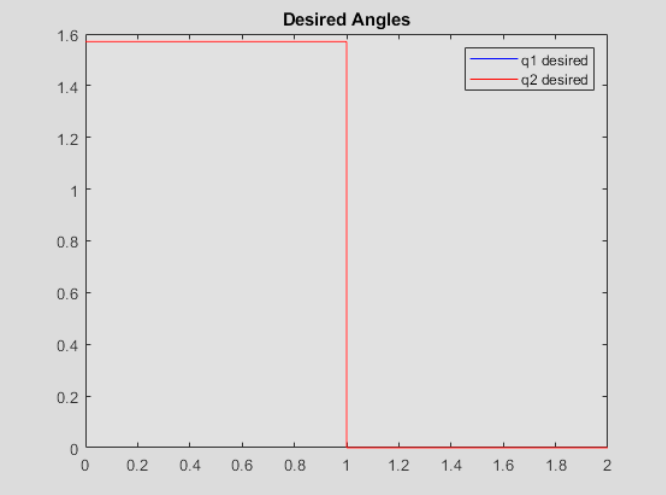
Ryan Shaw

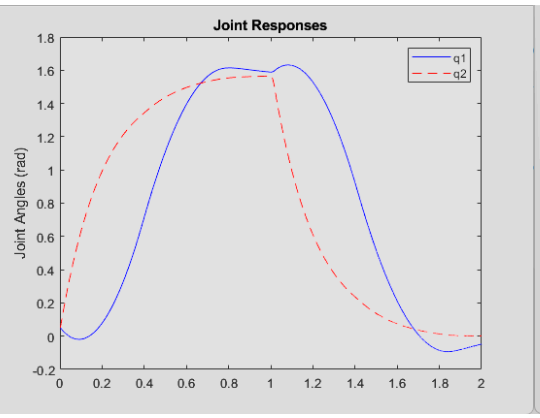
HW 6

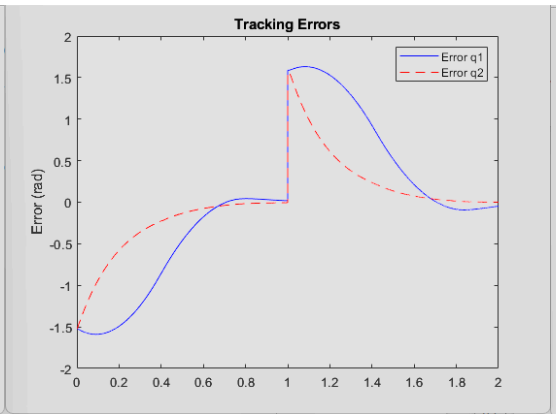
1.

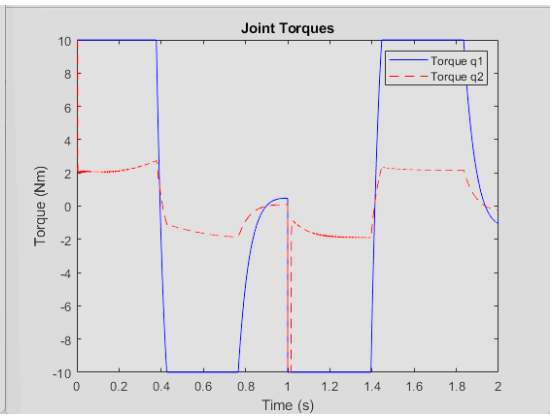
a.

(a) Since one can never guarantee that the initial conditions will be exactly at zero, simulate the response again, this time using the initial conditions, $q_1(0) = 0.05 = q_2(0)$ which corresponds to about 3-degrees. Generate, label, and print out plots of the joint responses, input torques, and tracking errors for each link.









You will notice how when the error is large the torque spikes to saturation. We also notice how the tracking error spikes at t=1sec due to our step input changing.

b.

(b) What is the error in position at t = 1 second? at t = 2 seconds? Did the torque inputs saturate? for how long? Explain the large initial tracking errors and the large initial torques.

