REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
Dec 94

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
Development of a TECS Control-Law for the Lateral Directional Axis of the McDonnell Douglas F-15 Eagle

5. FUNDING NUMBERS

6. AUTHOR(S)
Michael August Bruzzini

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AFIT Students Attending:
University of Washington

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/CI/CIA
94-159

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DEPARTNEMT OF THE AIR FORCE
AFIT/CI
2950 P STREET, BDLG 125
WRIGHT-PATTERSON AFB OH 45433-7765

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release IAW AFR 190-1
Distribution Unlimited
BRIAN D. GAUTHIER, MSgt, USAF
Chief Administration

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

19950117 013

14. SUBJECT TERMS

15. NUMBER OF PAGES
104

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT

18. SECURITY CLASSIFICATION OF THIS PAGE

19. SECURITY CLASSIFICATION OF ABSTRACT

20. LIMITATION OF ABSTRACT

94-159

Development of a TECS Control-Law for the Lateral Directional Axis of the McDonnell Douglas F-15 Eagle

by

Michael August Bruzzini

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Aeronautics and Astronautics

University of Washington
1994

Approved by________________________________
(Chairperson of Supervisory Committee)

____

____

Program Authorized
to Offer Degree: Department of Aeronautics and Astronautics
Date 12/16/94

Development of a TECS Control-Law for the Lateral Directional Axis of the McDonnell Douglas F-15 Eagle

by

Michael August Bruzzini

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Aeronautics and Astronautics

University of Washington
1994

Approved by____
(Chairperson of Supervisory Committee)

____

____

Program Authorized
to Offer Degree: Department of Aeronautics and Astronautics
Date____

In presenting this thesis in partial fulfillment of the requirements for the Master of Science degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this thesis is allowable only for scholarly purposes, consistent with "fairuse" as prescribed in the U.S. Copyright Law.

Signature____________________
Date 12-16-94

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank everyone who contributed in one way or another to the completion of this thesis as well as to those who helped me acquire my degree. First and foremost I would like to thank the United States Air Force and the Air Force Institute of Technology for sending me to the University of Washington during my banked assignment time. If it wasn't for their faith in my abilities I would not have received the funding or time in order to complete this degree as a full-time student.

Secondly, I would like to thank the faculty and staff here at the Aeronautics and Astronautics Department of the University of Washington. In particular, I would thank Dr. Ly for the numerous hours of instruction inside and outside of class. You and I spent long hours together on this thesis, sometimes continuing past the normal 9-to-5 business hours. Your dedication and work ethic are truly an inspiration. I would also like to thank Jane and Marlo who engineered my class schedule through countless revisions. Whenever I showed up in the office, they always had a smile for me.

Next I would like to thank the other graduate students who made the 18 months at the University enjoyable. I will now list their names here for the individual credit they deserve: Sylvia, Phil, Jimbo, Zeek, G, Matchu, Dougie, Tom, Ewald, SUPA-Dave, Paul, Tony, Rob, Dave, Jeff, and Lora. Thanks for the memories, and I hope to keep our friendship alive in the years to come.

Lastly I would like to thank The Lord above, who has been part of my learning and development throughout the years. Without Him, I could not have accomplished what I did.

Chapter 1 Introduction

1.1 Problem Description

The purpose of this report is to develop automatic aircraft flight control systems using the concept of Total Energy Control System (TECS) in the lateral directional axis for the McDonnell Douglas F-15 Eagle. Nonlinear simulation was originally developed from actual flight tests conducted by the U.S. Air Force and generated into a Fortran code. Lt. James P. Dutton then took the flight data code and generated a SIMULINK file [10]. Using the SIMULINK file a control law developed on the linearized model can then be tested on the nonlinear model.

1.2 Analysis of Data Provided

Due to the complexity of the flight data developed for the F-15, the next sections are used to describe important support modules contained in [5]. The numerical data were provided in the format of a Genesis simulation used at Wright-Patterson AFB and the coding was subsequently converted from FORTRAN to MATLAB. A detailed description of the model is located in Chapter 2.

Evaluation of the open-loop nonlinear model and design of the TECS control law will be accomplished at two distinct flight points shown in Table 1.1. Other flight conditions throughout the flight envelope can be investigated in future study.

Table 1.1: Selected Flight Conditions

Flight	Altitude	Vtas	Mach
Point	(ft)	(ft/s)	Number
1	9,800	539.1	0.5
2	30,000	497.3	0.5

1.2.1 Model Characteristics

The model is an integration of several modules, each performing a specific function. Included in these modules are the aerodynamic data, the propulsion data, the atmospheric conditions, as well as the nonlinear equations of motion for the F-15. As far as the modeling of the control actuators and sensors are concerned, they can be entered into the model using the capabilities within SIMULINK to create linear filter dynamics. Figure 1.1 summarizes the integration of the modules that form the complete system.

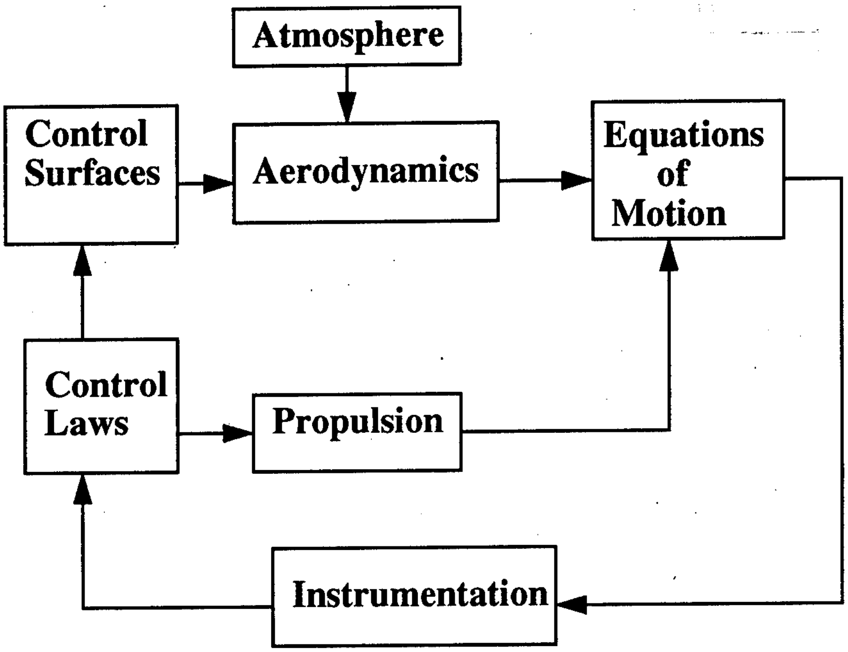


Figure 1.1 Integration of System Module Components

The aircraft is the McDonnell F-15 Eagle, the first line of defense for operational high-performance aircraft. It is powered by two afterburning turbofan engines, each providing approximately 32,000 pounds of thrust at maximum power. The primary flight controls are the horizontal stabilizers capable of both symmetric and differential movement, conventional ailerons, and dual vertical rudders. There are a total of six actuators, one for each surface mentioned above. All actuators are modeled identically with rate limits of 24 deg/sec and first-order response characteristics of

$$G(s) = \frac{20}{s+20} \tag{1.1}$$

Position limitations on the actual component and the sign conventions for positive deflections are summarized in Table 1.2. The aircraft mass and geometry characteristics are summarized in Table 1.3.

Table 1.2: Flight Control Surfaces Characteristics

Control Surface	Symbol	Limits	Sign Convention (+)
Symmetric Stabilator	δ_H	±20°	Trailing edge down
Differential Stabilator	δ_D	15°–25°	Left trailing edge down
Aileron	δ_A	±20°	Left trailing edge down
Rudder	δ_R	±30°	Trailing edge left

Table 1.3: F-15 Mass and Geometry Characteristics

Parameter	Symbol	Units	Value
Wing Area	S	ft^2	608.0
Wing span	b	ft	42.8
Mean aerodynamic chord	c	ft	15.95
Aircraft Weight	W	lb	45,000
Moments	I_x	slug-ft^2	28,700
of	I_y	slug-ft^2	165,100
Inertia	I_z	slug-ft^2	187,900
Products	I_{xz}	slug-ft^2	-520
of	I_{xy}	slug-ft^2	0
Inertia	I_{yz}	slug-ft^2	0

1.2.2 Aerodynamic Model

The aerodynamic model is composed of multidimensional table with interpolation to form nonlinear function generators. By using a vast database the highly nonlinear aerodynamics encountered in the large flight envelope of the F-15 are properly represented. This will be important for evaluating the TECS controller being developed for the linear model on the nonlinear system. Most of the aerodynamic qualities are determined from the Mach number. In addition, combination of the sideslip angle β, angle of attack α, and the symmetric stabilator deflection δ_H are also used to determine the flight environment.

The task of the aerodynamic module is to provide the nondimensionalized force and moment coefficients, which are then used to calculate the associated moments and forces of the F-15. The basic equations used for the coefficients are

Coefficients of forces:

$$C_L = C_{Lo} + \Delta C_{Lnz} n_z$$

$$C_D = C_{Do} + \Delta C_{Dalt} + \Delta C_{Dnoz}$$

$$C_Y = C_{Yo} + C_{Y\delta A}\delta_A + C_{Y\delta D}\delta_D - \Delta C_{Y\delta R}\delta_R$$

Coefficients of moments:

$$C_l = C_{lo} + C_{l\delta A}\delta_A + C_{l\delta D}\delta_D - \Delta C_{l\delta R}K_{\delta R} + \frac{b}{2V}(C_{lp}p + C_{lr}r)$$

$$C_m = C_{mo} + \Delta C_{mnz}n_z + \frac{\bar{c}}{2V}(C_{mq}q + C_{m\dot{\alpha}}\dot{\alpha} + C_{Lo}\Delta N_o)$$

$$C_n = C_{no} + C_{n\delta A}\delta_A + C_{n\delta D}\delta_D + \Delta C_{n\delta R}K_{\delta R} + \frac{b}{2V}(C_{np}p + C_{nr}r)$$

The terms C, ΔC, ΔN, and K are outputs from the function generation routines, and are either read directly or are derived from linear interpolation of tabular data. The parameters affecting each of the coefficients are located in Table 1.4.

Table 1.4: Independent Parameters Affecting the Aerodynamic Coefficients

Aero Coefficients	Independent Parameters	Aero Coefficients	Independent Parameters
C_{Lo}	M, α, δ_H	C_{lp}	M, α
ΔC_{Lnz}	M	C_{lr}	M, α
C_{mo}	M, α, δ_H	C_{no}	M, α, β
ΔC_{mnz}	M	$C_{n\delta a}$	M, α
C_{mq}	M, α	$C_{n\delta D}$	M, α
$C_{m\dot{\alpha}}$	M, α	$\Delta C_{n\delta r}$	M, α, β
ΔN_o	M	$K_{\delta Rn}$	M, α
C_D		$K_{\delta Rl}$	M
$(\alpha < 32)$	C_{Lo}, M	C_{np}	M, α
$(32 < \alpha < 40)$	C_{Lo}, M, α	C_{nr}	M, α
$(\alpha > 40)$	C_{Lo}, α	C_{yo}	M, α, β
ΔC_{Dalt}	h	$C_{y\delta A}$	M, α
ΔC_{Dnoz}	M, δ_{PLA}	$C_{y\delta D}$	M, α
C_{lo}	M, α, β	$\Delta C_{y\delta R}$	M, α, δ_R
$\Delta C_{l\delta R}$	M, α, δ_R	$K_{\delta Ry}$	M

The total forces and moments are calculated from the following equations

$$L = \bar{q}SC_L$$

$$D = \bar{q}SC_D$$

$$Y = \bar{q}SC_Y$$

and

$$\Sigma L = \bar{q}SbC_l$$

$$\Sigma M = \bar{q} S \bar{c} C_m$$

$$\Sigma N = \bar{q} S b C_n$$

where $\bar{q}$=1/2 ρV^2 is the dynamic pressure, S is the wing planform area, b is the wing span, and c is the mean aerodynamic chord.

1.2.3 Propulsion Model

The propulsion module has two distinct sub-modules, one for each engine. Although each engine is similar, they will not produce the same amount of thrust for the same power settings, thus the need for two different sub-modules. The thrust vectors are aligned with the aircraft's x-axis. The thrust produced by each engine is a function of altitude h, Mach number M, and throttle setting δ_{PLA}. Furthermore, each module is subdivided into two sections--the core and the afterburner section, each with its individual sequencing logic.

The throttle inputs to the engine module are in degrees of power-level-angle (PLA) with a minimum angle of 20^o and a maximum position of 127^o. The core section responds to the setting up to 83^o, while the afterburner initiates at 91^o. The core section has first-order dynamics and a rate limiter added in order to simulate the spool up time. The afterburner has a rate limiter and a sequencing logic to model the fuel pump and pressure regulator effects.

1.2.4 Atmospheric Module

The atmospheric model's data is based on tables from the U.S. Standard Atmosphere (1962). This model calculates values for the speed of sound, the acceleration due to gravity, air density, viscosity, and ambient static pressure and temperature based on the aircraft altitude. Linear interpolation is used between table values for altitudes from 0 to 90 km.

1.2.5 Equations of Motion

The nonlinear equations of motion used in the system model are based on the derivations by Duke, Antoniewicz, and Krambeer in [6]. These equations model the six-degree-of-freedom dynamics of a rigid aircraft flying over a flat, non-rotating Earth. These derivations are detailed in the next chapter.

Chapter 2 The Nonlinear F-15 Model

Linearized aircraft models are useful in the early stage of control law development. A design flight envelope can be defined based on a number of flight points. However, the control law will eventually have to be validated in nonlinear simulation, prior to the actual flight tests.

2.1 Derivation of Nonlinear State Equations

In practice, most control laws are formulated by decoupling the longitudinal motion from the lateral motion. This is not a correct practice in all cases. Certain simplifying assumptions must first be defined. The first of which, the aircraft must be symmetrical. In a generic sense, most aircraft are symmetrical, however a refueling boom on one side of the aircraft such as the A-4 or a different payload strapped under each wing on the aircraft can cause this assumption to be invalid. The other assumption is to have the aircraft trimmed at a specific reference trajectory; for instance straight and level flight. With this specific reference condition and a symmetrical aircraft, the longitudinal and lateral modes can always be decoupled. In the next section, the aircraft will not have either of these assumptions. Instead, the model will be considered as a rigid aircraft of constant mass flying over a flat, non-rotating earth.

2.1.1 Reference Systems

There are three primary reference systems associated with an aircraft in flight. These are the body-axis, wind-axis, the vehicle-carried vertical-axis systems. One axis system is preferred over another axis depending on the area of applications.

The rotational equations of motion are most easily referenced to the body-axis system. The body-axis rotational rates are measurable within the aircraft by sensors fixed in the body frame. The body axis-system has its origin at the aircraft's center of gravity, the x-axis extends out the nose of the aircraft, the y-axis exits the right wing of the aircraft, and the z-axis drops out the bottom of the aircraft. The positive rotation rates follow the right-hand rule. The x-, y-, and z- components of the aircraft velocity are u, v, w respectively. In addition the roll rates are p, q, and r with the moments being *L*, *M*, and *N* for the x-, y-, and z-axis respectively. Figure 2.1 shows these relationships.

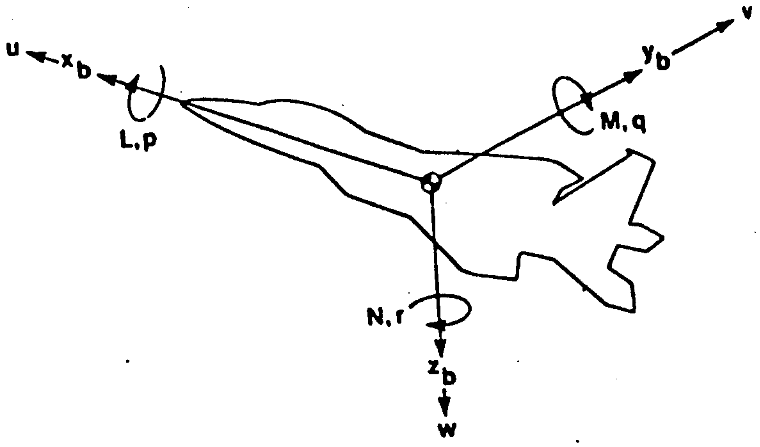


Figure 2.1 Aircraft Body-Axis System

The second-axis system is the wind-axis system which is mainly used for the translational equations of motion. Due to the fact that the aerodynamic forces acting on an aircraft are directly related to the wind direction, this system becomes very useful. Specifically, because the forces imparted on the aircraft are the result of the aerodynamic forces which are a function of the angle of attack α, the total velocity V, and the sideslip angle β. For this axis system, the origin is still located at the aircraft's center of gravity. The x-axis is located in the direction of the aircraft's velocity vector which is not necessarily out the nose. The y-axis again exits the aircraft's right wing, and the z-axis still exits the bottom. Due to the fact that the body and wind system have the same origin, the orientation can be defined from the angle of attack α and the sideslip angle β. Components of the total velocity V can be expressed in terms of the body-axis velocities as,

$$\begin{aligned} u &= V\cos\alpha\cos\beta \\ v &= V\sin\beta \\ w &= V\sin\alpha\cos\beta \end{aligned} \tag{2.1}$$

The vehicle carried, vertical-axis system also has its origin at the aircraft center of gravity. This axis system has its x-axis in the North direction, the y-axis points East and the z-axis is directed down. In other words, the vehicle carried, vertical-axis system is the earth-fixed reference system only translated to the aircraft's center of gravity. The method of comparing this axis system to the body-axis system is by the Euler angles ψ, θ, and φ as seen in Figure 2.2.

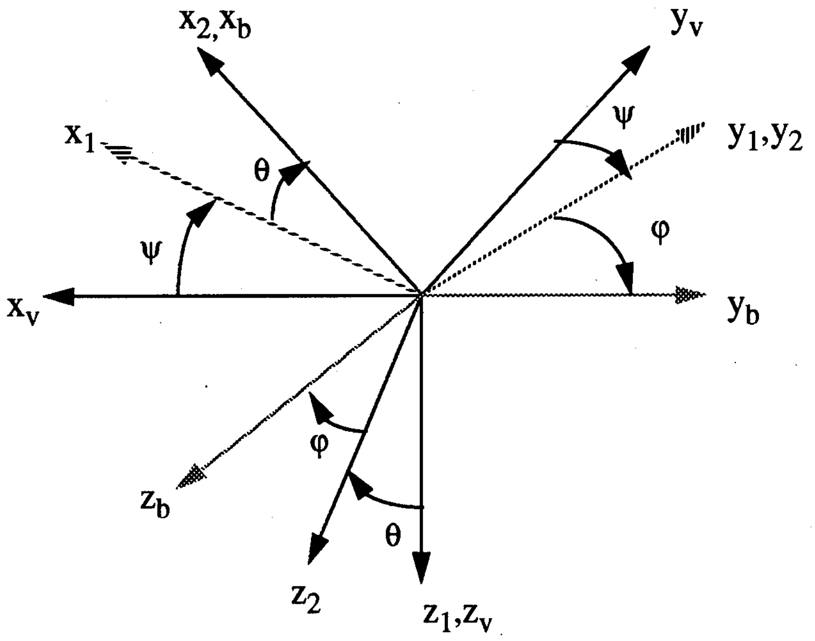


Figure 2.2 Orientation of Vehicle-Carried Vertical-Axis to the Body-Axis System

Based on these three different axis systems, the aircraft dynamics can be described by 12 states, each divided into 4 sets of three variables. The aircraft's rotational velocity is described by p, q, and r. The aircraft's translational motion is detailed with V, α, and β. The vehicle attitude is found through ψ, θ, and φ. And the aircraft's position is described by x, y, and h.

2.1.2 Force Equations

With the different axis systems described, the equations of motion can now be developed completely. The best starting place for this development is at the beginning with Newton's Second Law.

$$\frac{dp}{dt} = F \tag{2.2}$$

or

$$\frac{d}{dt}(mV_c) = F \tag{2.3}$$

In the above equation, p is the linear momentum of the body, m is the total body mass, and V_c is the velocity of the center of mass. For our derivation, mass is considered constant since the fuel weight is small compared to the aircraft weight as a whole. Thus equation 2.3 becomes

$$m\frac{d}{dt}V_c = F \tag{2.4}$$

Furthermore, an aircraft is assumed to behave like a rigid body, with angular accelerations. Thus making it

$$m\frac{d}{dt}V_c\big|_{OXYZ} = m\left[\frac{d}{dt}V_c\big|_{o'xyz} + \omega \times V_c\right] = F \tag{2.5}$$

Using the velocity vector from the aircraft in the body-axis system, $V_c = ui + vj + wk$ and the angular velocity $\omega = pi + qj + rk$. With this information, the force equation now become

$$F_x = m(\dot{u} + qw - rv) \tag{2.6}$$

$$F_y = m(\dot{v} - pw + ru) \tag{2.7}$$

$$F_z = m(\dot{w} + pv - qu) \tag{2.8}$$

2.1.3 Moment Equations

With the force equations defined above, the next step is to derive the moment equations. Again the starting place is with Newton's Law. Applying Newton's Laws to angular momentum, the following equation is developed.

$$\frac{d}{dt}H_c\big|_{OXYZ} = \frac{d}{dt}H_c\big|_{o'xyz} + \omega \times H_c = M \tag{2.9}$$

Given that $H_c = I\omega$ with I being the inertia matrix, the above equation becomes

$$M = I\dot{\omega} + \omega \times I\omega \tag{2.10}$$

Breaking it down into each individual axis, the moment equations are

$$M_x = I_{xx}\dot{p} - (\dot{q} - pr) I_{xy} - (\dot{r} + pq) I_{xz} - (q^2 - r^2) I_{yz} - (I_{yy} - I_{zz})\, qr \tag{2.11}$$

$$M_y = I_{yy}\dot{q} - (\dot{p} - qr) I_{xy} - (\dot{r} - pq) I_{yz} - (r^2 - p^2) I_{xz} - (I_{zz} - I_{xx})\, pr \tag{2.12}$$

$$M_z = I_{zz}\dot{r} - (p^2 - q^2) I_{xy} - (\dot{q} + pr) I_{yz} - (\dot{p} - qr) I_{xz} - (I_{xx} - I_{yy})\, pq \tag{2.13}$$

2.1.4 Euler's Angles

As alluded to above in the reference system section, the different axis systems are related through Euler angle transformations. When transforming coordinates from the vehicle-carried, vertical-axis-system to the body-axis system three different transformations must occur. The first is a rotation about the Z-axis through an angle ψ. The second is a rotation about the new y_1-axis through an angle θ. Last is a rotation about the new x_2-axis through an angle φ. At each rotation, components of a vector expressed in the coordinate frame before and after the rotation are related through a rotation matrix. Namely,

ψ Rotation:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.14}$$

θ Rotation:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{2.15}$$

φ Rotation:

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} \tag{2.16}$$

With the above rotations the next step is to apply them in the specific order to the angular velocity ω. After using the transformation matrices the angular velocities can be expressed in terms of the time derivatives of the Euler angles,

$$p = \dot{\varphi} - \dot{\psi}\sin\theta \tag{2.17}$$

$$q = \dot{\theta}\cos\varphi + \dot{\psi}\cos\theta\sin\varphi \tag{2.18}$$

$$r = \dot{\psi}\cos\theta\cos\varphi - \dot{\theta}\sin\varphi \tag{2.19}$$

The same transformation must also occur with the force on the aircraft due to gravity. Applying the transformations leads to the following equations.

$$F_x = -mg\sin\theta \tag{2.20}$$

$$F_y = mg\cos\theta\sin\varphi \tag{2.21}$$

$$F_z = mg\cos\theta\cos\varphi \tag{2.22}$$

Next, the linear acceleration terms must be found. Applying the transformations will result in the following acceleration equations.

$$\dot{x} = u\cos\psi\cos\theta + v(\cos\psi\sin\theta\sin\varphi - \sin\psi\cos\varphi) + w(\sin\psi\sin\varphi + \cos\psi\sin\theta\cos\varphi) \tag{2.23}$$

$$\dot{y} = u\sin\psi\cos\theta + v\cos\psi\cos\varphi + v\sin\psi\sin\theta\sin\varphi + w(\sin\psi\sin\theta\cos\varphi - \cos\psi\sin\varphi) \tag{2.24}$$

$$\dot{z} = -u\sin\theta + v\cos\theta\sin\varphi + w\cos\theta\cos\varphi \tag{2.25}$$

The final set of equations needed are the angular acceleration equations of the Euler angles as they apply to the aircraft. As the aircraft flies its attitudes will constantly be changing with time. This final set of equations describe the state dynamics of the Euler angles as a function of the vehicle angular relationship p, q, and r.

$$\dot{\theta} = q\cos\varphi - r\sin\varphi \tag{2.26}$$

$$\dot{\psi} = q\sin\varphi\sec\theta + r\cos\varphi\sec\theta \tag{2.27}$$

$$\dot{\varphi} = p + q\sin\varphi\tan\theta + r\cos\varphi\tan\theta \tag{2.28}$$

This completes the transformation from the vehicle-carried, vertical-axis system to the body-axis system through the Euler angles. It is important to note that the only assumptions made up to this point are 1) rigid airframe, 2) Flat earth 3) Axes fixed to the body with the origin at the center of gravity, and 4) Earth-fixed reference is treated as an inertial reference.

