The purpose of this assignment is to separate the complex dynamics of the UAV into a set of more simple systems. This can be done because the there are four coupled systems that do not depend on each other. This benefit is twofold: first, it is easier to calculate and intuit the dynamics of the UAV, and second we can give each coupled system a single input. We then are able to analyze the coupled systems for their properties individually, to find the properties of each state, such as eigenvalues, eigenvectors, mode functions, stability, controllability and observability.

1 Assumptions and constants

The assumptions and constants are largely unchanged from the previous assignment. To save space, I will just list the relevant constants and values for code.

Listing 1: Constants and Assumptions

```
%UAV
UAV_mass = 0.068; %kg ; total mass of UAV
Jxx = 0.0000582857;
Jyy = 0.0000716914;
Jzz = 0.0001;
J = [... % kg/m2 ; Moment of Inertia Matrix of UAV
     Jxx   0   0 ;...
      0   Jyy  0 ;...
      0    0  Jzz;...
];

%PHYSICAL CONSTANTS
rho = 1.22495; % kg/m3 ; density of air, given in presentation
F_g = [0;0;UAV_mass*9.81]; %m/s2 ; acceleration due to gravity in NED frame

%PROPELLER
prop_dist = sqrt(2)*0.09; %meters ; distance from CM to prop
prop_mass = 0.001; %kg (1 gram) ; mass of prop
prop_dia = 0.066; %meters ; diameter of prop
prop_mass_moment = 1/12*(prop_dia^2)*prop_mass; %kg/m2 ; Moment of Inertia of prop about its CM
Jprop = prop_mass_moment + prop_mass*(prop_dist^2); %kg/m2 ; MoI of prop in UAV frame, from Parallel Axis
C_T = [0.069075;4.95e-05]; %C_T(n) = C_T(1) + C_T(2)*n for prop spinning at n Hz
C_P = 0.041;
thrust_eq = @(n) [(C_T(1)+C_T(2)*n).*(n.^2)*rho*(prop_dia^4)]; %Thrust in N, a function of motor input, Hz
power_eq  = @(n) [(C_P)*(n.^3)*rho*(prop_dia^5)]; %Power in W, a function of motor input, Hz

%TRIM STATE
trim_state = [0,0,0,0,0,0,0,0,0,0,0,0]; %All states at 0
trim_input = [293,293,293,293]; %Hz for all propellers
```

The two big changes we are making are that we are changing the order of state variables in the state vector, and we are now using virtual inputs $TEAR$ instead of n_1, n_2, n_3, n_4. The state variables will be ordered as follows:

$$\begin{pmatrix} x & u & \theta & q & y & v & \phi & p & z & w & \psi & r \end{pmatrix}, \tag{1}$$

and the virtual input u is resolved to

$$v = M(u + u_0) \tag{2}$$

where v is the actual motor input and u_0 is the trim $TEAR$ input, $[293, 0, 0, 0]$. We will use the similarity transformation shown in equation 4 and Listing 2, adjusted to the state vector in equation 1 to create this change. We assume that the state matrices have been calculated by linearizing the system, as shown in the previous assignment.

Our last assumption that we make has to do with observation of states. We assume that our IMU is able to measure the angular velocities p, q, r, and the body-axis accelerations, a_x, a_y, a_z. Additionally, the UAV has an ultrasonic distance sensor that is able to sense its vertical height z, h, and a camera with Optical Flow to get the x, y velocities, u, v. We assume that we can perfectly observe these, and that all additional states and measurements will have to be reconstructed from an observer.

2 Derivations and Calculations

2.1 Similarity Transformations

This section is just an exercise to practice similarity transformations. In this case, we are using the transformation to reorder our state vector into the form of

$$\begin{pmatrix} z & w & \theta & q & \phi & p & \psi & r & x & u & y & v \end{pmatrix} \tag{3}$$

We create a transformation matrix T that maps each variable to a new position:

$$T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{4}$$

and define our new state X' to be

$$X' = TX \tag{5}$$

We can then use this to find our new A, B, C, D matrices:

$$\dot{X}' = TAT^{-1}X' + TBu \tag{6}$$

$$y = CT^{-1}X' + Du \tag{7}$$

We will call these new matricies $\bar{A}, \bar{B}, \bar{C}, \bar{D}$, and solved for them in matlab:

$$\bar{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \bar{B} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -\frac{9}{500} & -\frac{9}{500} & -\frac{9}{500} & -\frac{9}{500} \\ 0 & 0 & 0 & 0 \\ \frac{1553}{1000} & \frac{1553}{1000} & -\frac{1553}{1000} & -\frac{1553}{1000} \\ 0 & 0 & 0 & 0 \\ \frac{191}{100} & -\frac{191}{100} & -\frac{191}{100} & \frac{191}{100} \\ 0 & 0 & 0 & 0 \\ -\frac{59}{1000} & \frac{59}{1000} & -\frac{59}{1000} & \frac{59}{1000} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{8}$$

$$\bar{C} = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \bar{D} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{9}{500} & -\frac{9}{500} & -\frac{9}{500} & -\frac{9}{500} \end{pmatrix} \tag{9}$$

The code for this is shown below:

Listing 2: Similarity Transform

```
%starting state
states = [x,y,z,u,v,w,phi,theta,psi,p,q,r];
% new_states = [z,w,theta,q,phi,p,psi,r,x,u,y,v];

% Transformation Matrix X' = TX
T =     [... x  y  z  u  v  w phi theta psi p  q  r
             0  0  1  0  0  0  0    0    0  0  0  0 ;... x
             0  0  0  0  0  1  0    0    0  0  0  0 ;... y
             0  0  0  0  0  0  0    1    0  0  0  0 ;... z
             0  0  0  0  0  0  0    0    0  0  1  0 ;... u
             0  0  0  0  0  0  1    0    0  0  0  0 ;... v
             0  0  0  0  0  0  0    0    0  1  0  0 ;... w
             0  0  0  0  0  0  0    0    1  0  0  0 ;... phi
             0  0  0  0  0  0  0    0    0  0  0  1 ;... theta
             1  0  0  0  0  0  0    0    0  0  0  0 ;... psi
             0  0  0  1  0  0  0    0    0  0  0  0 ;... p
             0  1  0  0  0  0  0    0    0  0  0  0 ;... q
             0  0  0  0  1  0  0    0    0  0  0  0 ;... r
];

%% Transform states
% using TEAR as input
B = B*MotorMix;
D = D*MotorMix;

A = T*A/T;
B = T*B;
C = C/T;
% D = D;
```

Also note that in this process, we used the new input, $TEAR$, so we updated B, D accordingly. This exercise was done before I changed the order of the state variables, so the original state vector before the transform is from the previous assignment.

2.2 Decoupling the System

In this section we will split our system from one large MIMO system into 4 smaller SIMO systems that are easier to analyze. Where we decided to split the systems comes from two factors. First, we wanted each subsystem to have a single input from $TEAR$, and second, we wanted to avoid splitting coupled states. The state variables along with their given input as subsystems are shown below:

$$\text{Thrust: } \begin{pmatrix} z & w \end{pmatrix} \tag{10}$$
$$\text{Elevator: } \begin{pmatrix} x & u & \theta & q \end{pmatrix} \tag{11}$$
$$\text{Aileron: } \begin{pmatrix} y & v & \phi & p \end{pmatrix} \tag{12}$$
$$\text{Rudder: } \begin{pmatrix} \psi & r \end{pmatrix} \tag{13}$$

To construct our new state equations, we start by writing out A and B from the full system:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -425.3240 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 523.148 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1553}{1000} & \frac{1553}{1000} & \frac{1553}{1000} & -\frac{1553}{1000} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{191}{100} & \frac{191}{100} & \frac{191}{100} & -\frac{191}{100} \\ 0 & 0 & 0 & 0 \\ -\frac{9}{500} & -\frac{9}{500} & -\frac{9}{500} & \frac{9}{500} \\ 0 & 0 & 0 & 0 \\ -\frac{59}{1000} & -\frac{59}{1000} & -\frac{59}{1000} & \frac{59}{1000} \end{pmatrix} \tag{14}$$

Then, to get the z, w, Thrust sub system, we take the square of entries of A that relate to the two state variables to get

$$A_{thrust} = A(9:10, 9:10) \tag{15}$$

and the entries of B that are related to the states, with Thrust as an input,

$$B_{thrust} = B(9:10, 1) \tag{16}$$

Finally, we pick what we will be able to observe in each subsystem to construct C and D. From our assumptions, for thrust, we are able to observe Z, h, a_z, which we will combine to create the output of the system. We then have the following equations for the subsystem state space model:

$$A_{thrust} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} B_{thrust} = \begin{pmatrix} 0 \\ -\frac{9}{125} \end{pmatrix} \tag{17}$$

$$C_{thrust} = \begin{pmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 0 \end{pmatrix} D_{thrust} = \begin{pmatrix} 0 \\ 0 \\ -\frac{9}{125} \end{pmatrix} \tag{18}$$

We repeat this process for the remaining inputs. For elevator, we are able to observe u, q, a_x:

$$A_{elevator} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} B_{elevator} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{1553}{250} \end{pmatrix} \tag{19}$$

$$C_{elevator} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} D_{elevator} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{20}$$

Aileron can measure v, p, a_y:

$$A_{aileron} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} B_{aileron} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{191}{25} \end{pmatrix} \tag{21}$$

$$C_{aileron} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} D_{aileron} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{22}$$

And rudder can measure only r:

$$A_{rudder} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} B_{rudder} = \begin{pmatrix} 0 \\ \frac{59}{250} \end{pmatrix} \tag{23}$$

$$C_{rudder} = \begin{pmatrix} 0 & 1 \end{pmatrix} D_{rudder} = (0) \tag{24}$$

The matlab code that generated these matrices is shown below:

Listing 3: Decoupling States

```
%% Decouple States

% Thrust
% Z,w
Athrust = A(9:10,9:10);
Bthrust = B(9:10,1);
% Z,h,az
Cthrust = [ 1 0; -1 0; 0 0];
Dthrust = [0 ; 0 ; D(4,1)];

% Elevator
% X,u,theta,q
Aelevator = A(1:4,1:4);
Belevator = B(1:4,2);
% u,q,ax
Celevator = [0 1 0 0; 0 0 0 1; 0 0 0 0];
Delevator = [0 ; 0 ; 0];

% Aileron
% Y,v,phi,p
Aaileron = A(5:8,5:8);
Baileron = B(5:8,3);
% v,p,ay
Caileron = [0 1 0 0; 0 0 0 1; 0 0 0 0];
Daileron = [0 ; 0 ; 0];

% Rudder
% psi,r
Arudder = A(11:12,11:12);
Brudder = B(11:12,4);
% r
Crudder = [0 1];
Drudder = 0;
```

3 Analysis

For each of the four subsystems, we will do the following analysis:

1. Calculate the eigenvalues and eigenvectors
2. Determine the stability of the open-loop
3. Determine if the system is observable and controllable, and if not, which states are not
4. Construct a 2-state phase portrait

To start with, we calculate the eigenvalues of the subsystems. Because A matrices are all upper triangular, we just look at the main diagonal entries to get the eigenvalues: $\lambda_i = 0$ for all $i \in \{1, ..., n\}$. This is true for all four subsystems. This is consistent with our results from the previous assignment.

For the eigenvectors, we use Matlab's 'eig' command to get

$$V_{thrust} = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix} \tag{25}$$

$$V_{elevator} = \begin{pmatrix} 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{26}$$

$$V_{aileron} = \begin{pmatrix} 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{27}$$

$$V_{rudder} = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix} \tag{28}$$

where the columns of the matrices V_i are the ith eigenvectors. The fact that they all have repeated eigenvalues for repeated eigenvectors makes sense, since each A matrix is in Jordan form, or nearly in Jordan form, so we expect algebraic multiplicity of n and geometric multiplicity of 1.

Because each subsystem is a form of double of quadruple integrator, we would expect every state to be controllable. Additionally, based on how we defined our C matrices, we would expect a number of states to be unobservable. The states we don't have direct measurements for are $X, Y, w, \phi, \theta, \psi$. We therefore expect a minimum of the other 6 states to be observable.

To confirm our guesses, we create controllability and observability matrices,

$$W = \begin{pmatrix} B & AB & \dots & A^{n-1}B \end{pmatrix} \tag{29}$$

$$M = \begin{pmatrix} C \\ AC \\ \vdots \\ CA^{n-1} \end{pmatrix} \tag{30}$$

and test if they are full row-rank and full column-rank, respectively. The controllability matrices are shown below:

$$W_{thrust} = \begin{pmatrix} 0 & -\frac{9}{500} \\ -\frac{9}{500} & 0 \end{pmatrix} \tag{31}$$

$$W_{elevator} = \begin{pmatrix} 0 & 0 & 0 & -\frac{1523}{100} \\ 0 & 0 & -\frac{1523}{100} & 0 \\ 0 & \frac{31}{20} & 0 & 0 \\ \frac{31}{20} & 0 & 0 & 0 \end{pmatrix} \tag{32}$$

$$W_{aileron} = \begin{pmatrix} 0 & 0 & 0 & \frac{937}{50} \\ 0 & 0 & \frac{937}{50} & 0 \\ 0 & \frac{191}{100} & 0 & 0 \\ \frac{191}{100} & 0 & 0 & 0 \end{pmatrix} \tag{33}$$

$$W_{rudder} = \begin{pmatrix} 0 & \frac{3}{50} \\ \frac{3}{50} & 0 \end{pmatrix} \tag{34}$$

These are all visibly full row-rank, so all states are controllable. For visibility, I transposed my observability matrices, so the criterion for observability for my M matrices is also full row-rank. They are shown below:

$$M_{thrust} = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \end{pmatrix} \tag{35}$$

