The cart on a track

Author: Brent Justice

Course: AAE 36401

Date: 9/11/2014

TA: Abi Akomandu

# Introduction

This experiment had the following objectives:

Part (i)

* Determine equations of motion for a cart on a one dimensional track
* Determine damping on cart due to viscous friction

Part (ii)

* Investigate closed-loop feedback control on cart position
* Investigate effect of saturation on voltage input for feedback control

Part (iii)

* Determine effect of coulomb friction on cart motion
* Determine steady state error on cart position due to coulomb friction

Part (iv)

* Investigate effect of integral control feedback on cart position
* Investigate effect of integral controller on steady state error

Part (v)

* Investigate effect of PID control feedback on cart position
* Investigate effectiveness and accuracy of simulation software for designing PID controller with hardware out of the loop

An experimental apparatus containing a cart on a one dimensional track was utilized for reaching the above objectives. More specifically, the cart moved along the track with the use of an electric motor. The motor was controlled through the specification of input voltage. As the cart moved along the track, position was recorded.

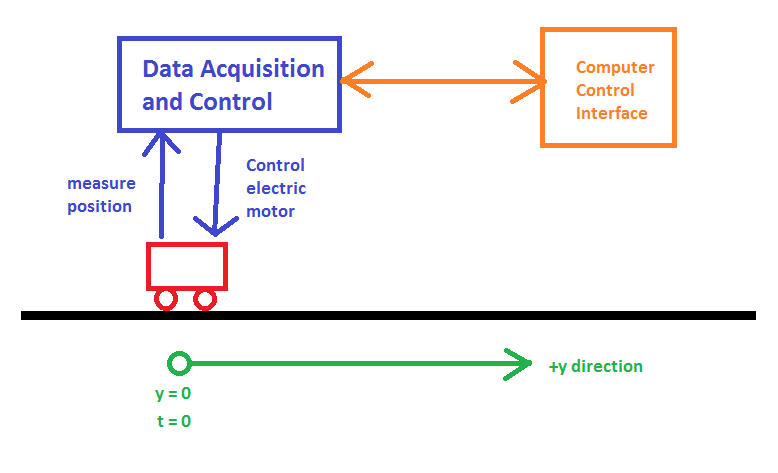
SIMULINK was utilized for modeling the cart motion and producing simulation data for comparing against experimentally observed data.

# Procedure

Definition of variables:

|  |  |  |
| --- | --- | --- |
| variable | units | Description |
|  |  | Motor armature resistance |
|  |  | Motor armature inductance |
|  |  | Motor torque efficiency |
|  | % | Motor efficiency |
|  | V-s/rad | Back electromotive force |
|  |  | Rotor moment of inertia |
|  |  | Planetary gearbox ratio |
|  | % | Planetary gearbox efficiency |
|  | kg | Cart mass |
|  | kg | Cart weight mass |
|  | m | Track length |
|  | m | Cart travel |
|  | m/tooth | Rack pitch |
|  | m | Motor pinion radius |
|  |  | Motor pinion number of teeth |
|  | m | Position pinion radius |
|  |  | Position pinion number of teeth |
|  | m/count | Cart encoder resolution |
| c | Kg/s | Total damping |
| y | m | Cart position |
| v | volts | Voltage input to motor |
| e | m | Cart position error |
|  | Kg/s | Viscous damping |
|  | Kg/s | Back emf damping |
|  | V/m | Proportional constant |
|  | V/m | Integral constant |
|  | V/m | Derivative constant |
|  | N/(Ω\*A) | Gain from voltage applied to motor to the cart force |
|  | N\*V/(Ω\*A) | Coulomb friction constant |
| |e(∞)| | m | Steady state error |

Schematic and Description of Apparatus



**Figure (0):** Experimental apparatus

From Figure 0, we see that the experimental apparatus consists of a car on a one dimensional track. At time = 0, the initial position is defined as y = 0. The cart motion is controlled by voltage input to an electric motor on the cart.

SIMULINK is utilized for control and data acquisition. More specifically, SIMULINK specifies the voltage input to the cart motor at run-time. Cart position is observed and saved by SIMULINK.

One limitation of the experiment is the length of the track – care must be taken to ensure that the cart does not crash into the end of the track. Additionally, the SIMULINK software only buffers 10 seconds of historical data. A test lasting longer than this time would require updated software.

Procedure of Experiments

Set SIMULINK data buffer to 10 seconds.

Part (i)

* Unplug electric motor
* Tap cart gently to simulate impulse input
* Record cart position over time

Part (ii)

* Initialize feedback loop control on cart position with the following inputs:
  + Proportional gain = 50 V/m
  + Target distance = 0.4 m (step input of 0.4)
* Record cart position and motor voltage input

Part (iii)

* Apply ramp voltage input to cart at 0.1 V/s
* Stop cart after it has begun moving
* Record cart position
* Initialize feedback loop control on cart position with the following inputs:
  + Proportional gain = 10 V/m
  + Target distance = 0.4 m (step input of 0.4)
* Record cart error (0.4 – y)

Part (iv)

* Initialize feedback loop control on cart position with the following inputs:
  + Proportional gain = 50 V/m
  + Integral gain = 15 V/m
  + Target distance = 0.4 m (step input of 0.4)
* Record cart error (0.4 – y)

Part (v)

* Initialize feedback loop control on cart position with the following inputs:
  + Proportional gain = 1000 V/m
  + Integral gain = 20 V/m
  + Derivative gain = 6 V/m
  + Target distance = 0.4 m (step input of 0.4)
* Record cart position, voltage input, and cart error
* Repeat test with user defined gain values

