Week 10 Overview

Tuesday, November 08, 2016 2:39 PM

- [x] Set up team log book
- [x] Find S4 motor model
    - [x] Add to git repo
    - [x] Update with specs from S5 motor
- [x] Set up GIT repo for project
    - [x] Add datasheets, charter, S4 motor model, etc

Motor Modelling

Thursday, November 10, 2016 3:48 PM

Semester 4 motor model parameters

From: http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SimulinkModeling

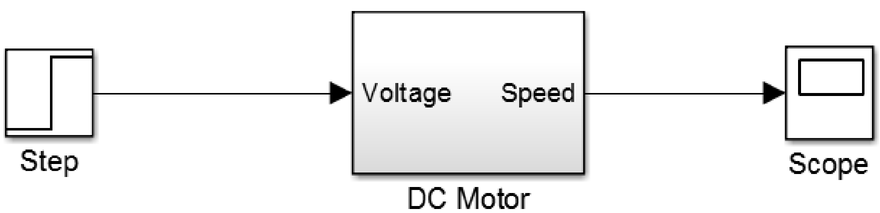
J = moment of inertia of the rotor = kg * m^2
b = motor viscous friction constant (damping) = N * m * s
Ke = electromotive force constant = V/rad/sec
Kt = motor torque constant = N * m / Amp
R = motor electrical resistance = ohms
L = motor electrical inductance = henrys

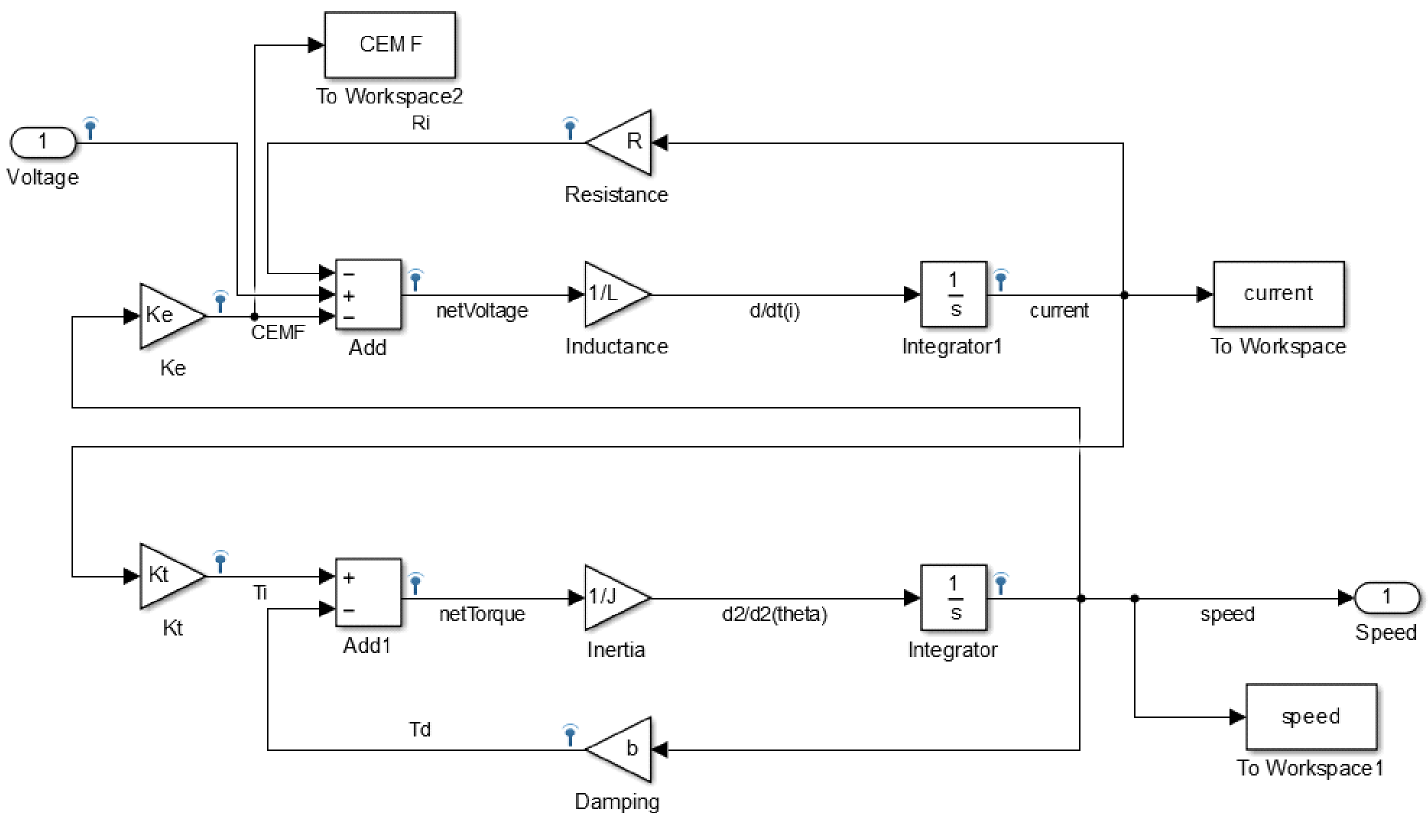
Semester 5 Motor:

ElectroCraft DPP240-29V48

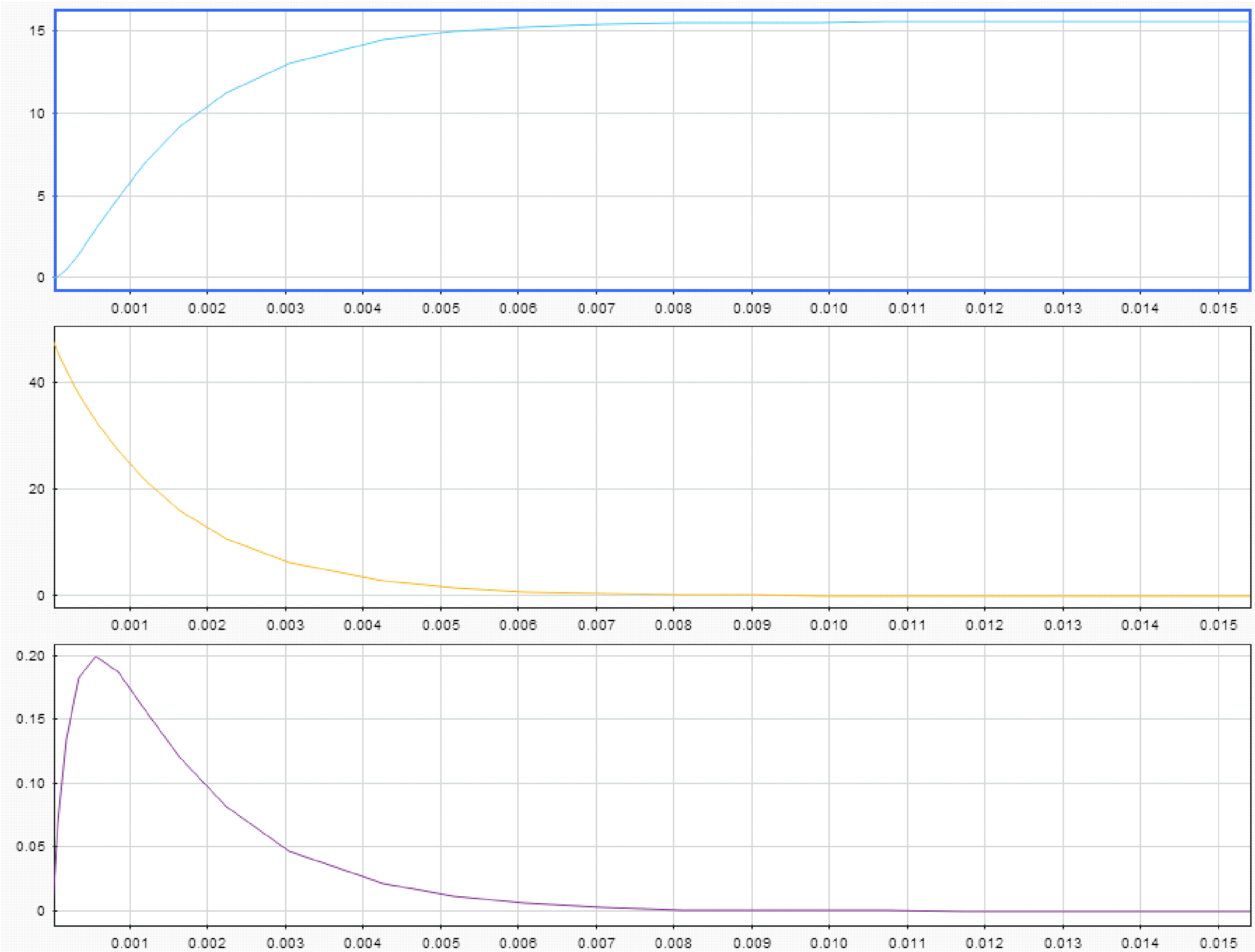
J = 282.5 g cm^2 = 2.825×10^{-5}
b = NOT LISTED = USE 0.1 for now per Bill instructions
Ke = 10.9 V/kRPM = 0.0109 V/rpm = 0.00141 V/rad/s = 1.141×10^{-3} V/rad/s
Kt = 14.7 oz-in / Amp = 0.1038 N*m / Amp
R = 3.2 Ohms
L = 4.8 mH = 4.8×10^{-3} H

Model In Simulink





Model Output - Driven by 48V Step Input - Speed - Net Voltage - Net Torque



Week 11 Overview

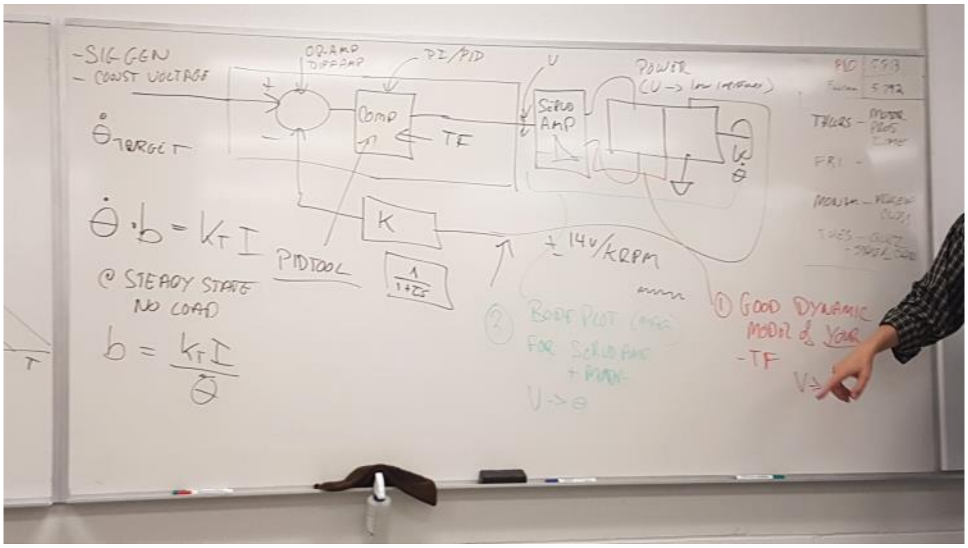
Thursday, November 17, 2016 2:56 PM

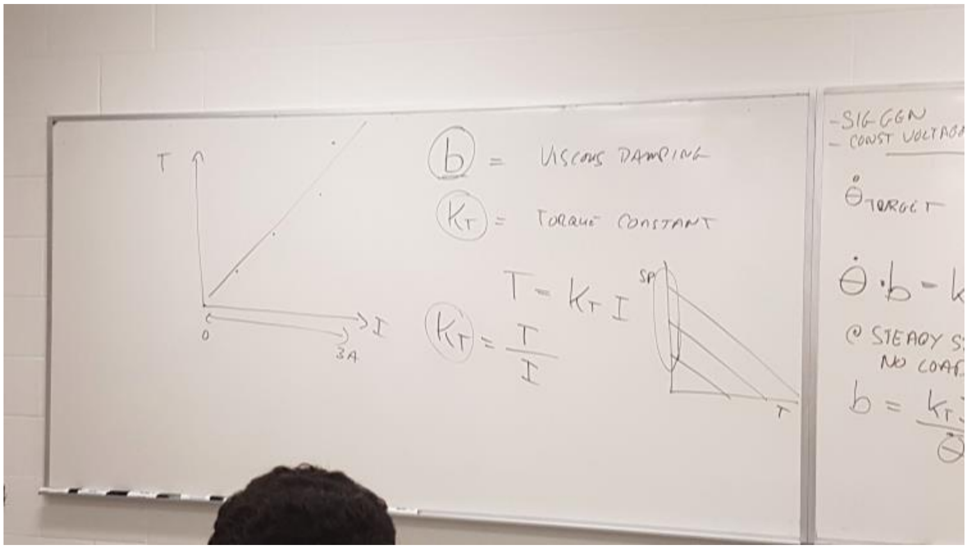
Objectives

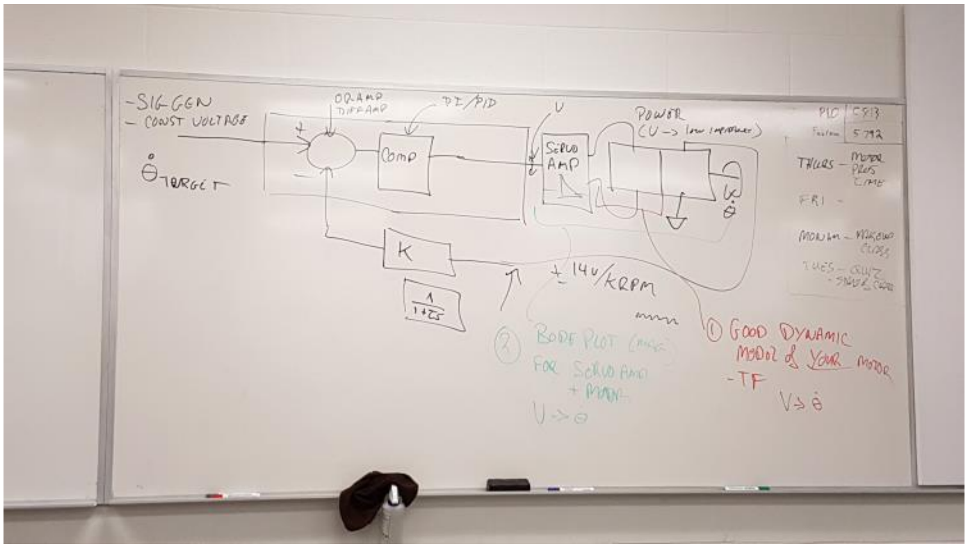
- [x] Measure real-world motor characteristics
- [x] Update motor model with real-world measurements
- [x] Tune motor model to match real-world step response
    - [x] Plot of step response from MATLAB model
    - [x] Scope shot of step response from real motor

Op-amp controlled motor

November 16, 2016 11:49 AM







Consider a low pass filter in the feedback section to eliminate high frequency ripple on the step response.

Use parallel form of the PI compensator.

Motor Measurements

Thursday, November 17, 2016 2:57 PM

Parameter	Units	Datasheet	Measurement	Method of Measurement
Rotor Inertia	Kg*m^2	$2.825 \times 10^{-5} + 9.886 \times 10^{-6} = 3.8136 \times 10^{-5}$	2.5×10^{-5}	MATLAB Tuning
Rotor Damping	N*m*s	(0.1 estimate)	0.000232	Current vs Speed measurements
Voltage Constant	V/rad/s	0.104087	0.100	Torque watch
Torque Constant	N*m /A	0.1038	0.100	Torque watch
R	Ohms	3.2	4	Fluke DMM
Inductance	Henry	4.8×10^{-3}	5.270×10^{-3} Or 7.169×10^{-3}	LCR

Making sure it spins (no load):

PSU Voltage: 10V
Measured Current: 0.23A
Measured Tach Voltage: 12.5V
Motor Speed: 14/12.5*1000 = 1120 rpm

Torque Constant Estimate

K_T = 0.100 N*m/A
Measurements in Excel Spreadsheet.

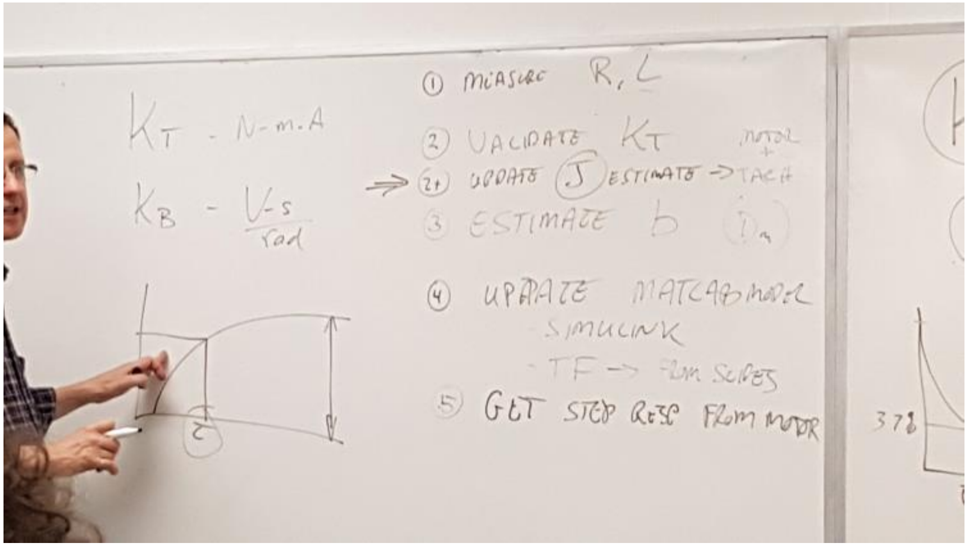
Damping Estimate

$$b = \frac{K_T I}{W} = \frac{Torque\ Constant\ * Current}{Angular\ Speed} = 0.000232\ N * m * s$$

Measurements in Excel Spreadsheet.

