Problem Set 7

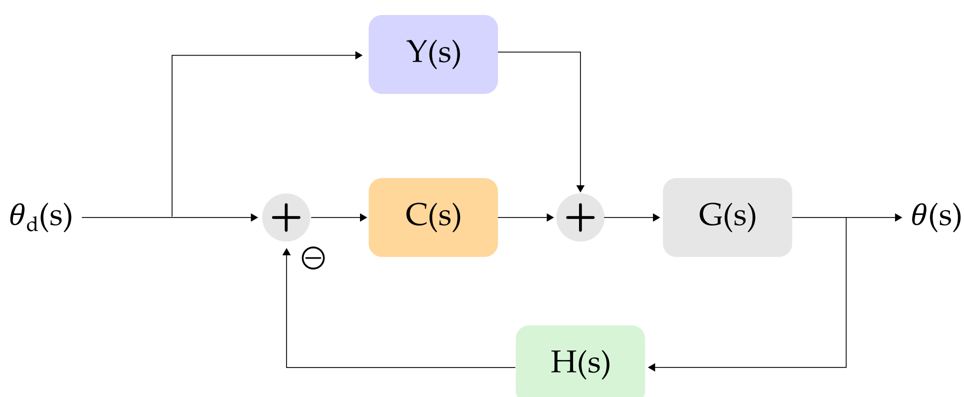
Robotics & Automation
Dylan Losey, Virginia Tech

Instructions. Please write legibly and do not attempt to fit your work into the smallest space possible. It is important to show all work, but basic arithmetic can be omitted. You are encouraged to use Matlab when possible to avoid hand calculations, but print and submit your commented code for non-trivial calculations. You can attach a pdf of your code to the homework, use live scripts or the publish feature in Matlab, or include a snapshot of your code. Do not submit .m files — we will not open or grade these files.

For this assignment we are asking you to also submit **videos** of your simulations. Follow the instructions to **label** these videos based on the problem number, and then submit them all within a **single zipped folder**.

1 Open-Loop and Closed-Loop Control

1.1 (10 points)



Find the closed-loop transfer function:

$$\frac{\theta(s)}{\theta_d(s)} \tag{1}$$

for the block diagram shown above. Here $Y(s)$ is a feed forward controller and $C(s)$ is a feedback controller.

Leave out the (s) for simplicity. From the block diagram:

$$\theta = \big(Y\theta_d + C(\theta_d - H\theta)\big)G \tag{2}$$
$$= YG\theta_d + CG\theta_d - CGH\theta \tag{3}$$
$$= -CGH\theta + (YG + CG)\theta_d \tag{4}$$

Moving $CGH\theta$ to the other side and factoring:

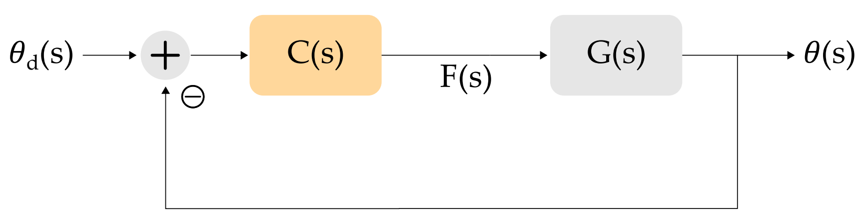
$$(1 + CGH)\theta = (YG + CG)\theta_d \tag{5}$$

Now solve for the closed-loop transfer function:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{Y(s)G(s) + C(s)G(s)}{1 + C(s)G(s)H(s)} \tag{6}$$

2 Stability

Imagine that you purchased three separate 1-DoF robots. Each system comes with its own links, joint, motor, and motor controller, and some of the systems are acting in strange ways. Your job is to design controllers that will result in closed-loop stability.



Throughout this problem assume that you can directly measure θ. Use the block diagram above as a reference when determining closed-loop stability.

2.1 (5 points)

The first 1-DoF robot is a mass-damper:

$$f(t) = m\ddot{\theta} - b\dot{\theta} \tag{7}$$

Design a controller that results in closed-loop stability.

