Robótica Espacial
Aula prática nº 3
Cinemática inversa

Vitor Santos

Universidade de Aveiro

27 fev 2025

Sumário

Exercício 1a - Cinemática inversa do RR planar

Criar um robot RR planar

- Definir os comprimentos dos elos LA = 2 e LB = 1
- E usar a seguinte tabela de parâmetros DH:

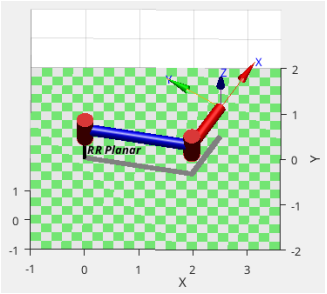
elo i	θ_i	d_i	a_i/l_i	α_i
1	θ_1	0	L_A	0
2	θ_2	0	L_B	0

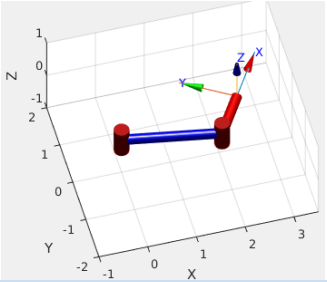
- O comando `SerialLink()` permite criar a cadeia cinemática do robô, o que pode ser feito de várias formas:
  - Criando primeiro cada elo com `L(i)=Link(...)` com os respetivos parâmetros DH e depois fazer algo como `SerialLink(DH, 'name', 'RR Planar')`;
  - Ou ...
  - criar diretamente o robô com `SerialLink(DH, 'name', 'RR Planar')` se se garantir que a tabela DH está na ordem correta.

Exercício 1b - Cinemática inversa do RR planar

Obter a cinemática inversa do robô

- Definir a posição da ponta: `target=[2.5 0.5 0]`
- Obter a transformação geométrica associada `T` (é uma simples translação)
- Obter o valor das juntas associadas através do método `ikine()` (*inverse kinematics*)
  - `q=RR_robot.ikine(T,'mask',[1 1 0 0 0 0])`
  - A máscara `[1 1 0 0 0 0]` indica para quais das 6 variáveis de posição no espaço $(x, y, z, \phi, \theta, \psi)$ se pretende a solução. Neste caso só (x, y), os dois 1s no vetor, têm relevância.
- Representar o robô nessa posição com o método `plot()`.
- Se se pretender omitir alguns elementos gráficos da vizualização, no `plot()` podem-se acrescentar opções como: `'notiles'`, `'nobase'`, `'noshadow'`, etc.





Exercício 2 - A outra redundância do RR planar

A função ikine não calcula as redundâncias!

Solução

Como o método `ikine()` da classe `SerialLink` não calcula as redundâncias (é um processo numérico), se as quisermos determinar temos de as calcular manualmente.
A solução é criar uma função dedicada que implemente as redundâncias usando as equações seguintes:

Equações da cinemática inversa analítica

$\theta_2 = \pm \arccos \frac{x^2+y^2-L_1^2-L_2^2}{2L_1L_2}$ $\quad \theta_1 = \arctan \left[\frac{y(L_1+L_2\cos\theta_2)-xL_2\sin\theta_2}{x(L_1+L_2\cos\theta_2)+yL_2\sin\theta_2} \right]$ (em Matlab usar `atan2()`)

Note-se que há duas soluções para θ_2 e como θ_1 é calculado em função de θ_2 logo haverá também duas soluções para θ_1.
A função a desenvolver deve retornar as duas soluções numa matriz, uma em cada linha.

Exercício 2 - Continuação

Criação da função invKinRRplanar

`QQ=invKinRRplanar(robot, target)`

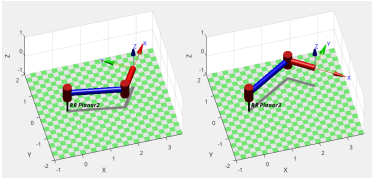
- QQ - matriz de 2 por 2 com as soluções: $\begin{bmatrix} q1_A & q2_A \\ q1_B & q2_B \end{bmatrix}$
- robot - robô previamente criado (da classse `SerialLink`)
- target - vetor com a posição de destino da ponta (x y, em geral, e z é ignorado)