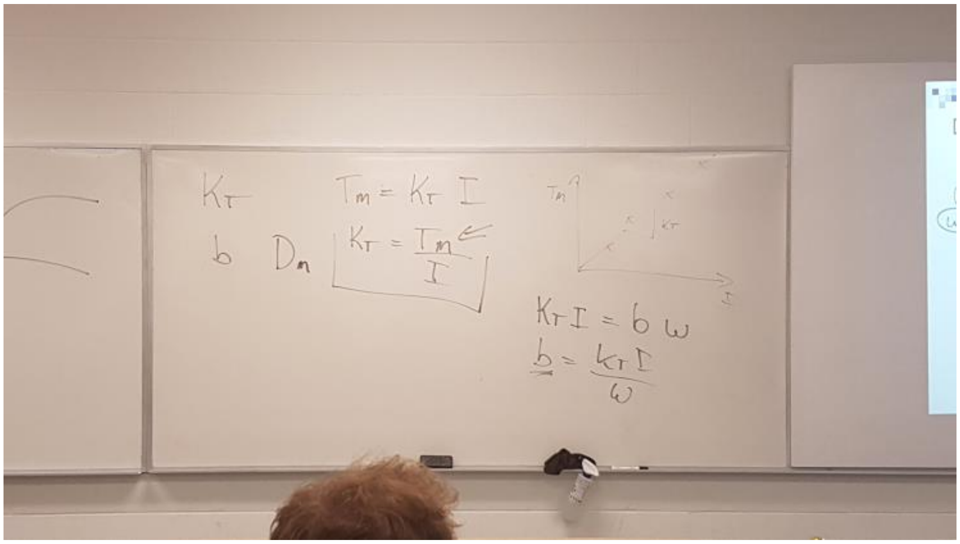
To do list

November 17, 2016 3:47 PM



Finding kt and b

November 17, 2016 3:48 PM

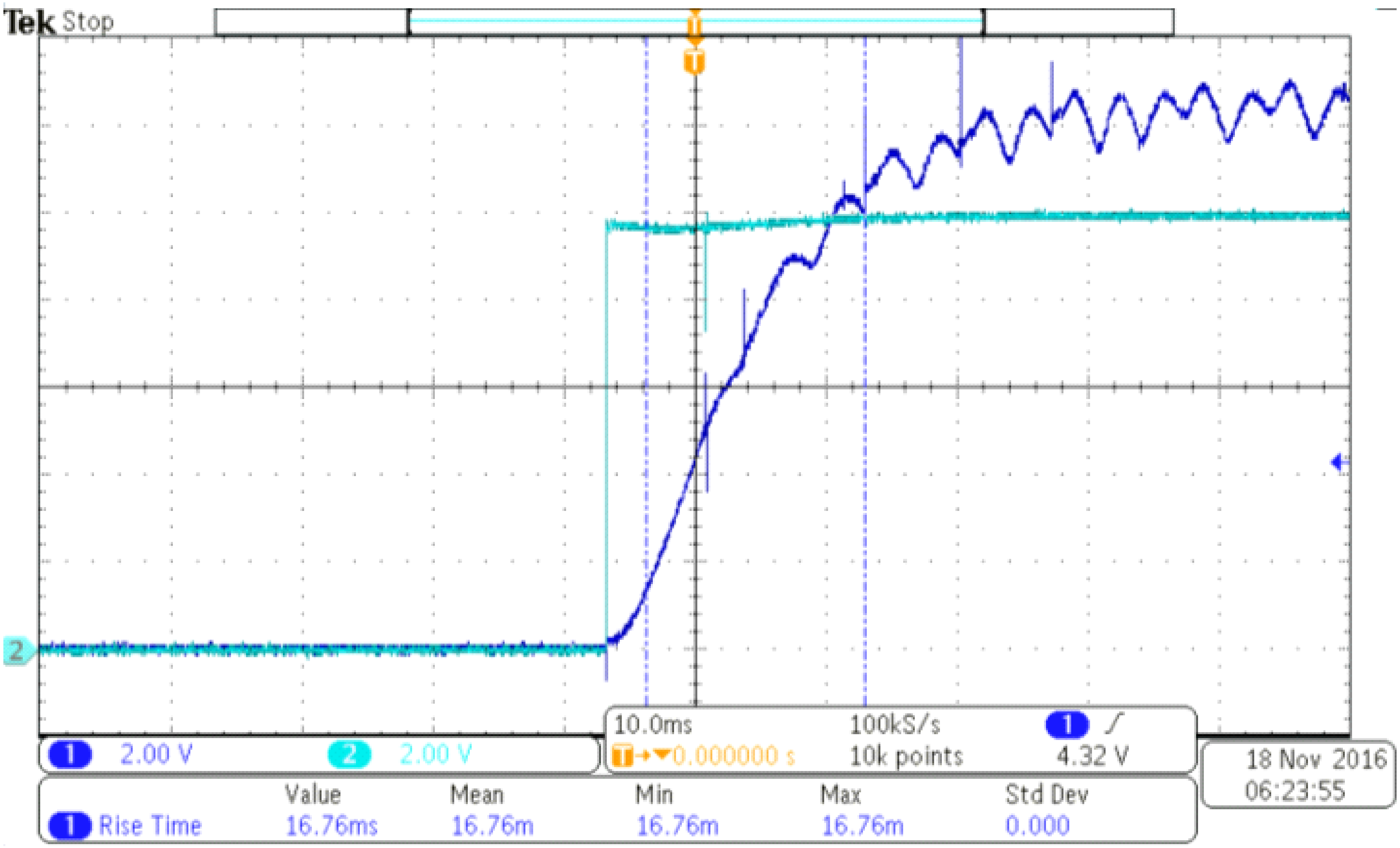


Step-Response Measurements & Tuning

Thursday, November 17, 2016 5:52 PM

Real-World Step Response

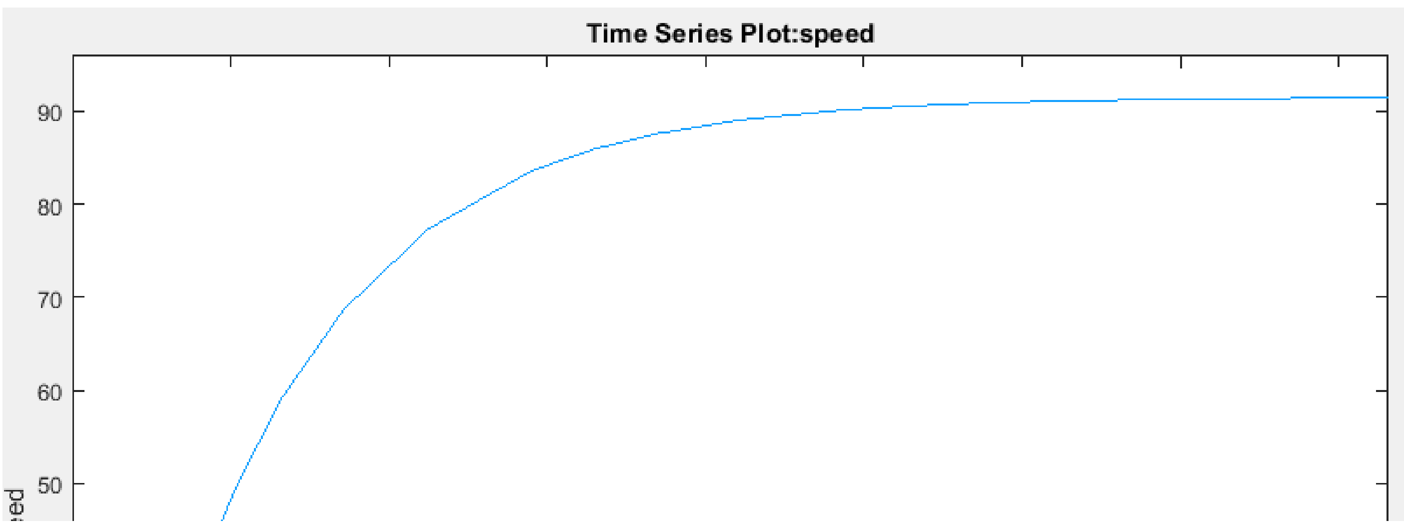
Tach Speed = 91.25 rad/s
10-90 Rise Time: 16.76ms

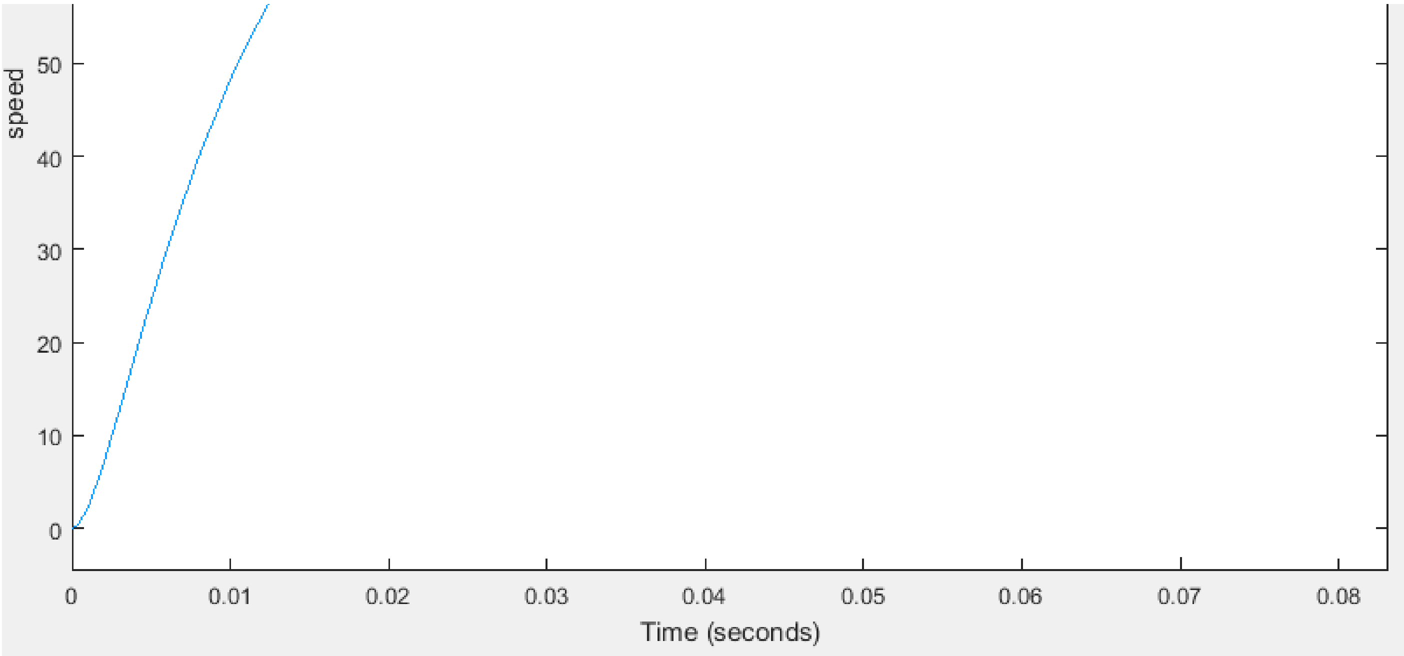


10V Step Response - Dark Blue = Tach Voltage

MATLAB Model Step Response

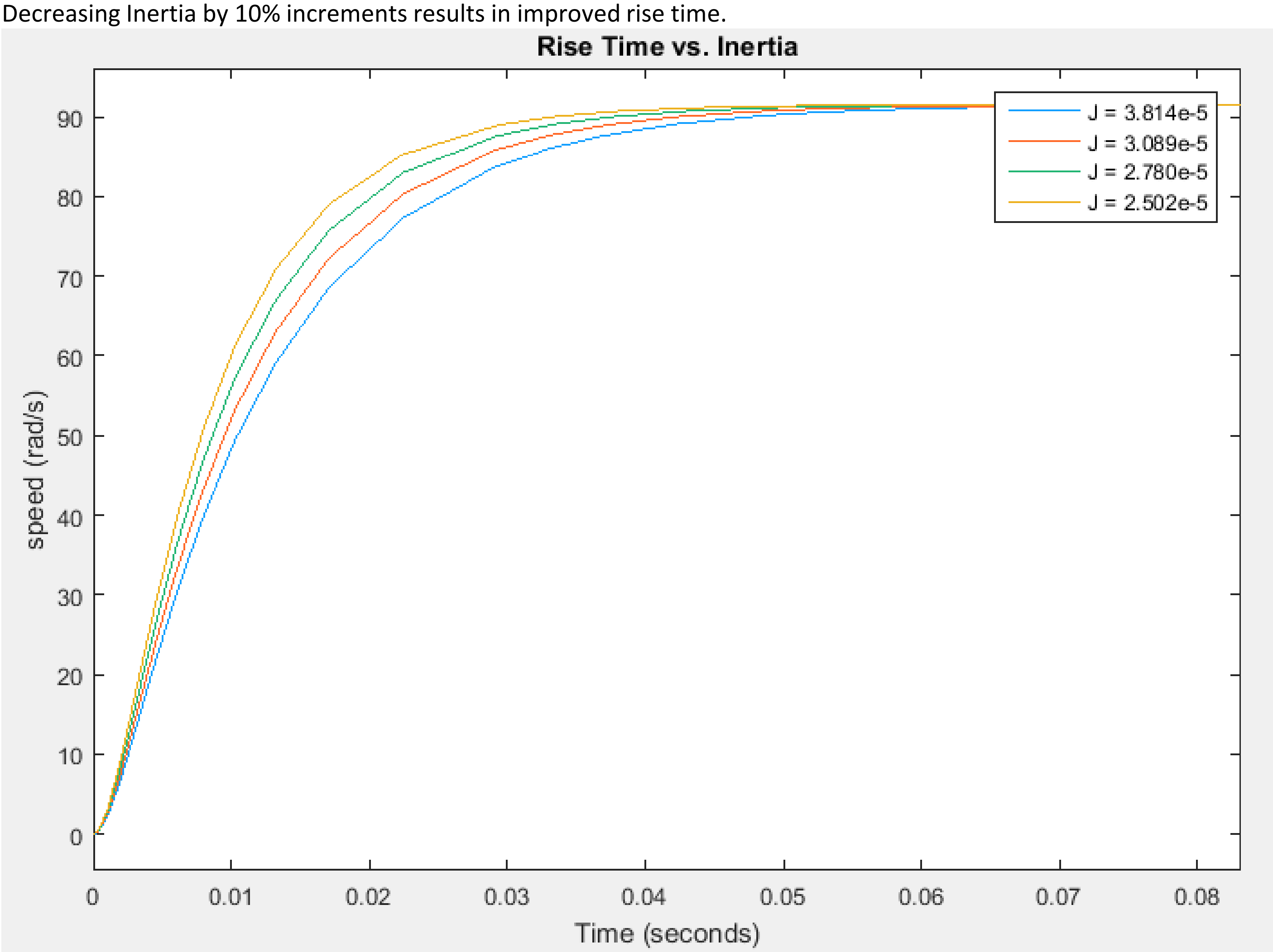
Steady State Speed = 91.3 rad/s
10-90 Rise Time: 26.5ms



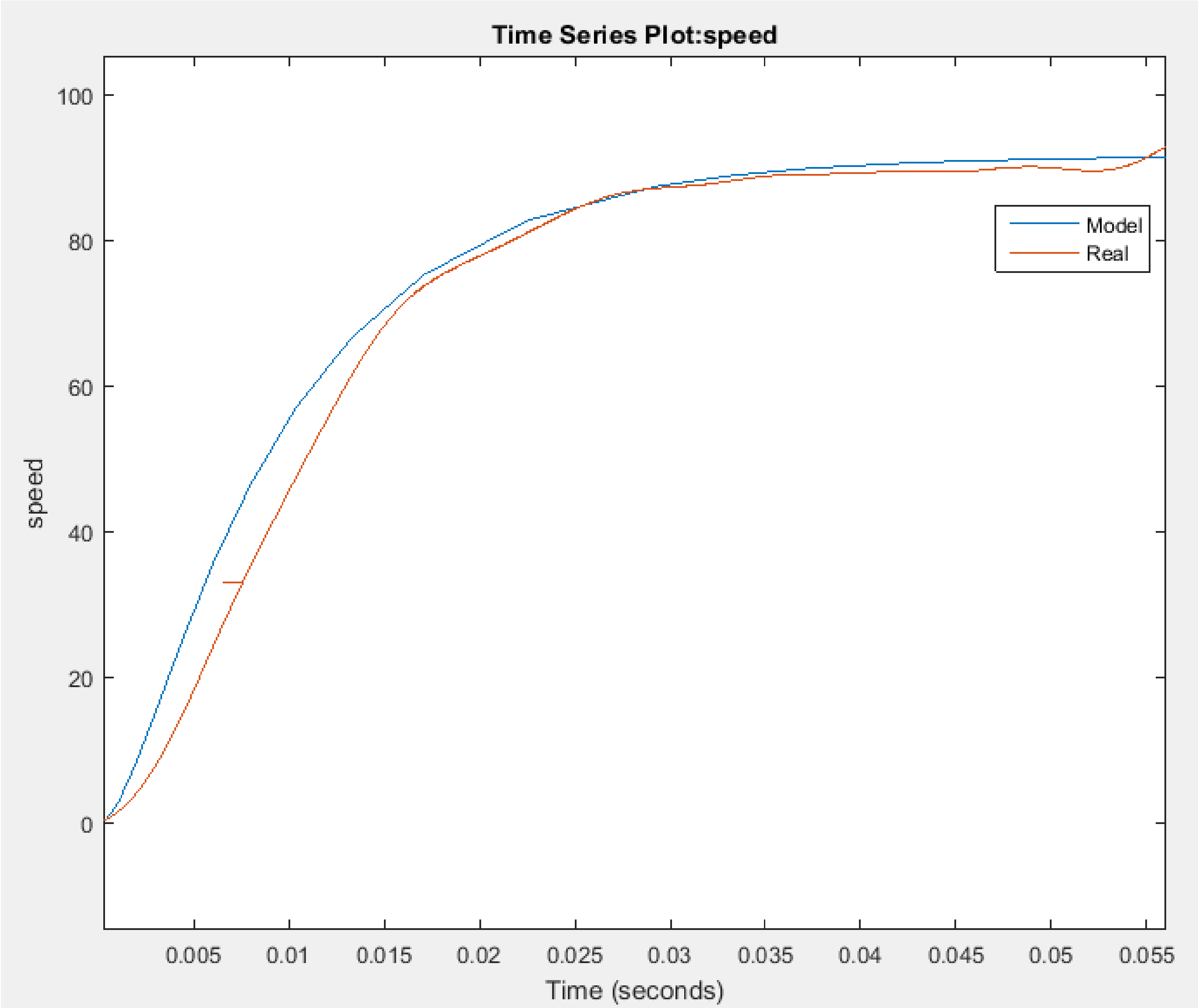


Tuning MATLAB Model (Inertia)

Decreasing Inertia by 10% increments results in improved rise time.



With Inertia set to be just the motor inertia, not adding the tach inertia, it looks good.



