Edward Venator
EECS 305 Project, Spring 2012

Introduction

In this project, I designed a controller for pitch control of wind turbine blades. The controller is given a desired turbine velocity Ω_r and must control the pitch angle of the blades to maintain that speed. The transfer function of the wind turbine determines its velocity as a function of the blade pitch angle β:

$$P(s) = \frac{\Omega_r}{\beta_r} = \frac{K_1 \omega_g}{(\tau_g s + 1)(s^2 + 2\zeta_g \omega_g + \omega_g^2)}$$

The architecture specified for this controller was a PID controller with a low-pass filter on the differential term. This controller can be described by the transfer function:

$$C(s) = \frac{\beta_r}{e} = k_p + \frac{k_i}{s} + \frac{k_d}{\gamma s + 1}$$

The parameters of the controller had to be tuned to meet several specifications. First, the gain margin of the controller was to exceed 6dB and the phase margin was to be in the range of 30° to 60°. Second, the transient response was to have a rise time T_r of less than 5 seconds and a peak time T_p of less than 11 seconds. Lastly, the step response steady state error E_{ss} was to be 0. My PID controller, which I tuned experimentally, exceeded these parameters.

Results

Parameters and Transfer Functions

My final PID gains and filter time constant were as follows:
k_p = -0.005
k_i = -0.0025
k_d = 0.005
γ = 1.3
With these parameters, the complete transfer function of the controller is:

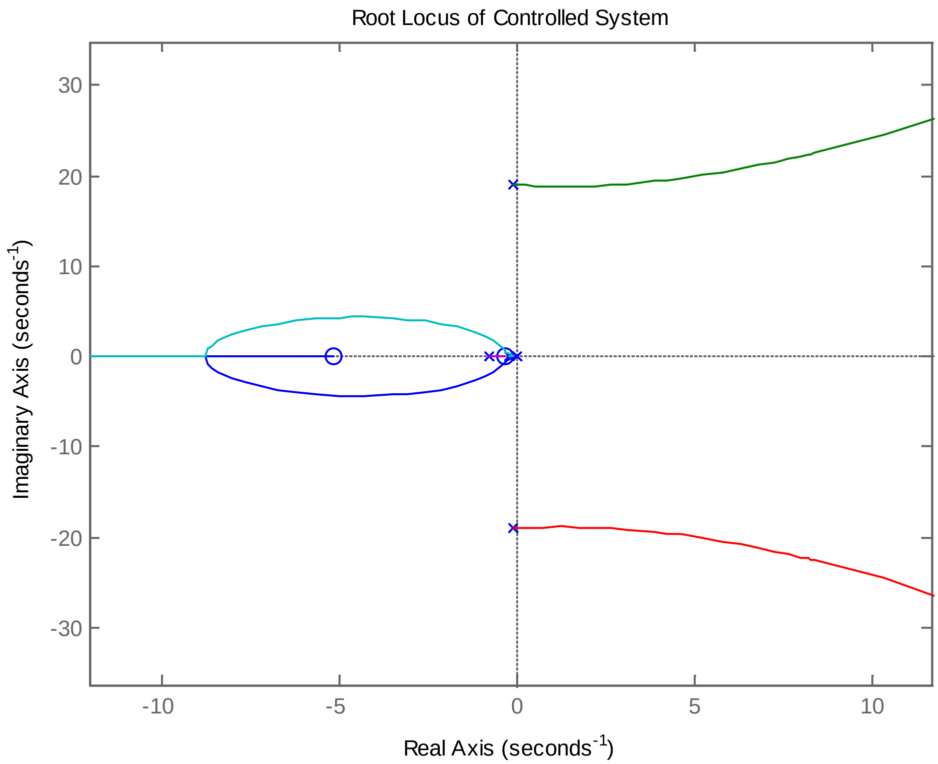
$$C(s) = \frac{-0.0015\, s^2 - 0.00825\, s - 0.0025}{1.3\, s^2 + s}$$

and the complete (open-loop) transfer function of the controlled system is:

$$L(s) = \frac{193.8\, s^2 + 1066\, s + 323}{6.24\, s^5 + 7.523\, s^4 + 2255\, s^3 + 2202\, s^2 + 361\, s}$$

