Case Study: Development of a Baseline Controller for Automatic Landing of an F-16 Aircraft Using Linear Matrix Inequalities (LMIs)

Danbing Seto
Enrique Ferreira
Theodore F. Marz

May 2000

TECHNICAL REPORT
CMU/SEI-99-TR-020
ESC-TR-99-020

Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Case Study: Development of a Baseline Controller for Automatic Landing of an F-16 Aircraft Using Linear Matrix Inequalities (LMIs)

CMU/SEI-99-TR-020
ESC-TR-99-020

Danbing Seto
Enrique Ferreira
Theodore F. Marz

May 2000

Dependable Systems Upgrade

This report was prepared for the

SEI Joint Program Office
HQ ESC/DIB
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

Norton L Compton

Norton L. Compton, Lt Col., USAF
SEI Joint Program Office

This work is sponsored by the SEI FFRDC primary sponsor, the Defense Advanced Research Projects Agency, and the Air Force Research Laboratory. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

List of Tables

Abstract

In this report, we present preliminary results on the design of the baseline controller for an F-16 aircraft automatic landing system using linear matrix inequalities (LMI)-based approaches. We start with a general study of aircraft control and dynamics to gain knowledge of the structure of an aircraft dynamic model and its inner loop control system. We then identify a linear model along the glide path for the inner loop control system in the simulator. With this linear model, the control objective is to solve a stabilization problem—stabilizing the aircraft along the glide path using linear state feedback controls. Expressing the stability criterion and the constraints in LMIs, we cast the stabilization problem as an optimization problem. Using the SDPSOL[1] software package developed by Wu and Boyd, we solve this optimization problem for the control gain and stability region, which completes the controller design [Wu 96].

[1] SDPSOL is a parser/solver for semidefinite programming and determinant maximization problems with matrix structure.

1 Introduction

As part of the Evolutionary Design of Complex Software (EDSC) Incremental Software Evolution for Real-Time Systems (INSERT) Project [INSERT 00], the Simplex[TM] architecture has been implemented in a legacy F-16 manned virtual desktop simulation to demonstrate its capability of supporting the safe insertion of new technology and the safe upgrading of current functionality in safety-critical real-time systems. Using the Simplex architecture, new capabilities can be rapidly installed in the aircraft to allow the fighter to perform with enhanced maneuverability. By tolerating possible faults that such installations may bring to the system, the Simplex architecture guarantees continuous operation of the aircraft even if the newly configured aircraft has not been fully tested in its operating environment.

Fault tolerance in the Simplex architecture is based on the concept of analytic redundancy. The Simplex architecture consists of multiple “replacement units” where different technologies can be implemented and where at least one of them will be highly reliable in the sense that it can serve as the fallback option when others fail to function properly. (For more information on the Simplex architecture, refer to [Feiler 99], [Pollak 98], and [Altman 99].) Where control of the aircraft is concerned, there must be a fallback controller in the system to support the upgraded control. This controller will be referred as the *baseline controller*. Since the baseline controller takes over control upon detection of a fault caused by the upgraded controller, it is important that the aircraft be in a state from which the baseline controller is capable of maintaining the aircraft’s stability. Therefore, in addition to the design of the control algorithm for the baseline controller, an operational region of the controller in the state space of the aircraft needs to be identified. An *operational region* of a controller means that, from any state inside the region, the controller achieves the control objectives without violating any constraints imposed upon the aircraft. Furthermore, such a region can serve as a switching criterion for fault detection.

In this report, we address the issues of control algorithm design and operational region identification for a baseline controller. In particular, we concentrate on the landing operation. We present the development and preliminary results of the design of a baseline controller for automatic landing of an F-16 using modern control techniques and linear matrix inequality (LMI)-based approaches. The control objective is to maintain the aircraft flying along the glide path, which is equivalent to a problem of stabilization at an aircraft steady state. A linear model is identified using data of typical landing maneuvers from the legacy F-16 simulation. LMIs are used to generate a feedback controller taking into account actuator constraints and flight specifications. As a result of the stabilization problem, a stability region of the

[TM] Simplex is a trademark of Carnegie Mellon University.

closed-loop system is identified to serve as the operational region of the baseline controller, a stabilizer.

The work reported here was carried out by using data from the F-16 Block 50 model simulation, referred as *the real-time simulator* (or *the simulator*) in this report, in parallel with a study of general aircraft control introduced by Stevens and Lewis and the F-16 modeling and simulation tools provided by Stevens [Stevens 92]. A software package implementing software illustrations in *Aircraft Control and Simulation* by B. Stevens was obtained from the author. This package contains a nonlinear model of an F-16, with which a linearized model can be derived and nonlinear simulation can be performed. To distinguish from the real-time simulator and the model within, we call this simulation a *book simulator* and the nonlinear model a *book model*.

Understanding the book model of the F-16 and inner loop control design as described by Stevens provided us a structure for the inner loop F-16 control system in the simulator, which is not available to us [Stevens 92]. With this structure, we were able to identify the control system in the simulator to which a control algorithm can be designed and implemented. Furthermore, with the book model, stability analysis, and LMI-based control design has been completed at an advanced stage.

This report is organized as the follows. Section 2 gives an introduction to aircraft dynamics and the notation used throughout the report. Section 3 presents the problem to be addressed. System identification issues are discussed in Section 4. Section 5 describes the design of control algorithms and derivation of stability regions. A summary of the work and main conclusions are presented in Section 6.

2 Dynamics and Control of an Aircraft

In this section, we present a review of aircraft dynamics and control described by Stevens [Stevens 92]. We first introduce the state variables and control inputs in a typical aircraft and various coordinate frames used to describe aircraft dynamics. Then we establish sets of equations of motion for an aircraft in different coordinate frames and flight conditions. Finally, we describe a software package for F-16 model trimming and linearization.

2.1 Aircraft State Variables and Control Inputs

As a rigid body moving in a three-dimensional space, an aircraft has a total of 6 degrees of freedom described by 12 state variables. These variables are divided into four groups:

1. three position variables [position of the aircraft center of gravity (cg)]
2. three linear velocity variables (translational velocity of the aircraft cg)
3. three attitude (orientation) variables
4. three angular velocity variables

Aircraft dynamics are normally controlled by four physical inputs: throttle, aileron, elevator, and rudder. The throttle controls the thrust to the aircraft, while the aileron, elevator, and rudder deflections generate aerodynamic forces. Figure 1 shows a sketch of an aircraft and its deflector components as well as the moments that they generate.

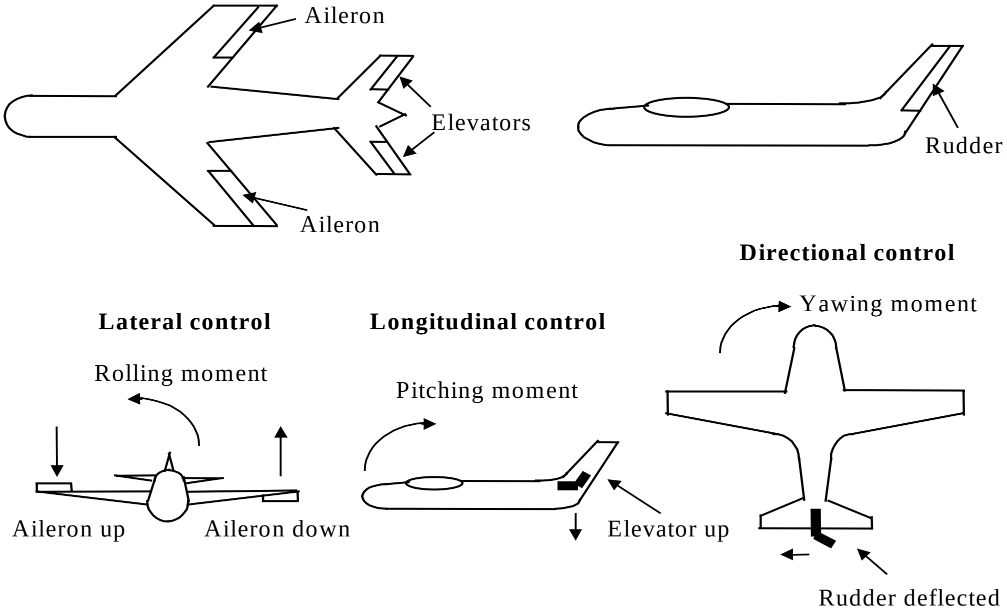


Figure 1: Deflection Actuators and Moments

2.2 Coordinate Frames

To describe the dynamics of an aircraft, we need to set up a coordinate frame. Depending on the flight conditions, different coordinate frames may be used to describe an aircraft's motion. The following summarizes some of the coordinate frames. Refer to Figure 3 for a picture of the relationship between the coordinate systems.

Earth-centered inertial (ECI) reference frame: centered at the origin of the Earth.

- *x*-axis along the Earth's spin axis, pointing to the North Pole
- *y*-axis perpendicular to *x*-*z* plane
- *z*-axis pointing to the 180° longitude point on the equator at $t = 0$

North-east-down (NED) frame: centered on the Earth's surface at the point vertically below the aircraft cg, with *x*-*y* plane tangent to the Earth's surface. It moves with the aircraft.

- *x*-axis pointing to the north
- *y*-axis pointing to the east
- *z*-axis normal to the Earth's surface and pointing inward

Aircraft-body coordinate (ABC) frames: refers to three coordinate frames, which are all centered at the aircraft cg with the conventional *right-handed* (forward, starboard, and down) set of axes. These frames are illustrated in Figure 2.

- **Body-fixed axes (ABC):** All axes are fixed on the body of the aircraft.
- **Stability axes (ABC-Stability):** Axes are obtained from the body-fixed axes by rotating about the *y*-axis for the angle of α, which is called the *angle of attack.* It is positive if the rotation about the body-fixed y-axis is negative.
- **Wind axes (ABC-Wind):** Axes are obtained from the stability axes by rotating about the *z*-axis for the angle of β, which is called the *side-slip angle.* It is positive if the rotation about the stability *z*-axis is positive.

The angles α and β are known as the *aerodynamic angles; these* are needed to specify the aerodynamic forces and moments.

Different coordinate frames will be used to simplify the description of the aircraft dynamics simple and to make them easier to understand. For example, to describe a flight around the Earth, the ECI frame is usually used, while flying on a flat earth is often described in the NED frame. Algebraic relations among these frames, namely the mapping from one frame to another, are given in Appendix A. Figure 3 shows the spatial relationship among the ECI frame, NED frame, and ABC frame.

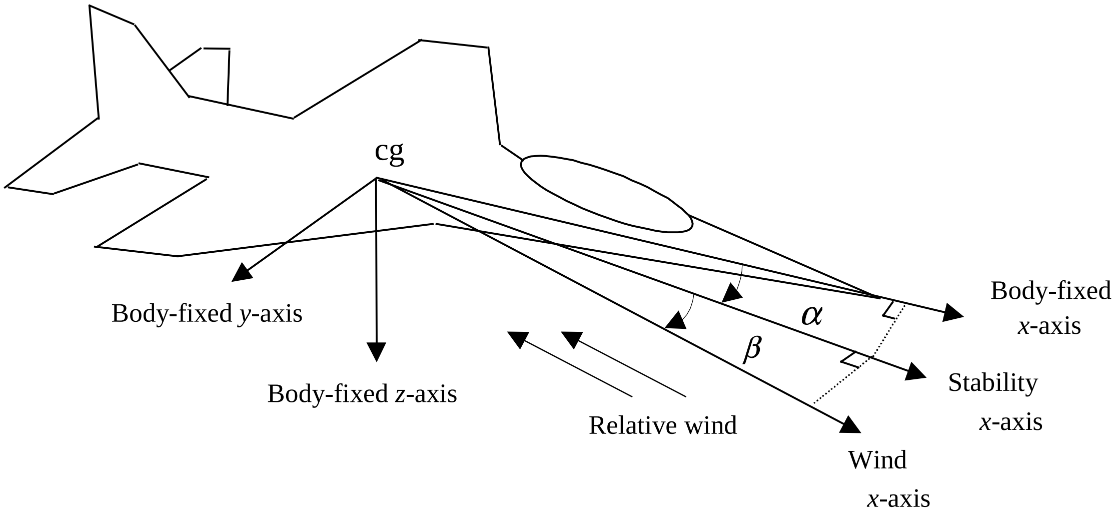


Figure 2: Aircraft Body Coordinate System

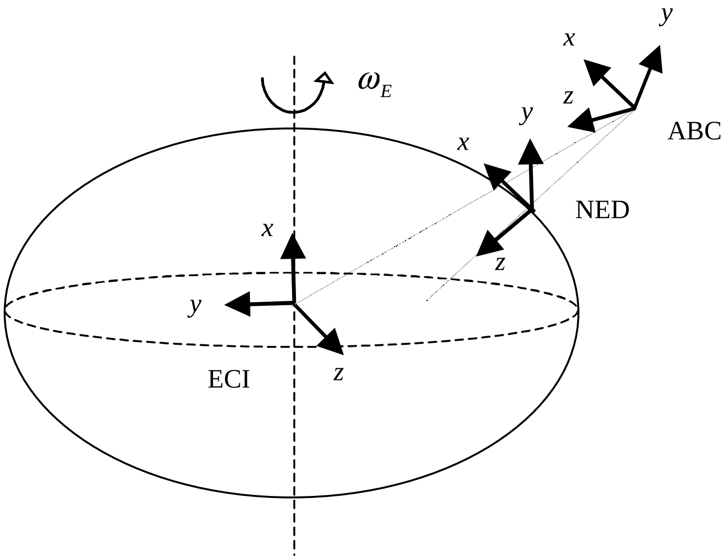


Figure 3: Relationship Among Different Coordinate Systems

2.3 Aircraft Modeling

A model of an aircraft is a mathematical description of the aircraft motion. It is given by a set of differential equations, called the *equations of motion*, which present the relations between the control inputs and the state variables. From the equations of motion, we hope to gain some physical intuitions that will help to structure the model in a real-time simulator.

The equations of motion are derived based on first principles. Detailed derivations and variable definitions are given in Appendix B. Since landing is the main concern in this report, we will consider the flat-earth model derived in Appendix B and rewrite the equations of motion as follows:

$$
\begin{aligned}
\dot{\mathbf{p}}_{\text{NED}} &= R_{AN}^T \mathbf{v}_B \\
\dot{\mathbf{v}}_B &= -\Omega_B \mathbf{v}_B + R_{AN}\mathbf{g}_0 + \frac{\mathbf{F}_B}{m} \\
\dot{\boldsymbol{\omega}}_B &= -J^{-1}\Omega_B J \boldsymbol{\omega}_B + J^{-1}\mathbf{T}_B \\
\dot{\boldsymbol{\Phi}} &= \Psi(\boldsymbol{\Phi})\boldsymbol{\omega}_B
\end{aligned}
$$

where $\mathbf{g}_0$ is the gravity expressed in the NED frame, $\mathbf{F}_B$ and $\mathbf{T}_B$ are the resultant aerodynamic and thrust forces and the moments they generate expressed in the ABC frame, and

$\mathbf{P}_{\text{NED}}$: the position of aircraft cg expressed in the NED frame,
$\mathbf{v}_B$: relative velocity of aircraft cg w.r.t air mass expressed in ABC frame,
$\boldsymbol{\omega}_B$: absolute angular velocity of ABC frame expressed in ABC frame,
$\boldsymbol{\Phi}$: Euler angle between ABC and NED frames.

