ECE 455: Computer Project #1 Description

Due Date: October 26, 2017 by 8 pm

Project Overview:
The first computer project will require you to read in two files, one specifying the joint positions and velocities for a robotic manipulator at key frames and another describing the position and orientation of an object at the same key frames. Using these two files, you will write a MATLAB script to generate a complete trajectory for the robot joints and object. The two trajectories will then be stored as separate files which can be displayed using the provided *Animate* program.

Input File Descriptions:
The input files your program should accept are *robot.key* and *object.key*. Both programs are defined as follows:

1. The file *robot.key* contains the robot joint variable positions and velocities at key frames. From this file, you are to generate the file *robot.ang*, used by animate, which contains the values of the joint variables of an n-joint manipulator for each frame. The format for *robot.key* is defined as:

$$\begin{array}{llll} k & t & & \\ q_{11} & q_{12} & \cdots & q_{1n} \\ \dot{q}_{11} & \dot{q}_{12} & \cdots & \dot{q}_{1n} \\ \vdots & \vdots & \dots & \vdots \\ q_{k1} & q_{k2} & \cdots & q_{kn} \\ \dot{q}_{k1} & \dot{q}_{k2} & \cdots & \dot{q}_{kn} \end{array}$$

   where k and t are integers denoting the number of key frames described and the number of total frames to be in the simulation, respectively. Note, t/k may not be an integer meaning during animation, you will not necessarily display each key frame. Keep this in mind when moving from one interpolant curve to the next! The position and velocity of joint j for key frame i are denoted by q_{ij} and $\dot{q}_{ij}$ respectively.

2. Similarly, *object.key* provides the homogeneous transformation describing an object's position and orientation at key frames. You will need to generate a file called *object.traj* that contains the homogeneous transformations for the object at every frame. The format for the file *object.key* is:

$$\begin{array}{llll} k & t & & \\ n_{1x} & o_{1x} & a_{1x} & p_{1x} \\ n_{1y} & o_{1y} & a_{1y} & p_{1y} \\ n_{1z} & o_{1z} & a_{1z} & p_{1z} \\ \vdots & \vdots & \vdots & \vdots \\ n_{kx} & o_{kx} & a_{kx} & p_{kx} \\ n_{ky} & o_{ky} & a_{ky} & p_{ky} \\ n_{kz} & o_{kz} & a_{kz} & p_{kz} \end{array}$$

   where k and t have the same definition as above. The interpolation is to be done by maintaining continuity in velocity. Assume the object is to start and end at rest.

Suggested Coding Procedure:
As with any large programming task, it is highly recommended you break the project into smaller segments. Focus on generating the joint trajectories for *robot.ang* first. You will need to use a reasonable position interpolation strategy for each joint and between each key frame. Once you have equations describing the joint trajectories between each key frame, you will need to step along the curves using appropriate u-values to generate the joint

positions for each of the total frames to be displayed. The joint positions should be stored in the file *robot.ang* using the format:

$$\begin{array}{llll} t & & & \\ q_{11} & q_{12} & \cdots & q_{1n} \\ \vdots & \vdots & \cdots & \vdots \\ q_{t1} & q_{t2} & \cdots & q_{tn} \end{array}$$

where t is again the total number of frames and q_{ij} is the position of joint j at frame i. The *robot.ang* file should be saved in a folder entitled *robot* along with provided files which describe the geometry of the robot and the position of the robot base.

Test your robot trajectory by running *animate.exe* similar to *persp_display* for the truncated pyramid. The input file for *animate* is the provided file labelled *script.* Before using, open *script* and make sure you understand what parameters are being specified. If you do not have an object, remove those lines from the script.

Once satisfied with the resulting smooth motion, begin working on the orientation interpolation needed to generate *object.traj* file from the provided *object.key* file. The format for *object.traj* is:

$$\begin{array}{llll} t & & & \\ n_{1x} & o_{1x} & a_{1x} & p_{1x} \\ n_{1y} & o_{1y} & a_{1y} & p_{1y} \\ n_{1z} & o_{1z} & a_{1z} & p_{1z} \\ \vdots & \vdots & \vdots & \vdots \\ n_{tx} & o_{tx} & a_{tx} & p_{tx} \\ n_{ty} & o_{ty} & a_{ty} & p_{ty} \\ n_{tz} & o_{tz} & a_{tz} & p_{tz} \end{array}$$

with parameters as defined above. The resulting *object.traj* file should be saved in the *object* folder which also contains the point polygon and point of view files for the object.

Useful MATLAB Commands:
The download folder includes a subfolder of MATLAB code which includes several scripts to help you complete the necessary interpolations. Look at the provided scripts and understand the assumed formatting, what the scripts accomplish, and how you can incorporate the scripts into your code. Feel free to modify the scripts or write your own if you prefer though!

As a final nudge, you will need to read in and write to external files. There are many ways to accomplish this in MATLAB as a Google search will reveal; however, one simple method is to use the commands:

- importdata('robot.key')
- dlmwrite('robot/robot.ang',output,'-append','delimiter',' ')

to read in and and write out data. It is still recommended you use the MATLAB 'help' command to understand the format of these commands.

What to Submit: When you are satisfied with the performance of you code, type a short README document as described in the *Guidelines* document under the *Computer Projects* module on Canvas. Upload a ¡last_name¿.zip folder containing your README file and all MATLAB code to Canvas by the assignment deadline.

Note: Once again, k and t are both variables and will be changed by Megan while grading so make sure your code allows for these modifications!