Controls Systems Design of an Autonomous, Supersonic UAV

a project presented to
The Faculty of the Department of Aerospace Engineering
San Jose State University

in partial fulfillment of the requirements for the degree
Master of Science in Aerospace Engineering

by

Kenneth Gorospe

May 2025

approved by

Dr. Periklis Papadopoulos
Faculty Advisor

ABSTRACT

Dynamics and Controls Systems Design of a Supersonic UAV

Kenneth Gorospe

This project will focus on the design of a control system and autopilot system of a supersonic unmanned aerial vehicle (UAV). The design of an autonomous, supersonic UAV will be achieved by implementing various feedback controllers such as PID, LQR, LQG, and others within the longitudinal and lateral-directional system of the UAV to ensure stability at supersonic speed. Afterwards, the design of stability augmentation systems (SAS), control augmentation systems (CAS), and autopilot systems will be implemented to complete the objectives of this project.

Acknowledgements

I would like to thank my friends and family back home for being supportive of my education at San Jose State these past 6 years. I missed a lot of events back home due to being in San Jose, but I hope to have made everyone proud of my achievements.

I want to thank my amazing girlfriend for being by my side while I navigated post-undergraduate life before ultimately deciding to return to school. Without your unconditional support and being the joy of my life, I wouldn't be where I am without you. I love you so much, my love.

A special thanks to Dr. Papadopoulos for his guidance throughout this project. Your insight on the Aerospace industry does not go unnoticed. I aim to apply what I've learned to my career and be the best engineer I can be.

Table of Contents

List of Figures

List of Tables

Symbols

Symbols	Definitions	Units (SI)
X_u	X-force due to change in forward velocity	1/s
X_α	X-force due to change in angle of attack	ft/s^2
Z_u	Z-force due to change in forward velocity	1/s
Z_α	Z-force due to change in angle of attack	ft/s^2
M_u	Pitching moment due to change in forward velocity	1/ft*s
M_α	Pitching moment due to change in angle of attack	1/s^2
$M_{\dot{\alpha}}$	Pitching moment due to rate of change of angle of attack	1/s
M_q	Pitching moment due to pitch acceleration	1/s
X_{δ_e}	X-force due to change in elevator deflection	ft/s^2
Z_{δ_e}	Z-force due to change in elevator deflection	ft/s^2
M_{δ_e}	Pitching moment due to change in elevator deflection	1/s^2
Y_β	Side force due to change in side slip angle	ft/s^2
L_β	Rolling moment due to change in side slip angle	1/s^2
L_p	Rolling moment due to roll rate	1/s
L_r	Rolling moment due to pitch rate	1/s

N_{β}	Yawing moment due to change in side slip angle	$1/s^2$
N_{p}	Yawing moment due to change in roll rate	1/s
N_{r}	Yawing moment due to change in yaw rate	1/s
Y_{δ_r}	Side force due to change in rudder deflection	$(ft/s)^2$
N_{δ_r}	Yawing moment due to change in rudder deflection	$1/s^2$
Y_{δ_a}	Side force moment due to change in aileron deflection	$1/s^2$
L_{δ_a}	Rolling moment due to change in aileron deflection	ft/s^2
N_{δ_a}	Yawing moment due to change in aileron deflection	$1/s^2$
Greek Symbols		
u	Forward velocity	ft/s
α	Angle of attack	deg
q	Pitch rate	deg/s
θ	Pitch angle	deg
δ_e	Elevator deflection	deg
ϕ	Roll angle	deg
p	Roll rate	deg/s
β	Side slip angle	deg
r	Yaw rate	deg/s
ψ	Heading angle	deg
δ_r	Rudder deflection	deg

δ_a	Aileron deflection	deg
Acronyms		
UAV	Unmanned Aerial Vehicle	----------
CAS	Control Augmentation System	----------
SAS	Stability Augmentation System	----------
PID	Proportional-Integral-Derivative	----------
LQR	Linear Quadratic Regulator	----------
LQG	Linear Quadratic Gaussian	----------
LTI	Linear Time Invariant	----------
ISE	Integral Squared Error	----------
KKT	Karush-Kuhn-Tucker	----------
SLC	Successive Loop Closure	----------
TECS	Total Energy Control System	----------
OL	Open Loop	----------

Chapter 1: Introduction

1.1 Motivation

Unmanned Aerial Vehicles (UAVs) are starting to receive more attention worldwide in both military and commercial usages due to their effectiveness and multipurpose use. While UAVs are receiving its rightful attention, it can be further advanced by focusing on how it can travel at faster speeds, supersonic speeds specifically, autonomously. A supersonic UAV can become a reality by improving its control systems as it would allow the UAV to be dynamically stable at such high speeds. As for autonomous flight, the development of various autopilot systems paired with the improved control system creates a product appeasing whomever desires such UAV.

1.2 Literature Review

This section of the report will provide a literature review on the foundation of the dynamics and control systems of an aircraft, applicable to a UAV. When discussing the dynamics and control systems, the following key points will be discussed:

- Longitudinal and lateral-directional dynamics
- Various controllers such as PID, LQR, LQG
- Autopilot/Flight path tracking.

1.2.1 Longitudinal Dynamics and Control

The longitudinal dynamics of a UAV involves the states/perturbation variables of forward velocity, angle of attack, pitch angle, and pitch rate. In addition to the perturbation variables, a control surface of elevator deflection is involved. Due to the aerodynamic forces a UAV would endure in flight such as climb, descent, and level flight, it is essential to control these variables to maintain stability and performance by developing various control systems to improve the responsiveness of the UAV.

One way to improve the responsiveness of the UAV is with the use of Proportional-Integral-Derivative (PID) controllers. PID controllers are commonly used in longitudinal dynamics due to their simplicity and reliability. PID controllers are able to reduce the error between the analytical and true states by adjusting control surfaces, elevator deflection in this instance. With the usage of this controller, it is able to assist in regulating the pitch and altitude of an aircraft by minimizing the oscillations such aircraft would experience [1].

While PID controllers have demonstrated their reliability, it can be further optimized by focusing on the Integral Squared Error (ISE) [2]. In this study on the ISE, it further explores PID controller optimization by minimizing their designated performance index function and finding optimal PID controller gains via Parseval's Theorem and Karush-Kuhn-Tucker (KKT) necessary conditions [2]. Figure 1 demonstrates the Open-Loop (OL) step response of their UAV without ISE optimization. Figure 2 shows the response of the UAV after including the optimized PID

controller gains. While PID controller gains can be tinkered with, this study demonstrates how the controller gains can be derived to directly improve stability and performance of a UAV.

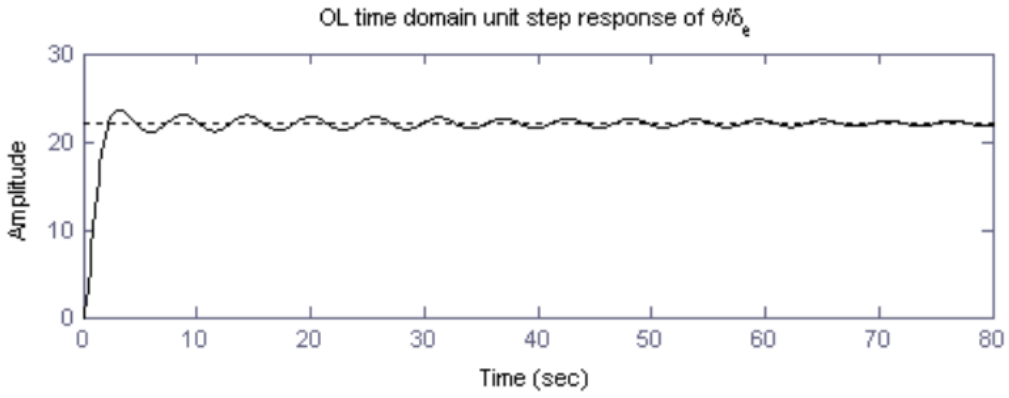

Figure 1 - OL step response without ISE optimization [2]

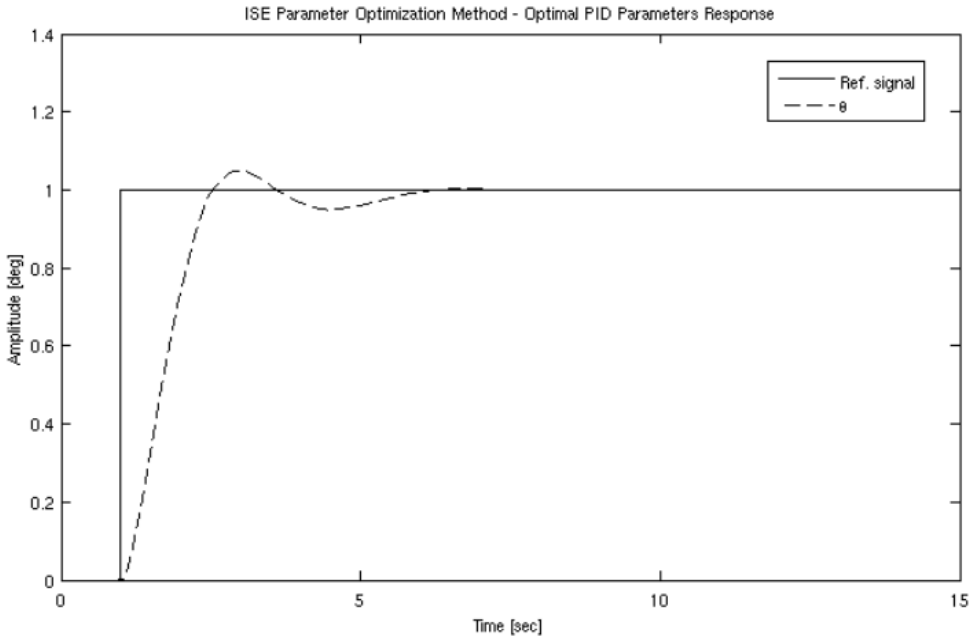

Figure 2 - OL step response with ISE optimization [2]

The previous study optimized a PID controller via ISE optimization. Another technique of PID optimization is with Successive Loop Closure (SLC) [3]. With SLC, it aimed to use four separate closed loops where the pitch attitude angle or airspeed were tuned with respect to either the elevator or throttle of the UAV. With these closed loops, the transfer functions are then extracted and a PID controller is used to form the longitudinal autopilot of pitch attitude hold, altitude hold, airspeed hold with the pitch angle, and airspeed hold using the throttle [3]. To compare the performance of SLC, it is compared to a Total Energy Control System (TECS) controller. TECS aimed to control the total power used by the aircraft via specific energy rate error and distributed energy rate error. Upon completion of this study, it reveals how SLC has

superior performance to TECS, but TECS is able to reduce flight costs and improve endurance of the aircraft [3].

PID controllers are able to demonstrate their effectiveness in improving stability and performance for control systems. Whether a technique of ISE, SLC, TECS, or the usage of multiple closed-loop systems involving attitude, position, or multiple states [4, 5], PID controllers are able to provide the stability needed for flight.

1.2.2 Lateral-Directional Dynamics and Control

When dealing with the lateral-directional dynamics and control of a UAV, its states involve the sideslip angle, roll rate, yaw rate, yaw angle, and bank angle. As for the control inputs, they would be aileron and rudder deflection. Based on the states involved with lateral-directional dynamics, it is expected to stabilize and improve yaw and roll responsiveness for the aircraft.

Similarly to longitudinal dynamics, lateral-directional dynamics are able to utilize PID controllers to its own advantage. For instance, the three modes that a UAV would exhibit would be the spiral mode, dutch roll mode, and roll mode [1]. Due to the existence of these three modes, stabilizing the rolling motion, increasing the damping of the dutch roll, and improving the stability of the spiral mode is necessary. With the usage of various feedback loops such as roll angle rate, roll angle, and course angle, PID controllers are able to achieve these objectives to stabilize the UAV laterally [1].

1.2.3 LQR, LQG controllers and Kalman Filter

After touching on the effectiveness of PID controllers, there exists other controllers such as Linear Quadratic Regulators (LQR) and Linear Quadratic Gaussian (LQG). LQR controllers are extremely effective for linear time-invariant (LTI) systems as it aims to minimize a desired measurement, one which could be a performance index of a system [6]. Similar to a PID controller, it relies on feedback gains that can either be analytically derived or experimentally found. Rather than focusing a gain K with PID, LQR utilizes a diagonal matrix Q and constant value R to tune the LQR controller. Using the LTI system that was provided along with constructing the LQR feedback gain, the LQR control system is created as seen in Figure 3. By choosing the weights of Q and R for the controller, it affects either the weight of the control inputs or how fast the states of the system decays overtime [7]. Regardless, a LQR controller is able to demonstrate its effectiveness when tuned properly.

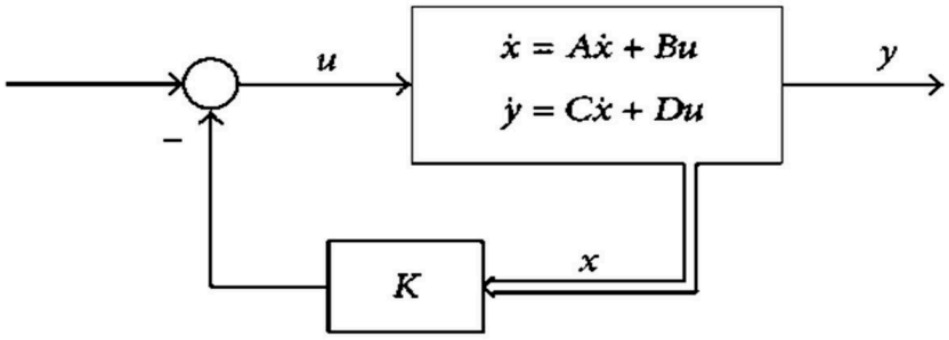

Figure 3 - Architecture of LQR control system [6]

As for LQG controllers, it is considered to be a combination of a LQR controller and a Kalman Filter [6]. To touch on the Kalman Filter, it estimates the state of a system along with minimizing a performance index. To initially construct the LQG controller, designing a Kalman Filter gain matrix through matrix derivation is necessary before it can be implemented alongside a LQR controller. By combining both the Kalman Filter gain matrix and even the previously mentioned LQR controller, the LQG control system is created as seen in Figure 4.

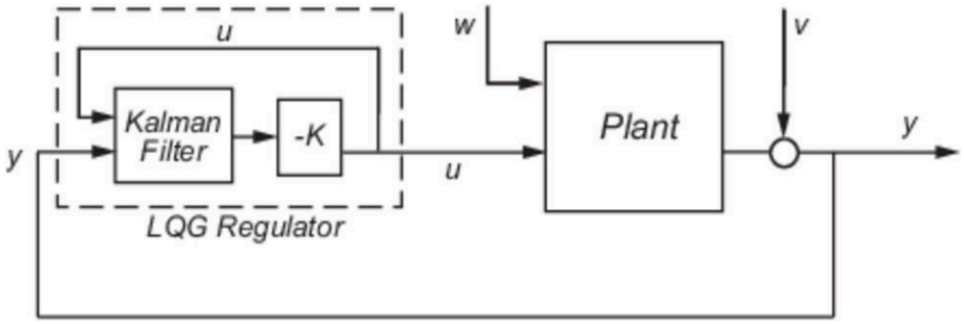

Figure 4 - Architecture of LQG control system [6]

1.2.4 Autonomous Flight

When developing autonomous flight for the proposed supersonic UAV, three aspects are required:

- Stability augmentation systems (SAS)
- Control augmentation systems (CAS)
- Autopilot systems.

SAS are capable of enhancing the stability and reliability of UAVs with controllers mentioned throughout this review. For example, a tuned LQG controller as seen in Figure 5 is able to stabilize the longitudinal and lateral-directional modes of a UAV despite experiencing disturbances in the simulation [8]. Furthermore, a SAS utilizing a tuned LQR controller demonstrates the ability to improve the dynamic stability characteristics of a UAV used in a study by Hanif and Sasongko [9].

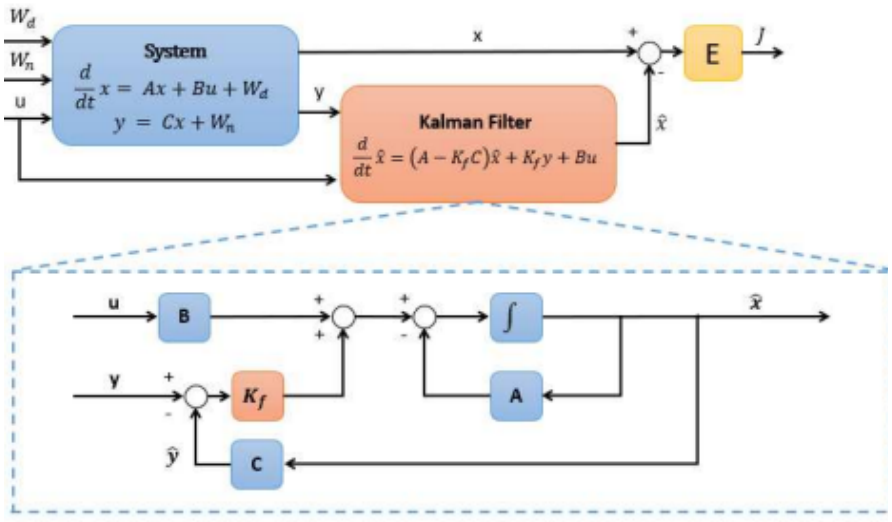

Figure 5 - Architecture of SAS using LQG [8]

CAS maintains the same role as SAS in terms of emphasizing stability for a UAV, enabling the UAV to fly autonomously and handle complex maneuvers in varying environments. In the same study by Hanif and Sasongko, a CAS was used in order to assist tracking for the pitch and flight path angle of their used aircraft [9]. However, the difference between the used SAS and CAS in the study is how CAS did have difficulty in tracking their designated error variable ɣ. Along with the difficulty tracking, its lateral-directional mode still exhibited some oscillations in their tracking [9]. Due to the rareness of not having complete success in improving stability when the CAS designed as it utilized a LQR controller, it is a study to keep in mind as their project progresses.

While SAS and CAS have been discussed individually thus far, both systems can be combined into one controller for both longitudinal and lateral-directional modes as seen in Figure 6 [10]. In this study of the combined controller by Tran et al., their controller is compared to a PI controller (a PID controller without derivatives). Comparing the two controllers and how the UAVs react to changes in aerodynamic forces, the designed controller demonstrates the ability to improve tracking of their attitude angles [10].

