AIRCRAFT AUTOPILOT DESIGN USING A SAMPLED-DATA GAIN SCHEDULING TECHNIQUE

A Thesis Presented to
The Faculty of the
School of Electrical Engineering and Computer Science
and Fritz J. and Dolores H. Russ
College of Engineering and Technology
Ohio University

In Partial Fulfillment
of the Requirement for the Degree
Master of Science

Chao Wang

March, 1999
Acknowledgment

I would like to express my special thanks to my advisor, Dr. Douglas A. Lawrence, for the insights he has provided to accomplish this thesis and the guidance given to me. Without this, it is impossible for me to accomplish this thesis.

I would like to thank Dr. Frank van Graas, Dr. Michael Braasch and Dr. David Bayless for agreeing to be members of my thesis committee and giving valuable suggestions. Finally, I would like to thank my husband and family for the help, support, and encouragement.
Contents

1 Introduction .......................... 1
  1.1 Gain Scheduling Methodology for Sampled-Data Systems ............ 3
  1.2 Project Objective ................... 3
  1.3 Notation and Definitions ................. 4

2 Sampled-Data Gain Scheduling Theory 5
  2.1 Sampled-Data Gain Scheduling ............... 5
    2.1.1 Family of Constant Operating Points and Parameterization . 6
    2.1.2 Parameterized Family of Linear Controllers .......... 7
    2.1.3 Gain Scheduled Controller .................. 10
  2.2 An Example .......................... 12

3 F-16 Aircraft Model 16
  3.1 F-16 Aircraft Mathematical Model ................. 16
  3.2 Implementation of F-16 Aircraft Model .................. 19
  3.3 Design Objective .......................... 20

4 F-16 Autopilot Design With Gain Scheduling Technique 22
  4.1 Constant Operating Points and Aircraft Linearization .......... 23
5 Conclusions

Bibliography
List of Figures

3.1 F-16 aircraft model ........................................... 19
4.1 General block diagram of autopilots in landing phase .......... 22
4.2 Constant operating values of angle of attack. .................. 25
4.3 Constant operating values of elevator deflection. ............... 25
4.4 Bode magnitude plot for aircraft pitch angle. .................. 27
4.5 Block diagram of pitch attitude hold autopilot. ................. 28
4.6 Step response of the closed-loop system with pitch attitude hold autopilot. ........................................... 30
4.7 Block diagram of velocity controller. ........................... 31
4.8 Step response of the system with the velocity controller ....... 32
4.9 Block diagram of altitude hold autopilot. ....................... 33
4.10 Step response of the system with altitude hold autopilot. ....... 34
4.11 Block diagram of glide slope autopilot. ........................ 34
4.12 Step response of the system with glide slope autopilot. ......... 36
4.13 Block diagram of flare control autopilot. ....................... 37
4.14 The step response of system with the flare control autopilot. ... 38
4.15 Simulation block diagram of linear continuous-time autopilot. ... 40
4.16 Simulation block diagram of linear sampled-data autopilot. .... 40
4.17 Simulation result of linear continuous-time altitude hold autopilot. . . 42
4.18 Simulation result of linear sampled-data altitude hold autopilot. . . 42
4.19 Simulation result of linear continuous-time glide slope autopilot. . . 43
4.20 Simulation result of linear sampled-data glide slope autopilot. . . . 43
4.21 Simulation of continuous-time flare control autopilot. . . . . . . . 44
4.22 Simulation of sampled-data flare control autopilot. . . . . . . . . 45
4.23 Block diagram of gain scheduled pitch attitude hold autopilot. . . . 52
4.24 Block diagram of linear pitch attitude hold autopilot . . . . . . . . 52
4.25 Gain scheduled pitch attitude hold autopilot step response. . . . . . 53
4.26 Linear pitch attitude hold autopilot step response. . . . . . . . . . 53
4.27 Simulation block diagram of gain scheduled altitude hold autopilot. . 55
4.28 Simulink block diagram of linear altitude hold autopilot. . . . . . 55
4.29 Simulation result of gain scheduled altitude hold autopilot. . . . . . 56
4.30 Simulation result of linear altitude hold autopilot. . . . . . . . . . 56
4.31 Simulation result of gain scheduled glide slope autopilot. . . . . . . 57
4.32 Simulation result of linear glide slope autopilot. . . . . . . . . . . 58
List of Tables

3.1 Test Result of F-16 Aircraft Model ......................... 20

4.1 Operating Points Of Altitude and Mach Number ............... 24
Chapter 1

Introduction

Realistic engineering systems are often nonlinear. There are many approaches in nonlinear control system design, such as linearization, robust control, adaptive control, etc. Through linearizing the nonlinear system, the controller can be designed using the approaches of linear control theory. Therefore, linearization is the most common approach in nonlinear system design. The basic limitation of the design via the linearization approach is the fact that the controller is guaranteed to work only in the neighborhood of a single operating (equilibrium) point. The dynamic behavior of a system changes with the operating region. A typical approach in this situation is gain scheduling.

In many situations, how the dynamics of a system change with its operating points can be identified. The system can be modeled in such a way that the operating points are parameterized by one or more variables, which are called scheduling variables. In such situations, the system may be linearized at several equilibrium points and design a linear feedback controller at each point that performs satisfactorily when the system is operating near the respective operating points. A family of linear controllers can be implemented as a single controller whose parameters change by monitoring the
scheduling variables, which are intended to handle the nonlinear aspects of the design problem. Gain scheduling is a practical method that can extend the validity of the controller operation to a set of operating points and has been proven to be a successful method in many applications in nonlinear control systems design, especially in flight control systems.

There have been numerous theoretical studies on the gain scheduling technique. The framework of gain scheduled controller design and analysis are proposed in [9],[7] and [10]. In [5] and [12], the methods and examples of the application of the gain scheduled technique in flight control systems are given. In [2], Kaminer et al. introduce a nonlinear gain scheduled controller that preserves the input-output properties of the linear closed-loop systems locally about each equilibrium point by providing integral action at the inputs to the plant and differentiating some of the measured outputs before they are fed back to the gain scheduled controller. Lawrence et al. present a stability theorem for slowly varying inputs [6]. The widespread use of digital hardware in controller implementation leads naturally to the investigation of approaches for sampled-data gain scheduling. The framework of sampled-data gain scheduling that builds upon previous results for the continuous-time case is presented in [3], and a stability property of nonlinear sampled-data systems with slowing varying inputs is provided in [4].

In this thesis, the focus is on applying the sampled-data gain scheduling method to aircraft autopilot design. Section 1.1 describes the gain scheduling concept and methodology. Section 1.2 presents the objective of this thesis. Section 1.3 sets the notion and definitions used in this thesis.
1.1 Gain Scheduling Methodology for Sampled-Data Systems

A framework for sampled-data gain scheduled controller design is proposed in [3]. The development of such controllers involves the following steps.

- Compute the constant operating points of a nonlinear plant, parameterized by constant values of a set of scheduling variables, and obtain the corresponding linearized plants.

- For the family of linearized plants, design a parameterized family of linear discrete-time controllers to meet the design goals at each operating point.

- Construct a gain scheduling controller that linearizes to the correct linear controller at each constant operating point.

- Verify the performance of the gain scheduled controller by simulation.

1.2 Project Objective

The objective of this project is to use this gain scheduling technique to design longitudinal autopilots for an F-16 fighter over an automatic landing procedure. The experience gained will be used in Unmanned Aerial Vehicle autopilot design in the future.

The autopilots include pitch attitude control, altitude control, glide slope acquisition, and flare control. Linear sampled-data autopilots are designed by classical design techniques to realize the design goals at operating points. The family of linear
autopilots is obtained by linear interpolation of autopilot parameters at intermediate operating points. Nonlinear gain scheduled controllers are constructed such that their linearizations match the linear autopilots performance at each operating point. The six degree-of-freedom F-16 model is implemented in SIMULINK based on the FORTRAN simulation in [11].

1.3 Notation and Definitions

The notation and definitions in this thesis are as follows:

- Matrices and sub-matrices will be denoted by capital letters, e.g., $A$ denotes a matrix, while $A_{11}$ denotes the upper left submatrix of $A$.

- Constant operating points are denoted with superscript $^0$, e.g., $x^0$.

- MATLAB and SIMULINK commands will be denoted in upper case letters, e.g., TRIM.

- Vectors will be denoted by lower case letters, e.g., $x$, while $x_i$ denotes the $i$-th element of $x$.

- Vectors of scheduling variables will be denoted by $\Theta$.

- A matrix that is a function of the scheduling variable $\Theta$ will be denoted by $A(\Theta)$.

- The Jacobean matrix is defined as follows:
  If $f : \mathbb{R}^n \to \mathbb{R}^m$, then $\frac{\partial f}{\partial x}$ denotes the $p \times n$ Jacobian matrix whose $(i,j)$ entry is the partial derivative $\frac{\partial f_i}{\partial x_j}$.
Chapter 2

Sampled-Data Gain Scheduling Theory

In this chapter, background material on sampled-data system gain scheduling is presented. Section 2.1 deals with the theory of sampled-data gain scheduling. Section 2.2 gives an example to illustrated the harmful effect of the hidden coupler term in gain scheduled controller.

2.1 Sampled-Data Gain Scheduling

As we mentioned in Chapter 1, the framework of sampled-data gain scheduling presented in [3] is similar to the continuous-time gain scheduling technique described in [7]. Consider a nonlinear continuous-time plant,

\[
\begin{align*}
\dot{x}(t) &= f(x(t), u(t), w(t)) \\
z(t) &= h_1(x(t), u(t), w(t), r(t)) \\
y(t) &= h_2(x(t), u(t), w(t)) 
\end{align*}
\]  
(2.1)
where $x(t)$ is state vector, $u(t)$ is control input, $w(t)$ is a measured exogenous signal, $r(t)$ is a reference input signal, $z(t)$ is a regulated signal, and $y(t)$ is an additional measured output for control and/or scheduling purposes.