Start by finding the plant $G(s)$:

$$G(s) = \frac{\theta(s)}{F(s)} = \frac{1}{ms^2 - bs} \tag{8}$$

Write the closed-loop transfer function using the block diagram:

$$\theta(s) = C(s)G(s)\Big(\theta_d(s) - \theta(s)\Big) \tag{9}$$

$$\frac{\theta(s)}{\theta_d(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} \tag{10}$$

Get the characteristic equation and plug in the plant $G(s)$:

$$1 + C(s)G(s) = 0, \qquad ms^2 - bs + C(s) = 0 \tag{11}$$

The system is stable if all poles of characteristic equation have negative real values. Using the Routh–Hurwitz criterion, we need $C(s)$ so that the characteristic equation is of the form $s^2 + a_1 s + a_0 = 0$, and both $a_1 > 0$ and $a_0 > 0$. Try PD control:

$$C(s) = k_d s + k_p \tag{12}$$

$$ms^2 + (k_d - b)s + k_p = 0 \tag{13}$$

$$s^2 + \frac{(k_d - b)}{m}s + \frac{k_p}{m} = 0 \tag{14}$$

The PD controller is stable for any choice of $k_d > b$ and $k_p > 0$.

2.2 (10 points)

The second 1-DoF robot has plant dynamics:

$$G(s) = \frac{1}{s(s-3)(s-5)} \tag{15}$$

Design a controller that results in closed-loop stability.

Get the characteristic equation and plug in the plant $G(s)$:

$$1 + C(s)G(s) = 0, \qquad s(s-3)(s-5) + C(s) = 0 \tag{16}$$

$$s^3 - 8s^2 + 15s + C(s) = 0 \tag{17}$$

The system is stable if all poles of characteristic equation have negative real values. Using the Routh–Hurwitz criterion, we need $C(s)$ so that the characteristic equation is of the form $s^3 + a_2 s^2 + a_1 s + a_0 = 0$, where $a_2 > 0$, $a_1 > 0$, and $a_0 > 0$, and $a_2 a_1 > a_0$. Let's design a custom controller:

$$C(s) = k_a s^2 + k_p \tag{18}$$

where k_a is the gain on acceleration error. Plugging in:

$$s^3 + (k_a - 8)s^2 + 15s + k_p = 0 \tag{19}$$

Our custom controller is stable when $k_a > 8$ and $15(k_a - 8) > k_p > 0$. For example, $k_a = 10$ and $k_p = 1$. **Aside –** Our choice of $C(s)$ will be tricky to implement in practice since it is challenging to get an accurate measure of acceleration. This is because we often only have sensors for position, and must take derivatives to get velocity and acceleration.

2.3 (5 points)

The third 1-DoF robot is a mass-spring-damper:

$$f(t) = 5\ddot{\theta} + 2\dot{\theta} - 15\theta \tag{20}$$

Design a controller that places both poles at $s = -5$. **Aside –** When both poles of a mass-spring-damper are equal negative real numbers, the system is *critically damped*.

Start by finding the plant $G(s)$:

$$G(s) = \frac{\theta(s)}{F(s)} = \frac{1}{5s^2 + 2s - 15} \tag{21}$$

Then substitute this plant $G(s)$ into the characteristic equation:

$$1 + C(s)G(s) = 0 \tag{22}$$

$$5s^2 + 2s - 15 + C(s) = 0 \tag{23}$$

We want to place both poles at $s = -5$. This means the characteristic equation should be: $(s+5)(s+5) = s^2 + 10s + 25 = 0$. We achieve this using a PD controller:

$$C(s) = k_d s + k_p \tag{24}$$

$$5s^2 + (k_d + 2)s + (k_p - 15) = 0 \tag{25}$$

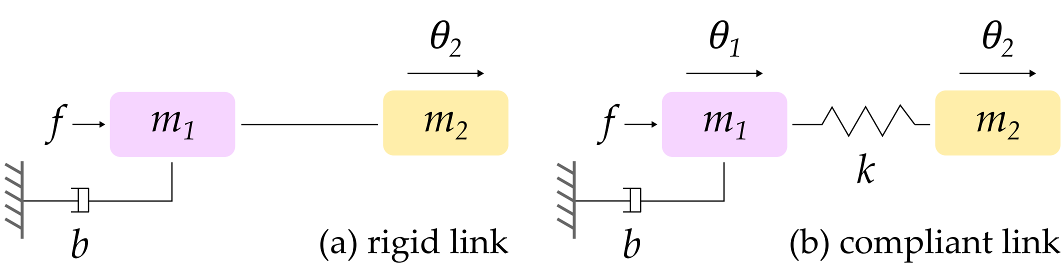
$$s^2 + \frac{k_d + 2}{5}s + \frac{k_p - 15}{5} = 0 \tag{26}$$

Solve for k_d and k_p to place both poles at $s = -5$:

$$\frac{k_d + 2}{5} = 10, \qquad \frac{k_p - 15}{5} = 25 \tag{27}$$

Using this we reach our critically-damped system: a PD controller with gains $k_d = 48$ and $k_p = 140$ places both closed-loop poles at $s = -5$.

3 Compliant Joints



(a) rigid link (b) compliant link

We often introduce mechanical compliance to make the robot soft and safe during interaction. The drawing above compares a rigid link and (Left) and a compliant link (Right). Here m_1 is the motor mass and m_2 is the link mass. For rigid systems we have a rigid connection between motor output and the link, and the total mass is $m_1 + m_2$. For compliant systems we introduce a spring k between the motor output and the link. In both systems we control the actuator force f to regulate θ_2, the position of the link.

3.1 (5 points)

Find the plant dynamics $G_1(s)$ and $G_2(s)$. Here $G_1(s)$ is the plant for the rigid 1-DoF robot and $G_2(s)$ is the plant for the compliant 1-DoF robot. Both plants should be of the form:

$$G(s) = \frac{\theta_2(s)}{F(s)} \tag{28}$$

Rigid Robot. Start by finding the equation of motion for mass $m_1 + m_2$, then write that equation in the Laplace domain:

$$F(s) = \Big((m_1 + m_2)s^2 + bs\Big)\theta_2(s) \tag{29}$$

Solve for the plant $G_1(s)$:

$$G_1(s) = \frac{\theta_2(s)}{F(s)} = \frac{1}{(m_1 + m_2)s^2 + bs} \tag{30}$$

Compliant Robot. Start by finding the equations of motion for each separate mass, then write those equations in the Laplace domain:

$$F(s) + k\Big(\theta_2(s) - \theta_1(s)\Big) = m_1 s^2\theta_1(s) + bs\theta_1(s) \tag{31}$$

$$-k\Big(\theta_2(s) - \theta_1(s)\Big) = m_2 s^2\theta_2(s) \tag{32}$$

Manipulate the equations to isolate the mass positions:

$$\theta_1(s) = \frac{F(s) + k\theta_2(s)}{m_1 s^2 + bs + k} \tag{33}$$

$$\theta_2(s) = \frac{k}{m_2 s^2 + k}\theta_1(s) \tag{34}$$

Then combine these equations to get $G_2(s)$:

$$\theta_2(s) = \frac{kF(s) + k^2\theta_2(s)}{(m_2 s^2 + k)(m_1 s^2 + bs + k)} \tag{35}$$

$$G_2(s) = \frac{k}{(m_2 s^2 + k)(m_1 s^2 + bs + k) - k^2} \tag{36}$$

$$G_2(s) = \frac{k}{m_1 m_2 s^4 + m_2 b s^3 + (m_1 + m_2)ks^2 + bks} \tag{37}$$

3.2 (10 points)

Assume we measure θ_2 in real-time and the closed-loop transfer function is:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} \tag{38}$$

We will use a proportional controller $C(s) = k_p$. Let $m_1 = 1$ kg, $m_2 = 2$ kg, $b = 0.1$ Ns/m, and $k = 100$ N/m.

