Aerospace Engineering Department

Faculty of Engineering

Cairo University

4th year

AER4420

Design of control systems for aeronautical and space vehicles

Prepared by team 11:

Student Name	S.N	B.N
احمد محمد عبد البديع عبدالرحمن	1	8
حامد احمد حامد احمد	1	20
عمرو ايهاب عبد الحميد عبد الفتاح	1	38
كريم عبدالعزيز عوض الله البوهى	1	40
يوسف احمد رمضان احمد	2	32

Submitted to:

Prof. Osama Mohamady

Eng. Mahmoud Gamal-Eldin Elewa

Table of Contents

Task 1: AUTOPILOT LITERATURE REVIEW

Part 1: Research Questions on Autopilot Systems

a) "What is an Autopilot?" i.e. define its main objective

- An autopilot is a system designed to control the trajectory of a vehicle, such as an aircraft or a ship, without constant manual input from a human operator. Its main objective is to automate the control of the vehicle's movement according to pre-programmed instructions or in response to real-time sensor data, thereby reducing the need for direct human intervention and allowing the operator to focus on other tasks or to intervene only when necessary.

b) When were the first autopilots invented, and what was their function?

- The concept of autopilots dates back to the early 20th century, with the first practical implementations occurring in the 1910s and 1920s. The first autopilots were primarily designed to help maintain stable flight by controlling the aircraft's pitch, roll, and yaw. One of the earliest documented autopilot systems was developed by Lawrence Sperry in 1914.

c) What are the inputs & outputs of an Autopilot system onboard an airplane from a control systems perspective?

- Inputs: Attitude and Heading Reference System (AHRS), Air Data System (ADS), Navigation System, Control Inputs.
- Outputs: Control Surfaces, Thrust, Autopilot Modes and Commands.

d) What would be the role of the pilot in an airplane equipped with an autopilot?

- The role of the pilot remains crucial for safe operation of the aircraft. While autopilot systems can automate many aspects of flight, pilots are responsible for overseeing the system, making critical decisions, and intervening when necessary.

e) What is the difference between an Autopilot & SAS (stability augmentation system)?

- Autopilot systems focus on automating navigation and flight management tasks, while SAS systems concentrate on enhancing stability and handling qualities, particularly in challenging or dynamic flight conditions.

f) What is the role of the onboard sensors like (GPS, gyroscopes, ..etc.)?

- Sensors include GPS (position, groundspeed), Gyroscopes (angular rates), Accelerometers (acceleration forces), AHRS (attitude), Air Data Sensors (airspeed, altitude), Magnetometers (magnetic heading).

g) What can we do if we need to know a state if it is not directly measured by a sensor?

- State Estimation Techniques, Observer Design, Sensor Fusion, System Identification.

h) What is a fly-by-wire flight control system?

- A fly-by-wire flight control system is a technological advancement in aircraft design where traditional mechanical control systems are replaced by electronic systems, allowing for more precise control and advanced features.

i) Find 2 open source autopilot software that can be used on UAVs

- ArduPilot, PX4 Autopilot.

j) Find 2 autopilot hardware that can be used on UAVs

- Pixhawk Autopilot, APM (ArduPilot Mega) Autopilot.

Part 2: Flight Mechanics

The general rigid body dynamics equations in 3D space:

Kinetics:

$$\sum \mathrm{F} = \frac{\mathrm{d}}{\mathrm{d}t}(m\mathrm{v})$$

$$\sum \mathrm{M} = \frac{\mathrm{d}}{\mathrm{d}t}(\mathrm{H})$$

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m\left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix}\right)$$

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = I\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I\begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

Kinematics:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = [J]\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \dfrac{\sin(\phi)}{\cos(\theta)} & \dfrac{\cos(\phi)}{\cos(\theta)} \end{bmatrix}\begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_E \\ \dot{y}_E \\ \dot{z}_E \end{bmatrix} = [T]_{EB}\begin{bmatrix} u \\ v \\ W \end{bmatrix}$$

$$= \begin{bmatrix} \mathrm{Cos}(\theta)\mathrm{Cos}(\psi) & -\mathrm{Cos}(\varphi)\mathrm{Sin}(\psi) + \mathrm{Sin}(\varphi)\mathrm{Sin}(\theta)\mathrm{Cos}(\psi) & \mathrm{Sin}(\varphi)\mathrm{Sin}(\psi) + \mathrm{Cos}(\varphi)\mathrm{Sin}(\theta)\mathrm{Cos}(\psi) \\ \mathrm{Cos}(\theta)\mathrm{Sin}(\psi) & \mathrm{Cos}(\varphi)\mathrm{Cos}(\psi) + \mathrm{Sin}(\varphi)\mathrm{Sin}(\varphi)\mathrm{Sin}(\psi) & -\mathrm{Sin}(\varphi)\mathrm{Cos}(\psi) + \mathrm{Cos}(\varphi)\mathrm{Sin}(\theta)\mathrm{Sin}(\psi) \\ -\mathrm{Sin}(\theta) & \mathrm{Sin}(\varphi)\mathrm{Cos}(\theta) & \mathrm{Cos}(\varphi)\mathrm{Cos}(\theta) \end{bmatrix}\begin{bmatrix} u \\ v \\ W \end{bmatrix}$$

Classification of the above equations:

- As I defined them above:
  The first six equations are kinetics, and last six equations are kinematics.

What are the assumptions introduced while deriving those equations?

- Mass (m) is constant ($dm/dt \ = 0$).
- The earth and atmosphere are fixed in inertial space.

The set of equations added to the (RBD) equations to form the Fixed wing Airplanes (EOM)

- Thrust, propulsive, Gravity and aerodynamic forces:

$$\begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + mg\begin{pmatrix} -\sin\ \theta \\ \cos\ \theta\sin\ \phi \\ \cos\ \theta\cos\ \phi \end{pmatrix}$$

$$\begin{pmatrix} \mathrm{L} \\ M \\ N \end{pmatrix} = \begin{pmatrix} I_x\dot{p} - I_{xz}\dot{r} \\ I_y\dot{q} \\ I_z\dot{r} - I_{xz}\dot{p} \end{pmatrix} + \begin{pmatrix} qr(I_z - Iy) - I_{xz}pq \\ pr(I_x - I_z) + I_{xz}(p^2 - r^2) \\ pq\left(I_y - Ix\right) + I_{xz}qr \end{pmatrix}$$

Classification of the airplanes EOM equations mathematically

All equations are 1st order non-linear ordinary differential equations.

Equations are coupled in lateral direction (rolling and yawing) and uncoupled in longitudinal direction (pitching).

The difference between the Body axes and the earth (or inertial) axes

The **difference** between the **Body axes** and the **Earth (or inertial) axes** lies in their reference frames and how they relate to an aircraft's motion:

I) **Body Axes**:

- The **body axes** are a set of three orthogonal axes (X_b, Y_b, Z_b) fixed to the aircraft itself.
- The X_b axis points forward along the aircraft's longitudinal axis (nose to tail).
- The Y_b axis points sideways along the aircraft's lateral axis (wingtip to wingtip).
- The Z_b axis points downward along the aircraft's vertical axis (from top to bottom).
- These axes move with the aircraft and are influenced by its attitude (yaw, pitch, and roll).
- The body axes are essential for describing aerodynamic forces and moments acting on the aircraft during flight.

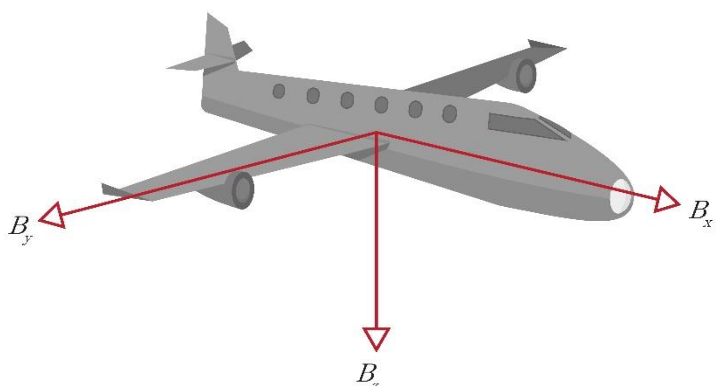


Figure 1: Body axes

II) **Earth (Inertial) Axes**:

- The **Earth (or inertial) axes** are a global reference frame fixed to the Earth.
- The X_e axis points north (towards the geographic North Pole).
- The Y_e axis points east (perpendicular to the X_e axis).
- The Z_e axis points upward (opposite to gravity).
- These axes remain fixed relative to the Earth and do not rotate with the aircraft.
- The Earth axes are useful for describing the absolute position and motion of the aircraft in space.

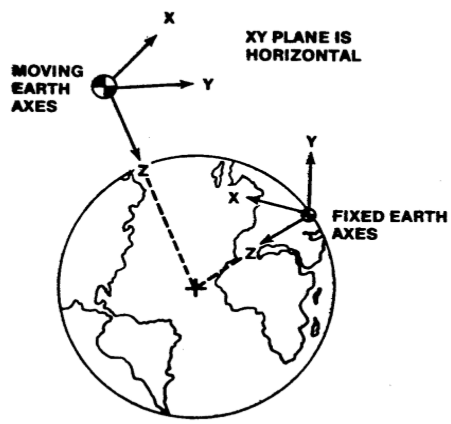


Figure 2: Earth axes

The difference between the pitch angle (θ) and the angle of attack (α), and between the sideslip angle (β) and the heading angle (ψ)

1. The difference between the pitch angle (θ) and the angle of attack (α)

I) **Pitch Angle (θ)**:

- The pitch angle refers to the orientation of an aircraft relative to the horizon.
- It is the angle between the longitudinal axis of the aircraft (from nose to tail) and the horizontal plane (the ground).
- In other words, it indicates whether the aircraft is pointing up or down with respect to the Earth's surface.
- Pilots control the pitch angle using the elevator (a control surface on the tail).

II) **Angle of Attack (α)**:

- The angle of attack is the angle between the chord line of an airfoil (such as a wing) and the relative wind.
- The relative wind is the airflow hitting the airfoil due to the aircraft's motion through the air.
- It determines how effectively the airfoil generates lift.
- A higher angle of attack increases lift, but beyond a certain point, it can lead to stalling (loss of lift).
- Pilots adjust the angle of attack by changing the pitch (nose-up or nose-down attitude).

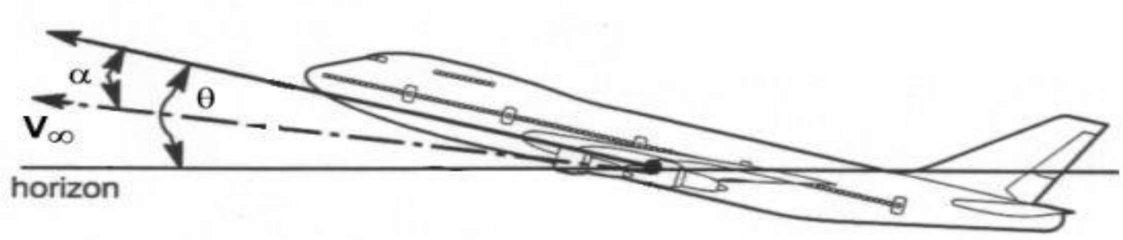


Figure 3: The difference between the pitch angle (θ) and the angle of attack (α)

2. The difference between the sideslip angle (β) and the heading angle (ψ)

I) **Sideslip Angle (β)**:

- The sideslip angle (also known as yaw angle) describes the deviation of an aircraft's flight path from its heading.
- It is the angle between the direction the aircraft is actually moving (known as the course angle, denoted as χ) and the direction it is pointing (the heading angle, denoted as ψ).
- In other words, β represents how much the aircraft is "slipping" or "skidding" sideways relative to its intended path.
- Pilots use rudder inputs to control the sideslip angle and maintain coordinated flight.

II) **Heading Angle (ψ)**:

- The heading angle is the orientation of the aircraft with respect to true north (or magnetic north, depending on the navigation system used).
- It is the angle between the longitudinal axis of the aircraft (from nose to tail) and the north direction.
- The heading angle provides information about the aircraft's overall orientation in the horizontal plane.

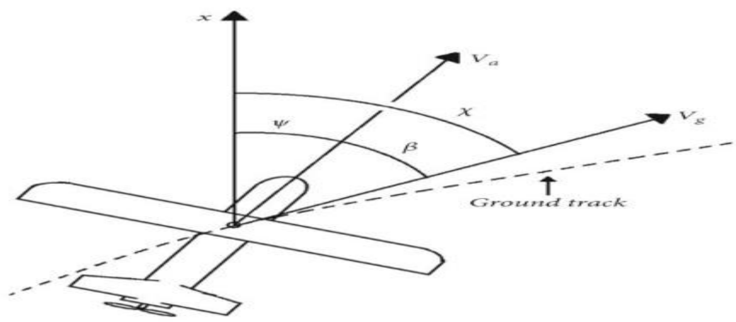


Figure 4: The difference between the sideslip angle (β) and the heading angle (ψ)

- Pilots adjust the heading angle using the ailerons and the rudder

Different attitude representations other than Euler angles such as Direction Cosine Matrix, quaternions and axis-angle representation advantages and disadvantages

1) Direction Cosine Matrix (DCM)

Advantages:

1. **Computational Tool from Linear Algebra**:
- The DCM (also known as the rotation matrix) is a powerful computational tool based on linear algebra.
- It directly applies to coordinate matrices (3-tuples of numbers) of vectors using matrix multiplication.
- It allows expressing a vector in one basis with respect to another (e.g., expressing a vector in the inertial frame using the DCM).
2. **Stability**:
- While the DCM can be complex to use, it provides stability.
- Stability ensures that the representation remains accurate even during numerical computations.

Disadvantages:

1. **Complexity and Lack of Intuition**:
- The DCM involves mathematical complexity.
- Understanding its properties and operations may require additional effort.
- Unlike Euler angles, which people are somewhat familiar with (roll, pitch, and yaw), the DCM lacks immediate intuition for visualizing rotations.
2. **Non-Uniqueness**:
- Similar to other rotation representations, the DCM is not unique.
- Different DCMs can represent the same rotation, leading to non-uniqueness.
- This can complicate inversion (finding a DCM corresponding to a given rotation matrix).
3. **Limited Interpretability**:
- While the DCM is stable, it lacks the ease of user interpretation and interaction.

- Users may find it challenging to mentally visualize and describe rotations using DCMs

2) Quaternions

Advantages:

1. Quaternions are more efficient and accurate for representing rotations than Euler angles because they avoid the Gimbal Lock problem.

2. Quaternions are compact, easily interpolated and their use does not cause numerical instability.

3. Quaternions provide a more intuitive way of representing rotational motion.

Disadvantages:

1. Quaternions are more complex to use than Euler angles.

2. Quaternions provide a different view of the world, which means some concepts, such as yaw, are defined differently than in Euler angles.

3. Quaternions cannot represent all possible rotations.

3) Axis-angle representation

Advantages:

1. **Concise Representation**: Only two numbers (the angle and the axis) are needed to define a rotation. This simplicity makes it convenient for characterizing rotations.
2. **Euler's Rotation Theorem**: The axis-angle representation is based on Euler's rotation theorem, which states that any rotation of a rigid body in three-dimensional space can be expressed as a pure rotation about a single fixed axis. This property simplifies the analysis of rotations.
3. **Useful in Rigid-Body Dynamics**: It is particularly useful when dealing with rigid-body dynamics, where understanding and characterizing rotations are essential.

Disadvantages:

1. **Non-Unique Representations**: Different axis-angle pairs can represent the same rotation. For example, a rotation vector of length $\theta + 2\pi M$ (for any integer M) encodes the same rotation as a vector of length θ. This non-uniqueness can complicate inversion (finding a rotation vector corresponding to a given rotation matrix).
2. **Not Intuitive for Visualization**: While the representation is concise mathematically, it may not be as intuitive for visualizing rotations compared to other representations like Euler angles or quaternions.

3. **Conversion Complexity**: Although it characterizes rotations well, applying the actual rotation often requires converting to other representations (such as quaternions or rotation matrices).

Part 3: Numerical solution of ODEs

Some of the numerical solving algorithms for ODEs

1) **Euler's Method**: This is the simplest numerical method for solving differential equations. It involves discretizing time and updating the state variables (position, velocity, etc.) based on the derivatives of these variables at each time step.
2) **Runge-Kutta Methods**: These are more accurate than Euler's method and come in various orders (e.g., RK2, RK4, RK45). RK4, the fourth-order Runge-Kutta method, is particularly popular for its balance of accuracy and computational efficiency.
3) **Finite Element Methods (FEM)**: FEM discretizes the aircraft's structure and aerodynamics into finite elements, allowing for the solution of complex EOM by solving algebraic equations. FEM is commonly used in structural analysis and coupled with computational fluid dynamics for aerodynamic analysis.
4) **Backward Differentiation Formulas (BDF)**: These implicit methods are often used for stiff differential equations, where the dynamics of the system change rapidly. They are more computationally expensive but can provide accurate solutions for stiff systems.

Using **Runge-Kutta** method (RK4) for solving the Airplanes EOM

One commonly used algorithm for solving the equations of motion (EOM) for airplanes is the **Runge-Kutta** method, specifically the fourth-order **Runge-Kutta** method (RK4). This method is a numerical technique used to solve ordinary differential equations (ODEs), which are often used to describe the motion of aircraft.

Initial conditions needed:

1. Initial position (x, y, z)
2. Initial velocity components (vx, vy, vz)
3. Initial Euler angles (roll, pitch, yaw)
4. Mass of the airplane
5. Aerodynamic coefficients
6. Atmospheric conditions (density, temperature, pressure, wind speed, etc.)
7. Time step size for the integration

Inputs needed in each iteration:

1. Current state variables (position, velocity, Euler angles)
2. Time step size

Outputs calculated in each iteration:

1. Updated position (x, y, z)
2. Updated velocity components (vx, vy, vz)
3. Updated Euler angles (roll, pitch, yaw)
4. Forces and moments acting on the airplane (lift, drag, thrust, weight, moments due to control surfaces)
5. Aerodynamic forces and moments (calculated based on the current state variables and aerodynamic coefficients)
6. Accelerations in each direction (ax, ay, az)
7. Angular rates (p, q, r)

The RK4 algorithm iteratively calculates the state variables at each time step using a weighted average of several slope estimates calculated at different points within the time step. These slopes are derived from the differential equations governing the motion of the airplane. By repeatedly applying this process, the algorithm produces a numerical approximation of the solution to the equations of motion over a specified time interval.

The RK4 method for simultaneous differential equations:

$$y_{n+1} = y_n + \frac{1}{6} * dt * (k_1 + 2k_2 + 2k_3 + k_4)$$

$$t_{n+1} = t_n + dt$$

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{dt}{2}, y_n + dt\frac{k_1}{2}\right)$$

$$k_3 = f\left(t_n + \frac{dt}{2}, y_n + dt\frac{k_2}{2}\right)$$

$$k_4 = f(t_n, y_n + dt)$$

Where “h” is time step and $(k1, k2, k3, k4)$ are slopes derived from the differential equations governing the motion of the airplane.

Solving the system of first-order ODEs

$$\frac{dy_1}{dt} = \cos(t) + \cos(y_1) + \sin(y_2)$$

$$\frac{dy_2}{dt} = \sin(t) + \sin(y_2)$$

The Initial conditions are:

$$\text{at } t = 0, \qquad \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

End of solution interval:

$$t_f = 20$$

Number of intervals:

$$n = 100$$

Matlab code

```
function [time_array,sol] = ode_rk4_solver(ode_eq,time_array,init_val)
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
n=length(time_array);
sol(:,1)=init_val;
dt=(time_array(n)-time_array(1))/n;
for i =1:n
 k1=ode_eq(time_array(i),sol(:,i));
 k2=ode_eq(time_array(i)+dt/2,sol(:,i)+k1*dt/2);
 k3=ode_eq(time_array(i)+dt/2,sol(:,i)+k2*dt/2);
 k4=ode_eq(time_array(i)+dt,sol(:,i)+k3*dt);
 sol(:,i+1)=sol(:,i)+dt/6*(k1+2*k2+2*k3+k4);
```

```
end
sol=sol(:,1:n);
end
```

Clear section

```
clc;clearvars;close all;
```

Initializing the input values

```
y_init=[-1 ; 1];

dy_dt=@(t,y) [cos(t)+cos(y(1))+sin(y(2)) ; sin(t)+sin(y(2))];

n=100;

t_init=0;

t_final=20;

dt=(-t_init+t_final)/n;

t_v=(t_init:dt:t_final);
```

Using our function and the ode45 function in matlab to get the solution

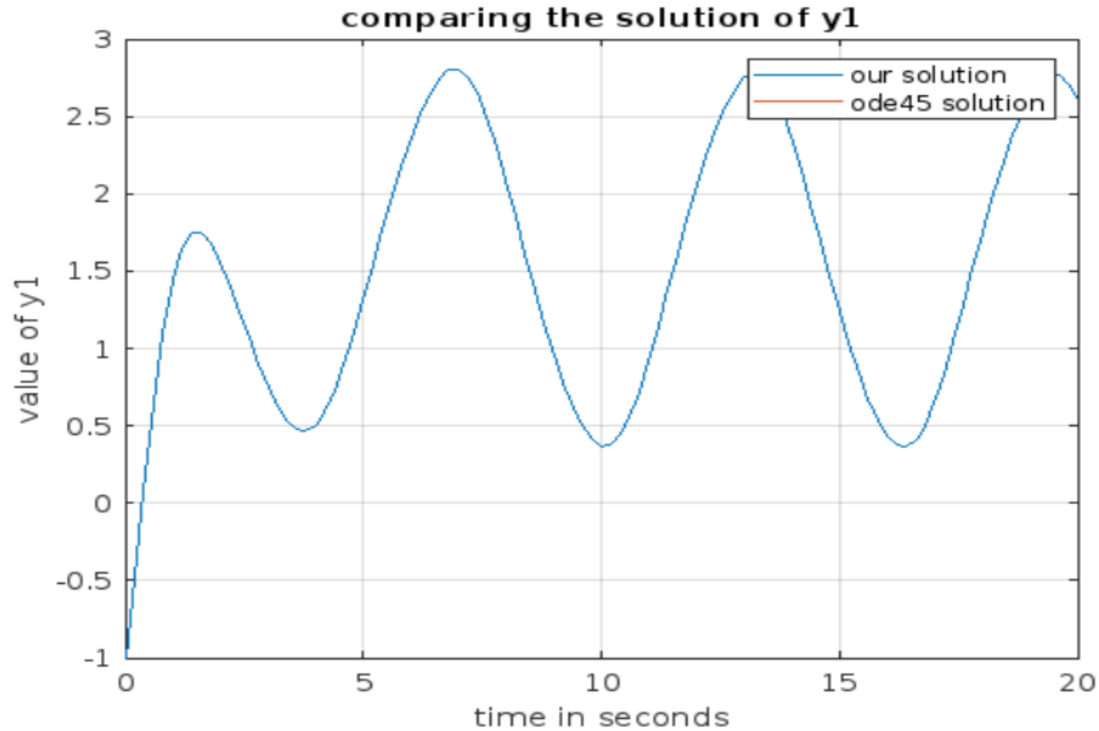
```
[our_t,our_sol] = ode_rk4_solver(dy_dt,t_v,y_init);

[ode45_t,ode45_sol] = ode45(dy_dt,t_v,y_init);
```

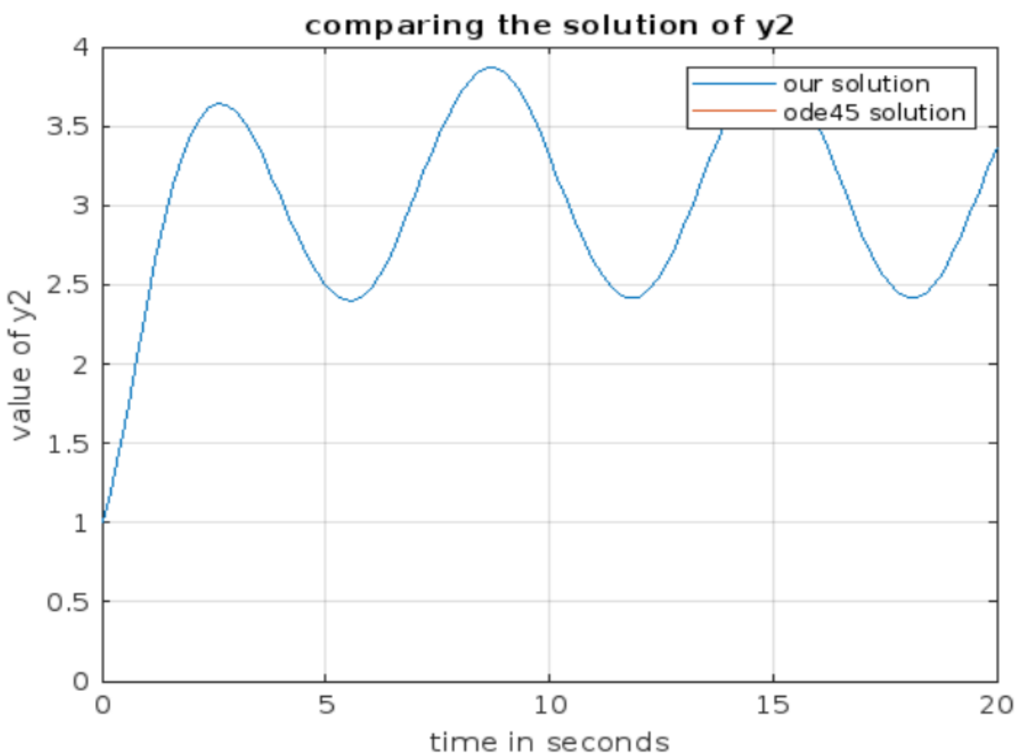
Graphing section to compare results obtained from the two solutions

```
figure(1)

plot(t_v,our_sol(1,:))

hold on;grid;
```

```
plot(t_v,ode45_t(1,:))
title("comparing the solution of y1")
legend(["our solution","ode45 solution"])
xlabel("time in seconds")
ylabel("value of y1")

figure(2)

plot(t_v,our_sol(2,:))
hold on;grid;
plot(t_v,ode45_t(2,:))
title("comparing the solution of y2")
legend(["our solution","ode45 solution"])
xlabel("time in seconds")
ylabel("value of y2")
```

Comparing the results of our solution with ode45 solution


comparing the solution of y1
our solution
ode45 solution
value of y1
time in seconds
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
5
10
15
20


comparing the solution of y2
our solution
ode45 solution
value of y2
time in seconds
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
20

Task 2: NUMERICAL SOLUTION OF ODE (RK4) AIRPLANE SIMULATOR PART I

a) Solving the Rigid body dynamics (RBD) equations for given constant values of forces & moments, using "Runge-Kutta" 4th order method

$$t_{final} = 25\ sec$$

$$Forces = [8; 5; 4]\ N$$

$$Moments = [5; 10; 20]\ N.m$$

$$mass = 10\ kg$$

$$I = \begin{bmatrix} 1 & -2 & -1 \\ -2 & 5 & -4 \\ -1 & -4 & 0.2 \end{bmatrix} kg.m^2$$

Initial conditions ($S.I.\ units$):

$$[u, v, w, p, q, r, \phi, \theta, \psi, x, y, z]_{t=0} = \left[10{,}2{,}0, \frac{2\pi}{180}, \frac{\pi}{180}, 0, \frac{20\pi}{180}, \frac{15\pi}{180}, \frac{30\pi}{180}, 2{,}4{,}7\right]$$

RBD Equations in vector form:

Kinetics:

$$\sum \mathrm{F} = \frac{\mathrm{d}}{\mathrm{d}t}(m\mathrm{v})$$

$$\sum \mathrm{M} = \frac{\mathrm{d}}{\mathrm{d}t}(\mathrm{H})$$

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m\left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix}\right)$$

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = I\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I\begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

Kinematics:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = [J]\begin{bmatrix}p\\ q\\ r\end{bmatrix} = \begin{bmatrix}1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta)\\ 0 & \cos(\phi) & -\sin(\phi)\\ 0 & \dfrac{\sin(\phi)}{\cos(\theta)} & \dfrac{\cos(\phi)}{\cos(\theta)}\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}$$

$$\begin{bmatrix}\dot{x}_E\\ \dot{y}_E\\ \dot{z}_E\end{bmatrix} = [T]_{EB}\begin{bmatrix}u\\ v\\ W\end{bmatrix}$$

$$= \begin{bmatrix}\mathrm{Cos}(\theta)\mathrm{Cos}(\psi) & -\mathrm{Cos}(\varphi)\mathrm{Sin}(\psi) + \mathrm{Sin}(\varphi)\mathrm{Sin}(\theta)\mathrm{Cos}(\psi) & \mathrm{Sin}(\varphi)\mathrm{Sin}(\psi) + \mathrm{Cos}(\varphi)\mathrm{Sin}(\theta)\mathrm{Cos}(\psi)\\ \mathrm{Cos}(\theta)\mathrm{Sin}(\psi) & \mathrm{Cos}(\varphi)\mathrm{Cos}(\psi) + \mathrm{Sin}(\varphi)\mathrm{Sin}(\varphi)\mathrm{Sin}(\psi) & -\mathrm{Sin}(\varphi)\mathrm{Cos}(\psi) + \mathrm{Cos}(\varphi)\mathrm{Sin}(\theta)\mathrm{Sin}(\psi)\\ -\mathrm{Sin}(\theta) & \mathrm{Sin}(\varphi)\mathrm{Cos}(\theta) & \mathrm{Cos}(\varphi)\mathrm{Cos}(\theta)\end{bmatrix}\begin{bmatrix}u\\ v\\ W\end{bmatrix}$$

The "Runge-Kutta 4" method for simultaneous differential equations:

$$y_{n+1} = y_n + \frac{1}{6} * dt * (k_1 + 2k_2 + 2k_3 + k_4)$$

$$t_{n+1} = t_n + dt$$

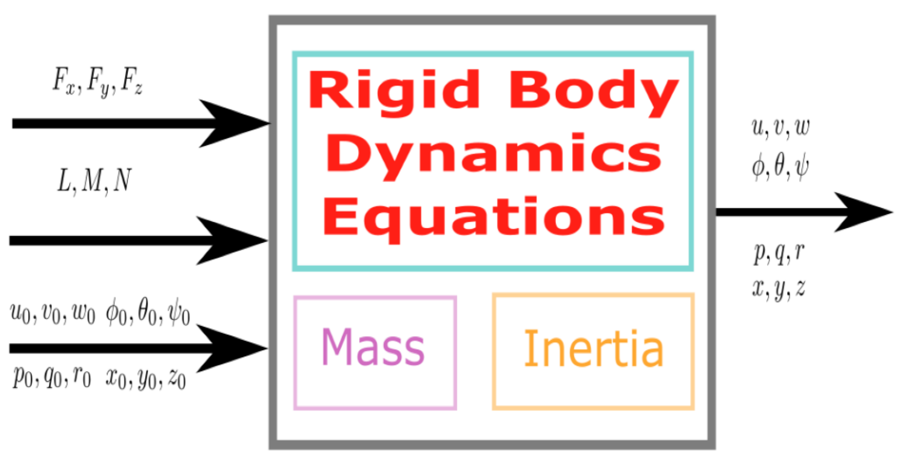
$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{dt}{2}, y_n + dt\frac{k_1}{2}\right)$$

$$k_3 = f\left(t_n + \frac{dt}{2}, y_n + dt\frac{k_2}{2}\right)$$

$$k_4 = f(t_n, y_n + dt)$$

RBD Solver overview:



Inputs to the RBD solver code:

- $Forces\ and\ Moments\ [F_x, F_y,\ F_z, L, M, N]$.
- $Initial\ conditions\ for\ all\ 12\ states\ [u, v, w, p, q, r, \phi, \theta, \psi, x, y, z]$.

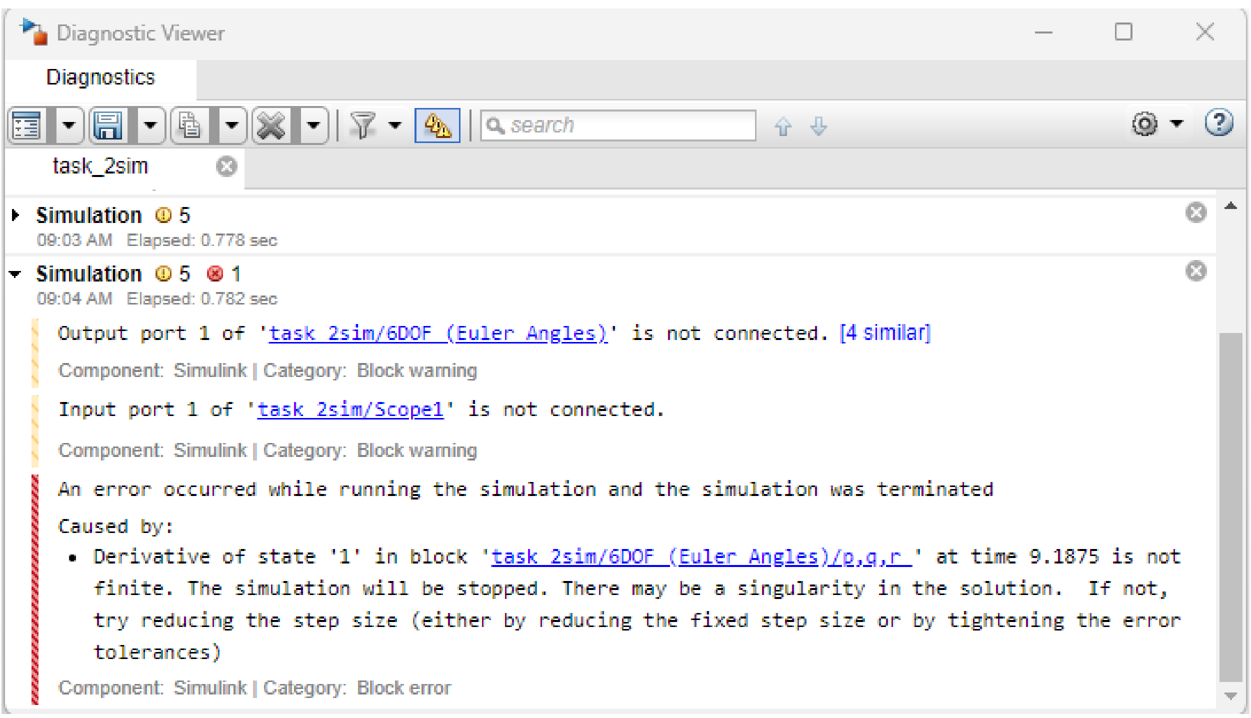
Outputs from the RBD solver code:

- $All\ 12\ states\ at\ the\ next\ timestep\ [u, v, w, p, q, r, \phi, \theta, \psi, x, y, z]$.

Constants to be defined:

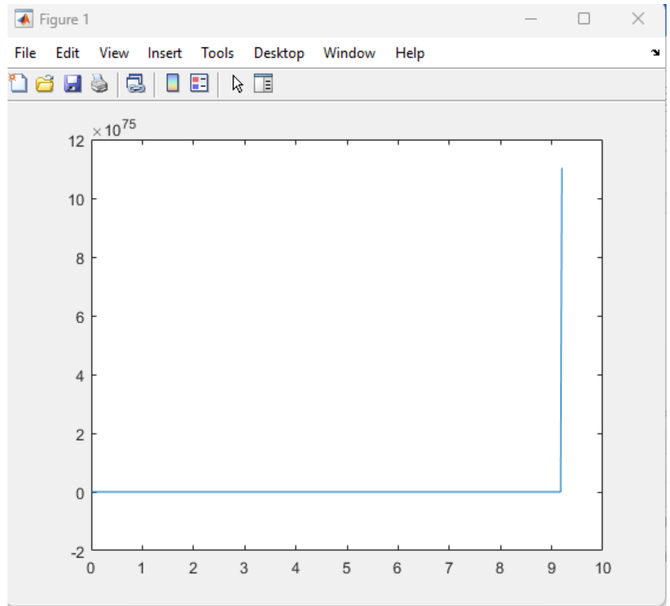
- $Mass\ (m)$
- $Inertia\ matrix\ (I)$
- $Time\ interval: (t_{final}\&\ timestep)$

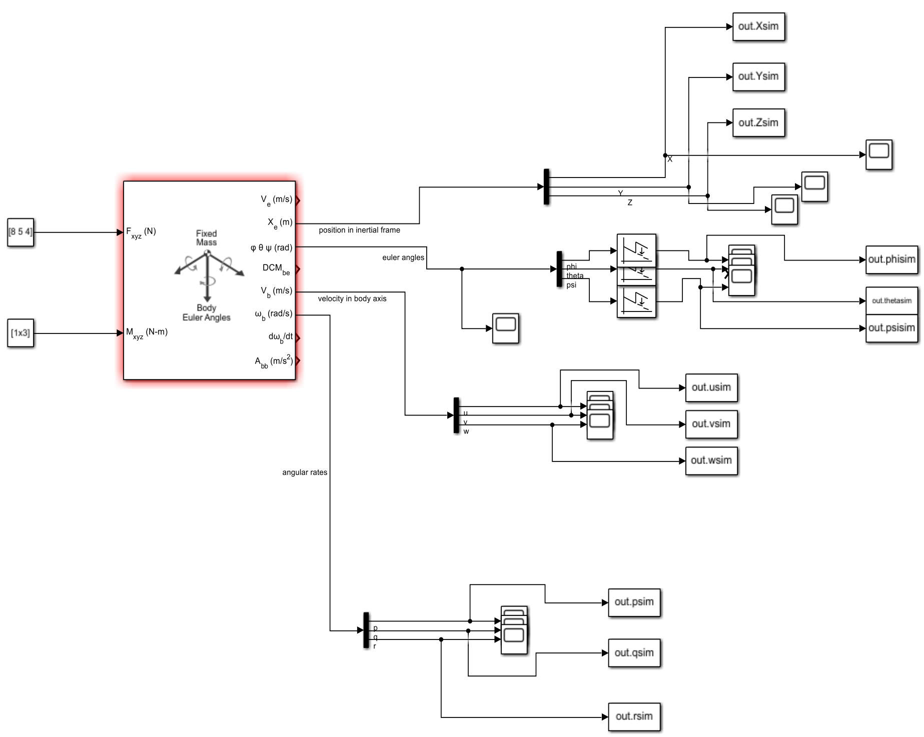
The importance of number of points in our scheme (the value of dt):



$We\ tried\ making\ n = 1000\ that\ means\ dt = 0.025$, Simulink showed the following error:

The error said that state 1 at time $9.1875\ sec$ goes to infinity that means the rk4 scheme failed to converge to a solution, by more investigation we used our function that we constructed at the previous task to draw a graph of the answer for u values, and the value of u approaches infinity just after the 9th second, the solution of that problem is to increase the number of points in time so decreasing the value of dt , $we\ choose\ n = 10000\ and\ dt\ = 0.0025$ to show the solutions provided in this report.

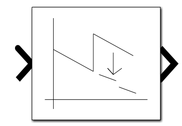




Simulink model

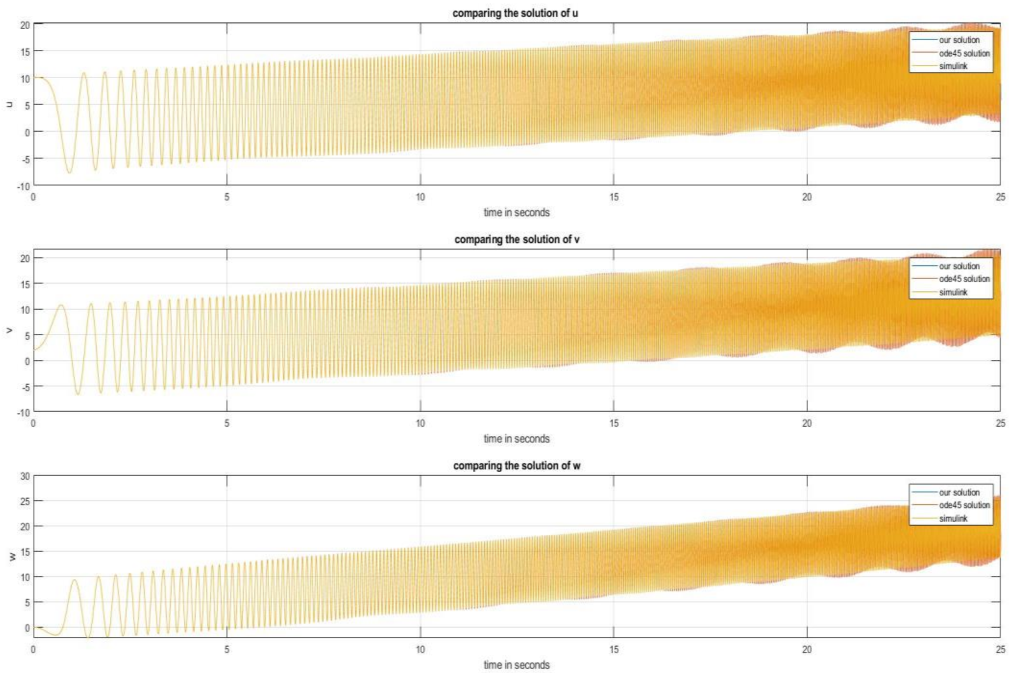
Note:

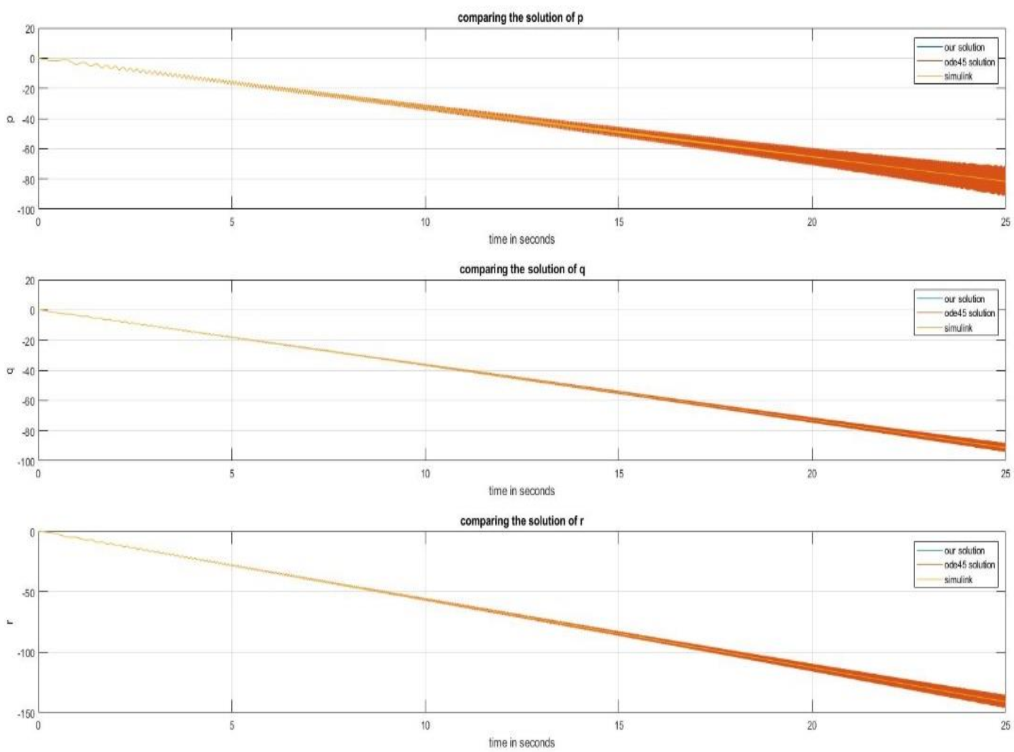
- We used the “unwrap block” which unwraps each channel of the input by adding or subtracting appropriate multiples of 2π to each channel element. The block recognizes phase discontinuities larger than the value of the Tolerance parameter.

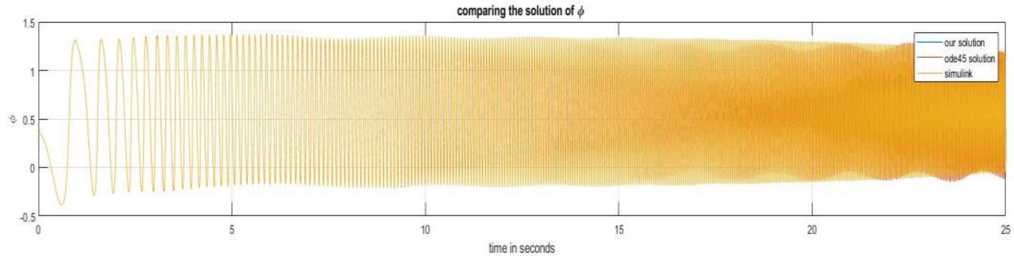


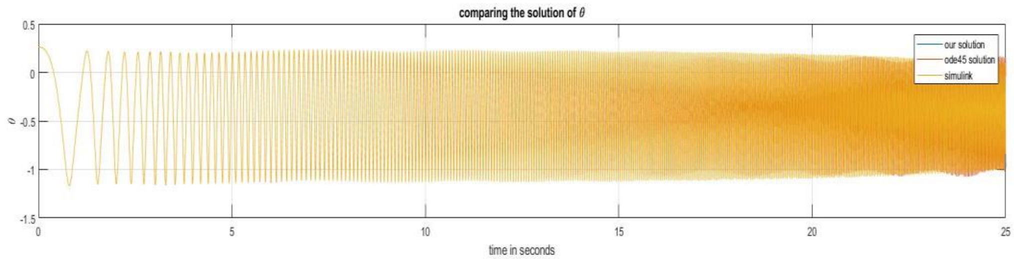
- **Phase unwrap** is a process often used to reconstruct a signal's original phase. Unwrap algorithms add appropriate multiples of 2π to each phase input to restore original phase values.
- Algorithms that compute the phase of a signal often only output phases $between -\pi\ and\ \pi$. For instance, such algorithms compute the phase of $sin(2\pi\ +\ 3)\ to\ be\ 3, since\ sin(3)\ =\ sin(2\pi\ +\ 3)$, and since the actual phase, $2\pi\ +\ 3, is\ not\ between -\pi\ and\ \pi$. Such algorithms compute the phases of $sin(-4\pi\ +\ 3)\ and\ sin(16\pi\ +\ 3)\ to\ be\ 3\ as\ well$.

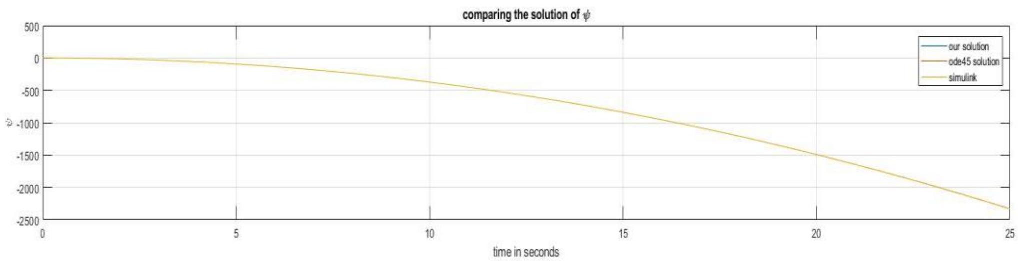
b) Comparing the results of our solution with ode45 solution and Simulink solution

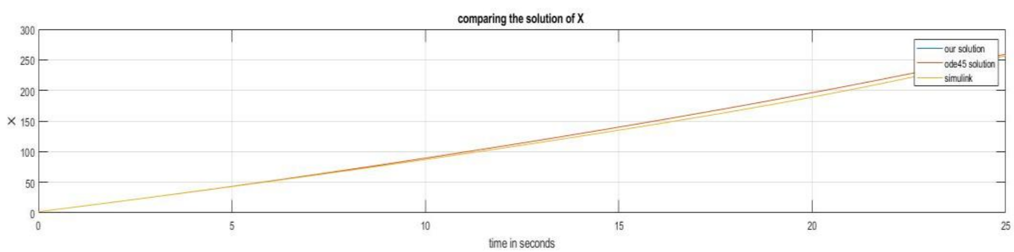


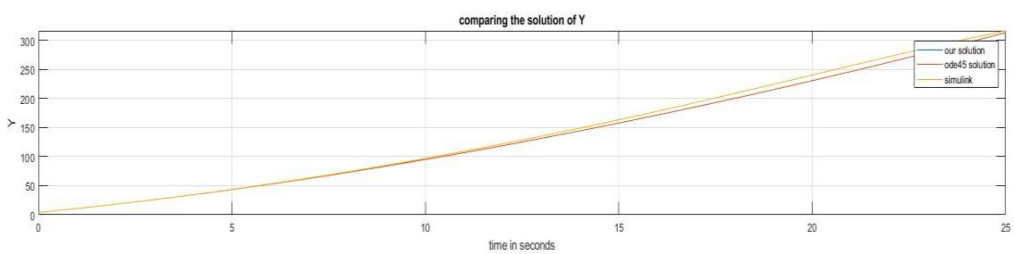


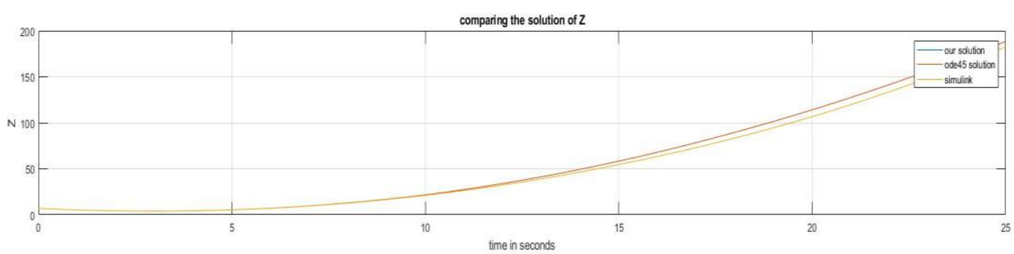

comparing the solution of ϕ
our solution
ode45 solution
simulink
ϕ
time in seconds


comparing the solution of θ
our solution
ode45 solution
simulink
θ
time in seconds


comparing the solution of ψ
our solution
ode45 solution
simulink
ψ
time in seconds

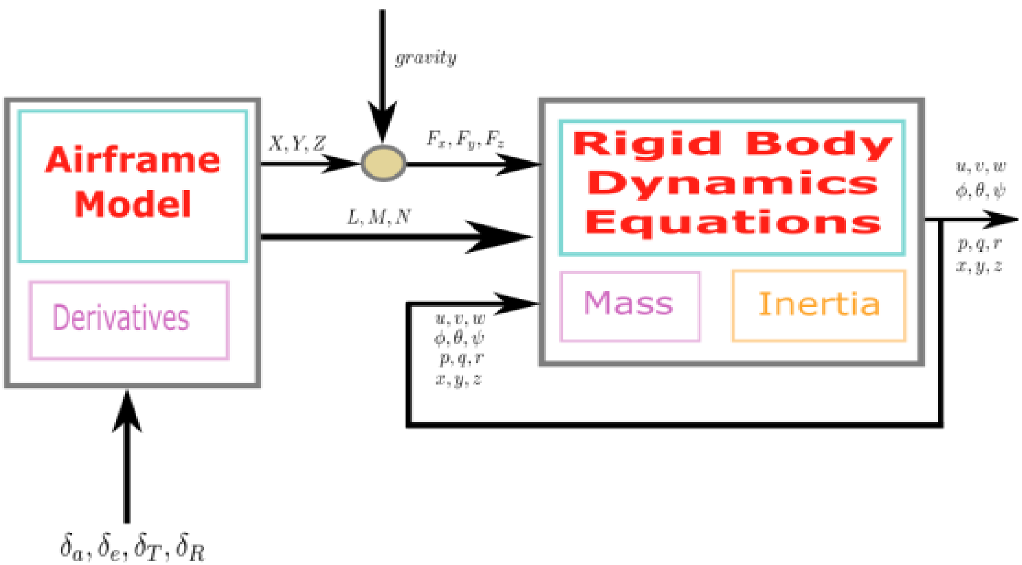

comparing the solution of X
our solution
ode45 solution
simulink
X
time in seconds


comparing the solution of Y
our solution
ode45 solution
simulink
Y
time in seconds


comparing the solution of Z
our solution
ode45 solution
simulink
Z
time in seconds

Task 3: AIRFRAME MODEL: AIRPLANE SIMULATOR PART II

A) OVERVIEW



IN the Second task we study The Rigid Body Dynamics Equations solver But It is On general vehicle, so if we need to study specific vehicle we will add the Airframe model.

Note:

1- we add gravity force to forces of the body.

2- we calculate the control action (input of Airframe model) from controller

B) Forces

The following set of linear equations represents the change in the Aerodynamic & thrust forces & moments, they are function of:

• Stability Derivatives.

• Control Derivatives.

• The perturbation changes in the states and the control surfaces deflections from their values at the trim condition.

$$\Delta X = \frac{\partial X}{\partial u}\Delta u + \frac{\partial X}{\partial w}\Delta w + \frac{\partial X}{\partial \delta_e}\Delta \delta_e + \frac{\partial X}{\partial \delta_T}\Delta \delta_T$$
$$\Delta Y = \frac{\partial Y}{\partial v}\Delta v + \frac{\partial Y}{\partial p}\Delta p + \frac{\partial Y}{\partial r}\Delta r + \frac{\partial Y}{\partial \delta_r}\Delta \delta_r$$
$$\Delta Z = \frac{\partial Z}{\partial u}\Delta u + \frac{\partial Z}{\partial w}\Delta w + \frac{\partial Z}{\partial \dot{w}}\Delta \dot{w} + \frac{\partial Z}{\partial q}\Delta q + \frac{\partial Z}{\partial \delta_e}\Delta \delta_e + \frac{\partial Z}{\partial \delta_T}\Delta \delta_T$$
$$\Delta L = \frac{\partial L}{\partial v}\Delta v + \frac{\partial L}{\partial p}\Delta p + \frac{\partial L}{\partial r}\Delta r + \frac{\partial L}{\partial \delta_r}\Delta \delta_r + \frac{\partial L}{\partial \delta_a}\Delta \delta_a$$
$$\Delta M = \frac{\partial M}{\partial u}\Delta u + \frac{\partial M}{\partial w}\Delta w + \frac{\partial M}{\partial \dot{w}}\Delta \dot{w} + \frac{\partial M}{\partial q}\Delta q + \frac{\partial M}{\partial \delta_e}\Delta \delta_e + \frac{\partial M}{\partial \delta_T}\Delta \delta_T$$
$$\Delta N = \frac{\partial N}{\partial v}\Delta v + \frac{\partial N}{\partial p}\Delta p + \frac{\partial N}{\partial r}\Delta r + \frac{\partial N}{\partial \delta_r}\Delta \delta_r + \frac{\partial N}{\partial \delta_a}\Delta \delta_a$$

Note:

- $\Delta X, \Delta Y, \Delta Z, \Delta L, \Delta M, \Delta N$ are the changes in the forces & moments, i.e. these are not the absolute values of the forces and moments. They should be added to the reference values at the trim condition $X_0, Y_0, Z_0, L_0, M_0, N_0$ to calculate the absolute values X, Y, Z, L, M, N.
- Similarly, $\Delta u, \Delta v, \Delta w, \ldots$ are the changes in the states values from their values at the reference condition $\Delta u = u - u_0, \Delta v = v - v_0, \Delta w = w - w_0, \ldots$
- There should be linearization point
- Inputs and outputs of the (Airframe Model) are perturbations from the reference values.
- Inputs and outputs of the RBD are absolute values.

So: "Perturbation values resulting from the (Airframe Model) should be added to the reference values before passing them to the (RBD) and the absolute values resulting from the (RBD) should be converted to perturbation values by subtracting the reference values from them."

The total forces acting on an airplane are:

- Aerodynamic forces.
- Thrust force.
- Gravity force.

$$X - \mathrm{mgsin}\ \theta = m(\dot{u}^E + qw^E - rv^E)$$
$$Y + \mathrm{mgcos}\ \theta\mathrm{sin}\ \phi = m(\dot{v}^E + ru^E - pw^E)$$
$$Z + \mathrm{mgcos}\ \theta\mathrm{cos}\ \phi = m(\dot{w}^E + pv^E - qu^E)$$

Equilibrium state:

Initially at the reference flight condition the airplane is in an equilibrium state, which means:

$$\sum \text{ Forces } = 0\ \&\ \sum M \text{oments } = 0$$
$$X_0 - mg\mathrm{sin}\ \theta_0 = 0 \rightarrow X_0 = mg\mathrm{sin}\ \theta_0$$
$$Y_0 - mg\mathrm{cos}\ \theta_0\mathrm{sin}\ \emptyset_0 = 0 \rightarrow Y_0 = -mg\mathrm{cos}\ \theta_0\mathrm{sin}\ \emptyset_0$$
$$Z_0 - mg\mathrm{cos}\ \theta_0\mathrm{cos}\ \emptyset_0 = 0 \rightarrow Z_0 = -mg\mathrm{cos}\ \theta_0\mathrm{cos}\ \emptyset_0$$
$$\therefore X = X_0 + \Delta X = \Delta X + mg\mathrm{sin}\ \theta_0$$
$$Y = Y_0 + \Delta Y = \Delta Y - mg\mathrm{cos}\ \theta_0\mathrm{sin}\ \emptyset_0$$
$$Z = Z_0 + \Delta Z = \Delta Z - mg\mathrm{cos}\ \theta_0\mathrm{cos}\ \emptyset_0$$

And the total force acting on the airplane (this value is the input which you will give to the RBD)

$$F_x = X - mg\mathrm{sin}\ \theta = \Delta X + mg\mathrm{sin}\ \theta_0 - mg\mathrm{sin}\ \theta$$
$$F_y = Y + mg\mathrm{cos}\ \theta\mathrm{sin}\ \emptyset = \Delta Y + mg\mathrm{cos}\ \theta\mathrm{sin}\ \emptyset$$
$$F_z = Z + mg\mathrm{cos}\ \theta\mathrm{cos}\ \emptyset = \Delta Z - mg\mathrm{cos}\ \theta_0\mathrm{cos}\ \emptyset_0 + mg\mathrm{cos}\ \theta\mathrm{sin}\ \emptyset$$

$$\begin{Bmatrix} F_x \\ F_y \\ F_z \\ L \\ M \\ N \end{Bmatrix} = \begin{Bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \\ \Delta L \\ \Delta M \\ \Delta N \end{Bmatrix} + \begin{Bmatrix} \mathrm{X}_0 \\ \mathrm{Y}_0 \\ \mathrm{Z}_0 \\ L_0 \\ M_0 \\ N_0 \end{Bmatrix} + \begin{Bmatrix} -mg\sin\,\theta \\ mg\cos\,\theta\sin\,\emptyset \\ mg\cos\,\theta\cos\,\emptyset \\ 0 \\ 0 \end{Bmatrix}$$

$$\begin{Bmatrix} X_0 \\ Y_0 \\ Z_0 \end{Bmatrix} = \begin{Bmatrix} mg\sin\,\theta_0 \\ -mg\cos\,\theta_0\sin\,\emptyset_0 \\ -mg\cos\,\theta_0\cos\,\emptyset_0 \end{Bmatrix} \& \begin{Bmatrix} L_0 \\ M_0 \\ N_0 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix}$$

C) Bode axes

- Body axis at Cg
- XZ plane is a planned symmetry ($I_{xy} = I_{yz} = 0$)
- 3 axes are orthogonal.

Note: we have types of body axes (principal axes – stability axes – body axes) and choose body axes make calculate of derivative easy.

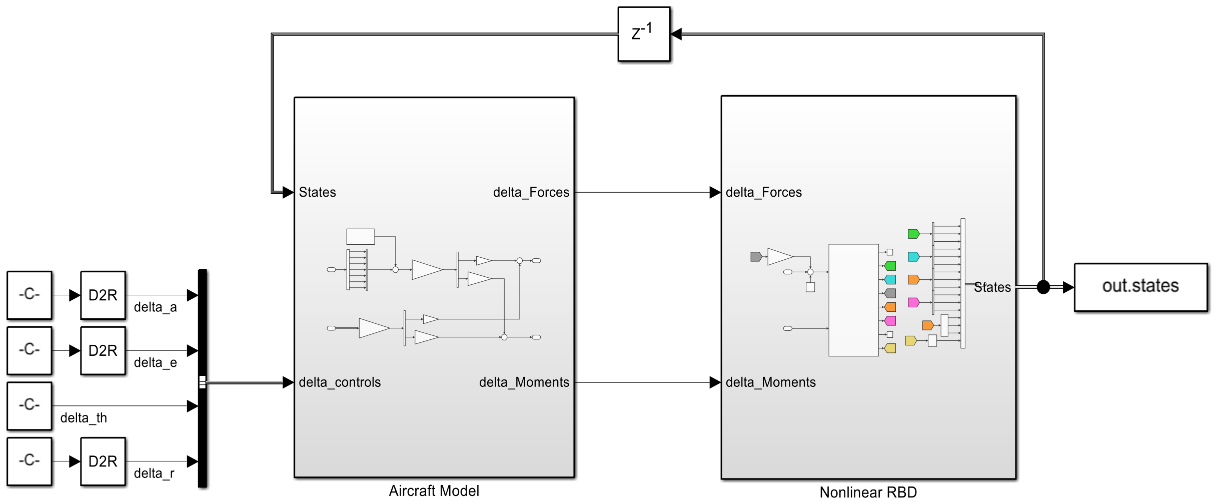
D) Airplane simulator

From the task 2 we use block RBD, and the inputs of the block are forces and moments and the outputs are states of plane,… so in this task we will add another block called Airframe model and its inputs are Control actions and the states we got from RBD block, and its outputs are the forces and moments which the inputs of RBD block.

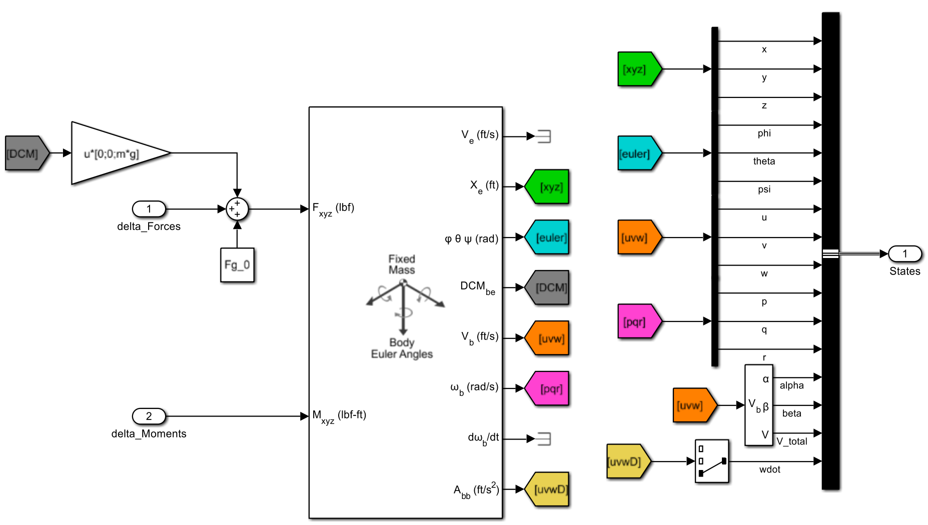
1)Simulink model

Airplane Simulator

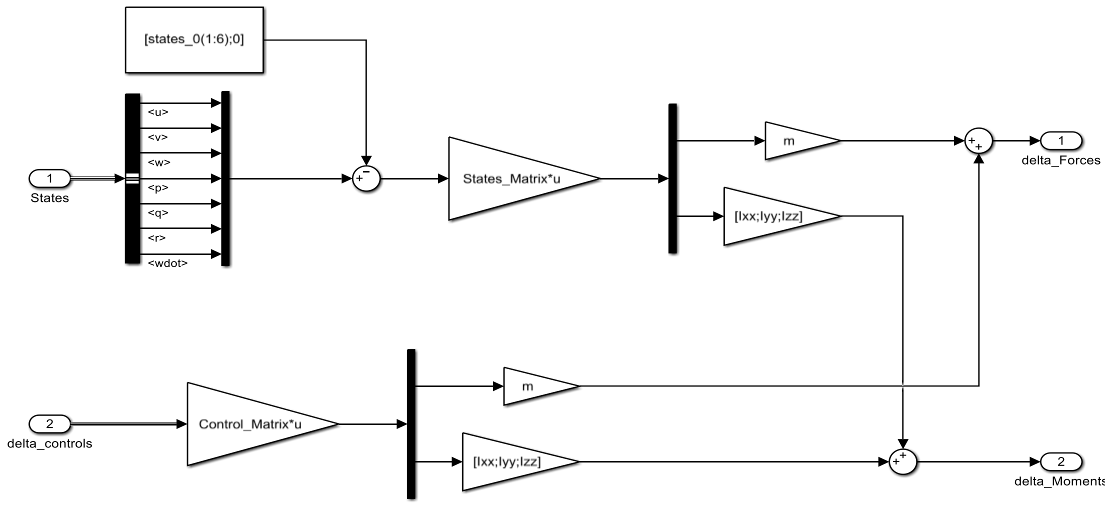
We use block z^{-1} for delay.



