EE128 Lab6 Writeup

Chris Mitchell, Shangge Liu

October 29th, 2018

Contents

1 Purpose

The goal of this lab is to implement a basic state-space controller to balance a pendulum with the aim of eventually using this controller in order to program a self-erecting inverted pendulum.

2 Prelab

2.1 Equations of Motion of the Mechanical System

$$-m\ddot{x}L_p cos\theta + mgL_p sin\theta = \frac{4}{3}mL_p^2\ddot{\theta} = m\ddot{x} + mgL_p\theta \tag{1}$$

$$\Rightarrow mL_p\ddot{x} + \frac{4mL_p^2}{3}\ddot{\theta} - mgL_p\theta = 0 \tag{2}$$

$$N - m\ddot{x} = mL_p\ddot{\theta} \tag{3}$$

$$F_a = N + M\ddot{x} \tag{4}$$

$$F_a = (m+M)\ddot{x} + mL_p\ddot{\theta} \tag{5}$$

2.2 Full System Dynamics of Linearized System

$$F_a = \frac{rK_gK_tV - K_g^2K_tK_m\dot{x} - K_g^2J_mR_m\ddot{x}}{r^2R_m} = (m+M)\ddot{x} + mL_p\ddot{\theta}$$

(6)

$$mL_p\ddot{x} + \frac{4mL_p^2}{3}\ddot{\theta} - mgL_p\theta = 0 \tag{7}$$

After combining these two equations and calculating, we got:

$$\ddot{\theta} = \frac{-K_g^2K_tK_m\dot{x} - [K_g^2J_mR_m + (m+M)r^2R_m]g\theta + rK_gK_tV}{-\frac{4}{3}K_g^2J_mR_mL_p - \frac{4}{3}(m+M)r^2R_mL_p + mr^2L_p} = 14.47\dot{x} + 25.67\theta - 3.46V \tag{8}$$

$$\ddot{x} = g\theta - \frac{4}{3}L_p\ddot{\theta} = -6.81\dot{x} - 1.5\theta + 1.52V \tag{9}$$

Therefore,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -6.81 & -1.5 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 15.47 & 25.67 & 0 \end{bmatrix} \tag{10}$$

$$B = \begin{bmatrix} 0 \\ 1.52 \\ 0 \\ -3.46 \end{bmatrix} \tag{11}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{12}$$

$$D = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{13}$$

2.3 Analysis and Controller Design

2.3.1 Eigenvalues and BIBO Stability

eig(A)=0 4.8665 -7.5504 -4.1260,which means:

$$\Delta(s) = s(s - 4.8665)(s + 7.5504)(s + 4.1260) \tag{14}$$

From that, we can see that it is neither internally stable nor BIBO stable as there is a pole in the RHP.

2.3.2 Simulated Output Response

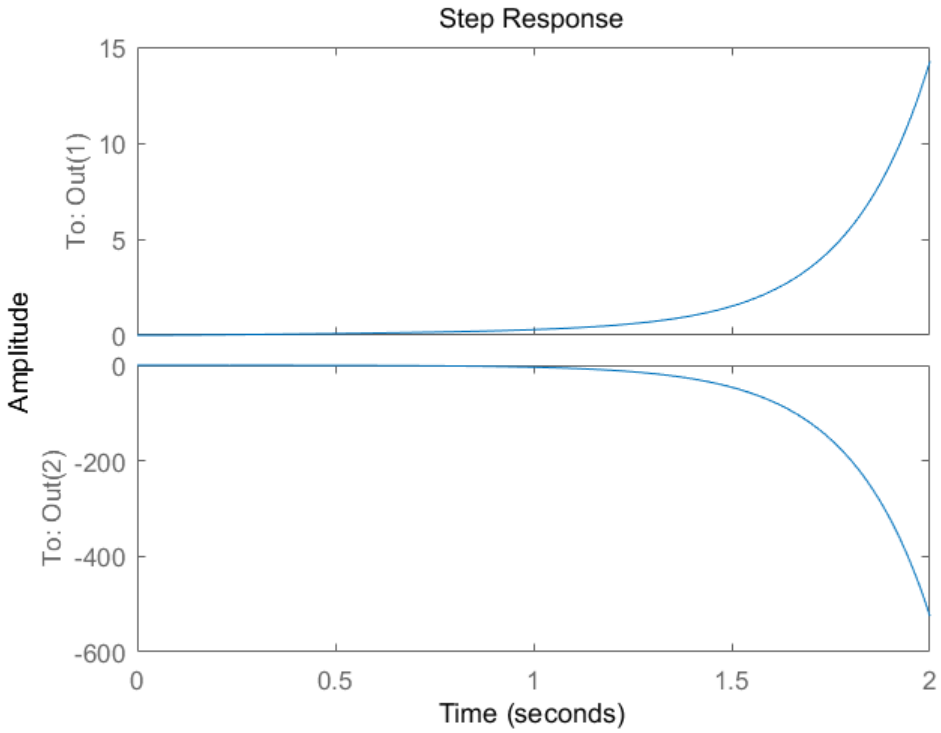

Figure 1: the step response of the original system

From the plot, we can see that the x and angle rise exponentially. This supposes an infinite track and infinite input power, both of which we do not have. This also avoids the equilibrium points of $sin\theta$ due to our linearization of the system. In reality, after getting to a certain point the velocity and angular velocity will become a constant. It is because as the velocity increases, the power the voltage source provide will increase as well and finally, after the sources reaches

its highest power, the two velocities will stop increasing. With the finite track constraint, the cart will just slam into the end of the track.