2.2 Linearized Equations of Motion

Linear aircraft models are very useful in aircraft control design. The quickness in which control laws can be validated saves many days of evaluation time that would be spent solely on a nonlinear simulation. It is common practice to first derive a control law for the linear model, and then apply it to the nonlinear model. If the controller is robust enough, it will also work for the nonlinear model. Therefore, the next section will discuss the derivation of the linear equations of motion for an aircraft from the nonlinear equations already presented.

2.2.1 Linearized Linear Acceleration Equations

The majority of the forces and moments applied to an aircraft are due to the aerodynamics of the wings, body, and tail surfaces. It would be difficult to express these forces in terms of u, v, and w. On the other hand, it is much easier to express these forces and moments in terms of vehicle velocity V, angle of attack α, and sideslip angle β. As shown in Figure 2.3, we can express the linear velocities (u,v,w) directly in terms of V, α, and β through the following relations.

$$u = V\cos\beta\cos\alpha \tag{2.29}$$

$$v = V\sin\beta \tag{2.30}$$

$$w = V\cos\beta\sin\alpha \tag{2.31}$$

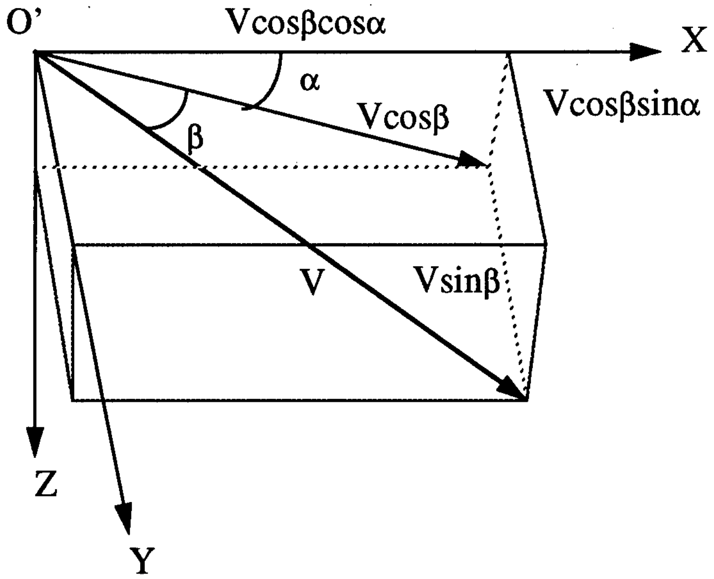


Figure 2.3 Linear Velocities

Furthermore, equations (2.6)-(2.8) can be rewritten as

$$\dot{u} = rv - qw - g\sin\theta + \frac{F_x}{m} \tag{2.32}$$

$$\dot{v} = pw - ru + g\sin\varphi\cos\theta + \frac{F_y}{m} \tag{2.33}$$

$$\dot{w} = qu - pv + g\cos\theta\cos\varphi + \frac{F_z}{m} \tag{2.34}$$

The above linear accelerations can also be derived in terms of V, α, and β by differentiating equations (2-29)-(2-31) with respect to time to obtain

$$\dot{u} = \dot{V}\cos\alpha\cos\beta - V\dot{\alpha}\sin\alpha\cos\beta - V\dot{\beta}\cos\alpha\sin\beta \tag{2.35}$$

$$\dot{v} = \dot{V}\sin\beta + \dot{\beta}V\cos\beta \tag{2.36}$$

$$\dot{v} = \dot{V}\sin\alpha\cos\beta + V\dot{\alpha}\cos\alpha\cos\beta - V\dot{\beta}\sin\alpha\sin\beta \tag{2.37}$$

Combining the last two sets of equations and expanding them into separate components, the following equations are determined.

$$\dot{V} = -g\,(\cos\alpha\cos\beta\sin\theta - \sin\beta\sin\varphi\cos\theta - \sin\alpha\cos\beta\cos\theta\cos\varphi) + \frac{F_x}{m}\cos\alpha\cos\beta + \frac{F_y}{m}\sin\beta + \frac{F_z}{m}\sin\alpha\cos\beta \tag{2.38}$$

$$\dot{\alpha} = -r\sin\alpha\tan\beta + q + g\,(\frac{\sin\alpha\sin\theta + \cos\alpha\cos\theta\cos\varphi}{V\cos\beta}) - p\cos\alpha\tan\beta - \frac{F_x}{mV\cos\beta}\sin\alpha + \frac{F_z}{cV\cos\beta}\cos\alpha \tag{2.39}$$

$$\dot{\beta} = \frac{g}{V}(\cos\alpha\sin\beta\sin\theta + \cos\beta\sin\varphi\cos\theta - \sin\alpha\sin\beta\cos\theta\cos\varphi) - r\cos\alpha + p\sin\alpha - \frac{F_x\cos\alpha\sin\beta}{mV} + \frac{F_y\cos\beta}{mV} - \frac{F_z\sin\alpha\sin\beta}{mV} \tag{2.40}$$

Now we begin the process of linearizing the above equations. This is accomplished by assuming small perturbations for all the variables. Specifically the linear velocities are now $V=V_0+\Delta V$, $\alpha=\alpha_0+\Delta\alpha$, and $\beta=\beta_0+\Delta\beta$. Likewise the angular velocities are replaced by $p=p_0+\Delta p$, $q=q_0+\Delta q$, and $r=r_0+\Delta r$. The force equations become $F_x=F_{xo}+\Delta F_x$, $F_y=F_{yo}+\Delta F_y$, and $F_z=F_{zo}+\Delta F_z$. Lastly the airplane attitudes become $\theta=\theta_0+\Delta\theta$, $\varphi=\varphi_0+\Delta\varphi$, and $\psi=\psi_0+\Delta\psi$. In the above expression, the subscript $(-)_0$ refers to the trim value, and the Δ-term is its perturbation. In order to finish the linearization, a specific flight condition must be chosen. For this section a symmetric climb condition is chosen. This flight condition has the following trim values equal to zero: $\beta_o = p_o = q_o = r_o = F_{yo} = \varphi_o = \psi_o = 0$.

Substituting the perturbation equations into equations (2.38)-(2.40), the linearized equations of motion can be developed. The linearization is done by neglecting higher-order terms such as $\Delta\beta\Delta\alpha$ which is set equal to zero, and at the same time assuming small angle approximations where $\cos\Delta\beta \cong 1$, and $\sin\Delta\beta \cong \Delta\beta$. After long mathematical manipulations the linearized equations of motion are

$$\Delta\dot{V} = -g\cos(\theta_o - \alpha_o)\Delta\theta + \frac{\Delta F_x \cos\alpha_o}{m} + \frac{\Delta F_z \sin\alpha_o}{m} \qquad (2.41)$$

$$\Delta\dot{\alpha} = \Delta q + \frac{g\Delta\theta}{V_o}(-\cos\alpha_o \sin\theta_o + \sin\alpha_o \cos\theta_o) - \frac{\Delta F_x \sin\alpha_o}{V_o} + \frac{\Delta F_z \cos\alpha_o}{V_o} \qquad (2.42)$$

$$\Delta\dot{\beta} = \Delta p \sin\alpha_o - \Delta r \cos\alpha_o + \frac{\Delta F_y}{mV_o} + \frac{g\Delta\theta}{V_o}\cos\theta_o \qquad (2.43)$$

The next step is to convert the forces from directional forces in the F_x and F_z terms and convert them to better known aircraft forces such as Thrust, Lift, and Drag as Figure 2.3 shows the relationship between F_x, F_z, and Thrust, Lift and Drag.

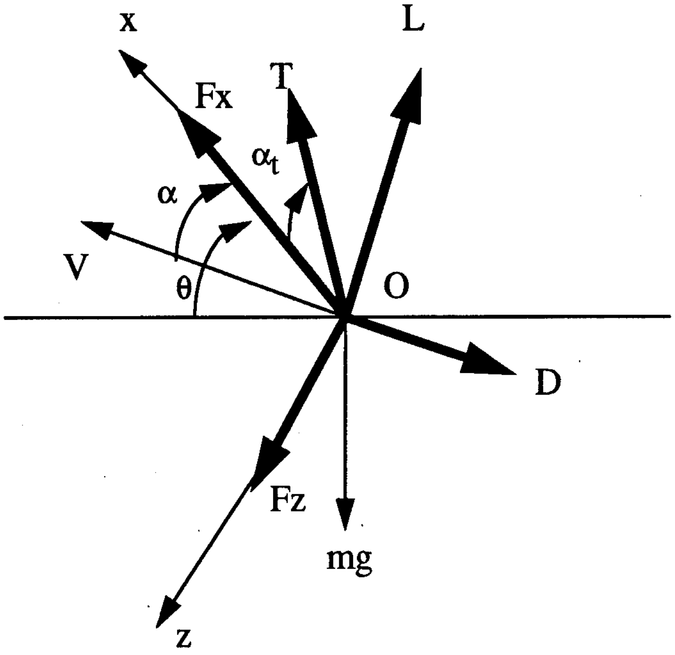


Figure 2.4 F_x and F_z Components in terms of L, D, and T

It is from this figure that the following equations are derived.

$$F_x = L\sin\alpha - D\cos\alpha - T\cos\alpha_t \tag{2.44}$$

$$F_z = -L\cos\alpha - D\sin\alpha - T\sin\alpha_t \tag{2.45}$$

Small perturbation theory is again used to define L=L$_0$+ΔL, D=D$_0$+ΔD, and T=T$_0$+ΔT. Expanding each term, keeping first-order effects, and using small angle approximations, we obtain a new set of equations for ΔF$_x$ and ΔF$_z$. These equations are then re-substituted into the linearized equations of the linear acceleration equations (2.41)-(2.43) to give us the final form of the linearized acceleration equations.

$$\Delta\dot{V} = -g\cos(\theta_o - \alpha_o)\Delta\theta + \frac{L_o\Delta\alpha}{m} - \frac{\Delta D}{m} + \frac{\Delta T}{m}\cos(\alpha_o + \alpha_t) \tag{2.46}$$

$$\Delta\dot{\alpha} = \Delta q - \frac{g}{V_o}\sin(\theta_o - \alpha_o)\Delta\theta - \frac{\Delta\alpha}{mV_o}D_o - \frac{\Delta L}{mV_o} - \frac{\Delta T}{mV_o}\sin(\alpha_o + \alpha_t) \tag{2.47}$$

$$\Delta\dot{\beta} = \Delta p\sin\alpha_o - \Delta r\cos\alpha_o + \frac{g}{V_o}\cos\theta_o\Delta\varphi + \frac{\Delta Y}{mV_o} \tag{2.48}$$

2.2.2 Linearized Angular Acceleration Equations

With the linear acceleration terms derived, the next step is to derive the linearized angular acceleration equations. Again a steady-level climb will be considered, with angular velocities redefined as p=p$_0$+Δp, q=q$_0$+Δq, and r=r$_0$+Δr. The moments are also defined using small perturbations about a trim condition in order to make *L*=*L*$_0$+Δ*L*, *M*=*M*$_0$+Δ*M*, and *N*=*N*$_0$+Δ*N*. For a steady-level climb, p$_0$=q$_0$=r$_0$=0 and *L*$_0$=*M*$_0$=*N*$_0$=0. Substituting the above expressions into the nonlinear angular acceleration equations (2.11)-(2.13) and using small angle approximations as well as neglecting higher order terms, the linearized angular acceleration equations are

$$\Delta L = I_{xx}\Delta\dot{p} - I_{xz}\Delta\dot{r} \tag{2.49}$$

$$\Delta M = I_{yy}\Delta\dot{q} \tag{2.50}$$

$$\Delta N = I_{zz}\Delta\dot{r} - I_{xz}\Delta\dot{p} \tag{2.51}$$

2.2.3 Linearized Euler Angles

In order to linearize the Euler angles, the perturbation theory applied to the previous equations will be used again. The perturbations will be applied to equations (2.17)-

(2.19) with the same simplified assumptions of small angle approximation and deletion of higher order terms. Making these substitutions and applying the small perturbation assumptions yields the following equations

$$\Delta\dot{\theta} = \Delta q \tag{2.52}$$

$$\Delta\dot{\psi} = \frac{\Delta r}{\cos\theta_o} \tag{2.53}$$

$$\Delta\dot{\varphi} = \Delta p + \tan\theta_o \Delta r \tag{2.54}$$

2.3 Nonlinear Simulation Model

The twelve nonlinear simultaneous second-order differential equations derived in the previous sections are implemented in the simulation in the form of a MATLAB special function, or *s-function.* The exact methodology of the MATLAB function is not important. What is important is the basic format of the function, as well as its integration with the simulation program SIMULINK. The next sections will outline these points.

2.3.1 Component Integration--the S-Function

Systems that are saved in the SIMULINK window as s-functions are treated in a similar manner as MATLAB treats m-functions. In MATLAB, the program calls a certain subprogram to be run. Likewise, SIMULINK calls a certain subprogram to be run. The subprogram must be in the same directory, but that is the only stipulation on running it. The s-function makes it possible for the user to write specific routines for the simulation. The F-15 dynamics that were acquired from Wright-Patterson AFB are located in two primary routines, **f25aero** and **f25eng**, quantifying the aircraft aerodynamic and propulsion characteristics respectively. The routine **atmos** calculates the current atmospheric parameters such as density and temperature to be used by the previous two routines. All three of these routines were provided in FORTRAN format from the AIAA Design Challenge [5]. The s-function used for the open-loop simulation is shown in Appendix A.

Initial conditions for the aircraft states are also established inside the s-function. These particular values are very important when determining the aircraft trim points for a given flight condition. The s-function also makes it possible to determine the output variables. For the open-loop case the 12 states were chosen in an effort to validate the equilibrium state responses at the two chosen flight conditions.

2.3.2 The SIMULINK Model

As far as the SIMULINK model is concerned, its job is made easier by the s-function. SIMULINK only needs to set up the inputs and outputs of the s-function. A clock is also needed in order to have a time response for the actual simulation. That is all there is to the SIMULINK model. As seen in Figure 2.5 the inputs to the s-function are chosen based on the inputs needed to run the s-function. Likewise the outputs were determined based on the outputs deemed necessary by the user when he creates the s-function.

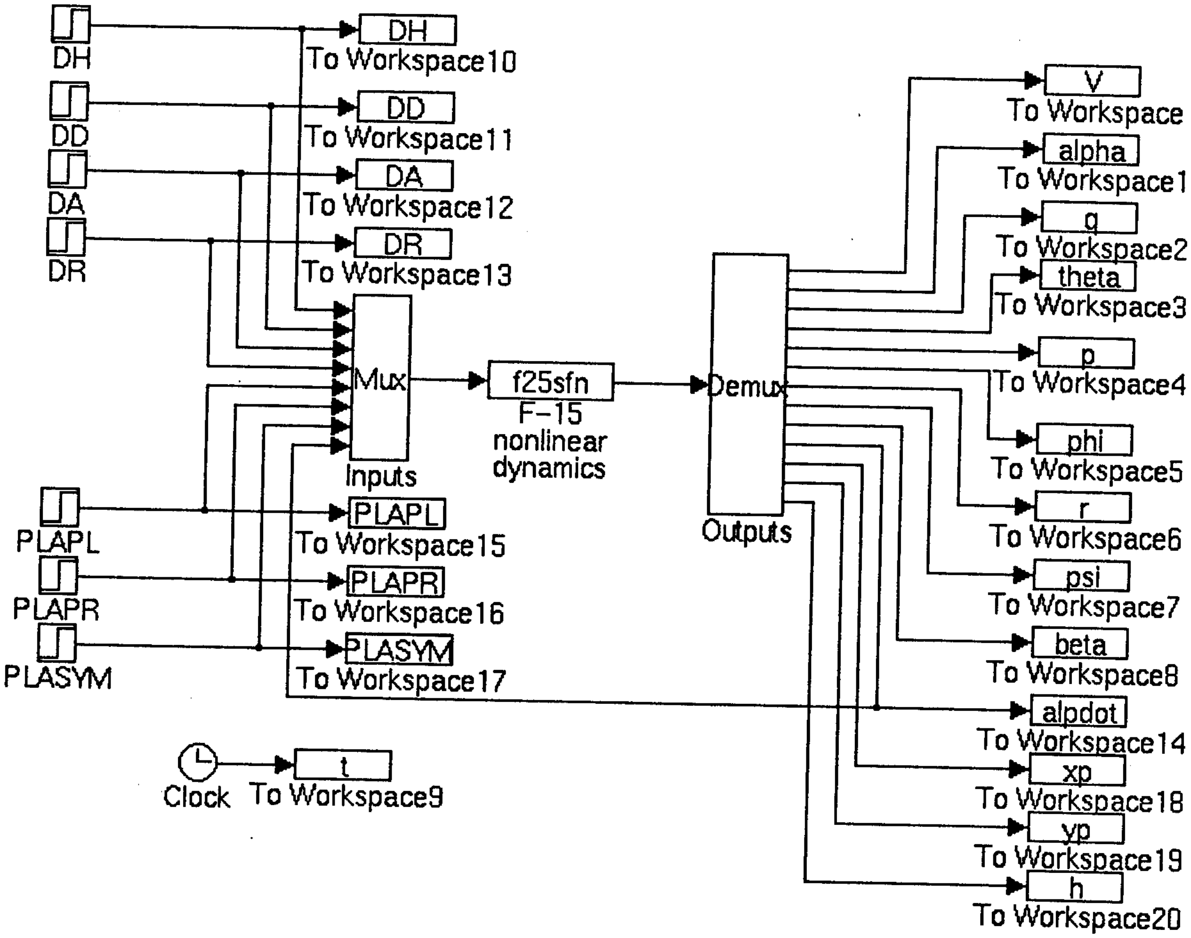


Figure 2.5 SIMULINK model

The inputs for the s-function **f25sfn** are typical inputs into most aircraft. These include the horizontal stabilizers both differential (DD) and symmetric (DH). Ailerons (DA) and rudder (DR) are also needed. And finally, the thrust from the engines is also an input. The inputs PLAR and PLAL refer to the right and left power level angles respectively, PLASYM assures that both are set equally. As for the outputs, they were chosen to be the 12 states.

2.4 Linearization of the Model

It has already been discussed that a linear model facilitates the design process. Granted several different flight points must be designed for in order to use the controller in the nonlinear environment. Therefore, different flight points must be chosen. In previous work, Lt. James Dutton chose two flight points in order to set up his control law [10]. In order to further his research and to validate my own, I chose to use the same flight points. The original decision for the flight points was not just a guess. They were chosen through the use of a program that calculates the equilibrium states of an aircraft, namely **trim**. This program doesn't allow any variable to remain constant. Instead it varies the trim values in an effort to find an equilibrium point. By allowing these variations, the two trim points were determined for this report. Table 2.1 shows how the initial guess settled at the trim point for flight point #1.

Table 2.1: Trim Data FP1 (9,800ft, 0.5 M)

Parameter	Units	Initial Guess	Trim Value	State Derivative
V	ft/s	539.08	539.08	-7.1636^{-9}
α	rad	0.0801	0.0801	-1.8594^{-9}
q	rad/s	0	-2.79^{-19}	-2.1328^{-5}
θ	rad	0.0801	0.0803	-2.7949^{-19}
p	rad/s	0	-2.54^{-20}	2.3895^{-19}
φ	rad	0	-1.02^{-18}	-2.8053^{-20}
r	rad/s	0	-3.22^{-20}	-1.8936^{-20}
ψ	rad	0	-1.66^{-13}	-3.2388^{-20}
β	rad	0	-8.41^{-21}	-2.9103^{-20}
δ_h	deg	-2.827	-2.880	N/A
δ_{PLA}	deg	37.4	37.4	N/A
δ_A	deg	0	0	N/A
δ_R	deg	0	0	N/A
γ	rad	0	0.0002	N/A

Once the two flight points have been determined, linear model describing the F-15 at a specific flight point can be obtained in the form of the following matrix equations.

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.55}$$

$$y(t) = Cx(t) + Du(t) \tag{2.56}$$

The data collected from the trim program was converted into the system of equations above, and then saved under two different files to be used later. The first flight point is at 9,800 ft flying at Mach 0.5 which translates to 539.0 ft/sec. The second flight point is at 30,000 ft again flying at Mach 0.5. For this altitude, the velocity is 497 ft/sec. The files that contain the matrix information are **trim539ss** and **trim497ss** for flight points FP1 and FP2 respectively. These two files were created by the MATLAB function **linmod**, which took the aircraft data and translated it into the respective state space models.

Chapter 3 F-15 Open-Loop Analysis

The purpose of this chapter is to discuss the open-loop analysis of the F-15. Open loop analysis is very important at the initial stage of a control-law design. If an aircraft is inherently unstable, it will manifest itself in the open-loop simulation. This will in turn provide engineers with knowledge concerning which modes will need what type of modifications in order to control the aircraft.

Open-loop analysis is also important to linear simulation verification. It has been previously stated that linear design points are used by engineers in order to design an adequate control law. If responses from the linear simulation and the nonlinear simulation are very different, the control law designed for the nonlinear simulation simply will not work. Therefore, by looking at the open-loop time responses of the F-15, the designer will know whether or not the linear simulation accurately describes the nonlinear case.

3.1 Linearized Model

The nonlinear model for the F-15 has already been shown in detail. In order to run a similar simulation for the linear case one must create a separate SIMULINK file. In Figure 3.1 one sees the SIMULINK file used to test the linear case.

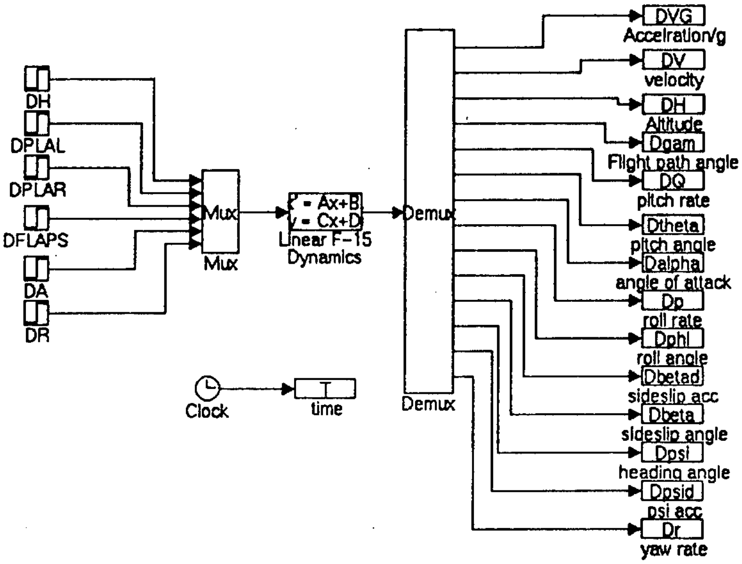


Figure 3.1 Linear SIMULINK model

Notice that the inputs and outputs are exactly the same as in the nonlinear case depicted in Figure 2.5 for consistency. As long as the inputs are the same and the outputs measured are the same, it stands to reason that the linear model and the nonlinear model have a good chance of having the same time responses.

3.2 Evaluation of the Linearized Model

The nonlinear and linear models were run with a variety of inputs in order to validate the two systems. As you will see in the next few pages, the linear simulation accurately describes the nonlinear dynamics found in the F-15. For most cases, nonlinear model had more fluctuations, but the overall responses of the linear simulations were accurate enough to be used for the control-law application.

3.2.1 Short-Period Time Response

The short period mode was excited by inputting a 20-second pulse of 2 degrees down on the elevator of the aircraft at both flight points.The aircraft time responses are found in Figure 3.2 through Figure 3.5. In these figures, the dotted line represents the linear response and the solid line is the nonlinear time response.

3.2.2 Roll Time Response

For Flight Point 1, a pulse was applied to the right aileron down one degree for 20 seconds in order to check the roll mode of the aircraft. Again the linear simulation closely matched the nonlinear case. For Flight Point 2, the right aileron was deflected down 2 degrees for 20 seconds. The linear and nonlinear time responses are comparable. These time responses can be found in Figures 3.6-3.9, with the dotted lines again being the linear case and the solid lines being the nonlinear case.

3.2.3 Dutch-Roll Time Response

For both flight points, the rudder was deflected 1 degree to the left for 20 seconds in order to check the Dutch roll mode of the F-15. Both the nonlinear case and the linear case had similar time responses. In Figures 3.10-3.13, the time responses are shown with the dotted line depicting the linear case and the solid showing the nonlinear case.