The discrete-time controller has the form:

$$
\begin{align*}
  x_C[k+1] &= a_1(x_C[k], x_I[k], z[k], y[k], w[k], r[k]) \\
  x_I[k+1] &= a_2(x_C[k], x_I[k], z[k], y[k], w[k], r[k]) \\
  u[k] &= c(x_C[k], x_I[k], z[k], y[k], w[k], r[k])
\end{align*}
$$

(2.2)

In this system, the ideal sampler is used as the interface between the output of continuous-time plant and the input of the discrete-time controller. The zero order hold is used as the interface between the controller output and the plant input.

### 2.1.1 Family of Constant Operating Points and Parameterization

The first step in sampled-data methodology is to compute the family of constant operating points for the plant by solving following equations:

$$
\begin{align*}
  0 &= f(x^o, u^o, w^o) \\
  0 &= h_1(x^o, u^o, w^o, r^o) \\
  y^o &= h_2(x^o, u^o, w^o)
\end{align*}
$$

(2.3)

We assume that the equations above have a family of solutions (this is a general case), and the following parameterization is possible.
Assumption:

For the plant functions,

\[ f : \mathbb{R}^n \times \mathbb{R}^{m_u} \times \mathbb{R}^{m_w} \to \mathbb{R}^n \]

\[ h_1 : \mathbb{R}^n \times \mathbb{R}^{m_u} \times \mathbb{R}^{m_w} \times \mathbb{R}^{m_r} \to \mathbb{R}^q \]

\[ h_2 : \mathbb{R}^n \times \mathbb{R}^{m_u} \times \mathbb{R}^{m_w} \to \mathbb{R}^p \]

we assume that for an \( l \)-dimensional parameter vector \( \Theta = \Theta(w, r, y) \), there exist an open set \( 0 \in \mathcal{O} \subset \mathbb{R}^l \) and functions: \( x^o : \mathcal{O} \to \mathbb{R}^n \), \( u^o : \mathcal{O} \to \mathbb{R}^{m_u} \), \( w^o : \mathcal{O} \to \mathbb{R}^{m_w} \), \( r^o : \mathcal{O} \to \mathbb{R}^{m_r} \), \( y^o : \mathcal{O} \to \mathbb{R}^p \), all zero at the origin, such that

\[
\begin{align*}
0 &= f(x^o(\Theta), u^o(\Theta), w^o(\Theta)) \\
0 &= h_1(x^o(\Theta), u^o(\Theta), w^o(\Theta), r^o(\Theta)) \\
y^o(\Theta) &= h_2(x^o(\Theta), u^o(\Theta), w^o(\Theta))
\end{align*}
\]

for \( \Theta \in \mathcal{O} \). Here \( \Theta \) is the scheduling variable, depending on \( w, r \) and \( y \).

### 2.1.2 Parameterized Family of Linear Controllers

The plant is linearized about the parameterized family of constant operating points to yield:

\[
\begin{align*}
\dot{x}_\delta(t) &= A(\Theta)x_\delta(t) + B_1(\Theta)u_\delta(t) + B_2(\Theta)w_\delta(t) \\
z_\delta(t) &= H_1(\Theta)x_\delta(t) + J_{11}(\Theta)u_\delta(t) + J_{12}(\Theta)w_\delta(t) + J_{13}(\Theta)r_\delta(t) \\
y_\delta(t) &= H_2(\Theta)x_\delta(t) + J_{21}(\Theta)u_\delta(t) + J_{22}(\Theta)w_\delta(t)
\end{align*}
\]
where the deviation variables are defined by

\[
x_\delta(t) = x(t) - x^o(\Theta)
\]
\[
u_\delta(t) = u(t) - u^o(\Theta)
\]
\[
w_\delta(t) = w(t) - w^o(\Theta)
\]
\[
z_\delta(t) = z(t)
\]
\[
r_\delta(t) = r(t) - r^o(\Theta)
\]
\[
y_\delta(t) = y(t) - y^o(\Theta)
\]

and the matrices for the parameterized linear state space equation are computed by taking, for example,

\[
A(\Theta) = \frac{\partial f}{\partial x}(x^o(\Theta), u^o(\Theta), w^o(\Theta)), \quad B_1(\Theta) = \frac{\partial f}{\partial u}(x^o(\Theta), u^o(\Theta), w^o(\Theta)).
\]

The plant (2.1) can be discretized as following. Let \(\phi(t, \tau, x, u, w)\) denote the solution of \(\dot{x}(t) = f(x(t), u(t), w(t))\) at time \(t\), where the plant is initialized at state \(x\) at time \(\tau\). Assuming that signals \(w(t)\) and \(z_\delta(t)\) are constants on each sampling interval and writing

\[
u(t) = u[k], \quad w(t) = w[k], \quad t \in [kh, (k + 1)h).
\]

\(x[k] = x(kh)\) satisfies the nonlinear difference equation

\[
x[k + 1] = \phi_h(x[k], u[k], w[k]).
\]

where \(\phi_h(x, u, w) = \phi(h, 0, x, u, w)\). For a constant reference signal \(r[k] = r(t)\) on each sampling interval, we also have

\[
z[k] = h_1(x[k], u[k], w[k], r[k])
\]
\[ y[k] = h_2(x[k], u[k], w[k]) \]

Linearizing the discretized plant about its family of constant operating points yields

\[
x_{\delta}[k + 1] = F(\Theta)x_{\delta}[k] + G_1(\Theta)u_\delta[k] + G_2(\Theta)w_\delta[k]
\]

\[
z_\delta[k] = H_1x_{\delta}[k] + J_{11}(\Theta)u_\delta[k] + J_{12}(\Theta)w_\delta[k] + J_{13}(\Theta)r_\delta[k]
\]  

\[
y_\delta[k] = H_2(\Theta)x_{\delta}[k] + J_{21}u_\delta[k] + J_{22}(\Theta)w_\delta[k]
\]  

where the coefficient matrices are calculated under our smoothness assumption as,

\[
F(\Theta) = \frac{\partial \phi_h}{\partial x}(x^o(\Theta), u^o(\Theta), w^o(\Theta)) = e^{A(\Theta)h}
\]

\[
G_1(\Theta) = \frac{\partial \phi_h}{\partial u}(x^o(\Theta), u^o(\Theta), w^o(\Theta)) = \int_0^h e^{A(\Theta)\tau} d\tau B_1(\Theta)
\]

\[
G_2(\Theta) = \frac{\partial \phi_h}{\partial w}(x^o(\Theta), u^o(\Theta), w^o(\Theta)) = \int_0^h e^{A(\Theta)\tau} d\tau B_2(\Theta)
\]

Other matrices are the same as in (2.5). The deviation variables are defined, for example, \(x_{\delta}[k] = x[k] - x^o(\Theta)\), etc. Suppose a family of linear discrete-time controllers can be designed on the basis of the family of discretized plants (2.6) and can be expressed as

\[
x_{C\delta}[k + 1] = A_{11}(\Theta)x_{C\delta}[k] + A_{12}(\Theta)x_{I\delta}[k] + B_{10}(\Theta)z_\delta[k] + B_{11}(\Theta)y_\delta[k]
\]

\[+ B_{12}(\Theta)w_\delta[k] + B_{13}(\Theta)r_\delta[k] \]

\[
x_{I\delta}[k + 1] = A_{21}(\Theta)x_{C\delta}[k] + A_{22}(\Theta)x_{I\delta}[k] + B_{20}(\Theta)z_\delta[k] + B_{21}(\Theta)y_\delta[k]
\]

\[+ B_{22}(\Theta)w_\delta[k] + B_{23}(\Theta)r_\delta[k] \]  

\[
u_\delta[k] = C_1(\Theta)x_{C\delta}[k] + C_2(\Theta)x_{I\delta}[k] + D_0(\Theta)z_\delta[k] + D_1(\Theta)y_\delta[k]
\]  

(2.7)
+D_2(\Theta)w_\delta[k] + D_3(\Theta)r_\delta[k]

where \( x_{\delta} \) has dimension \( n_c \), \( x_{\delta} \) has dimension \( n_z \), and

\[
x_{\delta}[k] = x_c[k] - x_c^o(\Theta), \quad x_{\delta}[k] = x_z[k] - x_z^o(\Theta)
\]

This is a general form of a linear discrete-time controller family. It can be used in a variety of linear controller schemes involving the discrete-time counterpart of pure integral action. That means there are controller eigenvalues at \( z = 1 \). For example, 
\( A_{21}(\Theta) = 0, A_{22}(\Theta) = B_{20}(\Theta) = I \) and \( B_{2j}(\Theta) = 0, j = 1, 2, 3 \), yields

\[
x_{\delta}[k + 1] = x_{\delta}[k] + z_\delta[k]
\]

2.1.3 Gain Scheduled Controller

The third step in the design is to construct a discrete-time controller of the form (2.2), whose linearizations exactly match the family of linear discrete-time controller designed in the second step at the operating points. This requirement can be stated as

**Requirement 1:** The gain scheduled controller (2.2) must be such that for

\[
a_1 : R^{nc} \times R^{nz} \times R^q \times R^p \times R_{mw} \times R_{mr} \rightarrow R^{nc}
\]

\[
a_2 : R^{nc} \times R^{nz} \times R^q \times R^p \times R_{mw} \times R_{mr} \rightarrow R^{nz}
\]

\[
c : R^{nc} \times R^{nz} \times R^q \times R^p \times R_{mw} \times R_{mr} \rightarrow R^{m_z}
\]

there exist an open set \( 0 \in \mathcal{O} \subseteq R^l \) and functions \( x_c^o : 0 \rightarrow R^{nc} \) and \( x_z^o : 0 \rightarrow R^{nz} \), both zero at the origin, satisfying

\[
x_c^o(\Theta) = a_1(x_c^o(\Theta), x_z^o(\Theta), 0, y^o(\Theta), w^o(\Theta), r^o(\Theta))
\]
for $\Theta \in \mathcal{O}$. Furthermore, the controller (2.2) should be linearized to (2.7) at each constant operating point. Thus, for example, we require that

$$
\frac{\partial a_1}{\partial x_c}(x_c^0(\Theta), x_T^0(\Theta), 0, y^0(\Theta), w^0(\Theta), r^0(\Theta)) = A_{11}(\Theta)
$$

$$
\frac{\partial c}{\partial r}(x_c^0(\Theta), x_T^0(\Theta), 0, y^0(\Theta), w^0(\Theta), r^0(\Theta)) = D_3(\Theta)
$$

for $\Theta \in \mathcal{O}$.