2.3.3 Feedback Controller

A_k

$$A_k = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1.52k_1 & -6.81 - 1.52k_2 & -1.5 - 1.52k_3 & -1.52k_4 \\ 0 & 0 & 0 & 1 \\ 3.16k_1 & 3.16k_2 + 15.47 & 3.16k_3 + 25.67 & 3.16k_4 \end{bmatrix} \quad (15)$$

Characteristic Polynomial for A_k

$$P(k;s) = det(sI - A_k) = s^4 + (1.52k_2 - 3.46k_4 + 6.81)s^3 + (1.52k_1 - 3.46k_3 - 0.0482k_4 - 25.67)s^2$$

$$+(\text{-}0.0482\text{k}_3 - 151.6077 - 33.8284k_2)s - 33.8284k_1 (16)$$

Desired Characteristic Polynomial

$$P_d = (s+2+10j)(s+2-10j)(s+1.6+1.3j)(s+1.6-1.3j) = s^4 + 7.2s^3 + 121.05s^2 + 349.8s + 442 \quad (17)$$

K-Values By solving the equation P(k;s)=P_d, we got the K values:

$$K = [\ -13.06594459 \quad -14.75362592 \quad -48.05271711 \quad -6.594078438\ ] \quad (18)$$

Verification of Results We proved our K values to be right by the code in the Appendix.

2.4 Closed Loop Transfer Function and Bode Plot

We got the transfer function by the code in the Appendix.

$$tf = \frac{-19.8602s^2 + 442}{s^4 + 7.2s^3 + 121.05s^2 + 349.8s + 442} \quad (19)$$

Here is the Bode diagram.

