EE128 Lab 6d: Self-Erecting Inverted Pendulum

Lucine Oganesian, Edward Lai, Hillary Chen
Lab Section 104

December 23, 2015

1 Purpose

From the lab: "Design a controller that starts with the pendulum in the down position and then swings it into and maintains it in the up position."

2 Prelab

a. $E(\theta, \dot{\theta}) = mgL_p\cos(\theta) + \frac{1}{2}J\dot{\theta}^2 - mgL_p$

b. $E(\pi, 0) = mgL_p\cos(\pi) + 0 - mgL_p = -2mgL_p$

c. $\frac{\mathrm{d}E}{\mathrm{d}t} = -mgL_pg\sin(\theta)\dot{\theta} + J\dot{\theta}\ddot{\theta} = \dot{\theta}(-mgL_p\sin(\theta) + J\ddot{\theta})$.
From the dynamics relationship provided:

$\frac{\mathrm{d}E}{\mathrm{d}t} = \dot{\theta}(-mL_p\cos(\theta)\ddot{x}) = -mL_p\dot{\theta}\ddot{x}\cos(\theta)$

d. At the natural frequency $\omega = \sqrt{\frac{g}{l}}$ we have: $H = \frac{1}{c_{lp}\sqrt{\frac{g}{l}}j+1}$. The magnitude of the phase of this is $\left|-arctan(c_{lp}\sqrt{\frac{g}{l}})\right| \leq 5$ which will yield that $\sqrt{\frac{l}{g}}\tan(5) \geq c_{lp}$. $l = L_p = 0.3302m, g = 9.8m/s$

Thus, $c_{lp} = 0.01606$

e. For full simulink model, please see the lab section.

3 Lab

a. Simulink diagrams See Appendix

b. Self-erecting Pendulum
We initially implemented a linear system for keeping the pendulum inverted using state-feedback with a gain matrix defined as $K = [-13.0323 \ -14.7880 \ -48.1644 \ -6.6228]$ and a luenberger controller, with a L matrix defined as:

$$L = \begin{bmatrix} 16.1595 & -2.3767 \\ 254.9780 & -5.4829 \\ 15.7136 & 21.0282 \\ 180.7734 & 378.2117 \end{bmatrix}$$

This system would kick in once our pendulum was over a threshold of 10 degrees within vertical (0 degrees). Until then, an energy pumping 'bang bang' controller was used to lift the pendulum within the required threshold.
Figure 4, shows the entire connection of our controller. The subsystem IPhardware (figure 5) corresponds to our interface with the cart system. Here we read values from the encoder and converted them to relevant units of position and angle. The position conversion was a multiplication with $\frac{1}{43958}$ and the angle conversion was a multiplication with $\frac{-1}{859.1}$. Because the observer is not active when the swing up controller is active, there is no estimate of velocity and angular velocity, and thus these values must be calculated from the encoder readings (see figure 5), using a differentiator multiplied by a low pass filter ($c_{lp} = 0.0160593$). The angle reading must also be converted to make sense in the context of our energy pumping system, using the custom MATLAB function block (listing 1):

```
1 function theta = angleMod(theta_encode)
2 %#codegen
3 theta = mod(theta_encode, 2*pi) - pi;
4
5 end
```

Listing 1: angleMod()

The saturation block before the analog write block is a custom saturation block defined as such (listing 1):

```
1 function V_sat = fcn(xdot, V)
2 % Setting the bounds
3 max_lim = min(8,max(4,10*xdot+5));
4 min_lim = min(-4,max(-8,10*xdot-5));
5 %Dynamic Saturation
6 V_sat = max(min_lim,V);
7 V_sat = min(max_lim,V_sat);
8 end
```

Listing 2: saturation function

The next subsystem of interest is the mode selector (figure 6). The mode selector serves to pick whether to use the swing-up controller or the linear observer controller. This is done by looking at the angle position reading and comparing it to our threshold.