11/18/2016 2:10 PM - Screen Clipping

Transfer Function of Motor Model

Friday, November 18, 2016 12:38 PM

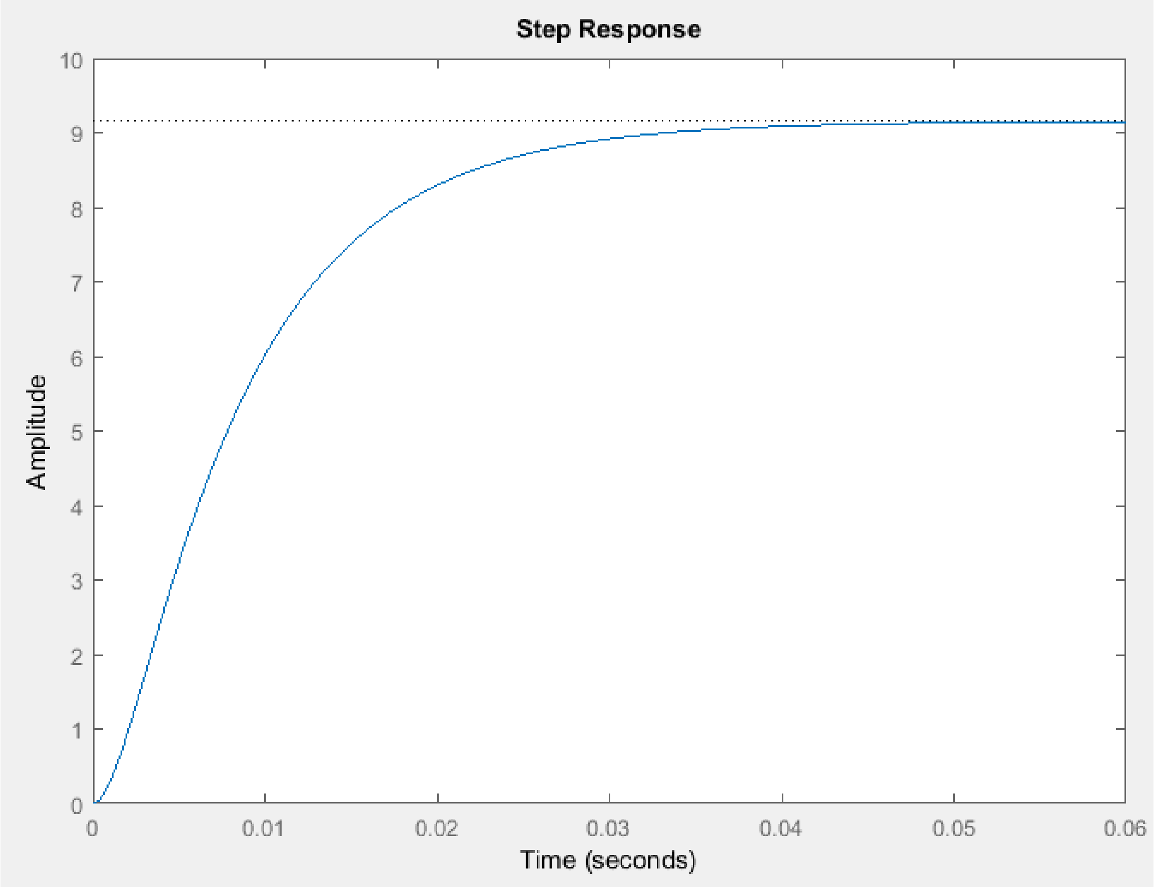
Transfer Function from Bill's Slides

$$G(s)=\frac{\omega(s)}{V(s)}=\frac{\frac{K_t}{LJ}}{s^2+\frac{(RJ+bL)}{LJ}s+\frac{Rb+K_eK_t}{LJ}}$$

Transfer Function as entered into Matlab

```
>> H=((Kt/(L*J))/((s^2)+((((R*J)+(b*L))/(L*J))*s)+(((R*b)+(Ke*Kt))/(L*J))))

H =

        7.59e05
  ------------------------
  s^2 + 768.3 s + 8.294e04

Continuous-time transfer function.
```

Step Response of Motor Model Transfer Function



Week 12 Overview

Tuesday, November 22, 2016 10:11 AM

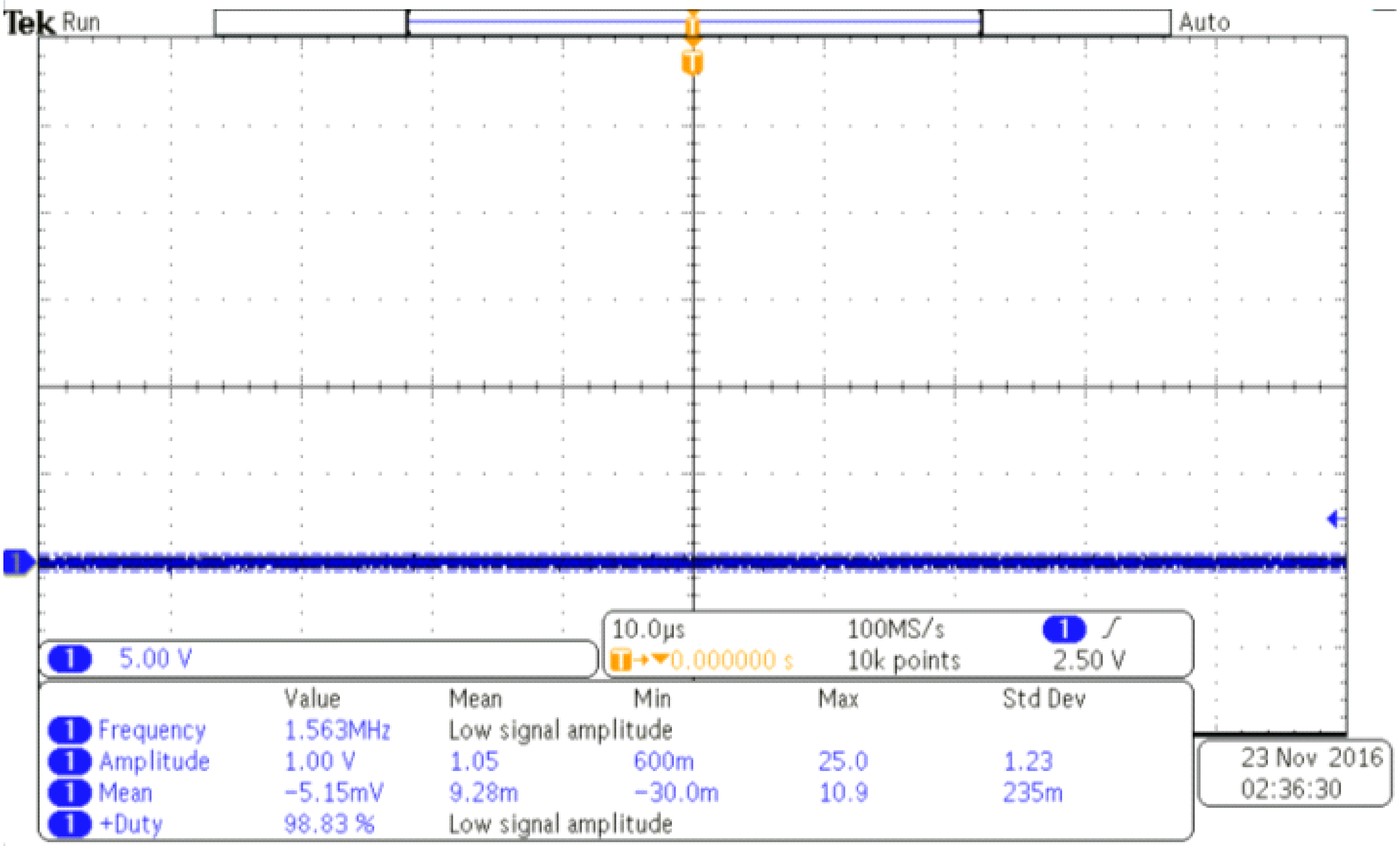
Objectives

- [x] Design & Build Op-Amp Speed Controller
    - [x] MATLAB Model of Servo-Amp
    - [x] MATLAB Model of Tachometer
    - [x] Updated Transfer Function
    - [x] PIDTOOL controller development
    - [x] Op-Amp Controller build/test

Servo-Amp Setup

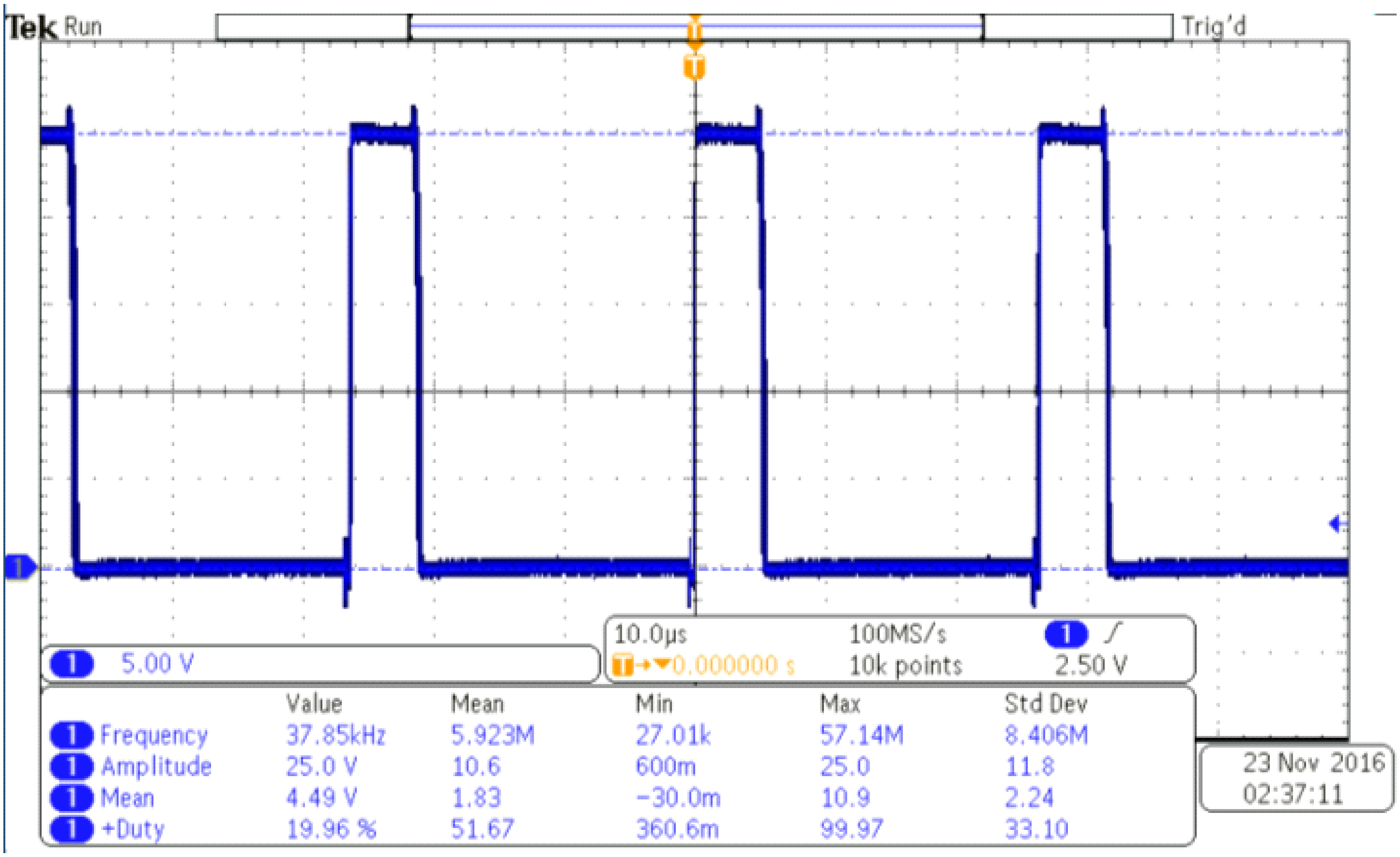
Tuesday, November 22, 2016 11:55 AM

Measuring Input Offset



Basically zero.

Setting Servo-Amp Gain = 1



5V Signal Input = 5V Output (20% * 25V)

PSU Set to 25V

Input Signal	Output Duty Cycle %	Effective Output Voltage
1	5.12	1.28
2	8.35	2.09
3	12.10	3.03
4	16.25	4.06
5	20.00	5.00
6	24.10	6.03
7	28.05	7.01
8	32.00	8.00
9	36.00	9.00
10	40.0	10.00

Conclusion: Good relationship between input signal & output voltage.

Transfer Function

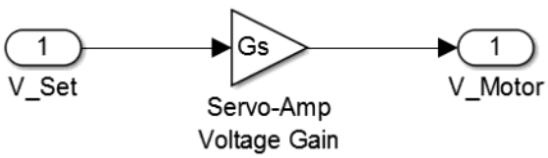
Tuesday, November 22, 2016 2:43 PM

Modelling Tachometer

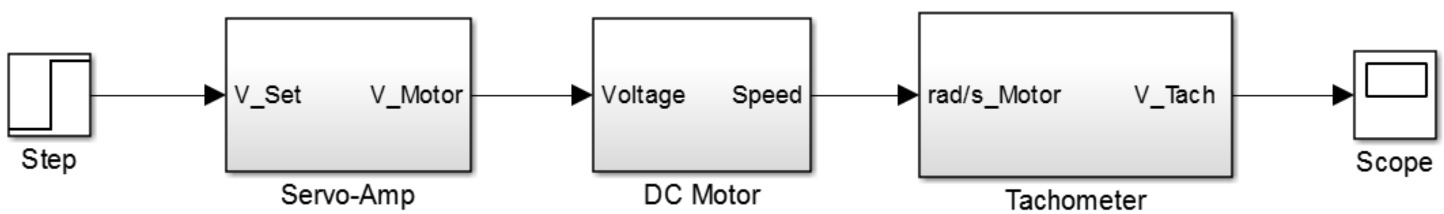
Tachometer Gain = 14 V / 1000 rpm
rad/s to RPM = 9.54929659643

Modelling Servo-Amp

Servo-Amp Gain = 1 V/V



Entire Motor Model



Updating Transfer Function

Just motor (from last week):

$$G(s) = \frac{\omega(s)}{V(s)} = \frac{\dfrac{K_t}{LJ}}{s^2 + \dfrac{(RJ + bL)}{LJ}s + \dfrac{Rb + K_e K_t}{LJ}}$$

Slide 49 - S&CS Systems of DE's Slides

MATLAB Function:
H=((Kt/(L*J))/((s^2)+((((R*J)+(b*L))/(L*J))*s)+(((R*b)+(Ke*Kt))/(L*J))))

Adding servo-amp:

Motor: $G(s) = \dfrac{\omega(s)}{V(s)} = \dfrac{\frac{K_T}{LJ}}{s^2 + \left(\frac{RJ+bL}{LJ}\right)s + \frac{Rb+K_eK_T}{LJ}}\ \dfrac{rad/s}{V}$

Servo-Amp: $H(s) = \dfrac{V_{motor}(s)}{V_{set}(s)} = 1$

Tachometer: $I(s) = \dfrac{V_{tach}(s)}{\omega(s)} = \left(\dfrac{14V}{1000\ rpm}\right)\left(9.5493\ \dfrac{rpm}{rad/s}\right) = 0.13369\ V/rad/s$

(Servo-Amp) (Motor) (Tachometer) $= H(s)G(s)I(s)$

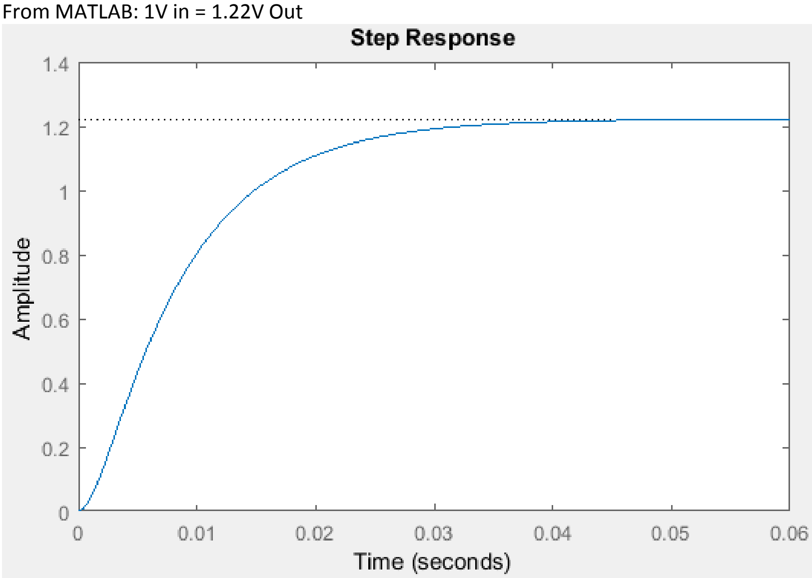
$\boxed{T.F. = (1)\,G(s)\,(0.13369)}$

MATLAB Function:
H= Gs * (Gt * Grr) * ((Kt/(L*J))/((s^2)+((((R*J)+(b*L))/(L*J))*s)+(((R*b)+(Ke*Kt))/(L*J))))

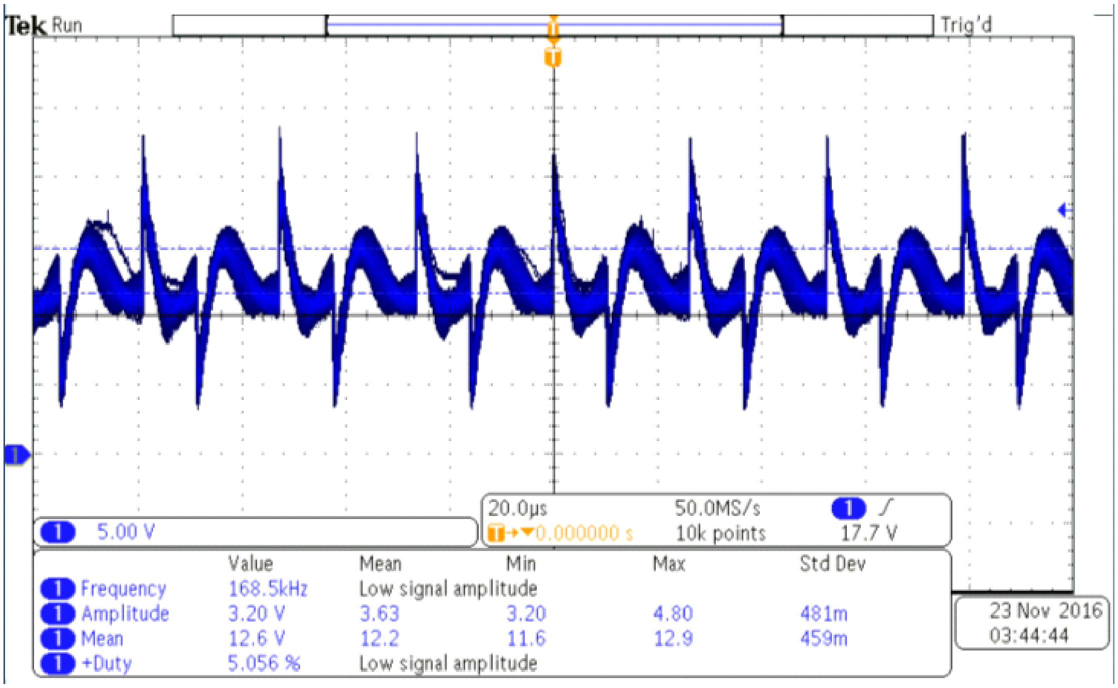
```
H =

         1.014e05
  ------------------------
  s^2 + 768.3 s + 8.288e04
```

Step-Response of H(s):

From MATLAB: 1V in = 1.22V Out



From Real-World: 10V in = 12.V Out



CONCLUSION: Our transfer function is a good estimation of the real world system.