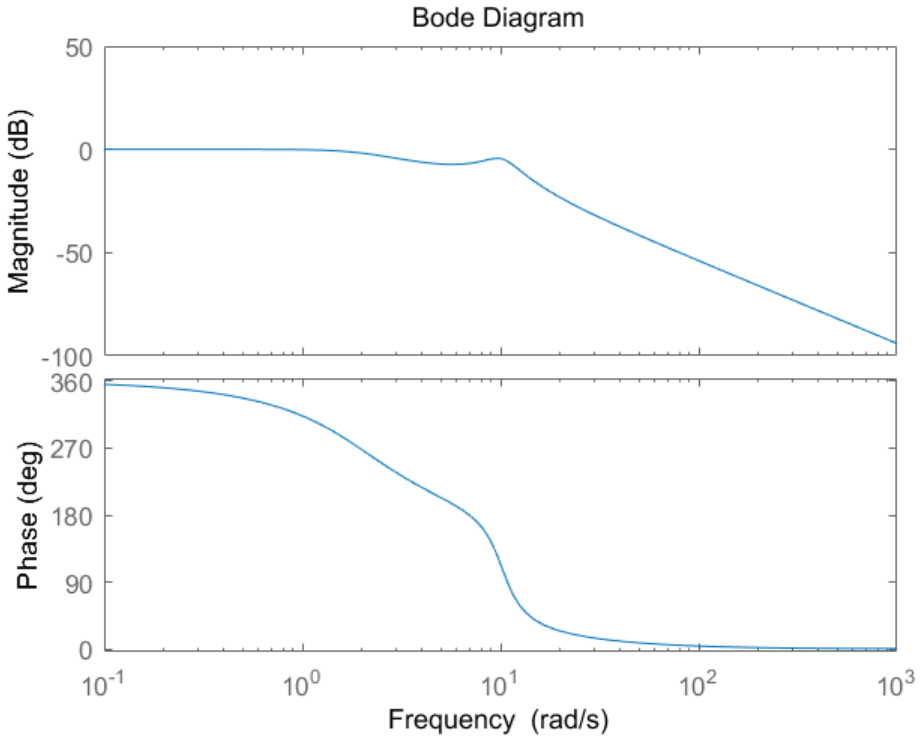

Figure 2: Bode diagram of the closed loop system

3 Lab Results and Analysis

3.1 Implementing the Controller in Simulink

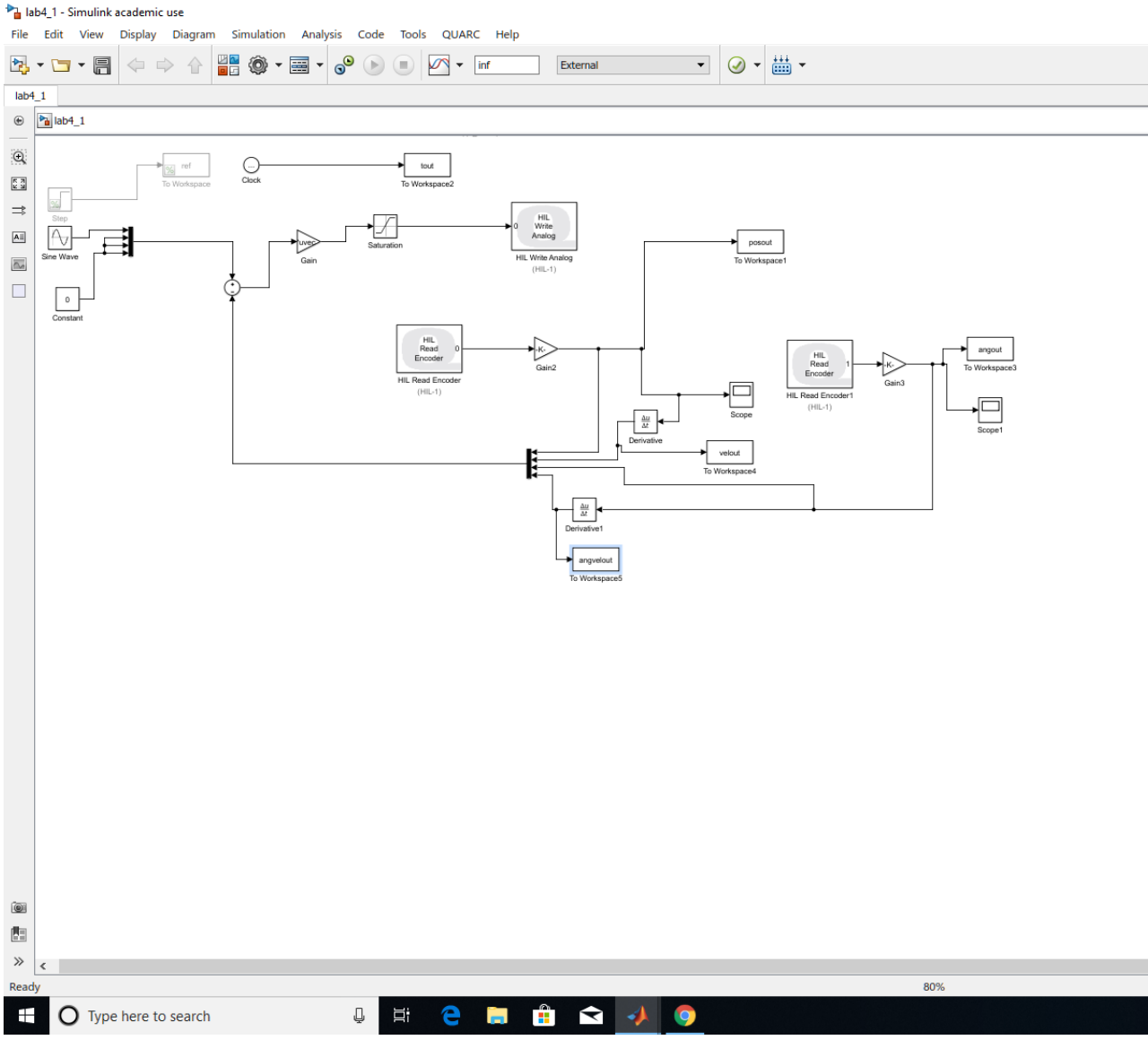

Figure 3: Hardware Model in Simulink

3.2 Running the Controller on Hardware

3.3 Running with Zero Input and with Perturbations

If you were to connect the pendulum to the hardware in it's stable equilibrium position, nothing would happen.