For the sake of linearization, we modified the above equations by expressing the relative velocity of aircraft cg and the absolute angular velocity of the ABC frame in the ABC-Wind frame. Since $\mathbf{v}_w = R_{WB}\mathbf{v}_B, \boldsymbol{\omega}_w = R_{WB}\boldsymbol{\omega}_B$ and $\mathbf{F}_w = R_{WB}\mathbf{F}_B$, we have the revised equations of motion as

$$
\begin{aligned}
&\textit{Navigation Equations}: && \dot{\mathbf{p}}_{\text{NED}} = R_{AN}^T R_{WB}^T \mathbf{v}_W \\
&\textit{Force Equations}: && \dot{\mathbf{v}}_w = -\Omega_{wb}^w \mathbf{v}_w - \Omega_w \mathbf{v}_w + R_{WB}R_{AN}\mathbf{g}_0 + \mathbf{F}_w / m \\
&\textit{Kinematics Equations}: && \dot{\boldsymbol{\Phi}} = \Psi(\boldsymbol{\Phi})R_{WB}^T\boldsymbol{\omega}_w \\
&\textit{Moment Equations}: && \dot{\boldsymbol{\omega}}_w = -\Omega_{wb}^w\boldsymbol{\omega}_w - J_w^{-1}(\boldsymbol{\omega}_w \times J_w\boldsymbol{\omega}_w) + J_w^{-1}\mathbf{T}_w
\end{aligned}
\qquad \textbf{(1)}
$$

where

$$
\Omega_{wb}^w = \begin{bmatrix} 0 & -\dot{\beta} & -\dot{\alpha}\cos\beta \\ \dot{\beta} & 0 & \dot{\alpha}\sin\beta \\ \dot{\alpha}\cos\beta & -\dot{\alpha}\sin\beta & 0 \end{bmatrix}, \quad \Omega_w = \begin{bmatrix} 0 & -\omega_w^z & \omega_w^y \\ \omega_w^z & 0 & -\omega_w^x \\ -\omega_w^y & \omega_w^x & 0 \end{bmatrix}
$$

Noticing that

$$\mathbf{v}_w = [v_T, 0, 0]^T \quad \text{with} \quad v_T \text{ the true air speed}$$

and

$$\dot{\mathbf{v}}_w + \Omega_{wb}^w \mathbf{v}_w = [\dot{v}_T, \ \ \dot{\beta} v_T, \ \ \dot{\alpha} v_T \cos\beta]^T ,$$

we can write the equations of motion with the state variables as follows:

$$x = [p_N ,\ p_E ,\ h,\ v_T ,\ \alpha,\ \beta,\ \phi,\ \theta,\ \varphi,\ P,\ Q,\ R]^T$$

by letting $\mathbf{p}_{\text{NED}} = [p_N , p_E , h]^T$, $\mathbf{v}_w = [v_T , 0, 0]^T$, $\mathbf{\Phi} = [\phi, \theta, \varphi]^T$, and $\boldsymbol{\omega}_w = [P, Q, R]^T$. Furthermore, we specify the force and torque in terms of aerodynamic and thrust forces and the moments they generate as

$$F_w = \begin{bmatrix} -D \\ Y \\ -L \end{bmatrix} + R_{WB} \begin{bmatrix} F_T \\ 0 \\ 0 \end{bmatrix}, \quad T_w = R_{WB} \begin{bmatrix} \overline{L} \\ M \\ N \end{bmatrix} \qquad \textbf{(2)}$$

where F_T is the thrust,

$D = \overline{q}SC_D$ – Drag	$\overline{L} = \overline{q}SbC_l$ – Rolling moment
$L = \overline{q}SC_L$ – Lift	$M = \overline{q}S\overline{c}C_M$ – Pitching moment
$Y = \overline{q}SC_Y$ – Side force	$N = \overline{q}SbC_N$ – Yawing moment

and

C_D – Drag coefficient	C_l – Rolling moment coefficient
C_L – Lift coefficient	C_M – Pitching moment coefficient
C_Y – Side - force coefficient	C_N – Yawing moment coefficient
$\overline{q}$ – free stream dynamic presure	S – wing reference area
$\overline{c}$ – wing mean geometric chord	b – wing span

The forces and moments in Equation 2 are the variables that affect the dynamics of the system, but they are not the control variables. In fact, the force and moment coefficients C_*, with * being the subscripts defined in Equation 2, are functions of the control variables:

$$u = [thl,\ el,\ ail,\ rdr]^T \qquad \textbf{(3)}$$

where

thl–Throttle	*ail*–Aileron deflection
el–Elevator deflection	*rdr*–Rudder deflection

Equations 1-3 compose the nonlinear mode of the aircraft considered in this report. The aircraft dynamics are often defined by the last three sets of equations in Equation 1 (i.e., the force equations, the kinematic equations, and the moment equations).

The aircraft motion can be further characterized as a combination of *longitudinal motion* and *lateral-directional motion*. While the former describes the pitching and translational part of the motion with variables α, θ, v_T, Q, the latter is about rolling, sideslipping, and yawing with variables $\beta, \phi, \varphi, P, R$. When an aircraft flies level, $\phi = P = 0$, and non-sideslipping (coordinated flight), $\beta = R = 0$, the longitudinal motion can be described by a set of decoupled equations

$$\begin{cases} m\dot{v}_T = F_T \cos\alpha - D - mg_0 \sin(\theta - \alpha) \\ m\dot{\alpha}v_T = -F_T \sin\alpha - L + mv_T\omega_w^y + mg_0 \cos(\theta - \alpha) \\ \dot{\theta} = \omega_w^y \\ J_y\dot{\omega}_w^y = M \end{cases}$$

Linearization of the nonlinear model of an aircraft can be derived with respect to a flight condition (i.e., a description of a particular flight situation). A commonly considered flight condition is level, coordinated flight, which is specified by

$$\beta,\ \phi,\ P, Q, R = 0,\ \ \dot{x} \equiv 0,\ \ u \equiv 0$$

This flight condition implies the following steady state:

$$\begin{aligned} x_s &= [v_T,\ \alpha,\ \theta,\ \phi, \theta, \psi, P, Q, R]|_{\beta,\phi,P,Q,R,\dot{\mathbf{x}}\equiv 0} = [v_{T_s},\ \ \alpha_s,\ 0,\ 0,\ \theta_s,\ \psi_s,\ 0,\ 0,\ 0] \\ u_s &= [thl, el, ail, rdr]|_{\beta,\phi,P,Q,R,\dot{\mathbf{x}}\equiv 0} = [thl_s,\ el_s,\ ail_s,\ rdr_s] \end{aligned} \tag{4}$$

Define $\delta x = x - x_s$, $\delta u = u - u_s$, then the linearized system at the steady state in Equation 4 is given by

$$E\delta\dot{x} = A\delta x + B\delta u$$

where *E*, *A*, and *B* are constant matrices which can be found in Section 2.5 of the book by Stevens [Stevens 92].

The objective of analytically deriving the linearized model is to understand the structure of the aircraft dynamics. In particular, we would like to see what elements in matrices *E*, *A*, and *B* are intrinsically zero or invariant as the type of aircraft or the flight condition change. Those elements will be treated as the fixed parameters in system identification. As can be seen from above, analytically derived linearization could be tedious, as the process has to be repeated when the flight conditions are changed. This motivates the development of numerical linearization, which is discussed in Section 2.4.

2.4 Modeling Tools

Stevens describes how insight into the dynamics of an F-16 fighter was gained by experimenting with a standard model described, and the simulation experiment runs with the main Fortran code provided by Stevens [Stevens 92]. Two different sets of programs were used, one to find a specific flying condition (and then the steady state) and the other to do a nonlinear simulation of landing. Appendix C contains a list of the files and subroutines for the original and modified source codes.

The core of both programs is a subroutine that contains a nonlinear model of the plane. To find the steady state, an optimization routine is needed. The simulation program uses an ordinary differential equation solver (Runge-Kutta method).

The aircraft system is described by

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$$

$$\mathbf{x} = [\ Vt, \alpha, \beta, \phi, \theta, \psi, p, q, r, x, y, z, pow\ ]$$

$$\mathbf{u} = [\ thl, el, ail, rdr]$$

Table 1 displays all the state variables and inputs with the corresponding units as they are used in the source code. This is also the order in which they appeared in the subroutine.

Vt (ft/s)	True air speed
α (rad)	Angle of attack
β (rad)	Side-slip angle
ϕ, θ, ψ (rad)	Roll, pitch, and yaw angles
p,q,r (rad/s)	Roll, pitch, and yaw angle rates
x,y,z (ft)	XYZ coordinates in flat-earth system frame
el (deg)	Elevator position (actuator)
ail (rad)	Aileron position (actuator)
rdr (rad)	Rudder position (actuator)
thl (unit)	Throttle position (actuator)
pow (unit)	Power/thrust

Table 1: Aircraft Nomenclature

The physical model must be enhanced with actuator dynamics and constraints and the use of the typical inputs that the pilot has available (i.e., thl, the roll command input Pcom, and the pitch command Qcom). This input transformation translates in an augmented system

$$\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u})$$

$\mathbf{x}$ = [Vt, α, β, ϕ, θ, ψ, p, q, r, x, y, z, pow, thls, els, ails, rdrs]

$\mathbf{u}$ = [thl, Pcom, Qcom],

with constraints on the actuator states thls, els, ails, and rdrs and their derivatives. It should be noticed that pitch and roll command inputs are actually indirect input commands. The aircraft is directly controlled by the position of the elevator, ailerons, and rudder flaps. Therefore, there is an implicit internal conversion system, perhaps a feedback controller, that translates pitch and roll commands into actuator commands. In Section 3, we show an example of this program.

3 Automatic Landing of an F-16 Aircraft

In this section, we are concerned with the last stages of the landing procedure (i.e., when the aircraft is in relatively good alignment with the runway). The automatic landing system should provide feedback control to allow the aircraft to follow a trajectory in the XYZ space.

The trajectory, called *glide-path*, is determined by the position and orientation of the runway and the flight-path angle γ_R. A typical automatic landing system uses a radio beam directed upward from the ground that can be detected by aircraft sensors to control the landing. Figure 4 shows a landing approach and some important characteristics.

The control commands to be used are the usual three: throttle, pitch, and roll command inputs. The throttle command is generally left open-loop and it is directly controlled by the pilot or by a separate subsystem. Pitch and roll commands usually have some level of feedback control to help the pilot. The pilot can modify the command actions using the stick.

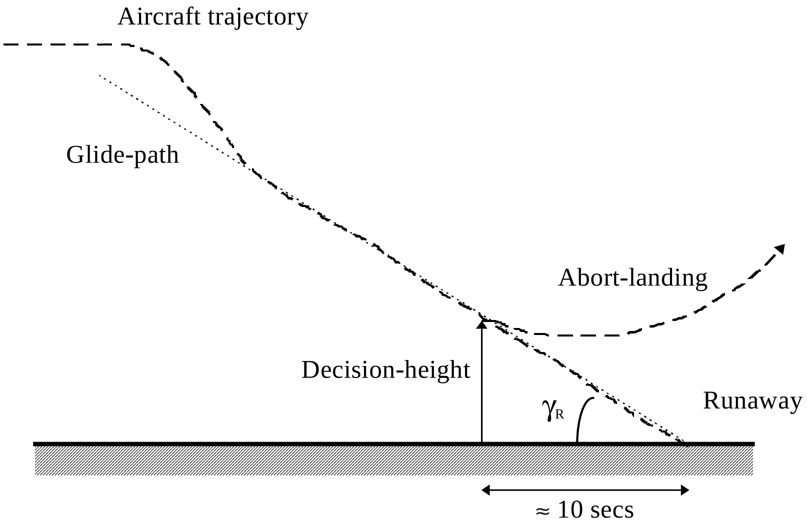


Figure 4: Landing Trajectoy

Landing specifications include constraints on the way the approach should be taken (i.e., a desired range of values for the aircraft state variables and flight-path). Table 2 shows typical values for a landing approach.

Flight-path angle	constant between 2.5 to 3.5 degrees
Airspeed	240-280 ft/s
Angle of attack	between 10-15 degrees

Table 2: General Landing Specifications

One of the most important features of an automatic landing system involves the decision of whether to land the aircraft according to how the approach has been going up to that time. A way to address this issue is by defining a position, taking into account the aircraft dynamics, when the decision to land or abort has to be made. This point is called *decision-height.* Tighter constraints on the state of the aircraft need to be met at *decision-height* to proceed with the landing. Table 3 specifies a typical set of values that are required at the decision-height point to continue and land for the state of an F-16 aircraft. Actually, the specification requires that these constraints be satisfied from the decision-height to the touchdown point.

Decision-height point	roughly 10 seconds before landing
Angle of attack (α)	between 10–15 degrees
Vertical deviation (d_v)	maximum 5 feet
Horizontal deviation (d_h)	maximum 15 feet
Roll error angle (ϕ)	Maximum 5 degrees
Pitch error angle (θ)	Maximum 5 degrees
Yaw error angle (ψ)	Maximum 5 degrees
Sink rate	Between 250 and 1000 ft/min

Table 3: Typical Range of Values for Aircraft Variables at Decision-Height Point

Rather than using the absolute position of the aircraft in the XYZ coordinate system, the error distance to the glide path is used to evaluate the approach and design the control system. Two scalar values are considered: the vertical and horizontal deviations d_v and d_h, and the vertical and horizontal projected distance of the aircraft with respect to the glide path. Angular magnitudes can be considered instead, but d_v and d_h give easier equations with which to work. Having set up this goal with respect to the decision-height point, the next step is to design a controller that brings the aircraft into that region. A similar goal is to assess the performance of a previous controller and predict when or where that controller would take us to a good landing. These problems are addressed in the next two sections.

4 Model Identification

The objective of this section is to identify a model for the F-16 simulator using the data collected while it is simulating landing. Rather than a black-box identification, it proves very useful to use the information from a first-principles model and standard control techniques to establish the best structure for the model to be identified. The overall aircraft control system can be illustrated as in Figure 5, where *thl, Pcom,* and *Qcom* are the control variables for which the control algorithms will be designed. Together with the inner loop control, these control variables generate control commands for the physical control surface *thl, el, ail,* and *rdr,* which in turn control the aircraft. The model that we need to identify consists of the aircraft dynamics, actuator dynamics and inner loop control. We will proceed in two steps. As studied in Stevens, the first step is to get the F-16 model by running the software provided by Stevens, and then the second step is to use the structure of the model obtained for the simulator and identify the unknown parameters [Stevens 92].

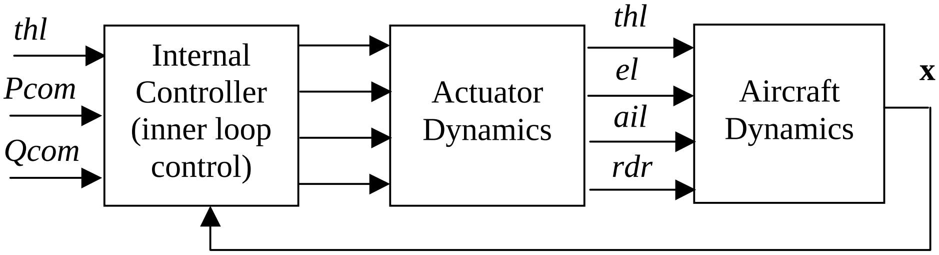


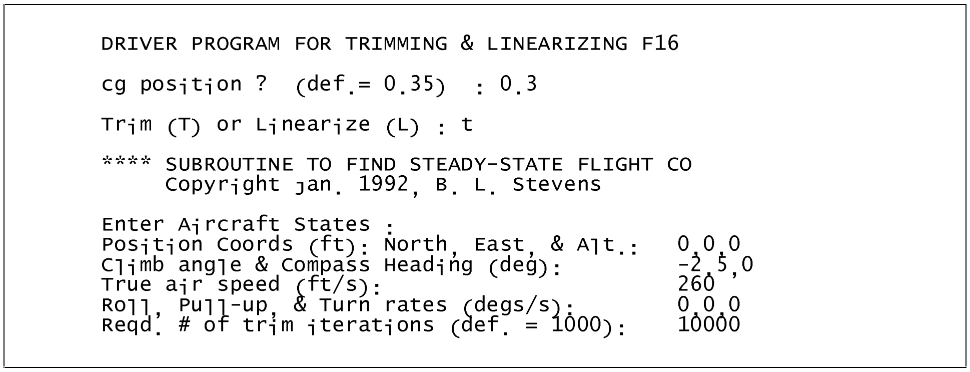
Figure 5: F-16 Control System

4.1 A Book Model for F-16 Aircraft

We used the nonlinear model studied in the book by Stevens to analyze the characteristics of the aircraft system and propose a sound structure for the linear model [Stevens 92]. Since we propose a control design based on a linear model, we need to find a linearized model for the aircraft in landing conditions. This was done by using the Fortran programs provided by Stevens, specifically the *trim* program. After that, we analyzed possible structures for the inner loop controller that meets the demands of the landing specifications. Simulations were carried out using a modified *simu* program, the book simulator.

