E102 Final Project: Stabilizing an Inverted Pendulum

Clayson Briggs and Gabriel Zwillinger

I. INTRODUCTION

Stabilizing systems is a crucial application of control theory. Frequently an unstable system will need to be controlled for various applications. In this case, a cart must move 1m while supporting an inverted pendulum, as seen in Fig. 1. Additional constraints are also given: the cart must move the full 1m in under 10 seconds; the cart may not overshoot the final position; the magnitude of the acceleration of the cart is limited to $|a(t)| < 0.5 m/s^2$; finally, the cart is subjected to a constant angular acceleration disturbance of $\alpha(t) = 0.5\, rad/s^2$. To stabilize the system, we can develop a full state feedback control system that maintains the angular displacement of the pendulum while satisfying the given constraints.

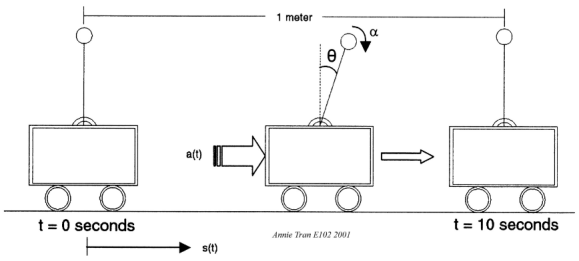

Fig. 1. Diagram of the cart-pendulum system to be stabilized. Courtesy of Annie Tran, 2001.

The governing equations of the pendulum-cart system are given as:

$$L\ddot{\theta}(t) - g\sin(\theta(t)) = a(t)\cos(\theta) + L\alpha(t)$$

$$\ddot{s}(t) = a(t)$$

where $L = 0.5m$ and $g = 9.81 m/s^2$ represent the length of the pendulum and gravitational acceleration, respectively.

II. STATE SPACE DESIGN OF LINEAR CONTROLLER FOR A LINEARIZED PLANT

A. Linearized State Space Equations

To formulate a state-space feedback control scheme for the total system, we first investigate the linearized state space equations for the cart-pendulum plant. The state vector used is:

$$\mathbf{x} = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ s(t) \\ \dot{s}(t) \end{bmatrix}$$

with the control input $u = a(t)$, disturbance input $w = \alpha(t)$ and output vector

$$\mathbf{y} = \begin{bmatrix} \theta(t) \\ s(t) \end{bmatrix}.$$

To linearize the system, we can use the small angle approximation such that $\sin(\theta(t)) = \theta(t)$ & $\cos(\theta(t)) = 1$. This results in the following state space equations:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{E}w \tag{1}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ g/L & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ -1/L \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{E} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \text{ and } \mathbf{D} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

With our state equations for system dynamics and outputs, we can move on to establishing the stability, controllability, and observability of the system.

B. Stability, Controllability, and Observability

The system stability is determined by the the eigenvalues of **A**, the system matrix. Using MATLAB to find the eigenvalues of **A**, we get $s_1 = 4.4294$, $s_2 = -4.4294$, and $s_3 = s_4 = 0$. Since s_1 has a positive real part, the original system is unstable, providing motivation for the implementation of a control scheme.

To determine if the system is controllable, we can examine the controllability matrix. We know that for a system to be stable, the controllability matrix must have rank n, where n is the dimension of the system matrix **A**. The controllability matrix is defined by:

$$\mathcal{C} = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \cdots & \mathbf{A}^{n-1}\mathbf{B} \end{bmatrix}$$

Using MATLAB, we find that $\text{rank}(\mathcal{C}) = 4$. Since the dimensions of **A** are 4 x 4, we know that $n = 4$ and thus the system is controllable.

Next, we analyze the observability of the system. The system is observable if the observability matrix $\mathcal{O}$ has $rank = n$. The observability matrix is given by:

$$\mathcal{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \vdots \\ \mathbf{CA}^{n-1} \end{bmatrix}$$

Using MATLAB, we find that $\text{rank}(\mathcal{O}) = 4$, which means that this system is observable.

C. State Feedback Control

State feedback was implemented using a linear proportional controller and integral action. A linear observer was also implemented to estimate the current state of the system. The block diagram of this system can be seen in Fig. 2.