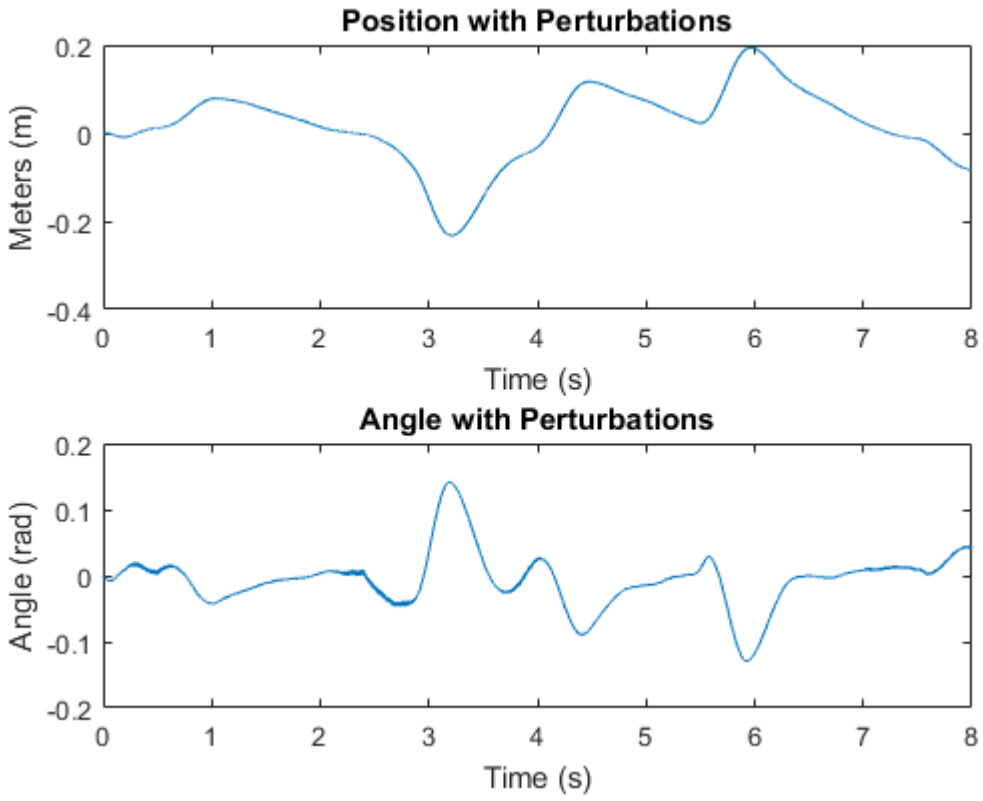

Figure 4: Plot of Position and Angle over Time with Perturbations

From the graph, we can see that when the angle is perturbed in the positive direction (represented by a spike on the angular portion of the graph), the cart compensates by moving in the negative x direction, and vise-versa. This makes sense due to equation (9), which shows that the acceleration of the cart is inversely proportionate to the angle of the cart. After such a positive perturbation of angle, the controller kicks in and moves the cart in the positive direction to compensate for and counteract the change of angle.

The controller generally appears to work well as it's able to correct perturbations, since we have a finite track length, there may be some issues if there

are too many perturbations in a short time span.