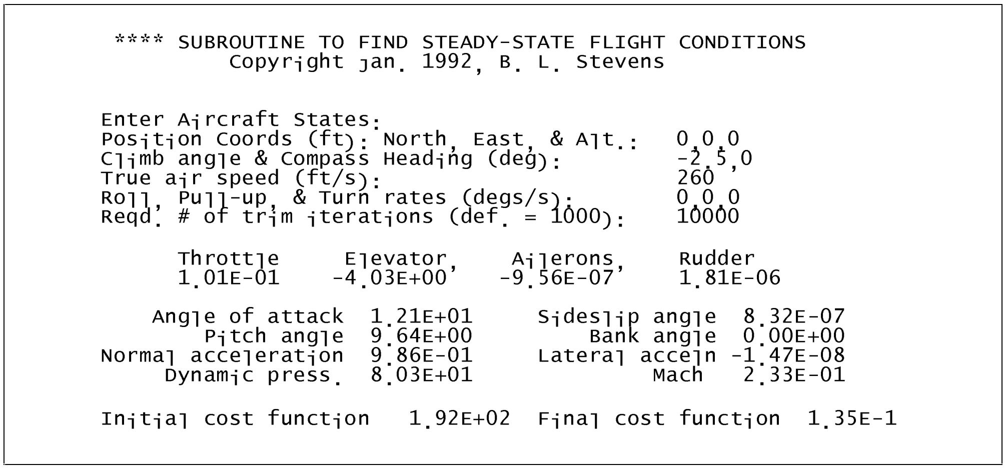
4.1.1 Trim Data

Based on the landing specifications discussed in Section 2, the *lf16* trimming program was used to generate a steady state and linearize the six-degrees of freedom nonlinear model embedded in the program. Figure 6 and Figure 7 show a listing of the input and output of the trimming program, respectively.



```
DRIVER PROGRAM FOR TRIMMING & LINEARIZING F16

cg position ?  (def.= 0.35)  : 0.3

Trim (T) or Linearize (L) : t

**** SUBROUTINE TO FIND STEADY-STATE FLIGHT CO
     Copyright jan. 1992, B. L. Stevens

Enter Aircraft States :
Position Coords (ft): North, East, & Alt.:   0,0,0
Climb angle & Compass Heading (deg):         -2.5,0
True air speed (ft/s):                       260
Roll, Pull-up, & Turn rates (degs/s):        0,0,0
Reqd. # of trim iterations (def. = 1000):    10000
```

Figure 6: Input Listing for the Trimming Program



```
 **** SUBROUTINE TO FIND STEADY-STATE FLIGHT CONDITIONS
          Copyright jan. 1992, B. L. Stevens


Enter Aircraft States:
Position Coords (ft): North, East, & Alt.:   0,0,0
Climb angle & Compass Heading (deg):         -2.5,0
True air speed (ft/s):                       260
Roll, Pull-up, & Turn rates (degs/s):        0,0,0
Reqd. # of trim iterations (def. = 1000):    10000

      Throttle     Elevator,    Ailerons,    Rudder
      1.01E-01    -4.03E+00    -9.56E-07    1.81E-06

    Angle of attack  1.21E+01     Sideslip angle  8.32E-07
        Pitch angle  9.64E+00         Bank angle  0.00E+00
Normal acceleration  9.86E-01     Lateral acceln -1.47E-08
     Dynamic press.  8.03E+01               Mach   2.33E-01

Initial cost function   1.92E+02  Final cost function  1.35E-1
```

Figure 7: Output of the Trimming Program to Screen

Saving these conditions in a file, the same program generates a linearization of the nonlinear model around these conditions. The linearized model is given in the following form:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

The matrices *A*, *B*, *C*, and *D* generated here allow us to evaluate which interactions are important for establishing the structure of the model to match with the model in the real-time simulator. Figures 8 and 9 show an example of the output of the trim and linearization results with some modifications to make them easier to understand. It is important to note at this point the strong structure of the matrices *A* and *B* that come out of the linearization program. They have many zeros establishing the decoupling between lateral and longitudinal motions for this particular case. Different conditions were used for the center of gravity, climb angle, and air velocity with similar results.

```
 13          4        14        0.000000E+00

 2.600000E+02  2.118568E-01  1.452425E-08  0.000000E+00  1.682236E-01
-1.453808E-08  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
 0.000000E+00  0.000000E+00  6.562361E+00

 1.010527E-01 -4.025289E+00 -9.560931E-07  1.811164E-06

 9.858838E-01 -1.465127E-08  1.674313E-01  8.034260E+01  2.328247E-01
 1.213850E+01  8.321781E-07  0.000000E+00  9.638502E+00  0.000000E+00
 0.000000E+00  0.000000E+00  0.000000E+00  9.858838E-01

 DATA ARE : N, M, L, TIME
            STATES
            INPUTS
            OUTPUTS
```

Figure 8: Output of Trimming Program to a File

```
       11          4           0           0           0

   Vt        alpha     beta     phi        theta    psi   P         Q          R        Z     Pow

 -3.94E-02 -2.36E+00 -1.10E-6  4.60E-7 -3.21E+1  0.0  7.74E-9 -3.40       3.06E-8  0.0  3.72E-01
 -9.27E-04 -5.47E-01  0.0      0.0      5.39E-3  0.0 -1.42E-8  9.02E-01 -3.05E-9  0.0 -3.07E-04
 -4.76E-12  1.32E-10 -1.66E-1  0.12     1.79E-9  0.0  2.12E-1  1.90E-10 -9.69E-1  0.0 -2.07E-11
  0.0       0.0       0.0      0.0      0.0      0.0  1.0      0.0       1.69E-1  0.0  0.0
  0.0       0.0       0.0      0.0      0.0      0.0  0.0      1.0       0.0      0.0  0.0
  0.0       0.0       0.0      0.0      0.0      0.0  0.0      0.0       1.01     0.0  0.0
  4.00E-11 -4.73E-07 -1.48E+1  0.0      0.0      0.0 -1.67     2.62E-04  9.27E-1  0.0  0.0
  1.71E-10  2.76E-01  0.0      0.0      0.0      0.0  0.0     -8.39E-01 -2.86E-3  0.0  0.0
  1.07E-10 -5.13E-08  2.45     0.0      0.0      0.0 -4.07E-2  2.53E-03 -2.65E-1  0.0  0.0
  4.36E-02  2.59E+02  0.0      0.0     -2.59E+2  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 -1.0


   THL          EL           AIL          RDR

  0.0000E+00 -1.7739E-02  5.7710E-10  1.5756E-09
  0.0000E+00 -1.1167E-03  0.0000E+00  0.0000E+00
  0.0000E+00  9.9150E-13  1.5282E-04  4.1723E-04
  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
  0.0000E+00  0.0000E+00 -1.8353E-01  3.3777E-02
  0.0000E+00 -5.1736E-02  0.0000E+00  0.0000E+00
  0.0000E+00  0.0000E+00 -7.1762E-03 -1.7061E-02
  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
  6.4940E+01  0.0000E+00  0.0000E+00  0.0000E+00
```

Figure 9: Example Output from Linearization Program Showing Matrices A and B

From this analysis we concluded that there was no reason to try to estimate all of the elements in the matrices that define the linear approximation model. Moreover, we determined which coefficients dominated the dynamic behavior and needed to be estimated using the experimental data.

4.1.2 Inner Loop Control Structure

In order to progress in the study and get a better understanding of aircraft dynamics, an inner loop controller as shown in Figure 5 had to be designed for the book simulator. Classical and modern control techniques could be used for such a task [Stevens 92]. However, we preferred to use modern control techniques that allowed us to introduce the constraints and a priori limits of the linear model into the picture.

For the landing conditions, it was fairly clear from the trim example that lateral and longitudinal dynamics could be separated in design of the internal controller and leave the possible coupling terms for the higher levels of control. Also, the inner loop controller does not have to deal with any specifics of the landing route, so the deviations from the glide path were not considered for feedback. However, it was necessary to introduce feedback from the angle positions to control the aircraft with a reasonable behavior.

A different problem was posed by the throttle command. In the Fortran simulator, we had complete control of the aircraft and could design a throttle control in the same way that we designed the directional feedback control. However, in the real-time simulator and because of flight specifications, we did not have control over the throttle, which introduced another disturbance and internal dynamics into the control design. Therefore, to make things similar in both environments, we designed a decoupled controller for the throttle and worked with that controller throughout the rest of the study.

Thus, the structure of the inner loop controller chosen is as follows:

$$\begin{aligned}
thc &= th_{st} + k_{th}\,(Vt - Vt_{st}) \\
elc &= el_{st} + k_{e1}(\theta - \theta_{st}) + k_{e2}\,(q - q_{st}) + k_{e3}\,Qcom \\
alc &= al_{st} + k_{a1}(\phi - \phi_{st}) + k_{a2}(\psi - \psi_{st}) + k_{a3}\,(p - p_{st}) + k_{a4}\,(r - r_{st}) + k_{a5}\,Pcom \\
rdc &= rd_{st} + k_{r1}(\phi - \phi_{st}) + k_{r2}(\psi - \psi_{st}) + k_{r3}\,(p - p_{st}) + k_{r4}\,(r - r_{st}) + k_{r5}\,Pcom
\end{aligned}$$

with *thc, elc, alc,* and *rdc* being the respective throttle, elevator, aileron, and rudder control commands that feed the actuator dynamics. This controller structure was verified during the minimization procedure involving LMIs and is explained in the next section.

4.1.3 Outer Loop Controller

To be able to meet the specifications of the automatic landing procedure, an outer loop controller was designed. It introduced feedback from the glide path deviations to make the necessary corrections. Again, the minimization procedure semidefinite programming (SDP) showed, as expected, that the vertical deviation error is important only for the elevator control command and that the horizontal deviation is significant only for the aileron and rudder commands. The outer loop controller structure is of the following form:

$$\begin{aligned}
Qcom &= k_{q1}(\theta - \theta_{st}) + k_{q2}\,(q - q_{st}) + k_{q3}\,d_v \\
Pcom &= k_{p1}(\phi - \phi_{st}) + k_{p2}(\psi - \psi_{st}) + k_{p3}\,(p - p_{st}) + k_{p4}\,(r - r_{st}) + k_{p5}\,d_h
\end{aligned}$$

4.2 A Linear Model of the Real-time F-16 Simulator

In this subsection, we apply the results obtained in the last subsection to identify a linear model for the real-time F-16 simulator in landing. In particular, we use the obtained aircraft structure, the actuator characteristics, and the inner loop control to build an augmented system for the real-time simulator with unknown parameters. Then identification of the system becomes an issue of parameter identification.

4.2.1 Model Structure

According to the results from Section 4.1.3, we construct the model as

$$\dot{x} = Ax + Bu, \quad y = Cx$$

with $x = [\delta\phi,\ \delta\theta,\ \delta\varphi,\ P,\ Q,\ R,\ \delta_{el},\ \delta_{ail},\ \delta_{rdr}]$,

$$A = \begin{bmatrix} 0 & 0 & 0 & 1.0 & 0 & 0.16983 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0143 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_{44} & 0 & a_{46} & 0 & a_{48} & a_{49} \\ 0 & a_{52} & 0 & 0 & a_{55} & 0 & a_{57} & 0 & 0 \\ 0 & 0 & 0 & a_{64} & 0 & a_{66} & 0 & a_{68} & a_{69} \\ 0 & 0 & 0 & 0 & a_{75} & 0 & a_{77} & 0 & 0 \\ a_{81} & 0 & a_{83} & a_{84} & 0 & a_{86} & 0 & a_{88} & 0 \\ a_{91} & 0 & a_{93} & a_{94} & 0 & a_{96} & 0 & 0 & a_{99} \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ b_{71} & 0 \\ 0 & b_{82} \\ 0 & b_{92} \end{bmatrix},$$

$$C = [I_{6\times 6},\ 0_{6\times 3}].$$

where $\delta *$ or δ_* are the differences between the actual variables and their steady state value, and parameters a_{ij} and b_{ij} in *A* and *B* matrices are to be identified.

The physical interpretation of this structure is clear. In fact, partitioning the state *x* as $x = [\bar{x}_1, \bar{x}_2]$ with $\bar{x}_1 = [\delta\phi,\ \delta\theta,\ \delta\varphi,\ P,\ Q,\ R]$ and $\bar{x}_2 = [\delta_{el},\ \delta_{ail},\ \delta_{rdr}]$, and

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} 0_{6\times 2} \\ B_2 \end{bmatrix}$$

accordingly, it can be seen that matrices A_{11} and A_{12} determine the aircraft dynamics, A_{21} constructs the inner loop controller, and A_{22} represents the actuator dynamics. The block diagram shown in Figure 10 illustrates such an interpretation.

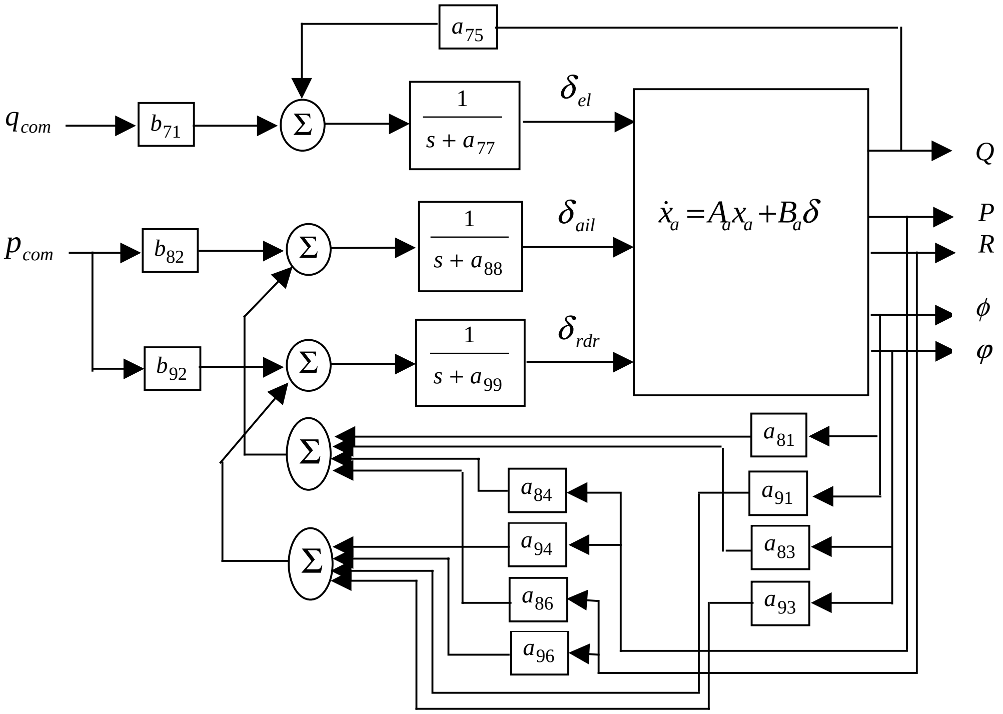


Figure 10: A Block Diagram for the Structure of the Model

4.2.2 Parameter Identification

To identify the parameters, we formulated a minimization problem and used the Matlab function *fmins* to solve it. The detailed procedure is described in the following steps.

Step 1: Collect data from the real-time simulator. Specifically, fly the aircraft through the landing stage and record the values of the state variables. The state variables used in parameter identification are

$$state\_data = [\phi,\ \theta,\ \varphi,\ P,\ Q,\ R] - steady\_state\_state$$

and the control variables are

$$control\_data = [Q_{com},\ \ P_{com}] - steady\_state\_control$$

where *steady_state_state* and *steady_state_control* are the mean values of the involved variables. It is important that the data are obtained from a successful landing.

Step 2: Solve a minimization problem using Matlab function *fmins*. The function *fmins* is used in the form

par = *fmins*('F_name', par_0)

which attempts to return a vector *par* that minimizes objective function *F*(*par*) defined in the file F_name.m. par_0 is an initial guess of the values of *par*. To use function *fmins* in parameter identification, we define the objective function as the absolute error between the data collected from the simulator and the trajectory generated by the model with trial parameters. Specifically, for each set of parameters (starting with an initial guess), we complete the matrices *A* and *B*, simulate the trajectory of system $\dot{x} = Ax + Bu$ with the initial condition

$$x(0) = [\phi(t_0), \theta(t_0), \varphi(t_0), P(t_0), Q(t_0), R(t_0), 0, 0, 0] - [steady\_state\_state, 0, 0, 0],$$

and control input *control_data*, and then compute the error

$$error = | x(1) - \phi | + | x(2) - \theta | + | x(3) - \varphi | + | x(4) - P | + | x(5) - Q | + | x(6) - R |.$$

After the execution of *fmins* terminates, vector *par* contains the parameters to be identified.

Step 3: Verify the obtained model by comparing the model trajectories with the real data obtained from another run of the landing simulation. If the result is satisfactory, model identification is complete. Otherwise, try another set of initial parameters and repeat Step 2.

We now apply the described procedure to parameter identification. As mentioned earlier, the model that we will identify is with respect to a particular flight condition (i.e., landing in this case). Hence, it is important to make sure that the real data we use are indeed collected when the aircraft is landing. Figure 11 shows the altitude of the aircraft in the group of data gathered.