This requirement has two parts. In the first part, it specifies that the nonlinear hybrid closed-loop system has a family of constant operating points. The second part of the requirement guarantees that the nonlinear hybrid closed-loop system has a family of linearizations that matches the interconnection of (2.5) and (2.7) via the samplers and zero-order hold.

**Theorem 1** Under Assumption 1 there exists a controller (2.2) satisfying Requirement 1 if and only if there exist an open set $0 \in \mathcal{O} \subset \mathbb{R}^l$ and functions $x_c^0(\Theta)$ and $x_T^0(\Theta)$ satisfying

$$
\begin{bmatrix}
A_{11}(\Theta) - I & A_{12}(\Theta) \\
A_{21}(\Theta) & A_{22}(\Theta) - I \\
C_1(\Theta) & C_2(\Theta)
\end{bmatrix}
\begin{bmatrix}
\frac{\partial x_c^0(\Theta)}{\partial \Theta} \\
\frac{\partial x_T^0(\Theta)}{\partial \Theta}
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
\frac{\partial w^0(\Theta)}{\partial \Theta}
\end{bmatrix}
- 
\begin{bmatrix}
B_{11}(\Theta) & B_{12}(\Theta) & B_{13}(\Theta) \\
B_{21}(\Theta) & B_{22}(\Theta) & B_{23}(\Theta) \\
D_1(\Theta) & D_2(\Theta) & D_3(\Theta)
\end{bmatrix}
\begin{bmatrix}
\frac{\partial y^0(\Theta)}{\partial \Theta} \\
\frac{\partial w^0(\Theta)}{\partial \Theta} \\
\frac{\partial r^0(\Theta)}{\partial \Theta}
\end{bmatrix}
$$

(2.9)

for $\Theta \in \mathcal{O}$.
The proof of the theorem is presented in [3]. This existence condition is restrictive. If it is not satisfied, the linearization of any controller of form (2.2) will not completely match the linear controller at any operating point, which is the so-called hidden coupling problem. If this kind of problem occurs, it may result in significant performance degradation.

The choice of gain scheduled controller meeting Requirement 1 under the existence condition (2.9) is not unique, but it has similar behaviors in the range of the family of operating points (2.8).

2.2 An Example

In this section, we give an example to show the necessity of Requirement 1 and the harmful effect of a hidden coupling term. Consider the first-order plant with linear dynamics and saturating output nonlinearity

\[ \dot{x}(t) = -x(t) + u(t) \]
\[ y(t) = \tanh(x(t)) \]

The regulated variable is the tracking error \( z(t) = r(t) - y(t) \).

Step 1: Choosing the output as the scheduling variable, the plant’s family of constant operating points is calculated as:

\[ 0 = -x^o(\Theta) + u^o(\Theta) \]
then we have:

\[ y^o = \Theta = \tanh(x^o(\Theta)) \]

Step 2: The family of the linearized discretized plant is derived from (2.6)

\[ x_\delta[k+1] = e^{A(\Theta)h} x_\delta[k] + \int_0^h e^{A(\Theta)\tau} d\tau B_1(\Theta) u_\delta[k] + \int_0^h e^{A(\Theta)\tau} d\tau B_2(\Theta) w_\delta[k] \]

with \( A(\Theta) = \frac{\partial f}{\partial x} = -1, \quad B_1(\Theta) = \frac{\partial f}{\partial u} = 1 \) and \( B_2(\Theta) = \frac{\partial f}{\partial w} = 0 \). We obtain,

\[ x_\delta[k+1] = e^{-h} x_\delta[k] + \int_0^h e^{-\tau} d\tau u_\delta[k] = e^{-h} x_\delta[k] + (1 - e^{-h}) u_\delta[k] \]

\[ y_\delta[k] = H(\Theta) x_\delta[k] \]

\[ H(\Theta) = \frac{d \tanh(x)}{dx} |_{x=x^o(\Theta)} = 1 - \Theta^2 \]

\[ z_\delta[k] = r_\delta[k] - y_\delta[k] \]

Step 3: Consider a family of discrete-time PI controllers

\[ x_{I\delta}[k+1] = x_{I\delta}[k] + z_\delta[k] \]

\[ u_\delta[k] = k_I(\Theta) x_{I\delta}[k] + k_p(\Theta) z_\delta[k] \]

Selecting the gains according to

\[ k_p(\Theta) = \frac{e^{-h}}{(1 - e^{-h}) H(\Theta)} \]

\[ k_I(\Theta) = \frac{1}{4(1 - e^{-h}) H(\Theta)} \]
results in a family of close-loop discrete-time systems which has characteristic polynomial

\[ z^2 - z + 1/4 = (z - 1/2)^2 \]

at each \( \Theta \in (-1, 1) \). For the family of linear controllers, the existence condition (2.9) reduces to

\[ k_I(\Theta) \frac{dx^o(\Theta)}{d\Theta} = \frac{du^o(\Theta)}{d\Theta} \]  

(2.10)

Consider the gain scheduled controller obtained by simply replacing \( \Theta \) with time varying sampled plant output \( y[k] \). The controller is

\[
x_I[k + 1] = x_I[k] + z[k]
\]

\[
u[k] = k_I(y[k])x_I[k] + k_p(y[k])z[k]
\]

This nonlinear gain scheduled controller satisfies the first part of Requirement 1 by taking

\[ x^o_1(\Theta) = \frac{1}{k_I(\Theta)} u^o(\Theta). \]

The controller obviously satisfies the partial derivatives with respect to \( x_I \) and \( z \). However, for the partial derivative with \( y \), a direct calculation leads to

\[
\frac{\partial c(x^o_1(\Theta), 0, y^o(\Theta))}{\partial y} = \frac{dk_I(\Theta)}{d\Theta} x^o_1(\Theta) + \frac{dk_p(\Theta)}{d\Theta} z^o(\Theta)
\]

\[
= \frac{du^o(\Theta)}{d\Theta} - k_I(\Theta) \frac{\partial x^o_1(\Theta)}{\partial \Theta}
\]

\[
= \frac{2\Theta \tanh^{-1}(\Theta)}{1 - \Theta^2}
\]
Clearly, the existance condition (2.10) is violated. This nonzero term above is the output of the sampled-data system when the controller is linearized at the operating point $\Theta$. It is different with the linear controller family at this point. This is a so-called hidden coupling term. The closed-loop characteristic polynomial is

$$z^2 - (1 + 2(1 - e^{-h})\Theta \tanh^{-1}(\Theta))z + (1/4 + 2(1 - e^{-h})\Theta \tanh^{-1}(\Theta)), \Theta \in (-1, 1)$$

which has roots outside the unit circle for sufficiently large $|\Theta| < 1$ for any sample period $h$. The closed-loop system is unstable under this condition.

But if $x_I^o(\Theta)$ is chosen to satisfy (2.10), then

$$x_I^o(\Theta) = \int_0^\Theta \frac{1}{k_I(\beta)} \frac{du^o(\beta)}{d\Theta} d\beta = 4(1 - e^{-h})\Theta,$$

and we get the controller

$$x_I[k + 1] = x_I[k] + z[k]$$
$$u[k] = k_I(y[k])[x_I[k] - x_I^o(y[k])] + k_p(y[k])z[k] + u^o(y[k])$$

which fully satisfies Requirement 1. The closed-loop system is always stable for any $|\Theta| < 1$. 
Chapter 3

F-16 Aircraft Model

This thesis applies the gain scheduling method described in Chapter 2 to F-16 aircraft autopilot design. Section 3.1 details the aircraft model and equations. Section 3.2 presents the implementation of an F-16 aircraft simulation. Section 3.3 lists the design objectives.

3.1 F-16 Aircraft Mathematical Model

The aircraft autopilot is a device which can maintain or change the aircraft flight conditions. It can hold aircraft attitude, altitude, and flight trajectory by controlling the control surface deflection of the aircraft and engine thrust.

The autopilot design in this thesis involves a six-degree-of freedom model of an F-16 aircraft. The mathematical model of the F-16 is built in SIMULINK based on the FORTRAN simulation in [11].