3.4 Sine Wave Inputs

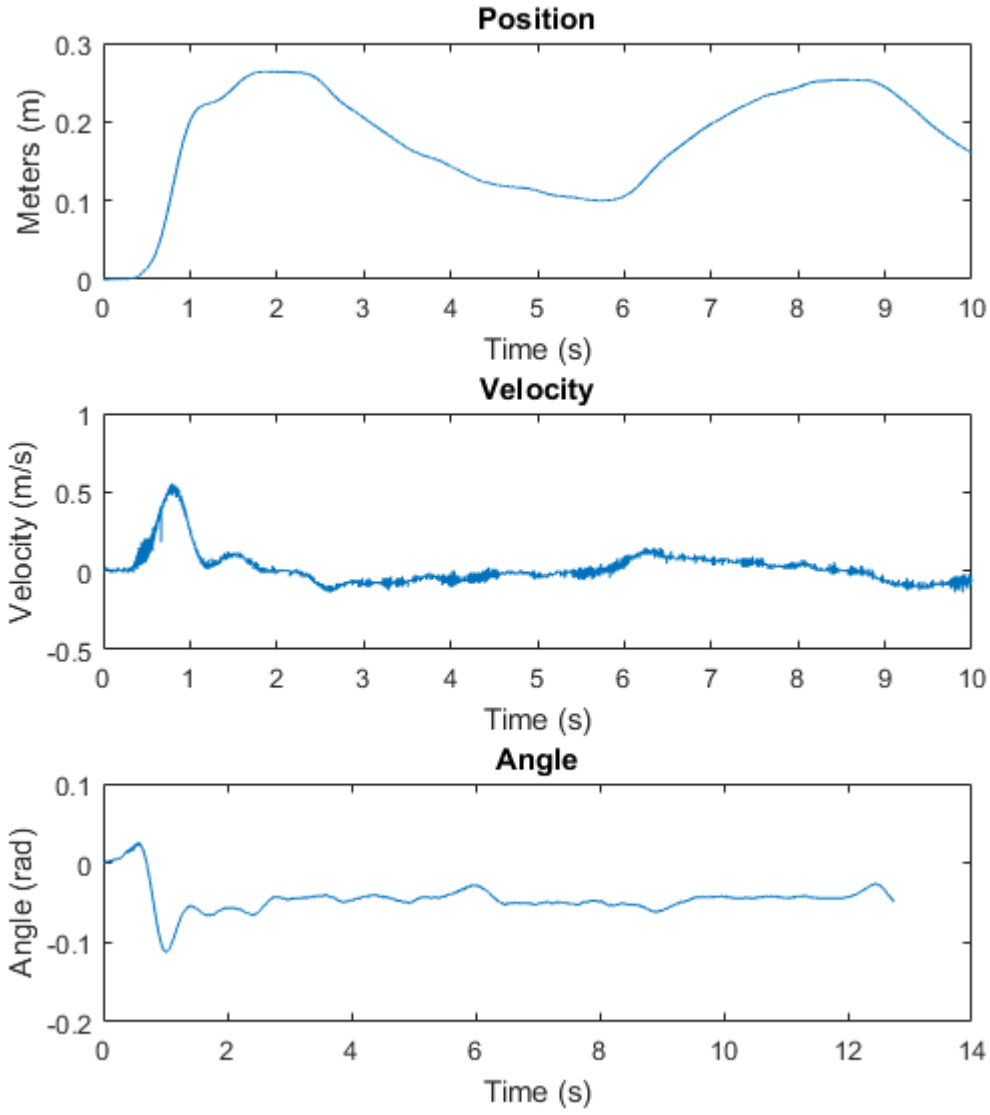

Figure 5: Plot of Response to $0.05sin(t)$ input

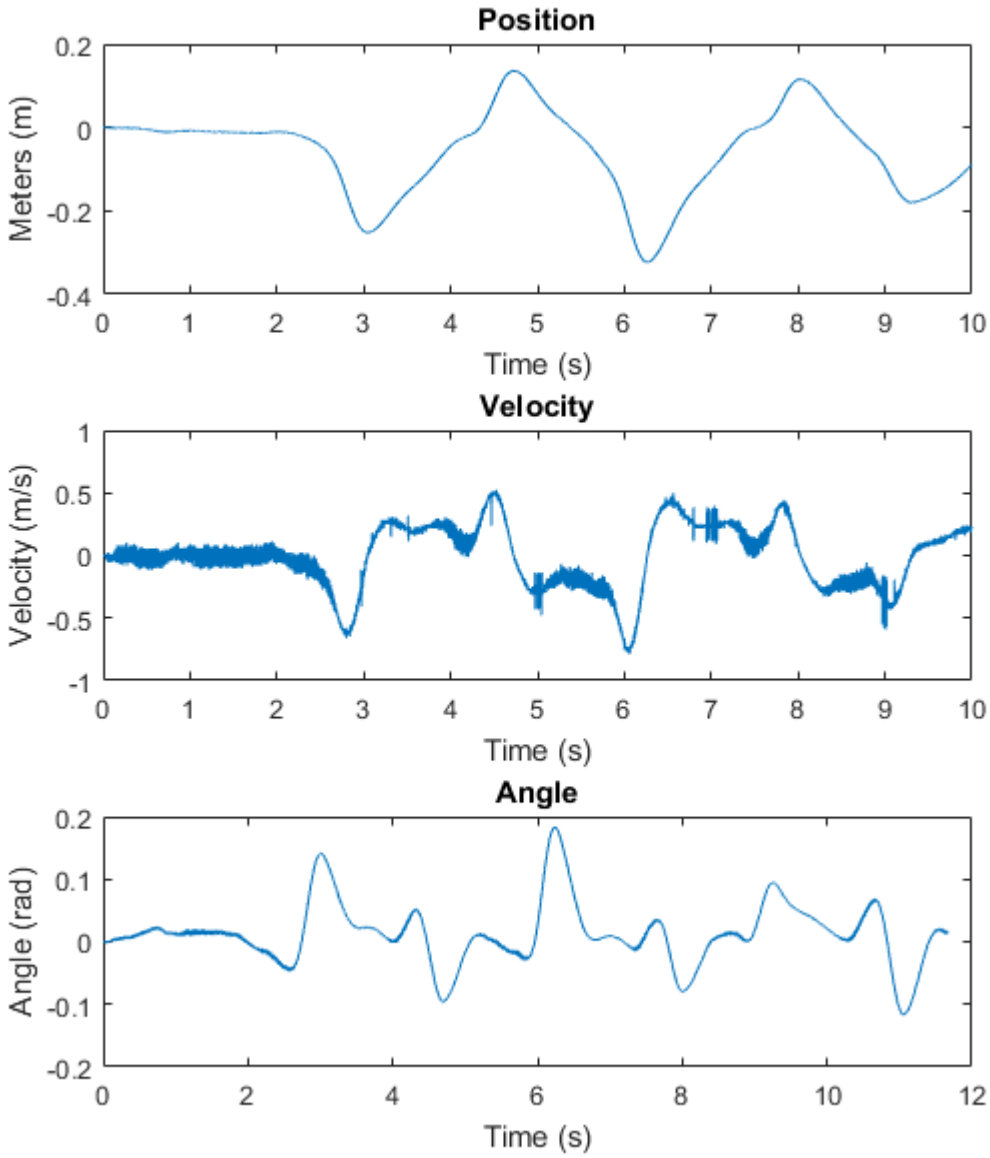

Figure 6: Plot of Response to $0.05sin(2t)$ input

Unfortunately, due to human error we realized after finishing this lab that our $0.05sin(5t)$ graph was not properly saved. However, we remember enough of

the lab results to still be able to continue analysis.