Nonlinear RBD

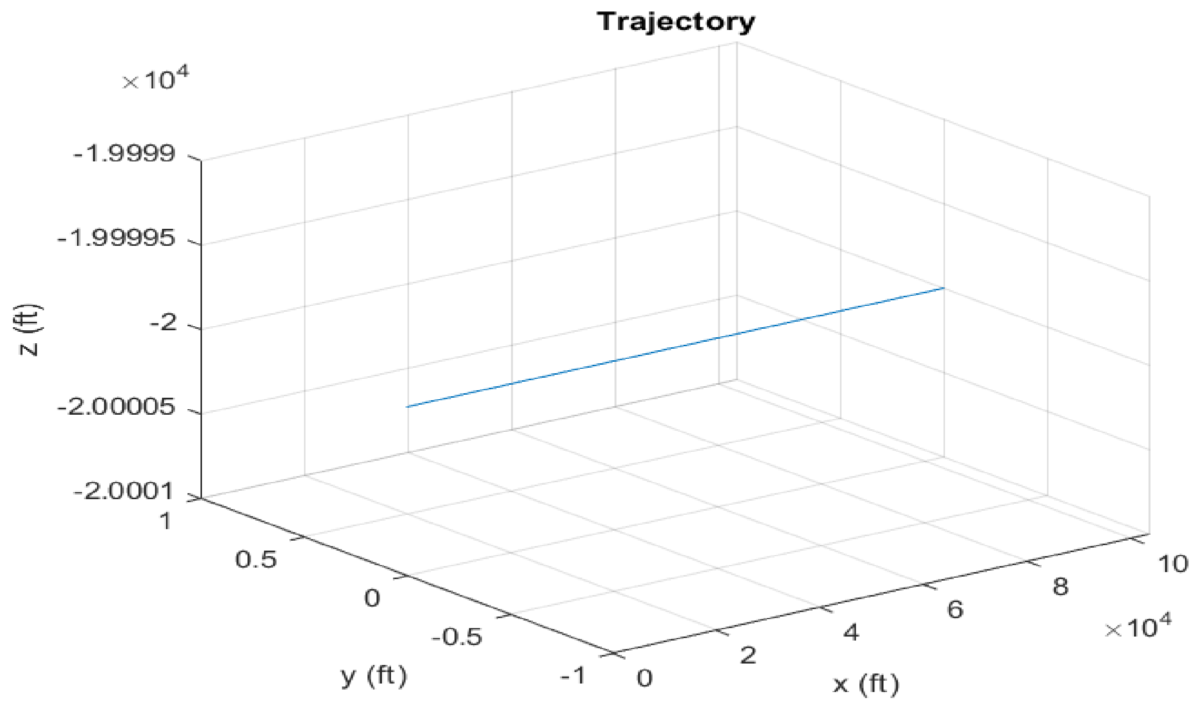


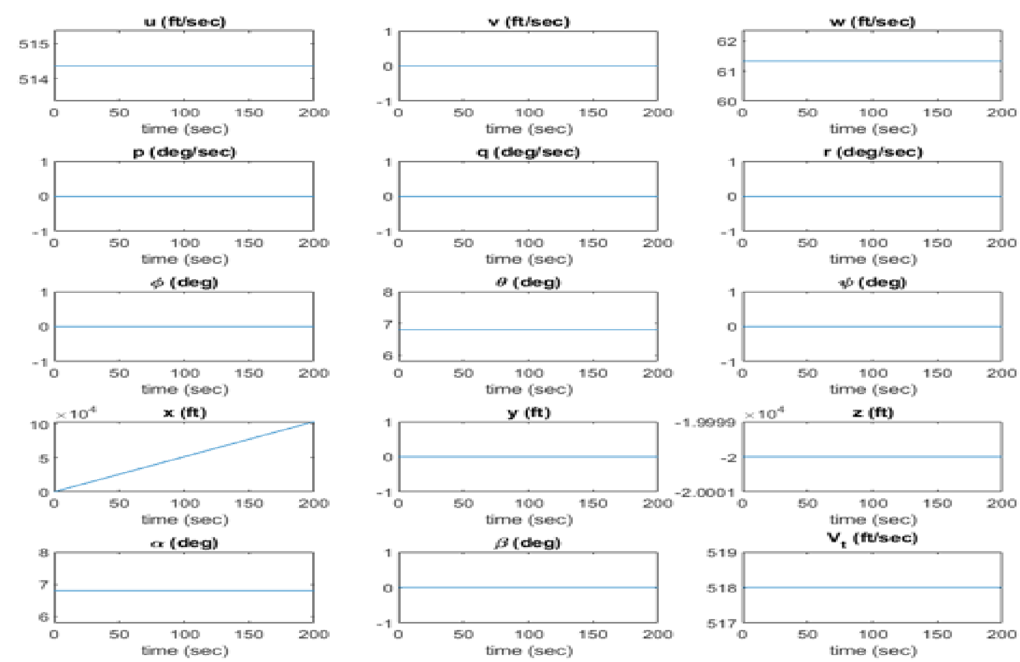
Aircraft Model



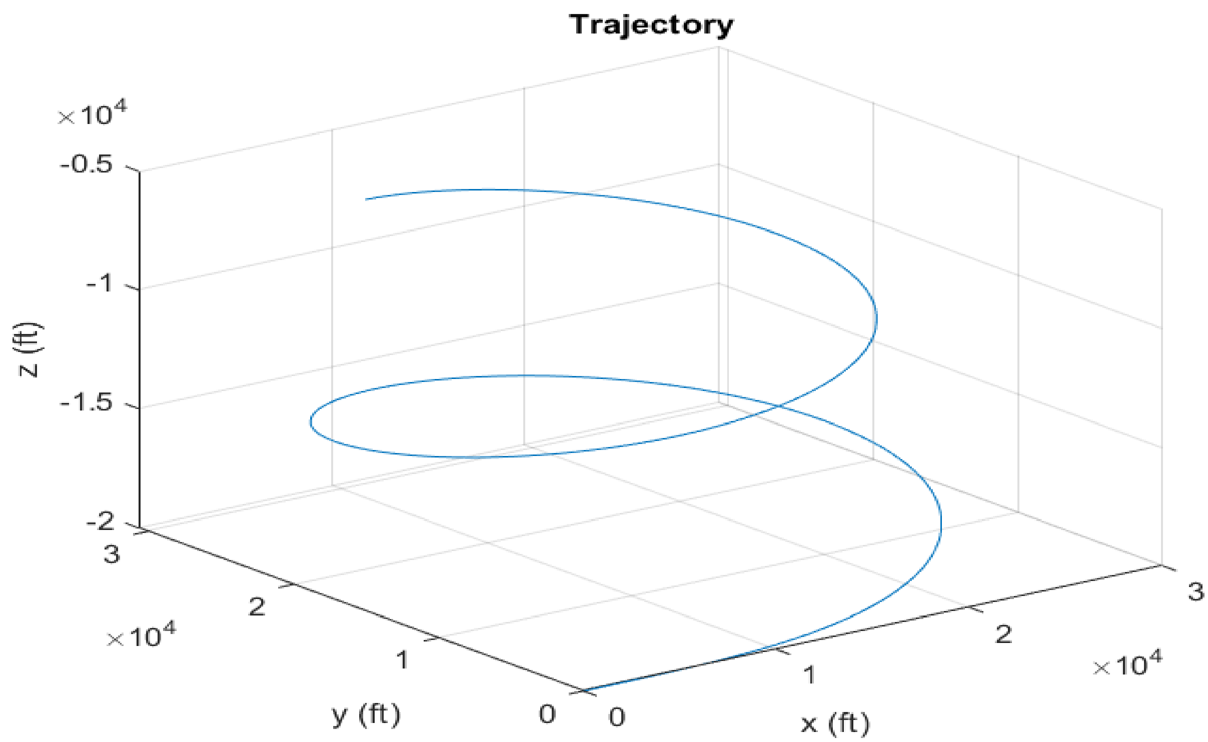
2) Validation: Response for B-747 flight condition 5 for various inputs

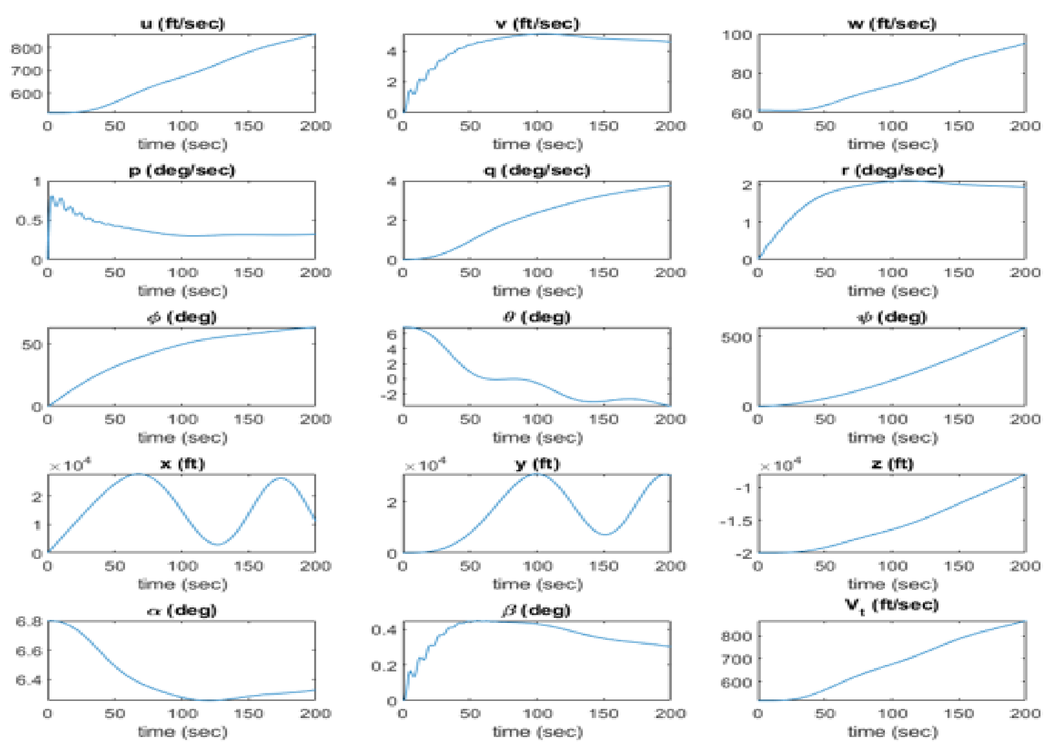
a) Response for no input



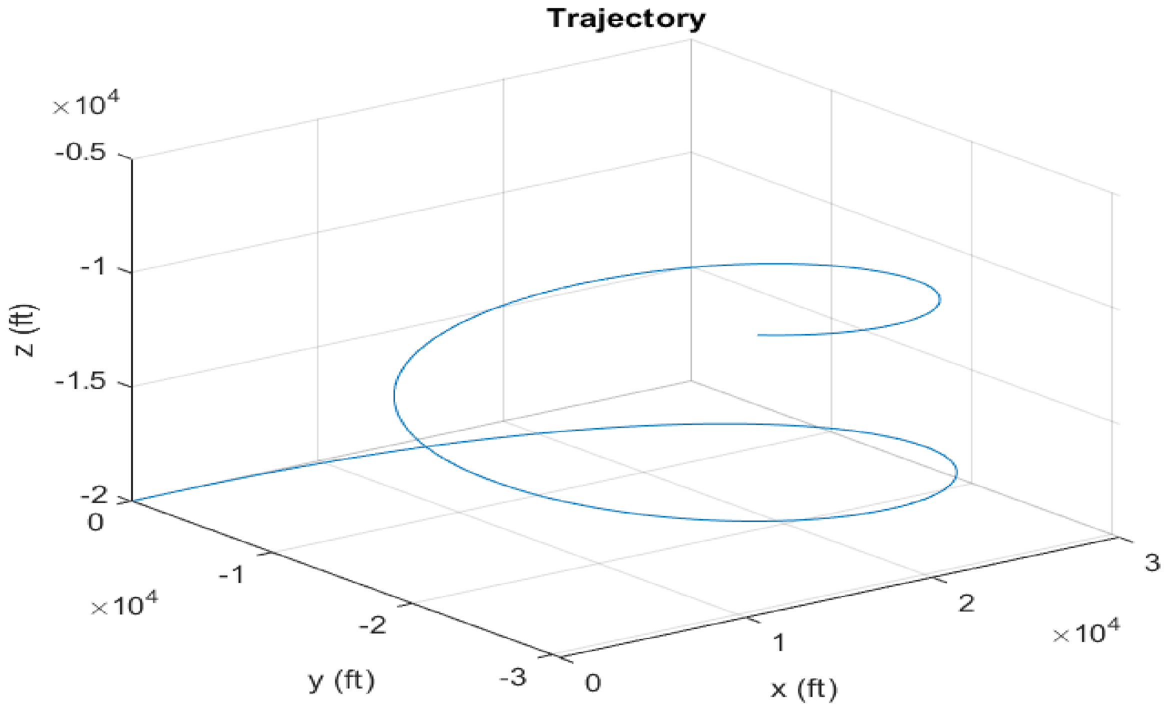


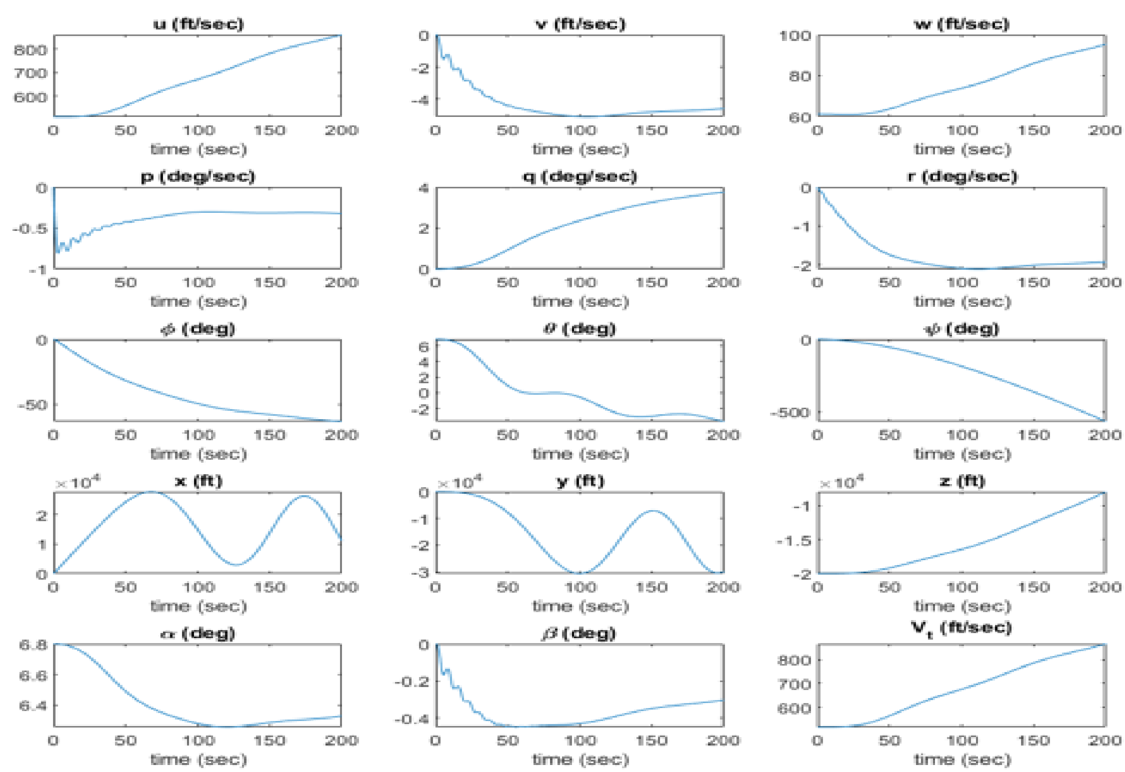
b) Response for +5° aileron



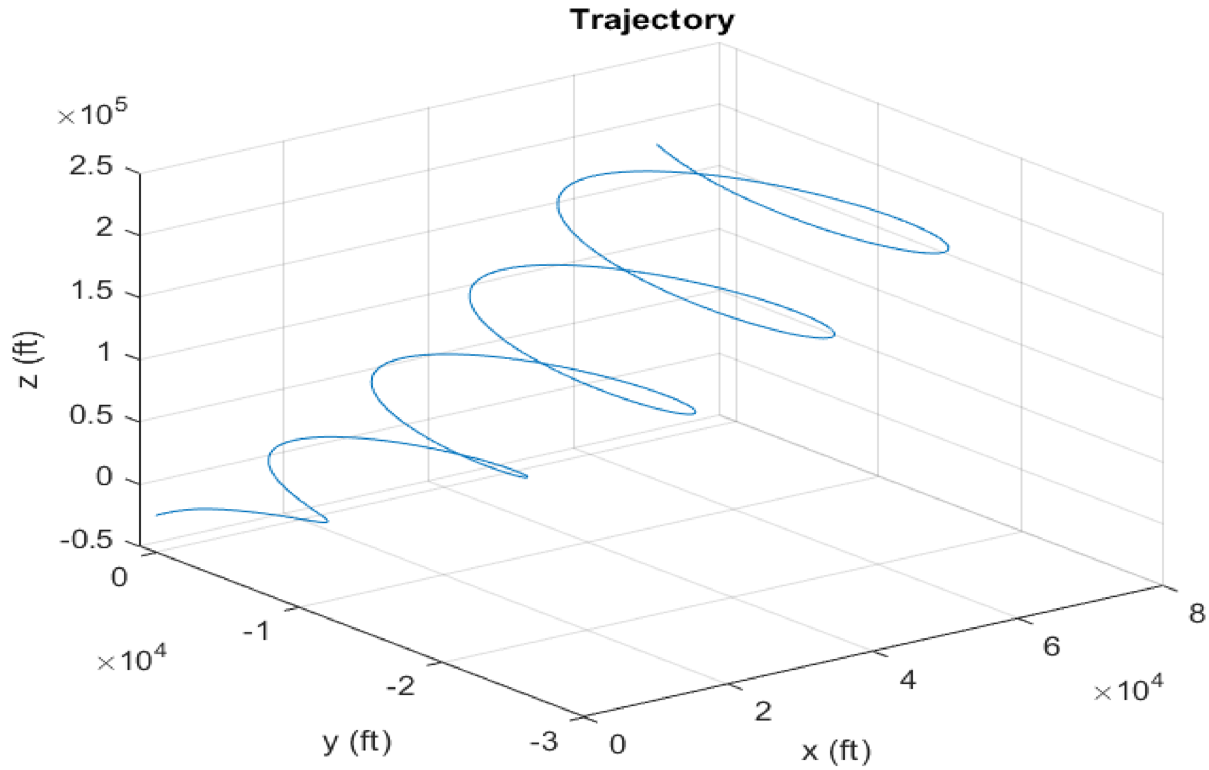


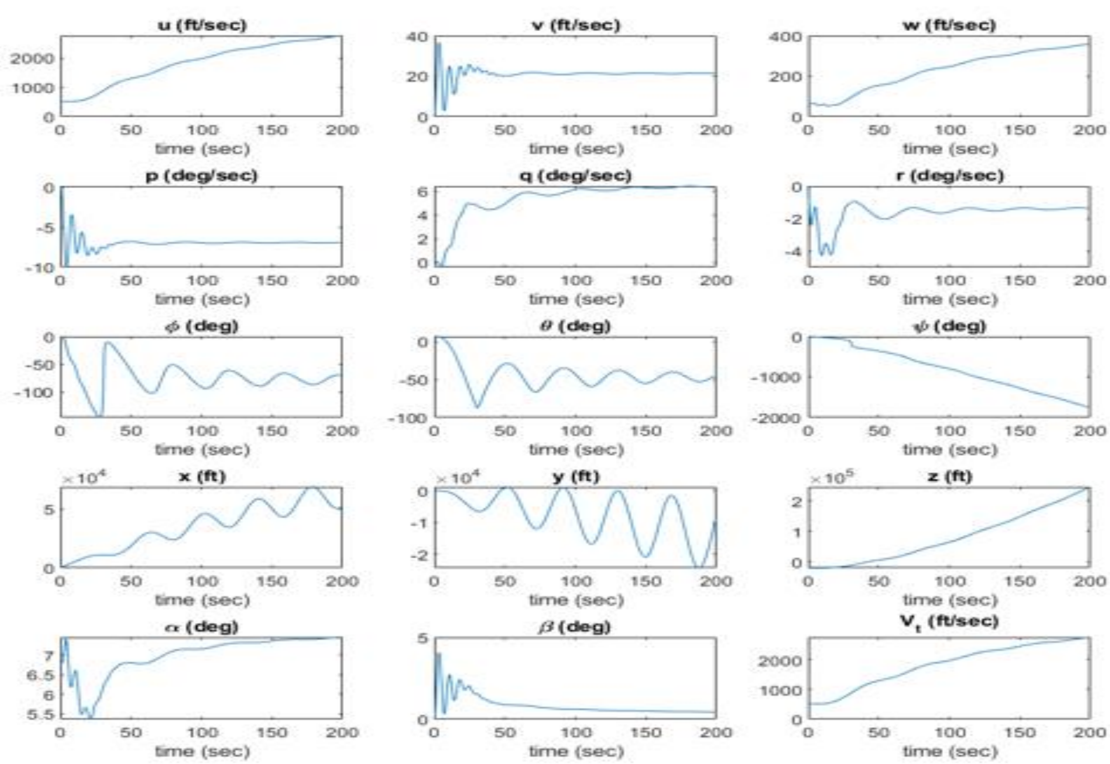
c) Response for -5° aileron



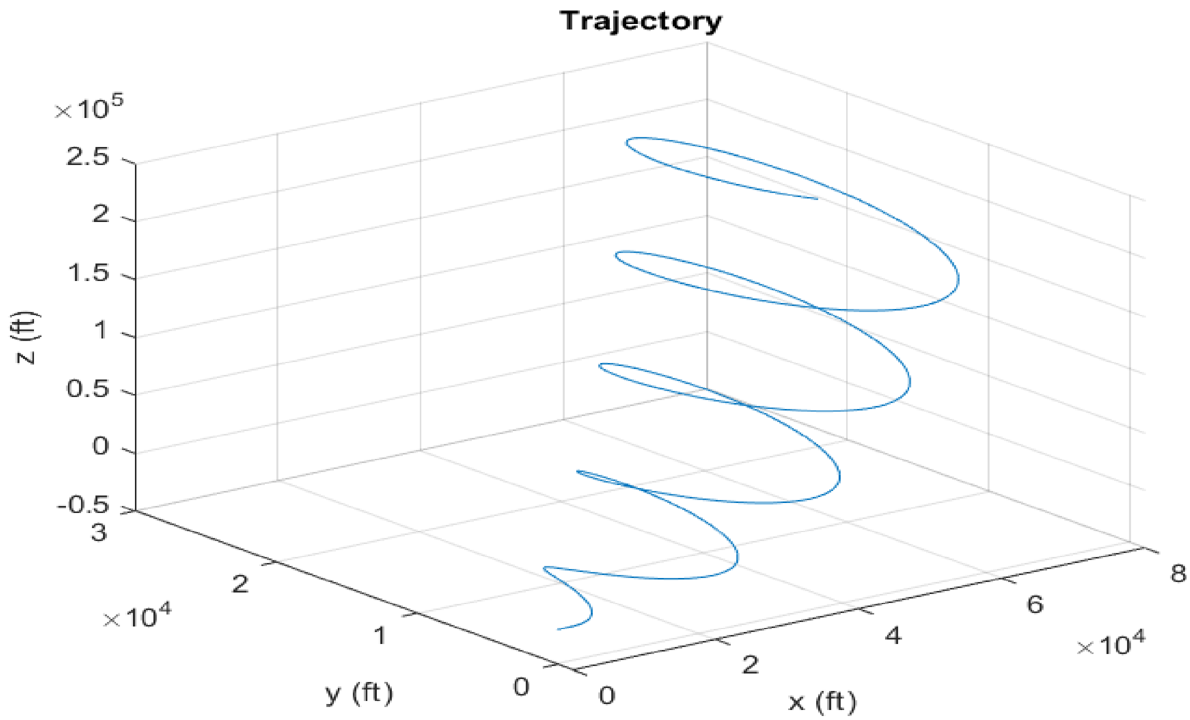


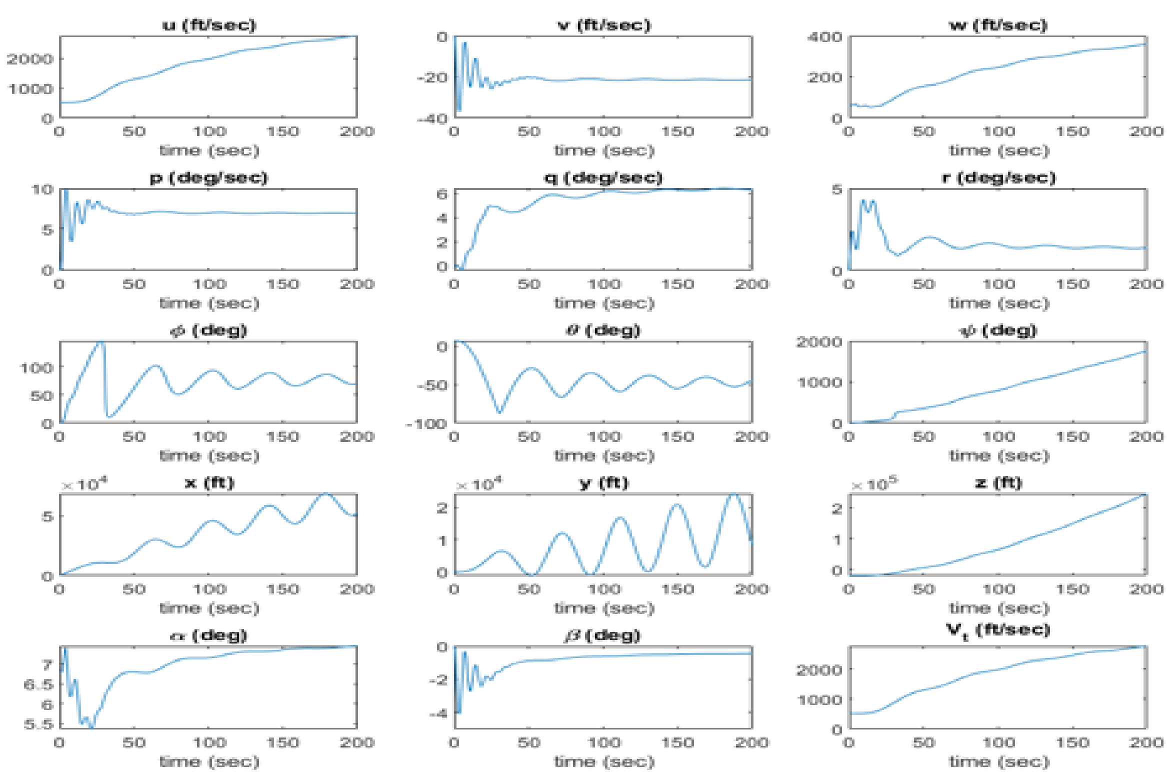
d) Response for +5° rudder



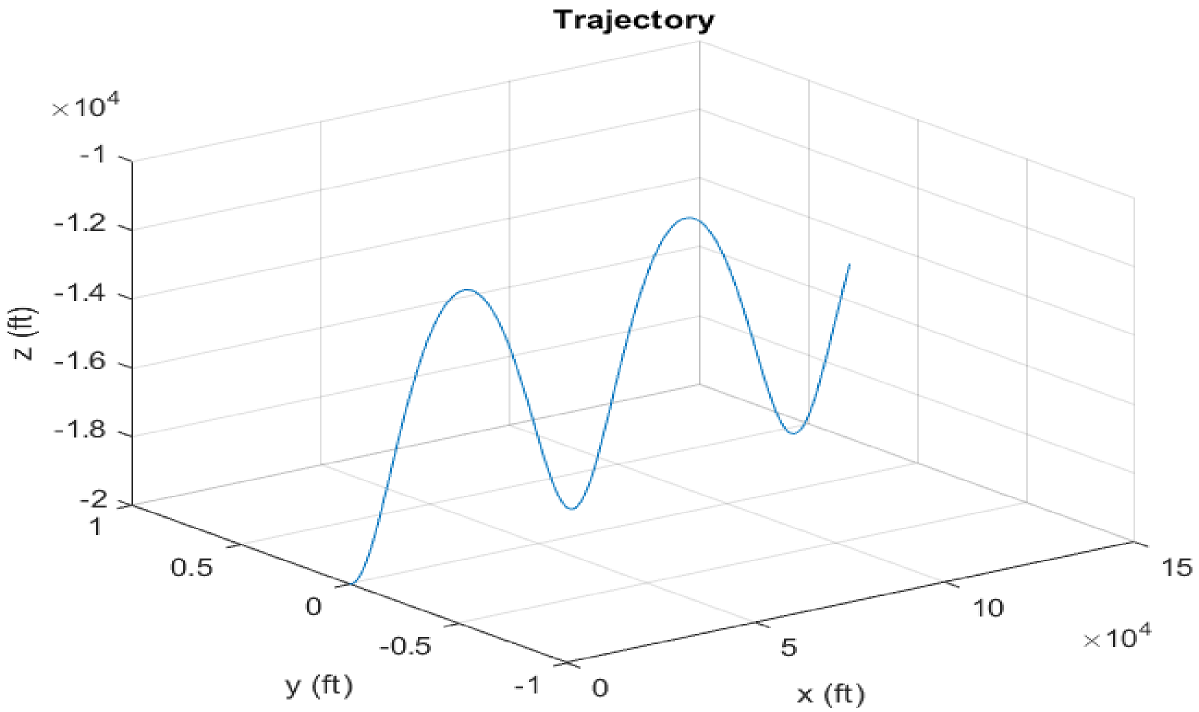


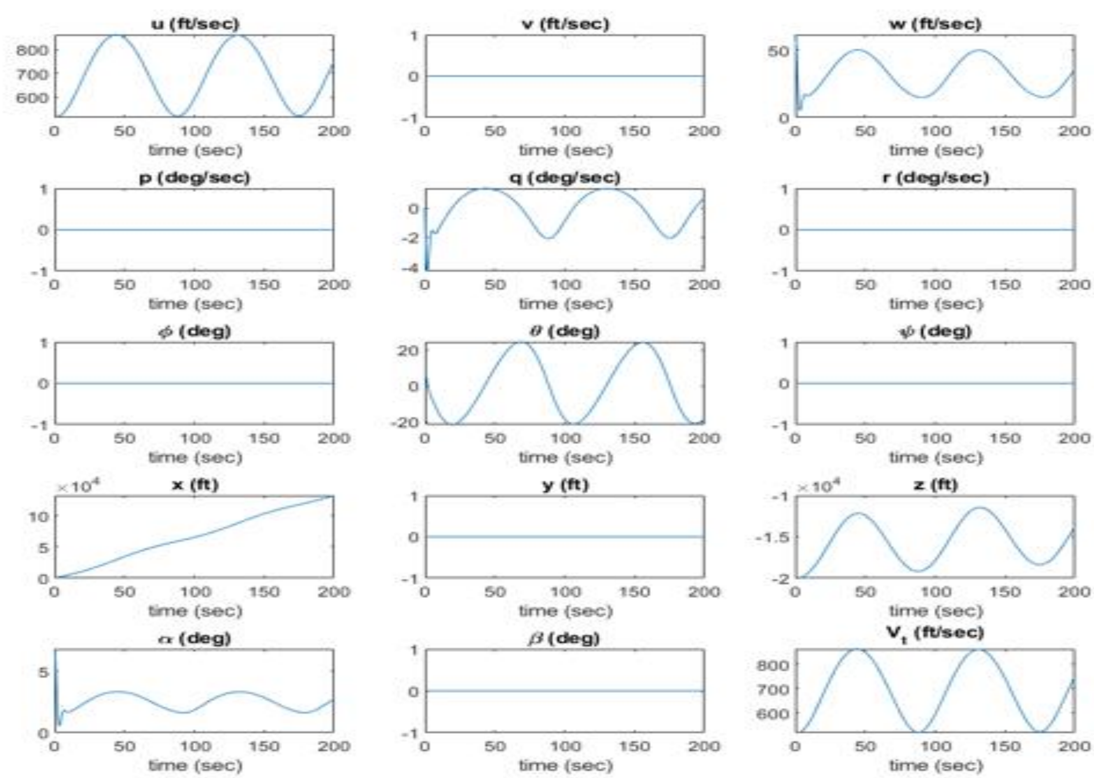
e) Response for -5° rudder



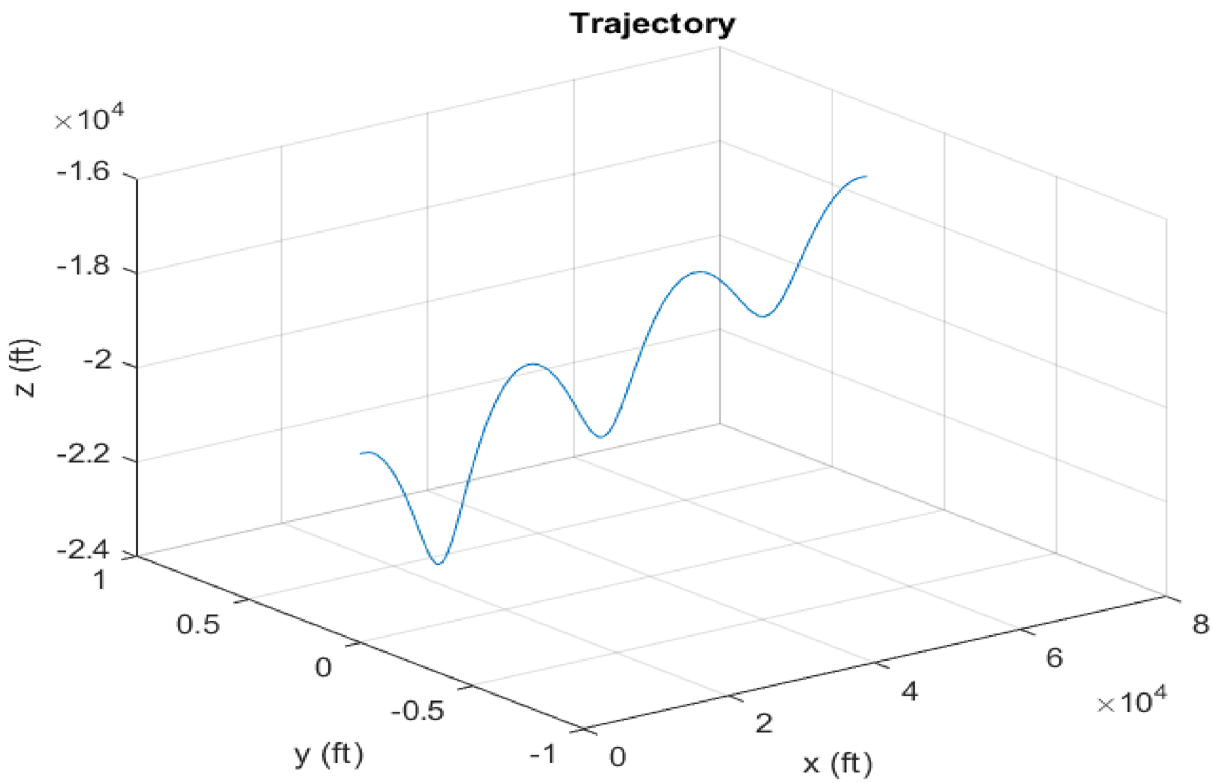


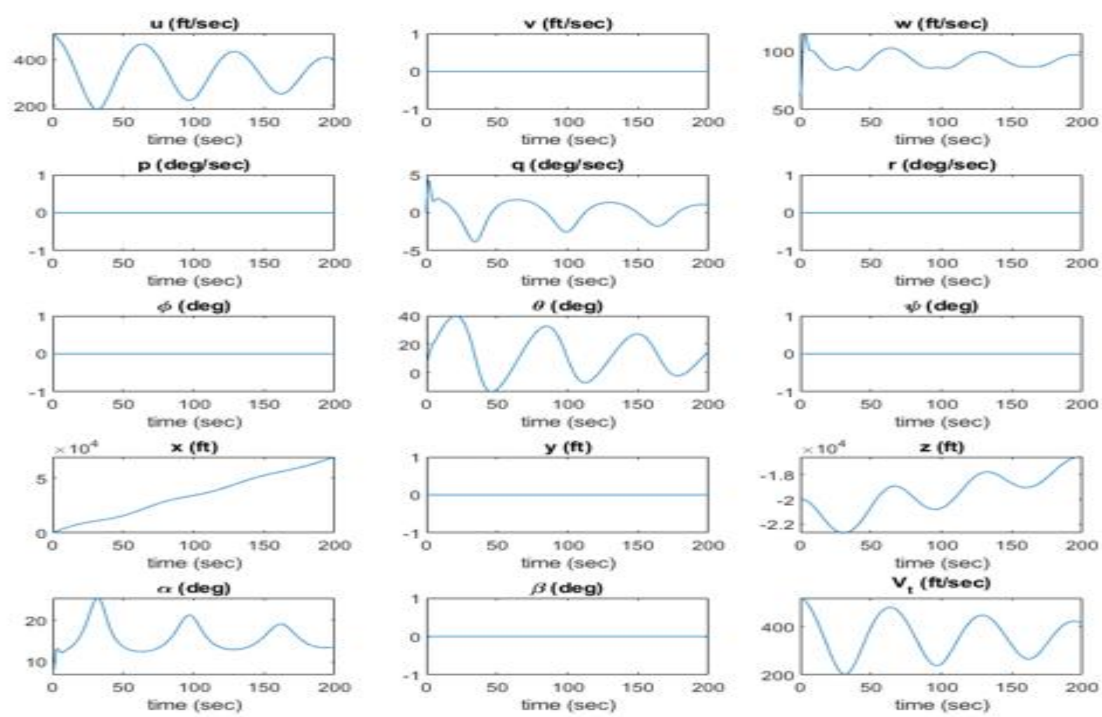
f) Response for +5° elevator



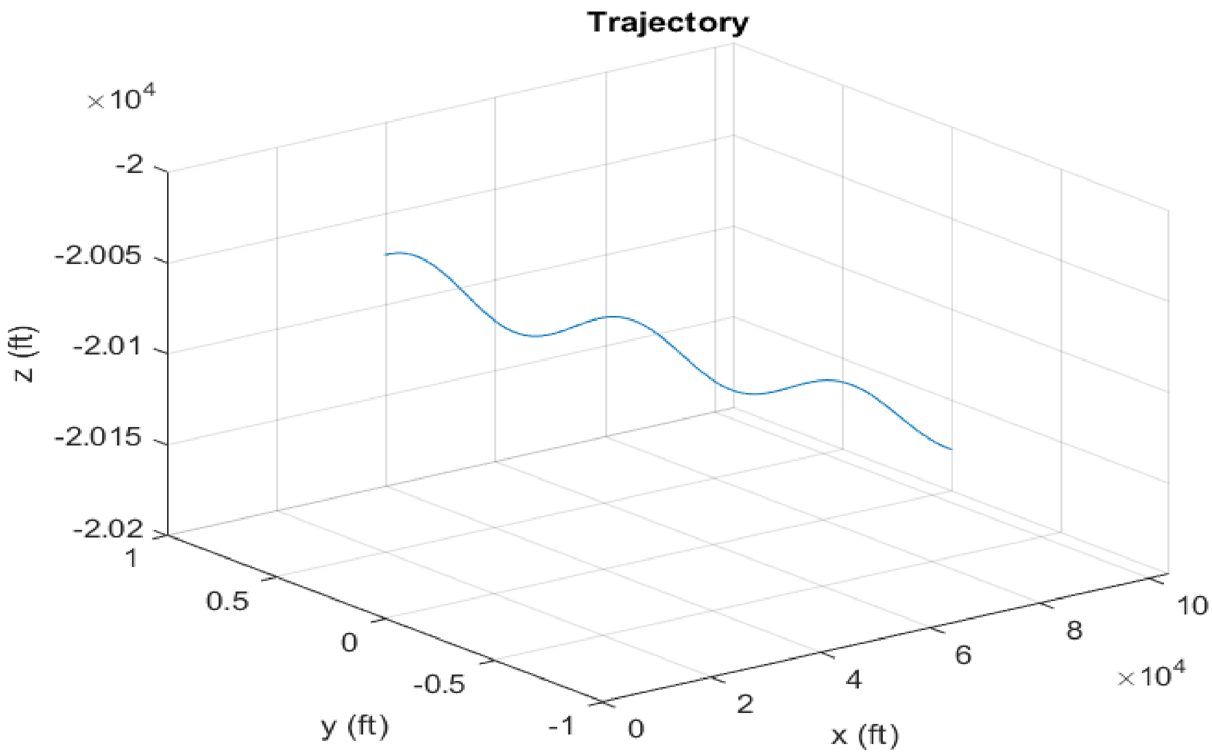


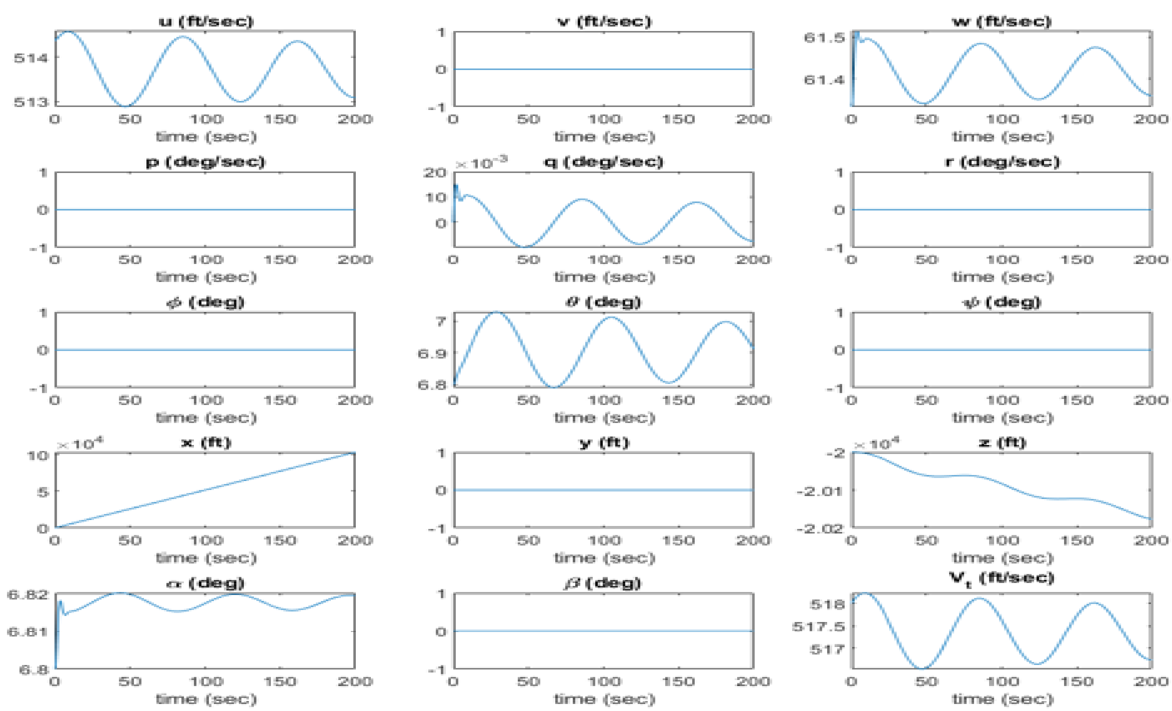
g) Response for -5° elevator



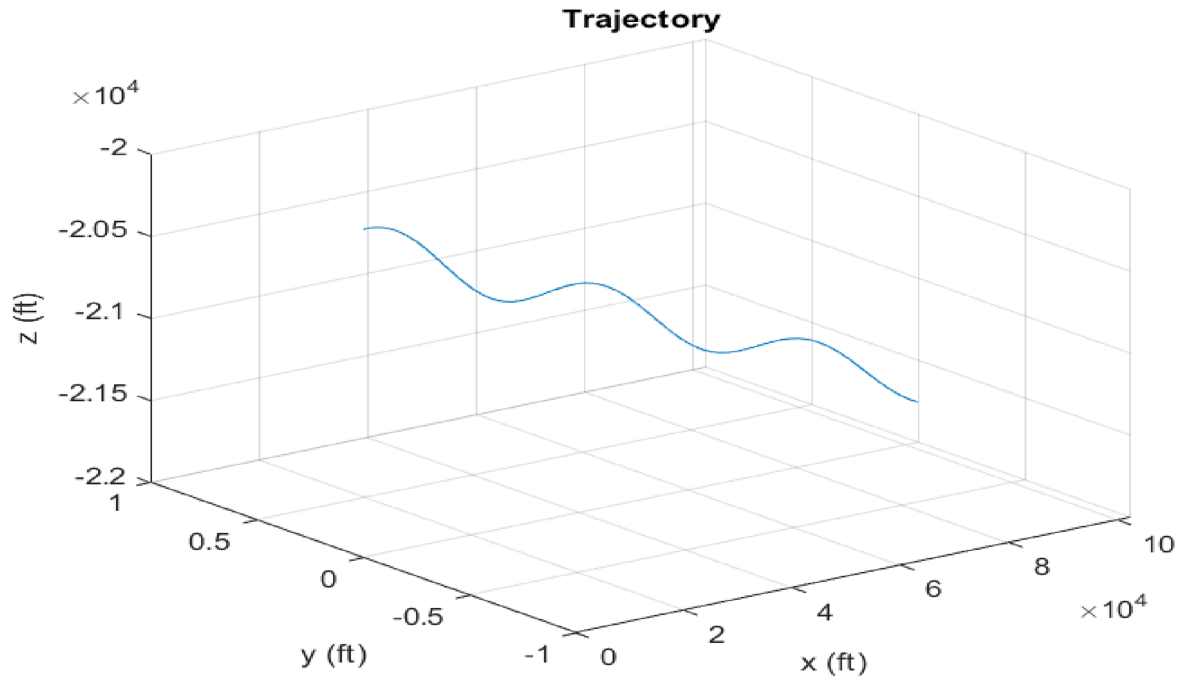


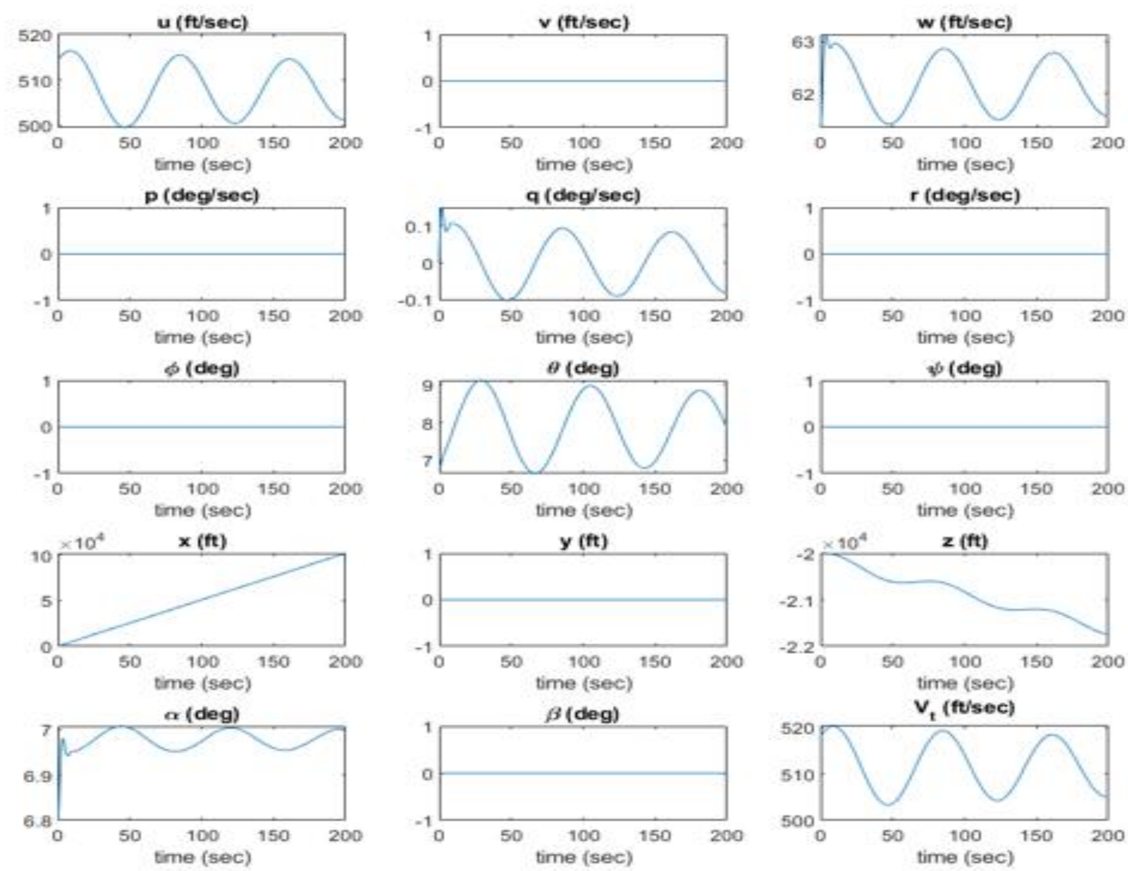
h) Response for 1000 in thrust





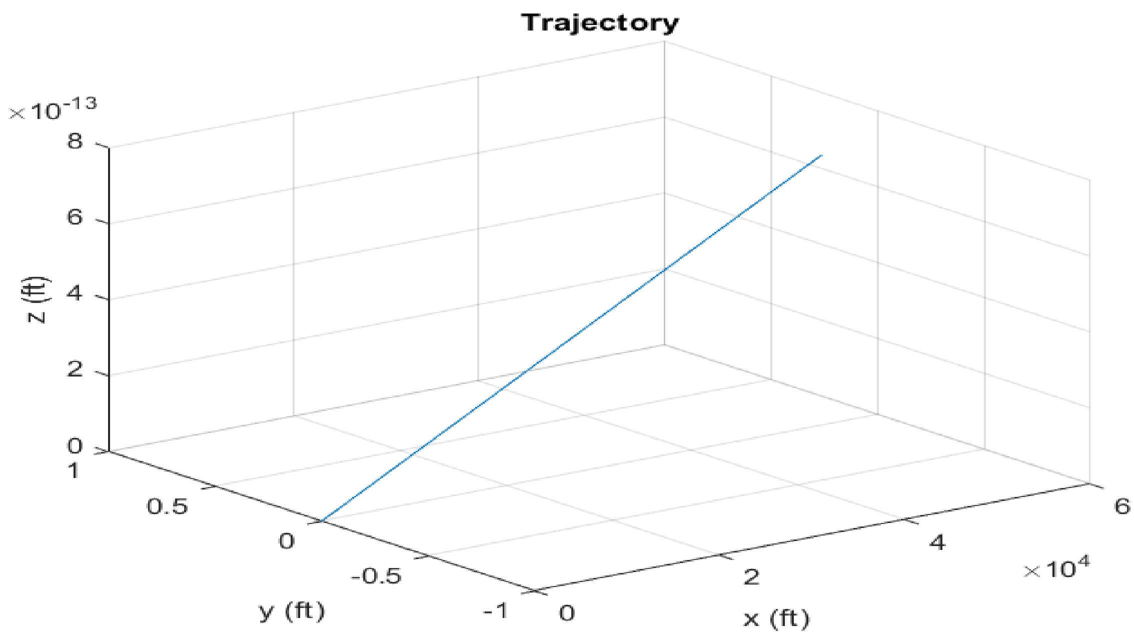
i) Response for 10000 in thrust

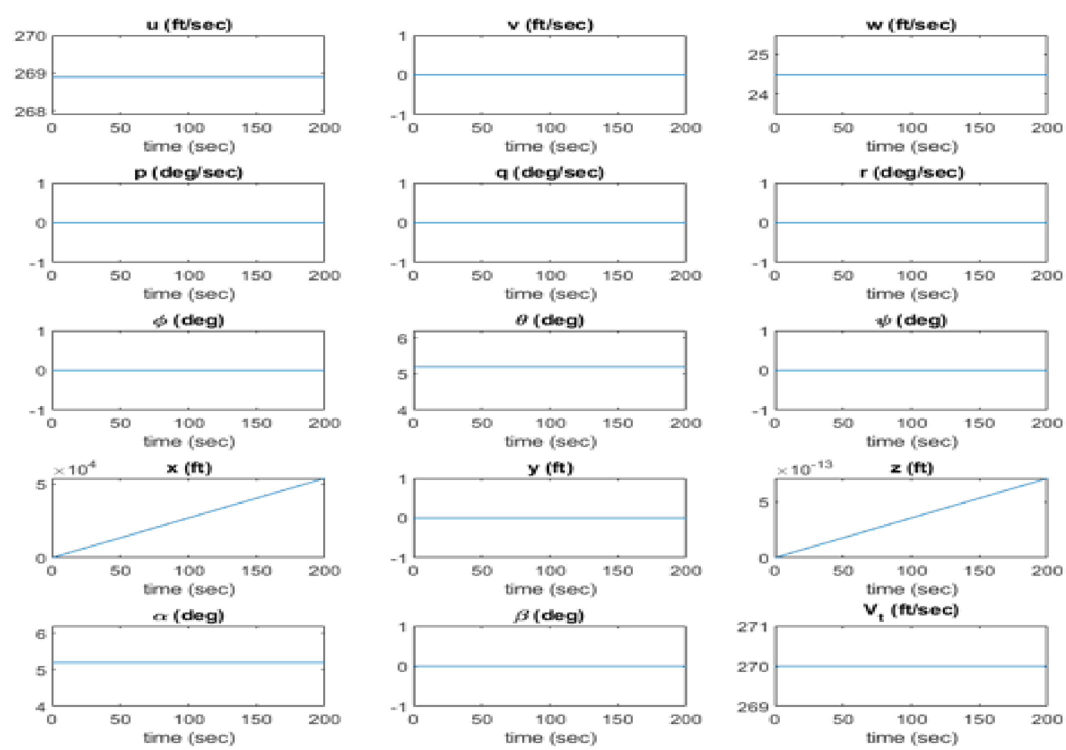




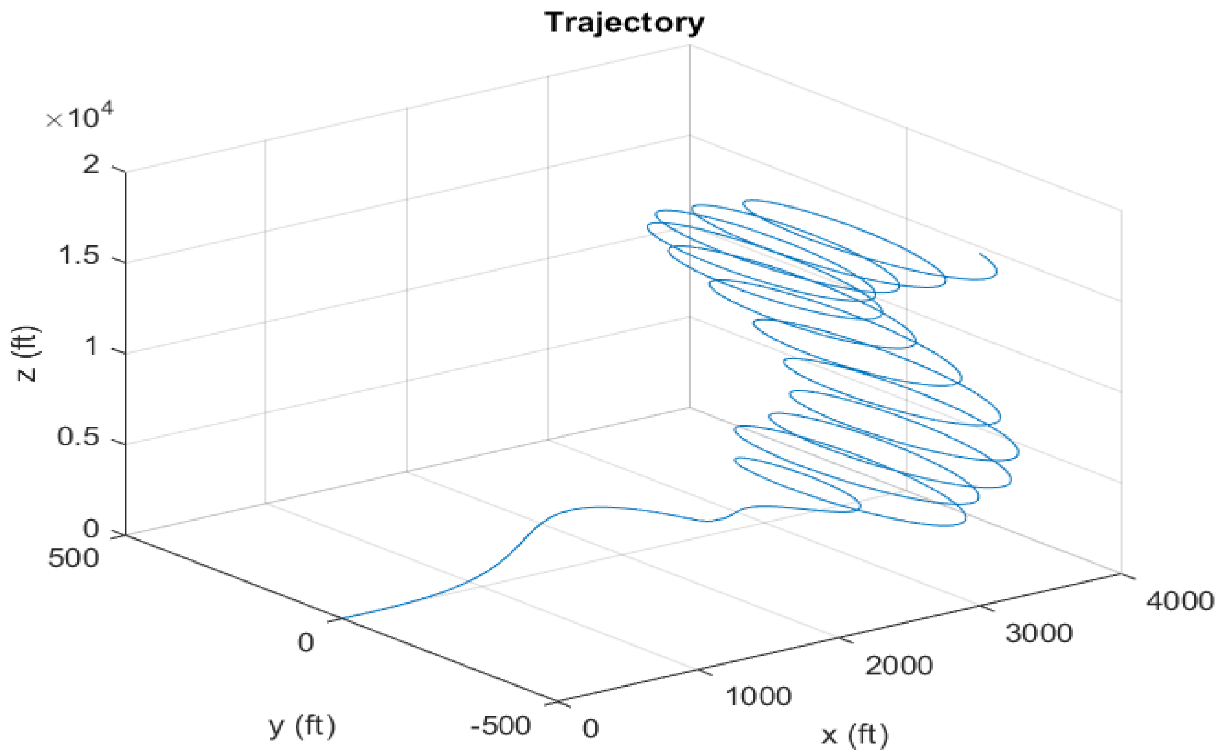
3) Response for NT_33A flight condition 2 for various inputs(before we change the plane to Boeing 747)