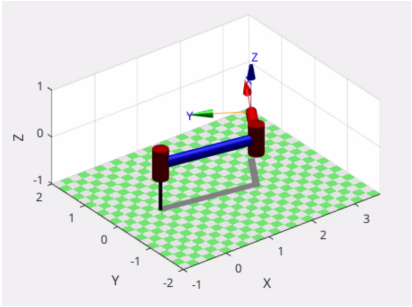
Notas

- A função deve extrair os comprimentos dos elos da variável `robot`, por exemplo: `robot.links(1).a`
- A função deve retornar uma matriz de `NaN` se não houver solução.
- Para maior robustez, a função pode também verificar se as variáveis passadas são do tipo certo e se o robô é um verdadeiro "RR".

Exercício 3 - Teste e representação da redundância

- Depois de criar a função anterior, escrever um programa para a testar.
- Ao lado da representação anterior dada pela função `ikine()`, representar um segundo robô similar mas ilustrando a redundância movendo-se alternadamente entre uma configuração e a outra, conforme os dois resultados dados por `invKinRRplanar()`.
- NOTA: para poder representar duas imagens distintas de um mesmo robô é necessário criar 2 `SerialLink` similares mas com nomes diferentes: e.g. `'RR planar', 'RR planar 2'`.
- Sugestões adicionais na página seguinte.





External player

Exercício 3 - Continuação: sugestões de implementação

- A função de cinemática inversa `Q=invKinRRplanarP(RR_robot,target)` retorna as duas configurações (conjunto de valores alternativos das duas juntas):
  - `Q(1,:)` e `Q(2,:)`
- Pode-se definir uma trajetória para um movimento entre `Q(1,:)` e `Q(2,:)` com um determinado número de amostras `NN`, por exemplo com a função `jtraj()`:
  - `fullTraj=jtraj(Q(1,:), Q(2,:),NN);`
- É possível fazer uma animação de forma rápida sem recorrer a um ciclo `for`:
  - `RR_robot2.plot(fullTraj, 'fps',30)`
  - Onde `fullTraj` é a trajetória completa a animar
  - e `...,'fps',30,...` indica que a animação decorre a 30 frames por segundo
- **N.B.** Esta forma de fazer animação é mais prática do que o ciclo `for` usado em exercícios anteriores, mas não permite que se animem outros objetos em simultâneo (por exemplo para animar um objeto transportando na ponta do robô).

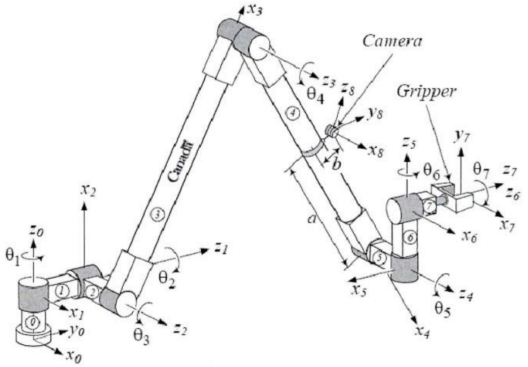
O manipulador da estação espacial internacional ISS - Canadarm2

- O manipulador a bordo da ISS tem 7 graus de liberdade
- É um sistema simétrico que consegue acoplar e desacoplar de forma reversível a ponta e base para permitir a sua recolocação em vários pontos da estação espacial.

Exercício 4 - Cinemática direta do Canadarm2

- Estabelecer a cinemática direta do Canadarm2 com base na sua tabela DH que pode ser aproximada pelo seguinte:

elo i	θ_i	d_i	a_i/l_i	α_i
1	θ_1	0.380	0	-$\pi/2$
2	θ_2	0.635	0	$\pi/2$
3	θ_3	0.504	6.85	0
4	θ_4	0.8	6.85	0
5	θ_5	0.504	0	-$\pi/2$
6	θ_6	0.635	0	$\pi/2$
7	θ_7	0.380	0	0