The following standard 6-DOF (degree of freedom) equations which describe the dynamics of the F-16 aircraft under a flat-Earth assumption. The derivation of the aircraft equations is the vector form of Newton’s second law of motion. The wind velocity components are assumed to be zero, i.e. a stationary air mass.
Force Equation

\[ \begin{align*}
\dot{U} & = RV - QW - g_0 \sin \theta + \frac{F_x}{m}; \\
\dot{V} & = -RU + PW + g_0 \sin \phi \cos \theta + \frac{F_y}{m}; \\
\dot{W} & = QU - PV + g_0 \cos \phi \cos \theta + \frac{F_z}{m}; 
\end{align*} \]  
(3.1)

Kinematic Equations

\[ \begin{align*}
\dot{\phi} & = P + \tan \theta (Q \sin \phi + R \cos \phi); \\
\dot{\theta} & = Q \cos \phi - R \sin \phi; \\
\dot{\psi} & = \frac{Q \sin \phi + R \sin \phi}{\cos \theta}; 
\end{align*} \]  
(3.2)

Moment Equations

\[ \begin{align*}
\dot{P} & = (c_1R + c_2P)Q + c_3L + c_4N; \\
\dot{Q} & = c_5PR - c_6(P^2 - R^2) + c_7M; \\
\dot{R} & = (c_8P - c_2R)Q + c_4L + c_9N; 
\end{align*} \]  
(3.3)

Navigation Equations

\[ \begin{align*}
\dot{p}_N & = U \cos \theta \cos \psi + V (\cos \phi \sin \psi + \sin \phi \sin \theta \cos \psi) \\
& \quad + W (\sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi); \\
\dot{p}_E & = U \cos \theta \sin \psi + V (\cos \phi \cos \psi + \sin \phi \sin \theta \sin \psi) \\
& \quad + W (\sin \phi \cos \psi + \cos \phi \sin \theta \sin \psi); 
\end{align*} \]  
(3.4)
\[ \dot{h} = U \sin \theta - V \sin \phi \cos \theta - W \cos \phi \cos \theta; \]

The elements of the state vector comprise the components of the Velocity Vector \( U, V \) and \( W (ft/s) \), the Vector of Euler Angles \( \phi, \theta \) and \( \psi (rad) \), the Angular Rate Vector \( P, Q \) and \( R (rad/s) \), and the Position Vector \( p_N, p_E \) and \( h (ft) \), where, in particular, \( h \) is altitude. The earth’s gravitational acceleration is \( g_0 = 32.17 ft/s^2 \), and the constants \( c_i \) are given by: \( c_1 = -0.77 \), \( c_2 = 0.02755 \), \( c_3 = 1.055 \times 10^{-4} \), \( c_4 = 1.642 \times 10^{-6} \), \( c_5 = 0.9604 \), \( c_6 = 1.759 \times 10^{-2} \), \( c_7 = 1.972 \times 10^{-5} \), \( c_8 = -0.7336 \) and \( c_9 = 1.587 \times 10^{-5} \).

The force vector \( F_x, F_y \) and \( F_z (lbs) \) and moment vector \( L, M \) and \( N (lbft) \) must be broken into aerodynamic and thrust contributions provided in [11].

The aerodynamic forces and moments on the aircraft depend on the angle of attack and sideslip angle. Also the true airspeed has an effect on forces and moments. It is convenient for us to use the true airspeed \( V_t (ft/s) \), angle of attack \( \alpha (rad) \), and sideslip angle \( \beta (rad) \) to replace the three velocity vector components. The new state derivatives can be derived as:

\[ \dot{V}_t = \frac{UU + VV + WW}{V_t}; \]
\[ \dot{\alpha} = \frac{UW - WU}{U^2 + W^2}; \]
\[ \dot{\beta} = \frac{VV_t - V_tV}{V_t^2 \cos \beta}; \]

Therefore, we have the state vector \( X^T = [V_t, \alpha, \beta, \phi, \theta, \psi, P, Q, R, p_N, p_E, h] \).

A number of the aerodynamic force and moment components depend on the control surface deflections, these are control inputs to the aircraft model. Throttle setting
is another control input. The control variables are

\[ U^T = [th, el, ail, rud] \]

where the elements of the vector are, respectively, throttle setting, elevator deflection, aileron deflection, and rudder deflection.

3.2 Implementation of F-16 Aircraft Model

In this thesis, all simulations are implemented in MATLAB SIMULINK. The block diagram of the F-16 model is

![Aircraft Model Simulation](image)

Figure 3.1: F-16 aircraft model

The algorithm of building the aircraft model is:

- Compute Mach number and dynamic pressure from a standard atmosphere model. Then compute engine thrust for use in the force equations.

- Compute the aerodynamic coefficients for the force equations, compute \( U, V, W \) from \( V_t, \alpha, \beta \), evaluate the force equations, and then compute the state derivatives \( \dot{V_t}, \dot{\alpha}, \dot{\beta} \).
Table 3.1: Test Result of F-16 Aircraft Model

<table>
<thead>
<tr>
<th>Element</th>
<th>$X_{CG}$</th>
<th>$U(i)$</th>
<th>$X(i)$</th>
<th>$\dot{X}(i)$</th>
<th>$\dot{x}_{actual}(i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.4</td>
<td>0.9</td>
<td>500.0</td>
<td>-75.2372</td>
<td>-75.2116</td>
</tr>
<tr>
<td>2</td>
<td>20.0</td>
<td>0.5</td>
<td>-20.0</td>
<td>-0.8814</td>
<td>-0.8812</td>
</tr>
<tr>
<td>3</td>
<td>-15.0</td>
<td>-0.2</td>
<td>-10.0</td>
<td>-0.4760</td>
<td>-0.4760</td>
</tr>
<tr>
<td>4</td>
<td>-20.0</td>
<td>-1.0</td>
<td>2.5057</td>
<td>2.5057</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1.0</td>
<td>0.3251</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-1.0</td>
<td>2.1459</td>
<td>2.1459</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.7</td>
<td>12.6267</td>
<td>12.6263</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-0.8</td>
<td>0.9650</td>
<td>0.9651</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.9</td>
<td>0.5810</td>
<td>0.5810</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10000.0</td>
<td>248.1241</td>
<td>248.1241</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>90.0</td>
<td>-58.6899</td>
<td>-58.6957</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Compute the aerodynamic coefficients for the moment equations, using $\dot{\alpha}$ and $\dot{\beta}$ if necessary, and then evaluate the moment equations.

- Evaluate the kinematic equations and the navigation equations.

A simple check on the aircraft model can be done by using the S-Function ($flag = 1$) with specified control variables and state variables to calculate the derivative vector. The test result $\dot{x}_{actual}(i)$ in Table 3.1 indicates that the model is built correctly.

### 3.3 Design Objective

According to the altitude-Mach range required of modern high-performance aircraft, the performance should be satisfied by the system in the following range.

- Attitude hold and altitude hold autopilots have satisfactory performance over the operating range specified by $0.3 \leq M \leq 0.9$ and $0 \leq h \leq 40k(\text{ft})$. 
• The glide slope autopilot needs to control the flight trajectory in aircraft landing over the operating range specified by $0.3 \leq M \leq 0.9$ and $0 \leq h \leq 40k(ft)$.

• The flare control autopilot must control the aircraft to follow an exponential trajectory from altitude, $h$ less than $30ft$ to touch down.
Chapter 4

F-16 Autopilot Design With Gain Scheduling Technique

In this chapter, we deal with the autopilot design with a sampled-data gain scheduling technique for F-16 aircraft automatic landing procedure. Three stages are included in the aircraft automatic landing sequence, which are altitude hold, glide slope acquisition and flare control. These three autopilots have the same structures shown in Figure 4.1.

Figure 4.1: General block diagram of autopilots in landing phase
The velocity controller is designed to control the airspeed, \( V_t \) at a specified constant \( V_{tc} \), and the pitch attitude hold autopilot in the inner loop is constructed to provide stability augmentation. Here \( \Theta \) is the scheduling variable, \( r \) represents the command altitude for altitude hold and flare control or the command flight path angle for glide slope acquisition, \( y \) represents the measurement output of altitude for altitude hold autopilot, of flight path angle for glide slope acquisition, and of altitude and the derivative of the altitude for flare control.

Section 4.1 gives the calculation of constant operating points and aircraft model linearization. Section 4.2 presents linear autopilot design for specified operating points. Section 4.3 details the nonlinear autopilot design with gain scheduling technique. Section 4.4 compares the simulation results between the linear autopilot and the nonlinear autopilot.

### 4.1 Constant Operating Points and Aircraft Linearization

According to the gain scheduling methodology, the first step of nonlinear gain scheduled controller design is to compute the family of constant operating points for specified steady-state flight conditions and linearize the nonlinear aircraft model about the specified flight condition. The operating points of the F-16 are determined by solving the nonlinear state equations given in Chapter 3 for the state and control variables to make the state derivatives \( \dot{V}_t, \dot{\alpha}, \dot{\beta}, \dot{\phi}, \dot{\psi} \) and \( \dot{P}, \dot{Q}, \dot{R} \) identically zero. The bank angle \( \phi \) and the sideslip angle \( \beta \) are set to be zero. MATLAB TRIM function is used to calculate the values of state and control input for given altitude,
Table 4.1: Operating Points Of Altitude and Mach Number

<table>
<thead>
<tr>
<th>(M,h(ft))</th>
<th>(M,h(ft))</th>
<th>(M,h(ft))</th>
<th>(M,h(ft))</th>
<th>(M,h(ft))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.3,0)</td>
<td>(0.3,10k)</td>
<td>(0.3,20k)</td>
<td>(0.3,30k)</td>
<td>(0.3,40k)</td>
</tr>
<tr>
<td>(0.45,0)</td>
<td>(0.45,10k)</td>
<td>(0.45,20k)</td>
<td>(0.45,30k)</td>
<td>(0.45,40k)</td>
</tr>
<tr>
<td>(0.6,0)</td>
<td>(0.6,10k)</td>
<td>(0.6,20k)</td>
<td>(0.6,30k)</td>
<td>(0.6,40k)</td>
</tr>
<tr>
<td>(0.75,0)</td>
<td>(0.75,10k)</td>
<td>(0.75,20k)</td>
<td>(0.75,30k)</td>
<td>(0.75,40k)</td>
</tr>
<tr>
<td>(0.9,0)</td>
<td>(0.9,10k)</td>
<td>(0.9,20k)</td>
<td>(0.9,30k)</td>
<td>(0.9,40k)</td>
</tr>
</tbody>
</table>

Mach number, flight path angle and center of gravity. The flight path angle, $\Gamma$ is an additional output of the S-function in the aircraft and given by $\Gamma = \theta - \alpha$ for zero roll and sideslip angle. The TRIM command is executed at each points given in the Table 4.1 in the range of altitude and Mach number.

