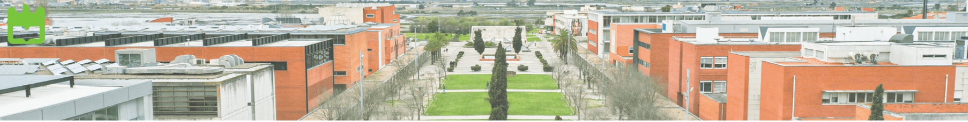


Robótica Industrial
Aula prática nº 5

Cinemática Direta
Funções adicionais
Múltiplas matrizes DH

Vítor Santos

Universidade de Aveiro

16 Out 2023

Exercício 1 - LinspaceVect

Criar a função `MQ=LinspaceVect(Qi, Qf, N)`

- A função deve emular a operação de `linspace` sobre vetores.
- Qi - vetor dos valores iniciais
- Qf - vetor dos valores finais
- N - número de elementos dos `linspace`
- MQ - matriz com todos os vetores — cada linha será o `linspace` dos elementos correspondentes de Qi até Qf.

Exemplo de aplicação

$$Q_i = \begin{bmatrix} 1 \\ 3 \\ -2 \end{bmatrix}, Q_f = \begin{bmatrix} 4 \\ 0 \\ -8 \end{bmatrix}, N = 4 \Longrightarrow MQ = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 0 \\ -2 & -4 & -6 & -8 \end{bmatrix}$$

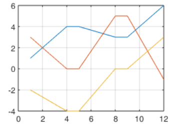
Exercício 2 - Teste da LinspaceVect

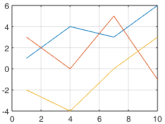
Criar e concatenar várias execuções da função `LinspaceVect`

Usar os seguintes vetores começando em Q_A, terminando em Q_D, criando uma matriz final global $MMQ = [Q_A \cdots Q_B \cdots Q_C \cdots Q_D]$ (e $N = 4$)

$$Q_A = \begin{bmatrix} 1 \\ 3 \\ -2 \end{bmatrix},\ Q_B = \begin{bmatrix} 4 \\ 0 \\ -4 \end{bmatrix},\ Q_C = \begin{bmatrix} 3 \\ 5 \\ 0 \end{bmatrix},\ Q_D = \begin{bmatrix} 6 \\ -1 \\ 3 \end{bmatrix},$$

Representar a matriz resultante *MMQ*





Repetições na matriz *MMQ*

Com a metodologia de concatenação, os vetores intermédios repetem-se. Propor uma solução em Matlab para remover essas repetições (direita).

Exercício 3 - GenerateMultiDH

Criar a função `MDH=GenerateMultiDH(DH, MQ)`

- A função permite obter as matrizes DH concretizadas (já sem variáveis) para as diversas posições das juntas de um robô. Essas diversas matrizes DH devem vir numa hipermatriz.
- Descrição dos parâmetros e retorno da função:
  - MDH - hipermatriz de matrizes DH definidas para os diversos vetores coluna de MQ.
  - DH - A matriz base de Denavit-Hartenberg que corresponde à posição zero do robô (juntas no valor de *home position*)
  - MQ - dado por `Linespacevect(Qi, Qf, N)`
    - Qi e Qf - vetores dos valores iniciais e finais das juntas
    - N - número de colunas de MQ, i.e., número de posições a calcular.

As variáveis na matriz DH

Para já, assume-se que as juntas do robô são apenas rotacionais; isso significa que as matrizes DH presentes na hipermatriz **MDH** vão ser todas iguais entre si, exceto na primeira coluna que corresponde às variáveis de cada junta θ_i onde $i = 1, 2, \dots, n$. As variáveis de junta θ_i são também usualmente designadas q_i constituindo o vetor de juntas $\mathbf{q} = [q_1, q_2, \dots, q_n]^T$

Explicação da MDH

Obtenção de MDH

- A hipermatriz **MDH** obtém-se a partir de **DH** e **MQ**.
- **DH** é matriz base de Denavit-Hartenberg para um robô com n elos, e os seus campos variáveis (assinalados abaixo) serão substituídos pelos diversos valores de juntas de MQ.