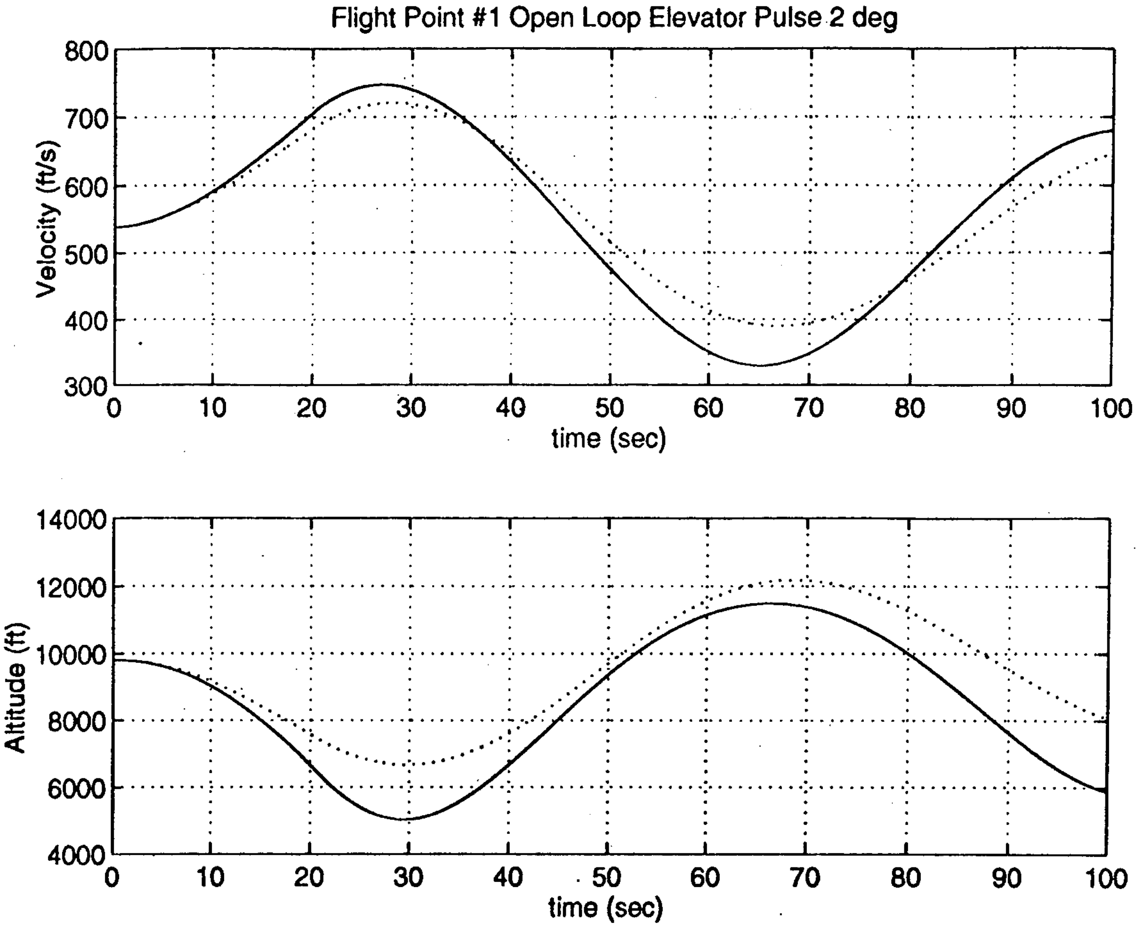


Figure 3.2 Aircraft Responses to a 20-second Elevator Pulse of 2^{o}.

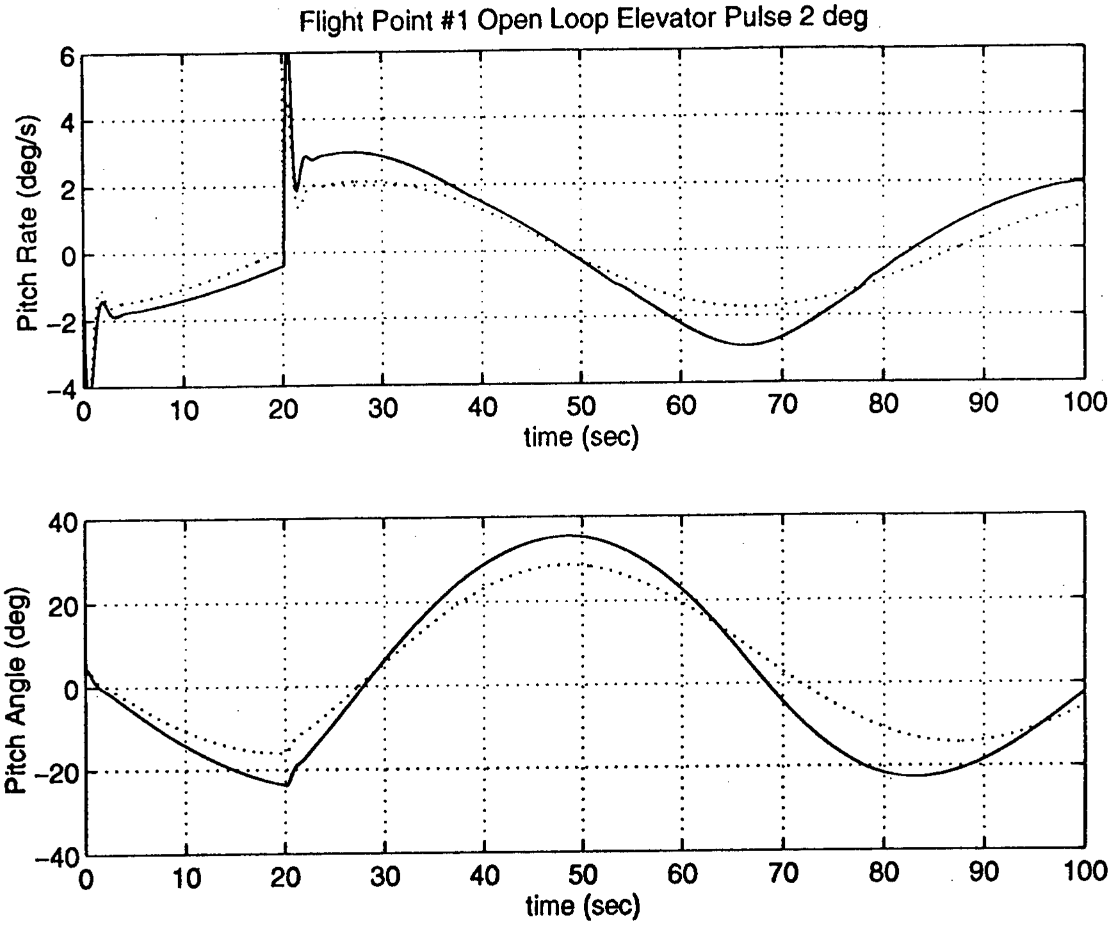


Figure 3.3 Aircraft Responses to a 20-second Elevator Pulse of 2^{o}.

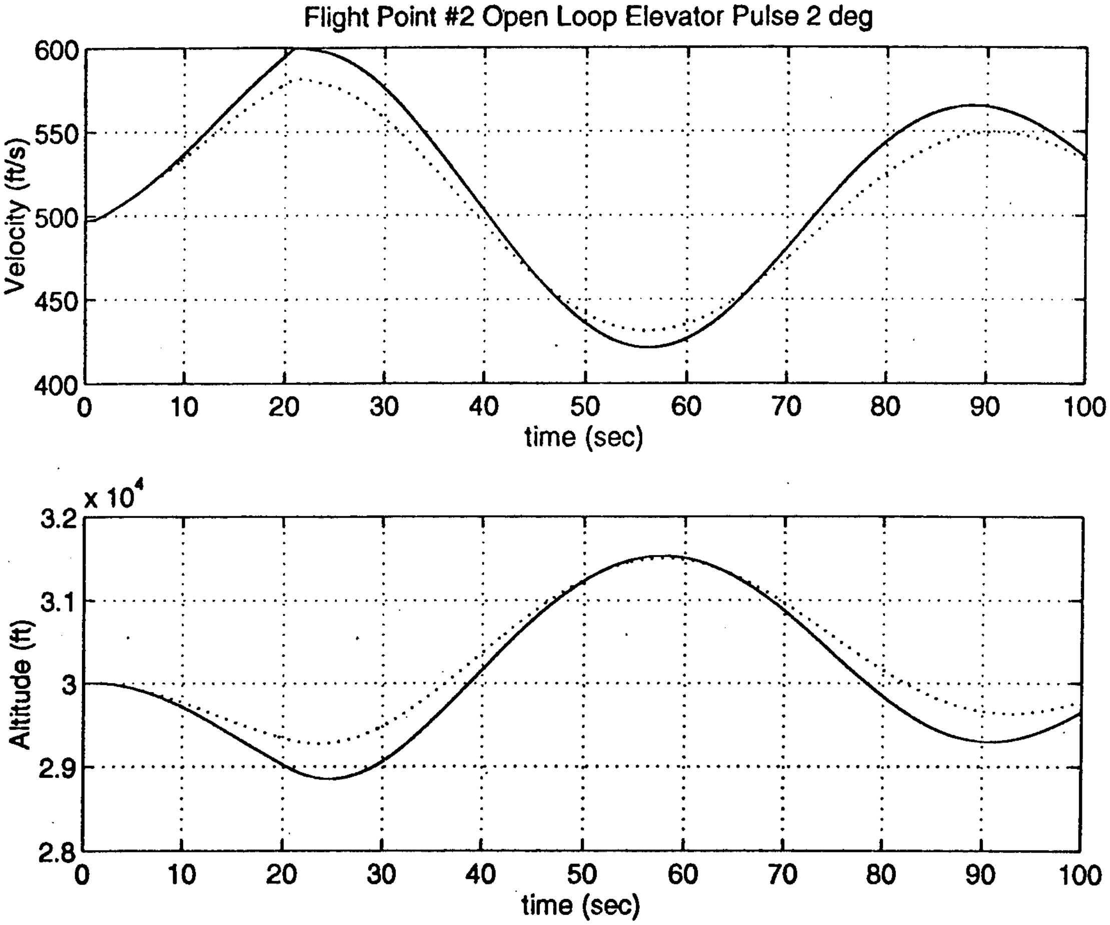


Figure 3.4 Aircraft Responses to a 20-second Elevator Pulse of 2°.

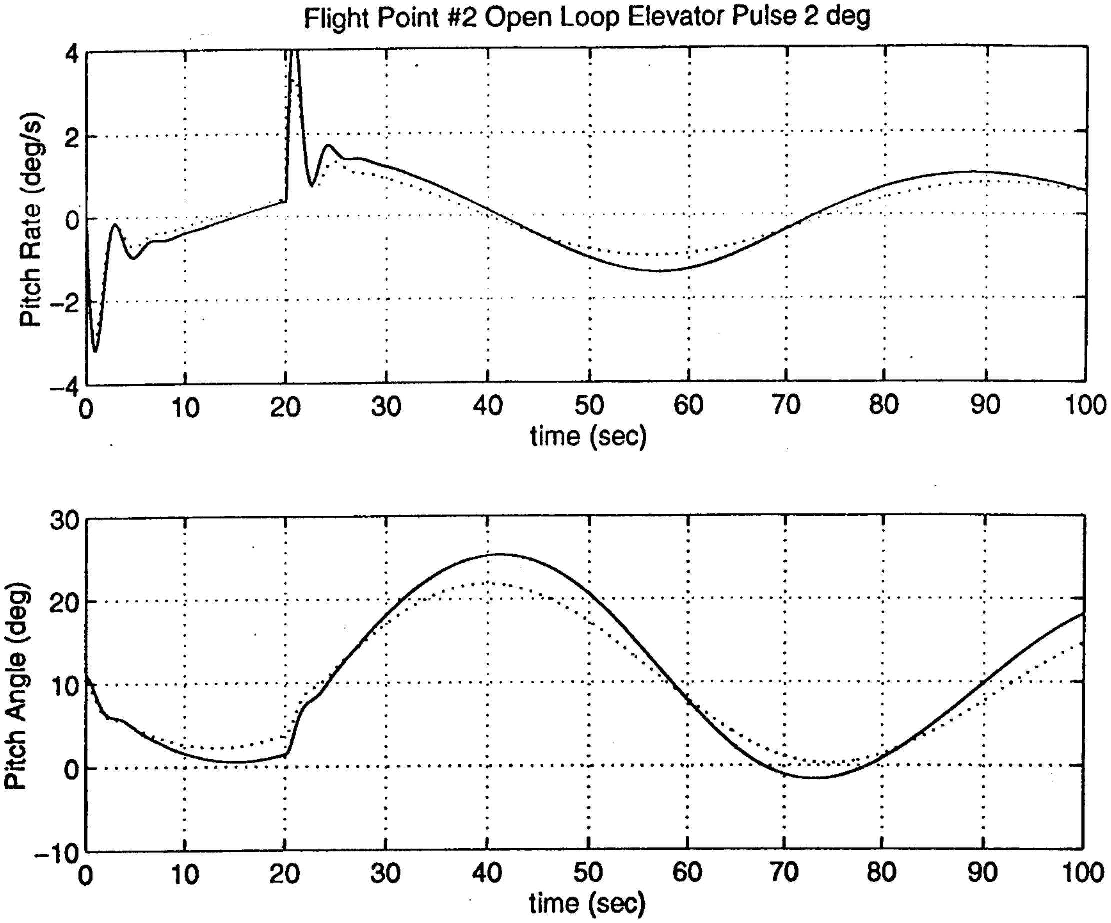


Figure 3.5 Aircraft Responses to a 20-second Elevator Pulse of 2°.

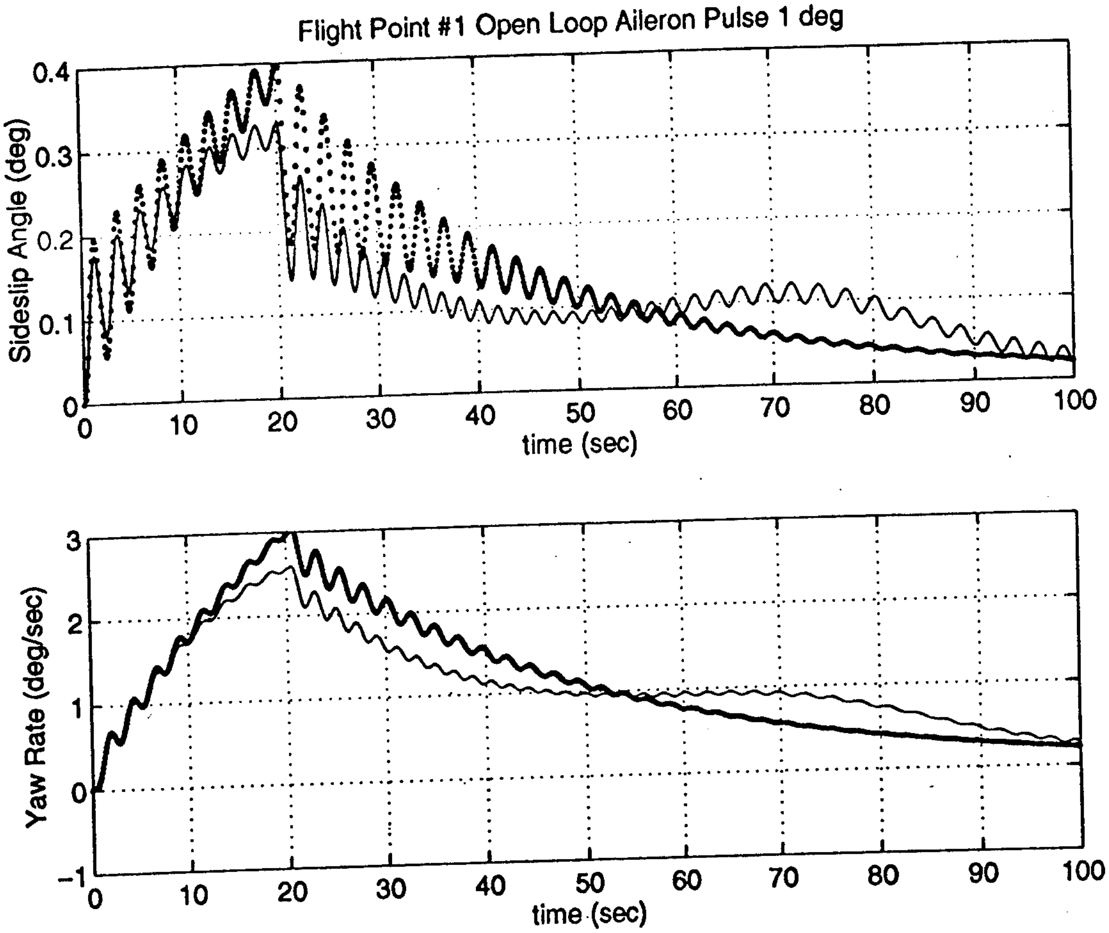


Figure 3.6 Aircraft Responses to a 20-second Aileron Pulse of 1^o.

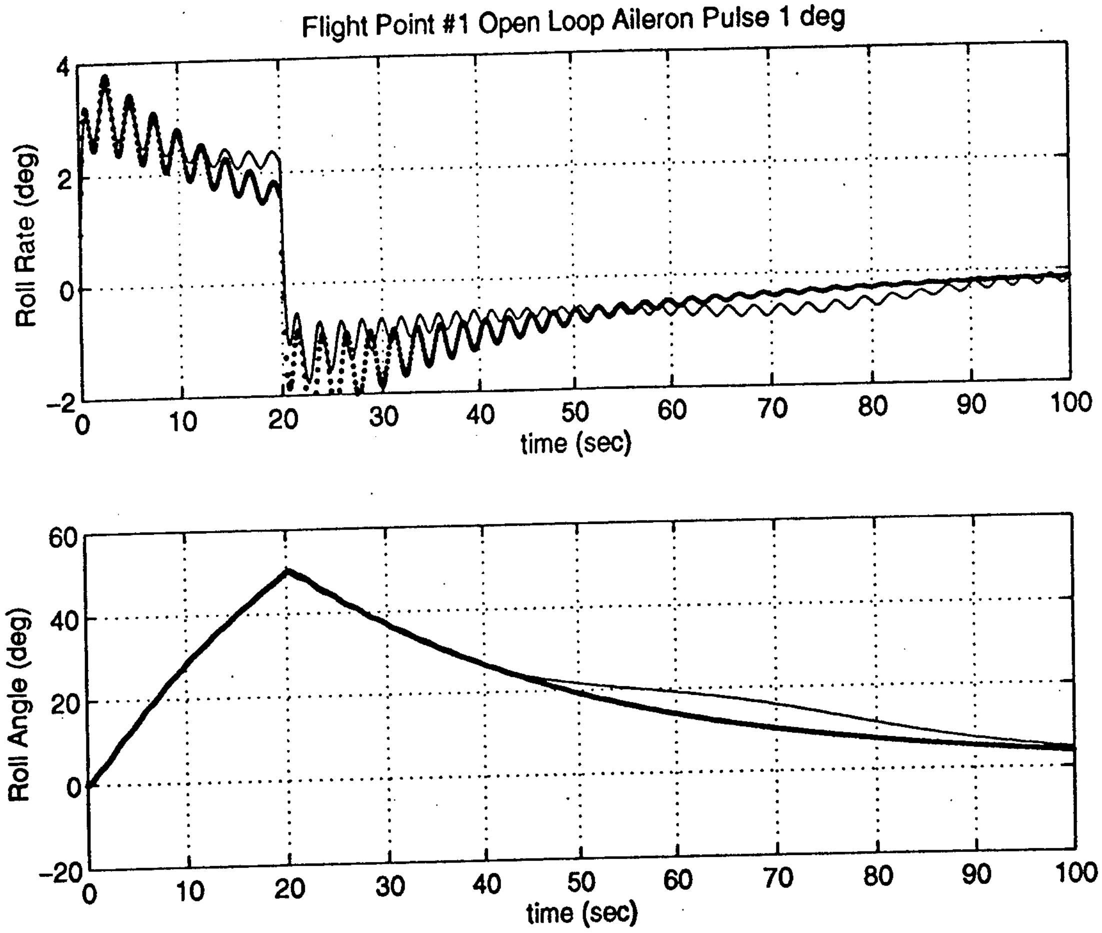


Figure 3.7 Aircraft Response to a 20-second Aileron Pulse of 1°.

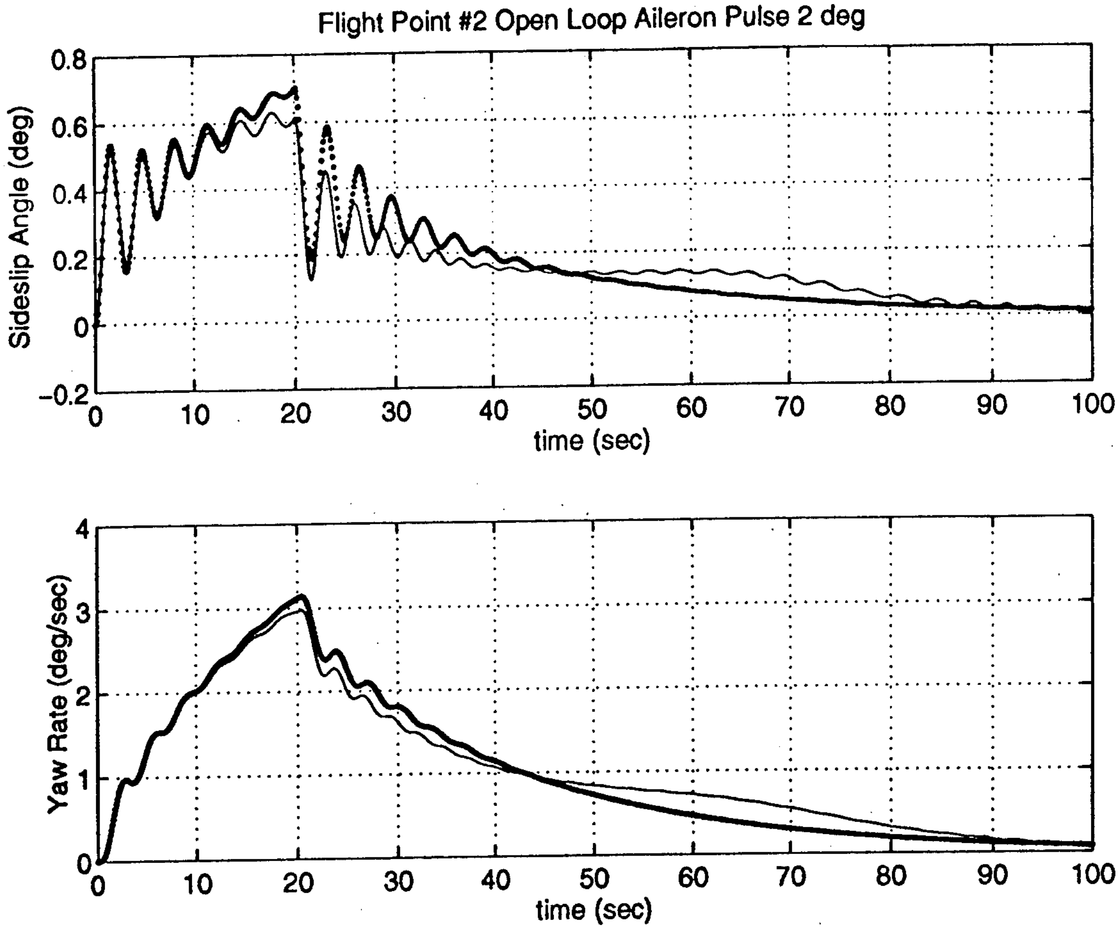


Figure 3.8 Aircraft Response to a 20-second Aileron Pulse of 2^{o}

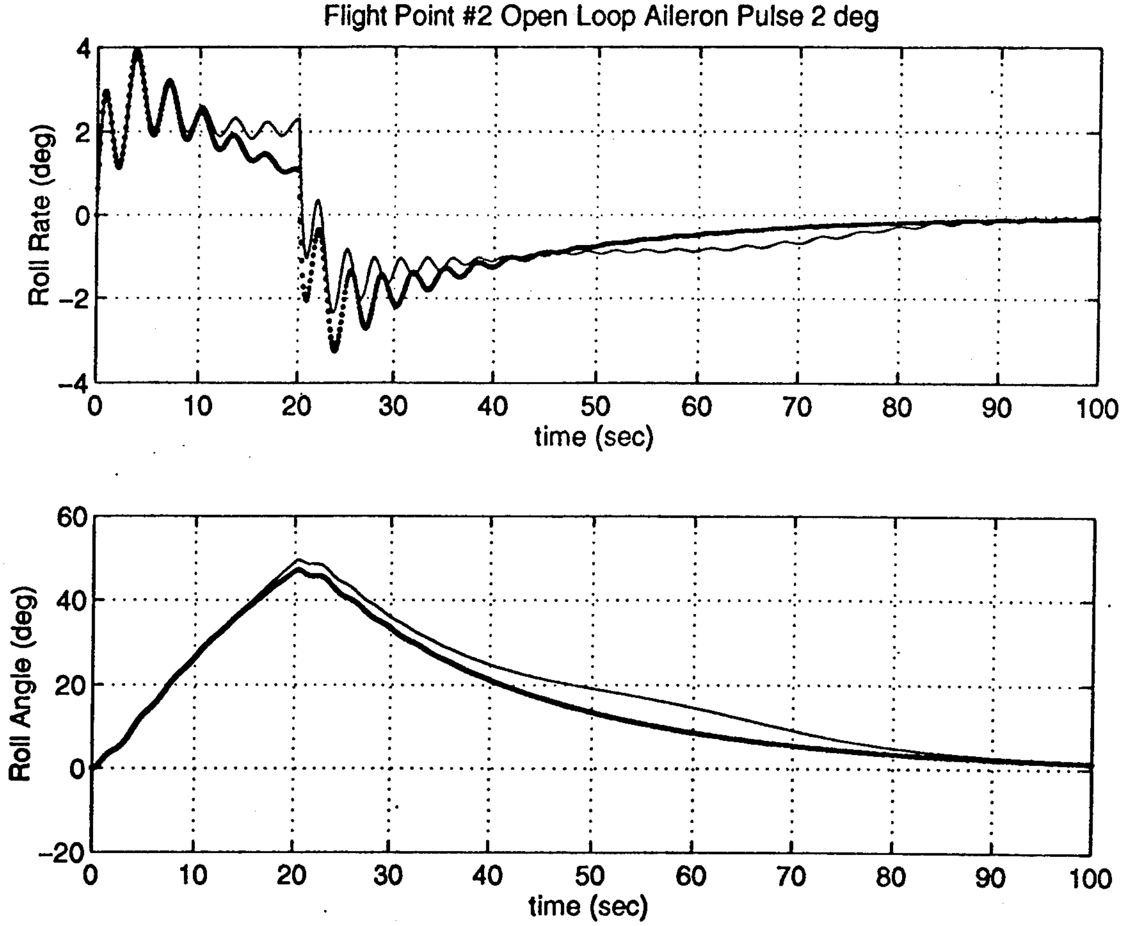


Figure 3.9 Aircraft Response to a 20-second Aileron Pulse of 2°.