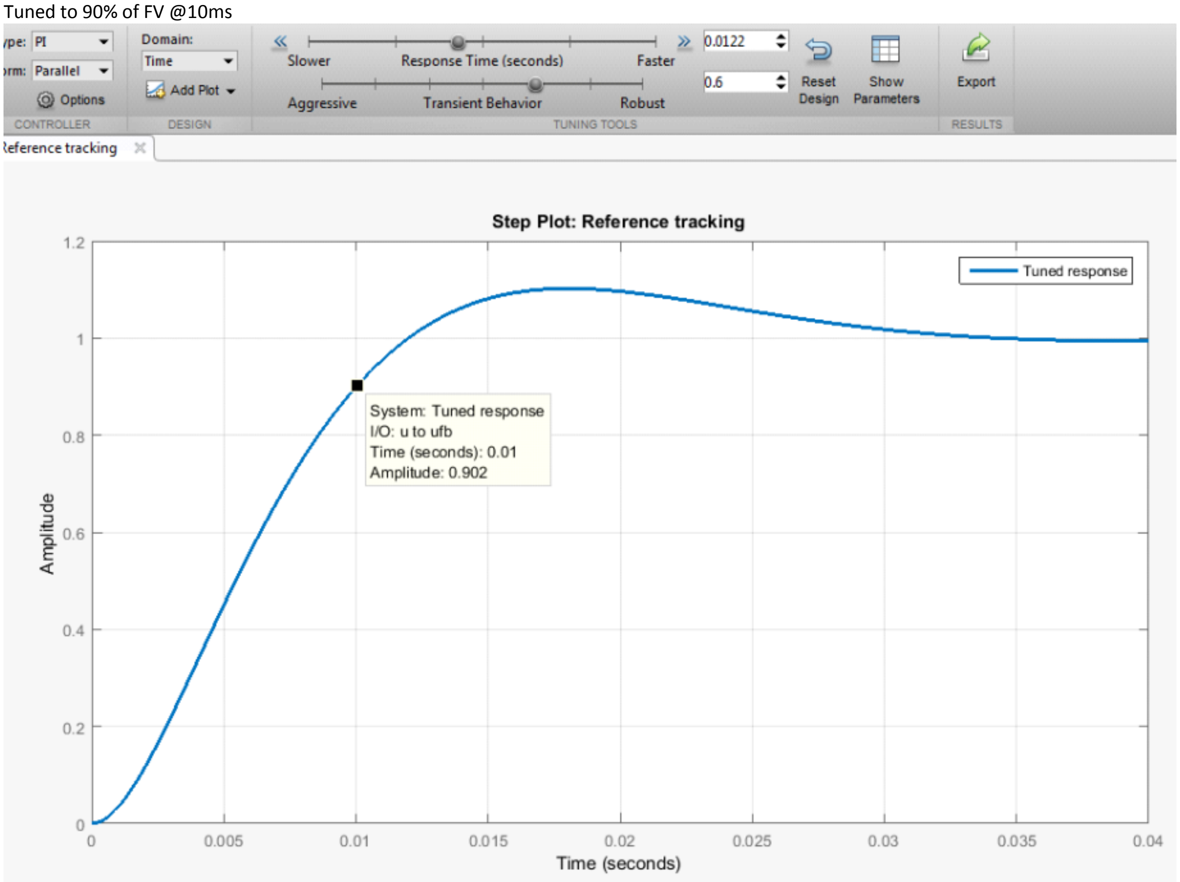
PI Controller Development

Tuesday, November 22, 2016 3:15 PM

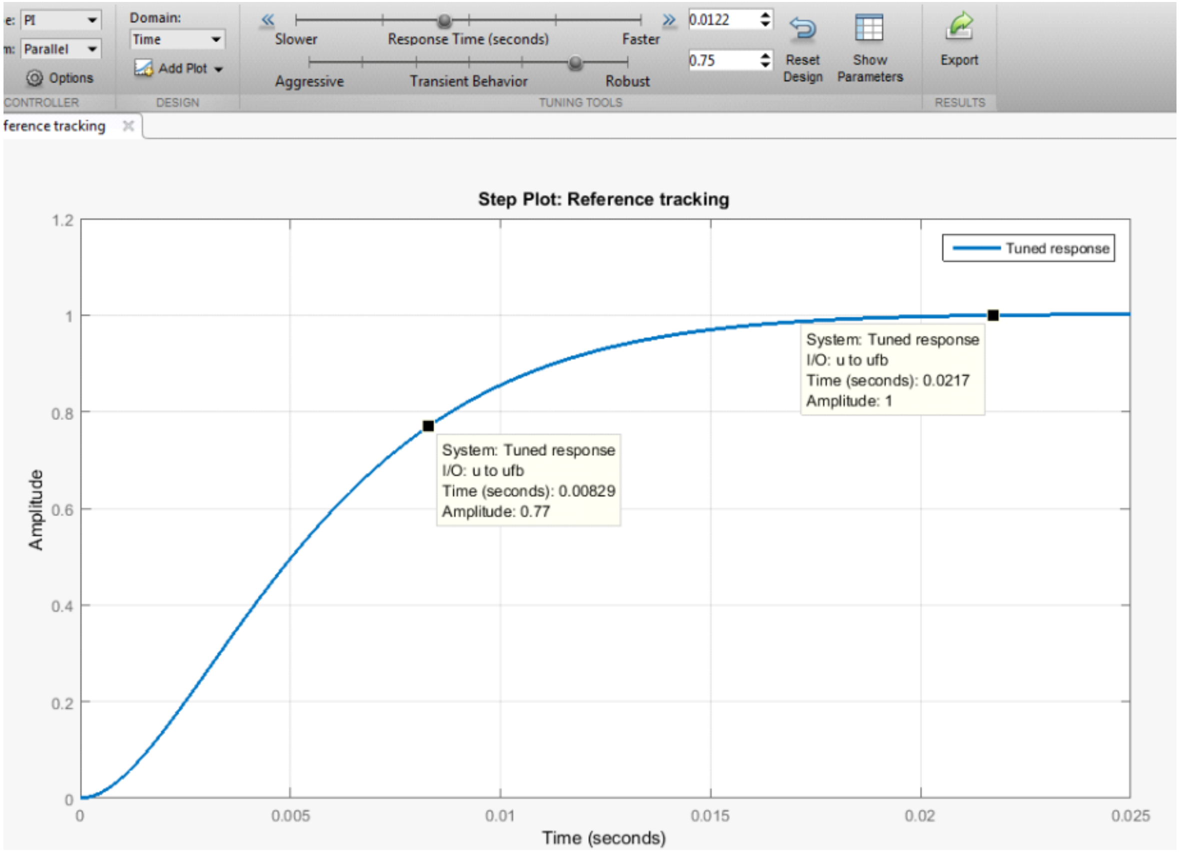
PIDTOOL Controller

Goals: 90% of final value in 10ms

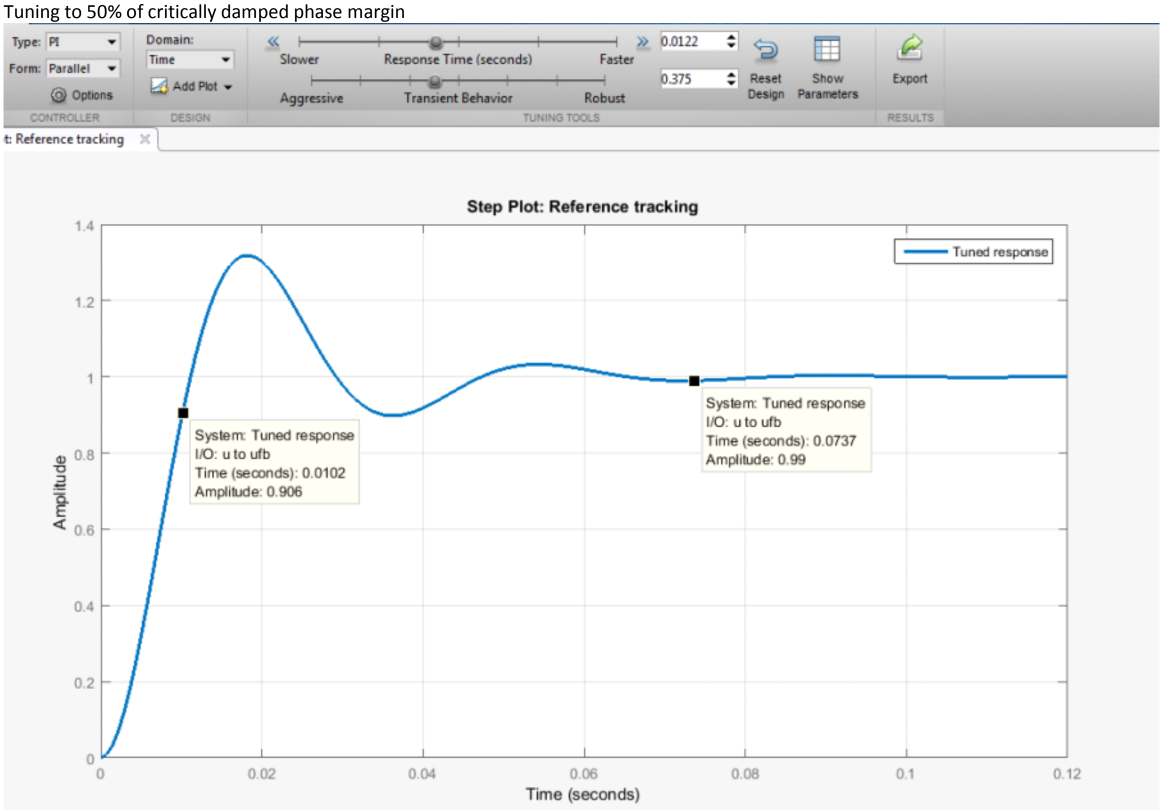
Tuned to 90% of FV @10ms



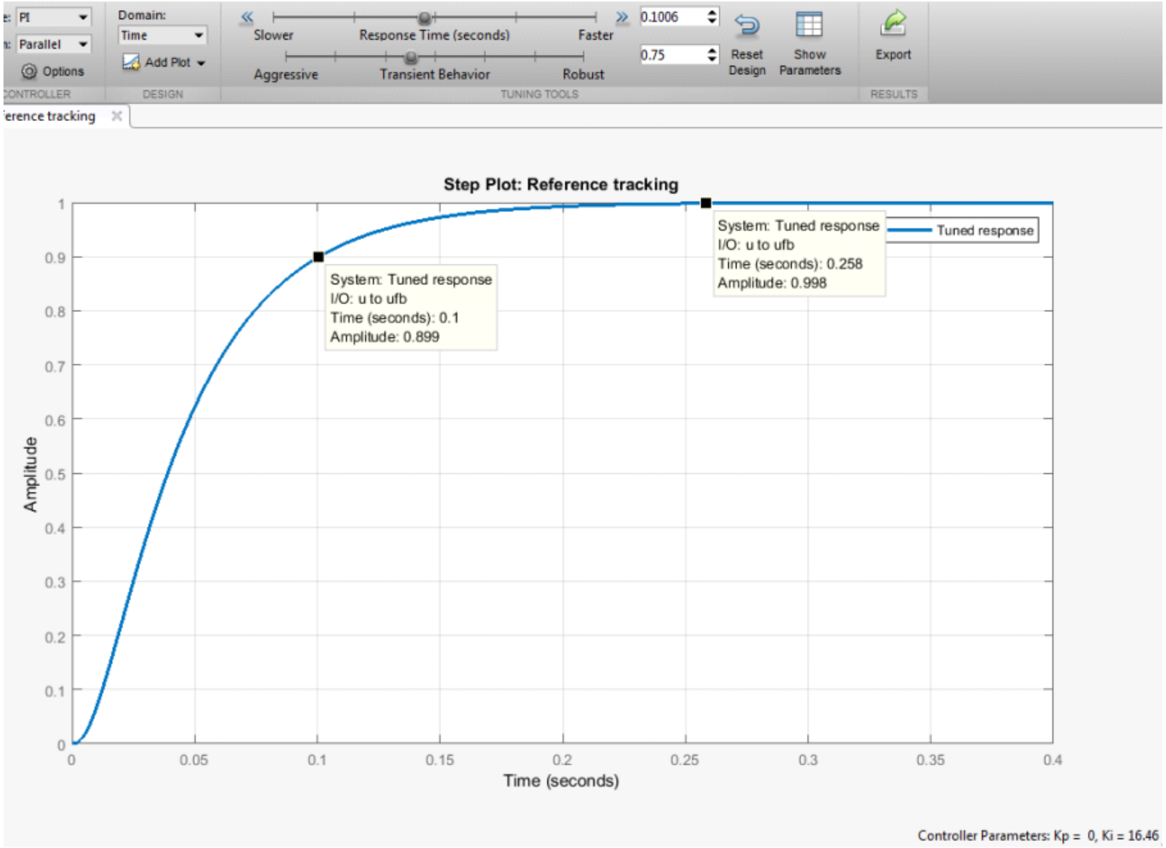
Tuning to be critically damped



Tuning to 50% of critically damped phase margin



Tuning to be close to critically damped with 10ms rise time to 90% F.V.

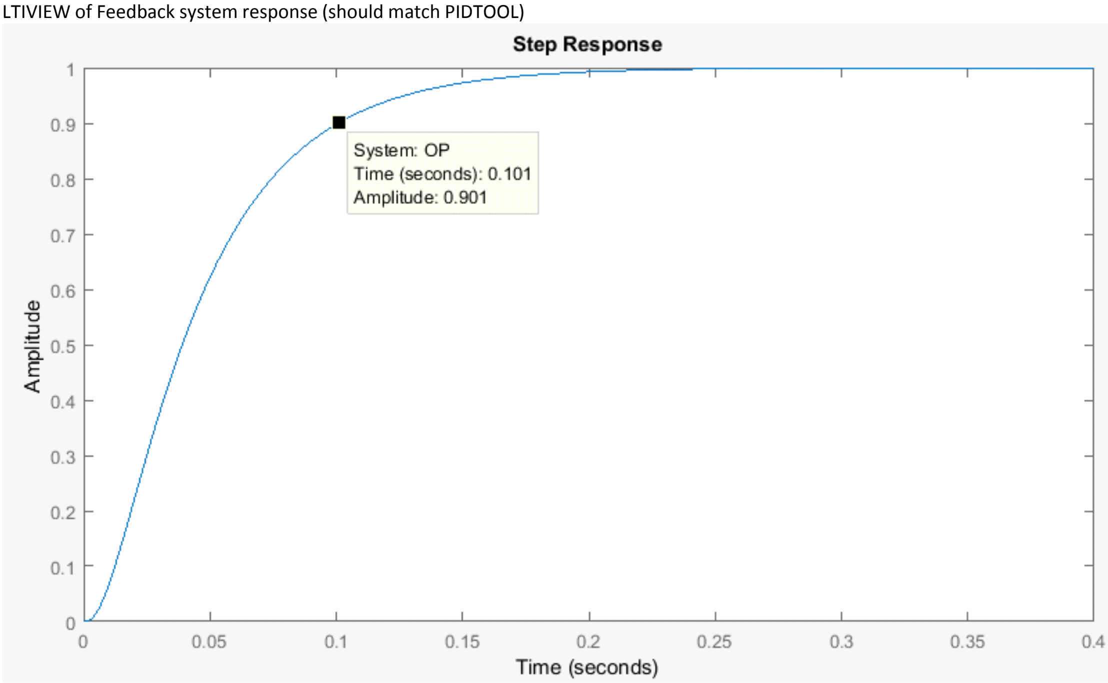


Adding PIDTOOL Controller to Model

Transfer function of feedback system

```
>> OP = (C*H) / (1 + C*H)

OP =

                1.668e06 s^3 + 1.282e09 s^2 + 1.383e11 s
  ---------------------------------------------------------------------
  s^6 + 1537 s^5 + 7.56e05 s^4 + 1.29e08 s^3 + 8.15e09 s^2 + 1.383e11 s

Continuous-time transfer function.
```

LTIVIEW of Feedback system response (should match PIDTOOL)



OUR PI CONTROLLER IS JUST AN I CONTROLLER ACTUALLY

```
>> PI = zpk(C)

PI =

  16.456
  ------
    s

Continuous-time zero/pole/gain model.
```

Op-Amp Controller Design

Motor: $G(s) = \dfrac{\omega(s)}{V(s)} = \dfrac{\frac{K_T}{LJ}}{s^2 + \left(\frac{RJ+bL}{LJ}\right)s + \frac{Rb+K_eK_T}{LJ}}$ $\dfrac{\text{rad/s}}{V}$

Servo-Amp: $H(s) = \dfrac{V_{motor}(s)}{V_{set}(s)} = 1$

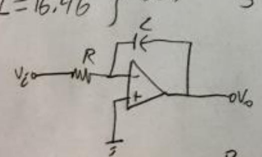
Tachometer: $I(s) = \dfrac{V_{tach}(s)}{\omega(s)} = \left(\frac{14\,V}{1000}\,\text{rpm}\right)\left(9.5493\ \text{rpm}/\text{rad/s}\right) = 0.13369\ V/\text{rad/s}$

$(\text{Servo-Amp})(\text{Motor})(\text{Tachometer}) = H(s)G(s)I(s)$

$\boxed{T.F. = (1)\,G(s)\,(0.13369)}$

PI Controller Design:

$\left.\begin{array}{l} P = 0 \\ I = 16.46 \end{array}\right\} G(s) = \dfrac{16.456}{s} = \dfrac{-Z_f}{Z_i} = \dfrac{-\frac{1}{sC}}{R}$



