Homework 6. ME 7752
Simple control of the two-link robot

All usual homework guidelines apply. Total = 20 points.

Note: Because this is the last week and I do not want to burden you with too much work, try to do and submit any 3-4 problems below, but read and think through other problems as well. Due last day of classes.

Q0) Make up a model of 'some' system in `simmechanics` and simulate it. This system should be something we did not do in class or I did not post on carmen in my list of examples.

Position regulation.

For everything below: Consider the two-link manipular we derived the equations of motion for in class. We wish to introduce "controllers" so that the robot does various things listed below. Use same parameters: $L_1 = L_2 = 1$, $m_1 = m_2 = 1$, $g = 10$, $I_1 = I_2 = 1/12$, etc.

Q1) Introduce torque motors with a simple proportional-derivative feedback on the joint angles (no integral terms, and constant proportional gain), to stabilize the robot about angles $\theta_1 = 0$ and $\theta_2 = \pi/2$. List all the things that is wrong with such a controller. How big do you (roughly) need to make the feedback gains so that the steady state error is within 5 degrees of the desired angles? Show a plot of angles versus time when $\theta_1(0) = -\pi/2$ and $\theta_2(0) = 0$. Show a plot of the two joint torques during this approach to steady state.

Q2) Introduce an integral component to the above controller (so make it a PID controller). Tune the three feedback gains by hand so that the steady state error does go pretty small in some reasonable amount of time of your choosing. Plot the angles and the torques starting from the same initial conditions as Q1. Are the torques smaller or larger than in Q1?

Tracking a path, now with dynamics.

Q3) Now we want to make the endpoint P2 track circle of radius 1 in exactly one second. Use closed-form inverse kinematics expressions from our first 2-3 lectures to find $\theta_1(t)$ and $\theta_2(t)$, so that the endpoint moves in a circle. Make an animation of this motion. This is purely a "kinematic" animation.

Q4) Now add a simple PID controller with constant feedback gains of the form :

$$\tau_i = -k_p \left(\theta_i - \theta_{i-desired}(t)\right) - k_v \left(\dot{\theta}_i - \dot{\theta}_{i-desired}(t)\right) - k_i \int_0^t \left(\theta_i - \theta_{i-desired}(t')\right) \, dt'$$

where the desired angle for each joint is from Q3, so that the control torque tries to correct deviations from going in a circle.

Start from the configuration in which the end-point is at $(0, 0)$ with $\theta_1(0) = 0$ and $\theta_2(0) = \pi$. Show how the end-point seem to approach the desired circular motion asymptotically (picking the feedback gains appropriately). And the angles approach the desired angles from Q3. Make sure you plot desired and actual angles as functions of time, to see how different they are. Can you reduce the gap by changing gains?

Q5) Now to the above trajectory-following simulation, in addition to the feedback terms, add an "extra noisy" joint torque equal to $\tau_{1extra}(t) = 1\sin(3t) + 2\cos(4t)$ and $\tau_{2extra} = 0.01\cos(3t) - 0.02\sin(2t)$. Start from $(0, 0)$ again. What is the motion now? Because of the extra torques the motion will be different from circular in steady state. Draw plots of desired and actual $\theta_1(t)$ and $\theta_(t)$ to show how good the tracking is.

If the tracking is reasonable, try to increase the extra torques, so as to make the motion more crazy.

Inverse dynamics.

Q6) Find the torques required at the joints exactly as a function of time, that are exactly consistent with the desired circular motion from Q3. Using whatever methods. Start at the point $(1, 0)$ at t = 0.

Bonus, just to think about.

Q7) Try using the joint torques as functions of time from Q7 to drive the system from some arbitrary initial condition, without any feedback. Does the motion approach the circular motion?

(Very likely not! It will be close to the circle if you start close to the circle, but may not be if you start far away.) This is feed-forward or open-loop (inverse-dynamics-based) control.

Q8) Add a PID term to the above torques from Q7 to see if the performance improves from Q8.

Q9) Do you remember the old 2D tracking program in which we used MATLAB to track a smooth curve connecting a bunch of point, using splines and so on? How would you design a path-following controller like above for general user-input curves like that? Explain the method in words in detail, but you do not have to implement it.

Q10) To reduce the gap between desired and actual from Q4-Q5, you can sometimes use both a single integral term and a double integral term, etc.