- For what range of k_p is the **rigid** robot stable?
- For what range of k_p is the **compliant** robot stable?

Rigid Robot. Plugging the plant and controller into the closed-loop dynamics, we reach the characteristic equation:

$$(m_1 + m_2)s^2 + bs + k_p = 0 \tag{39}$$

Using the given values for m_1, m_2, and b:

$$3s^2 + 0.1s + k_p = 0 \tag{40}$$

This rigid system is stable for any choice of $k_p > 0$.

Compliant Robot. Start by finding the characteristic equation:

$$m_1 m_2 s^4 + m_2 b s^3 + (m_1 + m_2)ks^2 + bks + kk_p = 0 \tag{41}$$

Plug in the given parameters:

$$2s^4 + 0.2s^3 + 300s^2 + 10s + 100k_p = 0 \tag{42}$$

A fourth-order polynomial $s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0$ has negative real poles if $a_i > 0$ and $a_3 a_2 a_1 > a_1^2 + a_3^2 a_0$. To apply this Routh-Hurwitz criterion it may help to divide the characteristic equation by 2:

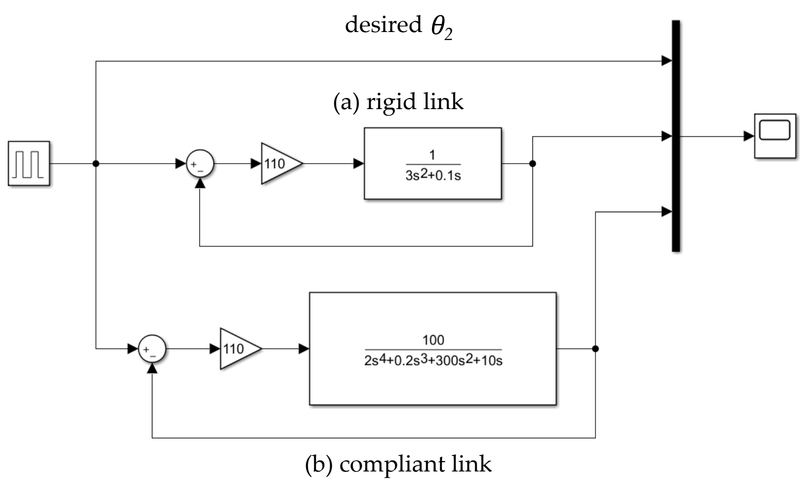
$$s^4 + 0.1s^3 + 150s^2 + 5s + 50k_p = 0 \tag{43}$$

We reach an interesting requirement for the controller. Here the compliant system is stable if and only if $k_p > 0$ and $k_p < 100$.

3.3 (10 points)

Simulate the rigid system and compliant system in Simulink (Matlab). For both systems set θ_d as a `Pulse Generator` with amplitude 1 m, period 5 s, and pulse width 50%. The model stop time should be 50 s.

Turn in **two separate plots**. One plot should show θ_d, θ_2 (for the rigid system) and θ_2 (for the compliant system) when $k_p = 10$. The other plot should show θ_d, θ_2 (for the rigid system) and θ_2 (for the compliant system) when $k_p = 110$. Include labels and captions.



Plots of the simulation are shown below. The Simulink model used to create these plots is shown above.