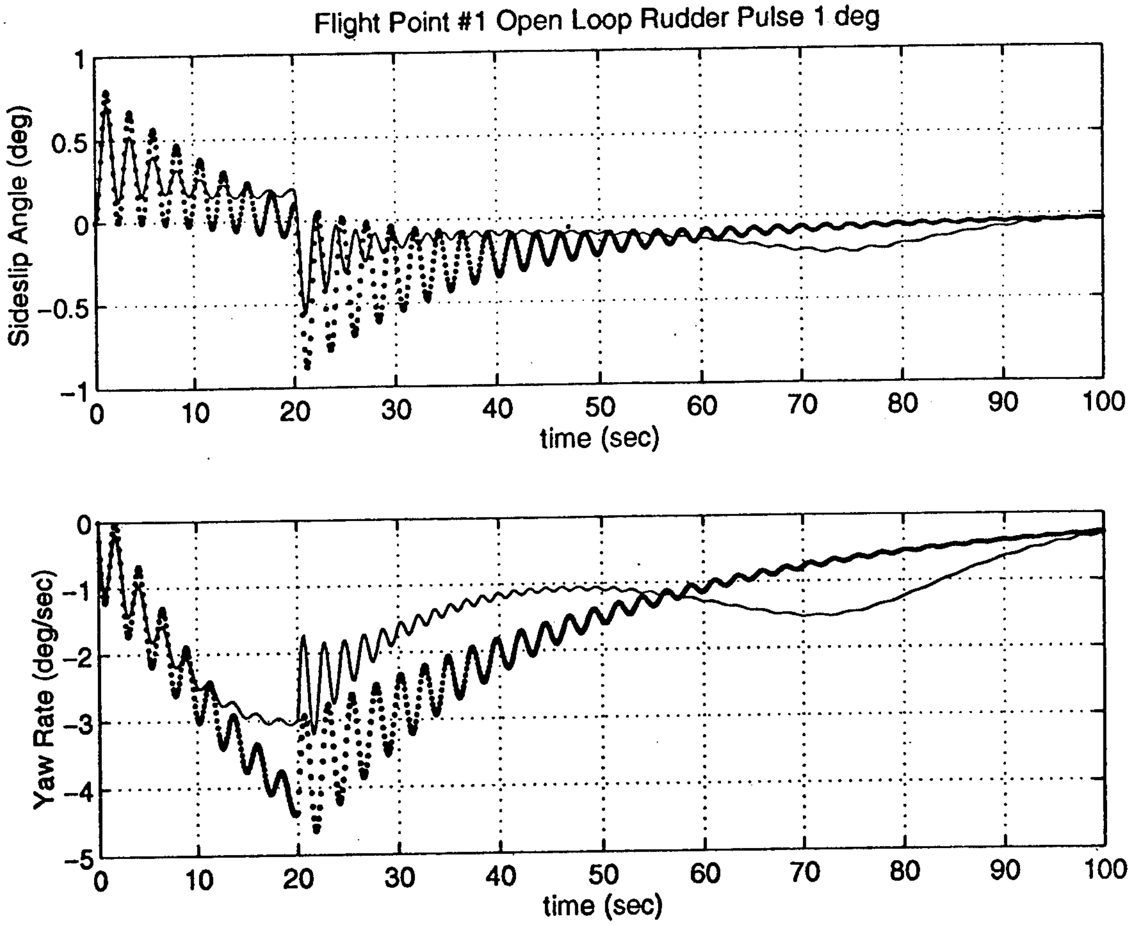


Figure 3.10 Aircraft Response to a 20-second Rudder Pulse of 1^{o}.

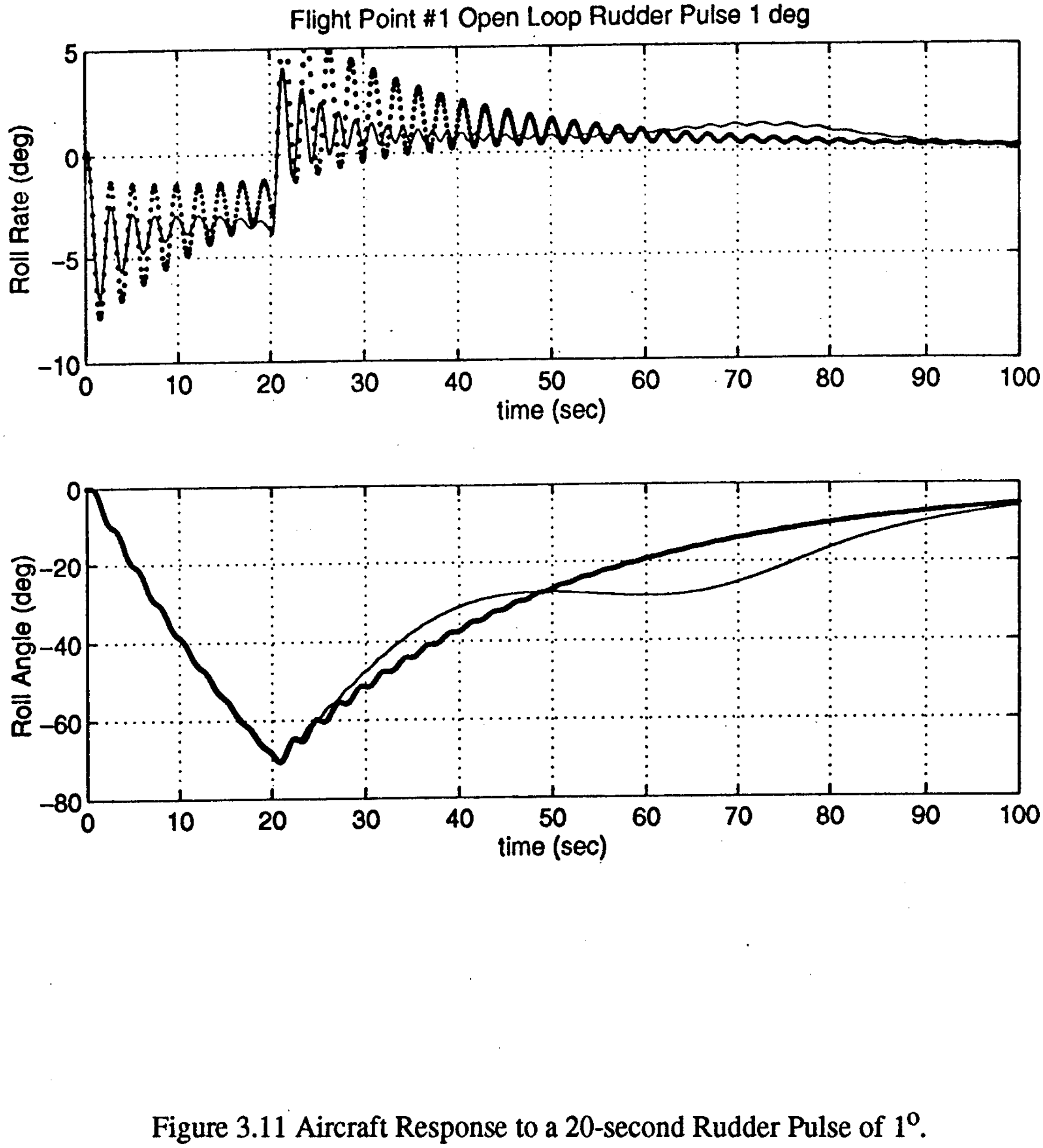


Figure 3.11 Aircraft Response to a 20-second Rudder Pulse of 1^{o}.

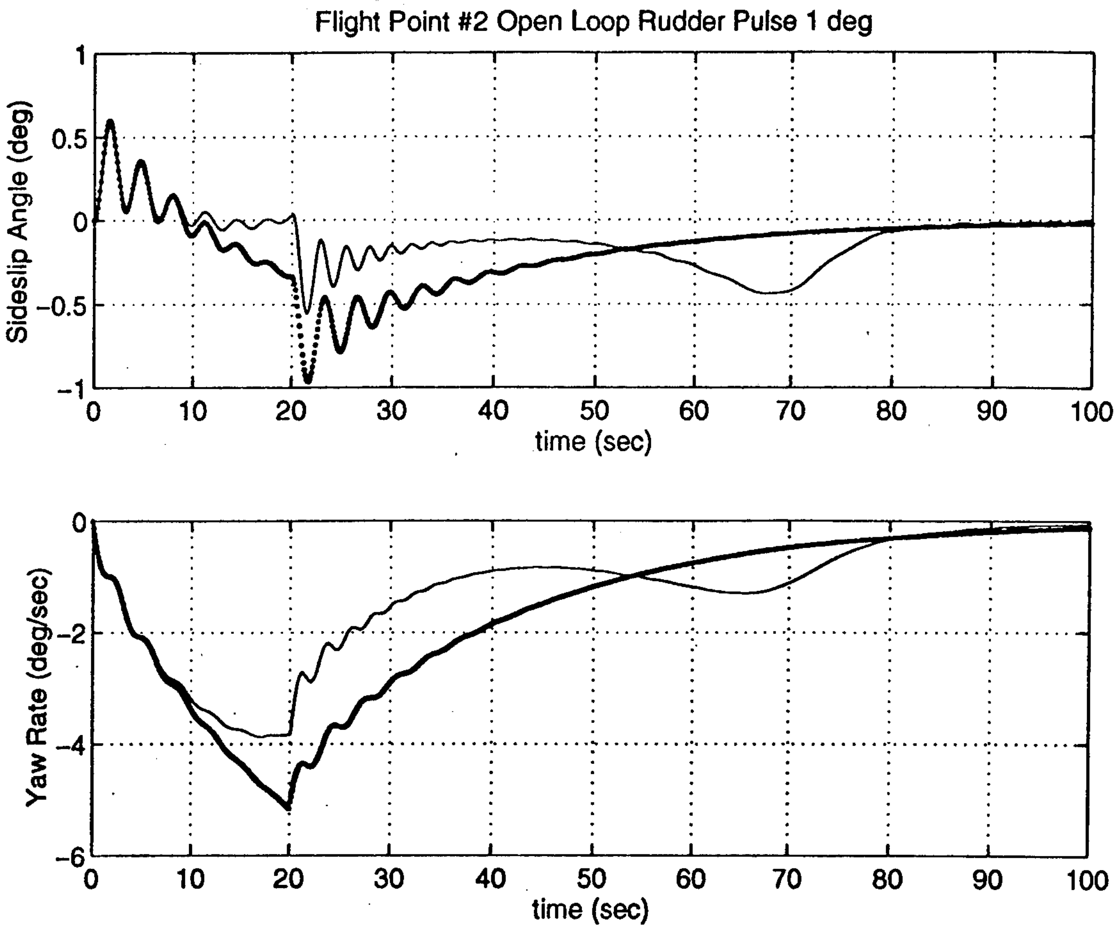


Figure 3.12 Aircraft Response to a 20-second Rudder Pulse of 1^{o}.

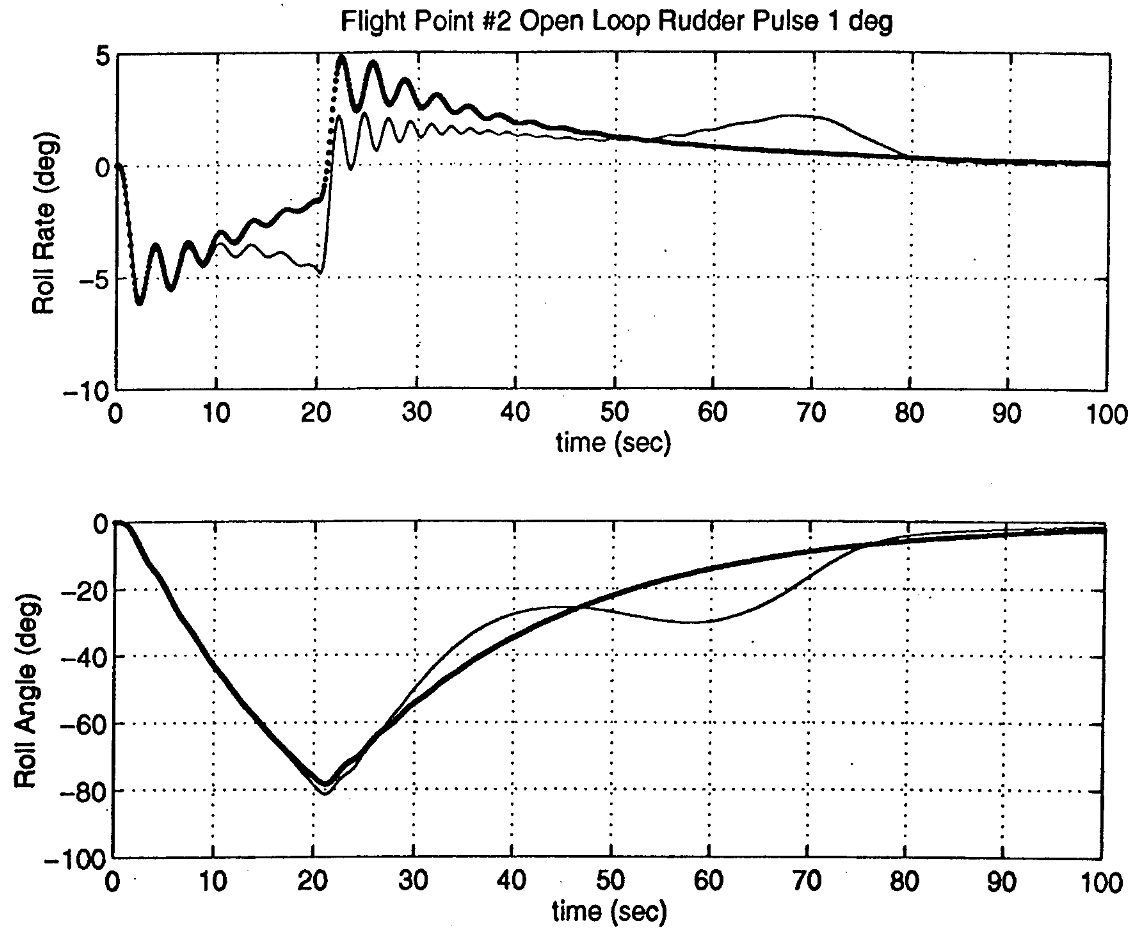


Figure 3.13 Aircraft Response to a 20-second Rudder Pulse of 1^{o}.

Chapter 4 Lateral Control Using TECS

4.1 Background

Modern control theory have focused on the improvement of the controller performance and robustness to plant uncertainties. Such multivariable control synthesis includes linear quadratic Gaussian/loop transfer recovery (LQG/LTR), H_2, and μ synthesis. However, these techniques are of high order, and hence not very cost effective. In order to lower the cost, classical design procedures based on single-loop closure and root locus have been applied to multi-loop control systems. However, these designs can have low performance and poor robustness to model uncertainties. The trade-off between performance and robustness is not a simple process and can be very time consuming. Furthermore, single-loop analysis does not incorporate the multivariable aspects of a design. This fact will also limit the maximum achievable performance. Single-loop design is easier, but not as effective nor as robust. Multivariable design would have better performance and robustness, but is of high order.

The TECS controller is a combination of the simplicity of the single-loop design, linked with the performance and robustness of the multivariable methods. Single-loop control analysis falls short where cross-coupling effects are concerned. For instance, most single-loop controller have a separate loop for the throttle and the elevator. This results in either over or under control of the surfaces in order to perform flight maneuvers. The TECS controller works by converting energy states. It positions the control surfaces simultaneously in order to achieve the maximum performance with little or no conflicting commands from the control surfaces. The exact manner is described in the next section.

4.2 Development of the TECS Concept

As stated above, the TECS controller functions by altering the energy states of the aircraft. Specifically, this means trading off potential or kinetic energy in order to perform a specific flight function. The key equations governing the total energy control concept were developed in [2 4 7 8]. They are reviewed here for completeness and to define the structure of the control system.

The total energy E(t) of the aircraft treated as a point mass is given by

$$E(t) = \frac{1}{2}m(t)V^2(t) + m(t)gh(t) \qquad (4.1)$$

where V(t) is the aircraft's total velocity, m(t) is the mass of the aircraft, h(t) is the altitude, and g is the gravitational acceleration. The total energy rate is then given by differentiating equation (4.1) with respect to time and assuming small flight path angle and that the aircraft mass is constant or slowly varying.

$$E'(t) = m(t)gV(t)\left(\frac{\dot{V}(t)}{g} + \gamma(t)\right) \qquad (4.2)$$

Now, the equation can be rewritten in terms of thrust required, T_{req}. Using the equations of motion developed in Chapter 2 we obtain

$$T_{req}(t) = m(t)g\left(\frac{\dot{V}(t)}{g} + \gamma(t)\right) + D(t) \qquad (4.3)$$

Assuming that the thrust is trimmed against the drag, and that the variation in drag is slow, then the change in thrust required is proportional to the change in energy divided by the aircraft velocity. This in turn leads directly to

$$\Delta T_{req}(t) = m(t)g\left(\frac{\dot{V}(t)}{g} + \gamma(t)\right) \qquad (4.4)$$

This last equation is important because it shows that the total energy of the aircraft can be regulated directly using thrust. In essence, the energy level of the aircraft is controlled by the throttle. It also shows that the distribution of the energy cannot be controlled by the throttle, hence the need for the elevator. The elevator distributes the energy available between kinetic and potential energies.

With the two control surfaces having their defined roles, integral and proportional control can be applied to the throttle and elevator. The following equations result from this application with δ_t and δ_e being the throttle and elevator commands respectively.

$$\delta_t = mg\left(K_{TP} + \frac{K_{TI}}{s}\right)\left(\frac{\dot{V}_\varepsilon}{g} + \gamma_\varepsilon\right) \qquad (4.5)$$

$$\delta_e = \left(K_{EP} + \frac{K_{EI}}{s}\right)\left(\frac{\dot{V}_\varepsilon}{g} - \gamma_\varepsilon\right) \qquad (4.6)$$

where $\gamma_\varepsilon = \gamma - \gamma_c$ and $\dot{V}_\varepsilon = \dot{V} - \dot{V}_c$ represent the errors of the flight path angle, γ_ε, and acceleration, $\dot{V}_{\varepsilon}$ from the commanded values γ_c and $\dot{V}_c$ respectively. Implementation of this system into SIMULINK format is shown in Figure 4.1.

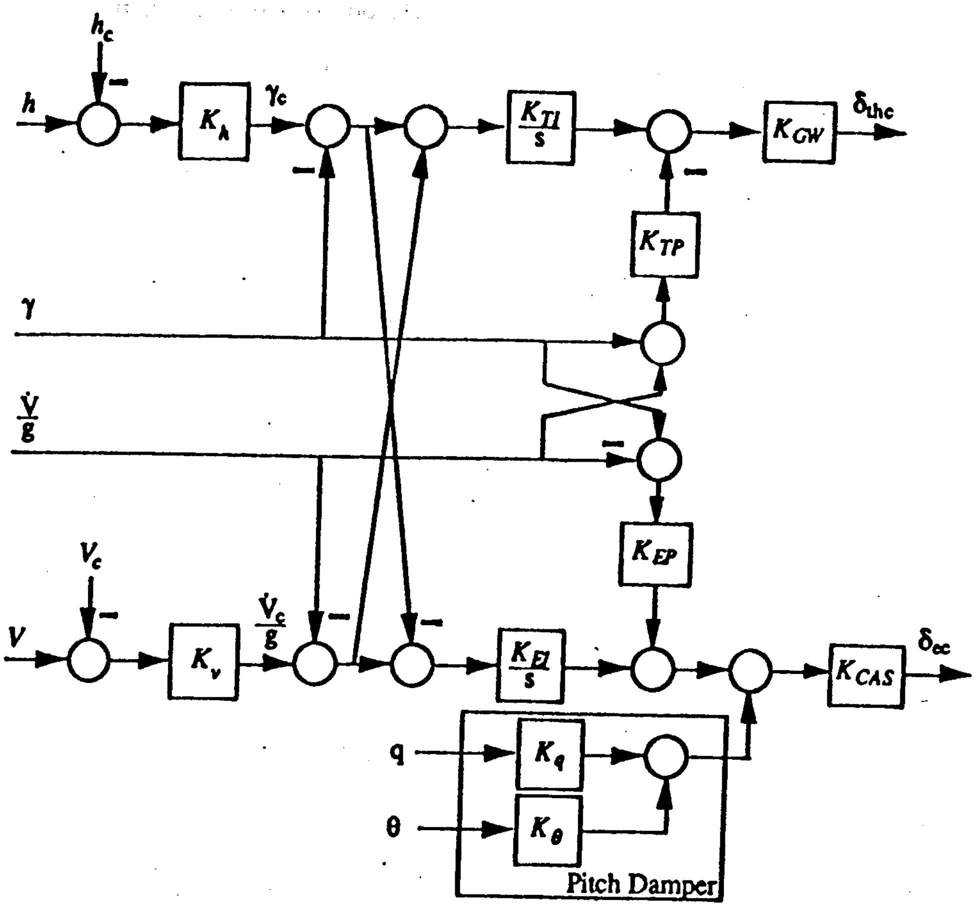


Figure 4.1 TECS Controller Structure for the Longitudinal Axis

For the above block diagram, the parameters K_{TP}, K_{TI}, K_{EP}, and K_{EI} are the proportional and integral feedback gains on the throttle and the elevator respectively. The gains K_q and K_θ are likened to a stability augmentation system (SAS) that improves the short period response. The gains K_v, and K_h provide feedback correction to the airspeed and altitude errors. Lastly, the gains K_{CAS} and K_{GW} are scheduled accordingly to compensate for the calibrated airspeed and the fluctuations in gross weight respectively. This completes a cursory explanation of the TECS controller structure for the longitudinal axis. This report is concerned with the lateral dynamics of the McDonnell Douglas F-15, therefore the next section will detail the TECS controller structure for the lateral axis.

4.3 Lateral TECS Controller

The total energy conservation principle is not as clearly seen in the lateral axis as it is in the longitudinal axis. For the lateral axis the energy balance concerns a coordinated turn. As an aircraft starts to turn, the natural tendency is for the nose to yaw against the direction of the turn, causing a slip through the air. This is not optimum performance of an aircraft. In order to perform the turn in a coordinated manner, the pilot needs to add rudder in order to counter the adverse yaw due to aileron deflection and the induced drag causing the yaw. The lateral TECS controller has as its foundation an energy balance between rolling motion (rolling energy) and yawing motion (yawing energy) in order to perform a coordinated turn. The side force equation and the requirement for a turn-coordination is depicted in equation (4.7).

$$\dot{\beta}(t) = -r(t) + \frac{g}{V_T}\phi(t) + \frac{\Sigma Fy}{mV_T} \tag{4.7}$$

For a coordinated turn in steady state, the aircraft bank angle and the yaw rate must have a certain relationship. This relationship is shown in Equation 4.8.

$$\phi_{ss} = \frac{V_T}{g} r_{ss} \tag{4.8}$$

Breaking the maneuver into small incremental steps is very helpful. The change in bank angle is then given by

$$\Delta\phi = \frac{V_T}{g}(\dot{\beta} + \Delta r) \tag{4.9}$$

Motion in the longitudinal axis movement is mainly governed by the throttle and the elevator. The lateral axis dynamics are controlled by the aileron and the rudder. Therefore, it only makes sense to have an aileron command that is a function of $\Delta\varphi$, and have the rudder be a function of $(\dot{\beta} - \Delta r)$. When this is done, the rudder functions effectively as a weathervane, so that at a constant bank angle, the difference $(\dot{\beta} - \Delta r)$ is equalled to zero.

Proportional and integral control are used for consistency in order to develop adequate control responses for the TECS controller. Applying proportional and integral control on the quantities $(\dot{\beta} - \Delta r)$ and $(\dot{\beta} + \Delta r)$, we develop the aileron and rudder command in equations 4.10 and 4.11.

$$\delta_{ac} = \frac{V_T}{g} K_a (K_{RP} + \frac{K_{RI}}{s}) (\dot{\beta}_{\varepsilon} + \dot{\psi}_{\varepsilon}) \qquad (4.10)$$

$$\delta_{rc} = Kr (K_{YP} + \frac{K_{YI}}{s}) (\dot{\beta}_{\varepsilon} - \dot{\psi}_{\varepsilon}) \qquad (4.11)$$

Implementation of the previous equations into SIMULINK is found in Figure 4.2.

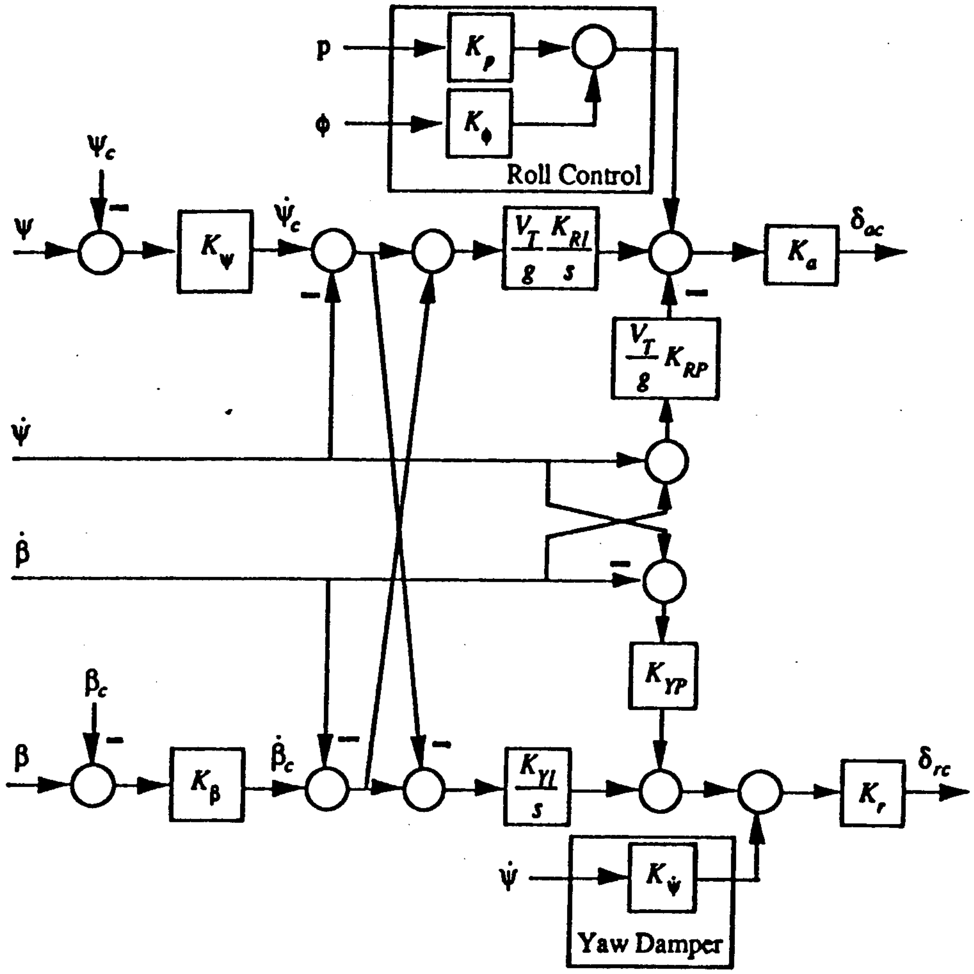


Figure 4.2 TECS Controller Structure for the Lateral Axis

The gains in lateral TECS controller are directly comparable to the longitudinal gains described in the previous section. Specifically the parameters K_{RP}, K_{YP}, K_{RI} and $K_{\dot{Y}I}$ are the proportional and integral gains on the aileron and rudder surfaces. The gains K_p and K_{φ} are used for the roll control system while $K_{\dot{\psi}}$ is incorporated in a yaw damper to better the Dutch-roll characteristics. The parameters K_{β} and K_{ψ} are proportional gains on the sideslip and heading errors, and finally K_a and K_r are used for gain scheduling for the appropriate flight dynamic pressures.

The above gains will be determined by a design method based on parameter optimization, namely the SANDY program. This optimization process requires a cost function that takes into account the command tracking problem as well as disturbance attenuation and good closed-loop stability. One such cost function is defined as,

$$J = \lim_{tf \to \infty} \int E\left[Q_1 (\beta(t) - \beta_c(t))^2 + Q_2 (\psi(t) - \psi_c(t))^2\right] \qquad (4.12)$$

where $\beta_c(t) = \beta_{cmd}(1-e^{-at})$ and $\psi_c(t)=\psi_{cmd}(1-e^{-at})$. The parameters Q_1 and Q_2 are selected to apply the proper weights to create trade-offs on the tracking variables. Another added constraint to this cost function is to establish guidelines for the optimization process. Specifically a certain eigen value and damping ratio were chosen as follows,

$$Real(\lambda_i) \le \sigma_{min} = -0.1 \qquad (4.13)$$

$$\zeta_i \ge \zeta_{min} = 0.7 \qquad (4.14)$$

The m-file **zzlatdes.m** was used to develop the necessary data and put it in the proper format for the program SANDY. A detailed discussion of this process is located in the next chapter.

Chapter 5 TECS Linear and Nonlinear Evaluations

The focus of this chapter is to first discuss the design process for the TECS control law starting with the linear model, then report the results on the linear model. Next, the gains acquired for the linear model will be tested on the nonlinear simulation. After testing the gains I will provide an evaluation of the overall design process as well as the results.

5.1 Linear Closed-Loop Model Design and Evaluation

The open-loop evaluation was covered in Chapter 3, therefore this section will only concern itself with the closed-loop system.

5.1.1 Initial Model Synthesis

Several tools were available in order to create an optimum solution to the closed-loop linear model with a TECS control law for the lateral dynamics of the McDonnell Douglas F-15. The major question was whether or not I would use the tools correctly. The primary tool was a numerical optimization program, SANDY. This program with a given set of inputs would numerically calculate the optimum gains for the control law. Before the gains could be calculated the system had to be properly input into the SANDY program. It only stands to reason that SANDY like most computer software only works as well as you set it up to work. The old adage “garbage in garbage out” is very true to the working of SANDY. In previous course applications, I was given a file, **latdes.m**, that would send SANDY the proper information in the proper format for the lateral dynamics of a TECS control law. The only difference was that this file dealt in degrees and per g units and it was designed for the Boeing 767 and not a F-15. I used **latdes.m** as a template in order to create **zzlatdes.m** which put linear model of the F-15 in the correct units and in the proper order for SANDY to use. The output of this file was **zzlatTECS1.dat** and **zzlatTECS2.dat** corresponding to flight point 1 and flight point 2 respectively. These MATLAB files can be found in Appendix C.

5.1.2 SANDY Design

The two above files were implemented into the SANDY program. Initially, SANDY was set only to run for a short time. Like most numerical optimization programs, this can be a lengthy process. By limiting the number of iterations, problems could be found and corrected before they compounded into an unfavorable solution. It was a good thing the number was limited because several unit miscalculations were found and fixed before too much computation time accrued. Once all the corrections were made, the gains for the second flight point came very quickly with acceptable results. Design of flight point 1 was a little more difficult to achieve. Specifically, the gain and phase margins were unacceptable for the aileron command loop. The gain margin was -8.894 dB and the phase margin was -29.21 degrees. The design created for the linear system will have to be implemented on a nonlinear system, thus robustness of the controller needed improvement. As a guide I used ±6 dB and ±45° for the gain and phase margins respectively.

In order to correct this shortcoming, the **zzlatTECS1.dat** file was modified. The aileron was using too much control, thus to improve the robustness I limited the aileron authority by placing an upper limit on its use. I needed to lower the upper limit several times in order to get acceptable results. These results are located in Table 5.1 which shows the worst-case eigenvalues and Table 5.2 which has the gain and phase margins for both controllers at both flight points. The gains calculated by SANDY are found in Appendix D.

Table 5.1: Closed-Loop Stability Characteristics

Flight Point	Eigenvalues	Damping	Frequency
1	-0.1 $-4.49 \pm 4.54i$	1.0 0.7	0.1 6.357
2	-0.1 $-0.326 \pm 0.333i$	1.0 0.7	0.1 0.466

Table 5.2: Single-Loop Stability Margins

Flight Point	δ_a G.M.	δ_a P.M.	δ_r G.M.	δ_r P.M.
1	-37.13	-47.39	∞	-65.14
2	∞	-82.27	∞	-70.72

These results were calculated by a file titled **sandybl.m**. This file would take the output from the SANDY program and evaluate each command loop of the closed-loop system by breaking the respective control loop. The Bode plots of the closed-loop system for each flight point and each control surface are in Figure 5.1 though Figure 5.4.

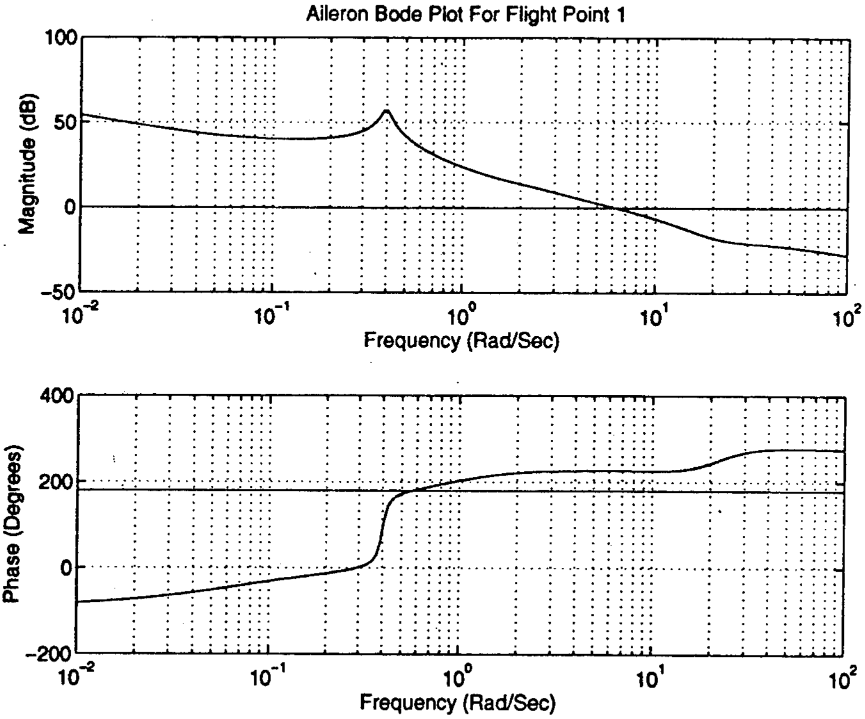


Figure 5.1 Flight Point 1 Aileron Bode Plot

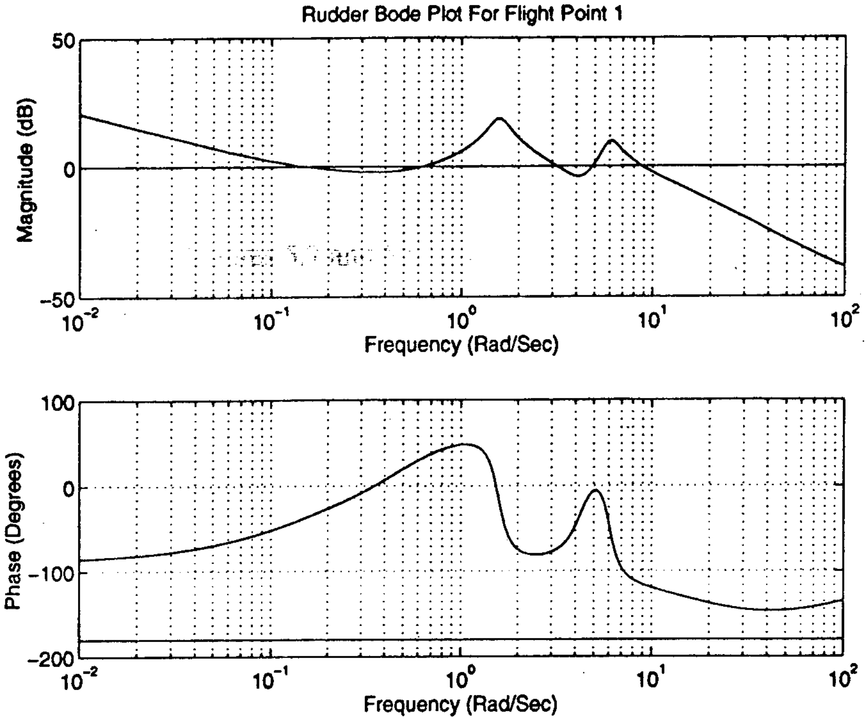


Figure 5.2 Flight Point 1 Rudder Bode Plot

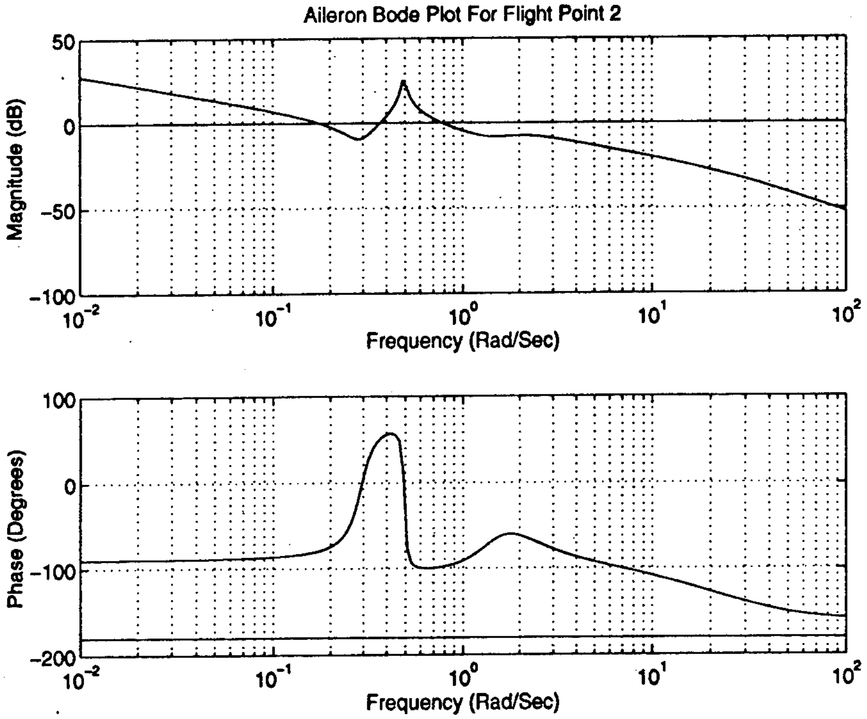
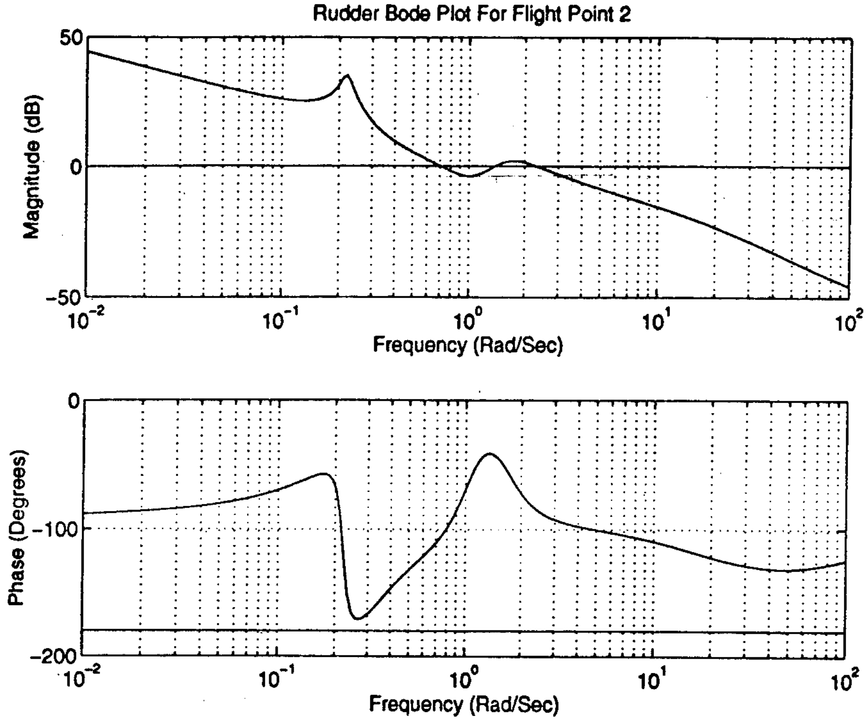


Figure 5.3 Flight Point 2 Aileron Bode Plot



5.1.3 Linear Model Closed-Loop Command Responses

The closed-loop command responses were determined through the use of the *m-file* **linan.m** which takes the aircraft's closed-loop matrices and inputs them into a linear simulation function of MATLAB. The response for the sideslip β, heading ψ, aileron δ_a, and rudder δ_r are discussed below. In all figures describing the time responses the thicker line is the input command and the thin line is the MATLAB linear solution.

Heading Command

The linearized aircraft responses to a 30-degree heading command are shown for both flight conditions in Figures 5.5 and 5.6 respectively. The heading tracking is very good in both cases showing no overshoot and very few deviations from the command. In addition, the amount of rudder and aileron deflections to acquire these results are well within component limits as well as within common sense parameters.

Sideslip Command

The aircraft was subjected to a 5-degree constant sideslip command in order to evaluate the sideslip performance. Flight point 2 tracked the sideslip command very well with no overshoots. In addition, the control deflections are moderate without large fluctuations that would add vibrations to the aircraft. For flight point 1, the aircraft did not track closely the command. However, the time response was reasonable enough to continue the design process. The control deflections for this command are also within tolerances. Figures 5.7 and 5.8 show the sideslip command responses for flight points 1 and 2 respectively.

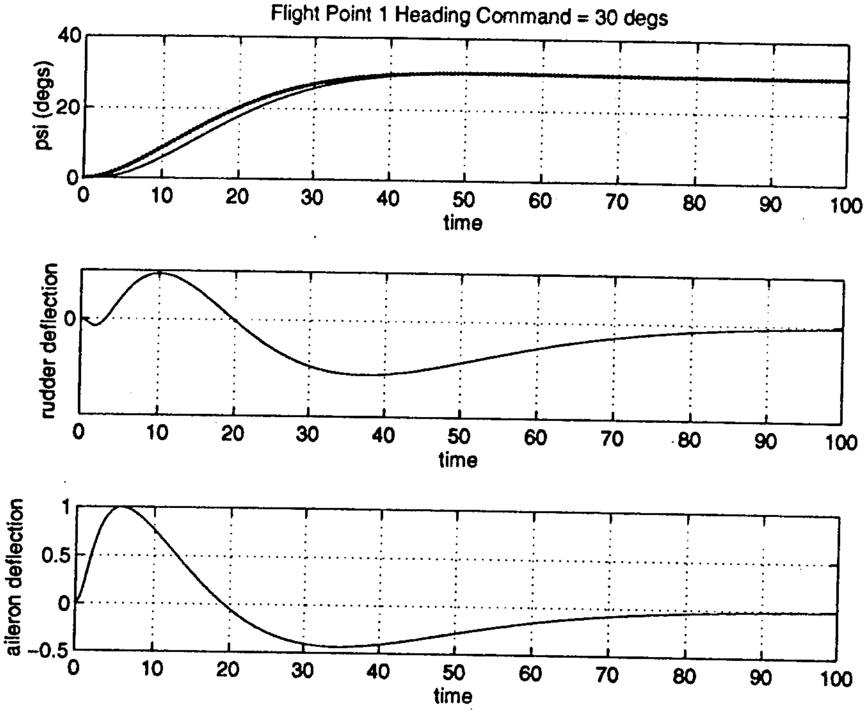


Figure 5.5 Flight Point 1 Linear Aircraft Response to a 30°-Heading Command

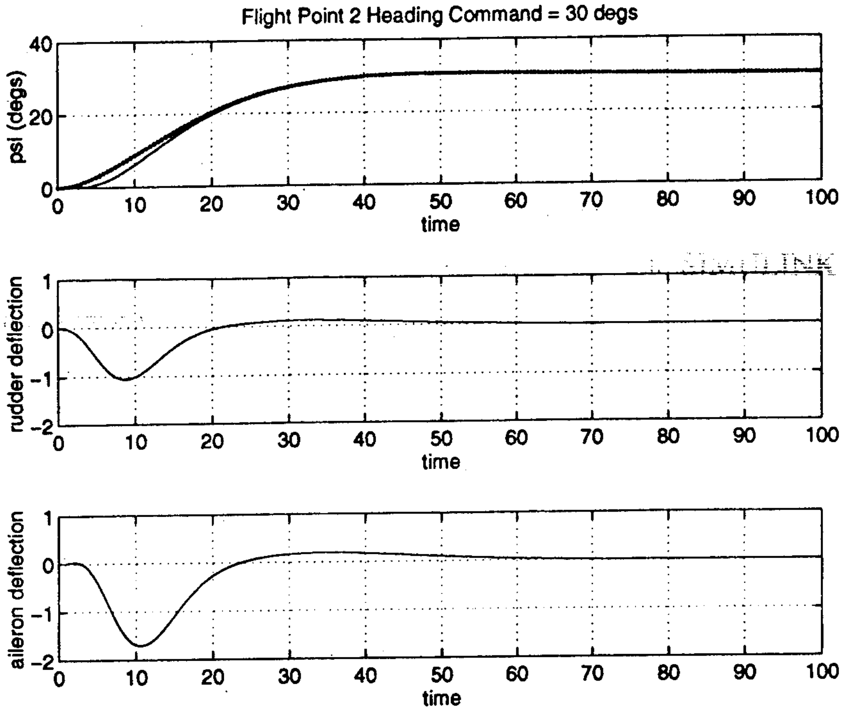


Figure 5.6 Flight Point 2 Linear Aircraft Response to a 30º-Heading Command

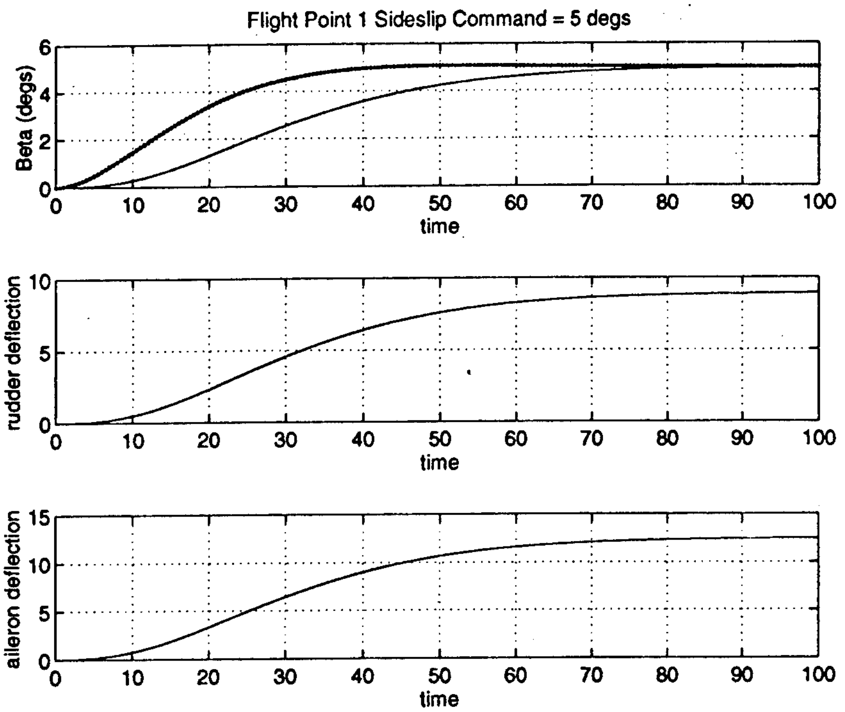


Figure 5.7 Flight Point 1 Linear Aircraft Response to a 5º-Sideslip Command

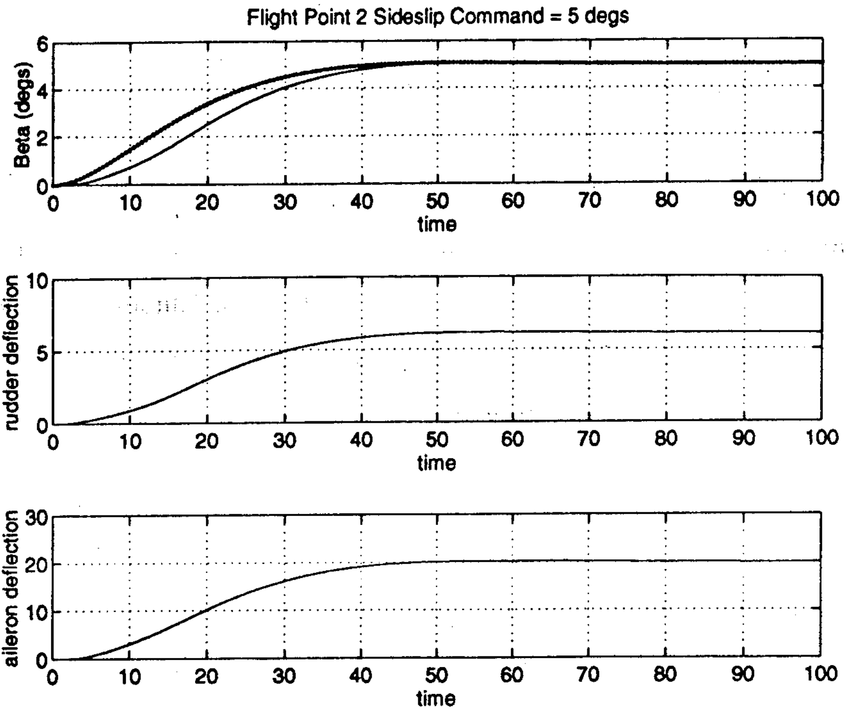


Figure 5.8 Flight Point 2 Linear Aircraft Response to a 5^{o}-Sideslip Command

5.1.4 Evaluating the Linear SIMULINK Model

Up until this point the linear simulation of the F-15 was done in the MATLAB environment. The nonlinear simulation is set up in the SIMULINK environment. Previous assignments have proven that what is in the MATLAB environment will not always directly transfer into the SIMULINK environment correctly. Therefore, the linear SIMULINK environment must be tested and the results verified with those attain from the MATLAB functions.