$G(s) = \dfrac{16.456}{\cancel{s}} = \dfrac{-1}{\cancel{s}RC}$

$-\dfrac{1}{RC} = 16.456$

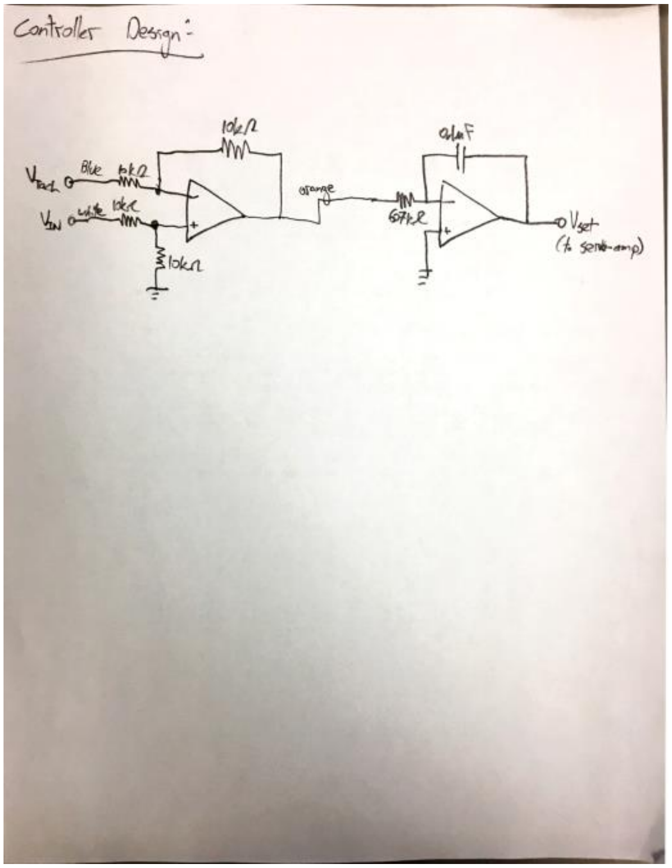
$R = \dfrac{-1}{16.456\,(C)}$

Try $C = 0.1\,\mu F$

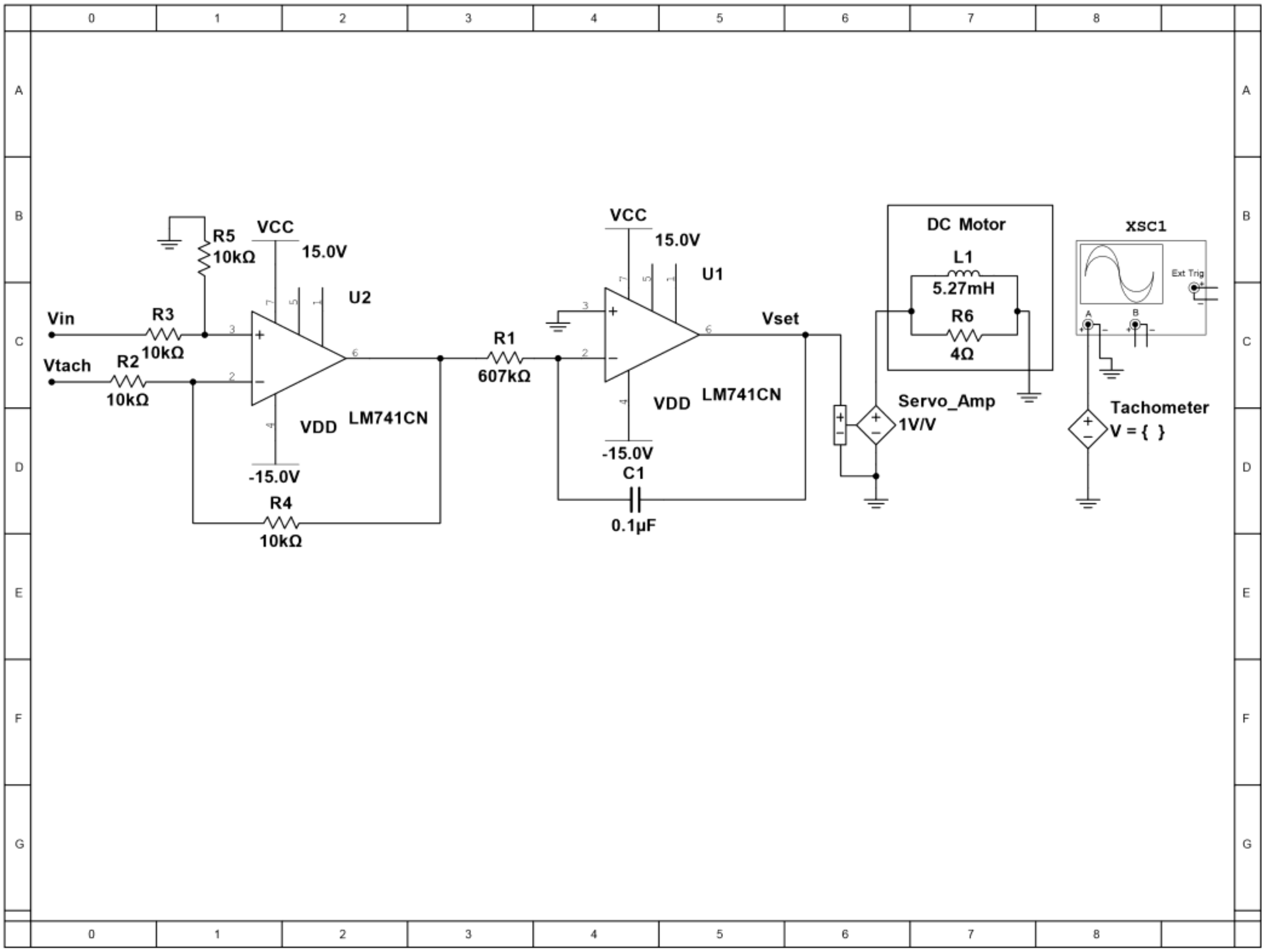
$R = 607.681\ k\Omega$

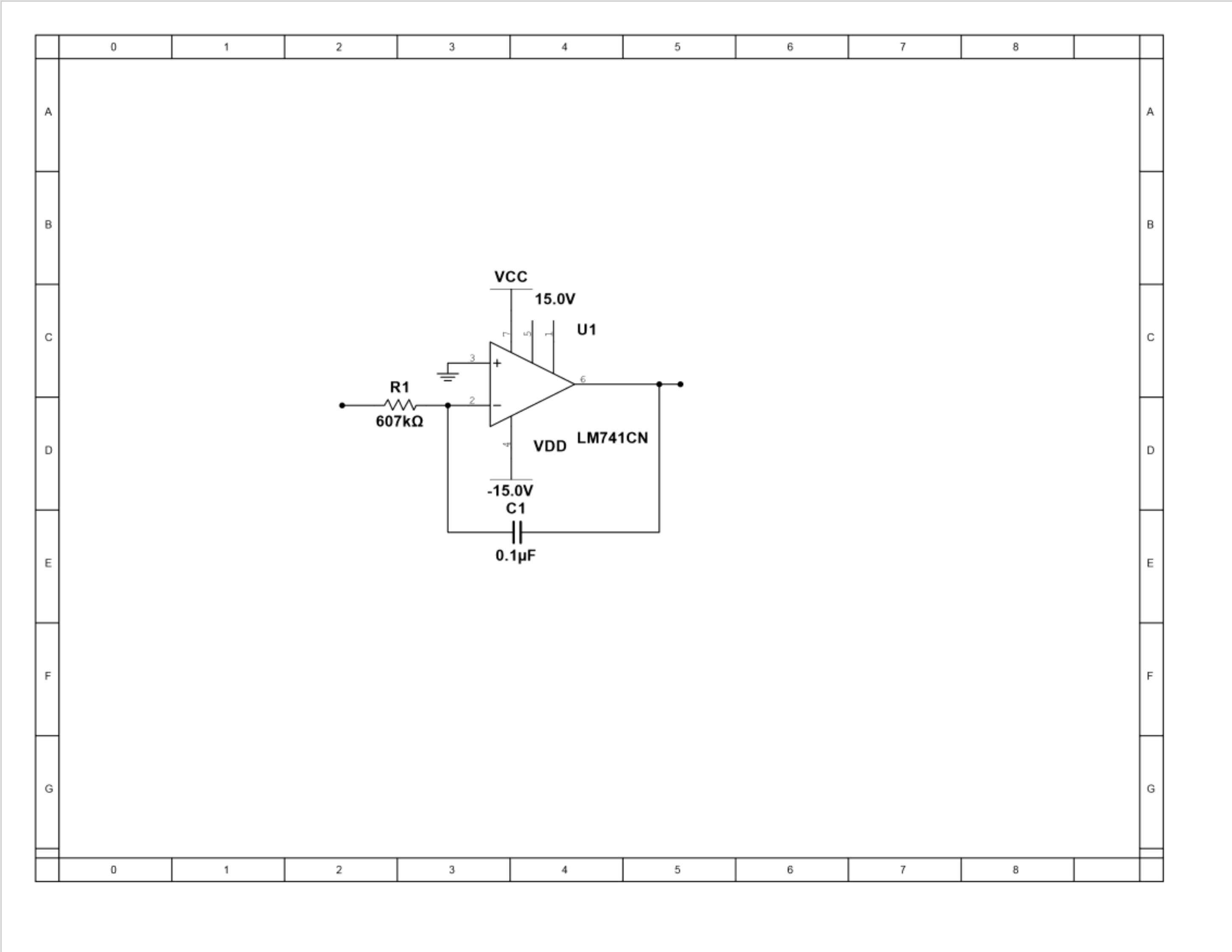
$390k\Omega + 220k\Omega = 0,606M\Omega$

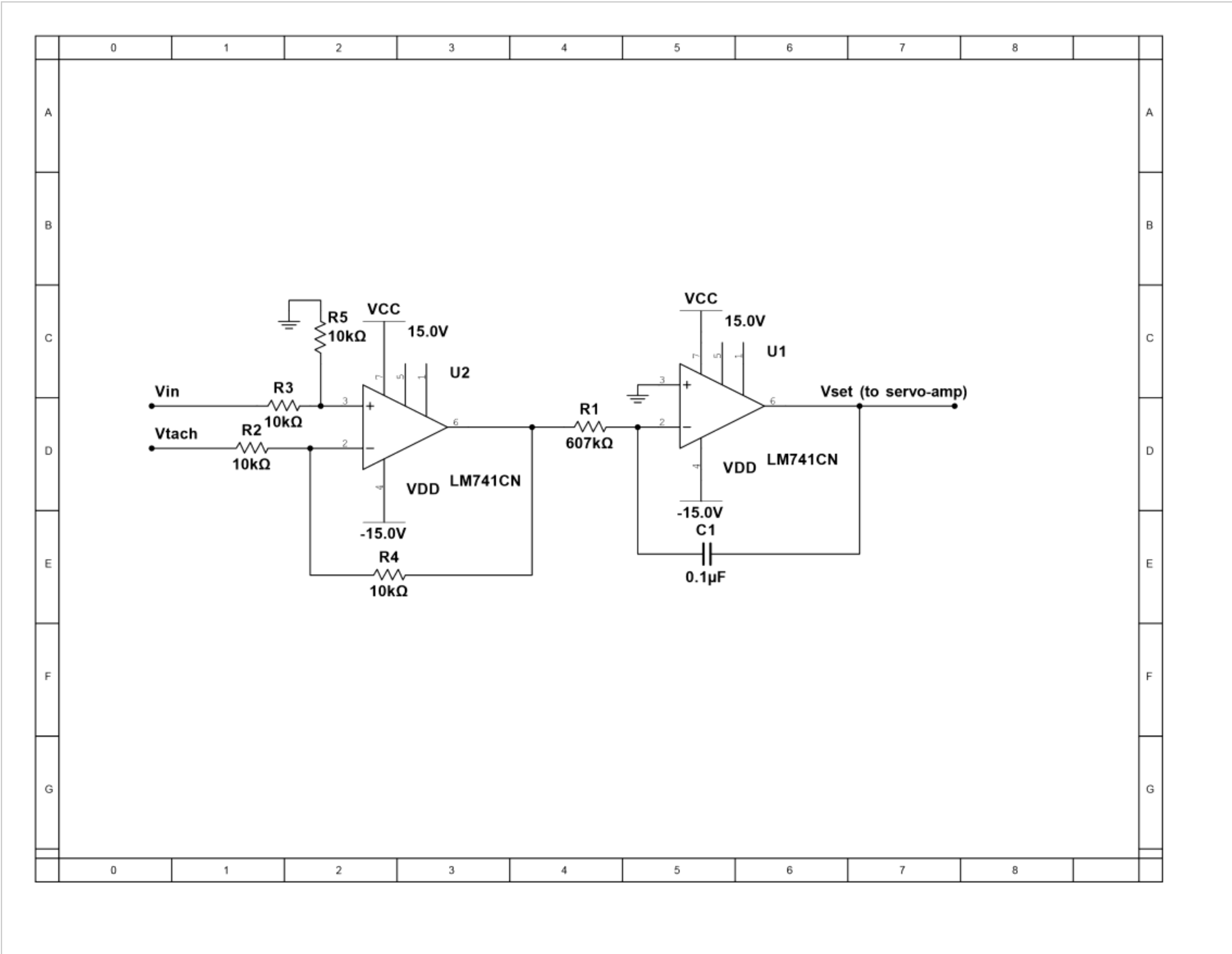
$560k\Omega + 47k\Omega = 0.610M\Omega$



$$G(s) = \frac{16.456}{s} = -\frac{Z_f}{Z_i} = -\frac{\frac{1}{sC}}{R}$$


VCC
15.0V
R5
10kΩ
U2
Vin
R3
10kΩ
Vtach
R2
10kΩ
VDD
LM741CN
-15.0V
R4
10kΩ
R1
607kΩ
VCC
15.0V
U1
VDD
LM741CN
-15.0V
C1
0.1µF
Vset
DC Motor
L1
5.27mH
R6
4Ω
Servo_Amp
1V/V
XSC1
Ext Trig
Tachometer
V = { }


VCC
15.0V
U1
R1
607kΩ
VDD
LM741CN
-15.0V
C1
0.1µF



0
1
2
3
4
5
6
7
8
A
B
C
D
E
F
G
R5
10kΩ
VCC
15.0V
U2
Vin
R3
10kΩ
Vtach
R2
10kΩ
VDD
LM741CN
-15.0V
R4
10kΩ
R1
607kΩ
VCC
15.0V
U1
Vset (to servo-amp)
VDD
LM741CN
-15.0V
C1
0.1µF

Controller Performance Measurements

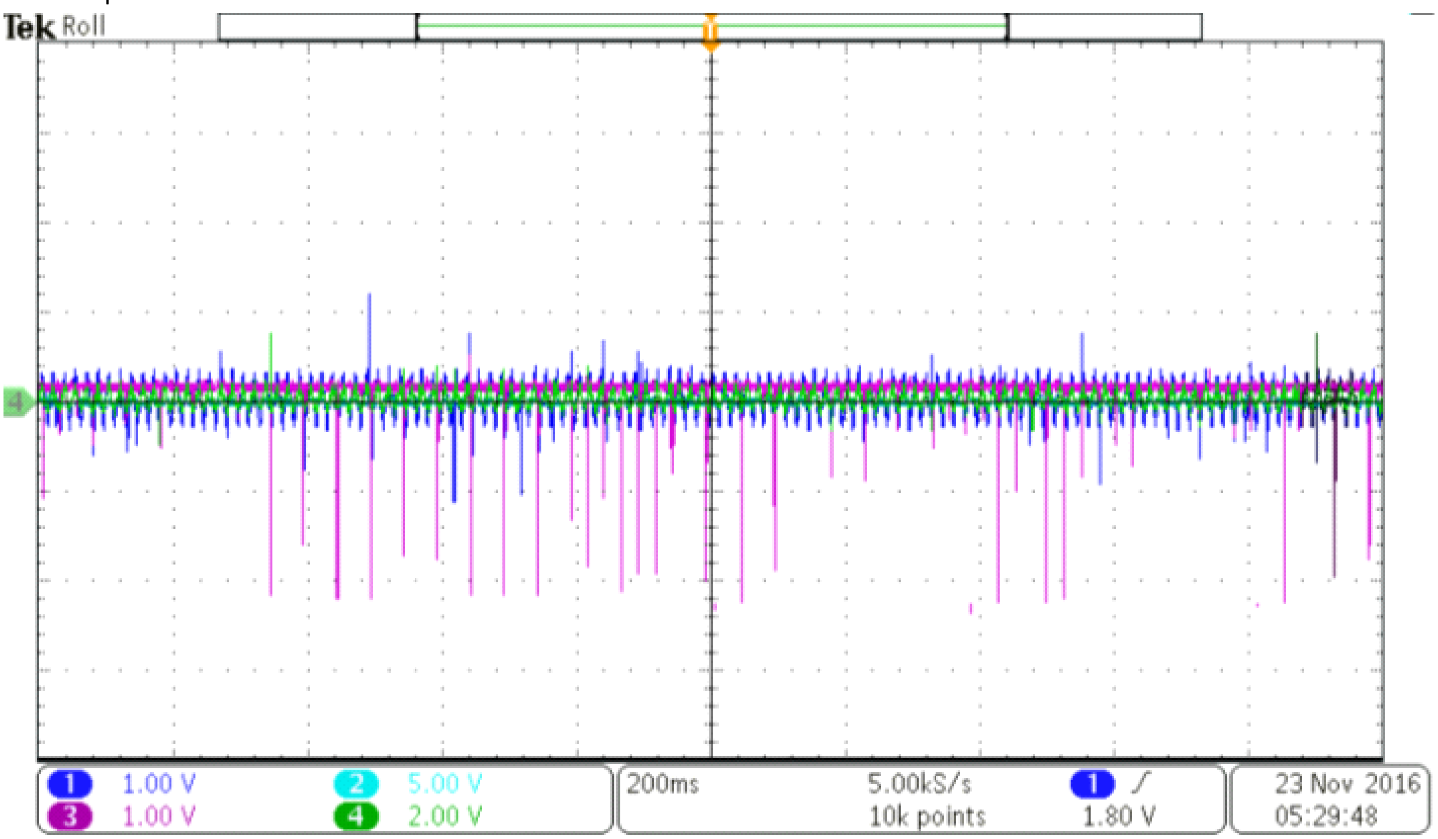
Tuesday, November 22, 2016 4:57 PM

Measurement Setup

Channel 1 - Yellow - Setpoint
Channel 2 - Blue - Error
Channel 3 - Pink - Controller Output
Channel 4 - Green - Tachometer Output

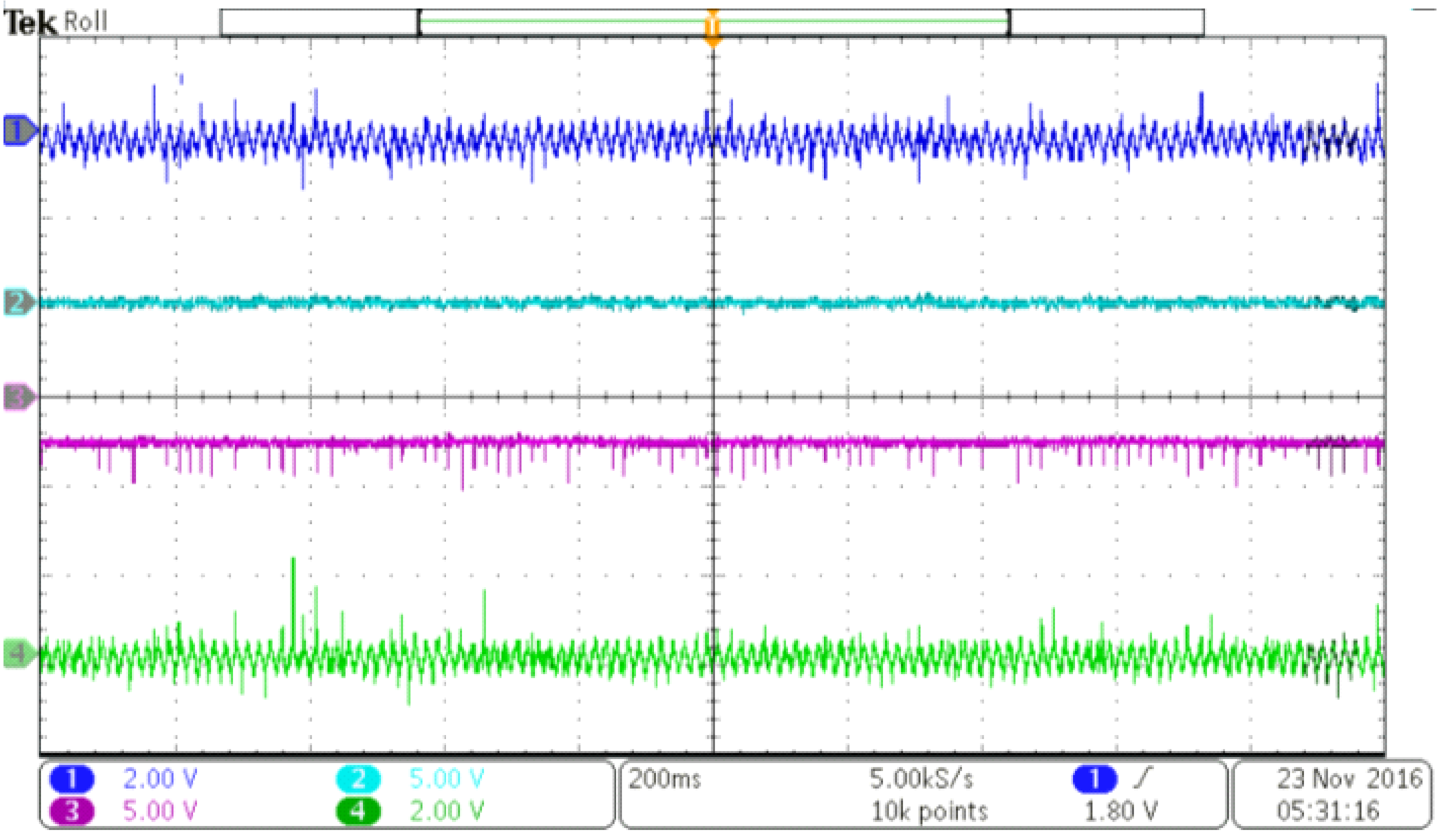
Zero Input

Zero output! Zero error! Zero Tach!

