1ª Lista de Simulação
PTC3441 - Modelagem e Controle de Manipuladores Robóticos

Prof. Fábio de Oliveira Fialho

Versão 1.01

Data de entrega: esta lista deverá ser entregue via moodle do STOA USP da disciplina até às 23:00 hs do dia lá indicado.

Créditos: essa lista foi retirada dos exercícios propostos em CRAIG, J. J., Introduction to Robotics: Mechanics and Control, Addison-Weslg, 1989, 2nd edition. Ela foi redigida por Fabián Núñez.

Instruções para documentação do código: o MATLAB possui uma ferramenta nativa para formatação e publicação de códigos escritos em seu ambiente chamada *Publishing*. Ela permite ao mesmo tempo a programação e documentação do código, eliminando uma tarefa crítica e enfadonha de atualização manual da documentação a cada vez que uma linha de código é modificada. Dessa forma, esta será nossa ferramenta de documentação do trabalho de simulação e, assim, todas as listagens deverão ser entregues em formato *Publishing* pdf, além dos arquivos .m originais.

Objetivo: o objetivo deste trabalho de simulação é produzir um simulador de um braço robótico RRR planar. Para tanto, 3 listas de exercícios de simulação foram previstas, sendo esta a primeira da série. Elas estão divididas em tarefas relacionadas a cada capítulo da disciplina e levam gradativamente ao desenvolvimento das funções necessárias ao simulador.

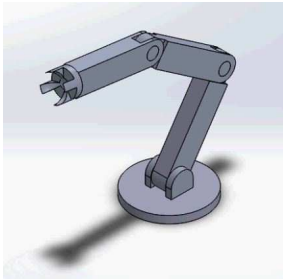

Figura 1: Manipulador planar de três juntas.

1 Descrições espaciais e transformações

1. Para fazer uma interface amigável com o usuário, desejamos descrever orientações no mundo planar através de um único ângulo, θ, ao invés de uma matriz de rotação 3x3. O usuário sempre se comunicará em termos do ângulo θ, mas internamente usaremos a forma de matrizes de rotação. Para a descrição do vetor posição de um sistema, o usuário especificará ainda os valores x e y. Assim, permitiremos ao usuário especificar um sistema como um conjunto de três valores: (x, y, θ). Internamente, desejamos utilizar uma matriz de transformação homogênea. Portanto, serão necessárias rotinas de conversão! Desenvolva uma função cuja definição em Matlab será:

   ```
   function [iform]=utoi(uform)
   ```

   onde *utoi* vem do inglês e significa *User form TO Internal form*, `iform` é uma matriz de transformação homogênea (4x4) de saída, e o argumento `uform` é um vetor que corresponde a (x, y, θ), onde θ é dado em graus.

   A rotina inversa também será necessária:

   ```
   function [uform]=itou(iform)
   ```

2. Escreva uma função para multiplicar duas transformações entre si. Use a seguinte definição de função:

   ```
   function [crela]=tmult(brela,crelb)
   ```

   Note que os nomes dos argumentos de entrada e saída seguem a seguinte notação: $brela = {}^{A}_{B}T$.

3. Escreva uma função para inverter uma transformação. Use a seguinte definição de função:

   ```
   function [arelb]=tinvert(brela)
   ```

   Novamente, os nomes dos argumentos de entrada e saída seguem a seguinte notação: $brela = {}^{A}_{B}T$.

4. As seguintes definições de sistema são dadas:

$$\begin{aligned} {}^{U}_{A}T &= [x\ y\ \theta] = [11\ \ -1\ 30]\,, \\ {}^{B}_{A}T &= [x\ y\ \theta] = [0\ 7\ 45]\,, \\ {}^{C}_{U}T &= [x\ y\ \theta] = [-3\ \ -3\ \ -30]\,. \end{aligned} \tag{1}$$

   Estes sistemas são entradas na representação do usuário [x,y,θ] (θ dado em graus). Desenhe um diagrama desses sistemas em 2-D (inspire-se na Fig. 2.15 do livro do Craig) que qualitativamente mostre sua disposição. Escreva um programa que chame TMULT e TINVERT, desenvolvidas nos exercícios 2 e 3, tantas vezes quanto for necessário para resolver ${}^{B}_{C}T$. Depois imprima ${}^{B}_{C}T$ nas representações do usuário e interna.

2 Cinemática direta do manipulador

5. Escreva uma função para calcular a cinemática do robô planar 3R da Figura 1, isto é, uma rotina que tenha como entrada os valores dos ângulos de junta e como saída uma matriz de transformação (o sistema do punho com relação ao sistema da base). Use a seguinte definição de função:

```
function [wrelb]=kin(theta,L)
```

onde *theta* é o vetor de ângulos das juntas, L é o vetor de comprimentos dos ligamentos e *wrelb* é a transformação homogênea que leva o sistema do punho para o sistema da base, ${}^{B}_{W}T$. (Os dados do manipulador são $l_1 = 0.5$ e $l_2 = 0.3$ metros.)

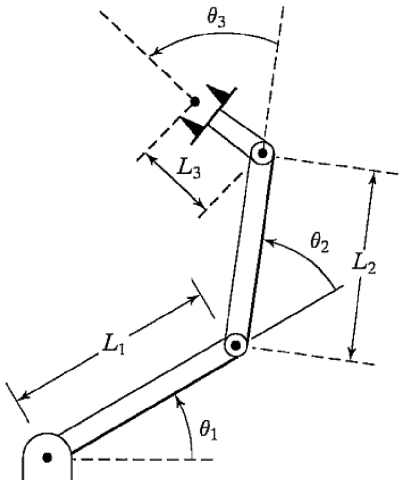

Figura 2: Braço planar de três juntas.

6. Escreva uma rotina que calcule onde a ferramenta está com relação ao sistema da estação. A entrada da rotina é um vetor de ângulos de junta:

```
function [trels]=where_robot(theta,trelw,srelb,L)
```

Obviamente, a função WHERE_ROBOT deve fazer uso das descrições do sistema da ferramenta (*trelw*) e do sistema da base do robô (*srelb*) para computar a localização da ferramenta com relação ao sistema da estação.
L é um vetor de comprimentos dos ligamentos e tem os mesmos valores do exercício 5.

7. Os sistemas da ferramenta e da estação são definidos para uma certa tarefa pelo usuário da seguinte forma:

$$
\begin{aligned}
{}^{W}_{T}T &= [x\ y\ \theta] = [0.1\ 0.2\ 30.0], \\
{}^{B}_{S}T &= [x\ y\ \theta] = [-0.1\ 0.3\ 0.0]
\end{aligned}
$$

Calcule a posição e a orientação da ferramenta com relação ao sistema da estação para as três seguintes configurações (em graus) do braço:

$$\begin{array}{c} [\theta_1\ \theta_2\ \theta_3] = [0.0\ 90.0\ -90.0]\,, \\ [\theta_1\ \theta_2\ \theta_3] = [-23.6\ -30.3\ 48.0]\,, \\ [\theta_1\ \theta_2\ \theta_3] = [130.0\ 40.0\ 12.0]\,. \end{array} \tag{2}$$

3 Cinemática inversa do manipulador

8. Escreva uma função para calcular a cinemática inversa para o manipulador de três juntas da Figura 2. A rotina deve ter a seguinte estrutura:

```
function [near,far,sol]=invkin(wrelb,current,L,thetalim)
```

onde

wrelb é a matriz de transformação do punho para a base e descreve para onde deseja-se levar o robô;
current é a posição atual do robô (dada como um vetor de ângulos das juntas) e serve como informação para definir qual a solução mais próxima;
L é um vetor de comprimentos dos ligamentos;
thetalim é uma matriz 2 x n de limites superiores e inferiores de operação para cada variável de junta;
near é a solução mais próxima;
far é a segunda solução;
sol é um flag que indica se soluções foram encontradas. Assim, *sol*=0 se nenhuma solução foi encontrada.

Os comprimentos dos ligamentos (em metros) são:

$$l_1 = 0.5 \text{ e } l_2 = 0.3$$

As faixas de movimentação das juntas são

$$-170^{\text{o}} \leq \theta_i \leq 170^{\text{o}}$$

Teste a sua rotina chamando-a com a função KIN para demonstrar que elas são inversas uma da outra.

9. Uma ferramenta foi incorporada ao manipulador de 3 juntas. Esta ferramenta é descrita por ${}^{W}_{T}T$, o sistema da ferramenta com relação ao sistema do punho. Além disso, um usuário descreveu sua área de trabalho, o sistema da estação com relação à base do robô, como ${}^{B}_{S}T$. Escreva uma função

```
function
[near,far,sol]=solve_robot(goal,current,trelw,srelb,L,thetalim)
```

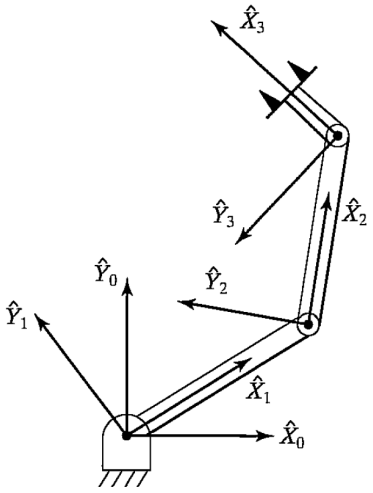

Figura 3: Manipulador planar de três juntas.

onde

goal é a posição objetivo do sistema $\{T\}$ com relação ao sistema $\{S\}$, descrita no formato do usuário;
trelw é a matriz de transformação do sistema $\{T\}$ para o sistema $\{W\}$;
srelb é a matriz de transformação do sistema $\{S\}$ para o sistema $\{B\}$;
Os outros parâmetros são exatamente como na função INVKIN.
O comando SOLVE_ROBOT deverá chamar as funções TMULT, TINVERT e INVKIN.

10. Escreva um programa principal que aceite um sistema objetivo $\{G\}$ expresso em termos de x,y e ϕ. Esta especificação do objetivo é dada em $\{T\}$ com relação a $\{S\}$, que é o jeito pelo qual o usuário deseja especificar objetivos.

    O robô está usando a mesma ferramenta na mesma área de trabalho dos exercícios de programação da Seção 2, então $\{T\}$ e $\{S\}$ são definidos assim:

$$\begin{aligned} {}^{W}_{T}T &= [x\ y\ \theta] = [0.1\ 0.2\ 30.0], \\ {}^{B}_{S}T &= [x\ y\ \theta] = [-0.1\ 0.3\ 0.0] \end{aligned}$$

    Calcule os ângulos de junta para cada um dos seguintes sistemas objetivo:

$$\begin{aligned} [x_1\ y_1\ \phi_1] &= [0.0\ 0.0\ -90.0], \\ [x_2\ y_2\ \phi_2] &= [0.6\ -0.3\ 45.0], \\ [x_3\ y_3\ \phi_3] &= [-0.4\ 0.3\ 120.0], \\ [x_4\ y_4\ \phi_4] &= [0.8\ 1.4\ 30.0] \end{aligned}$$

    Assuma que o robô começará com todos os seus ângulos em 0.0 e que se movimentará para esses três objetivos em sequência. O programa deverá encontrar a solução mais próxima com relação ao ponto objetivo anterior. Você deverá chamar SOLVE_ROBOT e WHERE_ROBOT como forma de validação para ter certeza que são realmente funções inversas.