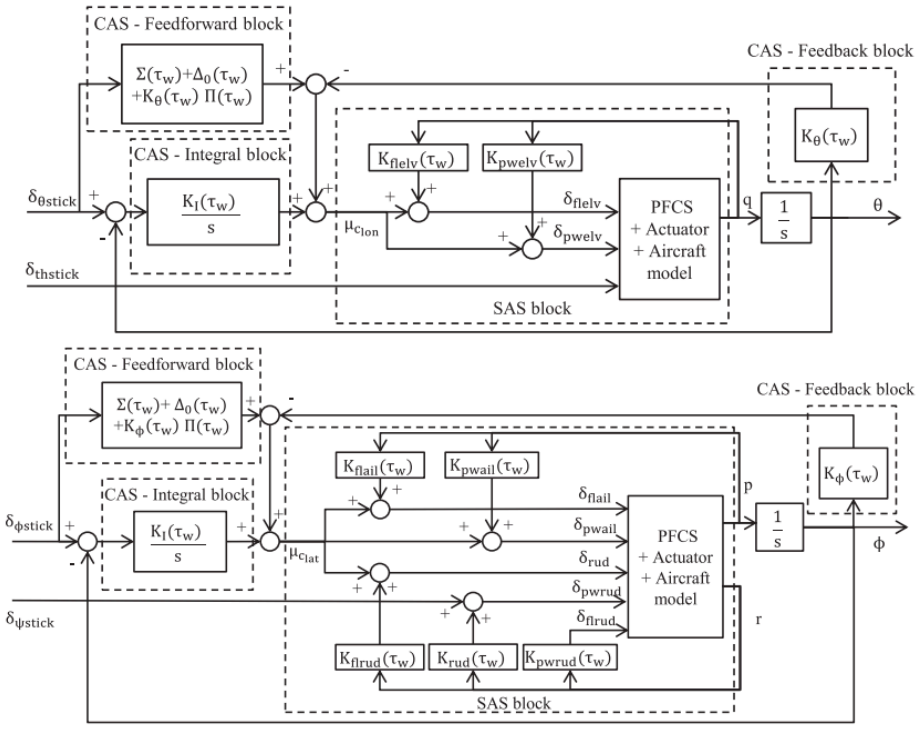

Figure 6 - Architecture of CAS and SAS [10]

For complete autonomous flight, various autopilot systems would need to be produced. Systems for various aspects of flight such as speed, altitude, and pitch hold can be designed accordingly to maintain stable flight [3, 11]. In addition to these variable holds, autopilots for both longitudinal and lateral-directional dynamics for different state variables such as roll and course control along others can be implemented [12]. Autopilot systems for takeoff, landing, and flight path following can be designed as well [13, 14, 15]. For the multitude of autopilot systems needed for autonomous flight, they all return to the foundational idea of controllers to develop the systems. Controllers discussed such as PID, LQR, and LQG are used. More advanced methods which even include the usage of AI to tune such controllers and energy optimization methods are used in certain studies [3, 11]. Now with the baseline of knowledge on longitudinal and lateral-directional dynamics and various controllers that can be used to improve stability and design autonomous flight, the project can proceed.

1.3 Objective

This project aims to design and simulate a control system that allows the UAV to be dynamically stable in flight and various autopilot systems, enabling autonomous flight. Throughout this report, there will be a focus on open-loop and closed-loop analysis of the selected aircraft, design of controllers to improve response times, and the design of SAS, CAS, and autopilot systems to allow autonomous flight.

1.4 Methodology

The objectives of this project can be achieved by dividing it into 4 sections:

- Open-Loop analysis
- SAS
- CAS
- Autopilot systems.

Due to the amount of designing and analysis needed for all sections, MATLAB-Simulink will be heavily relied upon in the development of an autonomous, supersonic UAV.

Chapter 2: Problem Description and Aircraft Modeling

2.1 Problem Setup

When choosing an aircraft to be the baseline model of the autonomous, supersonic UAV, the selected aircraft would be the Lockheed F-104 with flight conditions of an altitude of 55,000 ft at Mach 1.8 as the performance data of the aircraft is widely available.

In order to move forward with analysis on the F-104, assumptions have to be made to simplify the approach to the problem. Therefore, the assumptions of this project are as follows:

- Ideal weather conditions
- Constant dimensions of the aircraft
- Steady-state flight conditions.

Now that such assumptions have been made, establishing the longitudinal and lateral-directional state-space systems is needed. Equation 2.1 would be the model used for the longitudinal dynamics. Equation 2.2 is the model for the lateral-directional dynamics.

$$\begin{bmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_\alpha & 0 & -g \\ Z_u/U_1 & Z_\alpha/U_1 & 1 & 0 \\ \left(M_u + \frac{M_{\dot{\alpha}} Z_u}{U_1}\right) & \left(M_\alpha + \frac{M_{\dot{\alpha}} Z_\alpha}{U_1}\right) & \left(M_q + M_{\dot{\alpha}}\right) & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} X_{\delta e} \\ Z_{\delta e}/U_1 \\ \left(M_{\delta e} + \frac{M_{\dot{\alpha}} Z_{\delta e}}{U_1}\right) \\ 0 \end{bmatrix} \begin{bmatrix} \delta e \end{bmatrix} \tag{2.1}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{p} \\ \dot{\beta} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & \mathfrak{L}_p & \mathfrak{L}_\beta & \mathfrak{L}_r & 0 \\ \frac{g\cos(\Theta_1)}{U_1} & \frac{Y_p}{U_1} & \frac{Y_\beta}{U_1} & \frac{Y_r}{U_1} - 1 & 0 \\ 0 & \mathcal{N}_p & \mathcal{N}_\beta & \mathcal{N}_r & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ p \\ \beta \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \mathfrak{L}_{\delta_r} & \mathfrak{L}_{\delta_a} \\ \frac{Y_{\delta_r}}{U_1} & \frac{Y_{\delta_a}}{U_1} \\ \mathcal{N}_{\delta_r} & \mathcal{N}_{\delta_a} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_r \\ \delta_a \end{bmatrix} \tag{2.2}$$

Now having the longitudinal and lateral-directional dynamic state-space systems, Table 1 gives the necessary longitudinal and lateral-directional derivatives of the F-104 at the given flight conditions initially established.

Table 2.1 - Longitudinal and lateral-directional derivatives of F-104 at altitude of 55,000 ft at mach 1.8 [16]

Longitudinal Derivatives			**Lateral-Directional Derivatives**		
Derivative	Value	Units	Derivative	Value	Units
X_u	-0.0049	1/s	Y_β	-175.8047	ft/s^2
X_α	-32.4692	ft/s^2	L_β	-47.2783	1/s^2
Z_u	-0.0176	1/s	L_p	-0.8692	1/s
Z_α	-346.6128	ft/s^2	L_r	0.4921	1/s
M_u	0	1/ft*s	N_β	7.531	1/s^2
M_α	-18.1248	1/s^2	N_p	-0.0182	1/s
$M_{\dot{\alpha}}$	-0.0783	1/s	N_r	-0.127	1/s
M_q	-0.1844	1/s	Y_{δ_r}	14.6364	(ft/s)^2
X_{δ_e}	0	ft/s^2	L_{δ_r}	4.0161	1/s^2
Z_{δ_e}	-87.9865	ft/s^2	N_{δ_r}	-1.3537	1/s^2
M_{δ_e}	-18.1525	1/s^2	Y_{δ_a}	0	ft/s^2
			L_{δ_a}	8.7948	1/s^2
			N_{δ_a}	0.0778	1/s^2

2.2 Open-Loop Analysis of the F-104

With the longitudinal and lateral-directional state-space systems and derivatives established for the project, the next step is to analyze the OL system for the Lockheed F-104 using MATLAB-Simulink. With analysis for the OL system, determining the controllability and observability for both systems is needed. Afterwards, the poles for the longitudinal and lateral-directional systems are gathered to analyze the stability of the systems. The respective OL

responses are then simulated to visually determine the stability and oscillatory behaviors of the systems.

Starting with the controllability and observability of the longitudinal and lateral-directional dynamics of the F-104, using the MATLAB functions co and obsv after populating the respective state-space models, it is determined that the systems are controllable and observable as their ranks equals the number of states. With the longitudinal dynamics, since there are four states in forward velocity, angle of attack, pitch angle, and pitch rate, the rank of the controllability and observability matrices equals four. Therefore, the longitudinal dynamics are controllable and observable. As for the lateral-directional dynamics, it contains five states: roll angle, roll rate, side slip angle, yaw rate, and heading angle. The ranks for the controllability and observability matrices are five, thus the lateral-directional dynamics are controllable and observable. After completion of determining controllability and observability, the OL poles for the longitudinal and lateral-directional dynamics are found in Figures 7 and 8.

Pole	Damping	Frequency (rad/seconds)	Time Constant (seconds)
-2.44e-03 + 1.78e-02i	1.36e-01	1.80e-02	4.09e+02
-2.44e-03 - 1.78e-02i	1.36e-01	1.80e-02	4.09e+02
-2.31e-01 + 4.26e+00i	5.42e-02	4.26e+00	4.33e+00
-2.31e-01 - 4.26e+00i	5.42e-02	4.26e+00	4.33e+00

Figure 7 - Longitudinal directional OL poles

Pole	Damping	Frequency (rad/seconds)	Time Constant (seconds)
0.00e+00	-1.00e+00	0.00e+00	Inf
-5.14e-03	1.00e+00	5.14e-03	1.94e+02
-1.07e+00	1.00e+00	1.07e+00	9.36e-01
-1.16e-02 + 2.78e+00i	4.17e-03	2.78e+00	8.64e+01
-1.16e-02 - 2.78e+00i	4.17e-03	2.78e+00	8.64e+01

Figure 8 - Lateral-directional OL poles

In Figure 7, the longitudinal directional OL poles can be deemed stable as their poles contain negative real parts. However, due to the magnitude of the negative real parts, the F-104 can be declared marginally stable longitudinally. As for the lateral-directional OL poles in Figure 8, it is the same case as the magnitudes of the negative real parts are close to zero. In addition to analyzing the stability of the longitudinal and lateral-directional dynamics, dynamic modes for each system can be analyzed. The short-period phugoid modes of the longitudinal dynamics are observed within the OL poles. The first two lines of the longitudinal poles would be the phugoid

mode, and the last two lines are the short-period mode. As for the lateral-directional poles, the modes can be seen as:

- Second pole: Spiral mode
- Third pole: Roll mode
- Fourth and fifth poles: Dutch roll mode.

By determining which poles are their respective modes, approximations of each mode can be done if deemed necessary. Now that OL poles were analyzed, the longitudinal and lateral-directional OL systems can be designed in MATLAB-Simulink as seen in Figure 9 and 10.

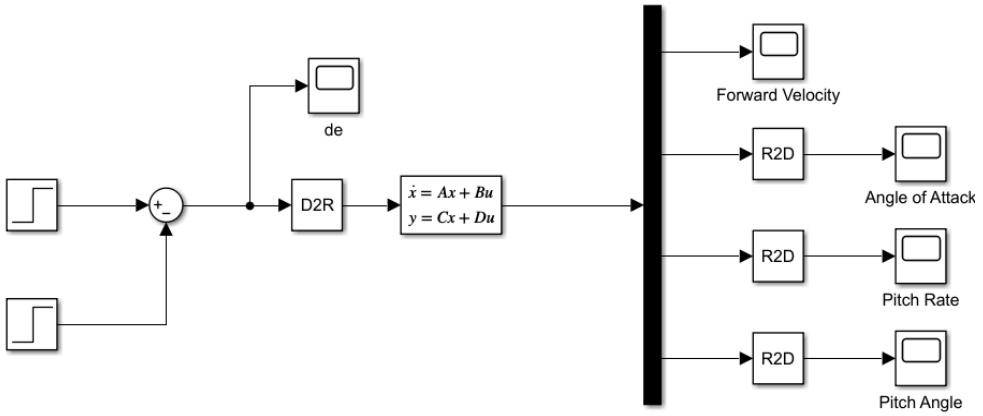

Figure 9 - Longitudinal OL system

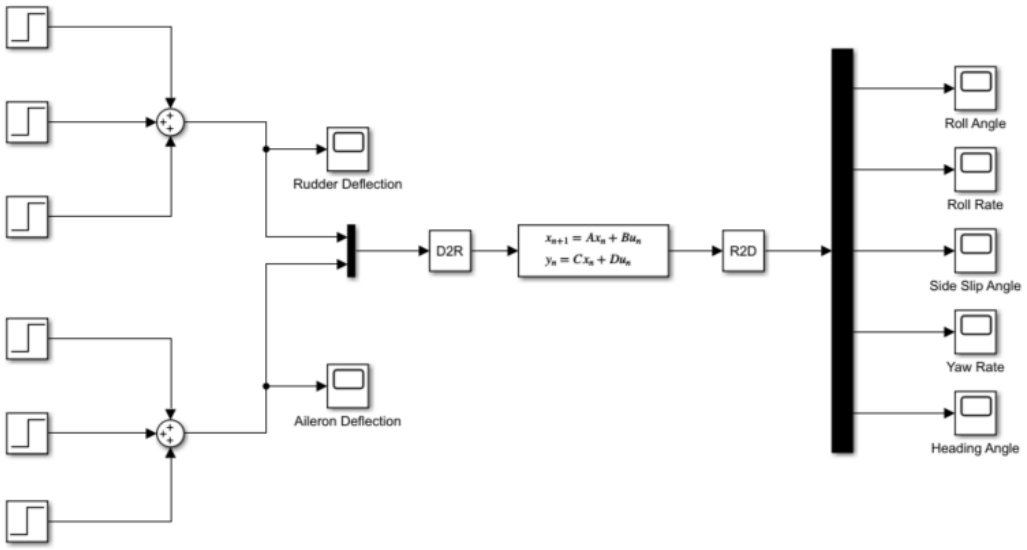

Figure 10 - Lateral-directional OL system

Before the longitudinal and lateral-directional OL systems can be analyzed, control inputs for elevator, rudder, and aileron deflections are inputted as seen in Figure 11. For

elevator deflection for the longitudinal OL system, a 4 degree input was simulated. As for rudder and aileron deflection, inputs of +/- 2 degrees are inputted. With these control inputs, the longitudinal and lateral-directional OL responses are plotted in Figures 12 and 13.

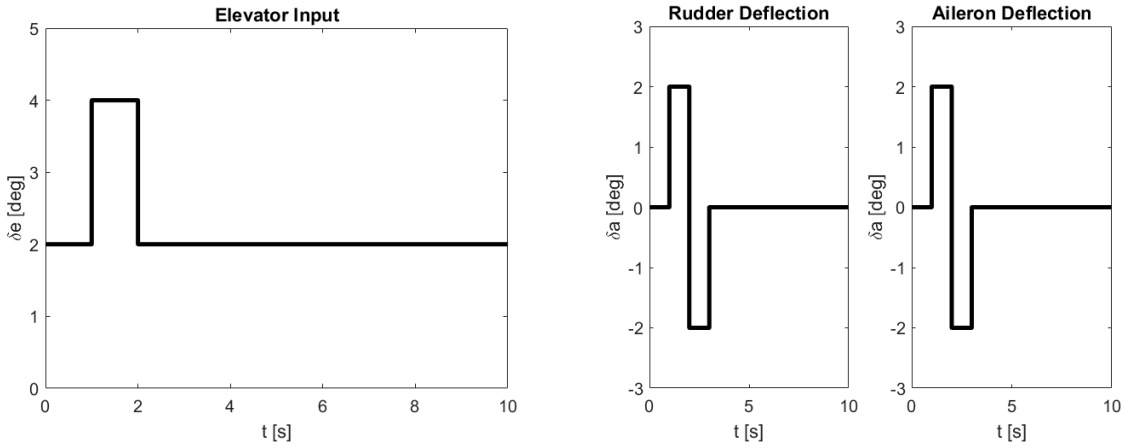

Figure 11 - Longitudinal and lateral-directional control inputs

In Figure 12, it can be seen how the longitudinal OL responses of the F-104 are stable. One thing to note is how the angle of attack and pitch rate states exhibit oscillatory behavior initially in the simulation. However, those oscillatory behaviors are later damped. As for the forward velocity and pitch angle, they still exhibit oscillations after 400 seconds of simulation time. Perhaps if the simulation time was increased, the oscillations would be damped as well.

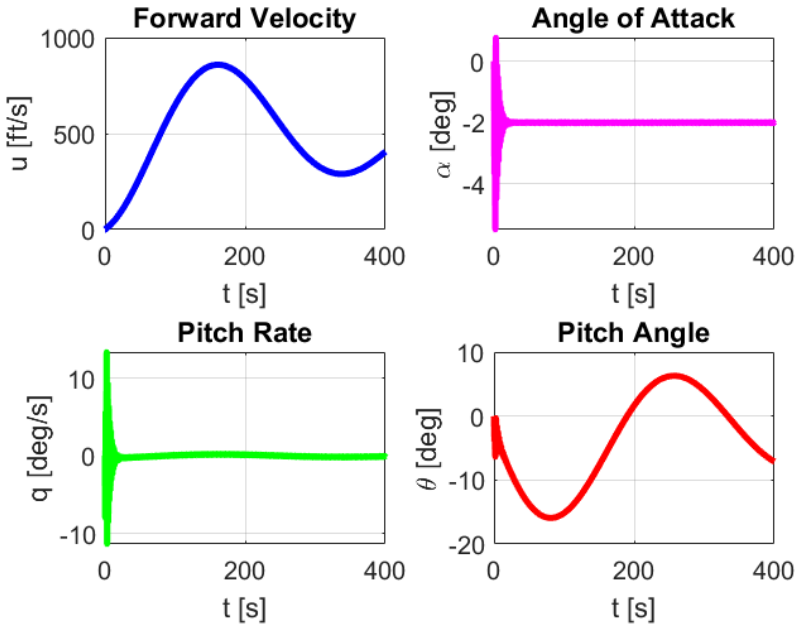

Figure 12 - Longitudinal directional OL response

When analyzing the lateral-directional OL responses in Figure 13, all states laterally exhibit oscillations that are dampened as the simulation progresses. Roll angle and roll rate have the highest magnitudes in 10 degrees and 10 degrees per second respectively. Based on the oscillatory behavior these states exhibit, the F-104 struggles with stability laterally.

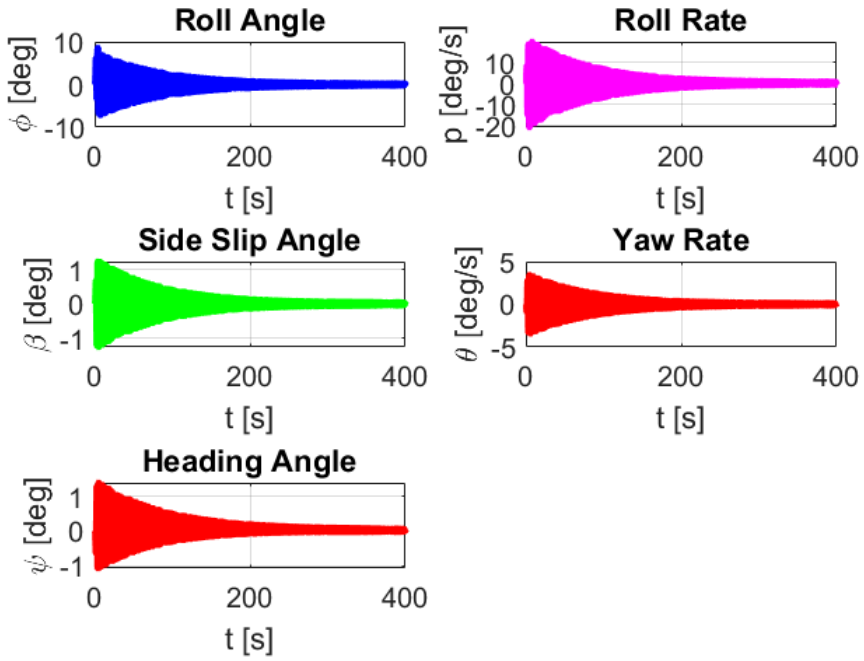

Figure 13 - Lateral-directional OL response

Chapter 3: Pitch-Altitude Hold Autopilot

This section of the report will begin the investigation of the various autopilot systems that would be necessary for the Lockheed F-104. Such autopilot systems would include:

- Pitch-attitude hold
- Altitude hold
- Speed/Mach hold
- Roll angle hold
- Heading hold/VOR hold.

When designing the necessary autopilot systems, various controllers such as PID, LQR, LQG, Kalman Filter, and Dynamic Inversion may be implemented to stabilize such systems.

3.1 Dynamic Inversion Control System

In Figures 14 and 15, when designing a control system meant for pitch-attitude hold, the chosen controller in order to stabilize the angle attack and pitch rate is a dynamic inversion control system. Between both figures, they are identical. The only difference is how each of the gain blocks are adjusted to account for the state variable that is desired to be outputted. Figure 14 has its dynamic inversion control system be designed to output the angle of attack of the aircraft. As for Figure 15, instead of the angle of attack, the pitch rate is outputted.

