Homework 6: Teleoperation

MEAM 520, University of Pennsylvania
Katherine J. Kuchenbecker, Ph.D.

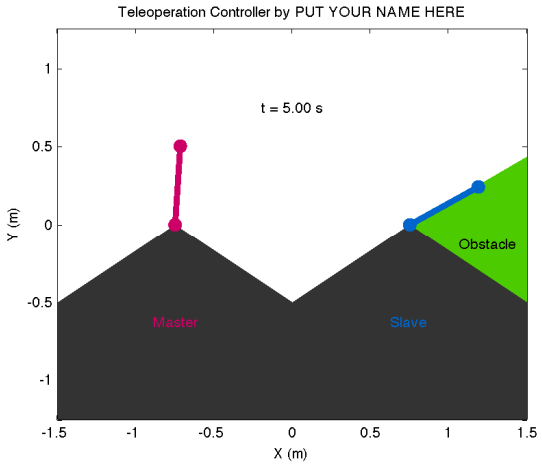
December 3, 2012

This assignment is due on **Friday, December 7**, by 5:00 p.m. If you don't finish by that time, you may turn it in with no penalty by 5:00 p.m. on Wednesday, December 12. After that deadline, no further assignments may be submitted. Because it is short, this assignment is worth 30 points (half the value of homework assignments 1 through 5).

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit should be your own work, not copied from a peer or a solution manual.

Teleoperation Controller *(30 points)*

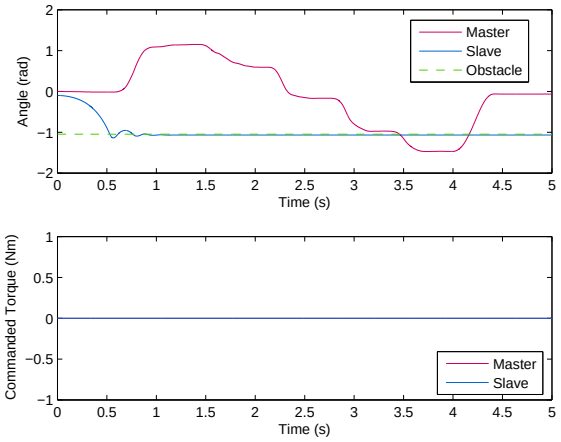
Your task is to write a good controller for a simple simulated teleoperation system. The image below shows a snapshot of the simulated teleoperator. It includes a one-degree-of-freedom master robot (left, in magenta) and an identical one-degree-of-freedom slave robot (right, in blue). Each device consists of a single revolute joint, much like the pair of Immersion Impulse Engine 2000 joysticks that Professor Kuchenbecker discussed in Lecture 18 (on November 20). Each robot's joint angle is measured in radians, with counterclockwise positive and straight up equal to zero.



The stationary bracket to which the robots are attached is shown in dark gray. The robots can move freely through this region because they are not in the same plane. There are no obstacles in the master's workspace, and the robots are too short to touch each other directly. There is one obstacle in the slave's workspace; shown in green, it begins at `obstacleAngle` and extends infinitely in the negative direction. You should move the obstacle around to test different environments; the controller that you write should work for any obstacle location, so it should not use the variable `obstacleAngle` in any way.

To simulate the presence of a human user holding onto the end of the master robot, the master moves through a pre-determined trajectory that you select. Six trajectories are provided (`masterMovement1.mat` ... `masterMovement6.mat`), and you can also write your own. The slave has pre-programmed dynamics that are hidden from your view inside the function `getSlaveTheta.p`. These dynamics include but are not

limited to inertia, gravity, friction, actuator saturation, and encoder quantization. When you first run the starter code, you will see that the slave just falls into the obstacle and stays there, while the master robot follows the default pre-determined trajectory. To help you understand what is happening in the simulation, the starter code animates the entire interaction and graphs the resulting angles and commanded torques over time, as shown in the sample graph below.



The simulated teleoperation system runs a servo loop at 1000 Hz, which you should not change. At each time step, it obtains the new position of the master (`masterTheta`) and the slave (`slaveTheta`) in radians. Your job is to specify the torque to command to the master (`masterTau`) and the slave (`slaveTau`) in newton·meters to yield good transparency (good tracking and good feel in free space, good feel in contact with the obstacle) and good stability (no extraneous ongoing oscillations). There should be no motion scaling or clutching between the two devices. The slave torque that you specify will directly affect the movement of the slave robot, while the master torque that you specify will merely be graphed. Following standard robotics convention, a positive torque moves the joint in the positive direction. It is expected that your controller will include gravity compensation, a proportional term, and a derivative term on both devices.

Download the starter code from this assignment's page on the class wiki, change the name of the provided script (`teleoperation_starter.m`) to include your PennKey, put your name where it says `'PUT YOUR NAME HERE'`, and make sure the starter code works correctly before starting to modify it. Near the top, you can change the movement of the master, the initial position of the slave, the angle of the obstacle, and the speed of the animation. When you're ready, put your controller code between the two lines of stars, modify whatever other simulation settings you want to elucidate the behavior of the system, and comment the final code you write. Follow the instructions below to submit your Matlab files.

Submitting Your Code

Follow these instructions to submit your code:

1. Start an email to `meam520@seas.upenn.edu`
2. Make the subject *Homework 6: Your Name*, replacing *Your Name* with your name.
3. Attach your correctly named MATLAB script (`teleoperation_yourpennkey.m`) to the email, along with any other files that you created. You do not need to submit the provided `masterMovement.mat` or `getSlaveTheta.p` files. Please **do not zip your files together** before attaching them; just attach them as individual files.
4. Optionally include any comments you have about this assignment.
5. Send the email.

You are welcome to resubmit your code if you want to make corrections. To avoid confusion, please state in the new email that it is a resubmission, and include all of your MATLAB files, even if you have updated only some of them.