The Cart on the Track

Joseph Le

AAE 364L

15 September 2021

TA: Himal Sapkota

1 – Introduction

1.1 – Objective, Goals, and Purposes

The objective of this lab was to learn the fundamentals of using a PID controller to move a cart within the performance specifications. The purpose was to be able to translate the handwritten calculations of PID gains and apply them to a real apparatus that had real world factors playing into its performance and how the gains affected the performance altogether.

1.2 – Intended Methods

The methods that were used included control of a cart on a track using inputs to the cart that were sent out and regulated by a computer through Simulink. The Simulink models modeled the cart, voltages, and other factors that went into consideration with the apparatus. PID gain values that were calculated were logged into the Simulink model and the experiment ran with those values and the data was logged into the workspace.

2 – Procedure

2.1– Definition of Variables

Variable	Description	Units
m	Mass of cart, weight, and induced mass from the motor	kg
c	Total damping of the cart $B_{eq} + B_{emf}$	kg/s
K	PID gains (p – proportional, i – integral, d – derivative)	
y	Position of the cart (displacement)	m
B_{emf}	Damping of cart due to voltage	kg/s
B_{eq}	Damping of cart due to friction	kg/s
f_c	Coulomb friction constant	
V	Voltage	V

2.2– Schematic and Description of Apparatus

Figure 1 – Experimental Apparatus

A cart is held on a track with a bar that guides it as it moves left and right. The cart is moved by a gear and motor that is connected to the computer which sends determines how the cart moves. The computer uses a connected between the cart and Simulink through sensors that allows it to send data and voltage to and from the cart and save it for later use.

2.3 – Procedure of Experiments

Part (i): The Open Loop Model

a) Determine B_{eq} by moving the cart with a push and logging the data for later calculations.
b) Calculate mass, voltage gain, and EMF damping on the cart using the equations from the lab manual.

Part (ii): Model Validation and Saturation

a) Run the cart with the labcartsat.mdl Simulink model to test the movement of the cart.
b) Save the cart position and voltage data.

Part (iii): The Effect of Coulomb Friction

a) This section was skipped

Part (iv): Integral Controller and Coulomb Friction

a) Set PID controller gains to specified values in lab manual into the labcartcolint.mdl file
b) Run the simulation and save the error data.

Part (v): Moving the Cart with PID Controller

a) Set max cart distance.
b) Set PID gains to calculated values from the prelab.
c) Run the simulation and save the position output.
d) Repeat steps b and c until desired performance is reached and save those PID gains.
e) Set desired position to a slider and move the cart around.

3 – Results

3.1 – Parameter Values

m	1.0731	kg
γ	1.7235	N/V
B_{emf}	7.7236	kg/s
B_{eq}	2.2857	kg/s
c	10.0093	kg/s

3.4 – Largest Stable k_i Value

k_i	131.2356	

3.5a – Initial k_p, k_i, and k_d values used in Simulink and Simulated Performance

k_p	37.5962	
k_i	0	
k_d	1.3466	
Rise Time	0.5845	sec
Percent Overshoot	1.12	%
Settling Time	0.9344	sec
Steady State Error	0	m

3.5b – Final k_p, k_i, and k_d values used in Experiment and Recorded Performance

k_p	80	
k_i	2	
k_d	10	
Rise Time	0.5797	sec
Percent Overshoot	0	%
Settling Time	0.9559	sec
Steady State Error	0.0033	m

4 – Analysis and Discussion

4.1 – The Open Loop Model

For part i of the lab, we calculated the values of B_{eq}, γ, m, and B_{emf}. To do so, we used the masses that were given to us to calculate m along with the equation for the virtual mass that was included to compensate for the motor, and the two other equations to solve for γ and B_{emf}. B_{eq} was computed using the equations of motion:

$$m\ddot{y} + c\dot{y} = \gamma v$$

$$m\ddot{y} + B_{eq}\dot{y} = 0$$

The second equation of motion given above is obtained by unplugging power from the cart such that there are no electromotive forces that are acting on the cart thus removing the gamma and B_{emf} terms from the damping and only includes the B_{eq} and mass terms.