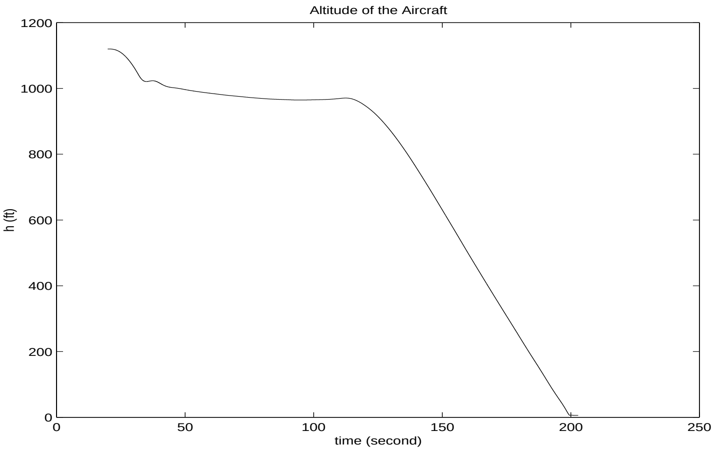


Figure 11: Altitude of the Aircraft in Landing

As can be seen from Figure 11, the aircraft was actually landing after 120 seconds. Therefore, the portion of data to be used for identification should be between time 120 and 190 seconds. With this selected set of data, the parameters are identified by working through Step 2 in the procedure. As a result, we obtain the model given as follows:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1.0 & 0 & 0.16983 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0143 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.1477 & 0 & -0.091927 & 0 & 0.36697 & -0.16544 \\ 0 & -0.018272 & 0 & 0 & -0.33702 & 0 & -0.1094 & 0 & 0 \\ 0 & 0 & 0 & 0.11259 & 0 & -0.24985 & 0 & -0.00333 & 0.048502 \\ 0 & 0 & 0 & 0 & 5.1649 & 0 & -6.8055 & 0 & 0 \\ -1.3393 & 0 & -1.1633 & -7.5556 & 0 & -6.7804 & 0 & -5.5285 & 0 \\ 1.018 & 0 & -1.0342 & -4.6727 & 0 & -4.4014 & 0 & 0 & -6.4675 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -30.105 & 0 \\ 0 & 9.9546 \\ 0 & 4.2838 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 12 shows the trajectory of the identified system plotted against the real data used for identification in the time interval [145, 190] seconds. It can be seen that the simulated data match the real data reasonably well. Figures 13 and 14 illustrate the model verification result, where Figure 13 depicts the result for the landing period and Figure 14 displays the comparison for the whole range of data. Again, the match between the simulated data and the real data is good except for some disturbance in the real data. In particular, the result in the long run demonstrates that the model identified is reasonably robust (i.e., it represents the real system fairly well even with variations of flight condition).

During landing, the aircraft should be controlled so it flies along the glide path. Hence the control problem becomes stabilization of the aircraft on the glide path. In the set of measured data, the deviations from the glide is described by two variables, g_v and g_h, the angles between the straight line from aircraft cg to the touch-down point and the glide path in the vertical plane and horizontal plane, respectively. Figure 15 shows these two angles.

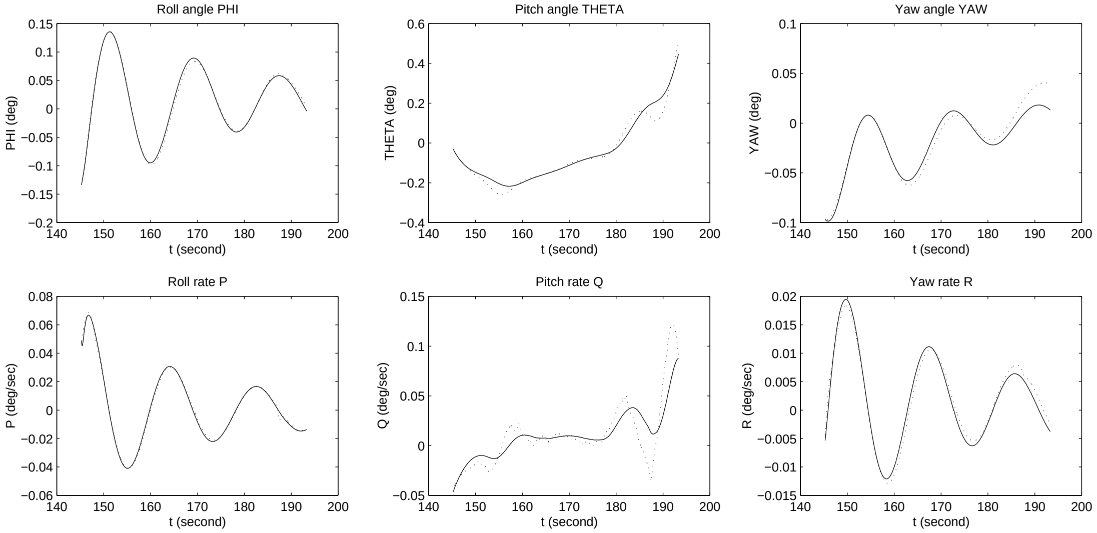


Figure 12: Trajectory of Identified Model (Solid Line) and Real Data Used for Identification (Dotted Line)

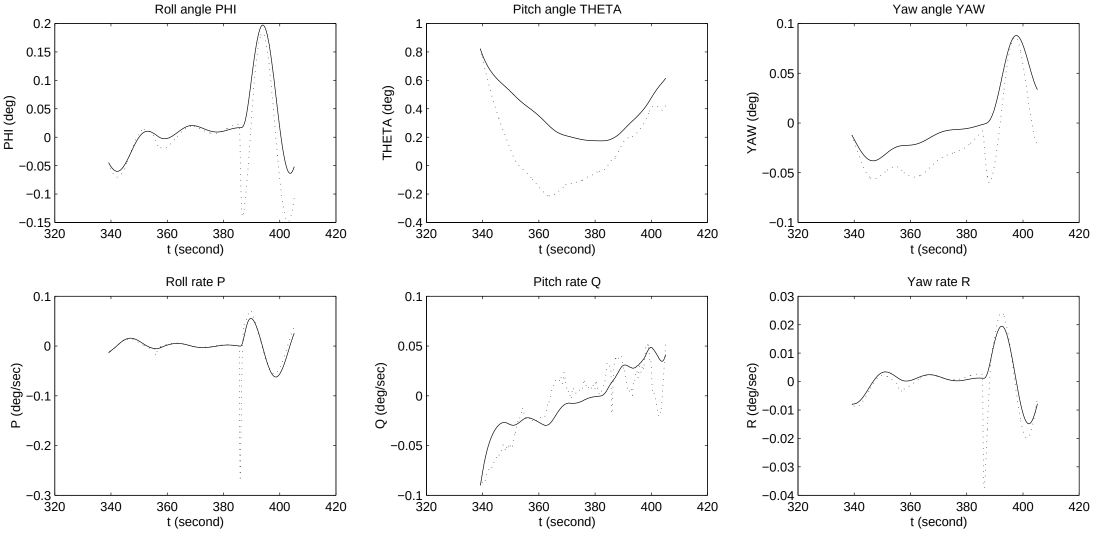


Figure 13: Results of Model Verification in the Landing Period (Solid Line:Trajectory of the Identified Model, Dotted Line: Real Data)

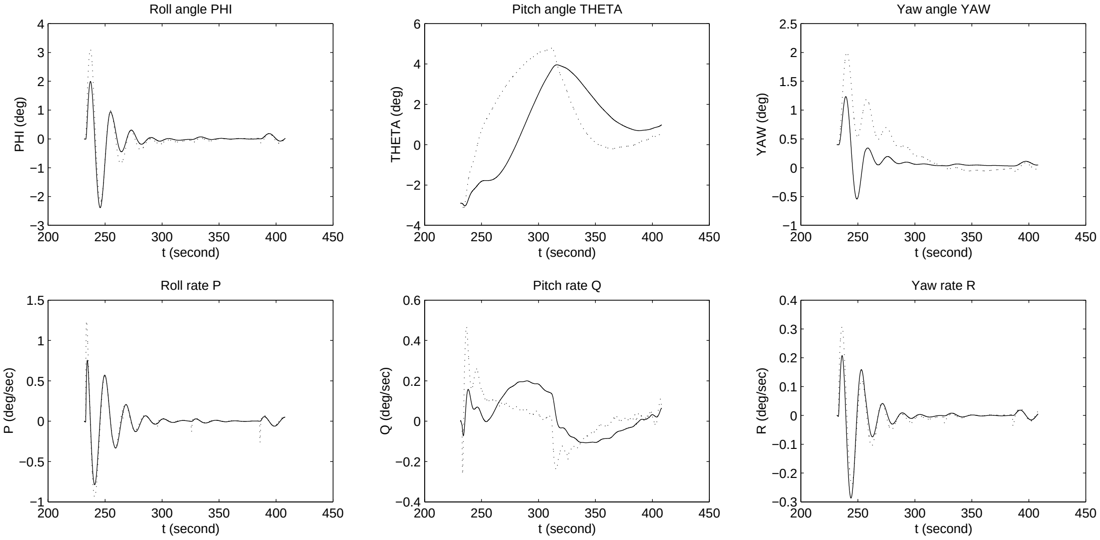


Figure 14: Results of Model Verification in a Long Run (Solid Line: Trajectory of Identified Model, Dotted Line: Real Data)

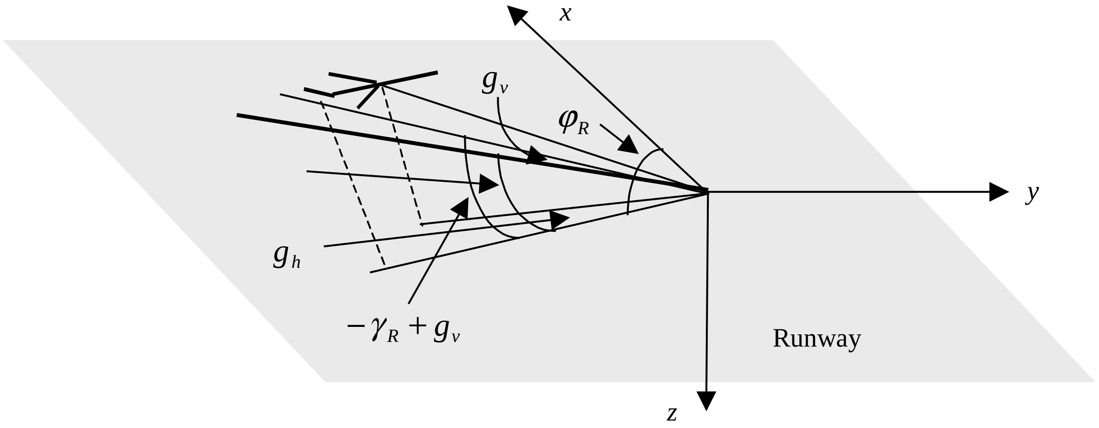


Figure 15: Angles Determining Glide Path Derivation

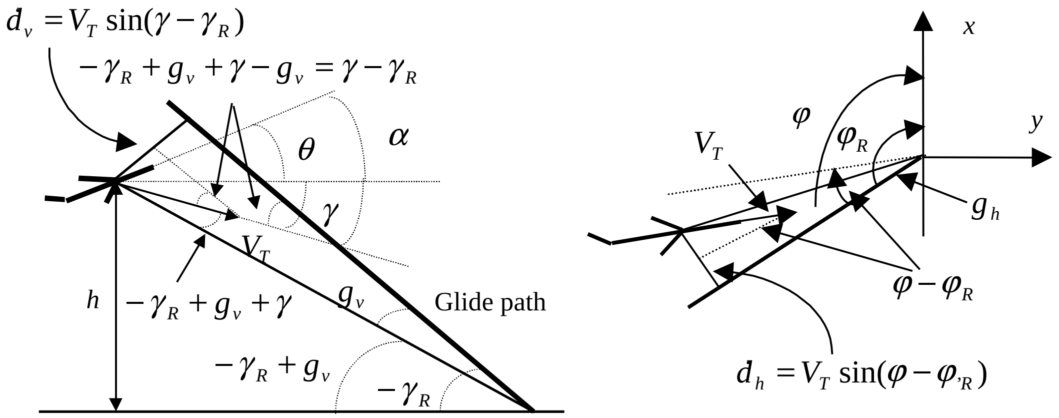
When a control algorithm is designed to stabilize the aircraft along the glide path, it is preferable to regulate the aircraft deviation from the glide path, namely the distance from the aircraft cg to the glide path. Such a distance can be resolved into two components, the horizontal deviation d_h in the horizontal plane parallel to the Earth's surface, and the vertical deviation d_v in the vertical plane perpendicular to the Earth's surface and containing the glide path. Then referring to Figure 16, we obtain the vertical deviation to the glide path as follows:

$$\dot{d}_v = V_T \sin(\gamma - \gamma_R), \qquad d_v = \frac{\sin g_v}{\sin(-\gamma_R + g_v)} h \tag{5}$$

and the horizontal derivation to be

$$\dot{d}_h = V_T \sin(\varphi - \varphi_R), \qquad d_h = \sqrt{p_N^2 + p_E^2} \sin g_h . \tag{6}$$

where V_T is the steady-state value of the air speed v_T .


$\dot{d}_v = V_T \sin(\gamma - \gamma_R)$
$-\gamma_R + g_v + \gamma - g_v = \gamma - \gamma_R$
θ
α
γ
V_T
h
$-\gamma_R + g_v + \gamma$
g_v
Glide path
$-\gamma_R + g_v$
$-\gamma_R$
x
y
φ
φ_R
V_T
g_h
$\varphi - \varphi_R$
$\dot{d}_h = V_T \sin(\varphi - \varphi_{,R})$

(a) Vertical derivation

(b) Horizontal derivation

Figure 16: Derivations to the Glide Path

To incorporate the derivations to the glide path in the model, we linearize the differential equations of in Equations 5 and 6 as

$$\dot{d}_v = V_T \delta\theta, \qquad \dot{d}_h = V_T \delta\varphi .$$

Here we have assumed that the air speed v_T and the angle of attack α are kept constant by the throttle control, and the steady-state value v_T is $V_T = 260\,\text{ft/sec}$. Then the complete model to be used in control design involve 11 states, which are as follows:

$$x = [\delta\phi,\ \delta\theta,\ \delta\varphi,\ P,\ Q,\ R,\ d_v,\ d_h,\ \delta_{el},\ \delta_{ail},\ \delta_{rdr}]$$

and matrices *A* and *B* are given by the following:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1.0 & 0 & 0.16983 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0143 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.1477 & 0 & -0.091927 & 0 & 0 & 0 & 0.36697 & -0.16544 \\ 0 & -0.018272 & 0 & 0 & -0.33702 & 0 & 0 & 0 & -0.1094 & 0 & 0 \\ 0 & 0 & 0 & 0.11259 & 0 & -0.24985 & 0 & 0 & 0 & -0.00333 & 0.048502 \\ 0 & 260 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 260 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5.1649 & 0 & 0 & 0 & -6.8055 & 0 & 0 \\ -1.3393 & 0 & -1.1633 & -7.5556 & 0 & -6.7804 & 0 & 0 & 0 & -5.5285 & 0 \\ 1.018 & 0 & -1.0342 & -4.6727 & 0 & -4.4014 & 0 & 0 & 0 & 0 & -6.4675 \end{bmatrix} \tag{7}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -30.105 & 0 \\ 0 & 9.9546 \\ 0 & 4.2838 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

This will be the model considered in the next section when the control algorithm is designed.

5 Stabilization Control and Stability Analysis

Based on the linearized model developed, control algorithms were designed to stabilize the aircraft along the glide path. Since the designed control laws were implemented as the baseline control, we also derived the stability region of the closed-loop system as the criterion for switching to the baseline controller when the upgraded controller causes the aircraft to behave abnormally. In this section, we review the LMI-based control design and stability derivation discussed in Seto's technical report, then apply the approach to the stabilization problem for F-16 landing [Seto 99]. Finally, we present some experiments with the book simulator, where the proposed design approach is used to design various control algorithms for the book model of the F-16 in [Stevens 92].