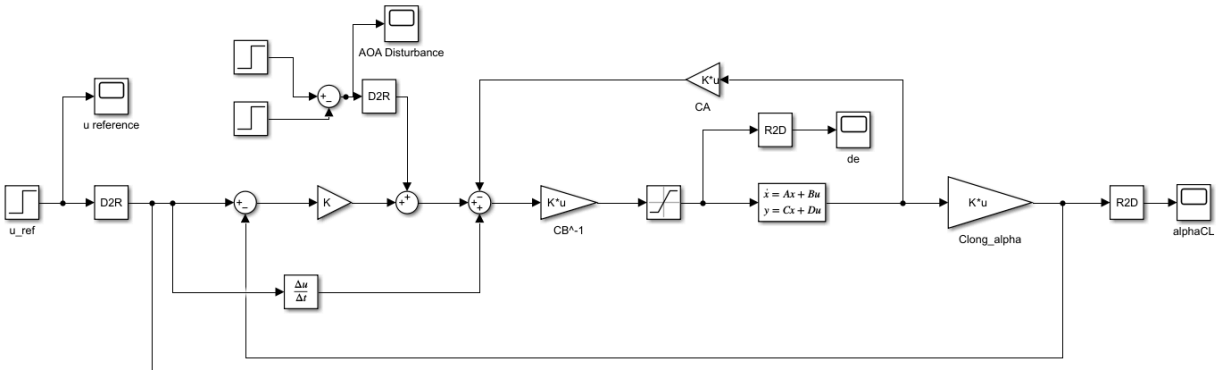

Figure 14 - Dynamic inversion control system for angle of attack

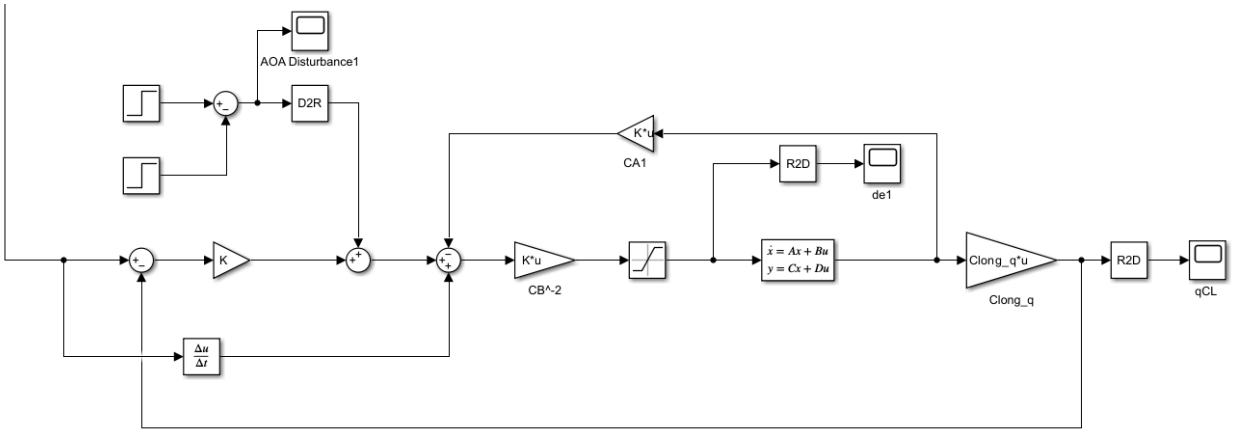

Figure 15 - Dynamic inversion control system for pitch rate

The dynamic inversion control systems contain a forward reference velocity of 0 feet per second due to the longitudinal derivatives accounting the F-104 traveling at Mach 1.8, thus inputting Mach 1.8 as the reference signal is unnecessary. The control system also includes a saturation block in order to limit the elevator deflection to +/- 20 degrees as seen in Figure 16. As for the gain block, it contains a value of 5. By establishing the contents of each block in the dynamic inversion control system, the angle of attack and pitch rate responses can be seen in Figure 17.

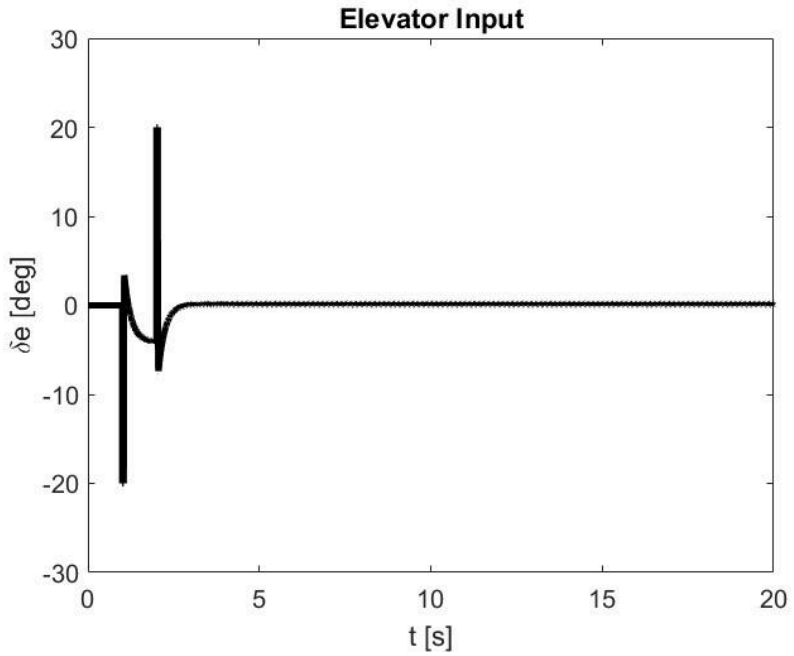

Figure 16 - Elevator input from dynamic inversion control system

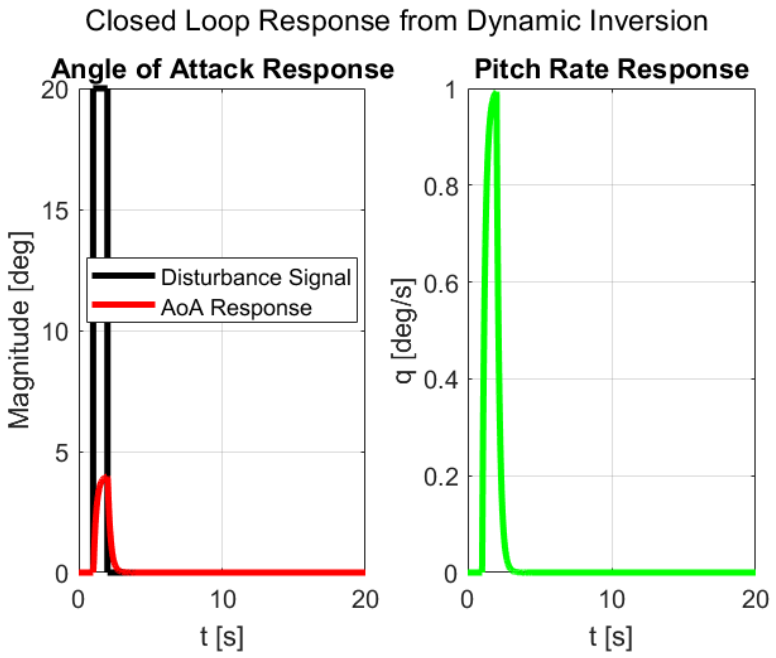

Figure 17 - Angle of attack and pitch rate response for pitch-attitude hold autopilot

The implementation of the dynamic inversion control system is able to stabilize the angle of attack and pitch rate of the F-104 in less than three seconds. The difference between the usage of dynamic inversion and the OL response are dramatic as there are no oscillations present and such magnitudes are 4 degrees and 1 degree per second, respectively.

One aspect of the dynamic inversion control system to note with this autopilot system is how MATLAB was unable to observe the pitch angle. When conducting the same process for each of the states, checking their observability and controllability when outputting angle of attack, pitch rate, and pitch angle, mathematically they are each observable and controllable. However, with inverse calculations, an infinite value appears, so proceeding with a control system outputting the pitch angle was unachievable.

A gain value of 5 was chosen temporarily in order to see its effect on the dynamic inversion control system. Such value was able to stabilize the angle of attack and pitch rate and decrease such magnitudes with ease. If the gain were to be increased, it would improve the performance of the dynamic inversion, but it shall be left at 5 for now.

Chapter 4: Altitude Hold Autopilot

4.1 LQG Control System

The altitude hold autopilot is designed using a LQG controller as seen in Figure 18. The LQG controller is able to filter out disturbances provided by a Kalman Filter that the F-104 may experience in flight. Due to initial flight conditions of the F-104 traveling at Mach 1.8 with an altitude of 55,000 feet, the controller's reference altitude would be 55,000 feet as well. The integrator block contains initial conditions of [1742, 0, 0, 0, 55000] as it simulates the real-time flight conditions where the F-104 flies at Mach 1.8 with an angle of attack of 0 degrees, pitch rate of 0 degrees per second, pitch angle of 0 degrees, and altitude of 55,000 feet. As for the LQR gain in the LQG controller, its state and control effort weighting matrix, Q_c and R_c, were optimized to focus on altitude hold. With the LQG controller defined, it can be seen that the controller is able to maintain altitude in Figure 19.

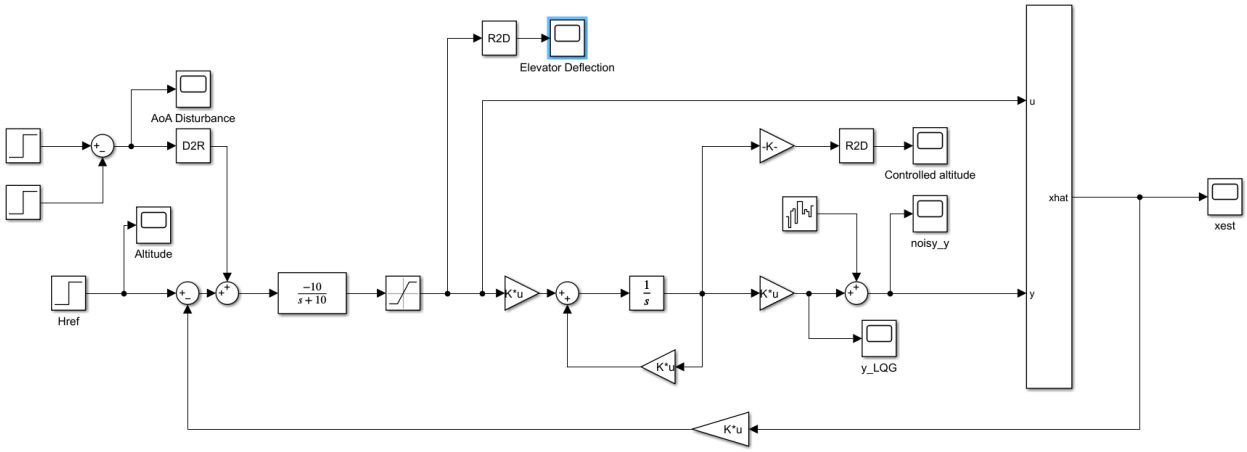

Figure 18 - LQG control system for altitude hold

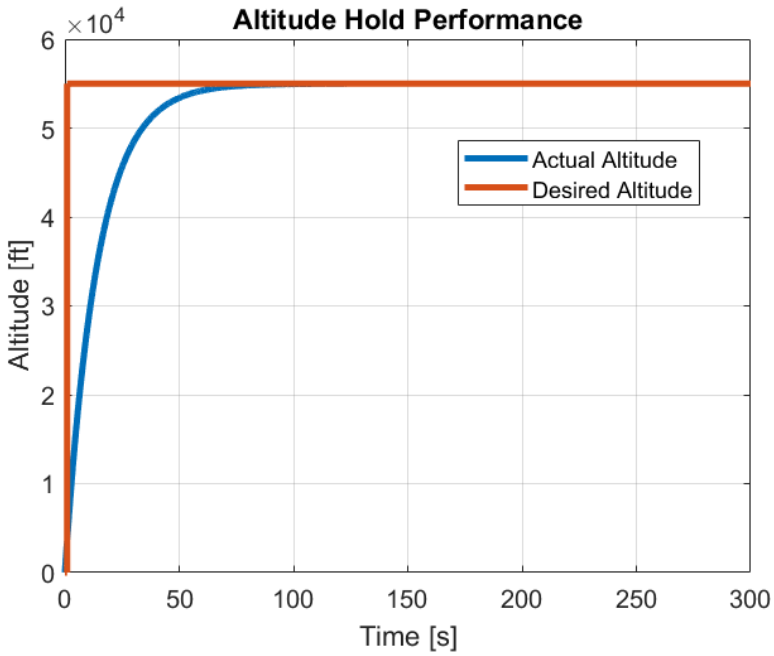

Figure 19 - Performance of altitude hold autopilot

With the assistance of the LQG controller, Figure 19 is able to demonstrate that the F-104 would be able to maintain an altitude of 55,000 feet after 75 seconds. The LQR gain is tuned with a matrix of [1 1 10 100 1000], putting more emphasis on the pitch rate, pitch angle, and altitude while not putting weight on the forward velocity and angle of attack. While the gain is optimized in such a format currently, perhaps with experimentation, forward velocity and angle of attack can have more weight in the gain while still being able to achieve the same results.

Chapter 5: Roll Angle Hold Autopilot

5.1 PID Control System

A PID controller was developed in order to maintain a reference roll angle as seen in Figure 20. PID was the choice for the development of the roll angle hold autopilot as it was the simplest controller to use while still maintaining effectiveness. When maintaining a specified roll angle, the PID controller is designated to track an angle of 5 degrees. The controller would utilize proportional control and derivative action with its tuning to track a roll angle of 5 degrees. After such tuning, the PID controller was able to track the desired angle as seen in Figure 21.

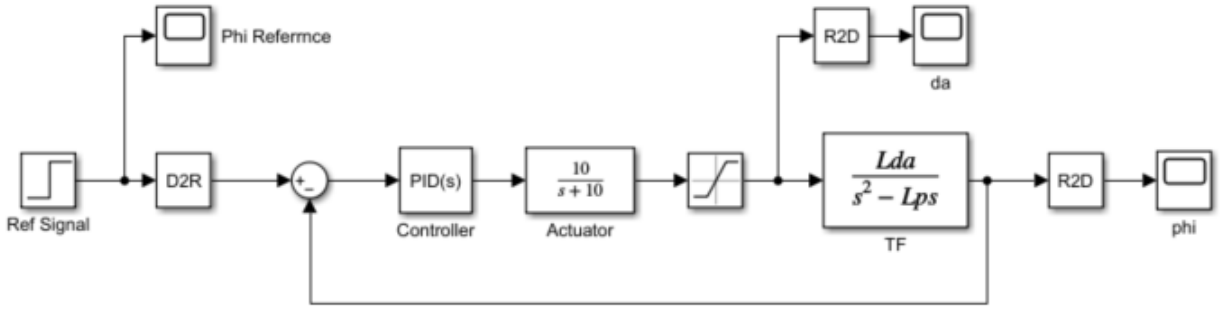

Figure 20 - PID control system for roll angle hold

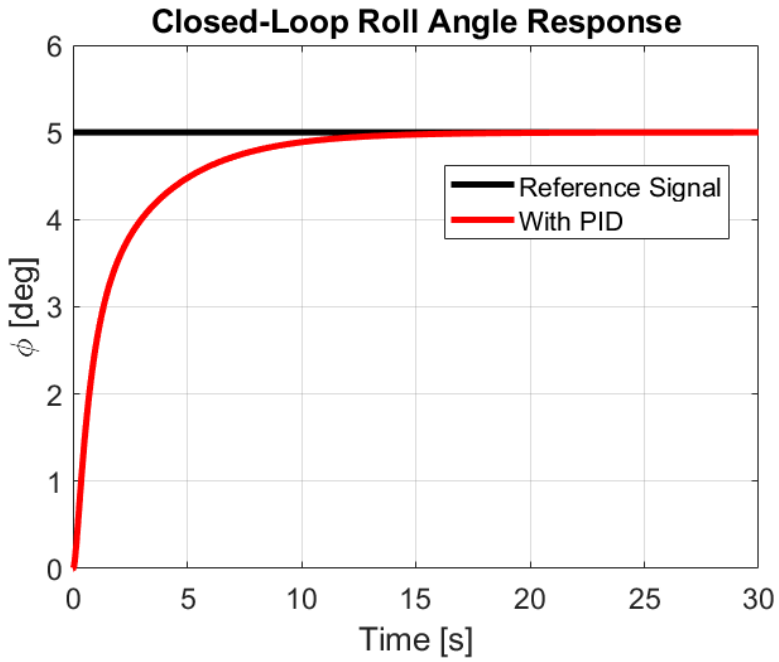

Figure 21 - Performance of roll angle hold autopilot

In Figure 21, the PID controller is able to demonstrate its simplicity and effectiveness as it is able to track the roll angle of 5 degrees within 15 seconds without any overshoot or oscillatory behavior. In order to do so, the controller was tuned to have a proportional control value of 0.05 and derivative action value of 0.1. These values help contribute to the smooth and damped response for the roll angle hold autopilot.

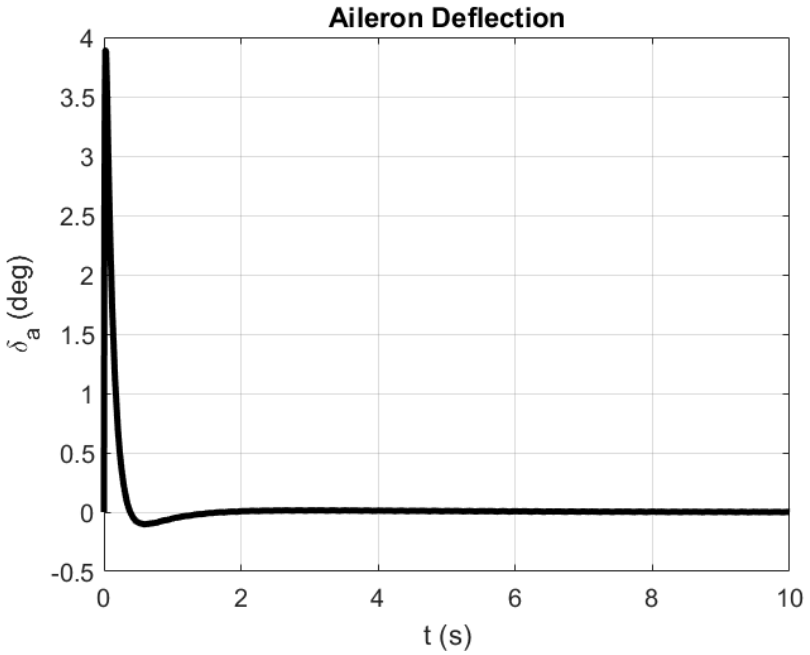

Figure 22 - Aileron deflection from roll angle autopilot

While the tuned PID controller was able to track the desired roll angle, it was able to correct the behavior of the aileron deflection as seen in Figure 22. Despite the deflection angle reaching nearly 4 degrees quickly, the controller was able to correct the aileron and settle to nearly 0 degrees as well. Since the roll angle and aileron are connected in regards to tuning and performance, these results are able to demonstrate the effectiveness of a tuned PID controller.

Chapter 6: Speed/Mach Hold Autopilot

6.1 LQG Control System

The development of maintaining a speed/mach hold autopilot to maintain steady flight for the Lockheed F-104 was done with a LQG controller as seen in Figure 23. The LQG controller was utilized for the autopilot since it has the capability to provide accurate performance despite noisy measurements being present during flight. Since the F-104 is traveling at a true airspeed of 1742 feet per second, accounted for within its derivatives, the LQG controller is set to track a reference signal of 0 feet per second. As seen in Figure 24, a tuned LQG controller is able to track the desired reference signal after 15 seconds.