T = 1 second:

- q1 error = 0.010
- q2 error = -0.006

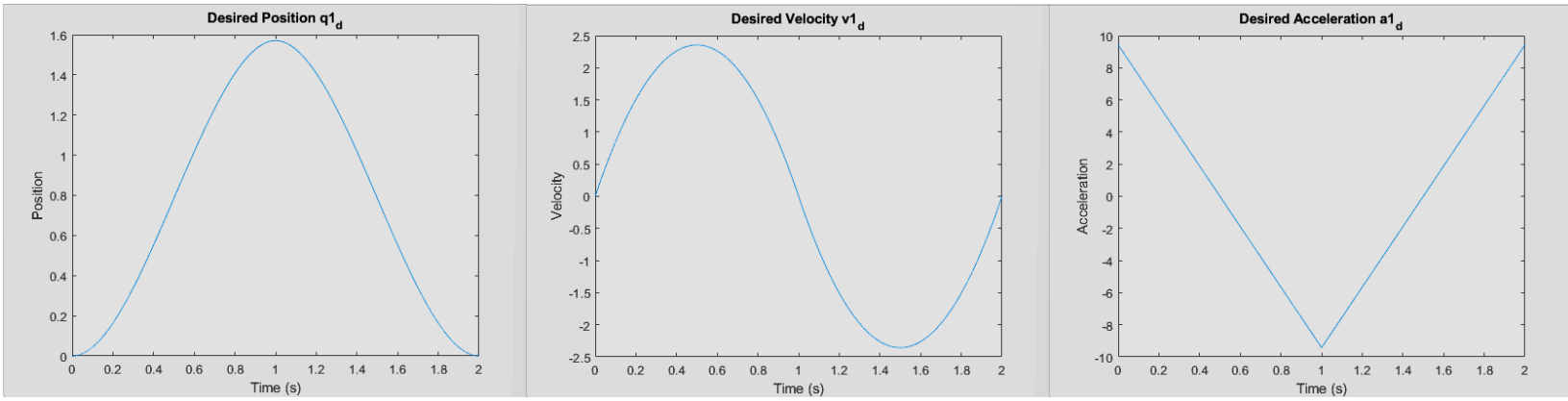
T = 2 seconds:

- q1 error = -0.054
- q2 error = 0.002

Torque q1 saturated 4 times for approximately 0.4s each time, q2 for a fraction of a second when our desired inputs changed. The large initial tracking errors and torques are due to our initial conditions being far (pi/2) from our desired position.

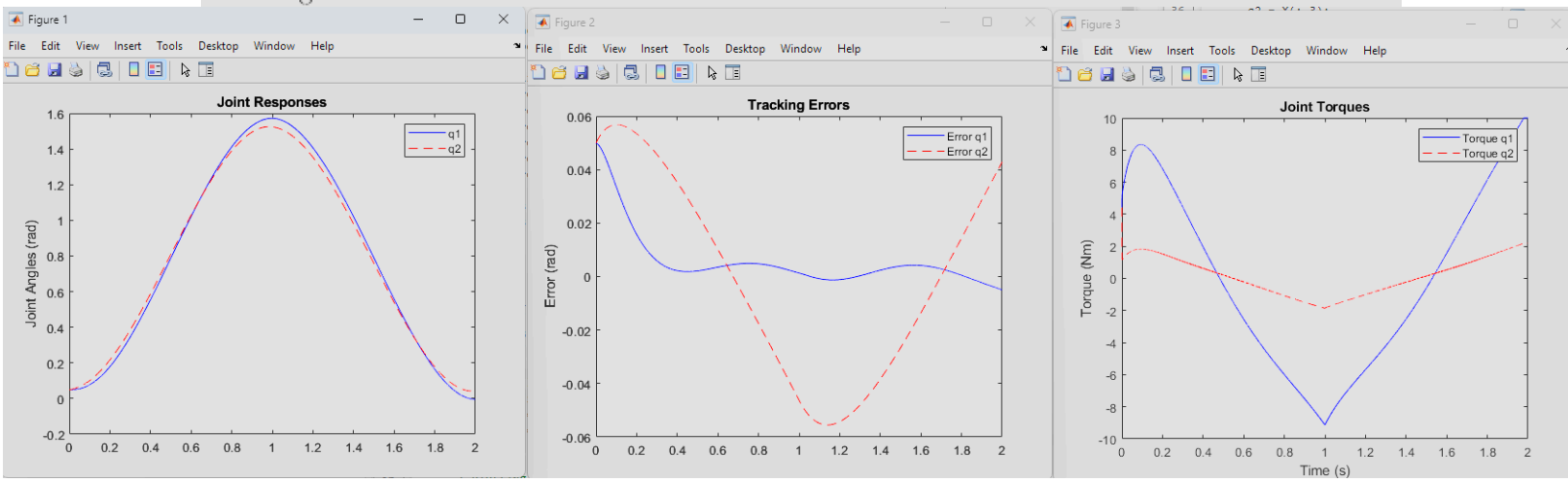
2.

Generated reference trajectory:



a.

(a) Since one can never guarantee that the initial conditions will be exactly at zero, simulate the response again, this time using the initial conditions, $q_1(0) = 0.05 = q_2(0)$ which corresponds to about 3-degrees. Generate, label, and print out plots of the joint responses, input torques, and tracking errors for each link.



