Multi-Rotor Aircraft Analysis

Riley Kenyon

December 14th, 2019

Abstract

This report describes the analysis of a multi-rotor aircraft and the design of a state feedback system and state observer. Measures of interest are controllability, stabilizability, observability, and detectibility.

1 Topic Discussion

1.1 Introduction

The system chosen to analyze is a multi-rotor aircraft, specifically a quadcopter. The aircraft has four rotors that provide motion in three dimension. Each opposing pair of rotors spin in the same direction, and adjacent rotors spin in the opposite direction. The model derived for this quadcopter assumes near level flight with the rotor's normal all tilted towards the center of the aircraft. This is assumed in order to have direct access to X, Y movement without pitching the quadcopter. For simplicity, it is also assumed an inner control loop is implemented that keeps the reference frame of the vehicle aligned with the coordinate axes.

1.2 State Space Representation

The state variables used in the model are

$$\mathbf{x} = [x \quad \dot{x} \quad y \quad \dot{y} \quad z \quad \dot{z}] \tag{1}$$

where the states describe the position and velocity in X, Y, Z. The four rotor inputs can describe the force felt in each coordinate direction by

$$F_X = \delta(T_3 - T_1) \tag{2}$$

$$F_Y = \delta(T_4 - T_2) \tag{3}$$

$$F_Z = \gamma(T_1 + T_2 + T_3 + T_4) \tag{4}$$

where T1 thorough T4 are the thrust produced by each rotor, and F_x, F_y, F_z are the respective coordinate forces. Drag is considered using vertical and horizontal coefficients ($k_{d,lat}$,$k_{d,vert}$) with mass m. Sensor data output is X, Y, Z from GPS. The general state-space representation is described by

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{5}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \tag{6}$$

The behavior desired from this system is a steady state position given a reference for the X,Y, and Z coordinates. This is what the GPS sensors were selected to provide. The specific matrices in this state-space model are

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -0.0104 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -0.0104 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -0.0208 \end{bmatrix} \tag{7}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -0.04167 & 0 & 0.04167 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -0.04167 & 0 & 0.04167 \\ 0 & 0 & 0 & 0 \\ 0.4 & 0.4 & 0.4 & 0.4 \end{bmatrix} \tag{8}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{9}$$

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

λ_1	0	m = 3
λ_2	-0.0104	m = 2
λ_3	-0.0208	m = 1

Table 1: Eigenvalues of matrix A describing the system

2 System Performance

The desired system performance is a rise time of 1 s for X, Y, and Z positioning. The other requirement is zero steady-state error to a step input, which will be addressed in the state feedback controller design. Rise time can be approximated as

$$t_r = 1.8/\omega_n \tag{11}$$

for a second order system. Using the performance measures to determine the closed loop poles of the system, the values needed to satisfy the constraints are

$$||\lambda_i|| > 1.8 \tag{12}$$

while λ_i should be negative to satisfy stability. As such, it is selected to have the pole locations of the closed loop transfer function be greater than -3 to account for the presence of the additional poles. The current poles of the system can be found by determining the eigenvalues of A by solving $DET(A - \lambda I) = 0$, the resulting values are found in table 1. Without a controller in place, the system is simulated to verify the initial performance and stability. For a step response to each individual rotor torque, the aircraft increases in Z exponentially with either X or Y depending on input as seen in figure 1. This makes sense, due to the rotors providing a constant force in those directions.

3 Stability and Observability

Before designing an observer or controller, it is important to see if the system is controllable and observable. Controllability is the strongest form of stabilizability, it means that the system can be driven to any state that exists in the space. If a system is controllable, that notion implies stabilizable. To check for controllability, the controllability matrix is used, defined as