3.4.1 Gain and Phase of Frequency Responses

For $\omega = 1$, the bode plot shows the magnitude around 0 (AKA gain of 1) and the phase around 300 °or 5.2 rad. For $\omega = 2$, we see a similar gain and a slight drop in phase. However, for $\omega = 1$ we see an amplitude of roughly .9 (meaning a gain of roughly 2 due to input amplitude of 0.05), and for $\omega = 2$ we see an amplitude of .2 (meaning a gain of 4). For the phase, we see a phase shift of .5s and 1.5s respectively, which means for 1rad/s and 2rad/s, we have actual phase shifts of .5 rad and 3 rad. This is far from what we expected based on the bode plot. We expected no noticeable change in amplitude and much larger phase shifts. This could be due to experimental error, the imperfectness of the encoders, the nonlinearities, and the assumption of lack of friction.

3.4.2 Changing the Poles

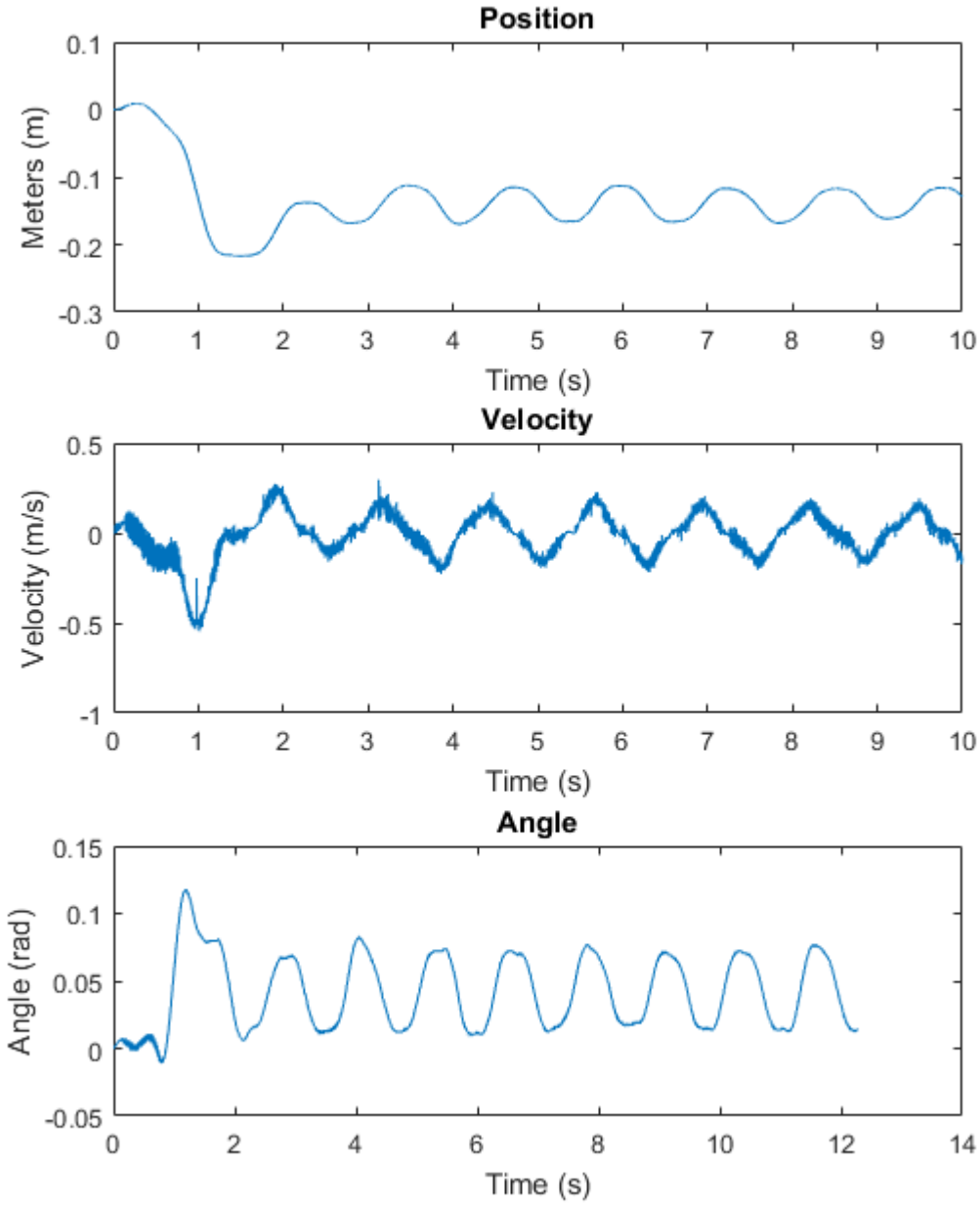

Figure 7: Plot of Response to $0.05sin(5t)$ Input with Larger Imaginary Component of Poles