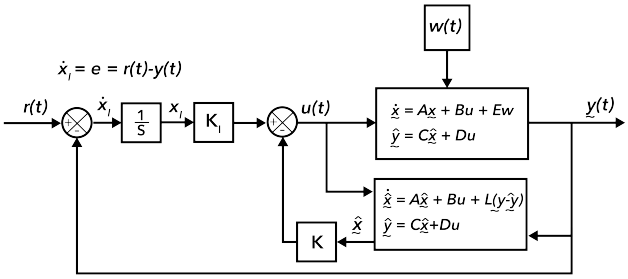

Fig. 2. Block diagram of the closed-loop cart pendulum system.

The first step in implementing full state feedback was finding the controller gains such that the closed-loop system was stable and met all constraints. To find the controller gains, the controller poles needed to be first determined. Because the system is fourth-order with integral control desired for the cart position, 5 controller poles are required. The method of dominant poles was used to simplify finding the poles, understanding that we would later tune the system to satisfy the constraints.

Using the method of dominant poles, we assume that the system can be approximated as second-order. We can then select values of ζ and ω_n knowing that the system may not have overshoot and must reach the final position in under 10 seconds. Since the equation derived for the settling time t_s is reliant on an underdamped second-order plant, we choose $\zeta = 0.99$ to minimize the overshoot, and we select $t_s = 7.5$ to introduce a factor of safety under 10 s. From here, we derive our values to be $\zeta = 0.99$ and $\omega_n = 0.884$.

Our dominant poles take the form of:

$$s_{1,2} = -\omega_n \zeta \pm \omega_n j\sqrt{1-\zeta^2}$$

We then chose the next 3 poles to have real parts 8, 9, and 10 times greater than the real part of our dominant poles, respectively. Altogether, after later tuning ζ and ω_n to better meet system constraints, our controller poles are selected to be

$$s_1 = -0.8752 + 0.1247j,\ s_2 = -0.8752 - 0.1247j$$

$$s_3 = -7.0013,\ s_4 = -7.8764, \text{ and } s_5 = -8.7516.$$

Equipped with our poles, we could now find our proportional gains $\mathbf{K}$ and our integral gain K_I. To do this, the state vector and system matrices were augmented to account for integral action:

$$\dot{x}_I = r(t) - y(t) = r - \mathbf{Cx} \tag{2}$$

and

$$\mathbf{u} = -\begin{bmatrix} -K_I & \mathbf{K} \end{bmatrix} \begin{bmatrix} x_I \\ \mathbf{x} \end{bmatrix} \tag{3}$$

.

Substituting Eq.s (2) and (3) into (1) results in our augmented state equation:

$$\begin{bmatrix} \dot{x}_I \\ \dot{\mathbf{x}} \end{bmatrix} = \mathcal{A} \begin{bmatrix} x_I \\ \mathbf{x} \end{bmatrix} + \mathcal{B}\mathbf{u}$$

where

$$\mathcal{A} = \begin{bmatrix} 0 & -\mathbf{C} \\ 0 & \mathbf{A} \end{bmatrix} \text{ and } \mathcal{B} = \begin{bmatrix} 0 \\ \mathbf{B} \end{bmatrix}$$

It is important to note that for the augmented matrices,

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

because only the displacement of the cart s is being controlled and not the rotation of the pendulum. Using the full $\mathbf{C}$ would result in an uncontrollable augmented system. Knowing the poles of the system and the augmented matrices $\mathcal{A}$ and $\mathcal{B}$ we can use the MATLAB command *place* to determine our controller gains. This results in:

$$K_I = -19.2219 \text{ and}$$

$$\mathbf{K} = \begin{bmatrix} -148.7722 & -34.2166 & -50.4360 & -43.0536 \end{bmatrix}.$$

D. Observer Design

The observer equations are given by:

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{Bu} + \mathbf{L}(\mathbf{y} - \hat{\mathbf{y}}) \tag{4}$$

$$\hat{\mathbf{y}} = \mathbf{C}\hat{\mathbf{x}}$$

We can substitute equations (1) and (4) into $\dot{\mathbf{e}} = \mathbf{x} - \hat{\mathbf{x}}$ to obtain:

$$\dot{\hat{\mathbf{e}}} = (\mathbf{A} - \mathbf{LC})\hat{\mathbf{e}} + \mathbf{E}w. \tag{5}$$

