ME751-Assignment 6

Problem 1. [MATLAB/Python/C] Implement two (possibly more, if you choose to) functions that provide all the computational kinematics quantities that are associated with the basic GCons Φ^{DP2} and Φ^{D}. Stick with the $\mathbf{r} - \mathbf{p}$ formulation. Specifically, your code should be able to return any or all of the following quantities:

(i) The value of the expression of the constraint

(ii) The right-hand side of the velocity equation ν

(iii) The right-hand side of the acceleration equation γ

(iv) The expression of the partial derivatives $\Phi_{\mathbf{r}}$ and $\Phi_{\mathbf{p}}$

You might decide to have one function for Φ^{DP2} and yet another function for Φ^{D} to produce for these two GCons the quantities in (i) through (iv) above. Since you don't need all these quantities all the time, you should devise a methodology (perhaps using flags) to instruct the subroutine what quantities are actually needed.

Observation 1: There are two ways to go about specifying the attributes of your GCon; i.e., who i is, who j is, where point P is, what $\bar{\mathbf{a}}_i$ is, etc.:

- Harder approach, but it puts you on a good trajectory: use an input file that contains information about the *attributes* of your model's GCons. You will also have to think about the format in which you expect the user to provide the required attributes. In conclusion, think about how you want the model attributes to be provided to you, and then generate a file with extension "`.mdl`" (from model) that actually stores the attributes. As an example, take a look at the ADAMS `adm` file that is available here. If you want to push it to an extreme, you can actually start using `adm` files as your model definition files. To put things in perspective, this `mdl` file is the input file; i.e., what the `simEngine3D` code parses to generate the actual GCons that you have in your model. For another possible setup of this "`.mdl`" input file that specifies the model's definition and initial conditions, take a look at the example provided with the previous assignment. This was a sample definition file that a ME451 student came up with. It was used for 2D dynamics though.

- Easier: Hard code for now the attributes of the GCons you have in your mechanism. This means that you have another file, call it `driver.py` that calls your two functions and provides the right arguments to your function. Upon return from the function call, your driver prints out the values that were computed by the two functions.

Observation 2: Keep in mind that body j can be the ground. In this case, $j = 0$. This is relevant since the number of columns in the Jacobian will be half – there are no partial derivatives with respect to $\mathbf{r}_j$ or $\mathbf{p}_j$. You will have to be able to properly dimension the size of the vectors/matrices that you work with to account for the fact that one of the bodies in the GCon is ground and as such it does not bring generalized coordinates along.

Observation 3: Keep in mind that the RHS of the acceleration equation needs to be the one for the $\mathbf{r} - \mathbf{p}$ formulation since this is what you implement. We discussed about how to derive $\hat{\gamma}$ in class.

Problem 2. [MATLAB/Python/C] In an input file "`revJoint.mdl`" (or in your `driver.py` file, if you chose to hardcode the GCon attributes), define your first mechanism model that will have one pendulum hanging as shown in Figure 1. The pendulum has $L = 2$, it is symmetric as shown in the Figure 1, and it is attached to ground at its tip (point Q). The revolute joint on the ground is at point O of location (0,0,0); that is, the origin of the G-RF. Note that the pendulum moves in the global OYZ plane; also note that the $O'x'y'$ plane of the L-RF is parallel to the OYZ plane of the G-RF. The pendulum is subjected to a motion specified as $f(t) = \frac{\pi}{4}\cos(2t)$. One way to implement this motion is through a *DP1* basic geometric constraint.

Your code should be able to provide the following information at $t = 0$: value of $\mathbf{\Phi}(\mathbf{q}, t)$, $\mathbf{\Phi}_{\mathbf{q}}$, ν, and γ. All units used are SI. The program that will be executed for this problem should be named "`simEngine3D-A6P2`" followed by the filename extension of your programming language.

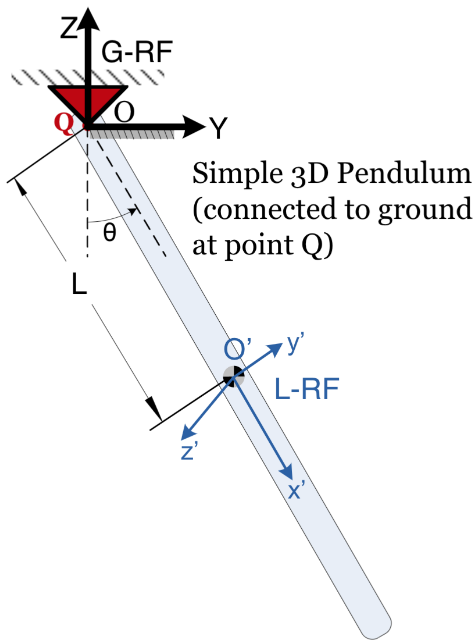

Figure 1: Pendulum with revolute joint.

Problem 3. [MATLAB/Python/C] This problem builds on Problem 2. The schematic of the mechanism is shown in Figure 1. The rigid body is subjected to a motion specified as $f(t) = \frac{\pi}{4}\cos(2t)$.

You will need to carry out a Kinematics Analysis for the mechanism for 10 seconds of its evolution. To this end, use a time grid with time steps of $dt = 10^{-3}$. Add **six plots** to your report including location/velocity/acceleration of point O'/Q in the G-RF as a function of time, and discuss the results that you obtained for point Q. The program that will be executed for this problem should be named "`simEngine3D-A6P3`" followed by the filename extension of your programming language.

Deliverables: Include your code for the three problems (use sub-folders if necessary) and a pdf report including the plots, discussion, and interesting findings (if any). Upload them as a zipped folder on Canvas.