Assignment 2: Teleoperation and Admittance Control

ME 328: Medical Robotics
Stanford University ◆ Winter 2019

Due in the drop box outside Allison's office by 5:00 pm on Friday, January 25

0. Guest Speaker: After the seminar on Friday, January 18, look for a question posted to Canvas about the seminar, and answer it online by the time this assignment is due.

1. Readings: You can download these papers from http://www.stanford.edu/class/me328/#readings

G. Niemeyer, C. Preusche, S. Stramigioli, D. Lee. Chapter 43: Telerobotics. In Springer Handbook of Robotics, pages 1085-1108, 2016. *This book chapter gives an overview of the field of teleorobotics.*

N. Enayati, E. De Momi and G. Ferrigno. Haptics in Robot-Assisted Surgery: Challenges and Benefits. IEEE Reviews in Biomedical Engineering, 9: 49-65, 2016. *This review article gives an overview of the state of the art and challenges in providing haptic feedback in minimally invasive robot-assisted surgery. Skim this one.*

J. Marescaux, J. Leroy, M. Gagner, F. Rubino, D. Mutter, M. Vix, S. E. Butner, M. K. Smith. Transatlantic Robot-Assisted Telesurgery. Nature, 413:379-380, 2001. *Current clinical robots are designed to work with the surgeon in the same room as the patient – where the time delay in communication between master and patient-side (follower) robot is minimal. However, the possibility of long-distance surgery enabled by telerobotics is intriguing. (This work was done with the Computer Motion ZEUS system, which is no longer commercially available.)*

A. (Niemeyer et al.) Briefly describe the three main control architectures for telerobotics.

B. (Niemeyer et al.) In the subsection on "Position Control and Kinematic Coupling," the authors describe *clutching*. Clutching is commonly used in teleoperated surgical robots. Explain why this is needed.

C. (Enayati) What are some important challenges with regard to sensing haptic information on the patient side?

D. (Marescaux et al.) Explain how the remote robotic surgery was evaluated and provide your opinion about whether this is a satisfactory evaluation method.

Important Notes for Problems 2 and 3: These problems should be done in a **team of two students**. When you have completed both of these problems, you will have a single executable file that can render any of the different parts of the problems, depending on which letter is pressed. Name your executable JaneDoeJohnRoeME328_Ass2.exe and your .cpp file JaneDoeJohnRoeME328_Ass2.cpp (with both team members' names). Each team member should upload both files to Assignment 2 on Canvas.

2. Teleoperation programming and data acquisition: Using a pair of Phantom Omnis, you will create several different types of teleoperators and analyze their performance. In all parts A through E, use identical controllers for all three Cartesian degrees of freedom of a particular robot (master or follower).

For each part of this problem, you will add your code to the main .cpp file of the code template linked next to the homework assignment at http://www.stanford.edu/class/me328/#assignments. For each part (A, B, etc.), you will add your code to the correct part in the switch statement in the function called "Teleoperation". You can also add variable definitions and any other necessary code anywhere between the comments that say "START EDITING HERE" and "STOP EDITING HERE". If you don't have experience with C programming, consult the link at the bottom of the course webpage.

A. Create a unilateral teleoperator. Begin by designing a controller that uses only a proportional gain (k_{ps}). Begin with a small gain and increase it (or decrease it if you've gone too far) until you get tracking behavior that seems reasonable. In your write-up, write out the control law you selected and explain why you selected the "best" value of the gain k_{ps}. Record data during teleoperation while you provide an approximately 5 cm, 2 Hz sinusoidal input in the x degree of freedom on the master. Provide a Matlab figure of the data with two plots (i.e., do a 2 x 1 subplot so that the two plots are stacked vertically): (1) the master and follower positions overlaid and (2) the error between the master and follower positions. You can scale the position and time axes to make the data easy to visualize. Be sure to provide axis labels and a legend in all plots where appropriate.

B. Improve your unilateral teleoperator. Add to your previous controller a derivative term with gain k_{ds}. Begin with a small gain and increase it (or decrease it if you've gone too far) until you get tracking behavior that seems better than what you had in part A. Then tweak both k_{ps} and k_{ds} in order to get the best tracking you can. In your write-up, write out the control law you selected. Record data during teleoperation while you provide an approximately 5 cm, 2 Hz sinusoidal input in the x degree of freedom on the master. Provide a Matlab figure of the data with two plots (i.e., do a 2 x 1 subplot): (1) the master and follower x positions overlaid – be sure to provide a legend – and (2) the error between the master and follower positions. Provide a short discussion comparing tracking behavior in part B to part A.