Equation (5) describes the observer dynamics. To determine the observer gains $\mathbf{L}$, we must first determine the observer poles. These are usually set to be 5-10x further from the origin than the controller poles so that the observer can quickly estimate the system state without stability concerns. With this in mind, we selected the observer poles to have real parts 7 times greater than the real part of the observer poles:

$$s_{observer} = 7\text{Re}(s_{controller}) + \text{Im}(s_{controller}).$$

Additionally, since only 4 observer poles are required, only the first 4 controller poles were used. The *place* command in MATLAB with the transposed $\mathbf{A}$ and $\mathbf{C}$ system matrices was

then used to obtain the initial observer gains. After tuning, our final observer gains are:

$$\mathbf{L} = \begin{bmatrix} 67.2655 & 0.1462 \\ 387.5825 & 2.6285 \\ 0.1562 & 61.3828 \\ 2.9371 & 344.3692 \end{bmatrix}.$$

E. Linear System Simulation and Verification

The next step in fully implementing state control is simulating the closed-loop system to verify that this control scheme meets the problem constraints. To do this, first note that when the observer is implemented, eq. (3) becomes:

$$\mathbf{u} = -\begin{bmatrix} -K_I & \mathbf{K} \end{bmatrix} \begin{bmatrix} x_I \\ \hat{\mathbf{x}} \end{bmatrix}. \tag{6}$$

After substituting equation (6) into (1) and using the fact that $\hat{\mathbf{e}} = \mathbf{x} - \hat{\mathbf{x}}$, we acquire the closed-loop system dynamics:

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{BK})\mathbf{x} + \mathbf{BK}\hat{\mathbf{e}} + \mathbf{B}K_I x_I + \mathbf{E}w \tag{7}$$

Equations (2), (5), and (7) can then be combined in one large augmented differential equation to encompass the controller, observer, and integral action dynamics:

$$\begin{bmatrix} \dot{x}_I \\ \dot{\hat{e}} \\ \dot{\mathbf{x}} \end{bmatrix} = \overline{\mathbf{A}} \begin{bmatrix} x_I \\ \hat{\mathbf{e}} \\ \mathbf{x} \end{bmatrix} + \overline{\mathbf{B}}r + \overline{\mathbf{E}}w$$

where

$$\overline{\mathbf{A}} = \begin{bmatrix} 0 & \mathbf{0} & \mathbf{C} \\ \mathbf{0} & \mathbf{A} - \mathbf{LC} & \mathbf{0} \\ \mathbf{B}K_I & \mathbf{BK} & \mathbf{A} - \mathbf{BK} \end{bmatrix},$$

$$\overline{\mathbf{B}} = \begin{bmatrix} 1 \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \text{ and } \overline{\mathbf{E}} = \begin{bmatrix} 0 \\ \mathbf{E} \\ \mathbf{E} \end{bmatrix}.$$

This linear ordinary differential equation was programmed into MATLAB and the solver *ode45* was used to derive the states over the 0 to 10 s interval. Initial conditions were all 0, as defined in the problem statement. Additionally, an *if* statement was used in the function definition to limit the maximum acceleration of the cart to 0.5 m/s^2, as given by the constraints. Using the controller, integral, and observer gains defined in sections C and D of this report, the system response is shown in Fig. 3. The cart acceleration is shown in Fig.4

As seen, these controller, integral, and observer gains result in a stable linear system that meets the constraints.

III. SIMULATION OF CONTROL OF NONLINEAR PLANT WITH THE LINEAR CONTROLLER

Up until now, all of the analysis has used the linearized governing equations resulting form the small angle approximation. Simulating the nonlinear plant will give more accurate behavior of the closed loop system, but the linear system provides a good starting point for controlling the nonlinear plant.

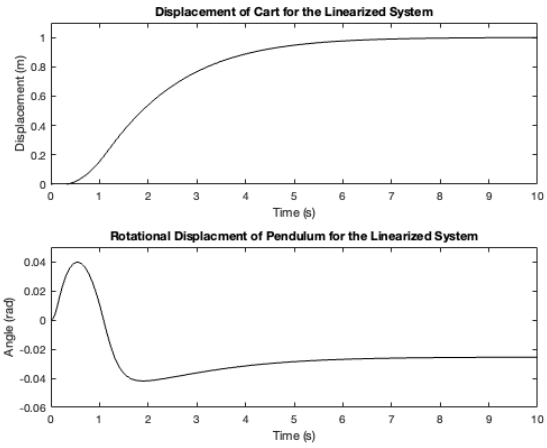

Fig. 3. Closed-loop system response of the linearized cart pendulum system. As seen in the upper diagram, the cart reaches a displacement of 1m without overshoot within 10 s. In the lower diagram, the constant angular acceleration disturbance results in a non-zero steady-state rotational displacement of the pendulum.