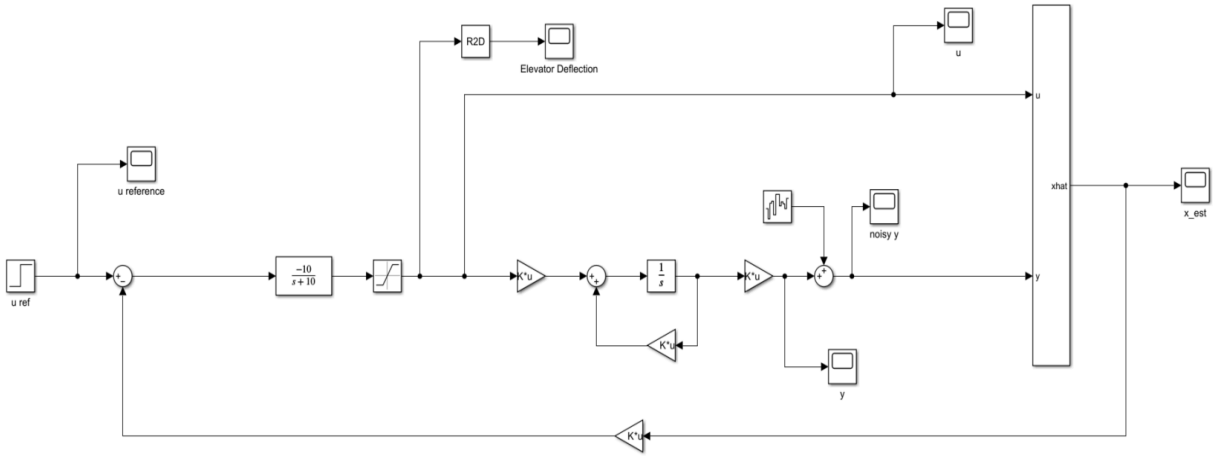

Figure 23 - LQG control system for speed/mach hold

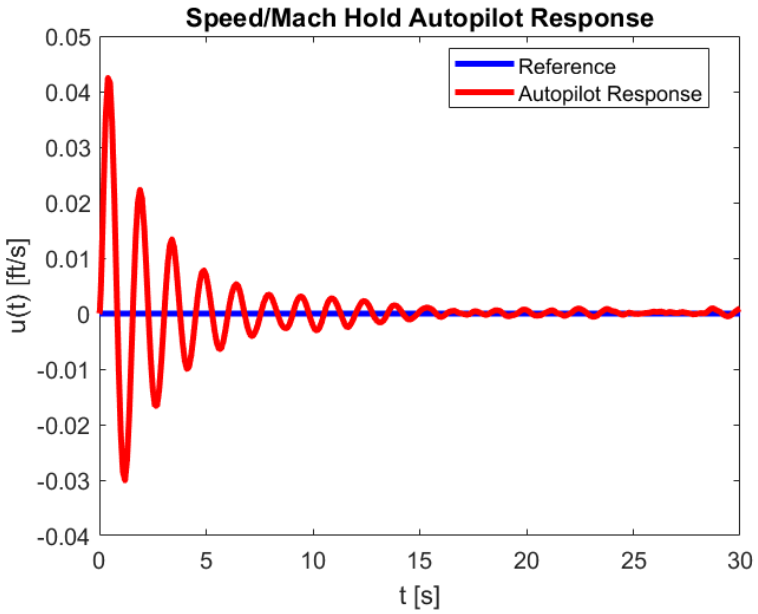

Figure 24 - Performance of speed/mach hold autopilot

The tuned LQG controller demonstrates the autopilot's ability to maintain the aircraft's speed within 15 seconds. Within Figure 24, it does not explicitly track the 1742 feet per second that the F-104 is traveling at. Rather, it measures the variation in speed. The tuned LQG controller does output an oscillatory response that does not disappear completely. However, the controller is able to dampen such oscillations within 15 seconds of flight, reducing the magnitude of varied speed from approximately 0.04 feet per second to eventually settling near the desired 0 feet per second. These results for the LQG controller were achieved by tuning the state and control effort weighting matrix, Q_c and R_c, to have an emphasis on penalizing the states of angle of attack and pitch rate. Within the specific Q_c diagonal matrix of [1, 500, 100, 1], the matrix was tuned in this manner in order to ensure steady flight for the F-104, which Figure 24 demonstrates steady flight was achieved.

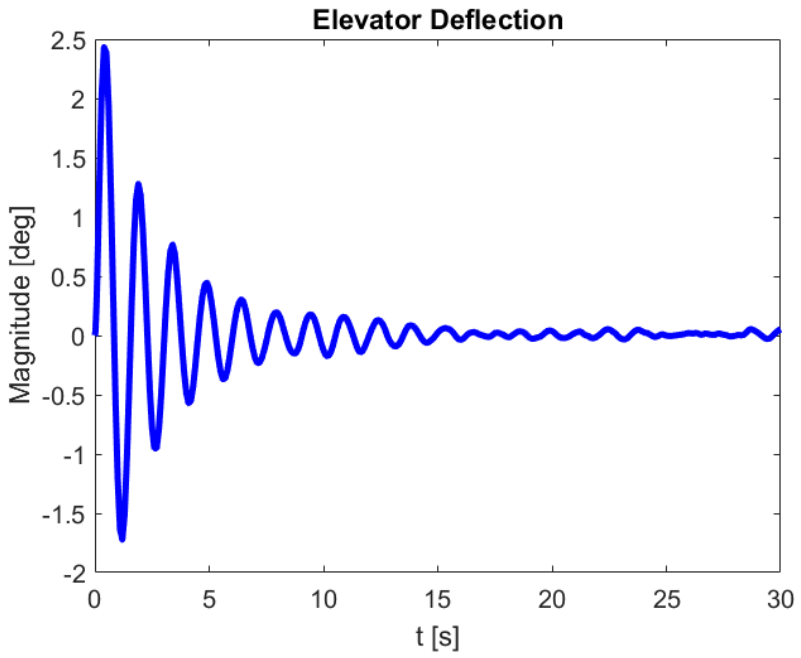

Figure 25 - Elevator deflection from speed/mach hold autopilot

In addition to the tuned LQG controller being able to track the desired reference speed, the elevator deflection for the F-104 is able to stay well within the deflection limits of +/- 20 degrees as seen in Figure 25. Similar to the speed/mach hold autopilot response, there is oscillatory behavior, peaking at 2.5 degrees. However, despite this behavior, it dampens to nearly 0 degrees after 15 seconds. Because of this response, it demonstrates the autopilot's ability to maintain the constraints the elevator deflection has and reinforces the performance of the autopilot as no drifting, erratic behavior is present in Figure 25.

Chapter 7: Heading Angle Hold Autopilot

7.1 LQG Control System

A heading angle hold autopilot was constructed with the usage of a LQG controller as seen in Figure 26. A LQG controller was chosen as the specified controller for the autopilot due to the same reasons as the speed/mach hold autopilot. Because of the flight conditions that the Lockheed F-104 is traveling at, a LQG controller is able to extract accurate responses despite external interference that the aircraft may experience. With such reasoning in mind, the controller was tasked with tracking a heading angle reference of 0 degrees since such angle is ideal for steady flight for the F-104. As seen in Figure 27, the LQG controller is able to track the desired 0 degrees reference angle.

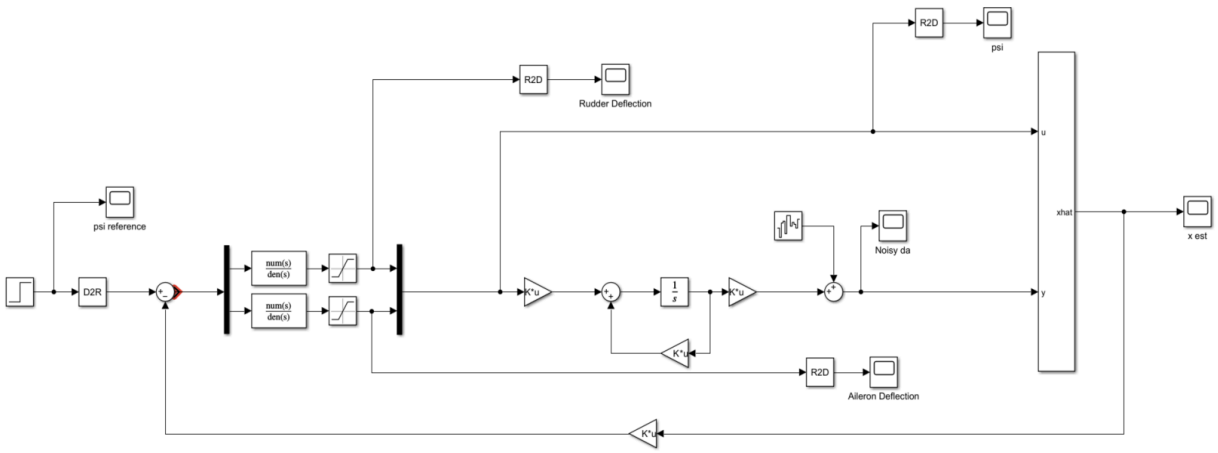

Figure 26 - LQG control system for heading angle hold

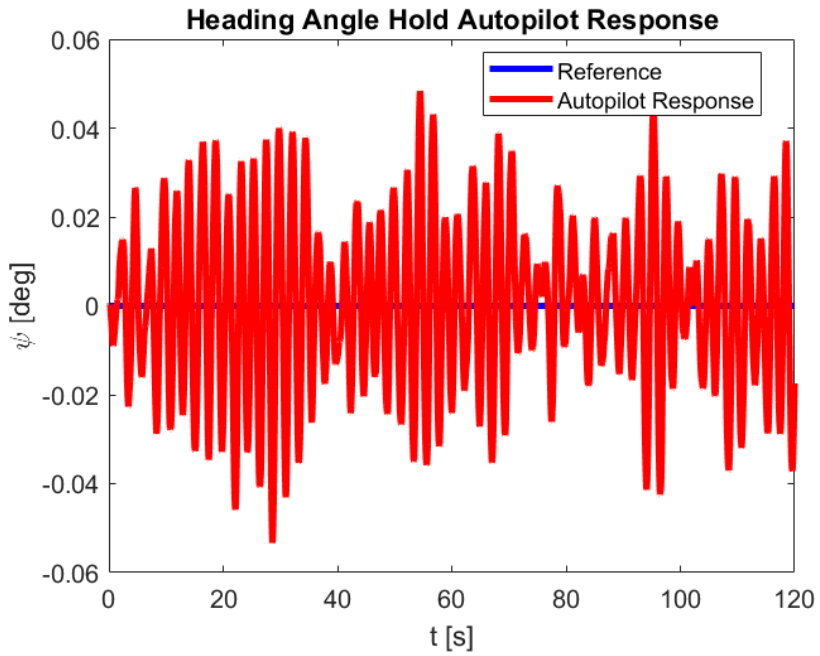

Figure 27 - Performance of heading angle hold autopilot

Figure 27 is able to demonstrate the functionality of the heading angle hold autopilot due to the low magnitudes the response exhibits. The LQG controller contains a tuned state weighting diagonal matrix , Q_c , of [0.5, 0.5, 1, 30, 10]. This state weighting diagonal matrix was tuned with an emphasis of suppressing yaw oscillations and tracking the desired heading angle. With this tuning, it allows the heading angle response to stay within +/- 0.05 degrees of the reference signal of 0 degrees. Because of the performance of the tuned LQG controller, the heading angle hold autopilot demonstrates its effectiveness in Figure 27 as there are no significant magnitude spikes exhibited within 120 seconds of operation.

In regards to the rudder and aileron deflections seen in Figures 28 and 29, they both stay well within their designated deflection limits of +/- 20 degrees. The rudder and aileron deflections exhibited are between +/- 0.05 degrees and 0.015 degrees respectively. These responses indicate that the rudder and ailerons of the F-104 are able to avoid unnecessary deflections while assisting in delivering the desired heading angle response.

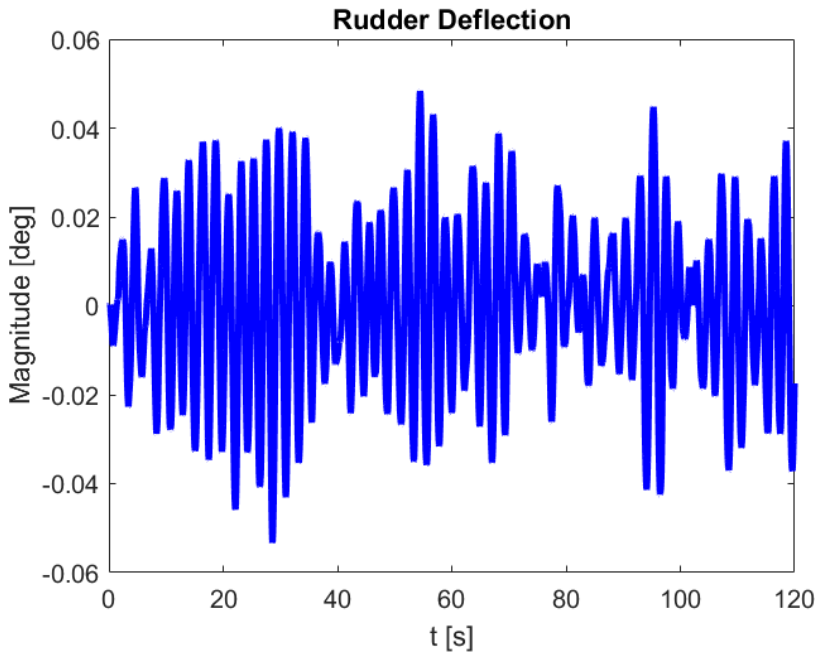

Figure 28 - Rudder deflection from heading angle hold autopilot

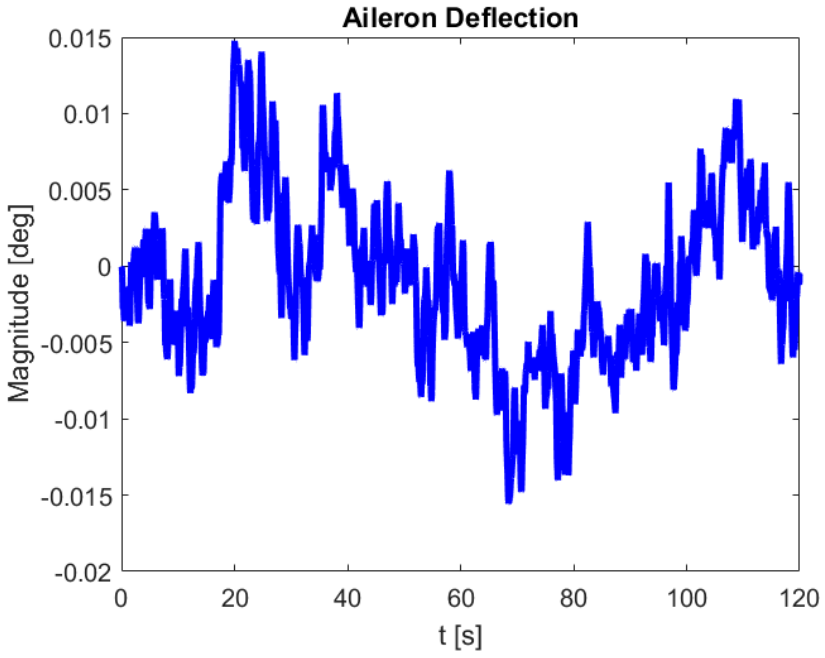

Figure 29 - Aileron deflection from heading angle hold autopilot

Chapter 8: Stability Augmentation System

8.1 Open-Loop Analysis

Due to the Lockheed F-104's thin wings, high wing loading, and difficulty with handling, the implementation of a Stability Augmentation System (SAS) is necessary for autonomous flight. In Figure 30, the F-104's longitudinal directional OL impulse response demonstrates oscillatory behavior within each of its states of forward velocity, angle of attack, pitch rate, and pitch angle. These states stabilize overtime with the exception of forward velocity, so the implementation of a pitch damper is needed to improve the F-104's response.

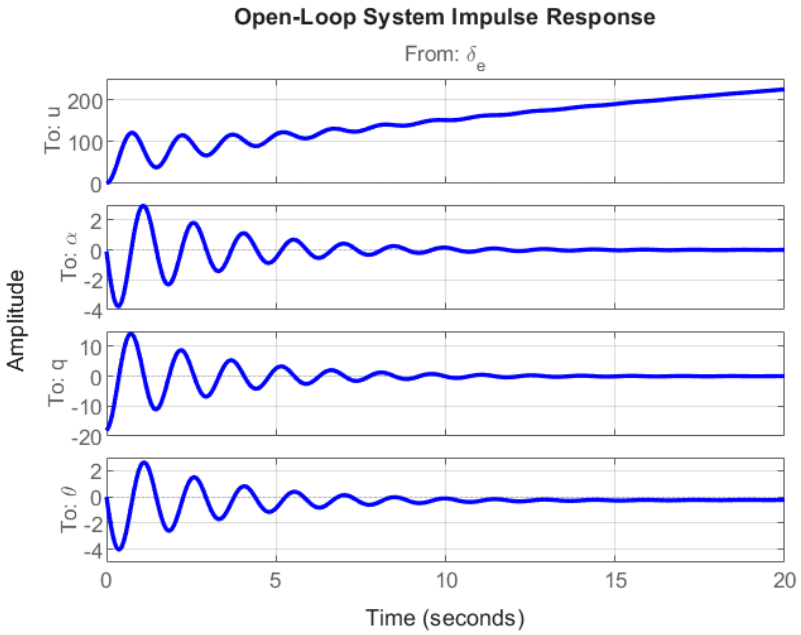

Figure 30 - Longitudinal directional OL impulse response

In regards to the lateral-directional OL impulse response in Figure 31, each states of roll angle, roll rate, side slip angle, yaw rate, and heading angle from the inputs of rudder and aileron deflection exhibit a lack of stability. The oscillations within each input and state combination reinforces the thought of lateral-directional instability of the Lockheed F-104, so yaw and roll damping is crucial to improve its flight performance. In order to improve the OL impulse response longitudinally and laterally, the yaw, pitch, and roll dampers will utilize the simplicity of tuned PID controllers to achieve desirable responses for the Lockheed F-104.

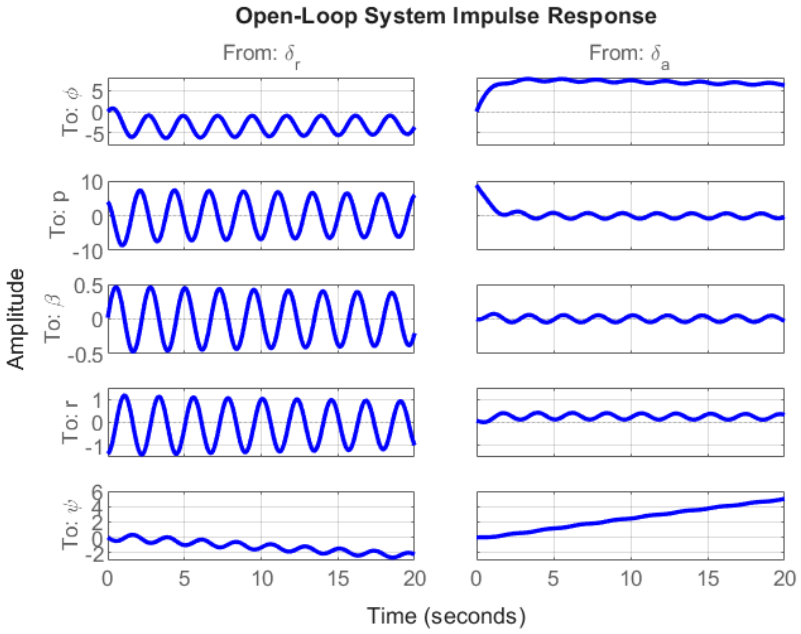

Figure 31 - Lateral-directional OL impulse response

8.2 Yaw Damper

The yaw damper utilizes only proportional and integral action from a PID controller. In addition to the control, a washout filter is implemented to assist in the damping of the impulse response via the input of the rudder. The constants used for proportional action, integral action, and the washout filter are: 1.5, 2, and 0.5 respectively. These values were chosen as it demonstrated the yaw damper's ability to suppress the oscillatory behavior seen from Figure 31 in Figure 32. In addition to suppressing oscillations, the yaw damper corrected the increasing drift behavior from the ailerons to the heading angle state.