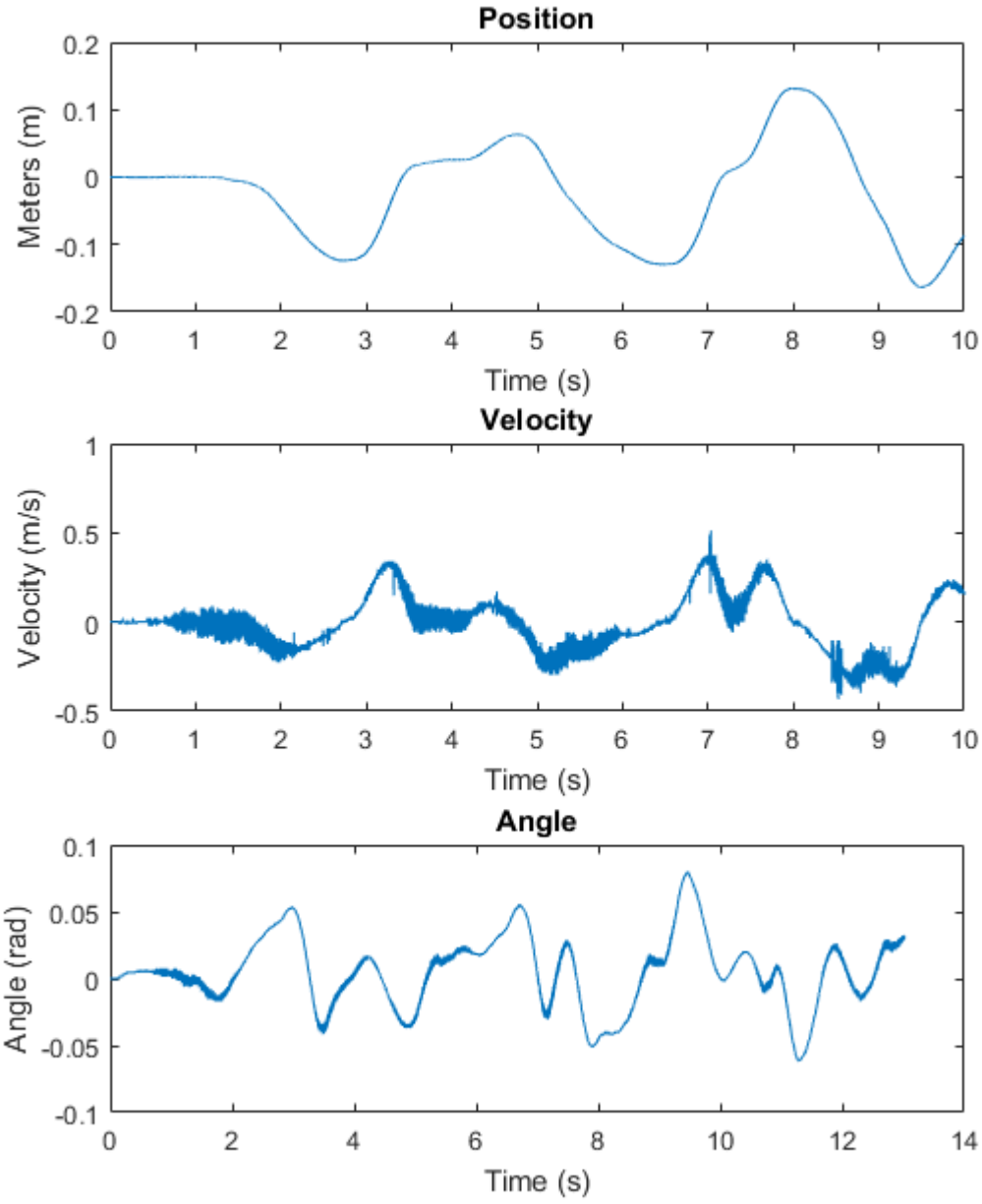

Figure 8: Plot of Response to $0.05sin(5t)$ Input with Smaller Imaginary Component of Poles

For pole manipulation, we focused on how changing the imaginary component of poles changes the system. Specifically, we changed the $2.0 \pm 10.0j$ to $2.0 \pm 12.0j$ and $2.0 \pm 8.0j$. For the case with larger imaginary input we see higher frequency waves for the position and angle. For the case with smaller imaginary input, we see a mixing of multiple waves, with a clear lower frequency component not present in the larger imaginary component graph.