5.1 LMI-Based Control Design Approaches

In this subsection, LMI techniques are used to design the feedback controller with state and input constraints [Boyd 94]. Since the performance of the controlled system is determined by the poles of the closed-loop system, constraints on the pole location in the complex plane may be specified to enforce certain performance. Such constraints can also be expressed in terms of LMI [Chilali 96]. Details are provided below. Consider a linear system in state space form

$$\dot{x} = Ax + Bu, \quad x \in R^n, u \in R^m$$

subject to the state, control, and rate constraints as

$$a_i x < 1,\ \ a_i \in R^n,\ \ i = 1,...,q,\ \ \ b_j u \le 1,\ \ b_j \in R^m,\ \ \ j = 1,...,r, \text{ and } c_k \dot{x} < 1, k = 1,...,l.$$

Our main goal was to design a state feedback controller $u = Kx$ that meets the constraint specifications and find out its maximal ellipsoidal stability region

$$S = \{ x : x^T Px < 1\}$$

with P a symmetric positive definite matrix, denoted by $P > 0$.

Stability, constraints, and pole placements can all be expressed in terms of LMI in this case. Stability reduces the existence of an $n \times n$ symmetric matrix *Q* and an $m \times n$ matrix *Y* that satisfy the following LMI constraints:

$$\begin{aligned} &Q > 0 \\ &AQ + QA^T + BY + Y^T B^T < 0 \end{aligned} \tag{8}$$

Then the control gain is given by $K = YQ^{-1}$, and the ellipsoid is determined by $P = Q^{-1}$. The constraints on the states, $a_i x < 1$, are translated into the following LMIs:

$$a_i Q a_i^T < 1, \quad i = 1, ..., q .$$

The constraints on the controls, $b_j u \leq 1$, can be expressed in LMI as

$$\begin{bmatrix} 1 & b_j Y \\ Y^T b_j^T & Q \end{bmatrix} > 0, \quad j = 1, ..., r ,$$

and the constraints on the rate of change of the states, $c_k \dot{x} < 1$, can be written in LMI as

$$\begin{bmatrix} 1 & c_k AQ + c_k BY \\ (c_k AQ + c_k BY)^T & Q \end{bmatrix} > 0, \ k = 1, \ldots, l .$$

The location of the poles of the closed-loop system can be restricted to a prescribed region in the complex plane, such as semi-planes, disks, or cones, to meet the performance requirement. For example, to reach a decay rate greater than α in the closed-loop response, the poles have to be in the complex semi-plane $\mathrm{Re}(pole) < -\alpha$. This is easily realized by modifying Equation 8 as follows:

$$\begin{aligned} &Q > 0 \\ &AQ + QA^t + BY + Y^t B^t + 2\alpha Q < 0 \end{aligned}$$

For the poles to be inside a disk of center –*q* and radius *r*, the following LMI has to be added to Equation 8:

$$\begin{bmatrix} -rQ & qQ + AQ \\ qQ + QA^t & -rQ \end{bmatrix} < 0.$$

When the desired region is a cone with its center in the origin of the complex plane and symmetric to the real axis with angle θ, the following LMI must be specified in addition to Equation 8:

$$\begin{bmatrix} \sin\theta(AQ+QA^T) & \cos\theta(AQ-QA^T) \\ \cos\theta(QA^T-AQ) & \sin\theta(AQ+QA^T) \end{bmatrix} < 0.$$

These constraints offer good options to set up the pole locations for the closed-loop system because it allows us to set up a decay rate and to limit overshoot and settling times. Figure 17 displays a possible combination of the LMI constraints described above.

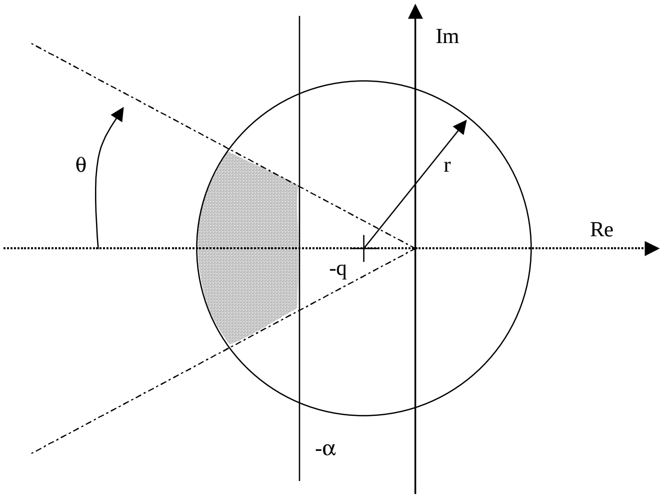


Figure 17: Pole Placement Constraint Options

To maximize the ellipsoid, we have to maximize *det*(*Q*) or equivalently minimize *–log det*(Q). The computation of the optimization problem is performed using the SDPSOL package developed by Shao-Po Wu and S. Boyd [Wu 96]. In the subsequent subsections, we formulate the stabilization problem for F-16 landing as an optimization problem in terms of LMIs and solve it with the SDPSOL software package. We first carry out the procedure with the identified model, and then apply it to the book model and test the rest with the book simulator. Pole location constraints are introduced to the control design with the book model.

5.2 Control Design for the Identified F-16 Model

We now apply the approach described above to the design of a state feedback control such that the aircraft will fly along the glide path and the stability region of the closed-loop system is maximized. Since the aircraft is in the landing mode, there are certain constraints on the state of the aircraft, called *flight envelope* that must be satisfied. Such an envelope restricts the aircraft’s dynamics to a greater extent when it is approaching the touchdown point. This requires the controller to be more sophisticated in term of providing precise control along the glide path as the aircraft approaches the runway. Therefore, we will design two controllers: one will be used before the aircraft reaches the decision height, and another will be in charge after the aircraft reaches the decision height. The former will be designed such that the stability region of the closed-loop system is larger than the latter controller’s stability region, but the former may take a longer time than the latter to drive the aircraft to converge to the glide path. These controllers are designed by solving a LMI problem described earlier for each, with different sets of state constraints. Consider a system in the form

$$\dot{x} = Ax + Bu$$

with matrices A and B given in Equation 5 and

$$x = [\delta\phi,\ \delta\theta,\ \delta\varphi,\ P,\ Q,\ R,\ d_v,\ d_h,\ \delta_{el},\ \delta_{ail},\ \delta_{rdr}],\ \ u = [\delta q_{com},\ \delta p_{com}]$$

subject to

$$\begin{array}{lllll}
|x_1| < 20\,\text{deg} & |x_4| < 40\,\text{deg/sec} & |x_7| < 50\,\text{ft} & |x_9| < 20\,\text{deg} & |\dot{x}_9| < 60\,\text{deg/sec} \\
|x_2| < 20\,\text{deg} & |x_5| < 40\,\text{deg/sec} & |x_8| < 50\text{ft} & |x_{10}| < 20\,\text{deg} & |\dot{x}_{10}| < 80\,\text{deg/sec} \\
|x_3| < 20\,\text{deg} & |x_6| < 40\,\text{deg/sec} & & |x_{11}| < 20\,\text{deg} & |\dot{x}_{11}| < 120\,\text{deg/sec}
\end{array}$$

before the aircraft reaches the decision height (before DH), and

$$\begin{array}{lllll}
|x_1| < 5\,\text{deg} & |x_4| < 10\,\text{deg/sec} & |x_7| < 5\,\text{ft} & |x_9| < 24\,\text{deg} & |\dot{x}_9| < 60\,\text{deg/sec} \\
|x_2| < 2.5\,\text{deg} & |x_5| < 10\,\text{deg/sec} & |x_8| < 15\text{ft} & |x_{10}| < 20\,\text{deg} & |\dot{x}_{10}| < 80\,\text{deg/sec} \\
|x_3| < 5\,\text{deg} & |x_6| < 10\,\text{deg/sec} & & |x_{11}| < 29\,\text{deg} & |\dot{x}_{11}| < 120\,\text{deg/sec}
\end{array}$$

after the aircraft reaches the decision height (after DH). The control objective is to find the control gain *K* for the linear state feedback control $u = Kx$ such that the closed-loop system $\dot{x} = (A + BK)x$ is asymptotically stable with the above constraints satisfied and its stability region is maximized.

The solution to the problem is obtained by solving the symmetric matrix *Q* and matrix *Y* from the following LMI problem:

$$\begin{array}{ll}
\text{minimize} & \log\det Q^{-1} \\
\text{subject to} & Q > 0; \\
 & AQ + QA^T + BY + Y^T B^T < 0;
\end{array}$$

$$a_i Q a_i^T < 1, \quad i = 1,\dots,11$$

$$\begin{bmatrix} 1 & c_k AQ + c_k BY \\ (c_k AQ + c_k BY)^T & Q \end{bmatrix} > 0, \quad k = 1,2,3$$

where $a_i, i = 1,\dots,11$, are the rows of the matrix

diag(1/20, 1/20, 1/20, 1/40, 1/40, 1/40, 1/50, 1/50, 1/24, 1/20, 1/29) before DH or diag(1/5, 1/2.5, 1/5, 1/10, 1/10, 1/10, 1/5, 1/15, 1/24, 1/20, 1/29) after DH,

and $c_1 = [0_{1\times 9}, 1, 0, 0]$, $c_2 = [0_{1\times 9}, 0, 1, 0]$, $c_3 = [0_{1\times 9}, 0, 0, 1]$. Therefore, the complete solution to the problem will be two sets of control gains with two corresponding stability regions. Using the SDPSOL software package [Wu 96], we get the solutions to these two LMI problems as follows:

Before the aircraft reaches the decision height, the control *K* should be as follows:

-0.0000	-0.2589	-0.0000	-0.0000	-0.1745	-0.0000	-0.0156	-0.0000	-0.1582	-0.0000	-0.0000
0.0207	-0.0000	-0.4737	0.7265	-0.0000	-0.5042	-0.0000	-0.0217	0.0000	0.3186	0.3200

and the symmetric matrix *Q* is as follows:

400.0000	-0.0008	-17.1723	-211.8347	0.0008	24.4667	-0.0317	-119.9860	-0.0035	-59.7096	113.7391
-0.0008	111.2293	0.0002	-0.0006	-70.6586	-0.0002	-198.2237	0.0008	31.5304	-0.0021	-0.0012
-17.1723	0.0002	48.7876	9.7757	-0.0002	-21.7717	-0.0022	-136.9569	0.0008	-8.4224	-7.4897
-211.8347	-0.0006	9.7757	436.7589	-0.0003	-30.2084	0.0030	17.4860	0.0029	-20.5461	-138.9358
0.0008	-70.6586	-0.0002	-0.0003	67.8700	0.0002	-0.5276	0.0007	43.0385	0.0003	0.0006
24.4667	-0.0002	-21.7717	-30.2084	0.0002	14.3932	0.0005	10.1714	-0.0005	-13.1903	1.5514
-0.0317	-198.2237	-0.0022	0.0030	-0.5276	0.0005	2500.0000	-0.0724	-84.7781	0.0187	-0.0003
-119.9860	0.0008	-136.9569	17.4860	0.0007	10.1714	-0.0724	2500.0000	-0.0014	105.3920	14.7288
-0.0035	31.5304	0.0008	0.0029	43.0385	-0.0005	-84.7781	-0.0014	576.0000	-0.0026	-0.0014
-59.7096	-0.0021	-8.4224	-20.5461	0.0003	-13.1903	0.0187	105.3920	-0.0026	400.0000	135.0018
113.7391	-0.0012	-7.4897	-138.9358	0.0006	1.5514	-0.0003	14.7288	-0.0014	135.0018	663.8984

After the aircraft reaches the decision height, the control gain *K* is changed to the following:

0.0000	-1.4162	-0.0001	0.0000	-0.9437	-0.0001	-0.1570	-0.0000	-0.1521	0.0000	-0.0000
-0.1313	-0.0000	-1.8460	0.5508	0.0000	-2.9336	-0.0000	-0.0921	-0.0000	0.3127	0.2981

and the symmetric matrix Q becomes the following:

25.0000	0.0001	-1.3287	-13.3985	-0.0001	1.7656	0.0009	-11.7317	0.0027	-19.7653	1.5411
0.0001	4.1497	-0.0000	-0.0000	-4.5740	0.0000	-3.7298	0.0001	7.4630	-0.0007	-0.0005
-1.3287	-0.0000	5.0008	-1.4746	-0.0000	-2.1862	-0.0001	-14.3404	0.0002	-1.4237	0.1915
-13.3985	-0.0000	-1.4746	48.7895	0.0004	0.0699	-0.0003	4.0581	0.0021	-25.9298	-15.1924

-0.0001	-4.5740	-0.0000	0.0004	9.1149	0.0000	-1.2334	0.0001	20.3156	0.0019	0.0010
1.7656	0.0000	-2.1862	0.0699	0.0000	1.3932	0.0001	1.2728	0.0004	-4.5244	-5.9309
0.0009	-3.7298	-0.0001	-0.0003	-1.2334	0.0001	25.0000	-0.0046	-13.6189	-0.0017	0.0000
-11.7317	0.0001	-14.3404	4.0581	0.0001	1.2728	-0.0046	225.0000	-0.0016	22.5049	6.1590
0.0027	7.4630	0.0002	0.0021	20.3156	0.0004	-13.6189	-0.0016	576.0000	-0.0045	-0.0024
-19.7653	-0.0007	-1.4237	-25.9298	0.0019	-4.5244	-0.0017	22.5049	-0.0045	400.0000	138.6215
1.5411	-0.0005	0.1915	-15.1924	0.0010	-5.9309	0.0000	6.1590	-0.0024	138.6215	605.2974

To illustrate the obtained result, we plot a series of two-dimensional projections of the stability region by zeroing out the other nine variables. Figure 18 shows the results for the case before the aircraft reaches the decision height (larger stability region), and Figure 19 displays the results for the case after the aircraft reaches the decision height (smaller stability region).

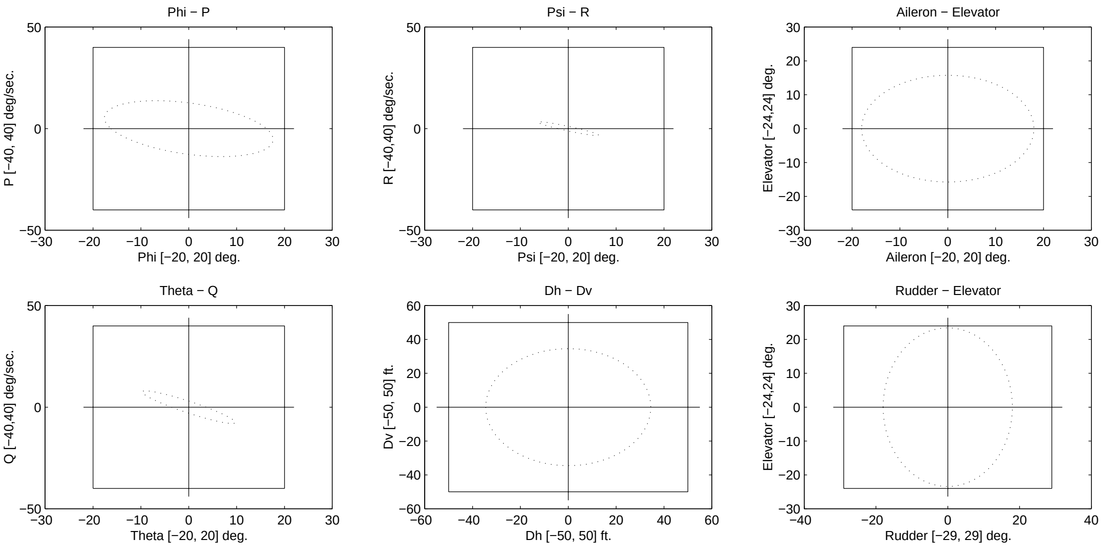


Figure 18: Two-Dimesional Projections of the Stability Region Before Aircraft Reaches Decision Height