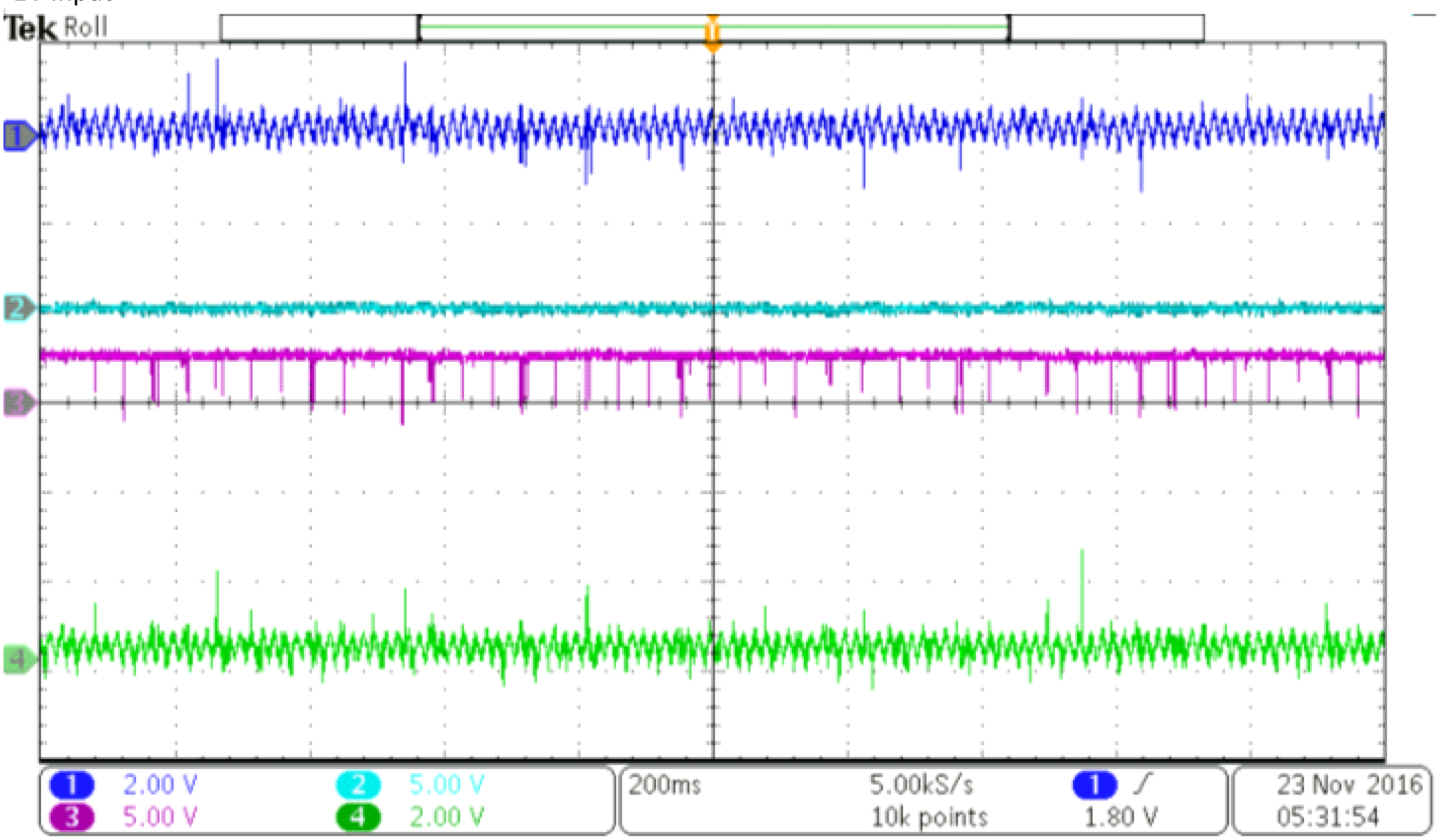


+/- 1V Input

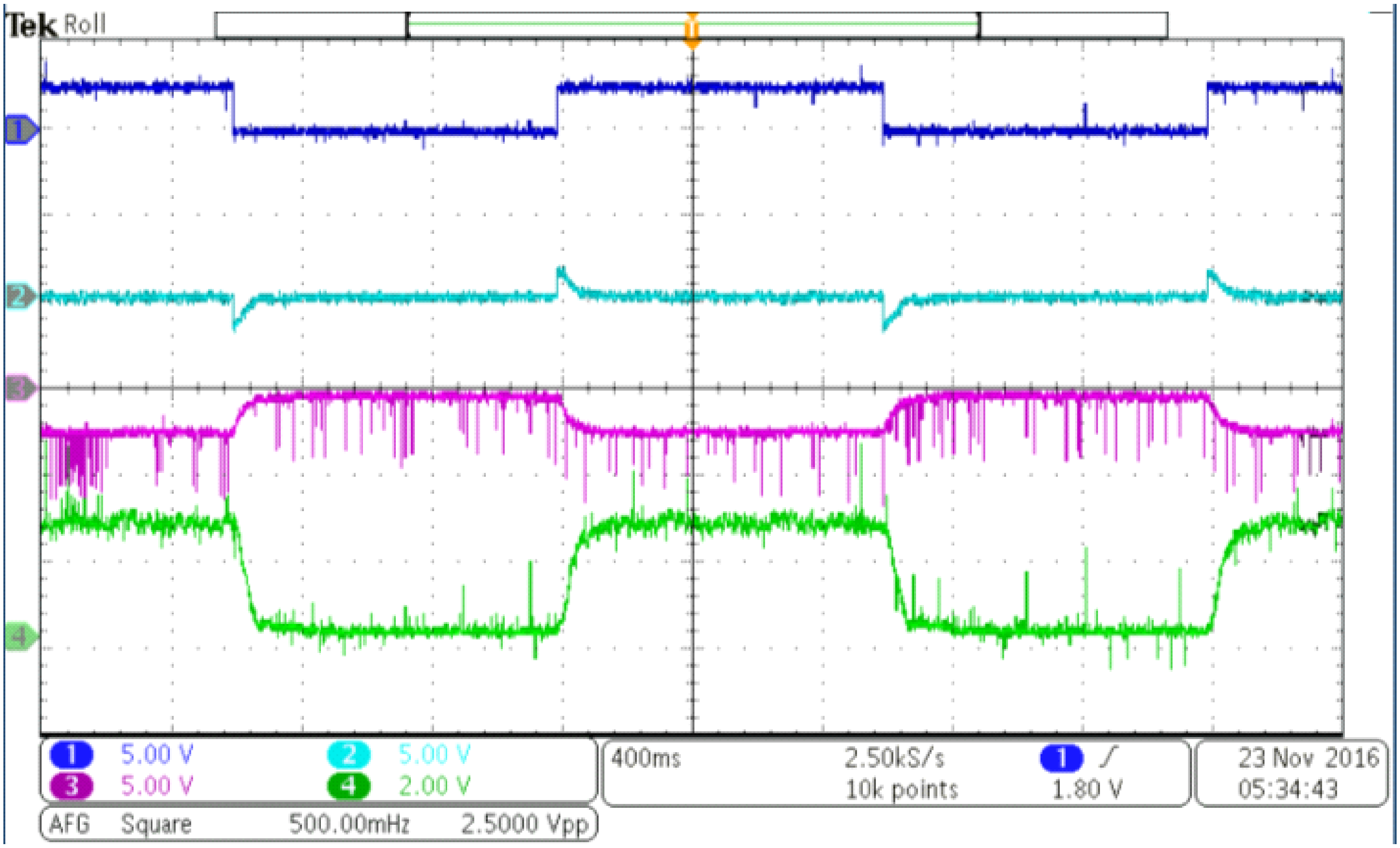
+1V Input - Motor Spins at Constant Speed



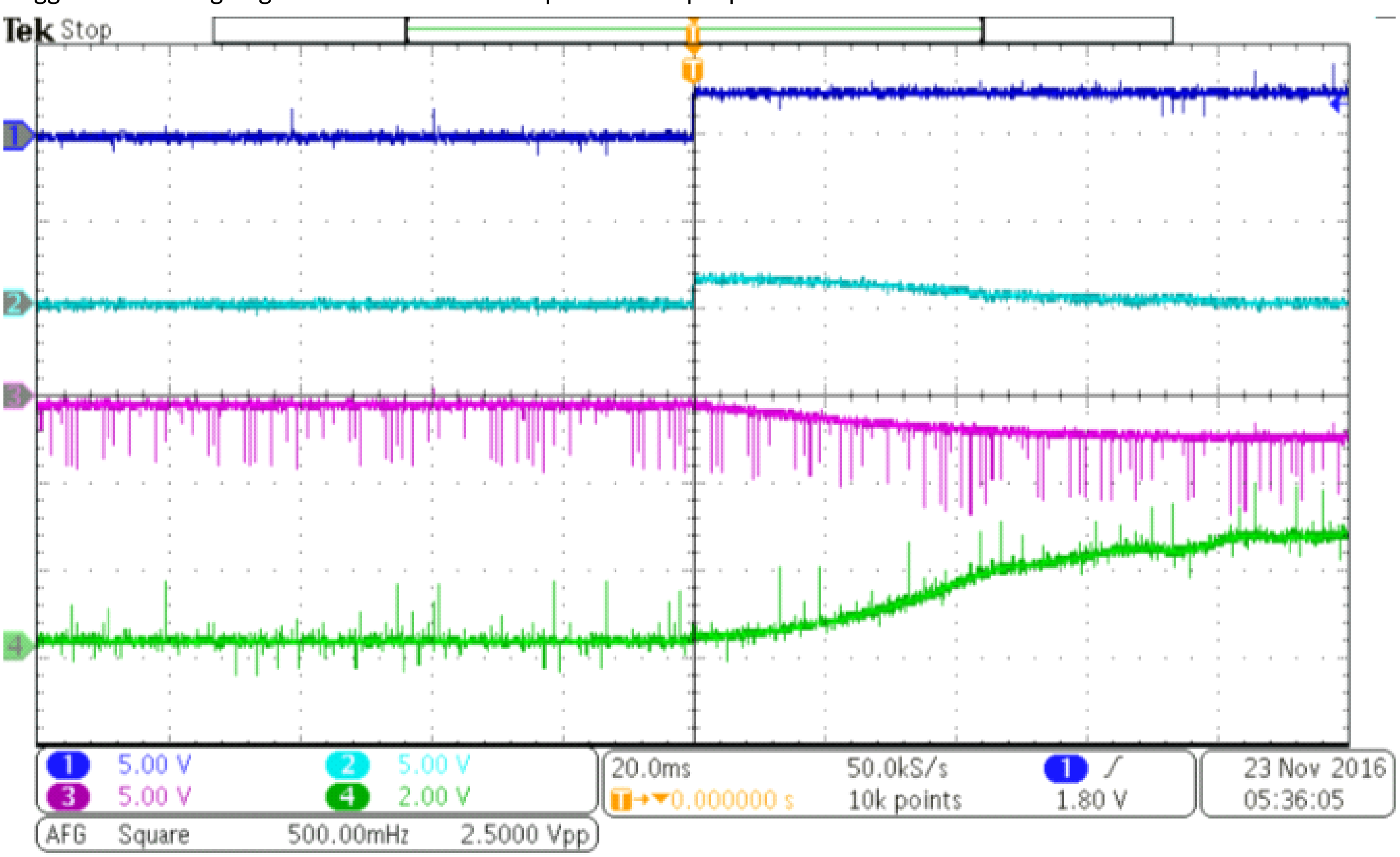
-1V Input



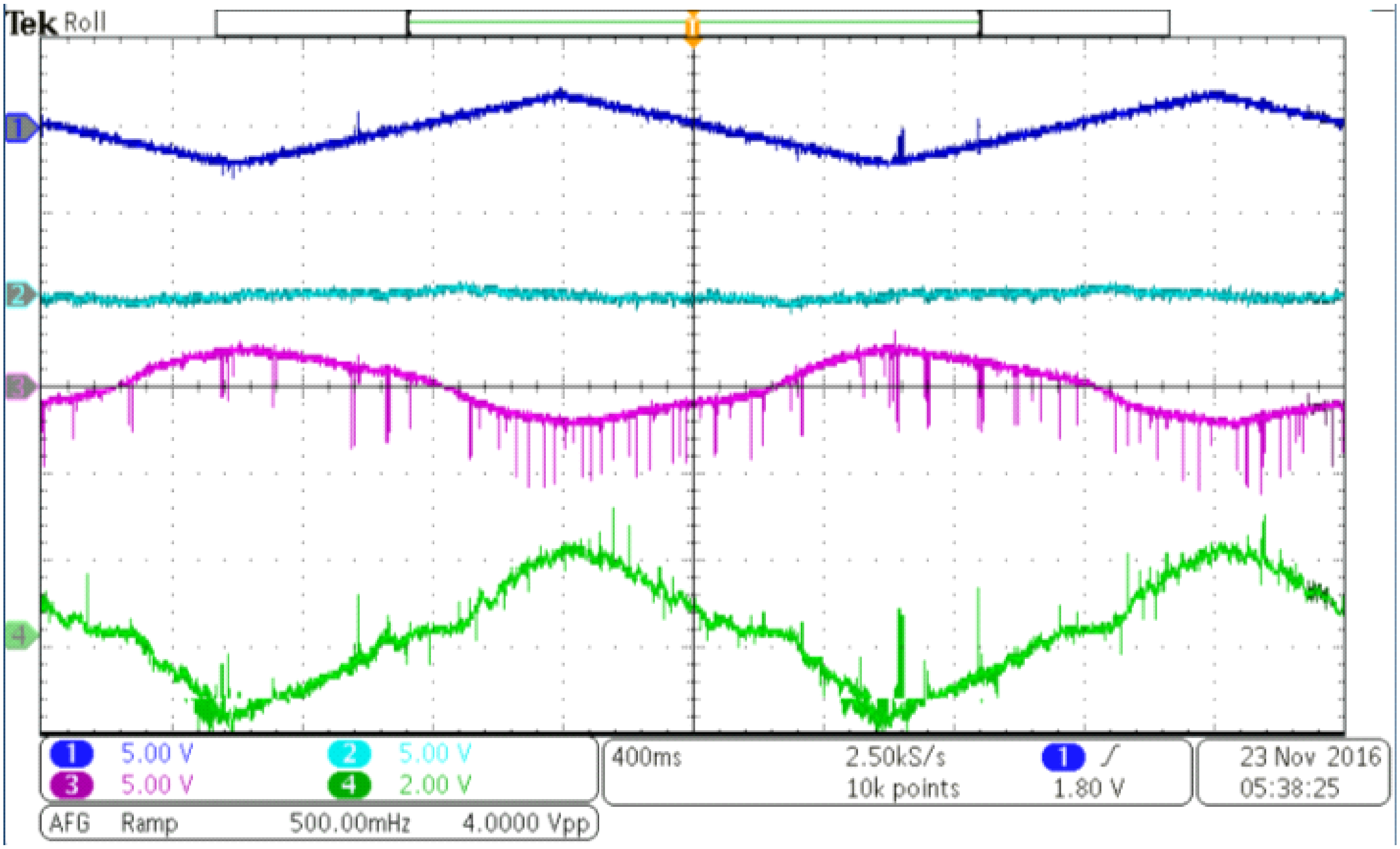
2.5V Square Wave - 0.5Hz



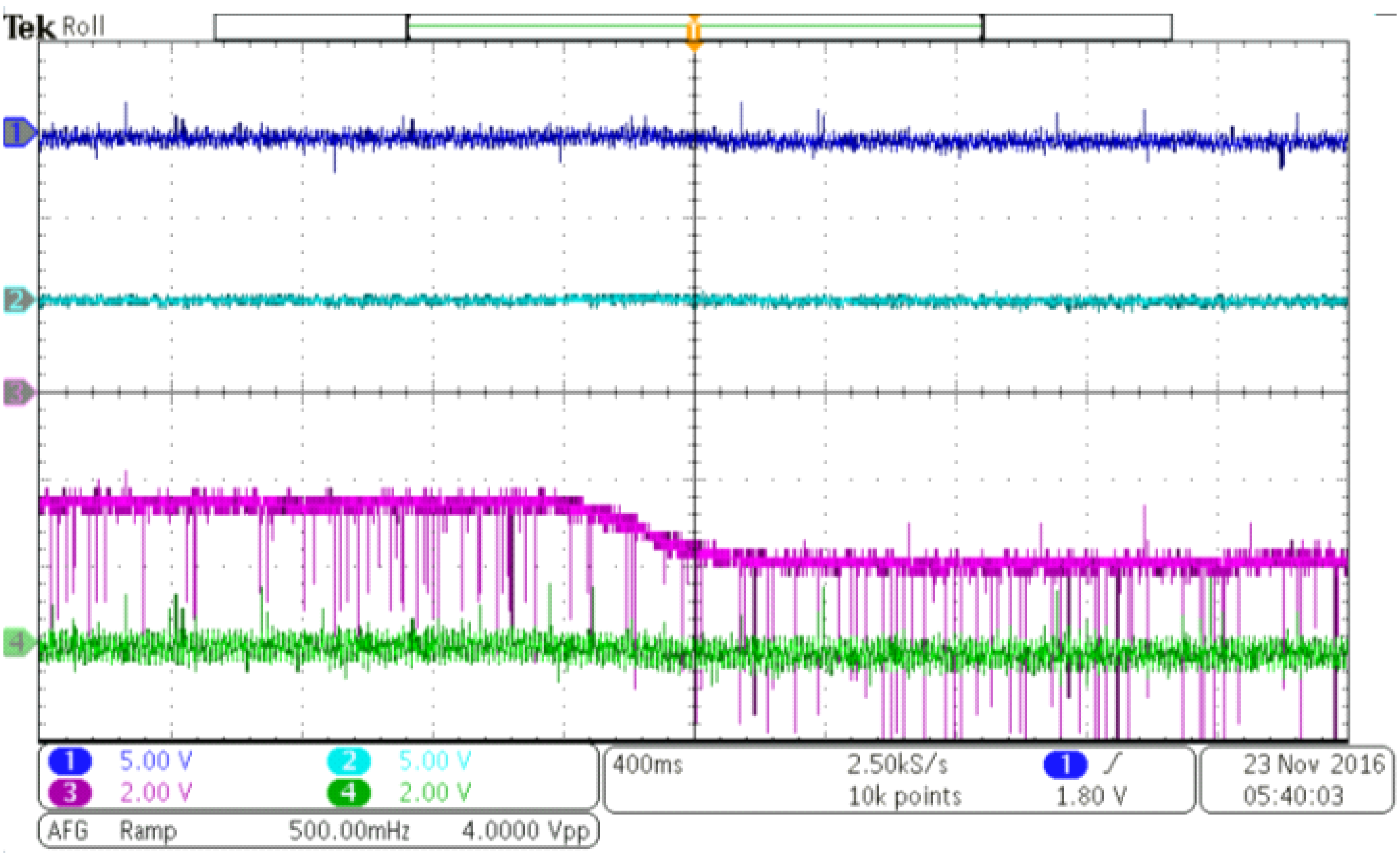
Triggered on Rising Edge to show controller response to step input



+/- 2V Triangle Wave - 0.5Hz



1V Input - Loaded by hand



Week 13 Overview

Tuesday, November 29, 2016 12:11 PM

We won't be doing testing with the elevators this week.