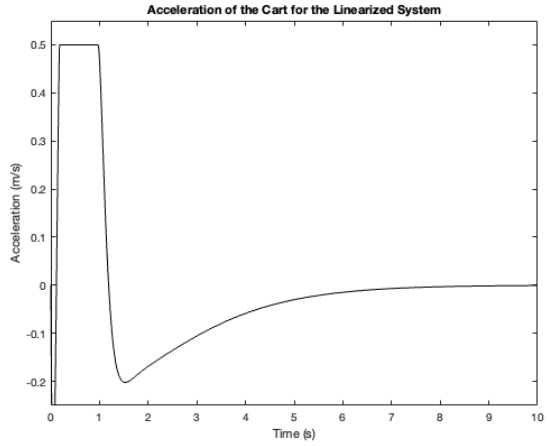

Fig. 4. Cart acceleration in the linearized system. The acceleration does not exceed 0.5 m/s^2 as specified.

A. Simulating the Nonlinear System

Using the same controller, integral, and observer gains as the linear system, the nonlinear plant, controller, and observer were programmed into a MATLAB function, and the states were derived using *ode45*. Just like in the linear system, an *if* statement was used to make sure the cart acceleration did not exceed 0.5 m/s^2 and initial conditions were all 0. As seen in Fig. 5 and Fig. 6, this closed-loop system response of the nonlinear system also satisfies all system constraints. Since the system already meets the constraints with the same system gains as the linear system, the parameters did not need to be tuned.

B. Largest Allowable Disturbance

We want to find the largest allowable disturbance $w(t)$ such that the system will become unstable. We are also interested in seeing at which disturbance $w(t)$ the system

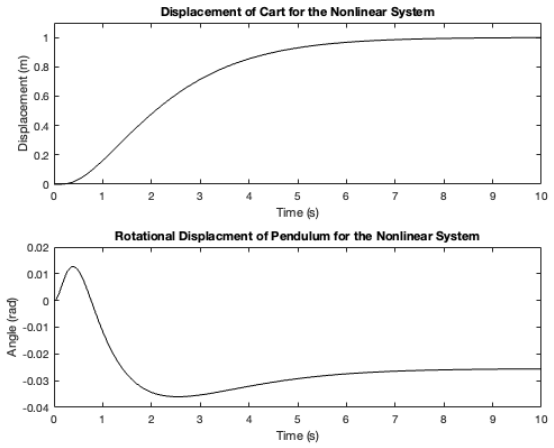

Fig. 5. Closed-loop system response of the nonlinear cart pendulum system. As seen in the upper diagram, the cart reaches a displacement of 1m without overshoot within 10 s. In the lower diagram, the constant angular acceleration disturbance results in a non-zero steady-state rotational displacement of the pendulum.

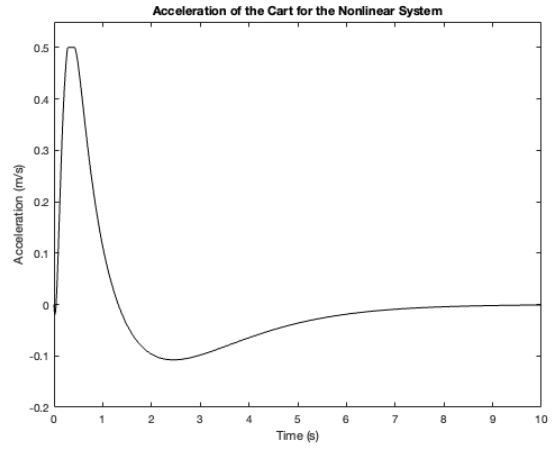

Fig. 6. Cart acceleration in the nonlinear system. The acceleration does not exceed 0.5 m/s^2 as specified.

no longer meets the system constraints. To consider system stability more generally, we no longer restricted time to 10 seconds, as we wanted to see if the systems were stable for all time, not necessarily within the constraints of the problem. We did, however, leave our controller and observer gains the same, as that provided a more interesting comparison to our previous system than tuning our system poles and gains further. Iterating through $w(t)$, we found that as long $w(t) \leq 7.5$, the system remained stable. That being said, the system had 115% overshoot and a settling time of 36 seconds. We could, in a future problem, maximize the disturbance the system could handle while staying within the problem constraints by retuning the poles and gains further, but that will remain a fun future challenge for a rainy day.