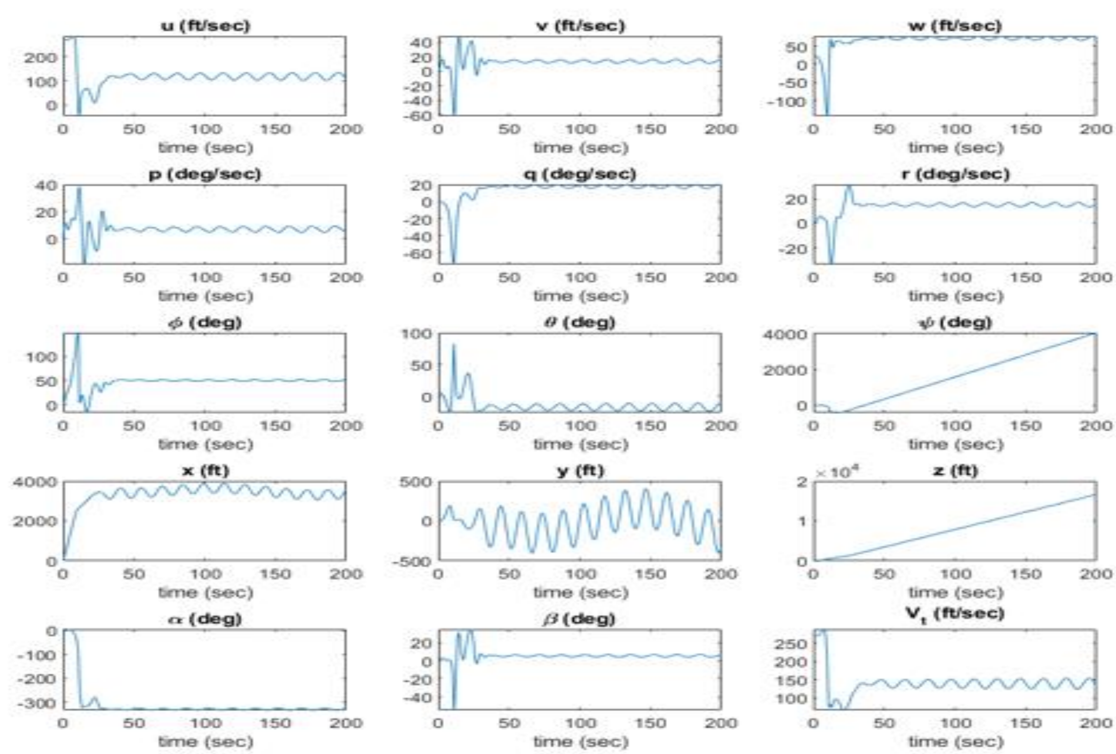
a) Response for no input



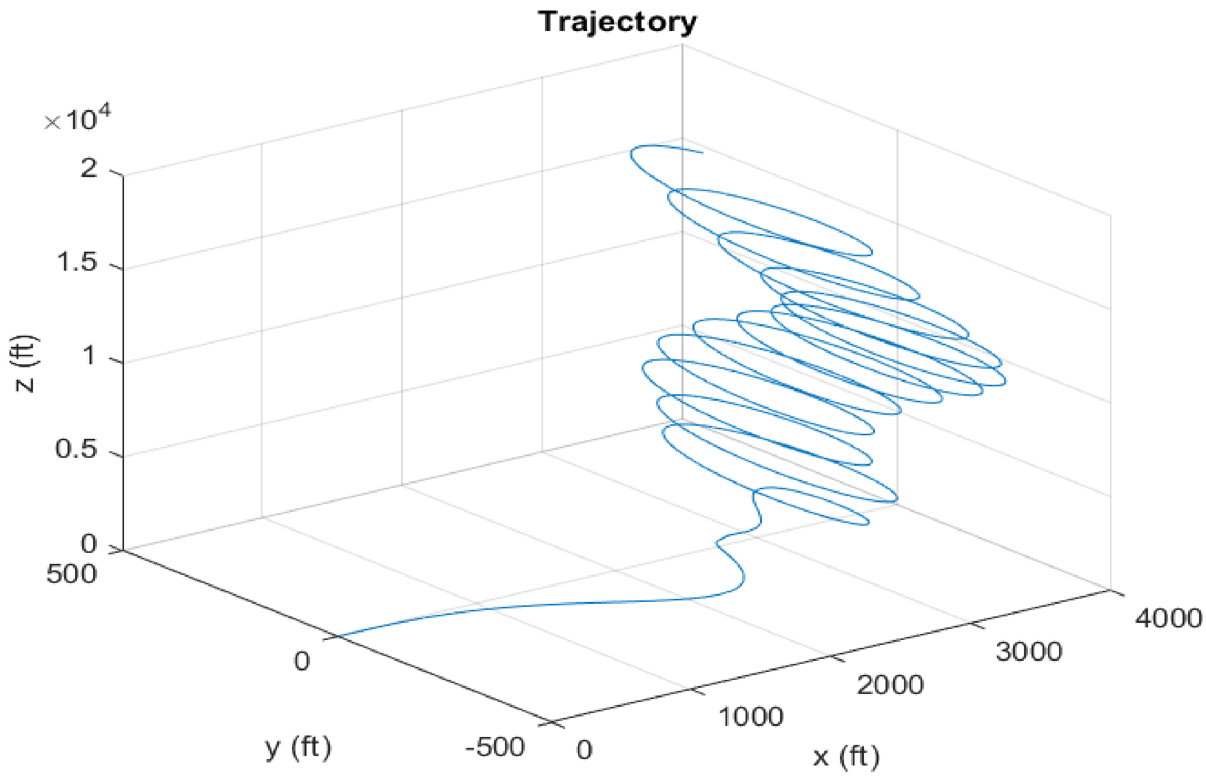


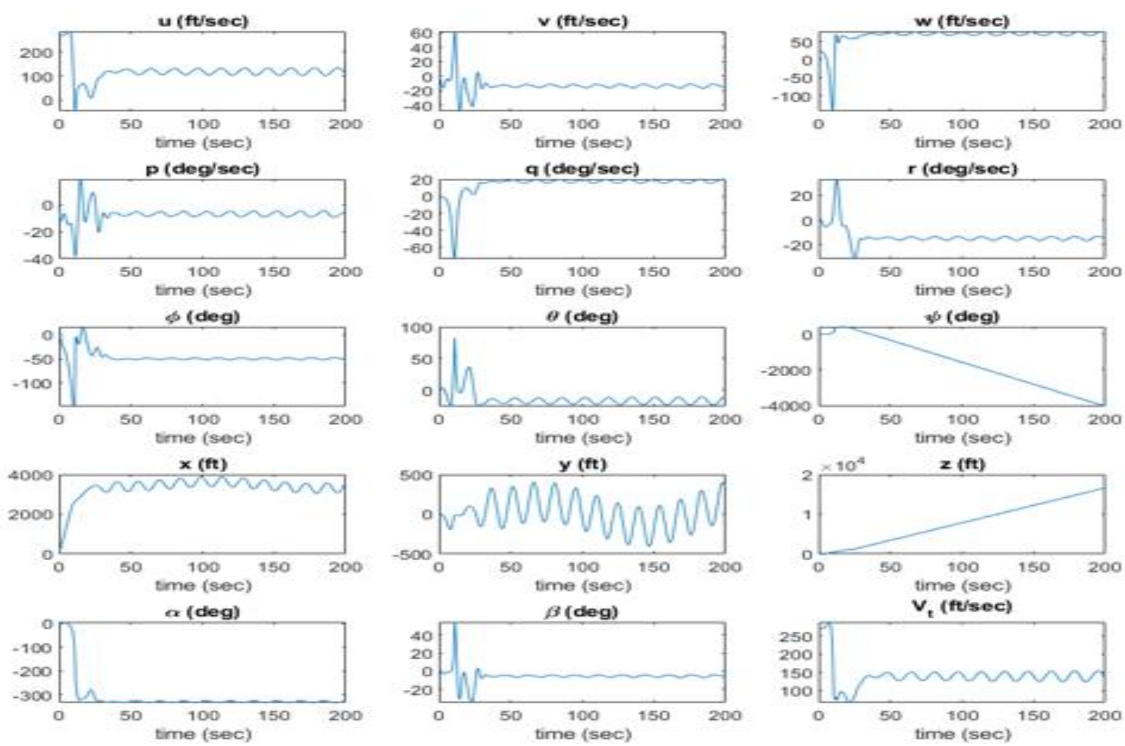
b) Response for +5° aileron



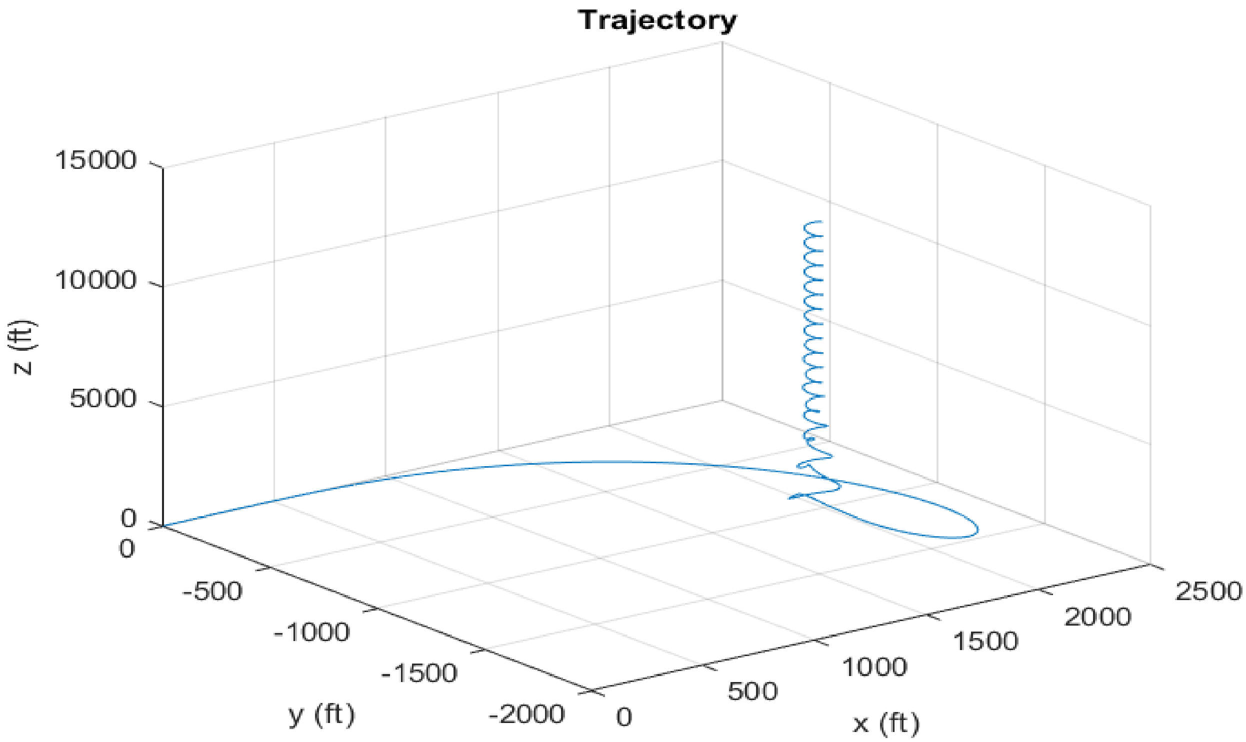


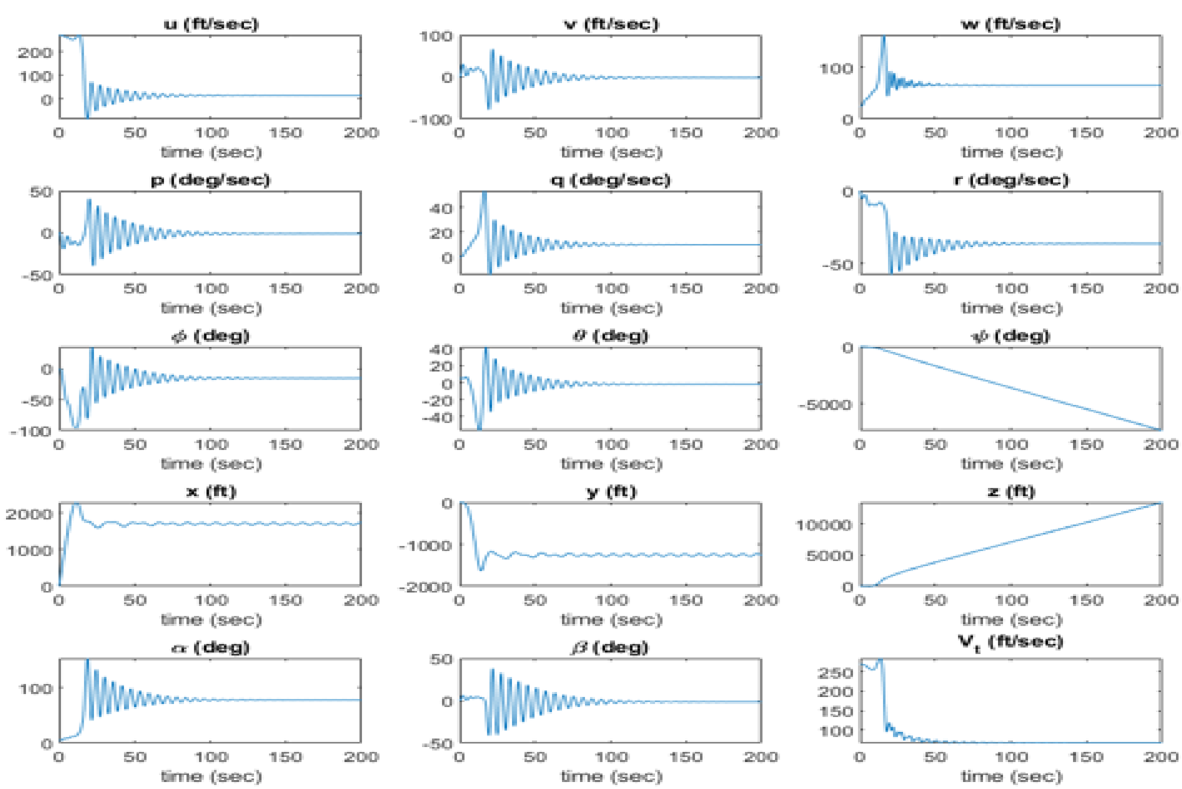
c) Response for -5° aileron



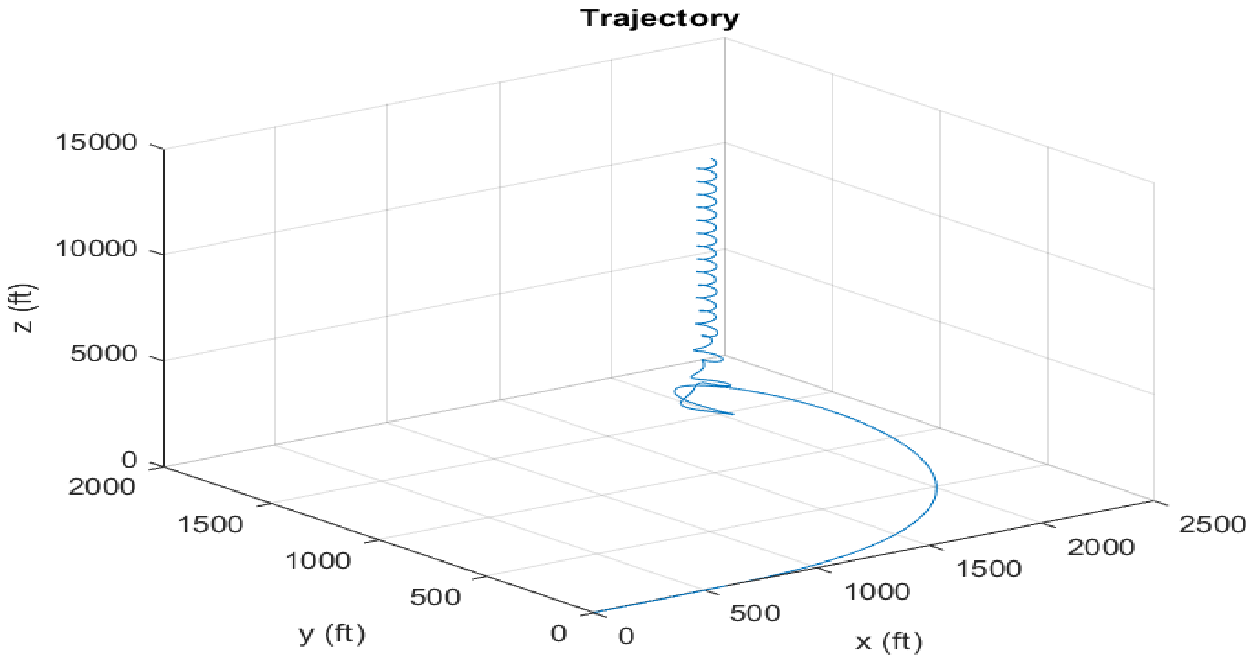


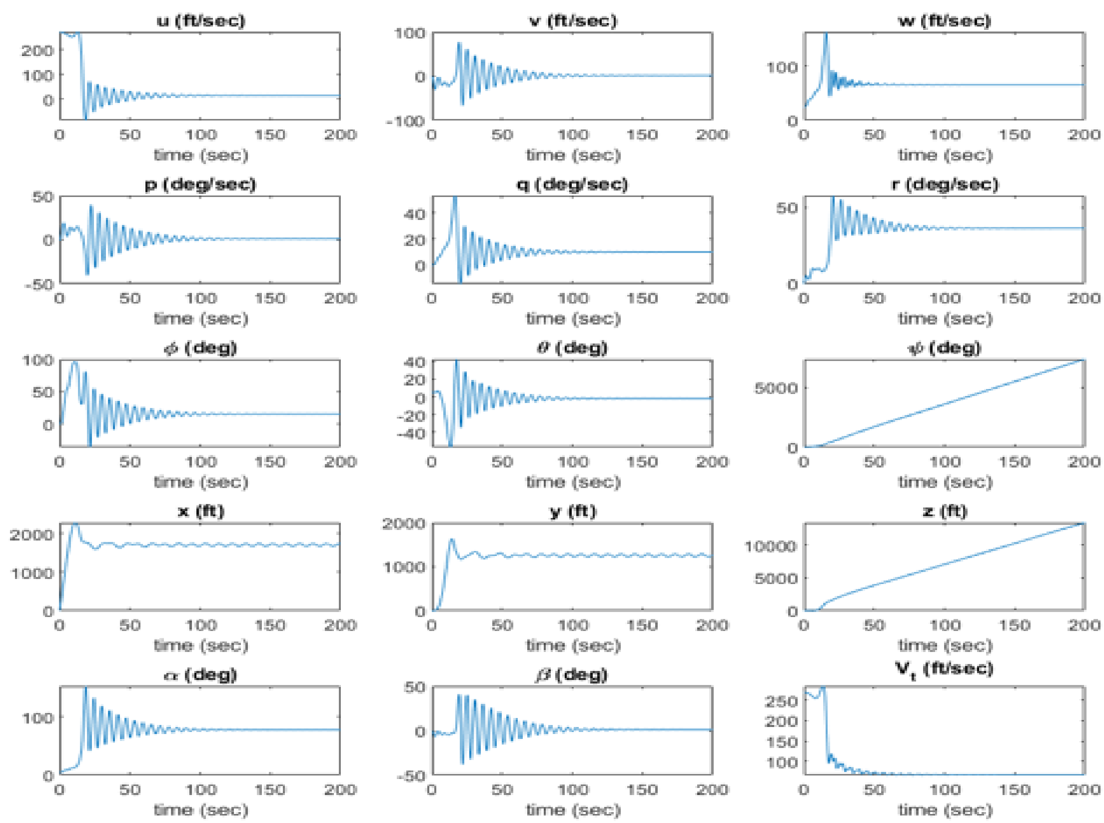
d) Response for +5° rudder



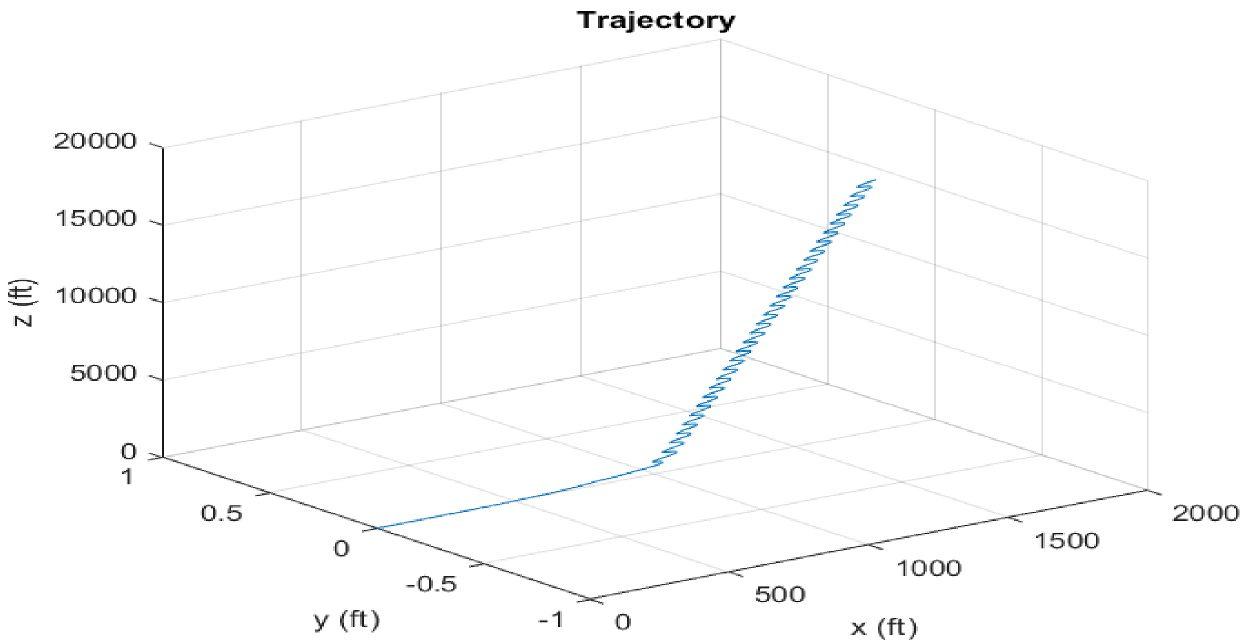


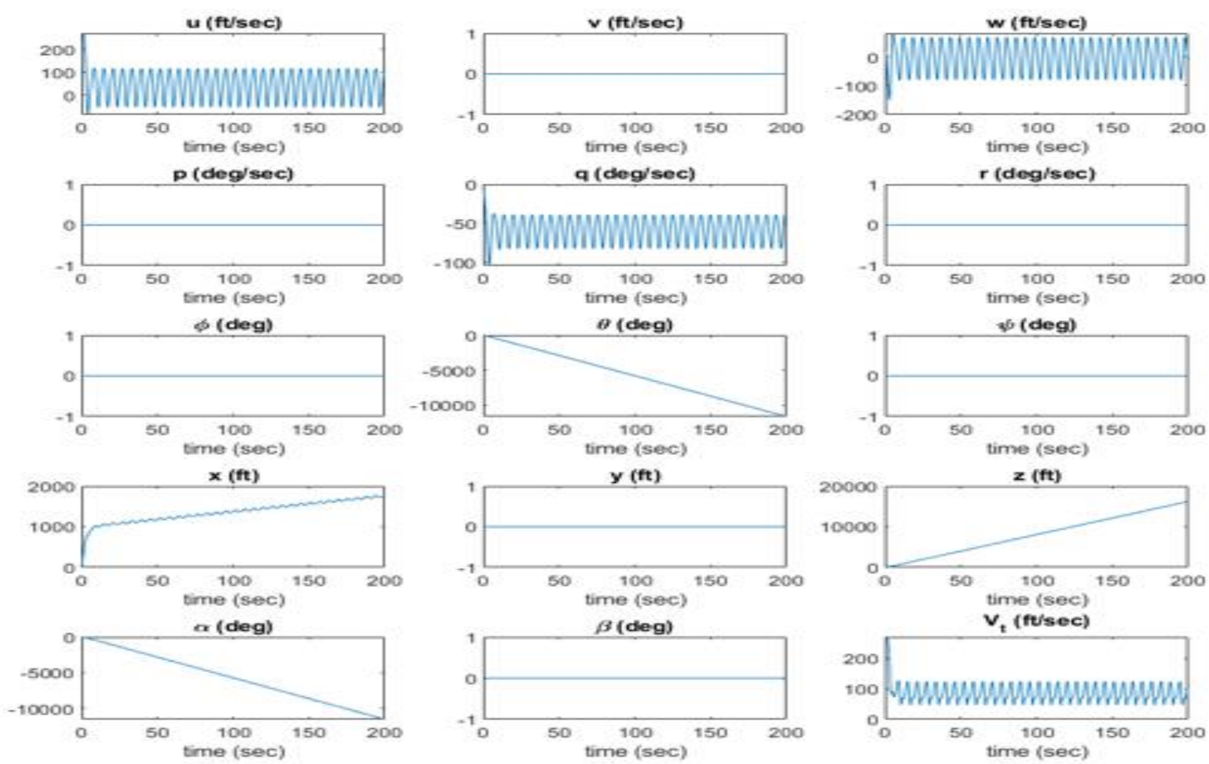
e) Response for -5° rudder



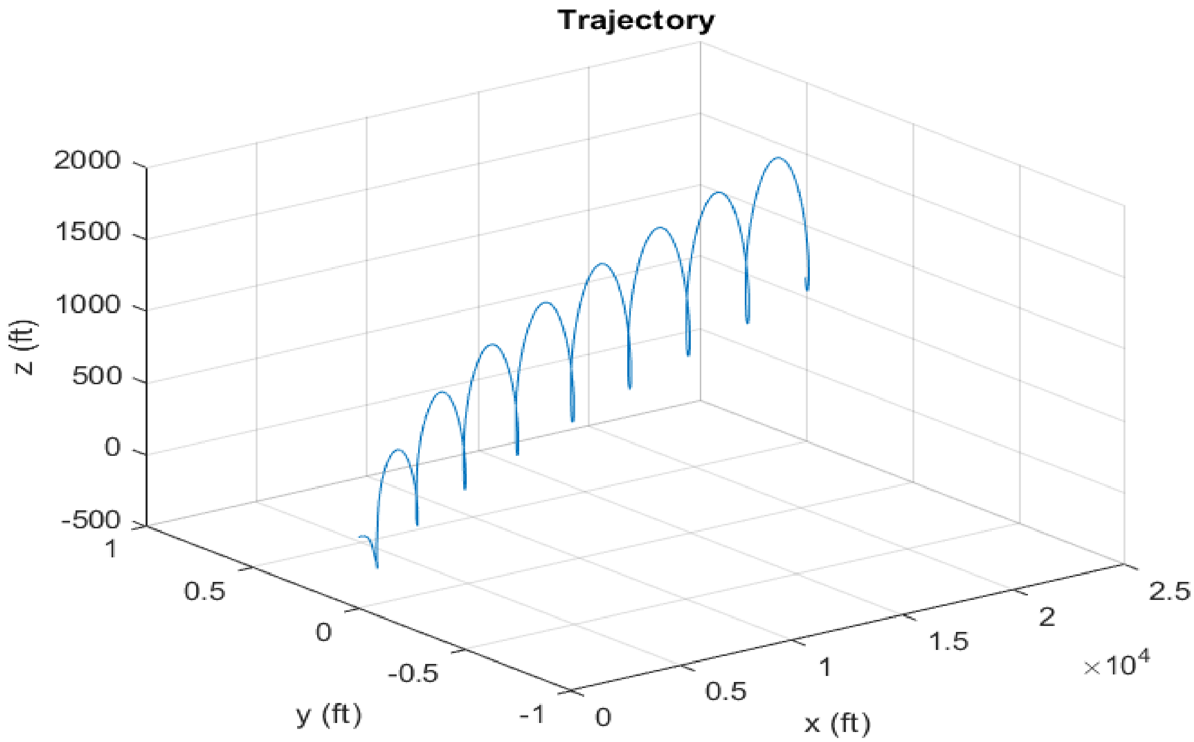


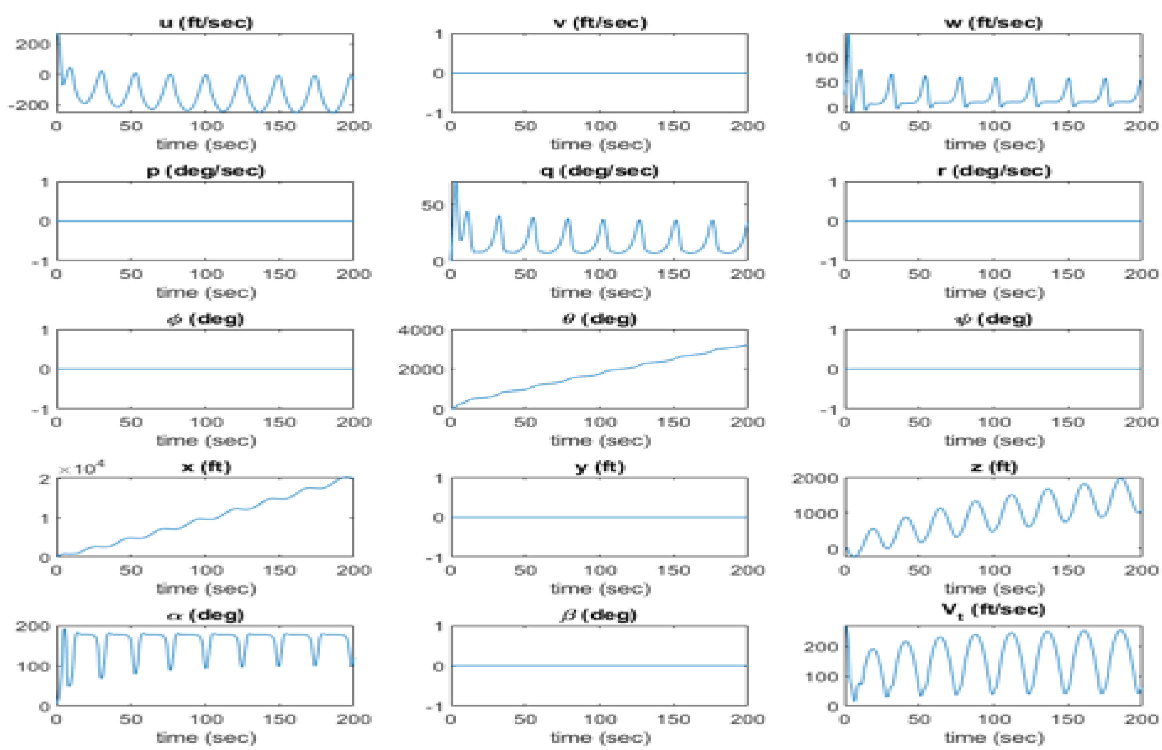
f) Response for +5° elevator



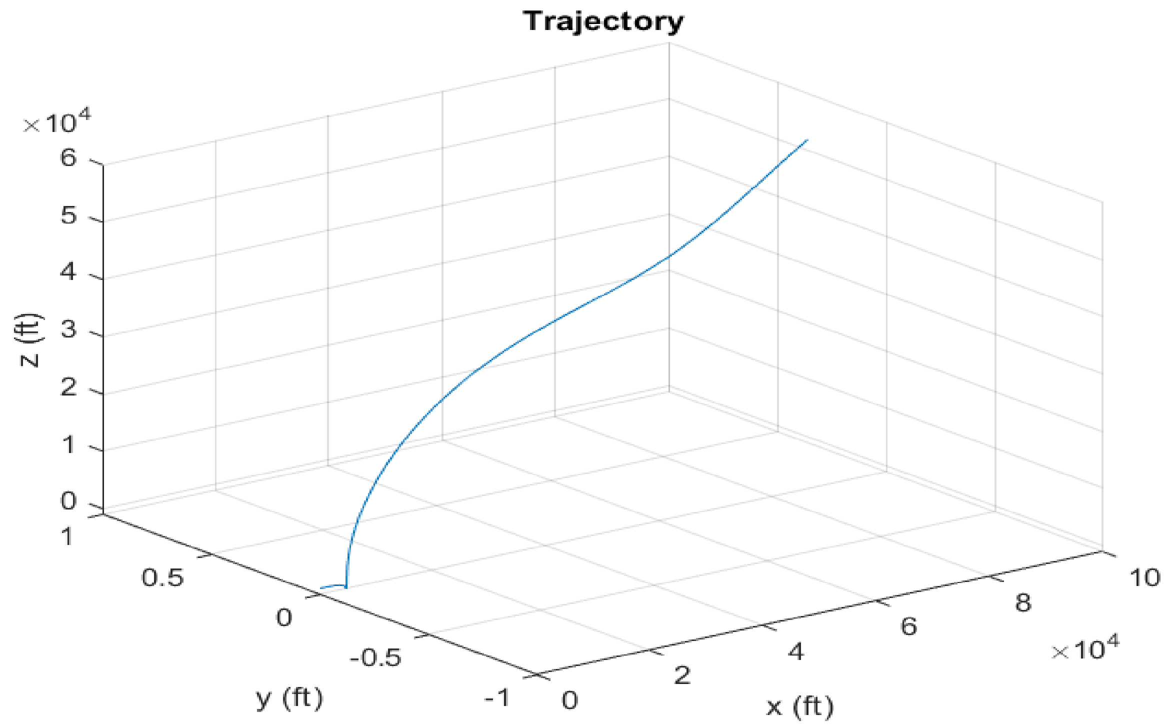


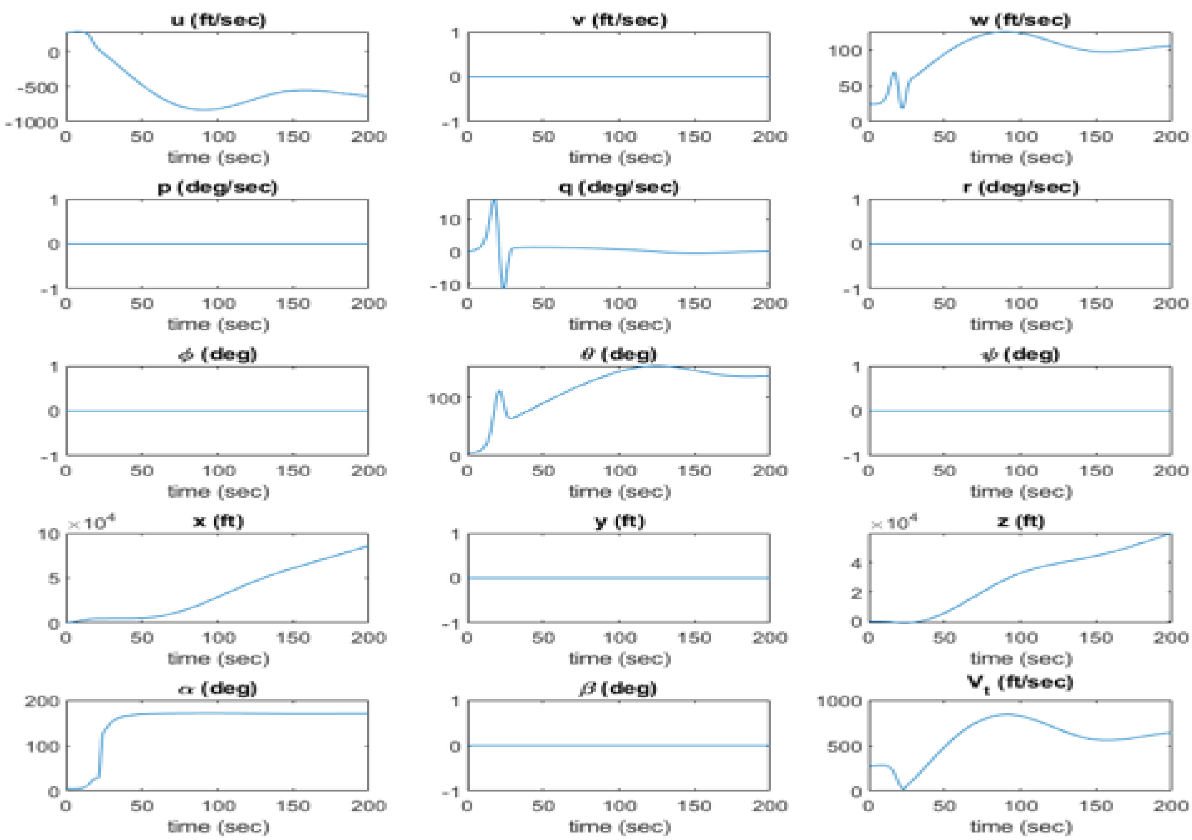
g) Response for -5° elevator



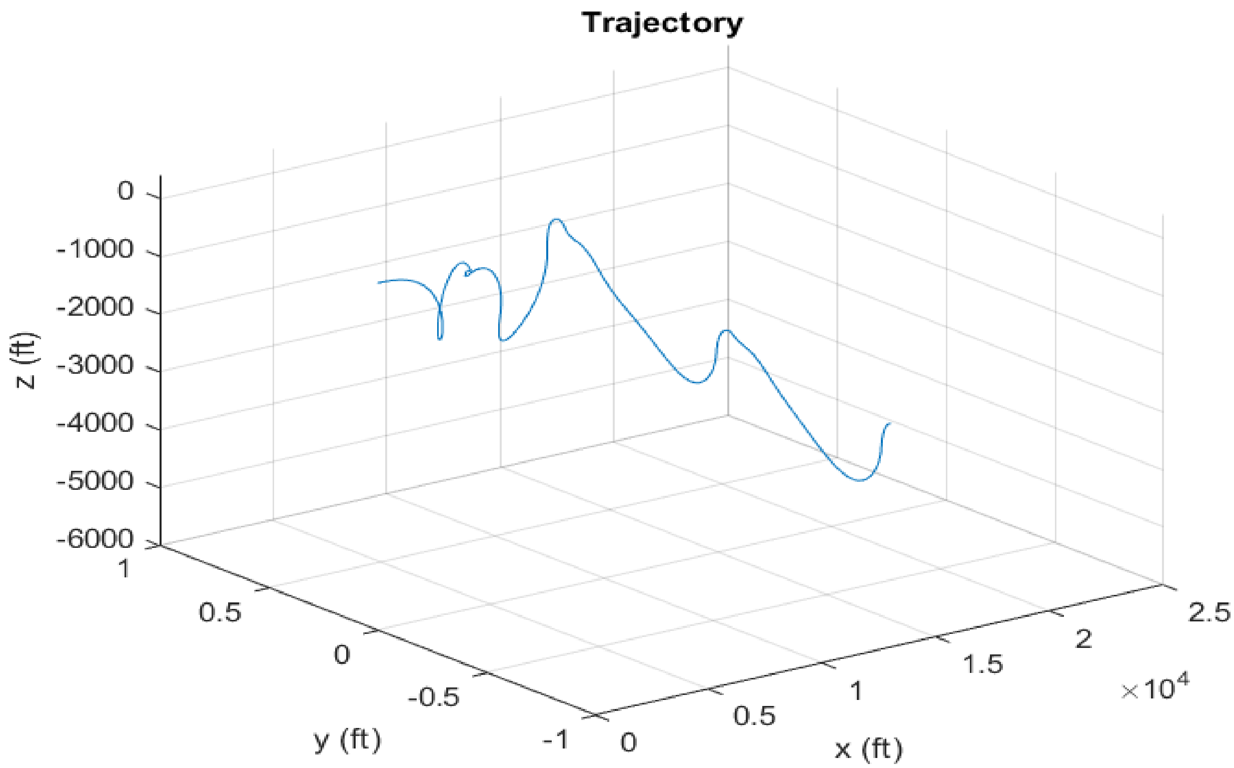


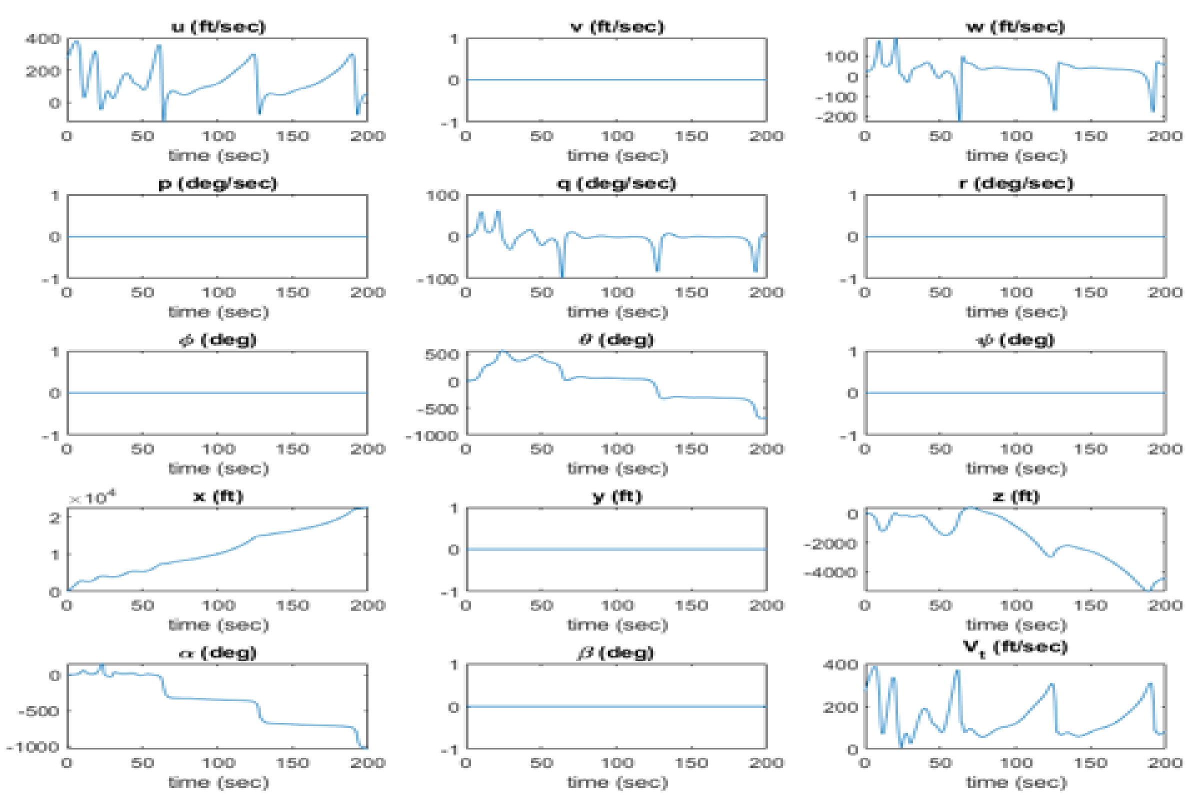
h) Response for 1000 in thrust





i) Response for 10000 in thrust





Task 4: Longitudinal dynamics

Overview

From task 3 and use etkin reference (linearization)

we get $\Delta x, \Delta y, \Delta z, \Delta N, \Delta M$ andΔL

and we get equation of motion.

And using small disturbance theory

$$\begin{aligned}\sin(\theta_0 + \Delta\theta) &= \sin\,\theta_0 \cos\,\Delta\theta + \cos\,\theta_0 \sin\,\Delta\theta \\ &\doteqdot \sin\,\theta_0 + \Delta\theta \cos\,\theta_0 \\ \cos(\theta_0 + \Delta\theta) &= \cos\,\theta_0 \cos\,\Delta\theta - \sin\,\theta_0 \sin\,\Delta\theta \\ &\doteqdot \cos\,\theta_0 - \Delta\theta \sin\,\theta_0\end{aligned}$$

And in reference steady state:

$$x_0 - \mathrm{g}\sin\theta_0 = 0\ ,\ \ y_0 = 0$$

$$z_0 + g\cos\,\theta_0 = 0\ ,\quad L_0 = M_0 = N_0 = 0$$

And for general: $\mathrm{H} = H_0 + \Delta H : using\ donald\ mclean\ reference$

$$\begin{aligned}X_0 &= m[Q_0W_0 - R_0V_0 + g\sin\,\Theta_0] \\ Y_0 &= m[U_0R_0 - P_0W_0 - g\cos\,\Theta_0 \sin\,\Phi_0] \\ Z_0 &= m[P_0V_0 - Q_0U_0 - g\cos\,\Theta_0 \cos\,\Phi_0] \\ L_0 &= Q_0R_0\left(I_{xz} - I_{yy}\right) - P_0Q_0I_{xz} \\ M_0 &= (P_0^2 - R_0^2)I_{xz} + (I_{xx} - I_{xz})P_0R_0 \\ N_0 &= I_{xz}Q_0R_0 + \left(I_{yy} - I_{xz}\right)P_0Q_0\end{aligned}$$

the perturbed equations of motion for an aircraft can be written as:

$$\begin{aligned}\mathrm{d}X = \quad & m[\dot{u} + W_0q + Q_0w - V_0r - R_0v + g\cos\,\Theta_0\theta] \\ \mathrm{d}Y = \quad & m[\dot{v} + U_0r + R_0u - W_0p - P_0w - (g\cos\,\Theta_0\cos\,\Phi_0)\phi \\ & +(g\sin\,\Theta_0\sin\,\Phi_0)\theta] \\ \mathrm{d}Z = \quad & m[\dot{w} + V_0p + P_0v - U_0q - Q_0u + (g\cos\,\Theta_0\sin\,\Phi_0)\phi \\ & +(g\sin\,\Theta_0\cos\,\Phi_0)\theta]\end{aligned}$$

Equations of Motion of a Rigid Body Aircraft

$$\begin{aligned}\mathrm{d}L \quad &= I_{xx}\dot{p} - I_{xz}\dot{r} + \left(I_{zz} - I_{yy}\right)(Q_0r + R_0q) - I_{xz}(P_0q + Q_0p) \\ \mathrm{d}M \quad &= I_{yy}\dot{q} + (I_{xx} - I_{zz})(P_0r + R_0p) - (2R_0r - 2P_{00}p)I_{xz} \\ dN \quad &= I_{xz}\dot{r} - I_{xz}\hat{p} + \left(I_{yy} - I_{xz}\right)(P_0q + Q_0p) + I_{xz}(Q_0r + R_0q)\end{aligned}$$

$$
\begin{aligned}
p = &\quad \phi - \Psi \sin\ \Theta_0 - \theta(\Psi_0 \cos\ \Theta_0) \\
q = &\quad \dot{\phi} \cos\ \Phi_0 - \theta(\Psi_0 \sin\ \Phi \sin\ \Theta_0) + \Psi \sin\ \Psi_0 \cos\ \Theta \\
&\quad +\phi\big(\Psi_0 \cos\ \Theta_0 \cos\ \Phi_0 - \dot{\Theta}_0 \sin\ \Phi_0\big) \\
r = &\quad \dot{\Psi} \cos\ \Theta_0 \cos\ \Phi_0 - \phi(\Psi_0 \cos\ \Theta_0 \sin\ \Phi_0 + \Psi_0 \cos\ \Phi_0) \\
&\quad -\dot{\theta} \sin\ \Phi_0 - \theta(\Psi_0 \sin\ \Theta_0 \cos\ \Phi_0)
\end{aligned}
$$

The significance of the specified trim conditions may be judged when the following implications are understood:

1-That straight flight implies $\Psi_0 = \Theta_0 = 0$.

2-That symmetric flight implies $\Psi_0 = V_0 = 0$.

3=That flying with wings level implies $\Phi_0 = 0$.

For this trimmed flight state, the aircraft will have particular values of

$$
\begin{aligned}
x &= m\big[u + W_{0q} + Q_0 w - R_0 v + g\cos\ \Theta_0 \theta\big] \\
y &= m[\hat{p} + U_0 r + R_0 u - W_0 p - P_0 w - g\cos\ \Theta_0 \phi] \\
z &= m[w + P_0 v - U_0 q - Q_0 \mu + g\sin\ \Theta_0 \theta]
\end{aligned}
$$

linearization

as we can get linearization as:

(a)

$$
\begin{aligned}
\dot{u} &= X_0 + \Delta x - g(\sin\ \theta_0 + \Delta\theta \cos\ \theta_0) \\
\dot{u} &= X_0 - Mg \sin\theta_0 - g\Delta\theta \cos\ \theta_0 + X_u \Delta u\ + X_w \Delta w + \Delta X_c - w_0 \mathrm{q}
\end{aligned}
$$

(1) $\dot{u} = x_u \Delta u + x_w \Delta w - g\Delta\theta \cos\theta_0 - w_0 \mathrm{q} + \Delta x_c$

(b)

$$
\begin{aligned}
\dot{v} &= y_0 + \Delta y + g\phi \cos\ \theta_0 - u_0 r \\
&= y_v v + y_\rho p + p w_0 + y_\mathrm{r} r + g\phi \cos\ \theta_0 - u_0 r + \Delta y_\mathrm{c}
\end{aligned}
$$

(2) $\dot{V} = y_v v + (y_p + W_0) p + (y_1 \cdot u_0) r + \phi g \cos\ \theta_0 + \Delta y_c$

(c)

$$\dot{w} = \Delta z + z_0 + g(\cos\ \theta_0 - \Delta\theta \sin\ \theta_0) + u_0 q$$
$$= \Delta z - g\Delta\theta \sin\ \theta_0 + u_0 q = z_{\dot{w}}\dot{w} + z_u \Delta u + z_w \Delta w + z_q q - g\Delta\theta \sin\ \theta_0 + u_0 q + \Delta z_c$$

$$(3)\dot{w} = \frac{\left[z_u \Delta u + z_w \Delta w + q\left(z_q + u_0\right) - g\Delta\theta \sin\ \theta_0 + \Delta z_c\right)]}{1 - z_{\dot{w}}}$$

(d) and (e) Find $\dot{p}$ and $\dot{r}$

$$\dot{P} = [L_0 + \Delta L + I_{2x}\dot{r}]/I_x$$
$$\dot{r} = [N_0 + \Delta N + I_{2x}\dot{P}]/I_2$$

We can substitute each other, $L_0 : N_0 = 0$.

$$\dot{P} = \left(I_x I_2 - I_{x_2}^2\right)^{-1}\left(I_2 \Delta L + I_{x_2}\Delta N\right)$$
$$\dot{r} = \left(I_x I_2 - I_{I_2}'\right)^{-1}(I_{2x}\Delta L + I_x \Delta N)$$

We as write them as:

$$\dot{P} = G\left(\frac{\Delta L}{I_x} + \frac{I_{x2}}{I_x I_2}\Delta N\right)$$
$$\dot{r} = G\left(\frac{\Delta N}{I_2} + \frac{I_{2x}}{I_x I_2}\Delta L\right)$$

G: is defined in task 3

[1] but based on our definition we take ΔL contains I_x and ΔN contains I_z

[2]

$$\Delta L = l_v \Delta V + l_p \Delta P + l_r \Delta r + \Delta L_c$$
$$\Delta N = N_v \Delta V + N_\rho \Delta P + N_r \Delta r + \Delta N_c$$

[3] from tasks 3 and Naca report We get: $L_{p'}, L_{r'}, \dots \dots$

so from [1], [2], [3] We can write

$$4)\dot{P} = L_{v'}\Delta V + L_{p'}\Delta P + L_{r'}\Delta r + \Delta L_c$$
$$5)\dot{r} = N_{v'}\Delta V + N_{p'}\Delta P' + N_{r'}\Delta r + \Delta N_c$$

(F)

$\dot{q} = \frac{\Delta M}{I_y}$, but based on our definition

$$\dot{q} = \Delta M = M_u \Delta u + M_w \Delta w_+ M_{\dot{w}} \Delta \dot{w} + M_q \Delta q + \Delta M_c$$

and from (c) we get $\dot{w}$ so

$$6)\dot{q} = \left(M_u + \frac{M_{\dot{w}} Z_u}{1 - Z_{\dot{w}}}\right)\Delta u + \left(M_w + \frac{M_{\dot{w}} Z_w}{1 - Z_w}\right)\Delta w + \left(M_q + \frac{M_{\dot{w}}\left(Z_q + U_0\right)}{1 - Z_{\dot{w}}}\right)\Delta q - \frac{M_{\dot{w}} g \sin\ \theta_0 \Delta\theta}{1 - Z_{\dot{w}}}$$

$$+\Delta M_c + \frac{M_{\dot{w}} \Delta Z_c}{1 - Z_{\dot{w}}}$$

(g) $\phi = p + r\ tan\theta_0$

(h) $\dot{\theta} = q$

(i) $\dot{\psi} = r\sec\ \theta_0$

Longitudinal in state space form

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_w & -w_0 & -g\cos\theta_0 \\ \dfrac{Z_u}{1 - Z_{\dot{w}}} & \dfrac{Z_w}{1 - Z_{\dot{w}}} & \dfrac{Z_q + u_0}{1 - Z_{\dot{w}}} & \dfrac{-g\sin\theta_0}{1 - Z_{\dot{w}}} \\ M_u + \dfrac{M_{\dot{w}} Z_u}{1 - Z_{\dot{w}}} & M_w + M_{\dot{w}} \dfrac{Z_w}{1 - Z_{\dot{w}}} & M_q + M_{\dot{w}} \dfrac{u_0}{1 - Z_{\dot{w}}} & \dfrac{-M_{\dot{w}} g\sin\theta_0}{1 - Z_{\dot{w}}} \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix}$$

$$+ \begin{bmatrix} X_{\delta_e} & X_{\delta_T} \\ \dfrac{Z_{\delta_e}}{1 - Z_{\dot{w}}} & \dfrac{Z_{\delta_T}}{1 - Z_{\dot{w}}} \\ M_\delta + \dfrac{M_{\dot{w}} Z_{\delta_e}}{1 - Z_{\dot{w}}} & M_{\delta_T} + \dfrac{M_{\dot{w}} Z_{\delta_T}}{1 - Z_{\dot{w}}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_T \end{bmatrix}$$

Full Lateral Mode (v, p_1, ϕ, r) $in\ state\ space\ form$

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_p + W_0 & Y_r - u_0 & g\cos\left(\theta_0\right) & 0 \\ L_{v'} & L_{p'} & L_{r'} & 0 & 0 \\ N_v' & N_p' & N_r' & 0 & 0 \\ 0 & 1 & \tan\theta_0 & 0 & 0 \\ 0 & 0 & 1/cos\theta_0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} Y_{\delta_a} & Y_{\delta_R} \\ L_{\delta_A}' & L_{\delta_R}' \\ N_{\delta_A}' & N_{\delta_R}' \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_A \\ \delta_R \end{bmatrix}$$

Linearization of sideslip angle

$$\Delta\beta = \tan^{-1}\frac{\Delta v}{u_0} = \frac{\Delta v}{u_0}\ \Delta v = \Delta\beta u_0$$

$$\begin{bmatrix}\dot{\beta}\\ \hat{p}\\ \dot{r}\\ \dot{\phi}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix} Y_\beta/u_0 & (Y_p + W_0)/u_0 & \frac{Y_r}{u_0} - 1 & g\cos(\theta_0)/u_0 & 0\\ L_\beta{}' & L_p{}' & L_r{}' & 0 & 0\\ N_\beta{}' & N_p{}' & N_r{}' & 0 & 0\\ 0 & 1 & \tan\theta_0 & 0 & 0\\ 0 & 0 & 1/cos\theta_0 & 0 & 0\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\\ \phi\\ \psi\end{bmatrix} + \begin{bmatrix} Y_{\delta_a}{}^* & Y_{\delta_R}{}^*\\ L_{\delta_a}{}' & L_{\delta_R}{}'\\ N_{\delta_a}{}' & N_{\delta_R}{}'\\ 0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_R\end{bmatrix}$$

Longitudinal Approx:

For long Period approx. [2*2]:

$$\dot{w} = 0$$
$$z_u u + \left(z_q + u_0\right)q - g\ \sin\theta_0\theta_t + z_{\delta_e}\delta_e$$
$$+z_{\sigma_T}\delta_T = 0$$
$$\sin\ \theta_0 = 0$$
$$\therefore q = \frac{-z_u u \cdot z_{\delta_e}\delta_e - z_{\delta_r}\delta_T}{z_q + u_0} = \dot{\theta}$$

So we can express as:

$$\begin{bmatrix}\dot{u}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix} \mathrm{X}_u & -g\\ -\frac{Z_u}{Z_q+U_0} & 0\end{bmatrix}\begin{bmatrix}u\\ \theta\end{bmatrix} + \begin{bmatrix} X_{\delta_e} & X_{\delta_t}\\ \frac{-Z_{\delta_e}}{Z_q+U_0} & \frac{-Z_{\delta T}}{Z_q+U_0}\end{bmatrix}\begin{bmatrix}\delta_e\\ \delta_T\end{bmatrix}$$

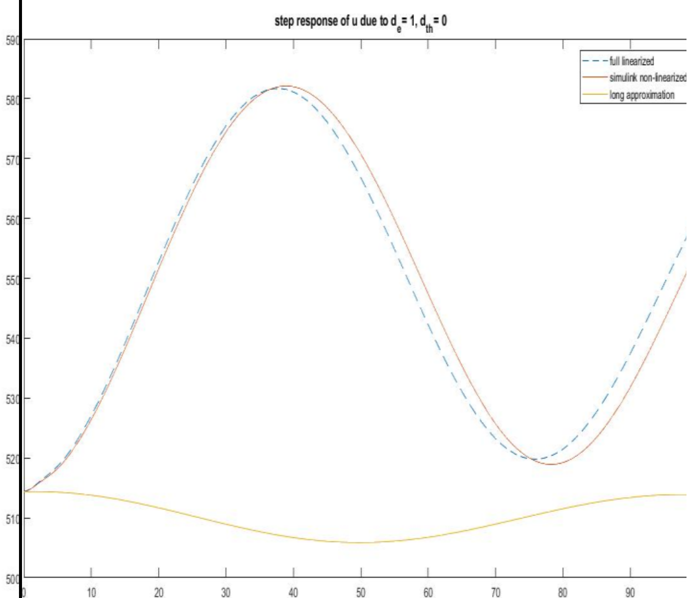
For short Period Approx [2*2]:

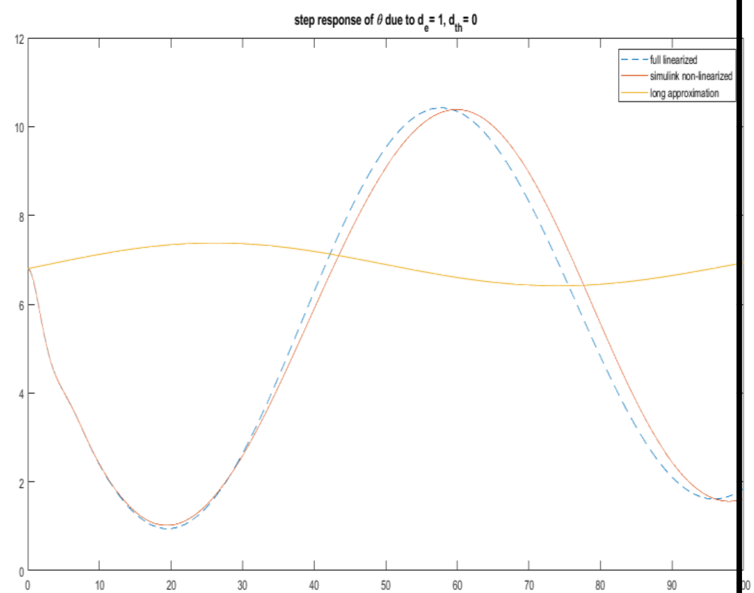
$$\Delta u = 0$$

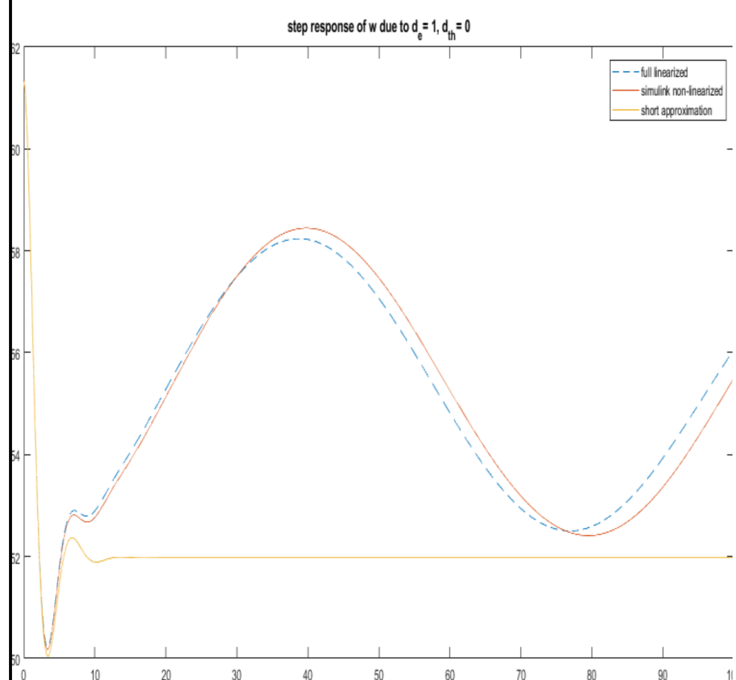
So we can express as:

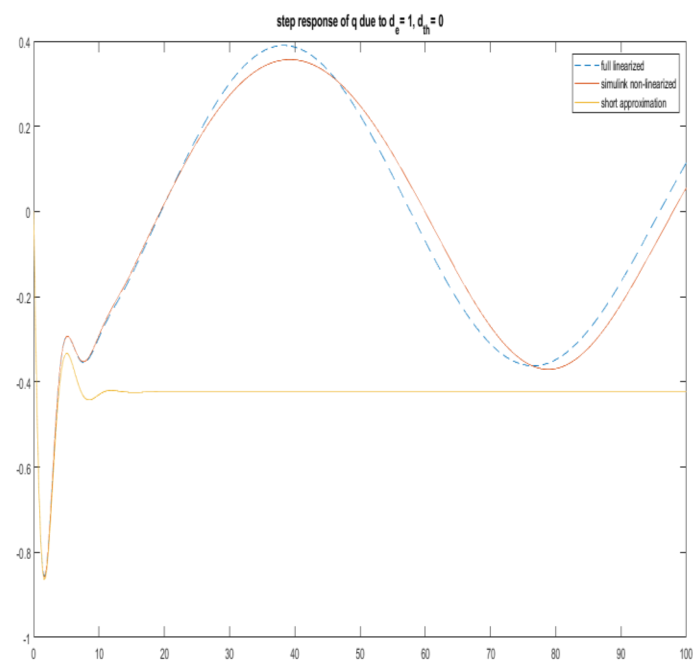
$$\begin{bmatrix}\dot{\omega}\\ \dot{q}\end{bmatrix} = \begin{bmatrix} \frac{Z_w}{1 - Z_{\dot{w}}} & \frac{Z_q + U_0}{1 - Z_{\dot{w}}}\\ M_\omega + \frac{M_{\dot{w}}Z_w}{1 - Z_{\dot{w}}} & M_q + \frac{M_{\dot{w}}\left(Z_q + u_0\right)}{1 - Z_{\dot{w}}}\end{bmatrix}\begin{bmatrix}w\\ q\end{bmatrix} + \begin{bmatrix} \frac{Z_{\delta_e}}{1 - Z_{\dot{w}}} & \frac{Z_{\delta T}}{1 - Z_i}\\ M_{\delta_e} + \frac{M_{\dot{w}}Z_{\delta_e}}{1 - Z_{\dot{w}}} & M_{\delta_t} + \frac{M_{\dot{w}}Z_{\delta T}}{1 - Z_{\dot{w}}}\end{bmatrix}\begin{bmatrix}\delta_e\\ \delta_T\end{bmatrix}$$

Step Response of longitudinal dynamics due to $d_e = 1, d_{th} = 0$:

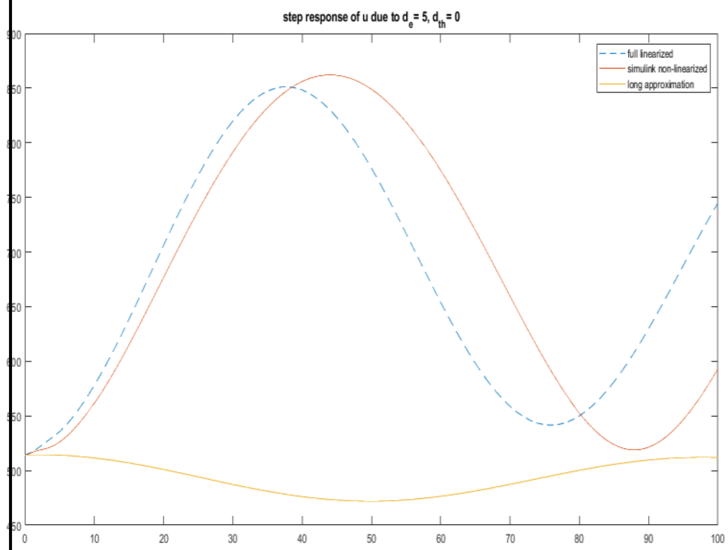


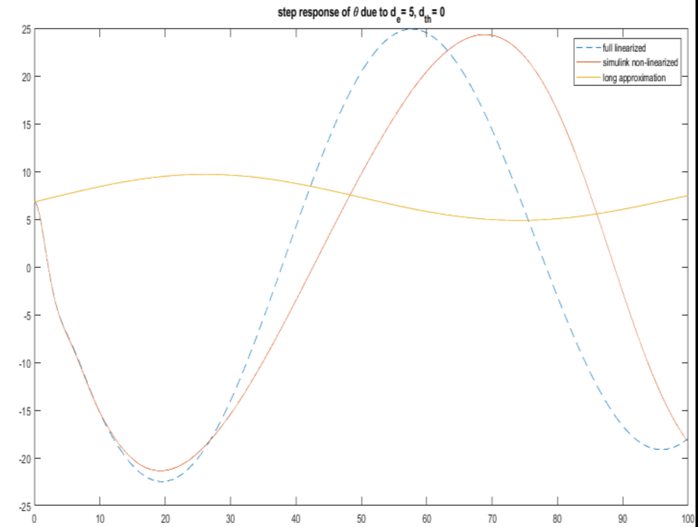


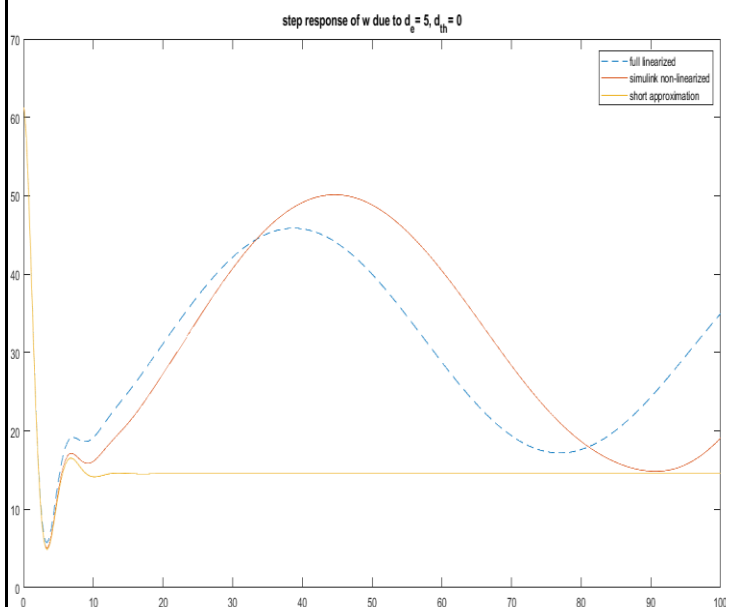


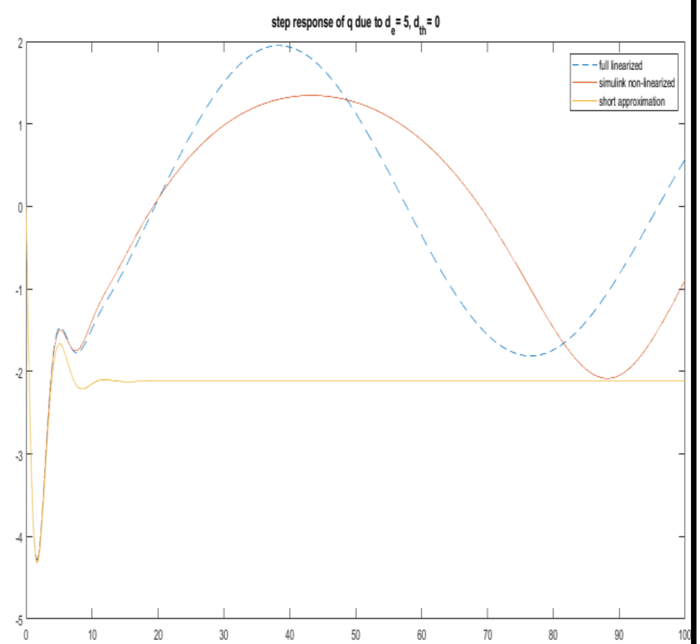


Step Response of longitudinal dynamics due to $d_e = 5, d_{th} = 0$:

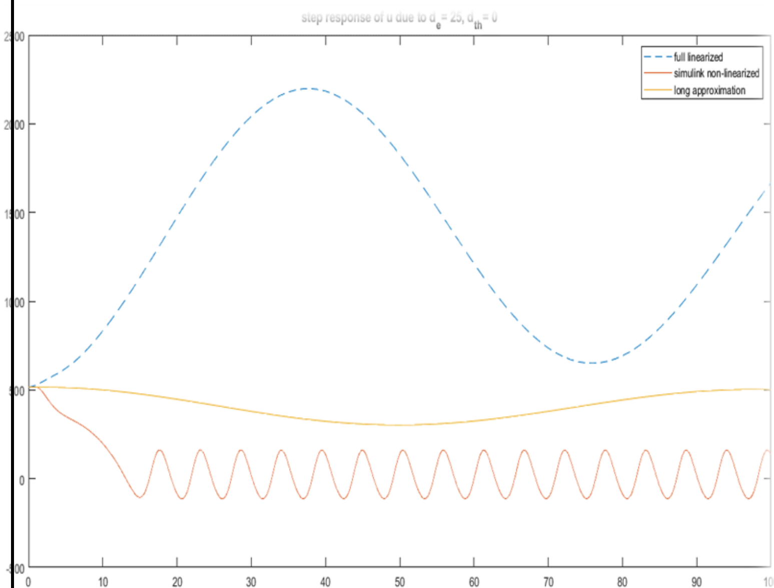


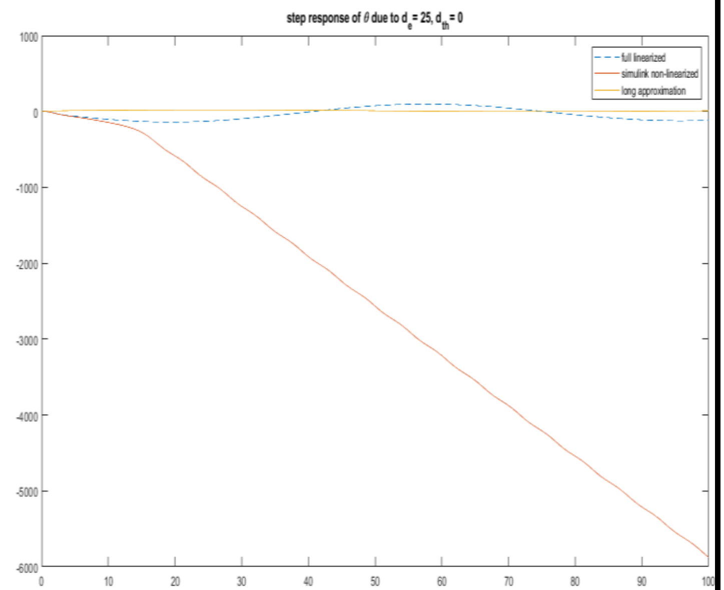


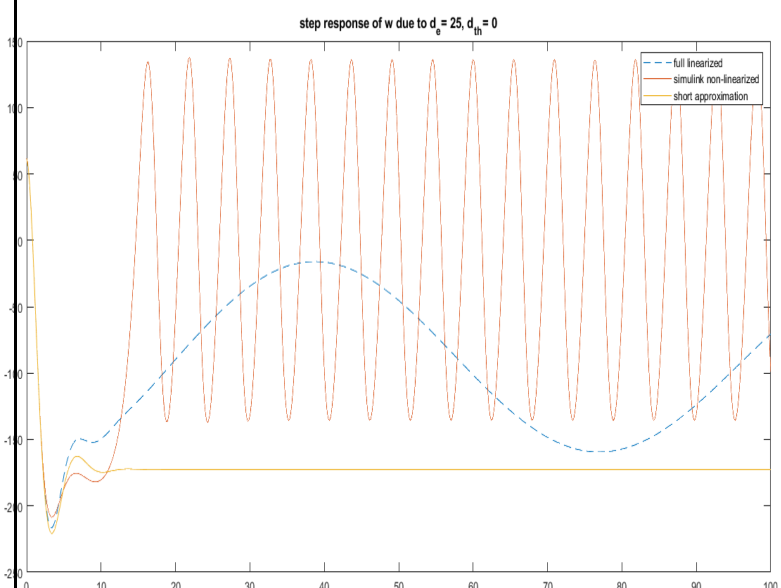


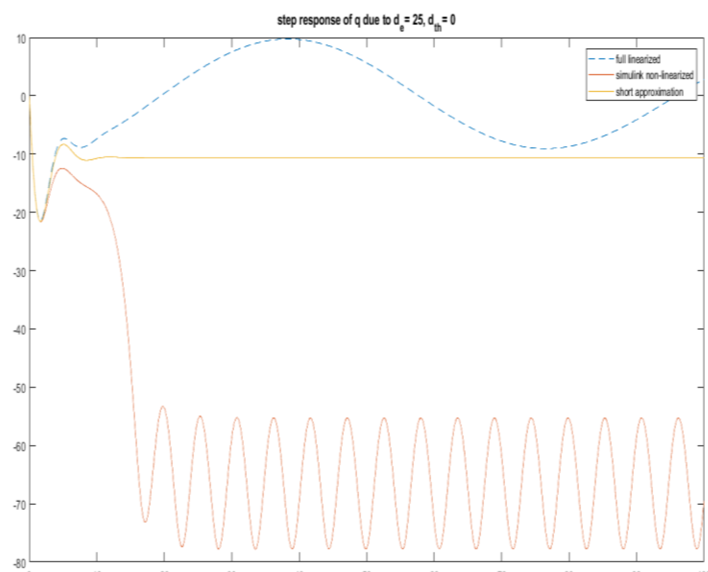


Step Response of longitudinal dynamics due to $d_e = 25, d_{th} = 0$:

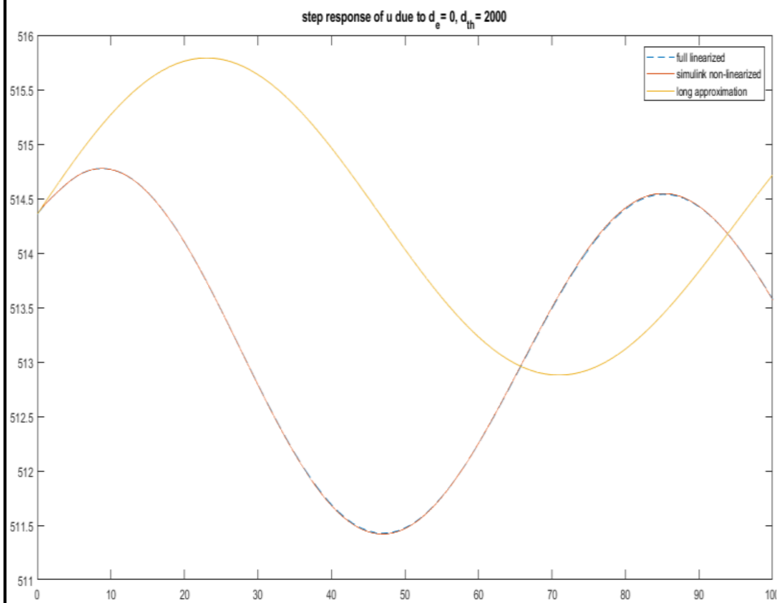


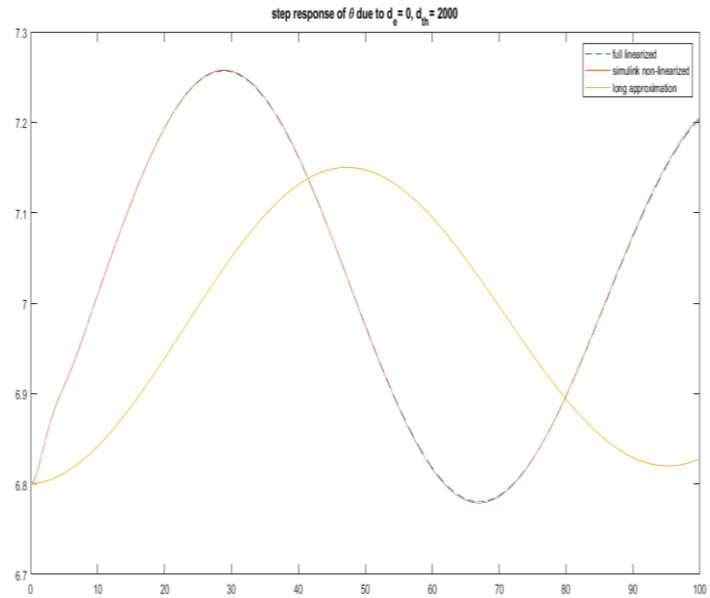


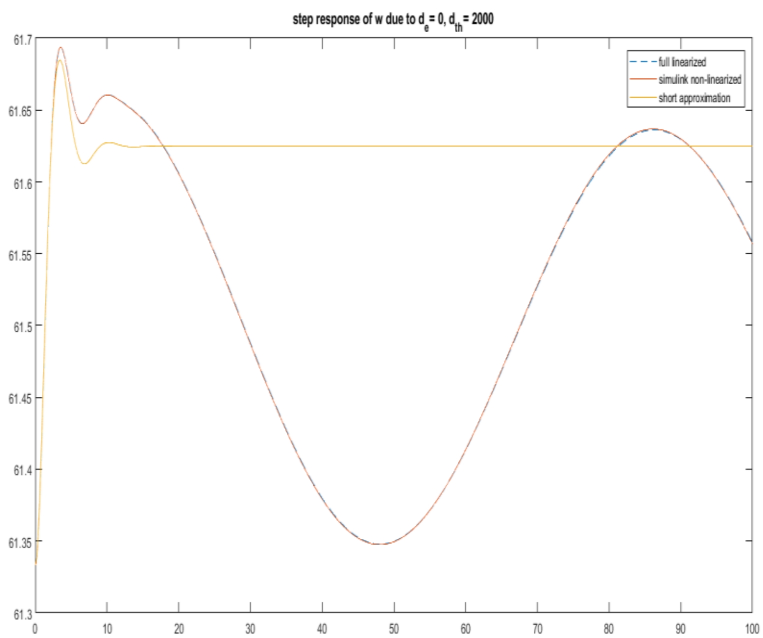


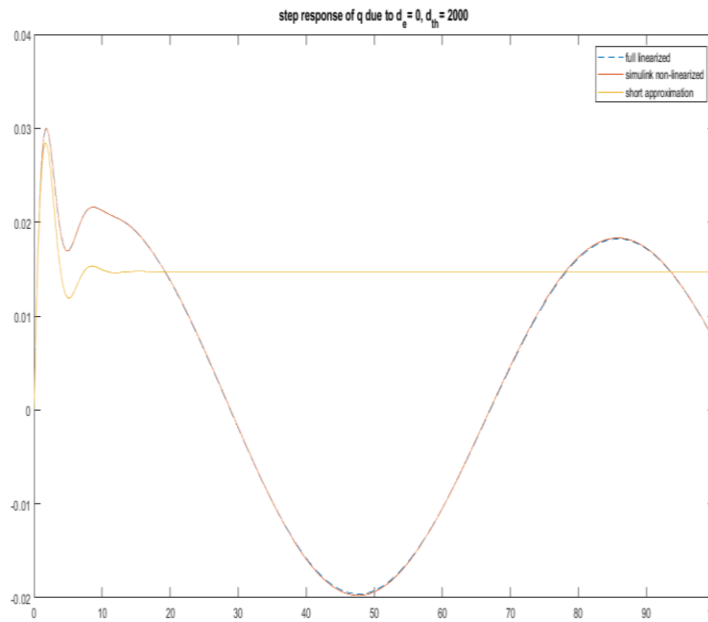


Step Response of longitudinal dynamics due to $d_e = 0, d_{th} = 2000$

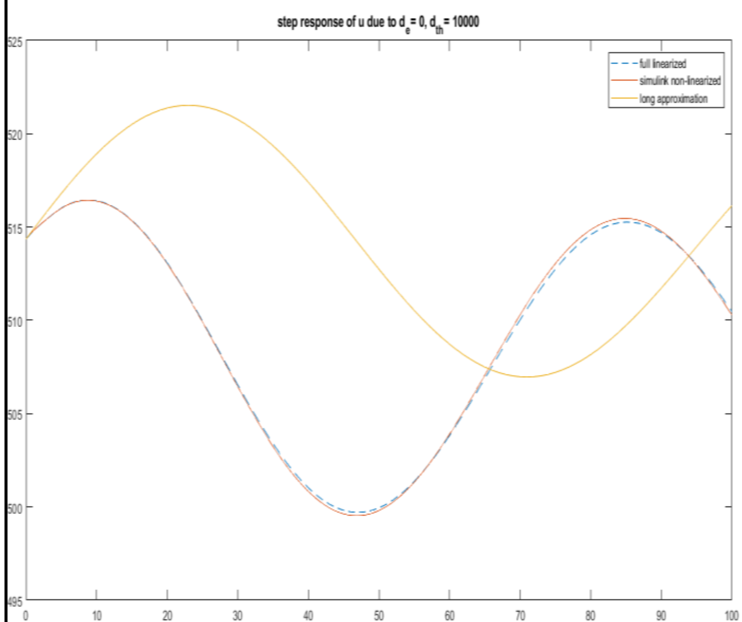


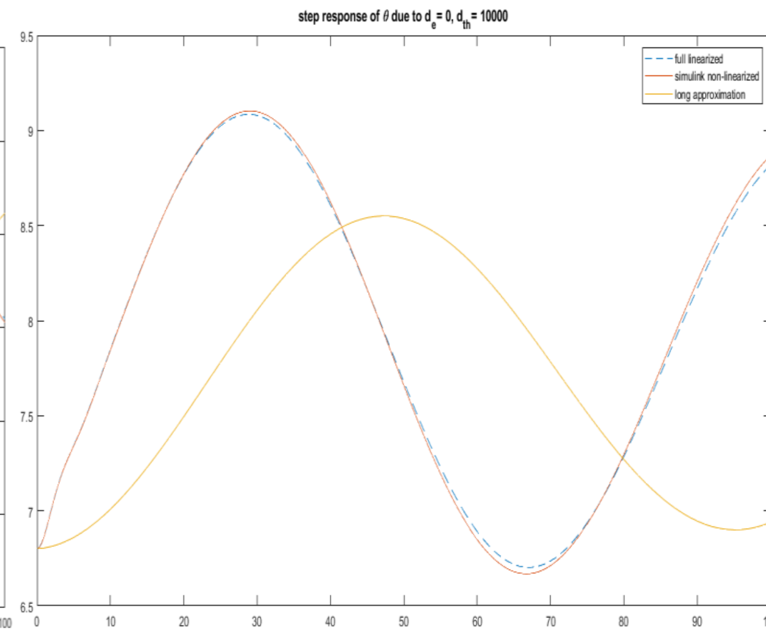


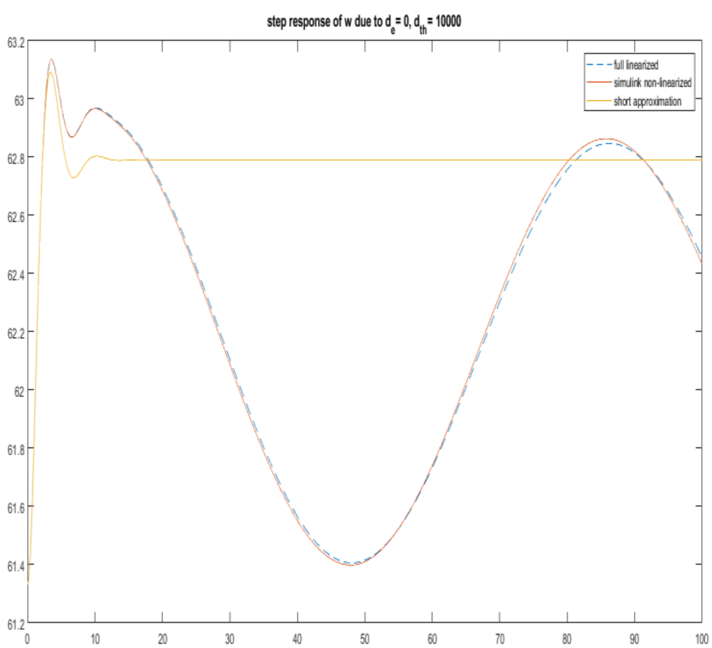


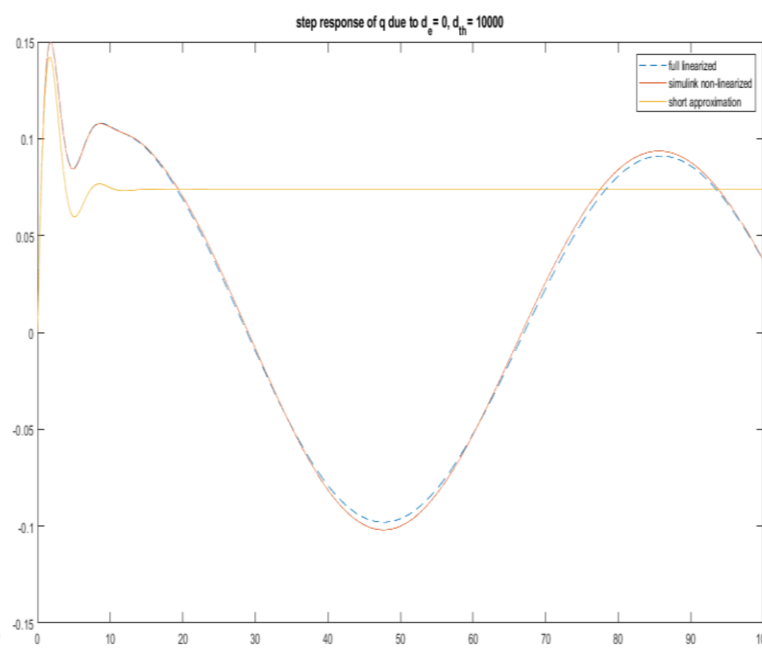


Step Response of longitudinal dynamics due to $d_e = 0, d_{th} = 10000$:

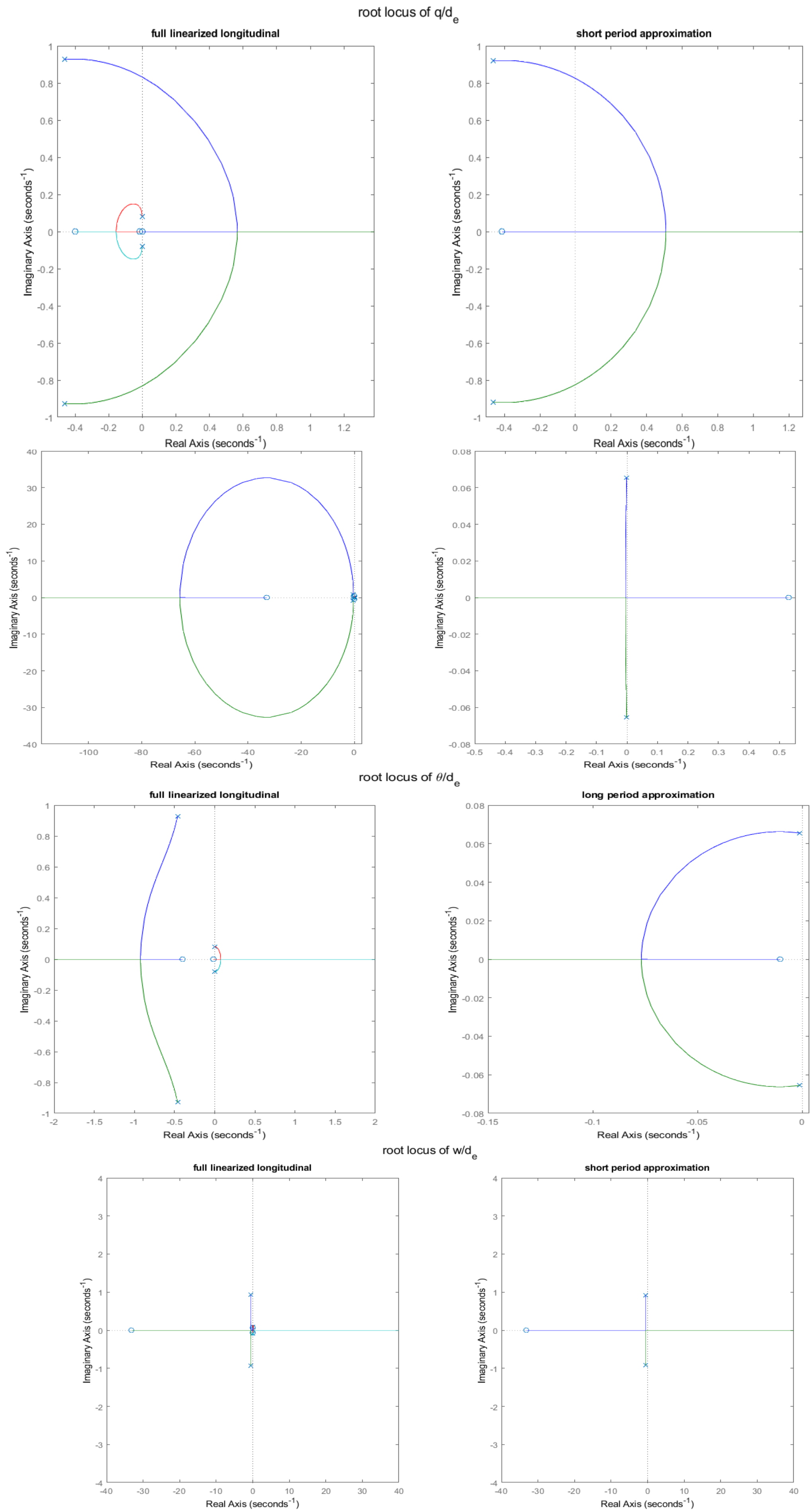


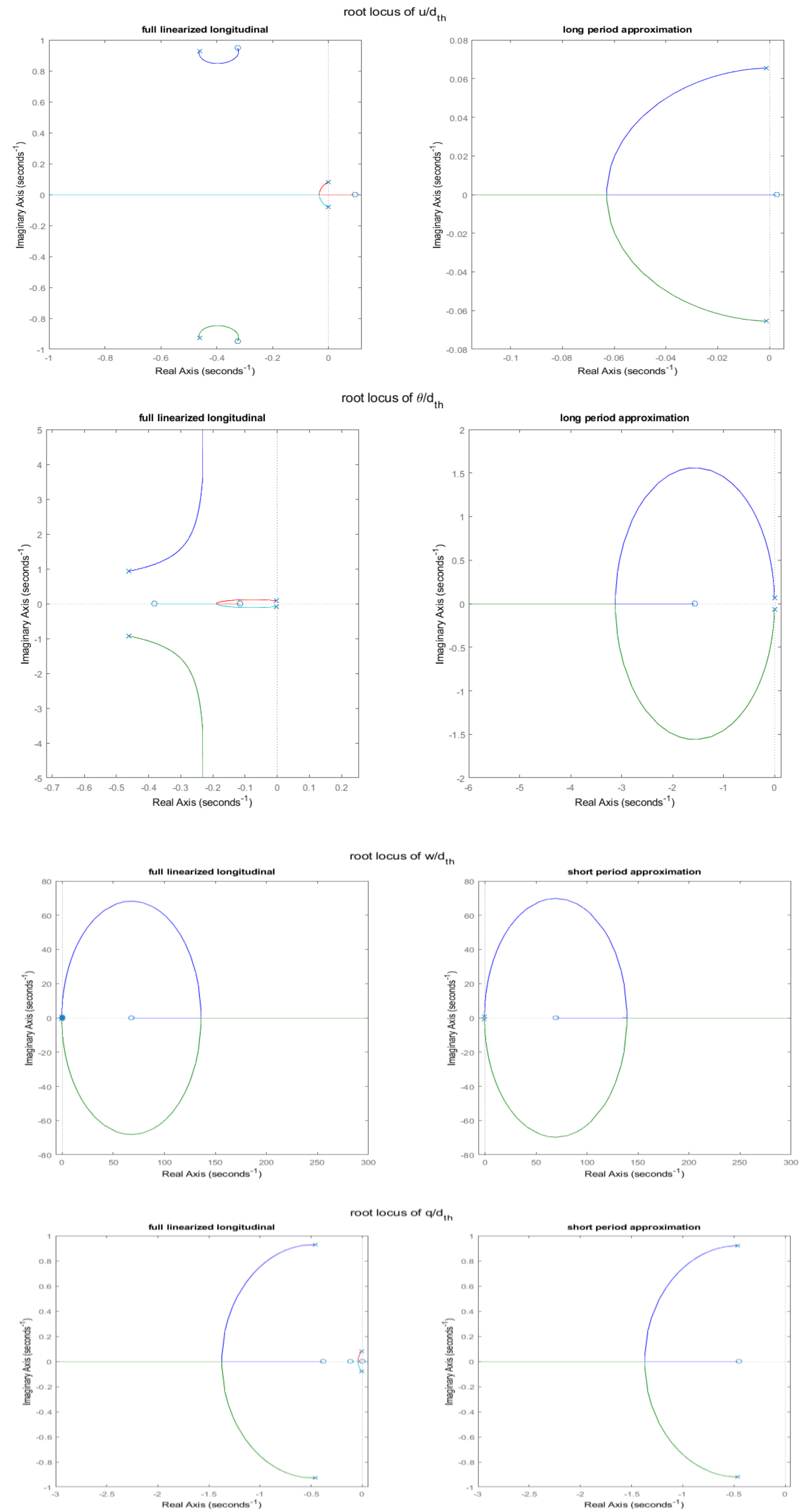




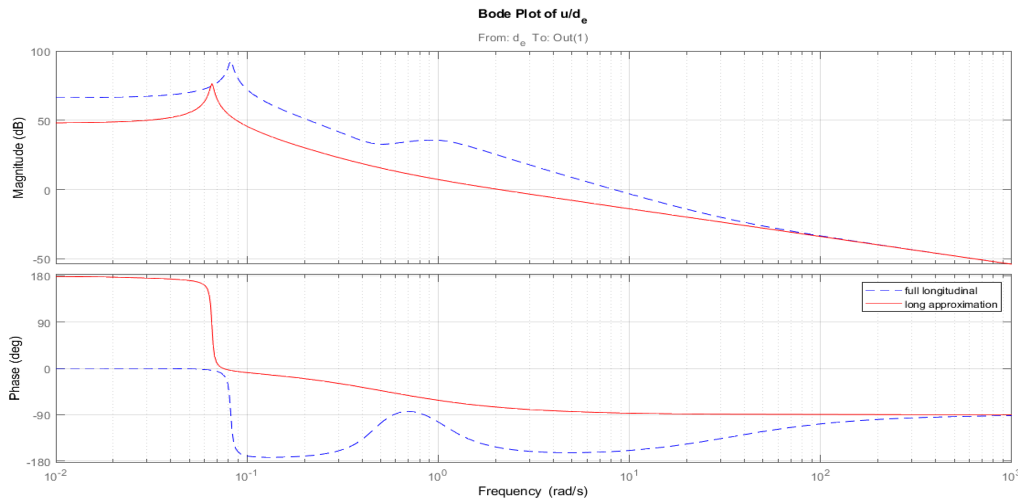


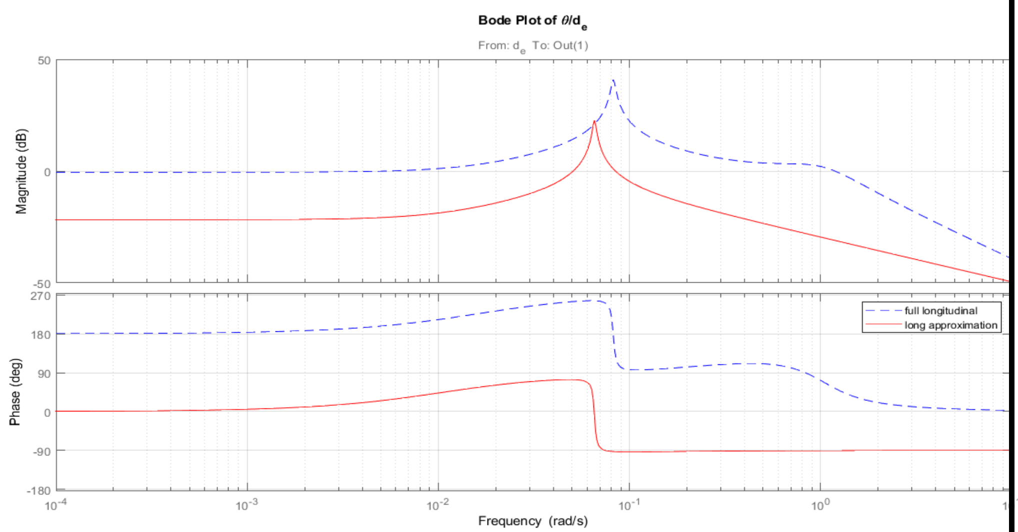
Root Locus:

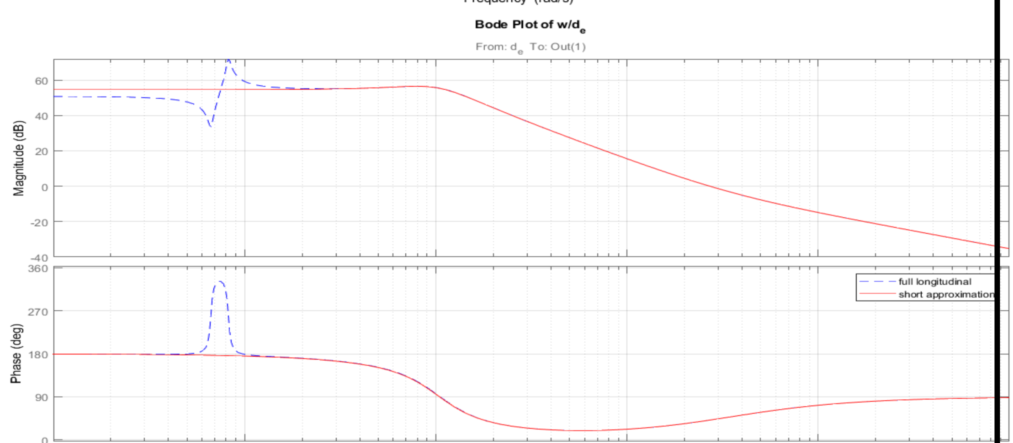

root locus of q/d_e
full linearized longitudinal
short period approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
root locus of θ/d_e
full linearized longitudinal
long period approximation
root locus of w/d_e
full linearized longitudinal
short period approximation

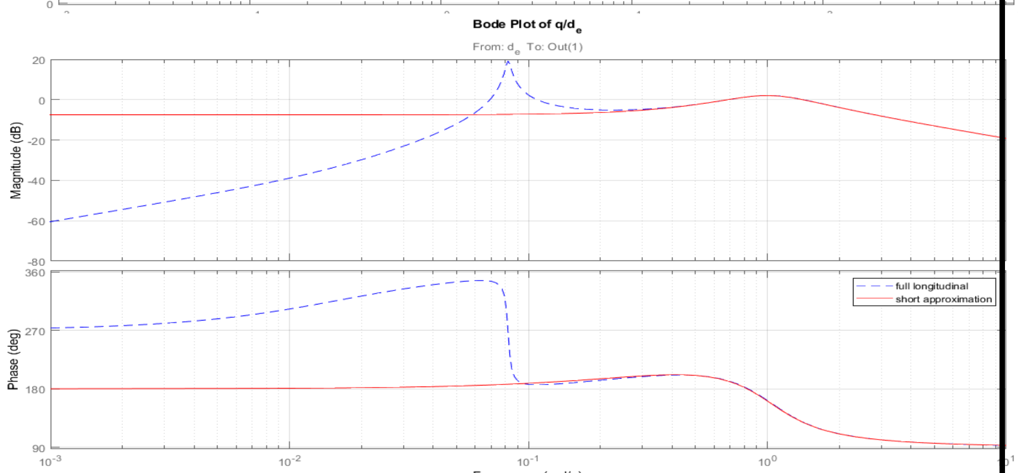

root locus of u/d_th
full linearized longitudinal
long period approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
root locus of θ/d_th
full linearized longitudinal
long period approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
root locus of w/d_th
full linearized longitudinal
short period approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
root locus of q/d_th
full linearized longitudinal
short period approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)

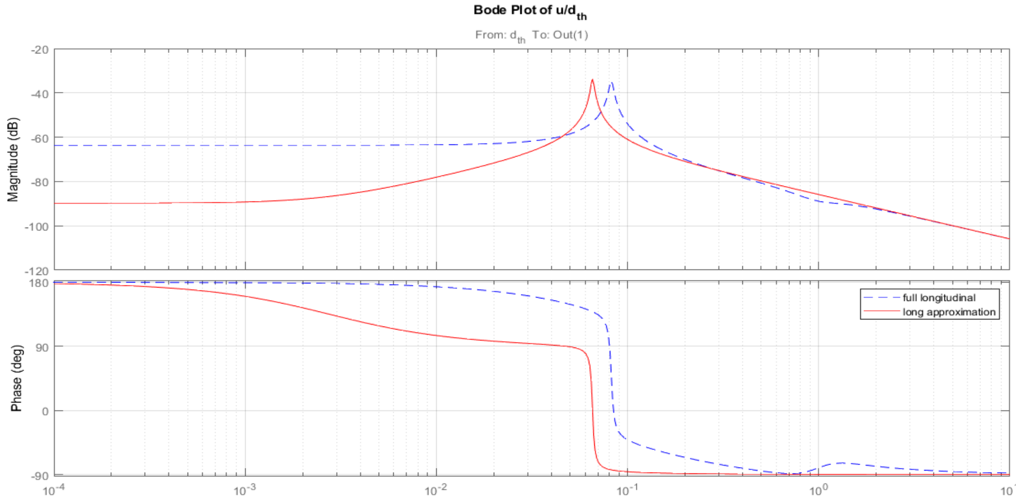
Bode Blot:

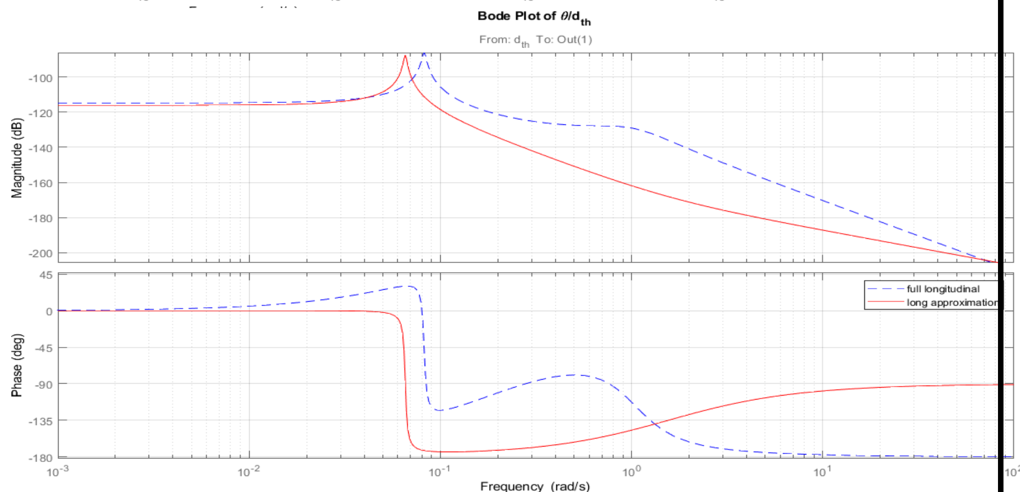

Bode Plot of u/d$_e$
From: d$_e$ To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
long approximation
Frequency (rad/s)