Figure 7 depicts our linear control system, with the gain (K) and L values as stated above. Our state space representation of the plant model had the following A, B, C, and D matrices:

$$A = \begin{bmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -6.8123 & -1.4968 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 15.4731 & 25.6749 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1.5226 \\ 0 \\ -3.4583 \end{bmatrix}$$

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

$$D = 0$$

Figure 8 shows the bang bang controller which outputs $V_{max} = +6V$ when the energy of the system is less than zero (i.e. not inverted) and when $\dot{\theta}cos(\theta) < 0$, $-V_{max} = -6V$ when the energy of the system is less than zero and $\dot{\theta}cos(\theta) > 0$, and 0 when the energy of the system is greater than or equal to 0 (achieved vertical state). We used a Kpump gain of 50. The SAT block (equal to 0.5) seen in figure 4 was used to set the initial direction of the cart before the bang bang controller kicked into effect (50 ms).

Lastly note the delay block that is highlighted in blue in figure 1. This is used to delay the output of the linear controller to the system until the observer has had enough time to converge to the actual values. We used a delay of 0.25 s. (The reference used was [0 0 0 0]. Everything else was kept the same.

3.1 Improving the controller

In order to improve performance with rebalancing the pendulum after it was knocked down, a LQR controller was implemented, with the following Q (state cost) and R matrices (controller action cost):

$$Q = \begin{bmatrix} 80.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 400.0000 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R = 0.0194$$

The resulting K and L matrices were:

$$K = \begin{bmatrix} -64.1427 & -56.7426 & -214.7453 & -33.5827 \end{bmatrix}$$

$$L = \begin{bmatrix} 41.6039 & 0.6927 \\ 541.4581 & 15.8748 \\ 16.2986 & 51.5838 \\ 760.8475 & 976.5979 \end{bmatrix}$$

These values were chosen such that the system would try to prioritize balancing the pendulum (q3 affects angle) then the position (q1) while keeping the input voltage to a minimum (low r). The SAT value was also changed to be 0.01 and Kpump gain was changed to 80. Everything else was kept the same as before. The updated controller did a better job with rebalancing the pendulum after it was knocked down.

3.2 Possible future improvements

The system seemed to fail to rebalance if the energy of the system started changing a lot (which would happen sometimes when the pendulum would be knocked down, see LQR energy plot below, and the behavior of the energy values at the end of the time period). Additional kinetic energy seems to make it difficult for the switch between swing-up and linear controller to happen successfully. Currently we are not sure of how best to address this issue. Lastly, a modified swing up controller might be necessary, such that it takes better account of the position of the cart on the track with respect to the reference.
The linear system also seemed to have a small steady state error, which might be worth trying to address

by increasing the system type (maybe by adding an integral before the gain matrix). This way the cart will consistently come back to center, thus making it less likely to crash into the walls of the track if it needs to rebalance the pendulum. Also the observer poles need to be moved further to the left (to be 5x away from the system poles).

3.3 Video of self inverting pendulum

To view the final pendulum in action: https://youtu.be/W3v7OwX6yGE.