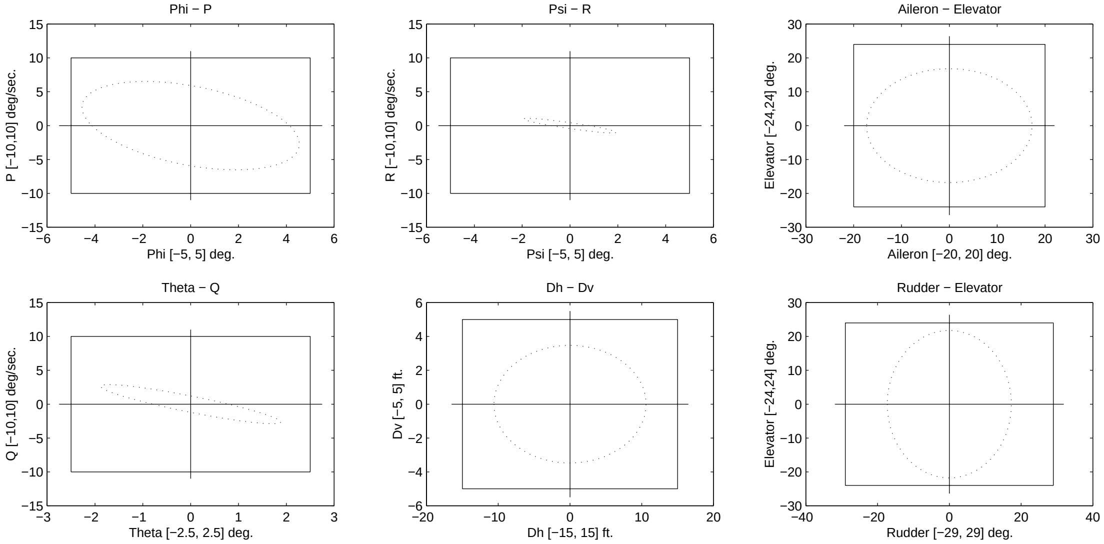


Figure 19: Two-Dimensional Projections of the Stability Region After Aircraft Reaches Decision Height

To check if an aircraft state is inside one of the stability regions obtained, we need to evaluate the function $V = x^T Px$. If the resulting $V < 1$, the state is inside the stability region; otherwise it is not. In reality, variables $\delta\phi$, $\delta\theta$, $\delta\varphi$, P, Q and R are measurable; d_v and d_h can be calculated from Equations 5 and 6; but δ_{el}, δ_{ail} and δ_{rdr} may not be available for computation in each sampling period. If this is the case, an observer is needed to estimate these variables. In our development, we used the identify observer in the following form:

$$\dot{\hat{x}} = A\hat{x} + L(y - C\hat{x}) + Bu$$

where $\hat{x}$ is the estimated state with the last three elements being the ones that we are interested in, matrices *A*, *B* and *C* are defined in Equation 7; *u* is the control command used in the real-time simulation; and the observer gain *L* is given by

16.9301	-0.0460	0.0844	0.9212	-0.0578	0.0891
-0.0252	12.5374	-0.8626	-0.0170	0.9634	-0.0481
0.0349	-0.8516	13.3853	0.0793	0.0139	1.1010
-0.9352	-0.5823	1.6391	20.4046	-3.2620	5.6179
0.0397	-0.0765	-0.0788	-1.1621	20.9738	-0.3720
-0.8829	-0.6246	1.4325	1.4397	-1.7598	21.2326
-12.3993	-4.6792	21.1685	72.1876	-391.3307	28.6433
-97.3066	-68.1473	163.0082	272.3444	-248.7508	713.2703
-144.2297	-105.5620	245.3321	220.5524	-333.9311	1143.7588

Figure 20 shows the comparisons of the real data $\delta\phi$, $\delta\theta$, $\delta\varphi$, P, Q, and R and the observed data, which demonstrate that the estimates match the real data very well after a short period of time.

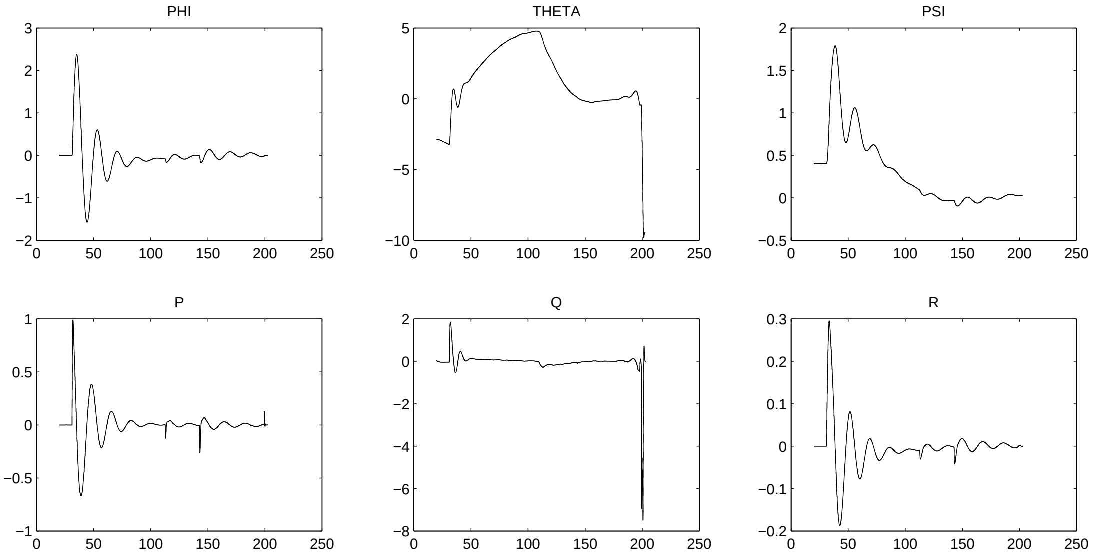
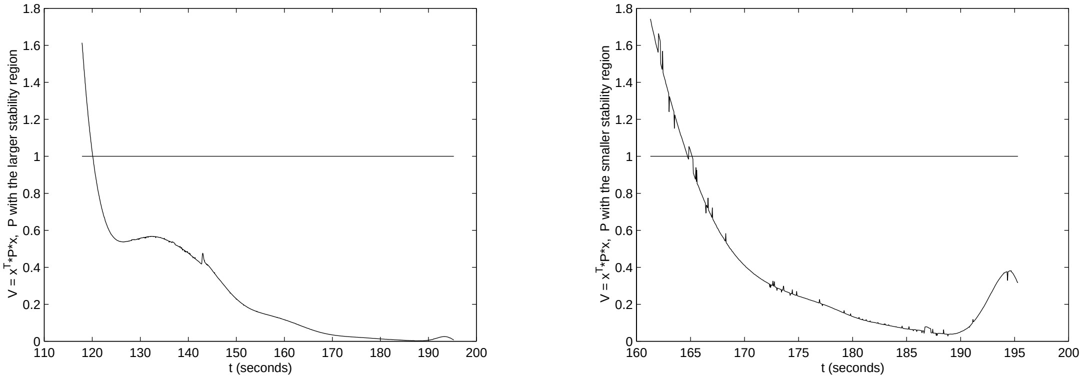


Figure 20: Observed State (Dotted Lines) and Real Data (Solid Lines)

To test the obtained controllers and the stability regions before actual implementation, we checked if the stability regions contain all the states of the aircraft obtained from running the real-time simulator. Since the data were collected in a successful landing, the states during the landing should be contained in the larger stability region first, and in the smaller stability region later. Figure 21 shows exactly this claim; namely, after 120 seconds, the aircraft started landing and the value of *V* was less than 1. From roughly 165 seconds on, the states of the aircraft were inside the smaller stability region until the aircraft touched down. The two controllers and the corresponding stability regions were thus ready for implementation.



(a) States inside the larger stability region.

(b) States inside the smaller stability region.

Figure 21: Verification of the Aircraft State Inside the Stability Regions

5.3 Experimental Tests with the Book Model

The control design approach described in Section 5.2 was tested with the book simulator. Specifically, we formulated the same optimization problem in terms of the LMI constraints for the book model, and solved for the control gain and the stability region. Then the results were implemented in the book simulator, and two types of switching experiments were run to verify the feasibility of the approach.

In one type of experiment, two different outer loop controllers (recall the outer loop controller is an implementation of the control laws for q_{com} and p_{com}) were designed to work in sequence; the first one controls the aircraft up to the decision-height point, and the second one takes over the control afterward. The first controller was derived with relaxed conditions on the states and glide-path errors and a decay rate to ensure that hard constraints were satisfied before the decision-height point was reached. The second controller, on the other hand, was designed with restricted state and glide-path constraints. Figures 22-25 show a switching experiment of this type.

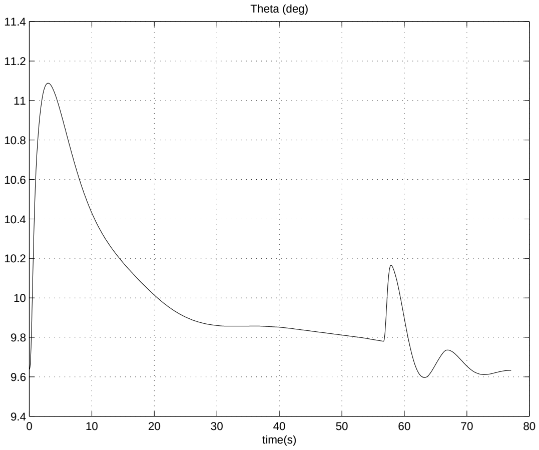


Figure 22: Pitch Evolution During Switching Experiment

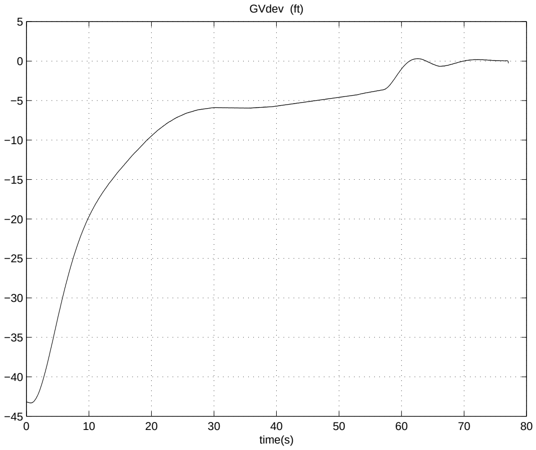


Figure 23: Vertical Derivation Behavior in a Switching Experiment

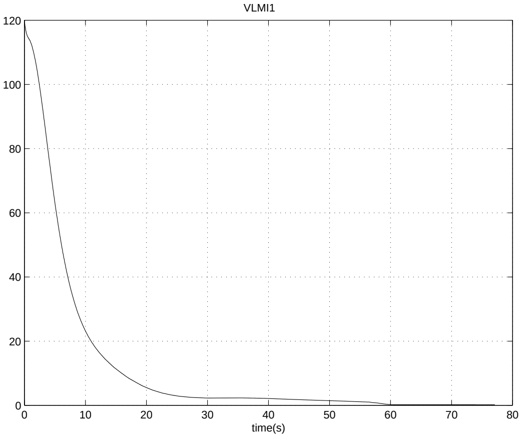


Figure 24: Stability Index of Controller for Decision Height Region

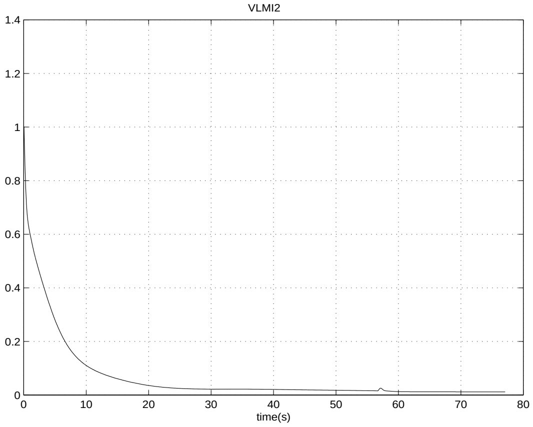


Figure 25: Stability Index of Controller for Larger Region

In the other types of experiments, we showed the recovery from a bad controller. Specifically, three controllers were running, and one of them, say, Controller 2, contained a bug. Controller 3 was designed to work in a larger state space than the decision-height specifications, and Controller 1 was fine-tuned for the decision height. Both Controller 1 and Controller 3 were reliable. A stability represented by a Lyapunov function was derived with respect to Controller 3, and Controller 3 guaranteed the stability of the system so long as the value of the Lyapunov function evaluated at the current state of the system was less than 1. As shown in Figures 26-30, a bug was introduced in Controller 2 at a time equal to 10 seconds, which eventually made the closed-loop system go out of the stability region. Controller 3 took over control when it predicted the out-of-bound Lyapunov function, and it recovered the system. At some later time, Controller 1 was switched on and landed the aircraft.

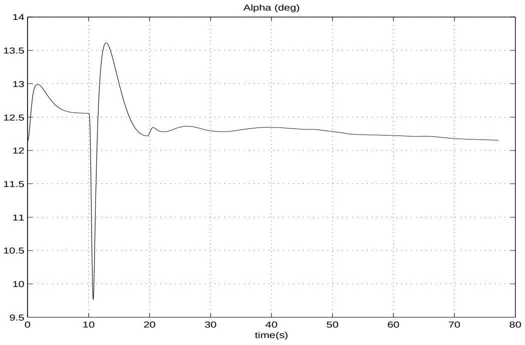


Figure 26: Behavior of Angle of Attack in a Recovery Experiment

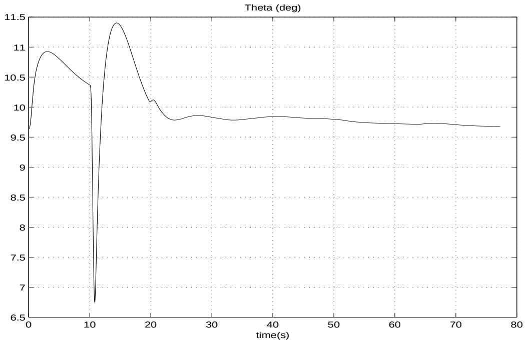
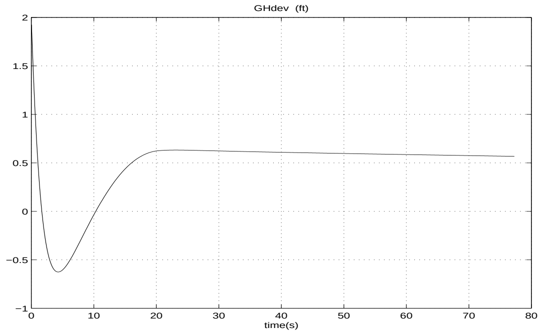


Figure 27: Pitch Angle Behavior During Recovery Experiment



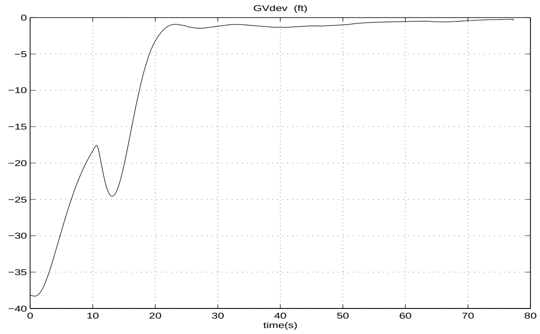


Figure 28: Horizontal Deviation from Gliding Path

Figure 29: Vertical Deviation from Gliding Path

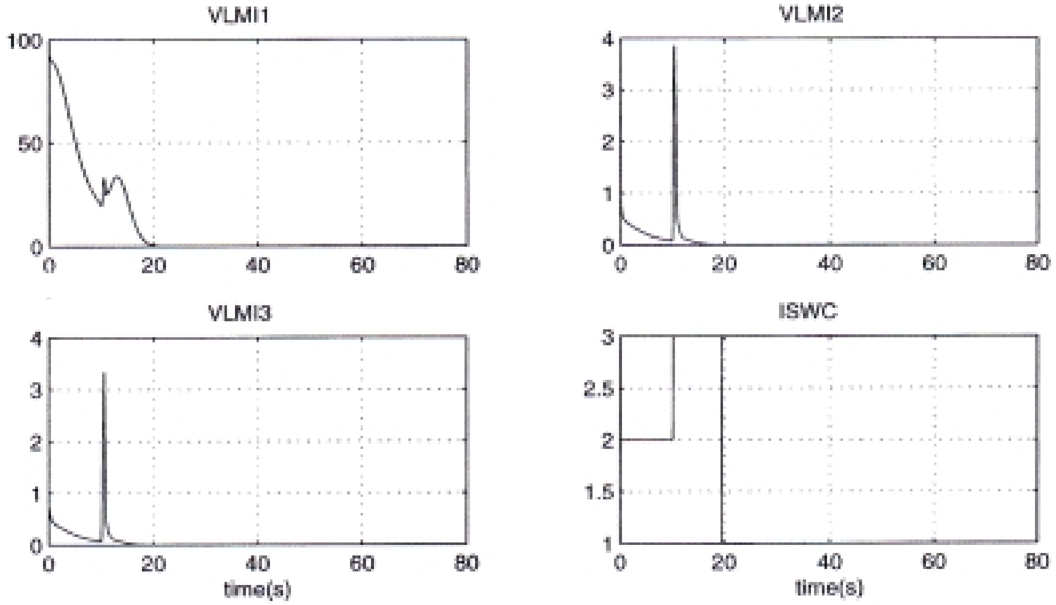


Figure 30: Stability Indices for the Three Controllers During Recovery Experiment and Switching Index Indicating Which Controller Is Being Applied

6 Summary and Conclusion

This report contains the analysis and design of control algorithms for automatic, safe landing with constraint specifications for an F-16 aircraft. The control objective of flying the aircraft along the glide path is formulated as a stabilization problem with constraints. This stabilization problem is then translated to an optimization problem (that is, maximizing the stability region subject to a set of LMI constraints, which characterize the stability criterion, the aircraft state and control constraints, and the pole location specifications). This optimization problem is solved by using the SDPSOL software package developed by Wu and Boyd [Wu 96]. As a result, a set of control gains for the linear state feedback control law and a stability region are obtained, and both of them are ready to be implemented in the real-time simulator.

