ,  0. ,  1. ]])
```

para **o0y0z0** e **o3x3y3z3**, teremos a rotação e a translação a seguir:

```
In [4]: rot_o3x3y3z3 = np.array([
# colunas
# x,y,z
[0,1, 0], # eixo y novo aponta para x do sistema anterior
[1,0, 0], # eixo x novo aponta para y do sistema anterior
[0,0,-1], # eixo z novo aponta para -z do sistema anterior
[0,0, 0] # dimensão extra da matriz homogenea
])

trans_o3x3y3z3 = np.array([
[-0.5], # translação em x = -0.5m
[ 1.5], # translação em y = 1m + 0.5m
[ 3. ], # translação em z = 3m
[ 1. ] # dimensão extra da matriz homogenea
])

np.concatenate((rot_o3x3y3z3, trans_o3x3y3z3), axis=1)
```

```
Out[4]: array([[ 0. ,  1. ,  0. , -0.5],
[ 1. ,  0. ,  0. ,  1.5],
[ 0. ,  0. , -1. ,  3. ],
[ 0. ,  0. ,  0. ,  1. ]])
```

Finalmente, para a transformação homogênea relacionando o sistema **o2x2y2z2** ao sistema **o3x3y3z3**, temos a mesma rotação entre o0 e o3, pois os eixos te o0 e o2 estão alinhados.

```
In [5]: rot2_para_3 = np.array([
# colunas
# x,y,z
[0,1, 0], # eixo y novo aponta para x do sistema anterior
[1,0, 0], # eixo x novo aponta para y do sistema anterior
[0,0,-1], # eixo z novo aponta para -z do sistema anterior
[0,0, 0] # dimensão extra da matriz homogenea
])

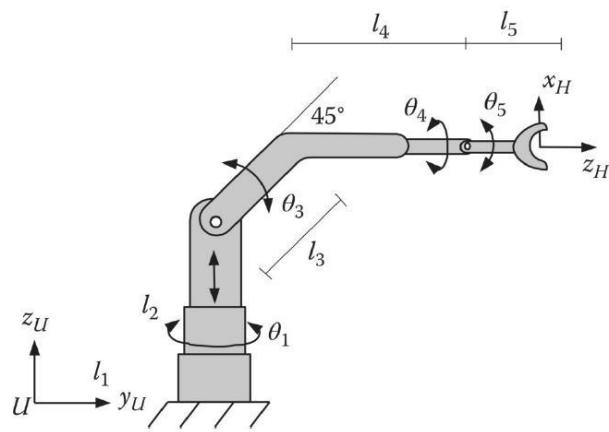
trans2_para_3 = np.array([
[0. ], # translação em x = 0m
[0. ], # translação em y = 0m
[1.9], # translação em z = 1.9m, pois precisamos considerar os 10 cm da superfície da mesa
# por essa razão, ao invés de 2m, ficariam 2m - 10cm, 1.9m
[1. ] # dimensão extra da matriz homogenea
])

np.concatenate((rot2_para_3, trans2_para_3), axis=1)
```

```
Out[5]: array([[ 0. ,  1. ,  0. ,  0. ],
[ 1. ,  0. ,  0. ,  0. ],
[ 0. ,  0. , -1. ,  1.9],
[ 0. ,  0. ,  0. ,  1. ]])
```

Questão 2 (peso 0,15):

Considere o manipulador de 5 graus de liberdade mostrado em sua posição zero. Derive as equações da sua cinemática direta, utilizando, para tanto, a convenção de Denavit-Hartenberg clássica. Não é preciso escrever a matriz geral do manipulador, basta desenvolver as matrizes que relacionam os elos. Derive as equações que resolvem cada um dos ângulos de orientação em X-Y-Z da ferramenta em função das variáveis de junta, na expressão mais simples possível.



Resposta

A especificação da matriz A é feita seguindo-se a ordem de transformações homogêneas de translação e rotação que reposicionam o sistema $i - 1$ ao i :

- rotação em torno do eixo Z_{i-1} do ângulo θ_i entre os elos;
- translação ao longo do eixo Z_{i-1} da distância d_i entre os elos;
- translação ao longo do eixo X_i (eixo X_{i-1} atual) do comprimento do elo (L_i);
- rotação em torno do eixo X_i do ângulo (α_i).

$$= \text{Rot}(Z, \theta_i) \cdot \text{Trans}(0, 0, d_i) \cdot \text{Trans}(L_i, 0, 0) \cdot \text{Rot}(X, \alpha_i)$$

```
In [6]: %%html
<style>
table {float:left}
</style>
```

Parâmetros de elos:

Elo	rot_z	trans_z	trans_x	rot_x
1	θ_1	l_2	0	-90
2	$\theta_3 + 45$	0	l_3	90
3	θ_4	l_4	0	0
4	θ_5	l_5	0	0

```
In [7]: text = """
translação inicial entre u e 0:

    [1, 0, 0, 0 ]
    [0, 1, 0, l1]
    [0, 0, 1, 0 ]
    [0, 0, 0, 1 ]
"""
print(text)
```

translação inicial entre u e 0:

```
[1, 0, 0, 0 ]
[0, 1, 0, l1]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

```
In [8]: print("""Matriz A, elo 1:
```

```
rotação de theta1 em Z
```

```
    [cos(theta1), -sen(theta1), 0, 0]
    [sen(theta1),  cos(theta1), 0, 0]
    [0           , 0           , 1, 0]
    [0           , 0           , 0, 1]
```

```
translação de l2 em Z
```

```
    [1, 0, 0, 0 ]
    [0, 1, 0, 0 ]
    [0, 0, 1, l2]
    [0, 0, 0, 1 ]
```

```
rotação de -90º em X
```

```
    [1, 0           , 0           , 0]
    [0, cos(90)    , -sen(90)    , 0]
    [0, sin(90)    , cos(90)     , 0]
    [0, 0           , 0           , 1]
```

```
Matriz final (produto das 3)
```

```
    [cos(theta1), -sen(theta1)*cos(90), sen(theta1)*sen(90) , 0]
    [sen(theta1), cos(theta1)*cos(90) , -cos(theta1)*sen(90), 0]
    [0           , sen(-90)           , cos(-90)           ,l2]
    [0           , 0                   , 0                   , 1]
```

```
ou (substituindo senos e cossenos de 90)
```

```
    [cos(theta1), 0 , sen(theta1) , 0]
    [sen(theta1), 0 , -cos(theta1), 0]
    [0           , 1 , 0           ,l2]
    [0           , 0 , 0           , 1]
```

```
""")
```

```
Matriz A, elo 1:
```

```
rotação de theta1 em Z
```

```
    [cos(theta1), -sen(theta1), 0, 0]
    [sen(theta1),  cos(theta1), 0, 0]
    [0           , 0           , 1, 0]
    [0           , 0           , 0, 1]
```

```
translação de l2 em Z
```

```
    [1, 0, 0, 0 ]
    [0, 1, 0, 0 ]
    [0, 0, 1, l2]
    [0, 0, 0, 1 ]
```

```
rotação de -90º em X
```

```
    [1, 0           , 0           , 0]
    [0, cos(90)    , -sen(90)    , 0]
    [0, sin(90)    , cos(90)     , 0]
    [0, 0           , 0           , 1]
```

```
Matriz final (produto das 3)
```

```
    [cos(theta1), -sen(theta1)*cos(90), sen(theta1)*sen(90) , 0]
    [sen(theta1), cos(theta1)*cos(90) , -cos(theta1)*sen(90), 0]
    [0           , sen(-90)           , cos(-90)           ,l2]
    [0           , 0                   , 0                   , 1]
```

```
ou (substituindo senos e cossenos de 90)
```

```
    [cos(theta1), 0 , sen(theta1) , 0]
    [sen(theta1), 0 , -cos(theta1), 0]
    [0           , 1 , 0           ,l2]
    [0           , 0 , 0           , 1]
```