4 Appendix

Simulink Models

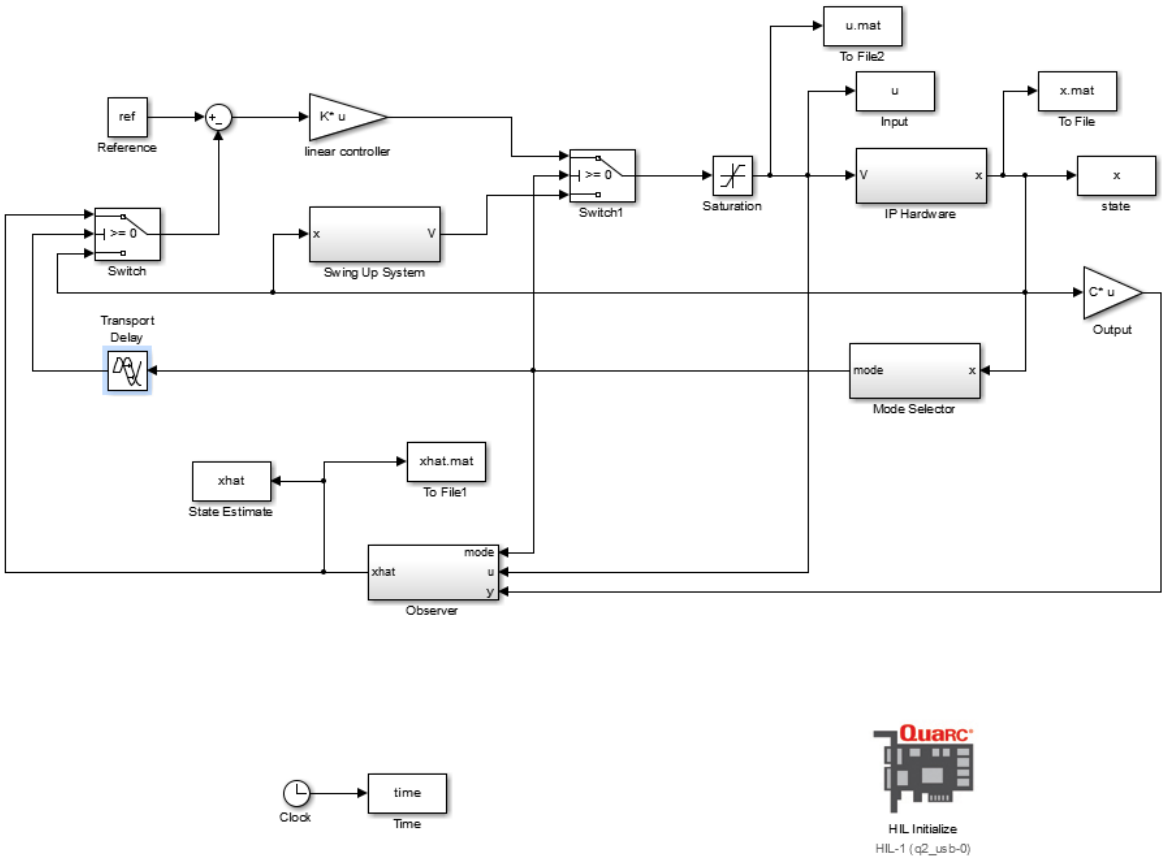

Figure 1: Full simulink model of entire system

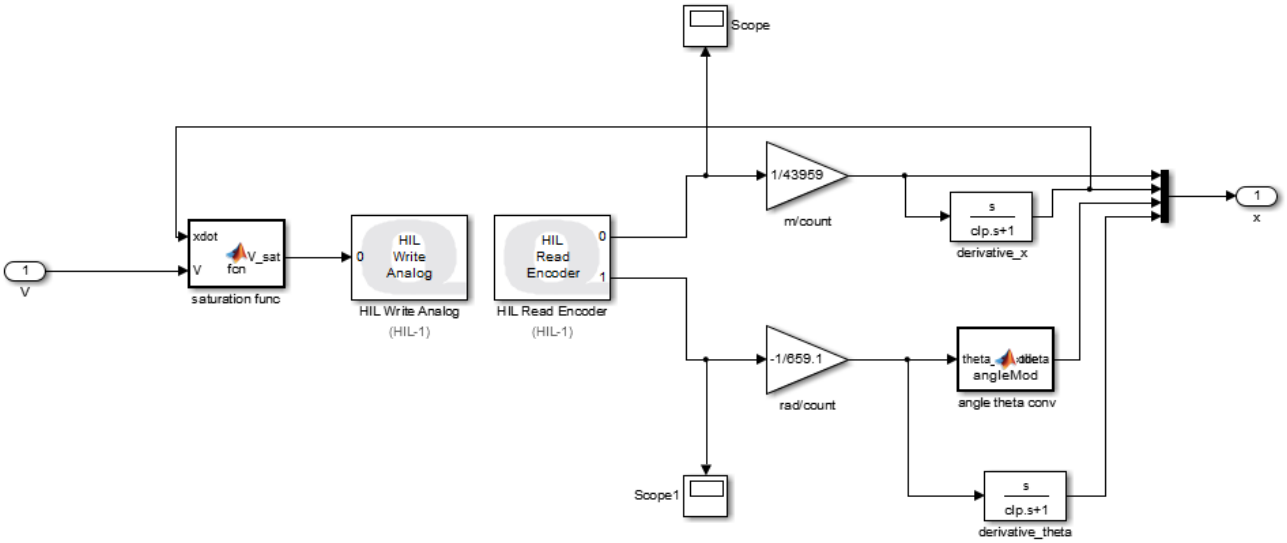

Figure 2: Simulink model of input (output) to (from) hardware

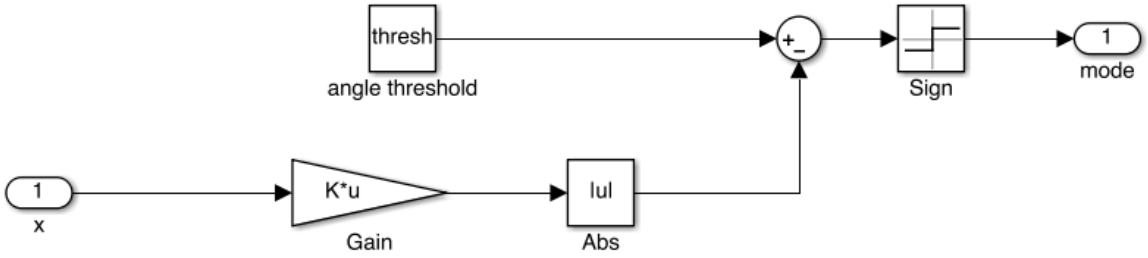

Figure 3: Simulink model of mode selector (swingup controller or linear controller selection)