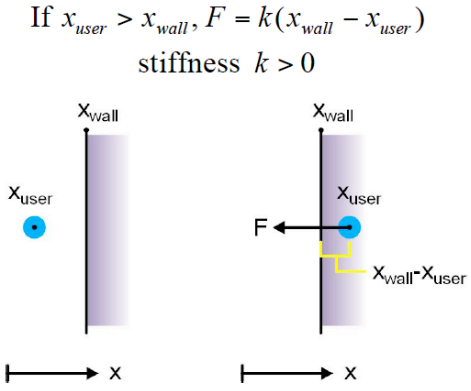
C. Motion scaling. Starting with your unilateral teleoperator from part B, create a motion scaling teleoperator in which the follower robot moves a noticeably smaller amount than the master robot, but you still have smooth and visible motion of the follower robot. In your write-up, write out the control law you selected. Record data during teleoperation while you provide an approximately 5 cm, 2 Hz sinusoidal input in the x degree of freedom on the master. Provide a Matlab figure of the master and follower x positions overlaid – be sure to provide a legend. Describe how this figure verifies that you have created the desired motion scaling.

D. Create a bilateral teleoperator using position feedback, in which the master and follower robots use identical (but reflected) proportional-derivative controllers. (Do not use any motion scaling.) Feel the controller from both the perspective of the master and the follower – they should feel identical. In your write-up, write out the control laws you selected. Record data during teleoperation and perform some movement/contact with an environment that demonstrates the force feedback. Provide a Matlab figure of the data with two plots (i.e., do a 2 x 1 subplot): (1) the master and follower x positions overlaid, and (2) the master and follower x-direction actuator forces overlaid. Explain what movement/contact you did and how the plot verifies that this controller is working properly.

E. Create a bilateral teleoperator using position feedback, in which the human operator receives *amplified* force feedback. (Begin with the controller you designed in part D.) Feel the controller from both the perspective of the master and the follower – they should no longer feel identical. In your write-up, write out the control laws you selected. Record data during teleoperation and perform some movement/contact with an environment that demonstrates the force feedback. Provide a Matlab figure of the data with two plots (i.e., do a 2 x 1 subplot): (1) the master and follower x positions overlaid, and (2) the master and follower x-direction actuator forces overlaid. Explain what movement/contact you did and how the plot verifies that this controller is working properly.

F. Develop your own interesting teleoperation controller, different from that in parts A through E. For example, you might try integral control, changing motion or force scaling for different parts of the workspace, or performing nonlinear motion/force scaling. Explain your algorithm in your write-up. Record data and provide plots that demonstrate that the controller is working as expected. Then say whether you think it would be appropriate/useful for teleoperated robot-assisted surgery and explain your reasoning.

G. Create a virtual box that attempts to prevent the master from moving outside of a 6 cm square area near the center of the workspace during teleoperation. To do this, create a set of 6 planes (walls) that are represented by linear springs. That is, they push back on the master with a linear stiffness ($f = kx$) when the master attempts to move through them. In other words, add an *additional* force to your f_{as} from part A that is proportional to master penetration into the virtual plane(s). You'll need to check whether there is penetration (i.e., is pos_1[0] greater than the position of the right side wall?) and add the additional spring force only when this is the case. See the figure at right for an example of rendering one wall. Explain your algorithm in your write-up. Explain how such a virtual box could be useful for a teleoperated surgical system.



3. Admittance control. In this problem, you will create a form of admittance control for a cooperatively manipulated Phantom Omni and apply virtual fixtures.

Note: For this problem, you only need to use one Omni. (Please do not move or unplug the Omnis, though, since resetting them requires administrator privileges on the computer.) We start with part H below so that you can include your code for this problem in the same program as the teleoperation problem.

H. Implement uniform, highly damped admittance control on all three degrees of freedom of the Phantom Omni. Because the Omni does not have a force sensor, you will use a "virtual" force sensor based on the difference between the previous position and actual position: $F_{measured} = -k_p(x_{previous} - x_{measured})$. Your admittance controller should effectively implement a high damping that (notwithstanding some freedom due to the virtual force sensor[1]) does not allow the user to make fast motions while moving in a single direction. Explain your algorithm and describe how you selected the relevant gains of the system in your write-up. Record data that demonstrates the effectiveness and limitations of your controller for movements in the x-direction only, and submit the plots and an explanation.

Hints: I suggest you start with admittance gain $k_a = 0$ to tune your virtual force sensor gain/controller. Then try different values of k_a. You can use the trapezoidal rule to integrate desired velocity from your admittance control law to obtain desired position for your robot control law. Since it is hard to match the initial conditions of the integrator to the actual position of the Omni, you should hold on tightly to the stylus when you start up. The initially large force will go away quickly. Additional Hint: To demonstrate your controller, you will want to show, along with other variables, the desired *position of the end-effector, which should exhibit more steady behavior than the* actual *position.*

I. In part G, you created a virtual box to maintain the teleoperated robot in a particular area of the workspace. Now, implement a similar virtual box in the framework of admittance control. Explain your algorithm in your write-up (no plots needed). How does this feel different from the impedance-type virtual box you implemented in the teleoperator setup, and why?

[1] While this would not make for a very accurate cooperative robot, if you were using the desired position as a command for a teleoperated patient-side robot, the patient-side robot would move very smoothly.