The first step of the process was to extract the gains acquired from the SANDY program. The programs **zzlatTECS1c.m** and **zzlatTECS2c.m** were both modified with the gains explicitly stated so the SIMULINK program would read them correctly. The next modification came from a units conversion. The states in the aircraft model are all in radians. SANDY solved for the states in degrees, and the inputs into the aircraft model are also in degrees. In order to keep the units consistent, the outputs of the state-space model needed to be converted to degrees. This was done in the **trim497ss.m** and **trim539ss.m** files before they were entered into the SIMULINK representation. These two MATLAB files can be found in Appendix B.

The SIMULINK model consists of both the lateral and the longitudinal models, therefore the gains gotten by Lt. Dutton needed to be added before the simulation could be run. Lt. Dutton saved his gains under **f25sd539_cx.mat** and **f25sd497_cx.-mat** for flight points 1 and 2 respectively [10]. The complete SIMULINK model is located in Figure 5.9.

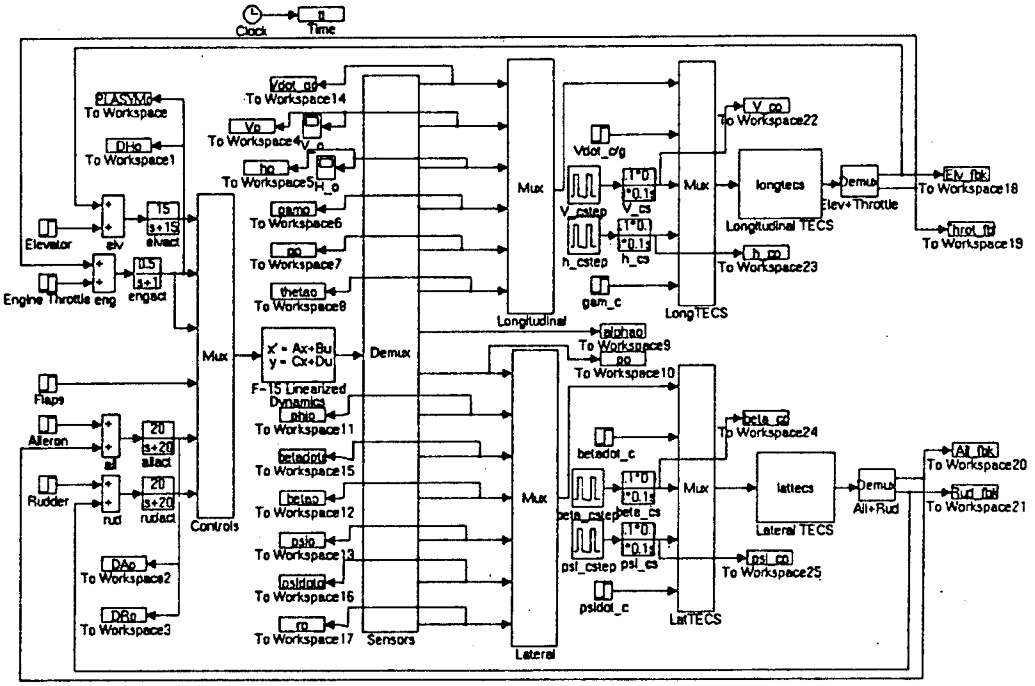


Figure 5.9 Linear SIMULINK Model Representation

Due to the fact that this was a linear model, the numerical integration method used to propagate the simulation was **linsim** which is found within the simulation parameters menu. A few bugs were discover during this process and they were quickly corrected. In the end, the results of the linear simulation provided by MATALB were consistent with the simulation provided by the linear simulation done in the SIMULINK environment The results of this comparison are found in Figures 5.10-5.13. The heavier line is the combination of the MATLAB and SIMULINK solutions while the thin line is the original command input that the simulations are trying to track.

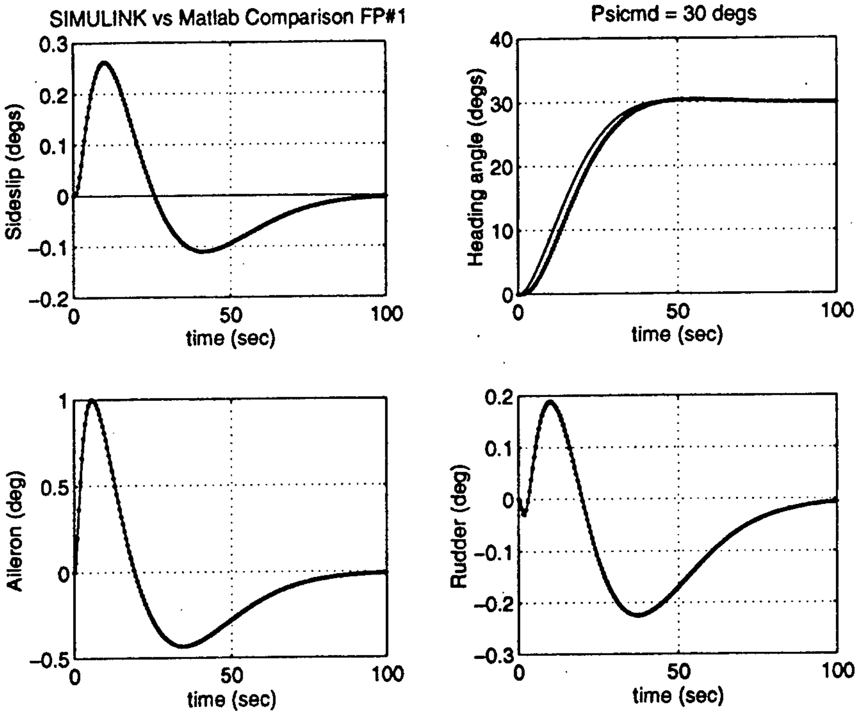


Figure 5.10 Flight Point 1 Linear Comparison Heading Command 30^{o}.

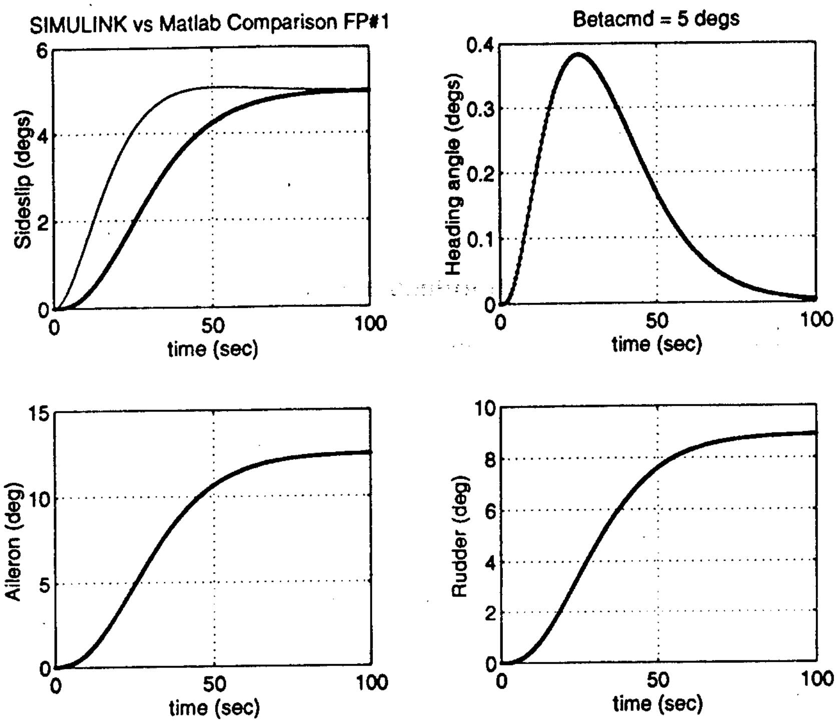


Figure 5.11 Flight Point 1 Linear Comparison Sideslip Command 5^{o}.

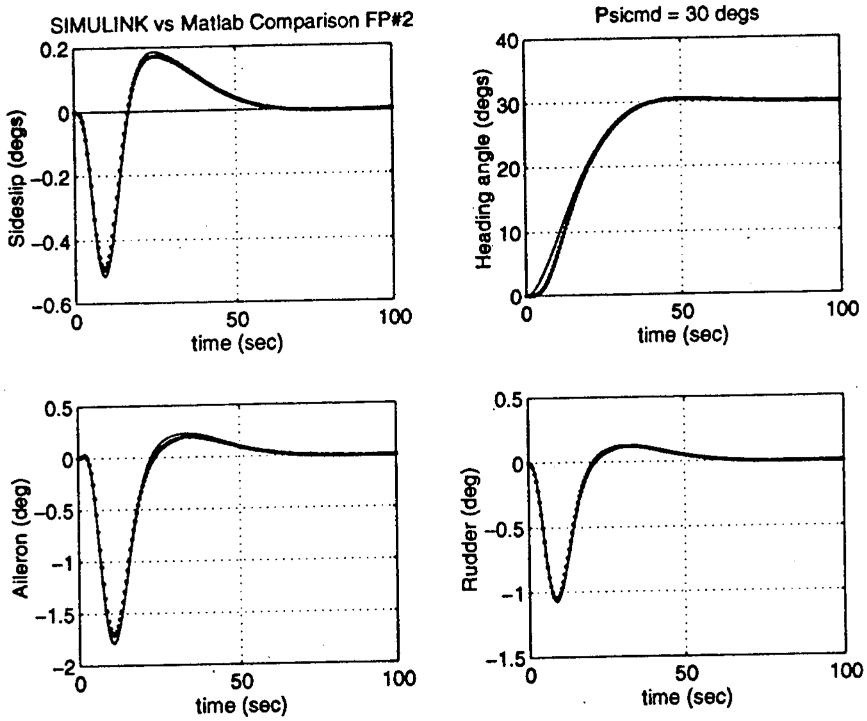


Figure 5.12 Flight Point 2 Linear Comparison Heading Command 30^{o}.

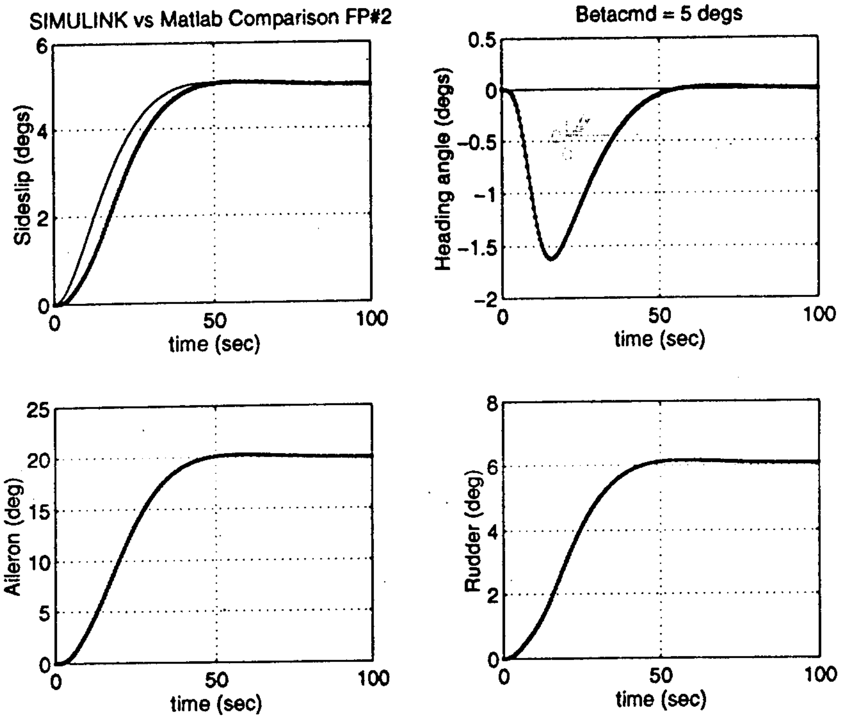


Figure 5.13 Flight Point 2 Linear Comparison Sideslip Command 5^{O}.

5.2 Nonlinear Closed-Loop Model Evaluations

In order to do the nonlinear closed-loop evaluations, the gains acquired from the linear closed-loop calculations were inserted into the nonlinear simulation shown in Chapter 2 of this report. The command responses were repeated for the nonlinear simulation with favorable results.

5.2.1 Nonlinear Closed-Loop Command Responses

The closed-loop time responses were calculated through the use of the SIMULINK model. The SIMULINK model has command inputs to be defined by the user. The previous commands were a 5-degree sideslip command and a 30-degree heading change, each done separately. These commands were repeated for the nonlinear model.

Heading Command

The nonlinear simulation was tasked to track a 30-degree heading change for both flight points. For flight points 1, the nonlinear and linear cases produced nearly identical results. The nonlinear solution tracked the heading command very accurately with no overshoots. In addition, the control deflections were well within tolerable limits. One difference between the linear and nonlinear control deflections was there was a single fluctuation in control surfaces found in the nonlinear simulation that was not present in the linear case. This only makes sense that noise in the control surfaces would represent itself in the nonlinear case and not the linear case. Flight point 2 also tracked the heading command very well. As with the first flight point, flight point 2 had no overshoots and quickly matched the input command. The control deflections for this flight point had more fluctuations when compared to the linear case. However, the deflections consistently matched the linear case deflections. Results of the simulations were graphed simultaneously in Figures 5.14-5.17 in order to show the differences described above graphically. In general, the thin line represents the linear model whereas the thicker line represents the nonlinear model. The exception is on the first graph of each set, where the input command is also plotted. In these graphs the dashed line is the linear case, the thick line is the nonlinear case, and the thin line is the command input.

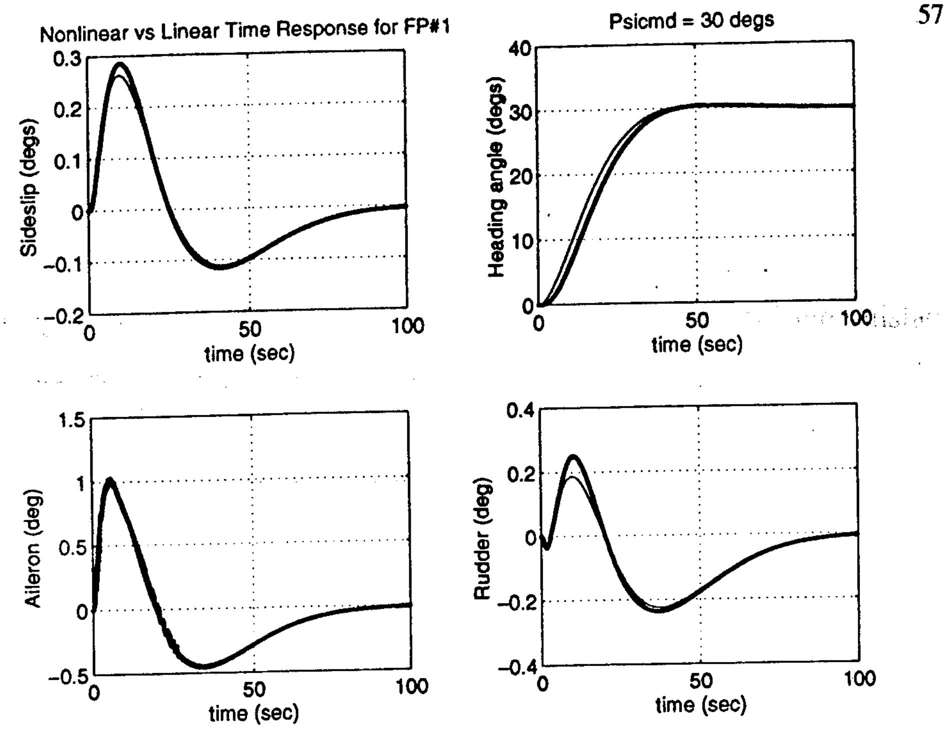


Figure 5.14 Flight Point 1 Nonlinear vs. Linear Time Responses

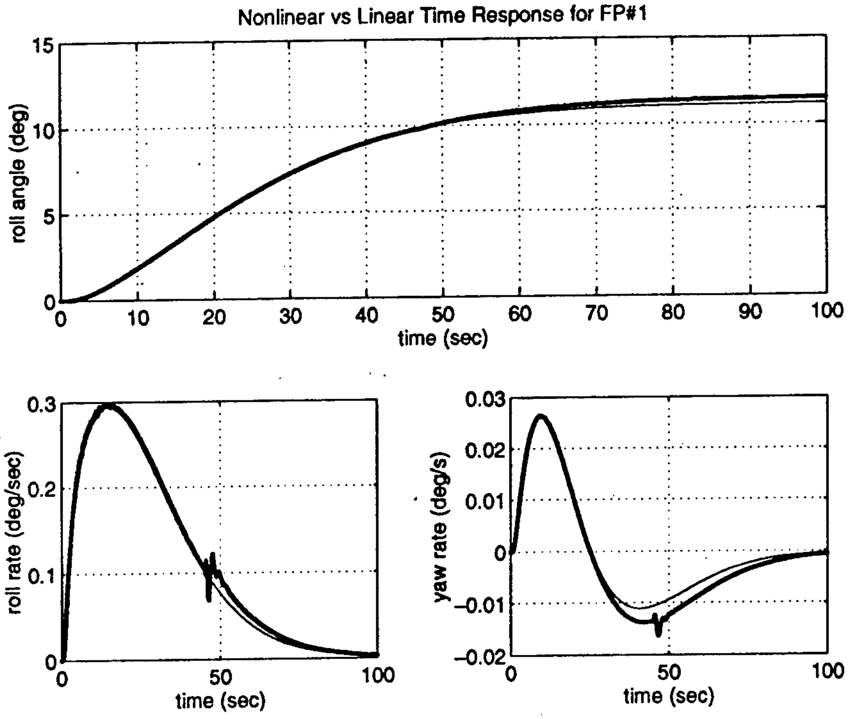


Figure 5.15 Flight Point 1 Nonlinear vs. Linear Time Responses

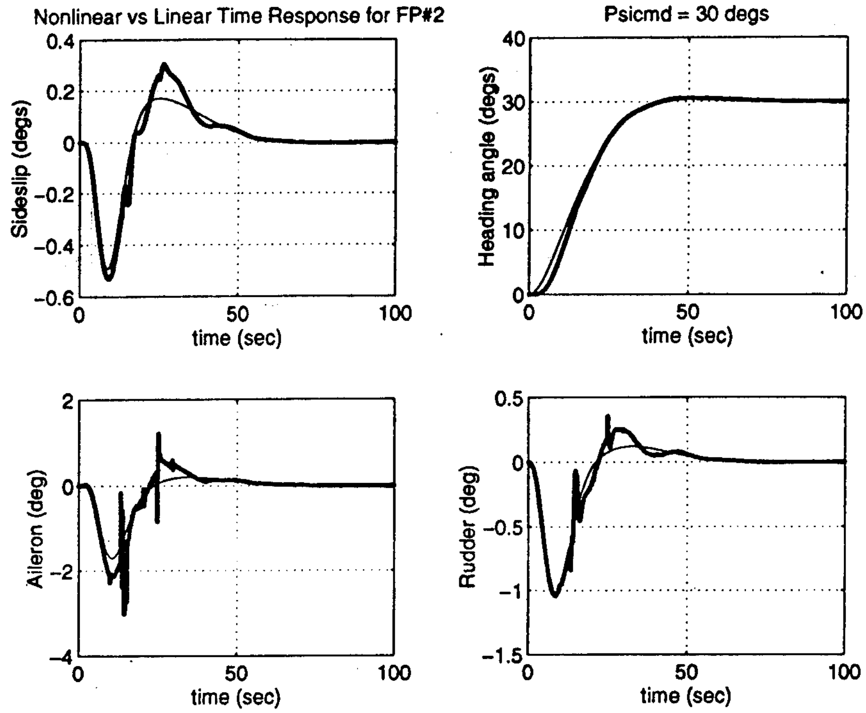


Figure 5.16 Flight Point 2 Nonlinear vs. Linear Time Responses

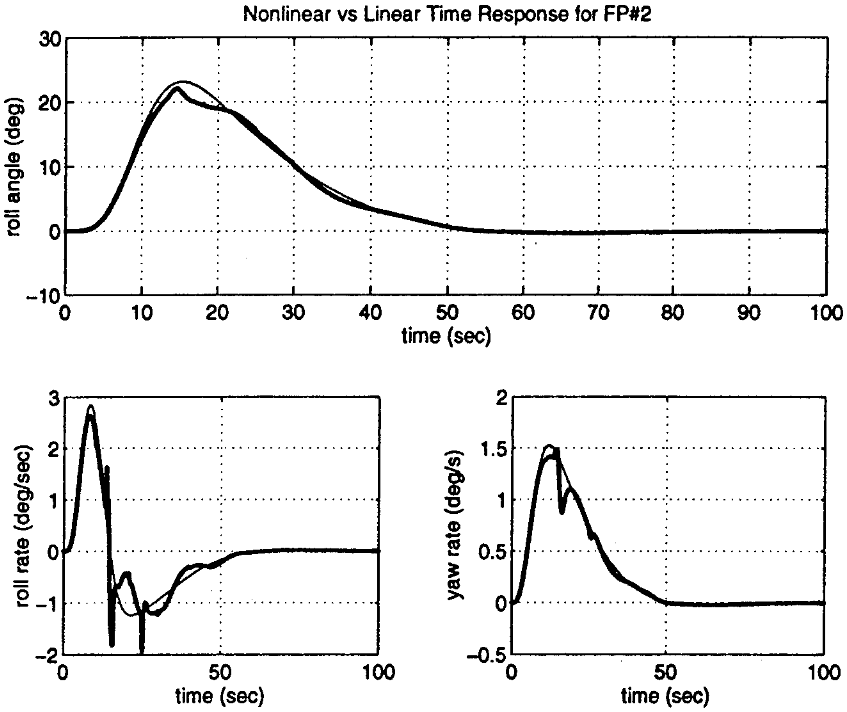


Figure 5.17 Flight Point 2 Nonlinear vs. Linear Time Responses

Sideslip Command

The aircraft was commanded to a 5-degree sideslip command for both flight points. The nonlinear simulation for Flight point 1 was a little slow in tracking the command. This could have been expected due to the fact that the linear model was also a little slow tracking the sideslip command. The results however were considered satisfactory due to the fact that the simulation had no overshoots and the initial error between the command and the aircraft response was not large. As for the control surface deflections, both the linear and the nonlinear curves were smooth and not excessive. As a matter of fact the nonlinear and the linear control deflections almost overlap completely.

The nonlinear simulation for flight point 2 was very successful in tracking the sideslip command. It quickly converged to the command without overshoots. The control deflection were smooth and within tolerances. The deflections were almost on top of one another for the linear and nonlinear case. Overall the nonlinear case was successful for the sideslip command.

A graphical description of the sideslip command time responses are found in Figures 5.18-5.21. In these graphs the nonlinear solution is always the thickest line. The linear solution is the thin line in all cases except where the input command is also plotted. In this case the thin line is the command input and the dashed line is the linear solution.