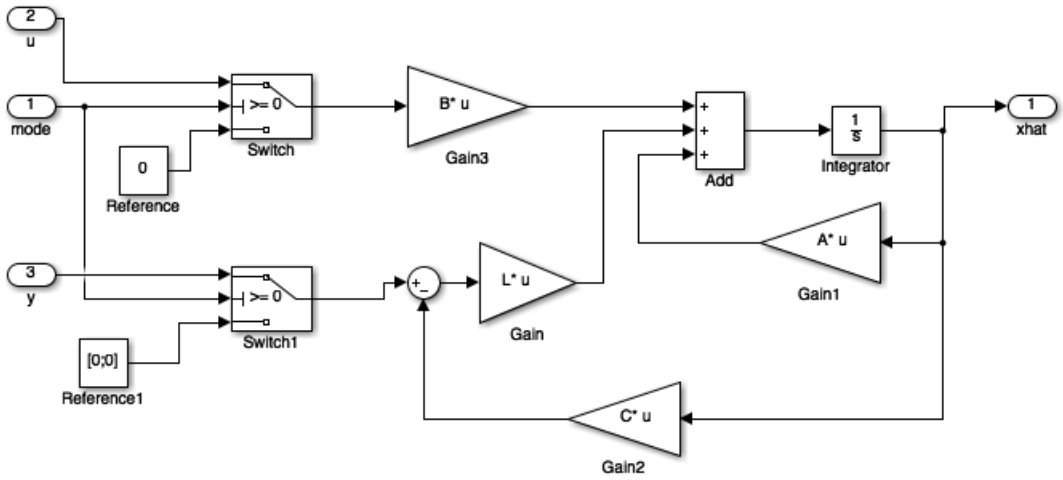

Figure 4: Simulink model of linear controller

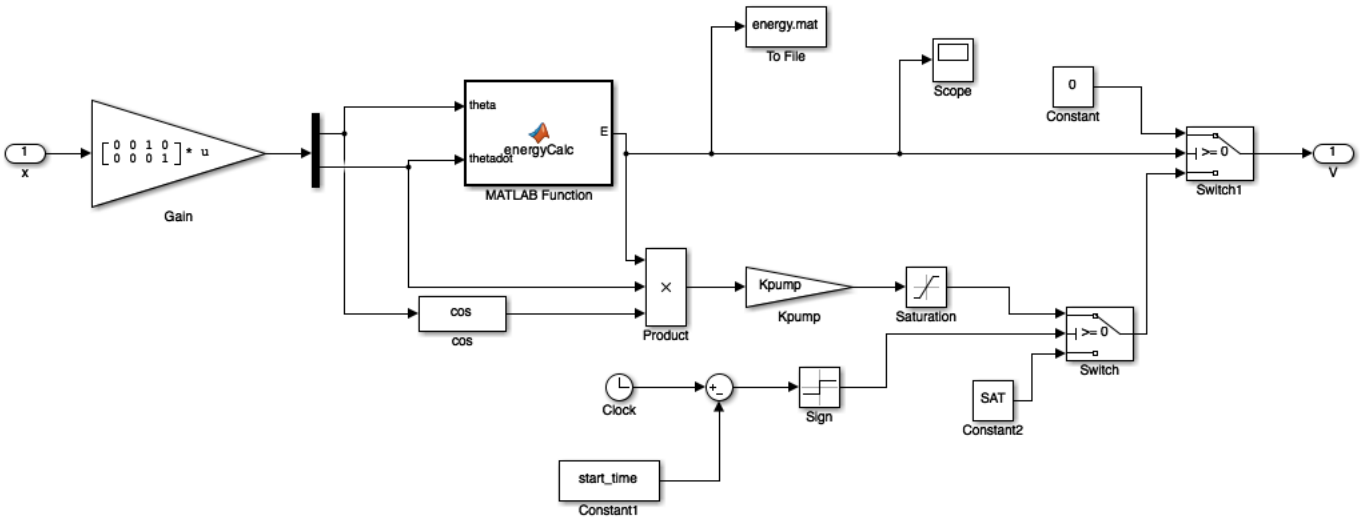

Figure 5: Simulink model of swingup, 'bang bang' controller

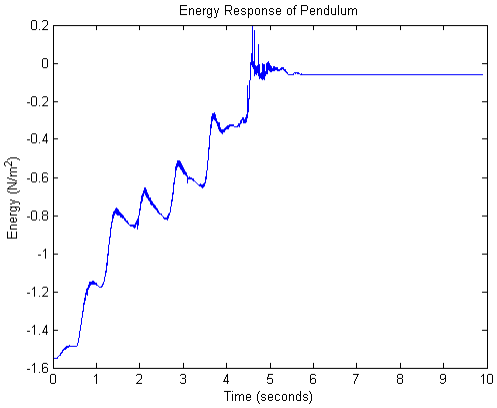

Figure 6: Graph of energy response for pendulum and cart while balancing

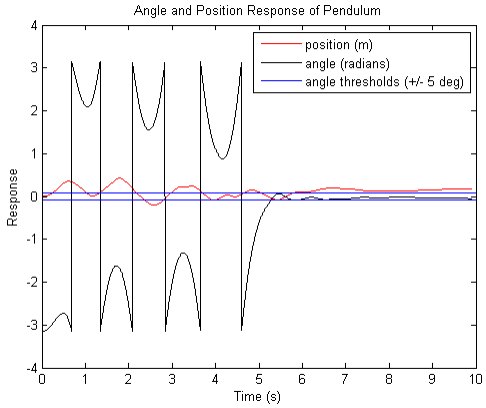

Figure 7: Graph of position and angle response for pendulum and cart while balancing, with angle thresholds for reference