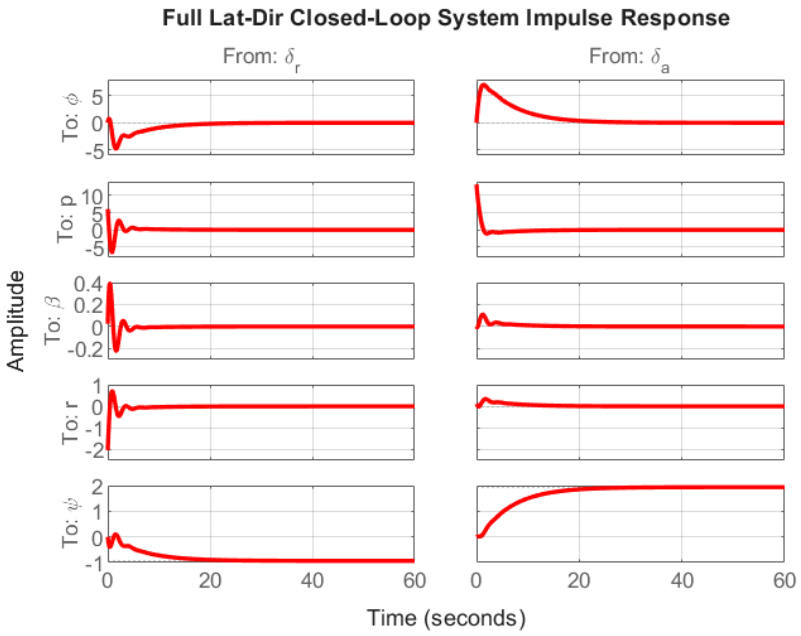

Figure 32 - Lateral-directional CL impulse response with yaw damper

8.3 Roll Damper

The roll damper is implemented within the F-104 with a PID controller as well. In contrast to the yaw damper, derivative action and the aileron input are necessary, but a washout filter serves no purpose for correcting roll. Within the PID controller used, the constants utilized for proportional, derivative, and integral action are: 2.2, 0.2, and 4. In Figure 33, these constants ensure that yaw and roll rates dampen to zero quickly, signifying the roll damper is functioning properly.

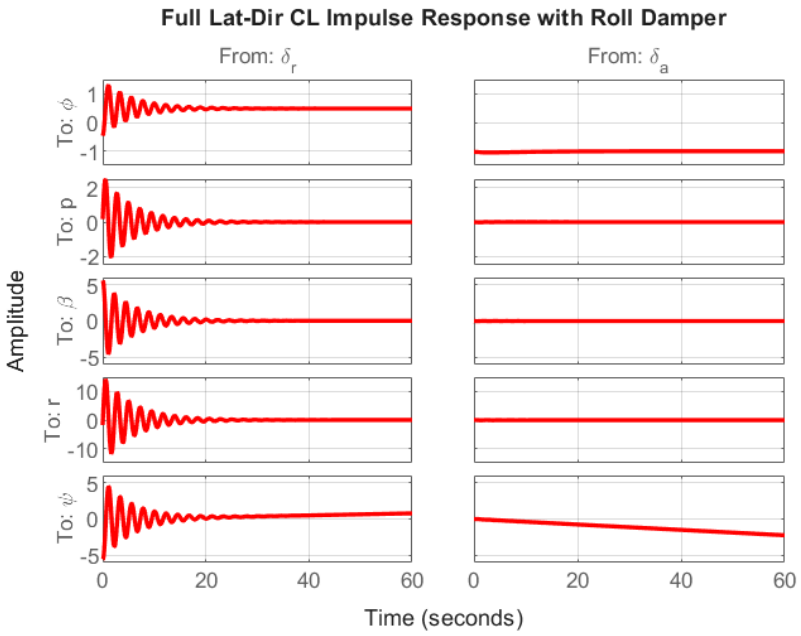

Figure 33 - Lateral-directional CL impulse response with roll damper

8.4 Pitch Damper

Similar to the yaw and roll dampers for the F-104, the pitch damper is integrated via a PID controller, utilizing all three actions in addition to a washout filter with the elevator as the only input. With the PID controller for the pitch damper, its constants for its actions and washout filter constant are respectively: proportional action of 25, integral action of 0.5, derivative action of 2, and a washout filter constant of 0.4. With this tuned PID controller, in Figure 34, pitch rate is able to be dampened to zero within 15 seconds. In addition, the angle of attack and pitch angle converge to zero degrees, demonstrating the effectiveness of the pitch damper via PID controller.

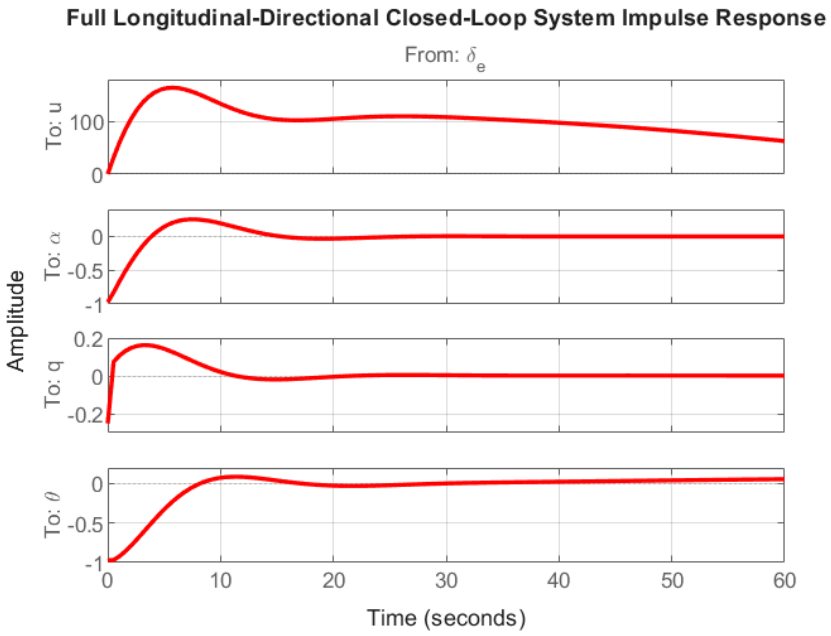

Figure 34 - Longitudinal directional CL impulse response with pitch damper

Chapter 9: Control Augmentation System

Upon completion of the various autopilots and Stability Augmentation Systems for the Lockheed F-104, the implementation of Control Augmentation Systems (CAS) are introduced in order to enhance the aircraft's controllability and maneuvering under its flight conditions. CAS implementation enables the ability to improve responsiveness by having direct control over the aircraft's dynamic states. Within this chapter, the usage of controllers such as LQR and dynamic inversion assist in addressing the longitudinal and lateral-directional control challenges that the Lockheed F-104 may have.

9.1 Roll Rate CAS

The roll rate CAS was developed using a LQR controller in order to improve the roll rate tracking performance of the Lockheed F-104 as seen in Figure 35. In designing the controller, reference signals were implemented to mimic the aircraft performing various maneuvers such as steady flight, rolling right, and rolling left when commanded. Additionally, its state weighting matrix Q was tuned to be a diagonal matrix of [1, 50, 1, 1, 200] in order to place significant emphasis on the roll rate and heading angle states. As for the control effort matrix R, it was tuned to be a diagonal matrix of [2, 2] with the purpose of having moderate penalties of rudder and aileron usage. With these tuning efforts, Figure 36 is able to demonstrate the LQR controller's ability to track the designated roll rate inputs.

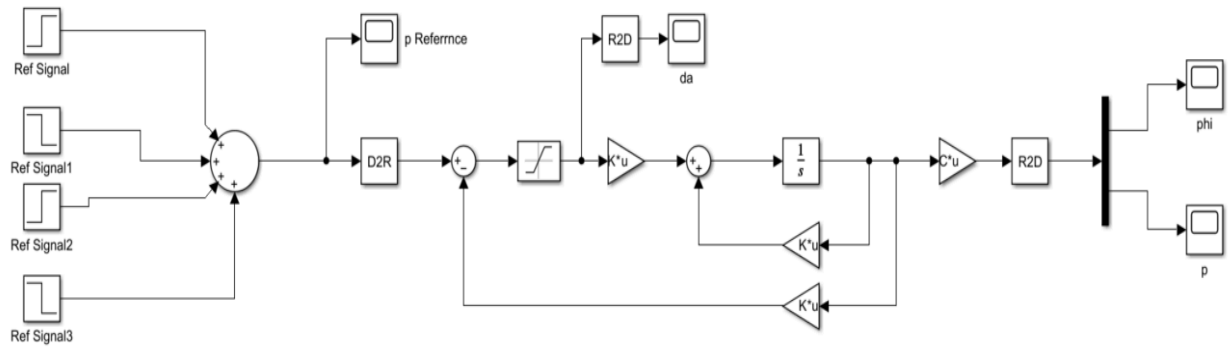

Figure 35 - LQR control system for roll rate CAS

Figure 36 is able to depict the LQR controller effectively tracking the commanded roll rate inputs consisting of various step variations between +/- 15 degrees per second. Additionally, the roll rate response is able to track such variations with no overshoot and oscillatory behavior. Because of this roll rate response, Figure 37 is able to depict the resulting roll angle for the F-104. It can be seen that the roll angle of the aircraft has brief oscillatory behavior, but such behavior is dampened after 20 seconds.

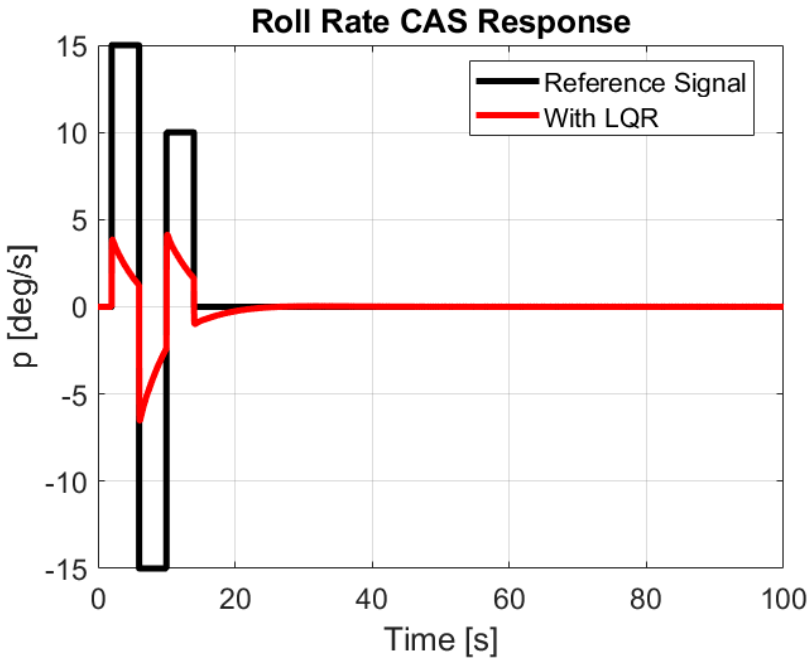

Figure 36 - Performance of roll rate CAS

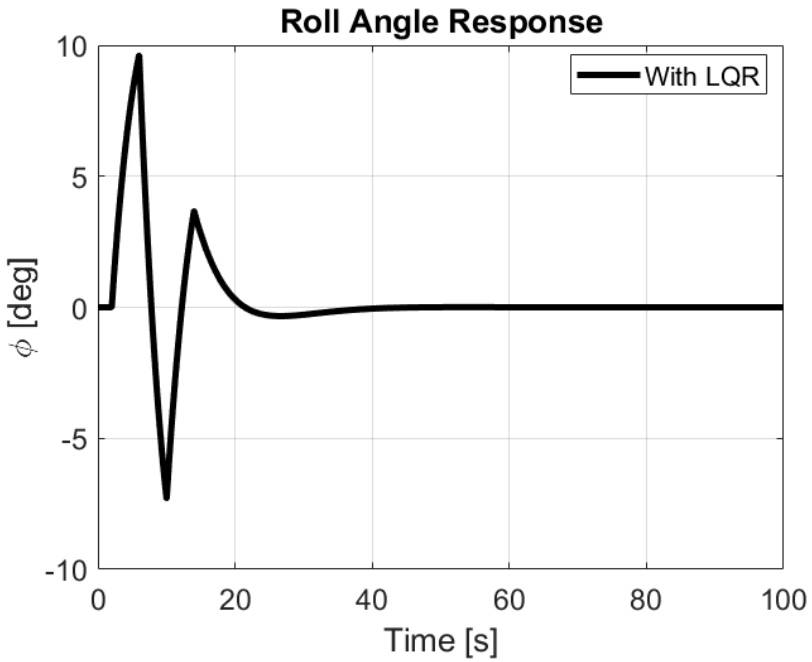

Figure 37 - Performance of roll angle due to roll rate CAS

With the roll rate maneuvering and resulting roll angle response from Figures 36 and 37, it has resulted in the aileron deflection to reach its maximum deflection limit in +/- 20 degrees in Figure 38. However, despite reaching such limits, this was only the case due to the aircraft making the appropriate rolling maneuver, thus the aileron reacting accordingly. Overall, the

performance of the LQR-based roll rate CAS demonstrates stable and responsive performance that is desired for the Lockheed F-104.

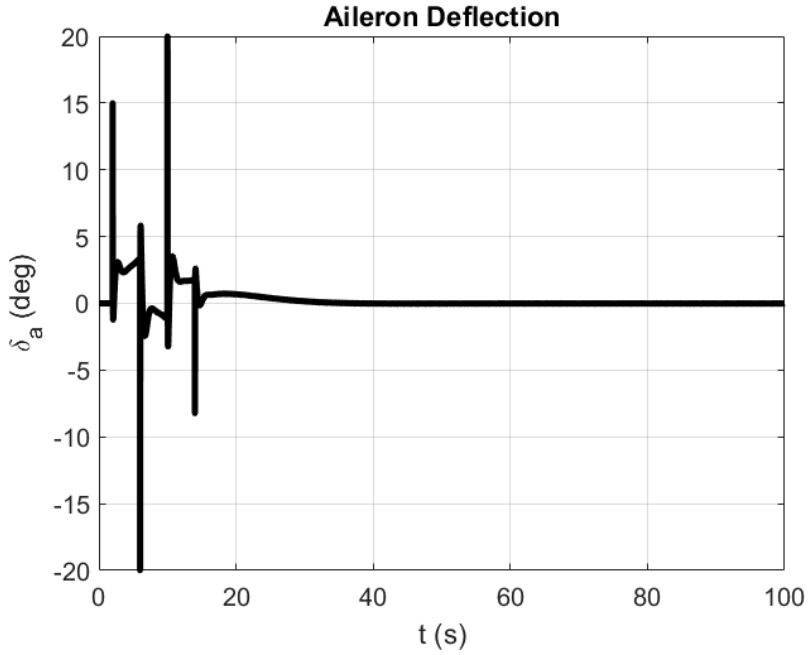

Figure 38 - Aileron deflection from roll rate CAS

9.2 Normal Acceleration CAS

The normal acceleration CAS was designed to improve the Lockheed F-104's longitudinal maneuvering performance by directly controlling the normal acceleration n_z via a dynamic inversion controller as seen in Figure 39. Within the longitudinal states, there is not an explicit state where normal acceleration can be individually outputted. Therefore, a proxy output equation is used as a form of approximating normal acceleration. Equation 9.1 would be the proxy output equation used, utilizing angle of attack as the primary state that contributes to normal acceleration.

$$n_z = \frac{Z_\alpha \alpha}{g} \tag{9.1}$$

With the proxy output equation established, it is then utilized when establishing matrix C to be $[0\ \frac{Z_\alpha}{g}\ 0\ 0]$. The dynamic inversion controller is then tuned with a K value of 3 in order to ensure appropriate tracking of the F-104 performing maneuvers such as either pulling the aircraft up or down. As seen in Figure 40, the normal acceleration maneuvers are able to be tracked by the controller.

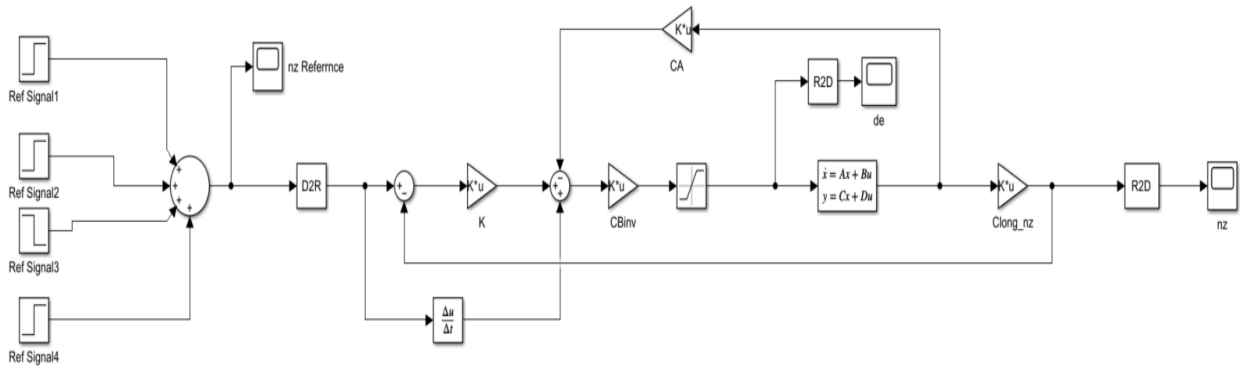

Figure 39 - Dynamic inversion control system for normal acceleration CAS

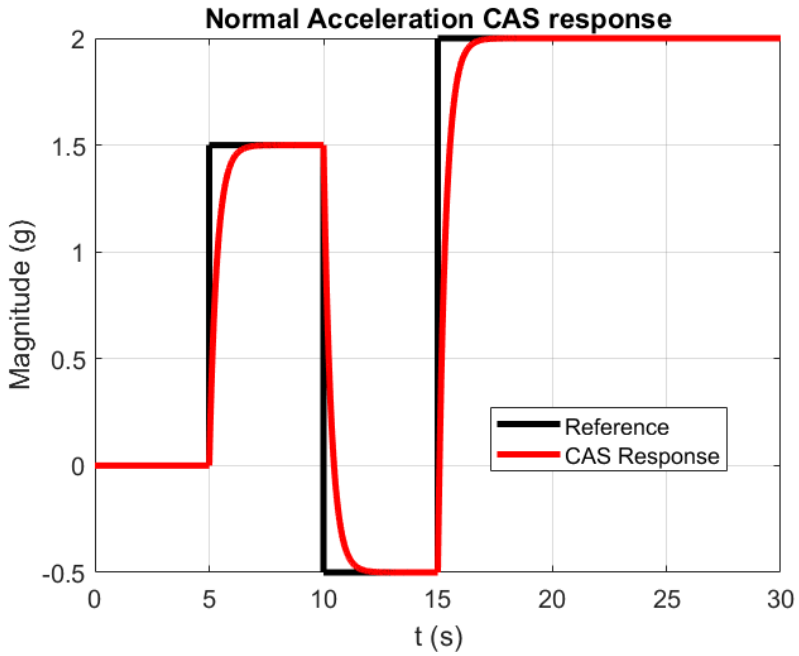

Figure 40 - Performance of normal acceleration CAS

Figure 40 indicates that the dynamic inversion-based normal acceleration CAS is able to track the various multi-step reference inputs as if it was nearly following the reference inputs every second of the simulation. In regards to the elevator deflection in Figure 41, it spikes to 20 degrees when the F-104 is performing its next maneuver in the reference steps. Otherwise, there is nearly no elevator usage by the aircraft. With these results, it demonstrates the normal acceleration CAS' longitudinal responsiveness for the Lockheed F-104 while being able to offer reliable normal acceleration tracking during the aircraft's high-speed flight conditions of Mach 1.8.