$$\mathcal{C} = [B \quad AB \quad A^2B \quad ... \quad A^{n-1}B] \tag{13}$$

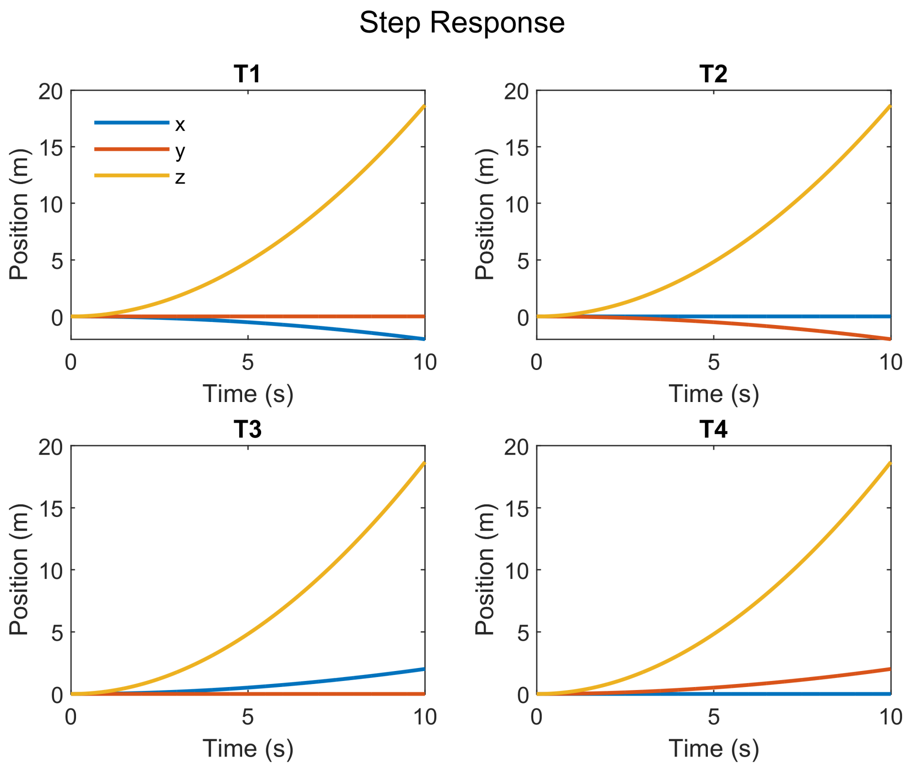
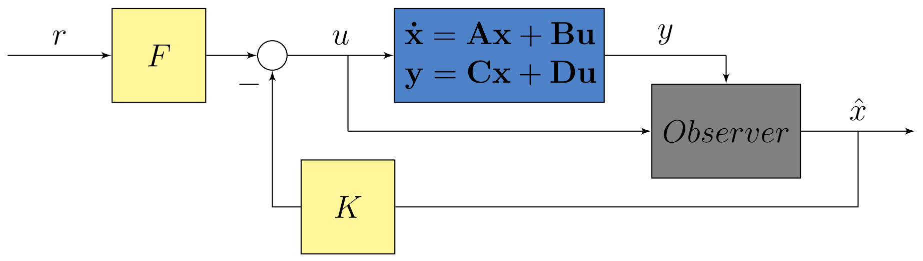


Figure 1: Step response from each individual rotor input.

for this system $n = 6$ and results in a 6x24 matrix of rank 6. The matrix has full rank and this result implies that any state $\vec{x}(t)$ can be written as a linear combination of the columns of $\mathcal{C}$. The system is completely controllable, and can be steered to any state by a specific input $u(t)$. Additionally, since the system is controllable it implies the weaker condition stabilizable. Looking at the zero-input response and the eigenvalues from table 1, the system is known to be ISL stable, due to the zero eigenvalues with algebraic multiplicity 3.

The other important to consider when implementing state feedback is observability which is the dual of controllablility. Essentially observability describes whether the time history from the system output $y(t)$ and system input $u(t)$ on the interval $[0, t]$ can the state be determined. Observability is the stronger form of detectability where the poles of the state observer can be placed to be stable. A test to determine observability is through the rank



r
F
$-$
u
$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$
$\mathbf{y} = \mathbf{Cx} + \mathbf{Du}$
y
$Observer$
$\hat{x}$
K

Figure 2: Closed loop block diagram for the state space system with feedback gain matrix K with an observer.

of the observability matrix defined as

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ . \\ . \\ CA^{n-1} \end{bmatrix} \tag{14}$$

for this system $n = 6$ and results in a 18x6 matrix. The observability matrix has full column rank and implies the system is fully observable. If a system is observable, the stronger condition of detectability, it thereby implies the system is detectable.

4 State Feedback Controller Design

Due to the performance conditions presented in the second section, the poles of the closed loop transfer function should be moved to locations $\lambda = -5$, and $\lambda = -3$. This is accomplished by solving the set of equations described by the feedback loop.