Next, we found the maximum allowable disturbance such that the system constraints were still met. Again leaving the controller and observer gains the same, we slowly incremented $w(t)$ until we found that the maximum value of $w(t)$ such that the cart reached 1m by 10 seconds, had no overshoot, and $|a(t)| < 0.5m/s^2$. This resulted in a maximum disturbance of $w(t) = 2.4$. For $w(t) \leq 2.4$, the system was stable and met all system constraints.

IV. DISCUSSION AND CONCLUSIONS

A cart supporting an inverted pendulum subject to a disturbance was stabilized by deriving control, observer, and integral gains. The cart was simulated and controlled both by analyzing the linearized system for simplicity and implementing the full nonlinear system in MATLAB. To control the system, we first determined approximate controller poles using the method of dominant poles. We then found observer poles that were 7 times faster than the controller poles. We were then able to find the corresponding controller, integral, and observer gains. Using *ode45* with a limit on α, we simulated the system and tuned our gains to verify that our system constraints were met. The same controller and observer gains were used in both the linear and nonlinear systems. As expected, the linearized model gave a moderately good approximation for the actual, nonlinear, system. A comparison of the two models can be seen in Fig. 7

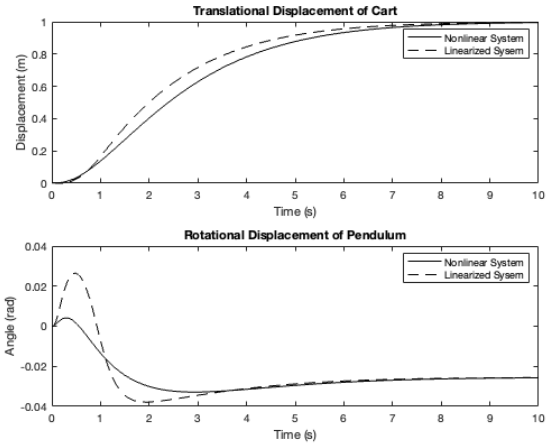

Fig. 7. Comparison of cart and pendulum displacement for the linear and nonlinear system

Increasing the disturbance $w(t)$ leads to the linear approximation deviating further and further from the nonlinear system. This is to be expected as the linear model relies on small-angle approximation and as the applied disturbance gets bigger, the angle of the pendulum increases.

V. ACKNOWLEDGMENTS

We would like to thank Professors Cha and Shia for their guidance throughout this project and for teaching us the fundamental skills that we learned in this course. We also wanted to give a big shoutout to Professor Bright for providing us with comprehensive notes on the class.

