Multibody simulation

Exercise 3

Dimitar Dimitrov, Henrik Andreasson

Örebro University

September 16, 2016

Main points covered

- FGM & IGM for a 2 DoF planar system
- FGM for a n DoF 3D system
- defining a model in the `bMSd` toolbox

FGM - Forward Geometric Model, IGM - Inverse Geometric Model

Supplementary materials used in this exercise

All materials are available for download at the course website

1. For this exercise, you might find the following functions from the bMSd toolbox useful
   - in directory `bMSd/general_purpose`: `Draw_System.m`, `poly3.m`, `trajC.m`
   - in directory `bMSd/examples`: `example_1.m`
   - in directory `bMSd/models`: `model_system_2.m`
2. `path_EE.m`
3. bMSd.pdf

Task 1 (FGM model of a 2 DoF planar manipulator)

Problems

1. implement a Matlab function that computes the FGM of a 2 DoF planar manipulator with revolute joints

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{bmatrix}$$

   function interface: `pe=fgm_2(q)`
   input: `q` - joints angles
   output: `pe` - Cartesian coordinates of the end-effector

2. implement the above function using homogeneous matrices
   function interface: `[p2,pe,R1,Re]=fgm_T2(q)`
   input: `q` - joints angles
   output: `p2` - Cartesian coordinates of joint 2
   output: `pe` - Cartesian coordinates of the end-effector
   output: `R1` - orientation of link 1
   output: `Re` - orientation of the end-effector

Task 1 (FGM model of a 2 DoF planar manipulator)

3. Plot the configuration of the 2 DoF manipulator for different joint angles (including the body-fixed frames). For example, you can use
   `q1 = poly3(0,pi/4,0,0,[0:0.1:1])`
   `q2 = poly3(0,pi/2,0,0,[0:0.1:1])`
   to generate values for the joint angles. If you want to plot a "continuous motion" you could use

```
n = length(q1);
for i=1:n
  q = [q1(i);q2(i)];
  [p2,pe,R1,Re]=fgm_T2(q); % evaluate the FGM

  cla;hold on;
  % ----------------------------------
  % put your display function here ...
  % set the plot axes ...
  % ----------------------------------
  drawnow
end
```

Task 2 (IGM model of a 2 DoF planar manipulator)

Problems

1. implement a Matlab function that computes the IGM of a 2 DoF planar manipulator with revolute joints
   function interface: `q = igm(pe,l,up_down)`
   input: `pe` - desired Cartesian coordinates of the end-effector
   input: `l` - lengths of the two links
   input: `up_down` - a flag for choosing **up-elbow** or **down-elbow**
   output: `q` - joints angles
2. generate a feasible profile for the Cartesian position of the end-effector (see file `path_EE.m`) and follow it with the manipulator by repeatedly
   - solving the IGM
   - solving the FGM (Task 1)
   - using your display function (Task 1)

   test using both up-elbow and down-elbow configurations
3. your function should return `q=NaN`, if the end-effector Cartesian position is not feasible (*i.e.*, the IGM does not have a solution)

Task 3 (parent-child structure)

consider $n+1$ coordinate frames $\{\mathcal{L}_0\}, \{\mathcal{L}_1\}, \dots, \{\mathcal{L}_n\}$

- the configuration of $\{\mathcal{L}_1\}$ is specified w.r.t. $\{\mathcal{L}_0\}$
- the configuration of $\{\mathcal{L}_2\}$ is specified w.r.t. $\{\mathcal{L}_1\}$, etc.
- frame $\{\mathcal{L}_k\}$ is a **parent** of frame $\{\mathcal{L}_{k+1}\}$
- frame $\{\mathcal{L}_{k+1}\}$ is a **child** of frame $\{\mathcal{L}_k\}$
- if frame $\{\mathcal{L}_k\}$ moves, so does its child $\{\mathcal{L}_{k+1}\}$

Problems

1. Choose a way to encode the relative configuration between two frames (and parameters that you can use to set it). You could use a Matlab "structure" to store your parameters, *e.g.*,

```
S(k).x = ...
S(k).y = ...
S(k).z = ...
S(k).alpha = ...
S(k).beta = ...
S(k).gamma = ...
```

2. set the configuration of frame $\{\mathcal{L}_{k+1}\}$ as seen from frame $\{\mathcal{L}_k\}, k = 0, \dots, n-1$ and plot the frames $\{\mathcal{L}_k\}$ and $\{\mathcal{L}_{k+1}\}$