The solution to the second equation is: $y = \frac{\dot{y}_0 m}{B_{eq}}(1 - e^{\frac{-B_{eq}t}{m}})$, where $y(\infty) = \frac{\dot{y}_0 m}{B_{eq}}$. From here the B_{eq} can be calculated using the slope of the experimental cart data and the mass that was calculated in the prelab.

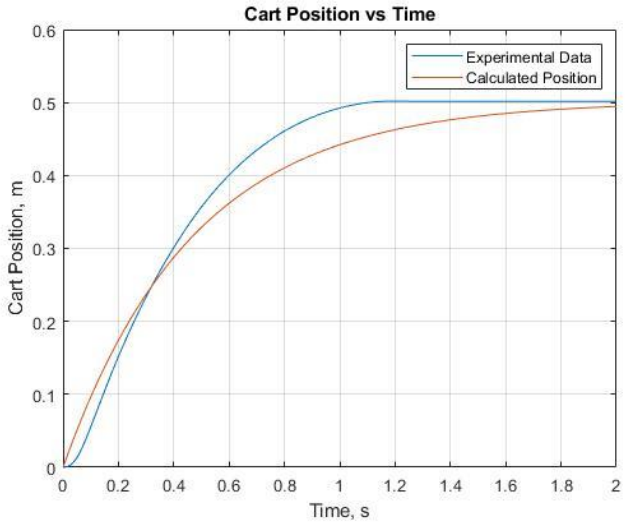

Figure 2 – Experimental vs Calculated Cart Position (B_{eq} calculation)

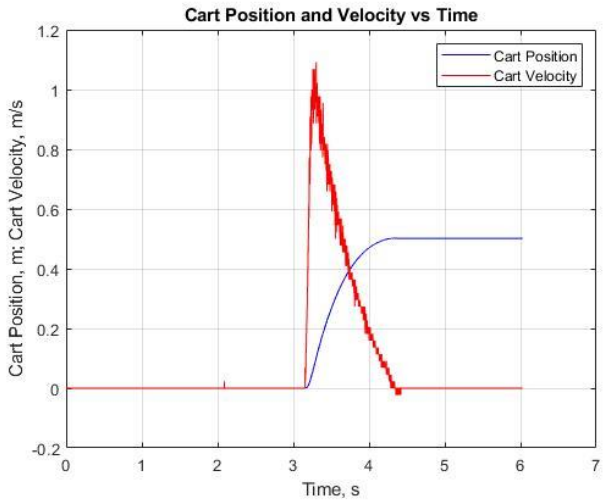

Figure 3 – Cart Position and Velocity

The carts do not initially coincide because of the estimations that were used to get the slope of the experimental data was inaccurate. To do so, the derivative of the experimental data was calculated using a MATLAB command (diff()), that data was then taken and analyzed to see where the slope leveled out for a bit and that slope was used for the calculation of B_{eq}. Another reason for the difference in the plots is that the moment and time the force was put on the cart may have been longer than anticipated which would have affected the experimental data.

4.2 – Model Validation and Saturation

The maximum voltage that the motor on the cart can handle is between +6V and -6V, thus the saturation of the cart motor has to be taken into consideration when modeling the system to not exceed the maximum.

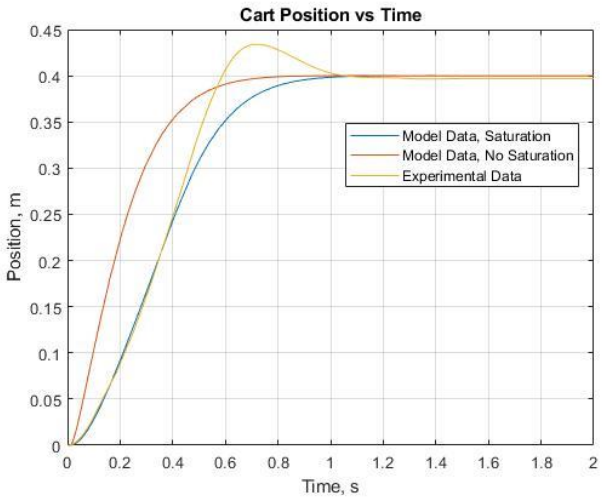

Figure 4 – Comparing Effect from Saturation, Position

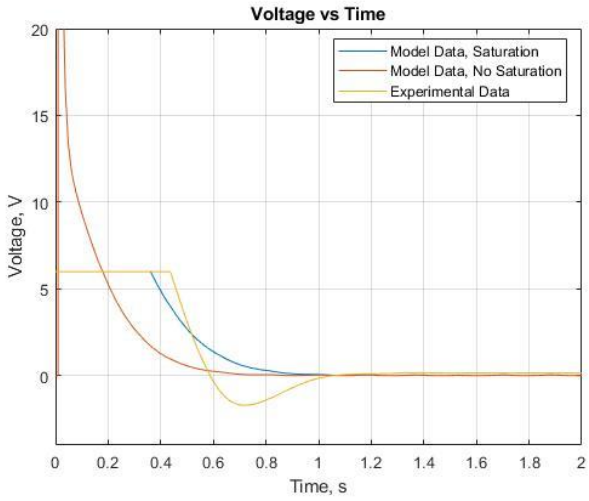

Figure 5 – Comparing Effect from Saturation, Voltage

These plots are generated from a Simulink model that was used in the prelab which matches figure 7 in the lab manual. The effect of saturation on the position of the cart is that it shifts the plot to the right, effectively making the response slower. This is because the maximum input is reduced and limited thus the cart cannot move as fast as it needs to in order to reach the desired distance in the shortest time. This explains why the y with no saturation moves faster since it can apply more force into its movement. To calculate the new estimate of B_{eq}, use the data from the Simulink model with and match it with the experimental data which yields: B_{eq} = 3.7107, this was done by increasing the k_p gain by 5 times

4.4 – Integral Controller and Coulomb Friction

An integral controller is a controller type that increases the system type of the transfer function to reduce or eliminate the steady state error of the response. As noted in the results section, the maximum k_i for the system would be 131.2356 before the system would become unstable which would cause the cart to go off of the track.

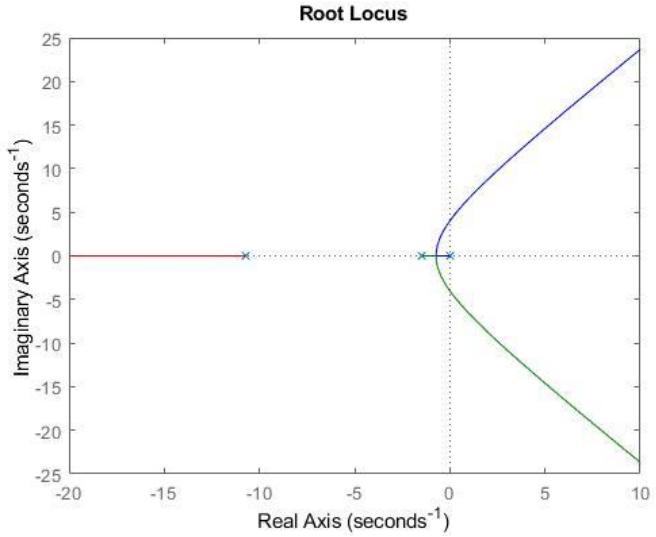

Figure 6 – Root Locus of System

The root locus for the system can be seen in the figure above. This can be used to calculate the k_i maximum value before the system becomes unstable. This value can be seen in table 3.4 which was a value of 131.2356.

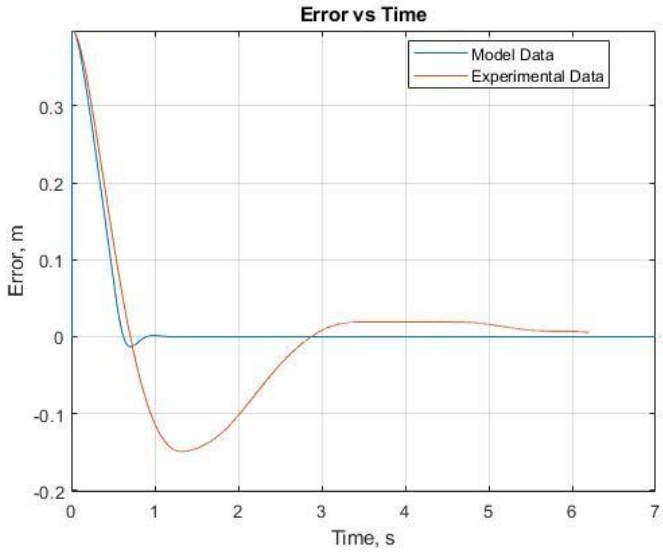

Figure 7 – Cart Position Error

The plots are generated in MATLAB using the data from the Simulink model and the experimental data. As can be seen from the plot above, the experimental error is much larger and is generally slower to reach zero along with the fact that the error for the experimental data doesn't actually go to zero. This may be due to the gears on the motor of the cart, the gears may keep the cart from moving into the desired point completely and drive the error to zero.

4.5 – Moving the Cart with PID Controller

The PIDs gains that were originally calculated are listed in table 3.5a. These values were calculated with the performance specifications that were outlined in the prelab. The final values that were used in the experiment are listed in table 3.5b, which was able to reach our design criterion.

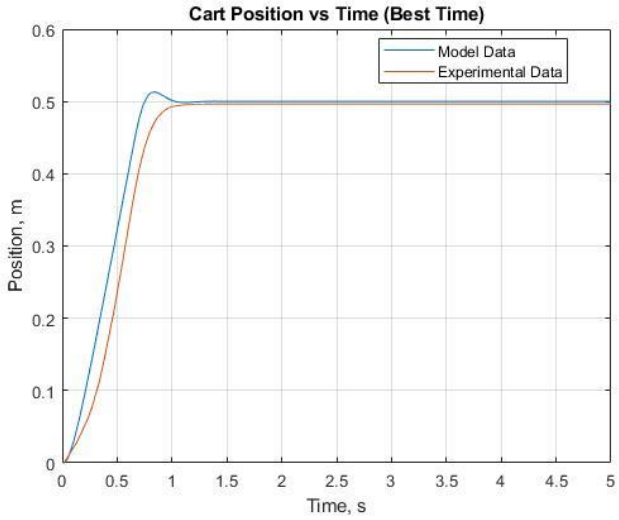

Figure 8 – Cart Position vs Time, Best Run

Using the PID gains from table 3.5b, the final motion was recorded in figure 8. This shows that the simulation was pretty close to the actual experiment. The differences are seen in the rise time and overshoot. The experiment had a slower rise time as seen in the shift to the right, but had little to no overshoot compared to the simulation. The overall performance and speed of the simulation was a little bit better compared to the experiment, but with little to no overshoot, the experiment was more accurate.

5 – Conclusion and Recommendation

5.1 – Main Points

This lab walked through the process of using a PID controller to control the movement of a cart. It detailed the considerations that are needed to take a model and confirm the behavior in real life. Such considerations included the saturation of the motors, damping forces and friction, and the design and performance criterion that are used as a basis on how the system would respond.

5.2 – Theoretical/Experimental Limitations

Limitations that may include and originate from the digital noise that may come from the sensors and the limited time steps that were used. One such occurrence may be the physical

interactions between the someone pushing the cart and the cart itself which would change the velocity measurements for calculating B_{eq}. Such interactions may have factors that happen faster than 1 millisecond which was the step time for the experiment, this caused some velocity graphs to become jagged which made it hard for calculations. Some theoretical limitations include

5.3 – Lessons Learned and Improvement

Lessons that were learned were to spend more time on fine tuning the performance of the cart movement by changing to PID gain values instead of settling for a "good enough" set of data. Other than that, no other suggestions for improvement come to mind other than more information on taking care of the equipment since damage seemed to build up over time.

Appendix: MATLAB code and Simulink models included in sections below

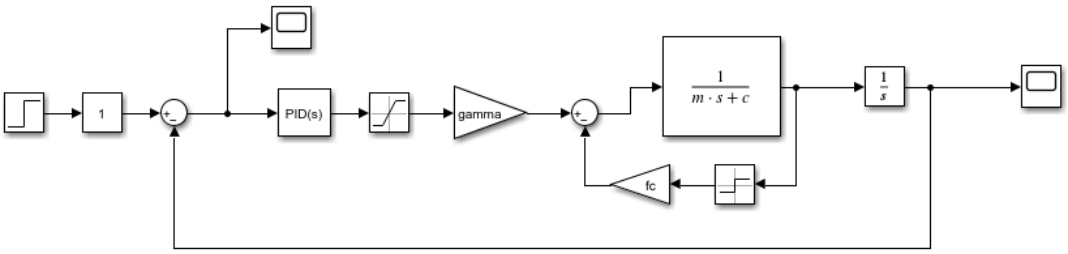

Figure 9 – Complete Simulated System

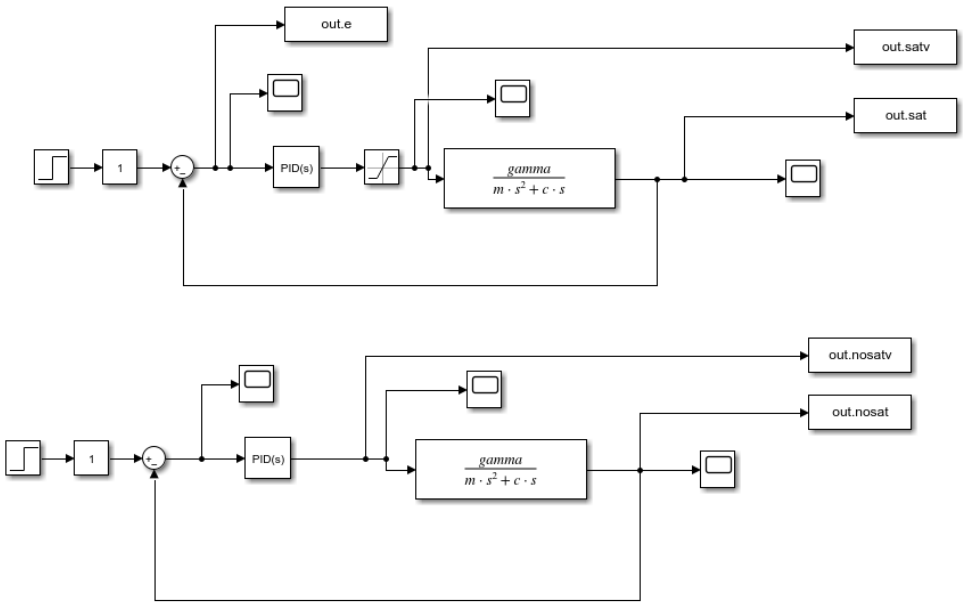

Figure 10 – Part ii Model With and Without Saturation

MATLAB code:

```
%% i)

nu_m = 1;
nu_g = 1;
K_g = 3.71;
J_m =3.9E-7;
r_mp = 6.35e-3;
M_c = .57;
M_w = .37;
M = M_c+M_w;
K_t = .00767;
R_m = 2.6;
K_m = .00767;

M_J = nu_g*K_g^2*J_m/(r_mp^2);
m = M + M_J
gamma = nu_g*K_g*nu_m*K_t/(R_m*r_mp)
B_emf = nu_g*K_g^2*nu_m*K_t*K_m/(R_m*r_mp^2)
%% ii)

k = 10;
B_eq = 5.4;
c = B_eq + B_emf
```

```
num = gamma;
den = [m c gamma*k 0];
Lpoles = roots(den)
Lzeros = roots(num)
thetaa = (180+360*[0,1,2])/3
sigmaa = (sum(Lpoles)-sum(Lzeros))/3
breakpoints = roots([3*m,2*c,gamma*k])
w = [0; roots([-1 0 gamma*k])]
ki = c*w(2)^2/gamma
num = [gamma];
den = [m, c, gamma*k, 0];
L = tf(num,den)
% rlocus(L)
%% iii)

Mp = 2; ts = .5;
zd = -log(Mp/100)/sqrt(pi^2+log(Mp/100)^2)
wnd = 4/(zd*ts)
sd = [-zd*wnd + 1i * wnd * sqrt(1-zd^2); -zd*wnd - 1i * wnd *
sqrt(1-zd^2)]
num2 = [gamma]; den2 = [m c 0];
G = tf(num2,den2);
gpole = roots(den2)
sdd = sd(1);
theta1 = rad2deg(atan2(imag(sdd),real(sdd)))
theta2 = rad2deg(atan(imag(sdd)/(-gpole(2)+real(sdd))))
Gangle = -theta1 -theta2
% Gangle = rad2deg(-(atan2(imag(sdd),real(sdd)) +
atan2(imag(m*sdd+c),real(m*sdd+c))))
phi = -180 - Gangle
rlocus(G)
zd = -(imag(sdd)/tand(phi)-real(sdd))
kd = norm(m*sdd^2+c*sdd)/norm(sdd+zd)/gamma
% kd = 1.47;
fc = 1;
kp = -zd*kd;
ki=0;
% stepinfo(out.x.data,out.x.time)
%%
%

plot(out.sat.time,out.sat.data)
hold on
% plot(out.nosat.time,out.nosat.data)
%
plot(linspace(0,2.705,length(v.signals.values)),y.signals.values
)
```

```
grid on
plot(Cartposition(:,1),Cartposition(:,2))
title("Cart Position vs Time (Best Time)")
xlabel("Time, s")
ylabel("Position, m")
legend("Model Data",'Experimental Data','Location','Best')
xlim([0,5])
hold off
diff(out.sat.data)/abs(9.9861-10.0000)

plot(out.satv.time,out.satv.data)
hold on
plot(out.nosatv.time,out.nosatv.data)
plot(linspace(0,2.705,length(v.signals.values)),v.signals.values
)
grid on
title("Voltage vs Time")
xlabel("Time, s")
ylabel("Voltage, V")
legend("Model Data, Saturation",'Model Data, No
Saturation','Experimental Data','Location','Best')
xlim([0,2])
ylim([-4,20])
hold off
%%
hold off
% plot(out.e.time,out.e.data)
% hold on
% time = 0:.001:6.2-.001;
% plot(time,e.signals.values)
% grid on
% title("Error vs Time")
% xlabel("Time, s")
% ylabel("Error, m")
% legend("Model Data",'Experimental Data','Location','Best')
% xlim([0,7])
% hold off

t = Cartposition(:,1);
dt = t(2)-t(1);
y = Cartposition(:,2);
ydot = diff(y)/dt; ydot(end+1) = ydot(end);
%%
figure(1)
plot(t,y,'b')
title("Cart Position and Velocity vs Time")
xlabel("Time, s")
```

```
ylabel("Cart Position, m; Cart Velocity, m/s")
grid on
hold on
plot(t,ydot,'r')
legend("Cart Position","Cart Velocity")
hold off
%%
m = 1.0731; gamma = 1.7235; Bemf = 7.7236;
% ydot_avg = mean(ydot(3593:3647))
ydot_avg = 1.06854386538119;
y_inf = y(end);
Beq = (y_inf/(ydot_avg*m))^-1
c = Bemf + Beq
%%
t2 = t(1:2875);
figure(2)
plot(t2,y(3156:end))
y_beq = ydot_avg * m / Beq *(1-exp(-Beq*(t)/m));
hold on
plot(t,y_beq)
grid on
title("Cart Position vs Time")
xlabel("Time, s")
ylabel("Cart Position, m")
legend("Experimental Data",'Calculated Position')
xlim([0 2])
```