**MEMORANDUM**

**TO:** Dr. Bedillion

**FROM:** Catherine Pavlov and Yigit Yakupoglu

**DATE:** April 30, 2018

**RE:** MLC Laboratory

Abstract

This lab provided a survey of the control techniques learned in 24-773: Multivariable Linear Control. We used five different methods for generating controllers to make the Quanser Aero track a desired trajectory. The Quanser Aero is a DIDO system with pitch and yaw control of a lever arm, and is actuated by a pair of perpendicularly mounted fans. The nominal plant and uncertainty bounds for the device are known, and were used to design controllers enabling the device to track a square wave trajectory with each of its degrees of freedom.

Five controllers were employed: inverse loopshaping, H2 optimal control, H­inf optimal control, μ synthesis, and Hinf loopshaping. All controllers were designed in simulation and executed on the hardware, to varying degrees of success. The H2 and Hinf controllers performed the best of all five, while H2 tracked better than any other on hardware. In simulation, the Hinf controller had the lowest error and it is the one closest to being robustly stable. In practice, no controllers met robust stability and robust performance requirements, though all were nominally able to track the trajectory.

This report has been proofread by all members of the group:

Catherine Pavlov 4/30/2018

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Print Name Signature and Date

Yigit Yakupoglu 4/30/2018

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Print Name Signature and Date

Introduction

In this lab, we investigated the performance of various controller generation techniques on the Quanser Aero. The Quanser Aero is a DIDO system consisting of two fans mounted perpendicularly on opposing ends of a 2DOF lever arm. The system is able to move about its pitch and yaw axes, with the voltages of the two fans used to control its motion. In this lab we employed loopshaping, H2 synthesis, H∞ synthesis, μ synthesis, and H∞ loop shaping in order to make the system track a desired trajectory.

For all parts of this lab, the desired trajectory consisted of square waves with a pitch angle of π/6 radians and a frequency of .4 rad/s, and a yaw angle of π/4 radians with a frequency of .5 rad/s. The control configuration for the plant is shown below.

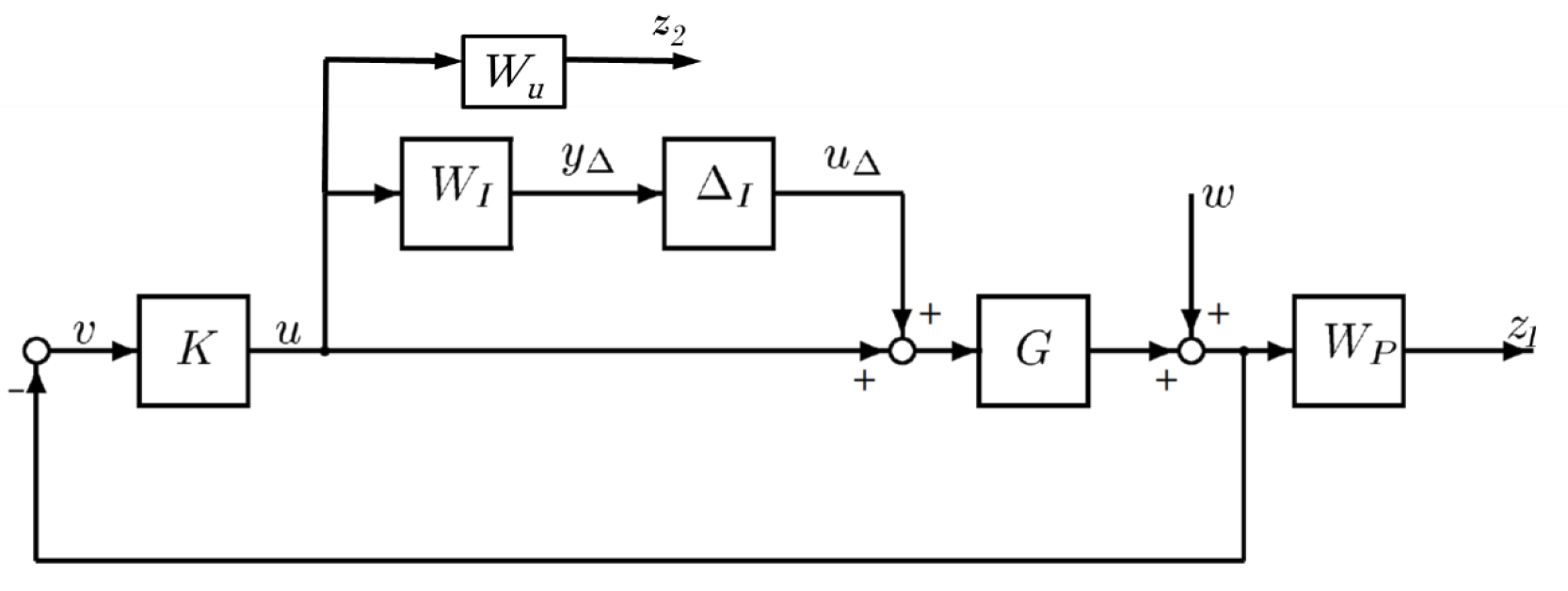


Figure . Control configuration for the Quanser Aero

The performance weight was designed to reject DC disturbances by a factor of 100 while keeping sensitivity peaking below 3, and the control weight was set to ensure controller usage did not exceed the maximum system voltage of 25 V.

Preliminaries

1. First, we found the poles and zeros of the nominal plant. We found that there are no zeros, and the poles are all in the closed left half plane. The poles are:
   * -0.1625 + 1.2982i
   * -0.1625 – 1.2982i
   * -1.0004
   * 0

As there are no RHP poles or zeros, we do not have any fundamental control limitations due to waterbed effects.

1. Below is shown the Bode plot for the nominal plant. Note that there is a peak at 1.34 rad/s, due to the complex conjugate pair of poles. Output direction one always has negative gain, so it will have poor tracking at DC.

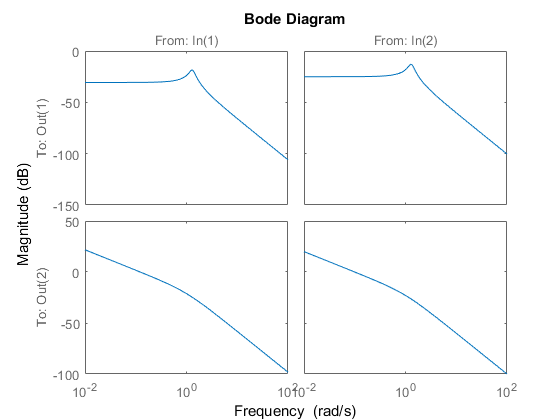


Figure . Bode plot of nominal plant

1. We next built a Simulink model based on the nominal plant, including a saturation nonlinearity to limit the voltages to +/- 25V.

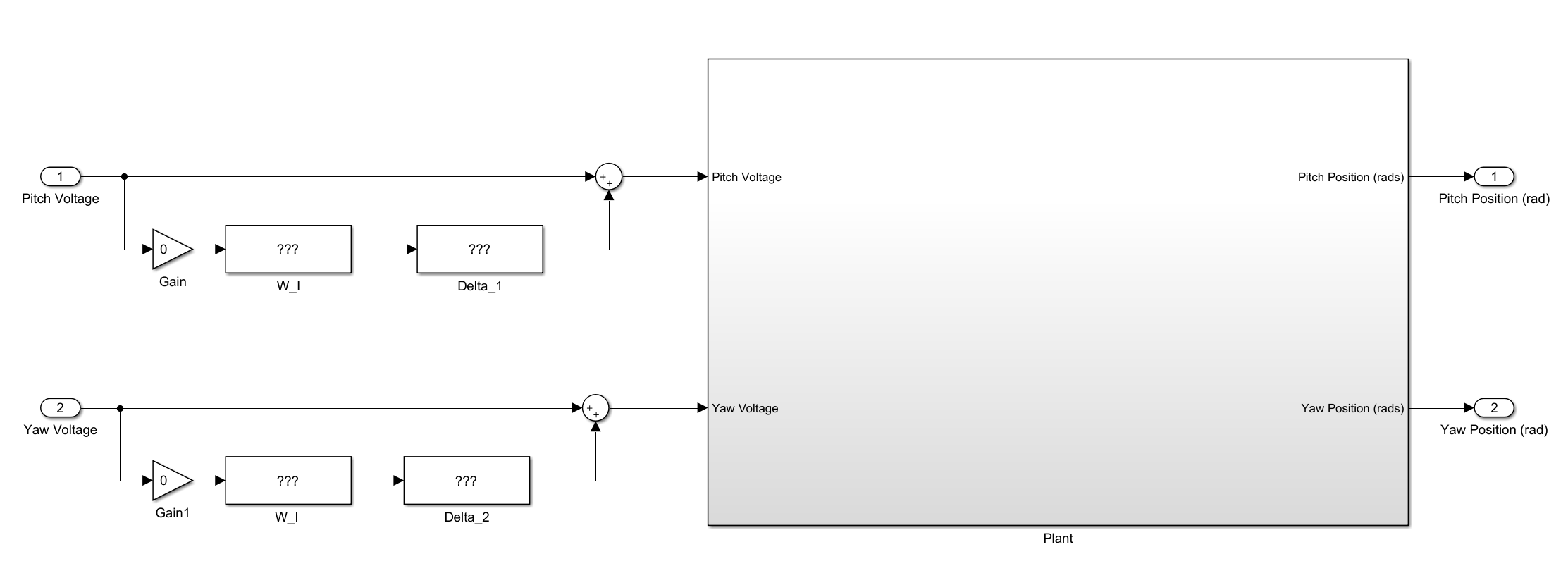


Figure . Simulink model of nominal plant.

1. Given 30% uncertainty in Jp and Jy, we built an uncertain model of the plant and fit input multiplicative uncertainty weights using *ucover*. Below, you can see samples from the uncertain plant (blue) plotted with the plant fit with our generated uncertainty weight (red).

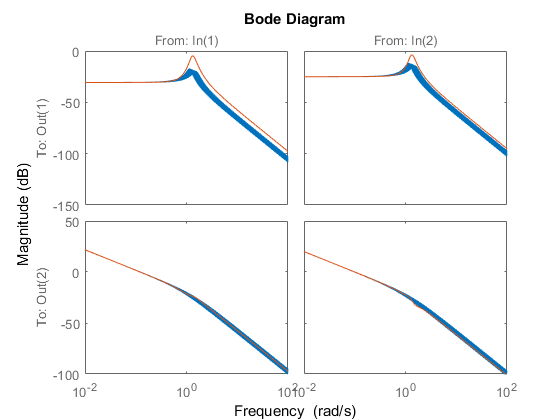


Figure . Generated uncertainty weights for the uncertain plant.

Loop Shaping

We first generated an inverse-based controller using classical loopshaping, using a desired loopshape of Ld = wc/s. The crossover frequency was initially set to 5 rad/s. As the generated controller was improper, with a zero excess of 2, we added a repeated high frequency pole at 1000 rad/s to make the controller realizable. Thus our controller is

1. Below, you can see the output of our simulation with the described controller implemented for no uncertainty. Note that there is a fair deal of overshoot, and that the system angles only vaguely resemble square waves.

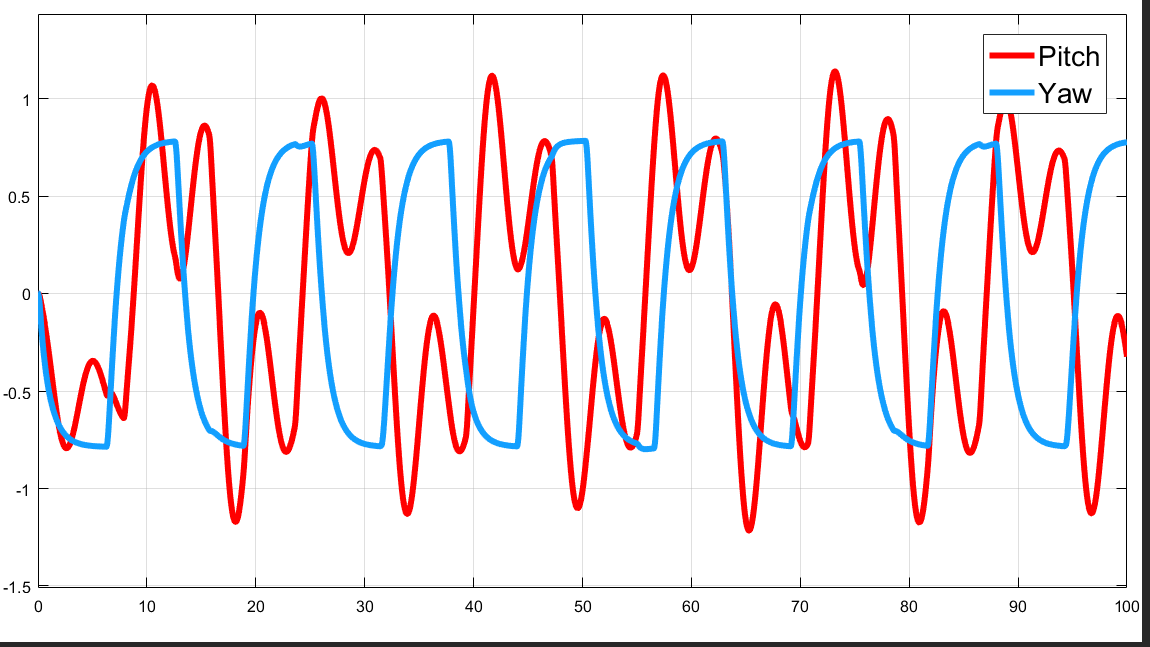


Figure . Pitch and yaw angles for loop shaping controller run on the nominal plant

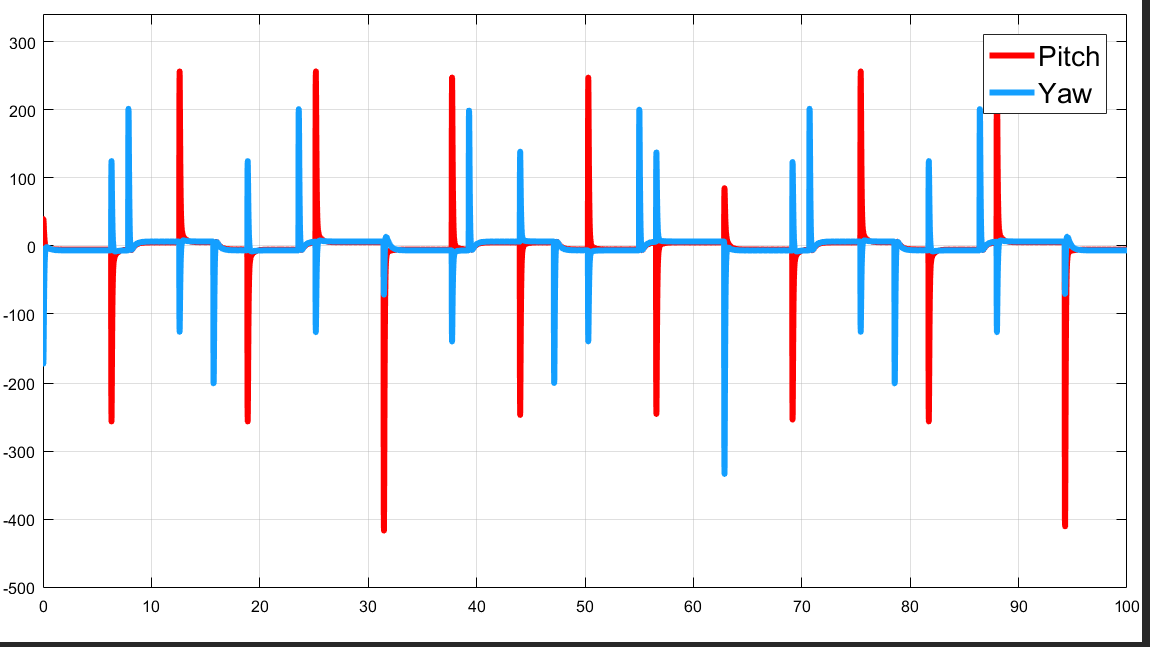


Figure . Pitch and yaw voltages for loop shaping controller implemented on the nominal plant

1. Next, we tested the system for robust stability by running it on 10 samples of the uncertain plant.

As you can see, the controller performs very poorly. While you can see the shape of the trajectory followed by the nominal plant, there is very high noise with large amounts of overshoot (enough to easily hit the hard stops). This controller is clearly not robustly stable, and does not have robust performance.

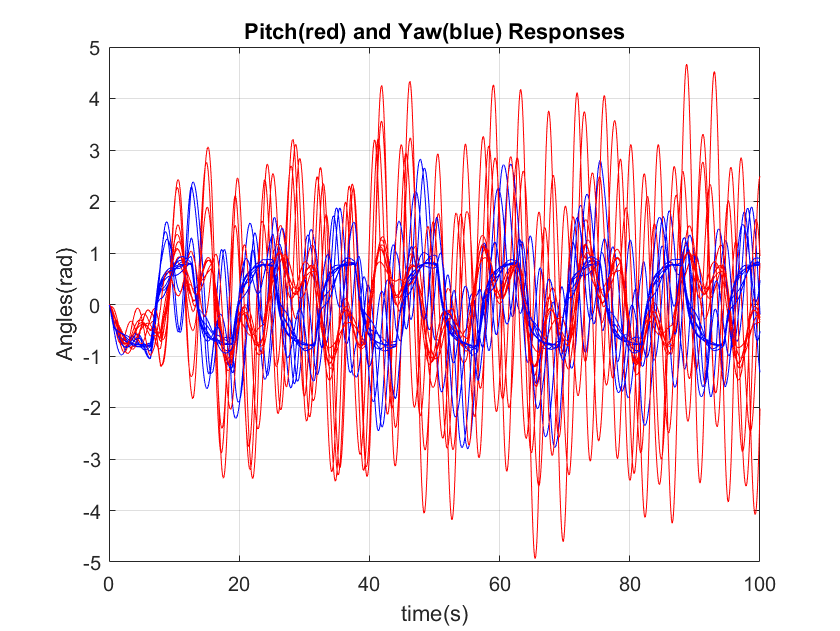


Figure . Pitch and yaw angles for loop shaping controller run on 10 samples of the uncertain plant.

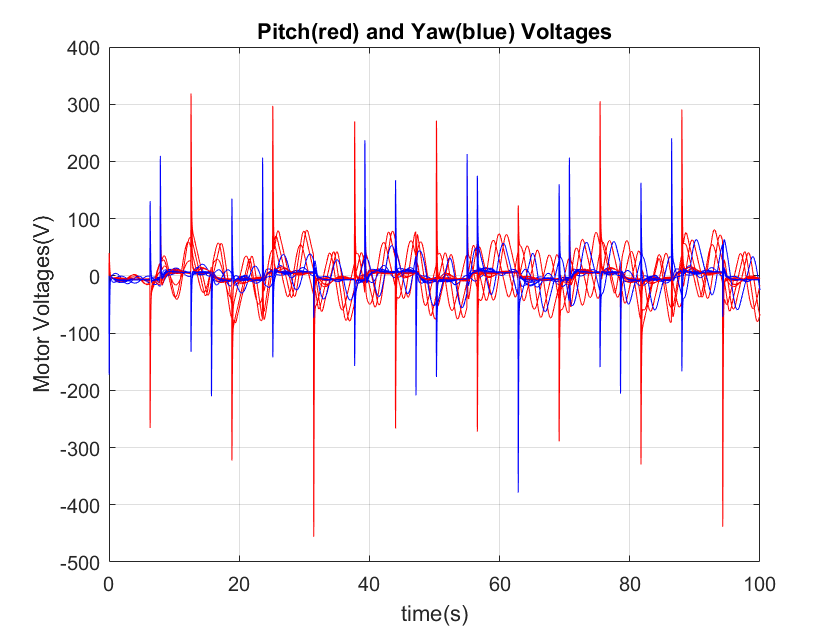


Figure . Motor voltages for the pitch and yaw angles for the loop shaping controller run on 10 samples of the uncertain plant.

1. We then implemented the controller on the hardware with no uncertainty. It did not track the desired trajectory very well, often hitting the hard stops of the system. Below are plots of the system angles and voltages.

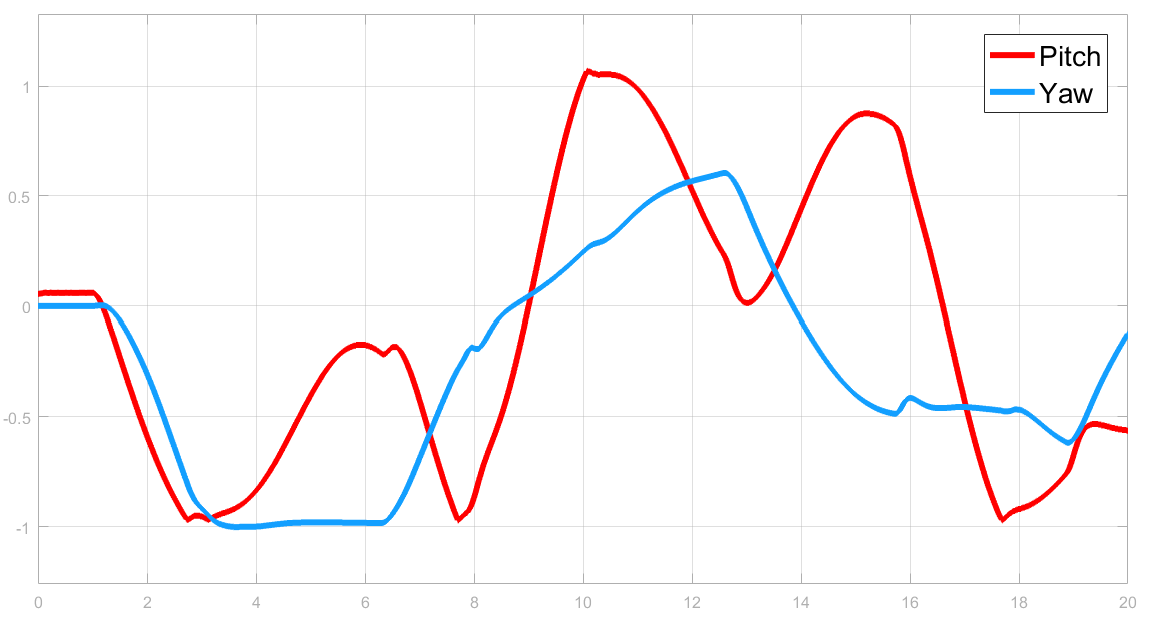


Figure . Pitch and yaw angles for the loop shaping controller implemented on the Quanser Aero for the nominal plant.

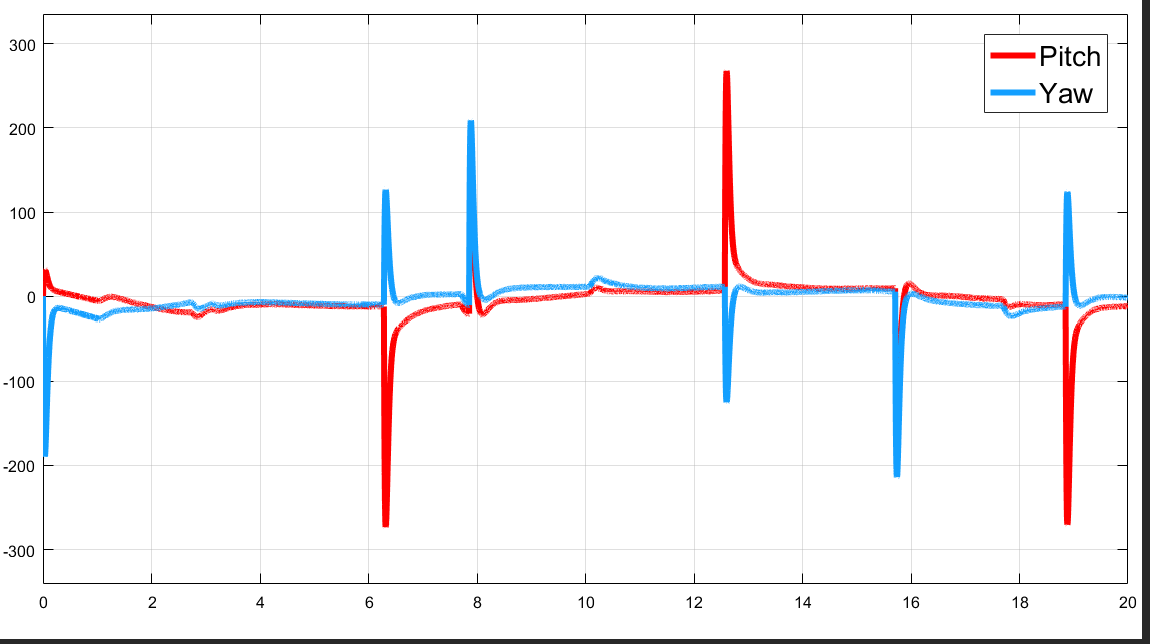


Figure . Controller voltages for the loop shaping pitch and yaw implemented on the Quanser Aero for the nominal plant.

1. Finally, we ran the hardware simulation for five samples of the uncertain plant. The system often hit the hard stops, though beyond this overshoot it did reasonably track the trajectory. Below are plots of the pitch and yaw angles and voltages for all five samples.

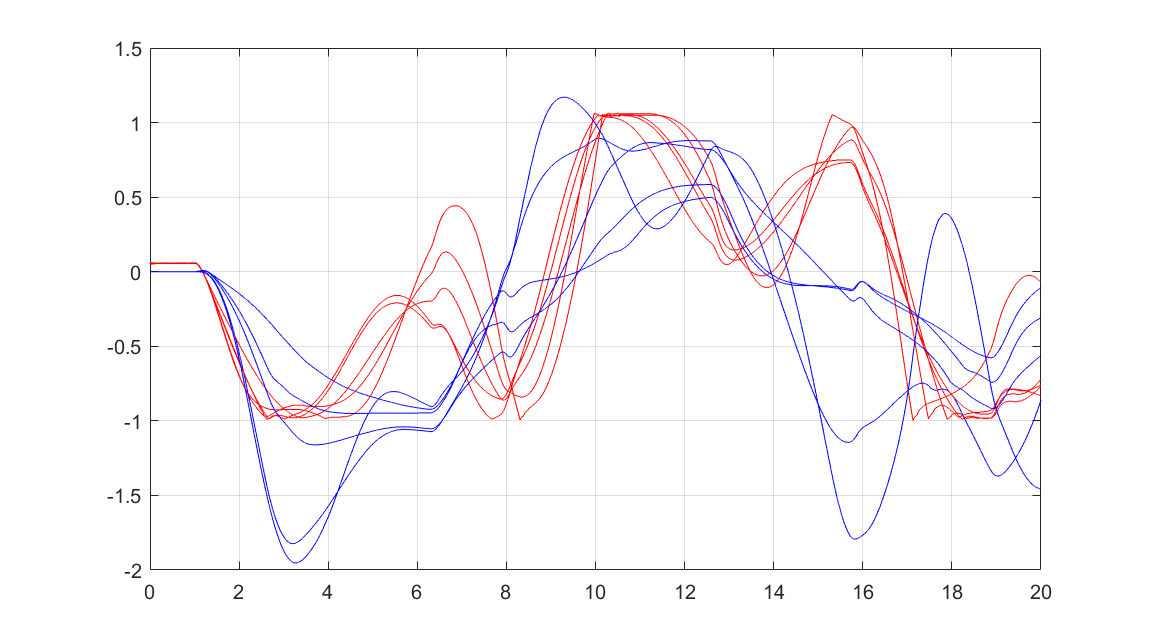


Figure . Pitch (red) and yaw(blue) angles for five trials of the loop shaping controller run on the uncertain plant.

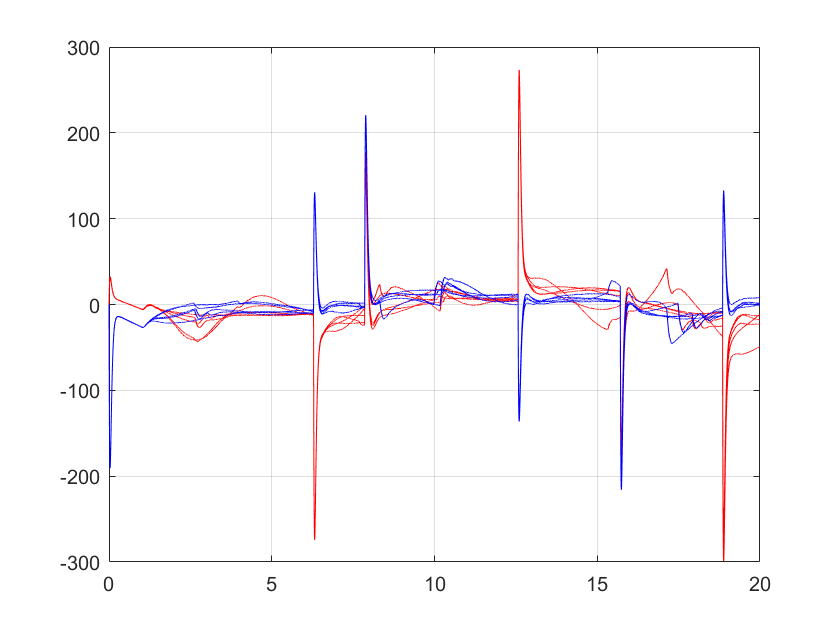


Figure . Pitch (red) and yaw(blue) voltages for five trials of the loop shaping controller run on the uncertain plant.

Video of the controller run on the nominal plant and five samples of the uncertain plant can be viewed here: <https://www.youtube.com/watch?v=RZdholooI4M>

H2 Optimal Synthesis

Next, we synthesized a controller using H2 optimal control, assuming no uncertainty. With tuning, we found a crossover frequency of wc = 10 rad/s to have good performance.

1. Below is the response from the system with no uncertainty, showing system angles and controller usage. Note that the angles have relatively minimal overshoot compared to the loopshaping controller, and overall track the reference much more closely.

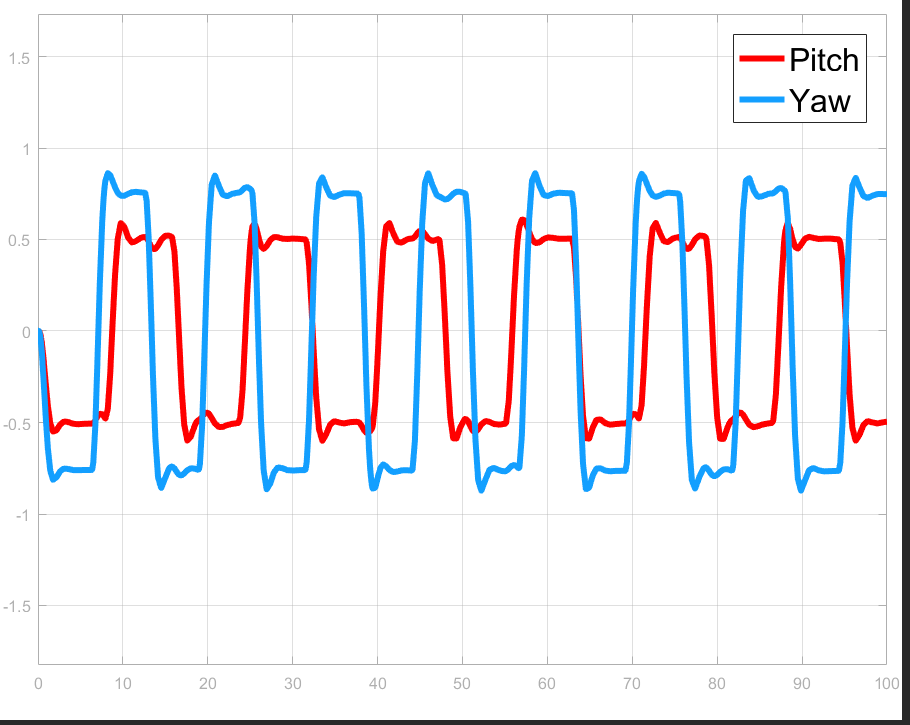


Figure . Output of angles from simulation of H2 optimal controller on plant with no uncertainty.

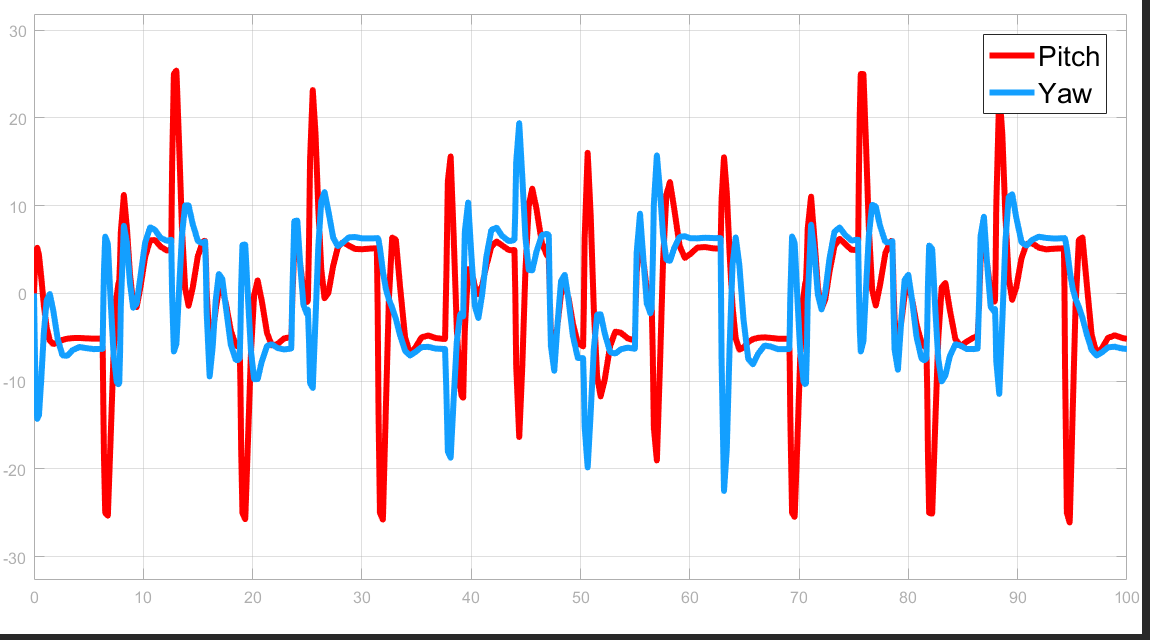


Figure . Controller voltages for pitch and yaw motors from simulation of H2 optimal controller on plant with no uncertainty

1. Next, we checked robust stability and robust performance of the system. The lower bound of the closed loop system turned out to be 0.6175, meaning the system is stabilizable if uncertain elements stay less than 0.6175 normalized units of their nominal values. However, since this number is less than 1, closed loop is not robustly stable, which can be as well checked from µ=1/0.6175=1.6194. The lower bound for robust performance is found to be 0.04, so the system is not robustly performing at all.
2. We then implemented the controller on the hardware. It performed far better than the loop shaped controller, not hitting the hard stops. The yaw reference is tracked much better than pitch reference. Also, the coupling effect between states can be observed. The pitch angle reacts to movements of yaw angle greatly.

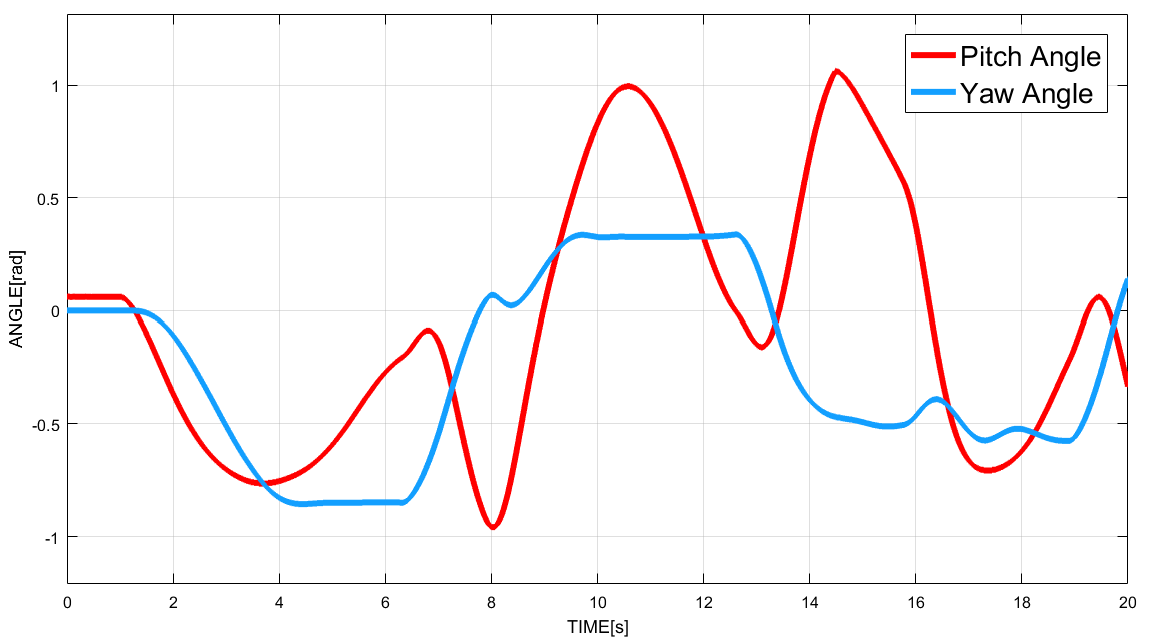


Figure . Pitch and yaw angles for H2 optimal controller implemented on the Quanser Aero for the nominal plant.

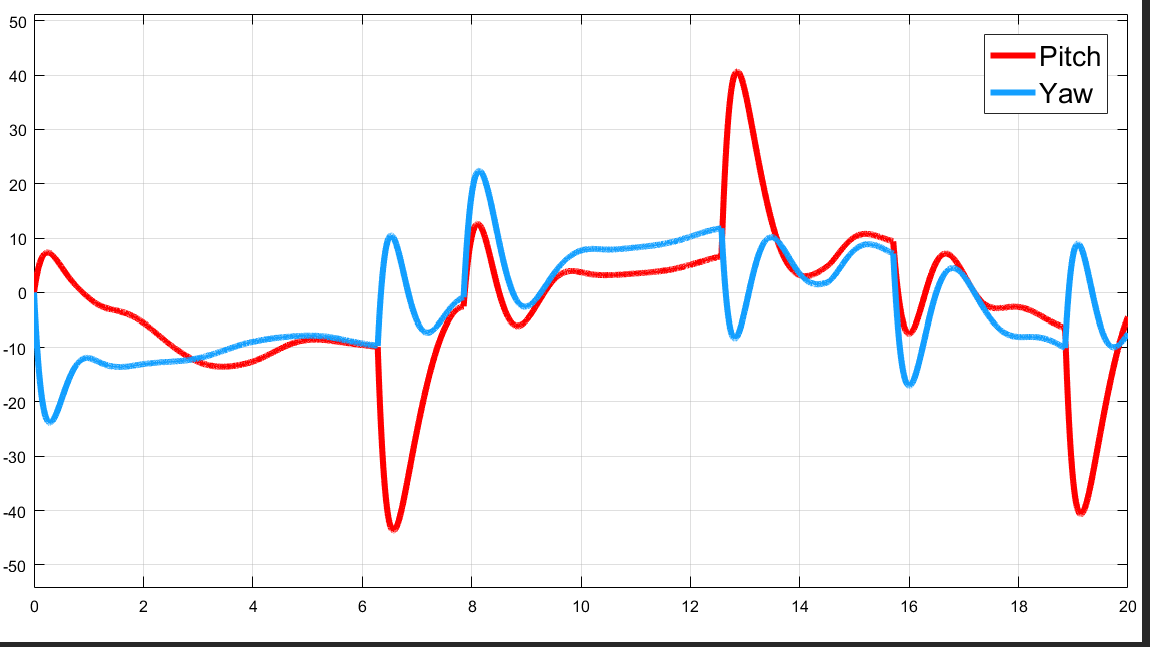


Figure . Pitch and yaw voltages for the H2 optimal controller implemented on the Quanser Aero for the nominal plant.

1. Finally, we ran the simulation for five samples of the uncertain plant. Again, the controller overall performed better than the loop shaped controller. One trial hit the hard stops rather hard, but the others hit only occasionally, following the trajectories relatively well.

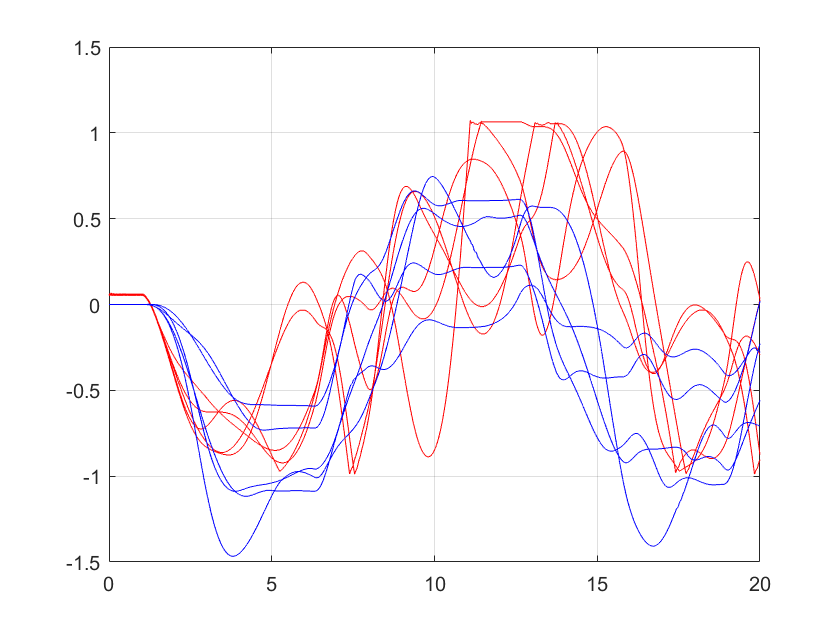


Figure . Pitch (red) and yaw (blue) angles for the H2 optimal controller implemented on the Quanser Aero for five samples of the uncertain plant.

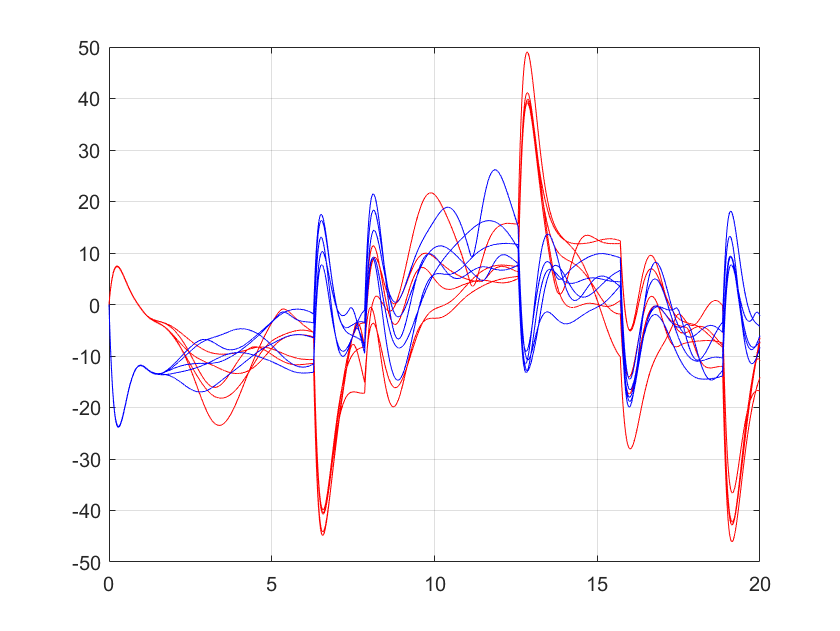


Figure . Pitch (red) and yaw (blue) voltages for the H2 optimal controller implemented on the Quanser Aero for five samples of the uncertain plant.

Video of the controller run on the nominal plant and five samples of the uncertain plant can be viewed here: <https://www.youtube.com/watch?v=qRbQcT-jgC8>

H∞ Optimal Control

We then solved the H∞ optimal control problem, maximizing the crossover frequency while maintaining γ < 1. Our ultimate crossover frequency was 1.07 rad/s, with a corresponding γ = 0.998.

1. Below is the response of our controller when simulated with no uncertainty. There is even less overshoot than observed with the H2 optimal controller.

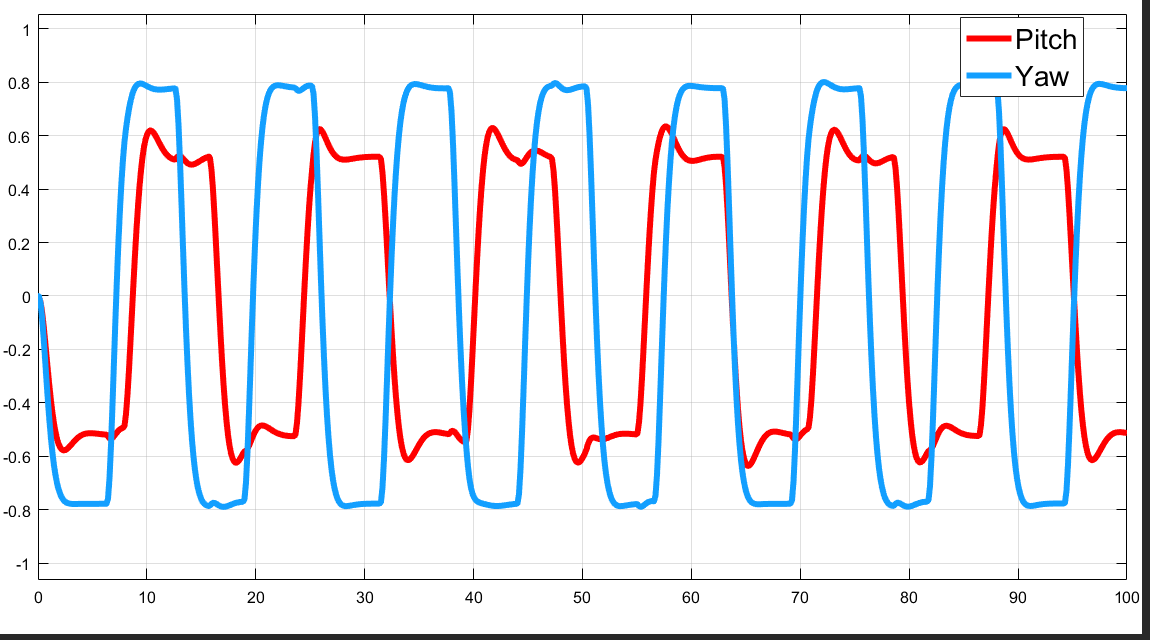


Figure . Output of angles from H infinity optimal controller on plant with no uncertainty.

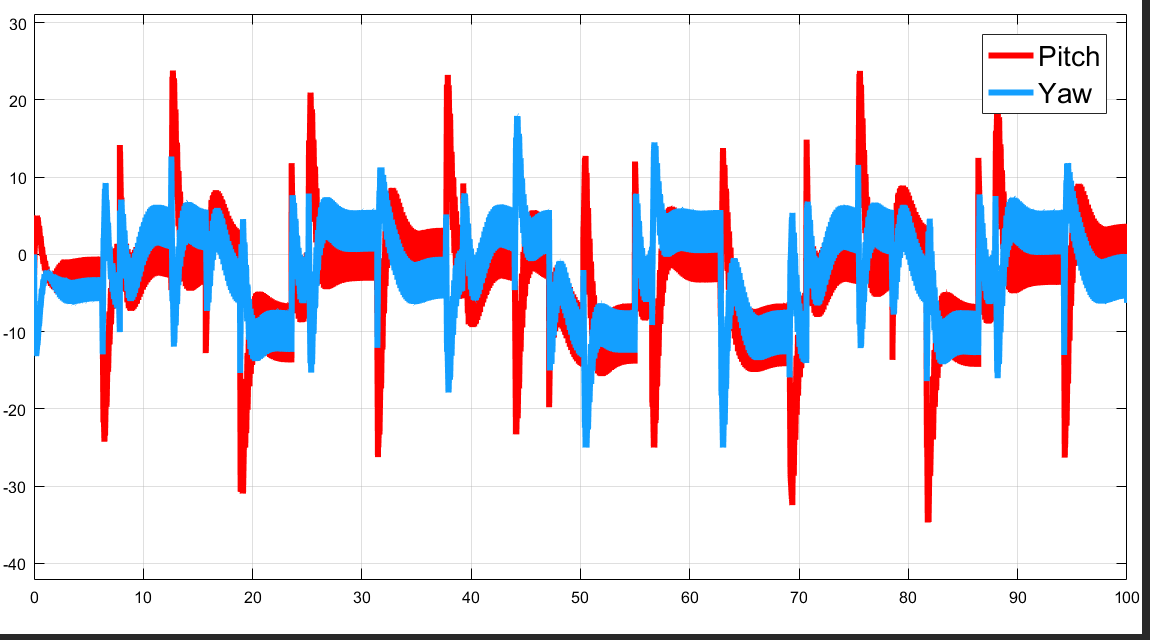


Figure . Controller voltages for pitch and yaw motors from simulation of Hinf optimal controller on plant with no uncertainty

The controller usage looks odd in this plot – note that while the lines appear to be thick, this is in fact due to the control usage oscillating at high frequency.

1. The closed loop lower bound is found to be 0.94, with µ=1.06. So, it can be said that the system is nearly robustly stable. The lower and the upper bound for robust performance is found to be 0.45. Meaning that for not all plants the performance has been achieved. When compared to H2, there is a great stability and performance increase, on the nominal plant.
2. We then implemented the controller on the hardware with no uncertainty. Below are plots of the angles and voltages of the Quanser Aero.

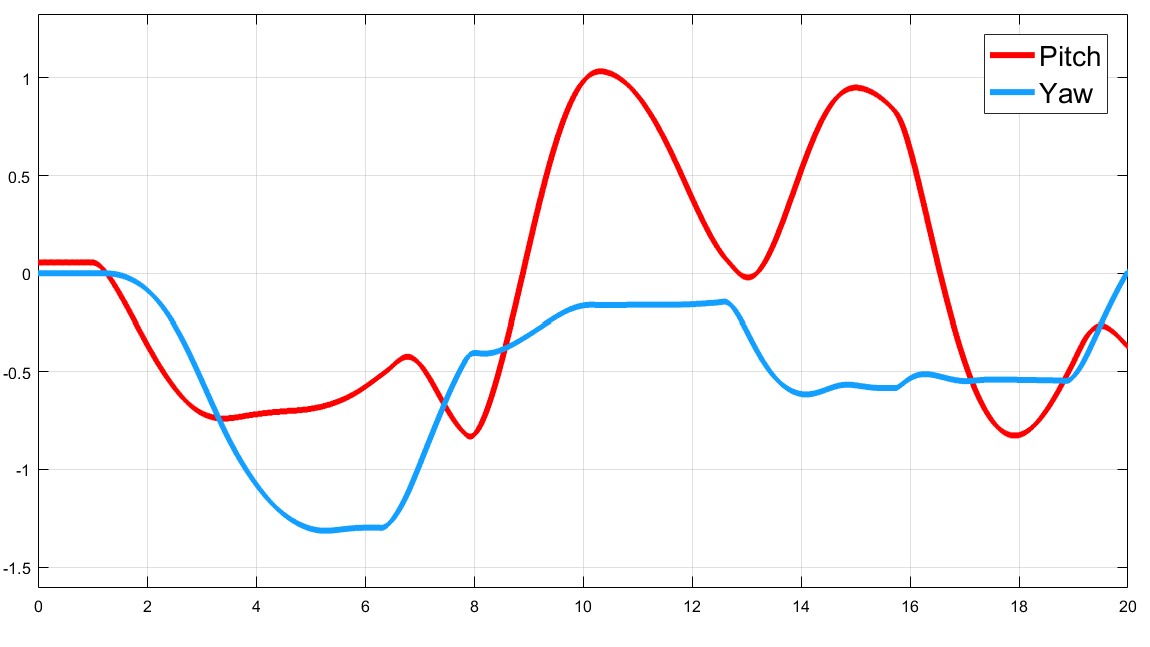


Figure . Pitch and yaw angles for the Hinf optimal controller implemented on the Quanser Aero for the nominal plant

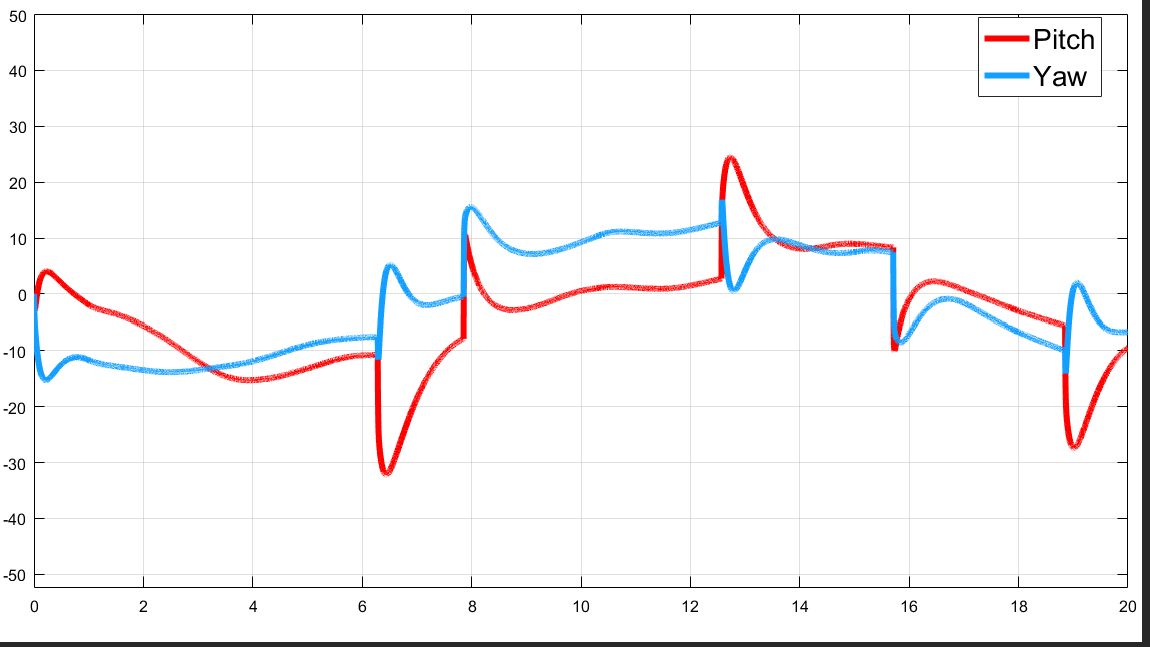


Figure . Pitch and yaw voltages for the Hinf optimal controller implemented on the Quanser Aero for the nominal plant.

1. We next ran the controller on the Quanser Aero for five samples of the uncertain plant. It tracked well, with minimal hitting of the hard stops.

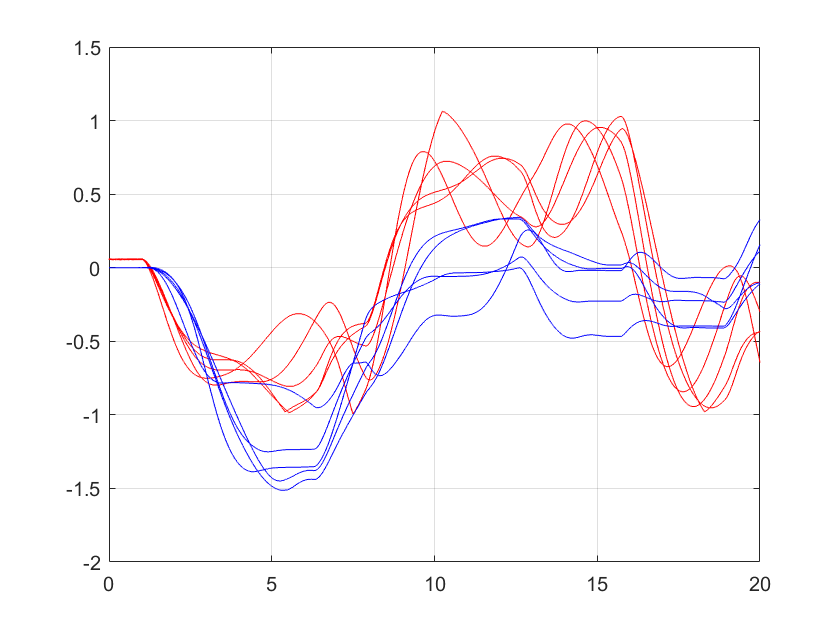


Figure . Pitch (red) and yaw (blue) angles for the Hinf optimal controller implemented on the Quanser Aero for five samples of the uncertain plant.

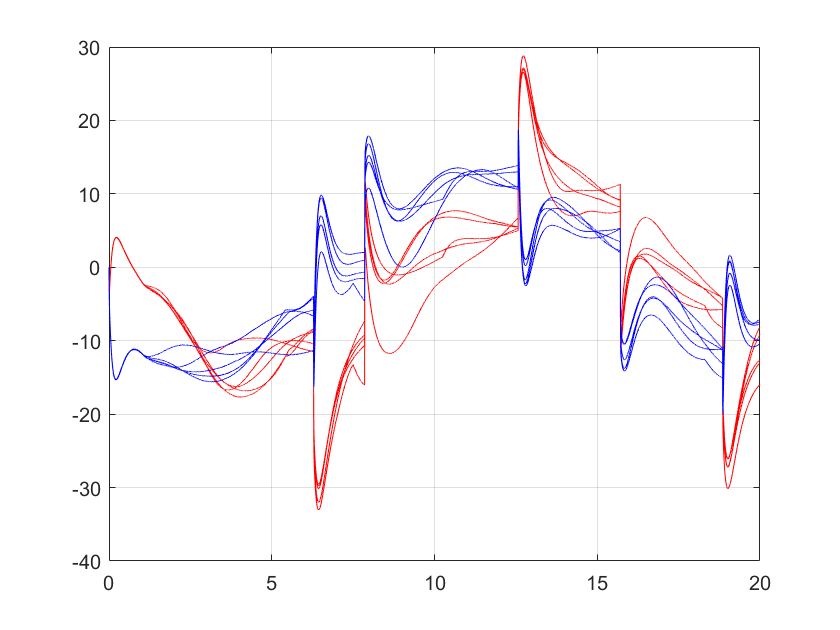


Figure . Pitch (red) and yaw (blue) voltages for the Hinf optimal controller implemented on the Quanser Aero for five samples of the uncertain plant.

Video of the controller run on the nominal plant and five samples of the uncertain plant can be viewed here: <https://www.youtube.com/watch?v=Ai3_BNFdNkM>

μ-Synthesis

Next, we used μ-synthesis to generate a controller, using the same weights as for the H∞ problem. We then reduced the controller order to sixth order to match that of the H∞ optimal controller.

1. Below is shown the controller run on the nominal plant. It appears to track decently well, with some oscillation and overshoot, but still roughly tracking a square wave.

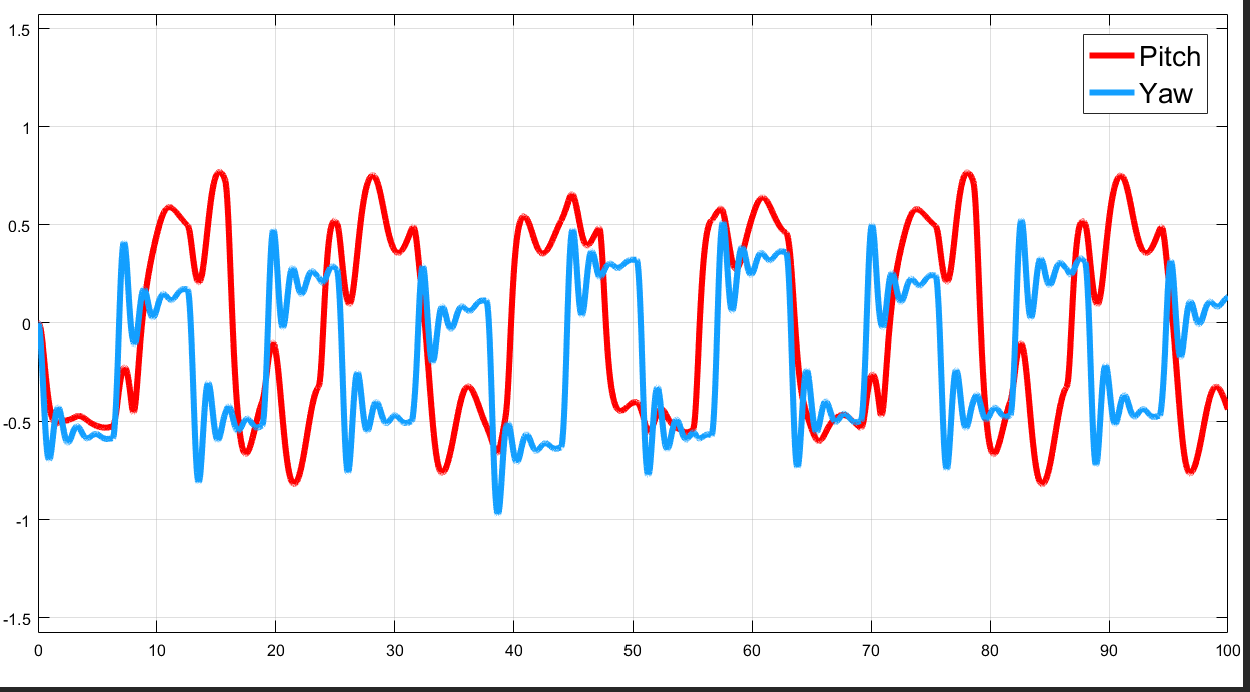


Figure . Output angles from μ synthesis controller run on nominal plant.

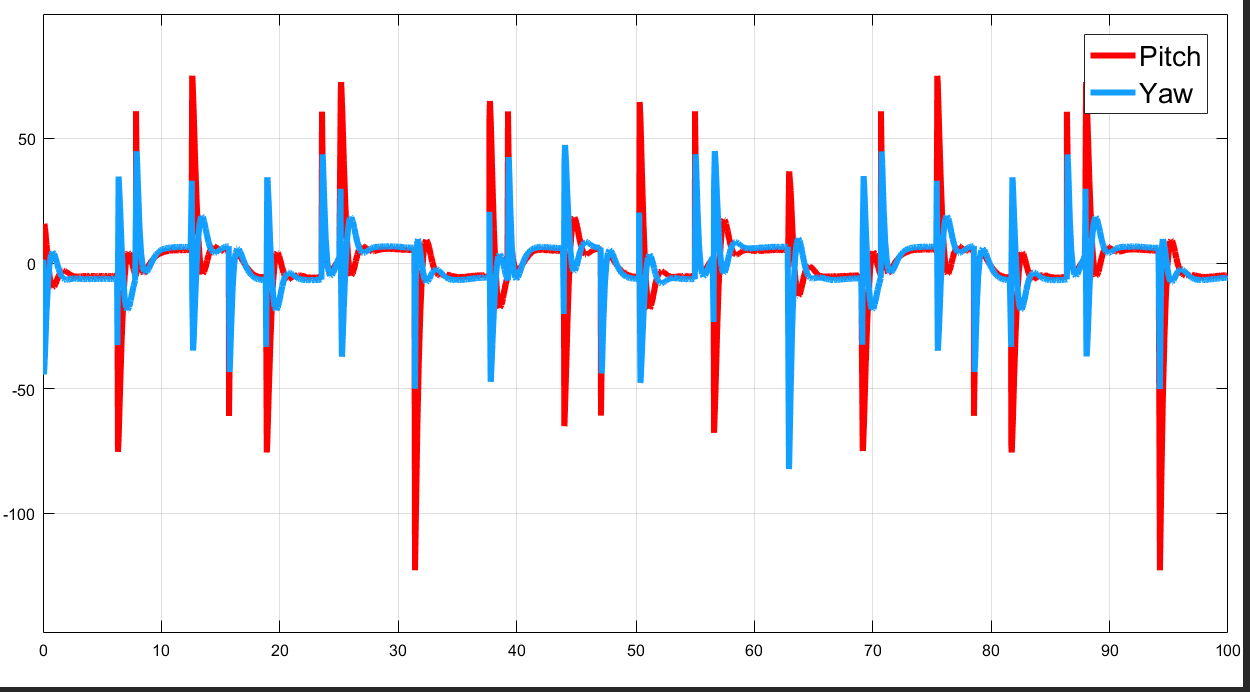


Figure . Controller voltages from μ synthesis controller run on nominal plant.

1. The simulation was then run on ten samples from the uncertain plant. The responses can be seen below. While the controller does not appear to go unstable, it has very poor performance.

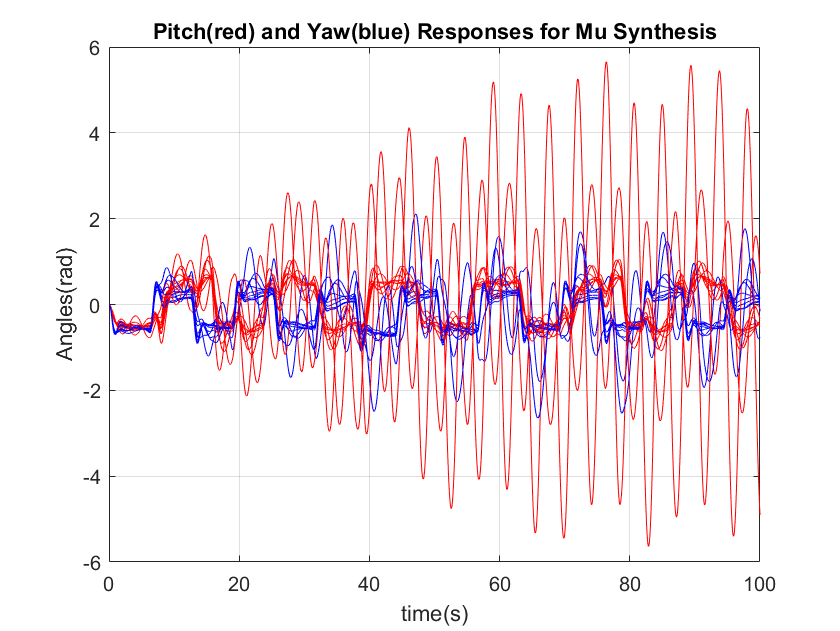


Figure . Pitch and yaw angles for 10 samples of the uncertain plan.

1. We then implemented the controller on the hardware for the nominal plant. Below are the pitch and yaw angles as well as the control voltages for the system.

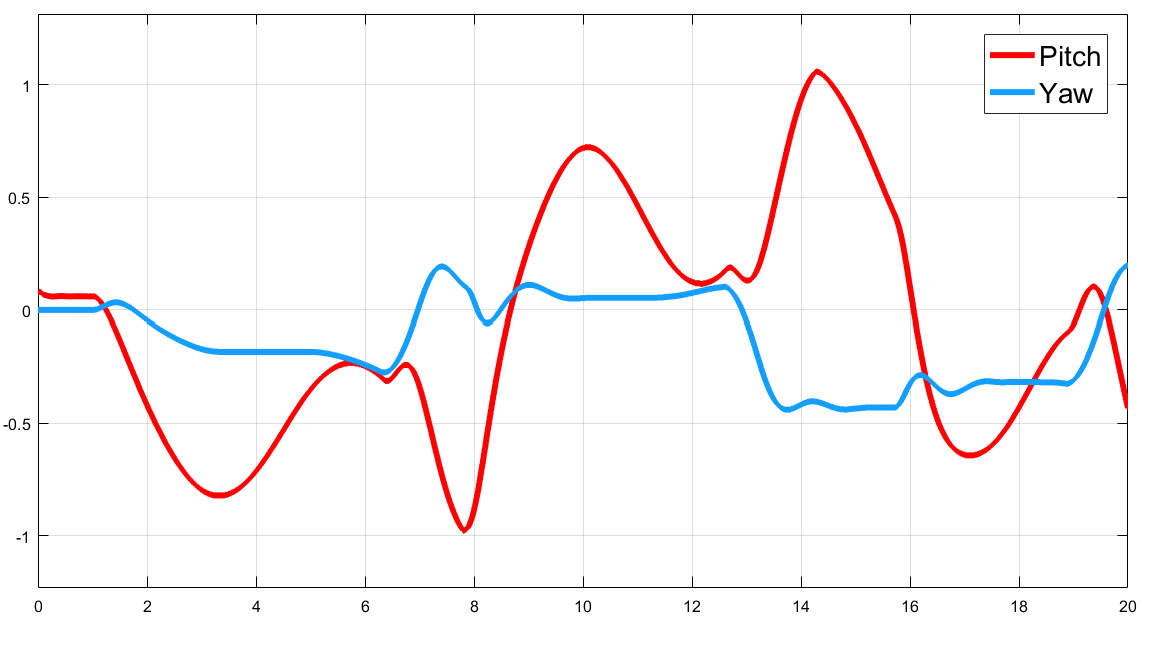


Figure . Pitch and yaw angles for the μ synthesis controller implemented on the Quanser Aero for the nominal plant

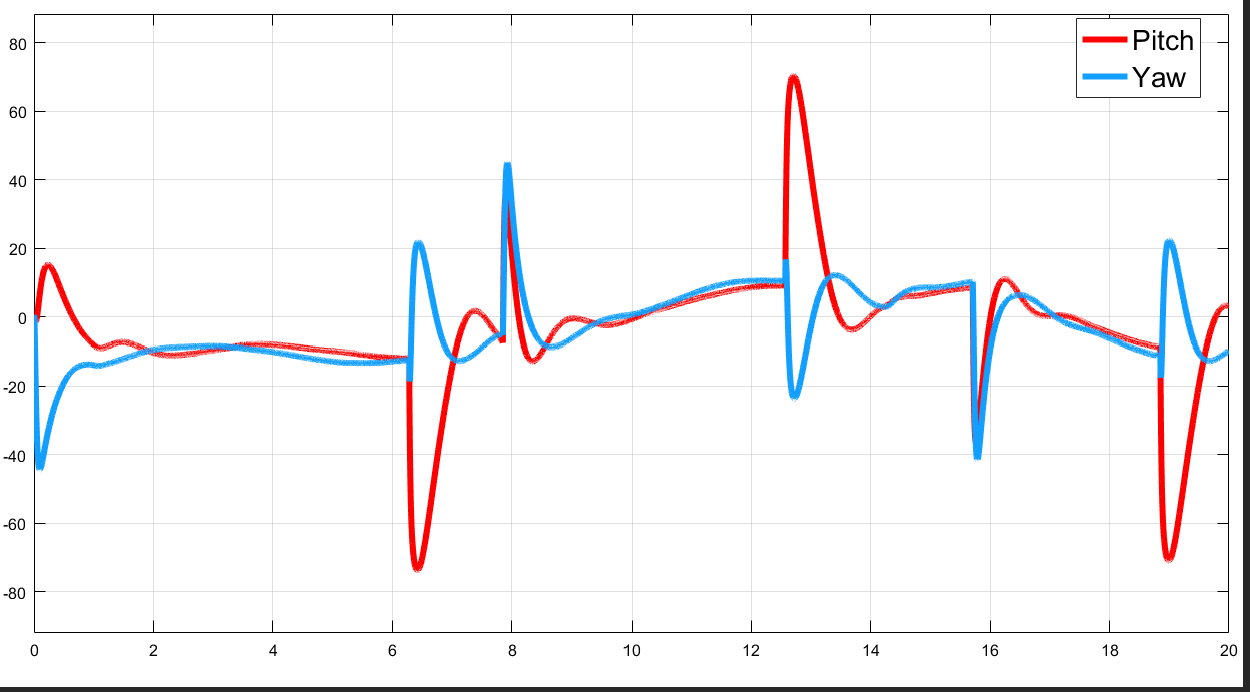


Figure . Pitch and yaw voltages for the μ synthesis controller implemented on the Quanser Aero for the nominal plant.

1. Lastly, we ran the controller on the Quanser Aero for five samples of the uncertain plant. There was a fair amount of oscillation, with the system often overshooting its desired positions.

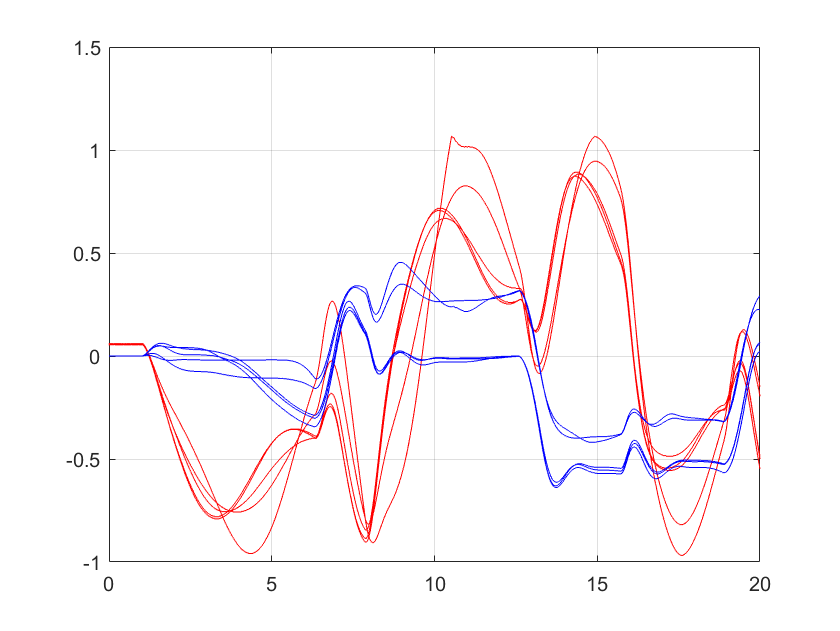


Figure . Pitch (red) and yaw (blue) angles for the μ synthesis controller run on the Quanser Aero for five samples of the uncertain plant.

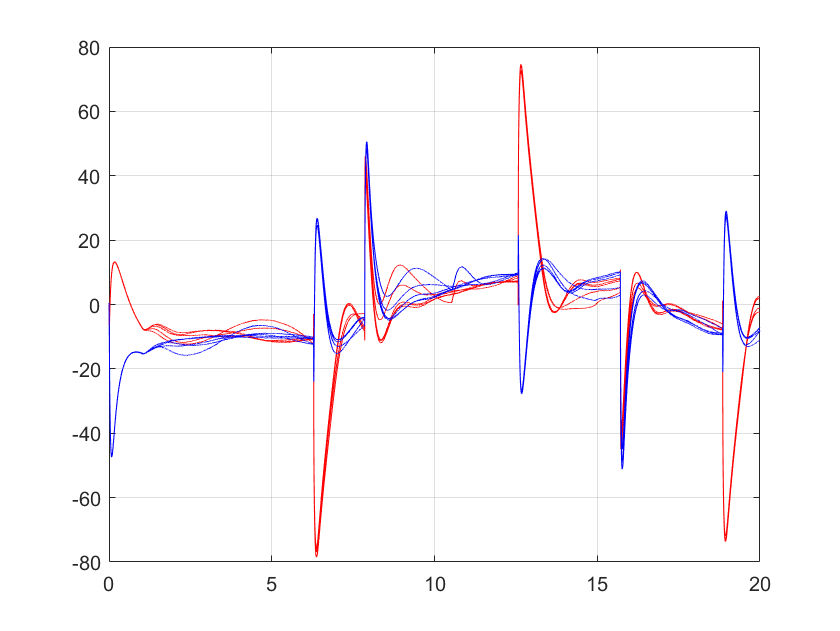


Figure . Pitch (red) and yaw (blue) voltages for the μ synthesis controller run on the Quanser Aero for five samples of the uncertain plant.

Video of the controller run on the nominal plant and five samples of the uncertain plant can be viewed here: <https://youtu.be/1y5eGrjnvVA>

H∞ Loop Shaping

Finally, we used H∞ loop shaping to generate a controller. The desired loopshape was the same employed during normal inverse loop shaping synthesis. We synthesized a controller and then reduced it to be sixth order, so that it matched the order of the H∞ optimal controller.

1. Below, you can see plots of the system response for the Hinf loop shaping controller. The yaw angle tracks well, however the pitch angle is slow to track the reference, so it is not able to converge.

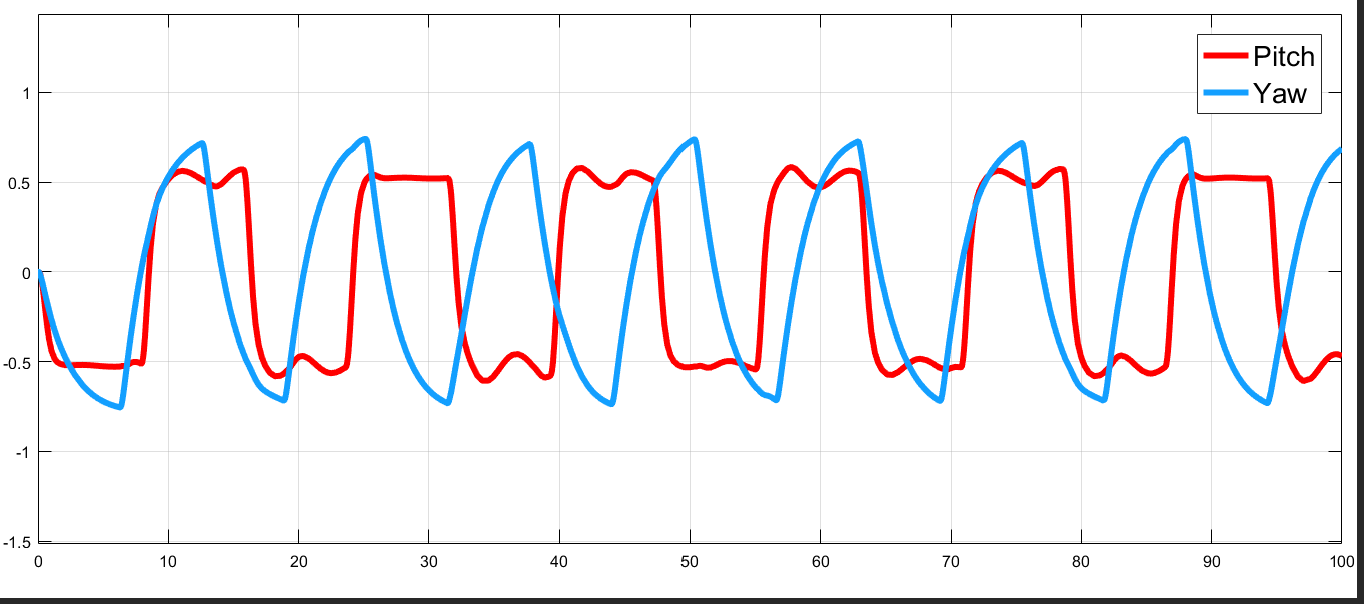


Figure . Pitch and yaw angles for H infinity loop shaping controller in simulation.

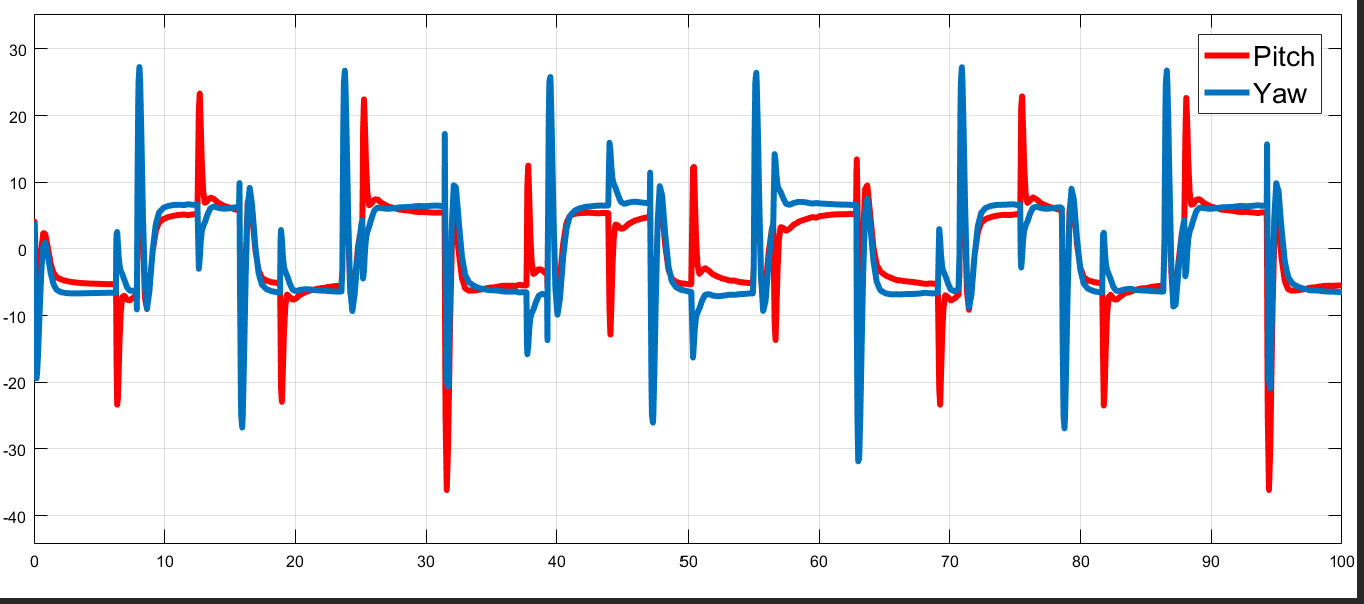


Figure . Pitch and yaw voltages for H infinity loop shaping controller in simulation.

1. Robust stability and robust performance could not be achieved with our design, as shown by the massive oscillation in the figure below. However, it should be noted that, that oscillation is the single one in 10 samples, meaning that that oscillation comes from an uncertainty with high variance from nominal value. Even though the design does not have robust stability, it has lower bound around 0.9.

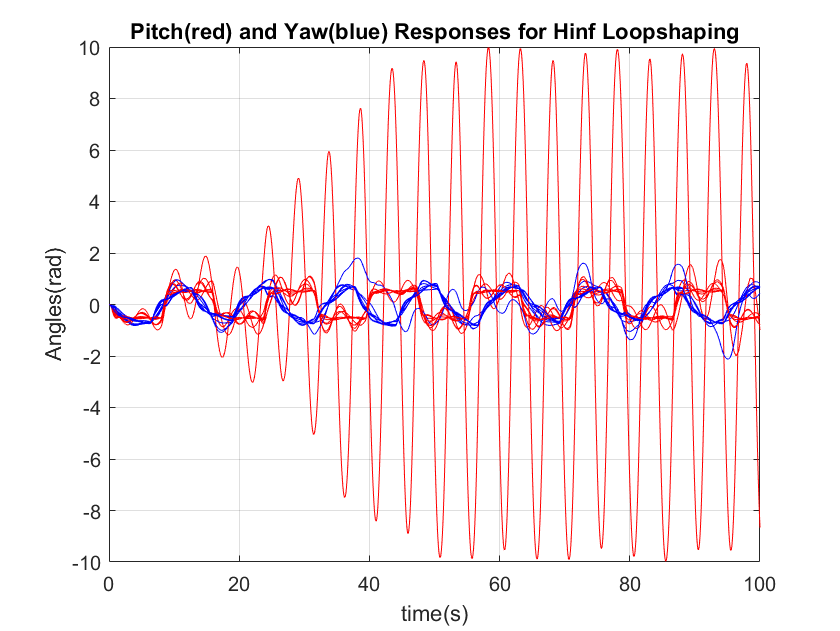


Figure . Pitch (red) and yaw (blue) angles for H infinity loop shaping controller on 10 samples of the uncertain plant.

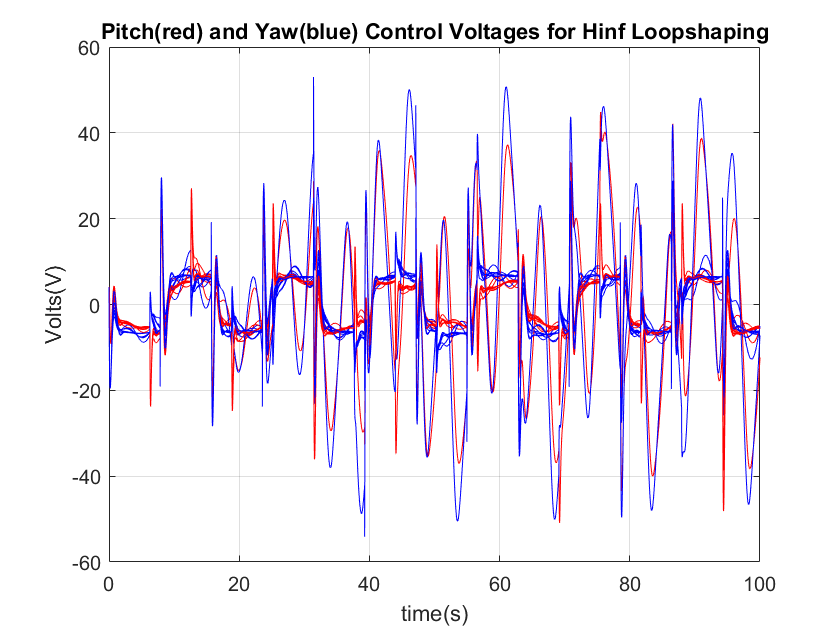


Figure . Pitch (red) and yaw (blue) angles for H infinity loop shaping controller on 10 samples of the uncertain plant.

1. Next, we implemented the controller on the Quanser Aero for the nominal plant. The pitch angle wobbled a fair amount, but the yaw was able to track reasonably well, except its magnitude. At this point we tried to check our loop shape and tune it for better performance, however it was extremely sensitive to variables such as extra pole location or cutoff frequency. With a slight change of the values, the system was going unstable, which led us to stick with this design.

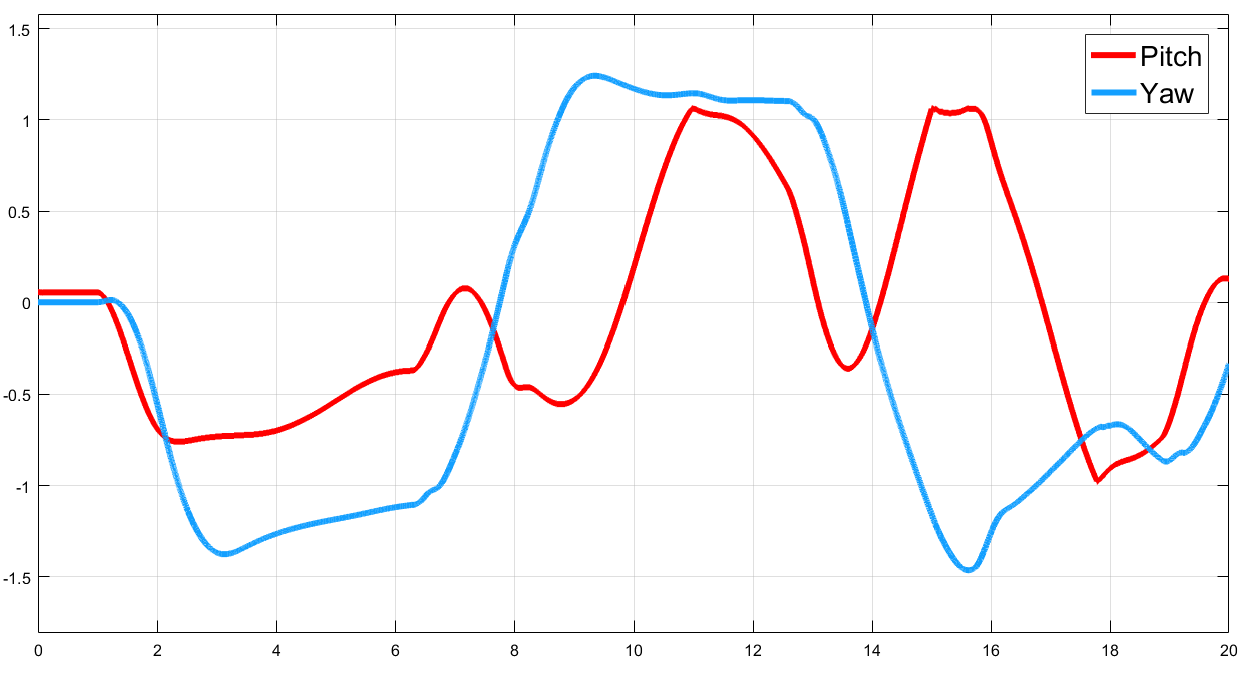


Figure . Pitch and yaw angles for the Hinf loop shaping controller implemented on the Quanser Aero for the nominal plant.

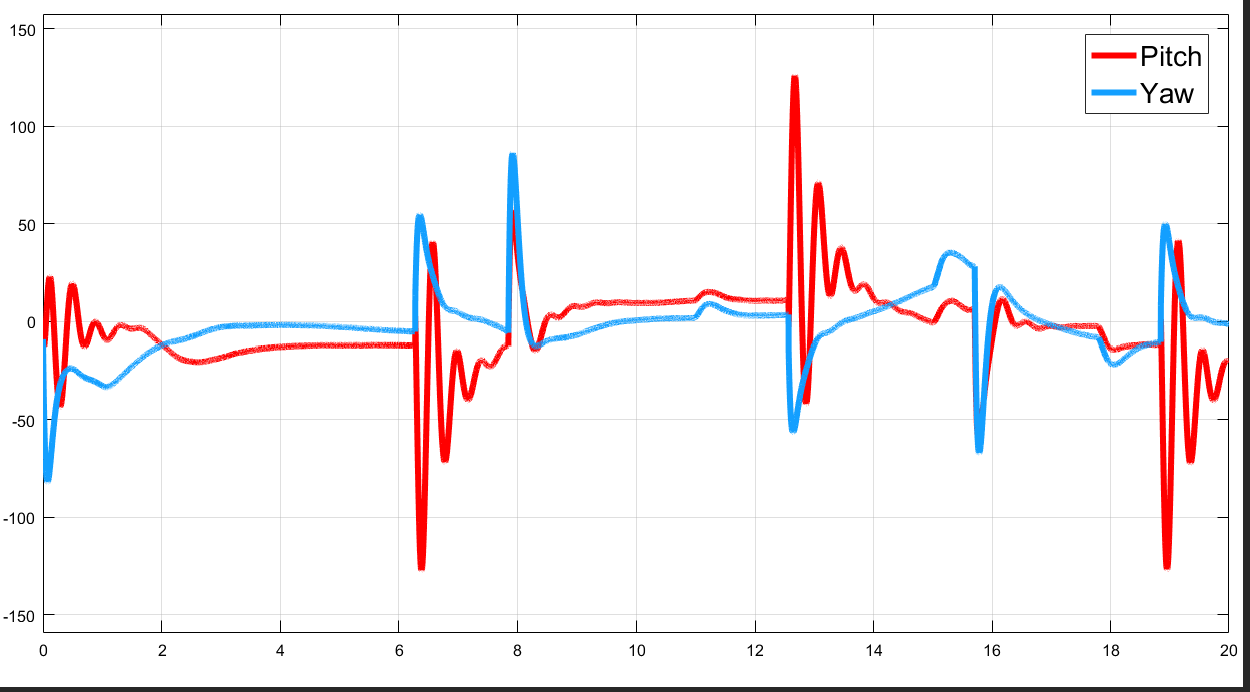


Figure . Pitch and yaw voltages for the Hinf loop shaping controller implemented on the Quanser Aero for the nominal plant.

1. Lastly, we ran the controller on five samples of the uncertain plant. Below are plots of all trials. Three trials performed very well, with minimal overshoot, but two had very large deviations in yaw angles.

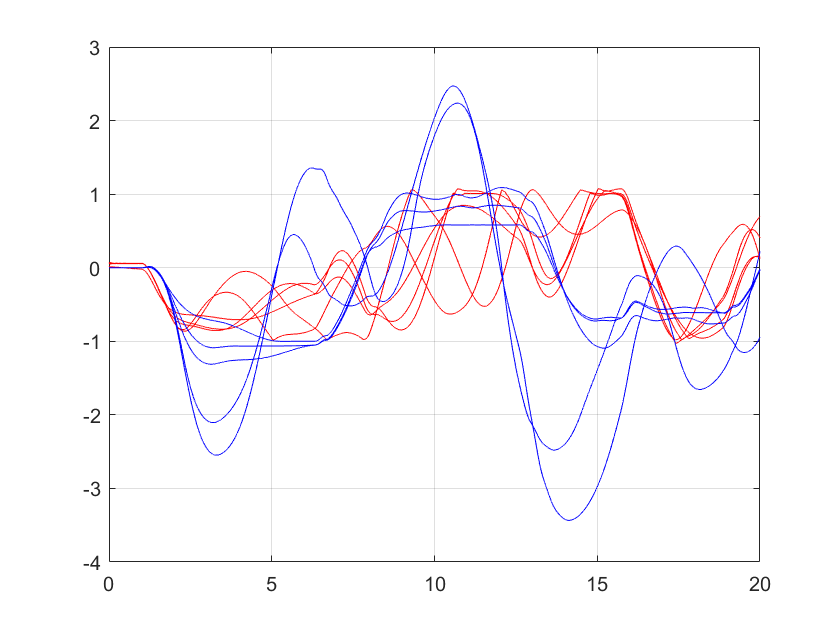


Figure . Pitch (red) and yaw (blue) angles for the Hinf loop shaping controller run on the Quanser Aero for five samples of the uncertain plant.

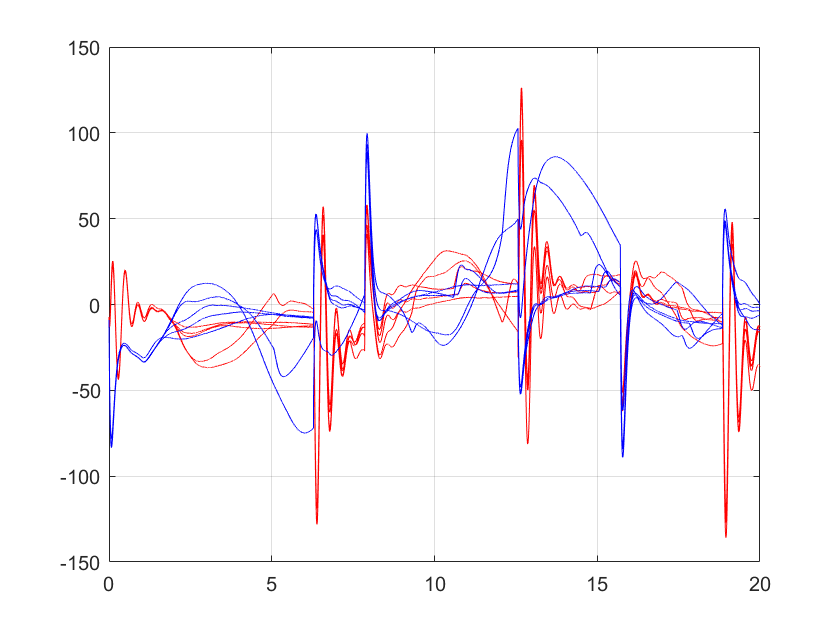


Figure . Pitch (red) and yaw (blue) voltages for the Hinf loop shaping controller run on the Quanser Aero for five samples of the uncertain plant.

Video of the controller run on the nominal plant and five samples of the uncertain plant can be viewed here: <https://youtu.be/0xQzf9-Uhno>

Conclusions

This lab explored the various controller generation techniques learned over the semester, implementing each on an uncertain DIDO system. While each technique was able to generate a nominally stable controller, they varied greatly in robust stability and performance.

The problem of using a DIDO system is hard because of the correlated relation of two states. The input in the pitch degree effects the movement in yaw, and vice versa. The biggest reason for this is the rotation of rotors effecting changing angular momentum of the system. The effect of coupling can be observed very clearly in our hardware plots. In all cases, our aim is to track two square signals, with different frequency. This results the need to make one state constant, and one to change at the same time instant. At each of our hardware plots, around 6.5 second, the yaw angle starts to go towards positive direction. Because of coupling, the pitch is disturbed as well, which reduces total tracking capability greatly. In general, it can be deduced that effects of yaw (blue) on pitch(red) is more than pitch effect on red, thus we were able to track yaw better in all controllers.

In Simulink, all our controllers performed well on nominal plant in terms of reaching to reference in desired time. However, most couldn’t perform well on hardware, due to the differences and limitations of hardware. The Quanser is not capable of operating at high frequency, and gave error signals when we attempted to do so. The H infinity loop shaping controller went unstable on hardware initially despite tracking well in simulation. Furthermore, we were unable to set the Quanser to the exact same position at the beginning of each trial. We observed that the initial conditions of the system had a nontrivial effect on the system performance.

H2 and Hinf controllers did a better job compared to other controllers. The tracking of H2 was better than any other on hardware. On simulation Hinf controller had minimum error and it is the one closest to being robustly stable.

While controller performance was less than ideal, we got hands-on experience at applying robust control techniques to a real system. We learned the importance of tuning your controller, as well as the need to consider hardware limitations when designing a controller. Overall, this lab was an excellent opportunity to put theory into practice.

Work Distribution

Nearly all the work had done together. Yigit contributed more on Simulink and Catherine contributed more to report.

Appendix

Preliminaries

clear all

close all

clc

%% Load Parameters & State Space

run('quanser\_aero\_parameters.m')

run('quanser\_aero\_state\_space.m')

%% Find Poles & Zeros

s=tf('s')

G\_nom=C\*inv(s\*eye(4)-A.NominalValue)\*B.NominalValue+D

zeross=tzero(G\_nom);

poles=eig(G\_nom);

%% Uncertain Plant

G\_unc=C\*inv(s\*eye(4)-A)\*B+D

G\_vec=usample(G\_unc,50);

[usys,info]=ucover(G\_vec,G\_nom,[2,2]);

%bodemag(G\_vec,G\_nom\*(eye(2)+info.W1\*info.W2));

%Wi = info.W1\*info.W2

W\_I\_Pitch=ss(info.W1.A(1:2,1:2),info.W1.B(1:2,1),info.W1.C(1,1:2), info.W1.D(1,1))

W\_I\_Yaw=ss(info.W1.A(3:4,3:4),info.W1.B(3:4,2),info.W1.C(2,3:4),info.W1.D(2,2))

Loopshape

preliminaries % load uncertain plant

%% Loop Shaping

s = tf('s');

wc = 5;

Ld = eye(2)\*(wc/s); %desired loopshape

A\_nom=A.NominalValue

B\_nom=B.NominalValue

K=0.5\*1/((s/30+1)^2)\*inv(G\_nom)\*(Ld);

L=G\_nom\*K;

%K=[(1711\*s+4910)/(s+50) (-1557\*s-5153)/(s+50)

% (2432\*s+7817)/(s+50) (921.5\*s+3308)/(s+50)];

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1a=usample(D1,10);

Delta\_2a=usample(D2,10);

%ncfsyn gkinverse

%%

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=angles.Time(:,1)

pitch=angles.Data(:,1)

yaw=angles.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Angles(rad)')

title('Pitch(red) and Yaw(blue) Responses for Inverse Loopshaping')

Delta\_1=Delta\_1a(:,:,1,1)

Delta\_2=Delta\_2a(:,:,1,1)

%%

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=mv.Time(:,1)

pitch=mv.Data(:,1)

yaw=mv.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Motor Voltages(V)')

title('Pitch(red) and Yaw(blue) Voltages')

H2syn

%Solves the H2 optimal control problem for the Quanser Aero, neglecting

%uncertainty

preliminaries

G = G\_nom;

J\_body

%% Lets not run preliminaries every time :)

G = G\_nom;J\_body

wc = 10; %crossover frequency needed for performance weight

Wu = 1/25\*eye(2); %control weight

Wp = makeweight(100, wc, 1/3)\*eye(2); %performance weight

P = augw(G, Wp, Wu, []);

[K,CL,GAM] = h2syn(P, 2, 2)

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1a=usample(D1);

Delta\_2a=usample(D2);

Delta\_1=Delta\_1a(:,:,1,1)

Delta\_2=Delta\_2a(:,:,1,1)

%% Plotting

bodemag(K)

CL=feedback(G\_unc\*K,eye(2))

stabmarg=robstab(CL)

mu=1/stabmarg.LowerBound

perfmargin=robustperf(CL)

%% Another Way to check

[STABMARG,DESTABUNC,REPORT,INFO] = robuststab(Sunc2); %NOTE: This DID NOT work with robstab!!!

mu2 = 1/STABMARG.LowerBound

% %Check RP + RS

S2 = eye(2)-feedback(G\_unc\*K,eye(2));

bodemag(S2,inv(Wp))

[STABMARG,DESTABUNC,REPORT,INFO] = robuststab(S2)

Hinfsyn

preliminaries

%% Hinf Optimal

%Wt = info.W1; %Not sure what sensitivity weight should be

Wu = 1/25\*eye(2); %control weight

Wt=[]

w\_max = 100;

w\_min = 0;

w\_try = (w\_max+w\_min)/2;

tol = .01;

%maximize wc

while(w\_max-w\_min > tol)

Wp = makeweight(100, w\_try, 1/10)\*eye(2); %performance weight

P = augw(G\_nom, Wp, Wu, Wt);

[Kinf,CL,GAM] = hinfsyn(P,2,2);

if GAM > 1

w\_max = w\_try;

else

w\_min = w\_try;

end

w\_try = (w\_max + w\_min)/2;

end

w\_try

K=Kinf

%Check RP + RS

Sinf = eye(2)-feedback(G\_unc\*Kinf,eye(2));

bodemag(Sinf,inv(Wp))

[STABMARG,DESTABUNC,REPORT,INFO] = robuststab(Sinf)

%% RS +RP

CL=feedback(G\_unc\*K,eye(2))

stabmarg=robstab(CL)

mu=1/stabmarg.LowerBound

perfmargin=robustperf(CL)

%%

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1=usample(D1);

Delta\_2=usample(D2);

Musyn

preliminaries %run only first time

%% Mu syn

wc = 5;

Wu = 1/25\*eye(2);

Wp = makeweight(100, wc, 1/3)\*eye(2);

%% Yigit Side

Wt=[]

InputUnc=ultidyn('DD',[2 2])

Gpert = G\_nom\*(eye(2)+InputUnc\*info.W1);

systemnames = 'Gpert Wu Wp';

inputvar = '[r{2};u{2}]';

outputvar = '[Wp;Wu;-r-Gpert]';

input\_to\_Gpert = '[u]';

input\_to\_Wu = '[u]';

input\_to\_Wp = '[r+Gpert]';

cleanupsysic = 'yes';

P = sysic;

[k,clp,bnd] = dksyn(P,2,2);

K=minreal(balred(k,6)\*(s/4785.4+1)/(s/200+1));

%%

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1=usample(D1,1);

Delta\_2=usample(D2,1);

%%

Delta\_1a=usample(D1,10);

Delta\_2a=usample(D2,10);

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=angles.Time(:,1)

pitch=angles.Data(:,1)

yaw=angles.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Angles(rad)')

title('Pitch(red) and Yaw(blue) Responses for Mu Synthesis')

%%

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=mv.Time(:,1)

pitch=mv.Data(:,1)

yaw=mv.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Volts(V)')

title('Pitch(red) and Yaw(blue) Control Voltages for Mu Synthesis')

Hinfloopsyn

preliminaries

s = tf('s');

wc = 3;

Ld = eye(2)\*(wc/s); %desired loopshape

K\_l=minreal((1/(s/10+1))^2\*(Ld));

[Ki,cl,gam,info] =ncfsyn(G\_unc,inv(G\_nom),K\_l);

Km=-Ki

K=balred(Km,6)

%%

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1=usample(D1,1);

Delta\_2=usample(D2,1);

%%

%Check robust stability

S = eye(2)-feedback(G\_unc\*K,eye(2));

bodemag(S,inv(Wp))

[STABMARG,DESTABUNC,REPORT,INFO] = robuststab(S)

%Check robust performance

[perfmarg,wcu,report,info] = robustperf(S)

%%

CL=feedback(G\_unc\*K,eye(2))

stabmarg=robstab(CL)

mu=1/stabmarg.LowerBound

perfmargin=robustperf(CL)

%%

%Generate deltas

D1=ultidyn('D1',[1 1]);

D2=ultidyn('D2',[1 1]);

Delta\_1a=usample(D1,10);

Delta\_2a=usample(D2,10);

%%

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=angles.Time(:,1)

pitch=angles.Data(:,1)

yaw=angles.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Angles(rad)')

title('Pitch(red) and Yaw(blue) Responses for Hinf Loopshaping')

%%

figure()

for i=1:10

Delta\_1=Delta\_1a(:,:,i,1)

Delta\_2=Delta\_2a(:,:,i,1)

simout=sim('prelim\_plant')

time=mv.Time(:,1)

pitch=mv.Data(:,1)

yaw=mv.Data(:,2)

plot(time,pitch,'r')

hold on

plot(time,yaw,'b')

end

xlabel('time(s)')

ylabel('Volts(V)')

title('Pitch(red) and Yaw(blue) Control Voltages for Hinf Loopshaping')