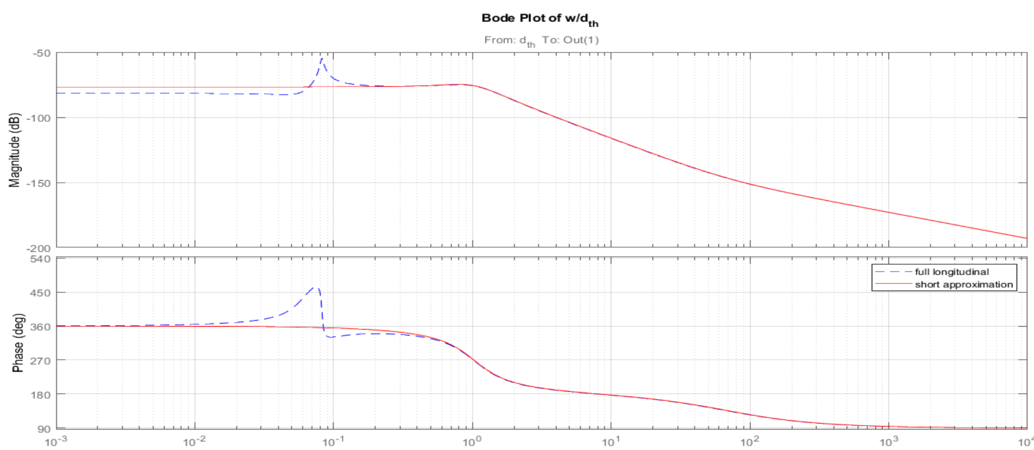

Bode Plot of θ/d$_e$
From: d$_e$ To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
long approximation
Frequency (rad/s)

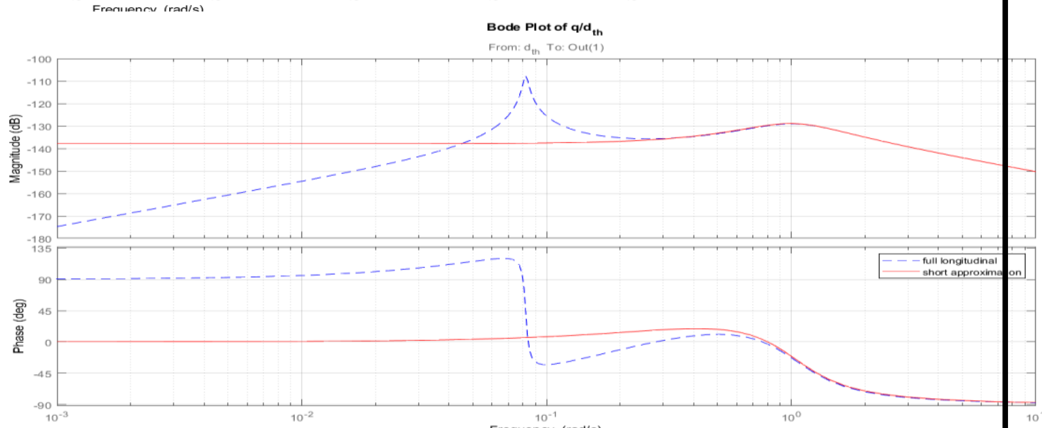

Bode Plot of w/d$_e$
From: d$_e$ To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
short approximation


Bode Plot of q/d$_e$
From: d$_e$ To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
short approximation
Frequency (rad/s)


Bode Plot of u/d_th
From: d_th To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
long approximation


Bode Plot of θ/d_th
From: d_th To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
long approximation
Frequency (rad/s)


Bode Plot of w/d_th
From: d_th To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
short approximation


Bode Plot of q/d_th
From: d_th To: Out(1)
Magnitude (dB)
Phase (deg)
full longitudinal
short approximation
Frequency (rad/s)

Task 4 part 2: Lateral dynamics

Overview

We get the linearized equations of lateral dynamics in state space form:

$$\begin{bmatrix}\dot{\beta}\\ \hat{p}\\ \dot{r}\\ \dot{\phi}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}\frac{Y_\beta}{u_0} & \frac{Y_p+W_0}{u_0} & \frac{Y_r}{u_0}-1 & \frac{g cos(\theta_0)}{u_0} & 0\\ L_\beta{}' & L_p{}' & L_r{}' & 0 & 0\\ N_\beta{}' & N_p{}' & N_r{}' & 0 & 0\\ 0 & 1 & tan\theta_0 & 0 & 0\\ 0 & 0 & \frac{1}{cos\theta_0} & 0 & 0\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\\ \phi\\ \psi\end{bmatrix}+\begin{bmatrix}Y_{\delta_a}{}^* & Y_{\delta_R}{}^*\\ L_{\delta_a}{}' & L_{\delta_R}{}'\\ N_{\delta_a}{}' & N_{\delta_R}{}'\\ 0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_R\end{bmatrix}$$

Approximation:

1) 3 DOF spiral

- Neglect side force equations.
- $\Delta\beta$=0

$$\begin{bmatrix}\dot{p}\\ \dot{r}\\ \dot{\phi}\end{bmatrix}=\begin{bmatrix}L_p{}' & L_r{}' & 0\\ N_p{}' & N_r{}' & 0\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}p\\ r\\ \phi\end{bmatrix}+\begin{bmatrix}L_{\delta_r}{}'\\ N_{\delta_r}{}'\\ 0\end{bmatrix}\delta_r$$

2) 3 DOF Dutch Roll

- Neglect $g\cos\,(\theta_0)/V_t$ for moderate and high speed.
- Neglect effect of yaw rate on rolling acc. $L_r' r=0$
- Neglect effect of roll rate on yawing acc. $N_p' p=0$

$$\begin{bmatrix}\dot{\beta}\\ \dot{p}\\ \dot{r}\end{bmatrix}=\begin{bmatrix}Y_v & 0 & -1\\ L_\beta{}' & L_p{}' & 0\\ N_\beta{}' & 0 & N_r{}'\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\end{bmatrix}+\begin{bmatrix}Y_{\delta_a}{}^* & Y_{\delta_r}{}^*\\ L_{\delta_a}{}' & L_{\delta_r}{}'\\ N_{\delta_a}{}' & N_{\delta_r}{}'\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_r\end{bmatrix}$$

3) 2 DOF Dutch Roll

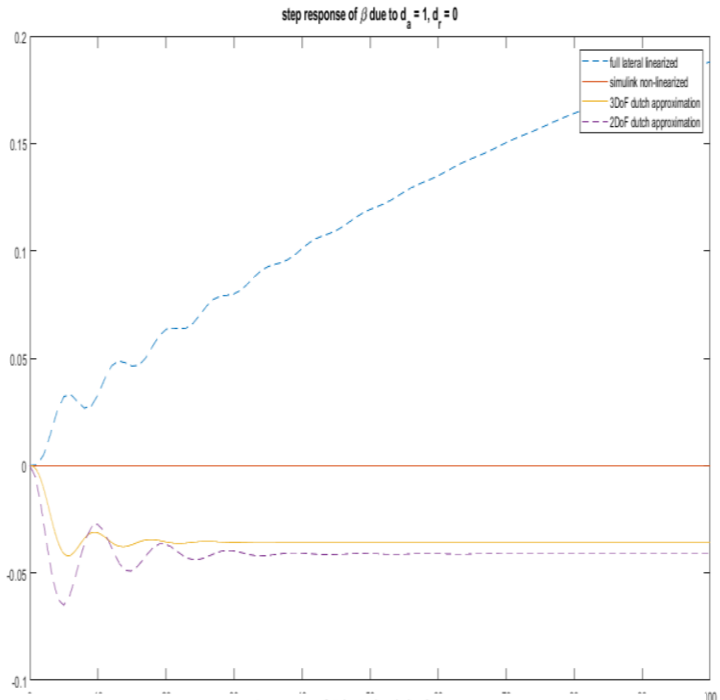
- Neglect rolling moment equation.
- $\Delta\phi=0$

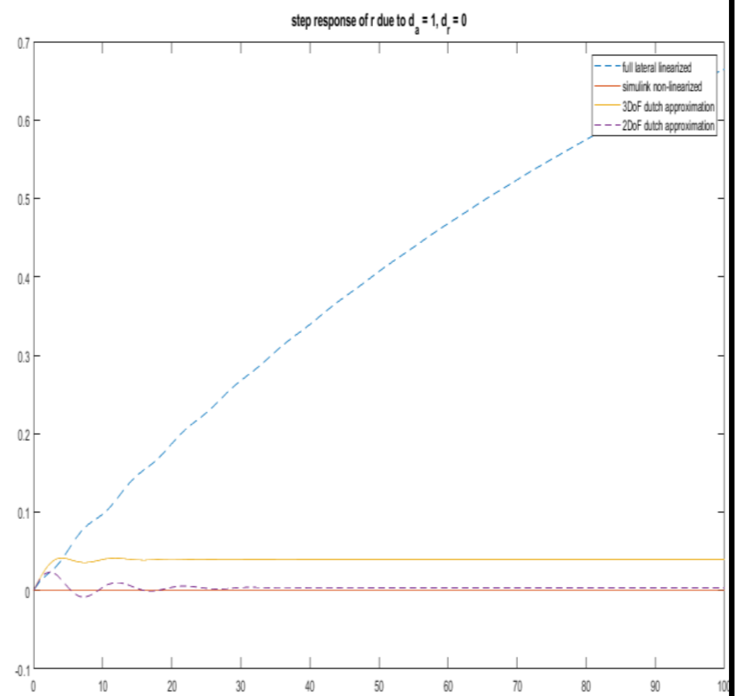
$$\begin{bmatrix}\dot{\beta}\\ \dot{r}\end{bmatrix}=\begin{bmatrix}Y_v & \frac{Y_r}{U_0}-1\\ N_\beta{}' & N_r{}'\end{bmatrix}\begin{bmatrix}\beta\\ r\end{bmatrix}+\begin{bmatrix}Y_{\delta_a}{}^* & Y_{\delta_r}{}^*\\ N_{\delta_a}{}' & N_{\delta_r}{}'\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_r\end{bmatrix}$$

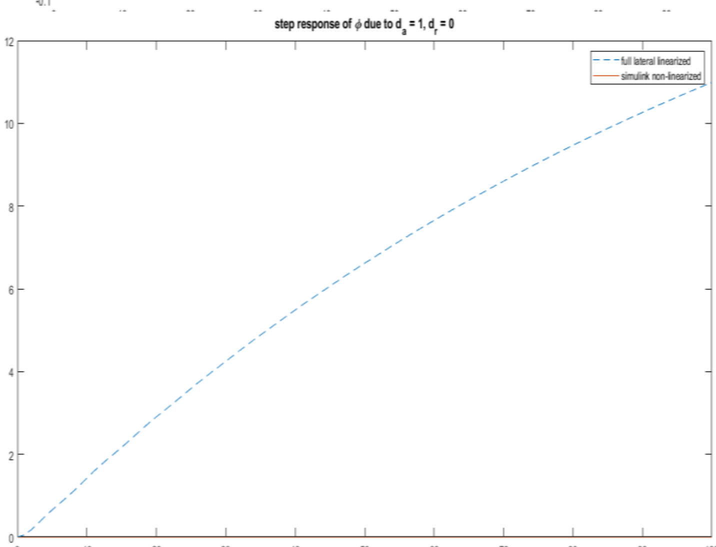
4) 1 DOF Roll

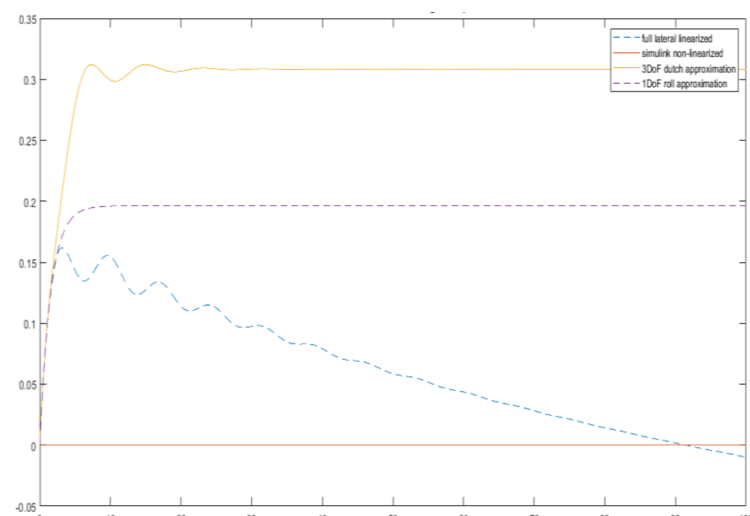
- $\dot{p} - L'_p p = L_{\delta_a}{}' \delta_a$

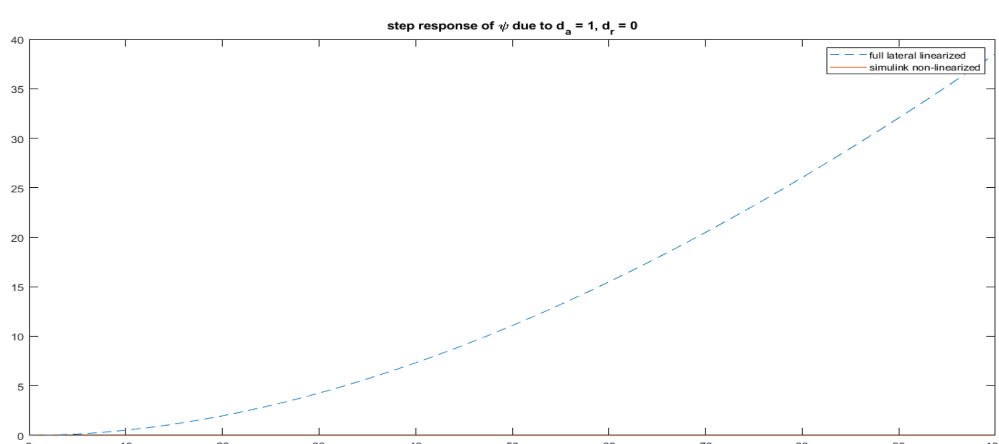
Step Response

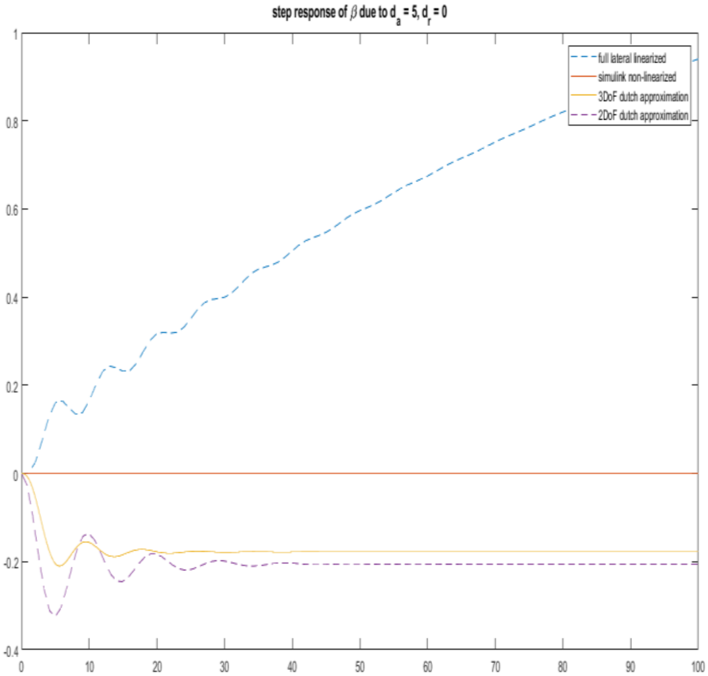


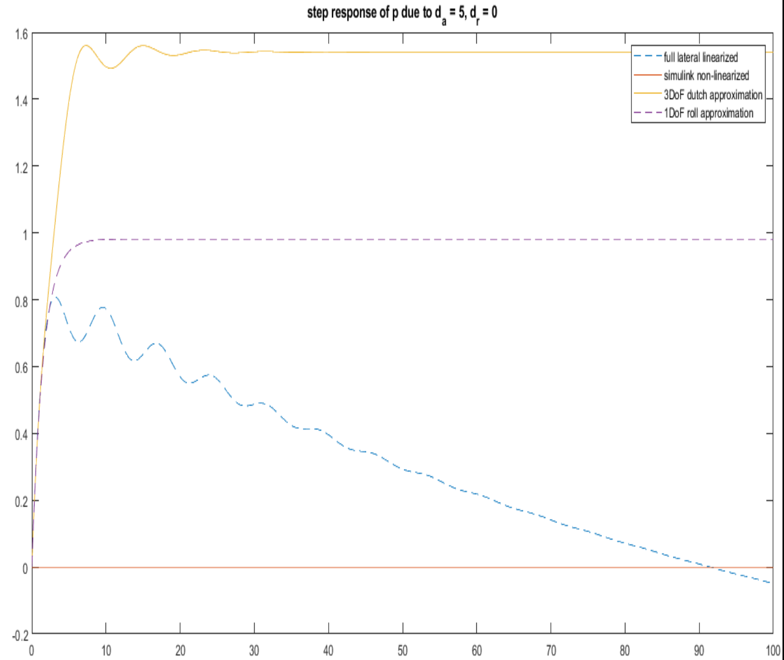


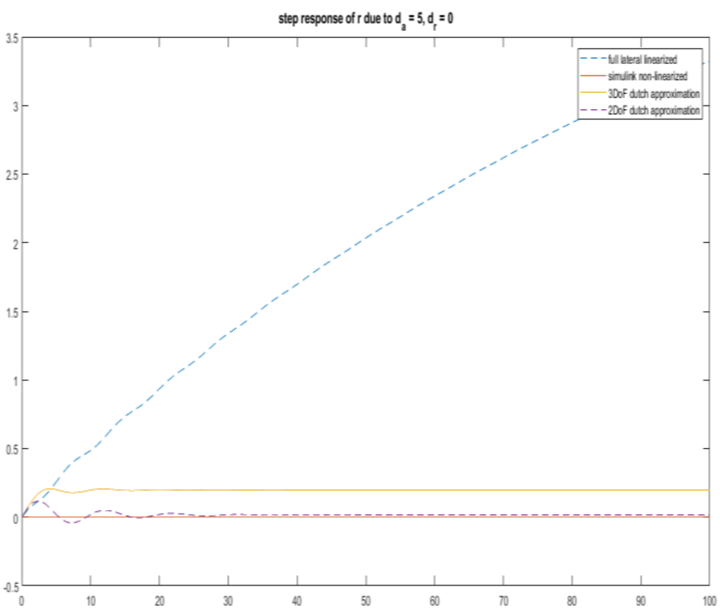


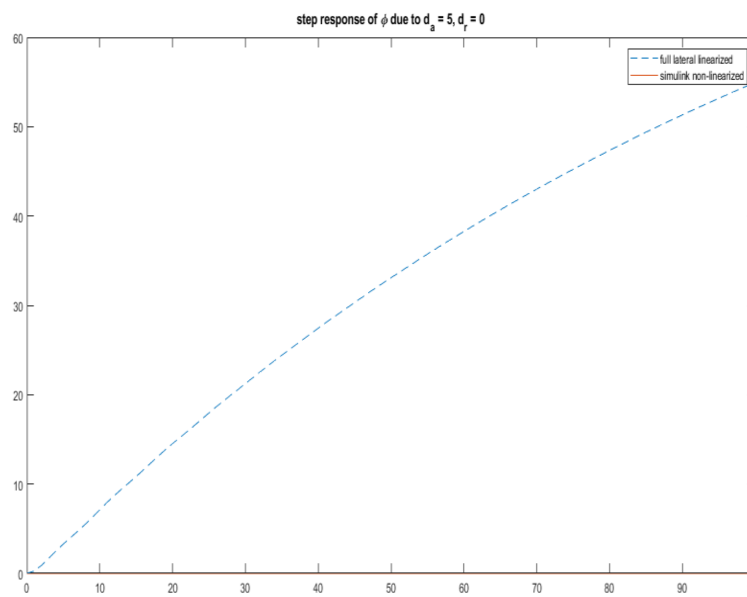


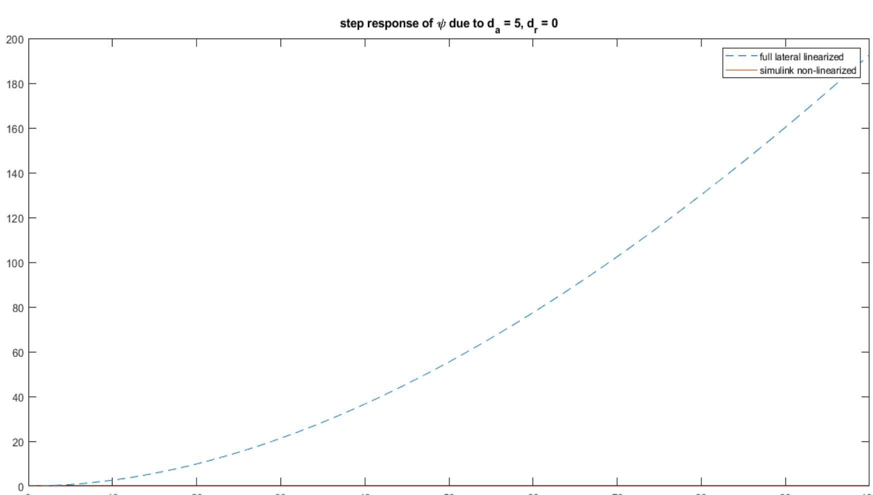


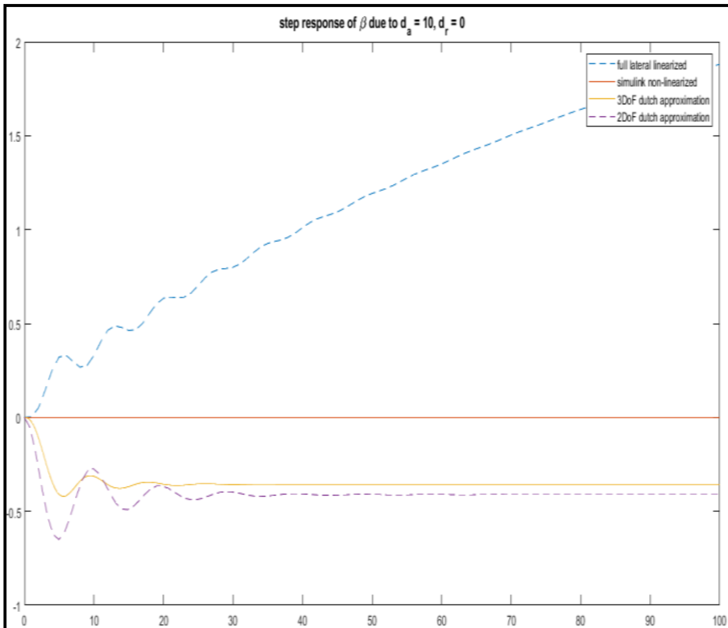

step response of β due to d_a = 5, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

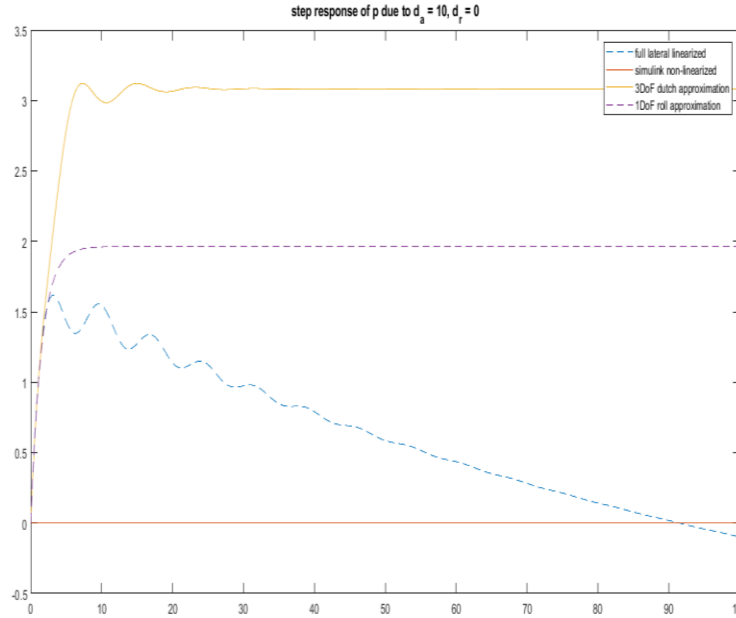

step response of p due to d_a = 5, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
1DoF roll approximation

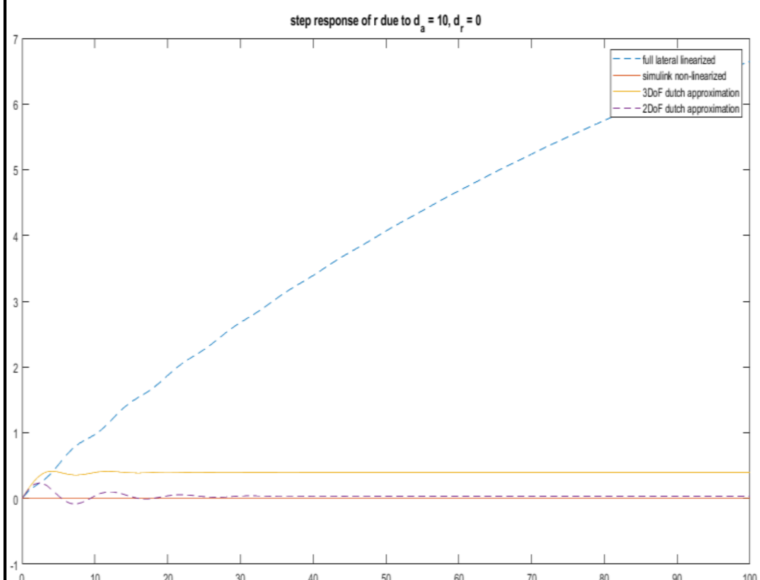

step response of r due to d_a = 5, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

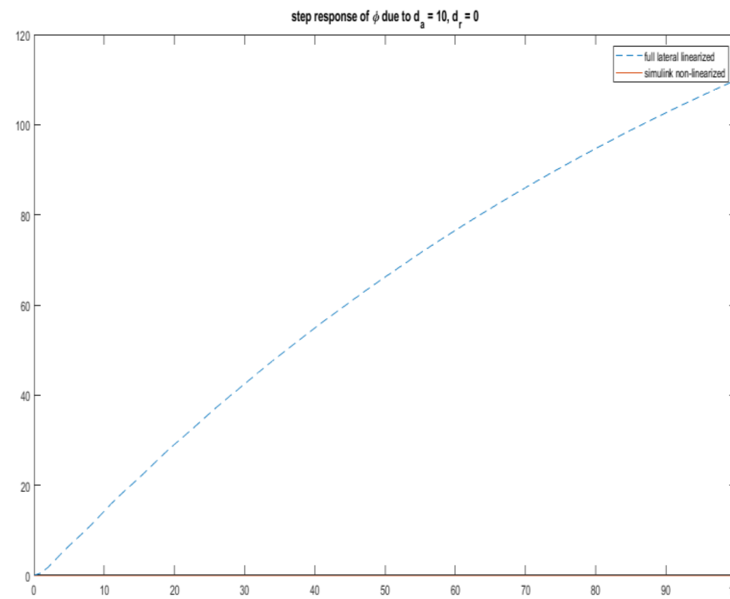

step response of ϕ due to d_a = 5, d_r = 0
full lateral linearized
simulink non-linearized

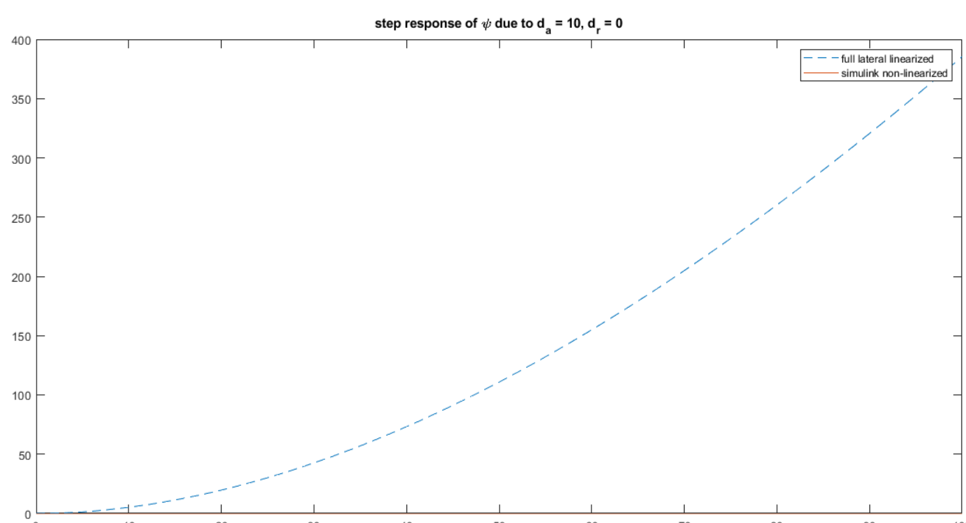

step response of ψ due to d_a = 5, d_r = 0
full lateral linearized
simulink non-linearized

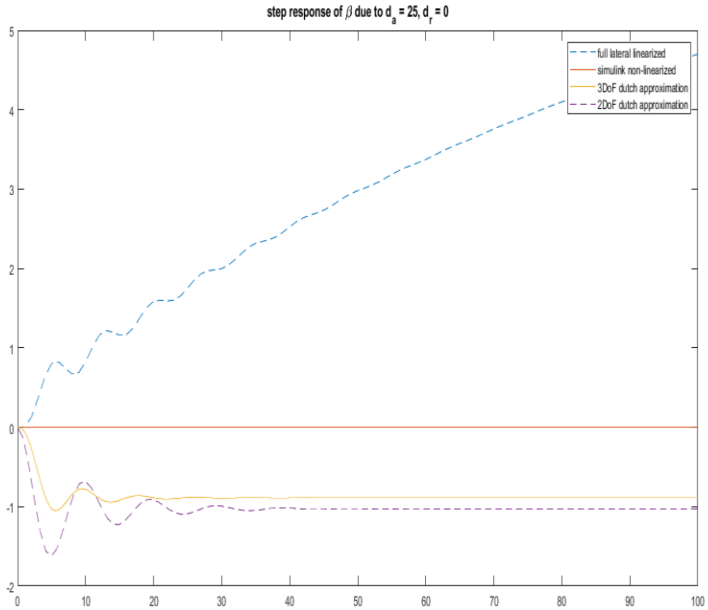

step response of β due to d_a = 10, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

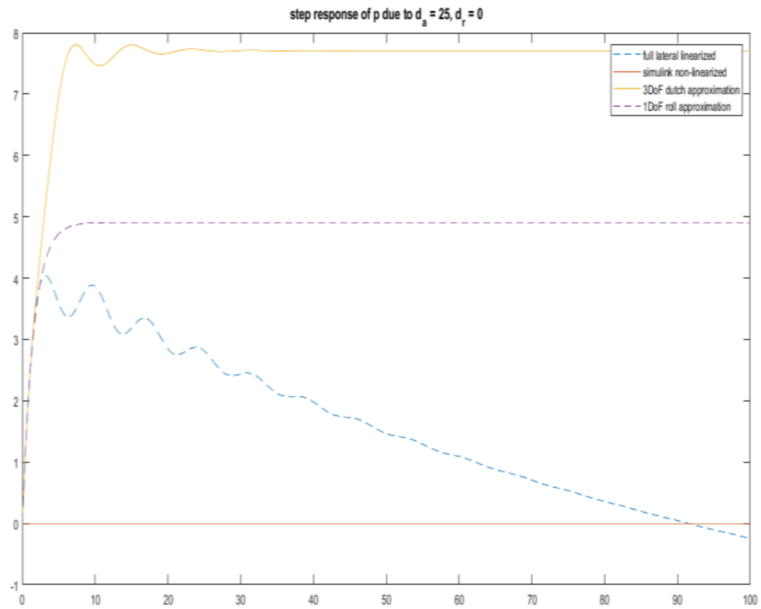

step response of p due to d_a = 10, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
1DoF roll approximation

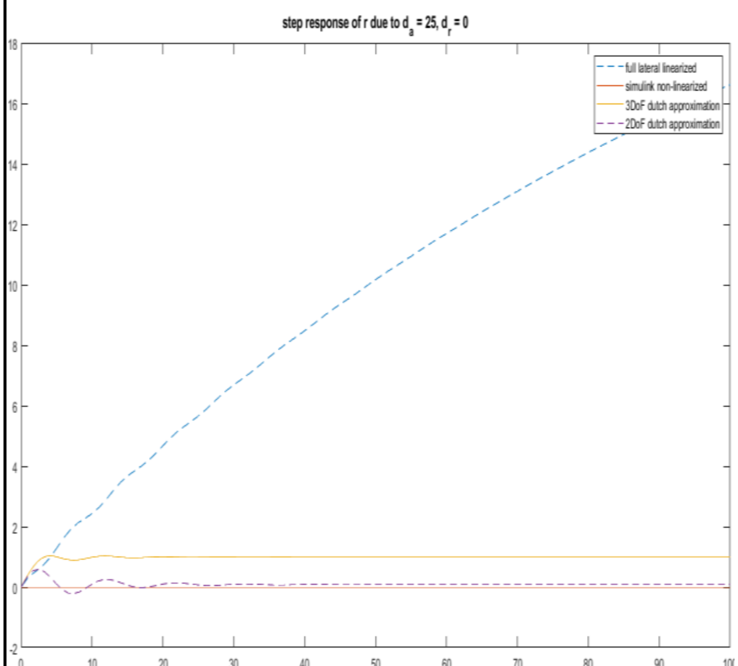

step response of r due to d_a = 10, d_r = 0
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

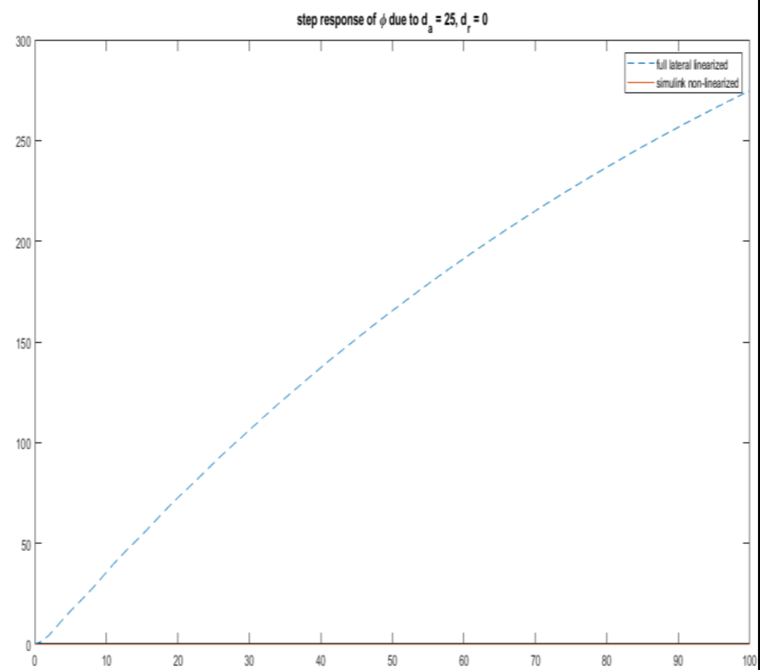

step response of ϕ due to d_a = 10, d_r = 0
full lateral linearized
simulink non-linearized

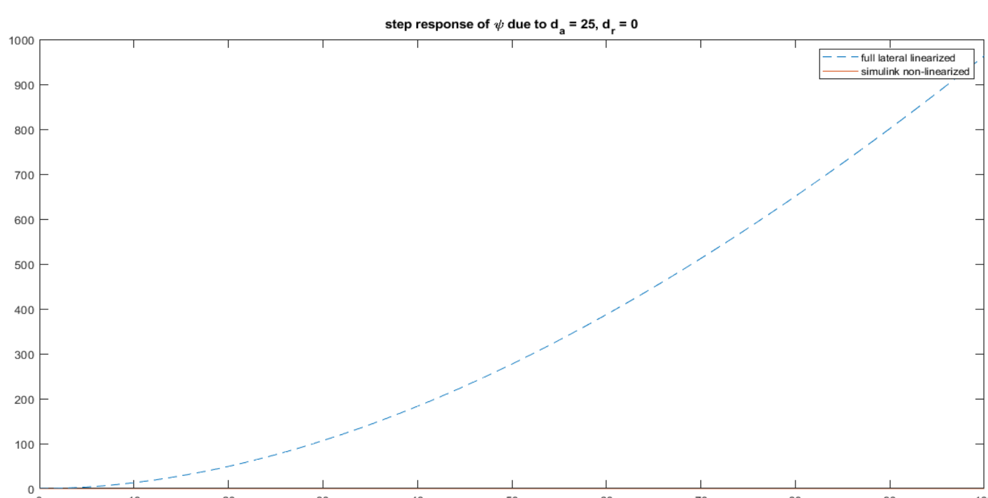

step response of ψ due to d_a = 10, d_r = 0
full lateral linearized
simulink non-linearized

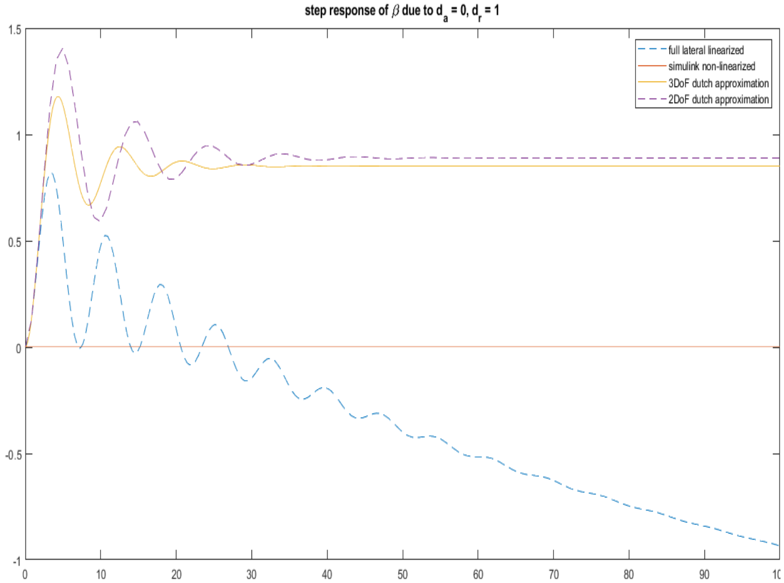

step response of β due to $d_a = 25, d_r = 0$
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation


step response of p due to $d_a = 25, d_r = 0$
full lateral linearized
simulink non-linearized
3DoF dutch approximation
1DoF roll approximation

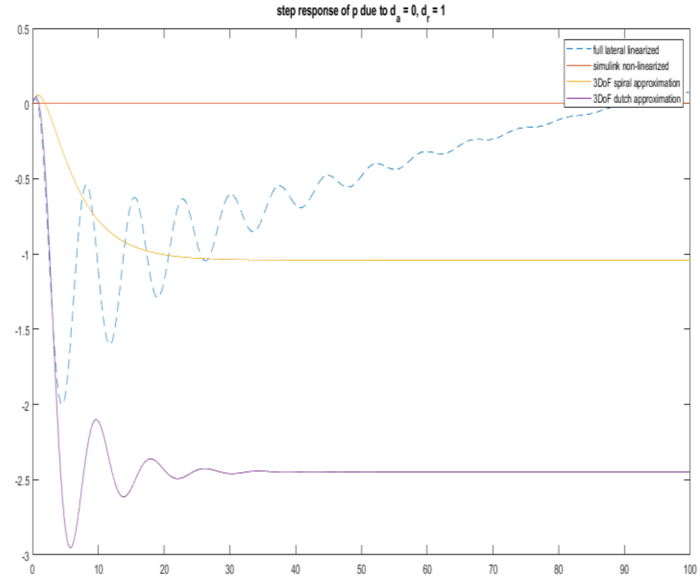

step response of r due to $d_a = 25, d_r = 0$
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation


step response of ϕ due to $d_a = 25, d_r = 0$
full lateral linearized
simulink non-linearized

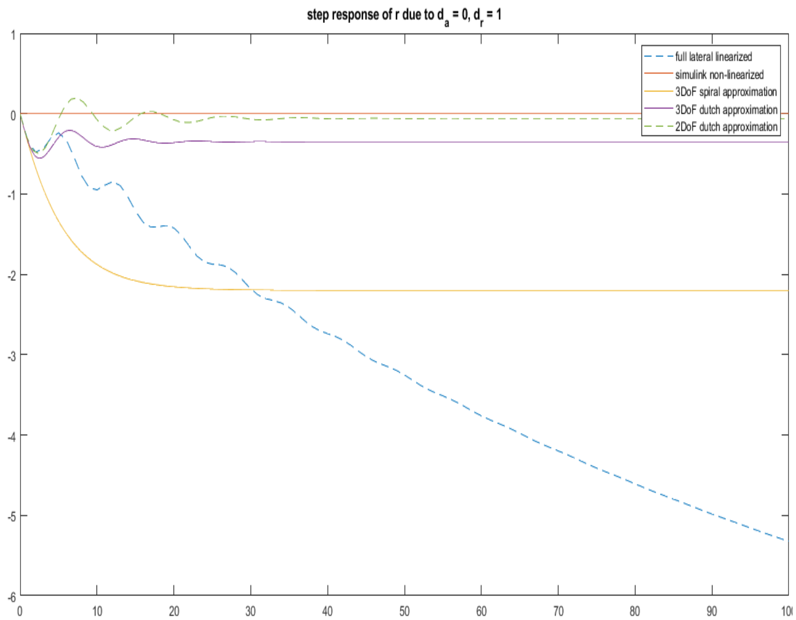

step response of ψ due to $d_a = 25, d_r = 0$
full lateral linearized
simulink non-linearized



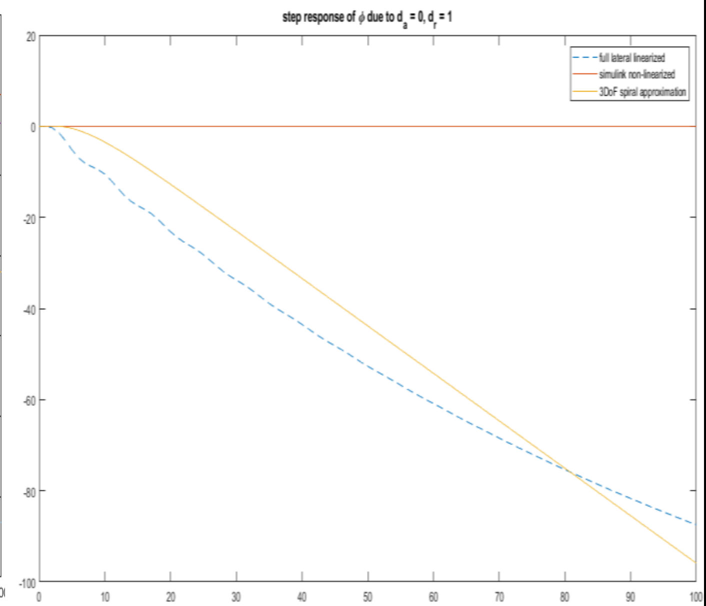
step response of β due to d_a = 0, d_r = 1
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation



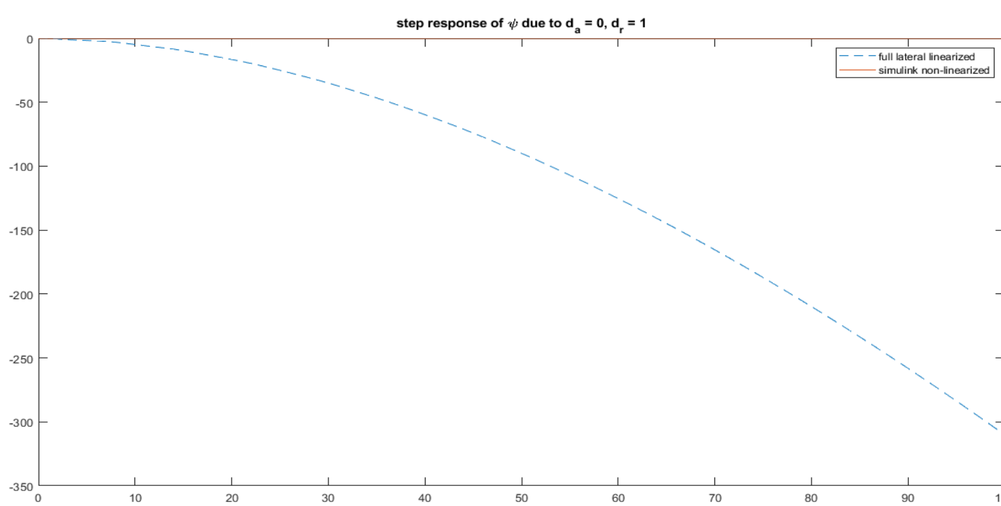
step response of p due to d_a = 0, d_r = 1
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation



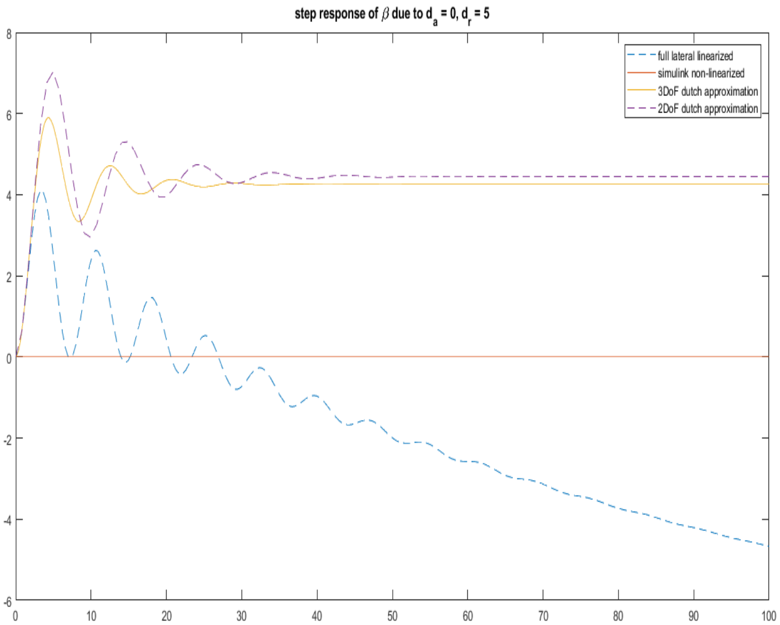
step response of r due to d_a = 0, d_r = 1
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation
2DoF dutch approximation

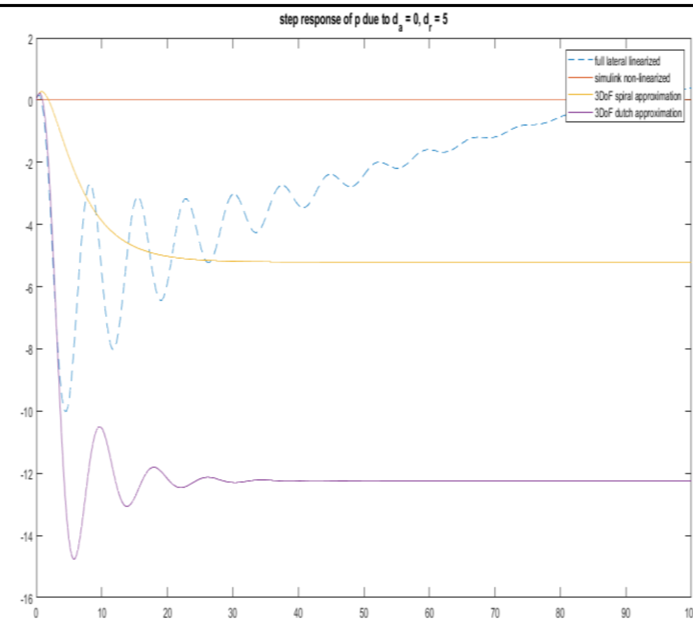


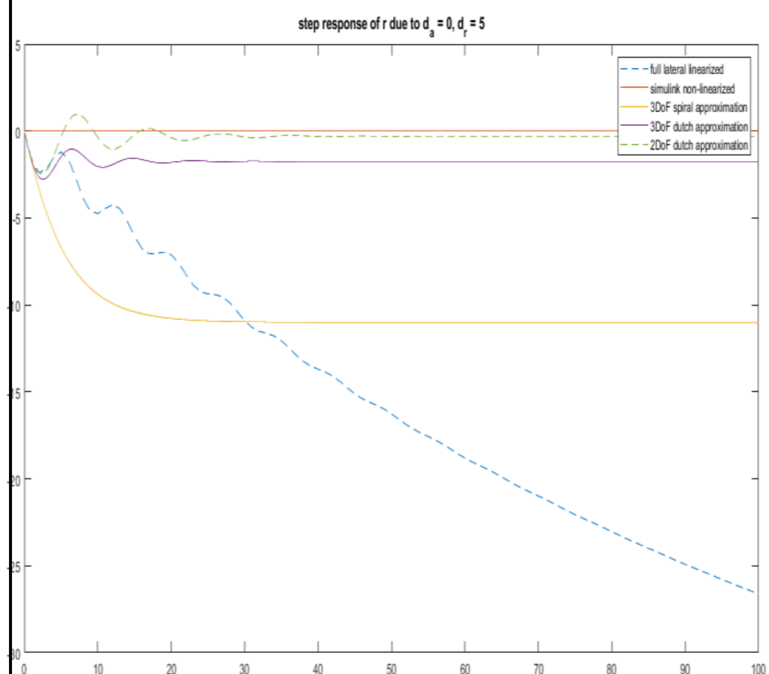
step response of ϕ due to d_a = 0, d_r = 1
full lateral linearized
simulink non-linearized
3DoF spiral approximation

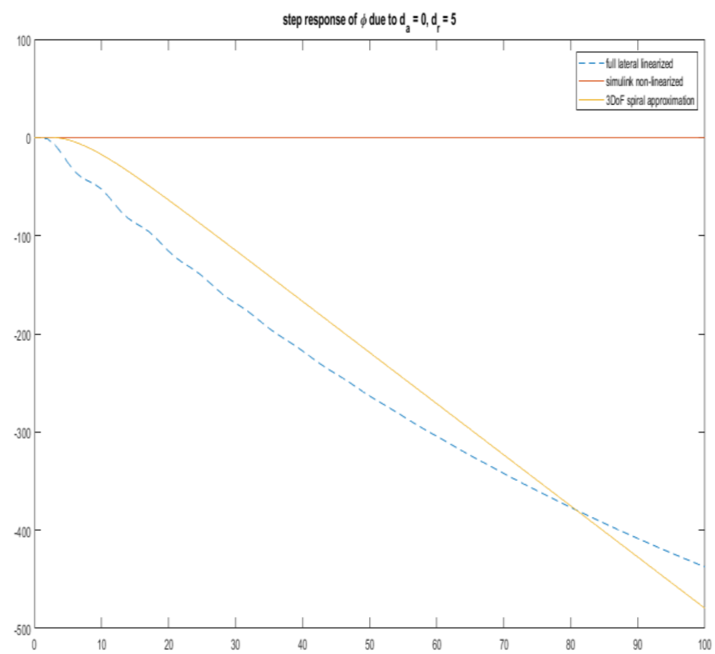


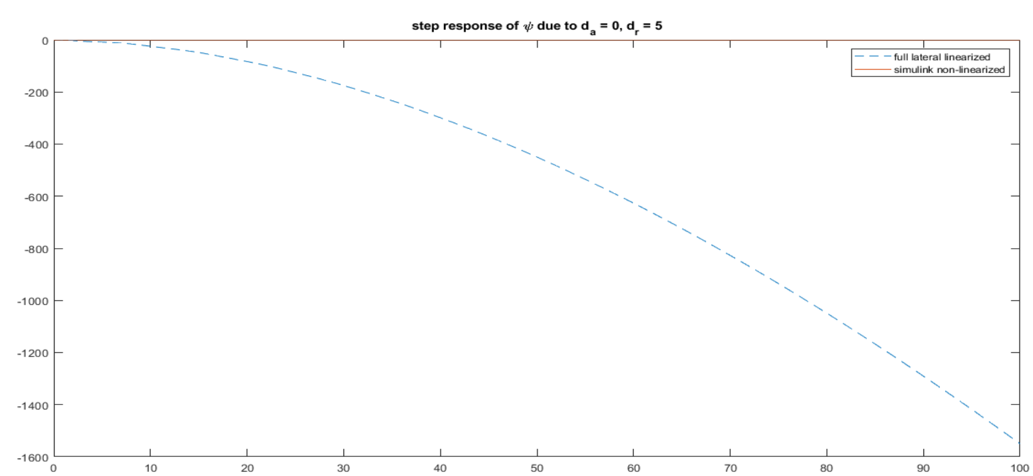
step response of ψ due to d_a = 0, d_r = 1
full lateral linearized
simulink non-linearized

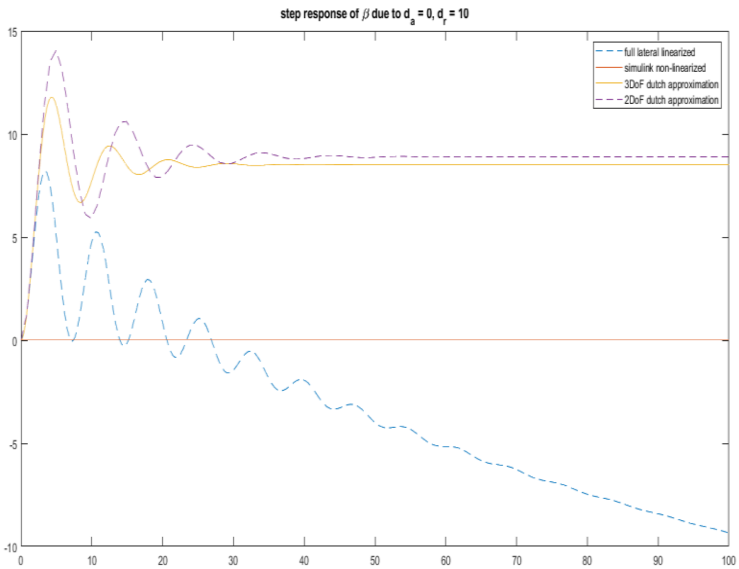

step response of β due to d_a = 0, d_r = 5
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

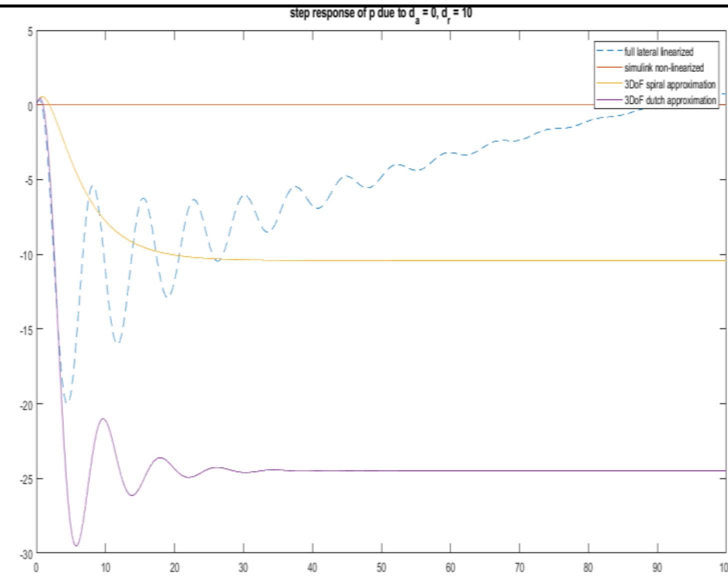

step response of p due to d_a = 0, d_r = 5
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation

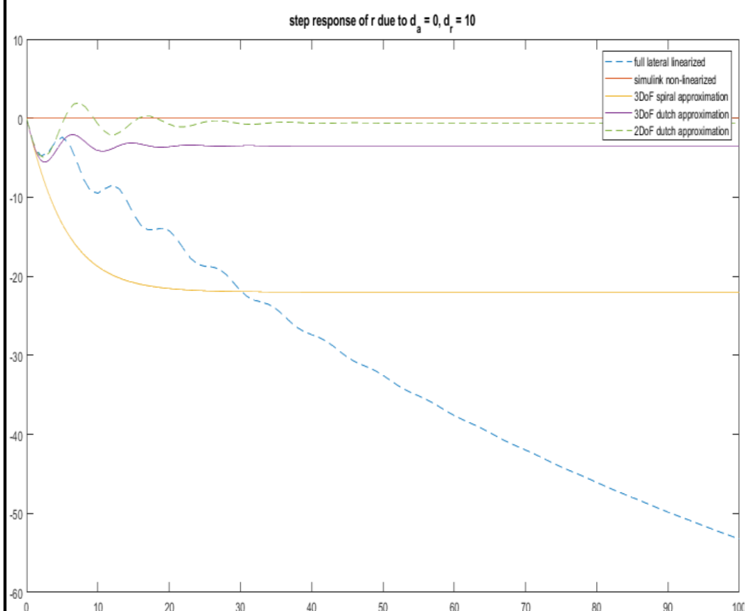

step response of r due to d_a = 0, d_r = 5
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation
2DoF dutch approximation

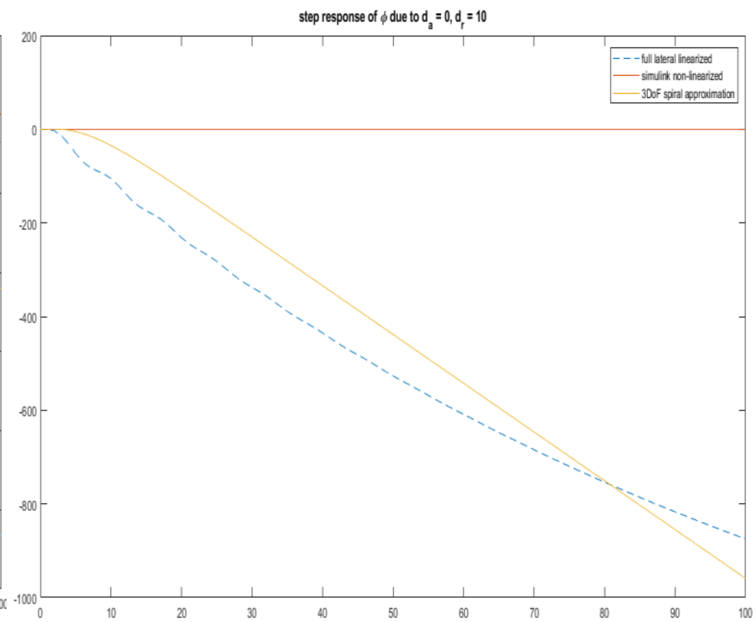

step response of ϕ due to d_a = 0, d_r = 5
full lateral linearized
simulink non-linearized
3DoF spiral approximation

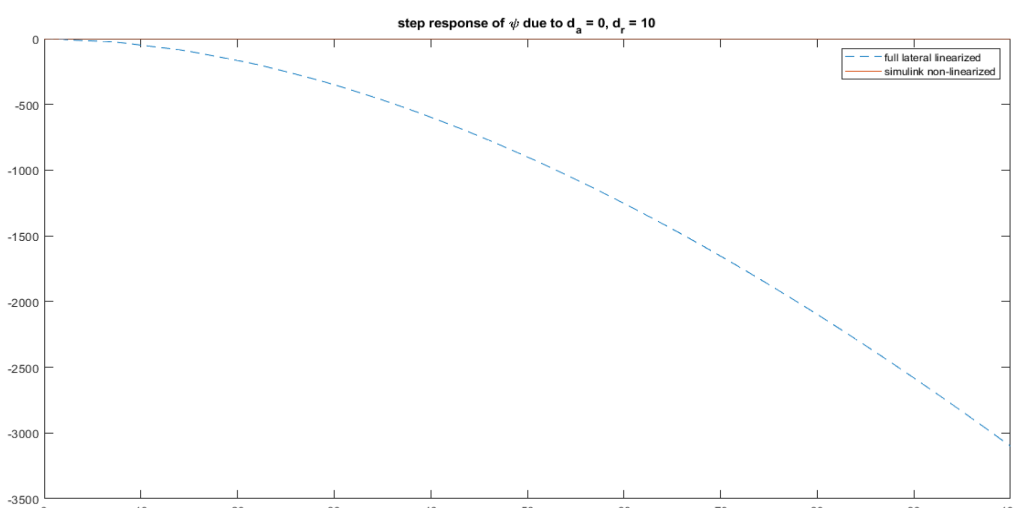

step response of ψ due to d_a = 0, d_r = 5
full lateral linearized
simulink non-linearized

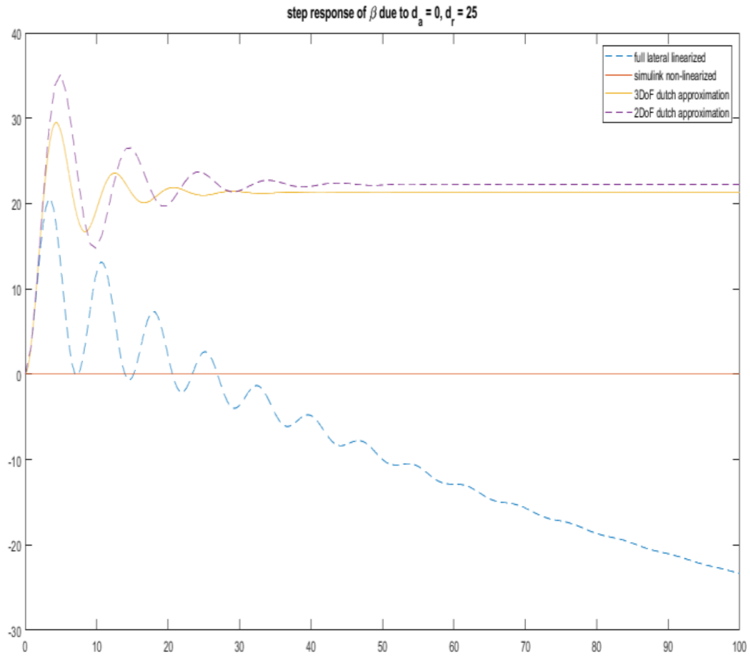

step response of β due to $d_a = 0, d_r = 10$
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

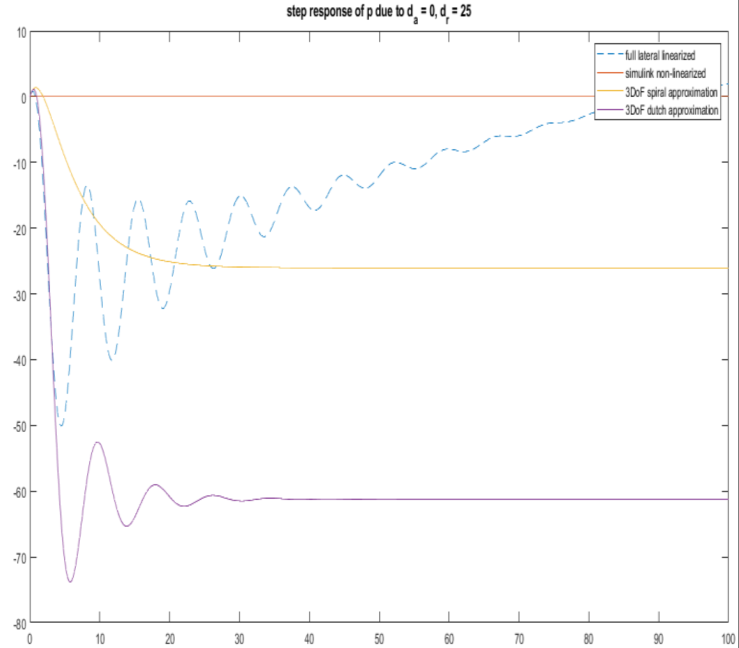

step response of p due to $d_a = 0, d_r = 10$
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation

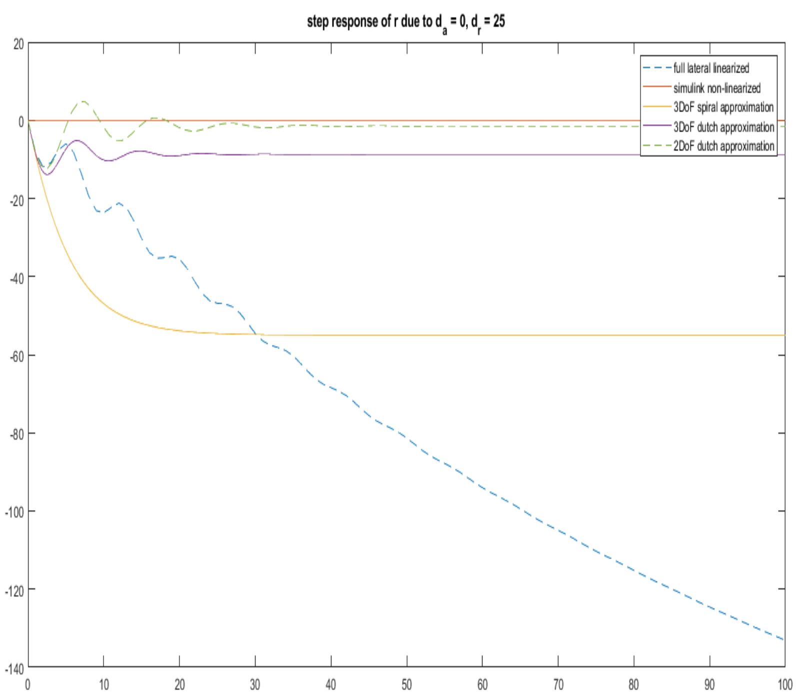

step response of r due to $d_a = 0, d_r = 10$
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation
2DoF dutch approximation

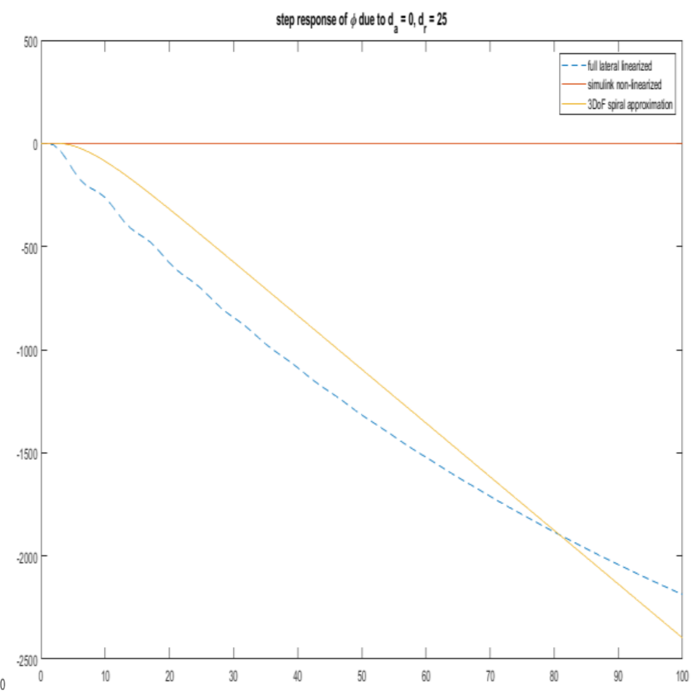

step response of ϕ due to $d_a = 0, d_r = 10$
full lateral linearized
simulink non-linearized
3DoF spiral approximation

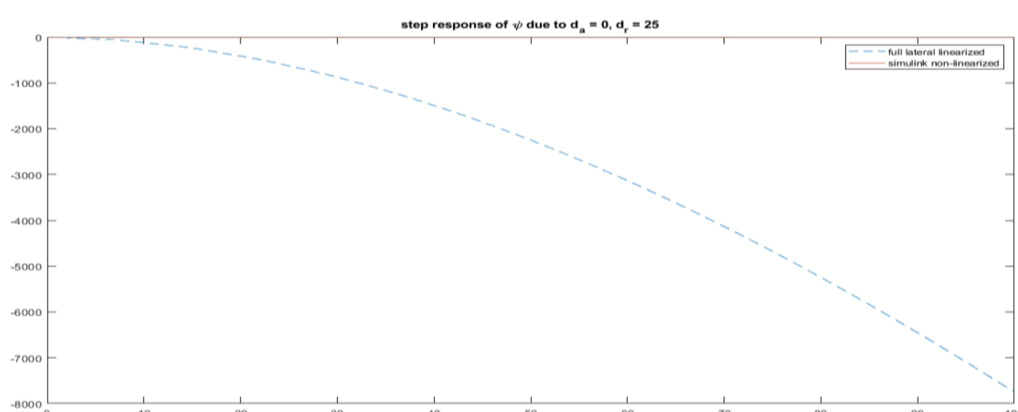

step response of ψ due to $d_a = 0, d_r = 10$
full lateral linearized
simulink non-linearized


step response of β due to d_a = 0, d_r = 25
full lateral linearized
simulink non-linearized
3DoF dutch approximation
2DoF dutch approximation

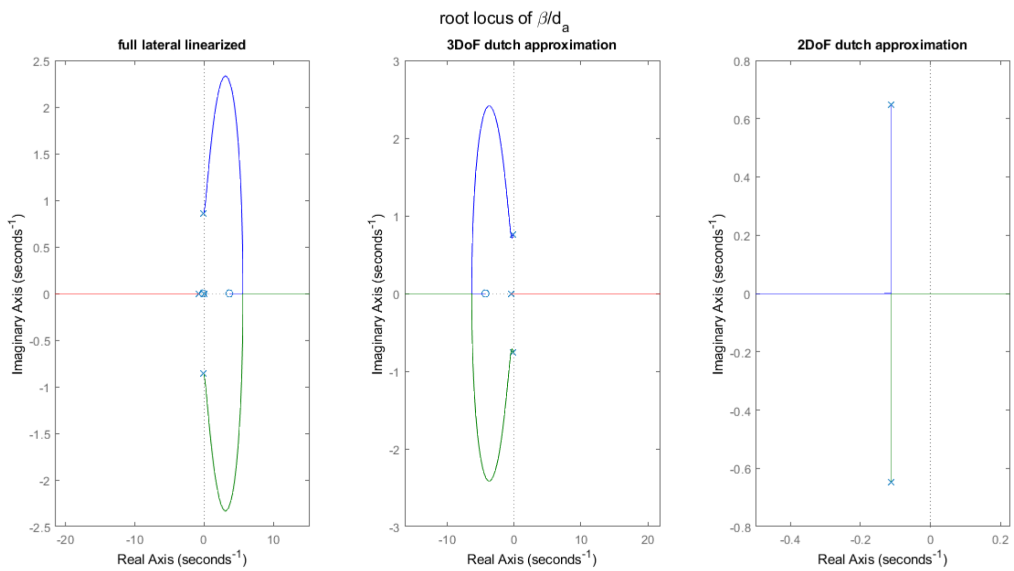

step response of p due to d_a = 0, d_r = 25
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation

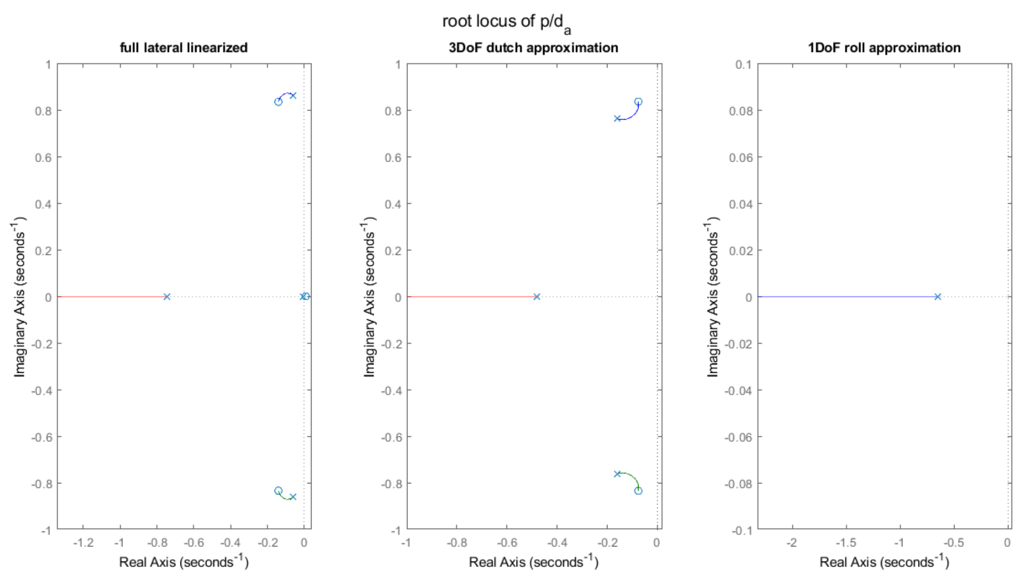

step response of r due to d_a = 0, d_r = 25
full lateral linearized
simulink non-linearized
3DoF spiral approximation
3DoF dutch approximation
2DoF dutch approximation

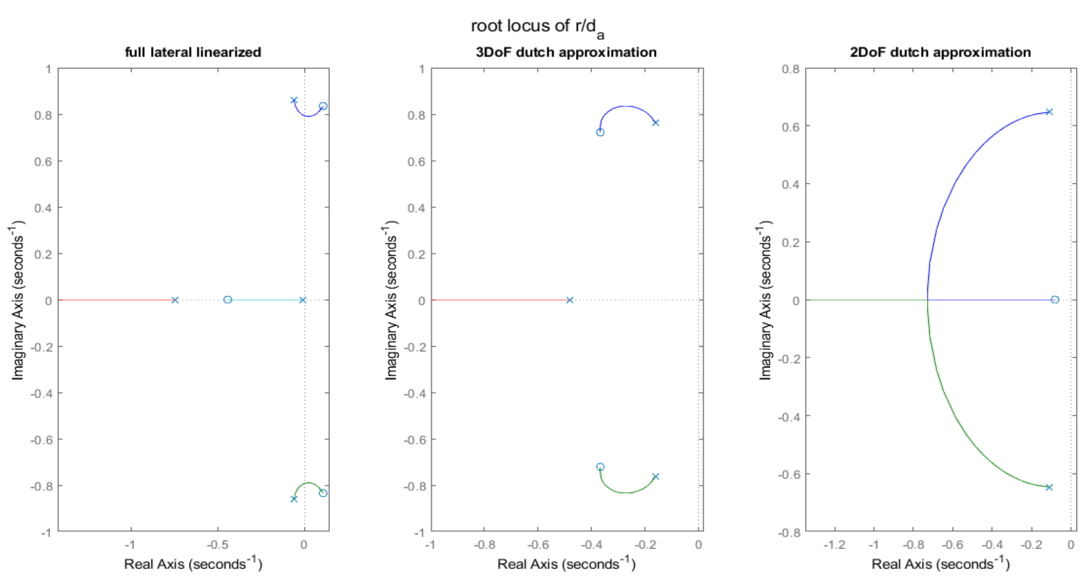

step response of ϕ due to d_a = 0, d_r = 25
full lateral linearized
simulink non-linearized
3DoF spiral approximation

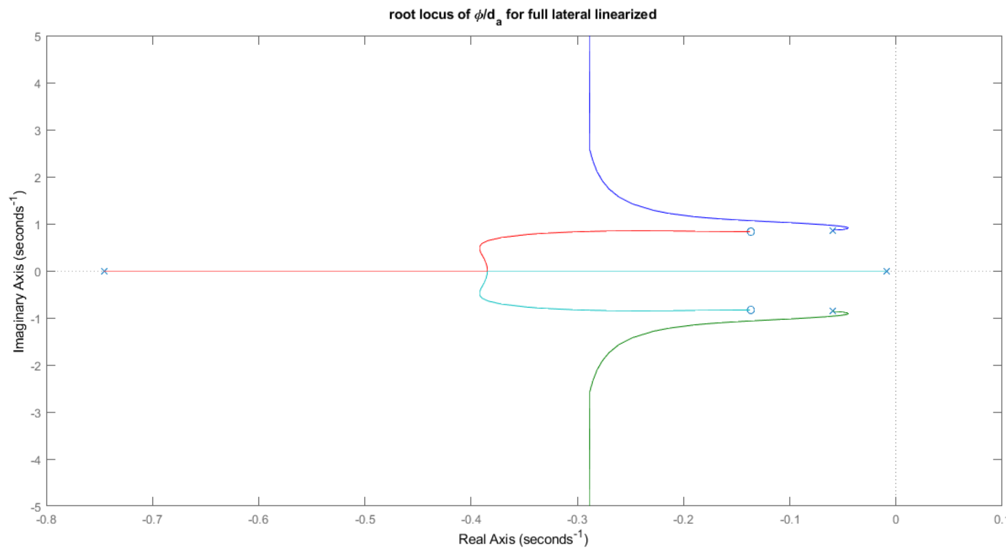

step response of ψ due to d_a = 0, d_r = 25
full lateral linearized
simulink non-linearized

Root locus:

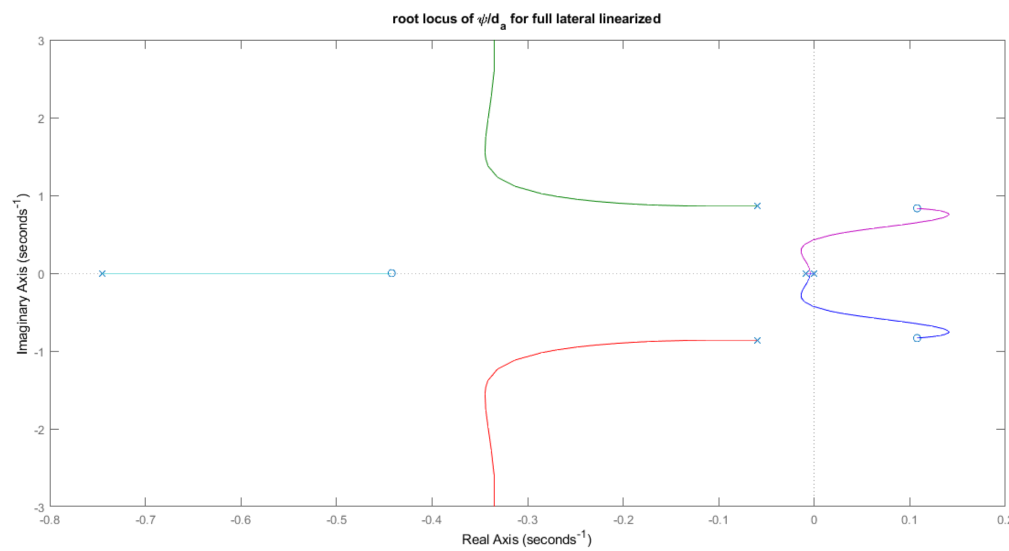

root locus of β/d_a
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)


root locus of p/d_a
full lateral linearized
3DoF dutch approximation
1DoF roll approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)

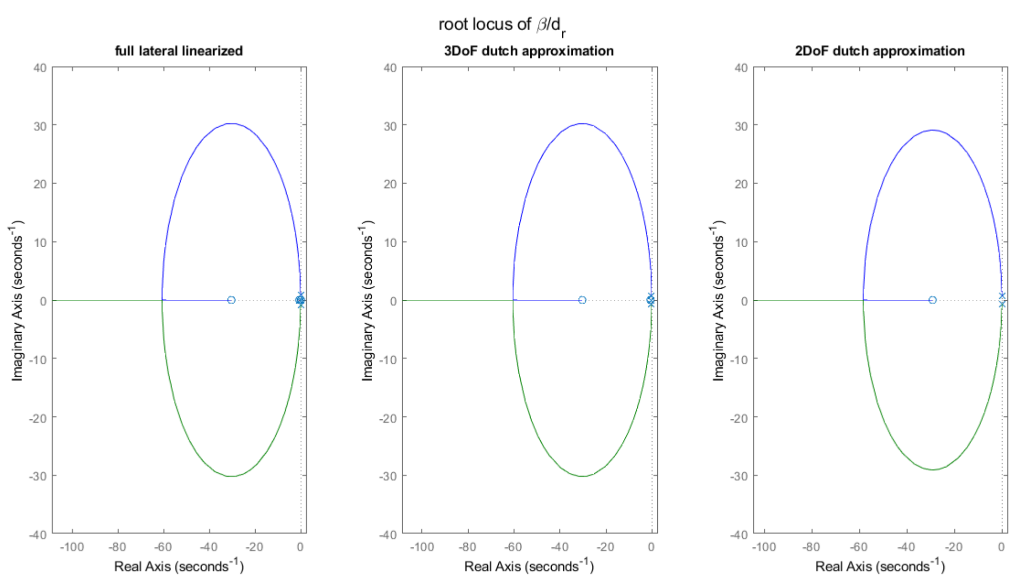

root locus of r/d_a
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)



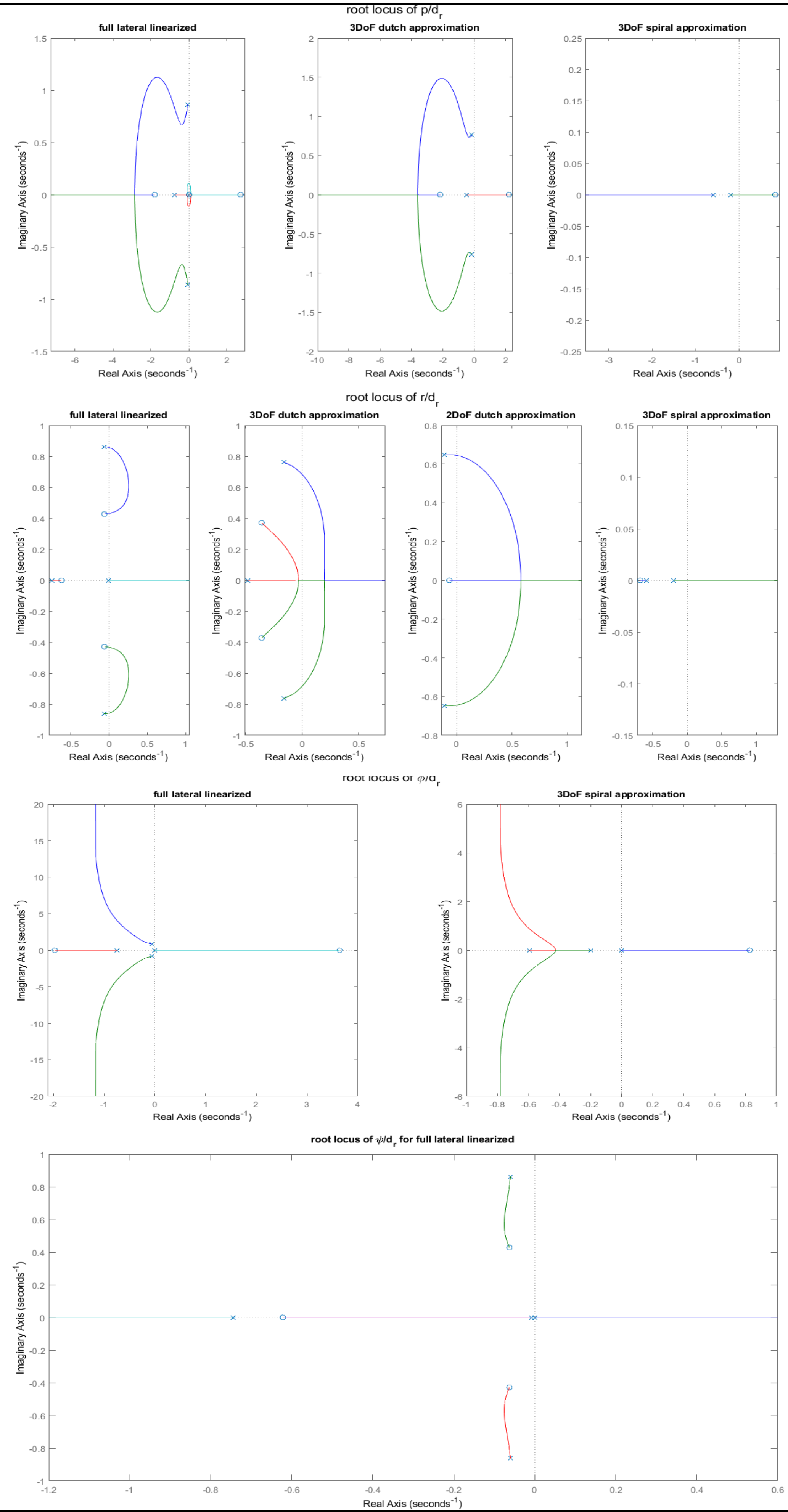
root locus of ϕ/d_a for full lateral linearized
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})



root locus of ψ/d_a for full lateral linearized
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})

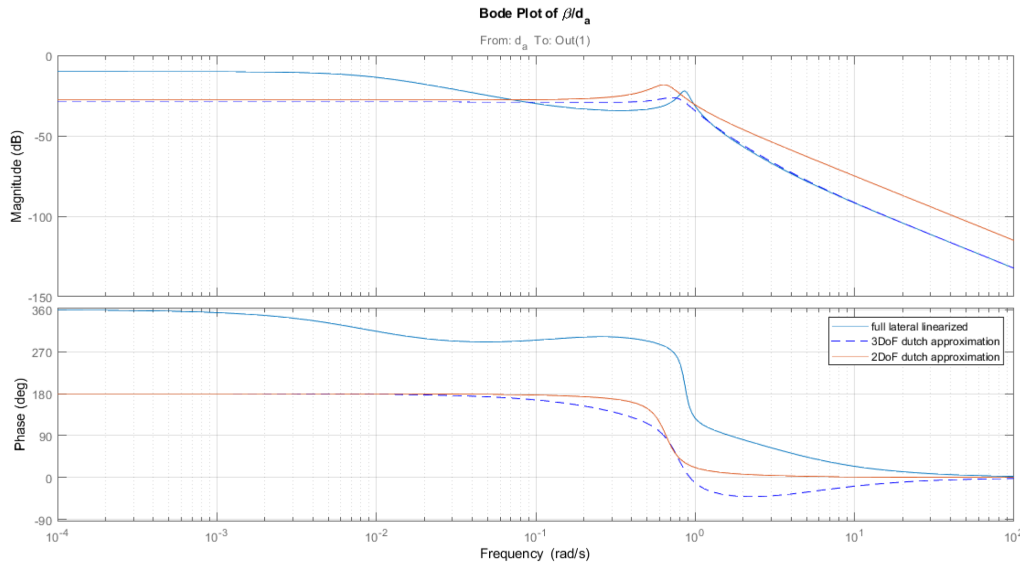


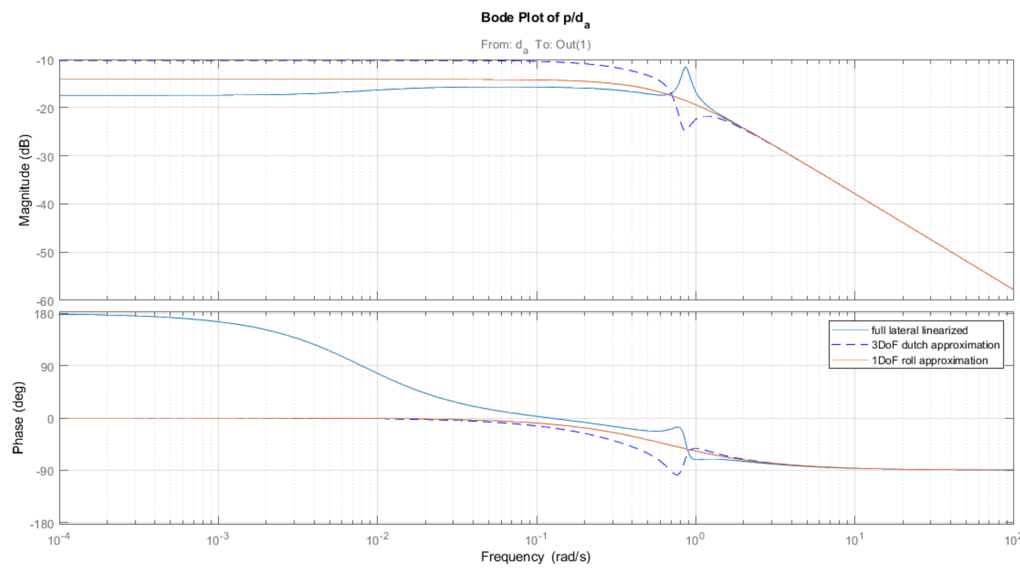
root locus of β/d_r
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})

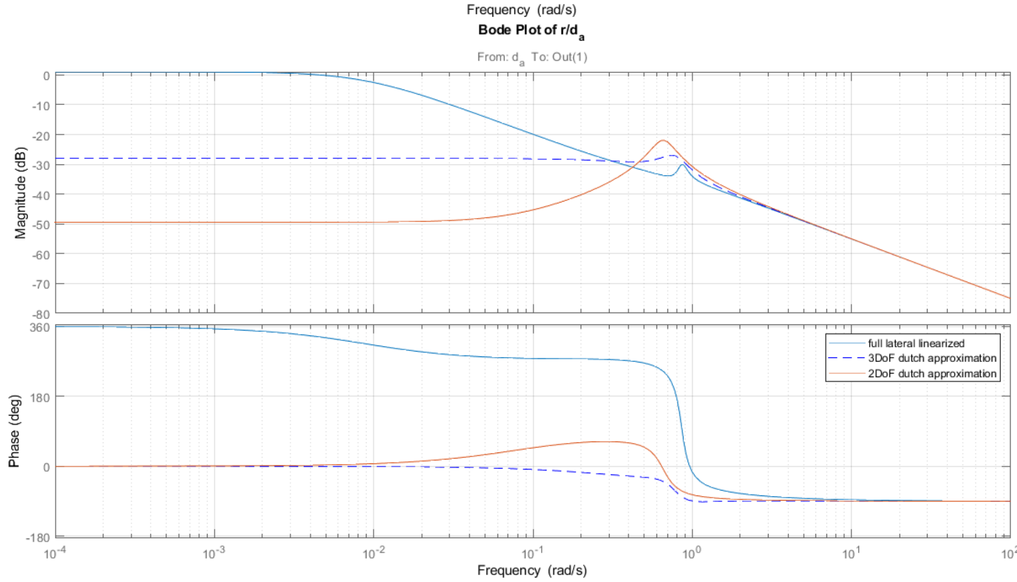


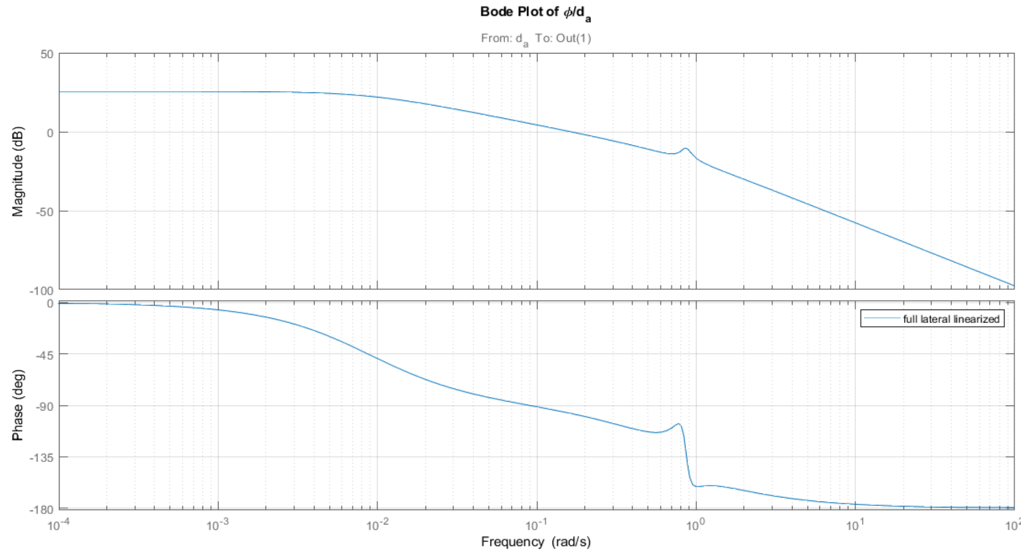
root locus of p/d_r
full lateral linearized
3DoF dutch approximation
3DoF spiral approximation
root locus of r/d_r
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
3DoF spiral approximation
root locus of φ/d_r
full lateral linearized
3DoF spiral approximation
root locus of ψ/d_r for full lateral linearized
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)

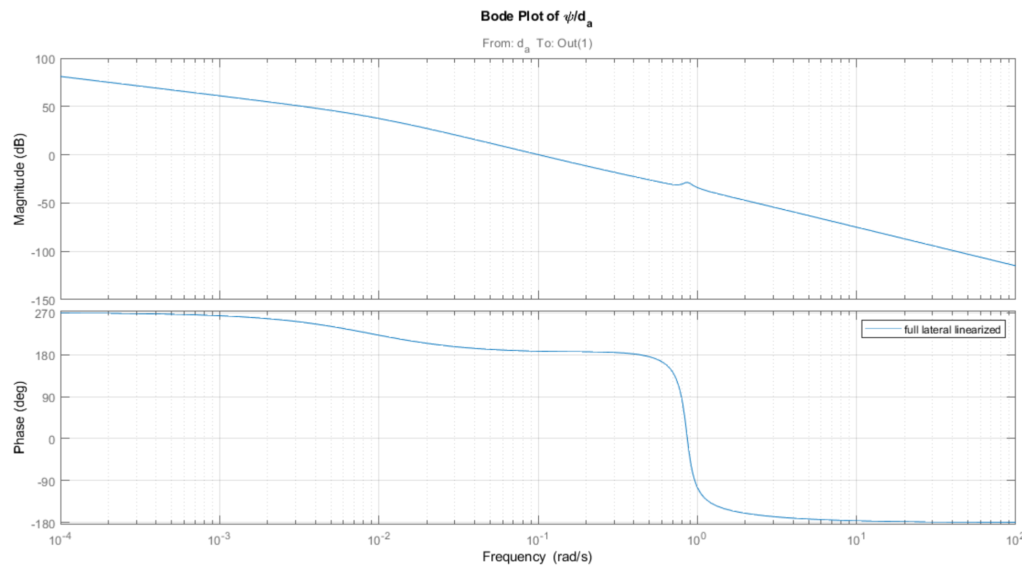
Bode Blot:

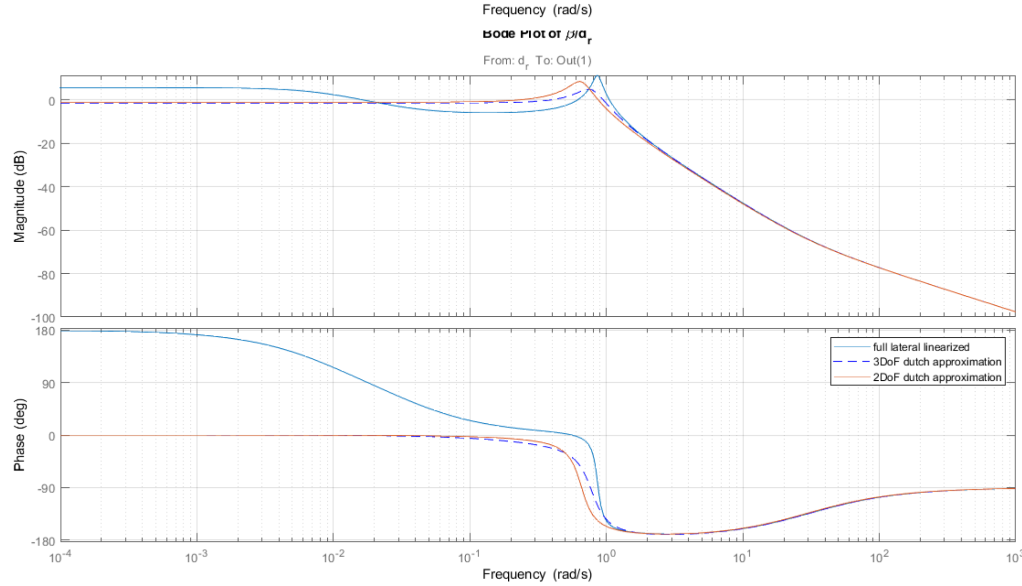

Bode Plot of β/d_a
From: d_a To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Frequency (rad/s)

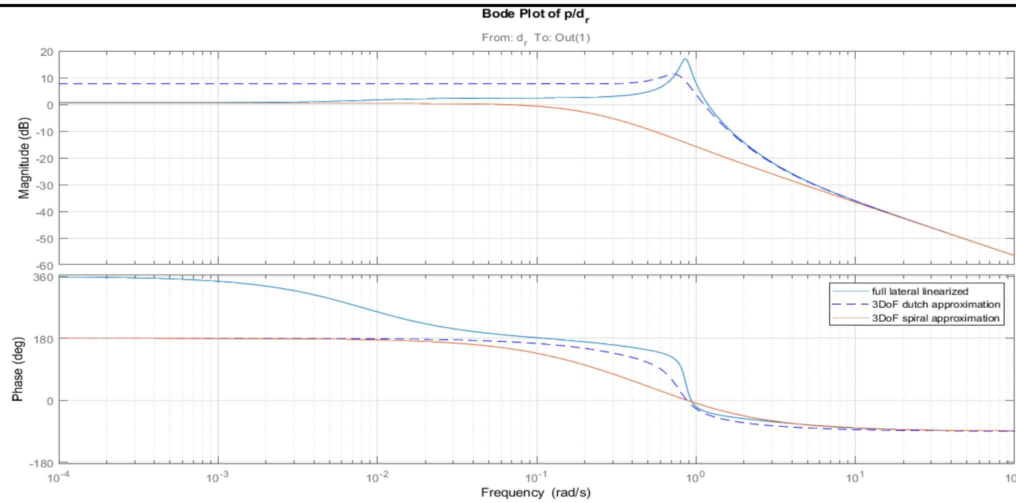

Bode Plot of p/d_a
From: d_a To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
3DoF dutch approximation
1DoF roll approximation
Frequency (rad/s)

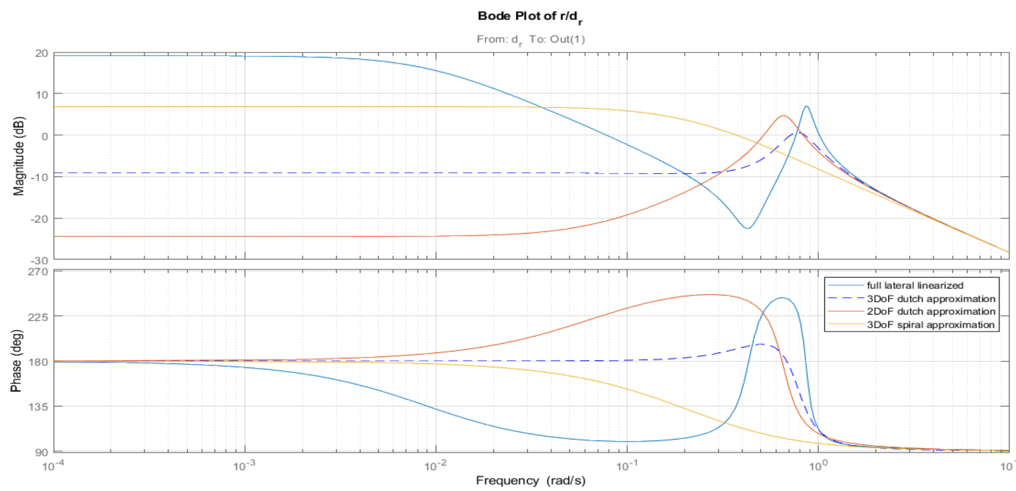

Bode Plot of r/d_a
From: d_a To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Frequency (rad/s)

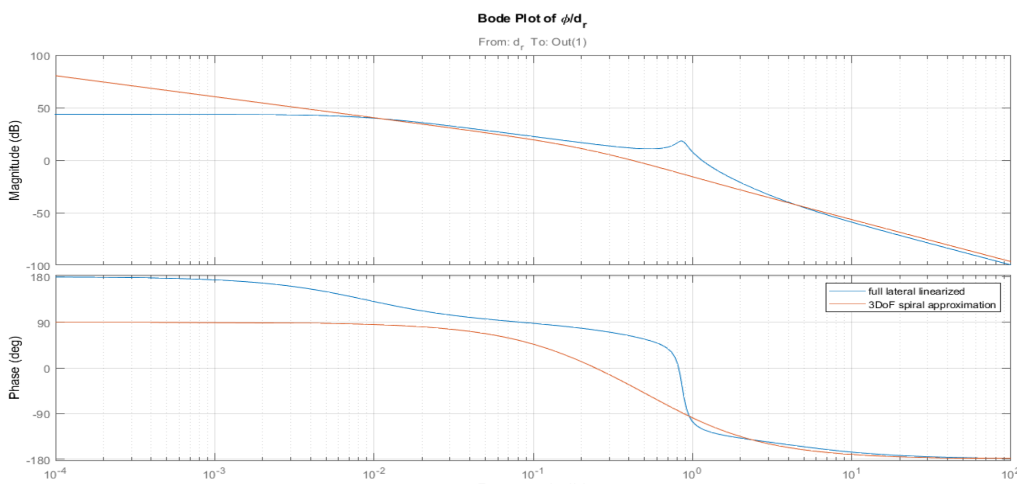

Bode Plot of ϕ/d_a
From: d_a To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
Frequency (rad/s)

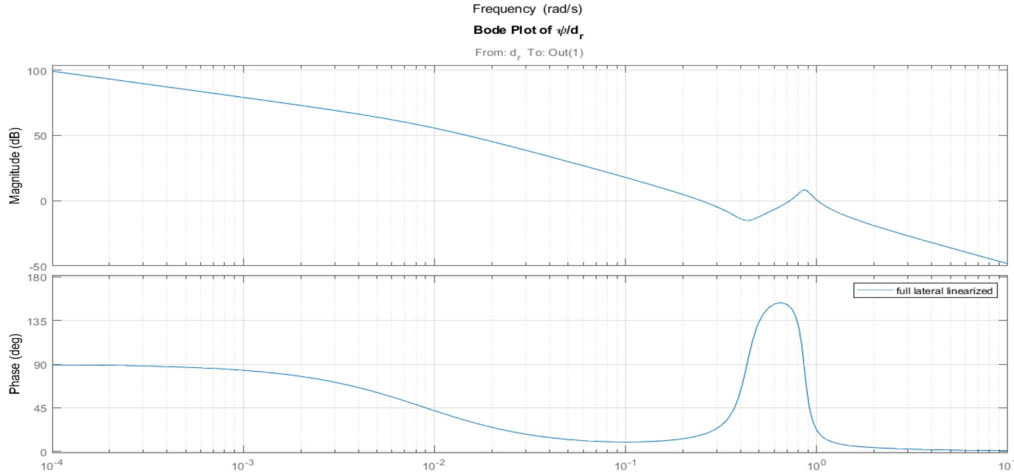

Bode Plot of ψ/d_a
From: d_a To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
Frequency (rad/s)


Bode Plot of β/d_r
From: d_r To: Out(1)
Magnitude (dB)
Phase (deg)
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
Frequency (rad/s)


Bode Plot of p/d_r
From: d_r To: Out(1)
Magnitude (dB)
Phase (deg)
Frequency (rad/s)
full lateral linearized
3DoF dutch approximation
3DoF spiral approximation


Bode Plot of r/d_r
From: d_r To: Out(1)
Magnitude (dB)
Phase (deg)
Frequency (rad/s)
full lateral linearized
3DoF dutch approximation
2DoF dutch approximation
3DoF spiral approximation


Bode Plot of φ/d_r
From: d_r To: Out(1)
Magnitude (dB)
Phase (deg)
Frequency (rad/s)
full lateral linearized
3DoF spiral approximation


Bode Plot of ψ/d_r
From: d_r To: Out(1)
Magnitude (dB)
Phase (deg)
Frequency (rad/s)
full lateral linearized

Task5.part1 [long.Autopilot]

Overview

The objective in this part of our project is to design "The longitudinal Autopilot" for a conventional fixed wing airplane. The requirement for the longitudinal autopilot is to control the motion of the airplane in the longitudinal plane, shortly "it controls the elevator & thrust to achieve the desired command of Altitude or Climb angle."

We will use the linearized state space model of the longitudinal dynamics (4x4) to represent the motion of the airplane in the longitudinal plane, which is a MIMO (Multi input Multi Output system), and to design our controllers we will use our previous studies about the LTI (linear time invariant) SISO (Single Input Single Output system) to design our controllers like (linear PID and compensators). This method is called "Successive loop closure".

The main problem is "How to use the tools of SISO systems to deal with a MIMO system", this is the main topic of any reference about autopilot design, i.e. the control loops architecture.

So, first we Prepare all the transfer functions of the full Longitudinal dynamics model (4x4) available in your MATLAB workspace [we already have them from previous tasks].

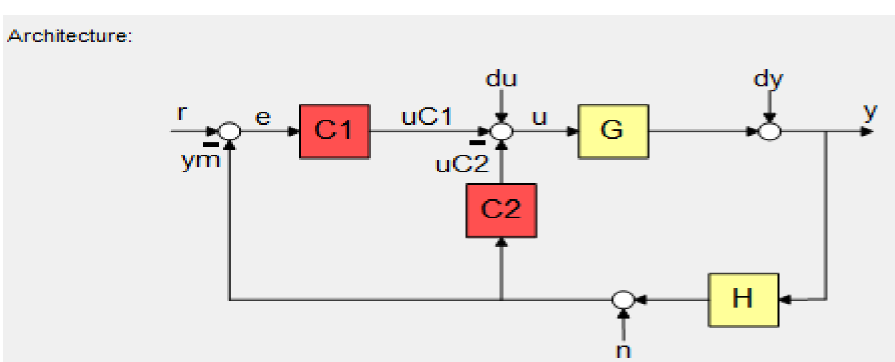
$$\frac{u}{\delta_e}, \frac{w}{\delta_e}, \frac{q}{\delta_e}, \frac{\theta}{\delta_e}, \frac{\alpha}{\delta_e}$$

$$\frac{u}{\delta_T}, \frac{w}{\delta_T}, \frac{q}{\delta_T}, \frac{\theta}{\delta_T}, \frac{\alpha}{\delta_T}$$

And create the transfer function as servo : tf(10,[1 10]) and then we will choose loop architecture and design a pitch controller with pitch rate (q) feedback.

We will try to get best results and by using tuning.i.e adding integrator

Designs
First design



Pitch controller design steps:

Step 1:

We will prepare all the transfer functions of the full longitudinal dynamics model (4x4)

$\frac{U}{\delta e}$:

$$\frac{2.02\ s^3\ +\ 67.25\ s^2\ +\ 19.96\ s\ +\ 14.7}{s^4\ +\ 0.9279\ s^3\ +\ 1.085\ s^2\ +\ 0.01037\ s\ +\ 0.007276}$$

$\frac{w}{\delta e}$:

$$\frac{-17.17\ s^3 - 569.9\ s^2 - 1.862\ s - 2.529}{s^4 + 0.9279\ s^3 + 1.085\ s^2 + 0.01037\ s + 0.007276}$$

$\frac{q}{\delta e}$:

$$\frac{-1.088\ s^3 - 0.4525\ s^2 - 0.006868\ s}{s^4 + 0.9279\ s^3 + 1.085\ s^2 + 0.01037\ s + 0.007276}$$

$\frac{\theta}{\delta e}$:

$$\frac{-1.088\ s^2\ -\ 0.4525\ s\ -\ 0.006868}{s^4\ +\ 0.9279\ s^3\ +\ 1.085\ s^2\ +\ 0.01037\ s\ +\ 0.007276}$$

$\frac{u}{\delta T}$:

$$\frac{5.05e-05\ s^3\ +\ 2.802e-05\ s^2\ +\ 4.771e-05\ s\ -\ 4.789e-06}{s^4\ +\ 0.9279\ s^3\ +\ 1.085\ s^2\ +\ 0.01037\ s\ +\ 0.007276}$$

$\frac{w}{\delta T}$:

$$\frac{2.235e-06\ s^3\ +\ 0.0001514\ s^2\ +\ 5.428e-06\ s\ +\ 5.968e-07}{s^4\ +\ 0.9279\ s^3\ +\ 1.085\ s^2\ +\ 0.01037\ s\ +\ 0.007276}$$

$\frac{q}{\delta T}$:

$$\frac{3.023e{-}07\ s^3 + 1.503e{-}07\ s^2 + 1.333e{-}08\ s}{s^4 + 0.9279\ s^3 + 1.085\ s^2 + 0.01037\ s + 0.007276}$$

$\frac{\theta}{\delta T}$:

$$\frac{3.023e{-}07\ s^2 + 1.503e{-}07\ s + 1.333e{-}08}{s^4 + 0.9279\ s^3 + 1.085\ s^2 + 0.01037\ s + 0.007276}$$

step 2:

we will use also the following transfer functions in our design

Servo : $\frac{10}{s+10}$ Differentiator : s Integrator : $\frac{1}{s}$

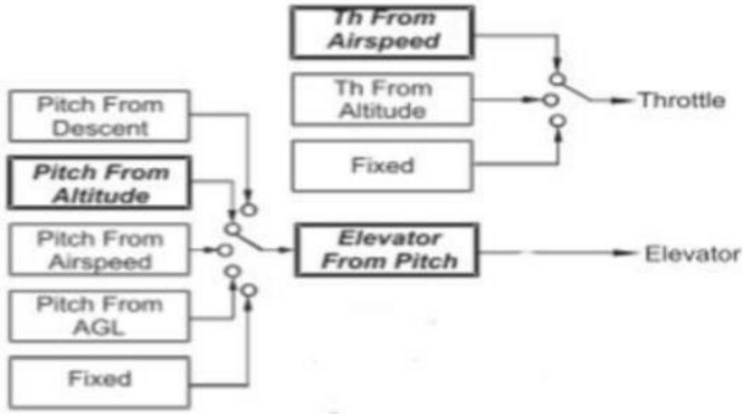
Note :

- We use differentiator to introduce damping into the system, helping to dampen out oscillations and reduce overshoot in response to sudden changes or disturbances.
- By looking at the rate of change of the error signal (the difference between desired and actual pitch), a differentiator can provide a predictive element to the control action, anticipating future changes and enabling faster response.
- We use integrator to eliminate steady-state errors by continuously summing the error signal over time. This ensures that even small, persistent errors are eventually corrected. And also to to improve the stability of the system, particularly in the presence of external disturbances or system uncertainties. They can effectively counteract biases and maintain the system close to its desired state over the long term.

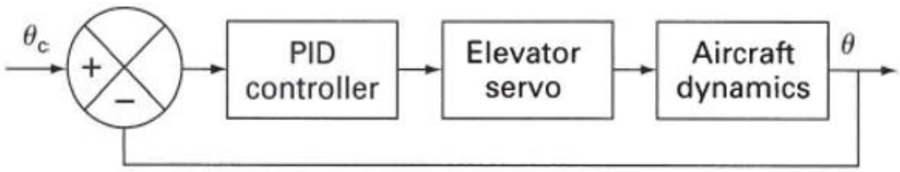
Finally we decide to use PID controller in our design

Step 3:

We will use successive loop closure in our design



The first loop which we will design it is the pitch control loop



PID	$k_p + \frac{k_i}{s} + k_d s$
Elevator servo	$-\frac{10}{s + 10}$

Step 4:

From the transfer function of q_de we note the the numerator has negative sign

$\frac{q}{\delta e}$:

$$\frac{-1.088\ s^{\wedge}3 - 0.4525\ s^{\wedge}2 - 0.006868\ s}{s^{\wedge}4 + 0.9279\ s^{\wedge}3 + 1.085\ s^{\wedge}2 + 0.01037\ s + 0.007276}$$

so that when the aircraft pitches up (nose rotates upward), the elevator control surface needs to move downward to counteract this motion and bring the aircraft back to the desired pitch angle. Similarly, when the aircraft pitches down (nose rotates downward), the elevator needs to move upward to oppose this motion.

It's the reason to add negative sign to the transfer function of servo

The negative sign ensures that the servo mechanism moves the control surface in the correct direction to achieve the desired pitch adjustment.

Step 5:

We must take care abiut zeda(dambing ratio) bocause it is important for pilot too. So we must design in zeda limits , see our plane in this limit not and if not and if not we must design to reach this limit.

Phugoid mode	
Level 1	ζ>0.04
Level 2	ζ>0
Level 3	T>55s

Short period mode				
	Category A and C		Category B	
	ζmin	ζmax	ζmin	ζmax
Level 1	0.35	1.30	0.3	2.0
Level 2	0.25	2.00	0.2	2.0
Level 3	0.15	-	0.15	-

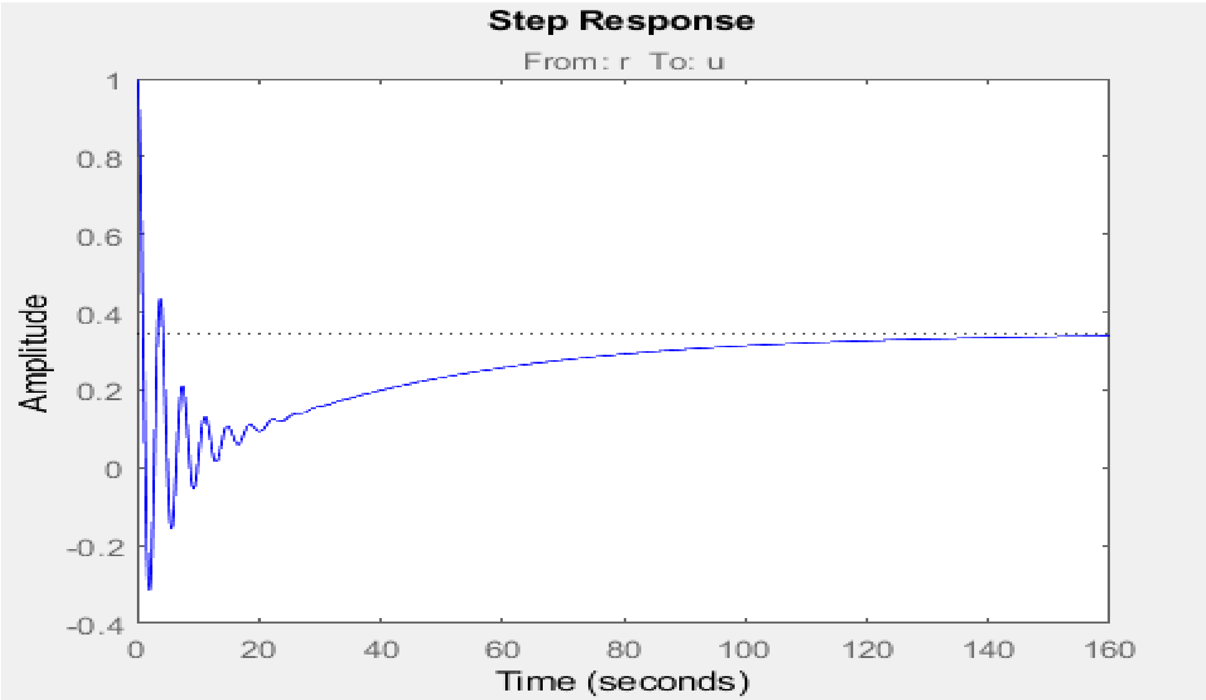
In our plane short period is ok , but the other is not so we must take this in our calculation and designs.

There are steps to get best results like we will do tuining to get best result as we can .

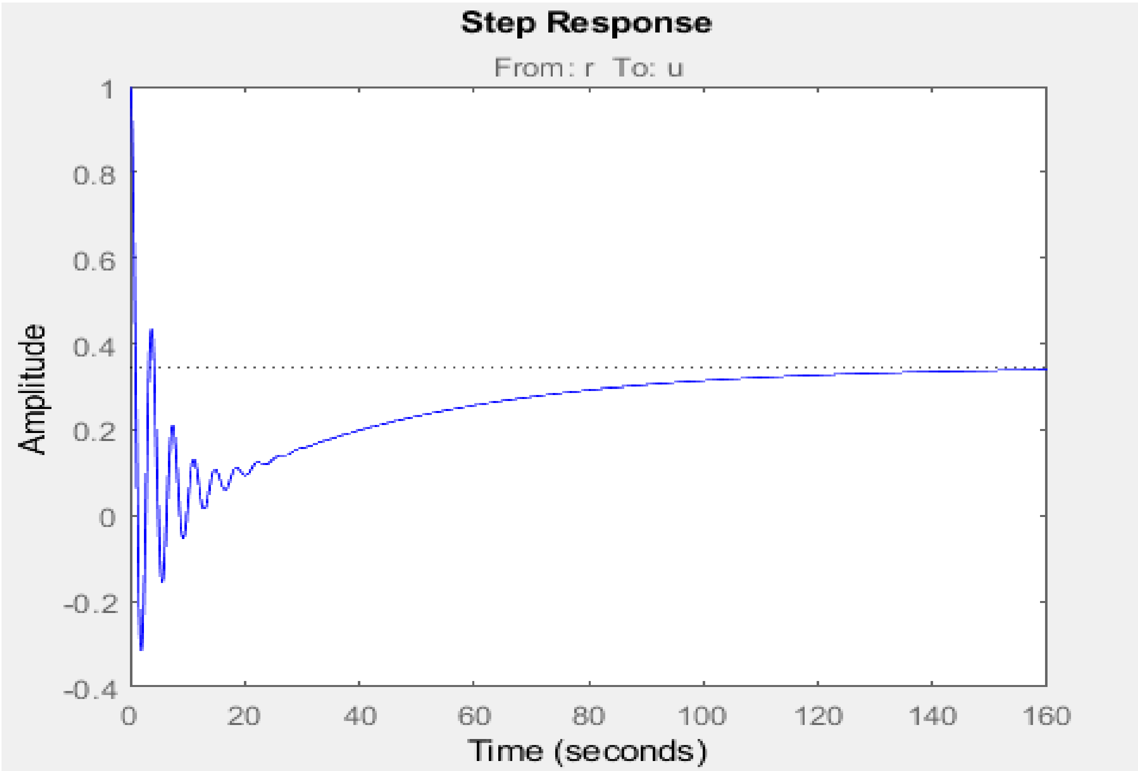
Designs:

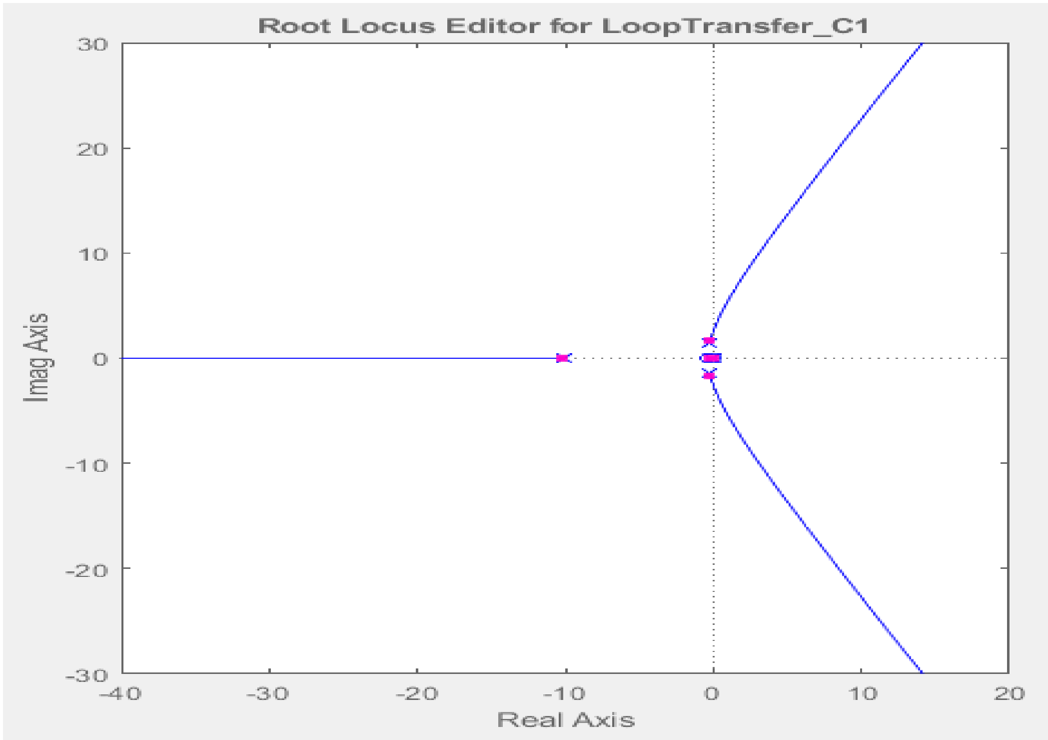
Design1

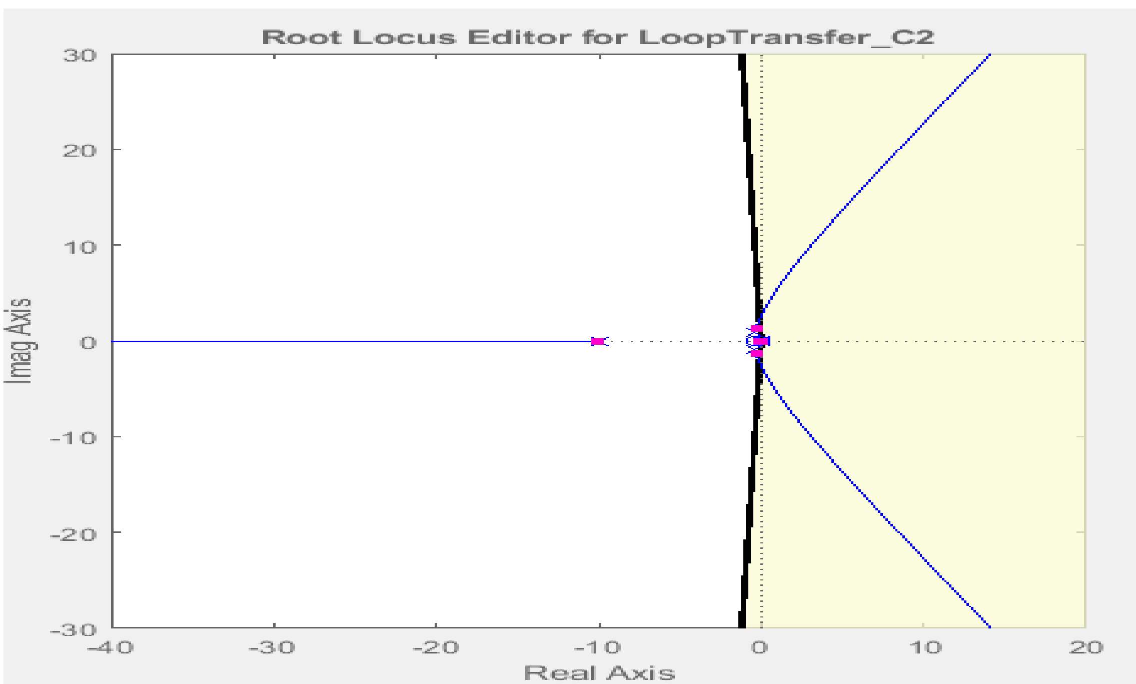
control action



Step response



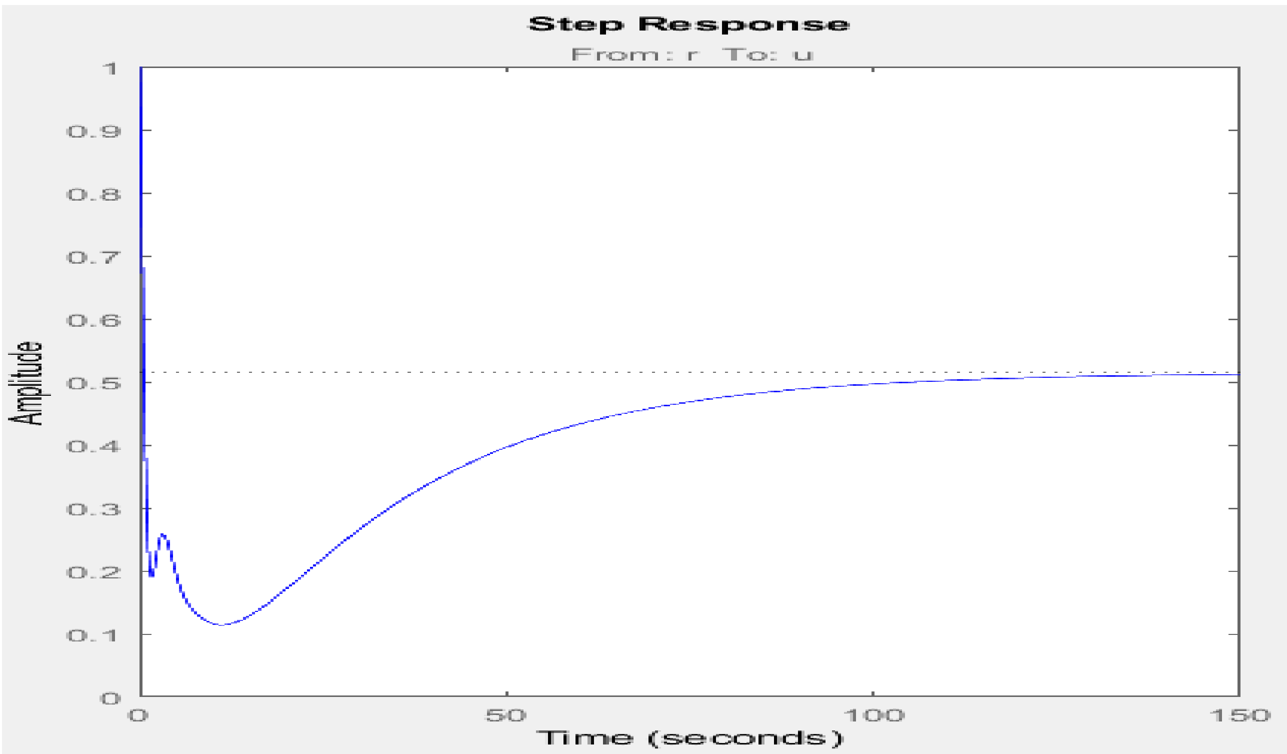


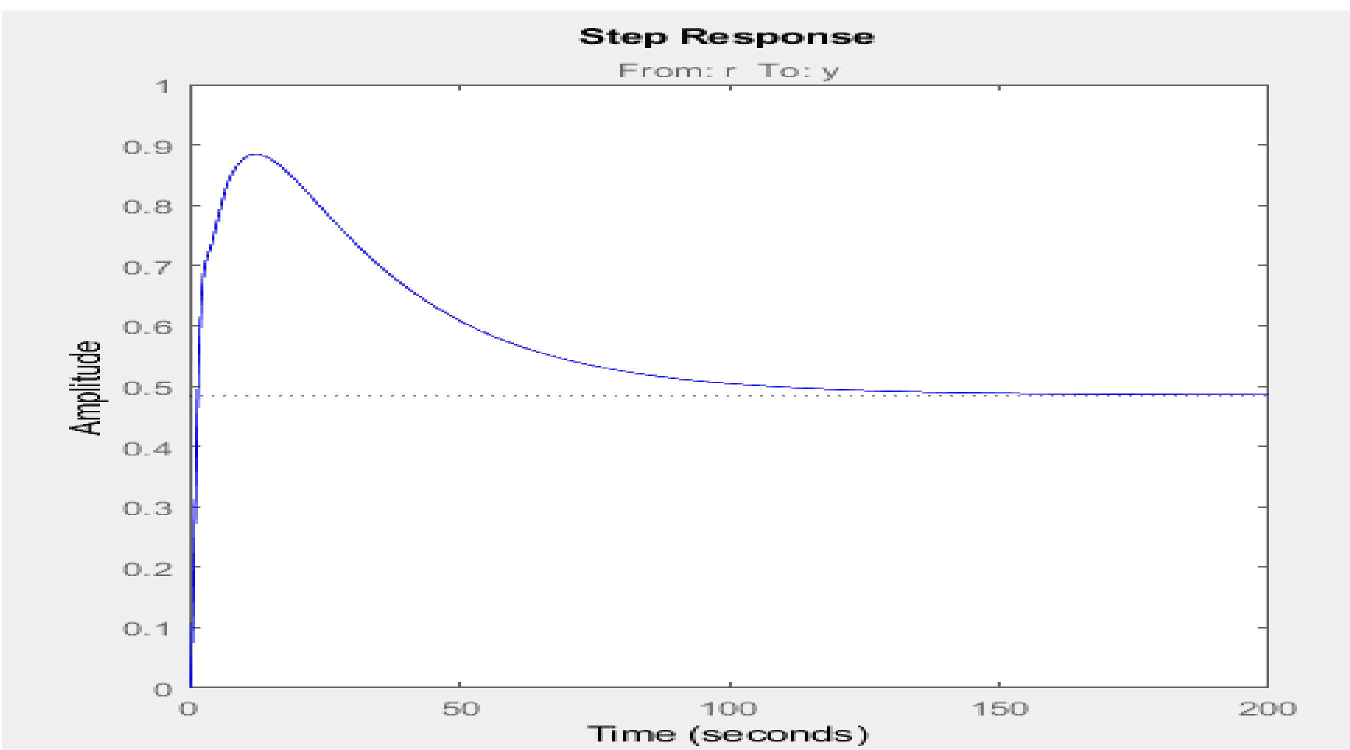


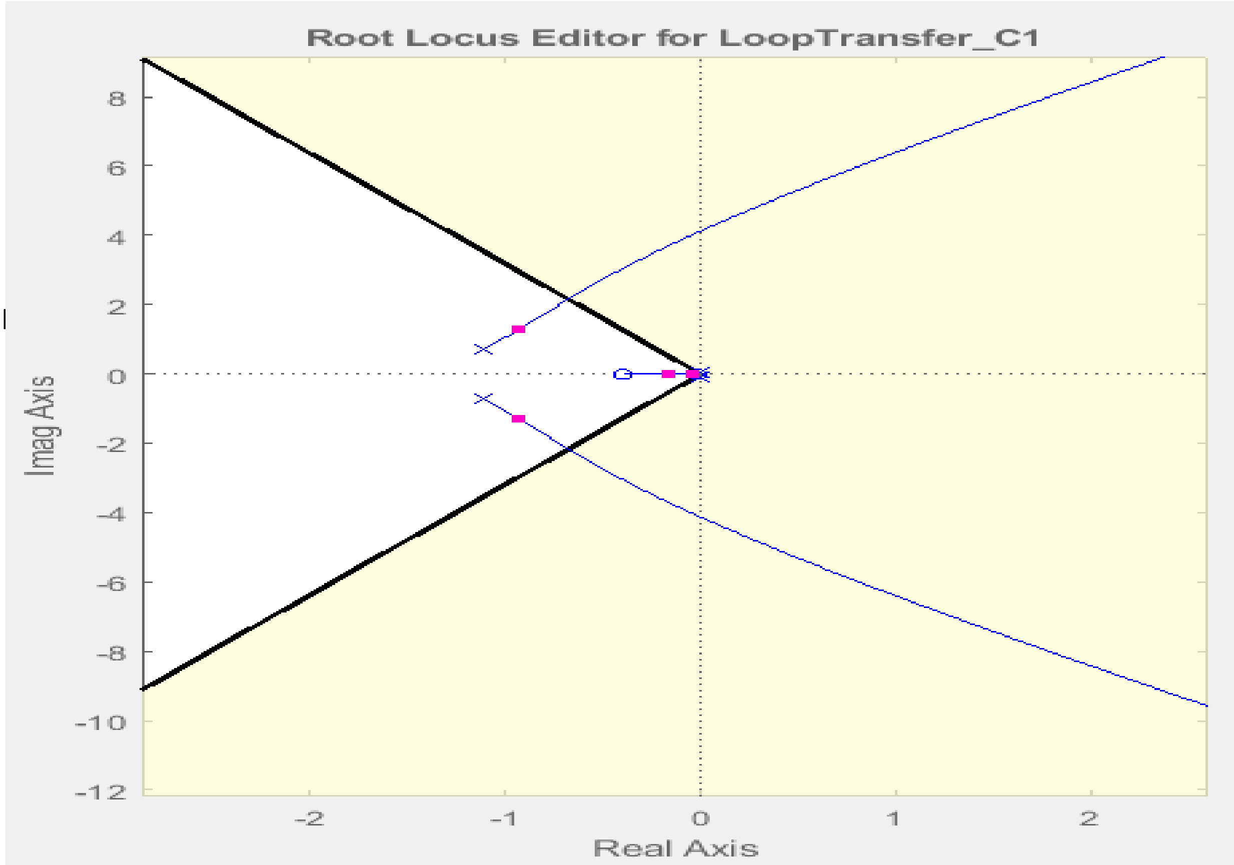
we must know that there are 2 loops inner and outer, and inner loop affect on outer loop ,at least we should put zeda >= 0.04 as we mention it before. And must see control action effect we can't neglect it, we do tuining by moving poles and get new values of gain and we get

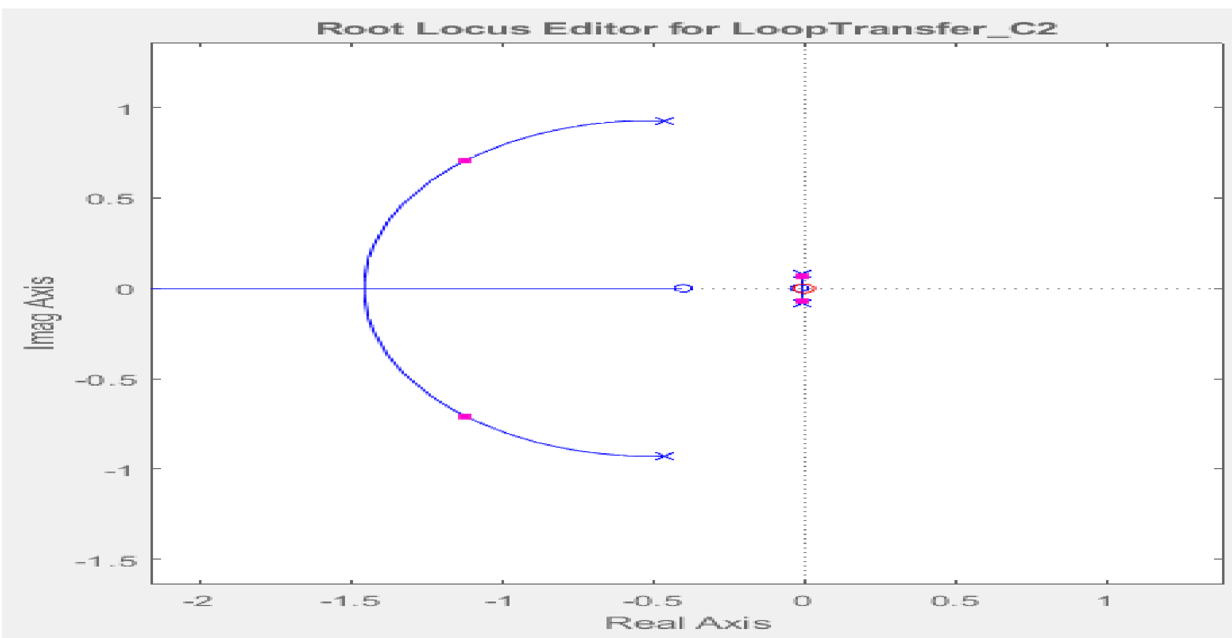
$C1=\frac{1.6941\,(s+0.4667)}{s}$, c2=1.696s but as we see it is not the best result so we will do tuning again.

Design 2





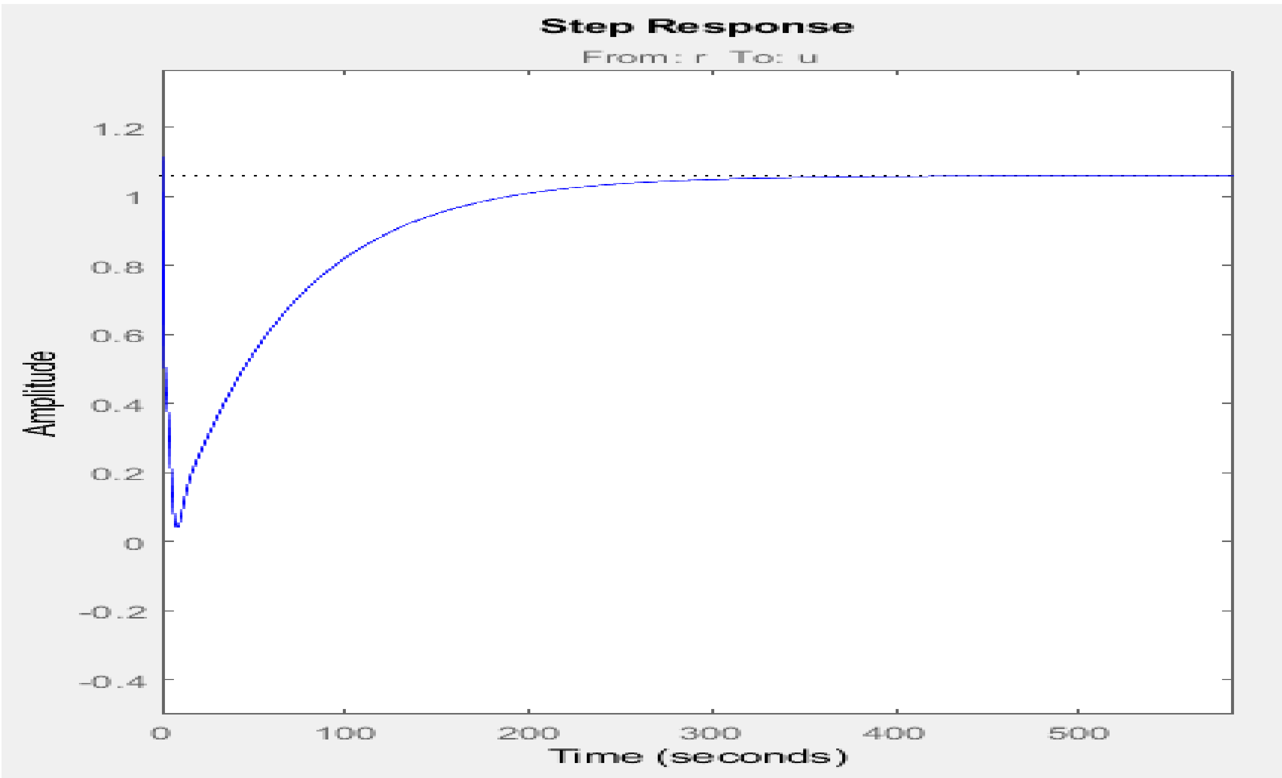


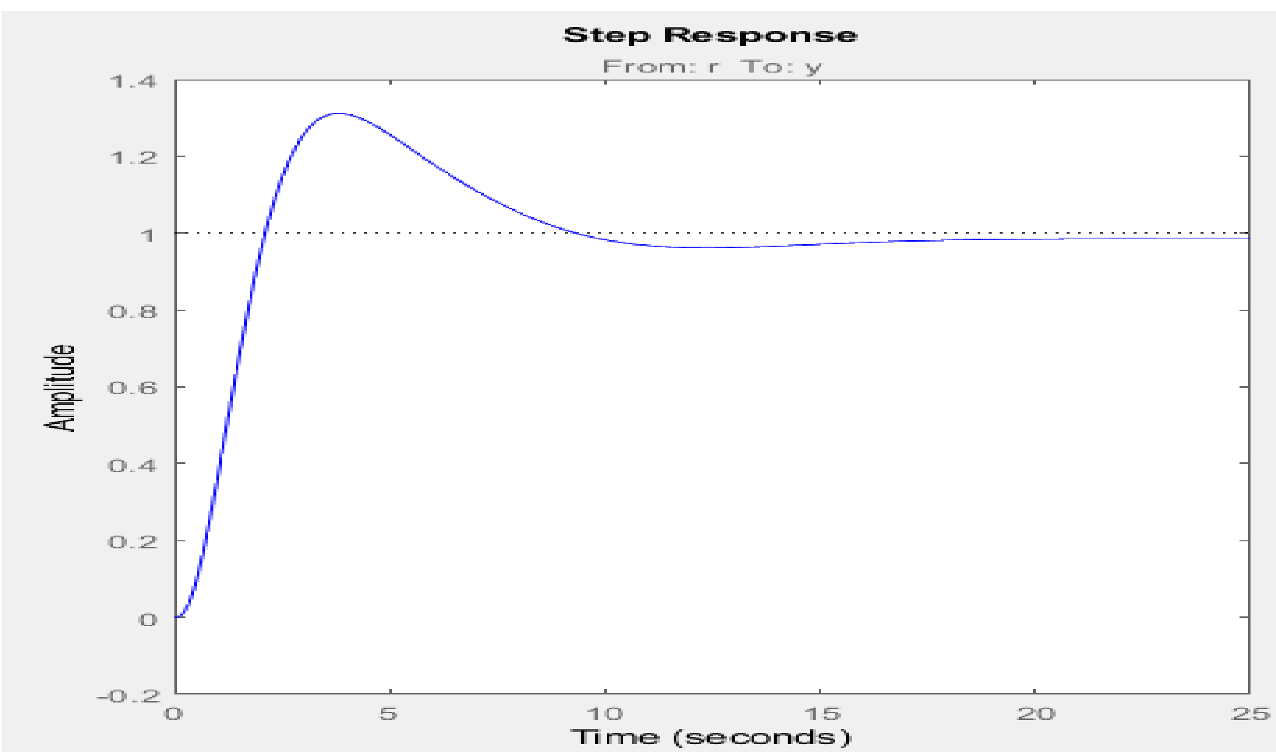


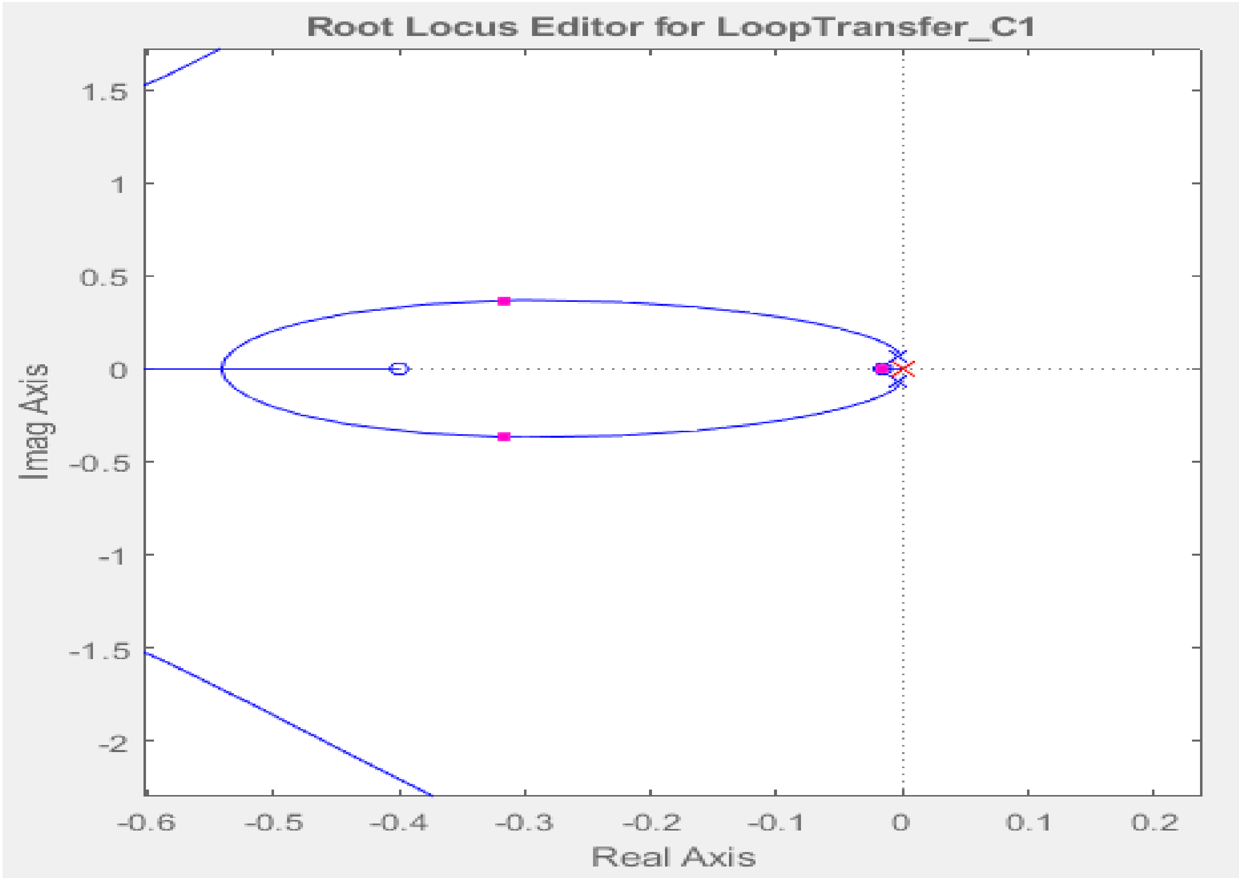
As we see when we do tuning we improve the result and we get new values of gains

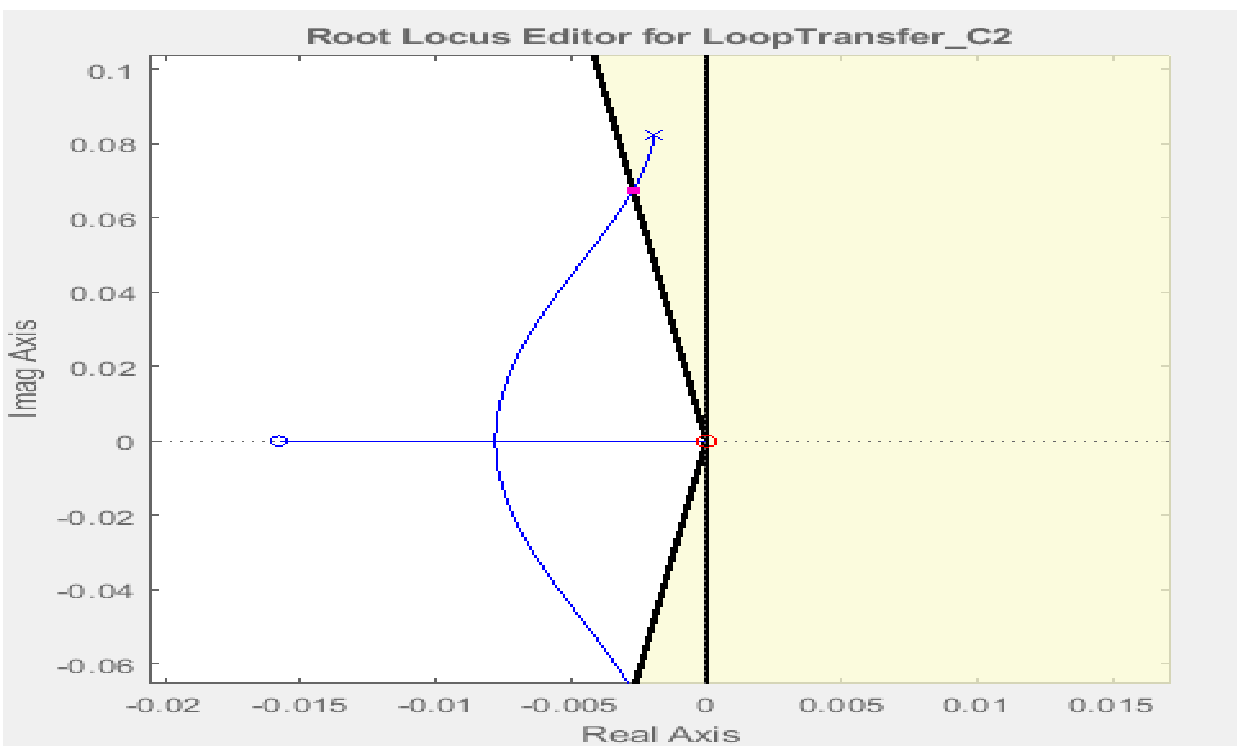
C1= $\frac{1.8564\ (s+0.4107)}{s}$, c2=1.696s, but it isn't the best result and we can get settling time and overshot better than that so we will another tuning.