In [9]: `print("""Matriz A, elo 2:`

rotação de $\theta_2 + 45^\circ$ em Z

```
[cos(theta2+45), -sen(theta2+45), 0, 0]
[sen(theta2+45),  cos(theta2+45), 0, 0]
[0              , 0              , 1, 0]
[0              , 0              , 0, 1]
```

translação de l_3 em X

```
[1, 0, 0, l3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

rotação de 90° em X

```
[1, 0      , 0      , 0]
[0, cos(90) , -sen(90) , 0]
[0, sin(90) , cos(90) , 0]
[0, 0      , 0      , 1]
```

Matriz A geral (produto das 3)

```
[cos(theta2+45), -sen(theta2+45)*cos(90), sen(theta2+45)*sen(90) , l3*cos(theta2+45)]
[sen(theta2+45), cos(theta2+45)*cos(90) , -cos(theta2+45)*sen(90), l3*sen(theta2+45)]
[0              , sen(90)              , cos(90)              , 0]
[0              , 0                    , 0                    , 1]
```

ou (substituindo senos e cossenos de 90°)

```
[cos(theta2+45), 0 , sen(theta2+45) , l3*cos(theta2+45)]
[sen(theta2+45), 0 , -cos(theta2+45), l3*sen(theta2+45)]
[0              , 1 , 0              , 0]
[0              , 0 , 0              , 1]
```

`""")`

Matriz A, elo 2:

rotação de $\theta_2 + 45^\circ$ em Z

$$\begin{bmatrix} \cos(\theta_2+45) & -\sin(\theta_2+45) & 0 & 0 \\ \sin(\theta_2+45) & \cos(\theta_2+45) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

translação de l_3 em X

$$\begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotação de 90° em X

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90) & -\sin(90) & 0 \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriz A geral (produto das 3)

$$\begin{bmatrix} \cos(\theta_2+45) & -\sin(\theta_2+45)\cos(90) & \sin(\theta_2+45)\sin(90) & l_3\cos(\theta_2+45) \\ \sin(\theta_2+45) & \cos(\theta_2+45)\cos(90) & -\cos(\theta_2+45)\sin(90) & l_3\sin(\theta_2+45) \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ou (substituindo senos e cossenos de 90°)

$$\begin{bmatrix} \cos(\theta_2+45) & 0 & \sin(\theta_2+45) & l_3\cos(\theta_2+45) \\ \sin(\theta_2+45) & 0 & -\cos(\theta_2+45) & l_3\sin(\theta_2+45) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In [10]: `print("""Matriz A, elo 3:`

`rotação de theta4 em Z`

```
[cos(theta4), -sen(theta4), 0, 0]
[sen(theta4),  cos(theta4), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação de l4 em Z`

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, l4]
[0, 0, 0, 1 ]
```

`rotação de 0º em X`

```
[1, 0      , 0      , 0]
[0, cos(0) , -sen(0) , 0]
[0, sin(0) , cos(0)  , 0]
[0, 0      , 0      , 1]
```

`Matriz A geral (produto das 3)`

```
[cos(theta4), -sen(theta4)*cos(0), sen(theta4)*sen(0) , 0]
[sen(theta4), cos(theta4)*cos(0) , -cos(theta4)*sen(0), 0]
[0           , sen(0)           , cos(0)           ,l4]
[0           , 0               , 0               , 1]
```

`ou (substituindo senos e cossenos de 90)`

```
[cos(theta4), -sen(theta4), 0 , 0]
[sen(theta4), cos(theta4) , 0 , 0]
[0           , 0           , 1 ,l4]
[0           , 0           , 0 , 1]
```

`""")`

Matriz A, elo 3:

rotação de theta4 em Z

$$\begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 0 \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

translação de l4 em Z

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotação de θ^0 em X

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriz A geral (produto das 3)

$$\begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4)\cos(\theta) & \sin(\theta_4)\sin(\theta) & 0 \\ \sin(\theta_4) & \cos(\theta_4)\cos(\theta) & -\cos(\theta_4)\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ou (substituindo senos e cossenos de 90)

$$\begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 0 \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 0 \\ 0 & 0 & 1 & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In [11]: `print("""Matriz A, elo 4:`

`rotação de theta5 em Z`

```
[cos(theta5), -sen(theta5), 0, 0]
[sen(theta5),  cos(theta5), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação de l5 em Z`

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, l5]
[0, 0, 0, 1 ]
```

`rotação de 0º em X`

```
[1, 0           , 0           , 0]
[0, cos(0)      , -sen(0)     , 0]
[0, sin(0)      , cos(0)      , 0]
[0, 0           , 0           , 1]
```

`Matriz A geral (produto das 3)`

```
[cos(theta5), -sen(theta5)*cos(0), sen(theta5)*sen(0) , 0]
[sen(theta5), cos(theta5)*cos(0) , -cos(theta5)*sen(0), 0]
[0           , sen(0)           , cos(0)           ,l5]
[0           , 0               , 0               , 1]
```

`ou (substituindo senos e cossenos de 90)`

```
[cos(theta5), -sen(theta5), 0 , 0]
[sen(theta5), cos(theta5) , 0 , 0]
[0           , 0           , 1 ,l5]
[0           , 0           , 0 , 1]
```

`""")`

Matriz A, elo 4:

rotação de θ_5 em Z

$$\begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & 0 \\ \sin(\theta_5) & \cos(\theta_5) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

translação de l_5 em Z

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotação de 90° em X

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90^\circ) & -\sin(90^\circ) & 0 \\ 0 & \sin(90^\circ) & \cos(90^\circ) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriz A geral (produto das 3)

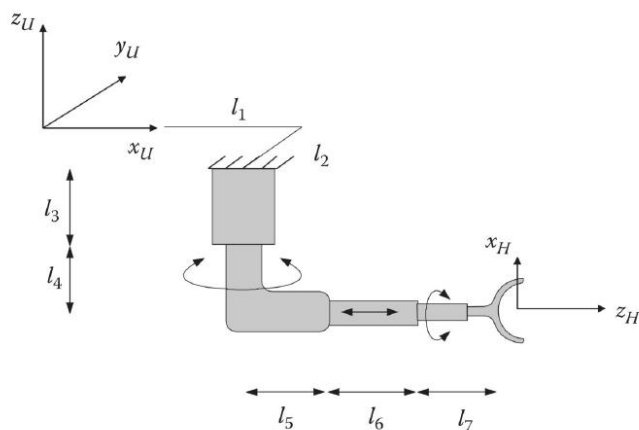
$$\begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5)\cos(90^\circ) & \sin(\theta_5)\sin(90^\circ) & 0 \\ \sin(\theta_5) & \cos(\theta_5)\cos(90^\circ) & -\cos(\theta_5)\sin(90^\circ) & 0 \\ 0 & \sin(90^\circ) & \cos(90^\circ) & l_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ou (substituindo senos e cossenos de 90°)

$$\begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & 0 \\ \sin(\theta_5) & \cos(\theta_5) & 0 & 0 \\ 0 & 1 & 0 & l_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Questão 3 (peso 0,10):

Resolva a cinemática inversa da posição para o manipulador mostrado na figura abaixo.



Resposta Para essa resposta vamos tentar o método geométrico.

- Passo 1: obter as matrizes A e cinemática direta, usando a convenção de Denavit-Hartenberg
- Passo 2: usar a solução geométrica do braço direito

Parâmetros de elos:

Elo	rot_z	trans_z	trans_x	rot_x
1	θ_1	$-l_3-l_4$	l_5+l_6	90
2	θ_2+180	l_7	0	0