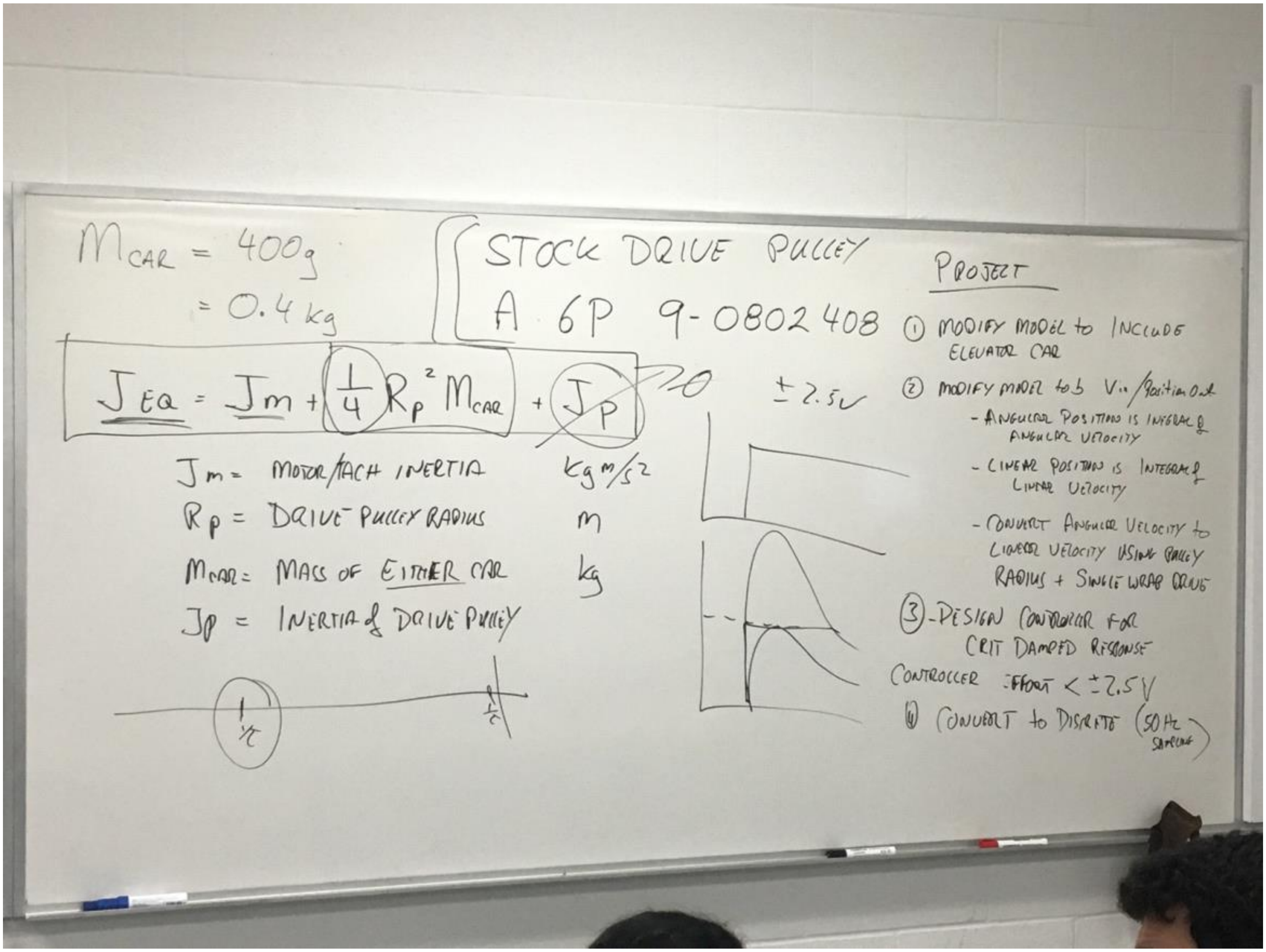
- [x] Updated Motor Model
    - [x] Add inertia of elevator cars
    - [x] Output = linear position (instead of rotational speed)
- [x] New PI Position Controller
    - [x] Continuous transfer function
    - [x] Discrete transfer function for compensator

Bill briefing

Wednesday, November 30, 2016 1:34 PM

Key Points

- Mass of elevator cars: 400g each
- Drive pulley = A 6P 9-0802408
- Controller effort limited to +/- 2.5V



Updated Motor Model

Wednesday, November 30, 2016 1:40 PM

Effective Inertia with Elevator Cars

Inertia of Motor + Tachometer: (from Weeks 10-12)

$J_m = 2.5021x10^{-5}$ kg*m^2

Effective Inertia of Elevator Cars:

$$J_c = \frac{1}{4} {R_p}^2 M_c$$

R_p = Pitch Radius of Drive Pulley (m) = 0.0145542m

Drive Pulley = A 6P-0802408

Pitch Diameter = 1.146" = 29.1084mm = $2.91084x10^{-2}$m

M_c = Mass of each elevator car (kg) = 0.4kg

Mass = 400g each = 0.4kg

$$J_c = \frac{1}{4}(0.0145542m^2)(0.4kg) = 1.45542x10^{-3}kgm^2$$

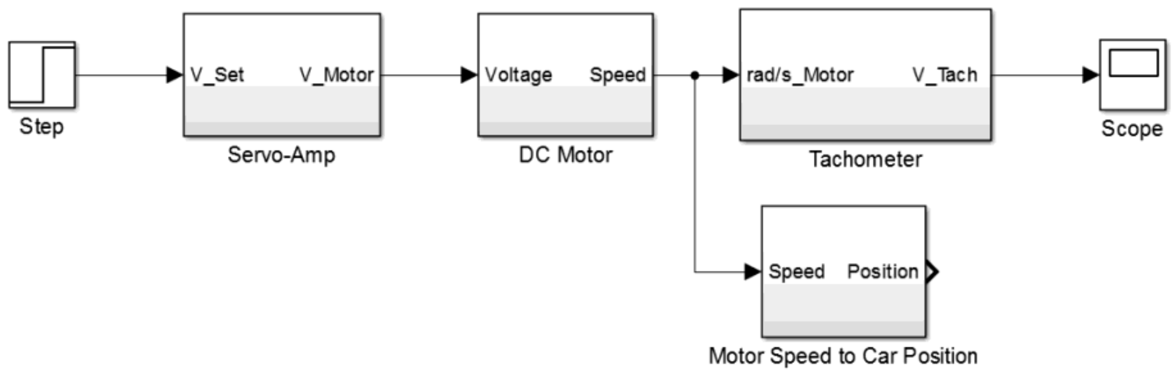
Effective Inertia of the System:

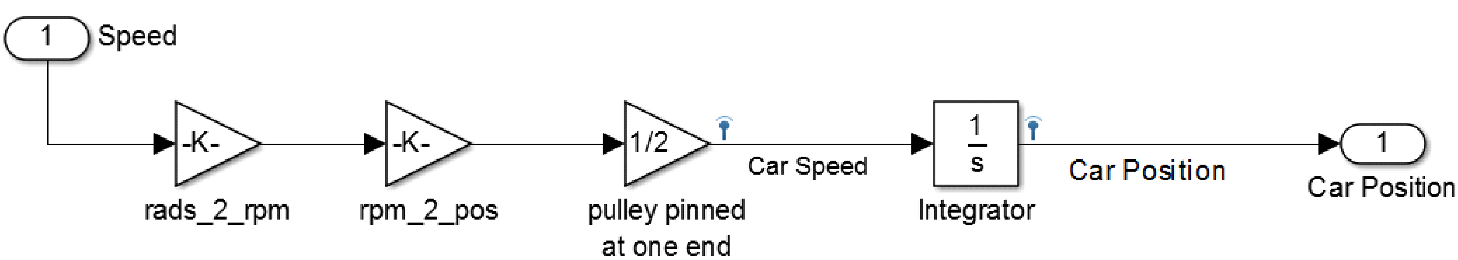
$J_{eq} = J_m + J_c = 1.480441x10^{-3}$kgm^2

Output Position Calculated as a function of Rotational Speed

Elevator Position = Rotational Speed of Motor * (Factor)

Factor = (1 Rad/s) * (1/2pi rev/rad) * (2piR m/rev) * 1/2 (factor for pinned end of elevator pulley cables)






```
rads_2_rpm = 1/(2*pi);  % Radians/second to RPM
rpm_2_pos = 2*pi*Rp;    % Angular speed in RPM to position in Metres
Rp = 29.1084/1000;      % Pitch radius of drive pulley
pos_car = speed * rads_2_rpm * rpm_2_pos * 1/2; % Car position calculation, 1/2 for pinned end.
```

Updated Transfer Function

Thursday, December 01, 2016 2:00 PM

Just motor (from week 11):

$$G(s)=\frac{\omega(s)}{V(s)}=\frac{\dfrac{K_t}{LJ}}{s^2+\dfrac{(RJ+bL)}{LJ}s+\dfrac{Rb+K_eK_t}{LJ}}$$

Slide 49 - S&CS Systems of DE's Slides

MATLAB Function of just motor + tach:
H=(Gs*(Kt/(L*J))/((s^2)+((((R*J)+(b*L))/(L*J))*s)+(((R*b)+(Ke*Kt))/(L*J))))

Adding car position:

Motor: $G(s) = \dfrac{\omega(s)}{V(s)} = \dfrac{\text{Angular Velocity}}{\text{Voltage Input}}$

Elevator: $H(s) = \dfrac{d(s)}{V(s)} = \dfrac{\text{Distance of elevator car}}{\text{Voltage Input}}$

Speed to Distance: $\dfrac{d(s)}{\omega(s)} = \left(\dfrac{1\ rpm}{2\pi\ rad/s}\right)\left(2\pi R_p \dfrac{rpm}{m/s}\right)\left(\overset{\text{one sided pulley}}{\dfrac{1}{2}}\right)\left(\underset{\text{speed to distance.}}{\dfrac{1}{s}}\right) = \left(\dfrac{R_p}{2s}\right)\dfrac{m}{rad/s}$, R_p = drive pulley pitch radius

$$H(s) = G(s)\left(\frac{d(s)}{\omega(s)}\right) = \left(\frac{\cancel{\omega(s)}}{V(s)}\right)\left(\frac{d(s)}{\cancel{\omega(s)}}\right) = \frac{d(s)}{V(s)}$$

$$H(s) = \frac{d(s)}{V(s)} = \left[\frac{\dfrac{R_T}{LJ}}{s^2+\dfrac{RJ+bL}{LJ}s+\dfrac{Rb+K_eK_T}{LJ}}\right]\left(\frac{R_p}{2s}\right)$$

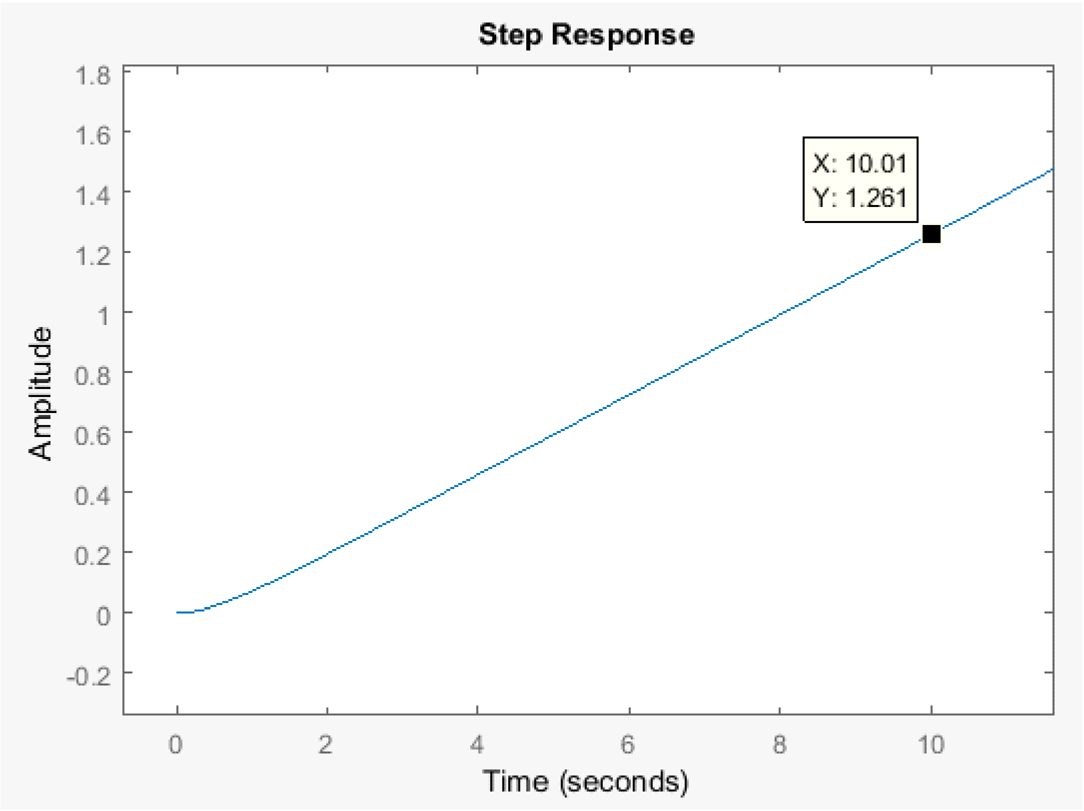
MATLAB Transfer Function:
H=(Gs*((Kt/(L*J))/((s^2)+((((R*J)+(b*L))/(L*J))*s)+(((R*b)+(Ke*Kt))/(L*J)))) * (Rp/(2*s)))

```
H =

            373.1
  -------------------------
  2 s^3 + 1518 s^2 + 2801 s

Continuous-time transfer function.
```

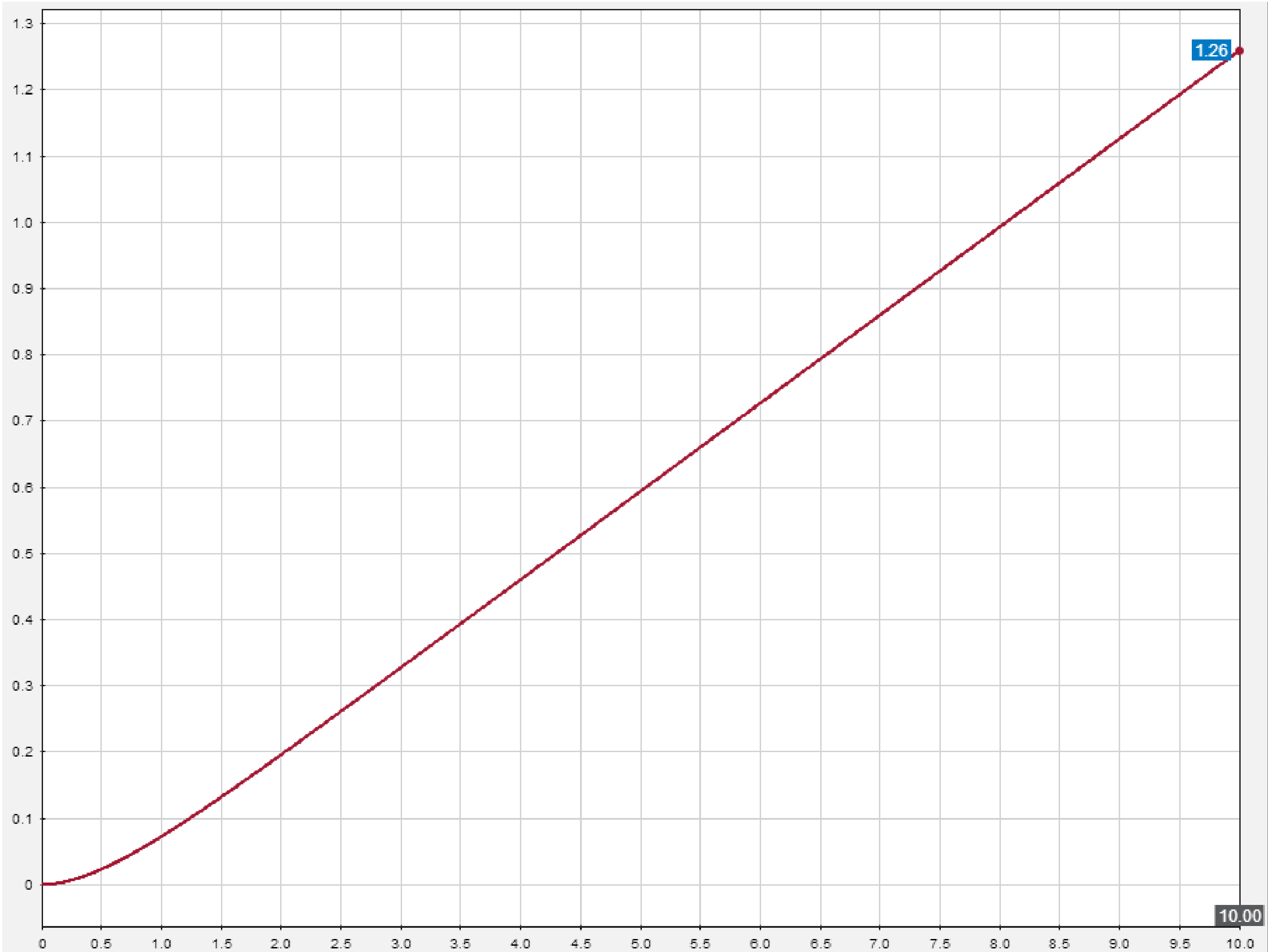
Factored transfer function:

```
>> TF = zpk(H)

TF =

         186.55
  --------------------
  s (s+757.3) (s+1.85)

Continuous-time zero/pole/gain model.
```

Step Response of H(s) Transfer Function



Step Response of Simulink Model for System



They match. This is good.

PI Controller Development

Thursday, December 01, 2016 2:51 PM

PIDTOOL Controller

Goals:

-Critically damped response.

-Settling time ~5 seconds --> 20cm/s elevator speed. Seems reasonable.

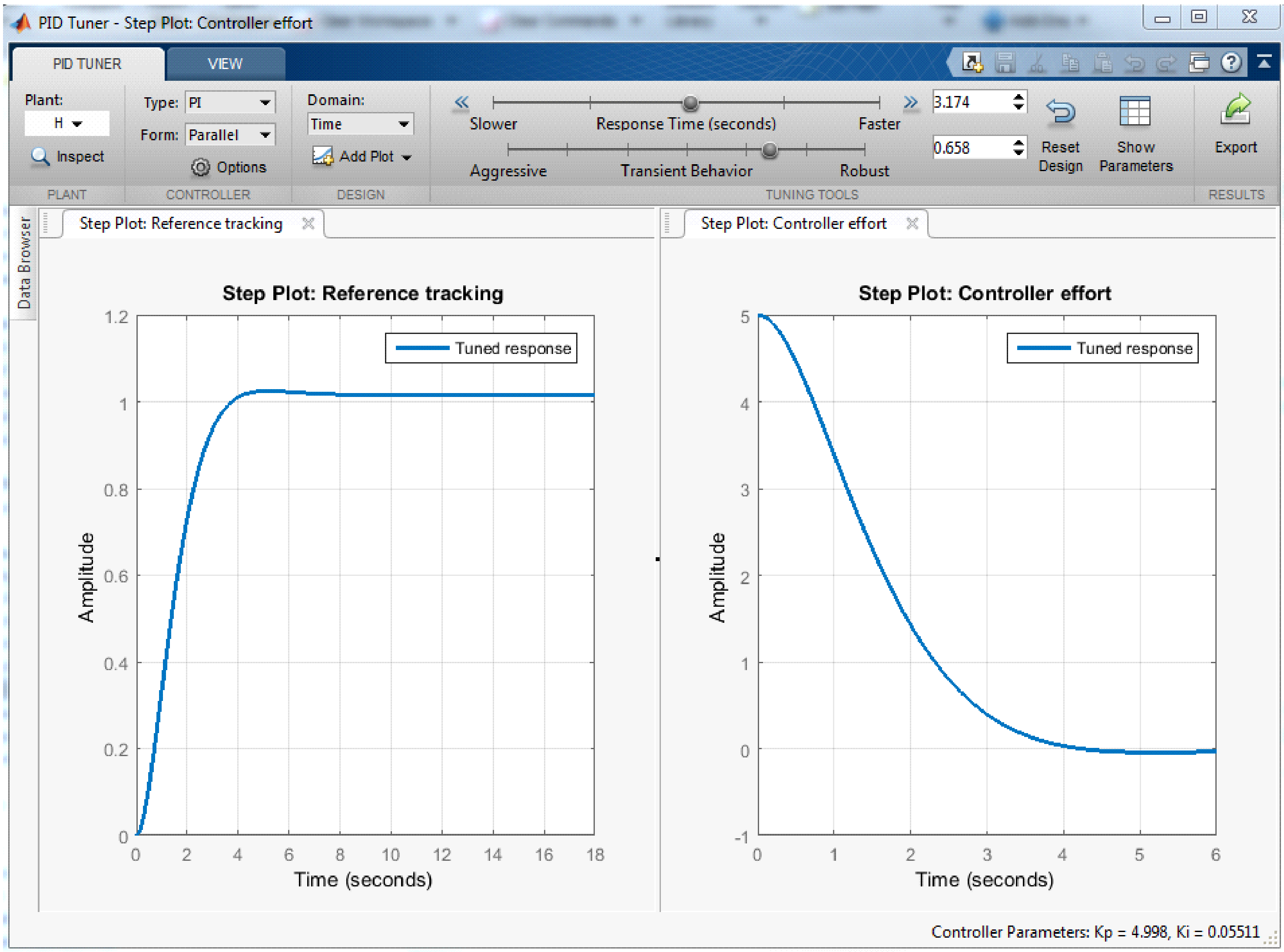
Limits: Controller effort limited to +/- 2.5V