The mesh plots of the constant operating point values of angle of attack $\alpha^o$ and the elevator deflection $\delta_e^o$ over the range of $0 \leq h \leq 40kft$, and $0.3 \leq M \leq 0.9$ are shown in Figure 4.2 and Figure 4.3. Other operating points of state and control variables are ommitted because these plots are representative of the constant operating point functions. The operating points can be checked by evaluating the S-function at each point and checking that the state derivatives are sufficiently small.

Linear aircraft model can be obtained by linearizing the nonlinear aircraft model based on different trim conditions. The linearized aircraft model is decoupled into two sets of equations: longitudinal equations that involve the variables of speed, angle of attack, pitch attitude, pitch rate and altitude, and lateral-directional equations that involve sideslpe angle, bank angle, roll, and yaw rates. In this thesis, we focus on the longitudinal direction.
Figure 4.2: Constant operating values of angle of attack.

Figure 4.3: Constant operating values of elevator deflection.
MATLAB LINMOD command generates the Jacobian matrices \((A, B, C, D)\) for the state-space linear aircraft model from nonlinear F-16 aircraft model corresponding to the specified trim condition. The transfer functions for different inputs and outputs are obtained by Laplace transform of the state-space equations. For example, the longitudinal linear aircraft model in state-space generated by LINMOD command at trim condition of \(M = 0.45, h = 10kft, cg = 0.3\overline{c},\) and \(\Gamma = 0^\circ\) is:

\[
\begin{bmatrix}
\dot{V}_{t\delta}(t) \\
\dot{\alpha}_{\delta}(t) \\
\dot{\theta}_{\delta}(t) \\
\dot{Q}_{\delta}(t) \\
\dot{h}_{\delta}(t) \\
\dot{p\omega}_{\delta}(t)
\end{bmatrix} =
\begin{bmatrix}
-0.0167 & -4.7739 & -32.1700 & -1.1316 & 0 & 0.3007 \\
-0.0003 & -0.7270 & 0 & 0.9277 & 0 & 0 \\
0 & 0 & 0 & 1.0000 & 0 & 0 \\
0 & -1.7260 & 0 & -0.9970 & 0 & 0 \\
0 & -484.5384 & 484.5384 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1.0000
\end{bmatrix}
\begin{bmatrix}
V_{t\delta}(t) \\
\alpha_{\delta}(t) \\
\theta_{\delta}(t) \\
Q_{\delta}(t) \\
h_{\delta}(t) \\
p\omega_{\delta}(t)
\end{bmatrix}
\begin{bmatrix}
0 & 0.0902 \\
0 & -0.0015 \\
0 & 0 \\
0 & -0.1253 \\
0 & 0 \\
64.9351 & 0
\end{bmatrix}
\begin{bmatrix}
V_{t\delta}(t) \\
\alpha_{\delta}(t) \\
\theta_{\delta}(t) \\
Q_{\delta}(t) \\
h_{\delta}(t) \\
p\omega_{\delta}(t)
\end{bmatrix}
\]

After canceling some very close pole-zero pairs, the corresponding transfer function from elevator deflection to pitch angle, \(\frac{\theta}{\delta_e}\), is given by:

\[
\frac{\theta}{\delta_e} = \frac{-7.1814(s + 0.7068)(s + 0.0146)}{(s + 0.8646 \pm 1.2570i)(s + 0.0051 \pm 0.0848i)}
\]

The transfer function from throttle setting to airspeed, \(\frac{V_t}{Th}\), is:

\[
\frac{V_t}{Th} = \frac{19.5241(s + 0.8641 \pm 1.2594i)(s - 0.0017 \pm 0.0272i)}{(s + 0.8646 \pm 1.2570i)(s + 1)(s + 0.0051 \pm 0.0848i)(s + 0.0012)}
\]
The various linear autopilots are designed based on the different transfer functions calculated from linearizations at the operating points listed in Table 4.1. The dynamic characteristics of linear aircraft model obtained by the linearization approach change with the operating points. For example, the different frequency response characteristics from elevator deflection to pitch angle at the following operating points \((M^o, h^o) = (0.3, 0\, ft), (0.3, 40k\, ft), (0.9, 0\, ft),\) and \((0.9, 40k\, ft)\) are illustrated in Figure 4.4. Different linear autopilots need to be designed for different trim conditions to meet the design goals.

![Bode magnitude plot for aircraft pitch angle.](image)

**Figure 4.4**: Bode magnitude plot for aircraft pitch angle.

### 4.2 Linear Autopilot Design

Linear autopilot design is the second step of the gain scheduling technique. Continuous-time linear autopilots are designed first; then sampled-data autopilots are designed using the bilinear transformation. The twenty five design points shown in Table 4.1 will cover the range of altitude and Mach number for altitude hold and glide slope
autopilots. The flare control autopilot designed via linearization approach has a satisfactory performance for the range of the flare phase, one linear flare control autopilot is designed for the landing sequence.

4.2.1 Continuous-Time Linear Autopilot Design

Figure 4.1 shows that the velocity controller and pitch attitude hold controller are involved in each phase of landing sequence. They are designed as follows.

Pitch Attitude Hold Autopilot and Velocity Controller

1. Pitch attitude hold autopilot.

The controlled variable of pitch attitude hold controller is pitch angle, $\theta$. The block diagram of this autopilot is shown in Figure 4.5.

![Figure 4.5: Block diagram of pitch attitude hold autopilot.](image)

The transfer function from elevator deflection to pitch angle, $\frac{\theta}{\delta_e}$, comes from the specified trim condition. Pitch attitude hold autopilots are designed at the 25 points of Table 4.1. The transfer function, $\frac{\theta}{\delta_e}$, in Section 4.1 is used as an example to show how to use the classical method in [8] to design pitch attitude hold controller. The
transfer function, \( \frac{\theta}{\delta_c} \), is:

\[
\frac{\theta}{\delta_c} = \frac{-7.1814(s + 0.7068)(s + 0.0146)}{(s + 0.8646 \pm 1.2570i)(s + 0.0051 \pm 0.0848i)}
\]

The dynamics above are augmented with the first-order lag actuator model shown in the Figure 4.5. Let \( G(s) = \frac{\theta}{\delta_c} \) denotes the transfer function of the original system. The transfer function, \( \frac{\theta}{\delta_c} \) is

\[
G(s) = \frac{\theta}{\delta_c} = \frac{-71.814(s + 0.7068)(s + 0.0146)}{(s + 0.8646 \pm 1.2570i)(s + 0.0051 \pm 0.0848i)(s + 10)}
\]

A PID controller which is a commonly employed controller in closed-loop control systems will be designed as following. Let \( G_c(s) = k_p + \frac{k_i}{s} + k_ds \) \( (k_p, k_i \text{ and } k_d \text{ are proportional, integral and derivative gains respectively}) \) denotes the controller transfer function. The following notations are through all this chapter. The frequency, \( \omega_1 \) denotes the compensated gain crossover frequency corresponding to the required phase margin, \( \Phi_m \) and is determined by the system settle time, \( T_s \) as:

\[
\omega_1 \geq \frac{8}{T_s \tan \Phi_m}
\]

If the required phase margin is 45° and the settle time is less than 4s, \( \omega_1 \) should be chosen from \( \omega_1 \geq 2rad/s \). We chose \( \omega_1 = 8rad/s \) by trial and error. The phase angle of the compensated open-loop system at \( \omega_1 \) is:

\[
arg(G_c(j\omega_1)G(j\omega_1)) = -180^\circ + \Phi_m
\]

The autopilot phase angle denoted by \( \varphi \) is calculated as:

\[
\varphi = argG_c(j\omega_1) = -180^\circ + \Phi_m - argG(j\omega_1)
\]
If the integral gain is chosen at \( k_i = 0.01 \), the proportional and derivative gains are given by:

\[
k_p = \frac{\cos \varphi}{|G(j\omega_1)|} = 3.6448
\]

\[
k_d = \frac{\sin \varphi}{w_1|G(j\omega_1)|} + \frac{k_i}{\omega^2} = 1.3276
\]

The unit step response of the closed-loop system is shown in Fig. 4.6.

![Step Response](image)

Figure 4.6: Step response of the closed-loop system with pitch attitude hold autopilot.

2. Velocity Control

The block diagram of the velocity controller is in Figure 4.7:

In the velocity controller designs, the transfer function from throttle setting to airspeed obtained in section 4.1 and throttle actuator model shown in Figure 4.7 act as the original system. The linear velocity controller designed at \((M^o, h^o) = (0.45, 10kft)\) performs satisfactorily in the range of altitude and Mach number. The first controller in the velocity controller is a PI controller which makes the system Type 1. A phase lead controller is used to increase the system stability margin.
Figure 4.7: Block diagram of velocity controller.