```
In [12]: text = """
translação inicial entre a origem e a base:

    [1, 0, 0, l1 ]
    [0, 1, 0, -l2]
    [0, 0, 1, 0  ]
    [0, 0, 0, 1  ]
"""
print(text)
```

translação inicial entre a origem e a base:

```
[1, 0, 0, l1 ]
[0, 1, 0, -l2]
[0, 0, 1, 0  ]
[0, 0, 0, 1  ]
```

In [13]: `print("""Matriz A, elo 1:`

`rotação de theta1 em Z`

```
[cos(theta1), -sen(theta1), 0, 0]
[sen(theta1),  cos(theta1), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação de -l3 e -l4 em Z`

`translação de +l5 e +l6 em X`

```
[1, 0, 0, l5+l6 ]
[0, 1, 0, 0      ]
[0, 0, 1, -l3-l4]
[0, 0, 0, 1      ]
```

`rotação de 90º em X`

```
[1, 0      , 0      , 0]
[0, cos(90) , -sen(90) , 0]
[0, sin(90) , cos(90) , 0]
[0, 0      , 0      , 1]
```

`Matriz A geral (produto das 3)`

```
[cos(theta1), -sen(theta1)*cos(90), sen(theta1)*sen(90) , (l5+l6)*cos(theta1)]
[sen(theta1), cos(theta1)*cos(90) , -cos(theta1)*sen(90), (l5+l6)*sen(theta1)]
[0           , sen(90)           , cos(90)           , -l3-l4           ]
[0           , 0                 , 0                 , 1                 ]
```

`ou (substituindo senos e cossenos de 90)`

```
[cos(theta1), 0 , sen(theta1) , (l5+l6)*cos(theta1)]
[sen(theta1), 0 , -cos(theta1), (l5+l6)*sen(theta1)]
[0           , 1 , 0           , -l3-l4           ]
[0           , 0 , 0           , 1                 ]
```

`""")`

Matriz A, elo 1:

rotação de θ em Z

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

translação de $-l_3$ e $-l_4$ em Z

translação de $+l_5$ e $+l_6$ em X

$$\begin{bmatrix} 1 & 0 & 0 & l_5+l_6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -l_3-l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotação de 90° em X

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90) & -\sin(90) & 0 \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriz A geral (produto das 3)

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta)\cos(90) & \sin(\theta)\sin(90) & (l_5+l_6)\cos(\theta) \\ \sin(\theta) & \cos(\theta)\cos(90) & -\cos(\theta)\sin(90) & (l_5+l_6)\sin(\theta) \\ 0 & \sin(90) & \cos(90) & -l_3-l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ou (substituindo senos e cossenos de 90°)

$$\begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & (l_5+l_6)\cos(\theta) \\ \sin(\theta) & 0 & -\cos(\theta) & (l_5+l_6)\sin(\theta) \\ 0 & 1 & 0 & -l_3-l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In [14]: `print("""Matriz A, elo 2:`

```
rotação de theta2 + 180 em Z (dessa forma, o eixo X ficará apontando para cima)

    [cos(theta2+180), -sen(theta2+180), 0, 0]
    [sen(theta2+180),  cos(theta2+180), 0, 0]
    [0                , 0                , 1, 0]
    [0                , 0                , 0, 1]

translação de l7 em Z

    [1, 0, 0, 0 ]
    [0, 1, 0, 0 ]
    [0, 0, 1, l7 ]
    [0, 0, 0, 1 ]

rotação de 0 em X

    [1, 0 , 0 , 0]
    [0, 1 , 0 , 0]
    [0, 0 , 1 , 0]
    [0, 0 , 0 , 1]

Matriz A geral (produto das 3)

    [cos(theta2+180), -sen(theta2+180) , sen(theta2+180) , 0 ]
    [sen(theta2+180),  cos(theta2+180) , -cos(theta2+180) , 0 ]
    [0                , 0                , 1                , l7]
    [0                , 0                , 0                , 1 ]

""")
```

Matriz A, elo 2:

```
rotação de theta2 + 180 em Z (dessa forma, o eixo X ficará apontando para cima)

    [cos(theta2+180), -sen(theta2+180), 0, 0]
    [sen(theta2+180),  cos(theta2+180), 0, 0]
    [0                , 0                , 1, 0]
    [0                , 0                , 0, 1]

translação de l7 em Z

    [1, 0, 0, 0 ]
    [0, 1, 0, 0 ]
    [0, 0, 1, l7 ]
    [0, 0, 0, 1 ]

rotação de 0 em X

    [1, 0 , 0 , 0]
    [0, 1 , 0 , 0]
    [0, 0 , 1 , 0]
    [0, 0 , 0 , 1]

Matriz A geral (produto das 3)

    [cos(theta2+180), -sen(theta2+180) , sen(theta2+180) , 0 ]
    [sen(theta2+180),  cos(theta2+180) , -cos(theta2+180) , 0 ]
    [0                , 0                , 1                , l7]
    [0                , 0                , 0                , 1 ]
```

In [15]: text = """

```
[Xx, Yx, Zx, px ]
[Xy, Yy, Zy, py ] =
[Xz, Yz, Zz, pz ]
[0 , 0 , 0 , 1 ]
```

```
[cos(theta1), 0 , sen(theta1) , (l5+l6)*cos(theta1)]
[sen(theta1), 0 , -cos(theta1), (l5+l6)*sen(theta1)]
[0           , 1 , 0           , -l3-l4           ]
[0           , 0 , 0           , 1                ]
```

.

```
[cos(theta2+180), -sen(theta2+180) , sen(theta2+180) , 0 ]
[sen(theta2+180), cos(theta2+180) , -cos(theta2+180) , 0 ]
[0           , 0           , 1           , l7]
[0           , 0           , 0           , 1 ]
```

=

A1.A2 =

```
[cos(theta1).cos(theta2+180), cos(theta1).-sen(theta2+180), cos(theta1).sen(theta2+180) + sen(theta1).cos(theta2+180), sen(theta1).-sen(theta2+180), sen(theta1).sen(theta2+180) - cos(theta1).cos(theta2+180), -cos(theta1).sen(theta2+180) - sen(theta1).cos(theta2+180)]
[sen(theta2+180)           , cos(theta2+180)           , -cos(theta2+180) -l3-l4
[0           , 0           , 0           , 0
```

Usando método algébrico:

theta1 = arctan(py/px) = arctan(0,0)... não consegui encontrar

```
Xz = sen(theta2+180)
Yz = cos(theta2+180)
theta2 = arctan2(Xz,Yz)
```

px = 0 ... não consegui encontrar

"""

```
print(text)
```

$$\begin{bmatrix} X_x & Y_x & Z_x & p_x \\ X_y & Y_y & Z_y & p_y \\ X_z & Y_z & Z_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & (l_5+l_6)\cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & (l_5+l_6)\sin(\theta_1) \\ 0 & 1 & 0 & -l_3-l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