Attempt 2: Servo-Amp Gain of 2

Settling time = 6.7 seconds -->14.925 cm/s

Can't reach critical damping with PI controller: Overshoot = 2.51%

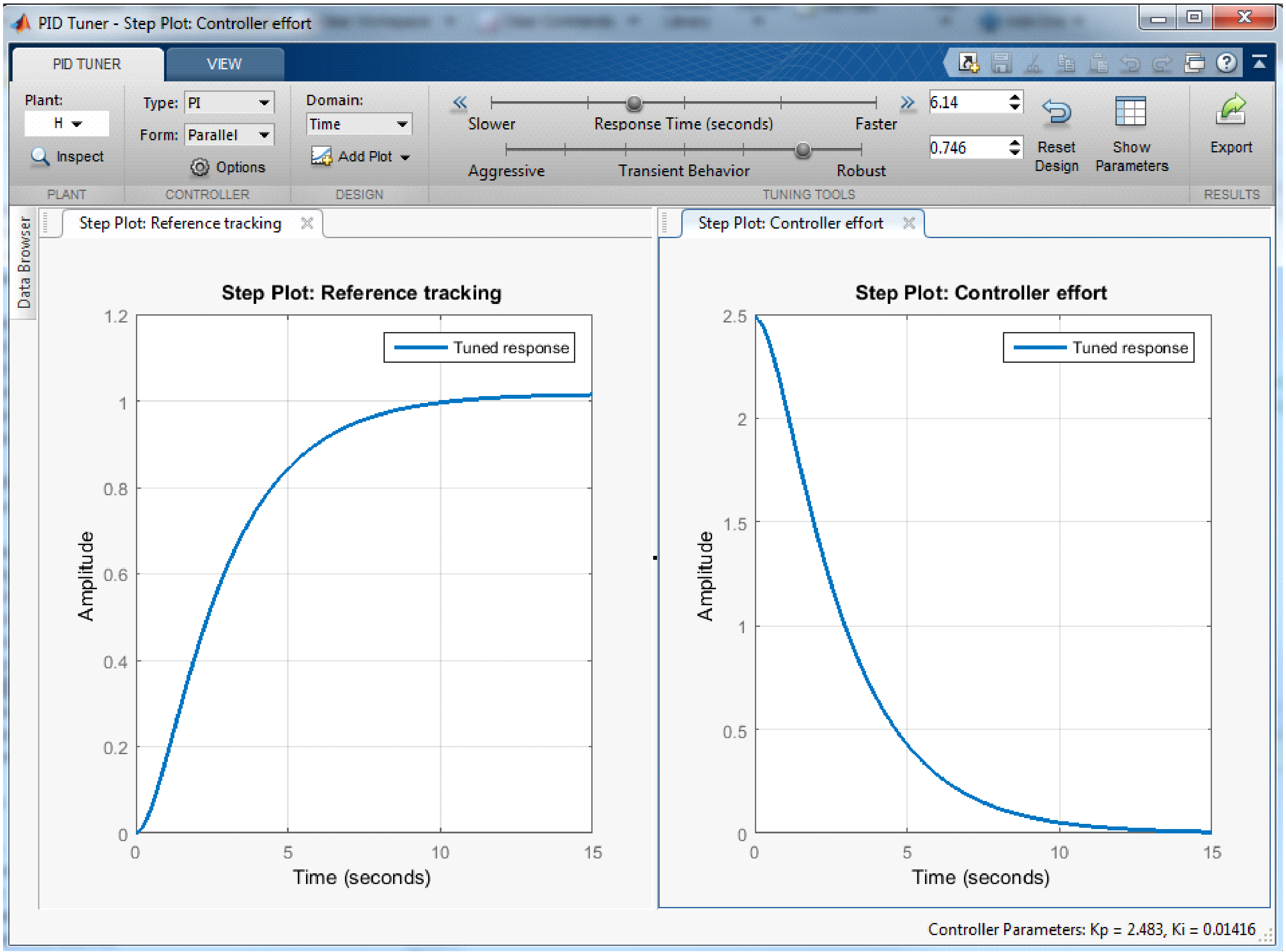
Controller effort required = +5V max



Attempt 3: Servo-Amp Gain of 2 - Tune for controller effort, not settling time

Fastest response with +/-2.5V controller effort limit: **Settling time = 8.65 seconds -->11.56cm/s**

Can't reach critical damping with PI controller: Overshoot = 1.58%

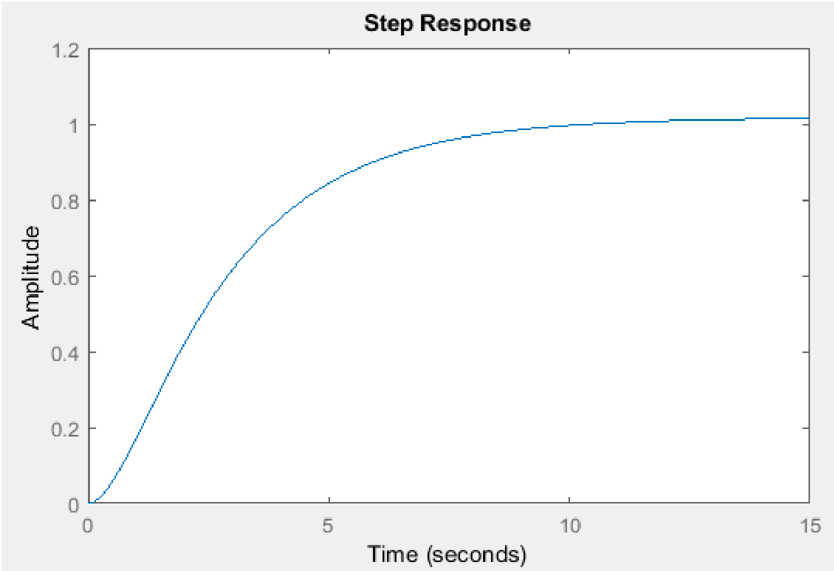


Adding PIDTOOL Controller to Model

Transfer function of feedback system

```
>> OP = (C*H) / (1+ C*H)

OP =

                    1853 s^5 + 1.407e06 s^4 + 2.603e06 s^3 + 1.48e04 s^2
  ------------------------------------------------------------------------------------------
   4 s^8 + 6073 s^7 + 2.317e06 s^6 + 8.509e06 s^5 + 9.254e06 s^4 + 2.603e06 s^3 + 1.48e04 s^2

Continuous-time transfer function.
```

LTIVIEW of Feedback system response (matches PIDTOOL)


Step Response
Amplitude
Time (seconds)
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15

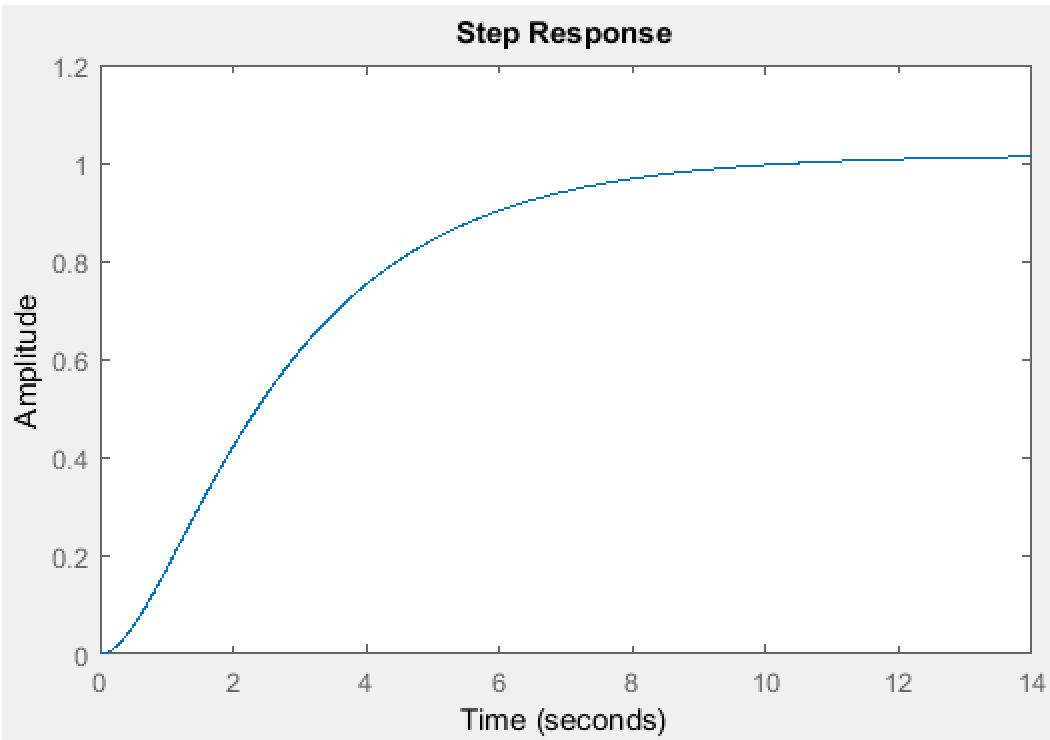
Discrete Controller Development

Thursday, December 01, 2016 3:35 PM

MATLAB Function: c2d does magic things to make a discrete TF from a continuous one.
C = controller model from PIDTOOL (P only controller)
Ts = 1/50 = 0.02 = Sampling time for 50Hz as Bill said to use that in class

```
C_D = c2d(C, Ts, 'tustin'); % Discrete TF for controller from pidtool
H_D = c2d(H, Ts, 'tustin'); % Discrete TF for elevator system model
OP_D = (C_D * H_D) / (1 + C_D*H_D); % Discrete TF for closed loop control system
```

Step-Response of Discrete Transfer Function OP_D



Matches continuous.

Discrete Transfer Function for PI Controller

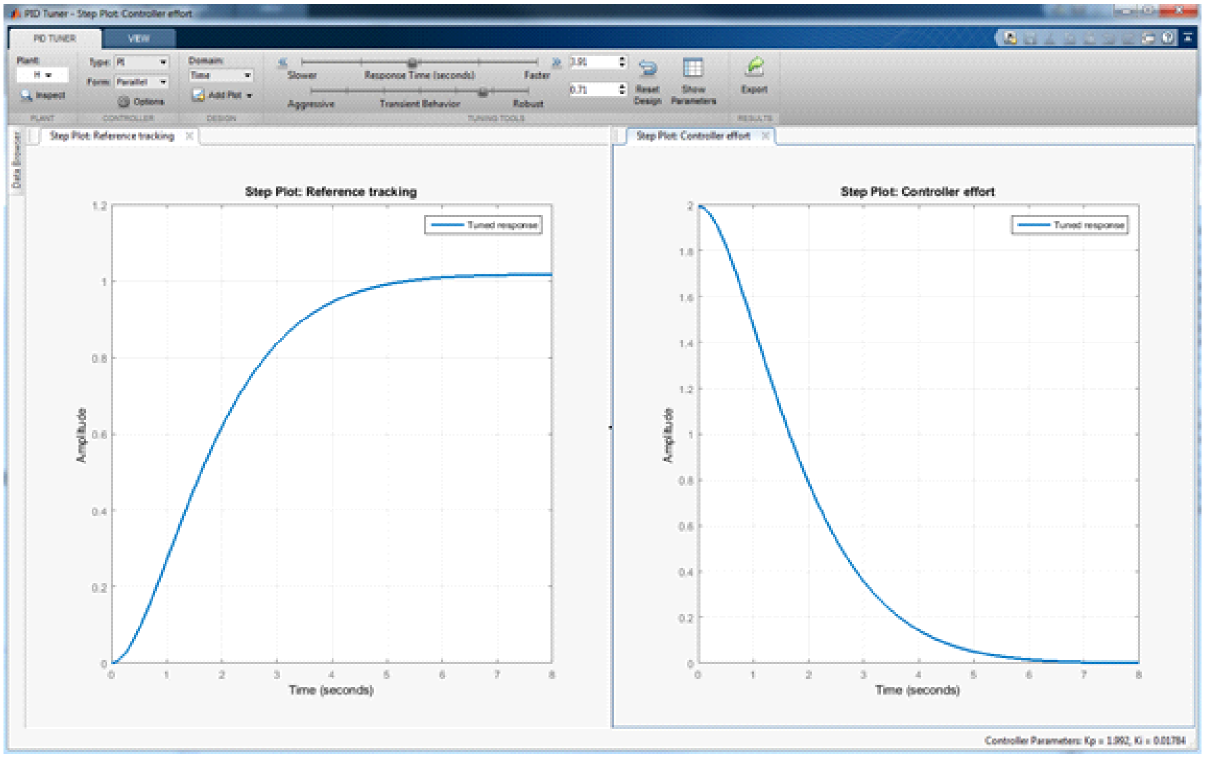
```
C_D =

             Ts*(z+1)
  Kp + Ki * --------
             2*(z-1)

  with Kp = 2.48, Ki = 0.0142, Ts = 0.02

Sample time: 0.02 seconds
Discrete-time PI controller in parallel form.
```

Week 14 Overview

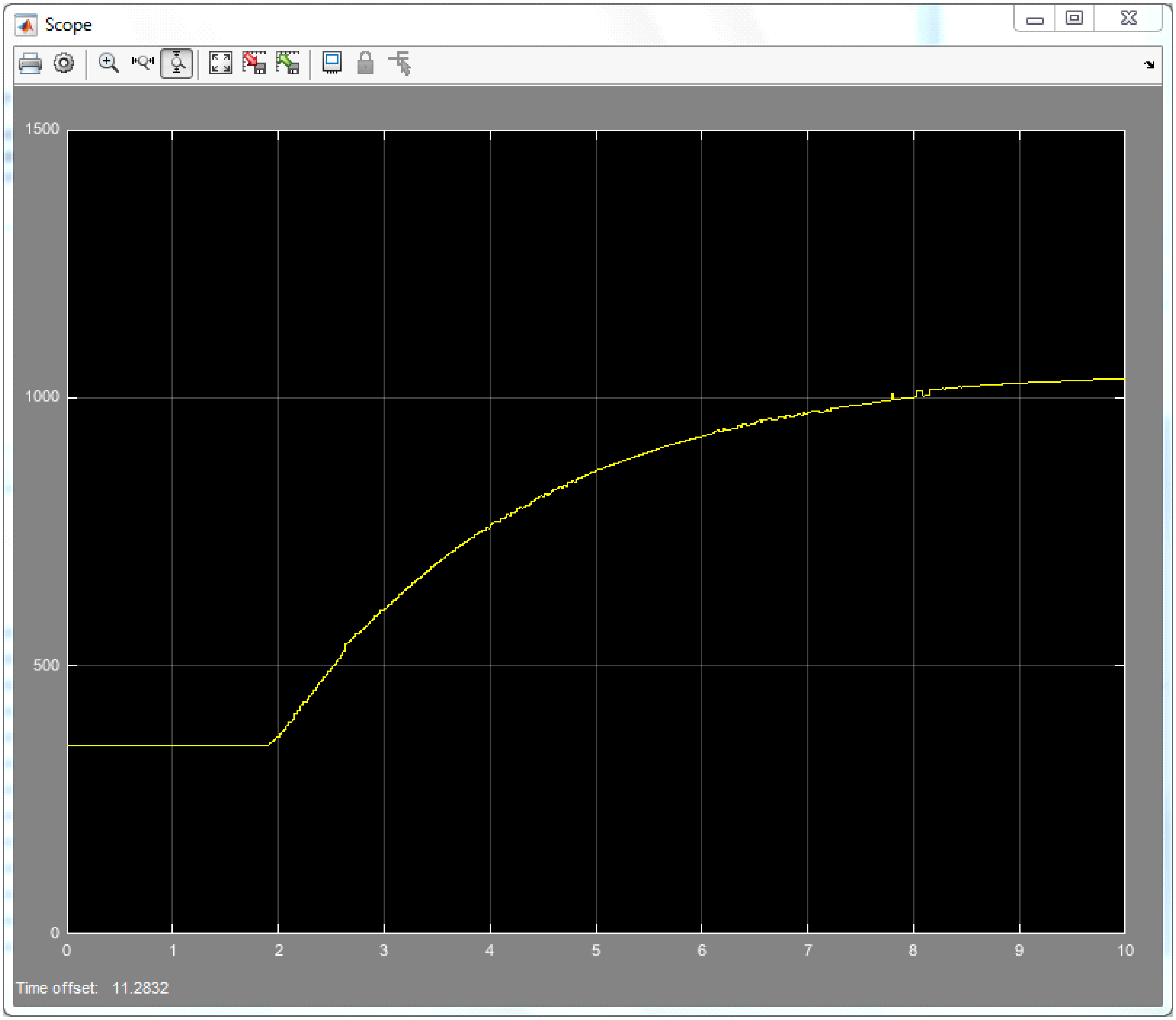
Wednesday, December 07, 2016 1:27 PM

- [x] Test controller on elevator
- [x] Project report

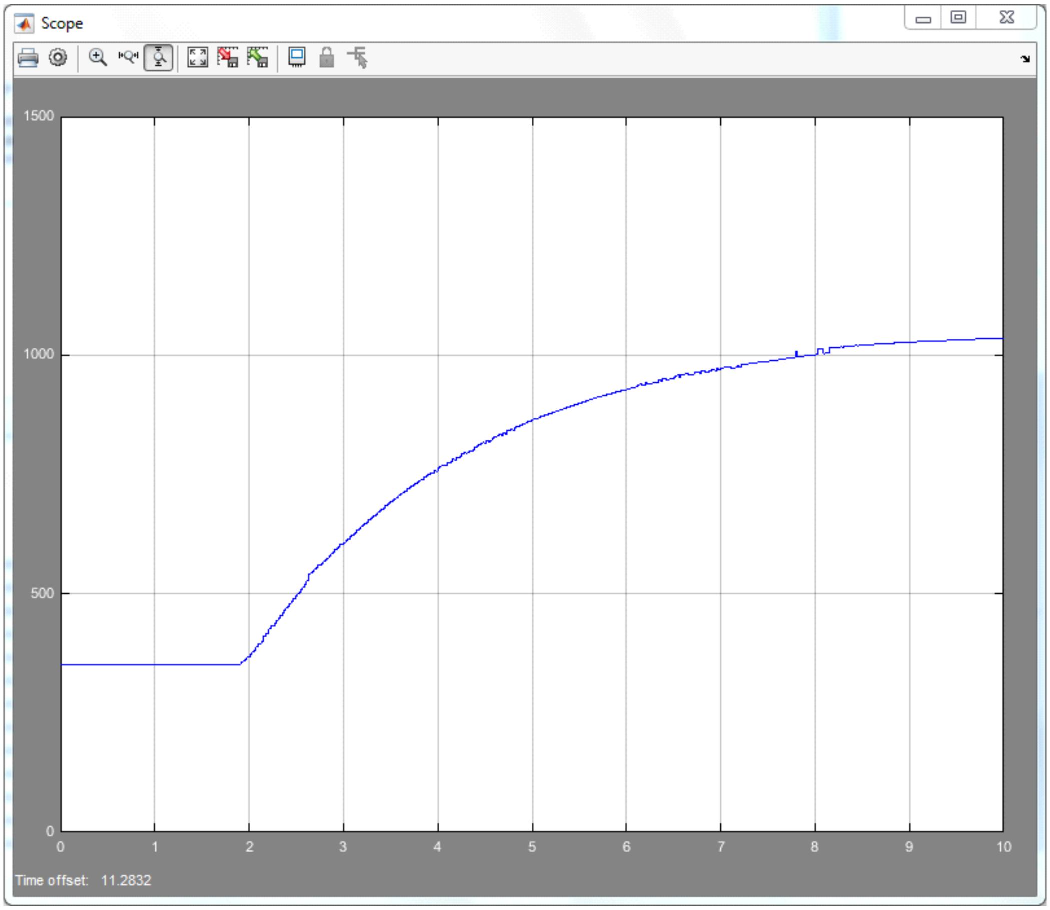
This is the controller we ran on the elevator: Rise Time = 2.96 seconds, Settling Time = 4.66 seconds.



This is the step response observed on the actual elevator:



Response time is slower & steady state error is larger due to friction which is not modelled.


Scope
1500
1000
500
0
0
1
2
3
4
5
6
7
8
9
10
Time offset: 11.2832