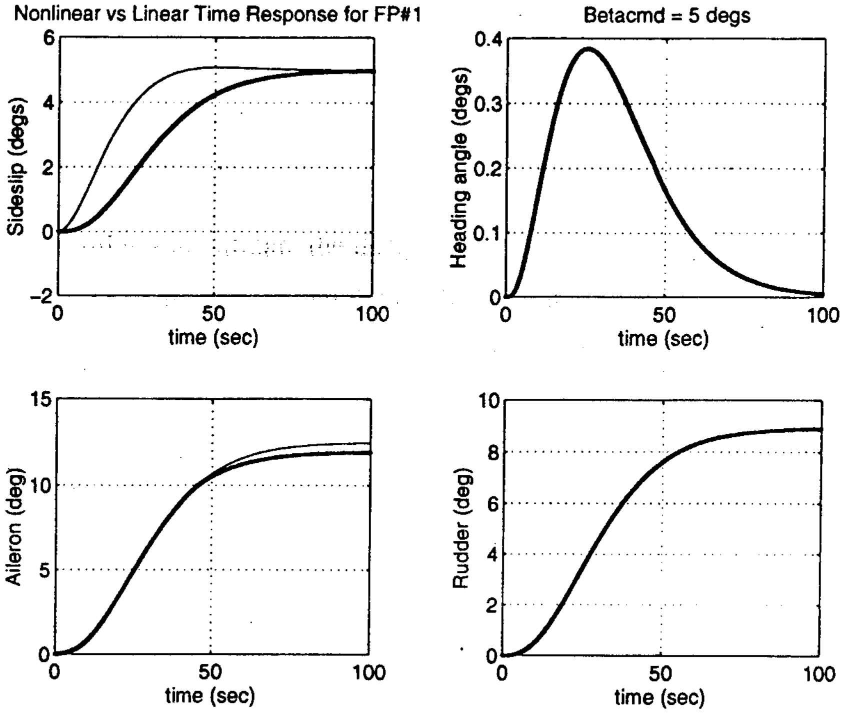


Figure 5.18 Flight Point 1 Nonlinear vs. Linear Time Responses

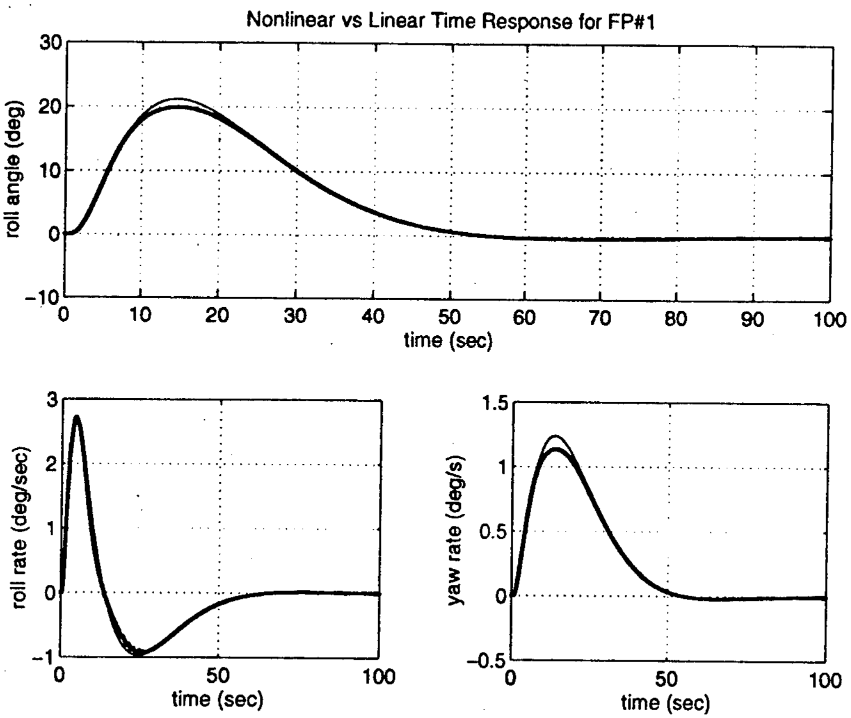


Figure 5.19 Flight Point 1 Nonlinear vs. Linear Time Responses

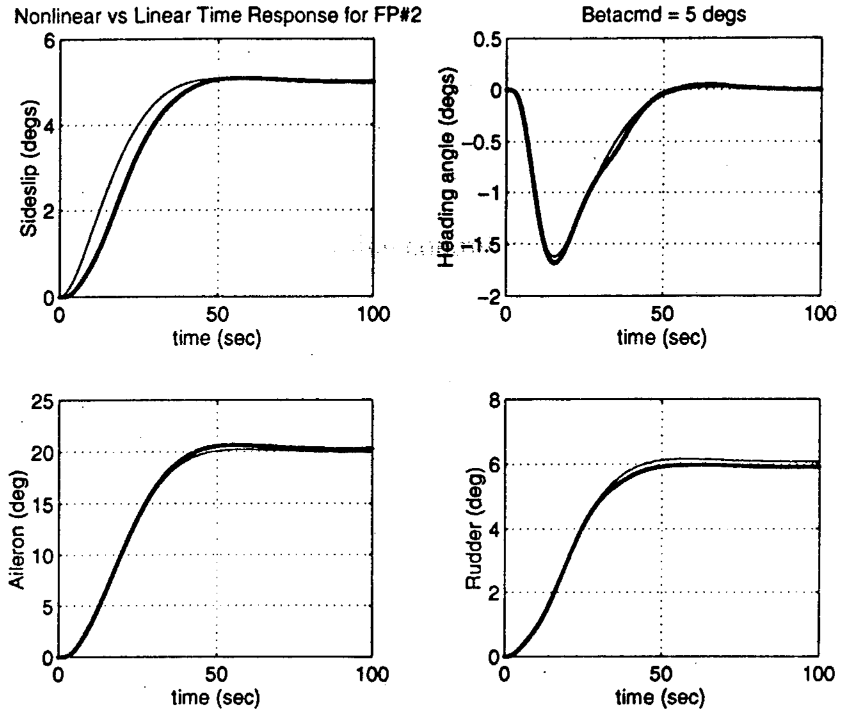


Figure 5.20 Flight Point 2 Nonlinear vs. Linear Time Responses

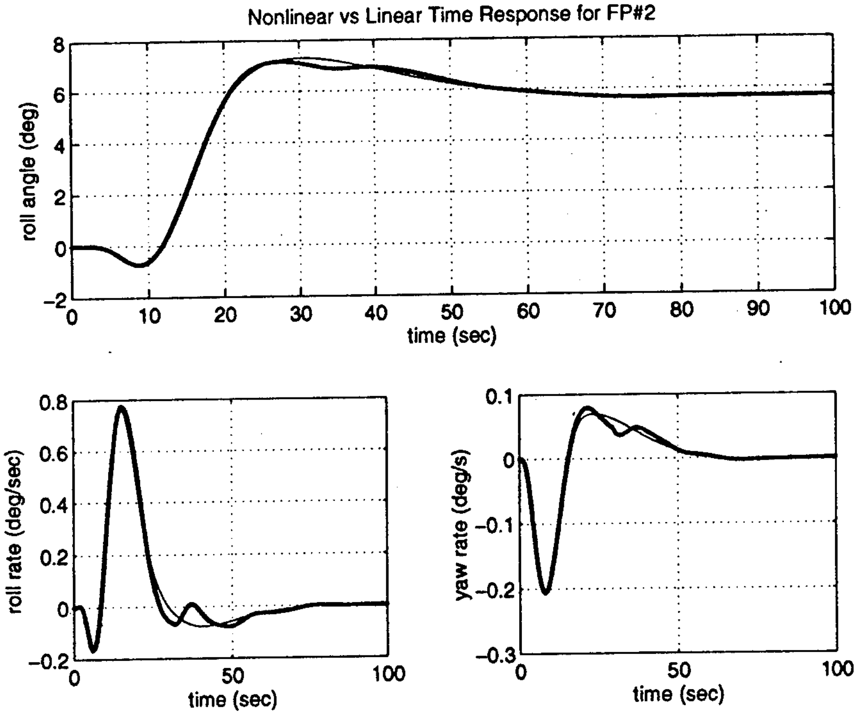


Figure 5.21 Flight Point 2 Nonlinear vs. Linear Time Responses

Chapter 6 Conclusions

6.1 Summary

The objective of this report was to develop a total energy control system (TECS) for the lateral dynamics and validate the design in a nonlinear simulation of the McDonnell Douglas F-15. This was accomplished by first analyzing the open-loop characteristics of both the linear and nonlinear simulations in order to validate the linear system. Once the linear system was proven to accurately describe the aircraft's nonlinear dynamics, the TECS controller was design for the linear system through the use of the SANDY program. SANDY is a numerical optimization program that came up with the optimum gains to be used to solve the control problem. These gains were effective in controlling the linear model both in performance and robustness. Next these gains were tested in the SIMULINK model of the linear model to make sure there was no extra-problems shifting from the MATLAB environment to the SIMULINK environment. This was accomplished locating several bugs in the system before the application to the nonlinear model was accomplished. The final step was to test the gains on the nonlinear simulation. The gains were as effective in tracking the heading and sideslip commands as the linear system. The end result is an effective Total Energy Control System for the lateral axis of the McDonnell Douglas F-15 fighter aircraft.

6.2 Recommendations for Future Study

There are several areas for future study. The following is a list where study can be furthered.

1) Lt. Dutton did the longitudinal axis and I did the lateral axis at two specific flight points [10]. Better understanding and learning could be accomplished by expanding the flight envelope for more than two flight points and then connecting the total design with a gain schedule.

2) This thesis dealt with the McDonnell Douglas F-15, another future study would be to apply the nonlinear model framework to another aircraft using its specific nonlinear aerodynamics and propulsion characteristics.

3) Control surface vibrations were discovered when applying the TECS control-law to the nonlinear simulation. Finding a way to eliminate these vibration is worth investigation.

These are just a few of the areas that can be investigated. I hope this thesis provokes thought into other aspects concerning control theory. If it does then this paper was a success not only in determining an effective control-law for the F-15, but also it has kept dreams of advanced flight alive.

Bibliography

[1] Anderson, John D. Jr., Introduction to Flight. McGraw-Hill Book Company, 3rd Edition, 1989.

[2] Bruce, K.R., *Integrated Autopilot/Autothrottle Based on a Total Energy Control Concept: Design and Evaluation of Additional Autopilot modes*. Technical Report NASA TCV Contract NAS1-16300, Boeing Commercial Airplane Company, August 1987.

[3] Bruce, K.R., *Integrated Autopilot/Autothrottle for the NASA TSRV B-737 Aircraft: Design and Verification by Nonlinear Simulation*. Technical Report NASA CR 4217, NASA Langley Research Center, February 1989.

[4] Bruce, K.R., J.R. Kelly, and L.H. Person, *NASA B737 Flight Test Results of the Total Energy Control System*. Technical Report AIAA 86-2143-CP, AIAA Guidance, Navigation and Control Conference, August 1986.

[5] Brumbaugh, Randal W., "An Aircraft Model for the AIAA Controls Design Challenge," PRC Inc., Edwards, CA.

[6] Duke, E.L., Antoniewicz, R.F., and Krambeer, K.D., *Derivation and Definition of a Linear Aircraft Model*, NASA RP-1207, August 1988.

[7] Lambregts, A.A., *Integrated System Design for FLight and Propulsion Control using Total Energy Principles*. Technical Report AIAA 83-2561, AIAA, October 1983.

[8] Lambregts, A.A., *Operational Aspects of the Integrated Vertical Flight Path and Speed Control System*. Technical Report SAE 831420, Aerospace Congress & Exposition, Long Beach, California, August 1983.

[9] Swamy, Sanjay, "Robust Integrated Design Using Constrained Parameter Optimization,' University of Washington, Department of Aeronautics and Astronautics, 1992.

[10] Dutton, James P. Jr., "Development of a Nonlinear Simulation for the McDonnell Douglas F-15 Eagle with a Longitudinal TECS Control-Law," University of Washington, Department of Aeronautics and Astronautics, 1994.

Appendix A F-15 Nonlinear Simulation S-Functions

A.1 S-Funtion for Open Loop F-15 Model

The following is a listing for the MATLAB code **f25sfn.m** which is used for the open-loop nonlinear F-15 simulation.

```
%%% NONLINEAR F-15 SIMULATION S-FUNCTION %%%
function [sys,x0] = f25sfn(t,x,u,flag)
global A IA
global V alpha q theta p phi r psi beta xp yp h
global DH DD DA DR PLAPL PLAPR PLASYM alpdot
global xx utrim

 %%% UPDATE A,IA-ARRAYS %%%
A(829) = V; A(914) = alpha; A(862) = q; A(713) = h;
 A(943) = theta; A(861) = p; A(942) = phi; A(715) = xp;
 A(863) = r; A(944) = psi; A(915) = beta; A(716) = yp;
A(1402) = DH; A(1403) = DD; A(1401) = DA;
 A(1404) = DR; A(1416) = PLAPL; A(1417) = PLAPR;
 A(1418) = PLASYM;

%%% STANDARD ATMOSPHERE %%%
 [AMCH,RHO,QBAR,G] = atmos(A);

%%% A-ARRAY UPDATE %%%
A(825) = AMCH; A(670) = RHO; A(669) = QBAR; A(772) = G;

 %%% STABILITY AXIS FORCES AND MOMENTS %%%
[CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = f25aero(A,IA);

%% Transfer to the array A (may not need to)
```

```
L = QBAR*S*CLFT;
D = QBAR*S*CD;
Y = QBAR*S*CY;
SumL = ALM;
SumM = AMM;
SumN = ANM;
XT = FPX;
YT = FPY;
ZT = FPZ;

%%% ANGLE CALCULATIONS %%%
COSTHETA = cos(theta);
SINTHETA = sin(theta);
TANTHETA = tan(theta);
SECTHETA = 1/cos(theta);
COSBETA = cos(beta);
SINBETA = sin(beta);
TANBETA = tan(beta);
COSALPHA = cos(alpha);
SINALPHA = sin(alpha);
COSPHI = cos(phi);
SINPHI = sin(phi);
COSPSI = cos(psi);
SINPSI = sin(psi);

%%% EQUATIONS OF MOTION %%%
V1 = -D*COSBETA+Y*SINBETA+XT*COSALPHA*COSBETA;
V2 = YT*SINBETA+ZT*SINALPHA*COSBETA;
V3 = -m*g*(COSALPHA*COSBETA*SINTHETA);
V4 = -m*g*(-SINBETA*SINPHI*COSTHETA);
```

```
V5 = -m*g*(-SINALPHA*COSBETA*COSPHI*COSTHETA);

al1 = -L+ZT*COSALPHA-XT*SINALPHA;
al2 = m*g*(COSALPHA*COSPHI*COSTHETA+SINALPHA*SINTHETA);
al3 = q-TANBETA*(p*COSALPHA+r*SINALPHA);

q1 = SumL*I2+SumM*I4+SumN*I5-p^2*(Ixz*I4-Ixy*I5);
q2 = p*q*(Ixz*I2-Iyz*I4-Dz*I5)-p*r*(Ixy*I2+Dy*I4-Iyz*I5);
q3 = q^2*(Iyz*I2-Ixy*I5)-q*r*(Dx*I2-Ixy*I4+Ixz*I5);
q4 = -r^2*(Iyz*I2-Ixz*I4);

p1 = SumL*I1+SumM*I2+SumN*I3-p^2*(Ixz*I2-Ixy*I3);
p2 = p*q*(Ixz*I1-Iyz*I2-Dz*I3)-p*r*(Ixy*I1+Dy*I2-Iyz*I3);
p3 = q^2*(Iyz*I1-Ixy*I3)-q*r*(Dx*I1-Ixy*I2+Ixz*I3);
p4 = -r^2*(Iyz*I1-Ixz*I2);

r1 = SumL*I3+SumM*I5+SumN*I6-p^2*(Ixz*I5-Ixy*I6);
r2 = p*q*(Ixz*I3-Iyz*I5-Dz*I6)-p*r*(Ixy*I3+Dy*I5-Iyz*I6);
r3 = q^2*(Iyz*I3-Ixy*I6)-q*r*(Dx*I3-Ixy*I5+Ixz*I6);
r4 = -r^2*(Iyz*I3-Ixz*I5);

be1 = D*SINBETA+Y*COSBETA-XT*COSALPHA*SINBETA;
be2 = YT*COSBETA-ZT*SINALPHA*SINBETA;
be3 = m*g*(COSALPHA*SINBETA*SINTHETA);
be4 = m*g*(COSBETA*SINPHI*COSTHETA);
be5 = m*g*(-SINALPHA*SINBETA*COSPHI*COSTHETA);
be6 = p*SINALPHA-r*COSALPHA;

xp1 = COSALPHA*COSBETA*COSTHETA*COSPSI;
xp2 = SINBETA*(SINPHI*SINTHETA*COSPSI-COSPHI*SINPSI);
```

```
    xp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*-
COSPSI);

    yp1 = COSALPHA*COSBETA*COSTHETA*SINPSI;
    yp2 = SINBETA*(COSPHI*COSPSI+SINPHI*SINTHETA*SINPSI);
     yp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*-
COSPSI);

    h1 = COSALPHA*COSBETA*SINTHETA;
    h2 = SINBETA*SINPHI*COSTHETA;
    h3 = SINALPHA*COSBETA*COSPHI*COSTHETA;

   if flag == 0

    %%% SYSTEM CHARACTERISTICS/INITIAL CONDITIONS %%%
    sys = [12 0 13 8 12 0];
    x0 = xx;

    %%% LONGITUDINAL STATES %%%
    V = x0(1); %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
    alpha = x0(2); %%% ANGLE OF ATTACK (RAD) %%%
    q = x0(3); %%% PITCH RATE (RAD/S) %%%
    theta = x0(4); %%% PITCH ANGLE (RAD) %%%

    %%% LATERAL/DIRECTIONAL STATES %%%
    p = x0(5); %%% ROLL RATE (RAD/S) %%%
    phi = x0(6); %%% ROLL ANGLE (RAD) %%%
    r = x0(7); %%% YAW RATE (RAD/S) %%%
    psi = x0(8); %%% YAW ANGLE (RAD) %%%
    beta = x0(9); %%% SIDESLIP ANGLE (RAD) %%%
```

```
%%% EARTH-RELATIVE POSITION STATES %%%
xp = x0(10); %%% X-DIRECTION POSITION (FT) %%%
yp = x0(11); %%% Y-DIRECTION POSITION (FT) %%%
h = x0(12); %%% ALTITUDE (FT) %%%

elseif abs(flag) == 1

%%% LONGITUDINAL STATES %%%
V = x(1); %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
alpha = x(2); %%% ANGLE OF ATTACK (RAD) %%%
q = x(3); %%% PITCH RATE (RAD/S) %%%
theta = x(4); %%% PITCH ANGLE (RAD) %%%

%%% LATERAL/DIRECTIONAL STATES %%%
p = x(5); %%% ROLL RATE (RAD/S) %%%
phi = x(6); %%% ROLL ANGLE (RAD) %%%
r = x(7); %%% YAW RATE (RAD/S) %%%
psi = x(8); %%% YAW ANGLE (RAD) %%%
beta = x(9); %%% SIDESLIP ANGLE (RAD) %%%

%%% EARTH-RELATIVE POSITION STATES %%%
xp = x(10); %%% X-DIRECTION POSITION (FT) %%%
yp = x(11); %%% Y-DIRECTION POSITION (FT) %%%
h = x(12); %%% ALTITUDE (FT) %%%

%%% INPUTS (U) %%%
DH = u(1)+utrim(1); %%% SYMETRIC STABILATOR (DEG) %%%
DD = u(2); %%% DIFFERENTIAL STABILATOR (DEG) %%%
DA = u(3)+utrim(5); %%% AILERON DEFLECTION (DEG) %%%
```

```
DR = u(4)+utrim(6); %%% RUDDER DEFLECTION (DEG) %%%
 PLAPL = u(5)+utrim(2); %%% LEFT PLA (DEG) %%%
 PLAPR = u(6)+utrim(3); %%% RIGHT PLA (DEG) %%%
 PLASYM = u(7); %%% SYMMETRIC PLA (DEG) %%%
 alpdot = u(8); %%% AOA RATE (RAD/S) %%%

 %%% STATE DERIVATIVES (dX/dT) %%%
 sys(1,1) = (V1+V2+V3+V4+V5)/m;
 sys(2,1) = al3+(al1+al2)/(V*m*COSBETA);
 sys(3,1) = (q1+q2+q3+q4)/detI;
 sys(4,1) = q*COSPHI-r*SINPHI;
 sys(5,1) = (p1+p2+p3+p4)/detI;
 sys(6,1) = p+q*SINPHI*TANTHETA+r*COSPHI*TANTHETA;
 sys(7,1) = (r1+r2+r3+r4)/detI;
 sys(8,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA;
 sys(9,1) = be6+(be1+be2+be3+be4+be5)/(m*V);
 sys(10,1) = V*(xp1+xp2+xp3);
 sys(11,1) = V*(yp1+yp2+yp3);
 sys(12,1) = V*(h1-h2-h3);

elseif flag == 3

 %%% SYSTEM OUTPUTS (Y) %%%
 sys(1,1) = x(1);
 sys(2,1) = x(2);
 sys(3,1) = x(3);
 sys(4,1) = x(4);
 sys(5,1) = x(5);
 sys(6,1) = x(6);
 sys(7,1) = x(7);
sys(8,1) = x(8);
sys(9,1) = x(9);
sys(10,1) = al3+(al1+al2)/(V*m*COSBETA);
sys(11,1) = x(10);
sys(12,1) = x(11);
sys(13,1) = x(12);

else

%%% ALL OTHER FLAGS UNDECLARED %%%
sys = [];

end

end
```

A.2 S-Function for Closed-Loop F-15 Model

The following is a listing of the MATLAB s-function **f25sfncl.m** whicih is used for the closed-loop nonlinear F-15 model.

```
%%% NONLINEAR F-15 SIMULATION S-FUNCTION %%%
function [sys,x0] = f25sfncl(t,x,u,flag)
global A IA
global V alpha q theta p phi r psi beta xp yp h
global DH PLAPL PLAPR FLAPS DA DR PLASYM DD
global xx utrim

%%% UPDATE A,IA-ARRAYS %%%
A(829) = V; A(914) = alpha; A(862) = q; A(713) = h;
A(943) = theta; A(861) = p; A(942) = phi; A(715) = xp;
A(863) = r; A(944) = psi; A(915) = beta; A(716) = yp;
```

```
A(1402) = DH; A(1403) = DD; A(1401) = DA;
 A(1404) = DR; A(1416) = PLAPL; A(1417) = PLAPR;
 A(1418) = PLASYM;

 %%% STANDARD ATMOSPHERE %%%
 [AMCH,RHO,QBAR,G] = atmos(A);

%%% A-ARRAY UPDATE %%%
A(825) = AMCH; A(670) = RHO; A(669) = QBAR; A(772) = G;

 %%% STABILITY AXIS FORCES AND MOMENTS %%%
[CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = f25aero(A,IA);

%% Transfer to the array A (may not need to)
A(1410) = CLFT; A(1411) = CD; A(1412) = CY;
A(1413) = CL; A(1414) = CM; A(1415) = CN;
A(748) = FAX; A(749) = FAY; A(750) = FAZ;
A(733) = ALM; A(734) = AMM; A(735) = ANM;

 %%% CALCULATE PROPULSION FORCES %%%
[FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C,FIRST] = f25eng(A,IA);

%% Transfer to the array A and IA
A(751) = FPX; A(752) = FPY; A(753) = FPZ;
A(736) = DCL; A(737) = DCM; A(738) = DCN;
A(1419) = TAUL; A(1420) = TAUR;
A(1431) = PLAL; A(1432) = PLAR;
```

```
A(667) = COUT1C; A(668) = COUT2C;
IA(502) = FIRST;

%%% A,IA-ARRAY VAR NAMES %%%
S = A(659); QBAR = A(669); g = A(772);
CLFT = A(1410); CD = A(1411); CY = A(1412);
ALM = A(733); AMM = A(734); ANM = A(735);
FPX = A(751); FPY = A(752); FPZ = A(753);
Ix = A(634); Iy = A(635); Iz = A(636);
Ixz = A(637); Ixy = A(638); Iyz = A(639);
I1 = A(632); I2 = A(631); I3 = A(630);
I4 = A(629); I5 = A(628); I6 = A(627);
Dx = A(626); Dy = A(625); Dz = A(624);
detI = A(633); m = A(658)/G;

%%% CALCULATE UPDATED AERO VALUES %%%
L = QBAR*S*CLFT;
D = QBAR*S*CD;
Y = QBAR*S*CY;
SumL = ALM;
SumM = AMM;
SumN = ANM;
XT = FPX;
YT = FPY;
ZT = FPZ;

%%% ANGLE CALCULATIONS %%%
COSTHETA = cos(theta);
SINTHETA = sin(theta);
TANTHETA = tan(theta);
```

```
SECTHETA = 1/cos(theta);
COSBETA = cos(beta);
SINBETA = sin(beta);
TANBETA = tan(beta);
COSALPHA = cos(alpha);
SINALPHA = sin(alpha);
COSPHI = cos(phi);
SINPHI = sin(phi);
COSPSI = cos(psi);
SINPSI = sin(psi);

%%% EQUATIONS OF MOTION %%%
V1 = -D*COSBETA+Y*SINBETA+XT*COSALPHA*COSBETA;
V2 = YT*SINBETA+ZT*SINALPHA*COSBETA;
V3 = -m*g*(COSALPHA*COSBETA*SINTHETA);
V4 = -m*g*(-SINBETA*SINPHI*COSTHETA);
V5 = -m*g*(-SINALPHA*COSBETA*COSPHI*COSTHETA);

al1 = -L+ZT*COSALPHA-XT*SINALPHA;
al2 = m*g*(COSALPHA*COSPHI*COSTHETA+SINALPHA*SINTHETA);
al3 = q-TANBETA*(p*COSALPHA+r*SINALPHA);

q1 = SumL*I2+SumM*I4+SumN*I5-p^2*(Ixz*I4-Ixy*I5);
q2 = p*q*(Ixz*I2-Iyz*I4-Dz*I5)-p*r*(Ixy*I2+Dy*I4-Iyz*I5);
q3 = q^2*(Iyz*I2-Ixy*I5)-q*r*(Dx*I2-Ixy*I4+Ixz*I5);
q4 = -r^2*(Iyz*I2-Ixz*I4);

p1 = SumL*I1+SumM*I2+SumN*I3-p^2*(Ixz*I2-Ixy*I3);
p2 = p*q*(Ixz*I1-Iyz*I2-Dz*I3)-p*r*(Ixy*I1+Dy*I2-Iyz*I3);
p3 = q^2*(Iyz*I1-Ixy*I3)-q*r*(Dx*I1-Ixy*I2+Ixz*I3);
```

```
    p4 = -r^2*(Iyz*I1-Ixz*I2);

    r1 = SumL*I3+SumM*I5+SumN*I6-p^2*(Ixz*I5-Ixy*I6);
    r2 = p*q*(Ixz*I3-Iyz*I5-Dz*I6)-p*r*(Ixy*I3+Dy*I5-Iyz*I6);
    r3 = q^2*(Iyz*I3-Ixy*I6)-q*r*(Dx*I3-Ixy*I5+Ixz*I6);
    r4 = -r^2*(Iyz*I3-Ixz*I5);

    be1 = D*SINBETA+Y*COSBETA-XT*COSALPHA*SINBETA;
    be2 = YT*COSBETA-ZT*SINALPHA*SINBETA;
    be3 = m*g*(COSALPHA*SINBETA*SINTHETA);
    be4 = m*g*(COSBETA*SINPHI*COSTHETA);
    be5 = m*g*(-SINALPHA*SINBETA*COSPHI*COSTHETA);
    be6 = p*SINALPHA-r*COSALPHA;

    xp1 = COSALPHA*COSBETA*COSTHETA*COSPSI;
    xp2 = SINBETA*(SINPHI*SINTHETA*COSPSI-COSPHI*SINPSI);
     xp3  =  SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*-
COSPSI);

    yp1 = COSALPHA*COSBETA*COSTHETA*SINPSI;
    yp2 = SINBETA*(COSPHI*COSPSI+SINPHI*SINTHETA*SINPSI);
     yp3  =  SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*-
COSPSI);

    h1 = COSALPHA*COSBETA*SINTHETA;
    h2 = SINBETA*SINPHI*COSTHETA;
    h3 = SINALPHA*COSBETA*COSPHI*COSTHETA;

   %disp('before flag check')
   %flag,t,x,u,sys
```

```
%pause

if flag == 0

 %%% SYSTEM CHARACTERISTICS/INITIAL CONDITIONS %%%
 sys = [12 0 14 6 12 0];
 x0 = xx;

 %%% LONGITUDINAL STATES %%%
 V = x0(1); %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
 alpha = x0(2); %%% ANGLE OF ATTACK (RAD) %%%
 q = x0(3); %%% PITCH RATE (RAD/S) %%%
 theta = x0(4); %%% PITCH ANGLE (RAD) %%%

 %%% LATERAL/DIRECTIONAL STATES %%%
 p = x0(5); %%% ROLL RATE (RAD/S) %%%
 phi = x0(6); %%% ROLL ANGLE (RAD) %%%
 r = x0(7); %%% YAW RATE (RAD/S) %%%
 psi = x0(8); %%% YAW ANGLE (RAD) %%%
 beta = x0(9); %%% SIDESLIP ANGLE (RAD) %%%

 %%% EARTH-RELATIVE POSITION STATES %%%
 xp = x0(10); %%% X-DIRECTION POSITION (FT) %%%
 yp = x0(11); %%% Y-DIRECTION POSITION (FT) %%%
 h = x0(12); %%% ALTITUDE (FT) %%%

elseif abs(flag) == 1

 %%% SYSTEM STATES (X) %%%
 %%% LONGITUDINAL STATES %%%
```

```
V = x(1); %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
alpha = x(2); %%% ANGLE OF ATTACK (RAD) %%%
q = x(3); %%% PITCH RATE (RAD/S) %%%
theta = x(4); %%% PITCH ANGLE (RAD) %%%

%%% LATERAL/DIRECTIONAL STATES %%%
p = x(5); %%% ROLL RATE (RAD/S) %%%
phi = x(6); %%% ROLL ANGLE (RAD) %%%
r = x(7); %%% YAW RATE (RAD/S) %%%
psi = x(8); %%% YAW ANGLE (RAD) %%%
beta = x(9); %%% SIDESLIP ANGLE (RAD) %%%

%%% EARTH-RELATIVE POSITION STATES %%%
xp = x(10); %%% X-DIRECTION POSITION (FT) %%%
yp = x(11); %%% Y-DIRECTION POSITION (FT) %%%
h = x(12); %%% ALTITUDE (FT) %%%

%%% INPUTS (U) %%%

DH = u(1)+utrim(1); %%% SYMETRIC STABILATOR DEFLECTION (DEG)
%%%
PLAPL = u(2)+utrim(2); %%% LEFT ENGINE POWER LEVEL ANGLE
(DEG) %%%
PLAPR = u(3)+utrim(3); %%% RIGHT ENGINE POWER LEVEL ANGLE
(DEG) %%%
FLAPS = u(4)+utrim(4); %%% Flaps (DEG) %%%
DA = u(5)+utrim(5); %%% DIFFERENTIAL AILERON DEFLECTION
(DEG) %%%
DR = u(6)+utrim(6); %%% RUDDER DEFLECTION (DEG) %%%
PLASYM = 0;
```

```
DD = 0;

%%% STATE DERIVATIVES (dX/dT) %%%
sys(1,1) = (V1+V2+V3+V4+V5)/m;
sys(2,1) = al3+(al1+al2)/(V*m*COSBETA);
sys(3,1) = (q1+q2+q3+q4)/detI;
sys(4,1) = q*COSPHI-r*SINPHI;
sys(5,1) = (p1+p2+p3+p4)/detI;
sys(6,1) = p+q*SINPHI*TANTHETA+r*COSPHI*TANTHETA;
sys(7,1) = (r1+r2+r3+r4)/detI;
sys(8,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA;
sys(9,1) = be6+(be1+be2+be3+be4+be5)/(m*V);
sys(10,1) = V*(xp1+xp2+xp3);
sys(11,1) = V*(yp1+yp2+yp3);
sys(12,1) = V*(h1-h2-h3);

elseif flag == 3

%%% SYSTEM OUTPUTS (Y) %%%

sys(1,1) = (V1+V2+V3+V4+V5)/m/g; %%% V_DOT/G %%%
sys(2,1) = x(1)-xx(1); %%% V %%%
sys(3,1) = x(12)-xx(12); %%% H %%%
sys(4,1) = x(4)-xx(4)-x(2)+xx(2); %%% GAMMA %%%
sys(5,1) = x(3)-xx(3); %%% Q %%%
sys(6,1) = x(4)-xx(4); %%% THETA %%%
sys(7,1) = x(2)-xx(2); %%% ALPHA %%%
sys(8,1) = x(5)-xx(5); %%% P %%%
sys(9,1) = x(6)-xx(6); %%% PHI %%%
if V < 1e-3
```

```
sys(10,1) = 0;
else
sys(10,1) = be6+(be1+be2+be3+be4+be5)/(m*V); %%% BETA_DOT %%%
end
sys(11,1) = x(9)-xx(9); %%% BETA %%%
sys(12,1) = x(8)-xx(8); %%% PSI %%%
sys(13,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA; %%% PSI_DOT %%%
sys(14,1) = x(7)-xx(7); %%% R %%%

else

%%% ALL OTHER FLAGS UNDECLARED %%%
sys = [];

end

end
```