.

$$\begin{bmatrix} \cos(\theta_2+180) & -\sin(\theta_2+180) & \sin(\theta_2+180) & 0 \\ \sin(\theta_2+180) & \cos(\theta_2+180) & -\cos(\theta_2+180) & 0 \\ 0 & 0 & 1 & l_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

=

A1.A2 =

$$\begin{bmatrix} \cos(\theta_1)\cos(\theta_2+180) & \cos(\theta_1)-\sin(\theta_2+180) & \cos(\theta_1)\sin(\theta_2+180) + \sin(\theta_1) & \sin(\theta_1)l_7 + (l_5+l_6)\cos(\theta_1) \\ \sin(\theta_1)\cos(\theta_2+180) & \sin(\theta_1)-\sin(\theta_2+180) & \sin(\theta_1)\sin(\theta_2+180) - \cos(\theta_1) & -\cos(\theta_1)l_7 + (l_5+l_6)\sin(\theta_1) \\ \sin(\theta_2+180) & \cos(\theta_2+180) & -\cos(\theta_2+180) & -l_3-l_4 \\ 0 & 0 & 1 & l_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Usando método algébrico:

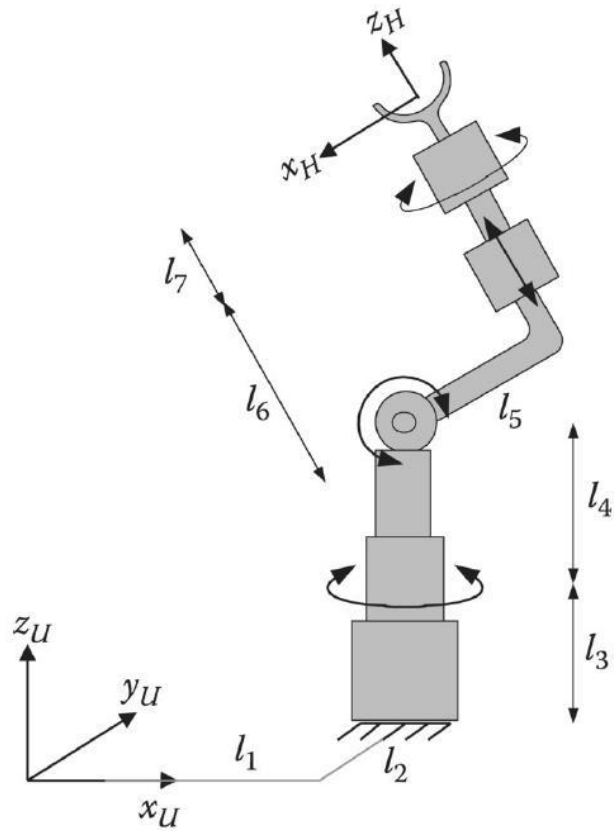
$\theta_1 = \arctan(p_y/p_x) = \arctan(0,0) \dots$ não consegui encontrar

$X_z = \sin(\theta_2+180)$
 $Y_z = \cos(\theta_2+180)$
 $\theta_2 = \arctan2(X_z, Y_z)$

$p_x = 0 \dots$ não consegui encontrar

Questão 4 (peso 0,15):

Resolva a cinemática inversa da posição para o manipulador cartesiano abaixo.



Resposta

Parâmetros de elos:

Elo	rot_z	trans_z	trans_x	rot_x
1	θ_1	l_3+l_4	0	90
2	θ_2	0	l_5	-90
3	$\theta_3 + 180$	l_6+l_7	0	0

```
In [16]: text = """
translação inicial entre u e 0:

    [1, 0, 0, l1]
    [0, 1, 0, l2]
    [0, 0, 1, 0 ]
    [0, 0, 0, 1 ]
"""
print(text)
```

translação inicial entre u e 0:

```
[1, 0, 0, l1]
[0, 1, 0, l2]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

In [17]: `print("""Matriz A, elo 1:`

`rotação de theta1 em Z`

```
[cos(theta1), -sen(theta1), 0, 0]
[sen(theta1),  cos(theta1), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação de l3 e l4 em Z`

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, l3+l4]
[0, 0, 0, 1 ]
```

`rotação de 90º em X`

```
[1, 0           , 0           , 0]
[0, cos(90)    , -sen(90)    , 0]
[0, sin(90)    , cos(90)     , 0]
[0, 0           , 0           , 1]
```

`Matriz A geral (produto das 3)`

```
[cos(theta1), -sen(theta1)*cos(90), sen(theta1)*sen(90) , 0 ]
[sen(theta1), cos(theta1)*cos(90) , -cos(theta1)*sen(90), 0 ]
[0           , sen(90)           , cos(90)           , l3+l4 ]
[0           , 0                 , 0                 , 1 ]
```

`ou (substituindo senos e cossenos de 90)`

```
[cos(theta1), 0 , sen(theta1) , 0 ]
[sen(theta1), 0 , -cos(theta1), 0 ]
[0           , 1 , 0           , l3+l4 ]
[0           , 0 , 0           , 1 ]
```

`""")`

Matriz A, elo 1:

rotação de theta1 em Z

```
[cos(theta1), -sen(theta1), 0, 0]
[sen(theta1),  cos(theta1), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

translação de l3 e l4 em Z

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, l3+l4]
[0, 0, 0, 1 ]
```

rotação de 90º em X

```
[1, 0           , 0           , 0]
[0, cos(90)    , -sen(90)    , 0]
[0, sin(90)    , cos(90)     , 0]
[0, 0           , 0           , 1]
```

Matriz A geral (produto das 3)

```
[cos(theta1), -sen(theta1)*cos(90), sen(theta1)*sen(90) , 0 ]
[sen(theta1), cos(theta1)*cos(90) , -cos(theta1)*sen(90), 0 ]
[0           , sen(90)           , cos(90)           , l3+l4 ]
[0           , 0                 , 0                 , 1 ]
```

ou (substituindo senos e cossenos de 90)

```
[cos(theta1), 0 , sen(theta1) , 0 ]
[sen(theta1), 0 , -cos(theta1), 0 ]
[0           , 1 , 0           , l3+l4 ]
[0           , 0 , 0           , 1 ]
```

In [18]: `print("""Matriz A, elo 2:`

`rotação de theta2 em Z`

```
[cos(theta2), -sen(theta2), 0, 0]
[sen(theta2),  cos(theta2), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação de l5 em X`

```
[1, 0, 0, l5]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`rotação de -90º em X`

```
[1, 0           , 0           , 0]
[0, cos(-90) , -sen(-90) , 0]
[0, sin(-90) , cos(-90) , 0]
[0, 0           , 0           , 1]
```

`Matriz A geral (produto das 3)`

```
[cos(theta2), 0 , -sen(theta2) , l5*cos(theta2)]
[sen(theta2), 0 ,  cos(theta2) , l5*sen(theta2)]
[0           , -1, 0           , 0           ]
[0           , 0 , 0           , 1           ]
```

`""")`

Matriz A, elo 2:

rotação de theta2 em Z

```
[cos(theta2), -sen(theta2), 0, 0]
[sen(theta2),  cos(theta2), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

translação de l5 em X

```
[1, 0, 0, l5]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

rotação de -90º em X

```
[1, 0           , 0           , 0]
[0, cos(-90) , -sen(-90) , 0]
[0, sin(-90) , cos(-90) , 0]
[0, 0           , 0           , 1]
```

Matriz A geral (produto das 3)

```
[cos(theta2), 0 , -sen(theta2) , l5*cos(theta2)]
[sen(theta2), 0 ,  cos(theta2) , l5*sen(theta2)]
[0           , -1, 0           , 0           ]
[0           , 0 , 0           , 1           ]
```

In [19]: `print("""Matriz A, elo 3:`

`rotação de theta1 em Z`

```
[cos(theta3+180), -sen(theta+180), 0, 0]
[sen(theta3+180), cos(theta3+180), 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`translação de l6 e l7 em Z`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, l6+l7]
[0, 0, 0, 1]
```

`rotação de 0º em X`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`Matriz A geral (produto das 3. valores invertem de sinal em razão dos 180 graus de giro)`

```
[-cos(theta3+180), sen(theta3+180), 0, 0]
[-sen(theta3+180), -cos(theta3+180), 0, 0]
[0, 0, 1, l6+l7]
[0, 0, 0, 1]
```

`""")`

Matriz A, elo 3:

rotação de theta1 em Z

```
[cos(theta3+180), -sen(theta+180), 0, 0]
[sen(theta3+180), cos(theta3+180), 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

translação de l6 e l7 em Z

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, l6+l7]
[0, 0, 0, 1]
```

rotação de 0º em X

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

Matriz A geral (produto das 3. valores invertem de sinal em razão dos 180 graus de giro)

```
[-cos(theta3+180), sen(theta3+180), 0, 0]
[-sen(theta3+180), -cos(theta3+180), 0, 0]
[0, 0, 1, l6+l7]
[0, 0, 0, 1]
```

Matriz do manipulador = ${}^R T_H = A_1.A_2.A_3$

$A_1^{-1}.{}^R T_H = A_2.A_3$

$A_1^{-1} = A_1^T$

In [20]: text = """

```
[cos(theta1), sen(theta1) , 0 , 0 ]
[0          , 0          , 1 , -l3-l4]
[sen(theta1), -cos(theta1), 0 , 0 ]
[0          , 0          , 0 , 1   ]

.

[Xx, Yx, Zx, px ]
[Xy, Yy, Zy, py ]
[Xz, Yz, Zz, pz ]
[0 , 0 , 0 , 1 ]

=

[cos(theta2), 0 , -sen(theta2) , l5*cos(theta2)]
[sen(theta2), 0 , cos(theta2)  , l5*sen(theta2)]
[0          , -1, 0          , 0          ]
[0          , 0 , 0          , 1          ]

.

[-cos(theta3+180), sen(theta3+180) , 0 , 0 ]
[-sen(theta3+180), -cos(theta3+180) , 0 , 0 ]
[0          , 0          , 1 , l6+l7 ]
[0          , 0          , 0 , 1   ]
```

"""

print(text)

```
[cos(theta1), sen(theta1) , 0 , 0 ]
[0          , 0          , 1 , -l3-l4]
[sen(theta1), -cos(theta1), 0 , 0 ]
[0          , 0          , 0 , 1   ]

.

[Xx, Yx, Zx, px ]
[Xy, Yy, Zy, py ]
[Xz, Yz, Zz, pz ]
[0 , 0 , 0 , 1 ]

=

[cos(theta2), 0 , -sen(theta2) , l5*cos(theta2)]
[sen(theta2), 0 , cos(theta2)  , l5*sen(theta2)]
[0          , -1, 0          , 0          ]
[0          , 0 , 0          , 1          ]

.

[-cos(theta3+180), sen(theta3+180) , 0 , 0 ]
[-sen(theta3+180), -cos(theta3+180) , 0 , 0 ]
[0          , 0          , 1 , l6+l7 ]
[0          , 0          , 0 , 1   ]
```

```
In [21]: text ="""
[cos(theta1).Xx + sen(theta1).Xy, cos(theta1).Yx + sen(theta1).Yy, cos(theta1).Zx + sen(theta1).Zy, cos(theta1).px + sen(theta1).py]
[Xz , Yz , Zz
[sen(theta1).Xx - cos(theta1).Xy, sen(theta1).Yx - cos(theta1).Yy, sen(theta1).Zx - cos(theta1).Zy, sen(theta1).px - cos(theta1).py]
[ 0 , 0 , 0

=

(por favor assumir que theta3 = theta3+180 caso contrário a matriz ficará muito grande)
[cos(theta2).-cos(theta3), cos(theta2).sen(theta3), -sen(theta2), -sen(theta2).(l6+l7) + l5*cos(theta2)]
[sen(theta2).-cos(theta3), sen(theta2).sen(theta3), cos(theta2) , cos(theta2).(l6+l7) + l5*sen(theta2)]
[sen(theta3) , cos(theta3) , 0 , 0
[0 , 0 , 0 , 1

"""
print(text)
```

```
[cos(theta1).Xx + sen(theta1).Xy, cos(theta1).Yx + sen(theta1).Yy, cos(theta1).Zx + sen(theta1).Zy, cos(theta1).px + sen(theta1).py]
[Xz , Yz , Zz
, pz - l3 -l4 ]
[sen(theta1).Xx - cos(theta1).Xy, sen(theta1).Yx - cos(theta1).Yy, sen(theta1).Zx - cos(theta1).Zy, sen(theta1).px - cos(theta1).py]
[ 0 , 0 , 0
, 1 ]

=

(por favor assumir que theta3 = theta3+180 caso contrário a matriz ficará muito grande)
[cos(theta2).-cos(theta3), cos(theta2).sen(theta3), -sen(theta2), -sen(theta2).(l6+l7) + l5*cos(theta2)]
[sen(theta2).-cos(theta3), sen(theta2).sen(theta3), cos(theta2) , cos(theta2).(l6+l7) + l5*sen(theta2)]
[sen(theta3) , cos(theta3) , 0 , 0
]
[0 , 0 , 0 , 1
]
```

```
In [22]: print("""
```

igualando as 2 matrizes temos que:

$\sin(\theta_1)Z_x - \cos(\theta_1)Z_y = 0$ (linha 3, coluna 3)

$\theta_1 = \cotan(Z_y/Z_x)$

$\sin(\theta_2) = -(\cos(\theta_1)Z_x + \sin(\theta_1)Z_y)$ (linha 1, coluna 3)

$\cos(\theta_2) = Z_z$ (linha 2, coluna 3)

$\theta_2 = \arctan2(-\cos(\theta_1)Z_x - \sin(\theta_1)Z_y, Z_z)$

$\sin(\theta_3) = \sin(\theta_1)X_x - \cos(\theta_1)X_y$ (linha 3, coluna 1)

$\cos(\theta_3) = \sin(\theta_1)Y_x - \cos(\theta_1)Y_y$ (linha 3, coluna 2)

$\theta_3 = \arctan2(\sin(\theta_1)X_x - \cos(\theta_1)X_y, \sin(\theta_1)Y_x - \cos(\theta_1)Y_y)$