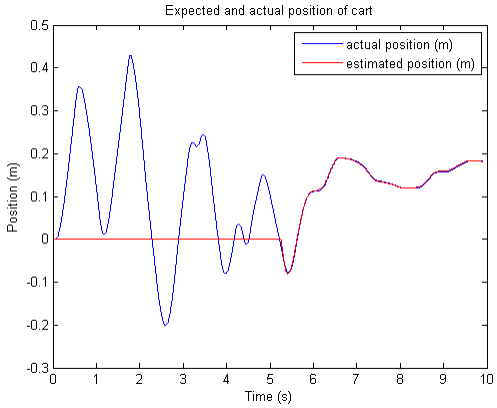

Figure 8: Position and position estimate for the system

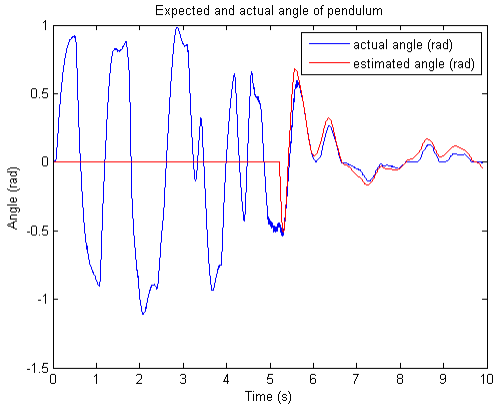

Figure 9: Angle and angle estimate for the system

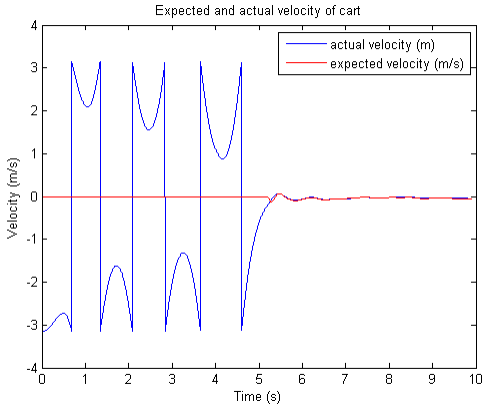

Figure 10: Velocity and velocity estimate for the system

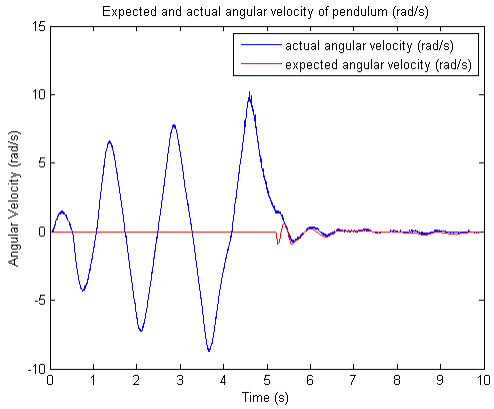

Figure 11: Angular velocity and angular velocity estimate for the system

The following script was used when running the simulations and calculating step response information in the prelab.

```
%% Lab 6D
clc
close all
clear all
```

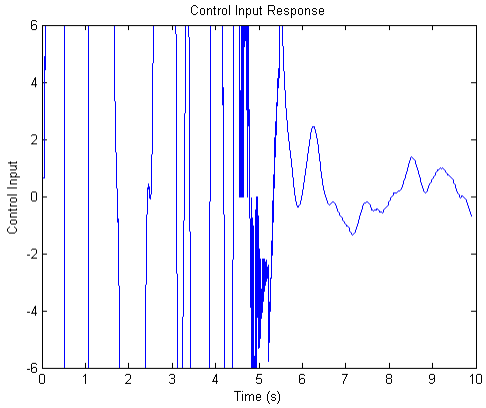

Figure 12: Graph of control input for pendulum and cart while balancing

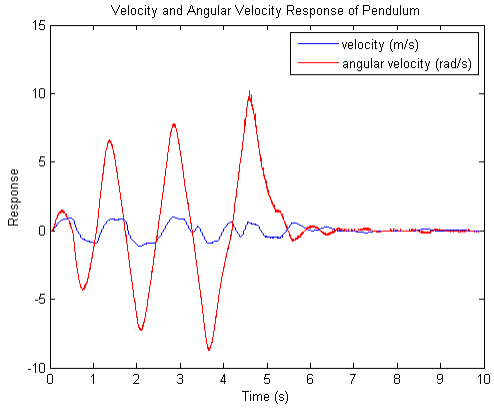

Figure 13: Graph of velocity of cart and angular velocity of the pendulum while balancing

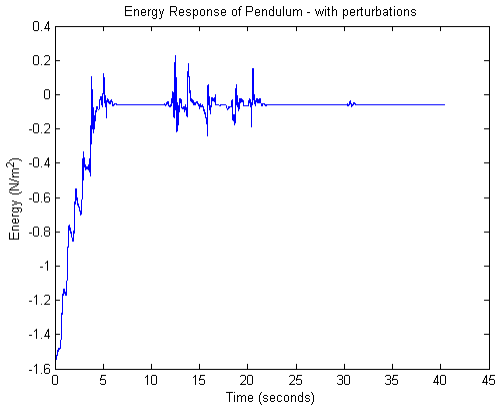

Figure 14: Graph of energy of pendulum and cart with perturbations

```
5
6 start_time = 0.25
7 Kpump = 50;
8 SAT = 0.5;
9 clp = 0.0160593;
10 thresh = (10*pi)/180;
```