Appendix B Linearized State-Space Models

The following are state-space models of the linearized F-15 model for the two flight points.

States = [ΔV(ft/sec), $\Delta\alpha$(rad) ,Δq(rad/sec), $\Delta\theta$(rad), Δp(rad/sec), $\Delta\varphi$(rad), Δr(rad/sec), $\Delta\psi$(rad), $\Delta\beta$(rad), Δh(ft)]^T

Inputs = [$\Delta\delta_H$(deg), $\Delta\delta_{PLAL}$(deg), $\Delta\delta_{PLAR}$(deg), $\Delta\delta_{Flap}$(deg), $\Delta\delta_a$(deg), $\Delta\delta_r$(deg)]^T

Outputs = [$\Delta\dot{V}/g$, ΔV(ft/sec), Dh(ft), $\Delta\gamma$(rad), Δq(rad/sec), $\Delta\theta$(rad), $\Delta\alpha$(rad), Δp(rad/sec), $\Delta\varphi$(rad), $\Delta\dot{\beta}$(rad/sec), $\Delta\beta$(rad), $\Delta\psi$(rad), $\Delta\dot{\psi}$(rad/sec), Δr(rad/sec)]^T

%%% F-15 State Space Model %%%

%%% FLIGHT POINT #2 %%%

%%% 9,800 ft, 0.5 Mach %%%

```
a1 = [-1.3658e-02 -3.4605e+00 0 -3.2144e+01 0 -1.2813e-05
 -2.1654e-04 -7.7897e-01 1.0000e+00 -2.9875e-05 0 -2.9715e-07
 4.0744e-04 -5.8027e+00 -2.5013e+00 -6.5003e-04 3.1496e-08 4.0402e-08
 0 0 1.0000e+00 0 0 3.2285e-20
 7.4534e-22 2.4167e-19 -1.5331e-19 0 -2.2133e+00 0
 0 0 -8.2426e-20 -3.2493e-20 1.0000e+00 -2.2495e-20
 -9.4443e-23 9.4468e-20 1.8812e-20 0 -7.4308e-02 0
 0 0 -1.0274e-18 -2.6070e-21 0 -2.8039e-19
 1.1581e-22 -2.6938e-20 0 4.3973e-21 7.9981e-02 5.9435e-02
 2.4600e-04 -5.3908e+02 0 5.3908e+02 0 2.1489e-04];

a2 = [0 0 -1.0357e-03 -8.7382e-05
```

```
 0 0 1.6619e-12 1.3835e-08
 -3.1496e-08 0 -4.5464e-13 -3.7846e-09
 1.0241e-18 0 0 0
 1.3971e+00 0 -2.7048e+01 0
 8.0486e-02 0 0 0
 -5.7386e-01 0 4.6745e+00 0
 1.0032e+00 0 0 0
 -9.9680e-01 0 -1.9213e-01 -1.3645e-27
 0 0 -6.6284e-07 0];

a = [a1 a2];

b = [-7.1779e-02 1.0670e-01 1.0147e-01 0 0 0
 -1.5698e-03 -1.5881e-05 -1.5103e-05 0 0 0
 -1.5079e-01 4.3444e-06 4.1314e-06 0 0 0
 0 0 0 0 0 0
 0 0 0 0 1.6930e-01 2.6623e-02
 0 0 0 0 0 0
 0 0 0 0 2.2413e-03 -4.8714e-02
 0 0 0 0 0 0
 -1.1210e-24 1.6663e-24 1.5846e-24 0 -3.8274e-05 6.3187e-04
 0 0 0 0 0 0];

c1 = [-4.2492e-04 -1.0766e-01 0 -1.0000e+00 0 -3.9862e-07
 1.0000e+00 0 0 0 0 0
 0 0 0 0 0 0
 0 -1.0000e+00 0 1.0000e+00 0 0
 0 0 1.0000e+00 0 0 0
 0 0 0 1.0000e+00 0 0
0 1.0000e+00 0 0 0];
```

```
03 3.1566e-03 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 -1.1210e-24 1.6663e-24 1.5846e-24 0 -3.8274e-05 6.3187e-04
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0];

                     %%% F-15 State Space Model %%%

                       %%% FLIGHT POINT #2 %%%
                       %%% 30,000 ft, 0.5 Mach %%%

a1 = [-2.0004e-02 -3.8596e+01 0 -3.2083e+01 0 -2.9032e-05
 -2.6078e-04 -3.5479e-01 1.0000e+00 -2.8115e-04 0 -3.1179e-07
 2.6521e-04 -2.3011e+00 -1.1476e+00 -6.4020e-04 3.1496e-08 1.1296e-08
 0 0 1.0000e+00 0 0 1.9763e-20
 1.3725e-21 -3.2396e-19 -9.3566e-20 0 -1.0499e+00 0
 0 0 -1.0213e-20 -2.0490e-20 1.0000e+00 2.6320e-23
 -1.4110e-22 1.7417e-19 5.3361e-20 0 -7.5227e-03 0
 0 0 -5.4237e-20 -3.8585e-21 0 1.3977e-22
 1.1038e-23 -7.7720e-20 0 -1.1030e-21 1.8428e-01 6.3359e-02
 4.1000e-03 -4.9731e+02 0 4.9731e+02 0 4.5002e-04];
```

```
a2 = [0 0 -4.6970e-04 -2.1911e-04
 0 0 5.3732e-12 8.3523e-08
 -3.1496e-08 0 -4.9397e-13 -7.6785e-09
 5.3267e-20 0 0 0
 1.2101e+00 0 -1.3163e+01 0
 1.9174e-01 0 0 0
 -3.0149e-01 0 1.4547e+00 0
 1.0182e+00 0 0 0
 -9.8287e-01 0 -9.4448e-02 -1.1953e-26
 0 0 -1.0194e-05 0];

a = [a1 a2];

b = [-1.4240e-01 3.8878e-02 3.6972e-02 0 0 0
 -7.0387e-04 -1.4657e-05 -1.3939e-05 0 0 0
 -5.5398e-02 1.3475e-06 1.2814e-06 0 0 0
 0 0 0 0 0 0
 0 0 0 0 5.5926e-02 5.4716e-03
 0 0 0 0 0 0
 0 0 0 0 1.4022e-04 -2.1371e-02
 0 0 0 0 0 0
 -7.7682e-24 2.1208e-24 2.0169e-24 0 2.9883e-06 3.1658e-04
 0 0 0 0 0 0];

c1 = [-6.2352e-04 -1.2030e+00 0 -9.9999e-01 0 -9.0492e-07
 1.0000e+00 0 0 0 0 0
 0 0 0 0 0 0
 0 -1.0000e+00 0 1.0000e+00 0 0
 0 0 1.0000e+00 0 0 0
 0 0 0 1.0000e+00 0 0
 0 1.0000e+00 0 0 0 0
 0 0 0 0 1.0000e+00 0
 0 0 0 0 0 1.0000e+00
 1.1038e-23 -7.7720e-20 0 -1.1030e-21 1.8428e-01 6.3359e-02
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 -5.4237e-20 -3.8585e-21 0 1.3977e-22
 0 0 0 0 0 0];

c2 = [0 0 -1.4640e-05 -6.8295e-06
 0 0 0 0
 0 0 0 1.0000e+00
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 -9.8287e-01 0 -9.4448e-02 -1.1953e-26
 0 0 1.0000e+00 0
 0 1.0000e+00 0 0
 1.0182e+00 0 0 0
 1.0000e+00 0 0 0];

c = [c1 c2];

d = [-4.4385e-03 1.2118e-03 1.1524e-03 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
```

```
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
-7.7682e-24 2.1208e-24 2.0169e-24 0 2.9883e-06 3.1658e-04
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];
```

Appendix C Files used by SANDY to Acquire the Optimum Gains

C.1 Flight Point 1

This file is **zzlatTECS1.dat** which contains the data used by the SANDY program to come up with the optimum gains for the first flight point.

```
 'Nfcmax' 1 1
  2000

 'Npm' 1 1
  1

 'Tf' 1 1
  100

 'Tfctor' 1 1
  2

 'Wp' 1 1
  1

 'F' 9 9
  -0.094448 0.18428 0.063359
  -0.98287 0.00017122959 0.018140034
  0 0.00164792149893461 0.010050294940281
 -13.163 -1.0499 0
  1.2101 3.2045598 0.31352268
  0 0.229667019846649 1.4006864337934
 0 1 2.632e-23
  0.19174 0 0
  0 -0 -0
```

```
1.4547 -0.0075227 0
 -0.30149 0.008034606 -1.2245583
0 -0.0253814946266748 -0.154795909385342
0 0 0
 0 -20 0
 0 -0 -0
0 0 0
 0 0 -20
 0 -0 -0
0 0 0
 1 0 0
 0 0 0
0 0 0
 0 0 0
 0 0 1
0 0 0
 0 0 0
 0 -0.080656 -0.5396

‘G’ 9 2
 0 0
 0 0
 0 0
 0 0
 20 0
 0 20
 0 0
 0 0
 0 0
```

```
'Gam' 9 3
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
1 0 0

'Hs' 6 9
0 0 0
1 0 0
0 0 0
-0.094448 0.18428 0.063359
-0.98287 0.00017122959 0.018140034
0 0.00164792149893461 0.010050294940281
0 1 0
0 0 0
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0
0 0 0
0 -0.0174479237139443 -0.106410881546258
0 0 0
0 0 0
1 0 0
```

```
'Dsu' 6 2
 0 0
 0 0
 0 0
 0 0
 0 0
 0 0

'Dsw' 6 3
 0 0 0
0 0 0
0 0 0
0 0 0
0 -1 0
0 0 -1

'Hc' 3 9
 -0.00538843614841794 -10.4180780028834 -0.0171376540600306
 -8.37362741591059 4.61237608695652e-05 0.00488634347826087
 0 9.4017022855056e-05 0.000573388240708870
 1 0 0
 0 0 0
 0 -0.0174479237139443 -0.106410881546258
0 0 0
 0 0 0
 1 0 0

'Dcu' 3 2
```

```
0 0
0 0
0 0
'Dcw' 3 3
0 0 0
0 -1 0
0 0 -1

'Wspec' 3 6
0 0 0
0 0 1
0 0 0.1
1 0.1 0
0 0 0.1
1 0.1 0

'Alpha' 1 1
0
'Sigmamax' 1 1
-0.15

'Zetamin' 1 1
0.7

'Q' 3 3
0 0 0
0 20 0
0 0 1

'R' 2 2
```

```
0 0
0 0
'nDircu' 1 1
0

'Dircu' 0 0

'nDircy' 1 1
0

'Dircy' 0 0

'A' 2 2
0 0
0 0
'B' 2 6
-1 -1 0
0 -3.65667335253509 -1.01743461636475
1 -1 0
0 -3.65667335253509 1.01743461636475

'C' 2 2
-0.253683784709487 0
0 0.10496436553704

'D' 2 6
1.3514438906174 1.3514438906174 -0.560850786142026
0.164780307434178 0 0
-0.371252547475194 -1.82026154119372 0
0 0 0
```

```
'nAid' 1 1
0
'nBid' 1 1
4
'Bid' 4 5
1 5 1
-10000 10000
1 6 1
-10000 10000
2 5 1
-10000 10000
2 6 1
-10000 10000
'nCid' 1 1
2
'Cid' 2 5
1 1 1
-10000 10000
2 2 1
-10000 10000
'nDid' 1 1
6
'Did' 6 5
1 1 1
-10000 10000
1 2 1
-10000 10000
1 3 1
```

```
-10000 10000
1 4 1
-10000 10000
2 1 1
-10000 10000
2 2 1
-10000 10000
'Nclin' 1 1
3
'nLincoef' 3 1
2
2
2
'Lincoef' 6 4
1 98 1
5
-1 98 2
5
1 98 1
6
1 98 2
6
1 100 1
1
-1 100 1
2
'Linbnds' 3 2
0 0
0 0
0 0
```

```
'Nudnlc' 1 1
0
```

C.2 Flight Point 2

Here is the MATLAB file **zzlatTECS2.dat** which the SANDY program used in to calculate the optimum gains for the second flight point.

```
'Nfcmax' 1 1
2000
'Npm' 1 1
1
'Tf' 1 1
100
'Tfctor' 1 1
2
'Wp' 1 1
1
'F' 9 9
-0.094448 0.18428 0.063359
-0.98287 0.00017122959 0.018140034
0 0.00164792149893461 0.010050294940281
-13.163 -1.0499 0
1.2101 3.2045598 0.31352268
0 0.229667019846649 1.4006864337934
0 1 2.632e-23
0.19174 0 0
0 -0 -0
1.4547 -0.0075227 0
-0.30149 0.008034606 -1.2245583
0 -0.0253814946266748 -0.154795909385342
0 0 0
```

```
0 -20 0
0 -0 -0
0 0 0
0 0 -20 0 -0 -0
0 0 0
1 0 0
0 0 0
0 0 0
0 0 0
0 0 1
0 0 0
0 0 0
0 -0.080656 -0.5396

'G' 9 2
0 0
0 0
0 0
0 0
20 0
0 20
0 0
0 0
0 0

'Gam' 9 3
0 0 0
0 0 0 0 0 0
0 0 0
0 0 0
```

```
0 0 0
0 0 0
0 0 0

1 0 0
‘Hs’ 6 9
0 0 0
1 0 0
0 0 0
-0.094448 0.18428 0.063359
-0.98287 0.00017122959 0.018140034
0 0.00164792149893461 0.010050294940281
0 1 0
0 0 0
0 0 0
0 0 1
0 0 0
0 0 0
1 0 0
0 0 0
0 -0.0174479237139443 -0.106410881546258
0 0 0
0 0 0
1 0 0

‘D0 0
0 0
0 0
0 0
0 0
```

```
‘Dsw’ 6 3
0 0 0
0 0 0

0 0 0

0 0 0

0 -1 0

0 0 -1
‘Hc’ 3 9
-0.00538843614841794 -10.4180780028834 -0.0171376540600306
-8.37362741591059 4.61237608695652e-05 0.00488634347826087
0 9.4017022855056e-05 0.000573388240708870

1 0 0
0 0 0
0 -0.0174479237139443 -0.106410881546258

0 0 0
0 0 0
1 0 0
‘Dcu’ 3 2
0 0
0 0
0 0
‘Dcw’ 3 3
0 0 0
```

```
0 -1 0
 0 0 -1


'Wspec' 3 6
 0 0 0
 0 0 1
0 0 0.1
 1 0.1 0
0 0 0.1
 1 0.1 0

'Alpha' 1 1
 0
'Sigmamax' 1 1
 -0.15
'Zetamin' 1 1
 0.7
'Q' 3 3
 0 0 0
 0 20 0
0 0 1

'R' 2 2
 0 0
 0 0

'nDircu' 1 1
 0
'Dircu' 0 0
```

```
‘nDircy’ 1 1
 0
‘Dircy’ 0 0
‘A’ 2 2
0 0
 0 0
‘B’ 2 6
 -1 -1 0
 0 -3.65667335253509 -1.01743461636475
1 -1 0
 0 -3.65667335253509 1.01743461636475

‘C’ 2 2
 -0.253683784709487 0
 0 0.10496436553704

‘D’ 2 6
 1.3514438906174 1.3514438906174 -0.560850786142026
 0.164780307434178 0 0
-0.371252547475194 -1.82026154119372 0
 0 0 0

‘nAid’ 1 1
 0
‘Aid’ 0 0
‘nBid’ 1 1
 4
‘Bid’ 4 5
 1 5 1
 -10000 10000
```

```
1 6 1
-10000 10000
2 5 1
-10000 10000
2 6 1
-10000 10000
'nCid' 1 1
2
'Cid' 2 5
1 1 1
-10000 10000
2 2 1
-10000 10000
'nDid' 1 1
6
'Did' 6 5
1 1 1
-10000 10000
1 2 1
-10000 10000
1 3 1
-10000 10000
1 4 1
-10000 10000
2 1 1
-10000 10000
2 2 1
-10000 10000
'Nclin' 1 1
3
```

```
‘nLincoef’ 3 1
2
2
2
‘Lincoef’ 6 4
1 98 1
5
-1 98 2
5
1 98 1
6
1 98 2
6
1 100 1
1
-1 100 1
2
‘Linbnds’ 3 2
0 0
0 0
0 0
‘Nudnlc’ 1 1
0
```

Appendix D SANDY Gains

D.1 SANDY Gains for Flight Point 1

$K_p = -7.8735$

$K_\varphi = -5.5103$

$K_\beta = -0.1563$

$K_\psi = -0.4172$

$K_r = 1$

$K_a = 1$

$K_{RI} = 41.6557$

$K_{RP} = -95.7255$

$K_{YP} = 1.0616$

$K_{YI} = 0.6251$

$K_{\dot{\psi}} = -0.3364$

D.2 SANDY Gains for Flight Point 2

$K_p = 1.7888$

$K_\varphi = 0.5208$

$K_\beta = -2.2818$

$K_\psi = -0.9177$

$K_r = 1$

$K_a = 1$

$K_{RI} = 0.2142$

$K_{RP} = 11.7688$

$K_{YP} = 1.0322$

$K_{YI} = 0.4382$

$K_{\dot{\psi}} = -0.5067$

Appendix E Operating Instructions

E.1 Nonlinear Open-Loop Simulation

The following is a step-by-step instruction needed to execute the nonlinear open-loop F-15 simulation:

1) Start MATLAB

2) Using an available editor, modify **trimmod.m** in order to chose the desired flight point.

3) On the MATLAB window type **f25load** in order to activate **f25load.m**.

4) Type **f25sim** in MATLAB in order to bring up the SIMULINK environment.

5) Select *Simulation* menu, select *Parameters* and set the desired time span as well as the method of calculation. This report used the EULER method with a minimum time step of 0.001 seconds in order to avoid numerical inconsistencies.

6) Chose the desired input by clicking on the respective input and typing in the magnitude of the input.

7) Select the *Simulation* menu and choose *Start* to start the simulation.

8) Once the simulation is complete, the selected parameters of the simulation will be stored in the MATLAB workspace. Saving the MATLAB variables will make them available for future usage.

E.2 Nonlinear Closed-Loop Simulation

The following is a summation of the commands used to run the nonlinear closed-loop simulation.

1) Start MATLAB

2) Using an available editor, modify **trimmod.m** in order to choose the desired flight point.

3) On the MATLAB window type **f25load** in order to activate **f25load.m**.

4) The simulations for each flight point have been saved under different file names for convenience. In order to run Flight Point 1 type **simu539n** and for Flight Point 2 type **simu497n**. These commands will set up the SIMULINK environment.

5) Select *Simulation* menu, select *Parameters* and set the desired time span as

well at the method of calculation. This report used the EULER method with a minimum time step of 0.001 seconds in order to avoid numerical inconsistencies.

6) Choose the desired input by clicking on the respective input and typing in the magnitude of the input.

7) Select the *Simulation* menu and choose *Start* to start the simulation.

8) Once the simulation is complete, the selected parameters of the simulation will be stored in the MATLAB workspace. Saving the MATLAB variables will make them available for future usage

E.3 Linear Closed-Loop Simulation

Below is a summary of the necessary commands to run the linearized closed-loop simulation.

1) Start MATLAB

2) The simulations for each flight point have been saved under different file names for convenience. In order to run Flight Point 1 type **simu539** and for Flight Point 2 type **simu497**. These commands will set up the SIMULINK environment.

3) Select *Simulation* menu, select *Parameters* and set the desired time span as well at the method of calculation. This report used the LINSIM method with a minimum time step of 0.01 seconds in order to avoid numerical inconsistencies.

4) Choose the desired input by clicking on the respective input and typing in the magnitude of the input.

5) Select the *Simulation* menu and choose *Start* to start the simulation.

6) Once the simulation is complete, the selected parameters of the simulation will be stored in the MATLAB workspace. Saving the MATLAB variables will make them available for future usage.