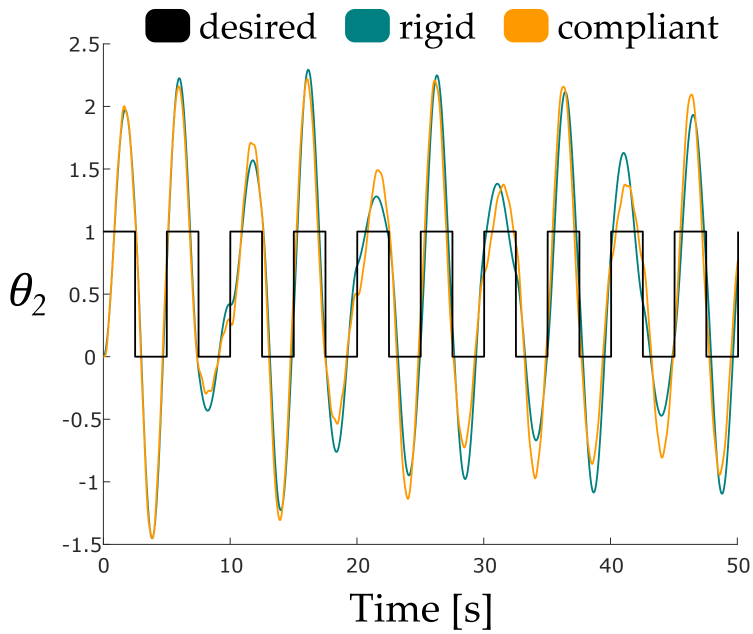


Figure 1: Closed-loop behavior of both systems when $k_p = 10$. Here both systems behave similarly: there is no clear difference between rigid and compliant.

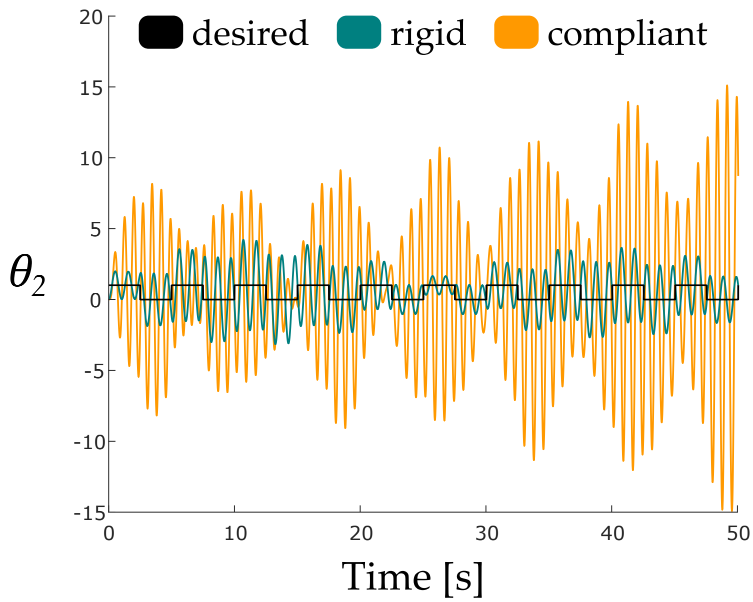


Figure 2: Closed-loop behavior of both systems when $k_p = 110$. Our analysis determined that the compliant system is unstable when $k_p > 100$, while the rigid system is stable for any $k_p > 0$. This plot supports our analysis and shows an unstable compliant system.

3.4 (5 points)

Does introducing compliance (e.g., the spring between the motor and link) make it *easier* or *harder* to control the robot? Write a few sentences to explain your answer.

Compliance makes it **harder** to control the system. Our analysis shows that the added compliance restricts the range of k_p for which the system is stable. Physically, the added spring causes the second mass to oscillate, and the controllers we design must work to mitigate these oscillations.

4 Lyapunov Stability Analysis

4.1 (20 points)

In this problem we will introduce *passivity-based motion control*. This approach is a basis for many robust and adaptive robot controllers. Consider a robot arm with dynamics:

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + g(\theta) \tag{44}$$

In passivity-based control we select the controller:

$$\tau = M(\theta)y + C(\theta, \dot{\theta})x + g(\theta) - Kr \tag{45}$$

$$y = \ddot{\theta}_d - \Lambda(\dot{\theta} - \dot{\theta}_d), \quad x = \dot{\theta}_d - \Lambda(\theta - \theta_d), \quad r = (\dot{\theta} - \dot{\theta}_d) + \Lambda(\theta - \theta_d) \tag{46}$$

Here K and Λ are constant positive definite matrices. Apply Lyapunov's method to show that the closed-loop system is stable. The equilibrium should be $\dot{\theta} = \dot{\theta}_d$, $\theta = \theta_d$.