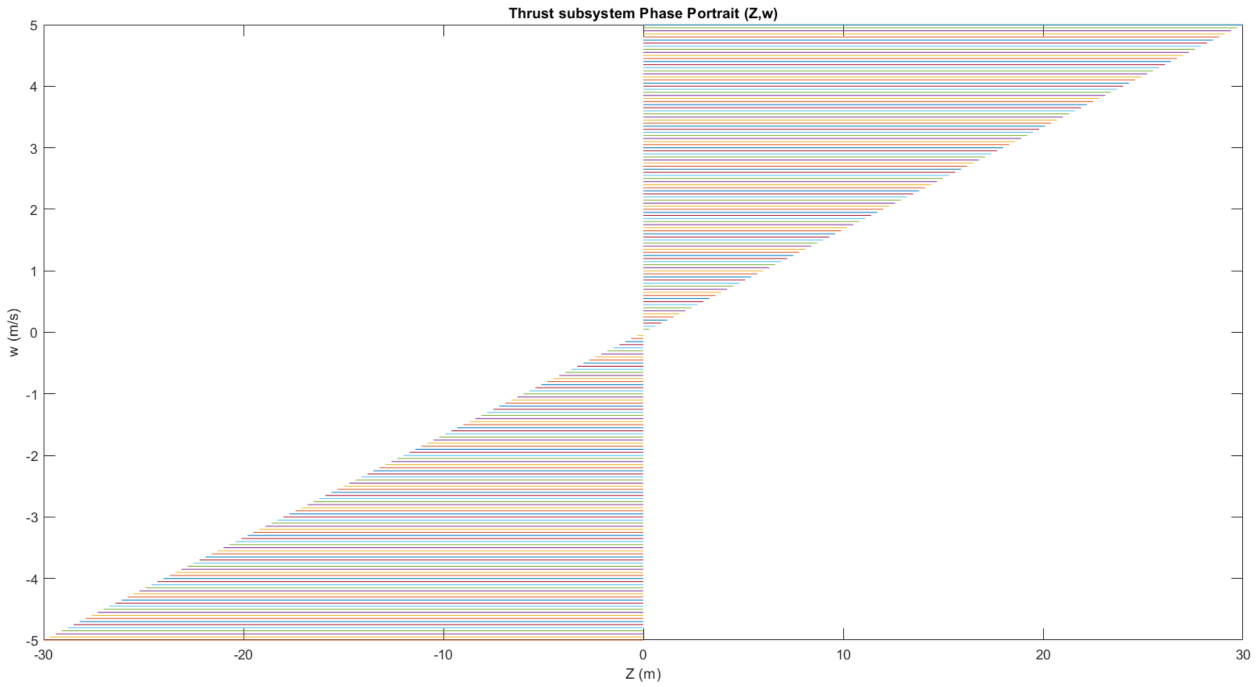
$$M_{elevator} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{981}{100} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -\frac{981}{100} & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{36}$$

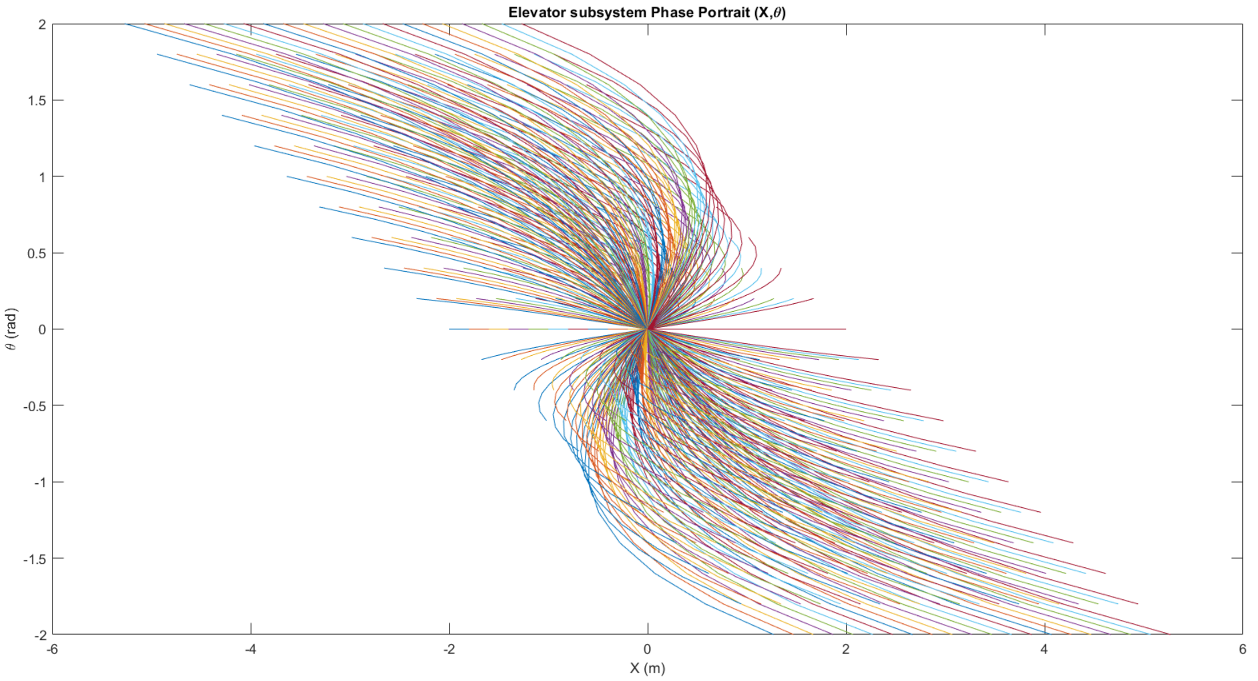
$$M_{aileron} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{981}{100} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \frac{981}{100} & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{37}$$

