MATLAB Assignment 6

Due Date: October 27, 2011

Turning in your homework: place all your files in a directory called "lastNameDate"; zip that directory and drop it in the mailbox at Learn@UW.

I suggest that you start working on this assignment no later than Monday and stop by during office hours on Mo and Wd with questions.

Problem 1 [straightforward]. In MATLAB assignment 2, you put together a parser capable of reading in the following definition of an absolute-X constraint:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
AbsoluteX: 3                   % id of this constraint
Body: 3                        % id of participating body
xPprime: 0.4                   % x of point P on moving body, expressed in LRF
yPprime: -0.3                  % y of point P on moving body, expressed in LRF
xPground: 4                    % x of point P on ground
yPground: 2.3                  % y of point P on ground
CmotionFunction: 0.1*t+1/9     % provides expression for C(t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

For this assignment, based on the discussion we had in class, slightly modify that piece of code such that it is capable of handling the following improved definition of the absolute-X constraint:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
AbsoluteX: 3                 % id of this constraint
Body: 3                      % id of participating body
xPprime: 0.4                 % x of point P on moving body, expressed in LRF
yPprime: -0.3                % y of point P on moving body, expressed in LRF
CmotionFunction: 4+0.1*t+1/9 % provides expression for C(t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

NOTE: CmotionFunction above can never be NONE. You always have to prescribe a function of time, which nonetheless in some cases it can be a constant function.

Problem 2 [straightforward]. This is similar to Problem 1, but this time around you'll have to modify the parser for the absolute-Y constraint. The old definition has redundant components and will be replaced by this new definition:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
AbsoluteY: 4                       % id of this constraint
Body: 2                            % id of participating body
xPprime: 0.4                       % x of point P on moving body, expressed in LRF
yPprime: -0.3                      % y of point P on moving body, expressed in LRF
CmotionFunction: 2*sin(3*t+pi/2) % provides expression for C(t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

NOTE: CmotionFunction above can never be NONE.

Problem 3 [not straightforward]. Consider the pendulum at page 60 of the textbook (Example 3.2.1), whose total length is 2. For that pendulum assume that its mass is 2.5 and its mass moment of inertia is 1. All units are S.I. Also assume that a motion is prescribed on the pendulum to the effect that its orientation should change like $\phi(t) = \pi/2 + 2\pi t$. The state of this model is to be characterized by an array of Cartesian generalized coordinates $\mathbf{q} = [x, y, \phi]^T$.

(a) Generate a pair of files, simplePend321.acf and simplePend321.adm that you would use in conjunction with simEngine2D to run a Kinematics Analysis for 1 second, with a time step of 0.01 seconds. For the constraints, rely on the new absolute-X and absolute-Y constraints defined above.

(b) Produce at time $t = 0$ and post on the forum the following set of quantities: $\Phi(\mathbf{q}_0, t)$, $\Phi_\mathbf{q}$, $\nu(\mathbf{q}_0, t)$, and $\gamma(\dot{\mathbf{q}}_0, \mathbf{q}_0, t)$. For the value of the velocity at $t = 0$ take $\dot{\mathbf{q}}_0 = [-2\pi, 0, 2\pi]^T$. Choose the generalized coordinates $\mathbf{q}_0$ to be consistent with the given motion when evaluated at time $t = 0$.

(c) (Not straightforward) Write a MATLAB program that does the following: first, it parses simplePend321.acf and simplePend321.adm to read in the model and type of analysis. Next, it runs a loop over time from $t_0 = 0$ to $t_{end} = 1$ in increments specified in the acf file; i.e., $\Delta t = 0.01$. At each time step t_1, t_2,..., $t_{end} = 1$, it should print out the value of $\Phi(\mathbf{q}, t)$, $\Phi_\mathbf{q}$. On the forum, report the value that you get at $t_{37} = 0.37$. Note that as time passes, the values of x, y, and ϕ change so make sure you work with the correct values when you compute $\Phi(\mathbf{q}, t)$ and $\Phi_\mathbf{q}$.

Problem 4[relatively straightforward]. Assume that $\mathbf{q} = [x, y]^T$ and

$$\boldsymbol{\Phi}(\mathbf{q}) = \begin{bmatrix} 3x + \sin(xy) - 4 \\ x - e^{cos(y)} \end{bmatrix}$$

(a) Compute the Jacobian (sensitivity) matrix $\boldsymbol{\Phi}_\mathbf{q}$.

(b) Write a MATLAB script that implements the Newton-Raphson method for solving the nonlinear system $\mathbf{\Phi}(\mathbf{q}) = \mathbf{0}_{2\times 1}$. For the stopping criteria, use the condition that the norm of the correction should be less than 10^{-8}.

(c) Post on the forum the results that you get when you run the MATALB script above using as a starting point $\mathbf{q}^{(0)} = [1.5, 1.5]^T$. Note that I already posted my results there, specifically, at each iteration I reported the norm-2 of the residual.