Hint 1. Let M be the mass matrix and let C be the Coriolis matrix. The matrix $\dot{M} - 2C$ is skew symmetric. For any vector a, this means $a^T(\dot{M} - 2C)a = 0$.

Hint 2. Try the following generalized energy function:

$$v = \frac{1}{2}r^T M(\theta)r + (\theta - \theta_d)^T \Lambda K(\theta - \theta_d) \tag{47}$$

Start by plugging the controller in the dynamics. We obtain:

$$0 = M(\theta)(\ddot{\theta} - y) + C(\theta, \dot{\theta})(\dot{\theta} - x) + Kr \tag{48}$$

Looking at the definitions for y, x, and r, the closed-loop dynamics become:

$$0 = M(\theta)\dot{r} + C(\theta, \dot{\theta})r + Kr \tag{49}$$

Now take the time dervative of v to see how energy changes of time:

$$\dot{v} = r^T M\dot{r} + \frac{1}{2}r^T \dot{M}r + 2(\theta - \theta_d)^T \Lambda K(\dot{\theta} - \dot{\theta}_d) \tag{50}$$

We are ready to plug in the closed-loop dynamics for $M\dot{r}$. We reach:

$$\dot{v} = r^T\Big(- C(\theta, \dot{\theta})r - Kr\Big) + \frac{1}{2}r^T\dot{M}r + 2(\theta - \theta_d)^T\Lambda K(\dot{\theta} - \dot{\theta}_d) \tag{51}$$

Rearrange and leverage the skew symmetric property for $\dot{M} - 2C$:

$$\dot{v} = r^T\left(\frac{1}{2}\Big(\dot{M} - 2C(\theta, \dot{\theta})\Big)r - Kr\right) + 2(\theta - \theta_d)^T\Lambda K(\dot{\theta} - \dot{\theta}_d) \tag{52}$$

$$\dot{v} = -r^T Kr + 2(\theta - \theta_d)^T\Lambda K(\dot{\theta} - \dot{\theta}_d) \tag{53}$$

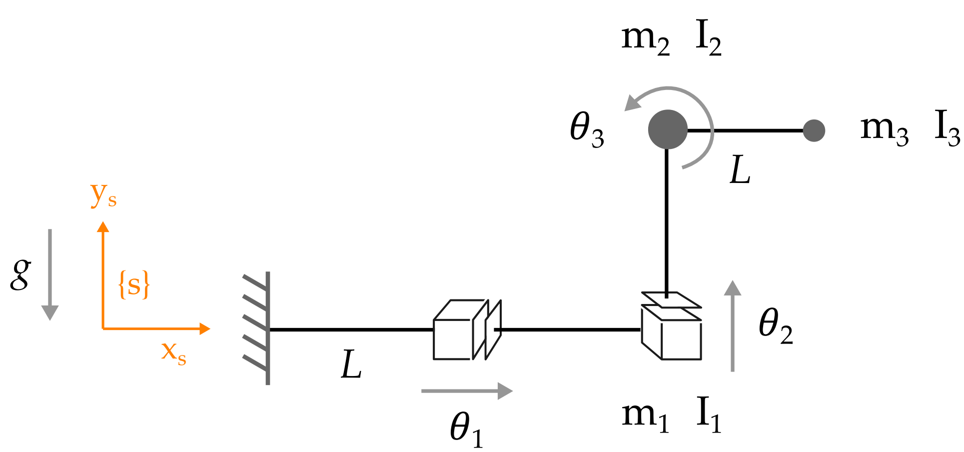
The final step is to expand $r^T Kr$ using the given definition for r. Note that the added term cancels out part of this expansion:

$$\dot{v} = -\Big((\dot{\theta} - \dot{\theta}_d) + \Lambda(\theta - \theta_d)\Big)^T K\Big((\dot{\theta} - \dot{\theta}_d) + \Lambda(\theta - \theta_d)\Big) + 2(\theta - \theta_d)^T\Lambda K(\dot{\theta} - \dot{\theta}_d) \tag{54}$$

$$\dot{v} = -(\dot{\theta} - \dot{\theta}_d)^T K(\dot{\theta} - \dot{\theta}_d) - (\theta - \theta_d)^T\Lambda^T K\Lambda(\theta - \theta_d) \tag{55}$$