VI. APPENDIX

A. MATLAB Script

```
% E102 Final Project
% CB + GZ

% Parameters
g = 9.81;
l = 0.5;
alpha = 0.5; % alpha (nonlinear: max for stability is 7.5, 2.4 for meeting constraints)

% Plant matrices
A = [0 1 0 0; g/l 0 0 0; 0 0 0 1; 0 0 0 0];
B = [0; -1/l; 0; 1];
C = [1 0 0 0; 0 0 1 0];
D = [0; 0]; %[0 0; 0 0];
E = [0; 1; 0; 0];

% Check stability
Aeig = eig(A);

% Check controllability
Con = [B A*B A^2*B A^3*B];
if rank(Con) == 4
    disp('This system is controllable!')
else
    disp("This system ain't controllable breh")
end

% Check observability
Obs = [C; C*A; C*A^2; C*A^3];
if rank(Obs) == 4
    disp('This system is observable')
else
    disp('this system is tragically not observable')
end

% Solve for initial poles
Mp = 0.001;
ts = 7.5;
ops = optimset('Display', 'off');
zG = 0.99; %fsolve(@(z) exp(-pi*z/sqrt(1-z^2))-Mp, 0.7, ops);
wG = fsolve(@(w) -log(0.01*sqrt(1-zG^2))/(zG*w)-ts, 0.7, ops);
pG = [wG*(-zG+1i*sqrt(1-zG^2)), wG*(-zG-1i*sqrt(1-zG^2)), -wG*zG*8, -wG*zG*9, -wG*zG*10]; % poles with dominant pole method

% Augmented system matrices and controller gains
Aa = [0 -C(2,:); zeros(4,1) A];
Ba = [0; B];

Kextended = place(Aa, Ba, pG);
Ki = -Kextended(1);
K = Kextended(2:end);

% Calculate observer poles and gains
obsPG = zeros(length(pG)-1,1);
for i = 1:(length(pG)-1)
    obsPG(i) = 7*real(pG(i))+imag(pG(i));
end
L = (place(A',C',obsPG))';

% Solve linear system
[tL,xLin] = ode45(@(t,x) finalLinear(t,x,1,alpha,A,B,C,K,Ki,L,E), [0 10], [0, zeros(1,4), 0, 0, 0, 0]);

% Plotting response of linear system
figure(1)
clf
subplot(1,2,1)
plot(tL,xLin(:,8))
xlabel('Time (s)')
ylabel('Displacement (m)')

subplot(1,2,2)
plot(tL,xLin(:,6))
xlabel('Time (s)')
ylabel('Angle (rad)')

% Solve and plot nonlinear system
[tN, xN] = ode45(@(t,x) finalNonlinear(t,x,1,alpha,A,B,C,K,Ki,L,E), [0 10], [0, zeros(1,4), 0, 0, 0, 0]);

figure(2)
clf
subplot(1,2,1)
plot(tN,xN(:,8))
xlabel('Time (s)')
ylabel('Displacement (m)')

subplot(1,2,2)
plot(tN,xN(:,6))
xlabel('Time (s)')
ylabel('Angle (rad)')

% Test integral control
Aint = [0 -C(2,:); zeros(4,1) A]; %[0 zeros(1,4) -C(2,:); zeros(4,1) A-L*C zeros(4,4); B*Ki B*K A-B*K];
Bint = [0; B]; %[1; zeros(4,1); zeros(4,1)];
```

```
Cint = [0 0 0 1 0]; %[0 0 0 0 0 0 0 1 0];

sysInt = ss(Aint-Bint*Kextended, [1; 0; 0; 0; 0], Cint, 0);

figure(3) % acceleration graph
accel = zeros(length(tL),9);
for i = 1:length(tL)
    accel(i,:) = finalLinear(tL(i), xLin(i,:)', 1, alpha, A, B, C, K, Ki, L, E);
end
plot(tL, accel(:,9))
xlabel('Time (s)')
ylabel('Acceleration (m/s)')
title('Acceleration of cart')

figure(4) % nonlinear system acceleration
accel = zeros(length(tN),9);
for i = 1:length(tN)
    accel(i,:) = finalNonlinear(tN(i), xN(i,:)', 1, alpha, A, B, C, K, Ki, L, E);
end
plot(tN, accel(:,9))
xlabel('Time (s)')
ylabel('Acceleration (m/s)')
title('Acceleration of nonlinear system')

figure(5)
clf
step(sysInt)
title('Integral Control, no observer')

figure(6) % debug menu
clf
for i = 1:5
    subplot(2,3,i)
    plot(tL,xLin(:,i))
end

max(xN(:,8))
```

B. Linear MATLAB Function

```
function out = finalLinear(t, x, r, w, A, B, C, K, Ki, L, E)
%The function for our E102 Final Project
%   Inputs are t (time), x (state), r (tracking input),
%   w (disturbance input), A, B, and C (the system
%   matrices), K and L (the controller and
%   observer gains), and E (the error matrix)
%   Output is xdot

Abar = [0 zeros(1,4) -C(2,:);
        zeros(4,1) A-L*C zeros(4,4);
        B*Ki B*K A-B*K];
Bbar = [1; zeros(8,1)];
Ebar = [0; E; E];

xdot = Abar*x + Bbar*r + Ebar*w;

% if t > 2 && t < 2.25
%     xdot = xdot + Ebar*w;
% end

if xdot(9) > 0.5
    xdot(9) = 0.5;
elseif xdot(9) < -0.5
    xdot(9) = -0.5;
end

out = xdot;

end
```

C. Nonlinear MATLAB Function

```
function xdot = finalNonlinear(t, x, r, w, A, B, C, K, Ki, L, E)
%finalNonlinear The nonlinear function for the E102 final project
%   Takes u, the system input, as well as alpha, disturbance input, as
%   system inputs
xdot = zeros(9,1);

g = 9.81;
l = 0.5;

%u = -K*x(2:5)+Ki*x(1);

xdot(1) = r - C(2,:)*x(6:9);
xdot(2:5) = (A-L*C)*x(2:5)+E*w;
xdot(6:9) = [x(7);
             g/l*sin(x(6))-1/l*cos(x(6))*(-[-Ki K]*[x(1); x(6:9)])+w;
             x(9)
             -[-Ki K]*[x(1); x(6:9)]];

if xdot(9) > 0.5
    xdot(9) = 0.5;
elseif xdot(9) < -0.5
    xdot(9) = -0.5;
end

end
```

D. Simulink Block Diagram

We did not use Simulink and thus have no Simulink block diagram. See Fig. 2 for system block diagram