# Results

Part (i): the open loop model

**Table 1: experimentally determined constants**

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| m | kg | 1.07 |
| Bemf | kg/s | 7.72 |
| Beq | kg/s | 3.39 |
| c | kg/s | 11.11 |
| λ | N/(Ω\*A) | 1.72 |

Part (ii): model validation and saturation

**Table 2:** Comparison of experimentally observed value for Beq and the adjusted Beq value.

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| Beq - measured | kg/s | 3.39 |
| Beq - adjusted | kg/s | 2.50 |

Part (iii): the effect of coulomb friction

**Table 3:** experimentally determined coulomb friction

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| fc | N\*V/(Ω\*A) | 0.44 |

**Table 4:** comparison of experimentally observed steady state error and theoretical steady state error

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| |e(∞)| upper bound | m | 0.026 |
| |e(∞)| experimental | m | 0.033 |
| |e(∞)| simulated | m | 0.026 |

Part (iv): integral controller and coulomb friction

**Table 5:** largest value of ki that yields stable roots (all negative roots)

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| ki - max | V/m | 51.6 |
| Resulting Roots | N\*V/(Ω\*A\*ms) | -8.7729 |
|  |  | -0.38 + 3.05i |
|  |  | -0.378 - 3.05i |

Part (v): moving the cart with PID controller

**Table 6:** PID controller inputs – determined using SIMULINK modeling software prior to experimentation. Measured performance parameter outputs also shown.

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| PID inputs | | |
| kp | V/m | 1000 |
| ki | V/m | 20 |
| kd | V/m | 5 |
|  |  |  |
| Performance Parameters Output | | |
| tr | s | 0.6 |
| percent overshoot | % | 7.32 |
| ts | s | 0.74 |
| |e(∞)| | m | 0.003 |

**Table 7:** PID controller inputs – determined through experimentation. Measured performance parameter outputs also shown.

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| PID inputs | | |
| kp | V/m | 400 |
| ki | V/m | 5 |
| kd | V/m | 20 |
|  |  |  |
| Performance Parameters Output | | |
| tr | s | 0.64 |
| percent overshoot | % | 0.73 |
| ts | s | 0.58 |
| |e(∞)| | m | 0.002 |

# Analysis and Discussion

Part (i): the open loop model

A simplified equation of motion for a cart on a one dimensional track is given by:

|  |  |
| --- | --- |
|  | (1) |

Please note that this equation of motion neglects coulomb friction. This complication is discussed in part (iii). The constants and are determined by:

|  |  |
| --- | --- |
|  | (2) |
|  | (3) |

Where,

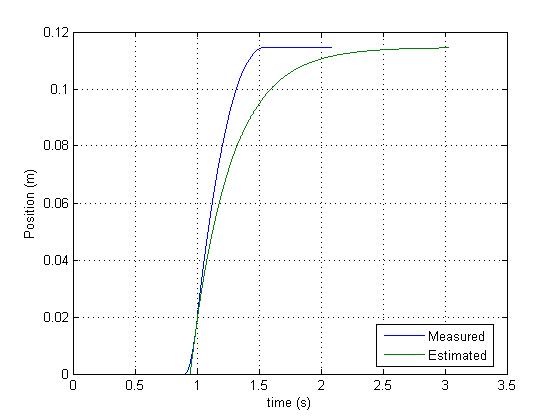
|  |  |
| --- | --- |
|  | (4) |
|  | (5) |
|  | (6) |

Note that from the given cart and track parameters, is readily calculated. However, cannot be calculated without determining . Recall that corresponds to the damping of the cart motion due to viscous friction. This value can be determined through experimentation.

More specifically, as explained in the procedure, the cart was given an initial push on the track without the motor plugged in. This eliminated back EMF. As such, the only damping force on the cart in motion is viscous friction force. By removing back EMF from the equation of motion, we observe that:

|  |  |
| --- | --- |
|  | (7) |

From the experimental data, all of the parameters in Equation 7 can be observed from the position output on the following figure.

****

**Figure 1: position of cart over time after cart receives a “push” to simulate an impulse input**

From Figure 1, we observe the experimental position when compared to the simulated position which utilizes the calculated value for .

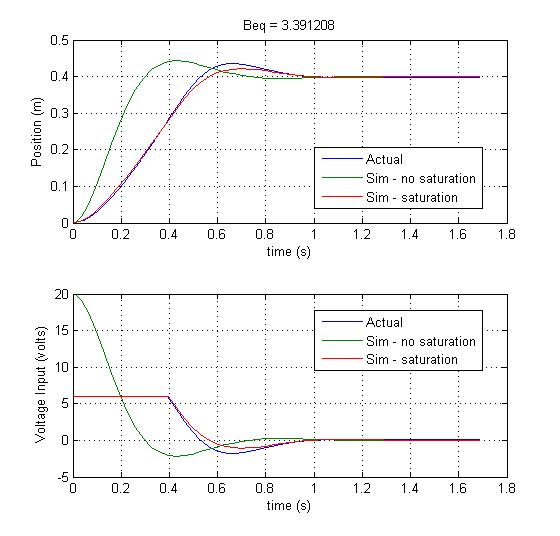
Note that the 2 plots do not coincide initially. This can be attributed to the fact that the simulated results assumed a perfect impulse input. This corresponds to an impulse delivered to the cart over an infinitesimally small period of time. In reality, this kind of push cannot be achieved – the force is delivered over a period of time. As such, we see that the cart position slowly speeds up initially until the pushing force is stopped. This causes error in the calculation of as well as the discrepancy in the data comparison.

Part (ii): model validation and saturation

Saturation is a nonlinear function where maximum and minimum limits are placed on the output variable. More specifically, in the case of the experiment, the motor voltage cannot exceed +/- 6 volts. These limits are the saturation factors for voltage input. A saturated voltage input looks like the following:

|  |  |
| --- | --- |
|  | (8) |

Part (ii) of the experiment demonstrated the effect of saturation on input voltage. See the following plot:



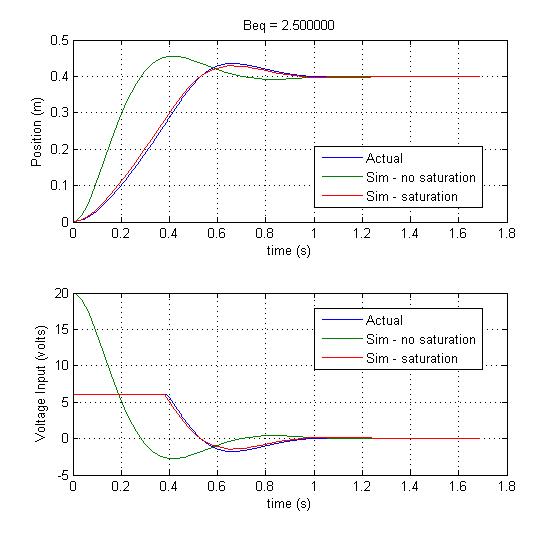
**Figure 2:** voltage input and position output with a constant gain feedback. Note that experimentally collected results are compared to simulation results.

These plots were generated through the use of MATLAB. The simulated data was obtained through the use of SIMULINK models – these models can be found in the Appendix.

From Figure 2, we observe that the SIMULINK model which did not utilize saturation on voltage input achieved a quicker response time. This can be attributed to the fact that the model associates large voltage with a large forcing function on the cart. With a larger forcing function, pushing the cart with more force, the cart is able to arrive at its destination sooner.

Conversely, the SIMULINK model with saturation is slower and matches the experimental data more closely. As seen, saturation forces the voltage input to never exceed +/- 6 volts.

We also observe from Figure 2 that the saturated SIMULINK model does not perfectly match the experimental data. This discrepancy can be attributed to the inaccurate value for calculated in part (i). In order to determine a better value for , the value was tweaked until the SIMULINK results matched the experimental results more closely.



**Figure 3:** Same as Figure 2 except that Beq was tweaked to make the simulation results match the experimentally determined results.

Part (iii): the effect of coulomb friction

Coulomb friction is a nonlinear damping generated by sliding friction. Equation 9 illustrates this:

|  |  |
| --- | --- |
|  | (9) |

When added to the simplified equation of motion described in Equation (1), we obtain the following completed equation of motion:

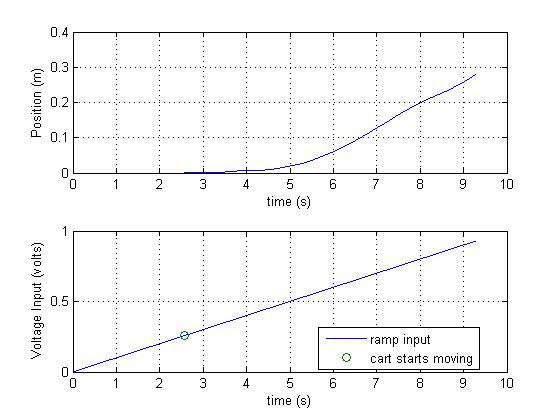
|  |  |
| --- | --- |
|  | (10) |

From Equations (9) and (10), we see that coulomb friction behaves similar to back EMF and viscous friction. Coulomb friction is not linear like back EMF and viscous friction, but it always works opposite of the direction of motion.

Similar to part (i), we can perform an experiment to observe the value for the coulomb friction constant: . As described in the procedure, we apply a ramp voltage input to the cart. In this situation, the cart will not move until the voltage input is strong enough to overcome the coulomb friction force. Noting that the position velocity and acceleration are zero until the cart starts moving, we can simplify Equation 10 to the following:

|  |  |
| --- | --- |
|  | (11) |
| Where: |  |

The following figure illustrates the experimental data:

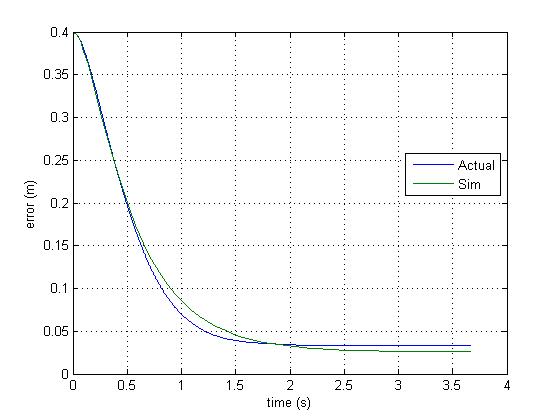


**Figure 4:** experimentally determined position output from a ramp voltage input

This plot was generated using MATLAB to load and analyze the experimental data.

From Figure 4, we observe that the cart begins to move at 2.57 seconds. This corresponds to a voltage input of 0.26 volts. With this, we are able to determine from Equation (11). This value is shown in Table 3.

In the 2nd portion of part (iii), the effect of coulomb friction on a closed loop system was observed. As described in the procedure, a constant proportional gain was applied. This resulted in the following data:



**Figure 5:** comparison of experimentally observed error and simulated error over time with a constant gain feedback on voltage input

From Figure 5, we observe that experimental steady-state error is nonzero. This can be attributed to coulomb friction. More specifically, coulomb friction allows for a theoretical upper bound for steady-state error. This bound was calculated and can be found in Table 4.

SIMULINK was used to model the effect of coulomb friction. This model can be found in the Appendix. Note that the calculated value for is utilized by this model. The resulting simulation data was plotted against experimental data in Figure 5.

The actual and simulated steady-state errors can be found in Table 4. Note from Table 4 that the simulated steady state error matches the upper bound. However, the measured steady-state error exceeds the upper bound. This can be attributed to error in measurement of . More specifically, the calculation of neglects the force of static friction.

Part (iv): integral controller and coulomb friction

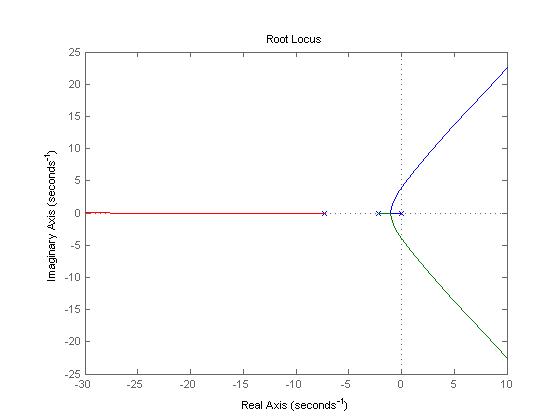
An integral control is a form of closed-loop feedback where the output term is proportional to both the magnitude and duration of input error. Integral controllers are useful for reducing steady-state error. At small error values, a proportional controller moves the output slowly. However, an integral controller accelerates the movement at small error values due to the accumulation of time in error.

An integral controller is described by the following equation:

|  |  |
| --- | --- |
|  | (12) |

Note that large values for can result in instability. More specifically, the integral control may respond to harshly and overshoot the target. This results in large error. Since cumulative error affects the integral controller, this previous overshoot may cause the controller to overshoot even more harshly on the next iteration. This results in an unstable divergence.

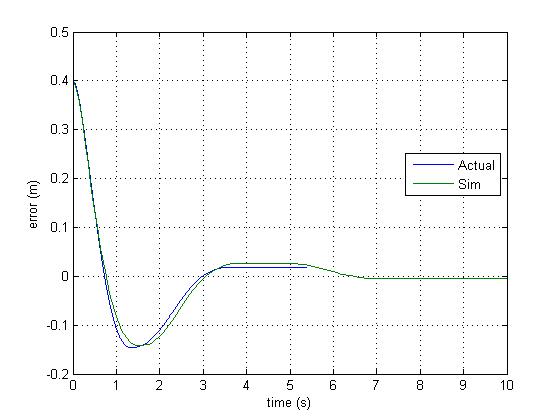
A root locus plot can be utilized to determine the range of stability for . This analysis and the root locus can be seen below:



**Figure 6:** root locus illustrating values for which ki is stable

From this root locus, we observe that the integral controller is stable when all roots lie in the left-hand plane. The largest value of that satisfies this stability requirement was determined from Figure 7 and can be found in Table 5. Note that the resulting roots, in Table 5, all lie in the left-hand plane of the root locus.

In the 2nd part of part (iv), the effect of an integral controller on cart position was demonstrated. This yielded the following data:



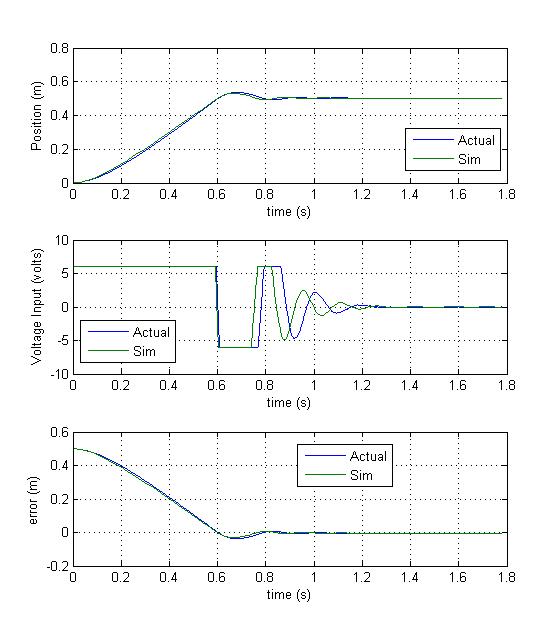
**Figure 7:** comparison of experimentally observed error and simulated error over time with a PI controller

SIMULINK was used to model the integral controller – the model can be found in the Appendix. From Figure 7, we see that the experimental and simulation results were very similar.

From the experimental data, we see that error was nonzero at 5 seconds. However, the simulated data suggests that steady-state error does indeed go to zero after a long period of time. This indicates that the controller gain values can be optimized for better feedback. This also indicates how an integral controller reduces steady-state error.

Part (v): moving the cart with PID controller

Part 5 demonstrated a PID controller – utilizing gain values for proportional, integral, and derivative feedback. Prior to experimentation, a SIMULINK model (in the Appendix) was utilized to simulate the cart motion with a given PID controller. Table 6 describes the PID controller designed in SIMULINK. The following figure illustrates the simulated cart position in comparison with experimentally collected data:

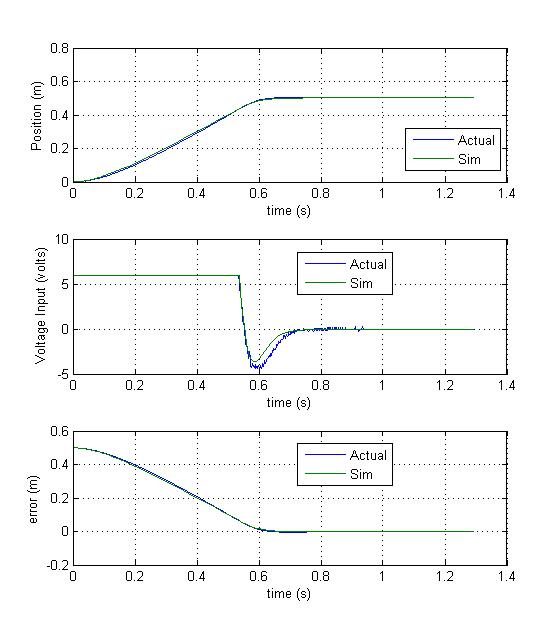


**Figure 8:** position, voltage input, and error resulting from PID inputs in Table 6

From this data, we see that the controller harshly overshoots the target position. With a max overshoot of 7%, this overshoot is harsh enough to cause the input voltage to saturate several times before damping to zero. The design objectives required that the controller be capable of achieving a settling time of less than one second with a max overshoot of 5%. While the controller satisfies the settling time requirement, the max overshoot requirement is not met. As such, the controller gain values must be modified to meet design objectives.

The simulation results and experimental results were very similar. The settling time between the two datasets was different by only 0.04 seconds. That difference in max overshoot is only 0.001%. This indicates an accurate simulation. The small error can be attributed to a simplification of the SIMULINK model – coulomb friction was neglected.

In the 2nd part of part (v), the PID gains were adjusted until the controller design objectives were met. Table 7 describes the designed controller and resulting performance parameter outputs. Figure 9 illustrates the data:

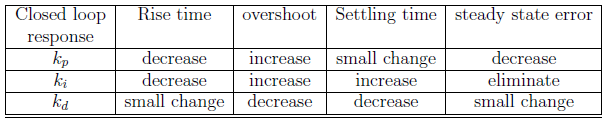


**Figure 9:** position, voltage input, and error resulting from PID inputs in Table 7

From Table 7, we see that the design criteria are met by this controller. Furthermore, Figure 7 illustrates how there is very little overshoot. This results in a very quick settling time.

Table 8 describes the effect of gains on a PID controller:

**Table 8:** effect of gains on PID controller



The proportional constant is the simplest. At large error, the input is large. At small error, the input is small. If this constant is very large, this may result in overshoot.

The integral constant functions very similarly to the proportional constant. At large values, the system is prone to overshoot. In the case of saturated input, the system may even become unstable due to integral windup. However, integral controllers are able to decrease the steady state error. At low error, the proportional controller has little effect. However, as time passes, error accumulates within the integral controller and drives an accelerated response, eliminating steady state error.

The derivative constant is used to decreases overshoot and settling time. By predicting system behavior, a derivative controller can be very powerful. However, derivative controllers are vulnerable to noisy input where the slope (and thus, the output) changes harshly.

The following equation illustrates a PID controller response:

|  |  |
| --- | --- |
|  | (13) |

# Conclusion and Recommendation

Main Points

The motion of a cart on a one dimensional track can be described by a relatively simple ordinary differential equation. In this case, the force acting on the cart is from an electric motor. Various damping forces work against the direction of motion of the cart and act to halt the cart. Viscous friction and back EMF are 2 identified sources of linear damping. Coulomb friction is also identified as a source of nonlinear damping.

Viscous friction can be obtained experimentally by tapping the cart and analyzing the time required for the cart to stop. Coulomb friction can be obtained experimentally by applying a ramp voltage input to the cart and observing the voltage at which the cart overcomes coulomb friction.

Voltage input to the cart saturates at +/- 6 volts. This nonlinearity cannot be ignored when designing modeling software.

SIMULINK is an effective tool for modeling the cart motion and predicting cart response to control inputs. With a reliable SIMULINK model, a better value for viscous friction damping constant can be obtained than through the tapping method denoted above.

Coulomb friction creates steady state error when the cart is controlled via proportional closed loop feedback. This steady state error can be eliminated with the use of an integral controller. However, care must be taken when specifying the integral controller constant since too large of a value can cause instability. A root locus can be used to identify the maximum possible integral constant for stability.

With an accurate SIMULINK model, a PID controller can be simulated and optimized to meet design requirements with hardware out of the loop. This is desirable for when it is not cheap, or possible to test PID control inputs with hardware.

Theoretical/ Experimental Limitations

In part (i) of the experiment, coulomb friction was ignored in order to experimentally obtain the viscous friction constant. Without this simplification, this would not have been possible.

Throughout the experiment, it is assumed that input voltage corresponds linearly to the force applied to the cart by the motor. (Within saturation range.) While this is close to accurate, no electric motor is perfectly linear in terms of output.

In part (iii), coulomb friction was obtained experimentally by determining the voltage at which the cart started moving from being at rest. This analysis excluded static friction for simplicity.

In part (v), the effect of coulomb friction was ignored for simplicity. The simulated results still matched the experimental results very closely.

The experiment is limited by quality of the cart, track, and sensors. The geared wheels may experience greater friction force as they wear out. Additionally, the power/control cable to the cart may apply a small resistance – possible noticeable at low speeds.

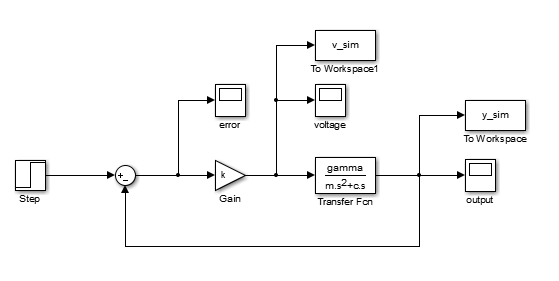
Personal Lessons Learned and Suggestions for Improvement

I personally benefited from learning how SIMULINK can be used to model and simulate real-world control problems. More importantly, SIMULINK can be used to prototype and optimize a PID controller with hardware out of the loop. This becomes advantageous when testing the hardware is not possible or is expensive.

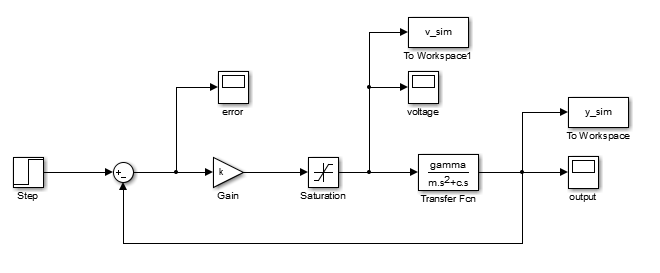
I also learned how the proportional, integral, and derivative gain each have a different effect on signal response. Care must be taken to ensure that these values aren’t too large such that they create instabilities.

My sole suggestion is that the TA hold a competition to see whose PID gains from pre-lab resulted in the best performance. This would inspire students to spend more time tweaking and becoming more familiar with PID controller optimization. Furthermore, our group finished the lab 1 hour of extra time. This indicates that the few extra minutes required for a competition would be feasible.

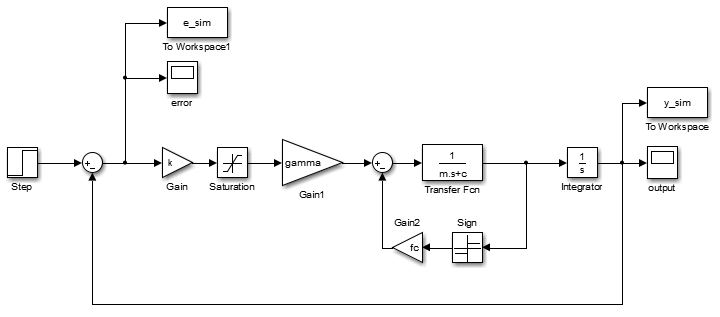
# Appendix



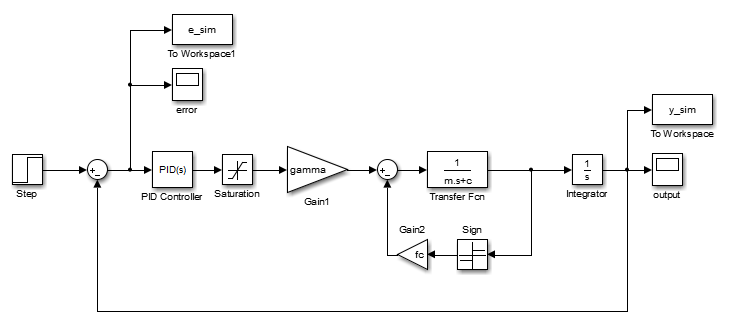
**Model A1:** proportional feedback without saturation



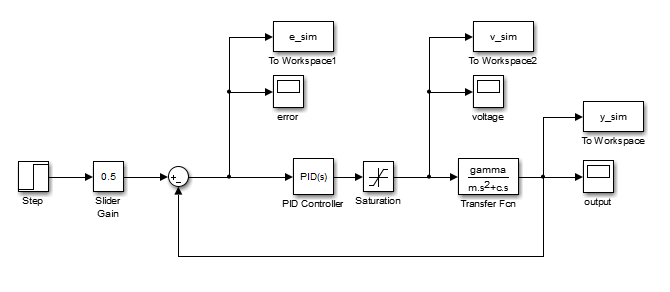
**Model 2A:** proportional feedback with saturation



**Model 3A:** Model simulating coulomb friction



**Model 4A:** PI controller with coulomb friction



**Model 5A:** PID controller without coulomb friction

Code:

clc; clear; close all;

addpath('Data');

%% Given Constants

Rm = 2.6; %motor armature resistance 2.6 ?

Lm = 0.18; %motor armature inductance 0.18 mH

Kt = 0.00767; %motor torque constant 0.00767 N.m/A

nu\_m = 1.00; %motor e?ciency 100% %

Km = 0.00767; %back-electromotive-force(EMF) 0.00767 V.s/rad

Jm = 3.9E-7; %rotor moment of inertia 3.9 × 10?7 kg.m2

Kg = 3.71; %planetary gearbox ratio 3.71

nu\_g = 1.00; %planetary gearbox e?ciency 100% %

Mc = 0.57; %cart mass 0.57 kg

Mw = 0.37; %cart weight mass 0.37 kg

Lt = 0.990; %track length 0.990 m

Tc = 0.814; %cart travel 0.814 m

Pr = 1.664E-3; %rack pitch 1.664 × 10?3 m/tooth

rmp = 6.35E-3; %motor pinion radius 6.35 × 10?3 m

Nmp = 24; %motor pinion number of teeth 24

rpp = 0.01482975; %position pinion radius 0.01482975 m

Npp = 56; %position pinion number of teeth 56

KEP = 2.275E-5; %cart encoder resolution 2.275 × 10?5 m/count

%% Part (i)

%find gamma, m, Bemf

M = Mc + Mw; %kg total cart system mass

Mj = (nu\_g\*Kg^2\*Jm)/(rmp^2); %effective mass added to the system due to the moment of inertia of the motor

m = M + Mj %

Bemf = (nu\_g\*Kg^2\*nu\_m\*Kt\*Km)/(Rm\*rmp^2)

gamma = (nu\_g\*Kg\*nu\_m\*Kt)/(Rm\*rmp)

%Load collected data

load('Justice\_1\_y')

t = y.time;

t = t-t(1);

y = y.signals.values;

figure(1), plot(t,y)

xlabel('time (s)')

ylabel('Position (m)')

grid on

hold all

%estiamte Beq

ydot\_0 = 0.3621; %observed from graph

syms Beq

Beq = double(solve(y(end) == ydot\_0\*m/Beq))

c = Beq + Bemf

%Plot esitamted curve with Beq

y\_estim = (ydot\_0\*m/Beq)\*(1 - exp(-Beq\*t/m));

plot(t+0.944,y\_estim)

legend('Measured','Estimated',0)

%% Part (ii)

for n = 1:1:2

if n == 2

Beq = 2.5;

c = Beq + Bemf;

end

%Load collected data

load('Justice\_2\_y')

load('Justice\_2\_v')

t = y.time;

y = y.signals.values;

v = v.signals.values;

figure(n+1)

subplot(2,1,1), plot(t,y)

title(sprintf('Beq = %f',Beq))

xlabel('time (s)')

ylabel('Position (m)')

grid on

hold all

subplot(2,1,2), plot(t,v)

xlabel('time (s)')

ylabel('Voltage Input (volts)')

grid on

hold all

%Sim 2a

k = 50; %V/m

r0 = 0.4; %m

t\_end = t(end);

sim('part2a')

subplot(2,1,1), plot(y\_sim.Time,y\_sim.Data)

hold all

subplot(2,1,2), plot(v\_sim.Time,v\_sim.Data)

hold all

%Sim 2b

k = 50; %V/m

r0 = 0.4; %m

t\_end = t(end);

sim('part2b')

subplot(2,1,1), plot(y\_sim.Time,y\_sim.Data)

legend('Actual','Sim - no saturation','Sim - saturation',0)

subplot(2,1,2), plot(v\_sim.Time,v\_sim.Data)

legend('Actual','Sim - no saturation','Sim - saturation',0)

end

%% Part (iii)a

%Load collected data

load('justice\_3\_y')

load('justice\_3\_v')

t = y.time;

y = y.signals.values;

v = v.signals.values;

figure(4)

subplot(2,1,1), plot(t,y)

xlabel('time (s)')

ylabel('Position (m)')

grid on

hold all

subplot(2,1,2), plot(t,v)

xlabel('time (s)')

ylabel('Voltage Input (volts)')

grid on

hold all

%determine when cart starts moving

n = 0;

flag = 0;

while flag == 0

n = n+1;

if y(n)>0

t0 = t(n);

v\_t0 = v(n);

flag = 1;

end

end

subplot(2,1,2), plot(t0,v\_t0,'o')

legend('ramp input','cart starts moving',0)

fc = gamma\*v\_t0

%% Part (iii)b

load('justice\_3\_e')

t = e.time;

e = e.signals.values;

figure(5)

plot(t,e)

xlabel('time (s)')

ylabel('error (m)')

grid on

hold all

%Sim

k = 10; %V/m

r0 = 0.4; %m

t\_end = t(end);

sim('part3')

plot(e\_sim.Time,e\_sim.Data)

legend('Actual','Sim',0)

%e\_inf

e\_inf\_max = fc/(k\*gamma)

e\_inf = e(end)

e\_inf\_sim = e\_sim.Data(end)

%% Part (iv)

load('justice\_4\_e')

t = e.time;

e = e.signals.values;

figure(6)

plot(t,e)

xlabel('time (s)')

ylabel('error (m)')

grid on

hold all

%sim

kp = 10;

ki = 15;

kd = 0;

t\_end = 10;

sim('part4')

plot(e\_sim.Time,e\_sim.Data)

legend('Actual','Sim',0)

%determine largest ki for stable root locus given parameters

k = 10; %V/m

num = [gamma];

denom = [m, c, gamma\*k, 0];

figure(7)

rlocus(num,denom)

ki\_max = 51.6; %observed from root locus plot

poly\_rooms = roots([m c gamma\*k gamma\*ki\_max])

%% Part (v)a

load('justice\_5\_y')

load('justice\_5\_v')

load('justice\_5\_e')

t = y.time(1:end/2);

y = y.signals.values(1:end/2);

v = v.signals.values(1:end/2);

e = e.signals.values(1:end/2);

figure(8)

subplot(3,1,1), plot(t,y)

xlabel('time (s)')

ylabel('Position (m)')

grid on

hold all

subplot(3,1,2), plot(t,v)

xlabel('time (s)')

ylabel('Voltage Input (volts)')

grid on

hold all

subplot(3,1,3), plot(t,e)

xlabel('time (s)')

ylabel('error (m)')

grid on

hold all

%analysis

Sa = stepinfo([0;y],[0;t],0.5,'RiseTimeLimits',[0 1],'SettlingTimeThreshold',0.05)

%sim

kp = 1000;

ki = 20;

kd = 5;

t\_end = t(end);

sim('part5')

subplot(3,1,1), plot(y\_sim.Time,y\_sim.Data)

legend('Actual','Sim',0)

subplot(3,1,2), plot(v\_sim.Time,v\_sim.Data)

legend('Actual','Sim',0)

subplot(3,1,3), plot(e\_sim.Time,e\_sim.Data)

legend('Actual','Sim',0)

%% Part (v)b

load('justice\_5b\_y')

load('justice\_5b\_v')

load('justice\_5b\_e')

t = y.time(1:end/2);

y = y.signals.values(1:end/2);

v = v.signals.values(1:end/2);

e = e.signals.values(1:end/2);

figure(9)

subplot(3,1,1), plot(t,y)

xlabel('time (s)')

ylabel('Position (m)')

grid on

hold all

subplot(3,1,2), plot(t,v)

xlabel('time (s)')

ylabel('Voltage Input (volts)')

grid on

hold all

subplot(3,1,3), plot(t,e)

xlabel('time (s)')

ylabel('error (m)')

grid on

hold all

%analysis

Sb = stepinfo([0;y],[0;t],0.5,'RiseTimeLimits',[0 1],'SettlingTimeThreshold',0.05)

%sim

kp = 400;

ki = 5;

kd = 20;

t\_end = t(end);

sim('part5')

subplot(3,1,1), plot(y\_sim.Time,y\_sim.Data)

legend('Actual','Sim',0)

subplot(3,1,2), plot(v\_sim.Time,v\_sim.Data)

legend('Actual','Sim',0)

subplot(3,1,3), plot(e\_sim.Time,e\_sim.Data)

legend('Actual','Sim',0)

Pre-lab

# Part (i)

%% Given

Rm = 2.6; %motor armature resistance 2.6 ?

Lm = 0.18; %motor armature inductance 0.18 mH

Kt = 0.00767; %motor torque constant 0.00767 N.m/A

nu\_m = 1.00; %motor e?ciency 100% %

Km = 0.00767; %back-electromotive-force(EMF) 0.00767 V.s/rad

Jm = 3.9E-7; %rotor moment of inertia 3.9 × 10?7 kg.m2

Kg = 3.71; %planetary gearbox ratio 3.71

nu\_g = 1.00; %planetary gearbox e?ciency 100% %

Mc = 0.57; %cart mass 0.57 kg

Mw = 0.37; %cart weight mass 0.37 kg

Lt = 0.990; %track length 0.990 m

Tc = 0.814; %cart travel 0.814 m

Pr = 1.664E-3; %rack pitch 1.664 × 10?3 m/tooth

rmp = 6.35E-3; %motor pinion radius 6.35 × 10?3 m

Nmp = 24; %motor pinion number of teeth 24

rpp = 0.01482975; %position pinion radius 0.01482975 m

Npp = 56; %position pinion number of teeth 56

KEP = 2.275E-5; %cart encoder resolution 2.275 × 10?5 m/count

%% Part (i)

%find gamma, m, Bemf

M = Mc + Mw; %kg total cart system mass

Mj = (nu\_g\*Kg^2\*Jm)/(rmp^2); %effective mass added to the system due to the moment of inertia of the motor

m = M + Mj %

Bemf = (nu\_g\*Kg^2\*nu\_m\*Kt\*Km)/(Rm\*rmp^2)

gamma = (nu\_g\*Kg\*nu\_m\*Kt)/(Rm\*rmp)

MATLAB OUTPUT:

m = 1.0731 (kg)

Bemf = 7.7236

gamma = 1.7235

# Part (ii)

%% Part (ii)

%determine largest ki for stable root locus given parameters

k = 10; %V/m

Beq = 5.4; %kg/s

c = Beq + Bemf;

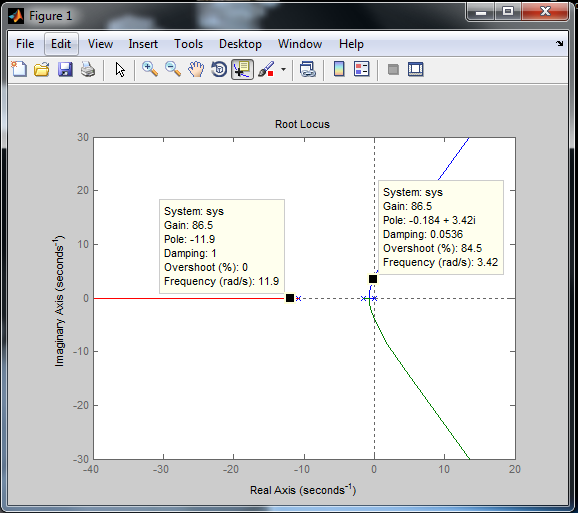
num = [gamma];

denom = [m, c, gamma\*k, 0];

rlocus(num,denom)

ki\_max = 86.5;

poly\_rooms = roots([m c gamma\*k gamma\*ki\_max])



At ki\_max = 86.5,

poly\_rooms =

-11.8626 + 0.0000i

-0.1833 + 3.4173i

-0.1833 - 3.4173i

# Part (iii)

%% Part (iii)

kp = 1000;

ki = 20;

kd = 5;

|  |  |  |
| --- | --- | --- |
| variable | units | value |
| PID inputs | | |
| kp | V/m | 1000 |
| ki | V/m | 20 |
| kd | V/m | 5 |
|  |  |  |
| Performance Parameters Output | | |
| tr | s | 0.6 |
| percent overshoot | % | 7.32 |
| ts | s | 0.74 |
| |e(∞)| | m | 0.003 |