Root Locus

With these PID gains, the root locus of the controlled system looks like this:



The final roots of the system are:

0

-0.1140 +18.9997i

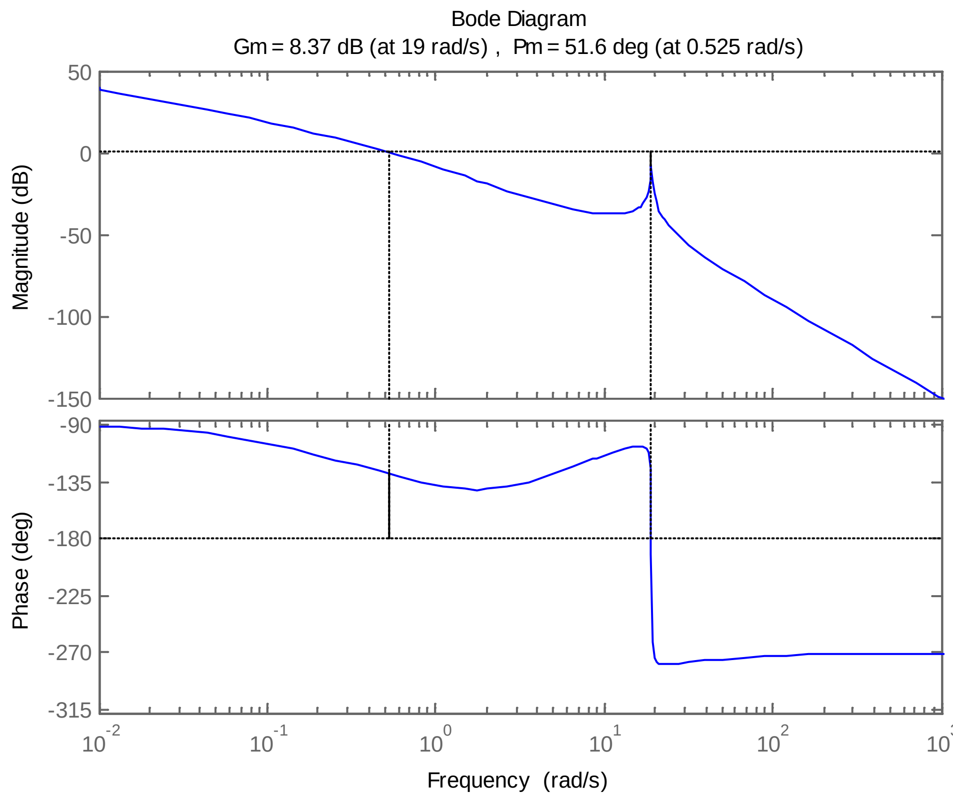
-0.1140 -18.9997i

-0.7692

-0.2083

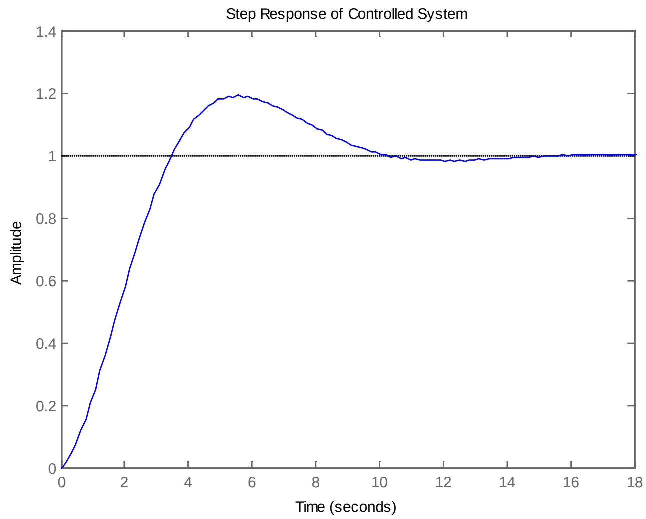
Bode Plot and Stability Margins

With these PID gains, the controlled system has the following Bode Plot:



Note that the gain margin (8.37 dB) and the phase margin (51.6°) are within the specifications.

Step Response and Steady State Error



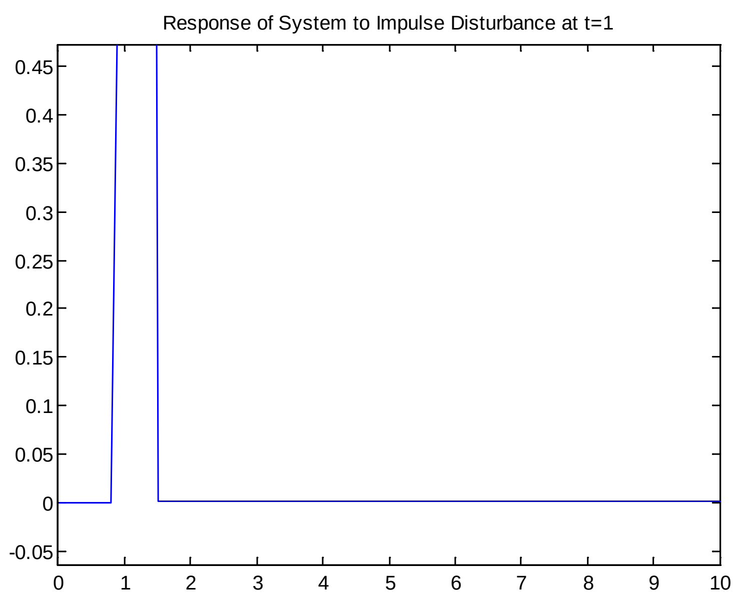
Note that T_r is under 4 seconds and T_p is under 6 seconds. The steady state error E_{ss} can be found using Final Value Theorem from the open loop transfer function L(s) (above).

$$E_{ss} = \frac{1}{1 + \lim_{s\to 0} L(s)}$$

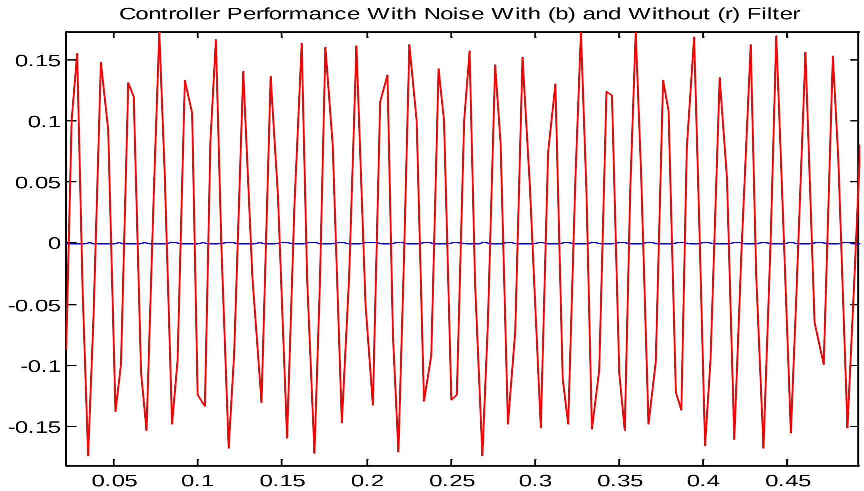
$$E_{ss} = \frac{1}{1 + \infty} = 0$$

Impulse Disturbance Response

For the impulse disturbance below, I used a 0.5 second pulse with an amplitude of 1 at t=1.

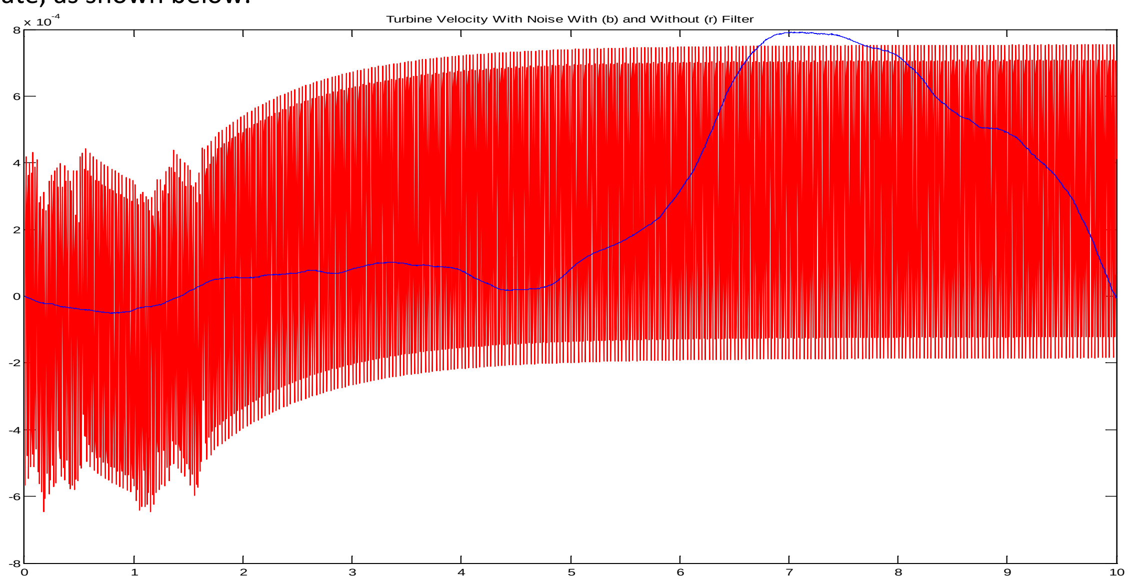


Noise Immunity Analysis



The plot above shows the output signal of the controller when 60Hz noise with an amplitude of .1 is added to the velocity measurement going into the controller. The blue line shows the controller I designed, and the red line shows the same controller without a low-pass filter.

As you can see, the low-pass filter greatly reduces the amount of spurious control effort due to noise. Whereas the noise is attenuated to practically nothing by the filtered controller, the unfiltered controller actually amplifies the noise. The resulting control effort causes the turbine's velocity to oscillate, as shown below.



Appendix 1: MatLab Code

```
%Edward Venator
%EECS 305 Spring 2012
%Project

%Plant Parameters
K1 = -6800;
T_g = 4.8;
g_g = 0.006;
w_g = 19;

%Plant Transfer Function
p_num = K1 * w_g;
p_den = conv([T_g 1], [1 (2 * g_g * w_g) (w_g^2)]);
p = tf(p_num, p_den);
figure(1);
margin(p);
%rltool(p);

%Controller Transfer Function
k_p = -.005;
k_i = k_p * .5;
k_d = -k_p * 1;
gamma = 1.3;
proportional = tf(k_p,1);
integral = tf(k_i,[1 0]);
derivative = tf([k_d 0], [gamma, 1]);
pi = parallel(proportional, integral);
c = parallel(pi, derivative)

%Controller and Gain in Series
g = series(c, p);
figure(2);
margin(g);
figure(3);
rlocus(g);
title('Root Locus of Controlled System');

%System with Feedback
sys = feedback(g,1);
figure(4);
step(sys);
title('Step Response of Controlled System');

%Impulse Disturbance Response
ref = 0;
noise = 0;
impulse = 1;
sim('plant.mdl');
figure(5);
plot(y.time,y.signals.values);
title('Response of System to Impulse Disturbance at t=1');
```

```
%Noise Response Plot
impulse = 0;
noise = .1;
sim('plant.mdl');
beta_filt = beta;
y_filt = y;
gamma = 0;
sim('plant.mdl');
figure(6);
plot(beta_filt.time,beta_filt.signals.values,'b',beta.time,beta.signals.values,'r');
title('Controller Performance With Noise With (b) and Without (r) Filter');
figure(7);
plot(y.time,y.signals.values,'r',y_filt.time,y_filt.signals.values,'b');
title('Turbine Velocity With Noise With (b) and Without (r) Filter');
```

Appendix 2: Plant.mdl