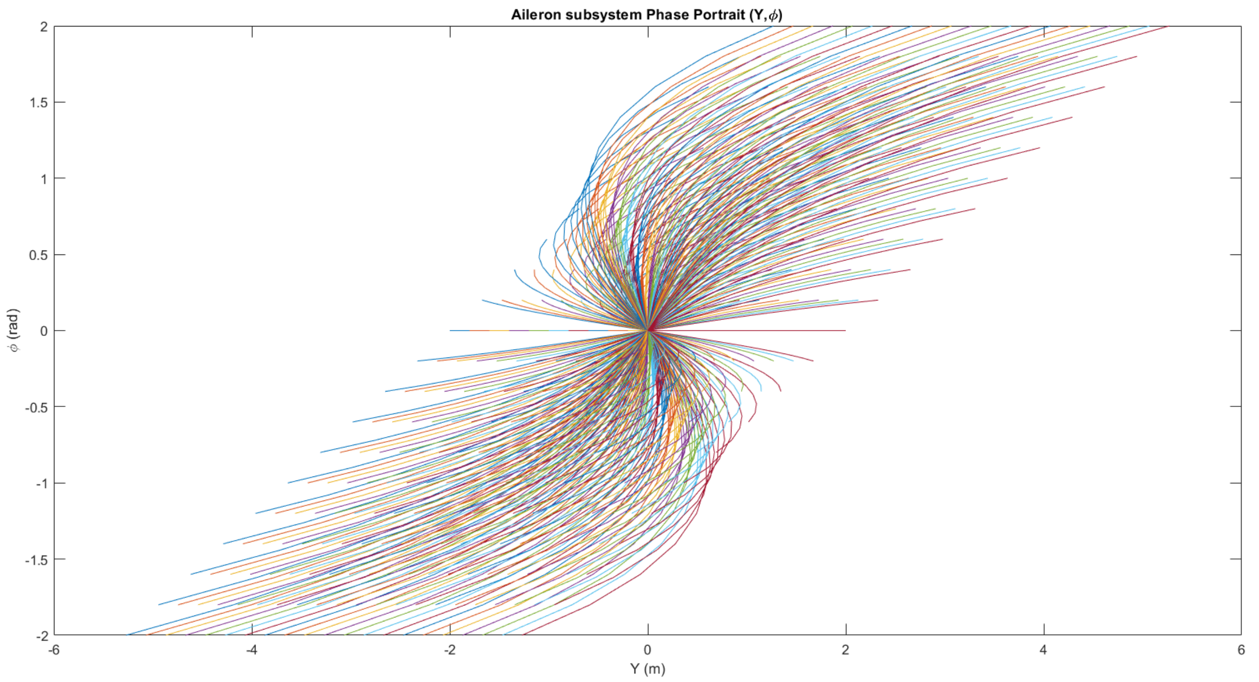
$$M_{rudder} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \tag{38}$$

Visibly, we can see the only subsystem that is completely observable is Thrust. This means that we will be able to reconstruct w mathematically. To find which states are not observable, we look at which of the rows are not independent, and compare them to their state. So, we cannot observer X, Y, ψ, but will be able to reconstruct θ, ψ. Note that the reconstructions may only work for the linear system, and not the full model, which is why they may be unintuitive.

Lastly, we will construct phase portraits of the decoupled systems. for Thrust and Rudder, they are simple double integrators, but for Elevator and Aileron, I chose the position and angle states because they looked interesting. The plots are all shown on the next pages in Figure 1.f


Thrust subsystem Phase Portrait (Z,w)
w (m/s)
Z (m)
5
4
3
2
1
0
-1
-2
-3
-4
-5
-30
-20
-10
0
10
20
30


Elevator subsystem Phase Portrait (X,θ)
θ (rad)
X (m)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-6
-4
-2
0
2
4
6