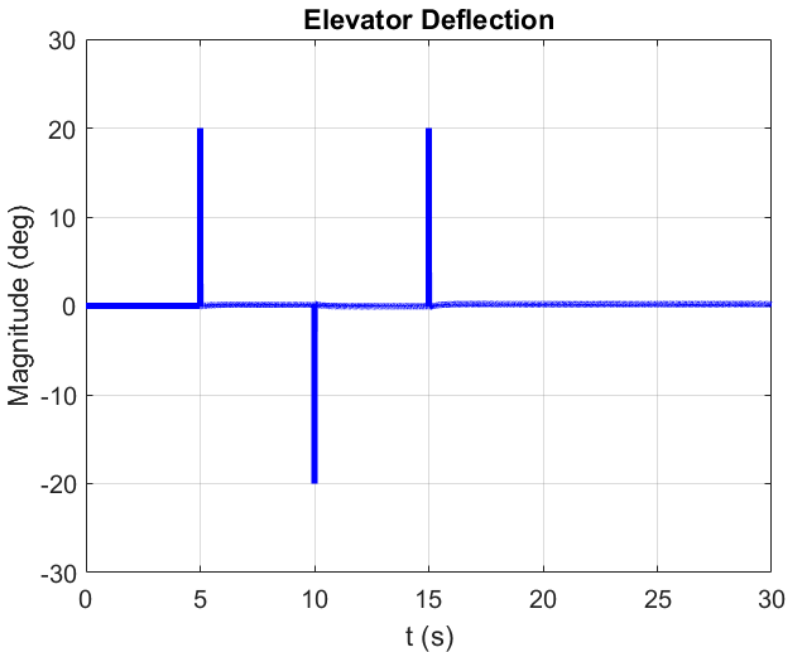

Figure 41 - Elevator deflection from normal acceleration CAS

9.3 Lateral-Directional CAS

The last CAS being designed, the lateral directional CAS, takes advantage of a dynamic inversion controller in order to regulate the yaw rate and sideslip angle of the F-104. By doing so, the dynamic inversion simulink seen in Figure 42 enables the ability to improve the aircraft's responsiveness in coordinated lateral maneuvers. The controller intakes multiple yaw rate step reference signals to simulate the aircraft performing aggressive or deliberate maneuvers in flight. It is tuned to have a diagonal controller gain matrix K of [1.75, 0.01]. The dynamic inversion controller is tuned in this fashion in order to place strong emphasis on yaw rate tracking while minimizing the control effort of the side slip angle. As seen in Figure 43, it depicts the CAS being able to track the multitude of yaw rate reference signals.

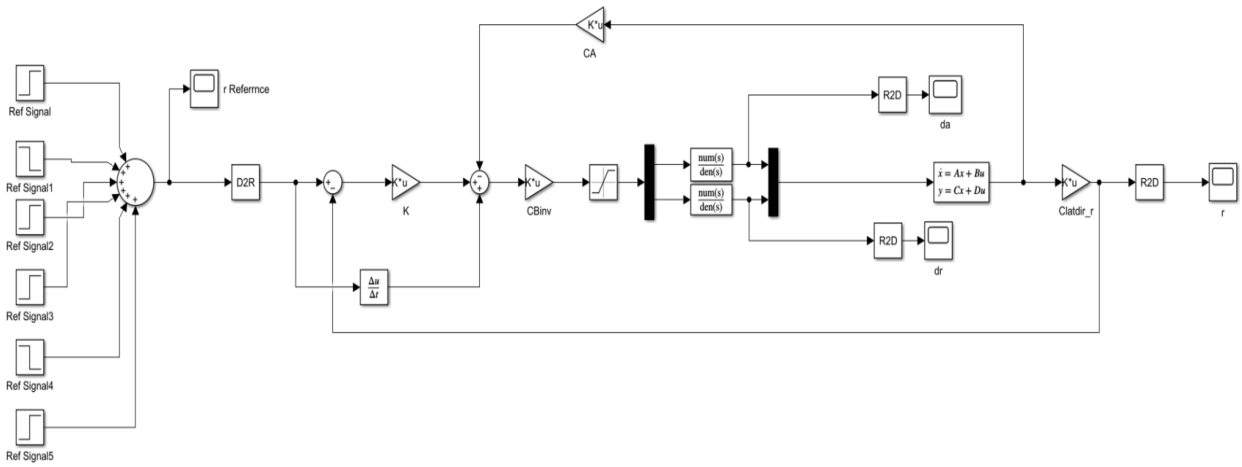

Figure 42 - Dynamic inversion control system for lateral directional CAS

Figure 43 demonstrates how the CAS is able to track the various yaw rate reference signals from +/- 10 degrees per second with minimal overshoot. The CAS smoothly tracks such reference signals, demonstrating the effectiveness of a dynamic inversion-based lateral directional CAS.

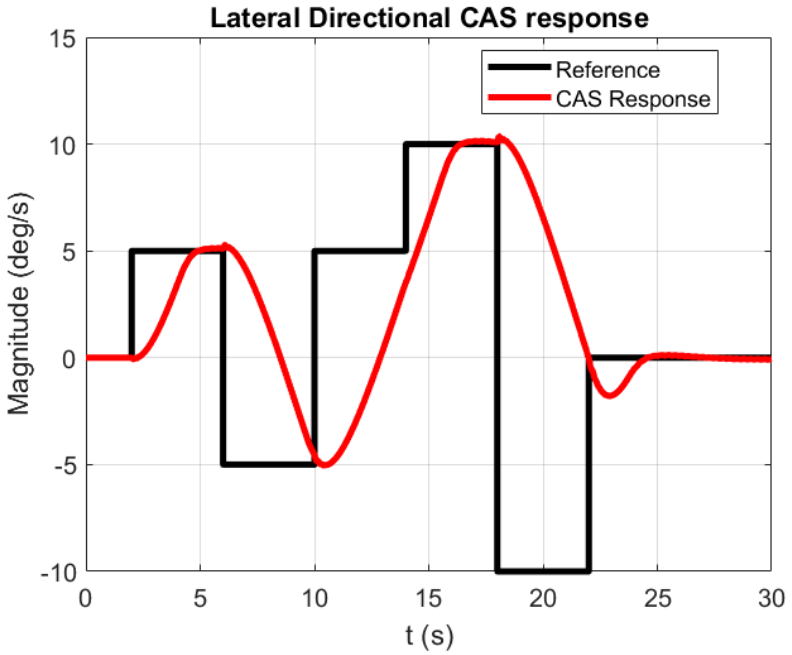

Figure 43 - Performance of lateral directional CAS

In regards to the rudder and aileron deflections, Figures 44 and 45 depict that they operate within the deflection limits of +/- 20 degrees. The rudder deflection peaks at such a limit of 20 degrees. In contrast, the aileron deflection peaks approximately at 7.5 degrees. While the behavior of the rudder and aileron seem concerning, especially the rudder, these behaviors are expected due to the changes in the yaw rate reference signals provided. In addition, these

deflections still operate within their designated limits. In totality, a dynamic inversion-based lateral directional CAS demonstrates its robustness in regulating the Lockheed F-104's yaw rate and sideslip angle. With this performance, the aircraft can intake various yaw rate inputs for coordinated turns, and it will respond accordingly.

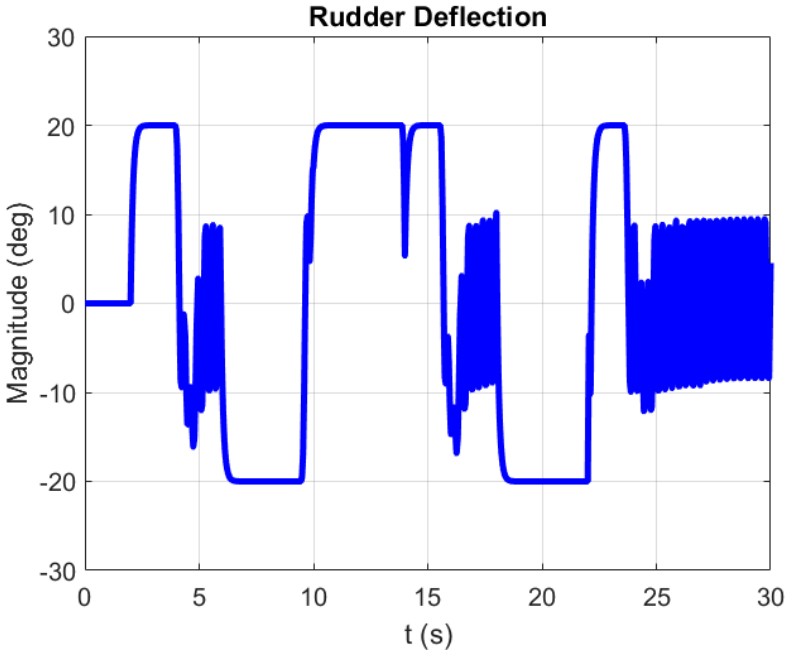

Figure 44 - Rudder deflection from lateral directional CAS

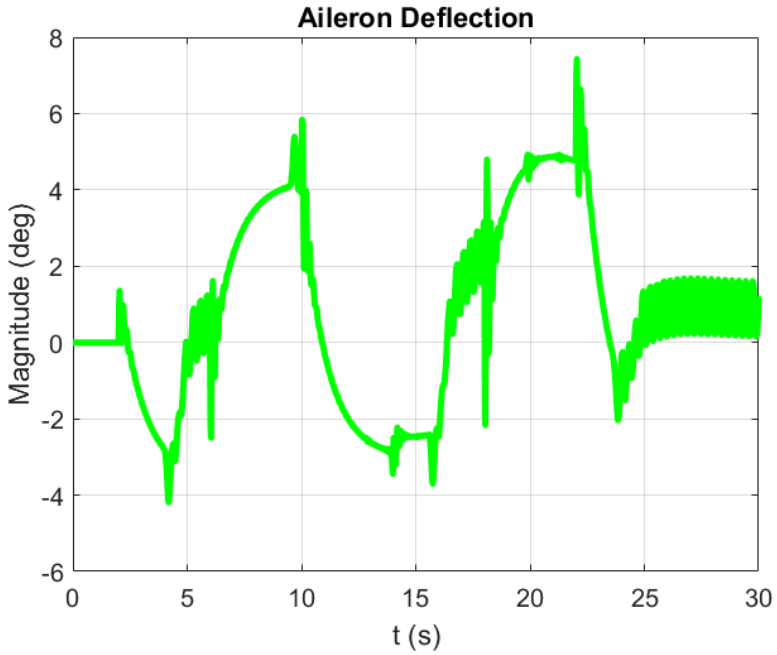

Figure 45 - Aileron deflection from lateral directional CAS

Chapter 10: Discussion

10.1 Importance of SAS, CAS, and Autopilots

Due to the Lockheed F-104's marginal stability and control difficulties at supersonic speeds, it necessitates the need for Stability Augmentation Systems, Control Augmentation Systems, and various autopilot systems. Originally, the F-104 was equipped with a SAS and other autopilot systems to address such instabilities as noted by Stoelinga [17]. However, these systems are limited in their responsiveness. Additionally, more than 50% of all F-104 accidents were due to outdated technology and their flight conditions [18]. With this known information, the SAS, CAS, and autopilot systems designed addresses the deficiencies that Lockheed F-104 may have by improving its damping, control precision, and autonomous tracking capabilities.

10.2 MATLAB/Simulink Results and Analysis

10.2.1 Pitch-Attitude Hold Autopilot

The pitch-attitude hold autopilot was developed with the usage of a dynamic inversion controller, demonstrating the controller's ability to stabilize the F-104's angle of attack and pitch rate within three seconds as seen in Figure 17. In comparison to the aircraft's OL behavior with its sustained oscillations, the CL system does not depict any oscillations at all. This improvement is significant as the OL response revealed short-period oscillation and pitch instability when the aircraft is traveling at Mach 1.8. WIth the assistance of the dynamic inversion controller to stabilize the angle of attack and pitch rate dynamically, it enables more reliable cruise and climbing maneuvers. Such assistance would also reduce pilot workload during aggressive pitch inputs, supporting safer flight operations as a result.

10.2.2 Altitude Hold Autopilot

The altitude hold autopilot utilizes an LQG controller in order to successfully maintain the F-104's altitude of 55,000 feet as seen in Figure 19. The controller was able to filter out external disturbances and be able to track with commanded altitude with the assistance of a Kalman Filter state estimator. This is crucial for flight operation as the F-104 has a low static margin and is sensitive to small pitch changes. By implementing altitude hold autopilot, it enables the aircraft's ability to maintain steady supersonic cruise without constant pilot correction, potentially improving overall mission endurance.

10.2.3 Roll Angle Hold Autopilot

A PID controller was utilized for a roll angle hold autopilot to track a commanded 5 degree roll angle with no overshoot or oscillatory behavior as depicted in Figure 21. With the controller's proportional and derivative tuning, it allowed a smooth and damped response for the roll angle while maintaining aileron deflections within operational limits. The implementation of such autopilot proves useful as the F-104 would have fast roll rates but poor spiral stability. This autopilot improves the aircraft's ability to perform precision banking maneuvers significantly, assisting in navigation or coordinated turns during supersonic flight.

10.2.4 Speed/Mach Hold Autopilot

Taking advantage of the capabilities of an LQG controller, the speed/mach hold autopilot is able to track speed variations from the F-104's true airspeed as seen in Figure 24. The aircraft being able to maintain a stable Mach number is necessary in order to prevent aerodynamic instabilities with potential drag, ensuring engine performance at such altitudes and speeds. This development of a speed/mach hold autopilot system reduces pilot burden and safe operations of the Lockheed F-104 during high-speed cruise as a result.

10.2.5 Heading Angle Hold Autopilot

Utilizing a tuned LQG controller, the heading able hold autopilot is able to suppress yaw oscillations and maintain a heading angle within 0.05 degrees of its commanded value of 0 degrees as seen in the autopilot's response in Figure 27. Due to the F-104's poor lateral-directional stability, the implementation of a heading angle hold autopilot would allow improved mission capability to navigate courses autonomously over a long duration without continuous pilot correction.

10.3 Stability Augmentation System Performance

The SAS systems were implemented into the Lockheed F-104 using PID controllers to address the aircraft's dynamic instability both longitudinal and lateral-directionally. As seen within the yaw, pitch, and roll dampers in Figures 32 to 34, these dampers are able to address such stability deficiencies. The yaw damper was able to suppress yaw rate oscillations that were caused by Dutch roll damping. The roll damper corrected spiral divergence and oscillatory roll responses. The pitch damper suppressed the phugoid mode and short-period oscillations longitudinally. WIth SAS implementation, it reduces the aircraft's risk to accidents when flying during turbulence, maneuvering, and high-altitude flight.

10.4 Control Augmentation System Performance

The CAS system builds upon the SAS systems, providing enhanced tracking capabilities and control authority in roll rate, normal acceleration, and lateral-directional as seen in Figures 36 to 45. It greatly enhances maneuverability, tracking precision, and controllability within the flight conditions that the Lockheed F-104 may endure. The roll rate CAS is able to follow commanded roll rates, improving coordinated roll maneuvers. The normal acceleration CAS regulates the F-104's angle of attack. The lateral-directional CAS improves yaw rate and sideslip angle control, assisting in coordinated turns and course corrections as well.

10.5 Additional Use Cases

With this project, the primary objective was to enhance the control systems of the Lockheed F-104, demonstrating autonomous capabilities at supersonic flight conditions. While such an objective was accomplished via various controllers, the design architecture and methodologies applied within this project could be utilized with broader use cases relevant to current and future aerospace applications.

A potential application would be the retrofitting of legacy aircrafts with modern control systems. The Lockheed F-104 was first introduced in 1958, and this project demonstrates that legacy aircrafts have the potential to have its control systems be upgraded. With such modernized upgrades, many other legacy aircrafts would have a pathway to enhanced stability, maneuverability, and autonomous functions without structural redesigns.

Along with retrofitting, these control systems are able to lay the foundation for further supersonic UAV development. As the aerospace industry continues to explore high-speed autonomous platforms for either commercial or non-commercial purposes, the ability to integrate various autopilot systems into an aircraft will be crucial. This foundation can then allow for testing of advanced concepts such as the usage of AI, recovery strategies, hypersonic UAV control designs, and other concepts that have yet to be imagined [3, 11].

Due to the flexibility and extensibility of the various control system designs, it demonstrates aerospace applications beyond the Lockheed F-104. Whether these designs are used for further research, industry implementations, or educational purposes, there are additional uses for such control systems outside of the used aircraft in this project.

Chapter 11: Conclusion

The development of an autonomous, supersonic UAV control system based on the Lockheed F-104 demonstrates the ability to enhance legacy aircrafts through modern control system design methodologies. With the implementation of PID, LQR, LQG, and dynamic inversion controllers, it was shown that the stability and control challenges that the F-104 may have at Mach 1.8 can be addressed and mitigated.

With an OL analysis of the F-104, the necessity of Stability Augmentation Systems and Control Augmentation Systems were crucial as the OL response revealed marginal dynamic stability characteristics. SAS implementation of yaw, pitch, and roll dampers were able to suppress Dutch roll and phugoid oscillations that were present, thus improving the stability of the aircraft. As for the CAS designs, they were able to improve the F-104's maneuverability and responsiveness with its tracking of commanded inputs of roll rate, normal acceleration, and lateral directional dynamics.

The various autopilot systems constructed in pitch-attitude hold, altitude hold, roll angle hold, speed/mach hold, and heading angle hold provided a baseline for autonomous flight of the Lockheed F-104. These autopilots were able to address the demands that supersonic flight may have, enabling the F-104 to maintain steady flight without continuous pilot corrections.

Overall, integrating the various autopilot systems, SAS, and CAS into a supersonic UAV results in an aircraft that is capable of autonomous flight while improving safety, stability, and maneuverability under such flight conditions of Mach 1.8 at an altitude of 55,000 feet. The methodologies and results presented provide a foundation for future research into developing high-speed UAVs through modern control system techniques, ultimately contributing to the advancement of supersonic technology.

References

[1] Liu, L., “Design of UAV Flight Control Law Based on PID Control,” *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML),* Stanford, CA, USA, 2021, pp. 98-101. https://doi.org/10.1109/CONF-SPML54095.2021.00028.

[2] Turkoglu, K., Ozdemir, U., Nikbay, M., and Jafarov, E., “PID Parameter Optimization of an UAV Longitudinal Flight Control System,” *International Journal of Mechanical, Industrial Science and Engineering*, 2008, pp. 1031-1036. https://doi.org/10.5281/zenodo.1079352.

[3] Ghelem, N., Boudana, D., and Bouchhida, O., “UAV longitudinal autopilot design using SLC and TECS controllers,” *CEAS Aeronautical Journal,* Vol. 15, 2024, pp. 351-362. https://doi.org/10.1007/s13272-023-00711-9.