Task 3 (parent-child structure)

3. write a function that rotates frame $\{\mathcal{L}_{k+1}\}$ around the local x-axis of frame $\{\mathcal{L}_k\}$ and plot the frames (do not forget to update the configurations of the children frames)
4. rotate frame $\{\mathcal{L}_{k+1}\}$ around the local y-axis of frame $\{\mathcal{L}_k\}$
5. rotate frame $\{\mathcal{L}_{k+1}\}$ around the local z-axis of frame $\{\mathcal{L}_k\}$
6. translate frame $\{\mathcal{L}_{k+1}\}$ along the x-axis of frame $\{\mathcal{L}_k\}$
7. translate frame $\{\mathcal{L}_{k+1}\}$ along the y-axis of frame $\{\mathcal{L}_k\}$
8. translate frame $\{\mathcal{L}_{k+1}\}$ along the z-axis of frame $\{\mathcal{L}_k\}$

Task 4 (FGM for a n DoF 3D system)

Consider a fixed-base, open-loop 3D manipulator with n rotational joints. In order to define its geometric structure, we can associate a coordinate frame with each of its rigid bodies. For example

- the origin of a frame $\{\mathcal{L}_k\}$ coincides with joint k
- the position of joint $k+1$ is expressed relative to frame $\{\mathcal{L}_k\}$
- the orientation of frame $\{\mathcal{L}_{k+1}\}$ is expressed relative to frame $\{\mathcal{L}_k\}$
- the relative one DoF of frame $\{\mathcal{L}_{k+1}\}$ w.r.t. frame $\{\mathcal{L}_k\}$ has to be defined

Essentially, the above points amount to associating a homogeneous matrix ${}^{j}_{k}\boldsymbol{T}(q_k)$ to the k^{th} link/joint where link j is the parent of link k ($i.e.$, link j directly supports link k).

- in ${}^{j}_{k}\boldsymbol{T}(q_k)$, the relative motion of link k w.r.t. link j is parametrized by the joint variable q_k
- ${}^{j}_{k}\boldsymbol{T}(0)$ gives the initial offset in configuration between links k and j
- you can use $\boldsymbol{q} = (q_1, \dots, q_n)$ so specify the configuration of the system

Task 4 (FGM for a n DoF 3D system)

Problems

1. Define the model of a fixed-base, open-loop 3D manipulator with n rotational (or/and prismatic) joints. It might be helpful to develop a Matlab "structure" that specifies what is the relative DoF between a child and a parent link. For example

```
% frame 'k' rotates around the 'z' axis
% of the parent frame
DoF(k).axis = 'z'
DoF(k).type = 'R'
% frame 'j' translates along the 'x' axis
% of the parent frame
DoF(j).axis = 'x'
DoF(j).type = 'P'
```

2. write a function that computes the FGM for your manipulator
   function interface: `T = fgm_Tn(q)`
   input: `q` - joint angles
   output: `T` - "cell array" of n homog. matrices (`T{k}` is the k^{th} one)

Task 4 (FGM for a n DoF 3D system)

Problems continued...

1. display the system (including the body-fixed frames)
2. create a keyboard interface to your system to be able to change the $\boldsymbol{q} = (q_1, \ldots, q_n)$ vector while running the visualization.
   The following example illustrates how keypresses can be handled in MATLAB. Note that a figure (and not the console) has to be active

```
q=zeros(2,1); stop=0; inc=0.01;
while ~stop
    waitforbuttonpress;
    if strcmp(get(gcf,'currentcharacter'),'a');
        q(1) = q(1) + inc
    elseif strcmp(get(gcf,'currentcharacter'),'z');
        q(1) = q(1) - inc
    elseif strcmp(get(gcf,'currentcharacter'),'q');
        stop=1; disp('quit');
    end
end
```

3. verify that the system behaves as intended

Defining a model in the bMSd toolbox

see

- file bMSd.pdf
- directory `bMSd/models`
- directory `bMSd/examples`

Lab 3 - receipt

After completing all exercises in the lab you need to present the results to the teacher/assistant in order to pass the lab. It should be completed latest 2 weeks after the lab was given. The receipt is g iven to you as a proof that you passed the lab. A separate record will be kept by the teacher.
Be prepared to demonstrate all results before the presentation.

Name:	Personal number: