Pre-Lab

1. Closed-Loop Control of SIXT33N

Last time, we discovered that open-loop control was not enough to ensure that our car goes straight in the event of model mismatch. In this problem, we will introduce closed-loop control which will hopefully make SIXT33N finally go straight.

Previously, we introduced $\delta[t] = d_L[t] - d_R[t]$ as the difference in positions between the two wheels. If both wheels of the car are going at the same velocity, then this difference $\delta$ should remain constant, since no wheel will advance by more ticks than the other. In our closed loop control scheme, we will consider a control scheme which will apply a simple proportional control $k_L$ and $k_R$ against $\delta[t]$ in order to try to prevent $|\delta[t]|$ from growing without bound.

$$v_L[t] = d_L[t + 1] - d_L[t] = \theta_L u_L[t] - \beta_L$$
$$v_R[t] = d_R[t + 1] - d_R[t] = \theta_R u_R[t] - \beta_R$$

We want to achieve the following equations:

$$v_L[t] = d_L[t + 1] - d_L[t] = v^* - k_L \delta[t]$$
$$v_R[t] = d_R[t + 1] - d_R[t] = v^* + k_R \delta[t]$$

We can put the equations in the following form to figure out how we should change our control inputs.

$$v_L[t] = d_L[t + 1] - d_L[t] = \theta_L \left( \frac{v^* + \beta_L}{\theta_L} - k_L \frac{\delta[t]}{\theta_L} \right) - \beta_L$$
$$v_R[t] = d_R[t + 1] - d_R[t] = \theta_R \left( \frac{v^* + \beta_R}{\theta_R} + k_R \frac{\delta[t]}{\theta_R} \right) - \beta_R$$

These are our new closed-loop control inputs – the new closed-loop proportional control is the $k_L/k_R$ term.

$$u_L[t] = \frac{v^* + \beta_L}{\theta_L} - k_L \frac{\delta[t]}{\theta_L}$$
$$u_R[t] = \frac{v^* + \beta_R}{\theta_R} + k_R \frac{\delta[t]}{\theta_R}$$
(a) Let’s examine the feedback proportions $k_L$ and $k_R$ more closely. Should they be positive or negative? What do they mean? Think about how they interact with $\delta[t]$.  

**Solution:**

If $\delta[t] > 0$, it means that $d_L[t] > d_R[t]$, so the left wheel is ahead of the right one. In order to correct for this, we should help the right wheel catch up, and we should do this by making $k_L > 0$ in order to apply less power on the left wheel and $k_R > 0$ in order to apply more power to the right wheel.

Likewise, if $\delta[t] < 0$, it means that $d_L[t] < d_R[t]$, so the right wheel is ahead of the left one. In this case, $k_L > 0$ is still valid, since $k_L \delta[t] > 0$ and so the left wheel speeds up, and likewise $k_R > 0$ is still correct since $k_R \delta[t] < 0$ so the right wheel slows down.

(b) Let’s look a bit more closely at picking $k_L$ and $k_R$. First, we need to figure out what happens to $\delta[t]$ over time. Find $\delta[t + 1]$ in terms of $\delta[t]$.

**Solution:**

$$
\delta[t + 1] = d_L[t + 1] - d_R[t + 1]
= v^* - k_L \delta[t] + d_L[t] - (v^* + k_R \delta[t] + d_R[t])
= v^* - k_L \delta[t] + d_L[t] - v^* - k_R \delta[t] - d_R[t]
= -k_L \delta[t] - k_R \delta[t] + (d_L[t] - d_R[t])
= -k_L \delta[t] - k_R \delta[t] + \delta[t]
= \delta[t](1 - k_L - k_R)
$$

(c) Given your work above, what is the eigenvalue of the system defined by $\delta[t]$? For discrete-time systems like our system, $\lambda \in (-1,1)$ is considered stable. Are $\lambda \in [0,1)$ and $\lambda \in (-1,0]$ identical in function for our system? Which one is “better”? *(Hint: Preventing oscillation is a desired benefit.)*

Based on your choice for the range of $\lambda$ above, how should we set $k_L$ and $k_R$ in the end?

**Solution:**

The eigenvalue is $\lambda = 1 - k_L - k_R$.

As a discrete system, both are stable, but $\lambda \in (-1,0]$ will cause the car to oscillate due to overly high gain. Therefore, we should choose $\lambda \in [0,1)$.  

As a result, $1 - k_L - k_R \in [0,1) \implies (k_L + k_R) \in (0,1)$ means that we should set the gains such that $(k_L + k_R) \in (0,1)$.

(d) Let’s re-introduce the model mismatch from last week in order to model environmental discrepancies, disturbances, etc. How does closed-loop control fare under model mismatch? Find $\delta_{ss} = \delta[t \to \infty]$, assuming that $\delta[0] = \delta_0$. What is $\delta_{ss}$? (To make this easier, you may leave your answer in terms of appropriately defined $c$ and $\lambda$ obtained from an equation in the form of $\delta[t + 1] = \delta[t] \lambda + c$.)  

Check your work by verifying that you reproduce the equation in part (c) if all model mismatch terms are zero. Is it better than the open-loop model mismatch case from last week?

$$
v_L[t] = d_L[t + 1] - d_L[t] = (\theta_L + \Delta \theta_L) u_L[t] - (\beta_L + \Delta \beta_L) \\
v_R[t] = d_R[t + 1] - d_R[t] = (\theta_R + \Delta \theta_R) u_R[t] - (\beta_R + \Delta \beta_R)
$$
\[ u_L[t] = \frac{v^r + \beta_L}{\theta_L} - k_L \frac{\delta[t]}{\theta_L} \]
\[ u_R[t] = \frac{v^r + \beta_R}{\theta_R} + k_R \frac{\delta[t]}{\theta_R} \]

Solution:

\( \delta[t + 1] = d_L[t + 1] - d_R[t + 1] \)
\( = (\theta_L + \Delta \theta_L)u_L[t] - (\beta_L + \Delta \beta_L) + d_L[t] - ((\theta_R + \Delta \theta_R)u_R[t] - (\beta_R + \Delta \beta_R) + d_R[t]) \)
\( = \theta_L u_L[t] - \beta_L + \Delta \theta_L u_L[t] - \Delta \beta_L + d_L[t] - \theta_R u_R[t] - \beta_R + \Delta \theta_R u_R[t] - \Delta \beta_R + d_R[t] \)
\( = v^r - k_L \delta[t] + \Delta \theta_L u_L[t] - \Delta \beta_L + d_L[t] - (v^r + k_R \delta[t] + \Delta \theta_R u_R[t] - \Delta \beta_R + d_R[t]) \)
\( = v^r - k_L \delta[t] + \Delta \theta_L u_L[t] - \Delta \beta_L + d_L[t] - v^r - k_R \delta[t] - \Delta \theta_R u_R[t] + \Delta \beta_R - d_R[t] \)
\( = \delta[t](1 - k_L - k_R) + \Delta \theta_L u_L[t] - \Delta \beta_L - \Delta \theta_R u_R[t] + \Delta \beta_R \)

Let us define \( c = (\frac{\Delta \theta_L}{\theta_L} (v^r + \beta_L) - \Delta \beta_L) - (\frac{\Delta \theta_R}{\theta_R} (v^r + \beta_R) - \Delta \beta_R) \), and our new eigenvalue \( \lambda = 1 - k_L - k_R - k_L \frac{\Delta \theta_L}{\theta_L} - k_R \frac{\Delta \theta_R}{\theta_R} \). In this case,

\( \delta[1] = \delta_0 \lambda + c \)
\( \delta[2] = \lambda (\delta_0 \lambda + c) + c \)
\( = \delta_0 \lambda^2 + c \lambda + c \)
\( \delta[3] = \lambda (\delta_0 \lambda^2 + c \lambda + c) + c \)
\( = \delta_0 \lambda^3 + c \lambda^2 + c \lambda + c \)
\( \delta[4] = \lambda (\delta_0 \lambda^3 + c \lambda^2 + c \lambda + c) + c \)
\( = \delta_0 \lambda^4 + c \lambda^3 + c \lambda^2 + c \lambda + c \)
\( \delta[5] = \delta_0 \lambda^5 + c (\lambda^4 + \lambda^3 + \lambda^2 + \lambda^1 + 1) \)
\( \delta[n] = \delta_0 \lambda^n + c (1 + \lambda^1 + \lambda^2 + \lambda^3 + \lambda^4 + \ldots + \lambda^n) \)
\( \delta[n] = \delta_0 \lambda^n + c \left( \sum_{k=0}^{n} \lambda^k \right) \) (rewriting in sum notation)
\( \delta[n] = \delta_0 \lambda^n + c \left( \frac{1 - \lambda^n}{1 - \lambda} \right) \) (sum of a geometric series)

If \( \lambda < 1 \), then \( \lambda^n = 0 \), so those terms drop out:
\[ \delta[n = t \to \infty] = \delta_0 \lambda^\infty + c \left( \frac{1 - \lambda^\infty}{1 - \lambda} \right) \]

\[ \delta[n = t \to \infty] = c \frac{1}{1 - \lambda} \]

\[ \delta_{ss} = c \frac{1}{1 - \lambda} \]

For your entertainment only: \( \delta_{ss} \) is fully-expanded form (not required) is

\[ \left( \frac{\Delta \theta_L}{\theta_L} (v^* + \beta_L) - \Delta \beta_L \right) - \left( \frac{\Delta \theta_R}{\theta_R} (v^* + \beta_R) - \Delta \beta_R \right) \]

\[ k_L + k_R + k_L \frac{\Delta \theta_L}{\theta_L} + k_R \frac{\Delta \theta_R}{\theta_R} \]

The answer is correct because plugging in zero into all the model mismatch terms into \( c \) causes \( c = 0 \), so \( \delta_{ss} = 0 \) if there is no model mismatch. Compared to the open-loop result of \( \delta_{ss} = \pm \infty \), the closed loop \( \delta_{ss} = c \frac{1}{1 - \lambda} \) is a much-desired improvement.

What does this mean for the car? It means that the car will turn initially for a bit but eventually converge to a fixed heading and keep going straight from there.

Problems

2. Design for Controllability and Observability I

We are given a system

\[ \dot{x}(t) = \begin{bmatrix} -3 & 3 \\ \gamma & -4 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \]

\[ y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \end{bmatrix} u(t) \]

with tunable parameter \( \gamma \).

(a) How should we tune \( \gamma \) to make the system controllable but not observable?

**Solution:**

The controllability matrix is

\[ \mathbf{C} = [B \ AB] = \begin{bmatrix} 1 & -3 \\ 0 & \gamma \end{bmatrix} \]

and the observability matrix is

\[ \mathbf{O} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ \gamma - 3 & -1 \end{bmatrix} \]

Thus, to make the system controllable but not observable, we should choose \( \gamma = 2 \).

(b) How should we tune \( \gamma \) to make the system observable but not controllable?

**Solution:**

To make the system observable but not controllable, we should choose \( \gamma = 0 \).
3. Design for Controllability and Observability II

We are given a new system
\[
\dot{x}(t) = \begin{bmatrix} 1 & 0 \\ -1 & -2 \end{bmatrix} x(t).
\]
along with only one sensor and one actuator to control and observe the system.

(a) Which state should we control with the actuator to make the system controllable?

Solution:
We should control the state \(x_1\). This makes the control matrix \(B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}\), so
\[
C = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}.
\]

(b) Which state should we measure with the sensor to make the system observable?

Solution:
We should measure the state \(x_2\). This makes the measurement matrix \(C = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}\), so
\[
\bar{C} = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}.
\]

4. Observability

Consider the following continuous time system.
\[
\dot{x}(t) = Ax(t) + Bu(t) \\
y(t) = Cx(t)
\]
(1)

We want to construct an estimate \(\hat{z}\) of the system state \(\bar{x}\). To do so, we construct a pretend system with the same \([A, B, C]\) models, the same input and the output of the last system along with an \(L\) system matrix. We do this to try and exploit the difference between the output of our pretend state and the actual output, with \(L\) being the “knob” that we can control.
\[
\dot{\hat{z}}(t) = A\hat{z}(t) + Bu(t) + L(C\hat{z}(t) - y(t))
\]
(2)

Define \(\bar{e}(t) = \hat{z}(t) - \bar{x}(t)\). This is the error term as a function of time.

(a) Using the two systems defined above, construct a system of the form,
\[
\frac{d\bar{e}}{dt}(t) = (A + LC)\bar{e}(t)
\]
(3)

Solution:
Subtracting (1) from (2), we get the desired answer:
\[
\begin{align*}
\dot{\hat{z}}(t) - \dot{\bar{x}}(t) &= A\hat{z}(t) + Bu(t) + L(C\hat{z}(t) - y(t)) - A\bar{x}(t) - Bu(t) \\
\dot{\bar{e}}(t) &= A(\hat{z}(t) - \bar{x}(t)) + L(C\hat{z}(t) - C\bar{x}(t)) \\
\dot{\hat{e}}(t) &= A\hat{e}(t) + LC(\hat{z}(t) - \bar{x}(t)) \\
\bar{e}(t) &= (A + LC)\bar{e}(t)
\end{align*}
\]
(b) We want
\[ \lim_{t \to \infty} \vec{e}(t) = \vec{0} \]
What does that imply about (3)?

**Solution:**
This means that we want (3) to be stable: all of the eigenvalues of \( A + LC \) should have a real component less than zero.

(c) Does the initial value of the guess \( \vec{z}(0) \) matter in the long term?

**Solution:**
Not really, since the \( \vec{e}(t) \) tends to \( \vec{0} \). This means that no matter how bad our initial guess, we will eventually have a good estimate.

5. **Rank 1 Decomposition**

In this problem, we will decompose a few images into linear combinations of rank 1 matrices.

(a) Consider a standard \( 8 \times 8 \) chessboard shown in Figure 1. Assume that black colors represent \(-1\) and that white colors represent \(1\).

![Figure 1: 8 x 8 chessboard.](image)

Decompose this image into rank 1 images. Draw out each of these rank 1 images and express the original chessboard as a linear combination of the individual rank 1 images.

**Solution:**
The chessboard is already a rank 1 image, so we do not need to decompose it further.

(b) Assume that the chessboard is given by the following \( 8 \times 8 \) matrix \( C_1 \):

\[
C_1 = \begin{bmatrix}
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
-1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
-1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
-1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
-1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
\end{bmatrix}
\]

Express \( C_1 \) as a linear combination of outer products.

**Solution:**
Using the result from the previous part,

\[
C_1 = \begin{bmatrix}
1 \\
-1 \\
1 \\
-1 \\
1 \\
-1 \\
1 \\
-1
\end{bmatrix}
\begin{bmatrix}
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1
\end{bmatrix}
\]

(c) For the same chessboard shown in Figure 1, now assume that black colors represent 0 and that white colors represent 1.

Decompose this image into rank 1 images. Draw out each of these rank 1 images and express the original chessboard as a linear combination of the individual rank 1 images.

**Solution:**

The chessboard is now a rank 2 image, so we need to decompose it.

*Image 1:*

![Image 1](image1.png)

*Image 2:*

![Image 2](image2.png)

The chessboard equals the sum of image 1 and image 2.
(d) Assume that the chessboard is given by the following $8 \times 8$ matrix $C_2$:

$$C_2 = \begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1
\end{bmatrix}$$

Express $C$ as a linear combination of outer products.

**Solution:**

Using the result from the previous part,

$$C_2 = \begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1
\end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

(e) Now consider the Swiss flag shown in Figure 2. Assume that red colors represent 0 and that white colors represent 1.

![Swiss flag](image)

Figure 2: Swiss flag.

Decompose this image into rank 1 images. Draw out each of these rank 1 images and express the Swiss flag as a linear combination of the individual rank 1 images.

**Solution:**

The Swiss flag is a rank 2 image, so we need to find two rank 1 images.

**Solution 1:**

Image 1:
The Swiss flag equals the sum of image 1 and image 2.

**Solution 2:**

Image 1:

![Image 1](image1.png)

Image 2:

![Image 2](image2.png)

The Swiss flag equals the difference between image 1 and image 2.
(f) Assume that the Swiss flag is given by the following $5 \times 5$ matrix $S$:

$$S = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}$$

Express $S$ as a linear combination of outer products.