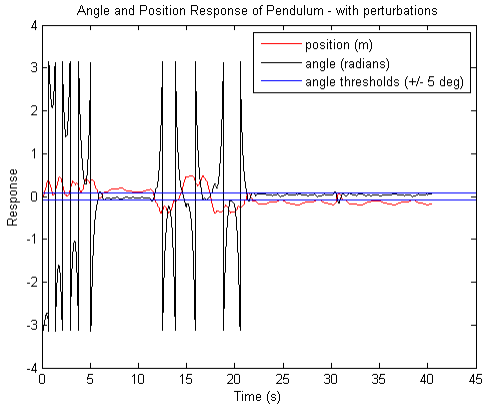

Figure 15: Graph of angle of pendulum and position of the cart with perturbations, with angle thresholds for reference

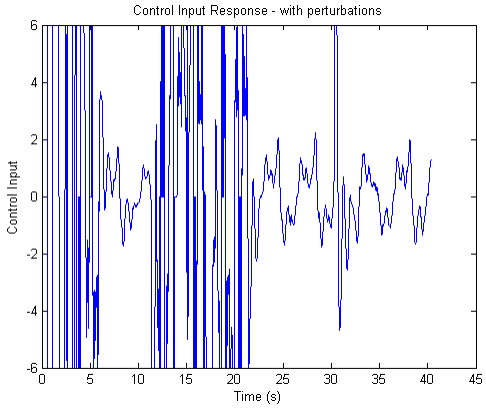

Figure 16: Graph of control input for pendulum and cart with perturbations

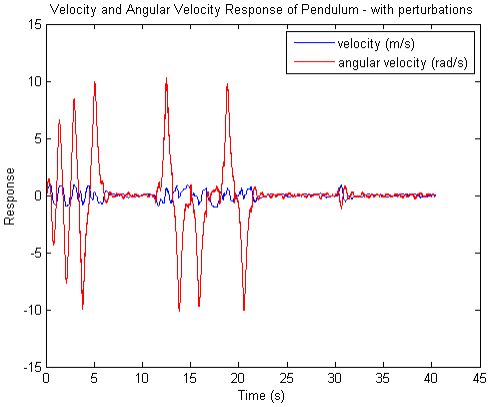

Figure 17: Graph of velocity of cart and angular velocity of pendulum with perturbations

```
ref = [0; 0; 0; 0];
M = 0.94;
m = 0.230;
Lp = 0.3302;
Ic = m*Lp^2/3;
```

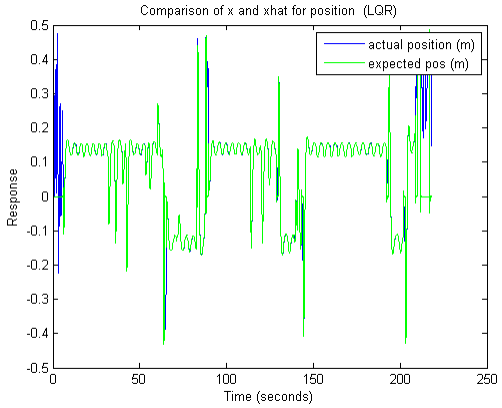

Figure 18: Position and Position estimate with lqr controller

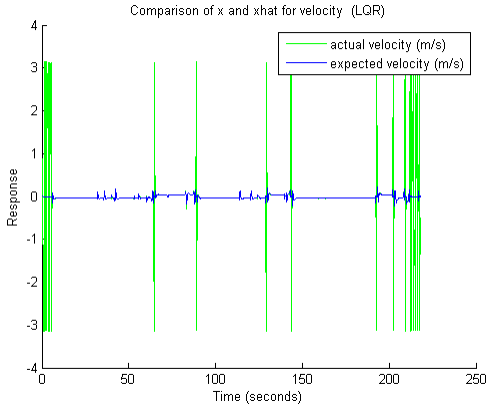

Figure 19: Velocity and velocity estimate with lqr controller

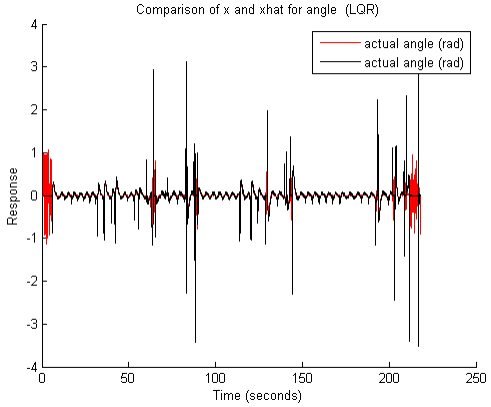

Figure 20: Angle and angle estimate with lqr controller

```
16  Ie = 4*m*Lp^2/3;
17  Kt = 7.67*10^(-3);
18  Km = 7.67*10^(-3);
19  r = 6.36 * 10^-3;
20  Rm = 2.6;
21  Kg = 3.71;
```