While the proposed design approach has proven to be feasible in the experiment with the book simulator, it still needs to be tested in the real-time simulator. Among all the possible practical issues, we foresee three that may need to be addressed further. First, the performance of the designed controller may need to be improved. As described in Section 5.2, the baseline controller is designed such that the resulting stability region needs to be maximized. While this would provide the largest region in the state space for the upgraded controller to explore new functionality, it may also make the convergence to the glide path slow when the baseline control is in charge. If this is the case, constraints on the pole location need to be imposed to enforce the minimum decay rate. On the other hand, merely restricting the pole location for a minimum decay rate may cause the closed-loop system to oscillate. When this is not acceptable, the magnitudes of the imaginary part of the poles must be limited as well. All of these can be tested with different specifications of pole location.

Second, the accuracy of the identified model may need to be further improved. Since the identified model is used to represent the actual system, it is always understood that some dynamics of the actual system may not be included in the identified model. Therefore, the actual system can be considered as the combination of the identified model and some uncertain dynamics. While the state feedback control is usually robust in the sense of controlling the system with uncertainties, the stability region built upon the identified model may be problematic. Furthermore, since the internal states, δ_{el}, δ_{ail}, and δ_{rdr}, may not be measurable, an observer may be needed to estimate these states. The accuracy of the observed states, of course, will depend on the precision of the model. Therefore, the accuracy of the identified model will need to be improved if it gives poor state estimates or false results on the safety check against the stability region. (See [Seto 99] for information about the safety check.)

Finally, the non-linearities in the real-time simulator may cause problems in the safety check and system performance evaluation. Although the non-linearities will result in model inaccuracy as mentioned above, they may not be addressed by improving the identified model. In this case, one needs to consider shrinking the stability region to make it more conservative and to experiment with the performance. If the model variation caused by the non-linearities can be characterized by a bounded term, robust control methodology may then be applied.

In summary, the designed controller and the derived stability region should work well in the real-time simulator after some second-order effect is addressed.

Bibliography

[Altman 99] Altman, Neal. *Simplex Architecture*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. Available WWW <URL: http://www.sei.cmu.edu/simplex/simplex_architecture.html> (1999).

[Boyd 94] Boyd, S.; El Ghaoui, L.; Feron, E.; and Balakrishnan, V. “Linear Matrix Inequalities in System and Control Theory.” *SIAM Studies in Applied Mathematics*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 1994.

[Chilali 96] Chilali, M. & Gahinet, P. “$H_{.}$ Design with Pole Placement Constraints: An LMI Approach.” *IEEE Transactions on Automatic Control 41*, 3 (March 1996).

[Feiler 99] Feiler, Peter. “Simplex: A Technology for Rapid, Reliable Upgrade.” *Proceedings of the 1999 SEI Software Engineering Symposium*. Pittsburgh, PA, Aug. 30-Sept. 2, 1999. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999.

[INSERT 00] INSERT Project. *INSERT: Incremental Software Evolution for Real-Time Applications*. Pittsburgh, PA: Carnegie Mellon University, 2000. Available WWW: <URL: http://www.cs.cmu.edu/Groups/real-time/insert> (2000).

[Pollak 98] Pollak, B. “Simplex: Upgrading Software Components Safely and Reliably.” *news@sei 1*, 2 (Summer 1998): 6-8.

[Rowan 90] Rowan, T. *Functional Stability Analysis of Numerical Algorithms*, Ph.D. thesis, Department of Computer Sciences, University of Texas at Austin, 1990.

[Stevens 92] Stevens, B. & Lewis, F. *Aircraft Control and Simulation*, New York, NY: Wiley, 1992.

[Seto 99] Seto, D. & Sha, L. *An Engineering Method for Safety Region Development* (CMU/SEI-99-TR-018, ADA 367624). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999. Available WWW <URL: http://www.sei.cmu.edu/publications/documents/99.reports/99tr018/99tr018abstract.html>.

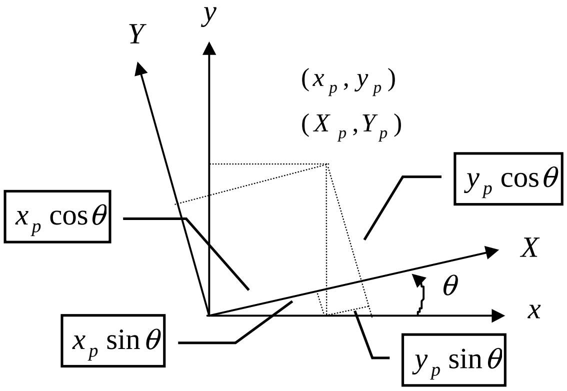
[Sha 97] Sha, L. “Shifting the computation paradigm of real-time control systems.” Invited paper in Real-Time Computing Symposium (SNART97). Lund, Sweden, August 1997.

[Wu 96] Wu, S-P. & Boyd, S. “A parser solver for semi-definite programming determinant maximization problems with matrix structure,” User’s Guide, Stanford University, May 1996.

Appendix A: Relationship Among Coordinate Frames

Coordinate Transformation $(x, y) \to (X, Y)$:

Given a point (x_p, y_p) in the old coordinate frame (x, y), find its expression (X_P, Y_P) in a new coordinate frame (X, Y).



$$X_P = x_p \cos\theta + y_p \sin\theta$$
$$Y_p = -x_p \sin\theta + y_p \cos\theta$$

or

$$\begin{bmatrix} X_p \\ Y_p \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_p \\ y_p \end{bmatrix}$$

Mapping from ECI to NED R_{NE} :

Celestial longitude angle l: the angle measured in the ECI *y*-*z* plane (Earth's equatorial plane), from the negative *z*-axis in a counterclockwise direction when viewed from the North Pole.

Geodetic latitude angle, μ: the angle between the line that is normal to the Earth's surface and passes the point of interest, and the ECI *y*-*z* plane.

These two angles are depicted in Figure 31.

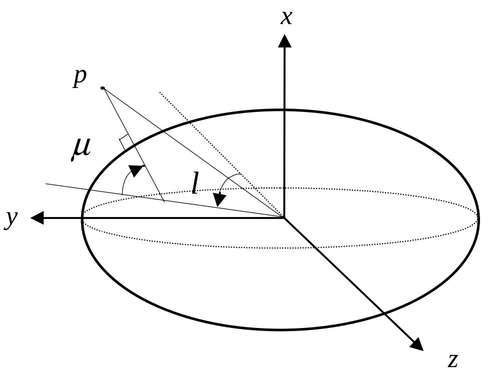


Figure 31: Celestial Longitude Angle and Geodetic Latitude Angle

Let $\mathbf{q}_{\text{NED}}$ and $\mathbf{q}_{\text{ECI}}$ be a vector expressed in the NED frame and in the ECI frame, respectively. Then, $\mathbf{q}_{\text{NED}} = R_{NE}\mathbf{q}_{\text{ECI}}$,

$$R_{NE} = R_{\mu} R_{l}$$

$$= \begin{bmatrix} \cos\mu & 0 & \sin\mu \\ 0 & 1 & 0 \\ -\sin\mu & 0 & \cos\mu \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos l & \sin l \\ 0 & -\sin l & \cos l \end{bmatrix} = \begin{bmatrix} \cos\mu & -\sin\mu\sin l & -\sin\mu\cos l \\ 0 & \cos l & \sin l \\ -\sin\mu & -\cos\mu\sin l & \cos\mu\cos l \end{bmatrix}$$

Mapping from NED to ABC R_{AN} :

Starting from NED,

1. Rotate ψ about the z-axis: yawing with nose right positive.
2. Rotate θ about the new y-axis: pitching with nose up positive.
3. Rotate ϕ about the new x-axis: rolling with right wing down positive, where ϕ, θ and ψ are called the Euler angles. Let $\mathbf{p}_{\text{ABC}}$ and $\mathbf{p}_{\text{NED}}$ be a vector expressed in the ABC and NED frames, respectively. Then, $\mathbf{p}_{\text{ABC}} = R_{AN}\mathbf{p}_{\text{NED}}$,

$$R_{AN} = R_{\phi} R_{\theta} R_{\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix}$$

Mapping from ABC fixed-body axes to ABC-Wind axes R_{WB}:

Starting from the ABC fixed-body frame,

1. Rotate about the *y*-axis for the angle of α .
2. Rotate about the new *z*-axis for the angle of β .

Let $\mathbf{p}_{\text{WIND}}$ and $\mathbf{p}_{\text{BODY}}$ be a vector expressed in the ABC-Wind and ABC fixed-body frames, respectively. Then $\mathbf{p}_{\text{WIND}} = R_{WB}\mathbf{p}_{\text{BODY}}$,

$$
\begin{aligned}
R_{WB} &= R_{\beta} R_{\alpha} \\
&= \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}
\end{aligned}
$$

Appendix B: Derivation of Equations of Motion of an Aircraft

In this section, we derive the equations of motion of an aircraft. Since an aircraft can be treated as a rigid body moving in three-dimensional space, we first review some materials on the motion of a rigid body, and then derive the equations of motion for an aircraft in the context of a rigid body.

B.1 Motion of a Rigid Body

Motion between coordinate systems: Let *A* and *B* be oriented Euclidean spaces, $\mathbf{q}, \mathbf{c} \in A$, and $\mathbf{q}_B \in B$. A motion of *B* relative to *A* is a mapping smoothly depending on *t*:

$$D_t : B \to A,$$

which preserves the metric and the orientation. Every motion D_t can be uniquely written as the composition of a rotation $R_t : B \to A$ and a translation $C_t : A \to A$, where $C_t\mathbf{q}(t) = \mathbf{q}(t) + \mathbf{c}(t)$. Namely,

$$D_t = C_t R_t \quad \Rightarrow \quad \mathbf{q}(t) = D_t \mathbf{q}_B(t) = R_t \mathbf{q}_B(t) + \mathbf{c}(t).$$

A motion D_t is called a rotation if it takes the origin of *B* to the origin of *A*(i.e., if D_t is a linear operator). A motion D_t is called translational if the mapping does not depend on *t* (i.e., $R_t = R_0 = R$ and $D_t \mathbf{q}_B = R\mathbf{q}_B + \mathbf{c}(t)$).

Motion of a vector in different coordinate systems:

$$\mathbf{q}(t) = R_t \mathbf{q}_B(t) + \mathbf{c}(t) \quad \Rightarrow \quad \dot{\mathbf{q}}(t) = \dot{R}_t \mathbf{q}_B(t) + R_t \dot{\mathbf{q}}_B(t) + \dot{\mathbf{c}}(t)$$

$\dot{R}_t \mathbf{q}_B = \boldsymbol{\omega} \times (\mathbf{q} - \mathbf{c}) = \boldsymbol{\omega} \times (R_t \mathbf{q}_B)$: the transferred velocity of rotation

$R_t \dot{\mathbf{q}}_B(t)$: the relative velocity

$\dot{\mathbf{c}}(t)$: the velocity of the origin of the moving coordinate system

Where $\boldsymbol{\omega}$ is the angular velocity of frame *B* with respect to frame *A* expressed in frame *A*, and is derived from a skew-symmetric matrix Ω as

$$\Omega = \dot{R}_t R_t^T = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \Rightarrow \Omega \mathbf{q} = \boldsymbol{\omega} \times \mathbf{q}, \forall \mathbf{q} \in R^3$$

Suppose $\mathbf{c}(t) = 0, \forall t \geq 0$. Let $\boldsymbol{\omega}_B$ be the angular velocity of frame *B* with respect to frame *A* expressed in frame B. Then $\boldsymbol{\omega} = R_t \boldsymbol{\omega}_B$, and the transferred velocity can be further written as

$$\dot{R}_t \mathbf{q}_B = \boldsymbol{\omega} \times \mathbf{q} = (R_t \boldsymbol{\omega}_B) \times (R_t \mathbf{q}_B) = R_t \boldsymbol{\omega}_B \times \mathbf{q}_B .$$

Define a skew-symmetric matrix Ω_B such that $\boldsymbol{\omega}_B \times \mathbf{v} = \Omega_B \mathbf{v}, \forall \mathbf{v} \in R^3$. Then

$$\dot{R}_t \mathbf{q}_B = R_t \Omega_B \mathbf{q}_B \Rightarrow \Omega_B = R_t^T \dot{R}_t .$$

B.2 Derivation of Aircraft Equations of Motion

Expression of a velocity vector in a different coordinate frame: Let *A* and *B* be two coordinate frames and *R* be the rotation matrix mapping $A \to B$. Then we may define a velocity vector as follows:

$\mathbf{v}_A$ with respect to frame *A* expressed in *A*

$\mathbf{v}_A^B$ with respect to frame *A* expressed in *B*

$\mathbf{v}_B$ with respect to frame *B* expressed in *B*

$$\mathbf{v}_A = \dot{\mathbf{p}}_A, \qquad \mathbf{v}_A^B = R\mathbf{v}_A, \qquad \mathbf{v}_B = \frac{d}{dt}(R\mathbf{p}_A) = \boldsymbol{\omega} \times \mathbf{p}_B + R\dot{\mathbf{p}}_A$$

where $\boldsymbol{\omega}$ is the angular velocity of frame *A* with respect to frame *B*, and $\mathbf{p}_A$ and $\mathbf{p}_B$ are the expressions of the position vector in frame A and frame B, respectively.

Example: Let frame *A* and frame *B* be aligned along the *z* axes as shown in Figure 32 (positive *z* points outwards). Suppose *A* is stationary and *B* rotates about the *z*-axis at an angular velocity $\dot{\theta}$. Consider a vector fixed on the *X*-axis of frame *B* with length *r* and expressions $\mathbf{p}_A$ and $\mathbf{p}_B$ in *A* and *B*, respectively. Then,

$$R = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{p}_A = \begin{bmatrix} r\cos\theta \\ r\sin\theta \\ 0 \end{bmatrix}, \qquad \mathbf{p}_B = \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix}$$

where $R : A \to B$. Hence the velocities can be derived as the following:

$$\mathbf{v}_A = \dot{\mathbf{p}}_A = \begin{bmatrix} -r\dot{\theta}\sin\theta \\ r\dot{\theta}\cos\theta \\ 0 \end{bmatrix}, \quad \mathbf{v}_A^B = R\mathbf{v}_A = \begin{bmatrix} 0 \\ r\dot{\theta} \\ 0 \end{bmatrix}$$

$$\mathbf{v}_B = \boldsymbol{\omega} \times \mathbf{p}_B + R\mathbf{v}_A = \begin{bmatrix} 0 & \dot{\theta} & 0 \\ -\dot{\theta} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ r\dot{\theta} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

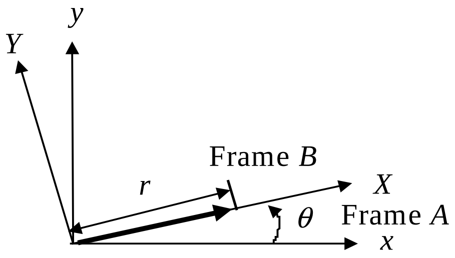


Figure 32: Vector Conversion Between Frames

External forces in the equations

Propulsion

Aerodynamic forces

Gravitational attraction (gravity)

State variables in the equations

Position vector in the inertial frame (ECI): $\mathbf{p} = [p_x,\ p_y,\ p_z]^T$.

Relative velocity of aircraft cg with respect to air mass and expressed in ABC frame: $\mathbf{v}_B = [v_B^x,\ v_B^y,\ v_B^z]^T$.

Absolute angular velocity of ABC frame expressed in ABC: $\boldsymbol{\omega}_B = [\omega_B^x,\ \omega_B^y,\ \omega_B^z]^T$.