**Solution:**

We can use the result from the previous part.

**Solution 1:**

$$S = \begin{bmatrix}
0 \\
1 \\
1 \\
1 \\
0 \\
\end{bmatrix} \begin{bmatrix}
0 & 0 & 1 & 0 & 0 \\
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
1 \\
0 \\
0 \\
\end{bmatrix} \begin{bmatrix}
0 & 1 & 0 & 1 & 0 \\
\end{bmatrix}$$

**Solution 2:**

$$S = \begin{bmatrix}
0 \\
1 \\
1 \\
1 \\
0 \\
\end{bmatrix} \begin{bmatrix}
0 & 1 & 1 & 1 & 0 \\
\end{bmatrix} - \begin{bmatrix}
0 \\
1 \\
0 \\
1 \\
0 \\
\end{bmatrix} \begin{bmatrix}
0 & 1 & 0 & 1 & 0 \\
\end{bmatrix}$$

6. **Balance**

Justin is working on a small jumping robot named Salto. Salto can bounce around on the ground, but Justin would like Salto to balance on its toe and stand still. In this problem, we’ll work on systems that could help Salto balance on its toe using its reaction wheel tail.
Standing on the ground, Salto’s dynamics in the x-z plane (called the sagittal plane in biology) look like an inverted pendulum with a flywheel on the end:

\[
\begin{align*}
(I_1 + (m_1 + m_2)l^2) \ddot{\theta}_1 &= -K_t u + (m_1 + m_2)lg \sin(\theta_1) \\
I_2 \ddot{\theta}_2 &= K_t u
\end{align*}
\]

where $\theta_1$ is the angle of the robot’s body relative to the ground (0 is straight up), $\dot{\theta}_1$ is its angular velocity, $\dot{\theta}_2$ is the angular velocity of the reaction wheel tail, and $u$ is the current input to the tail motor. $m_1, m_2, I_1, I_2, l, K_t$ are positive constants representing system parameters (masses and angular momentums of the body and tail, leg length, and motor torque constant respectively) and $g = 9.81 \text{ m/s}^2$ is the acceleration due to gravity.

Numerically substituting Salto’s physical parameters, the differential equations become approximately:

\[
\begin{align*}
0.001 \ddot{\theta}_1 &= -0.025 u + 0.1 \sin(\theta_1) \\
5(10^{-5}) \ddot{\theta}_2 &= 0.025 u
\end{align*}
\]

For this problem, we’ll look at a reduced suite of sensors on Salto. Our only output will be the tail encoder that measures the angular velocity of the tail relative to the body:

\[ y = \dot{\theta}_2 - \dot{\theta}_1 \]

(a) Using the state vector $[\theta_1 \quad \dot{\theta}_1 \quad \dot{\theta}_2]^\top$, input $u$, and output $y$ linearize the system about the point $[0 \quad 0 \quad 0]^\top$. Write the linearized equations as $\frac{d}{dt} \bar{x} = A \bar{x} + B u$ and $y = C \bar{x}$. Write out the matrices with the physical numerical values, not symbolically.

**Note:** Since the tail is like a wheel, we care only about its angular velocity $\dot{\theta}_2$ and not its angle $\theta_2$.

**Solution:**
Numerically, the dynamics are:

\[
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
100 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\theta_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
-25 \\
500
\end{bmatrix} u
\]

\[y =
\begin{bmatrix}
0 & -1 & 1
\end{bmatrix}
\begin{bmatrix}
\theta_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}
\]

For those interested, the symbolic dynamics are:

\[
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 \\
\frac{(m_1+m_2)g \sin(\theta)}{l_1+(m_1+m_2)l^2} & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\theta_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
\frac{-K_t}{l_1+(m_1+m_2)l^2} \\
\frac{K_t I_2}{l_2}
\end{bmatrix} u
\]

\[y =
\begin{bmatrix}
0 & -1 & 1
\end{bmatrix}
\begin{bmatrix}
\theta_1 \\
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}
\]

(b) Is the system fully controllable? Is the system fully observable?

**Solution:**

\[\mathcal{C} = [B \ AB \ A^2B] = \begin{bmatrix} 0 & -25 & 0 \\ -25 & 0 & -2500 \\ 500 & 0 & 0 \end{bmatrix}\]

which is full rank, so the system is fully controllable.

\[\mathcal{O} = [C \ CA \ CA^2] = \begin{bmatrix} 0 & -1 & 1 \\ -100 & 0 & 0 \\ 0 & -100 & 0 \end{bmatrix}\]

which is rank 3, so the system is fully observable.

(c) Design an observer of the form \(\dot{\hat{x}} = A\hat{x} + Bu + L(C\hat{x} - y)\) and solve for the gains in \(L\) that make the observer dynamics converge with all eigenvalues \(\lambda_1 = \lambda_2 = \lambda_3 = -10\).

**Solution:**

The observer dynamics are dictated by

\[\dot{\hat{e}} = (A + LC)\hat{e}\]

where \(\hat{e}\) is the error between the estimated state \(\hat{x}\) and the true state \(\bar{x}\). The characteristic polynomial is:

\[
\det(A + LC) = 0
\]

\[
\det \begin{bmatrix}
0 & 1 - l_1 & l_1 \\
100 & -l_2 & l_2 \\
0 & -l_3 & l_3
\end{bmatrix} = 0
\]

\[\lambda^3 + (-l_3 + l_2)\lambda^2 + (100l_1 - 100)\lambda + 100l_3 = 0\]
The desired characteristic polynomial is:

\[(\lambda + 10)^3 = 0\]

\[\lambda^3 + 30\lambda^2 + 300\lambda + 1000 = 0\]

which we can achieve by matching the coefficients of matching powers:

\[-l_3 + l_2 = 30\]
\[100l_1 - 100 = 300\]
\[100l_3 = 1000\]

These equations are solved by the gains: \(l_1 = 4\), \(l_2 = 40\), and \(l_3 = 10\). Written as a matrix,

\[L = \begin{bmatrix} 4 \\ 40 \\ 10 \end{bmatrix}\]

(d) Let’s implement a controller for our system using an analog electrical circuit! You can use the following circuit components in Figure 2:

<table>
<thead>
<tr>
<th>Block diagram symbol</th>
<th>Electrical Circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(v_n \rightarrow \int \rightarrow v_{out} )</td>
<td>(v_n \rightarrow \int \rightarrow v_{out} )</td>
</tr>
<tr>
<td>Integrator: (v_{out} = \int v_n )</td>
<td>(v_n \rightarrow \int \rightarrow v_{out} )</td>
</tr>
<tr>
<td>(v_n \rightarrow k \rightarrow v_{out} )</td>
<td>(v_n \rightarrow k \rightarrow v_{out} )</td>
</tr>
<tr>
<td>Amplifier: (v_{out} = kv_n )</td>
<td>(v_n \rightarrow k \rightarrow v_{out} )</td>
</tr>
<tr>
<td>Sum: (v_{out} = v_1 + v_2 + \ldots + v_n )</td>
<td>(v_n \rightarrow k \rightarrow v_{out} )</td>
</tr>
</tbody>
</table>

Figure 4: Circuit components and block diagram symbols.
Using state feedback, Justin has selected the control gains $K = \begin{bmatrix} -20 & -5 & -0.01 \end{bmatrix}$ for his input $u = -K \ddot{x}$. Draw a block diagram in the box in Figure 5 that implements this controller. Afterwards, design the circuit that implements the controller. Use relatively reasonable component values.

Optional: What are the eigenvalues of the closed loop dynamics for the given $K$?

![Figure 5: Fill in the box to implement the state feedback controller.]

Solution:

![Figure 6: Here is a possible circuit. Other resistor values will work as long as the gains are the same.]

Optional: The eigenvalues are $\lambda = -116.6$ and $\lambda = -1.697 \pm 1.187j$. 

EECS 16B, Spring 2018, Homework 7
Contributors:

- Edward Wang.
- John Maidens.
- Siddharth Iyer.
- Titan Yuan.
- Justin Yim.