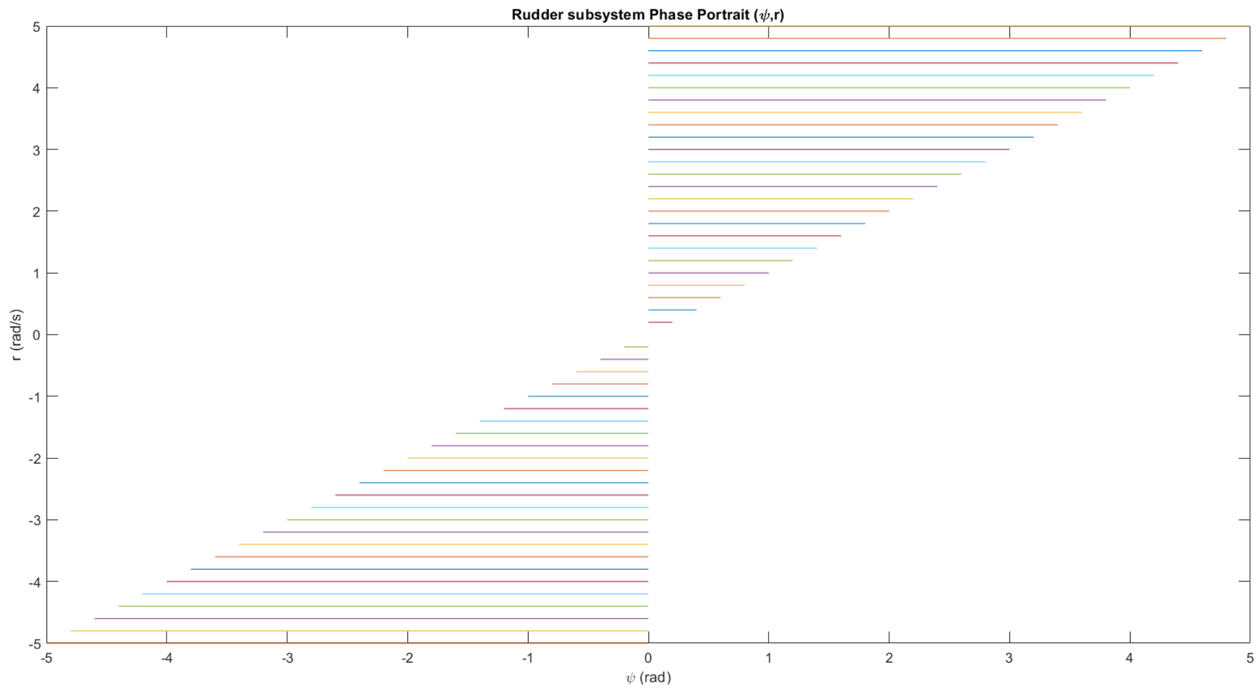


Figure 1: Phase Portraits

4 Updating the Model

The final step is to integrate our decoupled systems into the Simulink model previously made and compare the outputs to confirm they are valid. To do this I