Velocity and the rate of change of velocity are used in feedback path and $k_d$ is chosen at $k_d = 2.5$ according to [1]. The transfer function $G(s) = \frac{V}{u_t}$ is:

$$G(s) = \frac{1.95241(s + 0.8641 \pm 1.2594i)(s - 0.0017 \pm 0.0272i)}{s(s + 0.8646 \pm 1.2570i)(s + 1)(s + 0.0051 \pm 0.0848i)(s + 0.0012)}$$

The PI controller is designed based on the manner of [11]. The zero of the PI controller is chosen at $s = -0.1$ to cancel the pole of the throttle actuator. The phase lead controller is designed as described in [8] and is presented as following. The required phase margin is $\Phi_m = 50^\circ$, and $\omega_1$ is chosen at $1.4 rad/s$ by the same method as in PID controller design. The angle of the phase lead controller at $\omega_1$ is computed by:

$$\varphi = -180^\circ + \Phi_m - \angle G(j\omega_1)$$

Then the parameters of the phase lead compensator are:

$$a_1 = \frac{1 - |G(j\omega_1)| \cos \varphi}{\omega_1 |G(j\omega_1)| \sin \varphi} = 2.042$$

$$b_1 = \frac{\cos \varphi - |G(j\omega_1)|}{\omega_1 \sin \varphi} = 1.029$$

The transfer function of the velocity controller is obtained as

$$\frac{2.042s + 1}{1.029s + 1} \left( \frac{0.1}{s} \right)$$
The step response of the velocity controller is shown in Figure 4.8. It shows that the velocity loop does not have a significant effect on the short-period dynamics of the aircraft.

![Step Response](image)

Figure 4.8: Step response of the system with the velocity controller

**Altitude Hold Autopilot**

The altitude hold autopilot allows the aircraft to be held at a fixed altitude. The controlled variable is the altitude, $h$. A phase lead controller and a PI controller are designed for the altitude hold autopilot based on [11]. Twenty-five autopilots are designed for the specified operating range. The original system includes the transfer function from elevator deflection to altitude, $\frac{h}{\delta_e}$, and the pitch attitude hold controller. The block diagram of the altitude hold controller is shown in Figure 4.9:

The zero of the PI controller is chosen at $s = -0.1$; the phase lead controller is designed in the same way given in the velocity controller. For example, when the aircraft is trimmed at $M = 0.45, h = 10kft, \Gamma = 0^\circ$, and $cg = 0.3\bar{c}$ and the transfer
function from elevator deflection to altitude, $\frac{h}{\delta_e}$, is:

$$h = \frac{4.44 \times 10^{-16}(s + 1.6803 \times 10^{15})(s - 10.388)(s + 5.5327)}{(s + 0.86465 \pm 1.2570i)(s + 0.0051 \pm 0.08477i)}$$

The pitch attitude hold controller designed at the same operating point is used to provide the inner-loop of the design. The transfer function from $\theta_c$ to $h$ is calculated based on block diagram reduction techniques as:

$$h = \frac{-6.4744 \times 10^{-14}(s + 4.2 \times 10^{14})(s - 10.388)(s + 5.5327)}{s(s + 3.8019 \pm 8.1755i)(s + 3.4621)(s + 0.65538)}$$

The phase lead compensator is designed based on this transfer function for the required phase margin $\Phi_m = 55^\circ$. The frequency, $\omega_1$ is chosen at $\omega_1 = 6 \text{rad/s}$. The coefficients of the compensator are obtained as $a_1 = 1.6482, b_1 = 0.10207$. Hence, the transfer function of the altitude hold autopilot for $M^\circ = 0.45, h^\circ = 10 \text{ft}$ is:

$$G_c(s) = (1 + \frac{0.1}{s}) \frac{1.6482s + 1}{0.10207s + 1}.$$
Glide Slope Autopilot

The glide slope autopilot controls the aircraft tracking a reference flight path angle. When the glide slope autopilot is engaged, the aircraft will descend at a constant path angle and constant speed. This autopilot is designed based on [1]. Figure 4.11 is the block diagram of the glide slope autopilot. The glide slope autopilot includes a PI controller and a phase lead compensator in the outer-loop and the pitch attitude hold autopilot embedded in the inner-loop. A typical reference flight path angle for the

![Figure 4.10: Step response of the system with altitude hold autopilot.](image)

![Figure 4.11: Block diagram of glide slope autopilot.](image)
glide slope phase is from 2.5° to 3.5°. Twenty-five linear glide slope autopilots are designed in the range of altitude and Mach number given in Table 4.1.

Here we give an example to illustrate the glide slope autopilot design. The aircraft is trimmed at $V_t = 350\, ft/s$, $h = 1400\, ft$, $\Gamma_c = -2.5^\circ$, $cg = 0.3\, \text{ft}$. The transfer function from elevator deflection to flight path angle, $\frac{\Gamma}{\delta_e}$, and the transfer function from elevator deflection to pitch angle, $\frac{\theta}{\delta_e}$, which are gained from the linearized aircraft model at this trim condition are

$$\frac{\Gamma}{\delta_e} = \frac{0.082313(s - 8.851)(s + 4.3784)(s + 0.003429)}{(s + 8.483)(s + 0.57623)(s + 0.08754)(s - 0.053328)}$$

$$\frac{\theta}{\delta_e} = \frac{-4.7168(s + 0.66547)(s + 0.023478)}{(s + 0.8483)(s + 0.57623)(s + 0.08754)(s - 0.053328)}$$

The pitch attitude hold autopilot is designed in the way given before based on the transfer function $\frac{\theta}{\delta_e}$. If the integral gain is chosen as $k_i = 0.01$, the proportional gain is obtained as $k_p = 5.22$ and the derivative gain is $k_d = 2.0845$. The transfer function $\frac{\Gamma}{\theta_c}$ can be derived by the block diagram reduction techniques as:

$$\frac{\Gamma}{\theta_c} = \frac{-4.3003(s + 4.3784)(s - 8.8510)}{(s + 3.897 \pm 8.2374i)(s + 3.0896)(s + 0.5496)}$$

The phase lead controller is designed based on this transfer function and the coefficients, $a_1 = 7.2399, b_1 = 1.2648$ are calculated in the same way as in the altitude hold autopilot for the required phase margin, $\Phi_m = 55^\circ$. The zero of the PI controller is chosen at $s = -1$. The step response of the closed-loop system is shown in Figure 4.12.

**Flare Control Autopilot**

At an altitude of 30\, ft above the end of the runway, the automatic landing system starts to reduce the rate of descent of the aircraft, achieves the correct pitch attitude
for landing, and begins to reduce the airspeed. This causes the aircraft to reach a gentle but positive landing. A satisfactory performance requires tight control of aircraft altitude; And the aircraft needs to be controlled to track an exponential trajectory until touch down. Thus, we have the following equations and constraints for the exponential model that generate the flare command.

\[ h = h_0 e^{-\frac{t}{\tau}}, \quad \dot{h} = -\frac{h}{\tau} \]

The controlled variable is altitude. The time constant \( \tau = 2.67 \text{s} \) is obtained from the geometry of the flare path. The block diagram of flare control is shown in Figure 4.13:

The transfer function \( s + \frac{1}{\tau} \) in the block diagram is not implemented in the controller. The altitude derivative is an additional output of the s-function in the aircraft model. The autopilot is designed at trim condition of \( V_l = 250 \text{ft/s}, h = 29 \text{ft}, \Gamma = -2.5^\circ, \) and \( cg = 0.3 \bar{c} \) as follows. The transfer function from elevator deflection to

Figure 4.12: Step response of the system with glide slope autopilot.
The transfer function from elevator deflection to pitch angle is
\[
\theta(s) = \frac{-2.8046(s + 0.53846)(s + 0.033456)}{(s + 1.1729)(s + 1.7124 \pm 0.15628i)(s + 0.13125)}
\]

One PI controller and one phase lead controller are designed based on the transfer function \( G(s) = \frac{h}{\delta_e} \), which is derived based on the transfer function \( \frac{h}{\delta_e} \) and the pitch attitude hold autopilot. The zero of PI controller is chosen at \( s = -0.5 \), and the phase lead controller is designed for phase margin, \( \Phi_m = 55^\circ \) as before. The transfer function of the linear continuous-time flare control autopilot is
\[
G_c(s) = \frac{(s + 0.5)(1.8591s + 1)}{s(1.25s + 1)}
\]

Figure 4.14 shows the step response of the closed-loop system.

### 4.2.2 Sampled-Data Linear Autopilot Design

A typical sampled-data system includes a sampler and a data-reconstruction device. In this thesis, the ideal sampler and the zero-order hold are used for data...
Figure 4.14: The step response of system with the flare control autopilot.

sampling and reconstruction. Sample period $T$ is chosen at $T = 0.1s$ according to Shannon’s Theorem.

The sampled-data aircraft autopilot design deals with the characteristics of the system in the $z$-plane. However, many of the analysis and design techniques for a continuous-time system are based on the property in the $s$-plane, and the stability boundary is the imaginary axis. These techniques cannot be applied to the sampled-data system in the $z$-plane directly, since there the stability boundary is the unit circle.

The bilinear transformation defined as

$$w = \frac{2z - 1}{Tz + 1}$$

maps the unit circle of the $z$-plane into the imaginary axis of the $w$-plane. The classical design techniques for a continuous-time system can be used in the $w$-plane. During the autopilots design, the aircraft model is discretized and mapped into the $w$-plane from the $z$-plane first; then the same design techniques in continuous-time autopilot
design are used to design the pitch attitude hold, glide path acquisition, altitude hold and flare control autopilots; finally the autopilots are mapped into the z-plane by the bilinear transformation.