```
""")
```

igualando as 2 matrizes temos que:

$\sin(\theta_1)Z_x - \cos(\theta_1)Z_y = 0$ (linha 3, coluna 3)

$\theta_1 = \cotan(Z_y/Z_x)$

$\sin(\theta_2) = -(\cos(\theta_1)Z_x + \sin(\theta_1)Z_y)$ (linha 1, coluna 3)

$\cos(\theta_2) = Z_z$ (linha 2, coluna 3)

$\theta_2 = \arctan2(-\cos(\theta_1)Z_x - \sin(\theta_1)Z_y, Z_z)$

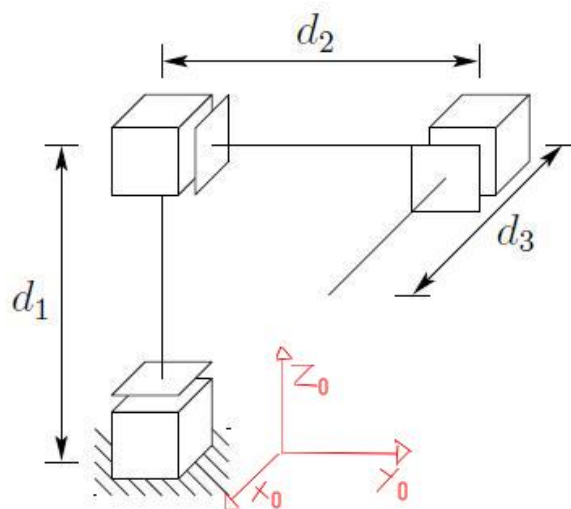
$\sin(\theta_3) = \sin(\theta_1)X_x - \cos(\theta_1)X_y$ (linha 3, coluna 1)

$\cos(\theta_3) = \sin(\theta_1)Y_x - \cos(\theta_1)Y_y$ (linha 3, coluna 2)

$\theta_3 = \arctan2(\sin(\theta_1)X_x - \cos(\theta_1)X_y, \sin(\theta_1)Y_x - \cos(\theta_1)Y_y)$

Questão 5 (peso 0,10):

Resolva a cinemática inversa da posição para o manipulador cartesiano abaixo.



Considerando que nessa questão não há mudança de orientação dos sistemas de coordenadas, temos que:

Parâmetros de elos:

Elo	rot_z	trans_z	trans_x	rot_x
1	0	d1	0	0
2	0	0	0	0
3	0	0	d3	0

** no elo 2 há uma translação de d2 em y, mas os parametros de denavich hartenberg não capturam isso

In [23]: `print("""Matriz A, elo 1:`

`rotação de 0 em Z`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`translação de d1 em Z`

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, d1]
[0, 0, 0, 1 ]
```

`rotação de 0º em X`

```
[1, 0 , 0 , 0]
[0, 1 , 0 , 0]
[0, 0 , 1 , 0]
[0, 0 , 0 , 1]
```

`Matriz final (produto das 3)`

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, d1]
[0, 0, 0, 1 ]
```

`""")`

Matriz A, elo 1:

rotação de 0 em Z

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

translação de d1 em Z

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, d1]
[0, 0, 0, 1 ]
```

rotação de 0º em X

```
[1, 0 , 0 , 0]
[0, 1 , 0 , 0]
[0, 0 , 1 , 0]
[0, 0 , 0 , 1]
```

Matriz final (produto das 3)

```
[1, 0, 0, 0 ]
[0, 1, 0, 0 ]
[0, 0, 1, d1]
[0, 0, 0, 1 ]
```


In [24]: `print("""Matriz A, elo 2:`

`rotação de 0 em Z`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`translação de d2 em Y`

```
[1, 0, 0, 0 ]
[0, 1, 0, d2]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`rotação de 0º em X`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`Matriz final (produto das 3)`

```
[1, 0, 0, 0 ]
[0, 1, 0, d2]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`""")`

Matriz A, elo 2:

rotação de 0 em Z

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

translação de d2 em Y

```
[1, 0, 0, 0 ]
[0, 1, 0, d2]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

rotação de 0º em X

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

Matriz final (produto das 3)

```
[1, 0, 0, 0 ]
[0, 1, 0, d2]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

In [25]: `print("""Matriz A, elo 3:`

`rotação de 0 em Z`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`translação de d3 em X`

```
[1, 0, 0, d3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`rotação de 0 em X`

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

`Matriz final (produto das 3)`

```
[1, 0, 0, d3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`""")`

Matriz A, elo 3:

rotação de 0 em Z

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

translação de d3 em X

```
[1, 0, 0, d3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

rotação de 0 em X

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

Matriz final (produto das 3)

```
[1, 0, 0, d3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

Matriz do manipulador = ${}^R T_H = A_1 \cdot A_2 \cdot A_3$

$A_1^{-1} \cdot {}^R T_H = A_2 \cdot A_3$

$A_1^{-1} = A_1^T$

In [26]: `text = ""`

```

        .
RTH:
    [Xx, Yx, Zx, px ]
    [Xy, Yy, Zy, py ]
    [Xz, Yz, Zz, pz ]
    [0 , 0 , 0 , 1 ]

    =

    [1, 0, 0, d3]
    [0, 1, 0, d2]
    [0, 0, 1, d1]
    [0, 0, 0, 1 ]

""
print(text)
```

```

        .
RTH:
    [Xx, Yx, Zx, px ]
    [Xy, Yy, Zy, py ]
    [Xz, Yz, Zz, pz ]
    [0 , 0 , 0 , 1 ]

    =

    [1, 0, 0, d3]
    [0, 1, 0, d2]
    [0, 0, 1, d1]
    [0, 0, 0, 1 ]
```

Dessa forma, a cinemática inversa torna-se simples:

para um ponto desejado, desloca-se as juntas prismáticas na mesma proporção, por exemplo:

para $x_1, y_1, z_1=1,2,3$ basta deslocar $d_3, d_2, d_1=1,2,3$

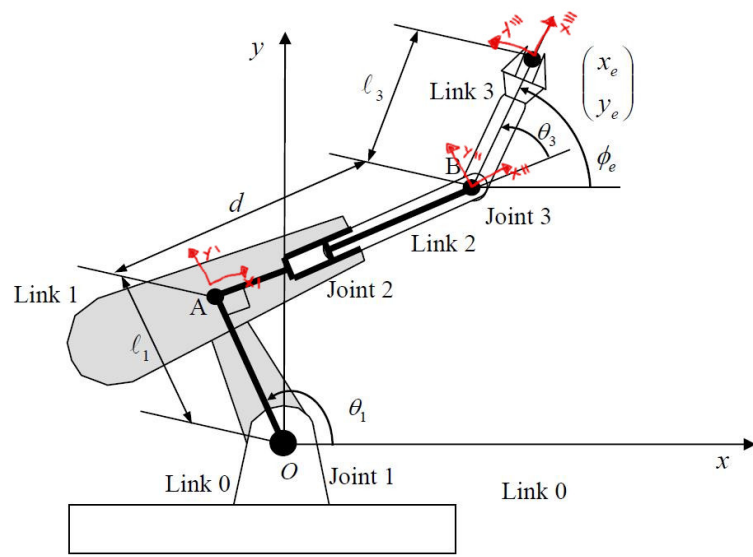
$d_1=pz$, $d_2=py$, $d_3=px$

Questão 6 (peso 0,20):

Está mostrada abaixo a construção de um robô que tem duas juntas rotativas e uma prismática. Observe que a junta prismática tem um desvio de l_1 a partir da primeira junta até a origem O. A distância entre A e O, l_1 , é constante e o ângulo entre AO e AB é de 90 graus, também constante.