Our result for $\dot{v}$ is always negative if either $\dot{\theta} \neq \dot{\theta}_d$ or $\theta \neq \theta_d$. Hence, energy leaves the system until we reach the equilibrium $\dot{\theta} = \dot{\theta}_d$, $\theta = \theta_d$ and the closed-loop robot is stable. If you reached this solution, congratulations! If you are stumped, also see Chapter 8.4 of Robot Modeling and Control.

5 Multivariable Control



In this problem you will simulate control the robot shown above. You have already obtained the dynamics of this robot in a previous assignment. The robot starts at joint position $\theta = 0$. Here $L = 1$, $m_1 = m_2 = m_3 = 1$, and $I_3 = 0.1$.

5.1 (20 points)

Download the Matlab file `make_controller.m` that was provided with this assignment. Use the given simulation parameters and frame rates; all videos should be 10 seconds in length. Modify the code as needed so that the robot reaches for θ_d using multivariable PD control with gravity compensation:

$$\tau = K_P(\theta_d - \theta) - K_D\dot{\theta} + g(\theta) \tag{56}$$

Turn in the following MP4 videos:

- Make a simulation where $K_P = I$ and $K_D = I$. Let the desired position be:

$$\theta_d = \begin{bmatrix} -2 \\ 2 \\ \pi/4 \end{bmatrix} \tag{57}$$

  Title this video **Problem2_1.mp4**

- Reach for the same θ_d as in the previous part, but now tune K_P and K_D to improve the robot's performance. Make a simulation with your best performing gains, and title this video **Problem2_2.mp4**

- Control the robot to move in a circle. Let t be the simulation time (variable `time` in the code), and define the desired trajectory as:

$$\theta_d = \begin{bmatrix} 2\cos(\frac{\pi t}{2}) \\ 2\sin(\frac{\pi t}{2}) \\ \pi/2 \end{bmatrix}, \qquad \dot{\theta}_d = \begin{bmatrix} -\pi\sin(\frac{\pi t}{2}) \\ \pi\cos(\frac{\pi t}{2}) \\ 0 \end{bmatrix} \tag{58}$$

  Update your PD controller to also include $\dot{\theta}_d$. *Write down* and submit the equation for your modified control law. Make a simulation with your best performing gains, and title this video **Problem2_3.mp4**

```
for idx = 1:1000

    % get desired position
    theta_d = [2*cos(0.5*pi*time); 2*sin(0.5*pi*time); pi/2];
    thetadot_d = [-pi*sin(0.5*pi*time); pi*cos(0.5*pi*time); 0];
    T_d = fk(M3, [S1 S2 S3], theta_d);

    % plot the robot
    p0 = [0; 0];
    p1 = p0 + [L + theta(1); 0];
    p2 = p1 + [0; theta(2)];
    p3 = p2 + L * [cos(theta(3)); sin(theta(3))];
    P = [p0, p1, p2, p3];
    cla;
```

```
    % choose your controller tau
    Kd = eye(3)*25;
    Kp = eye(3)*25;
    tau = Kp*(theta_d - theta) + Kd*(thetadot_d - thetadot) + G;

    % integrate to update velocity and position
    thetadotdot = M \ (tau - C*thetadot - G);
    thetadot = thetadot + deltaT * thetadotdot;
    theta = theta + deltaT * thetadot;
    time = time + deltaT;
```

See the figures above for the completed code. The videos for each case are uploaded in a separate solutions folder. To include the desired joint velocity, use:

$$\tau = K_P(\theta_d - \theta) + K_D(\dot{\theta}_d - \dot{\theta}) + g(\theta) \tag{59}$$