Sampled-data pitch attitude hold, altitude hold, and glide slope autopilot have the same structure as in continuous-time. The flare control has an additional phase lag compensator to filter the high frequency noise brought by sampling and data-reconstruction [8]. The phase lead controller and PID controller in the w-plane are designed in the same way as in the s-plane. The frequency, $\omega_{w1}$, at which the phase margin occurs in the w-plane relates to the frequency, $\omega_1$ in s-plane as:

$$\omega_{w1} = \frac{2}{T} \tan\left(\frac{\omega_1 T}{2}\right)$$

All autopilots in the w-plane can be mapped into the z-plane by C2DM command.

4.2.3 Example of Simulation with Linear Continuous-Time and Sampled-Data Autopilot

In this section, some simulation examples are given for both continuous-time and sampled-data autopilots. The simulations are implemented in MATLAB SIMULINK. The same trim conditions are used for both continuous-time and sampled-data autopilot simulation.

The altitude hold autopilot has a similar simulation block diagram with the glide slope autopilot, except for the different controller, input and output signal, therefore, only the simulation block diagrams of altitude hold autopilot are given. Figure 4.15 shows the block diagram of the continuous-time autopilot simulation. Figure 4.16 is block diagram of the linear sampled-data autopilot simulation.
Figure 4.15: Simulation block diagram of linear continuous-time autopilot.

Figure 4.16: Simulation block diagram of linear sampled-data autopilot.
Figure 4.17 shows the linear simulation result of linear continuous-time altitude hold autopilot. Figure 4.18 is the linear simulation result of the linear sampled-data altitude hold autopilot. The aircraft is trimmed at $h = 5000 ft$, $M = 0.45$, $\Gamma = 0^\circ$ and $cg = 0.3\bar{c}$. The simulations show that the sampled-data altitude hold autopilot performs as good as the continuous-time altitude hold autopilot.

The linear simulation result of the continuous-time glide slope autopilot is shown in Figure 4.19. The linear simulation result of the sampled-data glide slope autopilot is shown in Figure 4.20. The linear sampled-data glide slope autopilot has the same simulation result as the linear continuous-time glide slope autopilot.

The flare control autopilot designed via the linearization approach covers the range of the flare control phase for the nonlinear F-16 aircraft, therefore, nonlinear simulation results are presented here for the flare control procedure. Figure 4.21 is the nonlinear simulation result of the continuous-time flare control autopilot. The aircraft is initialized at $29 ft$ altitude, $250 ft/s$ airspeed, and $-2.5^\circ$ flight path angle. Figure 4.22 is the nonlinear simulation result of the sampled-data flare control autopilot. From the simulation results, the linear sampled-data flare control autopilot has bigger transient than the continuous-time flare control autopilot. However, it still shows satisfactory performance.

4.2.4 Family of Linear Autopilot Design

Once linear autopilots have been designed to meet the design goals at each operating point, the family of linear autopilots are computed by INTERP2 function to interpolate the various parameters of the linear autopilots in the range of altitude and Mach number.
Figure 4.17: Simulation result of linear continuous-time altitude hold autopilot.

Figure 4.18: Simulation result of linear sampled-data altitude hold autopilot.
Figure 4.19: Simulation result of linear continuous-time glide slope autopilot.

Figure 4.20: Simulation result of linear sampled-data glide slope autopilot.
Figure 4.21: Simulation of continuous-time flare control autopilot.
Figure 4.22: Simulation of sampled-data flare control autopilot.
4.3 Gain Scheduled Autopilot Design

After the family of linear autopilots are established, we can use the methodology described in Chapter 2 to build the nonlinear gain scheduled autopilots, which are intended to linearize to the corresponding linear autopilot at each constant operating point. Three gain scheduled autopilots will be designed.

4.3.1 Pitch Attitude Hold Autopilot

The nonlinear pitch attitude hold autopilot is constructed based on the family of linear sampled-data pitch attitude hold autopilot designed before. The commanded elevator deflection is the output of the gain scheduled controller. The commanded pitch angle is the reference input signal. The regulated variable is the tracking error $z = \theta_c - \theta$. The pitch rate is the additional output of the plant. The family of linear discrete-time PI controllers has the form

$$x_{I\delta}[k + 1] = x_{I\delta}[k] + z_{\delta}[k]$$

$$\delta_{e\delta}[k] = -k_i(\Theta)x_{I\delta}[k] - k_p(\Theta)z_{\delta}[k] + k_d(\Theta)Q_{\delta}[k]$$

where $x_{I\delta}[k] = x_I[k] - x^o(\Theta)$, and $z_{\delta}[k] = \theta_c[k] - \theta[k]$. The gains $k_i$, $k_p$ and $k_d$ are functions of scheduling variable $\Theta = [M, h]$. The parameterized transfer function from $(z_{\delta}, q_{\delta})$ to $\delta_{e\delta}$ associated with this state equations is:

$$\begin{bmatrix} -k_i(\Theta) & -k_p(\Theta) & k_d(\Theta) \end{bmatrix} \begin{bmatrix} \frac{1}{z-1}I & 0 \\ I & 0 \\ 0 & I \end{bmatrix}$$
In a linear system, it is equivalent to

\[
\frac{z}{z - 1} \begin{bmatrix}
-k_i(\Theta) & -k_p(\Theta) & k_d(\Theta)
\end{bmatrix}
\begin{bmatrix}
\frac{1}{z} & 0 & 0 \\
\frac{1}{z} & 0 & 0 \\
0 & \frac{1}{z} & 0
\end{bmatrix}
\] (4.1)

Defining

\[
\rho[k] = z[k - 1] \\
\xi[k] = z[k] - z[k - 1] \\
\eta[k] = Q[k] - Q[k - 1]
\] (4.2)

it is clear that \(\rho^\circ(\Theta) = 0\), \(\xi^\circ(\Theta) = 0\), and \(\eta^\circ(\Theta) = 0\) at the constant operating point. Therefore, \(\rho_\delta[k] = \rho[k]\), \(\xi_\delta[k] = \xi[k]\), and \(\eta_\delta[k] = \eta[k]\). Now, consider a linear sampled-data controller which has form:

\[
x_{\delta^\delta}[k + 1] = x_{\delta^\delta}[k] - k_i(\Theta)\rho_\delta[k] - k_p(\Theta)\xi_\delta[k] + k_d(\Theta)\eta_\delta[k]
\]

\[
\delta_{\delta^\delta}[k] = x_{\delta^\delta}[k] - k_i(\Theta)\rho_\delta[k] - k_p(\Theta)\xi_\delta[k] + k_d(\Theta)\eta_\delta[k]
\] (4.3)

The transfer function from \((\rho_\delta, \xi_\delta, \eta_\delta)\) to \(\delta_{\delta^\delta}\) is given as

\[
\frac{z}{z - 1} \begin{bmatrix}
-k_i(\Theta) & -k_p(\Theta) & k_d(\Theta)
\end{bmatrix}
\]

which together with (4.2) has the same transfer function from \((z_\delta, Q_\delta)\) to \(\delta_{\delta^\delta}\) with (4.1). The existence condition of a gain scheduled controller satisfies Requirement 1 given in Chapter 2 is

\[
\frac{\partial x^\circ_\delta(\Theta)}{\partial \Theta} = \frac{\partial \delta^\circ_\delta(\Theta)}{\partial \Theta}
\]
which satisfied by taking $x_7^p(\Theta) = \delta_c^p(\Theta)$. A sampled-data gain scheduled controller that meets Requirement 1 is

\[
x_I[k + 1] = x_I[k] - k_i(\Theta[k])\rho[k] - k_p(\Theta[k])\xi[k] + k_d(\Theta[k])\eta[k]
\]

\[
\delta_c[k] = x_I[k] - k_i(\Theta[k])\rho[k] - k_p(\Theta[k])\xi[k] + k_d(\Theta[k])\eta[k]
\]  \hspace{1cm} (4.4)

where $\Theta[k] = \Theta(M[k], h[k])$.

### 4.3.2 Altitude Hold Autopilot

From Section 4.2, altitude, $h$ is the control variable of the linear altitude hold controller. The command altitude, $h_c$ is the reference input. Tracking error $z = h_c - h$ is the regulated signal. Altitude and Mach number are the scheduling variables. Pitch angle and pitch rate are additional system outputs. The controller has three parts.