A junta 2 é prismática e tem um deslocamento d , variável. Usando os parâmetros geométricos e os deslocamentos de juntas mostrados na figura, responda as seguintes questões:

- a) Obtenha as equações cinemáticas relacionando a posição e orientação do elemento terminal em função dos deslocamentos das juntas;
- b) A junta 1 pode girar 45 graus e 135 graus, e a junta 3 pode girar de -90 graus a +90 graus, enquanto a junta 2 pode se mover de l_1 a $2l_1$. Esboce o espaço de trabalho do elemento terminal E dentro do plano XY.
- c) Resolva o problema da cinemática inversa e ache os deslocamentos de juntas que levem o elemento terminal à posição e orientação desejadas: x_e , y_e , φ_e .



a) posição e orientação do elemento terminal:

In [27]: `print("""Matriz A, elo 1:`

`rotação de theta1 em Z`

```
[cos(theta1), -sen(theta1), 0, 0]
[sen(theta1),  cos(theta1), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação constante de l1 em Y`

```
[1, 0, 0, 0 ]
[0, 1, 0, l1]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`Matriz final (produto das 2)`

```
[cos(theta1), -sen(theta1), 0, -sen(theta1)*l1]
[sen(theta1),  cos(theta1), 0,  cos(theta1)*l1 ]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`""")`

Matriz A, elo 1:

rotação de theta1 em Z

```
[cos(theta1), -sen(theta1), 0, 0]
[sen(theta1),  cos(theta1), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

translação constante de l1 em Y

```
[1, 0, 0, 0 ]
[0, 1, 0, l1]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

Matriz final (produto das 2)

```
[cos(theta1), -sen(theta1), 0, -sen(theta1)*l1]
[sen(theta1),  cos(theta1), 0,  cos(theta1)*l1 ]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

In [28]: `print("""Matriz A, elo 2:`

`translação variável de d em X`

```
[1, 0, 0, d ]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`""")`

Matriz A, elo 2:

translação variável de d em X

```
[1, 0, 0, d ]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

In [29]: `print("""Matriz A, elo 3:`

`rotação de theta3 em Z`

```
[cos(theta3), -sen(theta3), 0, 0]
[sen(theta3),  cos(theta3), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

`translação constante de l3 em X`

```
[1, 0, 0, l3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

`Matriz final (produto das 2)`

```
[cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
[sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```

`""")`

Matriz A, elo 3:

rotação de theta3 em Z

```
[cos(theta3), -sen(theta3), 0, 0]
[sen(theta3),  cos(theta3), 0, 0]
[0           , 0           , 1, 0]
[0           , 0           , 0, 1]
```

translação constante de l3 em X

```
[1, 0, 0, l3]
[0, 1, 0, 0 ]
[0, 0, 1, 0 ]
[0, 0, 0, 1 ]
```

Matriz final (produto das 2)

```
[cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
[sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```

In [30]: `print("""Matriz do manipulador RTH:`

```
    [cos(theta1), -sen(theta1), 0, -sen(theta1)*l1]
    [sen(theta1),  cos(theta1), 0, cos(theta1)*l1 ]
    [0           , 0           , 1, 0           ]
    [0           , 0           , 0, 1           ]

        *

    [1, 0, 0, d ]
    [0, 1, 0, 0 ]
    [0, 0, 1, 0 ]
    [0, 0, 0, 1 ]

        *

    [cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
    [sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
    [0           , 0           , 1, 0           ]
    [0           , 0           , 0, 1           ]

""")
```

Matriz do manipulador RTH:

```
    [cos(theta1), -sen(theta1), 0, -sen(theta1)*l1]
    [sen(theta1),  cos(theta1), 0, cos(theta1)*l1 ]
    [0           , 0           , 1, 0           ]
    [0           , 0           , 0, 1           ]

        *

    [1, 0, 0, d ]
    [0, 1, 0, 0 ]
    [0, 0, 1, 0 ]
    [0, 0, 0, 1 ]

        *

    [cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
    [sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
    [0           , 0           , 1, 0           ]
    [0           , 0           , 0, 1           ]
```

In [31]: `print("""`

```
[cos(theta1), -sen(theta1), 0, cos(theta1)*d -sen(theta1)*l1 ]
[sen(theta1),  cos(theta1), 0, sen(theta1)*d +cos(theta1)*l1 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```

`*`

```
[cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
[sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```

`""")`

```
[cos(theta1), -sen(theta1), 0, cos(theta1)*d -sen(theta1)*l1 ]
[sen(theta1),  cos(theta1), 0, sen(theta1)*d +cos(theta1)*l1 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```

`*`

```
[cos(theta3), -sen(theta3), 0, cos(theta3)*l3 ]
[sen(theta3),  cos(theta3), 0, sen(theta3)*l3 ]
[0           , 0           , 1, 0           ]
[0           , 0           , 0, 1           ]
```



```
In [32]: print("""
Matriz RTH:

    [cos(theta1)*cos(theta3) -sen(theta1)*sen(theta3), -cos(theta1)*sen(theta1) - sen(theta1)*cos(theta3), 0,
    cos(theta1)*cos(theta3)*l3 - sen(theta1) * sen(theta3)*l3 + cos(theta1)*d -sen(theta1)*l1]
    [sen(theta1)*cos(theta3) + cos(theta1)*sen(theta3), -sen(theta1)*sen(theta3) + cos(theta1)*cos(theta3), 0,
    sen(theta1)*cos(theta3)*l3 + cos(theta1) * sen(theta3)*l3 + sen(theta1)*d +cos(theta1)*l1]
    [0          , 0          , 1, 0
    [0          , 0          , 0, 1

da matriz RTH, podemos obter tanto a orientação quanto a posição:

    [Xx, Yx, Zx, px ]
    [Xy, Yy, Zy, py ]
    [Xz, Yz, Zz, pz ]
    [0 , 0 , 0 , 1 ]

posições em função das juntas:

posição X = cos(theta1)*cos(theta3)*l3 - sen(theta1) * sen(theta3)*l3 + cos(theta1)*d -sen(theta1)*l1
posição Y = sen(theta1)*cos(theta3)*l3 + cos(theta1) * sen(theta3)*l3 + sen(theta1)*d +cos(theta1)*l1
Posição Z = constante

""")
```

Matriz RTH:

```
    [cos(theta1)*cos(theta3) -sen(theta1)*sen(theta3), -cos(theta1)*sen(theta1) - sen(theta1)*cos(theta3), 0,
    cos(theta1)*cos(theta3)*l3 - sen(theta1) * sen(theta3)*l3 + cos(theta1)*d -sen(theta1)*l1]
    [sen(theta1)*cos(theta3) + cos(theta1)*sen(theta3), -sen(theta1)*sen(theta3) + cos(theta1)*cos(theta3), 0,
    sen(theta1)*cos(theta3)*l3 + cos(theta1) * sen(theta3)*l3 + sen(theta1)*d +cos(theta1)*l1]
    [0          , 0          , 1, 0
    [0          , 0          , 0, 1
```

da matriz RTH, podemos obter tanto a orientação quanto a posição:

```
[Xx, Yx, Zx, px ]
[Xy, Yy, Zy, py ]
[Xz, Yz, Zz, pz ]
[0 , 0 , 0 , 1 ]
```

posições em função das juntas:

```
posição X = cos(theta1)*cos(theta3)*l3 - sen(theta1) * sen(theta3)*l3 + cos(theta1)*d -sen(theta1)*l1
posição Y = sen(theta1)*cos(theta3)*l3 + cos(theta1) * sen(theta3)*l3 + sen(theta1)*d +cos(theta1)*l1
Posição Z = constante
```

```
In [33]: print("""
Orientações em função das juntas (constante em relação ao eixo z):

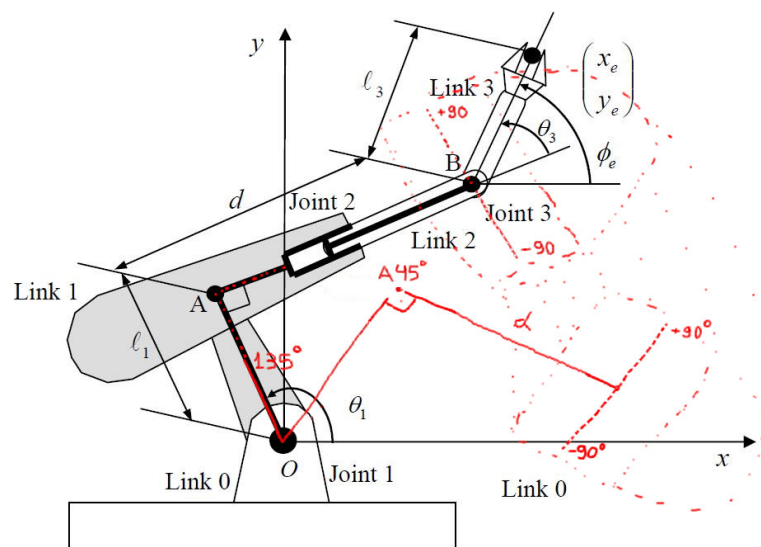
[cos(theta1)*cos(theta3) -sen(theta1)*sen(theta3), -cos(theta1)*sen(theta1) - sen(theta1)
[sen(theta1)*cos(theta3) +cos(theta1)*sen(theta3), -sen(theta1)*sen(theta3) + cos(theta1)
[0
, 0

""")
```

Orientações em função das juntas (constante em relação ao eixo z):

```
[cos(theta1)*cos(theta3) -sen(theta1)*sen(theta3), -cos(theta1)*sen(theta1) - sen(theta1)
1)*cos(theta3), 0 ]
[sen(theta1)*cos(theta3) +cos(theta1)*sen(theta3), -sen(theta1)*sen(theta3) + cos(theta1)
1)*cos(theta3), 0 ]
[0
, 0
, 1]
```

b) interpretando que θ_1 se move entre 45° e 135° , o espaço de trabalho está pontilhado em vermelho:



c) cinemática inversa, x_e , y_e , ϕ_e .

In []:

Questão 7 (peso 0,15):

Uma caixa foi girada de 30° em torno de um eixo colinear ao eixo 2. $\hat{i} + \hat{j} + \hat{k}$. Depois foi girada de 68° em torno de um outro eixo colinear a $\hat{i} + 3\hat{j} - 2\hat{k}$.

Ambos os eixos de rotação estão representados no sistema fixo de coordenadas. Você deseja retornar a caixa à sua posição e orientação original fazendo a caixa girar em torno de um único eixo. Determine qual a orientação deste eixo e de qual ângulo em torno dele a caixa deveria girar.

Obs.: Os valores positivos dos ângulos têm um significado. Use teoria de quaternions para resolver o problema.

Resposta:

```
In [34]: eixo_1 = np.array([0,2,1,1])
theta1 = np.radians(30)

eixo_2 = np.array([0,1,3,-2])
theta2 = np.radians(68)
```

Aplicamos os giros aos eixos da questão:

```
In [35]: # normaliza
norm_1 = np.sqrt(np.sum(eixo_1*eixo_1))
norm_2 = np.sqrt(np.sum(eixo_2*eixo_2))

eixo_1_normalizado = eixo_1/norm_1
eixo_2_normalizado = eixo_2/norm_2

#ordem errada?
#eixo_1_girado = (np.cos(angulo_1/2) + np.sin(angulo_1/2)) * eixo_1_normalizado
#eixo_2_girado = (np.cos(angulo_1/2) + np.sin(angulo_1/2)) * eixo_2_normalizado

## realizando os giros, aqui eu uso um
## gato para adicionar cos(theta/2) na posição 0 do vetor
eixo_1_girado = np.array([np.cos(theta1/2),0,0,0]) + (np.sin(theta1/2) * eixo_1_normalizado)
eixo_2_girado = np.array([np.cos(theta2/2),0,0,0]) + (np.sin(theta2/2) * eixo_2_normalizado)
```

$$\mathbf{q} = \sin\left(\frac{\theta}{2}\right)\hat{\mathbf{w}} \quad , \quad q_0 = \cos\left(\frac{\theta}{2}\right)$$

```
In [36]: #checa sos valores da transformação para os 2 ângulos
print("sen e cos de theta1/2:")
print(np.sin(theta1/2), np.cos(theta1/2))
print("sen e cos de theta2/2:")
print(np.sin(theta2/2), np.cos(theta2/2))
```

```
sen e cos de theta1/2:
0.25881904510252074 0.9659258262890683
sen e cos de theta2/2:
0.5591929034707469 0.8290375725550416
```

Eixo 1 girado:

```
In [37]: eixo_1_girado
```

```
Out[37]: array([0.96592583, 0.21132487, 0.10566243, 0.10566243])
```

Eixo 2 girado:

```
In [38]: eixo_2_girado
```

```
Out[38]: array([ 0.82903757, 0.14945059, 0.44835177, -0.29890118])
```

```
In [39]: # usando biblioteca nymphy-quaternion pra facilitar a multiplicação
import quaternion
q1 = np.quaternion(eixo_1_girado[0],eixo_1_girado[1],eixo_1_girado[2],eixo_1_girado[3])
q2 = np.quaternion(eixo_2_girado[0],eixo_2_girado[1],eixo_2_girado[2],eixo_2_girado[3])
q1
```

```
Out[39]: quaternion(0.965925826289068, 0.211324865405187, 0.105662432702594, 0.105662432702594)
```

Criamos um novo quaternion, que é o produto dos 2 primeiros giros:

```
In [40]: #novo quaternion
q3 = q1 * q2
q3
```

```
Out[40]: quaternion(0.753414863617681, 0.240597873464435, 0.599629244651418, -0.122161677800828)
```

Descobrimos o giro de retorno através do arco cosseno:

```
In [41]: # q3[0] escalar
escalar_q3 = 0.753414863617681
angulo = np.degrees(np.arccos(escalar_q3))*2
angulo
```

```
Out[41]: 82.2258876952747
```

```
In [42]: angulo_rad = np.radians(angulo/2)
angulo_rad
```

```
Out[42]: 0.7175562353288176
```

```
In [43]: np.sin(angulo_rad)
```

```
Out[43]: 0.6575454686027052
```

Descobrimos o novo eixo através da decomposição do eixo girado pelo ângulo, porém falta corrigir a parte escalar:

```
In [44]: q3 = q3/np.sin(angulo_rad)
q3
```

```
Out[44]: quaternion(1.14579888326004, 0.36590302108797, 0.911920579310872, -0.185784380904371)
```

```
In [45]: q3 = quaternion.as_float_array(q3)
q3[0] = 0
print(f"ângulo para voltar à posição original: {-angulo}\n")
print(f"novo eixo:{q3}")
```

ângulo para voltar à posição original: -82.2258876952747

novo eixo: [0. 0.36590302 0.91192058 -0.18578438]

```
In [ ]:
```