Design 3 and the best result

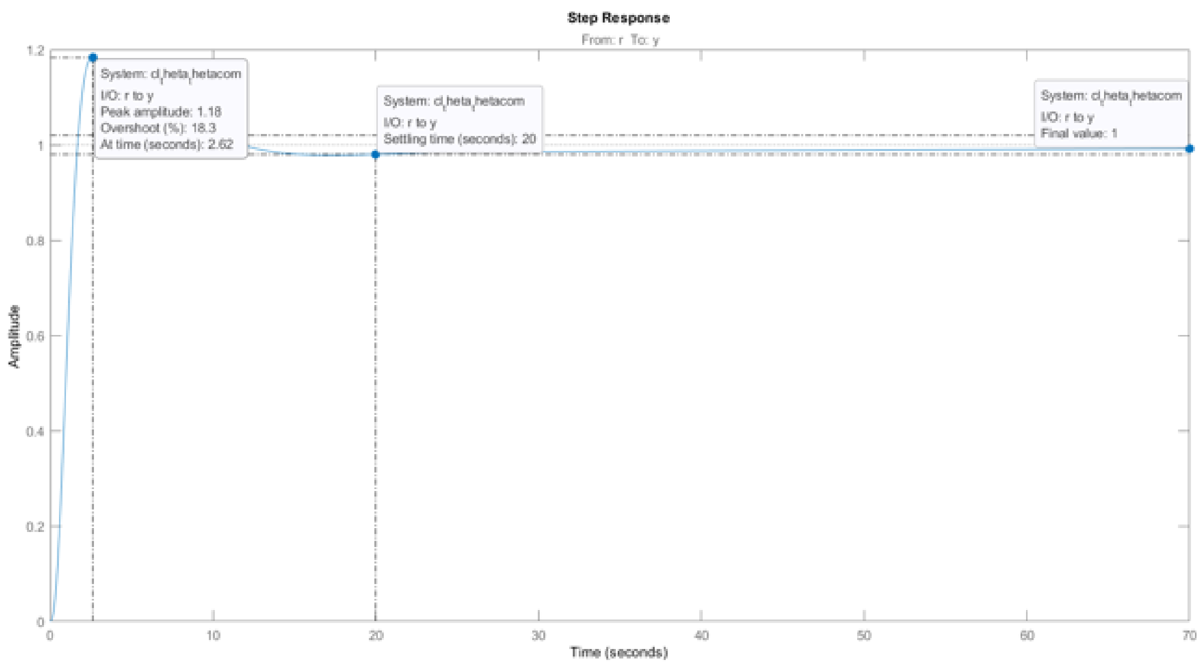





Root Locus Editor for LoopTransfer_C1
Imag Axis
Real Axis
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2


Root Locus Editor for LoopTransfer_C2
Imag Axis
Real Axis
0.1
0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
-0.02
-0.015
-0.01
-0.005
0
0.005
0.01
0.015

As we see we can say it is almost the best results by another tuning we get C1=$\frac{1.0861\ (s+0.728)}{s}$, c2= 1.691s , the settling time and overshoot and control action have improved we can show that in the next photo



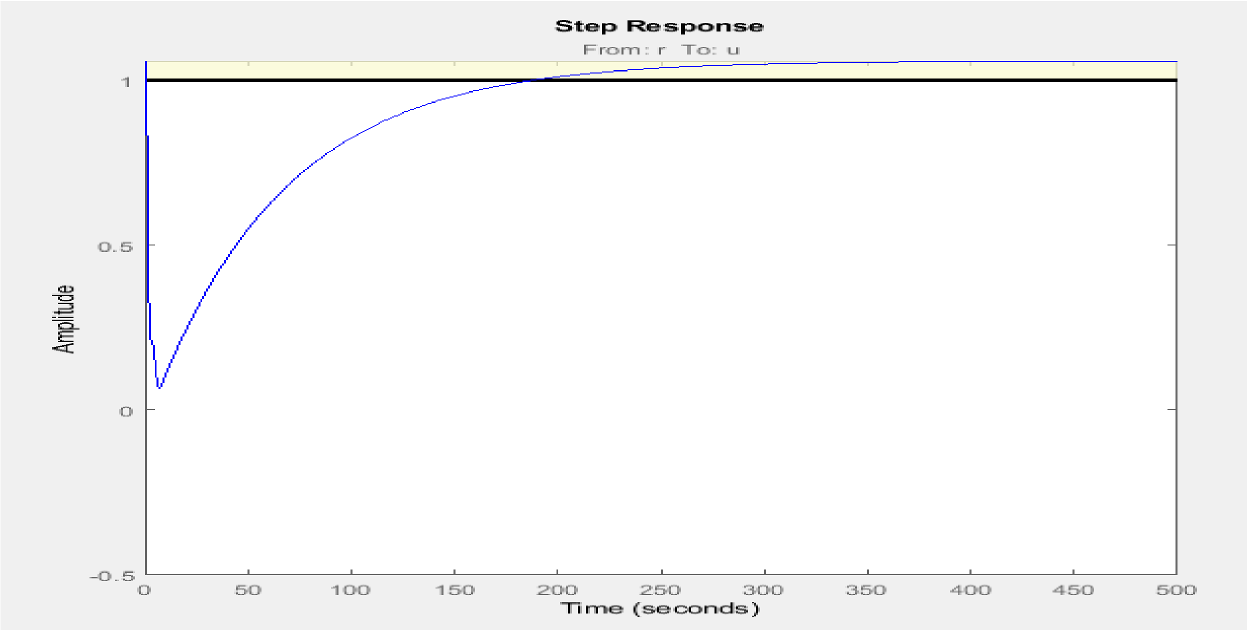
We can see by number the settling time and the overshoot and peak.it is better

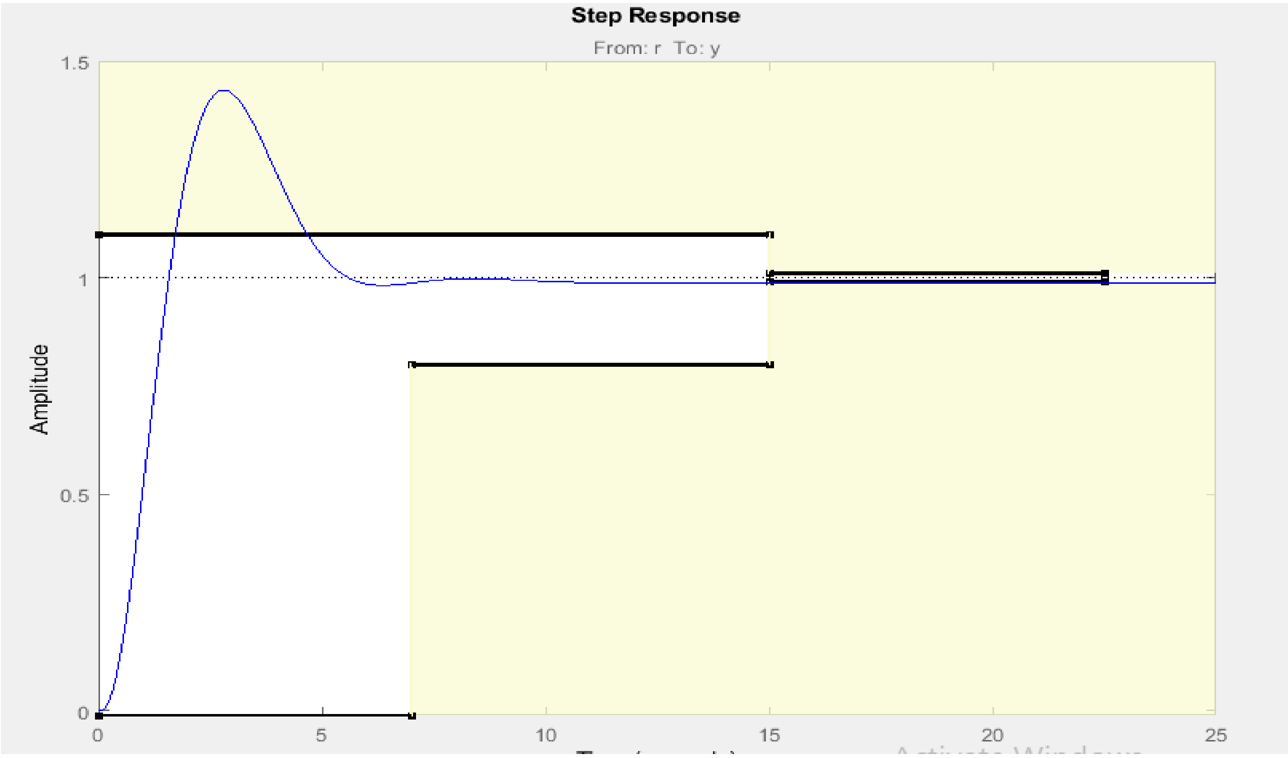
Iteration we tried to use:

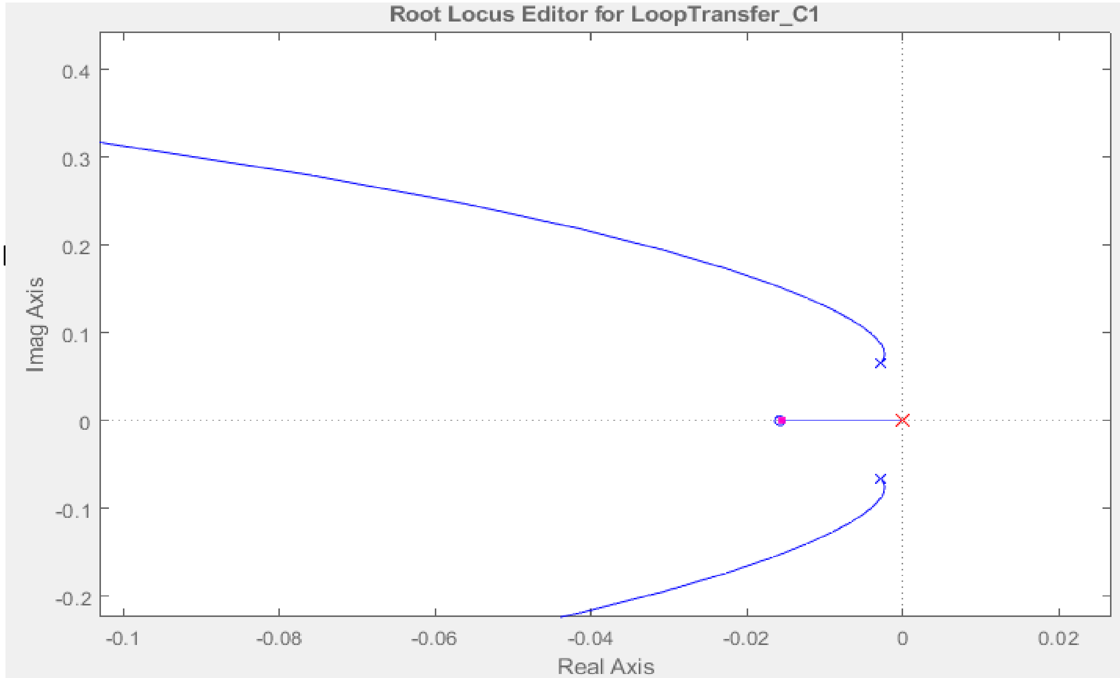
We know we have two loops inner and outer (c1 and c2), and From input "delta_elev" to output:

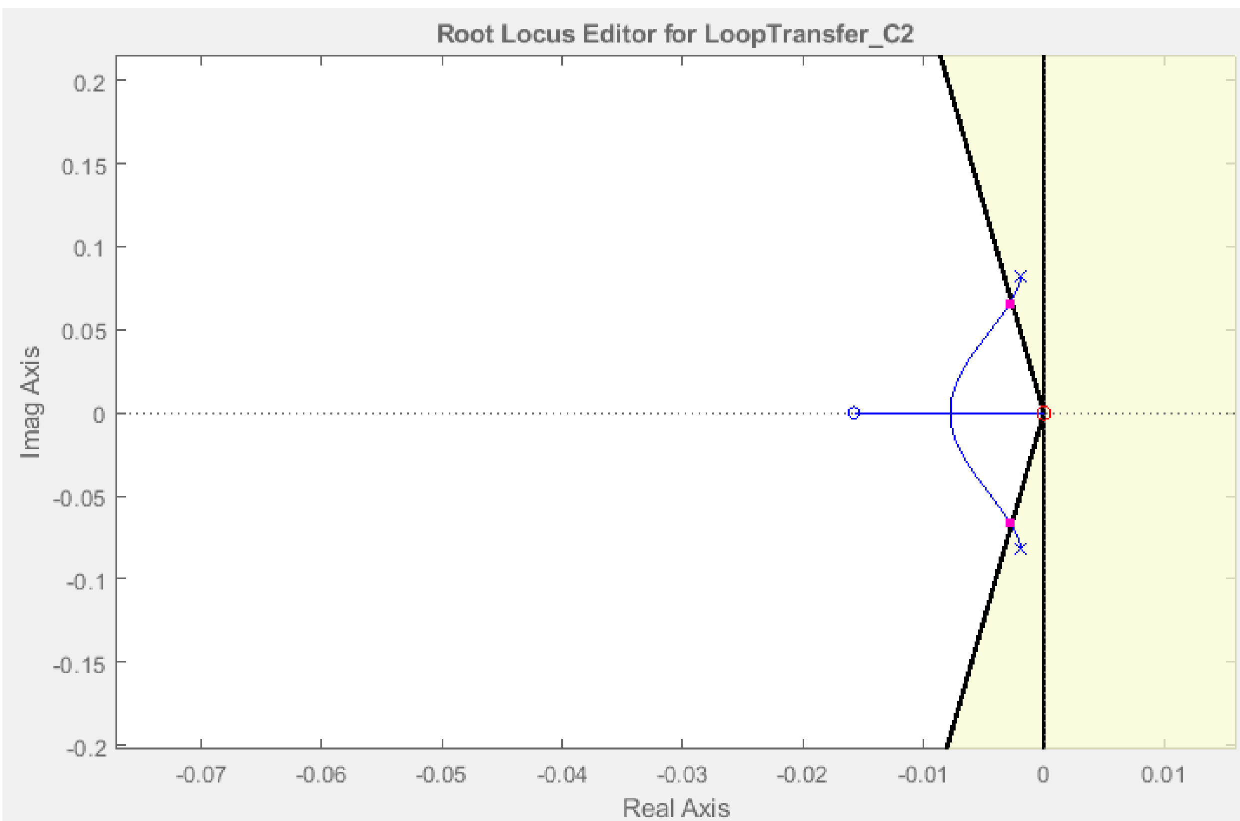
$$G=\frac{10.88\ s^{\wedge}2 + 4.525\ s + 0.06868}{s^{\wedge}5 + 10.93\ s^{\wedge}4 + 10.36\ s^{\wedge}3 + 10.86\ s^{\wedge}2 + 0.111\ s + 0.07276}$$

First iteration



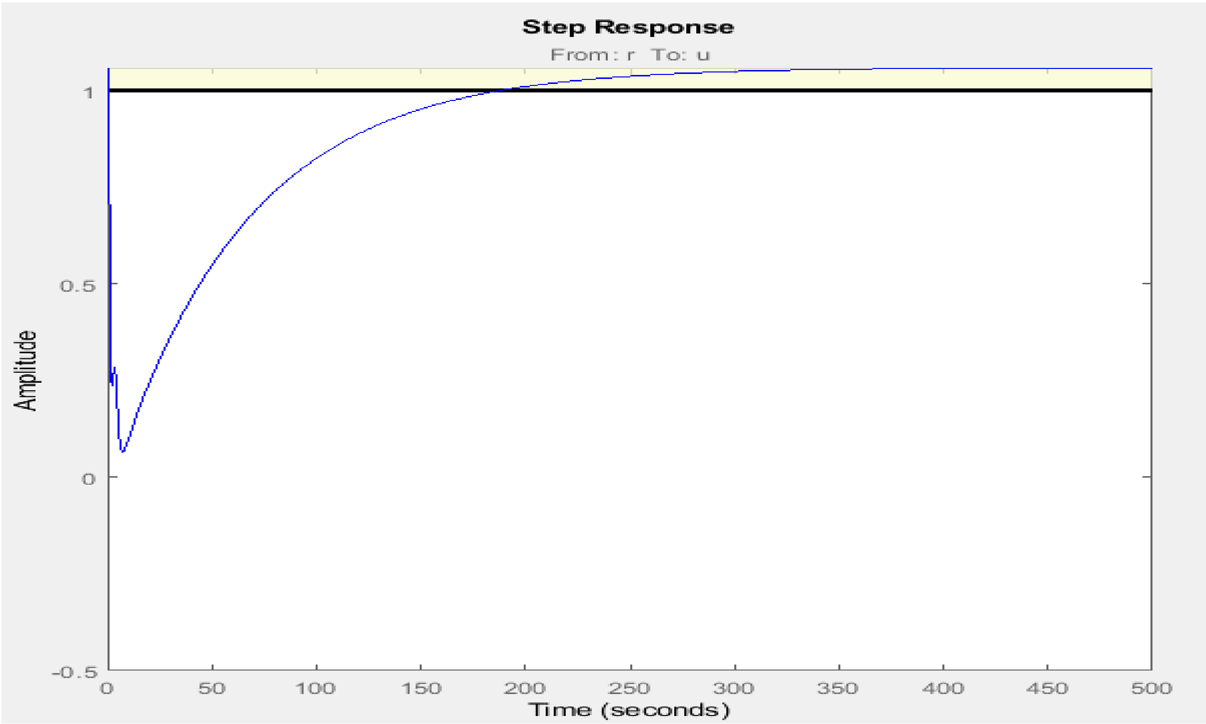


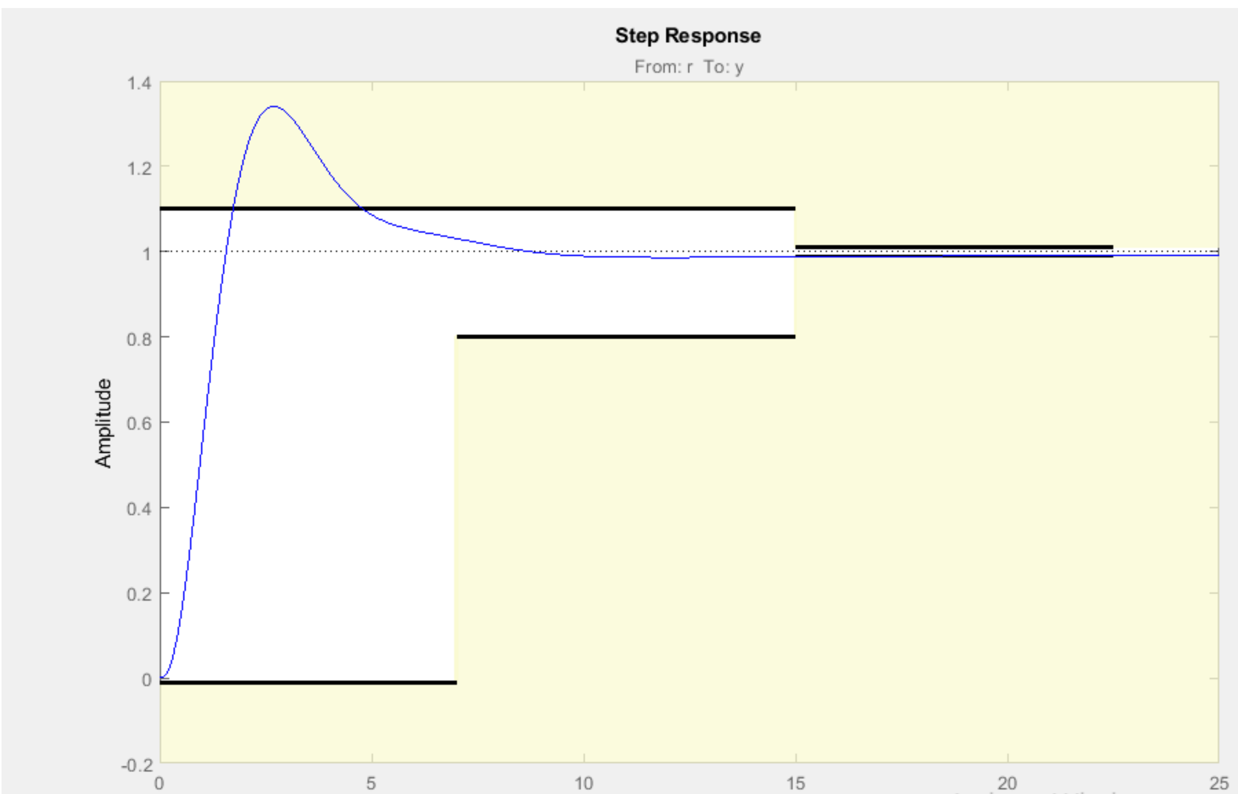


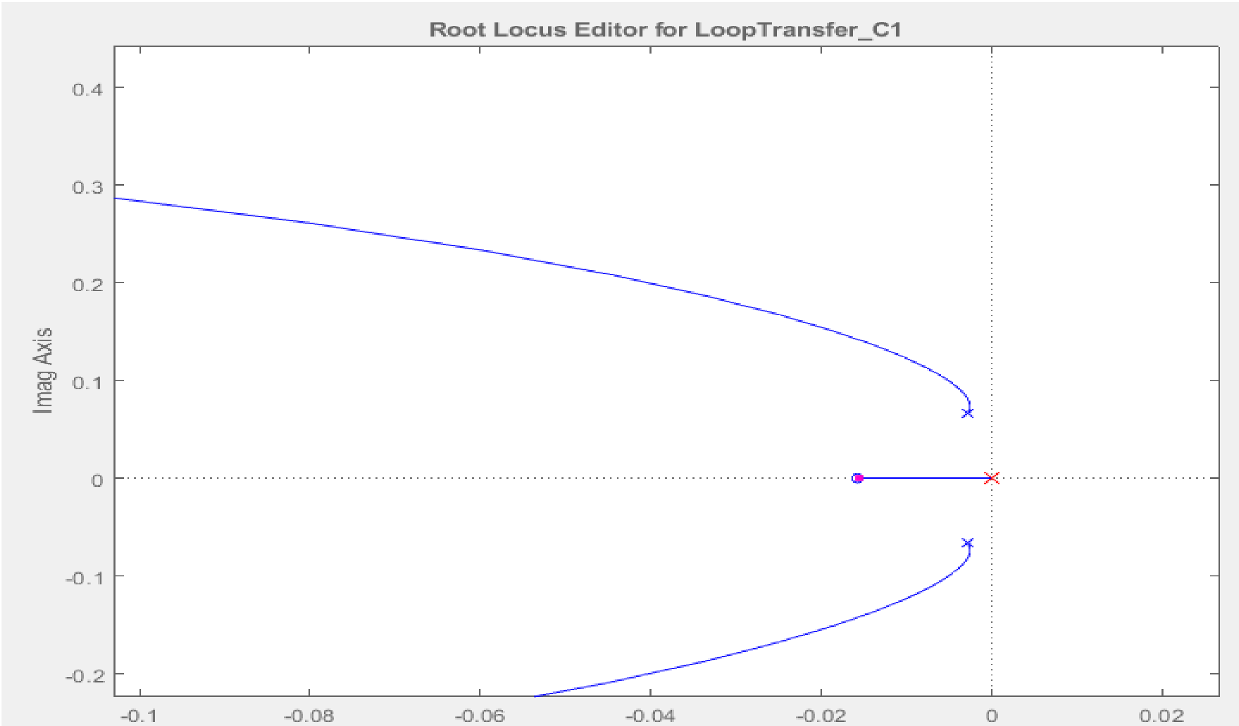


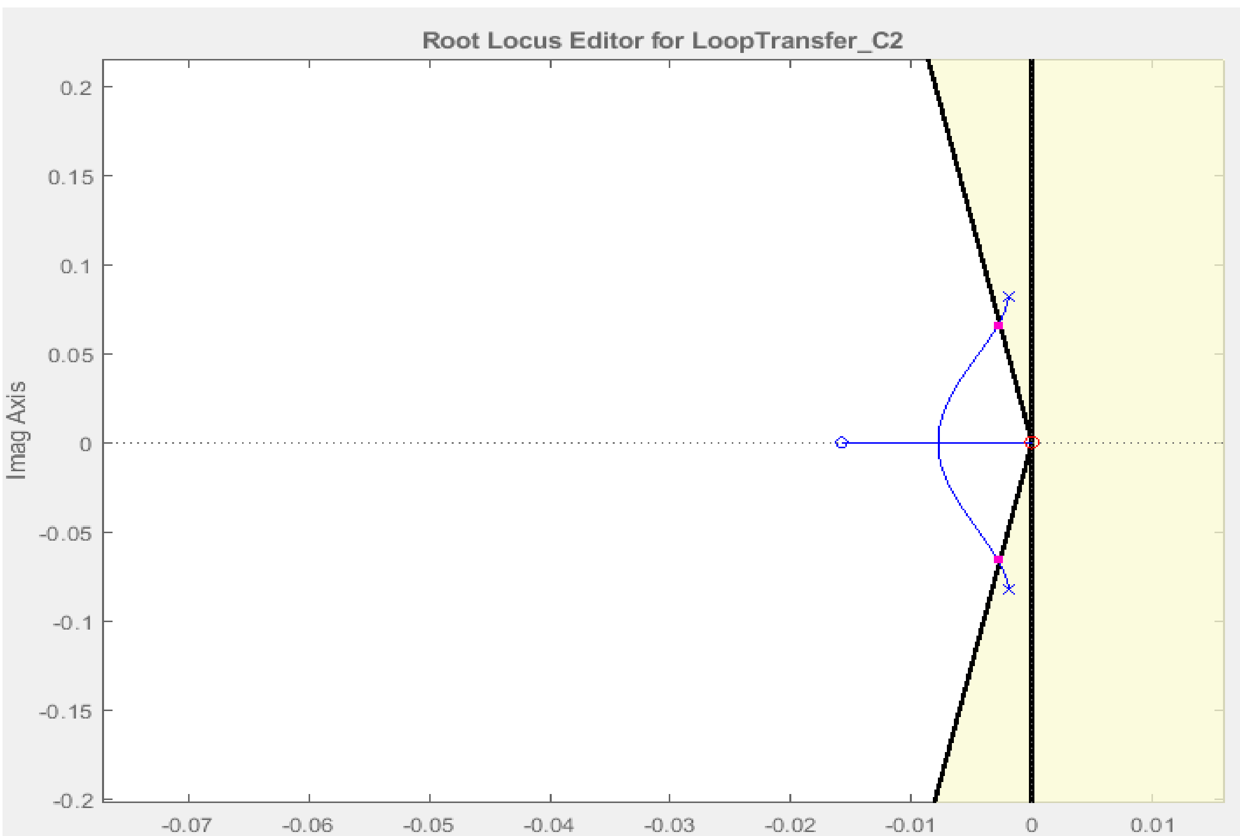
C1= $\frac{1.5511\,(s+0.949)}{S}$, c2= 1.3321s

Second iteration









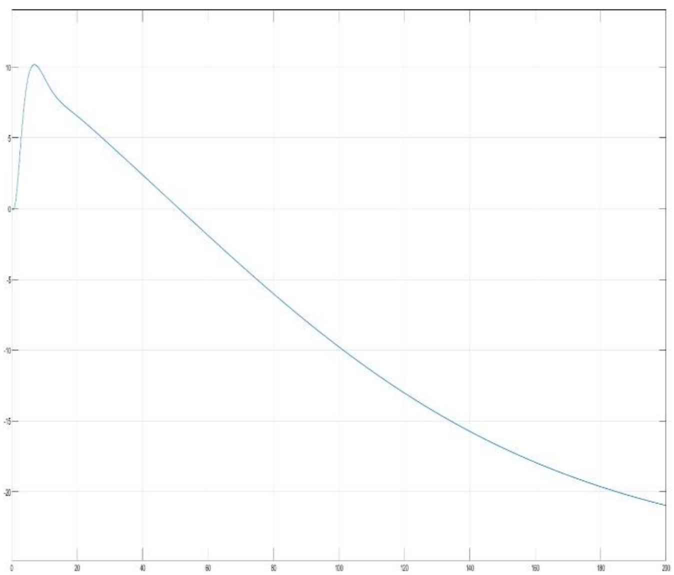
C1= $\frac{1.7623\,(s+0.728)}{s}$, c2=1.3321 , we can use this way automatically.

Test:

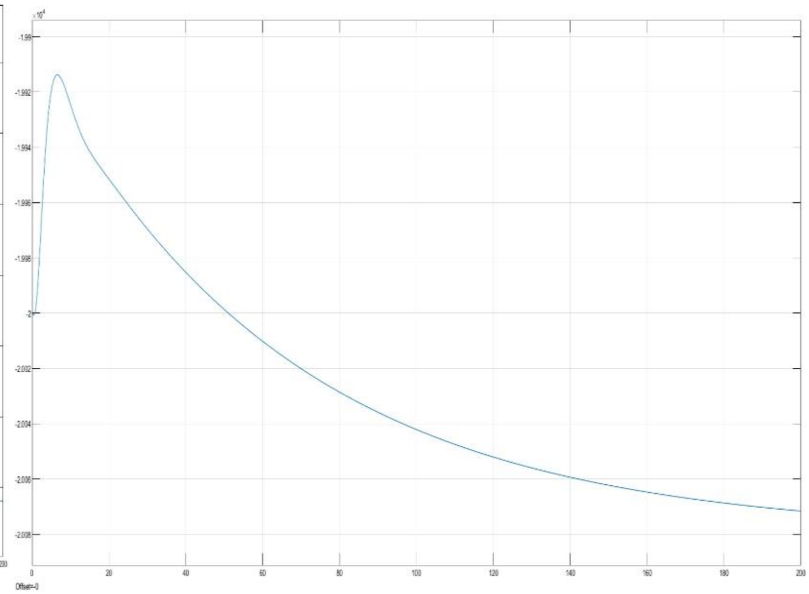
we have to test to make sure is every think ok and see what happened if we give inputs and see what the settling time , se use Simulink and we upload the model of it and see the results when you put input. And we try to put input pitch angle = 15 degree we get little settling time and deflection wasn't worse (about -23) .

but what about gamma and height

gamma



height



from gamma fig, if the give the plane positive theta it will pitch down not up and from height figure, first Hight increase but after that will decrease! And those opposite of our expections.

task 5 part2[long.Autopilot]

first we need to mention that before we changed our plane to Boeing 747_Fc5, we had worked on plane NT_33A_Fc2 because we faced problems with stability of the plane but we tried to search and solve this problem and we will discuss the solution.

Velocity control for NT_33A_Fc2:

initially we tried to control the velocity using thrust However, upon extracting the open-loop transfer function, represented as:

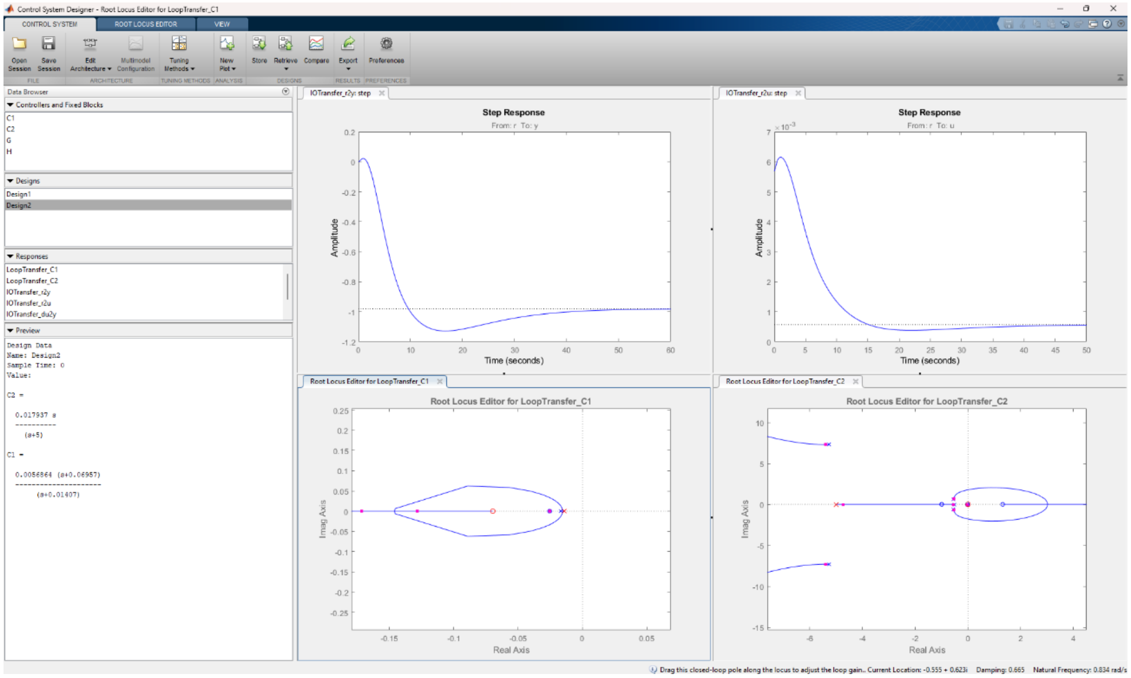
$$\frac{u_{command}}{\delta_{thrust}} = \frac{0.00235(s+2.581)(s-0.3869)(s-0.006554)}{(s+10)(s+2.577)(s-0.379)(s+0.1)(s^2+\ 0.004331s\ +\ 0.01364)}$$

We observed the presence of a pole and two zeros in the right half-plane, indicating system instability. This configuration implies that a portion of the root locus extends into the right half-plane, consequently resulting in a closed-loop pole residing in the unstable region. Hence, despite our control approach, the inherent instability of the system renders it uncontrollable.

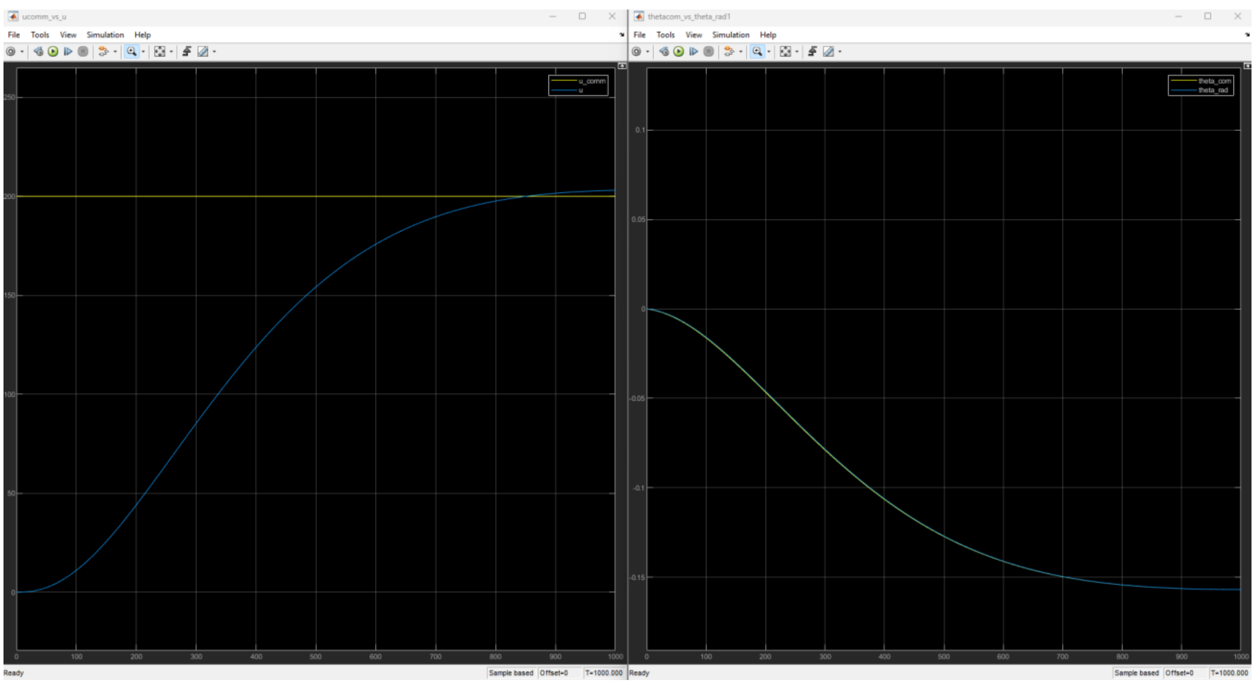
We tried to overcome that problem by controlling the velocity using theta command and we get the following transfer function :

$$\frac{u}{\theta_{command}} = \frac{-7.41\ (s+96.56)(s+1)(s^2+\ 0.8422s\ +\ 1.345)}{(s+0.5118)(s+0.02569)(s^2+\ 1.109s\ +\ 0.696)(s^2+\ 10.56s\ +\ 81.13)}$$

We managed to design a controller using siso tool in MATLAB as the following :



And after testing on the Simulink model we get the following response :



Height controller:

We proceeded with the design of the height controller, intending to utilize thrust for stabilization. However, our efforts were impeded when examination of the transfer function revealed instability, with a pole and two zeros located in the right half-plane:

$$:\frac{-2.6452*10^{-6}(s-9.277)(s+10.5)(s+3.362)(s-0.3067)(s+0.006833)}{s\,(s+10)(s+2.577)(s+0.9929)(s-0.379)(s+0.1)(s+0.02554)(s^2+0.004331s+0.01364)}$$

Subsequent scrutiny of all transfer functions derived from the longitudinal linear model unveiled a recurring pattern: every transfer function associated with the delta thrust command exhibited a pole and a zero in the right half-plane, rendering the system intrinsically unstable. Consequently, we were compelled to abandon this approach, necessitating a reassessment of our aircraft's control strategy.

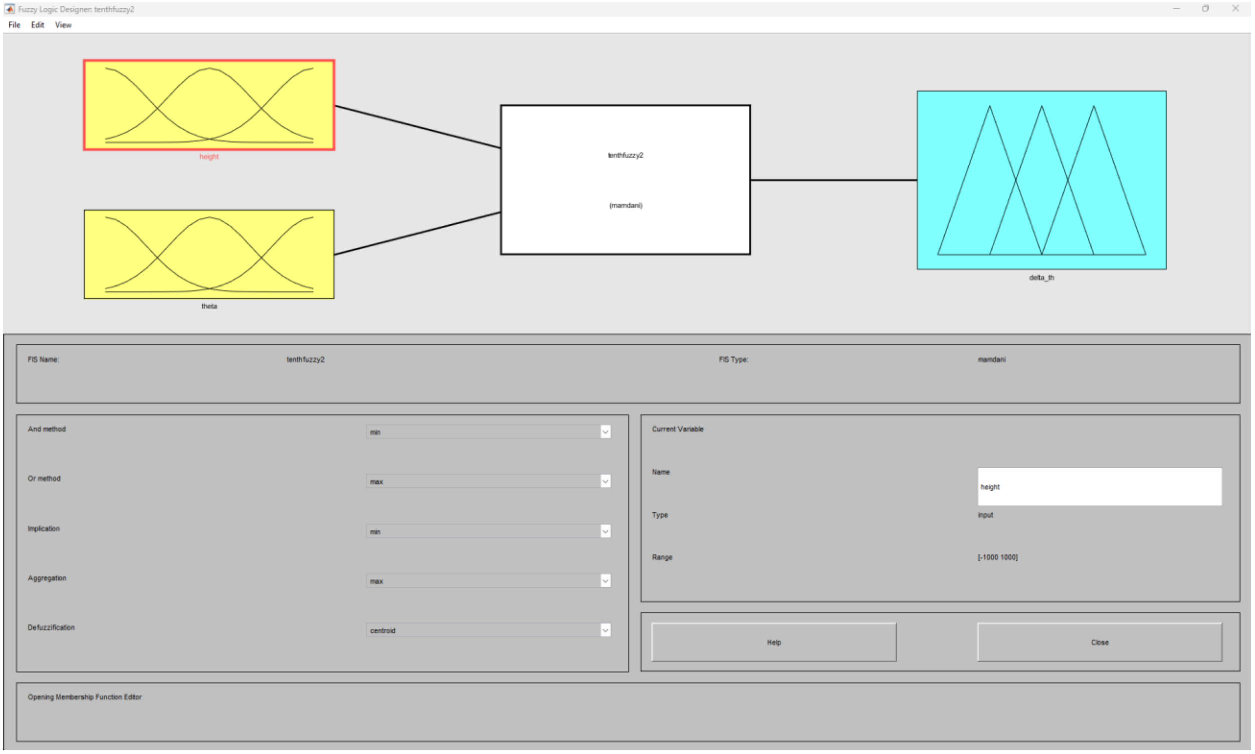
Searching for a solution that we faced:

We tried to search for a solution for the problem we have faced and we found the solution which is designing a fuzzy controller to control the height.

Fuzzy controller:

At its core, a fuzzy controller consists of three primary components: fuzzification, rule evaluation, and defuzzification. During fuzzification, crisp inputs are converted into fuzzy sets, representing degrees of membership in linguistic terms. These fuzzy sets are then evaluated against a set of predefined fuzzy rules, which capture the expert knowledge or empirical data about the system's behavior. The rule evaluation process aggregates the outputs of these rules to generate a fuzzy output, reflecting the system's response to the inputs. Finally, defuzzification translates the fuzzy output back into a crisp value, providing a concrete control action to be executed within the system. This holistic approach allows fuzzy controllers to effectively manage complex systems with uncertain or variable parameters, making them invaluable tools in engineering and control theory.

Our strategy involved prioritizing system stability, even if it meant compromising on achieving the ideal time response. In this endeavor, we opted to employ the difference between the height command and the height feedback, along with the theta feedback, as inputs. The output of interest was defined as delta thrust. The design process of the fuzzy controller was facilitated using the Fuzzy Designer tool within MATLAB, as illustrated in the following figures.



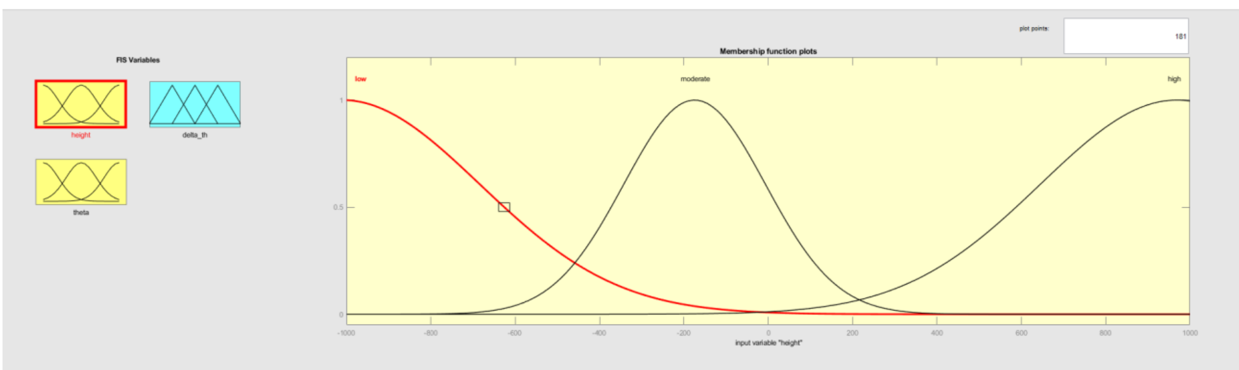


Figure 5(membership functions of input height)

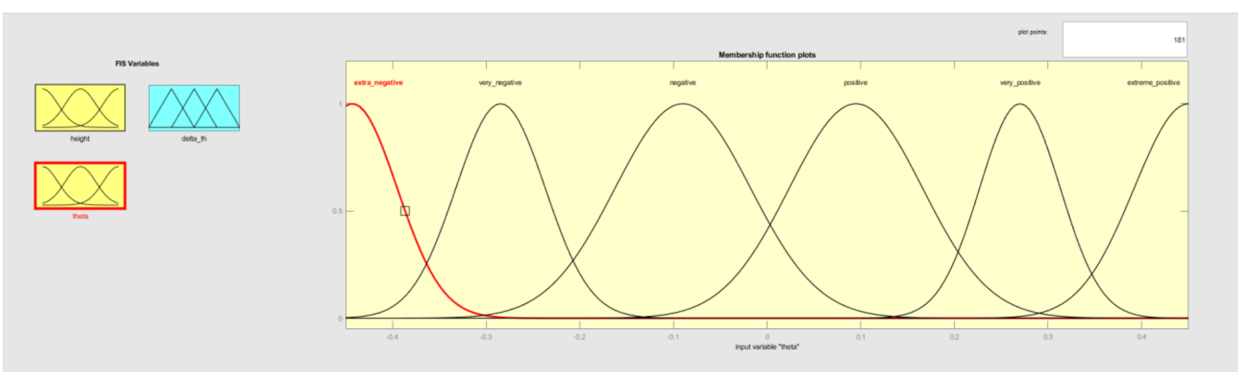


Figure 6(membership functions for theta input)

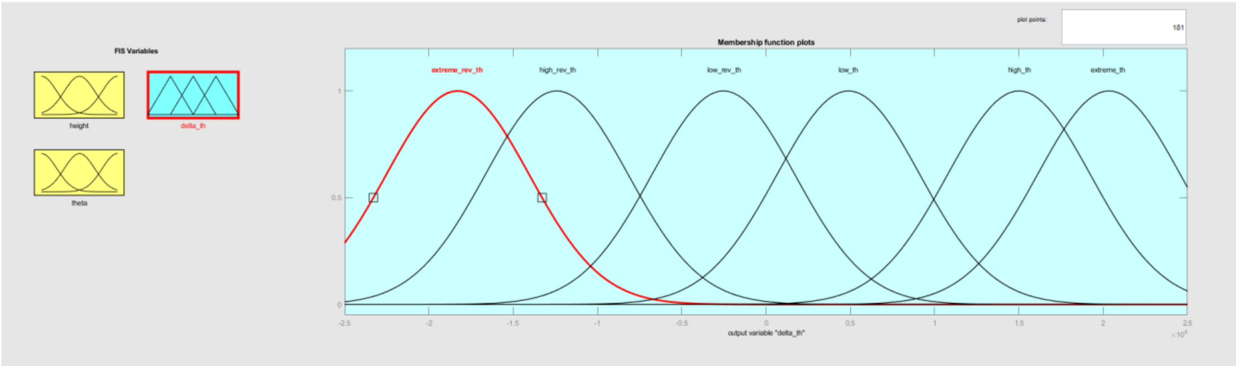
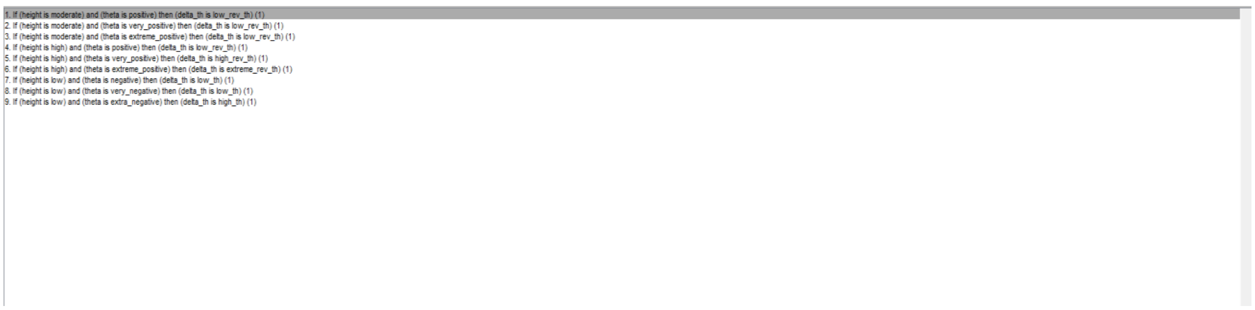


Figure 7(membership function for output delta thrust)



1. If (height is moderate) and (theta is positive) then (delta_th is low_rev_th) (1)
2. If (height is moderate) and (theta is very_positive) then (delta_th is low_rev_th) (1)
3. If (height is moderate) and (theta is extreme_positive) then (delta_th is low_rev_th) (1)
4. If (height is high) and (theta is positive) then (delta_th is low_rev_th) (1)
5. If (height is high) and (theta is very_positive) then (delta_th is high_rev_th) (1)
6. If (height is high) and (theta is extreme_positive) then (delta_th is extreme_rev_th) (1)
7. If (height is low) and (theta is negative) then (delta_th is low_th) (1)
8. If (height is low) and (theta is very_negative) then (delta_th is low_th) (1)
9. If (height is low) and (theta is extra_negative) then (delta_th is high_th) (1)

Figure 8(rules of fuzzy controller)

We got the following response after placing the fuzzy controller in Simulink :

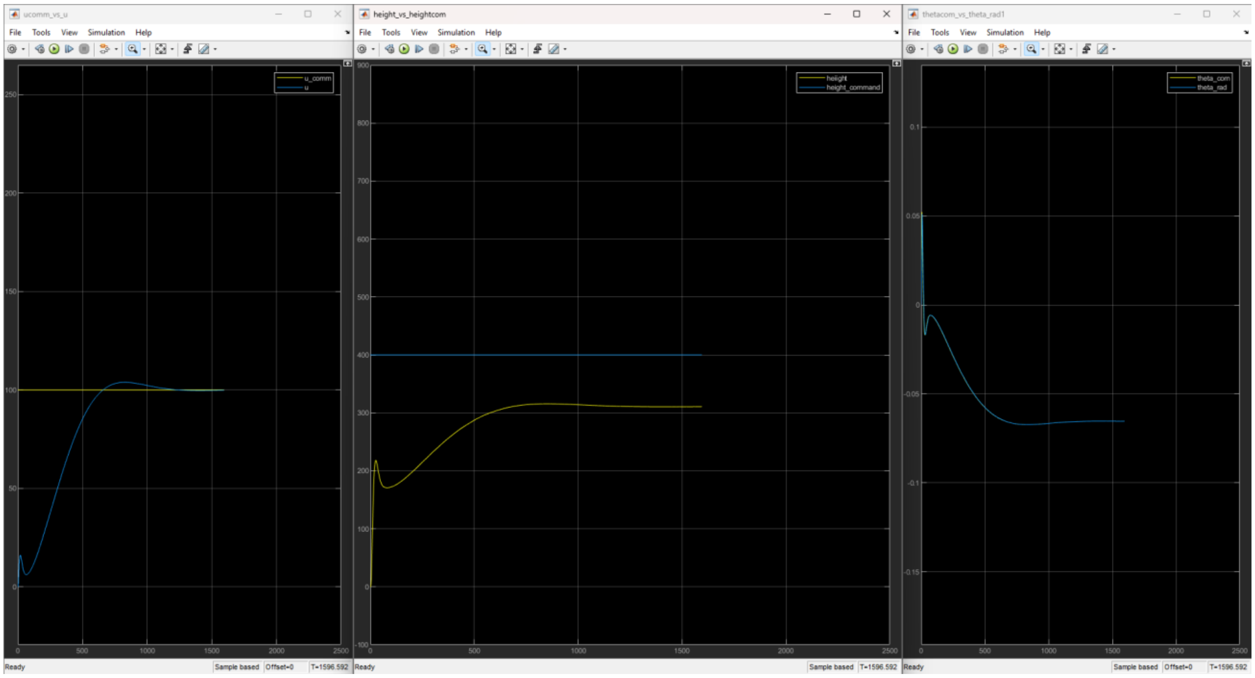


Figure 9(time response of u com=100 and height commad=400)

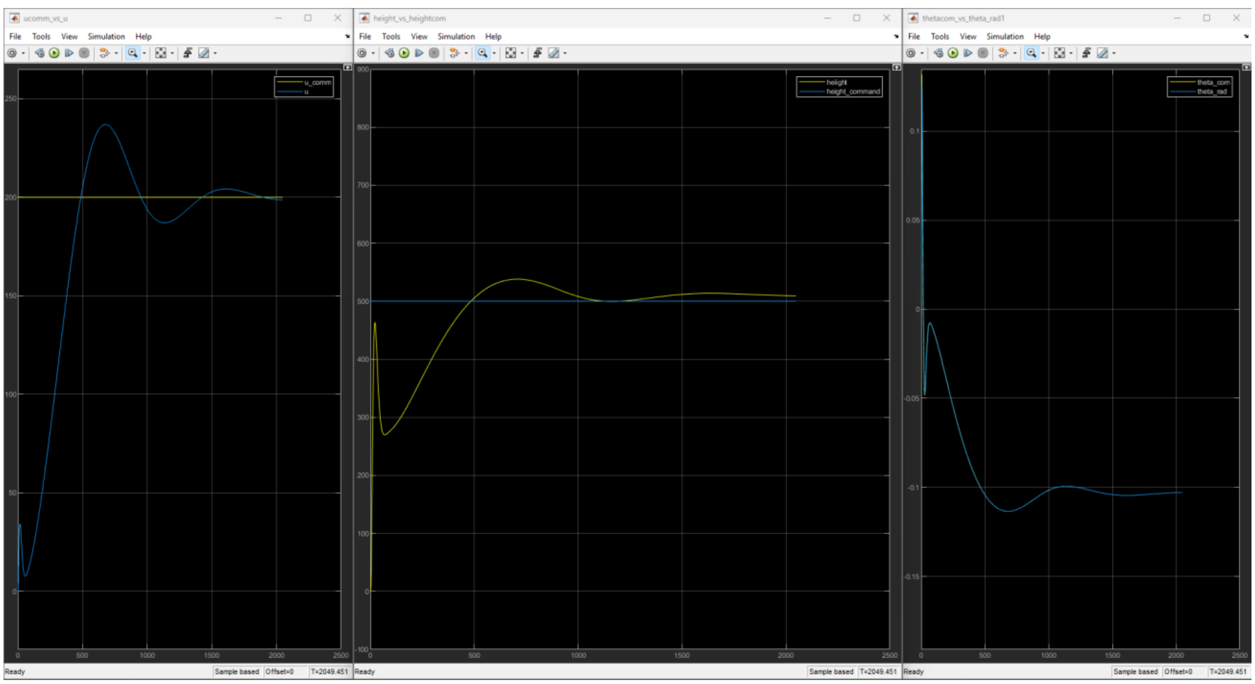


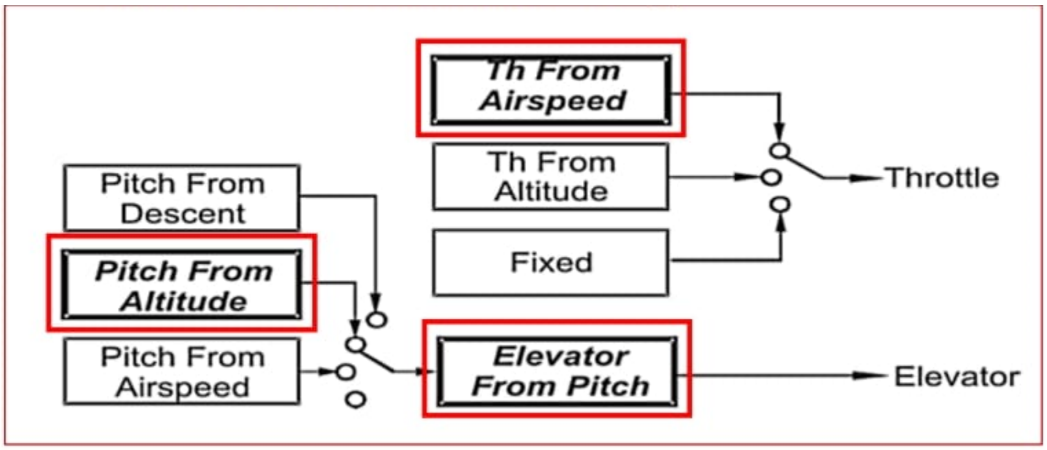
Figure 10(time response for u command =200 and height command =500)

Our observations revealed that despite not attaining the desired system characteristics, the implementation of the fuzzy controller successfully achieved stability within the thrust system. This outcome underscores the potential benefits of utilizing such a controller. Moreover, we recognized the opportunity for further improvement in the system's characteristics through meticulous tuning of the controller parameters.

overview

in this task we need to control the motion the airplane "control the elevator and thrust to achieve the desired command of altitude or climb angle.

In the previous task we had designed a pitch controller but we face to problem the plane down so it need energy and we make velocity control to solve this problem and the second problem height decrease and we will control it too.



We will ger elevator from pitch as inner loop and pitch from altitude , and throttle from airspeed

Velocity control:

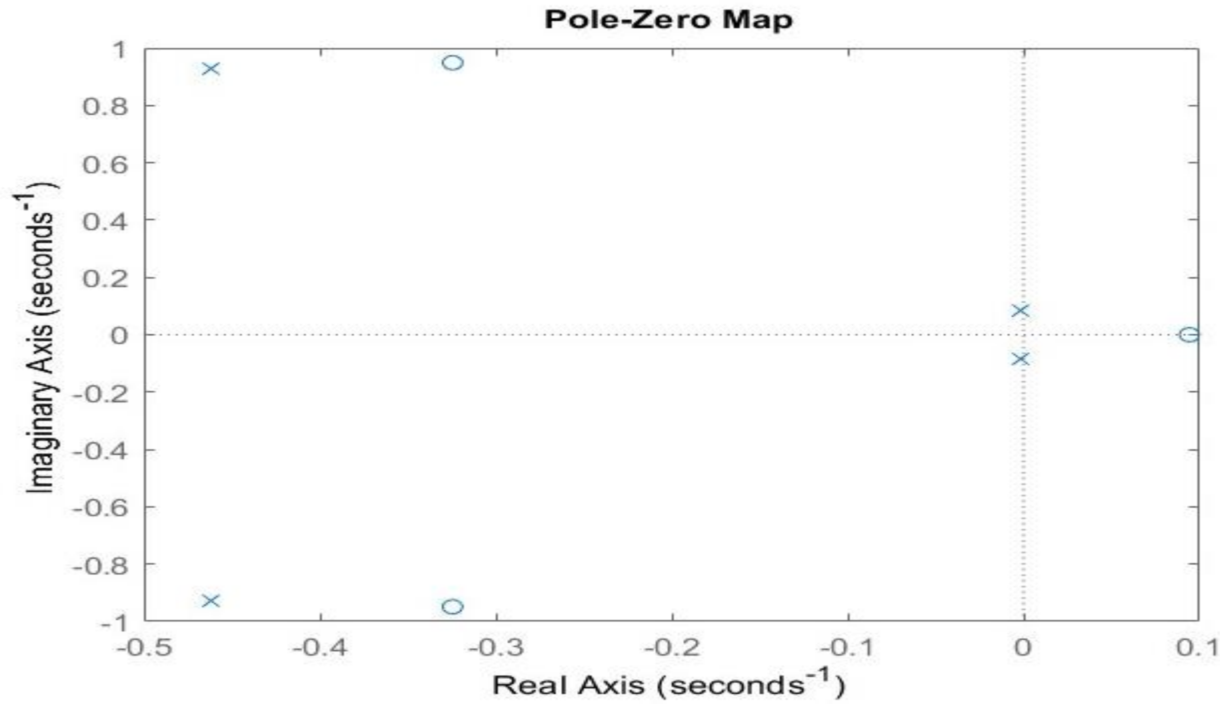
We get $\frac{u}{\delta_{th}}$ but we also will add the transfer function of throttle servo$\left(\frac{10}{s+10}\right)$

And engine time lag ($\frac{0.1}{s+0.1}$) and we notice that the taw of engine time lag is large and it makes the system slow.

So ol_u_ucomand= $\frac{u}{\delta_{th}} *$ throttle servo * And engine time lag

The rule of the velocity control loop is to keep the forward velocity constant

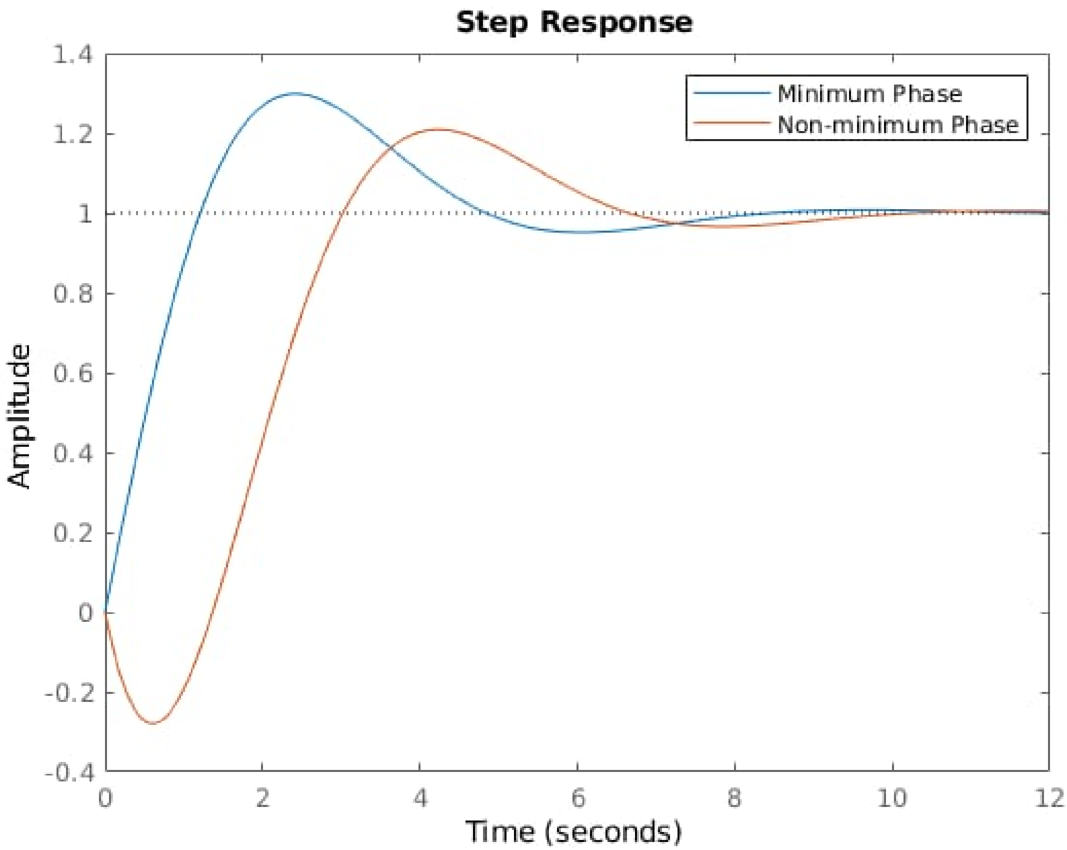
(+ve θ → +ve γ)



But when we look at pole and zero of the transfer function

We will see zero on the right and this system call non minimum phase system, and that not a good system as we will discuss:

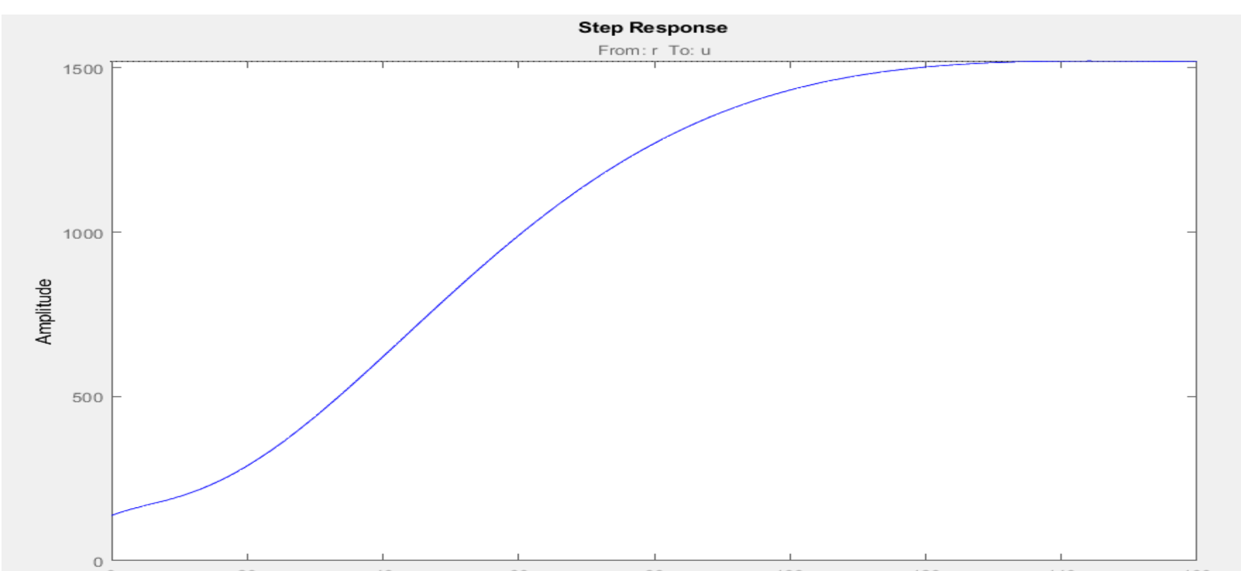
NM phase system

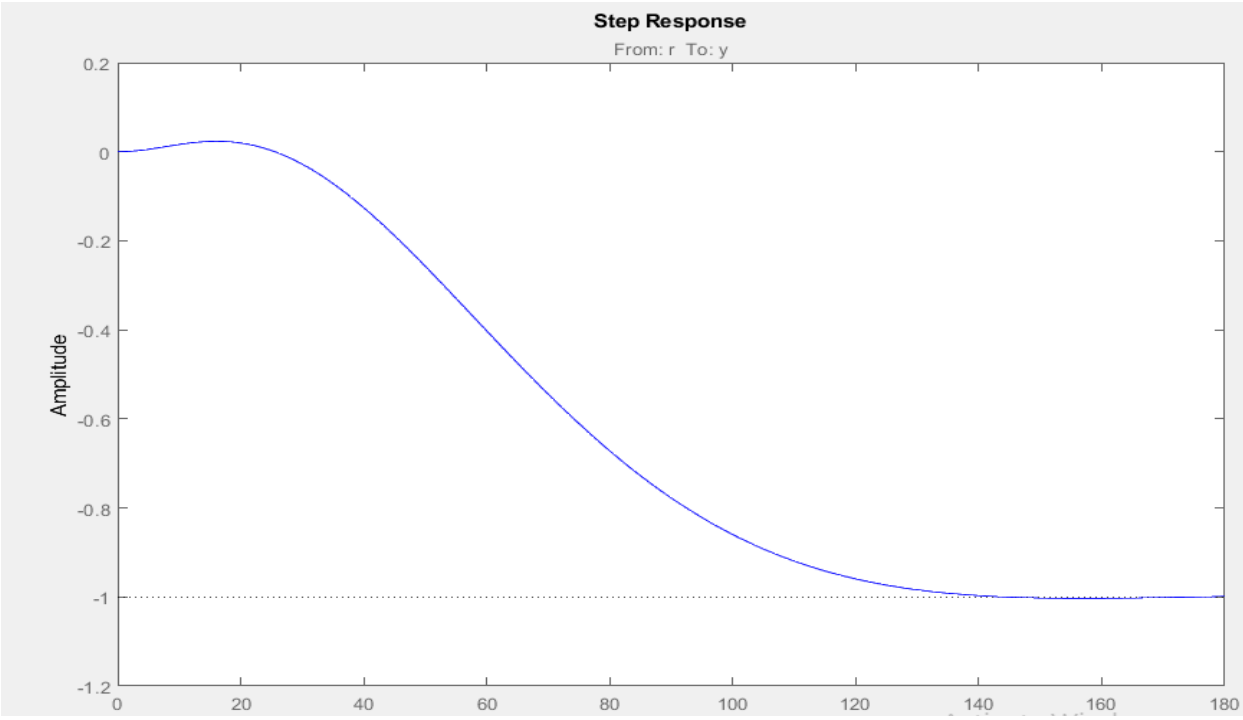


As we see Non-minimum phase system take time bigger to settling much than minimum phase system take so we have to deal and solve this problem in design

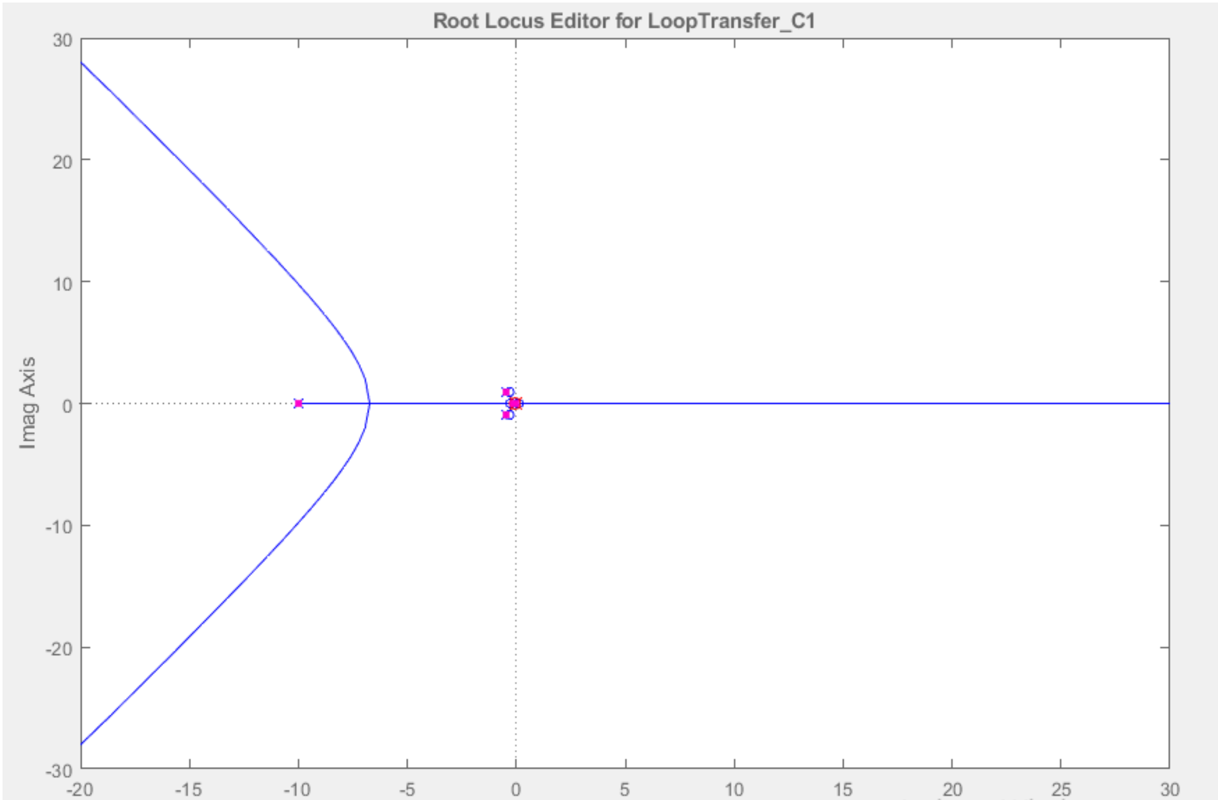
Design of velocity control:

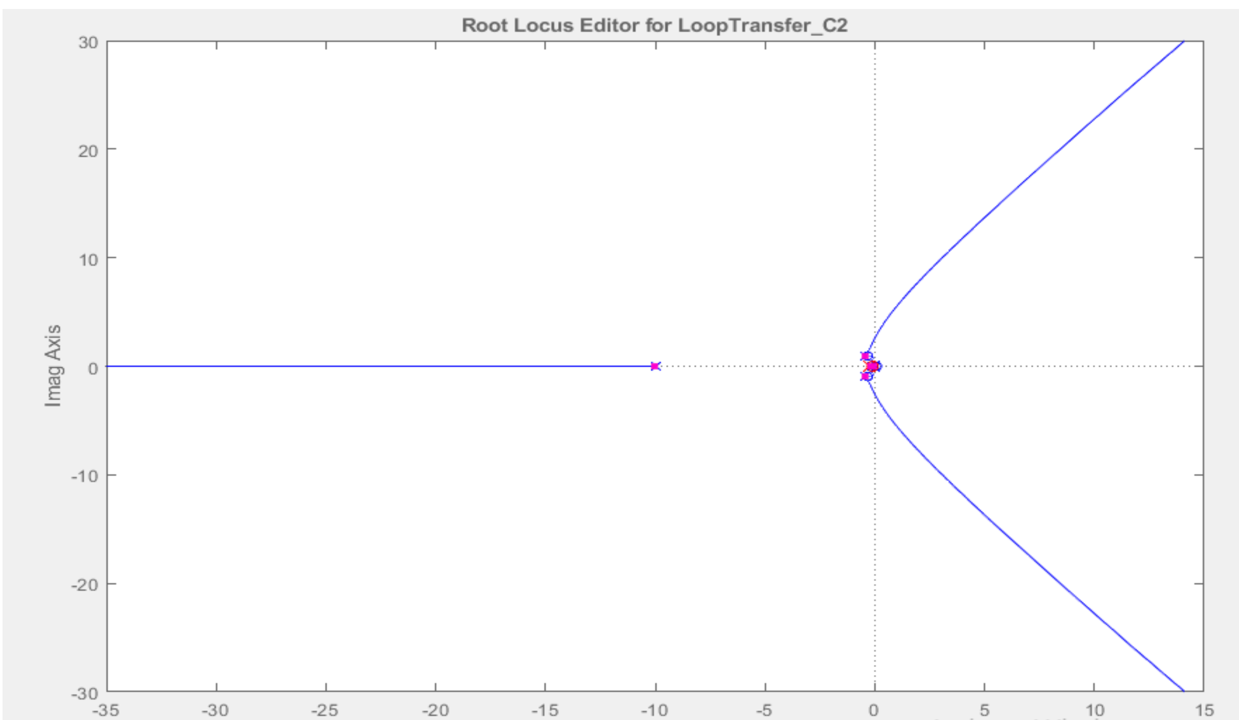
Control action response.





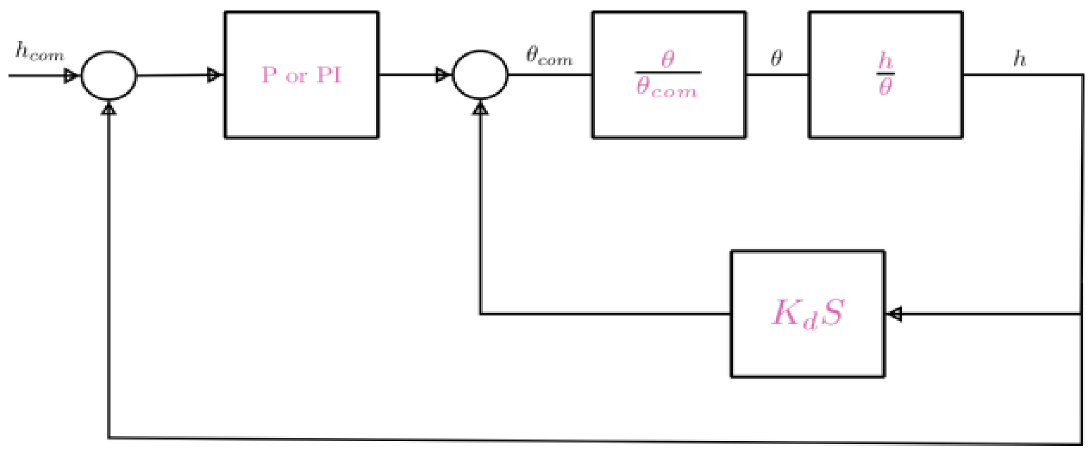
Step response





We should know the inner loop effects on outer loop, and at the first design we have zero at on right and large settling time even we add gain integrator, so we tried to make tuning but it wasn’t enough to solve this problem so change negative feedback outer to positive feedback and make tuning until we reached this result and get suitable settling time and we must not forget the control action it is very important and it depends on thrust of the plane.