Here we will notice the joint movement is much smoother than part one, with stable errors that oscillate slightly around zero. You'll also notice that the joint responses and torques greatly follow the trend of the generated trajectory.

b.

(b) What is the error in each joint angle at t = 1 second? at t = 2 second? How do the joint tracking errors and input torques compare to the pure PD control of Problem 1?

T = 1 second:

- q1 error = 0
- q2 error = -0.047

T = 2 seconds:

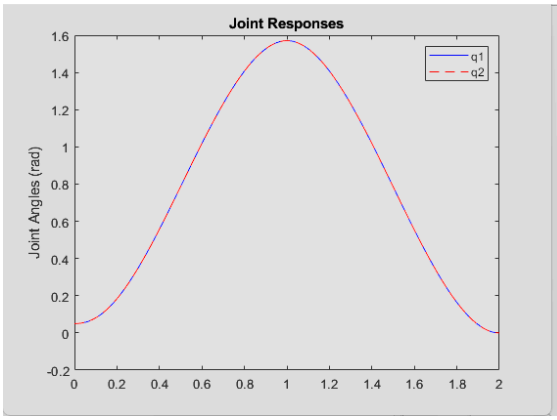
- q1 error = -0.005
- q2 error = 0.043

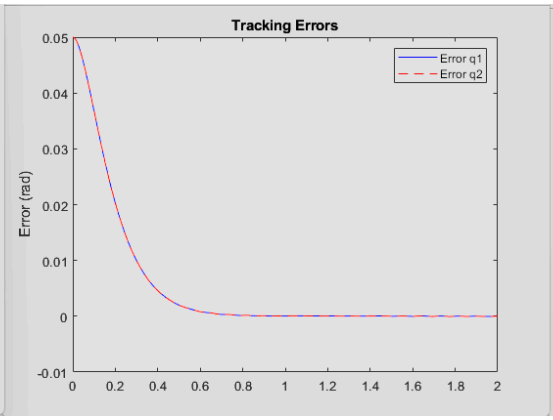
The joint torques are noticeably smoother than in problem 1; we never see saturation. There is also significantly less tracking error compared to part 1. This can be attributed to the benefit of feedforward control from the generated cubic trajectory.

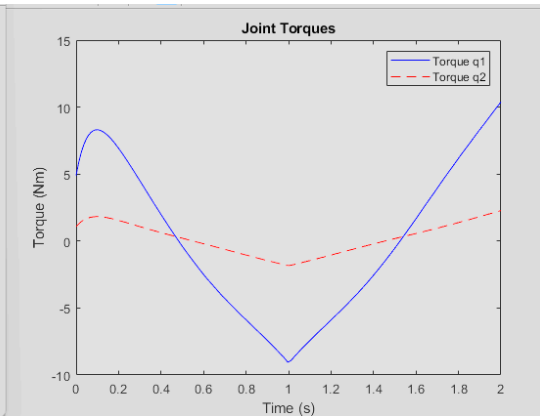
3.

a.

(a) Simulate the response for 2 seconds using the same nonzero initial conditions as in Problem 2 and generate plots of the joint responses, input torques, and tracking errors.







Here we see the joint responses follow our trajectory perfectly, with error converging quickly to zero.

b.

(b) What is the error after t = 1 second? after t = 2 second? How do the joint torques and tracking errors compare with those of the previous two cases?

There is essentially no tracking error. As mentioned in the problem description, the minuscule errors observed are an artifact of the simulation. The joint torques appear to be smooth much like the case of feedforward control, especially compared to normal PD control. However, since we utilize inverse dynamics, we ultimately see much better error tracking.

4.

Compare the performance of PD Control, PD + Feed Forward Control, and Inverse Dynamics Control based on the results obtained in the preceding questions. Analyze their respective strengths and weaknesses. Please limit your response to one paragraph.

PD control is great for its simplicity, though it struggles to provide smoothness in operation, posing potential dangers if there are people in its vicinity. It also has a tendency to saturate motor torques (though this can be dependent on saturation limits and transient specs) which may not be ideal for the motors themselves. PD with feedforward, on the other hand, is much better at providing smooth trajectories with predictable behavior and even further improved error tracking. This may also prove to be more energy efficient. However, it is important to consider that feedforward control in general is much more difficult to achieve properly, and requires more information. It may not always be possible to generate exact trajectories. Lastly, ID control has many of the same benefits and drawbacks as PD with feed forward control. However, IDC has the advantage of controlling our dynamic terms, allowing us to define our robot dynamics and in turn have perfect error tracking. On the other hand, this comes with the assumption that we know our model parameters definitively, but this may not always be the case. When we do not know these parameters, adaptive IDC would help us deal with uncertainties.