**Inner-loop PI controller:**

\[
x_{1\delta}[k + 1] = x_{1\delta}[k] + \theta_c[k] - \theta[k]
\]

\[
\delta_{e\delta}[k] = -k_{p1}(\Theta)(\theta_c[k] - \theta[k]) - k_{i1}(\Theta)x_{\delta}[k] + k_d(\Theta)Q[k]
\]

**Outer-loop PI controller:**

\[
x_{2\delta}[k + 1] = x_{2\delta}[k] + z[k]
\]

\[
u[k] = k_i x_{2\delta}[k] + k_p z[k]
\]

**Outer-loop phase lead controller:**

\[
x_{3\delta}[k + 1] = a(\Theta)x_{3\delta}[k] + b(\Theta)u_\delta[k]
\]
\[ \theta_c[k] = c(\Theta)x_{3\delta}[k] + d(\Theta)u_\delta[k] \]

where \(k_p, k_d, a, b, c,\) and \(d\) are functions of \(\Theta = [M, h]\). The family of linear altitude hold autopilots has the form

\[
x_{C\delta}[k + 1] = A_{11}(\Theta)x_{C\delta}[k] + A_{12}(\Theta)x_{I\delta}[k] + B_{10}(\Theta)z_\delta[k] + B_{11}(\Theta)y_\delta[k]
\]

\[
x_{I\delta}[k + 1] = x_{I\delta}[k] + z_\delta[k]
\]

\[\delta_{e\delta}[k] = C_1(\Theta)x_{C\delta}[k] + C_2(\Theta)x_{I\delta}[k] + D_0(\Theta)z_\delta[k] + D_1(\Theta)y_\delta[k]\]

where

\[ A_{11}(\Theta) = \begin{bmatrix} a(\Theta) & 0 \\ c(\Theta) & 1 \end{bmatrix}, \quad A_{12}(\Theta) = \begin{bmatrix} b(\Theta)k_i \\ d(\Theta)k_i \end{bmatrix} \]

\[ B_{10}(\Theta) = \begin{bmatrix} b(\Theta)k_p \\ d(\Theta)k_p \end{bmatrix}, \quad B_{11}(\Theta) = \begin{bmatrix} 0 & 0 \\ -1 & 0 \\ 0 & 0 \end{bmatrix} \]

\[ C_1(\Theta) = \begin{bmatrix} -k_{p1}(\Theta) & -k_i(\Theta) \end{bmatrix}, \quad C_2(\Theta) = -k_{p1}(\Theta)k_id(\Theta) \]

\[ D_0(\Theta) = -k_pk_{p1}(\Theta)d(\Theta), \quad D_1(\Theta) = \begin{bmatrix} k_{p1}(\Theta) & k_d(\Theta) \end{bmatrix} \]

\[ y_\delta[k] = \begin{bmatrix} \theta_\delta[k] \\ Q_\delta[k] \end{bmatrix} \]

The transfer function from \((z, y_\delta)\) to \(\delta_{e\delta}\) is

\[ [C_1(\Theta)[zI - A_{11}(\Theta)]^{-1}[A_{12}(\Theta)B_{10}(\Theta)B_{11}(\Theta)] + [C_2(\Theta)D_0(\Theta)D_1(\Theta)]] \begin{bmatrix} \frac{1}{z-1}I & 0 \\ I & 0 \\ 0 & I \end{bmatrix} \]

We use the same definitions of \(\rho\) and \(\xi\) as the pitch attitude hold controller and define \(\eta[k] = y[k] - y[k - 1]\). The family of linear discrete-time controllers has the
The existence condition of the gain scheduled controller to meet Requirement 1 has the form

\[
\begin{align*}
x_{C\delta}[k+1] &= A_{11}(\Theta)x_{C\delta}[k] + A_{12}(\Theta)\rho[k] + B_{1\gamma}(\Theta)\xi[k] + B_{11}(\Theta)\eta[k] \\
x_{I\delta}[k+1] &= C_{1}(\Theta)x_{C\delta}[k] + C_{2}(\Theta)\rho[k] + D_{0}(\Theta)\xi[k] + D_{1}(\Theta)\eta[k] \\
\delta_c[k] &= C_{1}(\Theta)x_{C\delta}[k] + C_{2}(\Theta)\rho[k] + D_{0}(\Theta)\xi[k] + D_{1}(\Theta)\eta[k]
\end{align*}
\]

(4.5)

which is satisfied by taking \(x_C^0(\Theta) = 0\) and \(x_I^0(\Theta) = \delta_c^0(\Theta)\). Here \(\Theta[k] = \Theta(M[k], h[k])\).

A gain scheduled controller which meets Requirement 1 is given by

\[
\begin{align*}
x_C[k+1] &= A_{11}(\Theta[k])x_C[k] + A_{12}(\Theta[k])\rho[k] + B_{1\gamma}(\Theta[k])\xi[k] + B_{11}(\Theta[k])\eta[k] \\
x_I[k+1] &= C_{1}(\Theta[k])x_C[k] + C_{2}(\Theta[k])\rho[k] + D_{0}(\Theta[k])\xi[k] + D_{1}(\Theta[k])\eta[k] \\
\delta_c[k] &= C_{1}(\Theta[k])x_C[k] + C_{2}(\Theta[k])\rho[k] + D_{0}(\Theta[k])\xi[k] + D_{1}(\Theta[k])\eta[k]
\end{align*}
\]

(4.6)

### 4.3.3 Glide Slope Autopilot

The linear glide slope controller has the same structure as the altitude hold controller, except that the reference input is the flight path angle and the regulated signal is \(z = \Gamma_c - \Gamma\). Therefore, the nonlinear gain scheduled controller design is also the same as the altitude hold controller.
4.4 Comparison of Simulation Results of Linear Autopilot and Nonlinear Gain Scheduled Autopilot

In this section, the nonlinear autopilots obtained in Section 4.3 is evaluated by simulation and compared to the linear autopilot for the same trim condition and command input. Figure 4.23 is the SIMULINK block diagram of the gain scheduled pitch attitude hold controller. Figure 4.24 is the SIMULINK block diagram of the linear attitude hold controller.

Figure 4.25 is the step response of the gain scheduled pitch attitude hold controller for the commanded pitch angle $\theta = 3.59^\circ$ (operating point) and $\theta = 30^\circ$. The step response of linear pitch attitude hold autopilot is shown in Figure 4.26 for the same commanded pitch angles. The aircraft is trimmed at 500 ft/s airspeed and 10000 ft altitude.

From the step response of the linear pitch attitude hold autopilot Figure 4.26 and nonlinear gain scheduled pitch attitude hold controller Figure 4.25, it is clear that both autopilots perform satisfactorily at the operating point, but the linear autopilot cannot hold the performance when the commanded pitch angle is far from the operating point. This is because the dynamic behavior of nonlinear system changes with the operating range. The gain scheduled autopilot performs its job well in a larger operating range.
Figure 4.23: Block diagram of gain scheduled pitch attitude hold autopilot.

\[
y(n) = Cx(n) + Du(n) \\
x(n+1) = Ax(n) + Bu(n)
\]

Figure 4.24: Block diagram of linear pitch attitude hold autopilot
Figure 4.25: Gain scheduled pitch attitude hold autopilot step response.

Figure 4.26: Linear pitch attitude hold autopilot step response.
Figure 4.27 is the SIMULINK block diagram of the gain scheduled altitude hold autopilot. Figure 4.28 is the SIMULINK block diagram of the linear altitude hold autopilot.

The aircraft is trimmed at $V_i = 500 ft/s$ and $h = 0 ft$ for both linear and nonlinear altitude hold autopilot simulations. The linear altitude hold autopilot is designed at $h = 0 ft$ and $V_i = 750 ft/s$. The commanded altitude is the function

$$h = \begin{cases} 40t & \text{if } h < 10000 ft \\ 10000 & \text{otherwise} \end{cases}$$

Figure 4.29 is the simulation result of the nonlinear gain scheduled altitude hold autopilot. Figure 4.30 is the simulation result of the linear altitude hold autopilot with the same trim condition and input signal. From the simulation results, the gain scheduled altitude hold controller shows very good performance in tracking the command input. But the linear autopilot can not hold the aircraft at 10kft altitude and the system goes to unstable. This is because the linear autopilot designed for one operating point can not control the aircraft at other different operating points.

The gain scheduled glide slope autopilot simulation block diagram has the same structure as Figure 4.27, except the different input and S-function are used. The linear glide slope autopilot simulation has the same structure as Figure 4.28. The aircraft is initialized at $h = 35k ft$, $V_i = 350 ft/s$ for glide slope autopilot simulation.
Figure 4.27: Simulation block diagram of gain scheduled altitude hold autopilot.

Figure 4.28: Simulink block diagram of linear altitude hold autopilot.
Figure 4.29: Simulation result of gain scheduled altitude hold autopilot.

Figure 4.30: Simulation result of linear altitude hold autopilot.
Figure 4.31 is the simulation result of the gain scheduled glide slope controller and Figure 4.32 is the simulation result of the linear glide slope autopilot. The linear autopilot designed at $h = 40kft$ and $V_t = 500ft/s$ makes the system unstable for the initial condition of the simulation. The gain scheduled controller achieve a good performance over the desired work range.

Figure 4.31: Simulation result of gain scheduled glide slope autopilot.
Figure 4.32: Simulation result of linear glide slope autopilot.
Chapter 5

Conclusions

In this thesis, autopilots for an F-16 aircraft were designed using a sampled-data gain scheduling technique. We introduced the theory and framework of the sampled-data gain scheduled technique. We also described the F-16 aircraft dynamic model and gave necessary explanations. The autopilots are designed according to the procedure listed in Chapter 2. We used SIMULINK to implement the systems and simulate their dynamic performance.

The first step in the gain scheduling technique is to compute the family of constant operating points and linearize the system at several equilibrium points. The second step is to design a family of linear controllers at each operating point that performs satisfactorily when the system is operating near the respective operating points. After exploring the frequency characteristic of the system, we designed several kinds of controllers to reach the design objective by the classical linear design techniques. Simulation showed that we achieved satisfactory performance around the operating points.

Gain scheduling can extend the validity of a controller’s operation to a set of operating points. We designed the sampled-data gain scheduled controller from the
family of linear controllers. Simulating results show that satisfactory performances are obtained in the required range.

Since the model used (F-16 aircraft model) is a general model, we can construct the models of many other aircraft in the same structure. The design method we used gave a framework for autopilot designs. This approach can also be extended to design autopilots for other aircrafts.

For further research work, lateral-direction autopilots for the F-16 aircraft should be designed to obtain six degree of freedom autopilots. More modern control design techniques will be involved in the full six degree of freedom linear autopilot designs. A wind gust model needs to be considered as a perturbation source of the nonlinear F-16 aircraft model. Finally, the experience we gained from the longitudinal autopilots design for the F-16 aircraft will be applied to the autopilot design for the Unmanned Aerial Vehicle project in the Avionics Engineering Center.
Bibliography