H controller:



Where $(\theta/\theta com)$ is the closed loop of the pitch control loop designed in Part I of this task, We then need to relate (θ),we will assume that (h) can be calculated by double integration of a_z

$$\begin{aligned} \ddot{h} &= -a_z \\ h &= -\frac{a_z}{s^2} \end{aligned}$$

From the (RBD) Equations of motion, acceleration in the Z-axis i
After linearization

$$a_z = \dot{w} + pv - qu$$

After linearization

$$a_z = \dot{w} - U_o q$$
$$h = -\frac{\dot{w} - U_0 q}{s^2}$$
$$\frac{h}{\dot{\theta}} = -\frac{1}{s^2} * \left(\frac{\dot{w}}{\dot{\theta}} - U_o \frac{q}{\dot{\theta}}\right)$$
$$\frac{h}{\dot{\theta}} = -\frac{1}{s^2} * \left(\frac{w/\delta e}{\theta/\delta e} - U_o\right)$$

Finally, we arrive at the transfer function relating the altitude (h)

$$\frac{h}{\theta} = -\frac{1}{s} * \left(\frac{w/\delta e}{\theta/\delta e} - Uo\right)$$

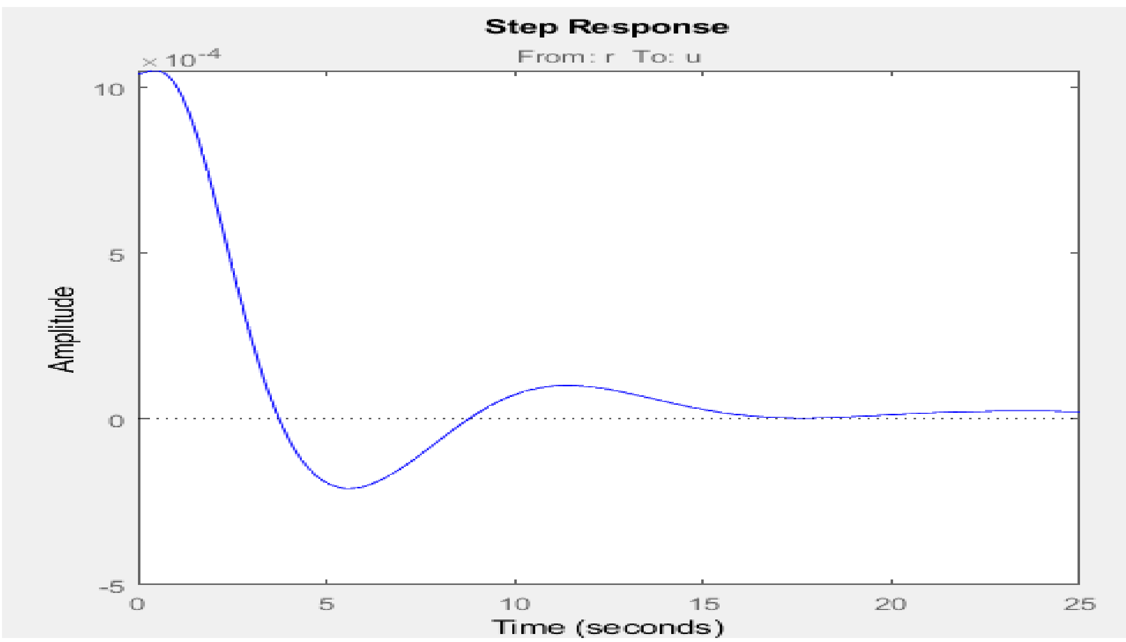
Or

$$\frac{h}{\theta} = -\frac{U_o}{s} * \left(\frac{\alpha/\delta e}{\theta/\delta e} - 1\right)$$

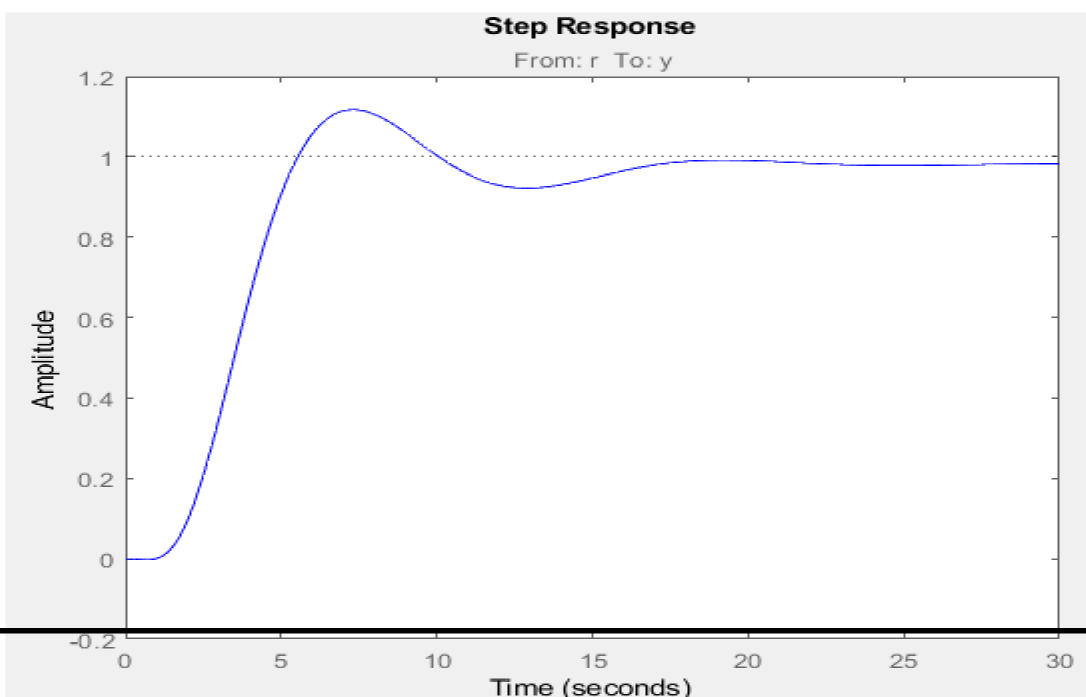
Now we can design after we reach this relation.

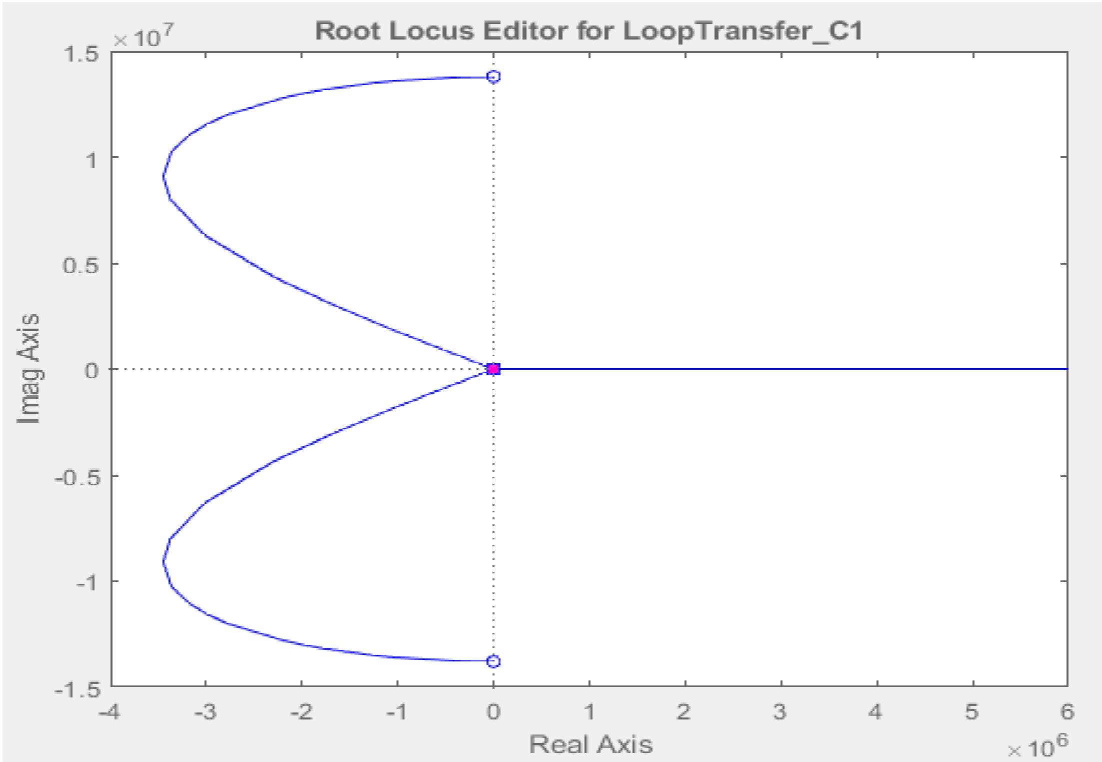
Design h controller.

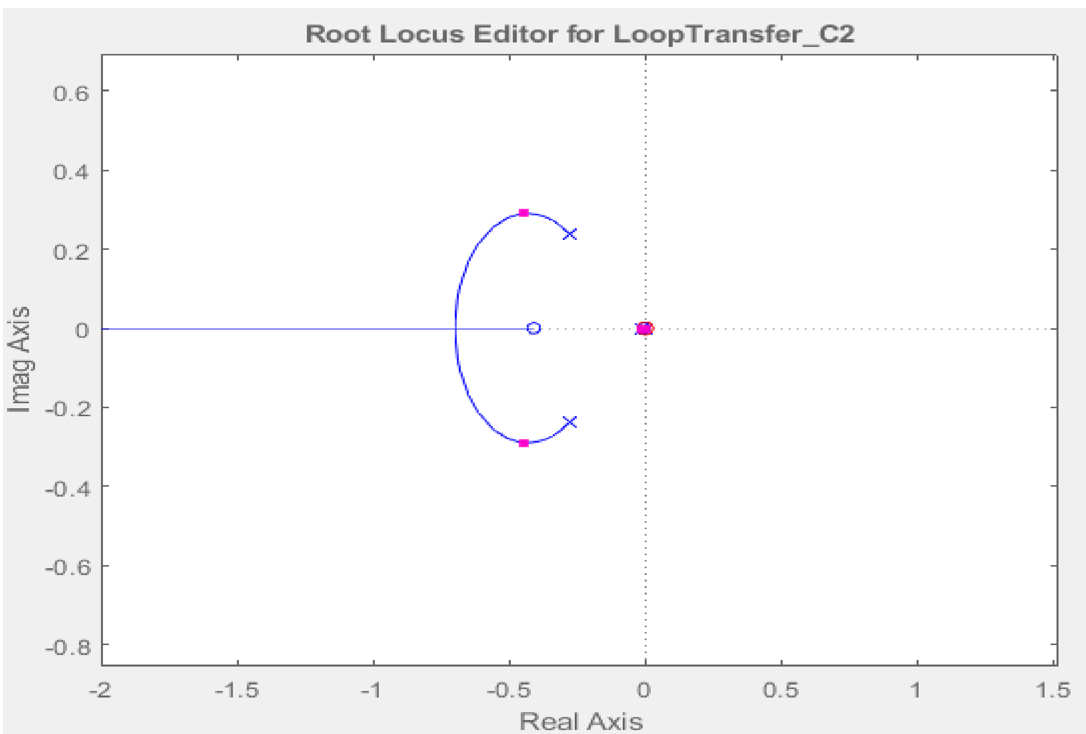
control action



Step response







It is easier than velocity control, we did tuning by changing K_d and K_i for changing damping and settling time until we reach the best result, and can't neglect the maximum of control action.

Task 7

`We discussed autopilot history and knowing aircraft equation and the assumption that we use to deal with it and to do non linear simulator for aircraft and how to linearization so we can apply control theory and what we learnt before in it and we got the approximations and what of it is valid or not valid.

And then we worked on the longitudinal and the lateral of autopilot, so we need to know if that actually work or not.

Our system is multi-input multi-output (after linearization) so we got decoupling and then we can work on the longitudinal or lateral alone and not related with each other, so we to test it to know it is valid or not.

But first make us show simply what are the differences between MIL, SIL, PIL and HIL.

MIL is model in the loop, we add the controller to aircraft model and test (simulation) and that we do, SIL is software in the loop, we can say it is a digital implementation and you need to generate a code and run the code on the model of aircraft, PIL is processor in the loop we take the controller in programming language and put it the hardware which worked on the plane and add it to the aircraft model and finally , HIL hardware in the loop we simulated aircraft hardware.

We worked on MIL (model in the loop).

The objective in this part of our project is to test our controllers “The lateral + longitudinal Autopilots” for a conventional fixed wing airplane on a more realistic model of the airplane dynamics, which is the 6DOF nonlinear airplane equations of motion.

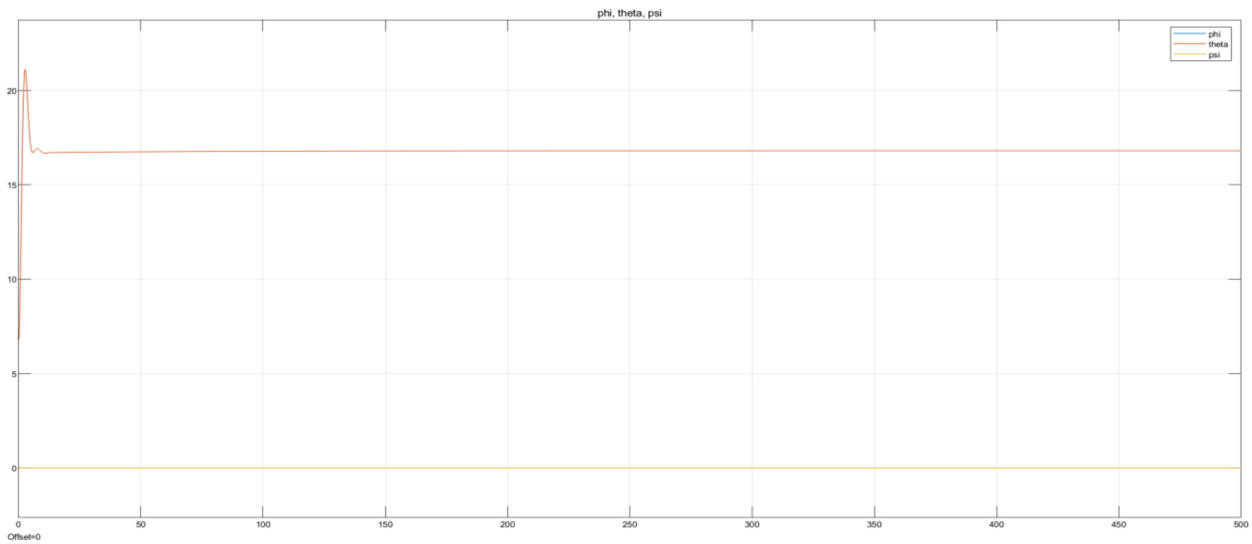
Test on nonlinear simulator.

We will test one by one to make sure every think is ok and if there any problem this way is easy to find the problem and solve it and then we do an integrated system and see the result.

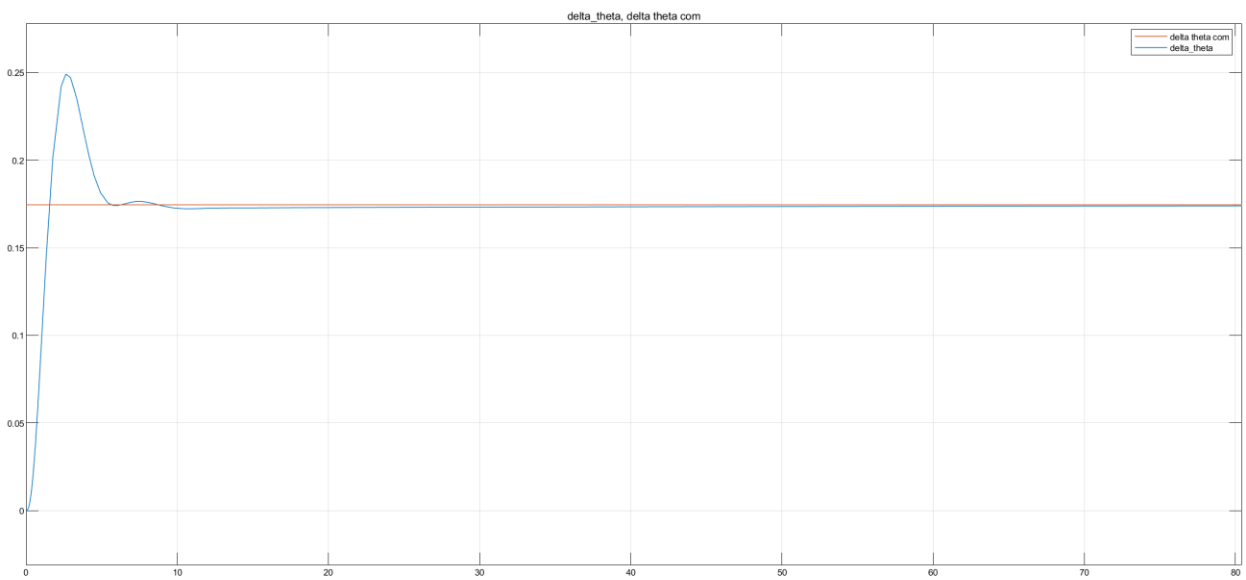
a) ***Test the "Pitch controller" and compare the response with the same test on the State space model(theta=10)***

phi_theta_psi

NL

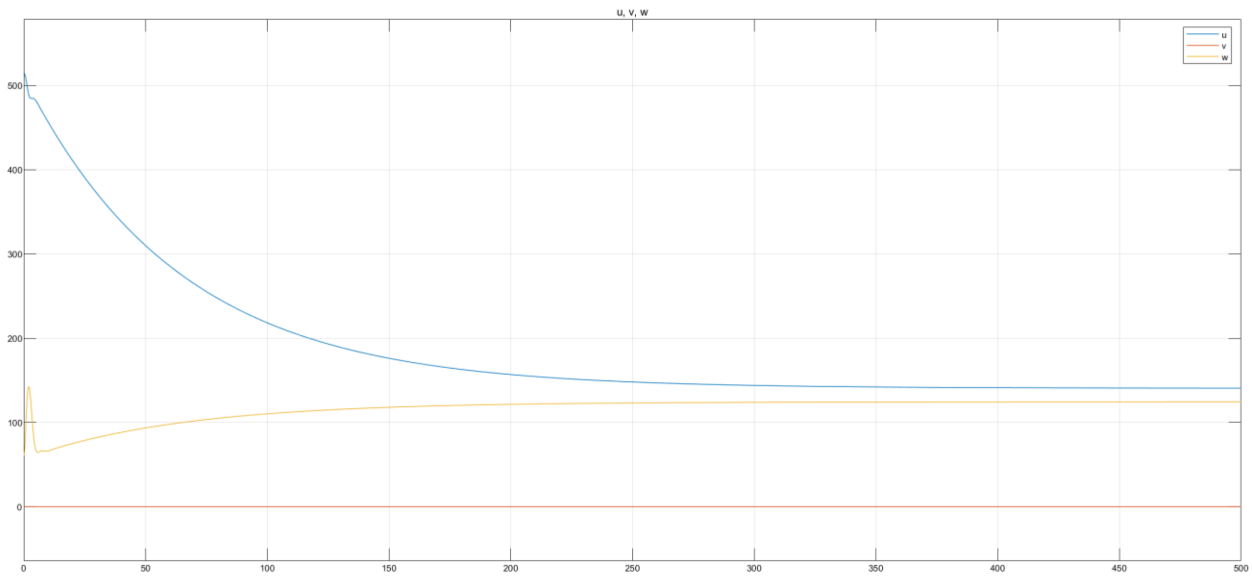


SS



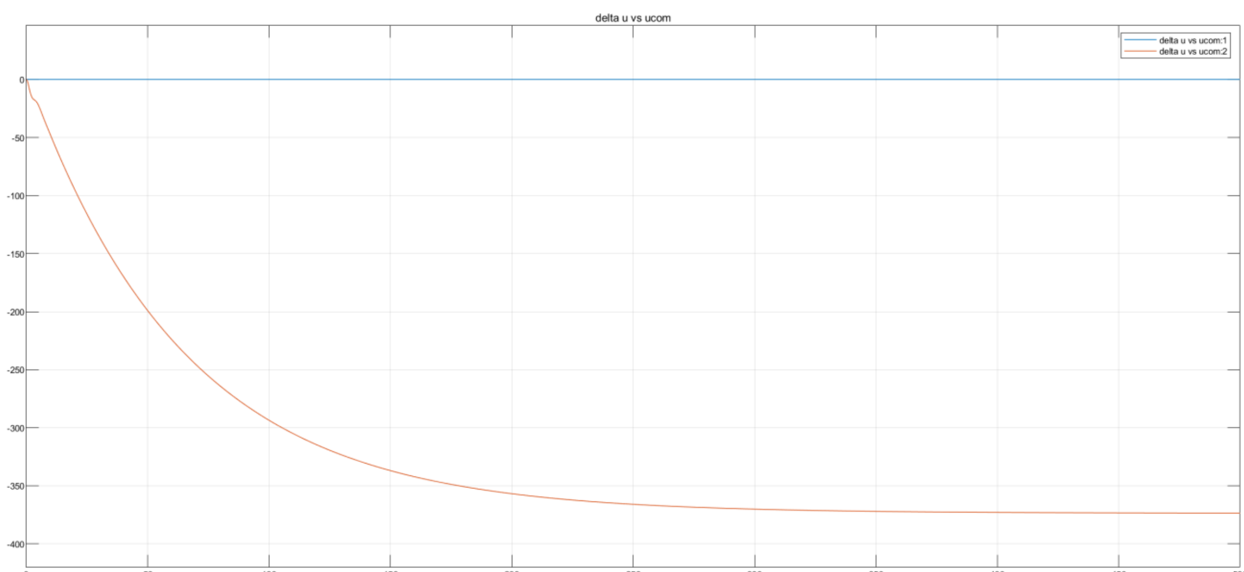
U,v,w

NL



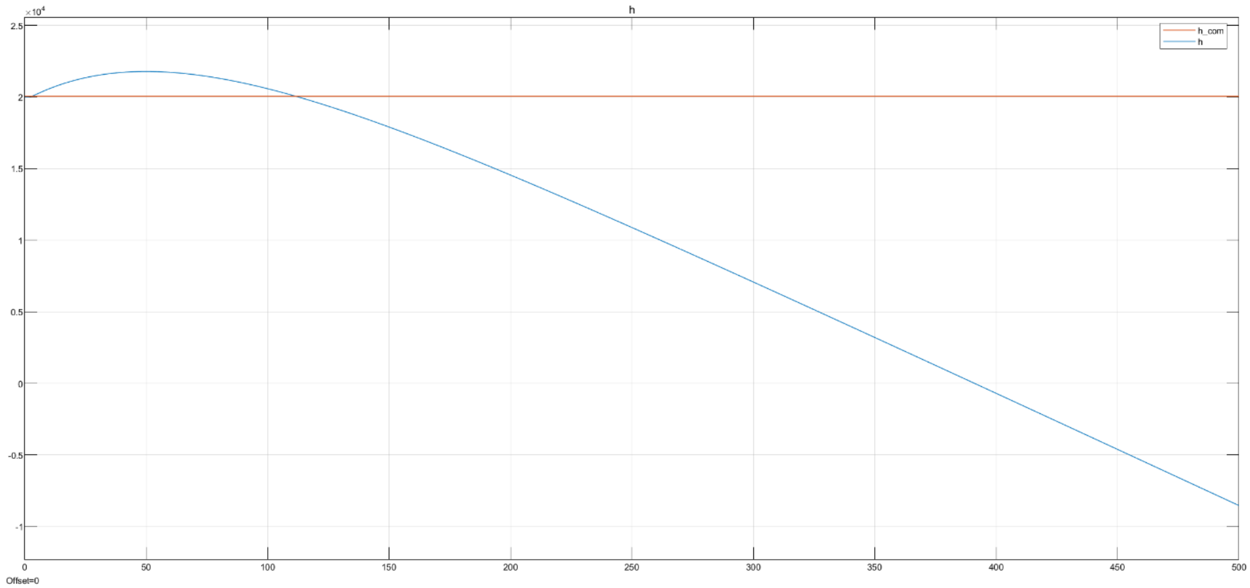
u, v, w
u
v
w
500
400
300
200
100
0
0
50
100
150
200
250
300
350
400
450
500

SS

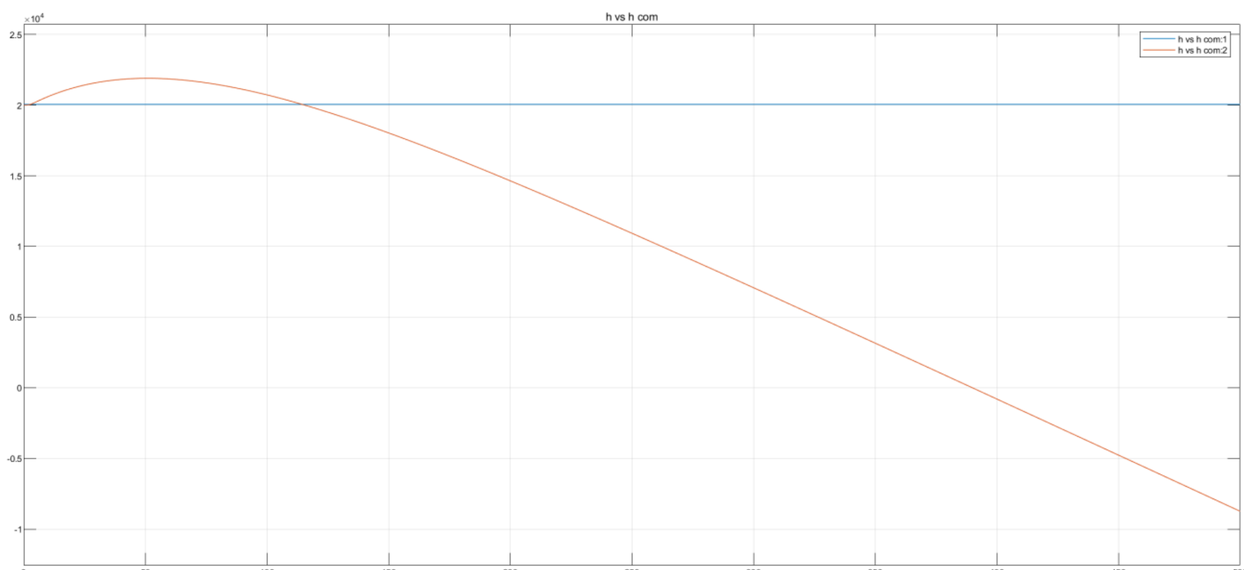


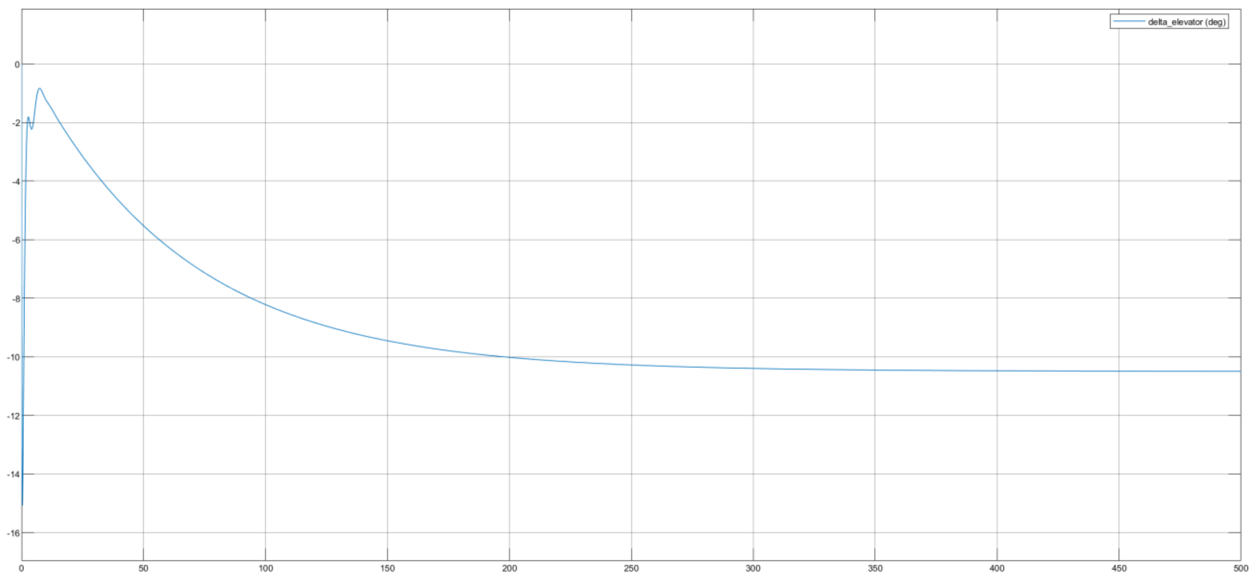
delta u vs ucom
delta u vs ucom:1
delta u vs ucom:2
0
-50
-100
-150
-200
-250
-300
-350
-400
0
50
100
150
200
250
300
350
400
450
500

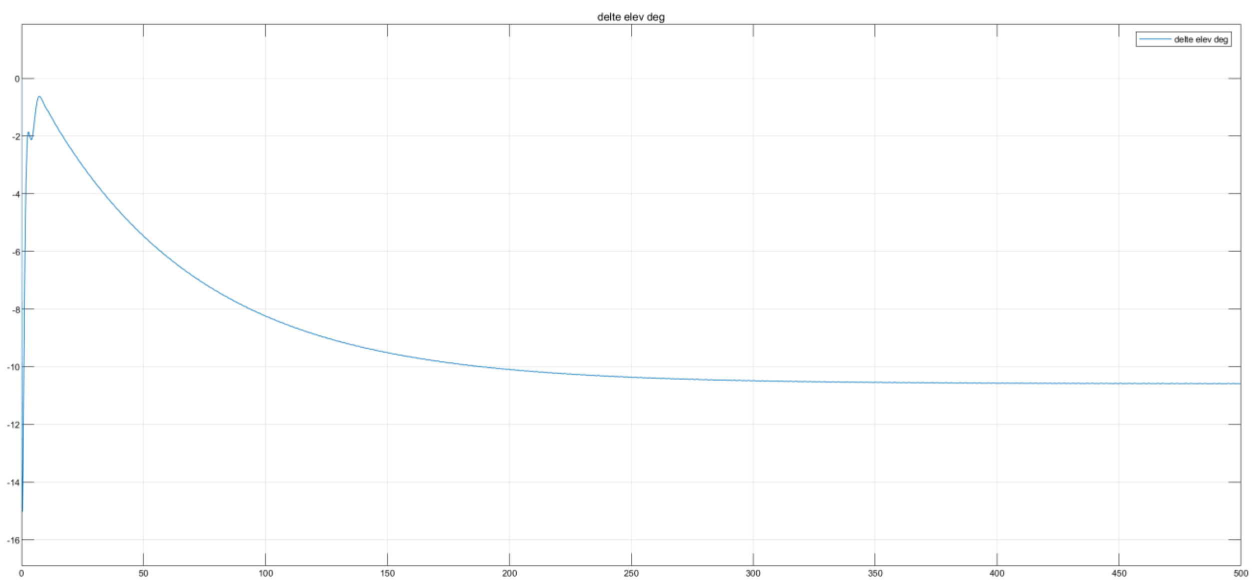
Nl

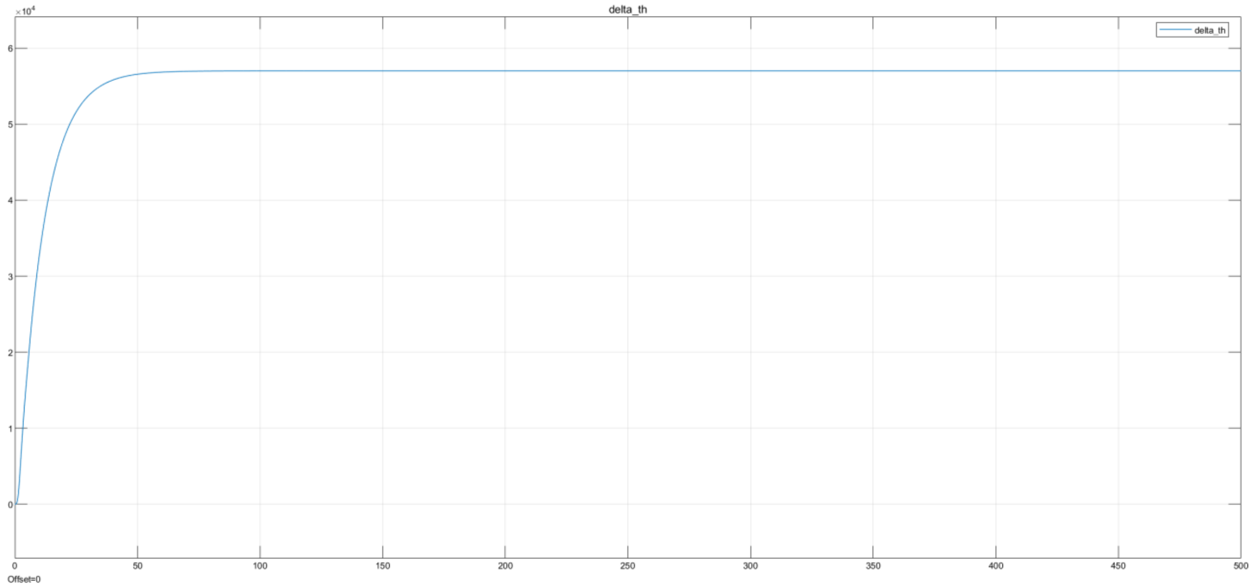

h
×10^4
h_com
h
2.5
2
1.5
1
0.5
0
-0.5
-1
0
50
100
150
200
250
300
350
400
450
500
Offset=0

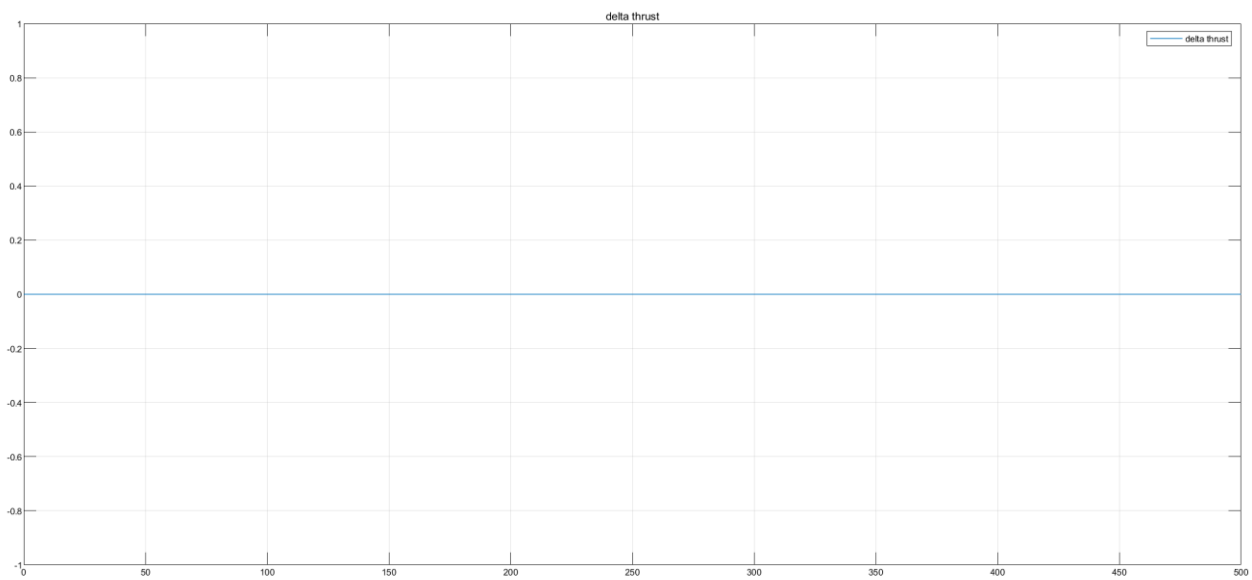
SS

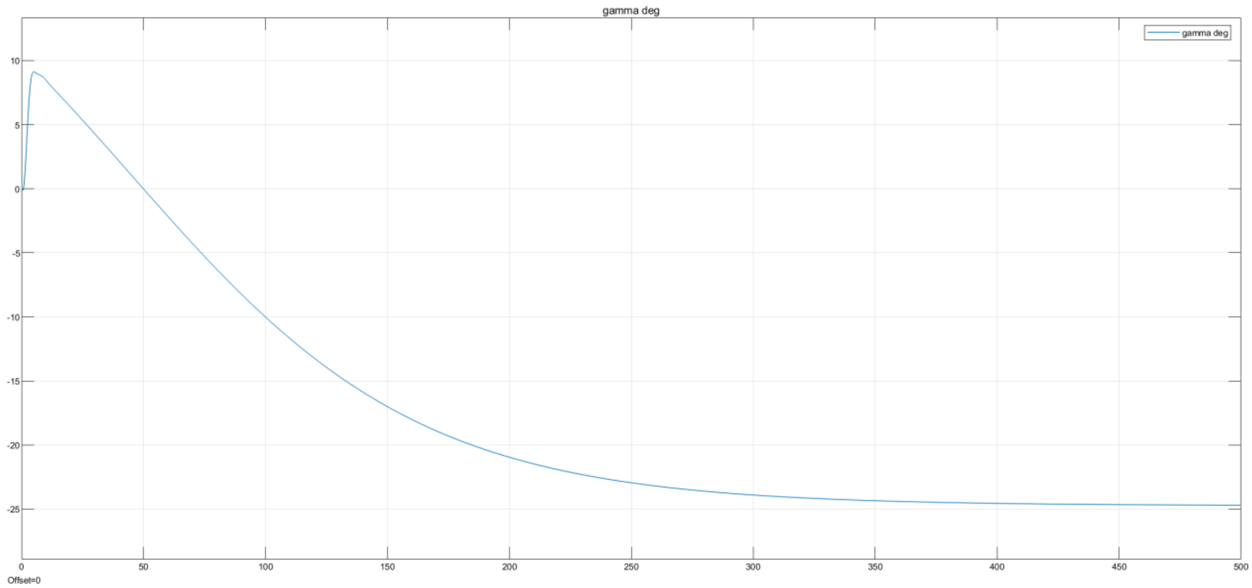

h vs h com
×10^4
h vs h com:1
h vs h com:2
2.5
2
1.5
1
0.5
0
-0.5
-1
0
50
100
150
200
250
300
350
400
450
500

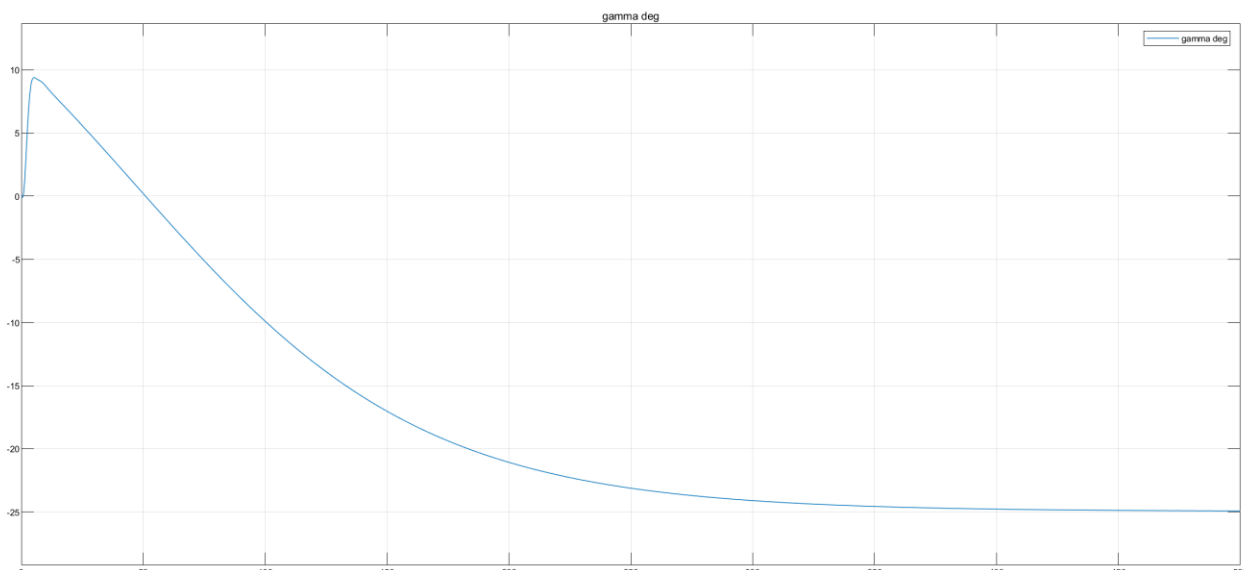

delta_elevator (deg)


delte elev deg
delte elev deg

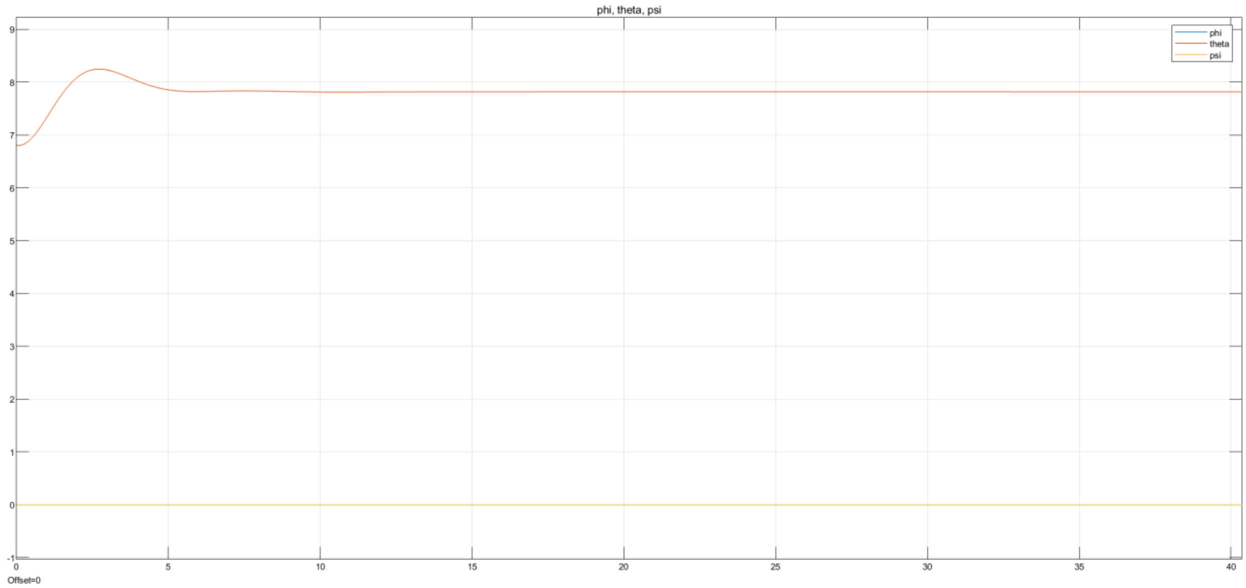

delta_th
×10^4
delta_th
Offset=0

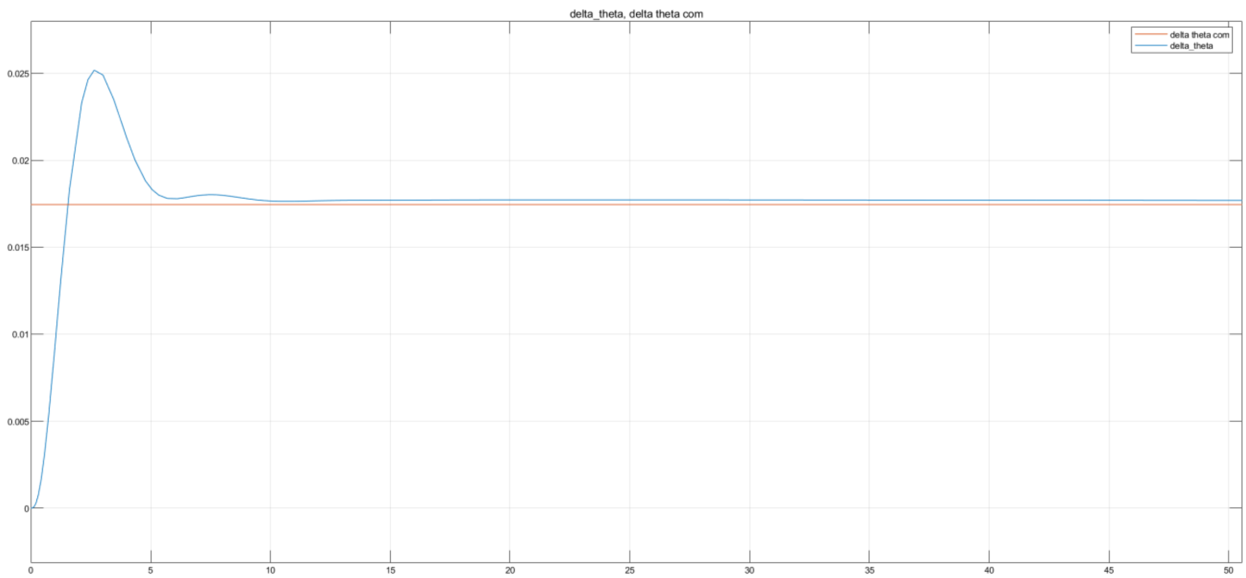

delta thrust
delta thrust

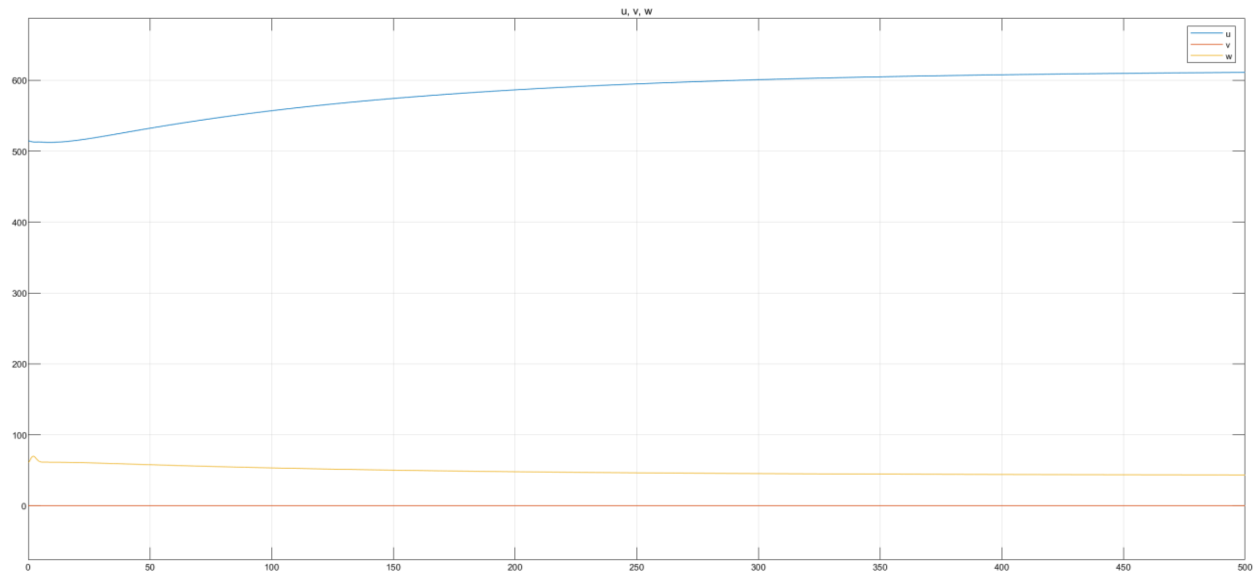

gamma deg
gamma deg
10
5
0
-5
-10
-15
-20
-25
0
50
100
150
200
250
300
350
400
450
500
Offset=0

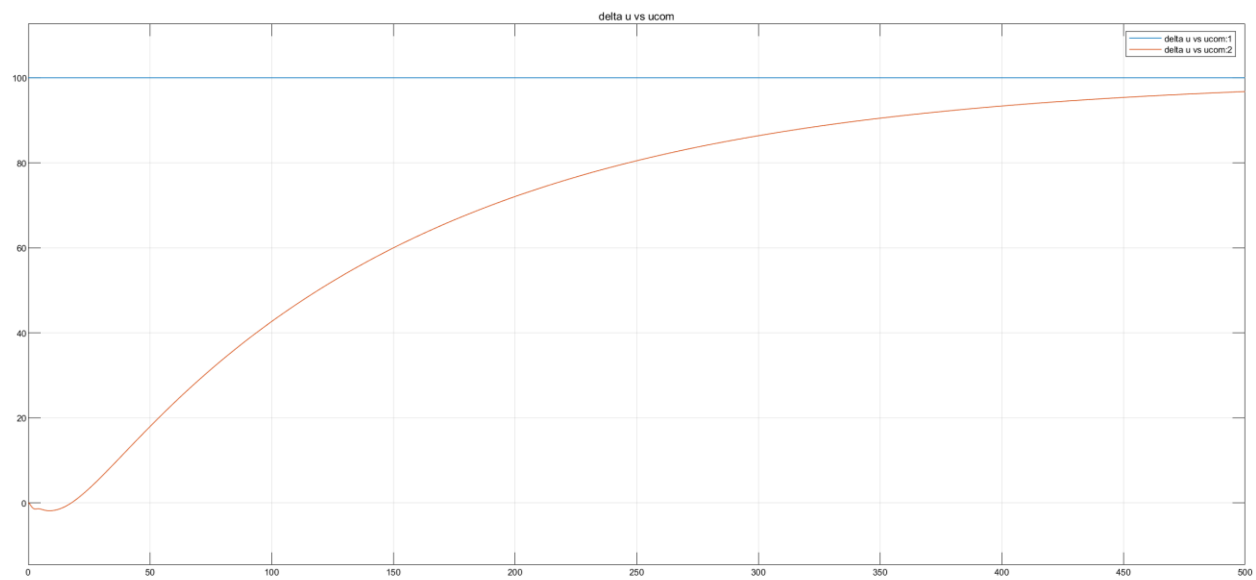

gamma deg
gamma deg
10
5
0
-5
-10
-15
-20
-25
0
50
100
150
200
250
300
350
400
450
500

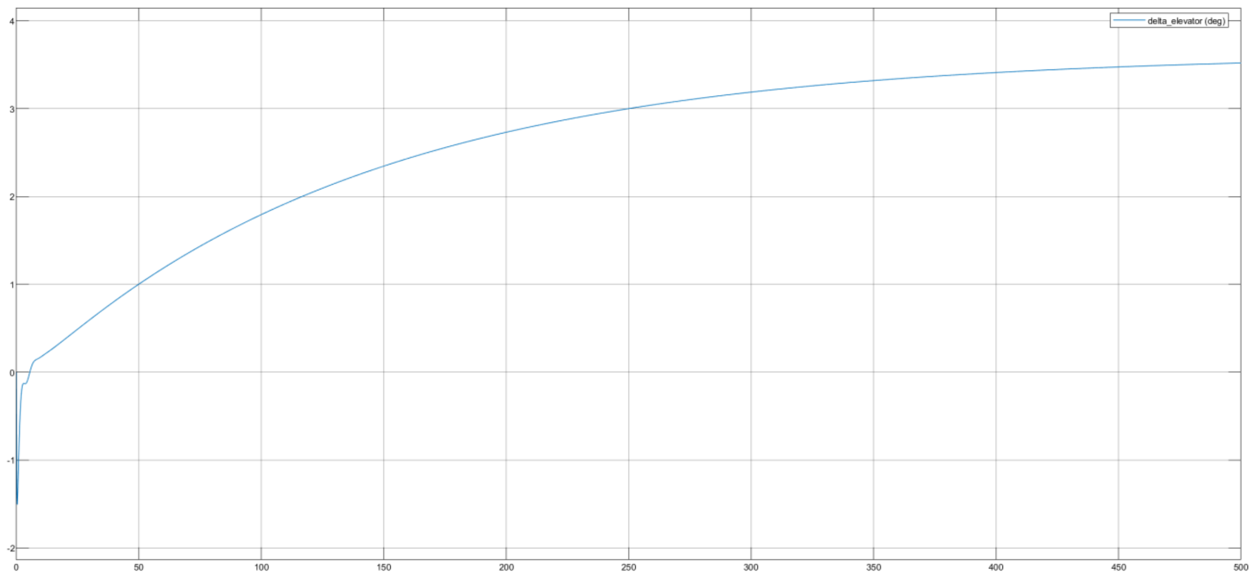
B) ***Test the "Pitch controller + Velocity controller" and compare the response with the same test on the State space model(theta 1 u100)***

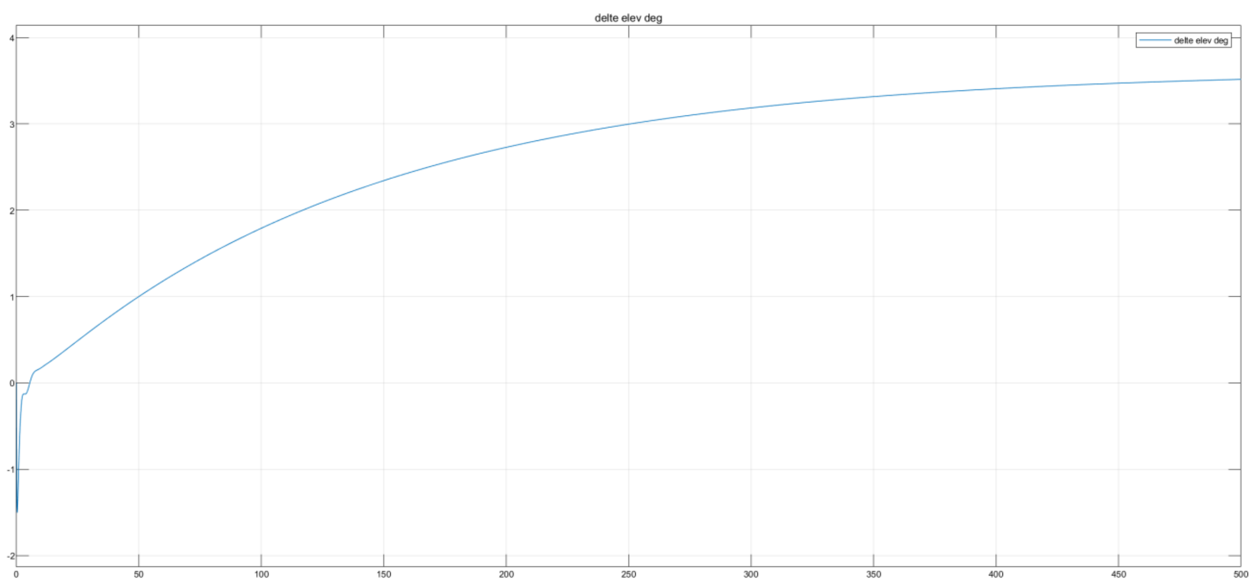




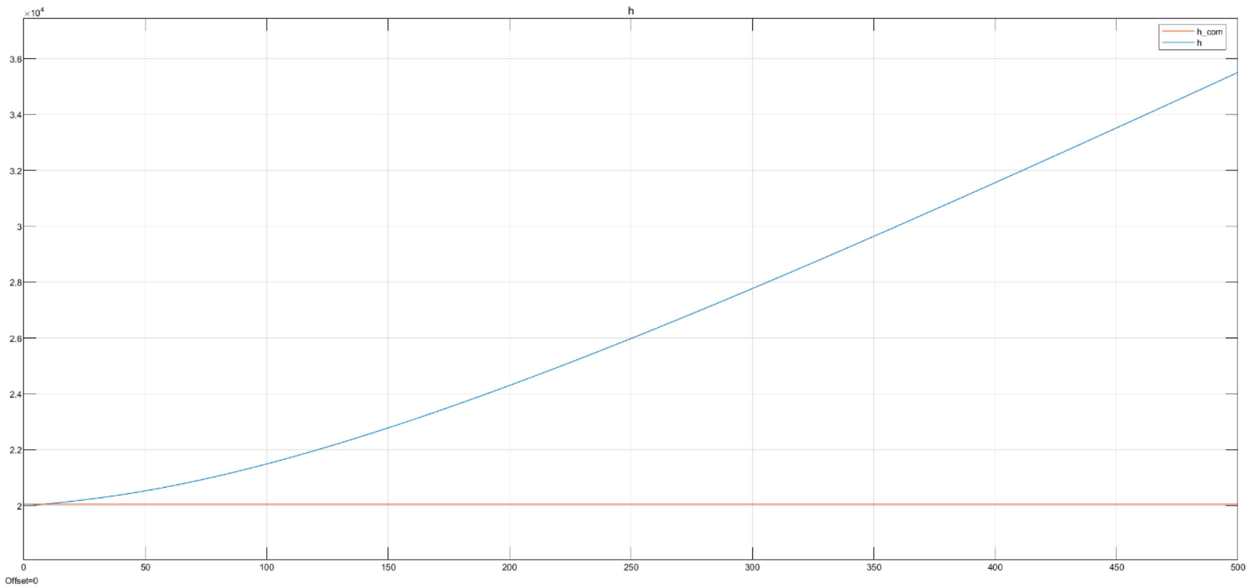

u, v, w
u
v
w
600
500
400
300
200
100
0
0
50
100
150
200
250
300
350
400
450
500


delta u vs ucom
delta u vs ucom:1
delta u vs ucom:2
100
80
60
40
20
0
0
50
100
150
200
250
300
350
400
450
500

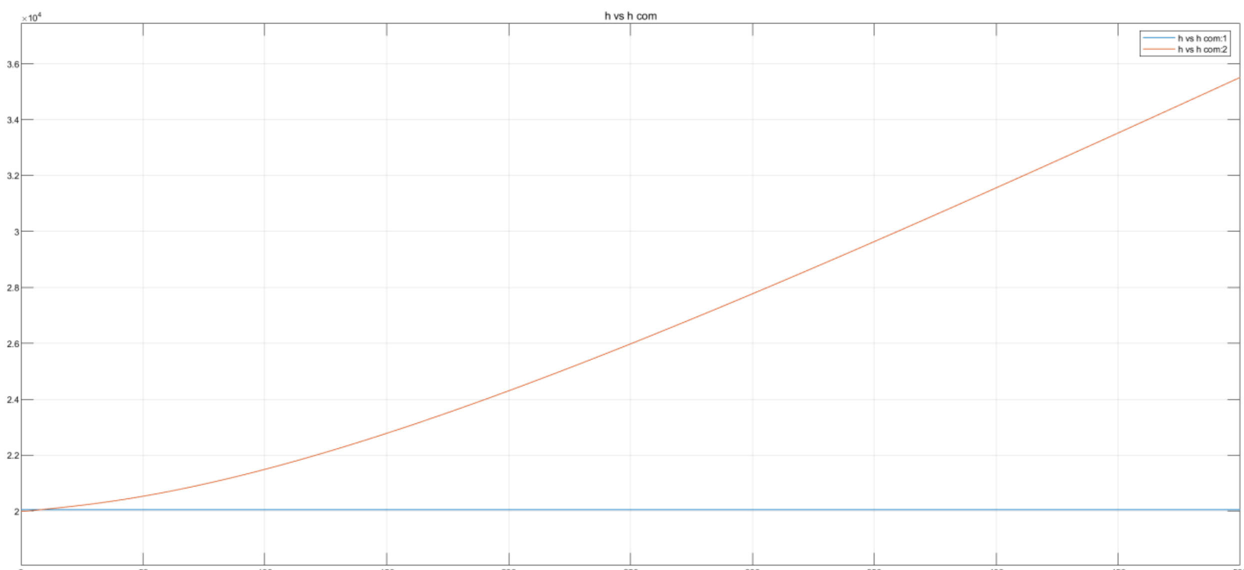

delta_elevator (deg)
4
3
2
1
0
-1
-2
0
50
100
150
200
250
300
350
400
450
500


delte elev deg
delte elev deg
4
3
2
1
0
-1
-2
0
50
100
150
200
250
300
350
400
450
500

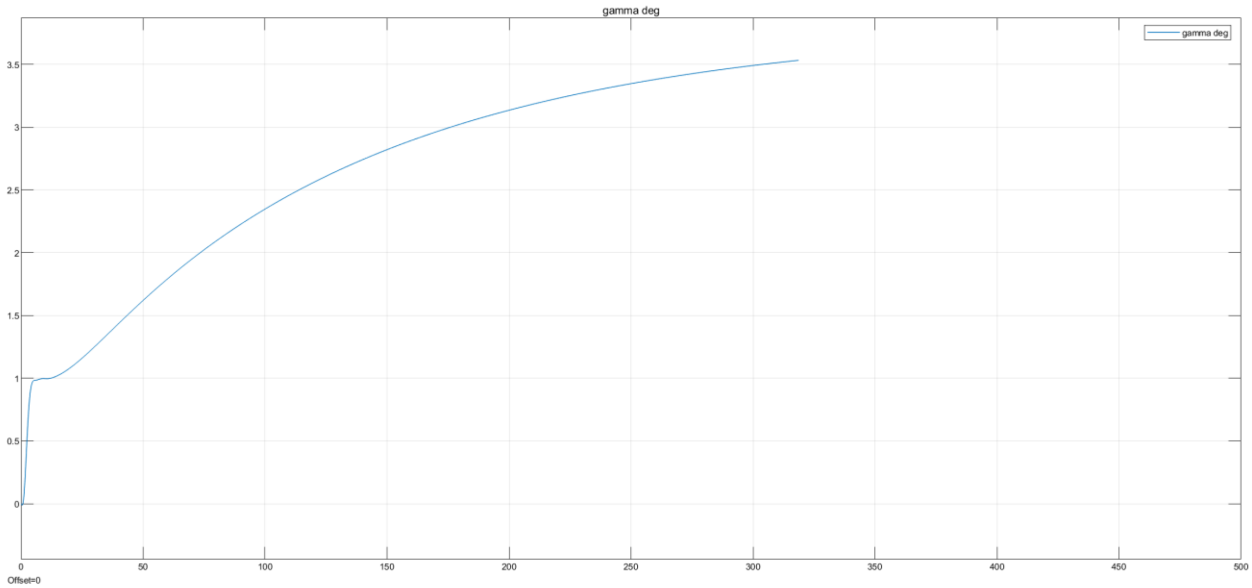
n

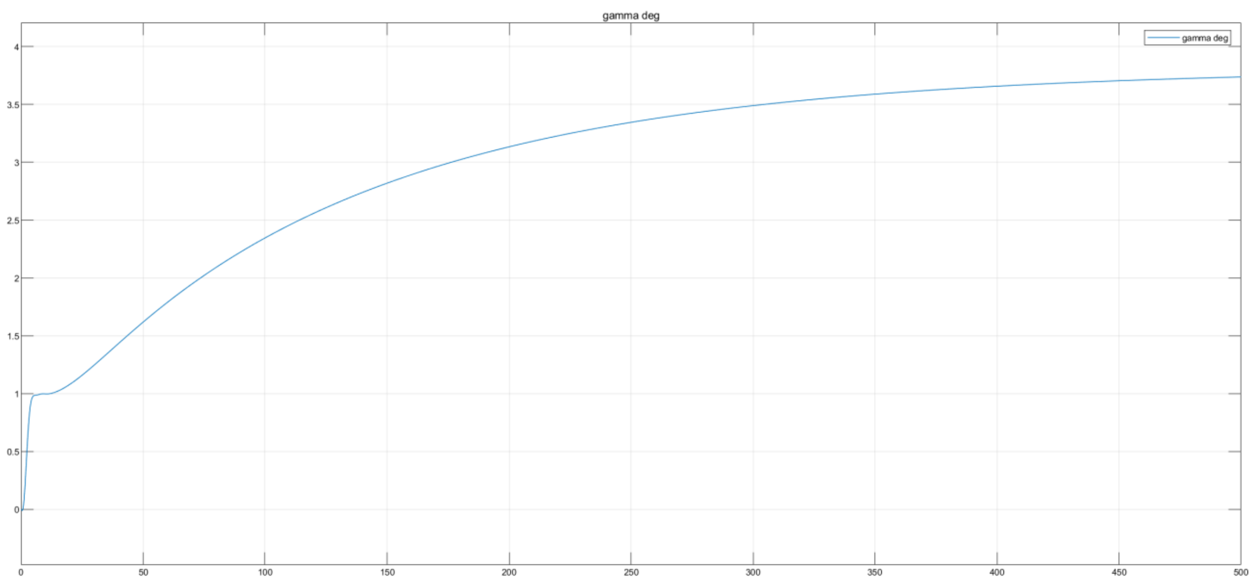


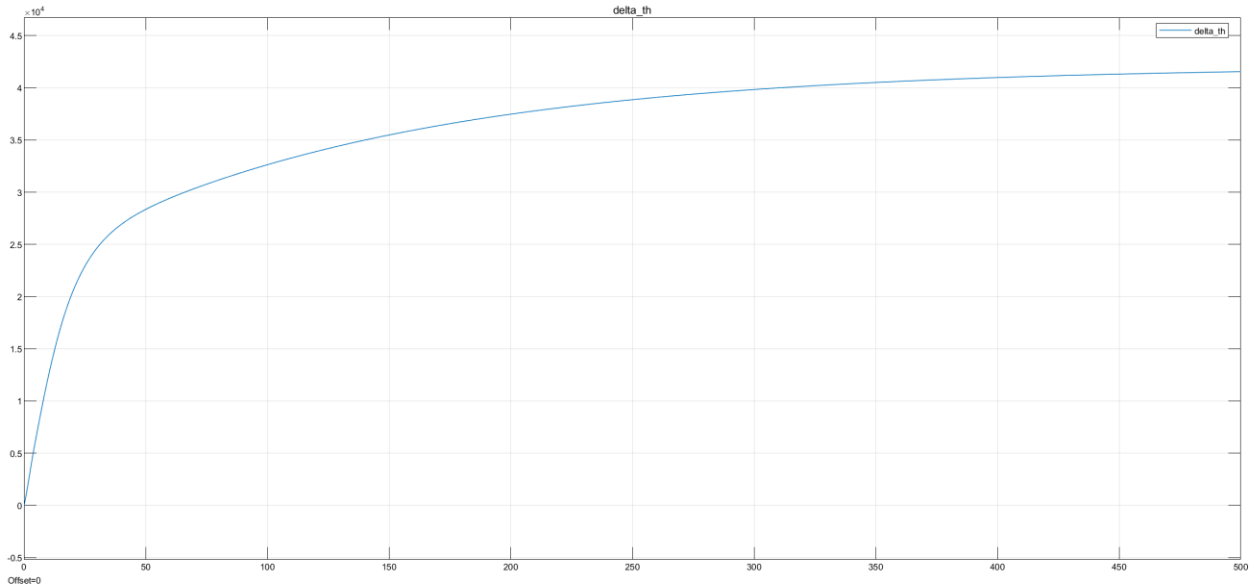
h
×10^4
h_com
h
3.6
3.4
3.2
3
2.8
2.6
2.4
2.2
2
0
50
100
150
200
250
300
350
400
450
500
Offset=0

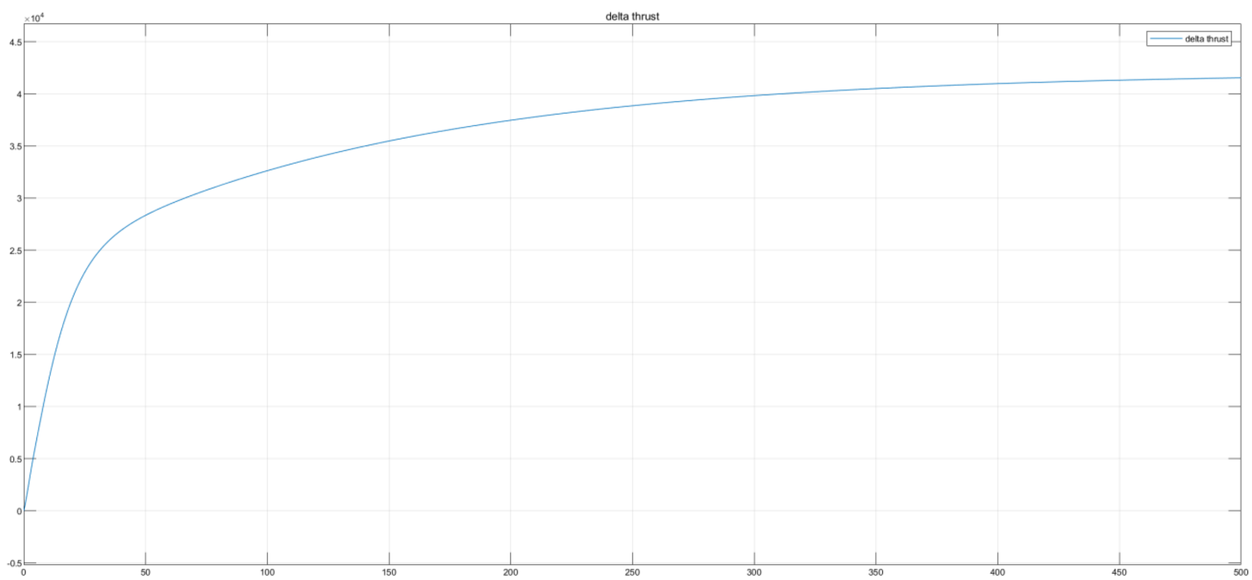


h vs h com
×10^4
h vs h com:1
h vs h com:2
3.6
3.4
3.2
3
2.8
2.6
2.4
2.2
2
0
50
100
150
200
250
300
350
400
450
500

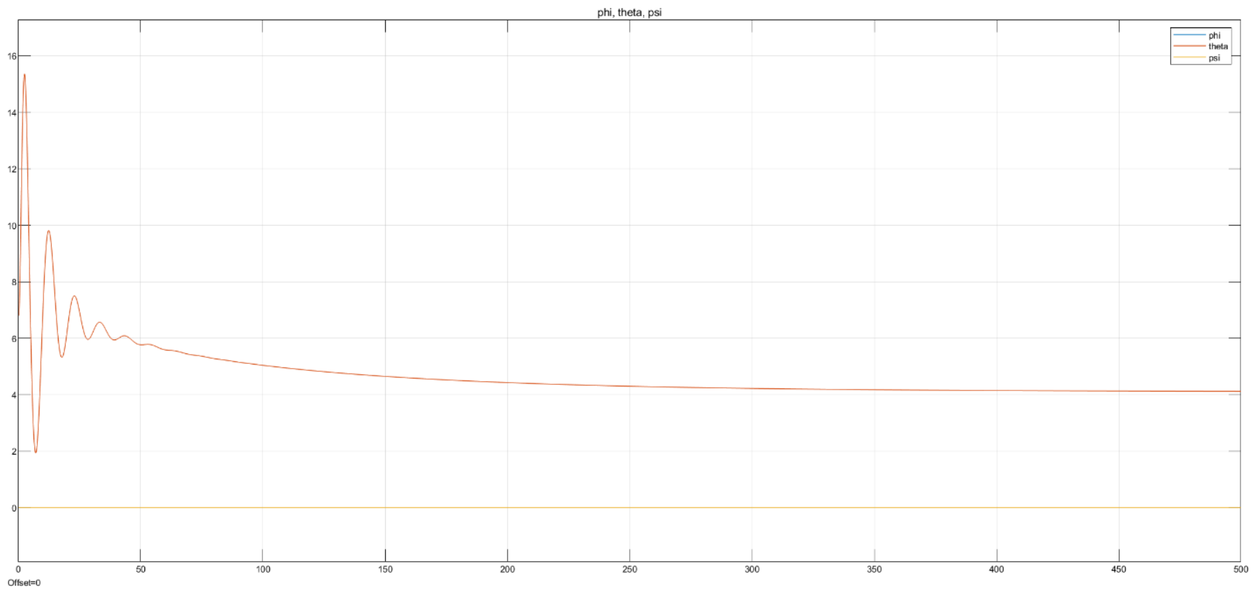

gamma deg
gamma deg
Offset=0

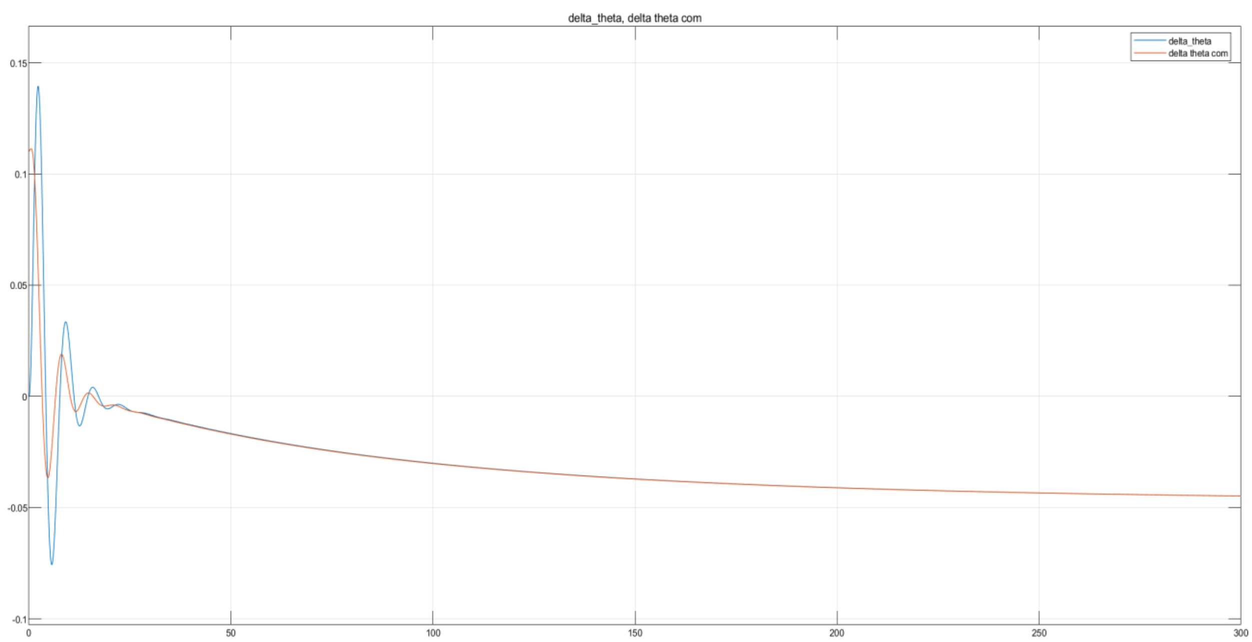

gamma deg
gamma deg


delta_th
×10^4
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
0
50
100
150
200
250
300
350
400
450
500
Offset=0
delta_th

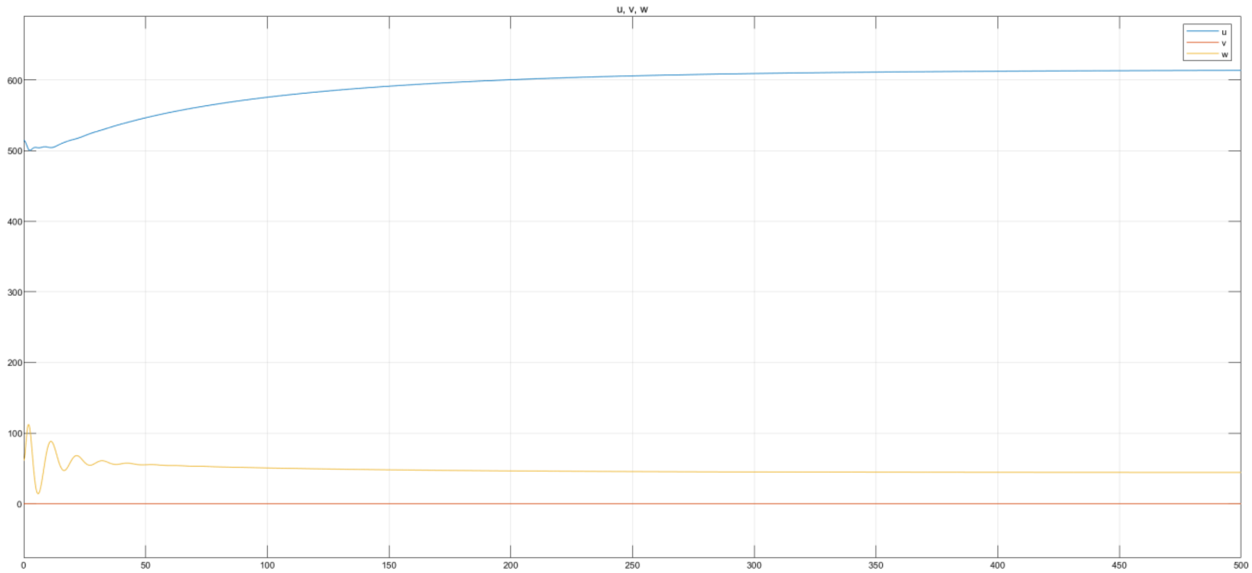

delta thrust
×10^4
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
0
50
100
150
200
250
300
350
400
450
500
delta thrust

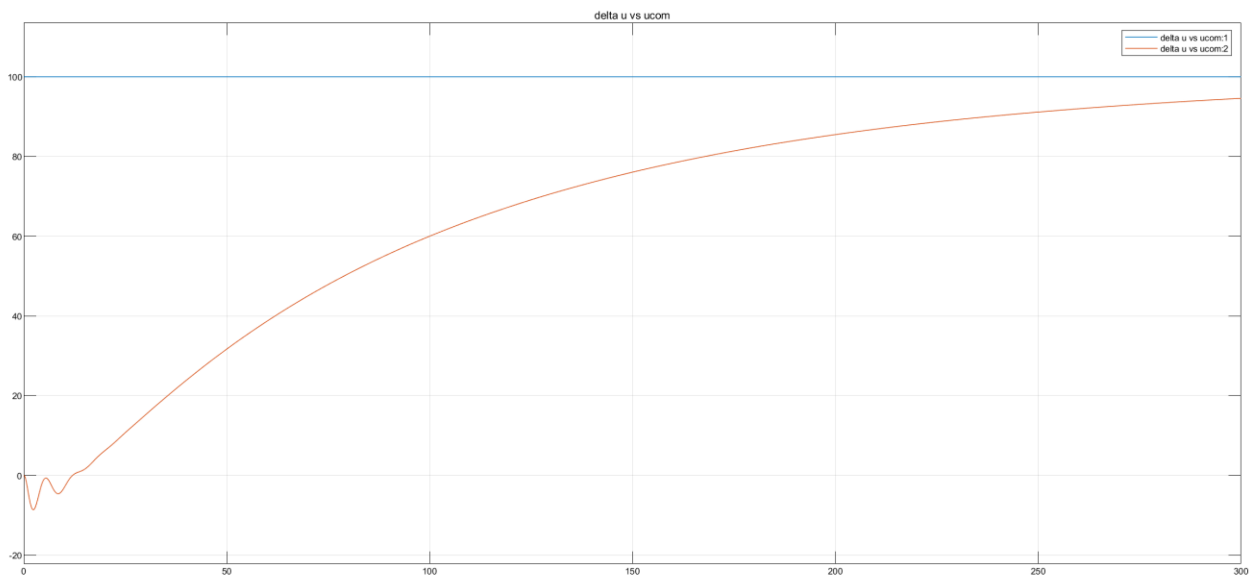
c) ***Test the "Altitude Hold" controller and compare the response with the same test on the State space mode(u=100,h2100)***