3.5 Angular and Linear Velocities

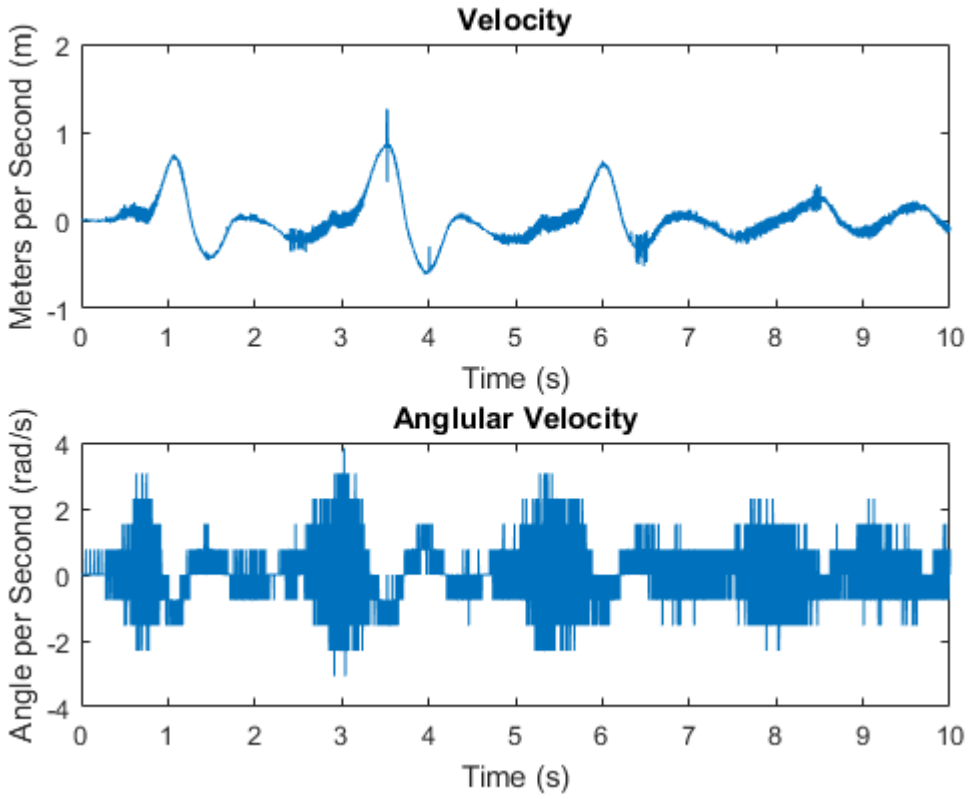

Figure 9: Plot of Numerical Derivatives-Linear and Angular Velocity

Both velocities have poor quality, with the angular velocity being significantly worse. This is why taking numerical derivatives should be avoided, especially since we are feeding these values back into the system.

Appendices

A Code for Prelab

```
M = .94;
m = .23;
Lp = .3302;
Ic = m*Lp*Lp /3;
Ie = 4*m*Lp*Lp/3;
Kt = 7.67e-3;
Km = 7.67e-3;
Kg = 3.71;
Rm = 2.6;
r = 6.36e-3;
Jm = 3.9e-7;


a22 = -6.81; % -6.81
a23 = -1.5; %-1.5
a42 = 15.47; %15.47
a43 = 25.67; %25.67

b2 = 1.52; %1.52
b4 = -3.46; %3.46

A = [0 1 0 0; 0 a22 a23 0; 0 0 0 1; 0 a42 a43 0];

B = [0; b2; 0 ; b4];

C = [1 0 0 0; 0 0 1 0];
D = [0; 0];

sys = ss(A, B, C, D);
step(sys)

%eig(A)=0 4.8665 -7.5504 -4.1260


K = sym('k', [1 4]);


Ak = A - B*K;
```

```
p = [-2+10j, -2-10j, -1.6-1.3j, -1.6+1.3j];
place(A, B, p);
%K=[ -13.0659  -14.7536  -48.0527   -6.5941]

K1=[-13.0659  -14.7536  -48.0527   -6.5941]
C1=[1 0 0 0]
D1=[0 0 0 0]
sys2=ss(A-B*K1,B*K1,C1,D1)
[num,den]=ss2tf(A-B*K1,B*K1,C1,D1,1)
%num =0         0  -19.8602   -0.0000  441.9985
%den = 1.0000    7.2001  121.0500  349.7991  441.9985

figure
bode(num,den)
```

A.1 Code for Perturbation Graph

```
subplot(2,1,1);
plot(tout, posout);

title('Position with Perturbations');
xlabel('Time (s)');
ylabel('Meters (m)');
xlim([0 8]);


subplot(2,1,2);
plot(tout, angout);

title('Angle with Perturbations');
xlabel('Time (s)');
ylabel('Angle (rad)');

xlim([0 8]);
```

A.2 Code for Plotting Position, Velocity, and Angle

```
subplot(3,1,1);
plot(tout, posout);

title('Position');
xlabel('Time (s)');
ylabel('Meters (m)');
xlim([0 10]);
```

```
subplot(3,1,2);
plot(tout, velout);

title('Velocity');
xlabel('Time (s)');
ylabel('Velocity (m/s)');

xlim([0 10]);

subplot(3, 1, 3)
plot(tout, angout)
title('Angle')
xlabel('Time (s)');
ylabel('Angle (rad)');
```

A.3 Code for Bode Plot

```
prelab_original;

K = [-13.0659  -14.7536  -48.0527   -6.594];
Ak = A-B*K;
Bk = B*K;
D = zeros(2,4);
sys = ss(Ak, Bk, C, D);
bode(sys(1,1));
```

A.4 Code for Angular and Linear Velocity Plots

```
subplot(2,1,1);
plot(tout, velout);

title('Velocity');
xlabel('Time (s)');
ylabel('Meters per Second (m)');
xlim([0 10]);


subplot(2,1,2);
plot(tout, angvelout);

title('Anglular Velocity');
xlabel('Time (s)');
ylabel('Angle per Second (rad/s)');

xlim([0 10]);
```