MeEn 537 Homework #7

1. Using the Simulink puma560 model that was given to you for the previous homework, develop a PD invidual joint controller and test it for multiple desired configurations throughout the workspace always starting at the default zero position. If the performance is very poor, you can add gravity compensation. Turn in plots of performance for at least two different positions as well as the K_p and K_d gain matrices that you used. Do not spend significant amounts of time tuning if you can't get good performance.

2. Do the following (using "sl_ctorque" and "sl_fforward") and for each part, turn in plots showing the trends that you found with at least a few sentences describing your results.

   (a) Compute and display the joint tracking error for the torque feedforward and computed torque cases. Experiment with different motions, control parameters and sample rates (higher and lower).

   (b) The dynamic model of a robot is never actually known, we can only invert our best estimate of the rigid-body dynamics. In simulation we can model this uncertainty by using the "perturb" method (see the online documentation), which returns a robot object with inertial parameters varied by plus and minus the specified percentage. Modify the models in Simulink so that the RNE block is using a robot model with parameters perturbed by 15%. Meaning that the inverse dynamics are computed for a slightly different dynamic model than the robot we are controlling. Investigate the effects on error for both the torque feedforward and computed torque cases.

3. Assuming a camera defined as follows "cam = CentralCamera('default');" and given the real world points (P) and pixel coordinates (p) in the file "hw7_prob3.mat", find the unknown transformation between the camera and the object for the following pairs (this problem is simple, but you will need to understand it well for the next problem, you may even want to plot the points to make sure you understand what is happening):

   (a) P1 and p1

   (b) P2 and p2

   (c) P3 and p3

4. Implement your own version of PBVS that includes puma560 robot arm. You may do this in simulink or a MATLAB script or both (it may be helpful to model your work off of the model in problem 5). You may assume that the object pose (and therefore tracked points) are known to make the problem easier. However, you must actually simulate the kinematics of the robot arm.

5. Using "sl_arm_ibvs" compare the results they obtain for IBVS with what you were able to do in problem 4.

6. Using your implementation from problem 4, now include robot dynamics and control in the model. This means that you can't just integrate the joinit velocities to get position but will

need to use the robot model blocks. In addition, after using PBVS to generate a desired velocity or position in time, use the computed torque controller to track the trajectory. Compare this to the performance in problem 4 and comment on the differences.