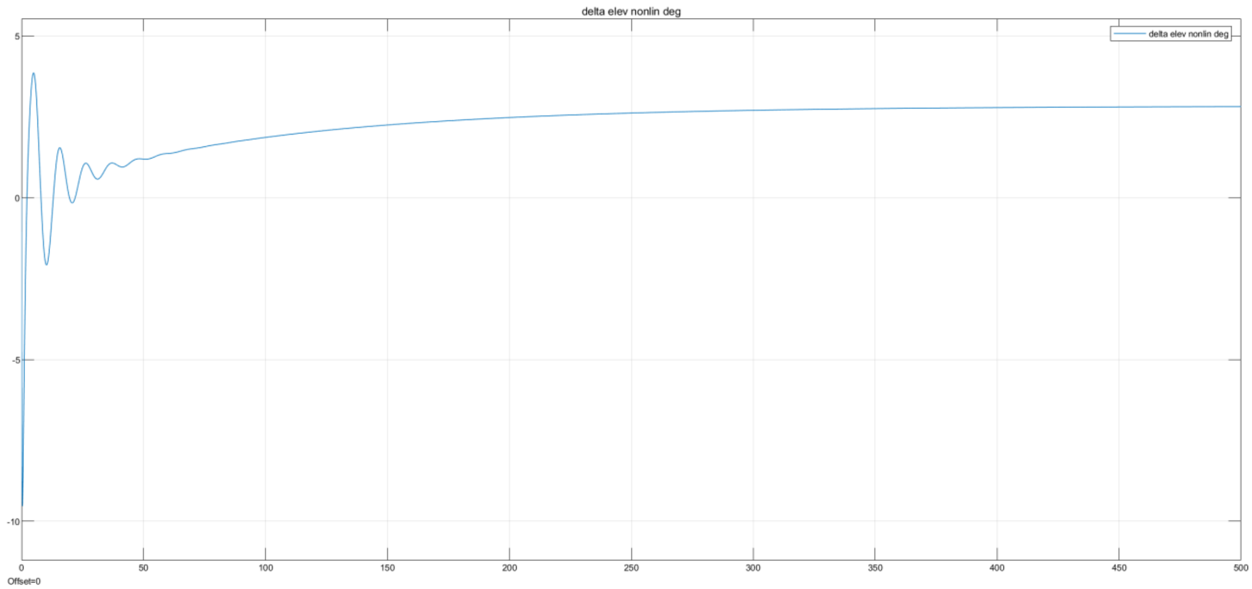


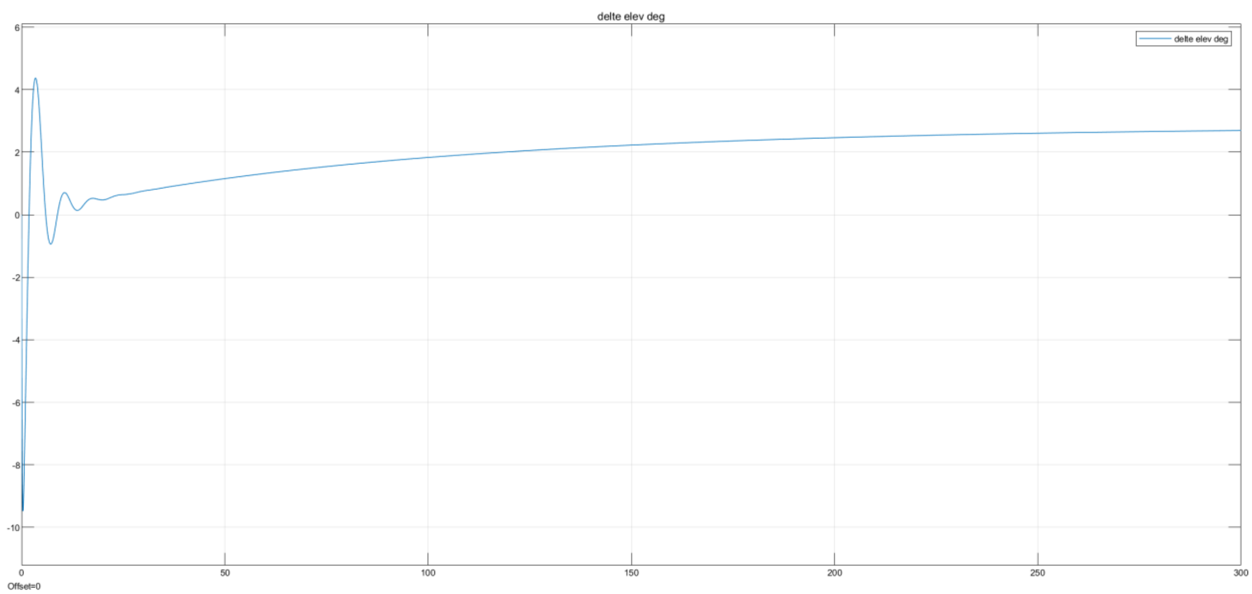


,

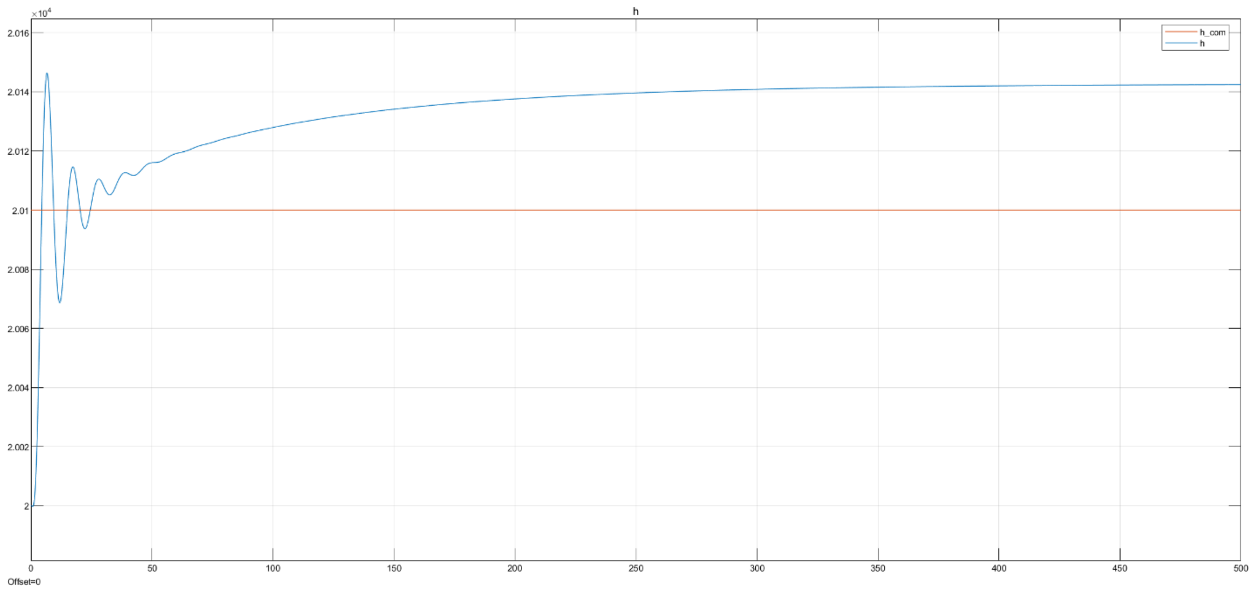

u, v, w
u
v
w
600
500
400
300
200
100
0
0
50
100
150
200
250
300
350
400
450
500


delta u vs ucom
delta u vs ucom:1
delta u vs ucom:2
100
80
60
40
20
0
-20
0
50
100
150
200
250
300
Offset=0

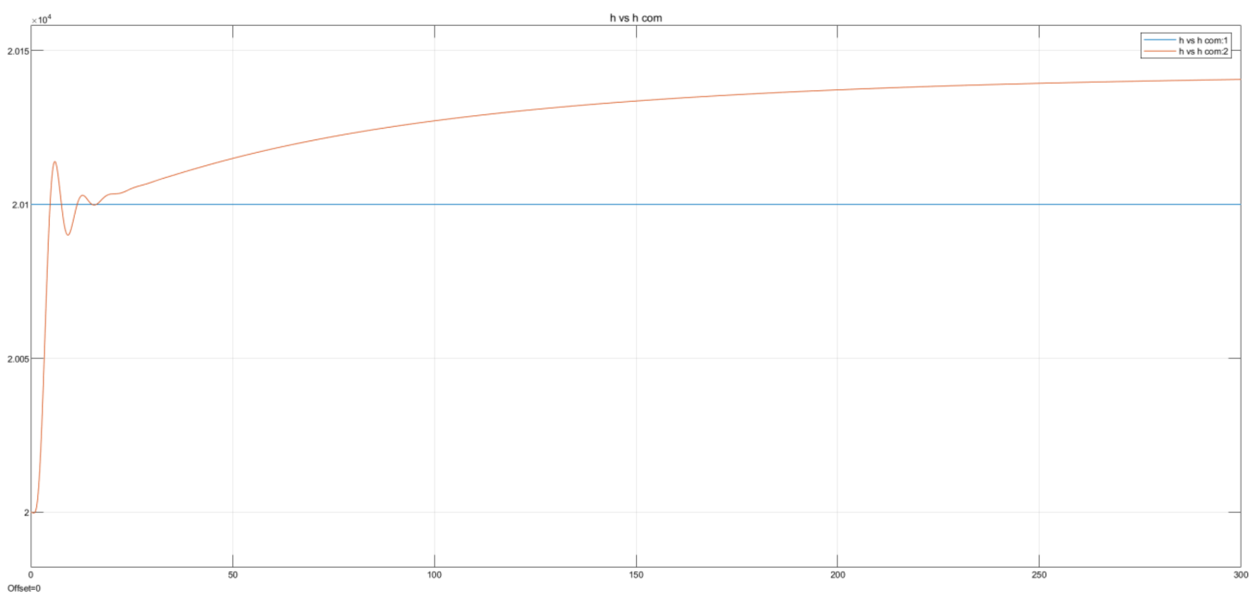

delta elev nonlin deg
delta elev nonlin deg
5
0
-5
-10
0
50
100
150
200
250
300
350
400
450
500
Offset=0


delte elev deg
delte elev deg
6
4
2
0
-2
-4
-6
-8
-10
0
50
100
150
200
250
300
Offset=0

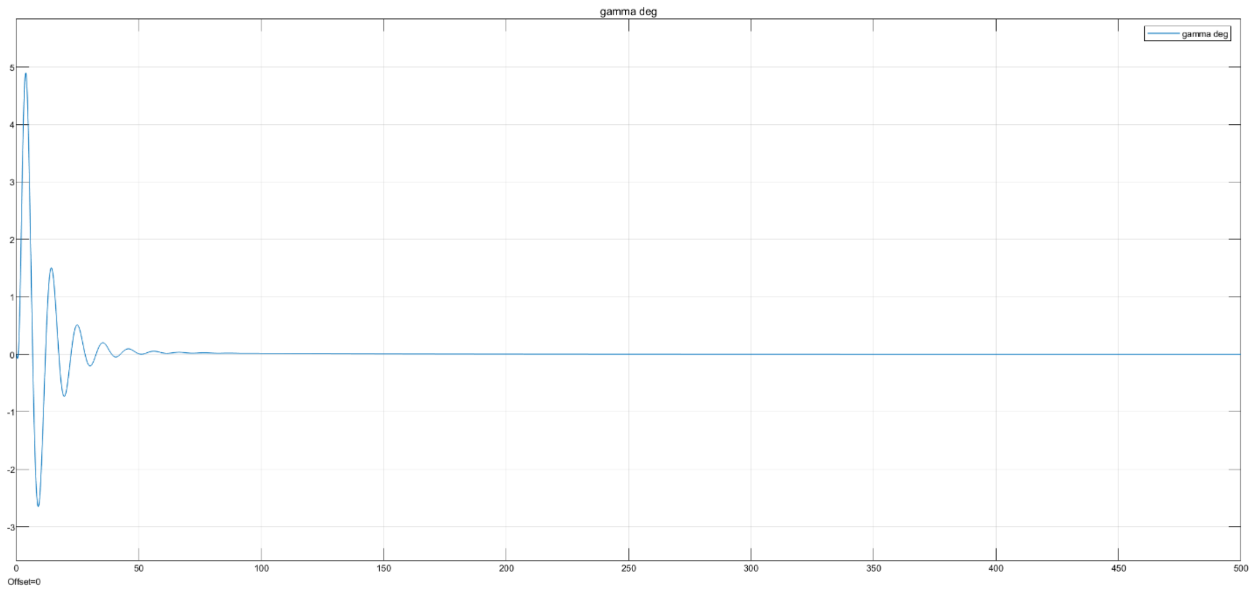
NL


h
×10^4
h_com
h
2.016
2.014
2.012
2.01
2.008
2.006
2.004
2.002
2
0
50
100
150
200
250
300
350
400
450
500
Offset=0

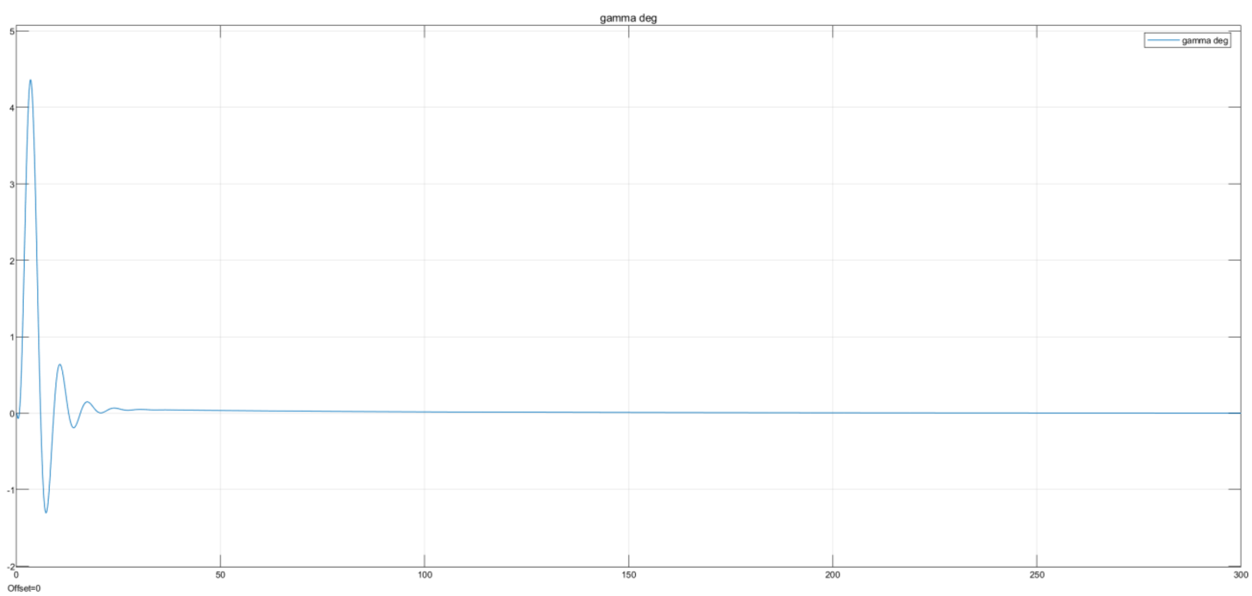
SS


h vs h com
×10^4
h vs h com:1
h vs h com:2
2.015
2.01
2.005
2
0
50
100
150
200
250
300
Offset=0

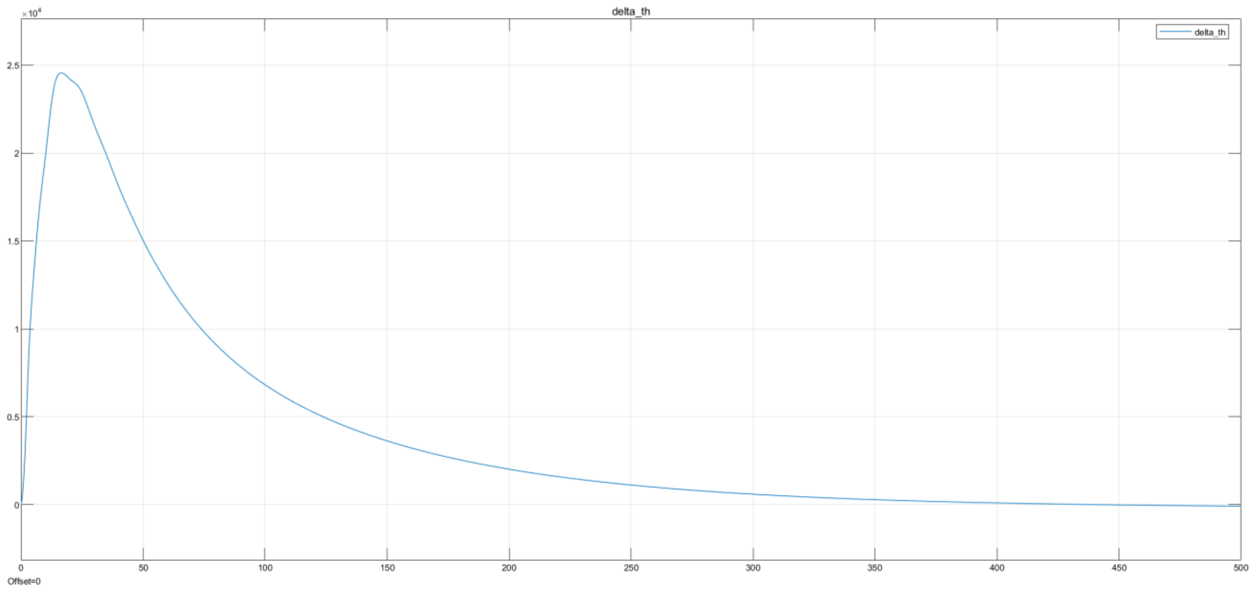
NL


gamma deg
gamma deg
5
4
3
2
1
0
-1
-2
-3
0
50
100
150
200
250
300
350
400
450
500
Offset=0

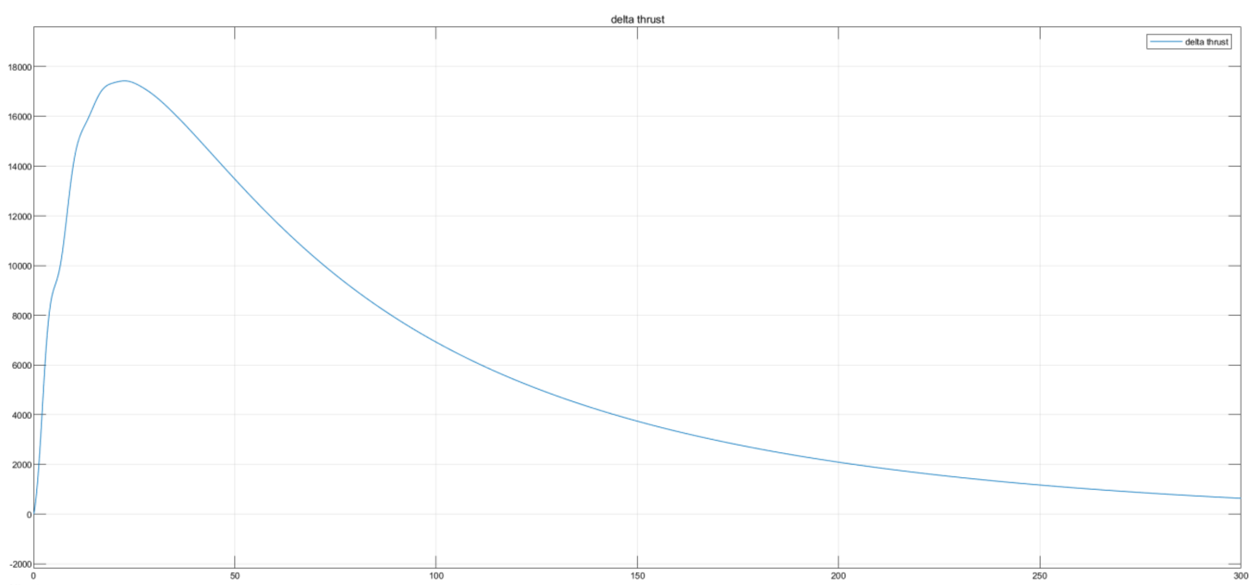
SS


gamma deg
gamma deg
5
4
3
2
1
0
-1
-2
0
50
100
150
200
250
300
Offset=0

NL


delta_th
×10^4
delta_th
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250
300
350
400
450
500
Offset=0

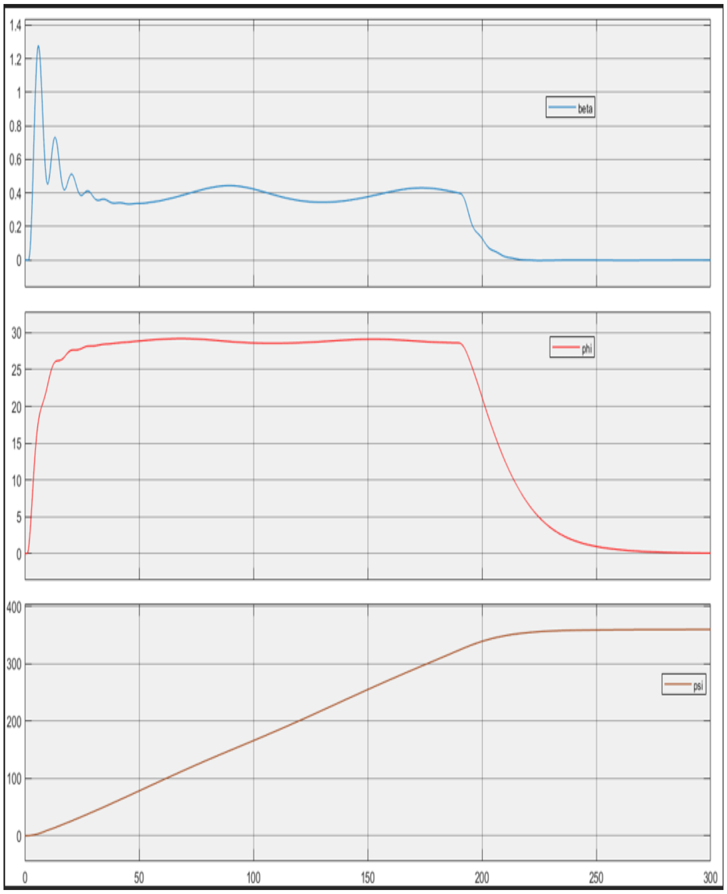
SS


delta thrust
delta thrust
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
-2000
0
50
100
150
200
250
300
Offset=0

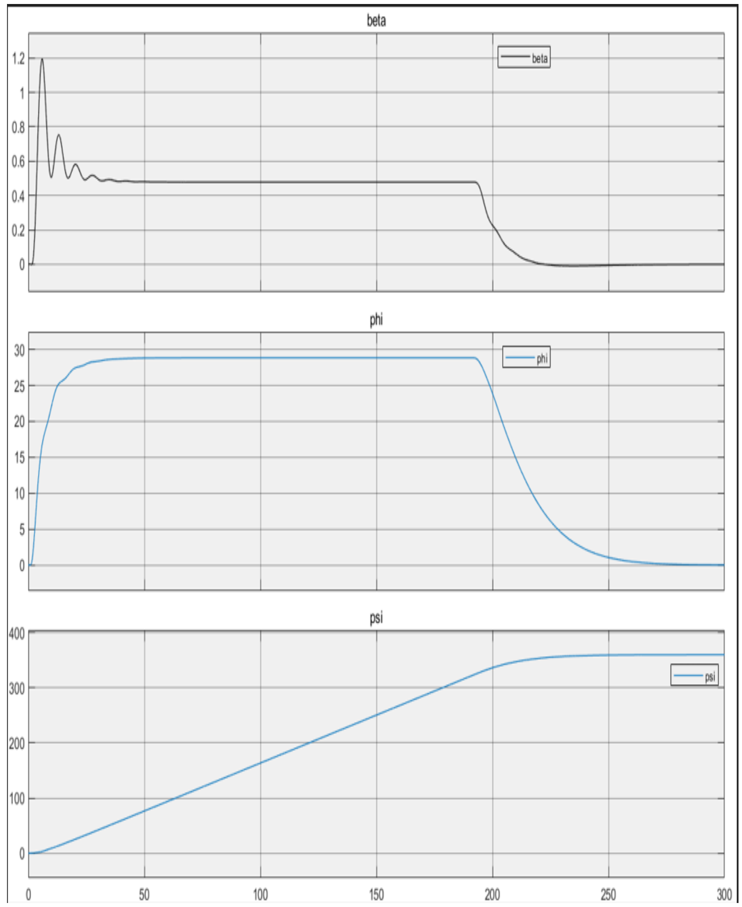
now we will test the lateral :

d) est β , ϕ, ψ

Nonlinear model



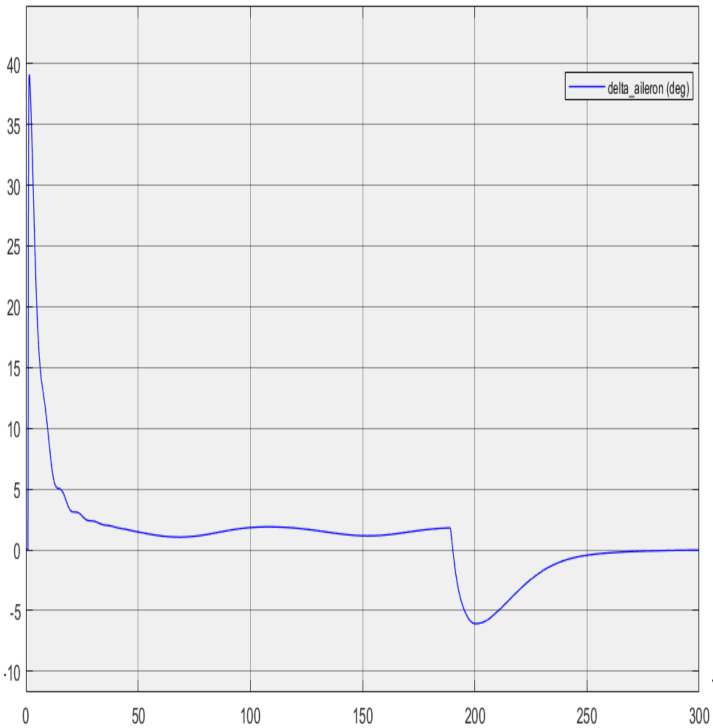
state space model



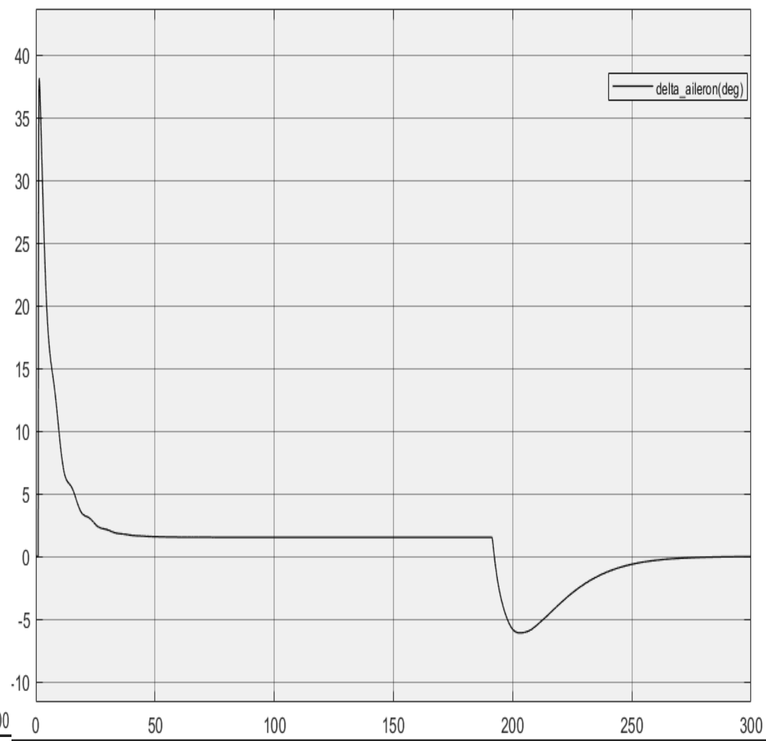
As we see we can say we have the same result

Test delta-aileron:

Nonlinear model



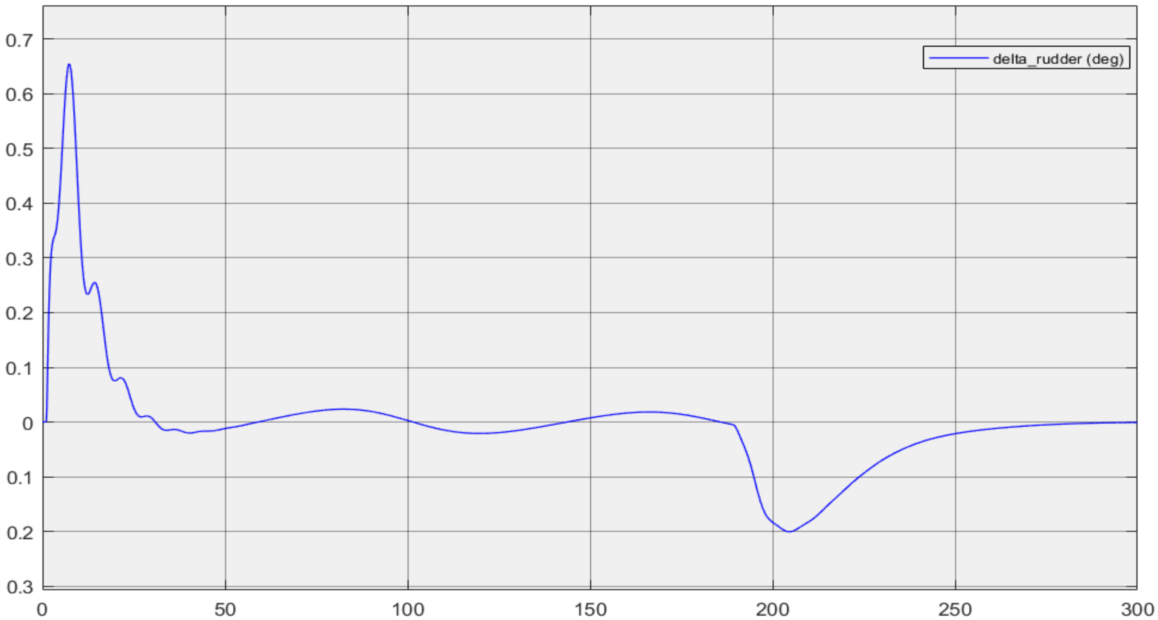
state space model

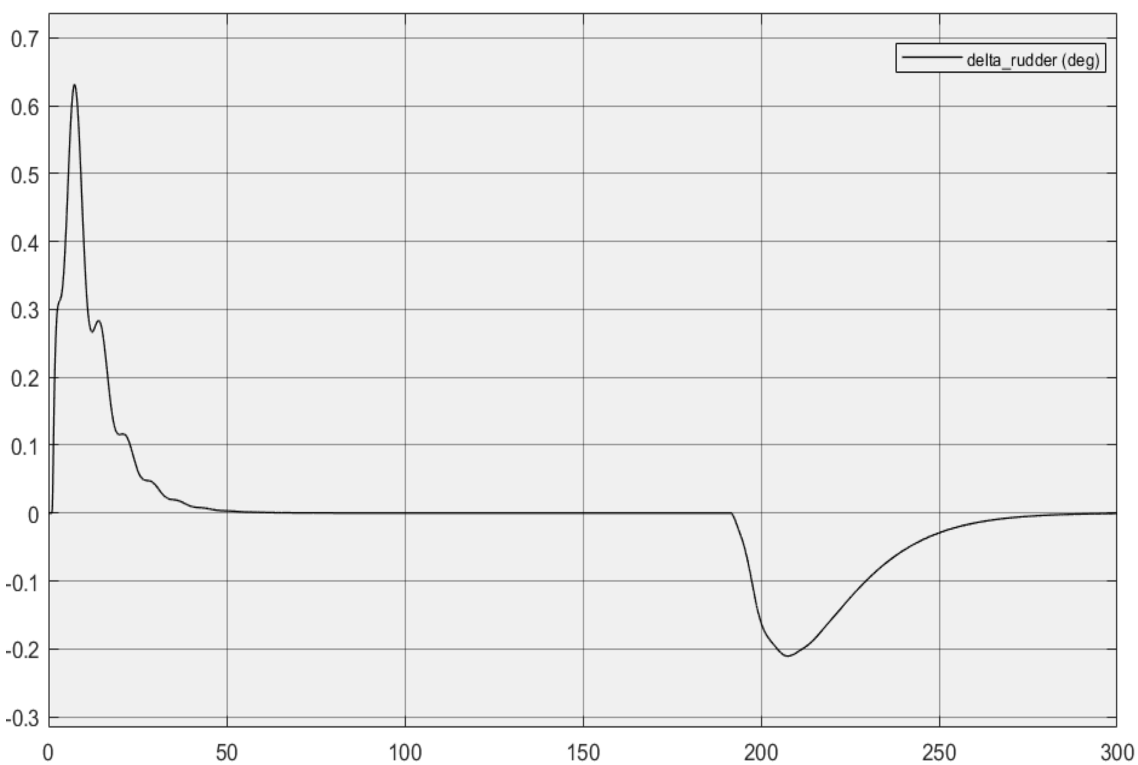


Same Result

Test delta-Rudder:

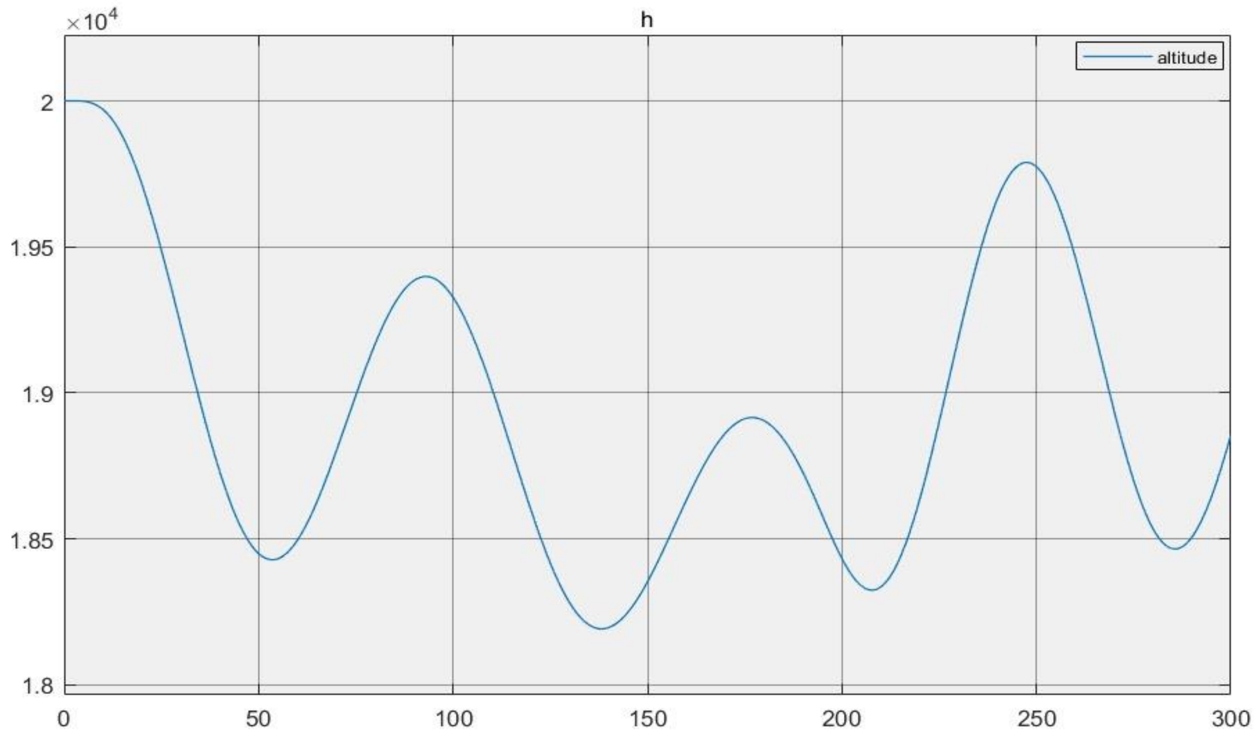
Nonlinear model state space model



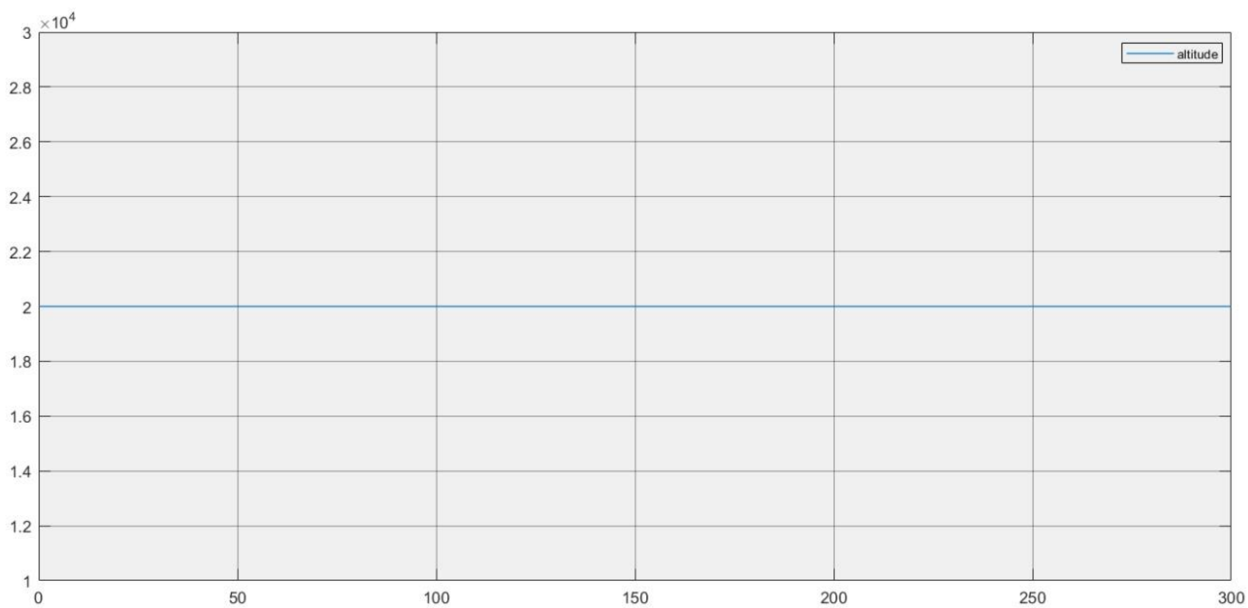


Test h:

Nonlinear model



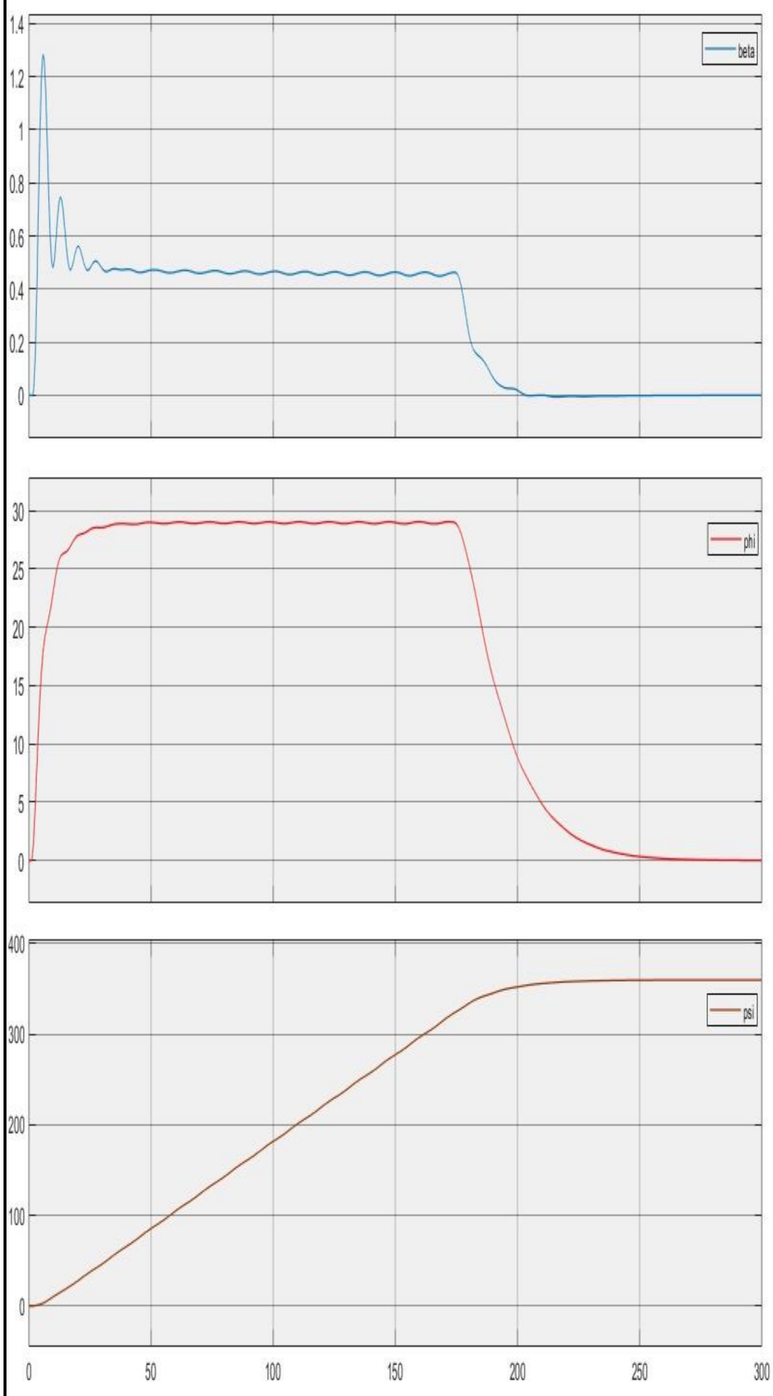
SS

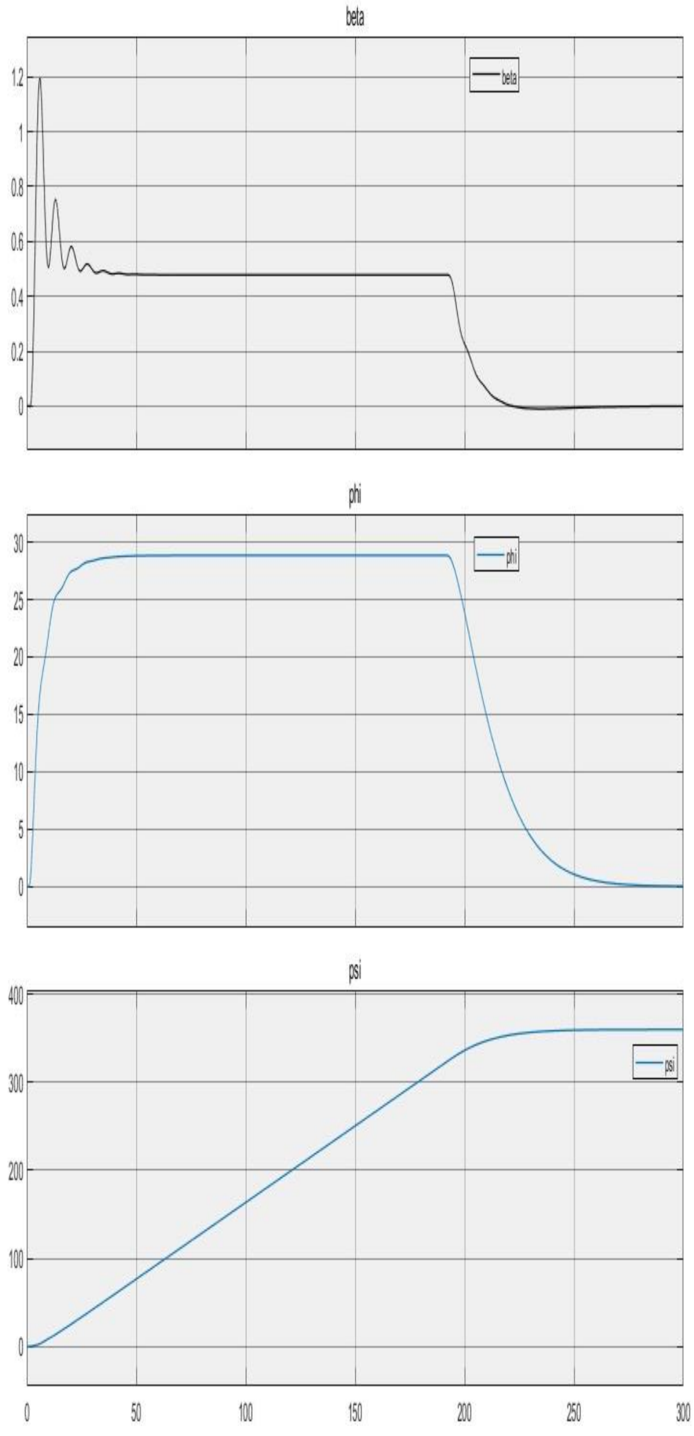


It is a coordinate level not turn

e) ***Test the “Lateral controller + Altitude Hold controller” and compare the response with the same test on the State space model***

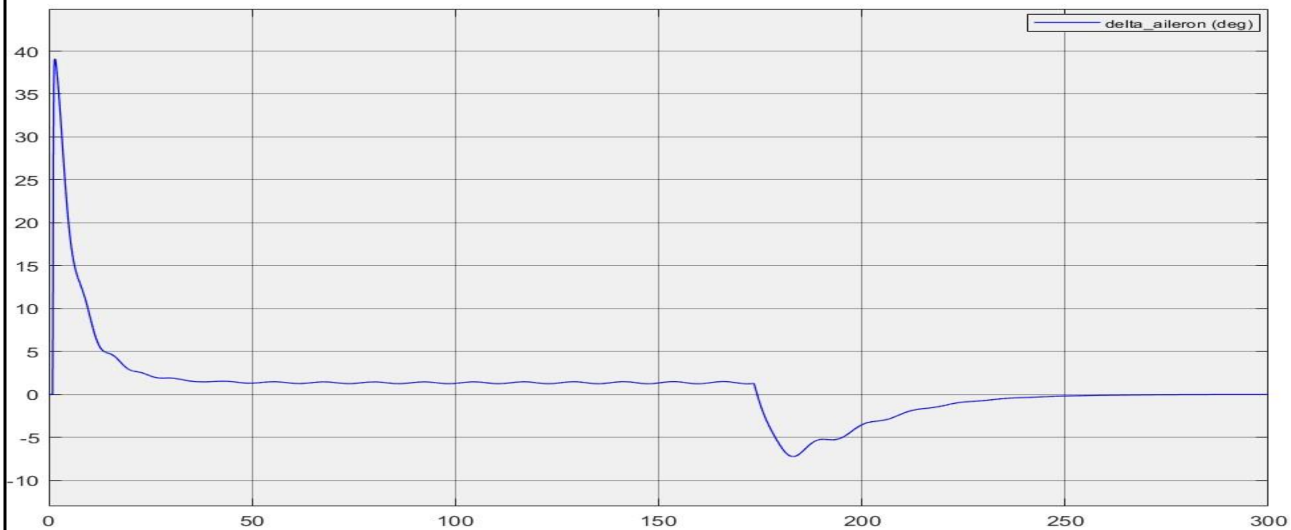
beta_phi_psi



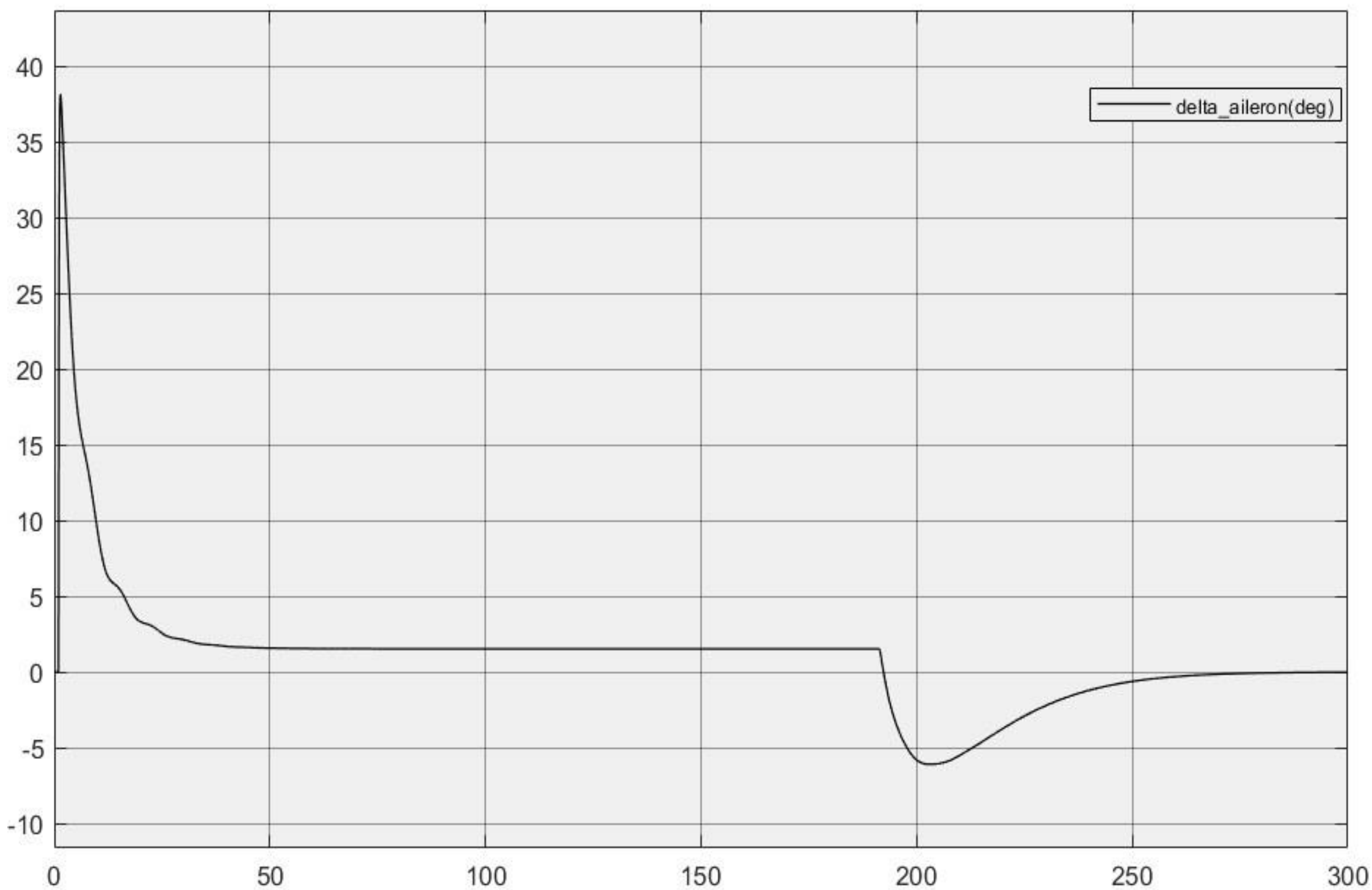


delta aileron

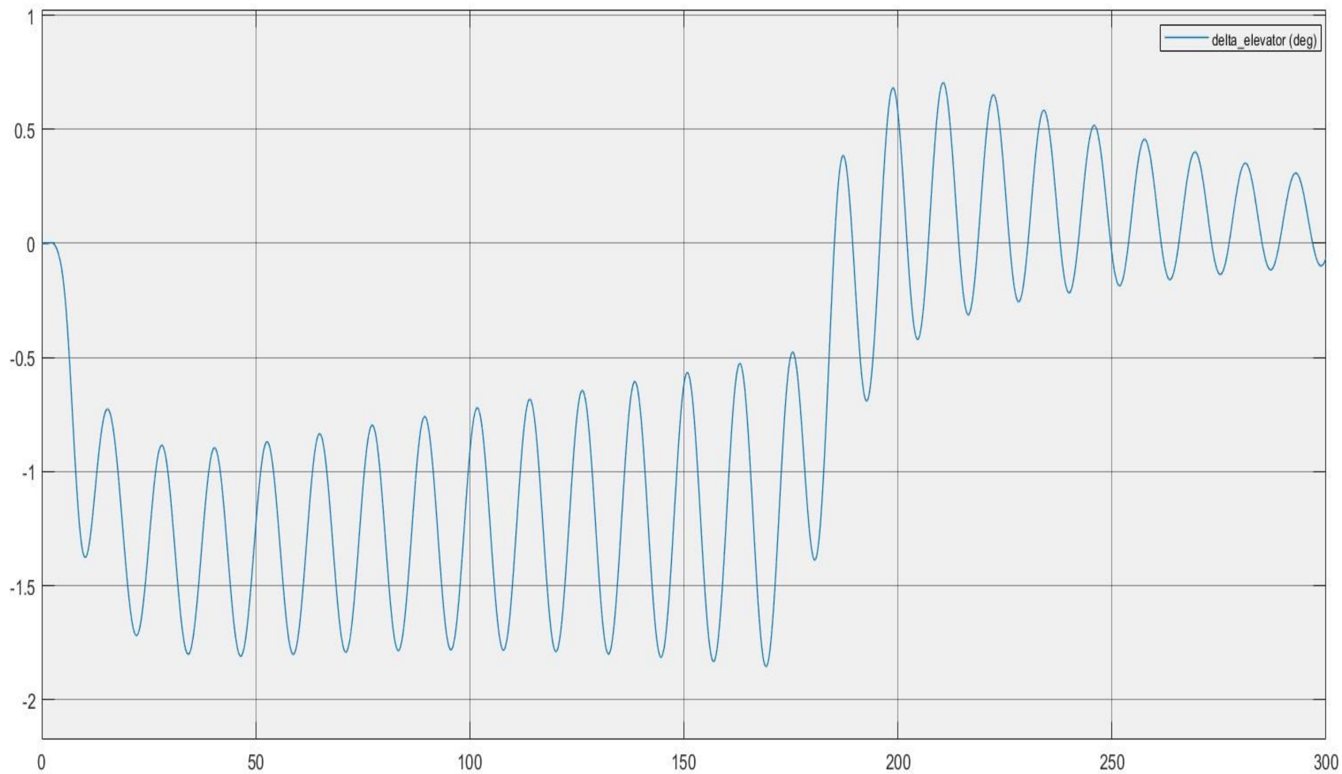
NL model

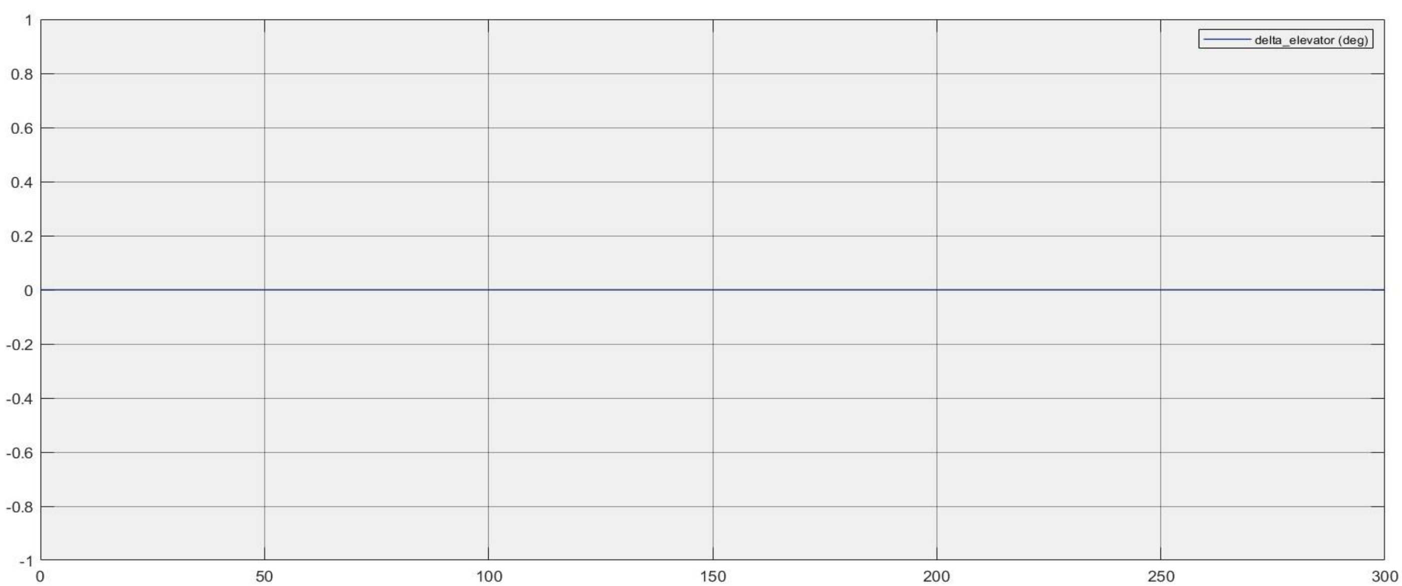


SS

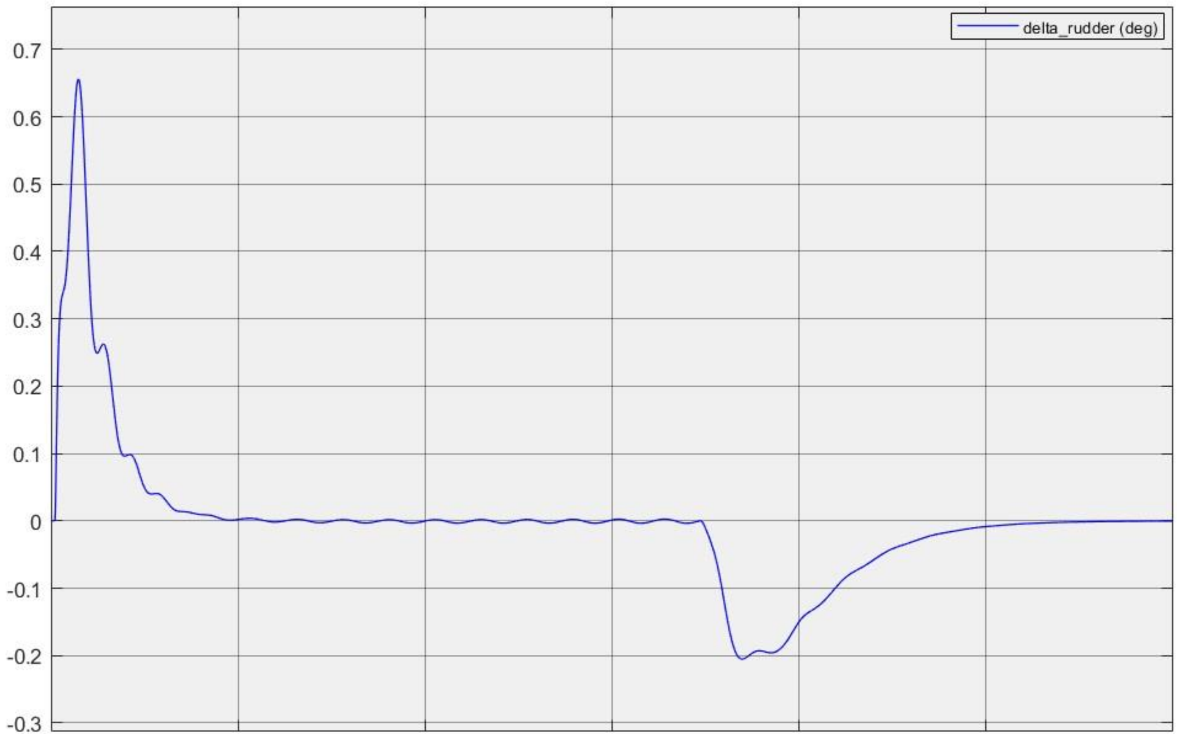


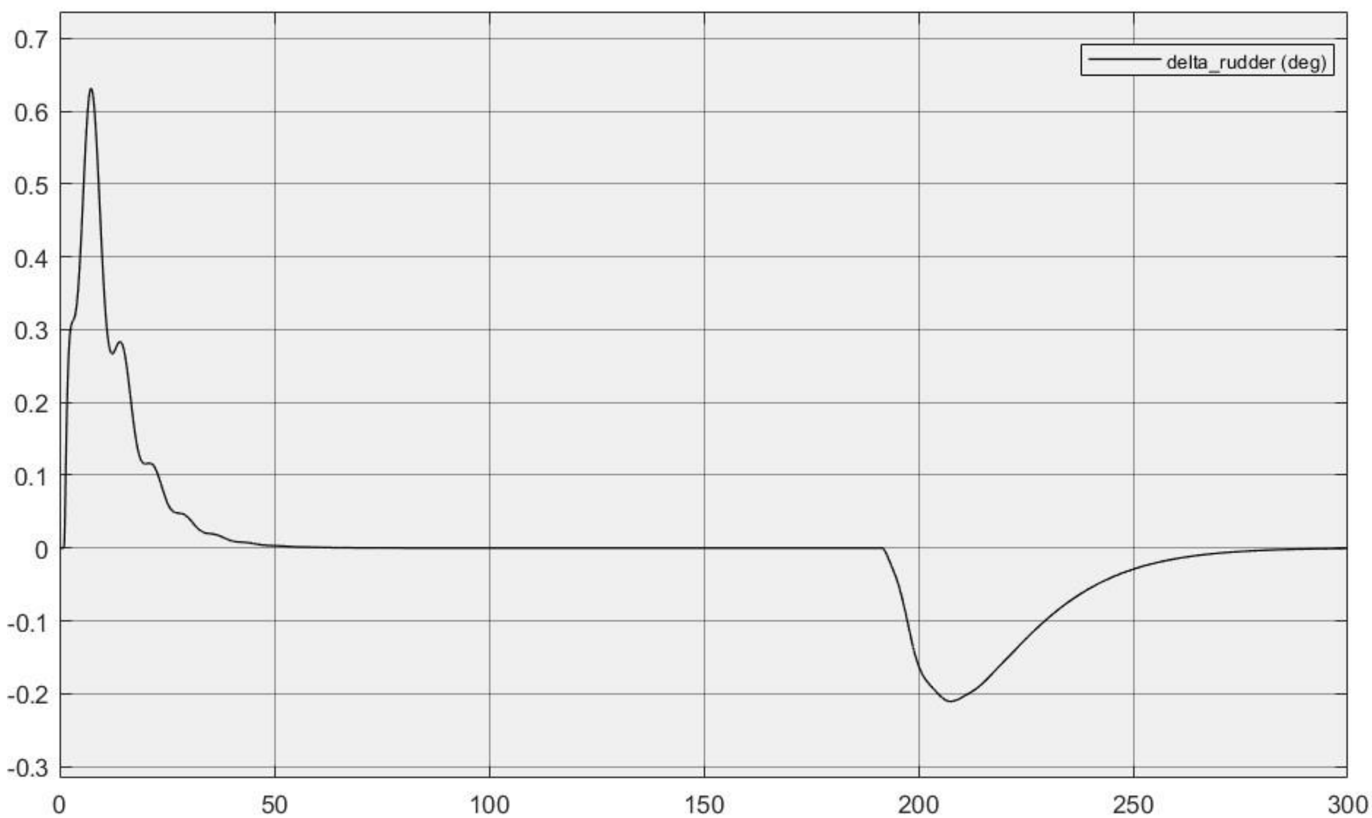
delta_elevator



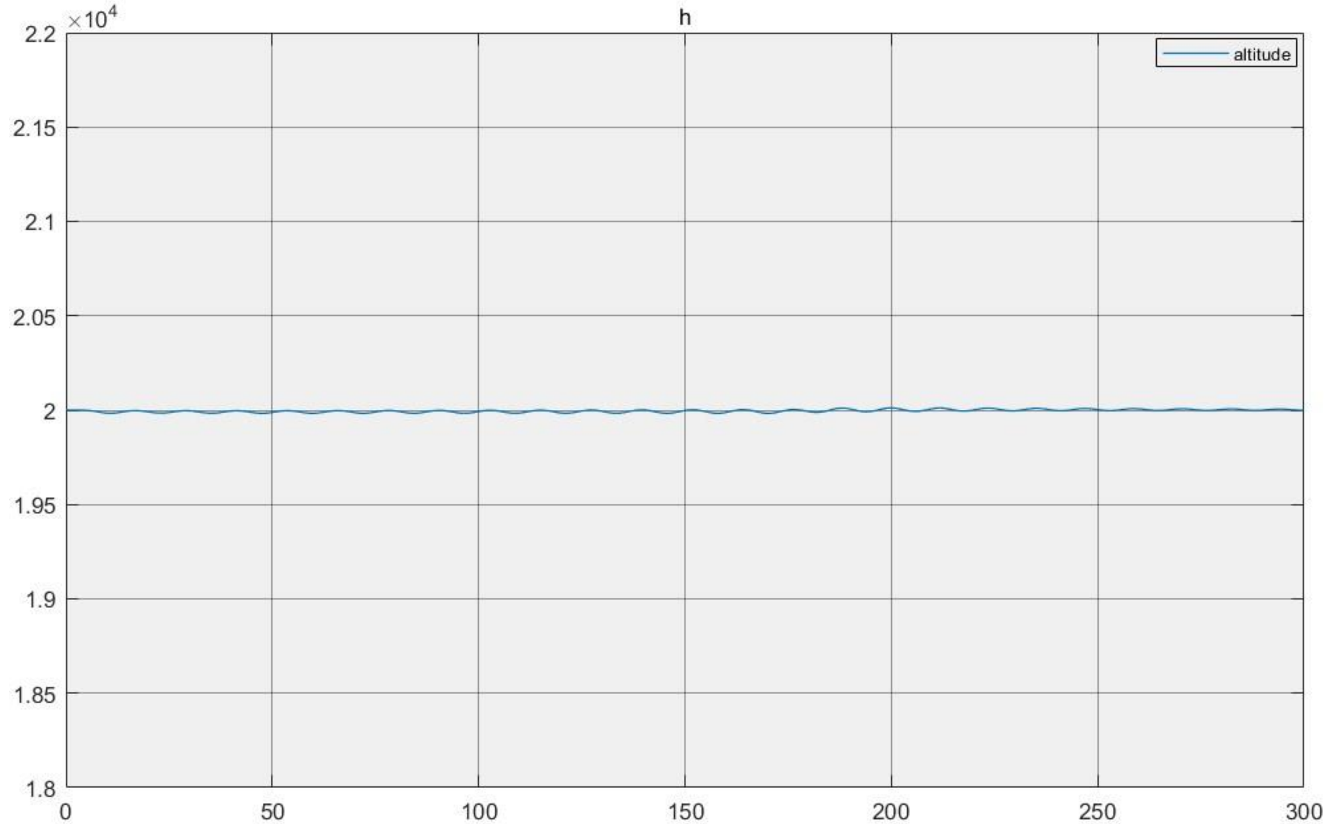


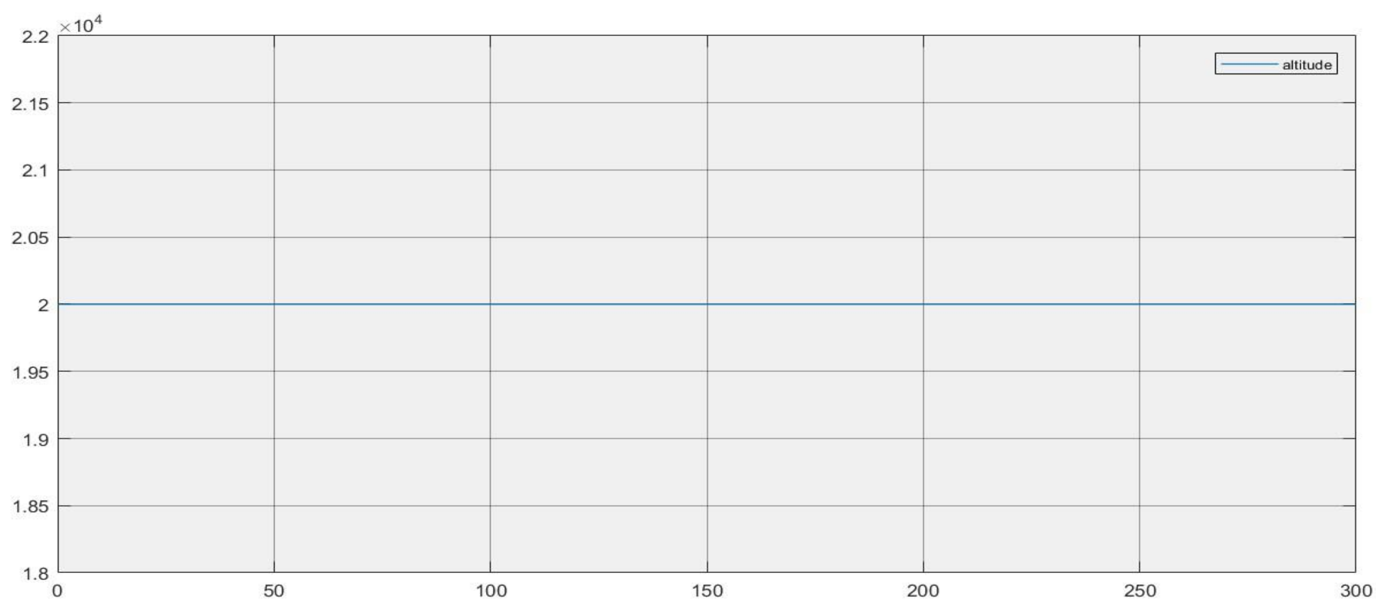
Delta_rudder





H





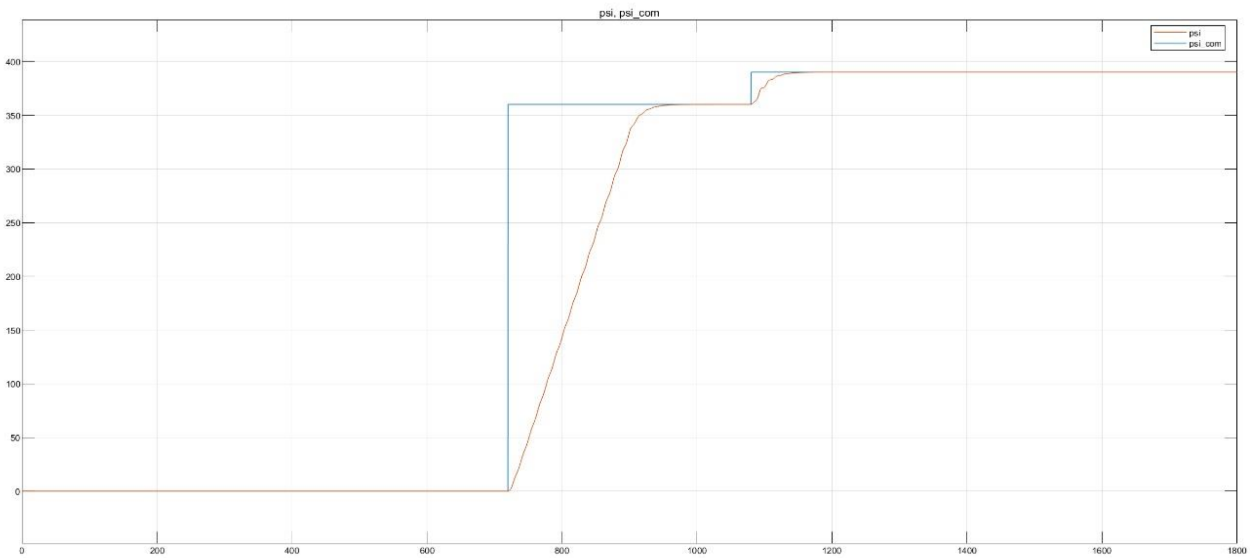
achive coordinates turn level

f) ***Perform a complete autonomous mission including "climb, cruise, turn, descent"***

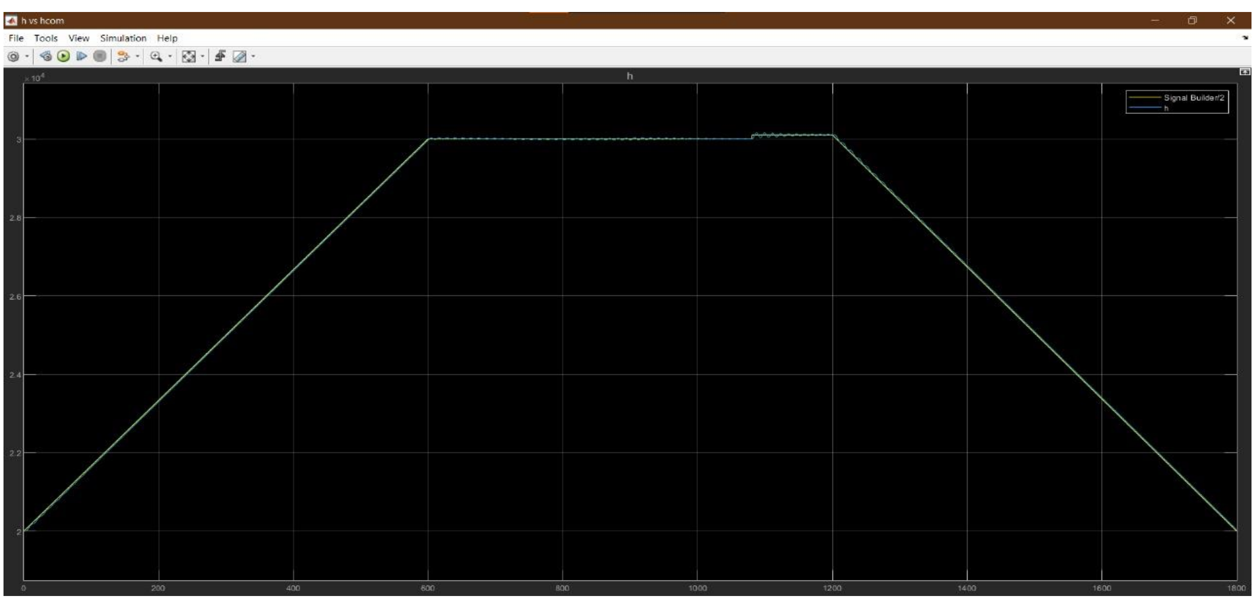
now we can do a mission and all Simulink model used in this task we uploaded it in file of code with video of mission.

We will show the Result

Psi_Psi command



H



Ascending from 20000 to 30000 and then constant and will up 100 ft and descending. And the visualization using flight gear uploaded with code