The block diagram corresponds the following set of equations.

$$\dot{\mathbf{x}} = (A - BK)x + BFr \tag{15}$$

solving for the eigenvalues in terms of the characteristic equation given by $DET(A-\lambda I) = 0$, or alternatively using the *matlab* command *place(A,B,[poles])*. Desiring the poles found in table 2. The resulting gain matrix K is shown

λ_1	-3	m = 3
λ_2	-5	m = 3

Table 2: Desired closed loop poles

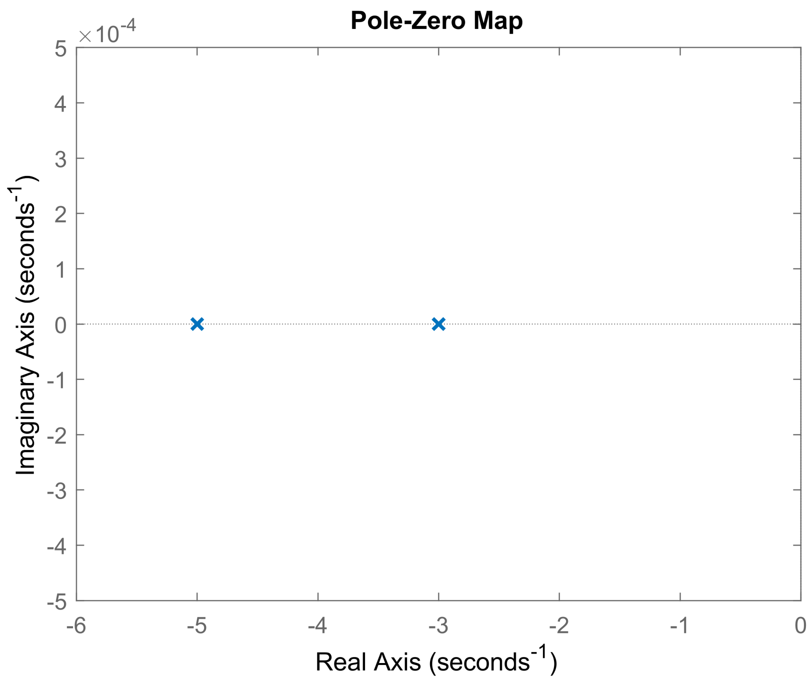


Figure 3: Pole-Zero map for closed loop system under feedback gain K. Note the desired poles under feedback are -3 and -5.

here,

$$K = \begin{bmatrix} -179.9856 & -95.8675 & 0 & 0 & 9.3750 & 4.9870 \\ 0 & 0 & -179.9856 & -95.8675 & 9.3750 & 4.9870 \\ 179.9856 & 95.8675 & 0 & 0 & 9.3750 & 4.9870 \\ 0 & 0 & 179.9856 & 95.8675 & 9.3750 & 4.9870 \end{bmatrix} \tag{16}$$

To confirm the poles of the closed loop transfer function have been moved successfully, the roots of (A-BK) are shown on a map of the complex plane in figure 3.

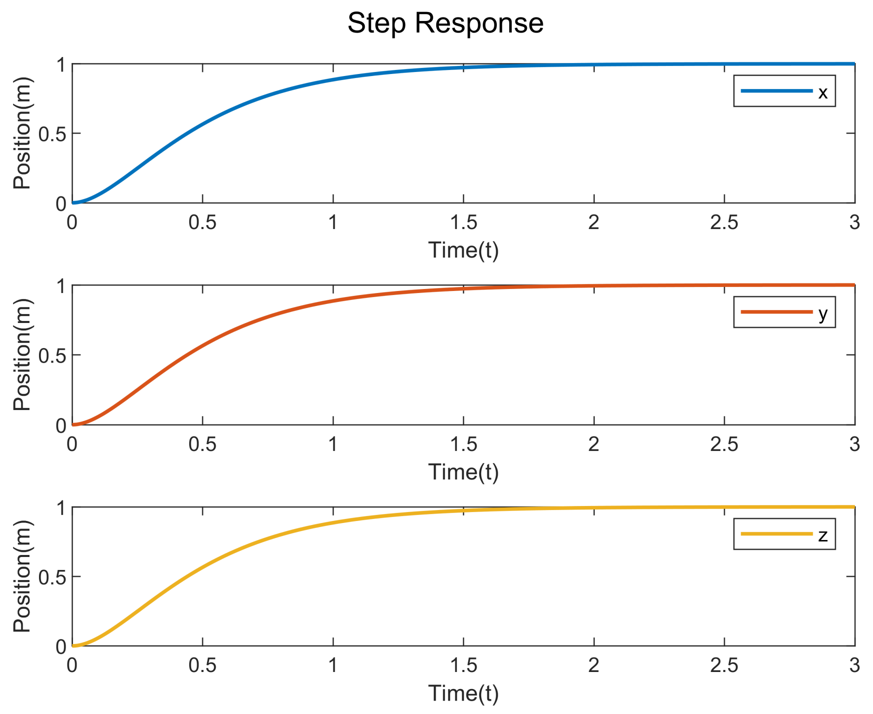


Figure 4: The step response to a 1 meter step reference for each direction X,Y,Z.

To determine the matrix F, equation 15 taking the Laplace transform and solving for x yields

$$x = (sI - (A - BK))^{-1} BFr \tag{17}$$

By solving for F and by evaluating the expression at $s = 0$ gives the matrix that will have a steady-state error of zero to a step input for position (X,Y,Z). Because B in not a square matrix and does not have an inverse, the psuedo-inverse is used to solve for F.

$$F = B^{*}(BB^{*})^{-1}(-(A - BK)) \tag{18}$$

Now for positional reference inputs, the performance of the closed loop transfer function is as expected, meeting the rise time and steady-state error requirement. The rise time to a step is 0.89 seconds.

5 State Observer Design

As a general rule, the poles of the observer should be 10x greater than those of the controller to process fast enough as to not add a delay into the system. Utilizing this assumption and the fact the system is observable, the poles of the observer are placed at -30 and -50. This operation is done in the same fashion as for the controller. An observer of the form given by the following equations

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{L}(\mathbf{y} - \hat{\mathbf{y}}) \tag{19}$$

$$\mathbf{y} = \mathbf{C}\hat{\mathbf{x}} + \mathbf{D}\mathbf{u} \tag{20}$$

using the substitution, $e = x - \hat{x}$ the expression is simplified to

$$\dot{\mathbf{e}} = (\mathbf{A} - \mathbf{LC})\mathbf{e}. \tag{21}$$

Note that the orientation of (A-LC) is similar to (A-BK) but the places of K and L are switched. In order to use the place command in *matlab*, the knowledge that the eigenvalues of A are the same as A^T, the expression can be written as

$$(A - LC)^T = A^T - C^T L^T \tag{22}$$

and the *matlab* command becomes $place(A^T, C^T, [poles])$. To look at how the error of the observer varies over time, we will construct a new matrix consisting of the two sets of state equation (for x and e). The resultant set of equations looks like

$$\dot{\hat{x}} = \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix} \begin{bmatrix} x \\ e \end{bmatrix} + \begin{bmatrix} BF \\ 0 \end{bmatrix} r \tag{23}$$

$$y = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x \\ e \end{bmatrix} + (0)r + D(u) \tag{24}$$

and is known as the meta system. Utilizing the *lsim* function in *matlab*, the states of the meta-state representation can be plotted. A step input is used to initialize an error in the observer. The response is as expected and occurs approximately 10x as fast as the system performs. Comparing the performance of the closed loop step response with the observer to the original state-feedback system as seen in figure 6, the rise time is approximately the same for both systems, with about a 0.4 second delay introduced by the initial error in the observer. The two converge at approximately 2.5 seconds.

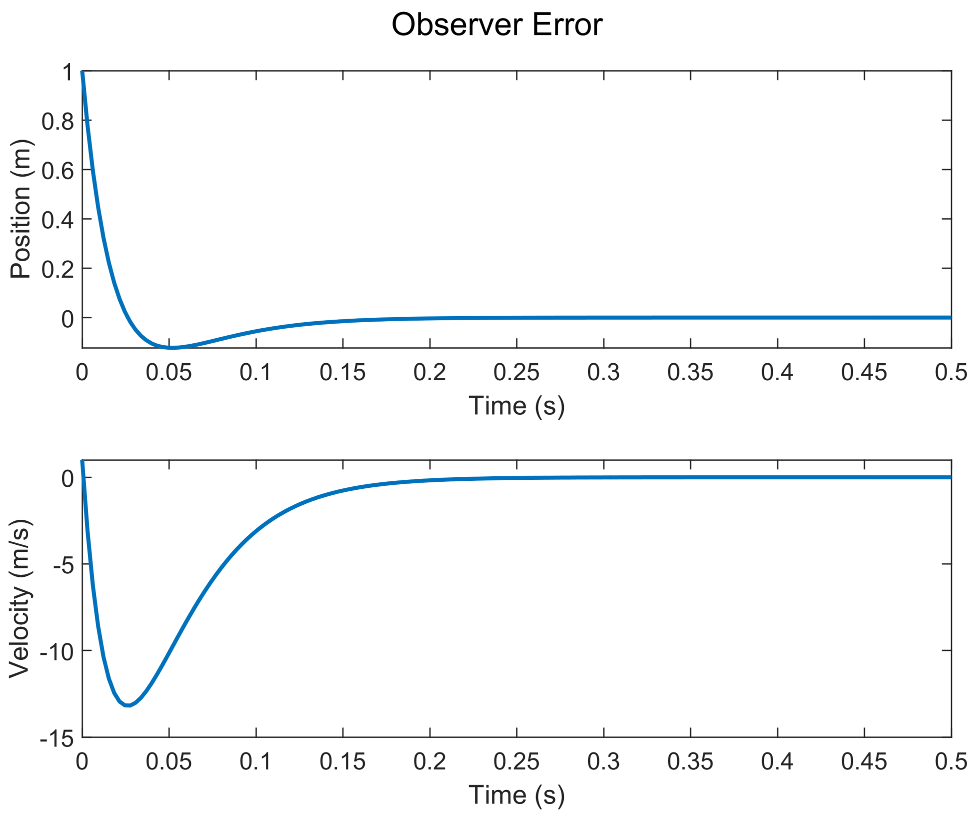


Figure 5: Observer error for velocity and position observer estimate with an initial error introduced of 1 m for position and 1m/s for velocity in the x-direction. The error reduces to zero approximately 10x as fast as the system closed loop poles.

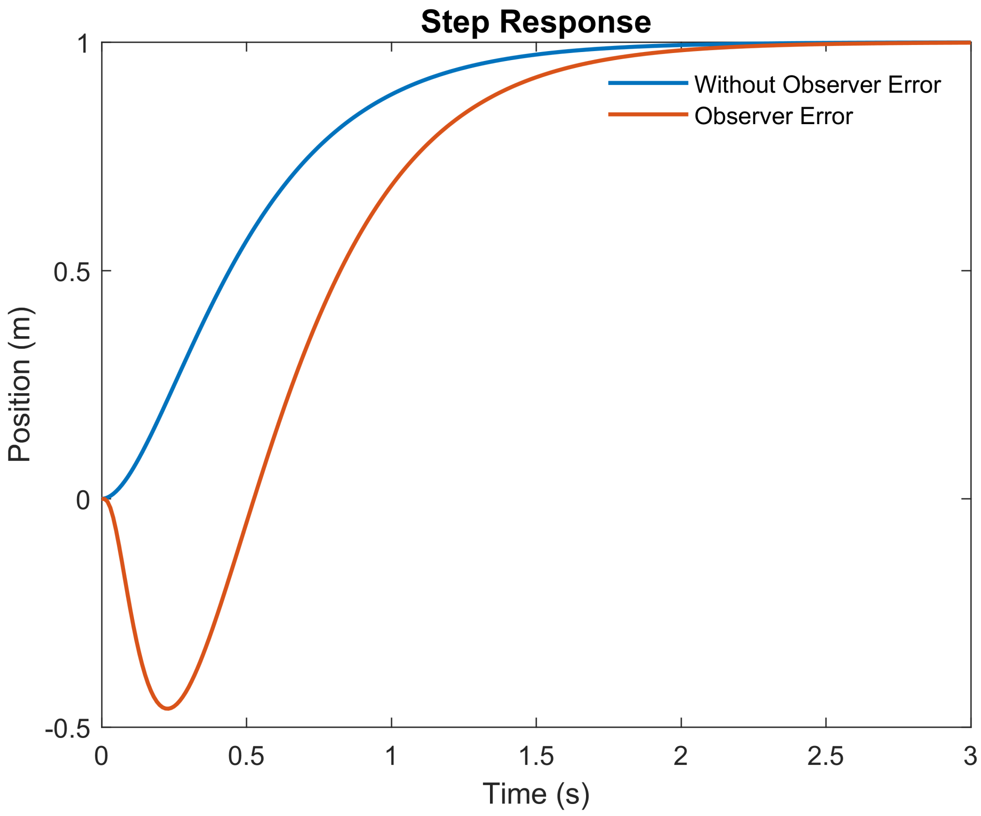


Figure 6: The step response to a 1 meter reference in the x-direction. The observer was initialized with a 1 meter error. The delay added to the system was approximately 0.4 seconds, and the two converged to zero steady-state error at approximately 2.5 seconds.

A Code

A.1 Observability Matrix

$$\mathcal{O} = \begin{bmatrix}
1.0000 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1.0000 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.0000 & 0 \\
0 & 1.0000 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1.0000 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1.0000 \\
0 & -0.0104 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -0.0104 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.0208 \\
0 & 0.0001 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.0001 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.0004 \\
0 & -0.0000 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -0.0000 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.0000 \\
0 & 0.0000 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.0000 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.0000
\end{bmatrix}$$

A.2 Controllability Matrix

$$\mathcal{C}^{\mathcal{T}} = \begin{bmatrix}
0 & -0.0417 & 0 & 0 & 0 & 0.4000 \\
0 & 0 & 0 & -0.0417 & 0 & 0.4000 \\
0 & 0.0417 & 0 & 0 & 0 & 0.4000 \\
0 & 0 & 0 & 0.0417 & 0 & 0.4000 \\
-0.0417 & 0.0004 & 0 & 0 & 0.4000 & -0.0083 \\
0 & 0 & -0.0417 & 0.0004 & 0.4000 & -0.0083 \\
0.0417 & -0.0004 & 0 & 0 & 0.4000 & -0.0083 \\
0 & 0 & 0.0417 & -0.0004 & 0.4000 & -0.0083 \\
0.0004 & -0.0000 & 0 & 0 & -0.0083 & 0.0002 \\
0 & 0 & 0.0004 & -0.0000 & -0.0083 & 0.0002 \\
-0.0004 & 0.0000 & 0 & 0 & -0.0083 & 0.0002 \\
0 & 0 & -0.0004 & 0.0000 & -0.0083 & 0.0002 \\
-0.0000 & 0.0000 & 0 & 0 & 0.0002 & -0.0000 \\
0 & 0 & -0.0000 & 0.0000 & 0.0002 & -0.0000 \\
0.0000 & -0.0000 & 0 & 0 & 0.0002 & -0.0000 \\
0 & 0 & 0.0000 & -0.0000 & 0.0002 & -0.0000 \\
0.0000 & -0.0000 & 0 & 0 & -0.0000 & 0.0000 \\
0 & 0 & 0.0000 & -0.0000 & -0.0000 & 0.0000 \\
-0.0000 & 0.0000 & 0 & 0 & -0.0000 & 0.0000 \\
0 & 0 & -0.0000 & 0.0000 & -0.0000 & 0.0000 \\
-0.0000 & 0.0000 & 0 & 0 & 0.0000 & -0.0000 \\
0 & 0 & -0.0000 & 0.0000 & 0.0000 & -0.0000 \\
0.0000 & -0.0000 & 0 & 0 & 0.0000 & -0.0000 \\
0 & 0 & 0.0000 & -0.0000 & 0.0000 & -0.0000
\end{bmatrix}$$

A.3 Matlab Analysis

```
%% Linear Systems Project MCEN 5228
%   Riley Kenyon 12/14/2019
close all; clear all; clc;

%% Derive System Matrices
% Model of Plant
A = [0 1 0 0 0 0; 0 -0.0104 0 0 0 0; 0 0 0 1 0 0; 0 0 0 -0.0104 0 0; 0 0 0 0 0 1;
B = [ 0 0 0 0; -0.04167 0 0.04167 0; 0 0 0 0; 0 -0.04167 0 0.04167; 0 0 0 0; 0.4
C = [1 0 0 0 0 0; 0 0 1 0 0 0; 0 0 0 0 1 0];
```

```
D = zeros(3,4);
model = ss(A,B,C,D);

%% Simulate Step Response
t = [0:0.01:10];
[y,~,~] = step(model,t); % This is a step to the rotor input
figure, hold on;
subplot(2,2,1)
plot(t,y(:,:,1),'LineWidth',1.5)
legend('x','y','z','Location','NorthWest','Box','Off')
title('T1')
xlabel('Time (s)')
ylabel('Position (m)')

subplot(2,2,2)
plot(t,y(:,:,2),'LineWidth',1.5)
title('T2')
xlabel('Time (s)')
ylabel('Position (m)')

subplot(2,2,3)
plot(t,y(:,:,3),'LineWidth',1.5)
title('T3')
xlabel('Time (s)')
ylabel('Position (m)')

subplot(2,2,4)
plot(t,y(:,:,4),'LineWidth',1.5)
title('T4')
xlabel('Time (s)')
ylabel('Position (m)')
sgtitle('Step Response','Fontsize',12)

% u = ones(4,length(t));
% figure, lsim(model,u,t);

% Zero input response
    % zir = ss(A,zeros(size(B)),C,zeros(size(D)));
    % initial(zir,[1,-1,0,0,0,0]')

%% System Properties
% Controllability
c_mat = [B A*B A^2*B A^3*B A^4*B A^5*B]; % could use ctrb(A,B)
r = rank(c_mat);
% Note that the rank of the controllability matrix is full row rank which
% implies that the system is  completely controllable

% Observability
o_mat = [C; C*A; C*A^2; C*A^3; C*A^4; C*A^5];   % Could use obsv(A,C)
% o_mat is full rank implies system is completely observable

%% State Feedback Controller
%p = [-10,-10, -10, -7, -7, -7]; % Desired closed loop poles
p = [-3,-3,-3,-5,-5,-5];
K = place(A,B,p); % Gain matrix to move poles to desired position
A_CL = A - B*K; % Closed loop system A
F = pinv(inv(-A_CL)*B); % Determine pre-feedback gain matrix
model2 = ss(A_CL,B*F,C,D*F); % Model of closed loop with desired pole loc.

% Simulate Response
t = [0:0.01:3];
r = [1 0 1 0 1 0]'*ones(1,length(t));
[y,t,x] = lsim(model2,r,t);
figure, subplot(3,1,1)
plot(t,y(:,1),'LineWidth',1.5)
ylabel('Position(m)')
xlabel('Time(t)')
legend('x')

subplot(3,1,2)
plot(t,y(:,2),'color',[0.8500 0.3250 0.0980],'LineWidth',1.5)
ylabel('Position(m)')
xlabel('Time(t)')
legend('y')

subplot(3,1,3)
```

```
plot(t,y(:,3),'color',[0.9290 0.6940 0.1250],'LineWidth',1.5)
ylabel('Position(m)')
xlabel('Time(t)')
legend('z')
sgtitle('Step Response','fontsize',12)
% Zero-input response of closed loop
% zir2 = ss(A_CL,zeros(size(B*F)),C,zeros(size(D*F)));
% initial(zir2,[1,-1,0,0,0,0]')

%% State Observer
%p = [-100,-100,-100,-110,-110,-110];
p = [-50,-50,-50,-30,-30,-30];
L = place(A',C',p);
L = L'; % using place to determine L requires transpose

% Meta-state Matrix
C_tilde = [C zeros(size(C))];
A_tilde = [(A - B*K) B*K; zeros(size(A)) A-L*C]; % Matrix for system and error
B_tilde = [B*F; zeros(size(B*F))];
D_tilde = zeros(size(C_tilde,1),size(B_tilde,2));
model3 = ss(A_tilde,B_tilde,C_tilde,D_tilde);

% State Observer Error given an initial offset
[~,t,x] = initial(model3,[zeros(1,6) 1 1 0 0 0 0]'); % offset observer error
figure,hold on
subplot(2,1,1)
plot(t,x(:,7),'LineWidth',1.5)
xlim([0 0.5])
xlabel('Time (s)')
ylabel('Position (m)')
subplot(2,1,2)
plot(t,x(:,8),'LineWidth',1.5)
xlim([0 0.5])
xlabel('Time (s)')
ylabel('Velocity (m/s)')
sgtitle('Observer Error','fontsize',12)

% step response using observer
```

```
t = [0:0.01:3];
u = [1 0 0 0 0 0]'*ones(1,length(t));
[y_offset,t_offset,x] = lsim(model3,u,t,[zeros(1,6),1,zeros(1,5)]'); %Error Case
figure, hold on
plot(t,y(:,1),'LineWidth',1.5) % Without observer error
plot(t_offset,y_offset(:,1),'LineWidth',1.5)
xlabel('Time (s)')
ylabel('Position (m)')
legend('Without Observer Error','Observer Error','Box','off')
title('Step Response','Fontsize',12)
```