[4] Ding, W., Quan, L., Li, S., Zhang, T., Lu, Y., and Lu, L., “Design and Simulation of Flight Control System for Quadrotor UAV,” *2023 5th International Conference on Intelligent Control, Measurement and Signal Processing,* 2023, pp. 237-240. https://doi.org/10.1109/ICMSP58539.2023.10170982.

[5] Liu, Z., Huang, J., Shi, F., Wu, J., and Ren, Y., “Arduino-based fixed-wing UAV control system design and implementation,” *Proceedings of SPIE,* Naval Aeronautical University, China, 2023, pp. 129211S-20. https://doi.org/10.1117/12.2688632.

[6] Ingabire, A., and Sklyarov, A.A., “Control of longitudinal flight dynamics of a fixed-wing UAV using LQR, LQG and nonlinear control,” *E3S Web of Conferences,* Vol. 104, 2019, pp. 02001. https://doi.org/10.1051/e3sconf/201910402001.

[7] Langston, S., “Low-Speed Stability and Control of Exploratory Tailless Long-Range Supersonic Configurations,” *ProQuest Dissertations & Theses,* 2015.

[8] Setiawan, M.N., Mustakim, M., Ananda, M.K., Hondro, W.A., Ramadhan, H., and Munadi, M., “The Development of Stability Augmentation System of Unmanned Aerial Vehicle Based on Linear Quadratic Gaussian,” *2022 9th International Conference on Information Technology, Computer, and Electrical Engineering,* 2022, pp. 13-18, https://doi.org/10.1109/ICITACEE55701.2022.9924138.

[9] Hanif, A., and Sasongko, R.A., “Attitude and flight path control systems of unmanned combat aerial vehicle,” *IOP Conference Series: Materials Science and Engineering,* Vol. 645, 2019, pp. 012019. https://doi.org/10.1088/1757-899X/645/1/012019.

[10] Tran, A., Sakamoto, N., and Muraoka, K., “Control Augmentation System Design for Quad-Tilt-Wing Unmanned Aerial Vehicle via Robust Output Regulation Method,” *IEEE Transactions on Aerospace and Electronic Systems,* Vol. 53, 2017, pp. 357-369. https://doi.org/10.1109/TAES.2017.2650618.

[11] Yadav, A.K., and Gaur, P., "AI-based adaptive control and design of autopilot system for nonlinear UAV," *Academy Proceedings in Engineering Sciences,* Vol. 39, 2014, pp. 765-783. https://doi.org/10.1007/s12046-014-0275-0.

[12] Osman, M.A.A., Abdalla, H.E.A., and Nawari, M.O., "Lateral and Longitudinal Controllers Design for a Fixed Wing UAV," *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering,* 2019, pp. 1-6. https://doi.org/10.1109/ICCCEEE46830.2019.9071049.

[13] Ueba, M., Takaku, Y., Takahashi, K., and Kamata, T., "Design and Tests of Guidance and Control Systems for Autonomous Flight of a Low-Speed Model Airplane for Application to a Small-Scale Unmanned Supersonic Airplane," *Transactions of the Japan Society for Aeronautical and Space Sciences,* Vol. 17, Aerospace Technology Japan, 2019, pp. 220-226. https://doi.org/10.2322/tastj.17.220.

[14] Lee, D., Kim, S., and Suk, J., "Formation flight of unmanned aerial vehicles using track guidance," *Aerospace Science and Technology*, Vol. 76, 2018, pp. 412-420. https://doi.org/10.1016/j.ast.2018.01.026.

[15] Letizia, N.A., Salaamat, B., and Tonello, A.M., "A New Recursive Framework for Trajectory Generation of UAVs," *2020 IEEE Aerospace Conference*, 2020, pp. 1-8. https://doi.org/10.1109/AERO47225.2020.9172656.

[16] Yechout, T., Morris, S., Bossert, D., Hallgren, W., and Hall, J., "Introduction to Aircraft Flight Mechanics," *American Institute of Aeronautics and Astronautics, Inc.,* 2014, pp. 665-680. https://doi.org/10.2514/5.9781624102547.0665.0680.

[17] Stoelinga, T., "Information regarding the Lockheed F-104 Starfighter F-104 Flight Controls," *Zipper Magazine*, Vol. #46, 2001.

[18] Esser, S., and Ruff-Stahl, H.J., "An HFACS Analysis of German F-104 Starfighter Accidents," *Journal of Aviation Technology and Engineering,* 2020, pp. 19-34. https://doi.org/10.7771/2159-6670.1218.

Appendices

Appendix A - Pitch Attitude Hold Autopilot Code

```
%% AE 295 Project, Pitch Attitude Hold Code
%% F-104 @ 55000 ft and M = 1.8
clear; clc;

% Parameters
g = 32.17405; % [ft/s^2] Gravitational Acceleration
U1 = 1742; % [ft/s] True Airspeed
tf = 400; % [s] Simulation Time
%% Longitudinal Directional system
% Stability and Control Derivatives
Xu = -0.0049;
Xa = -32.4692;
Zu = -0.0176;
Za = -346.6128;
Mu = 0;
Ma = -18.1248;
Madot = -0.0783;
Mq = -0.1844;
Xde = 0;
Zde = -87.9865;
Mde = -18.1525;
%% Longitudinal State-Space system
Along = [Xu                Xa                0          -g;...
      Zu/U1              Za/U1             1          0;...
      Mu+(Madot*Zu)/U1   Ma+(Madot*Za)/U1  Mq+Madot   0;...
      0                  0                 1          0];

Blong = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0];

Clong = eye(length(Along));

Dlong = zeros(size(Clong,1),size(Blong,2));

Clong_u = [1 0 0 0]; % Output Forward Velocity
Clong_alpha = [0 1 0 0]; % Output Angle of Attack
```

```
Clong_q = [0 0 1 0]; % Output Pitch Rate
Clong_theta = [0 0 0 1]; % Output Pitch Angle
%% Define OL SS system
longsys = ss(Along,Blong,Clong,Dlong);
%% Check Controllability and Observability of AOA, Pitch Rate, Pitch Angle
% Controllability
Co = ctrb(Along, Blong);
rankCo = rank(Co);
disp('rankCo = ');
disp(rankCo);

% Observability
Ob = obsv(Along, Clong);
rankOb = rank(Ob);
disp('rankOb = ');
disp(rankOb);

alphaOb = obsv(Along, Clong_alpha);
alpharankOb = rank(alphaOb);
disp('alpha rankOb = ');
disp(alpharankOb);

qOb = obsv(Along, Clong_q);
qrankOb = rank(qOb);
disp('q rankOb = ');
disp(qrankOb);

thetaOb = obsv(Along, Clong_theta);
thetarankOb = rank(thetaOb);
disp('theta rankOb = ');
disp(thetarankOb);
%% Pitch Attitude Hold (Dynamic Inversion)
% Compute matrices required for Dynamic Inversion
CBinv_alpha = inv(Clong_alpha*Blong);
CA_alpha = Clong_alpha*Along;

CBinv_q = inv(Clong_q*Blong);
CA_q = Clong_q*Along;

CBinv_theta = inv(Clong_theta*Blong);
```

```
CA_theta = Clong_theta*Along;

% Define controller gain
K = 5;

% Start the Simulink simulation
open_system('AE295_DynamicInversion_PitchAttitudeHold_KG.slx');
sim('AE295_DynamicInversion_PitchAttitudeHold_KG.slx');

% Plot results
figure,
plot(ans.de(:,1),ans.de(:,2),'k')
xlabel('t [s]');
ylabel('\deltae [deg]');
title('Elevator Input');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

figure,
subplot(1,2,1)
plot(ans.AOAdisturbance(:,1),ans.AOAdisturbance(:,2),'k')
hold on
plot(ans.alphaCL(:,1),ans.alphaCL(:,2),'r')
legend('Disturbance Signal', 'AoA Response', 'location', 'best');
grid on
xlabel('t [s]');
ylabel('Magnitude [deg]');
title('Angle of Attack Response');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

subplot(1,2,2)
plot(ans.qCL(:,1),ans.qCL(:,2),'g')
title('Pitch Rate Response');
xlabel('t [s]');
ylabel('q [deg/s]');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
grid on
sgtitle('Closed Loop Response from Dynamic Inversion');
```

Appendix B - Altitude Hold Autopilot Code

```
%% AE 295 Project, Altitude Hold
%% F-104 @ 55000 ft and M = 1.8
clear; clc;

% Parameters
g = 32.17405; % [ft/s^2] Gravitational Acceleration
U1 = 1742; % [ft/s] True Airspeed
tf = 400; % [s] Simulation Time
%% Longitudinal Directional system
% Stability and Control Derivatives
Xu = -0.0049;
Xa = -32.4692;
Zu = -0.0176;
Za = -346.6128;
Mu = 0;
Ma = -18.1248;
Madot = -0.0783;
Mq = -0.1844;
Xde = 0;
Zde = -87.9865;
Mde = -18.1525;
%% Longitudinal State-Space system
Along = [Xu              Xa              0          -g;...
      Zu/U1           Za/U1           1          0;...
      Mu+(Madot*Zu)/U1    Ma+(Madot*Za)/U1    Mq+Madot    0;...
      0               0               1          0];

Blong = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0];

Clong = eye(length(Along));

Dlong = zeros(size(Clong,1),size(Blong,2));

Clong_u = [1 0 0 0]; % Output Forward Velocity
Clong_alpha = [0 1 0 0]; % Output Angle of Attack
Clong_q = [0 0 1 0]; % Output Pitch Rate
```

```
Clong_theta = [0 0 0 1]; % Output Pitch Angle
%% Define OL SS system
longsys = ss(Along,Blong,Clong,Dlong);
%% Check Controllability and Observability of AOA, Pitch Rate, Pitch Angle
% Controllability
Co = ctrb(Along, Blong);
rankCo = rank(Co);
disp('rankCo = ');
disp(rankCo);

% Observability
Ob = obsv(Along, Clong);
rankOb = rank(Ob);
disp('rankOb = ');
disp(rankOb);

alphaOb = obsv(Along, Clong_alpha);
alpharankOb = rank(alphaOb);
disp('alpha rankOb = ');
disp(alpharankOb);

qOb = obsv(Along, Clong_q);
qrankOb = rank(qOb);
disp('q rankOb = ');
disp(qrankOb);

thetaOb = obsv(Along, Clong_theta);
thetarankOb = rank(thetaOb);
disp('theta rankOb = ');
disp(thetarankOb);
%% Altitude Hold (LQG/Kalman Filter)
% Longitudinal State-Space system inclduing Altitude
altitudeTheta = 0; % [deg] Assume zero degrees to maintain altitude

Along_altitude = [Xu                Xa                0          -g 0;...
      Zu/U1             Za/U1             1          0 0;...
      Mu+(Madot*Zu)/U1    Ma+(Madot*Za)/U1    Mq+Madot    0 0;...
      0                 0                 1          0 0
      0 0 0 0 -U1*sind(altitudeTheta)];
```

```
Blong_altitude = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0
      0];

Clong_altitude = eye(length(Along_altitude));

Dlong_altitude = zeros(size(Clong_altitude,1),size(Blong_altitude,2));

Cpartial_altitude = [0 0 0 0 1]; % Output altitude only
Dpartial_altitude  = zeros(size(Cpartial_altitude,1),size(Blong_altitude,2));

% Obtain the observability matrix
Ob_altitude = obsv(Along_altitude,Cpartial_altitude);

% Investigate the rank of the observability matrix
rankOb_altitude = rank(Ob_altitude);

% Define simulation time parameters
simtime = 20; % total sim time
dt = 1e-2; % time-step

% Define process noise covariance matrix
Qn = 0.5*eye(length(Along_altitude));

% Define measurement noise covariance matrix
Rn = 100;

% Define the randomized ICs
x0rand = 10*[rand, rand*pi/180, rand*pi/180, rand*pi/180];

% Open and run the Simulink simulation
open_system('AE295_KalmanFilter_AltitudeHold.slx');
sim('AE295_KalmanFilter_AltitudeHold.slx');

% Plot results from Kalman Filter
figure,
plot(ans.noisy_de(:,1),ans.noisy_de(:,2),'k');
hold on
```

```
plot(ans.de(:,1),ans.de(:,2),'r');
xlabel('t [s]');
ylabel('\deltae [deg]');
title('Noisy Elevator Input');
legend('Noisy Signal','Clean Signal');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

figure,
subplot(2,2,1)
plot(ans.xest(:,1),(180/pi)*ans.xest(:,3),'g');
hold on
plot(ans.xtrue(:,1),(180/pi)*ans.xtrue(:,3),'r--');
legend('Estimated','True','location','northeast');
title('Angle of Attack');
xlabel('t [s]');
ylabel('\alpha [deg]');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

subplot(2,2,2)
plot(ans.xest(:,1),(180/pi)*ans.xest(:,4),'k');
hold on
plot(ans.xtrue(:,1),(180/pi)*ans.xtrue(:,4),'r--');
legend('Estimated','True','location','northeast');
title('Pitch Rate');
xlabel('t [s]');
ylabel('q [deg/s]');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

subplot(2,2,3)
plot(ans.xest(:,1),(180/pi)*ans.xest(:,5),'m');
hold on
plot(ans.xtrue(:,1),(180/pi)*ans.xtrue(:,5),'r--');
legend('Estimated','True','location','northeast');
title('Pitch Angle');
xlabel('t [s]');
ylabel('\theta [deg]');
set(gca,'fontsize',12);
```

```
set(findall(gcf,'type','line'),'linewidth',3);

subplot(2,2,4)
plot(ans.xest(:,1),ans.xest(:,6),'m');
hold on
plot(ans.xtrue(:,1),ans.xtrue(:,6),'r--');
legend('Estimated','True','location','northeast');
title('Altitude');
xlabel('t [s]');
ylabel('Altitude [ft]');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
sgtitle('Estimated vs. True Open-Loop State Responses')

% Implement LQG Controller for Altitude Hold
% Flight Conditions
simtimeAltitude = 300;
h_desired = 55000; % Flight Altitude
IC = [U1 0 0 0 55000]; % Initial Conditions

% Extract Transfer Function of State Space System with Altitude
tf = ss2tf(Along_altitude, Blong_altitude, Cpartial_altitude, Dpartial_altitude);

% Compute Kalman Filter Gain Matrix
Kf = (lqr(Along_altitude', Cpartial_altitude', Qn, Rn))';

% Define Qc and Rc Matrices for Controller
Qc = diag([1 1 10 100 1000]);
Rc = 3e4;

% Determine LQG Gain Matrix
K_LQG = lqr(Along_altitude-Kf*Cpartial_altitude, Blong_altitude, Qc, Rc);

% Open and Run the Simulink Simulation
open_system('AE295_LQG_AltitudeHold.slx');
sim('AE295_LQG_AltitudeHold.slx');

% Plot Results
figure,
plot(ans.x_est(:,1), ans.x_est(:,6));
```

```
hold on
plot(ans.href(:,1), ans.href(:,2));
xlabel('Time [s]');
ylabel('Altitude [ft]');
title('Altitude Hold Performance');
legend('Actual Altitude', 'Desired Altitude', 'location', 'best');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
grid on;
```

Appendix C - Roll Angle Hold Autopilot Code

```
%% AE 295 Project, F-104 Autopilots, Roll Angle Hold
%% F-104 @ 55000 ft and M = 1.8
clear; clc;

% Parameters
g = 32.17405; % [ft/s^2] Gravitational Acceleration
U1 = 1742; % [ft/s] True Airspeed
tf = 400; % [s] Simulation Time
%% Lateral-Directional system
% Dimensional Stability and Control Derivatives
theta1 = 0;
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Lateral-Directional State-Space System
Alat = [0 1 0 0 0;
   0 Lp Lb Lr 0;
   g*cosd(theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;
   0 Np Nb Nr 0;
   0 0 0 1 0];

Blat = [0 0;
   Ldr Lda;
   Ydr/U1 Yda/U1;
   Ndr Nda;
   0 0];
```

```
Clat = eye(length(Alat));

Dlat = zeros();
%% Roll Angle Hold (PID)
% Start Simulation
open('RollAngleHold_PID_Simulink.slx');
sim('RollAngleHold_PID_Simulink.slx')

% Plot Results
figure
plot(ans.PhiRef(:,1), ans.PhiRef(:,2), 'k');
hold on
plot(ans.phi(:,1), ans.phi(:,2), 'r');
legend('Reference Signal', 'With PID', 'location', 'best');
xlabel('Time [s]')
ylabel('\phi [deg]')
ylim([0 6])
grid on
title('Closed-Loop Roll Angle Response');
set(findall(gcf, 'type', 'line'), 'linewidth', 3);
set(gca, 'fontsize', 14);

figure
plot(ans.da(:,1), ans.da(:,2), 'k')
xlim ([0 10])
xlabel('t (s)')
ylabel('\delta_a (deg)')
grid on
title('Aileron Deflection')
set(gca,'fontsize', 12);
set(findall(gcf, 'type', 'line'), 'linewidth',3);
```

Appendix D - Speed/Mach Hold Autopilot Code

```
%% AE 295 Project, Speed/Mach Hold Autopilot Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 1742; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Longitudinal Directional Control Derivatives
Xu = -0.0049;
Xa = -32.4692;
Zu = -0.0176;
Za = -346.6128;
Mu = 0;
Ma = -18.1248;
Madot = -0.0783;
Mq = -0.1844;
Xde = 0;
Zde = -87.9865;
Mde = -18.1525;
%% Longitudinal State-Space system
Along = [Xu                Xa                 0          -g;...
      Zu/U1             Za/U1              1          0;...
      Mu+(Madot*Zu)/U1    Ma+(Madot*Za)/U1    Mq+Madot    0;...
      0                 0                  1          0];

Blong = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0];

Clong = eye(length(Along));

Dlong = zeros(size(Clong,1), size(Blong,2));
%% Define Cpartial and Dpartial matrices to measure forward velocity only
Cpartial = [1 0 0 0];
Dpartial = zeros(size(Cpartial,1),size(Blong,2));
%% Check Controllability and Observability of OL system
Ob = obsv (Along, Cpartial);
rankOb = rank(Ob);
```

```
Co = ctrb(Along, Blong);
rankCo = rank(Co);
%% Design Kalman Filter
% Define process noise covariance matrix
Qn = 0.5*eye(length(Along));

% Define measurement noise covariance matrix
Rn = 100;

% Define the randomized ICs
x0rand = 10*[rand, rand*pi/180, rand*pi/180, rand*pi/180];
%% LQG Controller Design for Speed/Mach Hold
% Compute the Kalman filter gain matrix
Kf = (lqr(Along',Cpartial',Qn,Rn))';

% Define Qc and Rc matrices for the controller
Qc = diag([1 500 100 1]);
Rc = 1e5;

% Determine the LQG gain matrix
K_LQG = lqr(Along-Kf*Cpartial,Blong,Qc,Rc);
%% Open and run Simulink simulation
open_system('LQG_SpeedMachHold_Autopilot_Simulink.slx');
sim('LQG_SpeedMachHold_Autopilot_Simulink.slx');
%% Plot Results
figure,
plot(ans.de(:,1),ans.de(:,2),'b');
xlabel('t [s]');
ylabel('Magnitude [deg]');
title('Elevator Deflection');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

figure,
plot(ans.u_ref(:,1),ans.u_ref(:,2),'b');
hold on
plot(ans.u(:,1),ans.u(:,2),'r');
xlabel('t [s]');
ylabel('u(t) [ft/s]');
```

```
title('Speed/Mach Hold Autopilot Response');
legend('Reference','Autopilot Response','location', 'best');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
```