Attitude variables - Euler angles $\Phi = [\phi,\ \theta,\ \psi]^T$.

Dynamic Equation-Translational Motion

Let F be the combined propulsion and aerodynamic forces and p be the position vector of the aircraft center of gravity, both expressed in the ECI coordinate frame. Then Newton's second law implies that

$$\mathbf{F} + m\mathbf{g} = \frac{d}{dt}(m\dot{\mathbf{p}}) \;\Rightarrow\; \mathbf{F} + m\mathbf{g} = \dot{m}\dot{\mathbf{p}} + m\ddot{\mathbf{p}}$$

For aircraft, we ignore the change of mass (i.e., $\dot{m} = 0$). To derive an expression for the vector $\dot{\mathbf{p}}$, let RECI be yet another coordinate system, which is centered at the origin of the ECI and coincident with ECI at time $t = 0$, and rotates with the Earth about its x axis. Furthermore, suppose R_{ER} and R_{RA} are the rotation matrices that result in the mapping $R_{ER}: \mathrm{RECI} \to \mathrm{ECI}$ and $R_{RA}: \mathrm{ABC} \to \mathrm{RECI}$. Let R be the rotation matrix mapping: $\mathrm{ECI} \to \mathrm{ABC}$ (i.e., $R = R_{AN}R_{NE}$). It follows that $R^T = R_{ER}R_{RA}$, and we have

$$\mathbf{p} = R_{ER}\mathbf{p}_R \;\Rightarrow\; \dot{\mathbf{p}} = \dot{R}_{ER}\mathbf{p}_R + R_{ER}\dot{\mathbf{p}}_R$$

where $\mathbf{p}_R$ is the position vector expressed in RECI frame. Clearly, the first term $\dot{R}_{ER}\mathbf{p}_R$ is the transferred velocity of the RECI's rotation (see Section B.1 for the definition of transferred velocity), and it can be written as $\dot{R}_{ER}\mathbf{p}_R = \boldsymbol{\omega}_E \times \mathbf{p}$ with $\boldsymbol{\omega}_E$ the absolute angular velocity of the Earth's rotation expressed in the ECI frame. The other term $R_{ER}\dot{\mathbf{p}}_R$ is the relative velocity that can be expressed as $R_{ER}\dot{\mathbf{p}}_r = R_{ER}(R_{RA}\mathbf{v}_B) = R^T\mathbf{v}_B$. Finally the equation of $\dot{\mathbf{p}}$ can be written as follows:

$$\dot{\mathbf{p}} = \boldsymbol{\omega}_E \times \mathbf{p} + R^T\mathbf{v}_B$$

Differentiating $\dot{\mathbf{p}}$, we obtain the following:

$$\ddot{\mathbf{p}} = \dot{\boldsymbol{\omega}}_E \times \mathbf{p} + \boldsymbol{\omega}_E \times \dot{\mathbf{p}} + \dot{R}^T\mathbf{v}_B + R^T\dot{\mathbf{v}}_B$$

Since the Earth is rotating at a constant angular velocity, $\dot{\boldsymbol{\omega}}_E = 0$. We also realize that the third term, $\dot{R}^T\mathbf{v}_B$, is the transferred acceleration due to the rotation of the ABC frame. Hence, $\dot{R}^T\mathbf{v}_B = (R^T\boldsymbol{\omega}_B)\times(R^T\mathbf{v}_B) = R^T(\boldsymbol{\omega}_B \times \mathbf{v}_B)$, where $R^T\boldsymbol{\omega}_B$ and $R^T\mathbf{v}_B$ are the ABC angular velocity and aircraft cg's relative velocity with respect to the air, both expressed in the ECI coordinate frame. Then the equation of $\ddot{\mathbf{p}}$ becomes

$$\ddot{\mathbf{p}} = \boldsymbol{\omega}_E \times (\boldsymbol{\omega}_B \times \mathbf{p} + R^T\mathbf{v}_B) + R^T(\boldsymbol{\omega}_B \times \mathbf{v}_B) + R^T\dot{\mathbf{v}}_B$$

and the equation of Newton's second law can be written as

$$\mathbf{F} + m\mathbf{g} = m[\boldsymbol{\omega}_E \times (\boldsymbol{\omega}_B \times \mathbf{p} + R^T\mathbf{v}_B) + R^T(\boldsymbol{\omega}_B \times \mathbf{v}_B) + R^T\dot{\mathbf{v}}_B]$$

Let $\mathbf{F}_B$ be the force F expressed in ABC coordinate frame(i.e., $\mathbf{F}_B = R\mathbf{F}$). Then the above dynamic equation can be written in ABC by pre-multiplying matrix R as

$$\begin{aligned} \mathbf{F}_B + Rm\mathbf{g} &= m[R\boldsymbol{\omega}_E \times (\boldsymbol{\omega}_B \times \mathbf{p} + R^T \mathbf{v}_B) + \boldsymbol{\omega}_B \times \mathbf{v}_B + \dot{\mathbf{v}}_B] \\ &= m[R\boldsymbol{\omega}_E \times (\boldsymbol{\omega}_B \times \mathbf{p}) + (R\boldsymbol{\omega}_E) \times \mathbf{v}_B + \boldsymbol{\omega}_B \times \mathbf{v}_B + \dot{\mathbf{v}}_B] \end{aligned}$$

Finally, we obtain

$$\boxed{\dot{\mathbf{v}}_B = \frac{\mathbf{F}_B}{m} - (\boldsymbol{\omega}_B + R\boldsymbol{\omega}_E) \times \mathbf{v}_B + R[\mathbf{g} - \boldsymbol{\omega}_E \times (\boldsymbol{\omega}_B \times \mathbf{p})]}$$

Dynamic Equation-Angular Motion

Let M be the angular momentum and T be the total torque acting about the aircraft cg, both expressed in the ECI coordinate frame. Then Newton's second law implies

$$\underline{\dot{\mathbf{M}} = \mathbf{T}}$$

Since the angular momentum of the aircraft in the ABC coordinate can be expressed as $J\boldsymbol{\omega}_B$, where J is the moment of inertial of the aircraft, then $\mathbf{M} = R^T J\boldsymbol{\omega}_B$ and $\dot{\mathbf{M}} = \dot{R}^T J\boldsymbol{\omega}_B + R^T J\dot{\boldsymbol{\omega}}_B$. Again, the first term is recognized as the transferred momentum and it can be written as $\dot{R}^T J\boldsymbol{\omega}_B = (R^T\boldsymbol{\omega}_B) \times \mathbf{M}_I = R^T(\boldsymbol{\omega}_B \times (J\boldsymbol{\omega}_B))$. Then the dynamic equation can be rewritten as follows:

$$\mathbf{T} = R^T(\boldsymbol{\omega}_B \times (J\boldsymbol{\omega}_B)) + R^T J\dot{\boldsymbol{\omega}}_B$$

Let $\mathbf{T}_B$ be the ABC expression of T (i.e., $\mathbf{T}_B = R\mathbf{T}$). Then the dynamic equation can be expressed in ABC coordinate as follows:

$$\boxed{\dot{\boldsymbol{\omega}}_B = -J^{-1}[\boldsymbol{\omega}_B \times (J\boldsymbol{\omega}_B)] + J^{-1}\mathbf{T}_B}$$

Kinematics Equations

Three-variable attitude equation:

Define the Euler angles $[\phi, \theta, \psi]$ between the ECI frame and the ABC frame, and let R_{AE} be the rotation matrix mapping from ECI to ABC. Then,

$$R_{AE} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix}$$

and

$$\Omega_B = R_{AE}\dot{R}_{AE}^T = -\dot{R}_{AE}R_{AE}^T \;\Rightarrow\; \dot{R}_{AE} = -\Omega_B R_{AE}$$

where Ω_B is the skew-symmetric matrix such that $\boldsymbol{\omega}_B \times \mathbf{q} = \Omega_B \mathbf{q}$. By evaluating the elements (1,2), (1,3), and (2,3) in the above equation, we obtain the following:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} \omega_B^x \\ \omega_B^y \\ \omega_B^z \end{bmatrix}$$

Four-variable attitude equation:

The Euler angles can be replaced by the so-called quaternion four-variable representation, $\mathbf{q} = [q_0, q_1, q_2, q_3]^T$. Then the rotation matrix R_{AE} from the ECI frame to the ABC frame can be expressed in terms of q as follows:

$$R_{AE} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

and the relation between q and $\mathbf{\Phi}$ is given by the following:

$$\phi = \tan^{-1}\frac{b_{23}}{b_{33}}, \quad \theta = -\sin^{-1} b_{13}, \quad \psi = \tan^{-1}\frac{b_{12}}{b_{11}}$$

with b_{ij} is the element in the *i*th row and *j*th column of R_{AE}, and

$$\begin{aligned} q_0 &= \pm[\cos(\phi/2)\cos(\theta/2)\cos(\varphi/2) + \sin(\phi/2)\sin(\theta/2)\sin(\varphi/2)] \\ q_1 &= \pm[\sin(\phi/2)\cos(\theta/2)\cos(\varphi/2) - \cos(\phi/2)\sin(\theta/2)\sin(\varphi/2)] \\ q_2 &= \pm[\cos(\phi/2)\sin(\theta/2)\cos(\varphi/2) + \sin(\phi/2)\cos(\theta/2)\sin(\varphi/2)] \\ q_3 &= \pm[\cos(\phi/2)\cos(\theta/2)\sin(\varphi/2) - \sin(\phi/2)\sin(\theta/2)\cos(\varphi/2)] \end{aligned}$$

Therefore, the kinematics equations in terms of q are given by the following:

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = -\frac{1}{2} \begin{bmatrix} 0 & \omega_B^x & \omega_B^y & \omega_B^z \\ -\omega_B^x & 0 & -\omega_B^z & \omega_B^y \\ -\omega_B^y & \omega_B^z & 0 & -\omega_B^x \\ -\omega_B^z & -\omega_B^y & \omega_B^x & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

B.3 Summary of Modelling

The Round-Earth Equations in ECI are as follows:

$$
\begin{aligned}
\dot{\mathbf{p}} &= \Omega_E \mathbf{p} + R_{AE}^T \mathbf{v}_B \\
\dot{\mathbf{v}}_B &= -R_{AE}\Omega_E^2 \mathbf{p} - (\Omega_B + R_{AE}\Omega_E R_{AE}^T)\mathbf{v}_B + R_{AE}\mathbf{g}(\mathbf{p}) + \frac{\mathbf{F}_B}{m} \\
\dot{\boldsymbol{\omega}}_B &= -J^{-1}\Omega_B J \boldsymbol{\omega}_B + J^{-1}\mathbf{T}_B \\
\dot{\mathbf{q}} &= -\frac{1}{2}\Omega_q\, \mathbf{q}
\end{aligned}
$$

where

$$
R_{AE} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix},
$$

$$
\Omega_E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\omega_x \\ 0 & \omega_x & 0 \end{bmatrix},\ \Omega_B = \begin{bmatrix} 0 & -\omega_B^z & \omega_B^y \\ \omega_B^z & 0 & -\omega_B^z \\ -\omega_B^y & \omega_B^x & 0 \end{bmatrix},\ \Omega_q = \begin{bmatrix} 0 & \omega_B^x & \omega_B^y & \omega_B^z \\ -\omega_B^x & 0 & -\omega_B^z & \omega_B^y \\ -\omega_B^y & \omega_B^z & 0 & -\omega_B^x \\ -\omega_B^z & -\omega_B^y & \omega_B^x & 0 \end{bmatrix},
$$

ω_x is the x-component of the absolute angular velocity of the Earth in ECI, $\mathbf{F}_B$ and $\mathbf{T}_B$ are the resultant force and torque of propulsion and aerodynamic forces, expressed in the ABC frame, and g(p) is the gravitational attraction.

The Flat-Earth Equations in NED

In this case, we also ignore the Earth's rate (i.e., $\boldsymbol{\omega}_E = 0$). Then the equations become

$$
\begin{aligned}
\dot{\mathbf{p}}_{\text{NED}} &= R_{AN}^T \mathbf{v}_B \\
\dot{\mathbf{v}}_B &= -\Omega_B \mathbf{v}_B + R_{AN}\mathbf{g}_0^{'} + \frac{\mathbf{F}_B}{m} \\
\dot{\boldsymbol{\omega}}_B &= -J^{-1}\Omega_B J \boldsymbol{\omega}_B + J^{-1}\mathbf{T}_B \\
\dot{\boldsymbol{\Phi}} &= \Psi(\boldsymbol{\Phi})\boldsymbol{\omega}_B
\end{aligned}
$$

where

$$
\Psi(\boldsymbol{\Phi}) = \begin{bmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix},\quad \mathbf{g}_0^{'} = \begin{bmatrix} 0 \\ 0 \\ 32.17\,\text{ft/s}^2 \end{bmatrix},\quad \mathbf{p}_{\text{NED}} = \begin{bmatrix} p_N \\ p_E \\ h \end{bmatrix}
$$

and the Euler angles are defined between the NED frame and the ABC frame.

Appendix C: Data Collection / Protection Methodology

The legacy F-16 manned virtual desktop simulation mentioned in the Introduction is an unclassified, but International Tariffs in Arms Regulations (ITARS)-restricted item. This meant that non-U.S. nationals could not have access to the artifact, which presented some difficulties in the collection and analysis of the data. This appendix outlines the procedures that the team employed to protect the information represented in the simulation artifact.

The simulation is a reduced order F-16 aircraft model running in a soft real-time desktop computational environment. This simulation is assumed by the authors to be representative of F-16 flight characteristics (indeed, of the entire class of fighter aircraft), but it should not be considered a ground truth information source for the F-16 aircraft. The simulation was not certified as correct by any process, and no flight test evaluation by any qualified pilot was performed. Any relationship between data collected from this simulation and real-world ground truth is therefore unknown.

In addition, the flight envelope inspected by this process has no tactical significance to F-16 flight operations. All data were collected in a "gear down" landing configuration. The simulation was initialized with the aircraft within 3 degrees laterally of the runway localizer, at 1120 feet AGL altitude (3400 feet MSL altitude), and a velocity of 218 KCAS.[2] The aircraft was hand-flown by engineering personnel from this point to touchdown.

Data were extracted from the model at a 50 Hz rate. The extracted data included the horizontal and vertical glide-path deviations, aircraft Euler angles and rates, altitude, airspeed, vertical velocity, angle-of-attack, pilot control inputs and control surface positions.

A U.S. national was the identifier, operator, and data collector for this system. At no time did non-U.S. nationals have unrestricted access to either the source code or executable simulation artifacts. Once collected, the data were transferred to other computers for analysis and model identification.

[2] The acronyms AGL, MSL, and KCAS are defined as follows: AGL is above ground level; MSL is mean sea level; and KCAS is knots calibrated air speed.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (LEAVE BLANK)	2. REPORT DATE **May 2000**	3. REPORT TYPE AND DATES COVERED Final

4. TITLE AND SUBTITLE
Case Study: Development of a Baseline Controller for Automatic Landing of an F-16 Aircraft Using Linear Matrix Inequalities (LMIs)

5. FUNDING NUMBERS
C — F19628-95-C-0003

6. AUTHOR(S)
Danbing Seto, Enrique Ferreira, Theodore F. Marz

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

8. PERFORMING ORGANIZATION REPORT NUMBER
CMU/SEI-99-TR-020

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
HQ ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2116

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ESC-TR-99-020

11. SUPPLEMENTARY NOTES

12.A DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified/Unlimited, DTIC, NTIS

12.B DISTRIBUTION CODE

13. ABSTRACT (MAXIMUM 200 WORDS)

In this report, we present preliminary results on the design of the baseline controller for an F-16 aircraft automatic landing system using linear matrix inequalities (LMI)-based approaches. We start with a general study of aircraft control and dynamics to gain knowledge of the structure of an aircraft dynamic model and its inner loop control system. We then identify a linear model along the glide path for the inner loop control system in the simulator. With this linear model, the control objective is to solve a stabilization problem—stabilizing the aircraft along the glide path using linear state feedback controls. Expressing the stability criterion and the constraints in LMIs, we cast the stabilization problem as an optimization problem. Using the SDPSOL software package developed by Wu and Boyd, we solve this optimization problem for the control gain and stability region, which completes the controller design [Wu 96].

14. SUBJECT TERMS **automatic landing system, baseline controller, F-16 aircraft, INSERT project, linear matrix inequalities (LMIs), SDPSOL, Simplex^{TM} architecture**

15. NUMBER OF PAGES
53

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL