# Lab 6b: Luenberger Observer Design for Inverted Pendulum

**Name:** Mingjun Wu, Matthew Domine, Mitchel Stokes

**Date:** 4/26/2021

## Objectives

The objective of this lab is to design a full-state observer to estimate the state of an inverted pendulum system given just the position of the cart and the pendulum. We will utilize this estimate for full state feedback control of the system.

## Theory

Pole placement design is performed under the assumption that measurements of all states of the system are available. However, in many physical systems not all states may be easily measurable and thus states need to be estimated based on the limited sensing available. In this case the state feedback becomes $u = -K\hat{x}$, where $\hat{x}$ is the estimated state. We cannot use the controller $u = -Kx$, because the only measurements we have available are $y$.

Recall from class the dynamics of a Luenberger observer:

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - \hat{y}) \qquad (1)$$

where $y = Cx$ and $\hat{y} = C\hat{x}$. The first two terms in the above equation, $A\hat{x} + Bu$, can be called the predictor part and is a replica of the plant dynamics. However, because of uncertainties or errors in the plant model, the estimate of the state using only the predictor ("open-loop") will generally not match the actual state of the system. The corrective term $L(y - \hat{y})$ is thus needed. Together, these form the Luenberger observer.

The $L(y - \hat{y})$ term corrects future estimates of the state based on the present error in estimation. The gain matrix $L$ can be considered a parameter which weighs the relative importance between the predictor and the corrector in state estimation. Intuitively, a "low" value for $L$ is chosen when our confidence in the model (i.e. the predictor) is high and/or confidence in measurement y is low (i.e. when the measurements are noisy) and vice-versa for a "high" value of $L$. The objective of this lab is to design the observer gain matrix $L$ and use the state estimator for feedback control of the inverted-pendulum system instead of our previous derivative-based approximation.

# Lab

1. Implement the state feedback controller operating on the state estimate $\hat{x}$ provided by the Luenberger observer on the hardware. For a zero reference signal, observe and record the output $\hat{y}$ of the observer and the actual measurement y when manually applying small perturbations. That is, plot both the estimated and actual signals on the same graph for the position of the cart and the pendulum. The difference between these two signals indicates how well the observer estimates the state of the system.

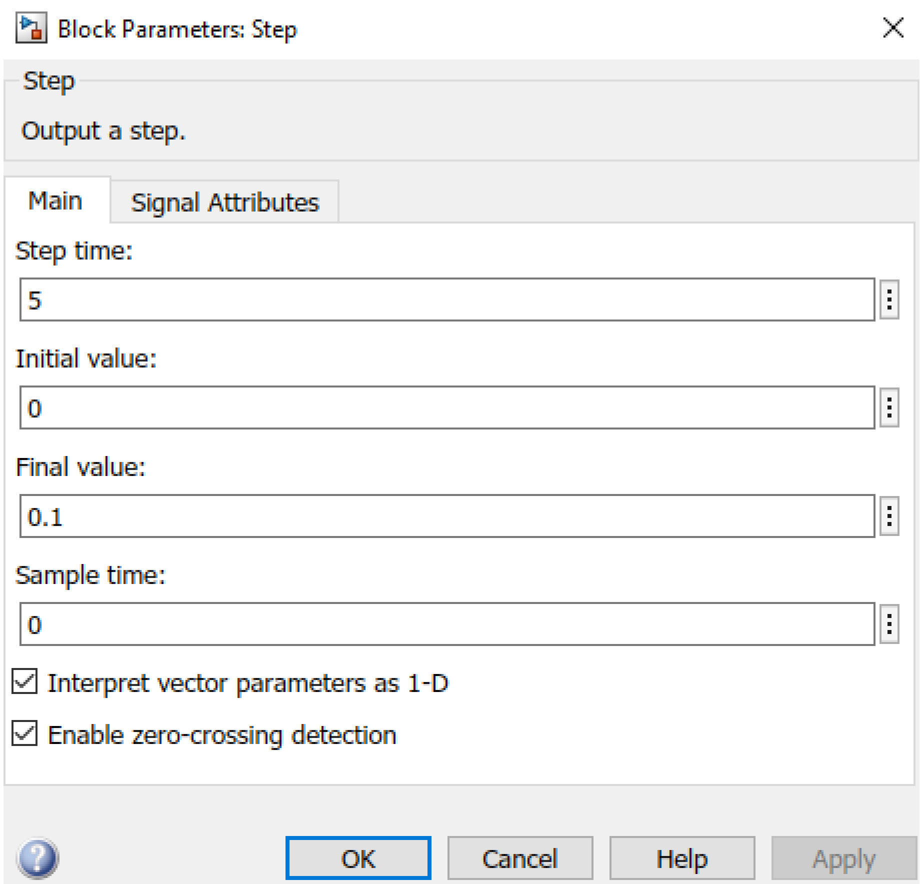


**Figure 1.** Settings For the Reference Step Function Used as A Perturbation.

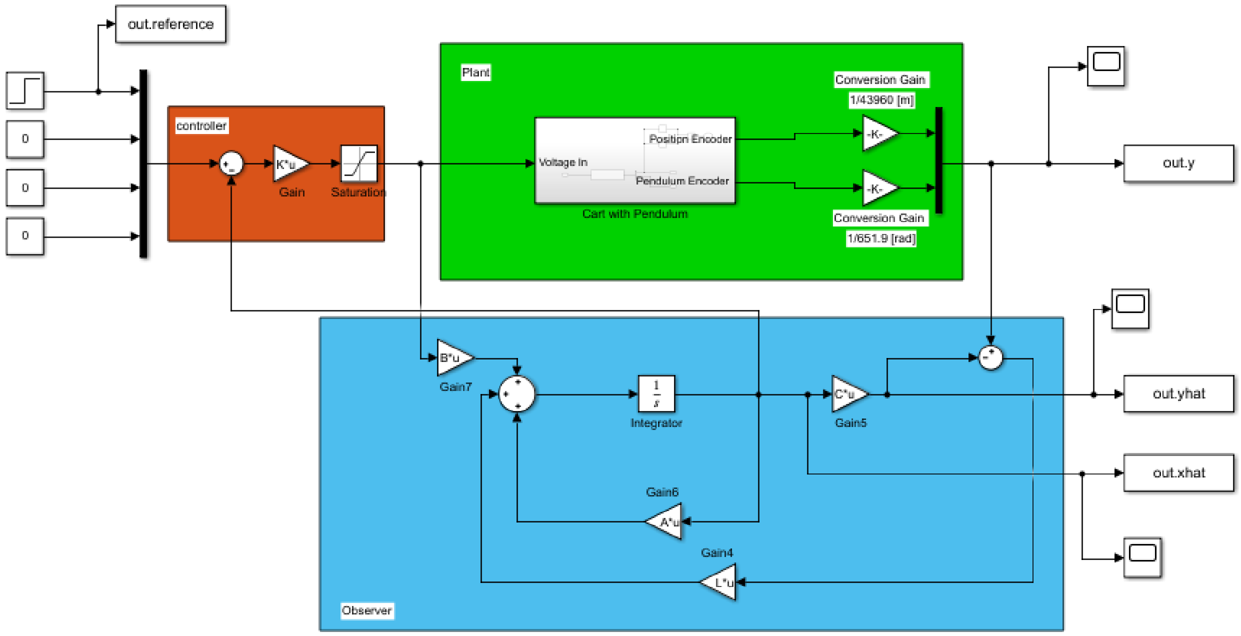


**Figure 2.** Simulink Block Diagram Of the Controller/Plant/Observer System.

### 4.1

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';

tstop = 10;
out = sim('statespaceobserverhstep',tstop);
y = out.y.data;
yhat = out.yhat.data;
reference = out.reference.data;

position = y(:,1);
theta = y(:,2);
positionhat = yhat(:,1);
thetahat = yhat(:,2);
t = out.tout;

plot(t,position)A
hold on
plot(t,positionhat)
hold on
plot(t,reference)
xlabel('Time (s)');ylabel('Position (m)');title('Position Output Actual vs Estimate')
legend('Actual Measurement y','Estimate Measurement $$\hat{y}$$','Reference','Interpreter','latex','Location',"best")

figure
plot(t,theta)
hold on
plot(t,thetahat)
hold on
yline(0)
xlabel('Time (s)');ylabel('Angle (rad)');title('Pendulum Angle Output Actual vs Estimate')
legend('Actual Measurement y','Estimate Measurement $$\hat{y}$$','Reference','Interpreter','latex','Location',"best")
```

**Figure 3.** MATLAB Script Used to Generate the Plots Needed For 4.1.

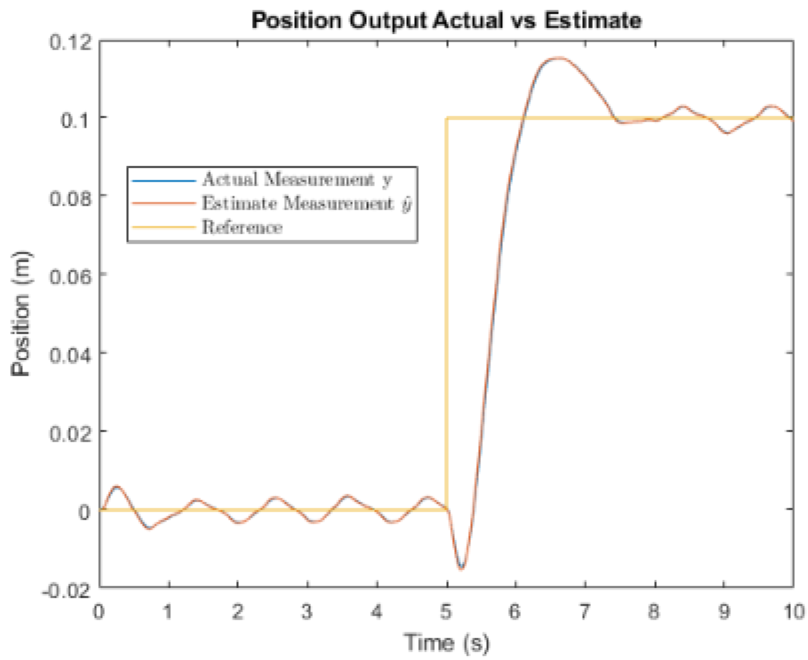


**Figure 4.** Estimated And Actual Position Signals $y$ And $\hat{y}$ for the Cart And Pendulum. Note the Relatively Small Differences Between the Estimated And Actual Position Signals.

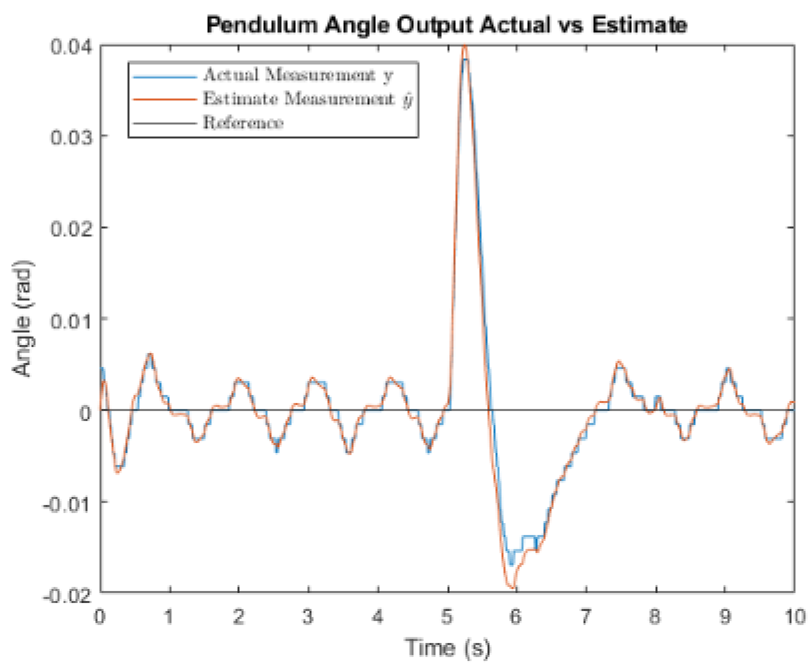


**Figure 5.** Estimated And Actual Pendulum Angle Signals $y$ And $\hat{y}$ for the Cart And Pendulum. Note the Relatively Small Differences Between the Estimated And Actual Position Signals.

2. We will now compare the controllers from Lab 6a and Lab 6b. Remember that the closed loop poles of both systems are the same. For each of the following reference signals, qualitatively describe any noticeable differences in performance, and plot the cart position and the angular position of the rod for both controllers on top of each other and compare their tracking abilities.

## 4.2.1 Zero reference

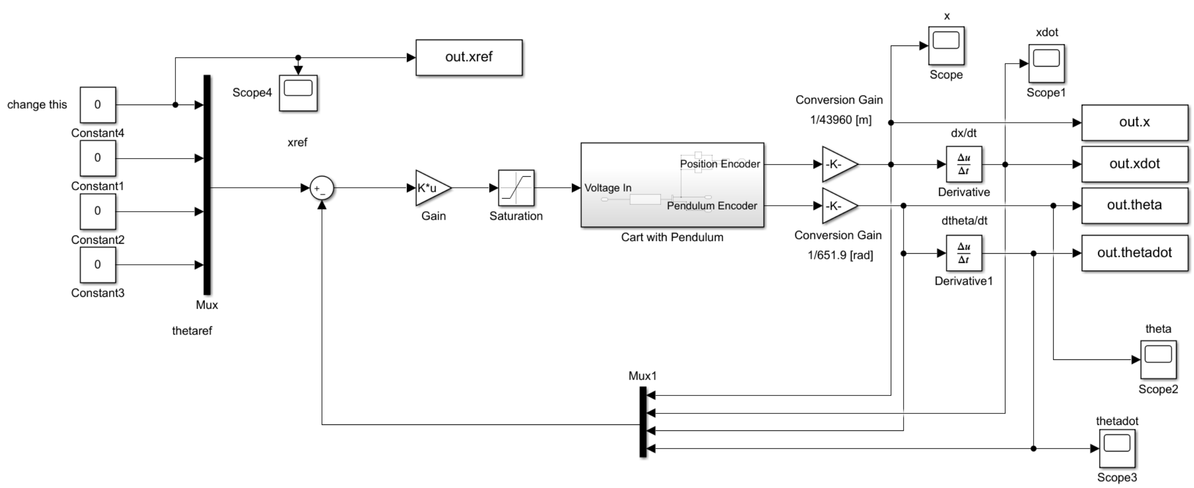


**Figure 6.** Simulink Block Diagrams for the Lab 6a Controller in the Zero Reference Case.

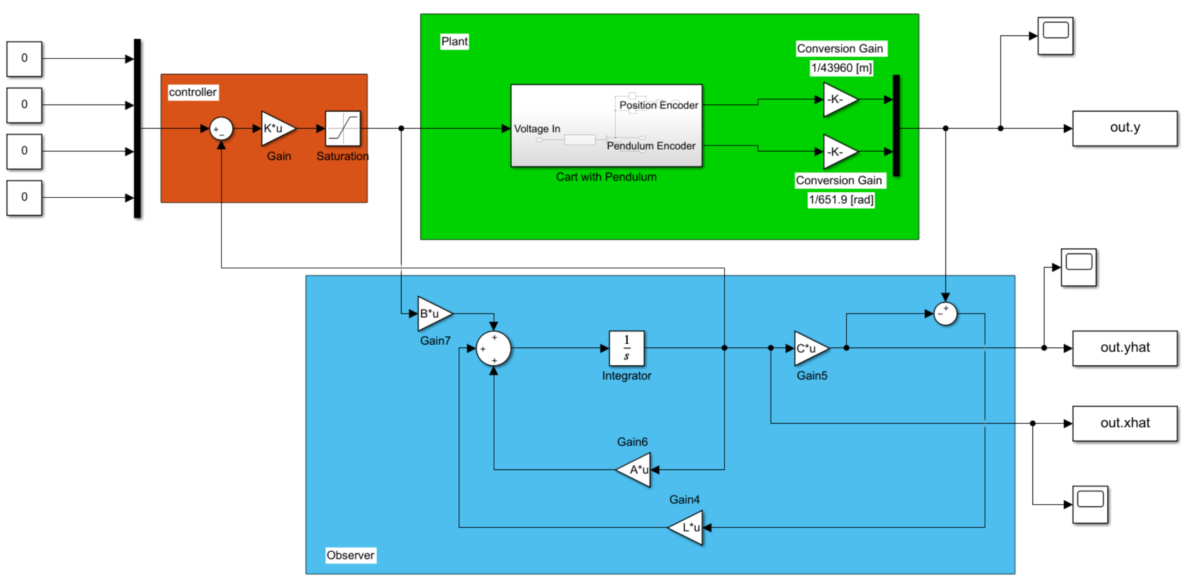


**Figure 7.** Simulink Block Diagrams For the Lab 6b Controller in the Zero Reference Case.

### 4.2.1

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverh',tstop);
outss = sim('cartpendulum1',tstop);

y = outob.y.data;
position_ob = y(:,1);
position_ss = outss.x.data;
% reference = outss.xref.data;

theta_ob = y(:,2);
theta_ss = outss.theta.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,position_ss)
hold on
plot(t,position_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Position (m)');title('Cart Position')

figure

plot(t2,theta_ss)
hold on
plot(t,theta_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angle (rad)');title('Pendulum Angle')
```

**Figure 8.** MATLAB Script Used to Generate the Plots For 4.2.1 (Zero Reference).

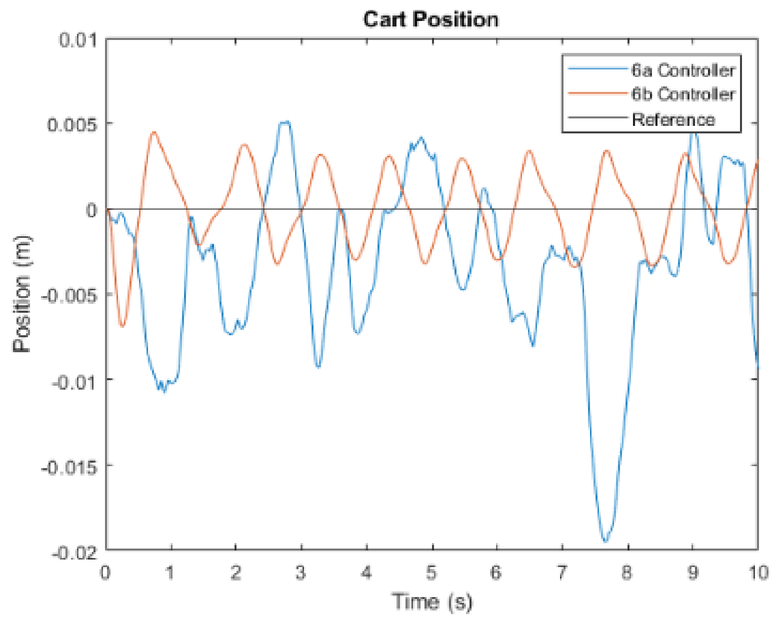


**Figure 9.** Position Output with 6a and 6b Controllers for the Cart and Pendulum System in the Zero Reference Case.

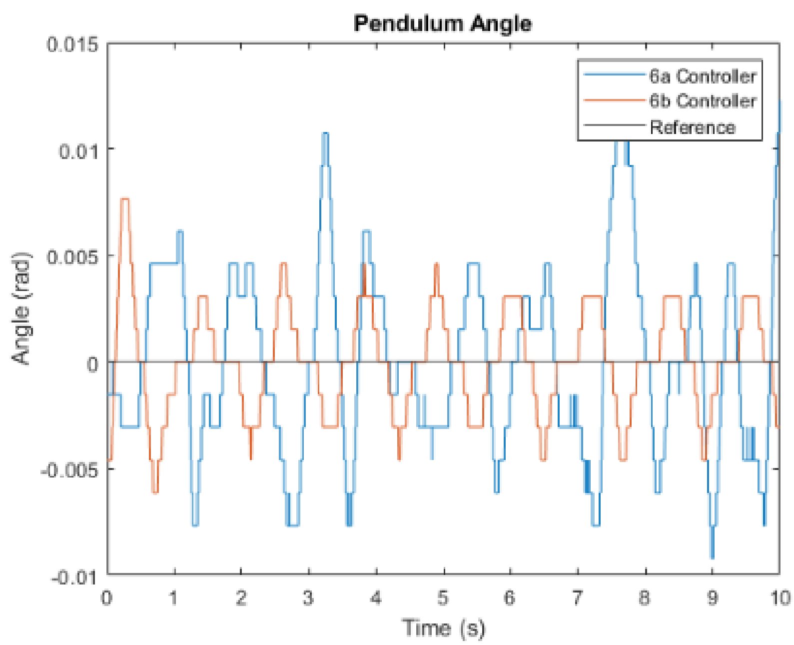


**Figure 10.** Pendulum Angle Output with 6a and 6b Controllers for the Cart and Pendulum System in the Zero Reference Case.

**Qualitative description:**

**Qualitative description: the lab 6B controller generates smoother and less noisy cart position response and pendulum angle response than the lab 6a controller. The responses using the 6b controller track the input reference better. Also the responses from the two controllers are completely out of phase most of the time.**

<u>4.2.2 Zero reference with small perturbations (try to be consistent in how you apply the perturbations)</u>

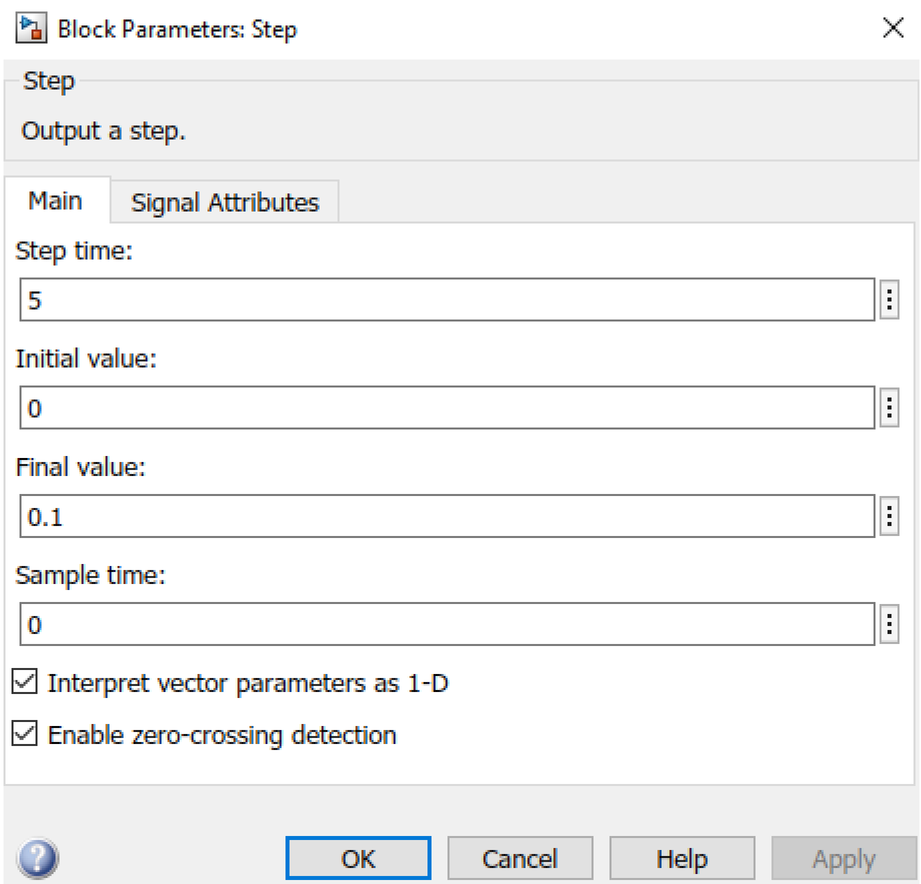


**Figure 11.** Step Function Settings Used for the Small Perturbation. This Is Identical To the Step Function Used Previously (**Figure 1**).

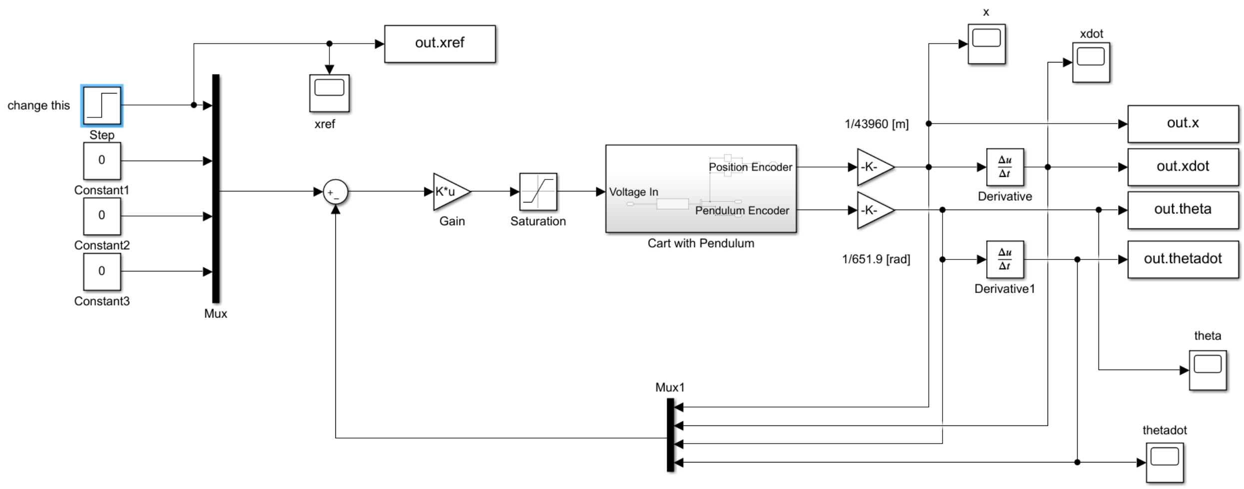


**Figure 12.** Simulink Block Diagrams for the Lab 6a Controller in the Zero Reference with Small Perturbations Case. Note the Addition of the Step Function Block to Create the Perturbation.

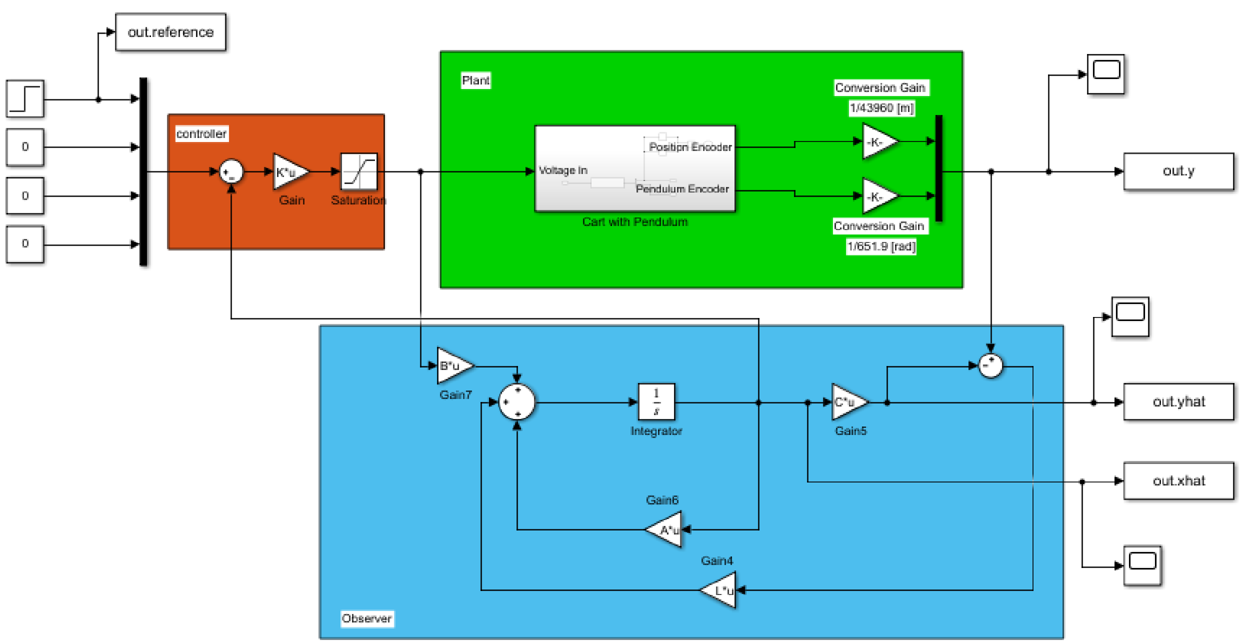


**Figure 13.** Simulink Block Diagrams for the Lab 6b Controller in the Zero Reference with Small Perturbations Case. Note the Addition of the Step Function Block to Create the Perturbation.

### 4.2.2

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverhstep',tstop);
outss = sim('cartpendulumstep1',tstop);

y = outob.y.data;
position_ob = y(:,1);
position_ss = outss.x.data;
reference = outss.xref.data;

theta_ob = y(:,2);
theta_ss = outss.theta.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,position_ss)
hold on
plot(t,position_ob)
hold on
plot(t2,reference)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Position (m)');title('Cart Position')

figure

plot(t2,theta_ss)
hold on
plot(t,theta_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angle (rad)');title('Pendulum Angle')
```

**Figure 14.** MATLAB Script Used to Generate the Plots for 4.2.2 (Zero Reference W/ Perturbation).

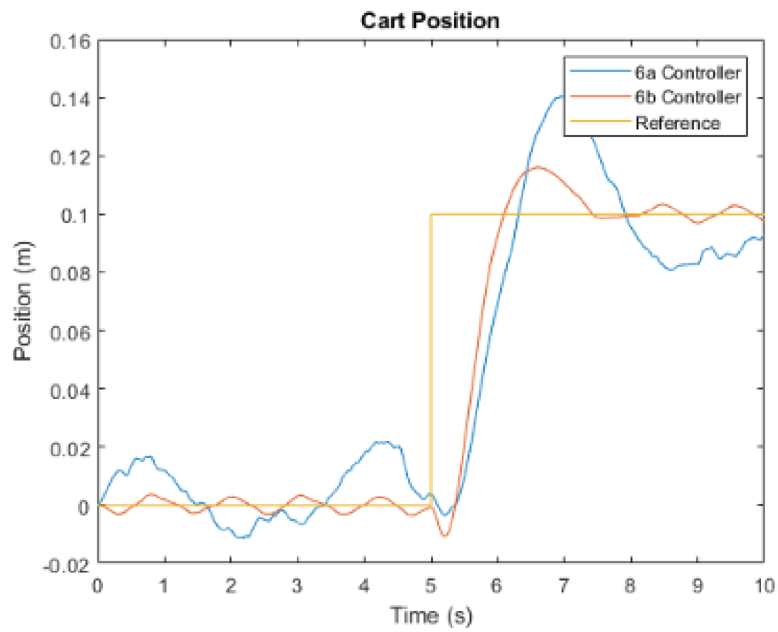


**Figure 15.** Position Output with 6a and 6b Controllers for the Cart and Pendulum System in the Zero Reference with Perturbation Case.

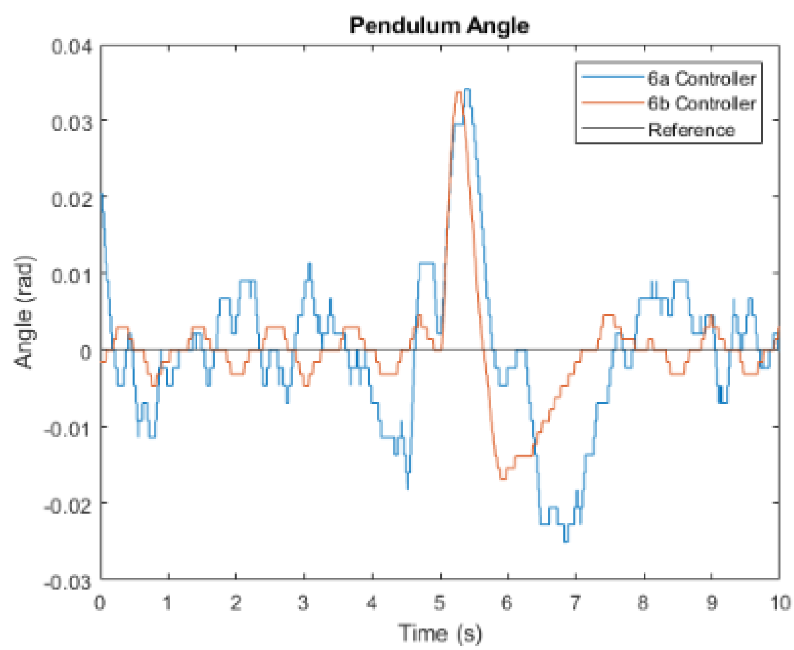


**Figure 16.** Pendulum Angle Output with 6a and 6b Controllers for the Cart and Pendulum System in the Zero Reference with Perturbation Case.

**Qualitative description: the lab 6B controller generates smoother and less noisy cart position response and pendulum angle response than the lab 6a controller. The responses using the 6b controller also track the reference input better. The responses generated by the lab 6A controller lag the ones generated by the lab 6B controller**.

4.2.3 Sinusoidal reference position with amplitude 5 cm and frequency 1 rad/s, i.e. r1(t) = .05 sin(t). The reference velocity, angle, and angular velocity should be set to 0.

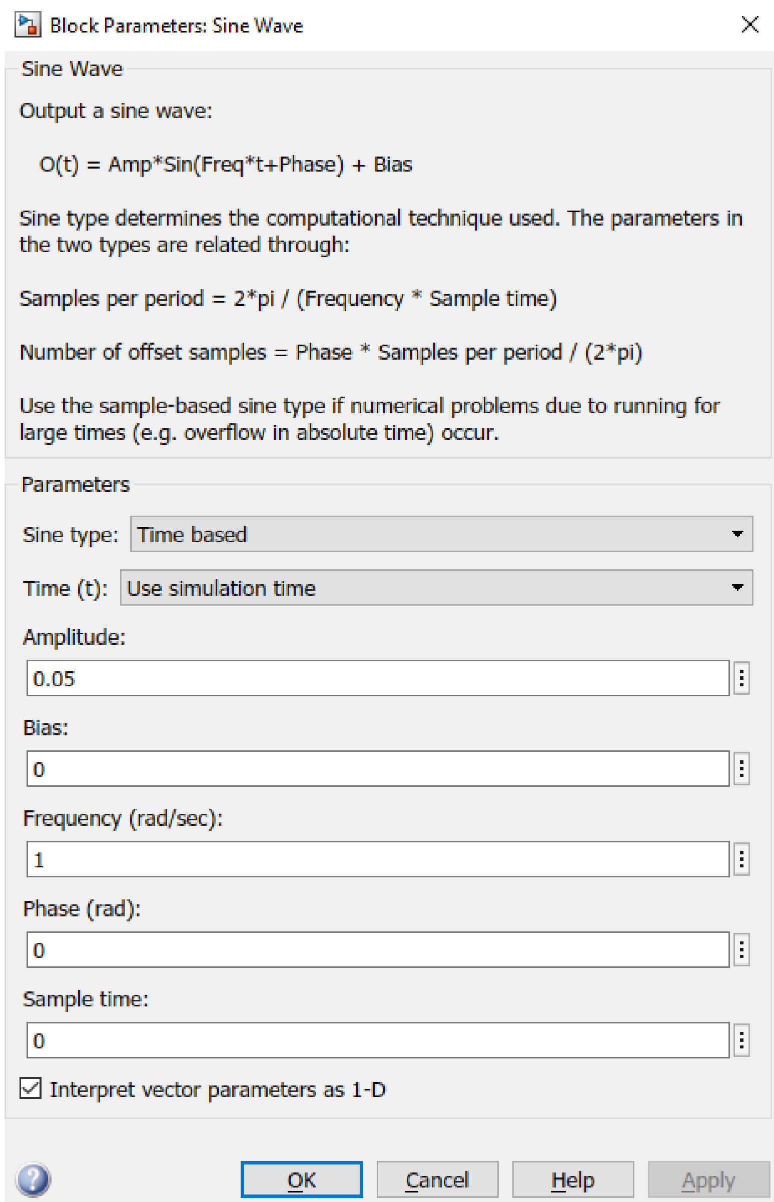


**Figure 17.** Sin Function Settings Used for the Sinusoidal Reference Position Input.

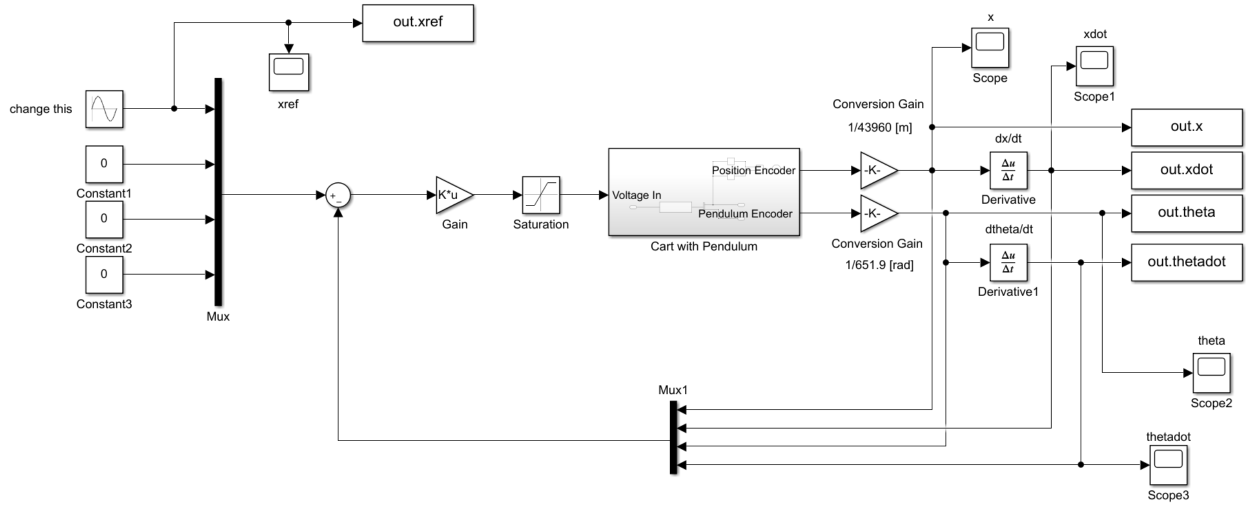


**Figure 18.** Simulink Block Diagrams for the Lab 6a and Lab 6b controllers in sinusoidal position reference case. Note the addition of the sine function block as an input to the controller

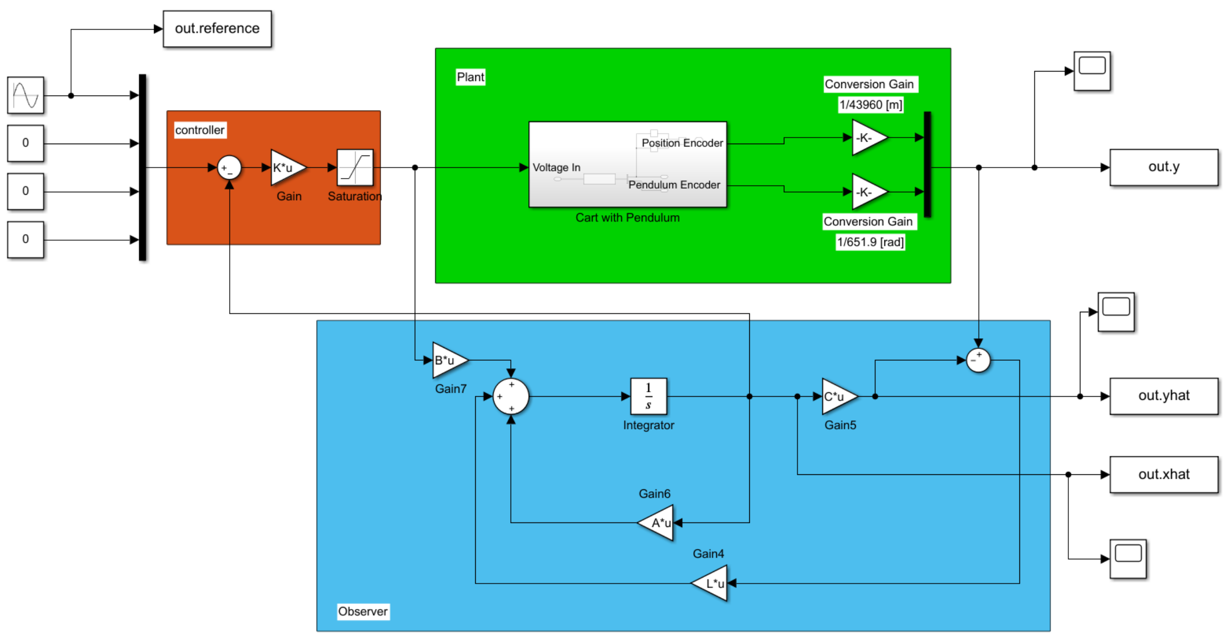


**Figure 19.** Simulink Block Diagrams for the Lab 6a and Lab 6b controllers in sinusoidal position reference case. Note the addition of the sine function block as an input to the controller.

### 4.2.3

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverhsine',tstop);
outss = sim('cartpendulumsineh1',tstop);

y = outob.y.data;
position_ob = y(:,1);
position_ss = outss.x.data;
reference = outss.xref.data;

theta_ob = y(:,2);
theta_ss = outss.theta.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,position_ss)
hold on
plot(t,position_ob)
hold on
plot(t2,reference)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Position (m)');title('Cart Position')

figure

plot(t2,theta_ss)
hold on
plot(t,theta_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angle (rad)');title('Pendulum Angle')
```

**Figure 20.** MATLAB script used to generate the plots for 4.2.3 (sinusoidal reference).

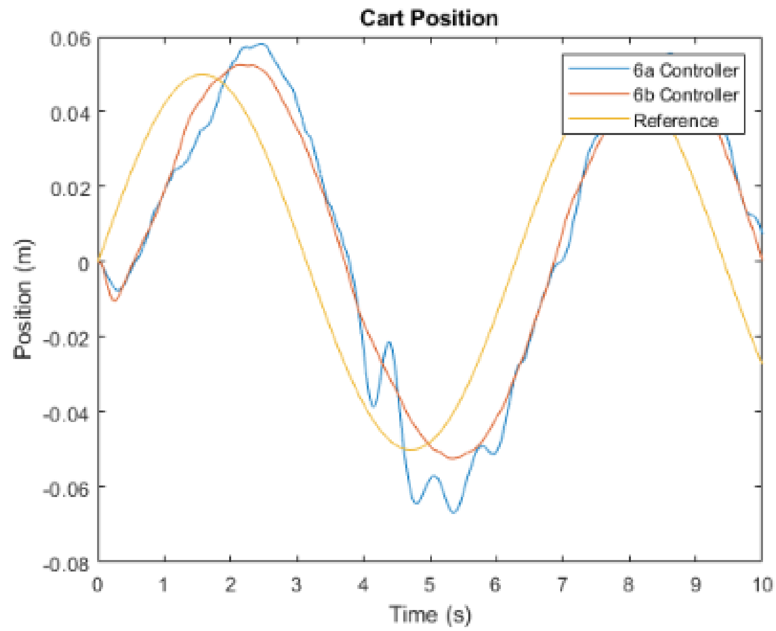


**Figure 21.** Position Output with 6a and 6b Controllers for the Cart and Pendulum System in the sinusoidal reference case.

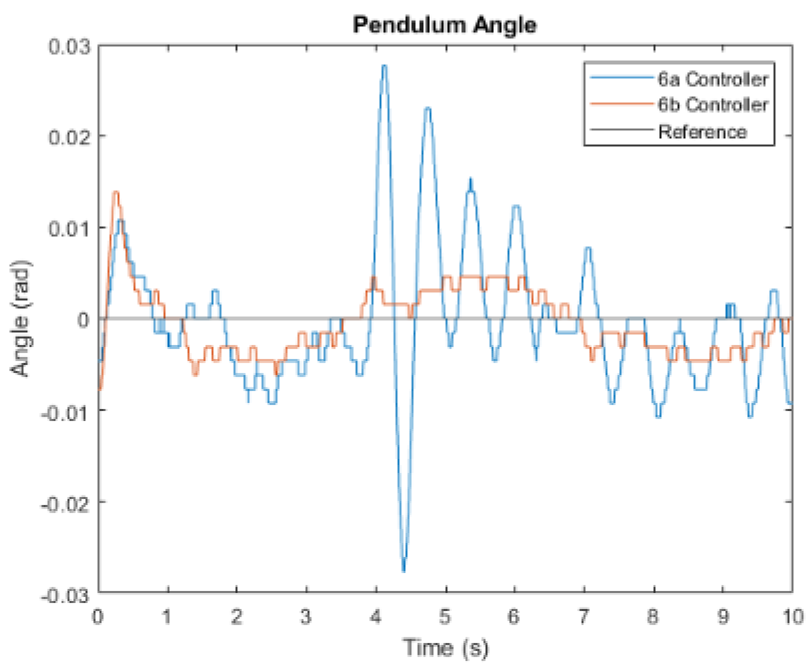


**Figure 22.** Pendulum Angle Output with 6a and 6b Controllers for the Cart and Pendulum System in the sinusoidal reference case.

**Qualitative description: the lab 6B controller generates smoother and less noisy cart position response and pendulum angle response than the lab 6a controller. The responses using the 6b controller also track the reference input better. The responses generated by the lab 6A controller lag the ones generated by the lab 6B controller slightly, but they also seem to be nearly in phase.**

3. Now we will look at the differences in performances a little more closely. Compare the estimates of the cart and pendulum velocities from this lab with the measurements obtained by taking the derivatives of the position and angle signals from the previous lab. How do these two schemes differ when a noise is present in the actual measurement of the positions?

**When noise is present in the actual measurement of the positions, we can see that the estimates from this lab (6b) are far less noisy than the derived velocity signals from the previous lab (6a). This is due to the fact that using a derivative block for noisy signals tends to greatly amplify the magnitude of the noise on the resulting signal. As can be seen by the following figures below, the estimates with controller 6b appear to be smooth curving signals with relatively small noise while the estimates with controller 6a are very noisy and the output can barely be seen. The estimates with controller 6a has the largest noise spikes at the beginning of the signals because there are discrete steps while using the encoder. There is a certain amount of degrees and the time step is very small. As time sampling before and right after the tick, the rise over the run for the velocity over a very small time step by using point slope method causes the large spike in the beginning. In the case of the observer design, the estimates with controller 6b are apparently much more robust to noise.**

4.3.1

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverh',tstop);
outss = sim('cartpendulum1',tstop);

xhat = outob.xhat.data;
velocity_ob = xhat(:,2);
velocity_ss = outss.xdot.data;
% reference = outss.xref.data; % Reference for xdot and thetadot is zero

thetadot_ob = xhat(:,4);
thetadot_ss = outss.thetadot.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Zero)')
figure
plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
ylim([-0.4 0.4])
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Zero)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Zero)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
ylim([-0.4 0.4])
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Zero)')
```

**Figure 23.** MATLAB script used to generate the cart velocity and angular velocity plots for the zero reference case.

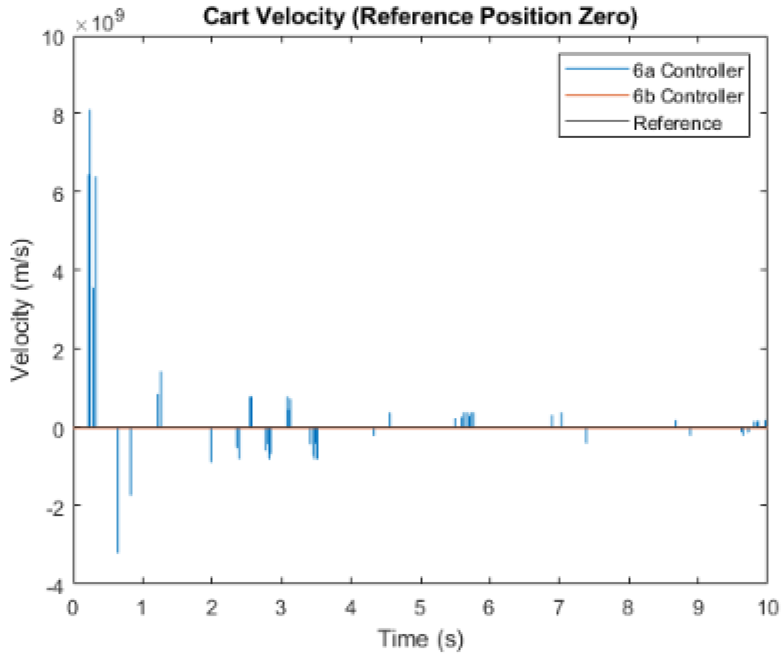


**Figure 24.** Cart velocity plots for the zero reference case.

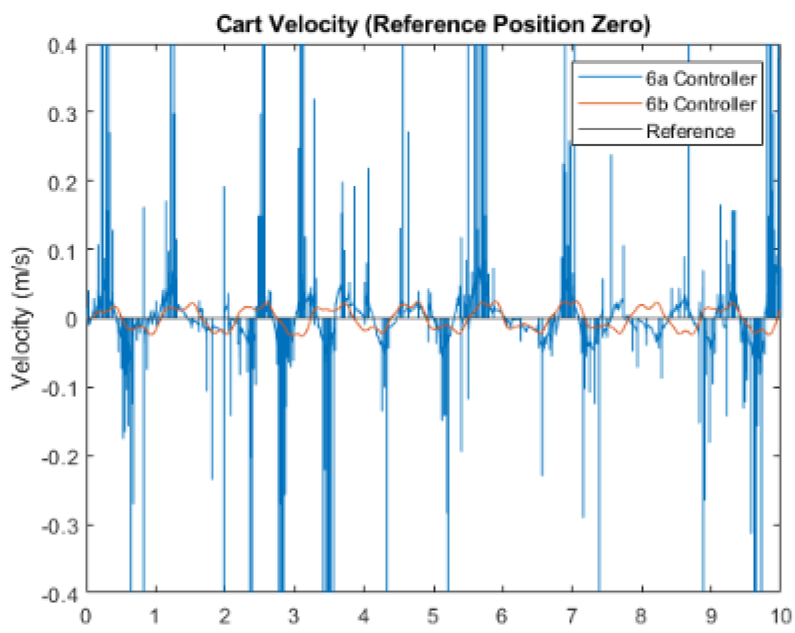


**Figure 25.** Cart velocity plots for the zero reference case (Zoomed in).

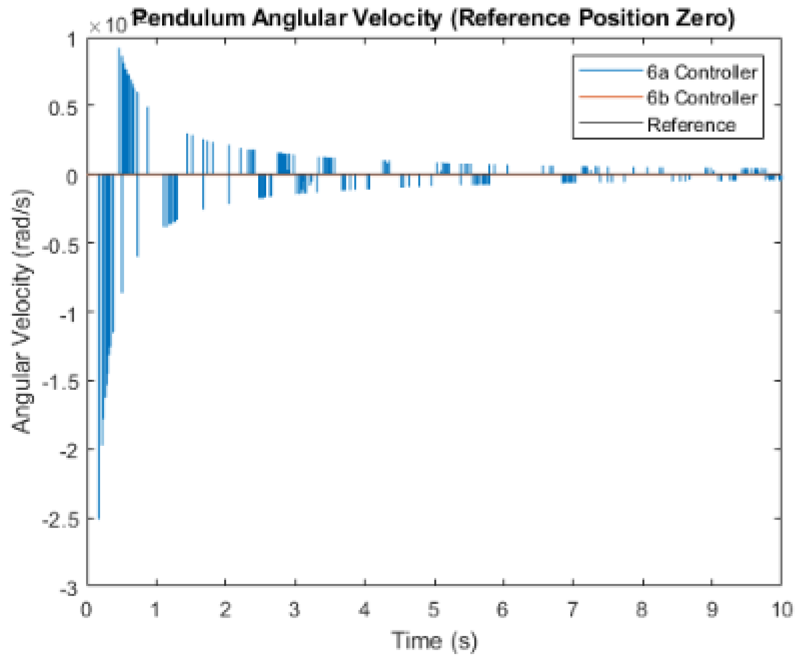


**Figure 26.** Pendulum angular velocity plots for the zero reference case.

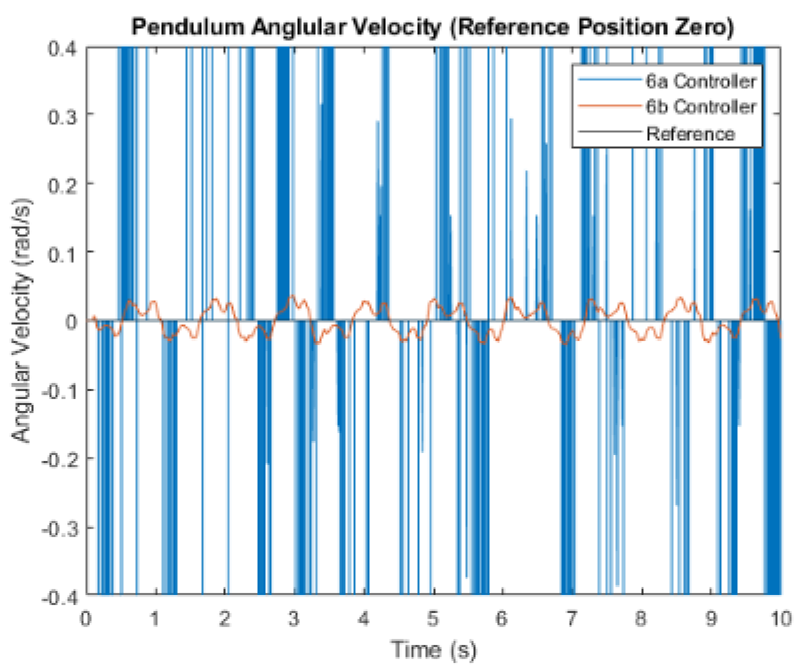


**Figure 27.** Pendulum angular velocity plots for the zero reference case **(**Zoomed in).

### 4.3.2

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverhstep',tstop);
outss = sim('cartpendulumstep1',tstop);

xhat = outob.xhat.data;
velocity_ob = xhat(:,2);
velocity_ss = outss.xdot.data;

thetadot_ob = xhat(:,4);
thetadot_ss = outss.thetadot.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Step)')
figure
plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
ylim([-0.4 0.4])
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Step)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Step)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
ylim([-0.4 0.4])
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Step)')
```

**Figure 28.** MATLAB script used to generate the cart velocity and angular velocity plots for the step reference case.

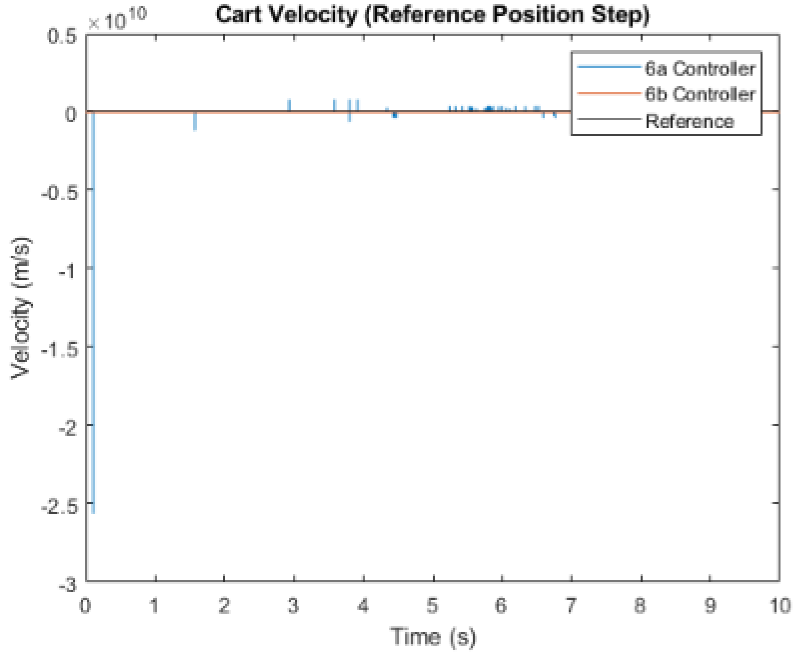


**Figure 29.** Cart velocity plots for the step reference case.

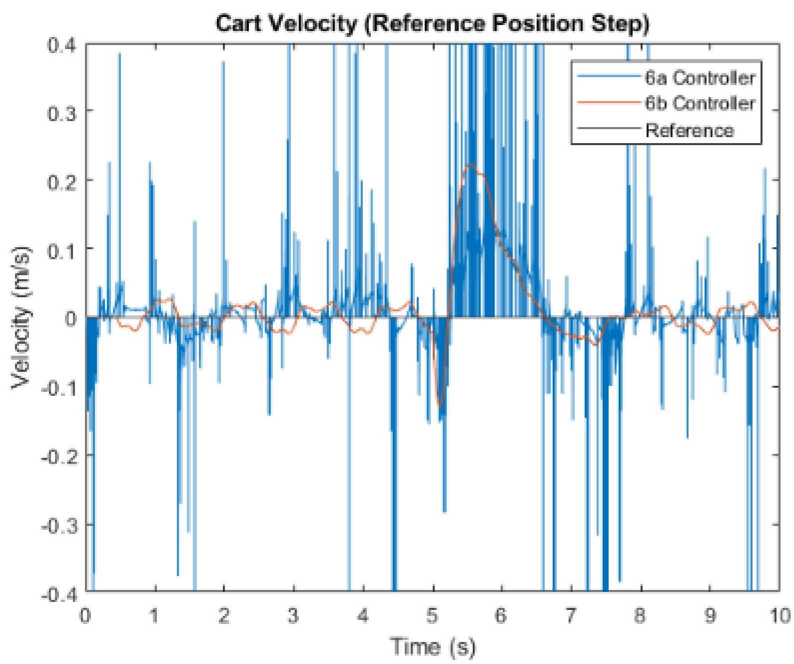


**Figure 30.** Cart velocity plots for the step reference case **(**Zoomed in).

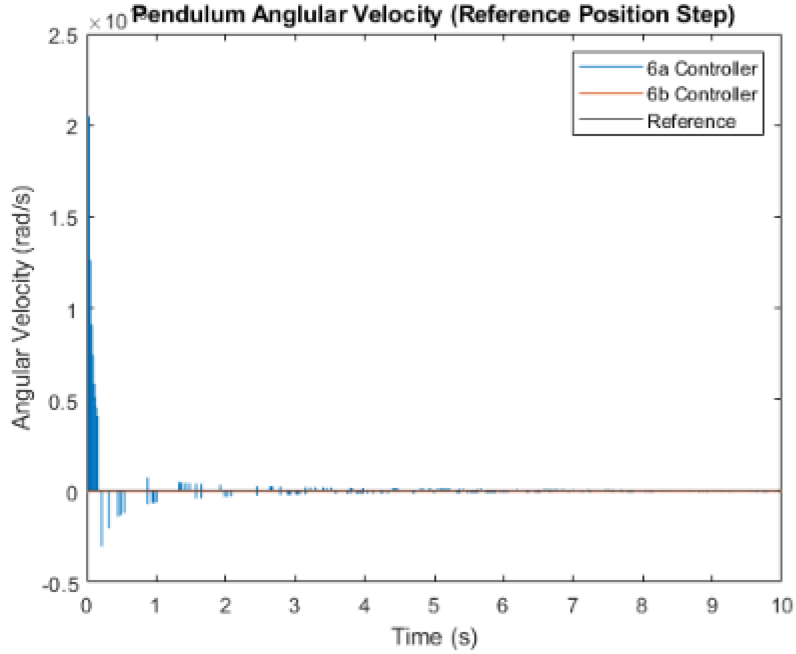


**Figure 31.** Pendulum angular velocity plots for the step reference case.

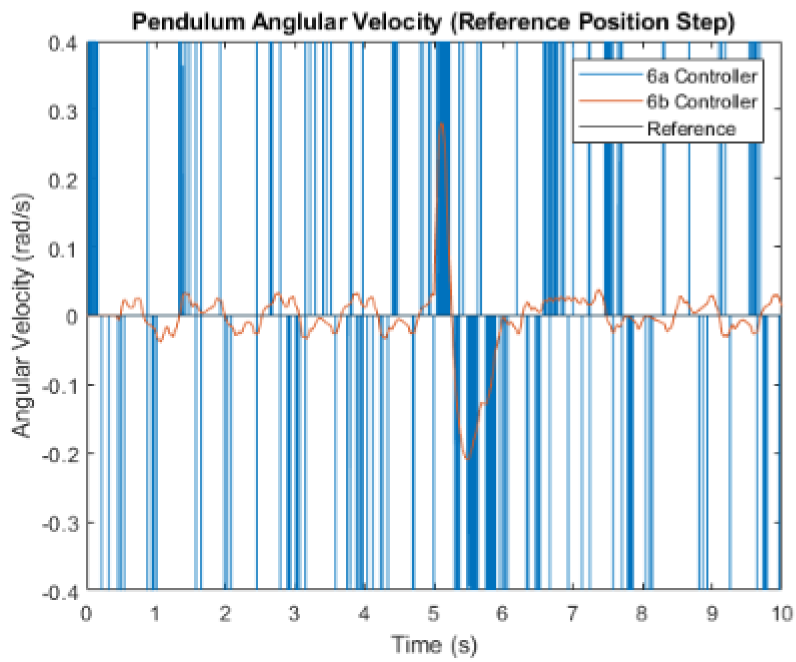


**Figure 32.** Pendulum angular velocity plots for the step reference case **(**Zoomed in).

### 4.3.3

```
clear; clc; close all;
A = [0 1 0 0; 0 -6.81 -1.50 0; 0 0 0 1; 0 15.47 25.68 0];
B = [0 ; 1.52 ; 0 ;-3.46];
C =[1 0 0 0; 0 0 1 0];
p = [-2.0+10j -2.0-10j -1.6+1.3j -1.6-1.3j];
K = place(A,B,p);
pobs = [-10 + 15j -10 - 15j -12 + 17j -12 - 17j];
L = (place(A',C',pobs))';
tstop = 10;
outob = sim('statespaceobserverh',tstop);
outss = sim('cartpendulum1',tstop);

xhat = outob.xhat.data;
velocity_ob = xhat(:,2);
velocity_ss = outss.xdot.data;

thetadot_ob = xhat(:,4);
thetadot_ss = outss.thetadot.data;
t = outob.tout;
t2 = outss.tout;

plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Sine Wave)')
figure
plot(t2,velocity_ss)
hold on
plot(t,velocity_ob)
hold on
yline(0)
ylim([-0.4 0.4])
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Velocity (m/s)');title('Cart Velocity (Reference Position Sine Wave)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Sine Wave)')
figure
plot(t2,thetadot_ss)
hold on
plot(t,thetadot_ob)
hold on
yline(0)
ylim([-0.4 0.4])
legend('6a Controller','6b Controller','Reference')
xlabel('Time (s)');ylabel('Angular Velocity (rad/s)');title('Pendulum Anglular Velocity (Reference Position Sine Wave)')
```

**Figure 33.** MATLAB script used to generate the cart velocity and angular velocity plots for the sinusoidal reference case.

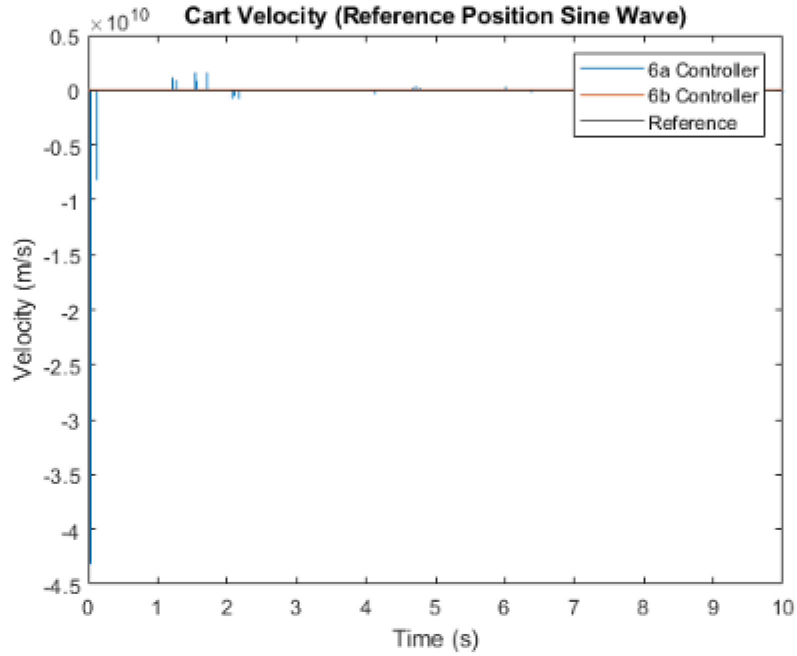


**Figure 34.** Cart velocity plots for the sinusoidal reference case.

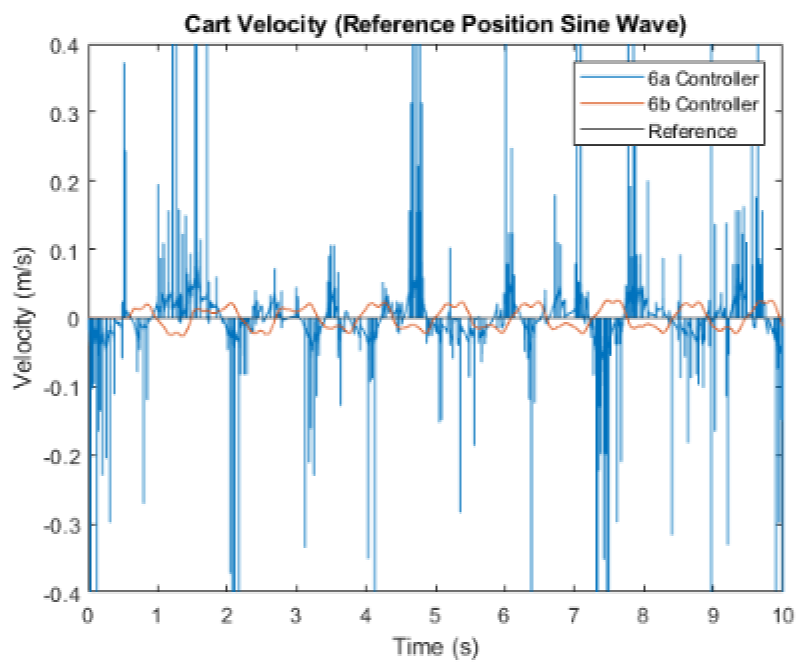


**Figure 35.** Cart velocity plots for the sinusoidal reference case **(**Zoomed in).

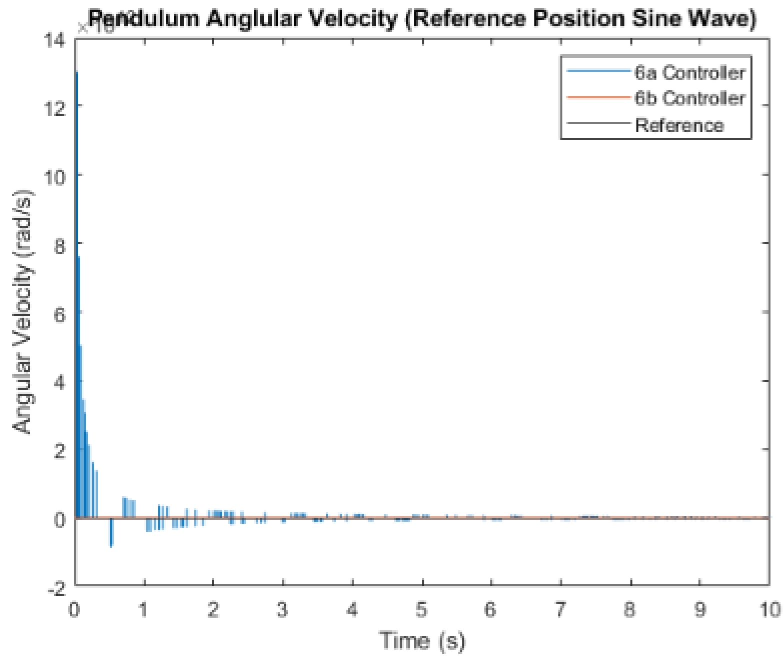


**Figure 36.** Pendulum angular velocity plots for the sinusoidal reference case (Zoomed

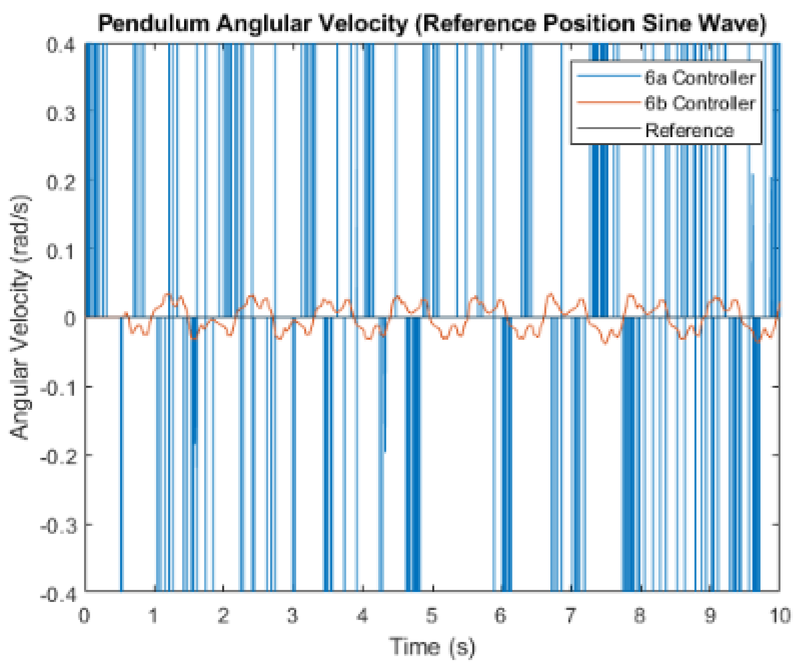


**Figure 37.** Pendulum angular velocity plots for the sinusoidal reference case (Zoomed in).

4. Which scheme do you think gives the "better" performance, and more importantly, why? There is no definite answer here. Just form your own opinion and defend it.

**The scheme from this lab performs far better than the scheme from the last lab. The Luenberger observer does a far better job at minimizing the effects of noise/perturbations on its estimations when compared to the differentiation method used in the previous lab. In observer design, despite there being a discrepancy between the actual and estimated state, after some time the estimate will eventually match the actual in time. The error between actual and estimated states can be controlled by adding a gain L to the actual and estimated outputs, where gain L can be designed to get quick convergence. The observer design will be less sensitive to noise than the model without the observer, especially when it comes to estimating the velocity and angular velocity.**