Appendix E - Heading Angle Hold Autopilot Code

```
%% AE 295 Project, Heading Angle Hold Autopilot Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 1742; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Define the A, B matrices for the Lateral-Directional System
Alatdir = [0 1 0 0 0;...
 0 Lp Lb Lr 0;...
 g*cosd(Theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;...
 0 Np Nb Nr 0;...
 0 0 0 1 0];

Blatdir = [0 0;...
 Ldr Lda;...
 Ydr/U1 Yda/U1;...
 Ndr Nda;...
 0 0];

Clatdir = eye(length(Alatdir));

Dlatdir = zeros(size(Clatdir,1), size(Blatdir,2));
```

```
%% Define Cpartial and Dpartial matrices to measure heading angle only
Cpartial = [0 0 0 0 1];
Dpartial = zeros(size(Cpartial,1),size(Blatdir,2));
%% Check Controllability and Observability of OL system
Ob = obsv (Alatdir, Cpartial);
rankOb = rank(Ob);

Co = ctrb(Alatdir, Blatdir);
rankCo = rank(Co);
%% Design Kalman Filter
% Define process noise covariance matrix
Qn = 0.1*eye(length(Alatdir));

% Define measurement noise covariance matrix
Rn = 75;

% Define the randomized ICs
x0rand = 10*[rand, rand*pi/180, rand*pi/180, rand*pi/180];
%% LQG Controller Design for Heading Angle Hold
% Compute the Kalman filter gain matrix
Kf = (lqr(Alatdir',Cpartial',Qn,Rn))';

% Define Qc and Rc matrices for the controller
Qc = diag([0.5 0.5 1 30 10]);
Rc = diag([6e4 6e4]);

% Determine the LQG gain matrix
K_LQG = lqr(Alatdir-Kf*Cpartial,Blatdir,Qc,Rc);
%% Open and run Simulink simulation
open_system('LQG_HeadingAngleHold_Simulink.slx');
sim('LQG_HeadingAngleHold_Simulink.slx');
%% Plot Results
figure,
plot(ans.dr(:,1),ans.dr(:,2),'b');
xlabel('t [s]');
ylabel('Magnitude [deg]');
title('Rudder Deflection');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
```

```
figure,
plot(ans.da(:,1),ans.da(:,2),'b');
xlabel('t [s]');
ylabel('Magnitude [deg]');
title('Aileron Deflection');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);

figure,
plot(ans.psi_ref(:,1),ans.psi_ref(:,2),'b');
hold on
plot(ans.psi(:,1),ans.psi(:,2),'r');
xlabel('t [s]');
ylabel('\psi [deg]');
title('Heading Angle Hold Autopilot Response');
legend('Reference','Autopilot Response','location', 'best');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
```

Appendix F - Yaw Damper SAS Code

```
%% AE 295 Project, Yaw Damper Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 871; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Define the A, B matrices for the Lateral-Directional System
Alatdir = [0 1 0 0 0;...
 0 Lp Lb Lr 0;...
 g*cosd(Theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;...
 0 Np Nb Nr 0;...
 0 0 0 1 0];

Blatdir = [0 0;...
 Ldr Lda;...
 Ydr/U1 Yda/U1;...
 Ndr Nda;...
 0 0];

Clatdir = eye(length(Alatdir));

Dlatdir = zeros(size(Clatdir,1),size(Blatdir,2));
```

```
%% Define the Open-Loop Full Lateral-Directional System
latdirsys = ss(Alatdir,Blatdir,Clatdir,Dlatdir);
set(latdirsys, 'statename', {'\phi','p','\beta','r','\psi'},...
 'inputname', {'\delta_r', '\delta_a'},...
 'outputname',{'\phi','p','\beta','r','\psi'});

% Check OL system pole locations and damping characteristics
damp(latdirsys)
%% OL System Response
tsim = 20; %[sec] sim time
figure
impulse(latdirsys, tsim,'b'); % Obtain OL sys impulse responses
title('Open-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% CL Control System Design
% Obtain the TF from rudder (delta_r) to yaw rate (r)
C1 = [0,0,0,1,0]; % Output r
B1 = Blatdir(:,1); % Use input delta_r only
D1 = 0; % Feedforward matrix since we have 1 TF only
[numTF_dr2r, denTF1] = ss2tf(Alatdir,B1,C1,D1);
fprintf('TF from rudder to yaw rate:');
TF_dr2r = tf(numTF_dr2r,denTF1);
%% Yaw Damper Design
%% Define the PID controller
s = tf('s'); % TF (Laplace's) variable
Kp = 20; % Proportional gain
Ki = 0.1; % Integral gain
Kd = 0.1; % Deivative gain
PID = Kp + Ki/s + Kd*s; % PID controller

% Close the loop
CLsys1 = feedback(PID*TF_dr2r,-1);

% Test the impulse response
figure,
impulse(CLsys1, tsim,'r');
title('Closed-Loop TF: r/dr Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
```

```
% Compare OL-TF vs. CL-TF impulse responses
figure,
impulse(TF_dr2r, tsim,'b');
hold on
impulse(CLsys1, tsim,'g');
title('Open-Loop vs. Closed-Loop TF: r/dr Impulse Response');
legend('Open-Loop','Closed-Loop','Location','best');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Test the PID controller on the whole Lat-Dir AC system
% Define the full CL control system with the yaw damper
CLsys_full = feedback(PID*latdirsys,-1,1,4); % The last two number stands for Input 1 (rudder)
and Output 4 (yaw rate)

% Test the impulse response of the full CL system
tsim = 60; %[sec]
figure,
impulse(CLsys_full, tsim,'r');
title('Full Lat-Dir Closed-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Retune the PID controller
Kp = 1.5; % Proportional gain
Ki = 2; % Integral gain
Kd = 0; % Derivative gain
Tw = 0.5; % Washout filter constant

Washout = s / (s + 1/Tw);
PID = (Kp + Ki*(1/s) + Kd*s) * Washout;

% Define the full CL control system with the yaw damper
CLsys_full = feedback(PID*latdirsys,-1,1,4); % The last two number stands for Input 1 (rudder)
and Output 4 (yaw rate)

% Test the impulse response of the full CL system
figure,
impulse(CLsys_full, tsim,'r');
title('Full Lat-Dir Closed-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
```

grid on

Appendix G - Roll Damper SAS Code

```
%% AE 295 Project, Roll Damper Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 871; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Define the A, B matrices for the Lateral-Directional System
Alatdir = [0 1 0 0 0;...
 0 Lp Lb Lr 0;...
 g*cosd(Theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;...
 0 Np Nb Nr 0;...
 0 0 0 1 0];

Blatdir = [0 0;...
 Ldr Lda;...
 Ydr/U1 Yda/U1;...
 Ndr Nda;...
 0 0];

Clatdir = eye(length(Alatdir));

Dlatdir = zeros(size(Clatdir,1),size(Blatdir,2));
```

```
%% Define the Open-Loop Full Lateral-Directional System
latdirsys = ss(Alatdir,Blatdir,Clatdir,Dlatdir);
set(latdirsys, 'statename', {'\phi','p','\beta','r','\psi'},...
 'inputname', {'\delta_r', '\delta_a'},...
 'outputname',{'\phi','p','\beta','r','\psi'});

% Check OL system pole locations and damping characteristics
damp(latdirsys)
%% OL System Response
tsim = 20; %[sec] sim time
figure
impulse(latdirsys, tsim,'b'); % Obtain OL sys impulse responses
title('Open-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% CL Control System Design
% Obtain the TF from aileron (delta_a) to roll rate (p)
C1 = [0,1,0,0,0]; % Output p
B1 = Blatdir(:,2); % Use input delta_a only
D1 = 0; % Feedforward matrix since we have 1 TF only
[numTF_da2p, denTF1] = ss2tf(Alatdir,B1,C1,D1);
fprintf('TF from aileron to roll rate:');
TF_da2p = tf(numTF_da2p,denTF1);
%% Roll Damper Design
%% Define the PID controller
s = tf('s'); % TF (Laplace's) variable
Kp = 0.25; % Proportional gain
Ki = 0.03; % Integral gain
Kd = 0.3; % Deivative gain
PID = Kp + Ki/s + Kd*s; % PID controller

% Close the loop
CLsys1 = feedback(PID*TF_da2p,-1);

% Test the impulse response
figure,
impulse(CLsys1, tsim,'r');
title('Closed-Loop TF: p/da Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
```

```
% Compare OL-TF vs. CL-TF impulse responses
figure,
impulse(TF_da2p, tsim,'b');
hold on
impulse(CLsys1, tsim,'g');
title('Open-Loop vs. Closed-Loop TF: p/da Impulse Response');
legend('Open-Loop','Closed-Loop','Location','best');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Test the PID controller on the whole Lat-Dir AC system
% Define the full CL control system with the roll damper
CLsys_full = feedback(PID*latdirsys,-1,2,2); % The last two number stands for Input 2 (aileron)
and Output 2 (roll rate)

% Test the impulse response of the full CL system
tsim = 60; %[sec]
figure,
impulse(CLsys_full, tsim,'r');
title('Full Lat-Dir Closed-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Retune the PID controller
Kp = 2.2; % Proportional gain
Ki = 0.2; % Integral gain
Kd = 4; % Deivative gain

PID = Kp + Ki*(1/s) + Kd*s;

% Define the full CL control system with the roll damper
CLsys_full = feedback(PID*latdirsys,-1,2,2); % The last two number stands for Input 2 (aileron)
and Output 2 (roll rate)

% Test the impulse response of the full CL system for roll damping
figure,
impulse(CLsys_full, tsim,'r');
title('Full Lat-Dir CL Impulse Response with Roll Damper');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
```

Appendix H - Pitch Damper SAS Code

```
%% AE 295 Project, Pitch Damper Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 871; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Xu = -0.0049;
Xa = -32.4692;
Zu = -0.0176;
Za = -346.6128;
Mu = 0;
Ma = -18.1248;
Madot = -0.0783;
Mq = -0.1844;
Xde = 0;
Zde = -87.9865;
Mde = -18.1525;
%% Longitudinal State-Space system
Along = [Xu                Xa                0          -g;...
      Zu/U1             Za/U1             1          0;...
      Mu+(Madot*Zu)/U1    Ma+(Madot*Za)/U1    Mq+Madot    0;...
      0                 0                 1          0];

Blong = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0];

Clong = eye(length(Along));

Dlong = zeros(size(Clong,1),size(Blong,2));
%% Define the Open-Loop Full Longitudinal-Directional System
longsys = ss(Along,Blong,Clong,Dlong);
set(longsys, 'statename', {'u','\alpha','q','\theta'},...
 'inputname', {'\delta_e'},...
 'outputname',{'u','\alpha','q','\theta'});
```

```
% Check OL system pole locations and damping characteristics
damp(longsys)
%% OL System Response
tsim = 20; %[sec] sim time
figure
impulse(longsys, tsim,'b'); % Obtain OL sys impulse responses
title('Open-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% CL Control System Design
% Obtain the TF from elevator deflection (delta_e) to pitch rate (q)
C1 = [0,0,1,0]; % Output q
B1 = Blong(:,1); % Use input delta_e only
D1 = 0; % Feedforward matrix since we have 1 TF only
[numTF_de2q, denTF1] = ss2tf(Along,B1,C1,D1);
fprintf('TF from elevator deflection to pitch rate:');
TF_de2q = tf(numTF_de2q,denTF1);
%% Pitch Damper Design
%% Define the PID controller
s = tf('s'); % TF (Laplace's) variable
Kp = 20; % Proportional gain
Ki = 0; % Integral gain
Kd = 1; % Deivative gain
PID = Kp + Ki/s + Kd*s; % PID controller

% Close the loop
CLsys1 = feedback(PID*TF_de2q,-1);

% Test the impulse response
figure,
impulse(CLsys1, tsim,'r');
title('Closed-Loop TF: q/de Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on

% Compare OL-TF vs. CL-TF impulse responses
figure,
impulse(TF_de2q, tsim,'b');
hold on
impulse(CLsys1, tsim,'g');
```

```
title('Open-Loop vs. Closed-Loop TF: q/de Impulse Response');
legend('Open-Loop','Closed-Loop','Location','best');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Test the PID controller on the whole Lat-Dir AC system
% Define the full CL control system with the yaw damper
CLsys_full = feedback(PID*longsys,-1,1,4); % The last two number stands for Input 1 (elevator
deflection) and Output 3 (pitch rate)

% Test the impulse response of the full CL system
tsim = 60; %[sec]
figure,
impulse(CLsys_full, tsim,'r');
title('Full Longitudinal-Directional Closed-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
%% Retune the PID controller
Kp = 25; % Proportional gain
Ki = 0.5; % Integral gain
Kd = 2; % Deivative gain
Tw = 0.4; % Washout filter constant

Washout = s / (s + 1/Tw);
PID = (Kp + Ki*(1/s) + Kd*s) * Washout;

% Define the full CL control system with the pitch damper
CLsys_full = feedback(PID*longsys,-1,1,3); % The last two number stands for Input 1 (elevator
deflection) and Output 3 (pitch rate)

% Test the impulse response of the full CL system
figure,
impulse(CLsys_full, tsim,'r');
title('Full Longitudinal-Directional Closed-Loop System Impulse Response');
set(findall(gcf,'type','line'),'linewidth',2);
grid on
```

Appendix I - Roll Rate CAS Code

```
%% AE 295 Project, Roll Rate CAS Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 871; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Define the A, B matrices for the Lateral-Directional System
Alatdir = [0 1 0 0 0;...
 0 Lp Lb Lr 0;...
 g*cosd(Theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;...
 0 Np Nb Nr 0;...
 0 0 0 1 0];

Blatdir = [0 0;...
 Ldr Lda;...
 Ydr/U1 Yda/U1;...
 Ndr Nda;...
 0 0];
%% Define the C, D matrices
C = [1 0 0 0 0;
    0 1 0 0 0];
```

```
D = [0 0;
   0 0];
%% Roll Rate CAS (LQR)
% Tune Q and R for LQR controller
Q = diag([1 50 1 1 200]);
R = diag([2 2]);

% Determine LQR gain
K_LQR = lqr(Alatdir, Blatdir, Q, R);

% Start Simulation
open('RollRate_CAS_Simulink.slx');
sim('RollRate_CAS_Simulink.slx')

% Plot Results
figure
plot(ans.p_ref(:,1), ans.p_ref(:,2), 'k');
hold on
plot(ans.p(:,1), ans.p(:,2), 'r');
legend('Reference Signal', 'With LQR', 'location', 'best');
xlabel('Time [s]');
ylabel('p [deg/s]');
grid on
title('Roll Rate CAS Response');
set(findall(gcf, 'type', 'line'), 'linewidth', 3);
set(gca, 'fontsize', 14);

figure
plot(ans.phi(:,1), ans.phi(:,2), 'k');
legend('With LQR', 'location', 'best');
xlabel('Time [s]');
ylabel('\phi [deg]');
grid on
title('Roll Angle Response');
set(findall(gcf, 'type', 'line'), 'linewidth', 3);
set(gca, 'fontsize', 14);

figure
plot(ans.da(:,1), ans.da(:,3), 'k')
xlabel('t (s)');
```

```
ylabel('\delta_a (deg)')
grid on
title('Aileron Deflection')
set(gca,'fontsize', 12);
set(findall(gcf, 'type', 'line'), 'linewidth',3);
```

Appendix J - Normal Acceleration CAS Code

```
%% AE 295 Project, Normal Acceleration CAS Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 1742; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
Xu = -0.0049;
Xa = -32.4692;
Zu = -0.0176;
Za = -346.6128;
Mu = 0;
Ma = -18.1248;
Madot = -0.0783;
Mq = -0.1844;
Xde = 0;
Zde = -87.9865;
Mde = -18.1525;
%% Longitudinal State-Space system
Along = [Xu                Xa                0          -g;...
      Zu/U1            Za/U1             1          0;...
      Mu+(Madot*Zu)/U1   Ma+(Madot*Za)/U1  Mq+Madot   0;...
      0                0                 1          0];

Blong = [Xde;...
      Zde/U1;...
      Mde+(Madot*Zde)/U1;...
      0];

Clong = eye(length(Along));

Dlong = zeros(size(Clong,1), size(Blong,2));

Clong_nz = [0 Za/g 0 0];

Dlong_nz = 0;
%% Dynamic Inversion Control System Design
% Compute matrices required for Dynamic Inversion
CBinv = inv(Clong_nz*Blong);
```

```
CA = Clong_nz*Along;

% Define controller gain
K = 3;

% Start Simulink Simulation
open_system('DI_NormalAcceleration_CAS_Simulink.slx');
sim('DI_NormalAcceleration_CAS_Simulink.slx');

% Plot Results
figure,
plot(ans.de(:,1), ans.de(:,2), 'b');
grid on
xlabel('t (s)');
ylabel('Magnitude (deg)');
title('Elevator Deflection');
set(gca, 'fontsize', 12);
set(findall(gcf, 'type', 'line'), 'linewidth', 3);

figure,
plot(ans.nz_ref(:,1), ans.nz_ref(:,2), 'k')
hold on
plot(ans.nz(:,1),ans.nz(:,2),'r')
legend('Reference', 'CAS Response', 'location', 'best');
grid on
xlabel('t (s)');
ylabel('Magnitude (g)');
title('Normal Acceleration CAS response');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
```

Appendix K - Lateral-Directional CAS Code

```
%% AE 295 Project, Lateral Directional CAS Design, Kenneth Gorospe
clear; clc;
%% Define parameters
g = 32.17405; % [ft/s^2] gravitational acceleration on Earth
U1 = 1742; % [ft/s]
Theta1 = 0; % [deg]
dt = 1e-2;
%% Define dimensional stability and control derivatives
Yb = -175.8047;
Lb = -47.2783;
Lp = -0.8692;
Lr = 0.4921;
Nb = 7.5310;
Np = -0.0182;
Nr = -0.1270;
Ydr = 14.6364;
Ldr = 4.0161;
Ndr = -1.3537;
Yda = 0;
Lda = 8.7948;
Nda = 0.0778;
Yp = 0;
Yr = 0;
%% Define the A, B matrices for the Lateral-Directional System
Alatdir = [0 1 0 0 0;...
 0 Lp Lb Lr 0;...
 g*cosd(Theta1)/U1 Yp/U1 Yb/U1 (Yr/U1)-1 0;...
 0 Np Nb Nr 0;...
 0 0 0 1 0];

Blatdir = [0 0;...
 Ldr Lda;...
 Ydr/U1 Yda/U1;...
 Ndr Nda;...
 0 0];

Clatdir = eye(length(Alatdir));

Dlatdir = zeros(size(Clatdir,1), size(Blatdir,2));
```

```
Clatdir_r = [0 0 0 1 0
   0 0 1 0 0];
%% Dynamic Inversion Control System Design
% Compute matrices required for Dynamic Inversion
CBinv = inv(Clatdir_r*Blatdir);
CA = Clatdir_r*Alatdir;

% Define controller gain
K = diag([1.75 0.01]);

% Start Simulink Simulation
open_system('DI_LatDir_CAS_Simulink.slx');
sim('DI_LatDir_CAS_Simulink.slx');

% Plot Results
figure,
plot(ans.dr(:,1), ans.dr(:,2), 'b');
grid on
xlabel('t (s)');
ylabel('Magnitude (deg)');
title('Rudder Deflection');
set(gca, 'fontsize', 12);
set(findall(gcf, 'type', 'line'), 'linewidth', 3);

figure,
plot(ans.da(:,1), ans.da(:,2), 'g');
grid on
xlabel('t (s)');
ylabel('Magnitude (deg)');
title('Aileron Deflection');
set(gca, 'fontsize', 12);
set(findall(gcf, 'type', 'line'), 'linewidth', 3);

figure,
plot(ans.r_ref(:,1), ans.r_ref(:,2), 'k')
hold on
plot(ans.r(:,1),ans.r(:,2),'r')
legend('Reference', 'CAS Response', 'location', 'best');
grid on
```

```
xlabel('t (s)');
ylabel('Magnitude (deg/s)');
title('Lateral Directional CAS response');
set(gca,'fontsize',12);
set(findall(gcf,'type','line'),'linewidth',3);
```