created a the linear subsystems, with no outputs, and ran the model. The Simulink decoupled systems are shown in Figure 2.

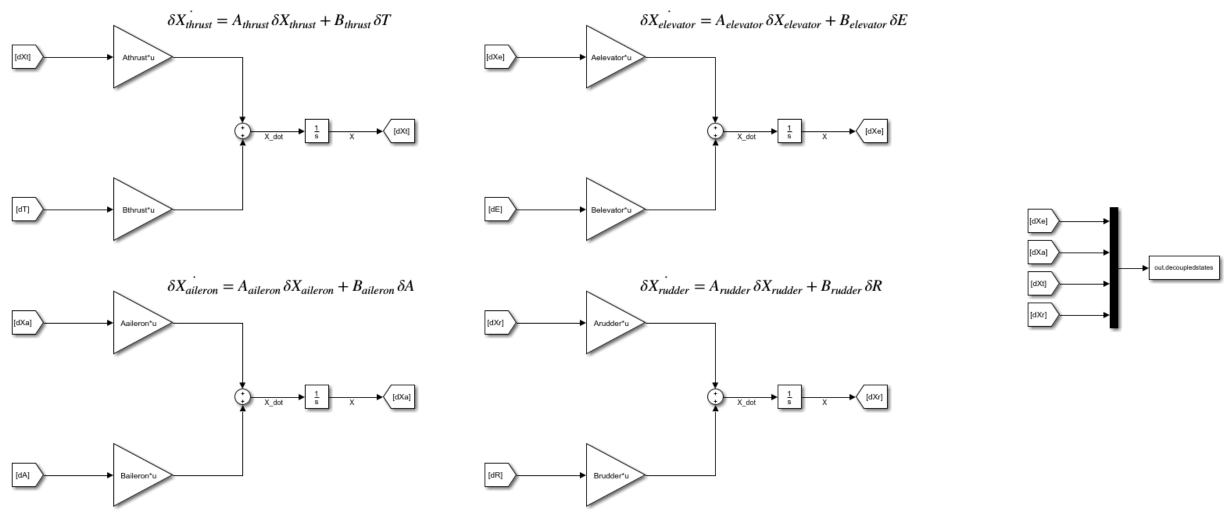
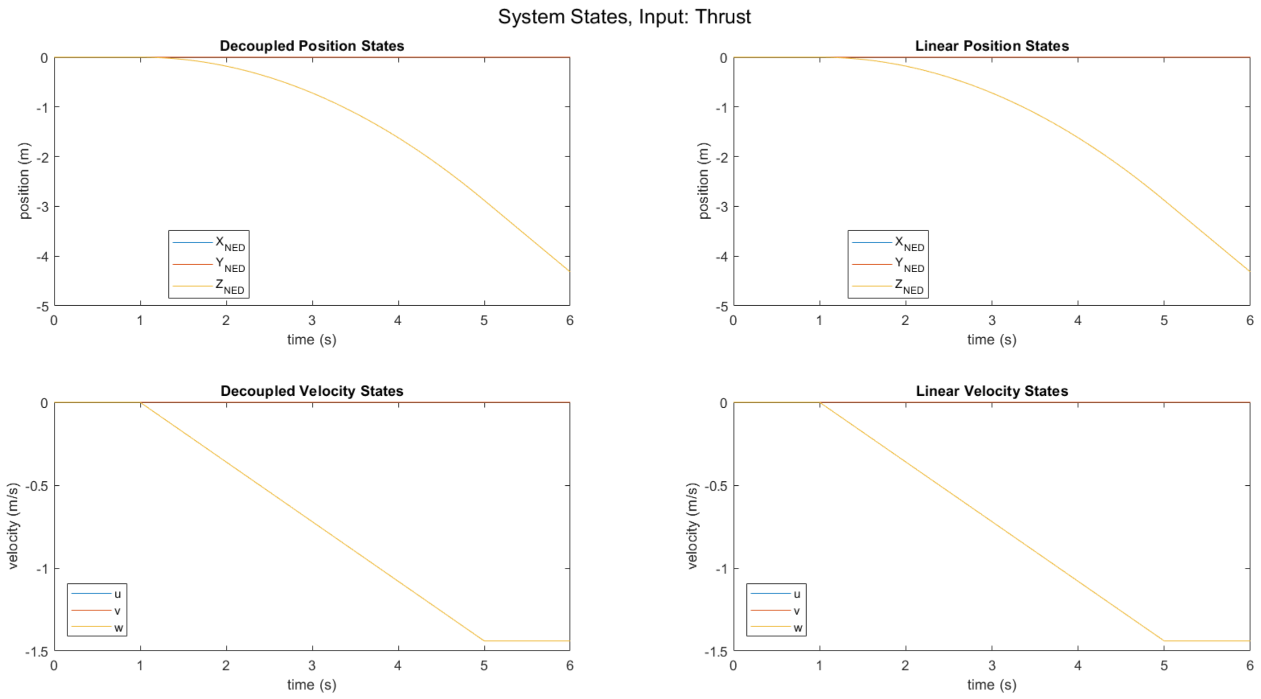
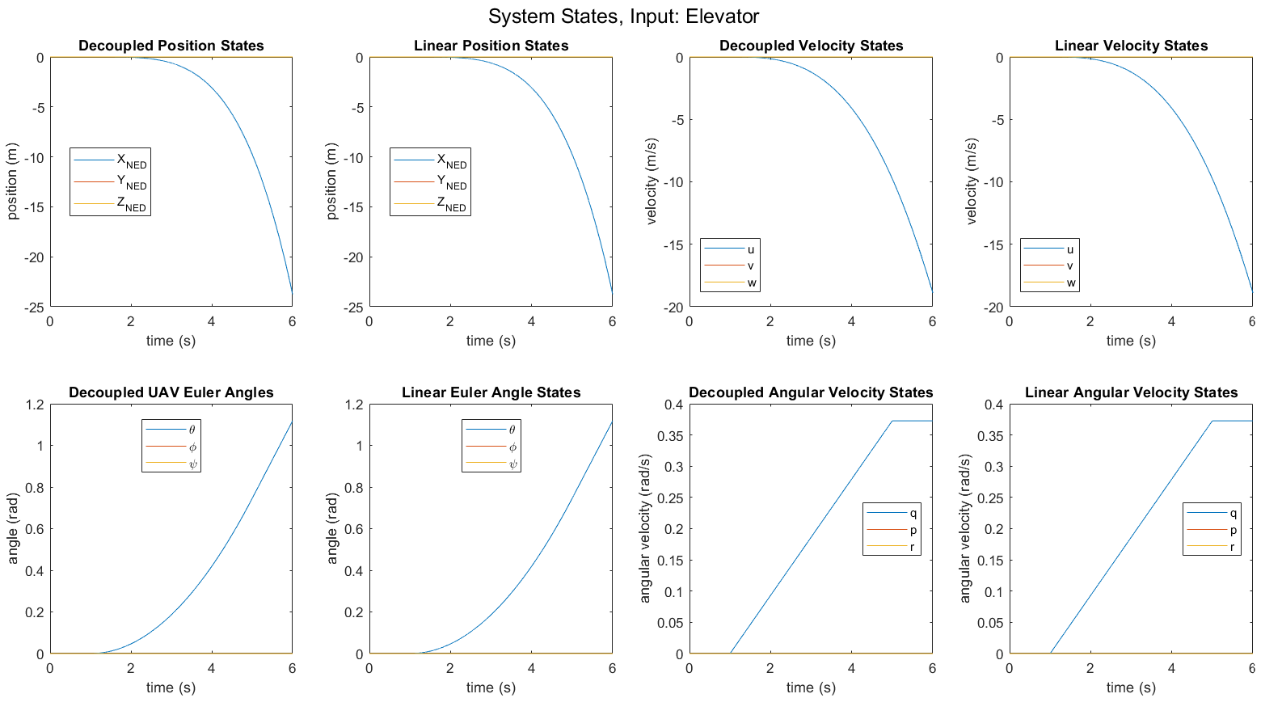

$\delta \dot{X}_{thrust} = A_{thrust}\,\delta X_{thrust} + B_{thrust}\,\delta T$
[dXt]
Athrust*u
[dT]
Bthrust*u
X_dot
1/s
X
[dXt]
$\delta \dot{X}_{elevator} = A_{elevator}\,\delta X_{elevator} + B_{elevator}\,\delta E$
[dXe]
Aelevator*u
[dE]
Belevator*u
X_dot
1/s
X
[dXe]
$\delta \dot{X}_{aileron} = A_{aileron}\,\delta X_{aileron} + B_{aileron}\,\delta A$
[dXa]
Aaileron*u
[dA]
Baileron*u
X_dot
1/s
X
[dXa]
$\delta \dot{X}_{rudder} = A_{rudder}\,\delta X_{rudder} + B_{rudder}\,\delta R$
[dXr]
Arudder*u
[dR]
Brudder*u
X_dot
1/s
X
[dXr]
[dXe]
[dXa]
[dXt]
[dXr]
out.decoupledstates

Figure 2: Decoupled Systems

Then, I ran the same code I used to compare the nonlinear and linear states, modified to compare the full set and decoupled systems. The results were that every state matched exactly. This is shown in Figure 3.


System States, Input: Thrust
Decoupled Position States
Linear Position States
position (m)
time (s)
X_{NED}
Y_{NED}
Z_{NED}
Decoupled Velocity States
Linear Velocity States
velocity (m/s)
u
v
w


System States, Input: Elevator
Decoupled Position States
Linear Position States
Decoupled Velocity States
Linear Velocity States
position (m)
velocity (m/s)
time (s)
X_{NED}
Y_{NED}
Z_{NED}
u
v
w
Decoupled UAV Euler Angles
Linear Euler Angle States
Decoupled Angular Velocity States
Linear Angular Velocity States
angle (rad)
angular velocity (rad/s)
θ
ϕ
ψ
q
p
r

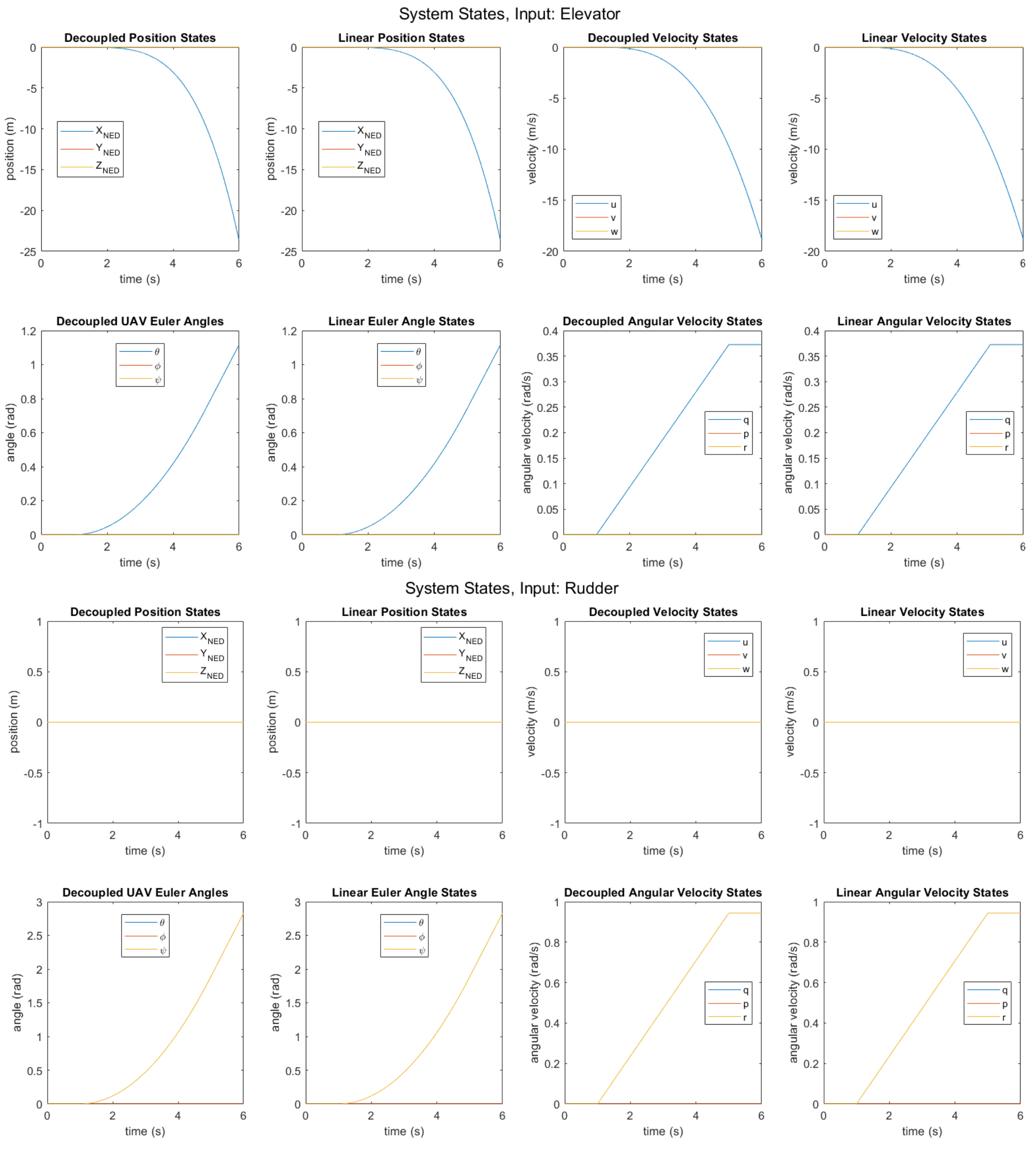


Figure 3: Decoupled and Full system Comparison

5 Conclusion

If we order our states intelligently, we can create a set of decoupled systems, each with a single input, that loses no information. The advantage of this is that we are able to isolate behaviors and modes of the UAV; analyzing a 12x12 system is difficult, but analyzing a double integrator is relatively simple. Instead of desiging a large, complex flight controller, we can instead design a set of more simple flight controllers for each subsystem.

Additionally, we are reducing the computational complexity of our model by creating smaller matrices, which may be relevant for the UAV's on board computer.