DH=

q_1	L_1	d_1	α_1
$\dots$	$\dots$	$\dots$	$\dots$
q_n	L_n	d_n	α_n

- **MQ** tem os vetores de juntas para as diversas posições desde A até f (final), onde cada coluna representa uma configuração do robô:

MQ=

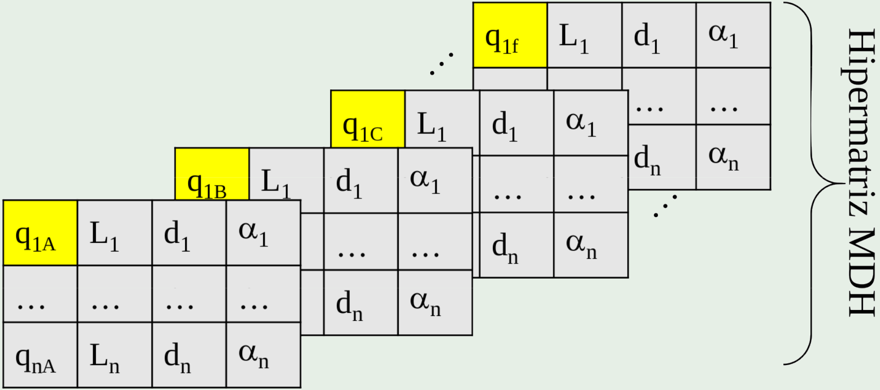
q_{1A}	q_{1B}	q_{1C}	$\dots$	q_{1f}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
q_{nA}	q_{nB}	q_{nC}	$\dots$	q_{nf}

Explicação da MDH - 2

Formato da MDH

A matriz **MDH** conterá as **DH** particulares para todas as posições intermédias de um robô representando um movimento das juntas.

Na figura realça-se a evolução da junta 1 desde a posição A até f.



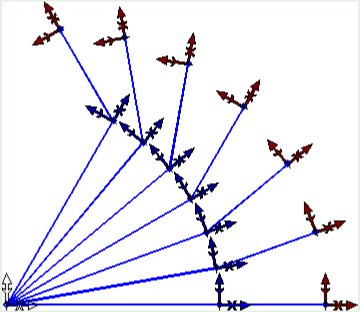
Exercício 4 - Aplicação a um caso concreto

- Obter a matriz MDH para um robo RR planar com os dados:
  - L1=2; L2=1
  - $Q_i = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$; $Q_f = \begin{bmatrix} 60^\circ & 60^\circ \end{bmatrix}^T$; $N = 7$
- Representar as 7 configurações em simultâneo com a invocação repetida da função `DrawLinks()` e `DrawFrames()`.

Ilustração do resultado

Cada uma das sete configurações usou a sua matriz `MDH(:,:,i)` para se obter a respectiva hipermatriz AA que serve para resolver o problema:

- `AA=Tlinks(MDH(:,:,i));`
- `Org=LinkOrigins(AA);`
- `h=DrawLinks(Org);`
- `H=DrawFrames(AA,...);`



Exercício 5 - Generalização da MDH

Adaptar a função `GenerateMultiDH`

- A função `GenerateMultiDH()` criada anteriormente admite que as variações em MQ são todas dos ângulos de junta θ_k . Se se quiser impor variações em d_k (junta prismática), será preciso usar um vetor adicional como argumento da função para indicar se as juntas são rotacionais ou prismáticas.
- Assim, deve-se adaptar a função `GenerateMultiDH`, para funcionar com esse vetor adicional.
- `MDH=GenerateMultiDH(DH, MQ, t)` onde t é vetor com tantos elementos quanto o número de linhas de DH.
  - Se t(k)=0, a junta k é rotacional (caso por defeito)
  - Se t(k)=1, a junta k é prismática
- A função `GenerateMultiDH()` deve estar preparada para receber 2 ou três argumentos. Em matlab isso faz-se dentro da função usando a variável intrínseca `nargin`.

Exercício 6 - Validação da nova `GenerateMultiDH`

Ilustrar a nova `GenerateMultiDH` num robô RRP (esférico).

- Estabelecer a matriz DH base usando os seguintes dados:
  - L1=L2=1; d3max=1
  - Se $t(k) = 1$, a junta k é prismática
- Obter a MDH para os seguintes intervalos das juntas:
  - $Q_i = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^\mathsf{T}$; $Q_f = \begin{bmatrix} 0 & 60^\circ & 1 \end{bmatrix}^\mathsf{T}$; $N = 5$
  - NB. A terceira junta é prismática, logo o vetor t em `MDH=GenerateMultiDH(DH,MQ,t)` será $t = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$.