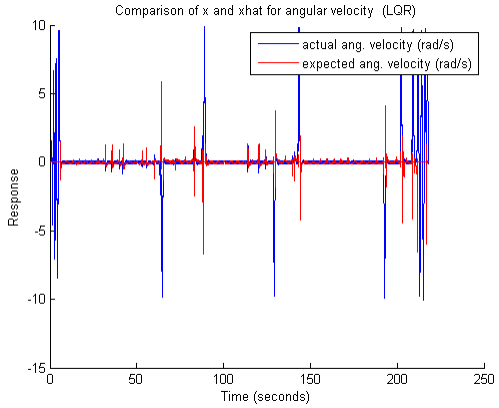

Figure 21: Angular velocity and angular velocity estimate with lqr controller

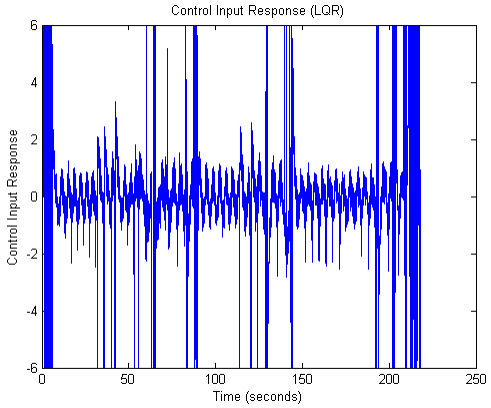

Figure 22: Voltage input graph with lqr controller

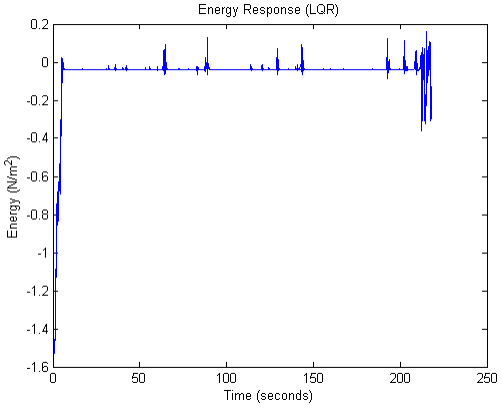

Figure 23: Energy graph with lqr controller

```
Jm = 3.9 * 10^-7;
g = 9.807;

den1 = (1/r^2)*Kg^2*Jm+M+(m/4);
den2 = (-4/(3*r^2)*Kg^2*Jm*Lp-(4/3)*Lp*(m+M)+m*Lp);
a12 = 1;
```

```
a34 = 1;
a22 = -(Km*Kg^2*Kt)/(r^2*Rm*den1);
a23 = (-3*g*m)/(4*den1);
a42 = -(Km*Kg^2*Kt)/(r^2*Rm*den2);
a43 = (-(1/r^2)*Kg^2*Jm*g - (m+M)*g)/den2;
b2 = (Kg*Kt)/(r*Rm*den1);
b4 = (Kg*Kt)/(r*Rm*den2);

A = [0 a12 0 0; 0 a22 a23 0; 0 0 0 a34; 0 a42 a43 0];
B = [0; b2; 0; b4];
C = [1 0 0 0; 0 0 1 0];
D = 0;

system = ss(A, B, C, D);

K = place(A, B, [-2+10j -2-10j -1.6+1.3j -1.6-1.3j]);
L = transpose(place(transpose(A), transpose(C), ...
            [-10+15j, -10-15j, -12+17j, -12-17j]));

% LQR
sys = ss(A, B, C, D);
Q = diag([5/0.25^2, 0, 1/0.05^2, 0]);
R = diag([0.7/6^2]);
[K, S, E] = lqr(sys, Q, R);
L = transpose(place(transpose(A), transpose(C), ...
            [-24+15j, -24-15j, -26+17j, -26-17j]));
```

Listing 3: lab6dcscript.m