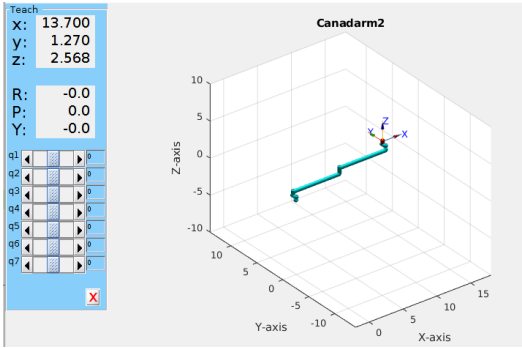


- Representar o robô na sua configuração com as juntas no valor zero

Exercício 4 - Representação e ajuste interativo das juntas

1. Representar o robô com parâmetros convenientes para melhor visualização.
2. Sugerem-se algumas indicações mas outras poderão ser usadas e o manual da toolbox tem mais informação:

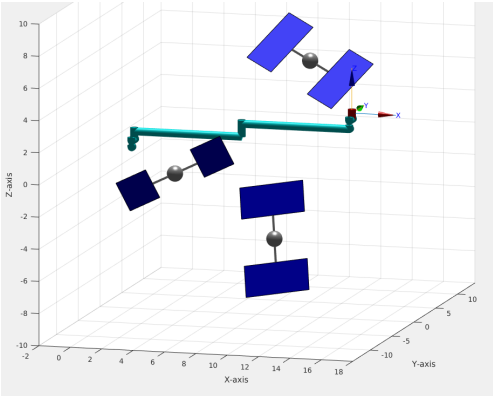
```
Canadarm2.plot(Q0,...
'scale', 0.4,...
'workspace',[-2 18 -14 14 -10 10],...
'notiles', ...
'nobase', ...
'noshadow', ...
'noname', ...
'jointdiam',0.5, ...
'nojoints', ...
'linkcolor','c' ...
);
```

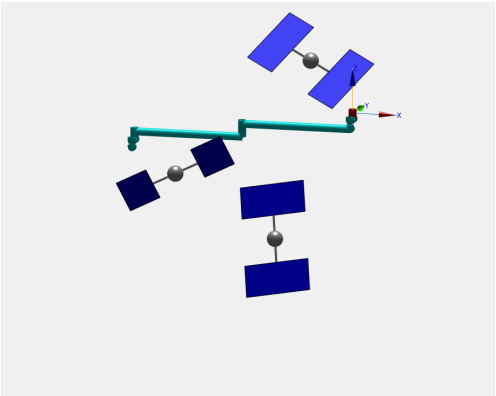


- Ativar o modo `teach` (execução do comando `Canadarm2.teach()`) e alterar as juntas manualmente para verificar os diversos movimentos.

Exercício 5 - Colocação de satélites para recolha

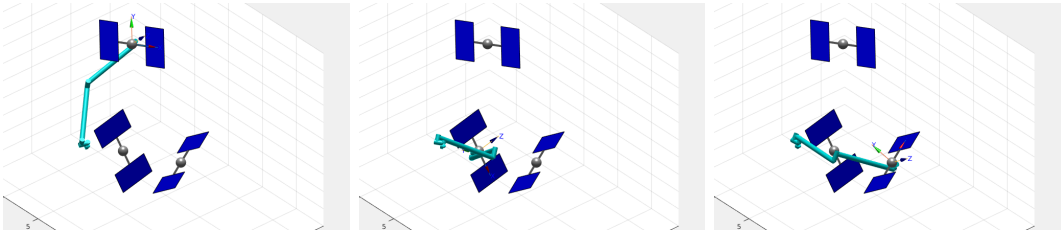
- Usando a função `drawSatellite(T)` fornecida, colocar três satélites em torno do manipulador conforme a ilustração. As coordenadas de cada um são:
  - `transl=[10 5 5 ], rotx=-45, roty=30`
  - `transl=[8 4 -6 ], rotx=-25, roty=80`
  - `transl=[5 -8 0 ], rotx=55, rotz=30`





Exercício 6 - Visita aos satélites com cinemática inversa

- Usando a cinemática inversa do manipulador (`ikine()`) fazer o movimento de visita a cada satélite, voltando entre cada um à posição de repouso inicial.
- Usar 50 pontos em cada trajetória e um `'fps'` de 20 na animação.



- O processo é meramente ilustrativo.
- Em rigor dever-se-ia definir um ponto e uma normal na superfície do satélite para o braço tocar e não colidir com os satélites, como sucede neste exemplo.
- Além desse ponto e dessa normal de contacto dever-se-ia definir também um ponto de aproximação em cada caso para garantir mais segurança no processo.