John Clouse ASEN 5014 Final project

Table of Contents

Init, sail force analysis 1
Set up the first controller 2
Observer in loop 4
Controller 1 + observer 4
Controller 1 + Observer with error 4
LQR 5
LQR, observer error 6
Print out the diff between the two controllers' observer errors 6
Monte Carlo Analysis 6
Model verification 7

Init, sail force analysis

```
close all
title_plots = 0; %zero for report figures.
lw = 1; % Line Weight
fs = 20; % Font Size
SysModel

if 0
alpha_range = 0:0.01:pi/2;
Ft = P*A_sail*(1-rho_s)*cos(alpha_range).*sin(alpha_range);
Fn = P*A_sail*((1+rho_s)*cos(alpha_range).*cos(alpha_range)...
    +2/3*rho_d*cos(alpha_range));
figure
plot(alpha_range*180/pi,Ft,'LineWidth',2)
hold on
plot(alpha_range*180/pi,Ft_max*alpha_range,'g','LineWidth',2)
for ii = 1:length(alpha_range)
    if abs((Ft(ii) - Ft_max*alpha_range(ii))/Ft(ii)) > 0.05
        x_five_percent = ii;
        break
    end
end
plot([x_five_percent x_five_percent],...
    [max(Ft_max*alpha_range) min(Ft_max*alpha_range)],'r--','LineWidth',2)
set(gca, 'FontSize', 20)
% title('F_{T} Solution', 'FontSize', 24)
legend('Actual', 'Linearized', '5% Difference')
xlabel('{\alpha} (degrees)', 'FontSize', 24)
ylabel('F_{T} (N)', 'FontSize', 24)

figure
plot(alpha_range*180/pi,Fn,'LineWidth',2)
```

```
hold on
plot(alpha_range*180/pi,Fn_max-0*alpha_range,'g','LineWidth',2)
for ii = 1:length(alpha_range)
    if abs((Fn(ii) - Fn_max-0*alpha_range(ii))/Fn(ii)) > 0.05
        x_five_percent = ii;
        break
    end
end
plot([x_five_percent x_five_percent],...
    [max(Fn) min(Fn)],'r--','LineWidth',2)
set(gca, 'FontSize', 20)
% title('F_{N} Solution', 'FontSize', 24)
legend('Actual', 'Linearized', '5% Difference')
xlabel('{\alpha} (degrees)', 'FontSize', 24)
ylabel('F_{N} (N)', 'FontSize', 24)
end

if 1
system = ss(A,B,[1 0 0 0;0 0 1 0],0);
figure
step(system,stepDataOptions('StepAmplitude',0.1));
end
```

Set up the first controller

```
close all
figWidth = 1120; % pixels
figHeight = 840; % pixels
r2d = 180/pi;


% The design parameters
PO_desired = 10/100;
PS_desired = 5/100;
PO = 9/100;
PS = 4/100; %Settle percentage
Ts = 3600*1.5;%1200;

t = 0:0.01:Ts*3;

% Get the desired dominant poles with SISO equations.
damp_times_wn = -log(PS)/Ts
damping_ratio = -log(PO)/sqrt(pi*pi+(log(PO))^2);
wn = damp_times_wn/damping_ratio;
wd = wn*sqrt(1-damping_ratio^2);

real_offset = -.1;% use this to tune the other poles
P = [complex(-damp_times_wn+real_offset, wd) ...
    complex(-damp_times_wn+real_offset, -wd) ...
    complex(-damp_times_wn, wd) complex(-damp_times_wn, -wd)];
K = place(A,B,P);

% Try out feedforward (not used)
```

```
F = inv(C*inv(-A+B*K)*B);
A_CL = A-B*K;
B_CL = B*F;
CL_system = ss(A_CL, B_CL, eye(4),0);
OL_system = ss(A,B,C,0);

% Integral control
A_OL_Aug = [A,zeros(4,1);-C, zeros(1)];
B_OL_Aug = [B;zeros(1)];
P_Aug = [-100,P];
K_Aug = place(A_OL_Aug,B_OL_Aug,P_Aug);
K = K_Aug(1:4); % gain for the nominal states
KI = K_Aug(5); % Integral gain
A_CL_Aug = [A-B*K, -B*KI; -C, zeros(1)];
B_CL_Aug = [zeros(4,1);eye(1)];
Int_sys = ss(A_CL_Aug, B_CL_Aug, [C 0], 0);

% Open-loop: stable response
r = 35*pi/180;
analysis_set = 'OL_Stable';
figure('Position', [0, 0, figWidth, figHeight]);
OL_t = 0:0.01:3600*5;
y_ol = lsim(OL_system,repmat(0.0001,1,length(OL_t)),OL_t);
plot(OL_t/3600,y_ol*r2d,'LineWidth',lw)
set(gca, 'FontSize', fs)
if title_plots
title(...
    sprintf('OL lsim Results: Step reference at %.1f degrees', r*180/pi));
end
ylabel('\alpha (deg)','FontSize',fs)
xlabel('Time (hr)','FontSize',fs)
print(['Report/' analysis_set],'-dpng')

% Open-loop: unstable response
analysis_set = 'OL_Unstable';
figure('Position', [0, 0, figWidth, figHeight]);
OL_t = 0:0.01:1800;
y_ol = lsim(OL_system,repmat(0.01,1,length(OL_t)),OL_t);
plot(OL_t,y_ol*r2d,'LineWidth',lw)
set(gca, 'FontSize', fs)
if title_plots
title(...
    sprintf('OL lsim Results: Step reference at %.1f degrees', r*180/pi));
end
ylabel('\alpha (deg)','FontSize',fs)
xlabel('Time (sec)','FontSize',fs)
print(['Report/' analysis_set],'-dpng')

% Feedforward controller (ended up opting for integral control)
figure
lsim(CL_system,repmat(r,1,length(t)),t)
title(...
    sprintf('CL lsim Results: Step reference at %.1f degrees', r*180/pi));
y = lsim(CL_system,repmat(r,1,length(t)),t);
```

```
% Test out the integral controller
analysis_set = 'Ctrl1';
y_int = lsim(ss(A_CL_Aug, B_CL_Aug, eye(5), 0),repmat(r,1,length(t)),t);

plotSailSysResp( analysis_set,y_int,t,K_Aug,r,Ts,600,title_plots )
```

Observer in loop

```
L = place(A',C',[-5,-6,-7,-8]*1e-3)';

A_Obs_Aug = [A_OL_Aug-B_OL_Aug*K_Aug,B_OL_Aug*K_Aug(1:4);
    zeros(4,5),A-L*C];
B_Obs_Aug = [zeros(size(B));1;zeros(length(L),1)];
C_Obs_Aug = [C, 0, zeros(1,length(L))];

rank(ctrb(A',C'));

C_Obs_AugFake = [eye(9)];
Obs_system = ss(A_Obs_Aug, B_Obs_Aug, C_Obs_AugFake, 0);
```

Controller 1 + observer

```
r = 35*pi/180;
analysis_set = 'Ctrl1Obs';
y_obs=lsim(Obs_system,repmat(r,1,length(t)),t);

plotSailSysResp( analysis_set,y_obs(:,1:5),t,K_Aug,r,Ts,150,title_plots )
figure('Position', [0, 0, figWidth, figHeight]);
colors = {'b','g','r','k'};
for ii = 6:9
plot(t/3600,y_obs(:,ii)*r2d,colors{ii-5},'LineWidth',lw)
hold on
end
set(gca, 'FontSize', 20)
xlabel('Time (hr)','FontSize',fs)
ylabel('Error','FontSize',fs)
l = legend({'\alpha error (deg)', '\alpha rate error (deg/s)', ...
    '\delta error (deg)', '\delta rate error (deg/s)'},'FontSize',fs);
print(['Report/' analysis_set '_ObsErr'],'-dpng')
```

Controller 1 + Observer with error

r = 0;

```
r = 35*pi/180;
analysis_set = 'Ctrl1ObsError';
sensor_error = .05*pi/180;
y_obs_error = ...
    lsim(Obs_system,repmat(r,1,length(t)),t,[0,0,0,0,0,sensor_error,0, 0, 0]);
```

```
plotSailSysResp( analysis_set,y_obs_error(:,1:5),t,K_Aug,r,Ts,3600,title_plots )

figure('Position', [0, 0, figWidth, figHeight]);
colors = {'b','g','r','k'};
for ii = 6:9
plot(t/3600,y_obs_error(:,ii)*r2d,colors{ii-5},'LineWidth',lw)
hold on
end
set(gca, 'FontSize', 20)
xlabel('Time (hr)','FontSize',fs)
ylabel('Error','FontSize',fs)
l = legend({'\alpha error (deg)', '\alpha rate error (deg/s)', ...
    '\delta error (deg)', '\delta rate error (deg/s)'},'FontSize',fs);
print(['Report/' analysis_set '_ObsErr'],'-dpng')
% set(l,'FontSize',12)
```

LQR

```
Q_wts = [1,1,10000,1,1];
Q_wts = Q_wts/sum(Q_wts);
state_max = [pi/2, 0.01, pi/6, 0.01, 0.01];
Q = diag(Q_wts.*Q_wts./(state_max.*state_max));
rho_R = 1000;
u_max = 100;
R = rho_R/u_max;
[K_LQR, W, E] = lqr(A_OL_Aug,B_OL_Aug,Q,R);

A_Obs_LQR = [A_OL_Aug-B_OL_Aug*K_LQR,B_OL_Aug*K_LQR(1:4);zeros(4,5),A-L*C];
B_Obs_LQR = [zeros(size(B));1;zeros(length(L),1)];
C_Obs_LQR = [C, 0, zeros(1,length(L))];
C_Obs_LQRFake = [eye(9)];
LQR_system = ss(A_Obs_LQR, B_Obs_LQR, C_Obs_LQRFake, 0);
eig(A_OL_Aug-B_OL_Aug*K_LQR); % Eigenvals of this CL system

r = 35*pi/180;
analysis_set = 'CtrlLqrObs';
y_lqr = lsim(LQR_system,repmat(r,1,length(t)),t);

plotSailSysResp( analysis_set,y_lqr(:,1:5),t,K_LQR,r,Ts,3600,title_plots )

% Observer error plot
figure('Position', [0, 0, figWidth, figHeight]);
colors = {'b','g','r','k'};
for ii = 6:9
plot(t/3600,y_lqr(:,ii)*r2d,colors{ii-5},'LineWidth',lw)
hold on
end
set(gca, 'FontSize', 20)
xlabel('Time (hr)','FontSize',fs)
ylabel('Error','FontSize',fs)
l = legend({'\alpha error (deg)', '\alpha rate error (deg/s)', ...
    '\delta error (deg)', '\delta rate error (deg/s)'},'FontSize',fs);
print(['Report/' analysis_set '_ObsErr'],'-dpng')
```

LQR, observer error

r=0;

```
r = 35*pi/180;
sensor_error = .05*pi/180;
analysis_set = 'CtrlLqrObsError';
y_lqr_error = lsim(LQR_system,repmat(r,1,length(t)),t,...
    [0,0,0,0,0,sensor_error,0, 0, 0]);

plotSailSysResp( analysis_set,y_lqr_error(:,1:5),t,K_LQR,r,Ts,3600,title_plots )

% Observer error plot
figure('Position', [0, 0, figWidth, figHeight]);
colors = {'b','g','r','k'};
for ii = 6:9
plot(t/3600,y_lqr_error(:,ii)*r2d,colors{ii-5},'LineWidth',lw)
hold on
end
set(gca, 'FontSize', 20)
xlabel('Time (hr)','FontSize',fs)
ylabel('Error','FontSize',fs)
l = legend({'\alpha error (deg)', '\alpha rate error (deg/s)', ...
    '\delta error (deg)', '\delta rate error (deg/s)'},'FontSize',fs);
print(['Report/' analysis_set '_ObsErr'],'-dpng')
```

Print out the diff between the two controllers' observer errors

```
analysis_set = 'Ctrl1_LQR_error_diff';
figure('Position', [0, 0, figWidth, figHeight]);
colors = {'b','g','r','k'};
for ii = 6:9
plot(t/3600,(y_lqr_error(:,ii)-y_obs_error(:,ii))*r2d,colors{ii-5},'LineWidth',lw)
hold on
end
set(gca, 'FontSize', 20)
xlabel('Time (hr)','FontSize',fs)
ylabel('Difference in Error','FontSize',fs)
l = legend({'\Delta \alpha error (deg)', '\Delta \alpha rate error (deg/s)', ...
    '\Delta \delta error (deg)', '\Delta \delta rate error (deg/s)'},...
    'FontSize',fs);
print(['Report/' analysis_set '_ObsErr'],'-dpng')
```

Monte Carlo Analysis

Disturbance solar torque

```
if 0
MC;
end
```

Model verification

A little ODE45 verification of the system. If the gimbal torque holds the boom still, there should be an oscillation of \alpha Anon fcn to compute the required torque to hold the boom still wrt the sail

```
torque_hold = @(X,A,B) -dot(A(4,:),X)/B(4);

% Anon fcn for state integration.
state_dot = @(t,X) A*X + B*torque_hold(X,A,B);

[t_out, X_out] = ode45(state_dot,[0 3600*12],[0;0;5*pi/180;0]);
T = [];
for ii = 1:length(X_out)
    T(ii) = torque_hold(X_out(ii,:)',A,B);
end
figure
plot(t_out, X_out(:,1))
hold on
plot(t_out, X_out(:,3),'r')
ylabel('Angle (rad)')
xlabel('Time (sec)')
legend('\alpha','\delta')

figure
plot(t_out, T)
ylabel('Gimbal Torque (N-m)')
xlabel('Time (sec)')
```

Published with MATLAB® R2013b

SysModel

Table of Contents

Constants 1
Set up ss model 1

Set up model

Constants

```
L_boom = 30;
h = L_boom*sin(pi/4)*2;
m_s = 40;
I = m_s*h*h/12
% I = I + ms*2.5^2

m_p = 116; %kg
m = m_p+m_s;
J_s = I; %kg*m2
J_p = 20; %kg*m2
r = 0.88;
s = 0.94;
Bf = 0.79;
Bb = 0.55;
ef = 0.05;
eb=0.55;
rho_s = r*s;
rho_d = (Bf*r*(1-s)+(ef*Bf-eb*Bb)/(ef+eb))*3/2;
P = 4.563e-6; %N/m2
A_sail = h*h;
l = 2; %m
d = l*m_p/m;
b = 0.5; %m
```

Set up ss model

```
Ft_max = P*A_sail*(1-rho_s);
Fn_max = P*A_sail*(1+rho_s+2/3*rho_d);

a_dd_LHS_1 = J_s + m_s*m_p/m*b*(b+l);
a_dd_LHS_2 = J_p + m_s*m_p/m*l*(b+l);
BIG = a_dd_LHS_2/a_dd_LHS_1;
d_dd_LHS = J_p + m_s*m_p/m*l*(l-b*BIG);

dd_da = m_p/m*(-l+b*BIG)*Ft_max/d_dd_LHS;
dd_dd = -m_p/m*l*Fn_max/d_dd_LHS;

da_da = (-m_p/m*b*Ft_max - m_s*m_p/m*b*l*dd_da)/a_dd_LHS_1;
da_dd = -m_s*m_p/m*b*l*dd_dd/a_dd_LHS_1;
```

```
A = [0 1 0 0;
    da_da 0 da_dd 0;
    0 0 0 1;
    dd_da 0 dd_dd 0];

B = [0; -1/a_dd_LHS_1; 0; (1+BIG)/d_dd_LHS];

C = [1 0 0 0];


P = ctrb(A,B);
fprintf('Rank of P: %d\n',rank(P))
O = obsv(A,C);
fprintf('Rank of O: %d\n',rank(O))

[Vecs, Eigs] = eig(A);
```

Published with MATLAB® R2013b

Monte Carlo Analysis

Table of Contents

Disturbance solar torque 1
LQR update - Make it meet the gimbal bounds with the disturbance 2

Disturbance solar torque

```
analysis_set = 'MonteCarlo';
r = 35*pi/180;
offset_max = 0.1;
offset_min = -offset_max;
t = 0:1:Ts*3;

num_runs = 100;
y_int_mc_store = [];
y_lqr_mc_store = [];

for ii = 1:num_runs
    fprintf('Run %d, ',ii)
    offset = offset_min + (offset_max-offset_min)*rand(1);
    fprintf('Offset %.4f\r',offset)
    dist_torque_max = offset*Fn_max;
    A_dist = A;
    % The disturbance will manifest itself as an input torque
    B_dist = [0; 1/a_dd_LHS_1; 0; -BIG/d_dd_LHS;zeros(5,1)];

    A_OL_Aug_MC = [A_dist,zeros(4,1);-C, zeros(1)];
    A_CL_Aug_MC = [A_dist-B*K, -B*KI; -C, zeros(1)];
    A_Obs_Aug_MC = [A_OL_Aug_MC-B_OL_Aug*K_Aug,B_OL_Aug*K_Aug(1:4);
        zeros(4,5),A_dist-L*C];
    Int_sys_MC = ss(A_Obs_Aug_MC, [B_Obs_Aug,B_dist], C_Obs_AugFake, 0);

    A_Obs_LQR_MC = ...
        [A_OL_Aug_MC-B_OL_Aug*K_LQR,B_OL_Aug*K_LQR(1:4);...
         zeros(4,5),A_dist-L*C];
    LQR_system_MC = ss(A_Obs_LQR_MC, [B_Obs_LQR,B_dist], C_Obs_LQRFake, 0);

    y_int_mc = lsim(Int_sys_MC,repmat([r;dist_torque_max],1,length(t)),t);
    y_lqr_mc = lsim(LQR_system_MC,repmat([r;dist_torque_max],1,length(t)),t);
%     y_int_mc = lsim(Int_sys_MC,repmat(r,1,length(t)),t);
%     y_lqr_mc = lsim(LQR_system_MC,repmat(r,1,length(t)),t);

    y_int_mc_store(:,:,ii) = y_int_mc;
    y_lqr_mc_store(:,:,ii) = y_lqr_mc;
end
fprintf('\n')
analysis_set = 'LQRMonteCarlo';
```

```
plotSailSysResp( analysis_set,y_lqr_mc_store(:,1:5,:),t,K_LQR,r,Ts,3600*2,...
    title_plots)
analysis_set = 'Ctrl1MonteCarlo';
plotSailSysResp( analysis_set,y_int_mc_store(:,1:5,:),t,K_Aug,r,Ts,3600*2,...
    title_plots)
t = 0:0.01:Ts*3;
```

LQR update - Make it meet the gimbal bounds with the disturbance

```
Q_wts = [1,1,11000,1,1];
Q_wts = Q_wts/sum(Q_wts);
state_max = [pi/2, 0.01, pi/6, 0.01, 0.01];
Q = diag(Q_wts.*Q_wts./(state_max.*state_max));
rho_R = 1000;
u_max = 100;
R = rho_R/u_max;
[K_LQR, W, E] = lqr(A_OL_Aug,B_OL_Aug,Q,R);

A_Obs_LQR = [A_OL_Aug-B_OL_Aug*K_LQR,B_OL_Aug*K_LQR(1:4);zeros(4,5),A-L*C];
B_Obs_LQR = [zeros(size(B));1;zeros(length(L),1)];
B_dist = [0; 1/a_dd_LHS_1; 0; -BIG/d_dd_LHS;zeros(5,1)];
C_Obs_LQR = [C, 0, zeros(1,length(L))];
C_Obs_LQRFake = [eye(9)];
y_lqr_mc_store = [];
t = 0:1:Ts*3;
for ii = 1:num_runs
    fprintf('Run %d, ',ii)
    offset = offset_min + (offset_max-offset_min)*rand(1);
    fprintf('Offset %.4f\r',offset)
    dist_torque_max = offset*Fn_max;

    LQR_system = ss(A_Obs_LQR, [B_Obs_LQR,B_dist], C_Obs_LQRFake, 0);
    y_lqr_mc = lsim(LQR_system,repmat([r;dist_torque_max],1,length(t)),t);

    y_lqr_mc_store(:,:,ii) = y_lqr_mc;
end
analysis_set = 'LQR2MonteCarlo';
plotSailSysResp( analysis_set,y_lqr_mc_store(:,1:5,:),t,K_LQR,r,Ts,3600*2,...
    title_plots)
t = 0:0.01:Ts*3;
```

Published with MATLAB® R2013b

```
function  plotSailSysResp( analysis_set,y,t,K,r,Ts,torque_tmax, use_title )
%plotSailSysResp All the controller plots


y_size = size(y);
if length(y_size) == 2
    num_y = 1;
elseif length(y_size) == 3
    num_y = y_size(end);
end

figWidth = 1120; % pixels
figHeight = 840; % pixels
r2d = 180/pi;
gimbal_angle_lim = pi/6*r2d;
lw = 1;
fs = 20;

% Coning angle plot
ctrl1Obs_Alpha_plot = figure('Position', [0, 0, figWidth, figHeight]);

for jj = 1:num_y
h_plot = plot(t/3600,y(:,1,jj)*r2d,'LineWidth',lw);
hold on
end
ylabel('\alpha (deg)', 'FontSize', fs)
xlabel('Time (hr)', 'FontSize', fs)
if use_title
title(...
    [sprintf('Sun Angle, Step Reference at %.1f degrees', r*180/pi)...
    ', ' analysis_set], 'FontSize', fs);
end
hold on
h_Ts = plot([Ts Ts]/3600,[0, r*1.10]*r2d,'r','LineWidth',lw);
h_OS = plot([t(1),t(end)]/3600,[1 1]*r*1.1*r2d,'r-.','LineWidth',lw);
h_hi = plot([t(1),t(end)]/3600,[1 1]*r*1.05*r2d,'r--','LineWidth',lw);
h_lo = plot([t(1),t(end)]/3600,[1 1]*r*.95*r2d,'r--','LineWidth',lw);
set(gca, 'FontSize', fs)
pl = legend([h_plot h_Ts h_OS h_hi],...
    '\alpha','T_{settle}','10% Overshoot', '5% Settling',...
    'Location','SouthEast');
set(pl, 'FontSize', fs)
print(['Report/' analysis_set '_Alpha'],'-dpng')

% Control torque
ctrl1Obs_Torque_plot = figure('Position', [0, 0, figWidth, figHeight]);
for jj = 1:num_y
u = [];
for ii = 1:length(t)
    u(ii) = r-K*y(ii,:,jj)';
end
plot(t(t<torque_tmax),u(t<torque_tmax),'LineWidth',lw)
```

```
hold on
end
if use_title
title([sprintf('Gimbal Torque, %.0f-degree Step',r*180/pi)...
    ', ' analysis_set], 'FontSize', fs)
end
set(gca, 'FontSize', fs)
ylabel('T_g (N-m)', 'FontSize', fs)
xlabel('Time (s)', 'FontSize', fs)
print(['Report/' analysis_set '_Torque'],'-dpng')

%Gimbal angle plot
ctrl1Obs_GimbalAng_plot = figure('Position', [0, 0, figWidth, figHeight]);
for jj = 1:num_y
h_plot = plot(t/3600,y(:,3,jj)*r2d,'LineWidth',lw);
hold on
end
h_hi = plot([t(1) t(end)]/3600,[1 1]*gimbal_angle_lim, 'r--','LineWidth',lw);
h_lo = plot([t(1) t(end)]/3600,[1 1]*-gimbal_angle_lim, 'r--','LineWidth',lw);
if use_title
title([sprintf('Gimbal Angle, %.0f-degree Step',r*180/pi)...
    ', ' analysis_set], 'FontSize', fs)
end
set(gca, 'FontSize', fs)
ylabel('\delta (deg)', 'FontSize', fs)
ylim([-gimbal_angle_lim-1, gimbal_angle_lim+1])
xlabel('Time (hr)', 'FontSize', fs)
pl = legend([h_plot, h_hi], '\delta','Actuator Limit',...
    'Location','SouthEast');
set(pl, 'FontSize', fs)
print(['Report/' analysis_set '_Delta'],'-dpng')

end
```

Published with MATLAB® R2013b