Abstract

Peterson, Joseph Scott, Design Study of an Autonomous Unmanned Air Vehicle Controller. (Under the direction of Dr. Charles Hall) The purpose of this document is to both quantitatively and qualitatively compare three varying approaches to control of unmanned air vehicles: dynamic inversion, classical gain scheduling, and robust gain scheduling through H_∞ synthesis. The quantitative comparisons include robust performance and robust stability measures of the aircraft and controller linearized about a trim operating point. A second study will look at the time response of the system with varying perturbation to the nominal plant dynamics. The qualitative analysis looks at the complexity of the controller and time required to implement, this included comparison of iterative solving methods for system gains both in the frequency and time domains. A full nonlinear form of the dynamic inversion controller was implemented with full non-linear simulation in Simulink and shows comparable results to the linearized studies. The results have shown that a dual loop controller with a dynamic inversion inner loop and H_∞ outer loop has the largest robust performance over the flight envelope but one of the more advanced forms of control for implementation.

Design Study of an Autonomous Unmanned Air Vehicle Controller

by

Joseph Scott Peterson

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

AEROSPACE ENGINEERING

Raleigh

2005

APPROVED BY:

_____ _____

_____

Chair of Advisory Committee

Dedication

Dedicated to my parents, who have provided me with support throughout my college career, and to my wife for both financial and emotional support during my studies.

Biography

Joseph Scott Peterson was born on October 24, 1981 in Columbus Ohio. He moved to Charlotte, North Carolina in 1995 when his father took a job with then First Union Bank. In August 2000, he started his undergraduate work at North Carolina State University. He married Rachele Shannon in August 2003. In May 2004, he graduated Cum Laude with a Bachelor of Science degree in Aerospace Engineering. He then interned at DRS Unmanned Technologies Inc., in Texas during summer 2004, before returning to NCSU to begin work on his Master of Science degree in Aerospace Engineering.

In December 2005, Mr. Peterson will start work with Analex Corporation in Cleveland, Ohio. He will be working in the Navigation and Controls Group on the verification and validation of Boeing and Lockheed's heavy lifter rocket missions.

Acknowledgments

I would like to sincerely thank those who have provided guidance and assistance during the course of this research:

- Dr. Charles Hall;
- Dr. Fen Wu;
- Dr. Sharon Lubkin; and,
- Mr. Stearns Heinzen.

Table of Contents

List of Figures

Nomenclature

A^T:	transpose of A
A^*:	complex conjugate transpose of A
g:	gravitational acceleration
I_x:	moment of inertia about the x-axis
I_y:	moment of inertia about the y-axis
I_z:	moment of inertia about the z axis
I_{xz}:	product of inertia about the xz plane
Ker(A)	Kernel or null space of A
L:	total rolling moment about the x-axis
M:	total pitching moment about the y-axis
N:	total yawing moment about the z-axis
p:	angular roll rate along the body frame x-axis
q:	angular pitch rate along the body frame y-axis
r:	angular yaw rate along the body frame z-axis
u:	input vector of the system in state space form
u:	forward velocity component of the aircrafts speed
v:	side velocity component of the aircrafts speed
w:	vertical velocity component of the aircrafts speed
X:	total force applied along the x-axis
x:	state vector of the system in state space form
x_d:	desired state vector response of the system
Y:	total force applied along the y-axis
y:	output vector of the system in state space form
Z:	total force applied along the z-axis
α:	angle of attack
β:	angle of side slip
Δ:	diagonal perturbation matrix
δa:	aileron deflection angle
δe:	elevator deflection angle
δr:	rudder deflection angle

ζ:	dampening ratio
θ:	pith Euler angle about the y-axis
η:	perturbation size
$\lambda(A)$:	eigenvalue of A
$\rho(A)$:	spectral radius of A
$\sigma_i(A)$:	ith singular value of A
$\overline{\sigma}(A)$ and $\underline{\sigma}(A)$:	largest and smallest singular value of A
ϕ:	roll Euler angle about the x-axis
ψ:	side slip Euler angle about the z-axis
ω_n:	natural frequency

1. Introduction

Technological advancements over the years, thereby increasing computational power on smaller and smaller platforms, have led to the use of more advanced control laws for autopilot and fly-by-wire systems on both manned and unmanned aerial vehicles (UAV). The driving goal has been to expand the operational envelope of the aircraft while maintaining a robust form of stability and performance requirements. At the same time, the synthesis of such a controller must be efficient in architecture and quickly systematized with today's technology.

The objective of this thesis is to present several methods for design of flight controllers for a UAV over a large flight envelope and compare the performance of these controllers with the difficulty of synthesis and implementation. In this study, feedback linearization techniques, also referred to as dynamic inversion (DI), are used to negate the nonlinear dynamics associated with aircraft over large flight envelopes.

Aircraft simulation and the plant models were based on the Aerosonde UAV. The Aerosonde Robotic Aircraft was first constructed in 1997 with the sponsorship of the US Office of Naval Research. The program's particular aim has been meteorological observations in remote and otherwise inaccessible regions. Aerosonde "Laima" was the first unmanned aircraft to cross the north Atlantic in August of 1998[1]. The aircraft is now on permanent display in the Seattle Air and Space Museum.

Figure 1-1: The Global Robotics Observation System developed by Aersonde

2. Dynamic Inversion

Dynamic inversion has been used in numerous aerospace applications for controllers operating in highly nonlinear flight, such as multiple effector arrays, super-maneuverability, and post stall. Nonlinear input behavior of the plant is canceled by feedback from the inversion of the nonlinear model, hence the term for this type of controller: linearization feedback[2]. This closed loop results in a reduced system of integrators. The approach works for both single input, single output (SISO) and multiple input multiple output (MIMO) systems - provided the system's control matrix is invertible.

With an aircraft, this control law is derived from directly solving for the required control surface deflections given desired angular and/or translational rates. The advantage of this approach over gain scheduling is that a unique set of equations only need be solved once for the entire nonlinear model. The classic gain scheduling approach would require a set of linearized cases to be obtained and then gain values solved independently for each linearization.

One fundamental assumption to this approach is that the plant dynamics are perfectly known and are canceled exactly. Since this condition is unrealistic of the physical plant, some form of outer loop control is required to suppress the undesired behavior due to plant uncertainty and in order to improve the closed loop system robustness.

For a general aircraft the rigid-body equations of motion can be formed in a state space representation by applying small perturbations about a trimmed operating point. This leads to the following mathematical form where F is input affine in u.

$$\begin{aligned} \dot{x} &= F(x,u) \\ y &= H(x) \end{aligned} \qquad \textbf{Eqn. 2-1}$$

where x is the state vector, u is the control vector, and y is the output vector. The nonlinear representation of the equations of motion can be constructed by separating the state and control effectiveness functions, f and g respectively. Here both f and g are nonlinear in form:

$$\dot{x} = f(x) + g(x)u \qquad \textbf{Eqn. 2-2}$$

As long as *g(x)* is invertible for all values of *x*, then a control law can be obtained by solving the equation above for *u*.

$$u = g(x)^{-1}\left[\dot{x} - f(x)\right] \qquad \textbf{Eqn. 2-3}$$

Once in this form, the control necessary for achieving the desired system response can be indirectly solved by substituting the state rates with the rate of the desired states.

$$u = g(x)^{-1}\left[\dot{x}_d - f(x)\right] \qquad \textbf{Eqn. 2-4}$$

When the system is closed using this control law, the resulting system is characterized by $\dot{x} = \dot{x}_d$. Because the control function essentially inverts the nonlinear plant, to guarantee internal stability the plant must be minimum phase. Furthermore, assuming *g(x)* is invertible is not a sufficient condition because if norm of *g(x)* were small, then the control input *u* would become large and could lead to actuator saturation. The simplest way to illustrate this is by assuming equation 2-4 describes a SISO system. As *g(x)* approaches zero, *u* approaches infinity.

2.1. Application to a Linear System

The in-flight motion of an aircraft can be extremely complicated. The main six degrees of freedom, three translational and three rotational, as well as numerous elastic degrees of freedom, lead to a highly nonlinear dynamic model. In addition, all of the force and moment derivatives vary in time based on dynamic pressure and change in mass with fuel consumption. While some have explored advanced techniques for analyzing the complete nonlinear system[3], this process is beyond the scope of this work.

To assess the closed system robustness using structured singular value decomposition techniques, the model must first be transformed to a linear time invariant system. Using Simulink and the complete nonlinear model presented in section 5, an array of trim conditions were solved covering a wide range of the

flight envelope. Secondly, the full 6-DOF model was separated into two parts where the X-force, Z-force, and pitching moment embody the longitudinal equations and the Y-force, roll, and yawing moments form the lateral equations. This separation is allowable as long as the two parts are not coupled, a reasonable assumption provided the flight path is not large in amplitude or does not involve rapid maneuvers for this aircraft.

From this point, a set of ordinary linear differential equations with constant coefficients can be formed. At each equilibrium point only small deviations were made allowing the use of small angle approximation. It should also be noted that the final form is only a first order approximation and ignores higher dynamics such as structural modes and vibration. The dynamics will later be added as uncertainty when doing performance analysis, see section 4.2.3.

2.2. Longitudinal Control

Because most actuation surfaces result in changing moments, only the three momentum equations are inverted to solve the required control law for command surface deflections. The Aerosonde UAV configuration allows for decoupling of the lateral and longitudinal states simplifying the linear time invariant longitudinal case to a single inversion of the equation for pitching moment.

$$\dot{q} = f(\alpha, \dot{\alpha}, q) + M_{\delta e}\delta e$$

$$f(\alpha, \dot{\alpha}, q) = M_{\alpha}\alpha + M_{\dot{\alpha}}\dot{\alpha} + M_{q}q \qquad \textbf{Eqn. 2-5}$$

Previous control laws have been developed using this wind axis[4]; however, to avoid the use of costly air data booms and low sampling frequencies associated with such devices, an inertial measurement unit, which can monitor aircraft angular rates and translational accelerations, will be used. This results in the following conversion:

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \text{ and } \dot{\alpha} = \frac{u\dot{w} - w\dot{u}}{u^2 - w^2} \qquad \textbf{Eqn. 2-6}$$

After applying the small angle assumption and the associated trim conditions for the linearized system, equation 2-7 shows the resulting solution for pitching moment.

$$\dot{q} = f(u, w, q) + (M_{\delta e} + M_{\dot{w}} Z_{\delta e})\delta e$$

$$f(u, w, q) = (M_u + M_{\dot{w}} Z_u)u + (M_w + M_{\dot{w}} Z_w)w + (M_q + M_{\dot{w}} u_0)q$$

Eqn. 2-7

where u_0 is the trim forward velocity of the aircraft. Looking at the form expressed by equation 2-2 the above equation can be mapped as follows: the states, x, are represented by q, u, and w; the control inputs, u, by δe; the state matrix $f(x)$ by $f(u, w, q)$; and the control matrix $g(x)$ by $M_{\delta e} + M_{\dot{w}} Z_{\delta e}$. For a linear time invariant system $g(x)$ is a constant; therefore, the inverse of the control function is always obtainable as $1/g$. As shown in equation 2-3, the system variables can be rearranged to solve for the elevator control input.

$$\delta e = \frac{1}{(M_{\delta e} + M_{\dot{w}} Z_{\delta e})}\left[\dot{q} - (M_u + M_{\dot{w}} Z_u)u - (M_w + M_{\dot{w}} Z_w)w - (M_q + M_{\dot{w}} u_0)q\right]$$

Eqn. 2-8

To construct a control law from the above form, desired pitch acceleration is substituted for the actual response, and the remaining variables become the measured system response.

$$\delta e_{cmd} = \frac{1}{(M_{\delta e} + M_{\dot{w}} Z_{\delta e})}\left[\dot{q}_d - (M_u + M_{\dot{w}} Z_u)u_m - (M_w + M_{\dot{w}} Z_w)w_m - (M_q + M_{\dot{w}} u_0)q_m\right]$$

Eqn. 2-9

In this simple case, both nonlinear and higher order terms in the actual aircraft dynamics were neglected, see equation 2-5. For this reason the controller cannot completely cancel out all undesirable aircraft dynamics. Secondly, the command elevator input does not directly equal the elevator response due to limitations in the control surface position, rate and actuator model. As mentioned earlier, as $M_{\delta e}$ becomes small, the commanded elevator angle becomes unbounded. Finally,

due to sensor noise in the measured angular rates, accelerations may further degrade the controller's performance.

2.3. Lateral Directional Control

The same type of system linearization that was done for the longitudinal case can also be done on the lateral equations of motion. However, instead of a SISO system as was the case before, there are two moment equations and two coupled control surfaces resulting in a multiple input multiple output (MIMO) system. The ailerons and rudder are used to control the roll and yaw rates of the aircraft respectively. Simplified linear equations for the roll and yaw moments can be represented as:

$$\begin{bmatrix} \dot{p} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L_\beta & L_p & L_r \\ N_\beta & N_p & N_r \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \end{bmatrix} + \begin{bmatrix} L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \end{bmatrix} \begin{bmatrix} \delta a \\ \delta r \end{bmatrix} \qquad \textbf{Eqn. 2-10}$$

In this form the aircraft is assumed to have a zero or negligible product of inertia allowing for further simplification and decoupling of angular rates. The mappings for the lateral case are shown below.

$$x = \begin{bmatrix} \beta \\ p \\ r \end{bmatrix}$$

$$u = \begin{bmatrix} \delta a \\ \delta r \end{bmatrix}$$

$$f = \begin{bmatrix} L_\beta & L_p & L_r \\ N_\beta & N_p & N_r \end{bmatrix}$$

$$g = \begin{bmatrix} L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \end{bmatrix}$$

Notice that g is a square matrix; therefore, in general the inverse exists if g is nonsingular. Now following the same procedure used for the longitudinal case, the rate dynamics are inverted and equation 2-10 is solved for the control deflections.

$$\begin{bmatrix} \delta a \\ \delta r \end{bmatrix} = \begin{bmatrix} L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \dot{p} \\ \dot{r} \end{bmatrix} - \begin{bmatrix} L_\beta & L_p & L_r \\ N_\beta & N_p & N_r \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \end{bmatrix} \right\} \qquad \textbf{Eqn. 2-11}$$

Since all the terms associated with the control matrix are constant for a linear time invariant system, the matrix inversion lemma can be applied. By replacing the angular accelerations with desired terms and converting the states to measured values, the lateral control law is formed.

$$\begin{bmatrix} \delta a \\ \delta r \end{bmatrix}_{cmd} = \frac{\begin{bmatrix} N_{\delta r} & -L_{\delta r} \\ -N_{\delta a} & L_{\delta a} \end{bmatrix}}{L_{\delta a} N_{\delta r} - L_{\delta r} N_{\delta a}} \left\{ \begin{bmatrix} \dot{p} \\ \dot{r} \end{bmatrix}_d - \begin{bmatrix} L_\beta & L_p & L_r \\ N_\beta & N_p & N_r \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \end{bmatrix}_m \right\} \qquad \textbf{Eqn. 2-12}$$

2.4. Application to a Nonlinear System

The previous control law derivations are of a simplified form for the three axis moment equations. This approach can extend to nonlinear representations of an aircraft such as the form in the equations below.

$$\begin{aligned} \dot{p} &= \frac{I_z L + I_{xz} N}{I_x I_z - I_{xz}^2} + \frac{I_{xz}\left(I_x - I_y + I_z\right)pq}{I_x I_z - I_{xz}^2} + \frac{\left[I_z\left(I_y - I_z\right) - I_{xz}^2\right]qr}{I_x I_z - I_{xz}^2} \\ \dot{q} &= I_y^{-1}\left[M + \left(I_z - I_x\right)pr + I_{xz}\left(r^2 - p^2\right)\right] \\ \dot{r} &= \frac{I_{xz} L + I_x N}{I_x I_z - I_{xz}^2} - \frac{I_{xz}\left(I_x - I_y + I_z\right)qr}{I_x I_z - I_{xz}^2} + \frac{\left[I_x\left(I_y - I_z\right) + I_{xz}^2\right]pq}{I_x I_z - I_{xz}^2} \end{aligned} \qquad \textbf{Eqn. 2-13}$$

While nonlinearities have now been introduced, for the time being we will assume that the total longitudinal and lateral moments are linear with respect to all aerodynamic derivatives. The assumption allows us to write the linear moment equations in the form:

$$\begin{aligned} L &= L_\beta \beta + L_p p + L_r r + L_{\delta a}\delta a + L_{\delta r}\delta r \\ M &= M_\alpha \alpha + M_{\dot{\alpha}}\dot{\alpha} + M_q q + M_{\delta e}\delta e \\ N &= N_\beta \beta + N_p p + N_r r + N_{\delta a}\delta a + N_{\delta r}\delta r \end{aligned} \qquad \textbf{Eqn. 2-14}$$

Because the aircraft will be using an inertial measurement unit for sensory input, the angle of attack and angle of side slip are expressed respectively: $\alpha = \tan^{-1}\left(\frac{w}{u}\right)$

and $\beta = \tan^{-1}\left(\frac{v}{u}\right)$. For the time being, though, we will leave expressions in terms of α and β. Substituting equations 2-14 into 2-13 the states can be separated from the control variables; in addition, the nonlinear terms can be separated from the linear.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & L_\beta & L_p & 0 & L_r \\ M_\alpha & 0 & 0 & M_q & 0 \\ 0 & N_\beta & N_p & 0 & N_r \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ p \\ q \\ r \end{bmatrix} + \begin{bmatrix} 0 & L_{\delta a} & L_{\delta r} \\ M_{\delta e} & 0 & 0 \\ 0 & N_{\delta a} & N_{\delta r} \end{bmatrix} \begin{bmatrix} \delta e \\ \delta a \\ \delta r \end{bmatrix}$$

$$+ \frac{\begin{bmatrix} I_y I_z & 0 & I_y I_{xz} \\ 0 & I_x I_y - I_{xz}^2 & 0 \\ I_y I_{xz} & 0 & I_x I_z \end{bmatrix}}{\Gamma} \begin{bmatrix} pq \\ I_{xz}\left(r^2 - p^2\right) + \left(I_z - I_x\right)pr \\ qr \end{bmatrix}$$

Eqn. 2-15

Where $\Gamma = I_x I_y I_z - I_y I_{xz}^2$. Note the control matrix is square to satisfy the first condition of matrix inversion. Also note that if the last part were ignored the result would be identical to the linear set of dynamic inversion equation obtained in sections 2.2 and 2.3. As was done in the previous sections, the control law is obtained by solving for the control vector and substituting the states with measurable response. In addition to the state vector, the nonlinear portion will be substituted with measured values.

$$\begin{bmatrix} \delta e \\ \delta a \\ \delta r \end{bmatrix}_{cmd} = \begin{bmatrix} 0 & L_{\delta a} & L_{\delta r} \\ M_{\delta e} & 0 & 0 \\ 0 & N_{\delta a} & N_{\delta r} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}_d - \begin{bmatrix} 0 & L_\beta & L_p & 0 & L_r \\ M_\alpha & 0 & 0 & M_q & 0 \\ 0 & N_\beta & N_p & 0 & N_r \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ p \\ q \\ r \end{bmatrix}_m \right.$$

$$\left. - \frac{\begin{bmatrix} I_y I_z & 0 & I_y I_{xz} \\ 0 & I_x I_y - I_{xz}^2 & 0 \\ I_y I_{xz} & 0 & I_x I_z \end{bmatrix}}{\Gamma} \begin{bmatrix} pq \\ I_{xz}\left(r^2 - p^2\right) + \left(I_z - I_x\right)pr \\ qr \end{bmatrix} \right\} \qquad \textbf{Eqn. 2-16}$$

2.5. Concerns with Dynamic Inversion

The main issues with the application of the dynamic inversion process deal with the method and accuracy of the matrix inversion. First, the inversion of the control effectiveness matrix must exist; secondly, it must not be near singularity. Depending on the complexity of the physical system, is the inversion accurate enough to maintain stability or performance despite error? This leads to the issue of designing an outer loop and what form this outer loop should take. Finally there is the issue of saturation and ensuring the inversion process does not require more than what the control effectors can provide.

3. Outer Loop Control Structures

The inversion laws obtained form the aircraft rates in section 2.2 and 2.3 will be linked with the plant to form the closed inner loop. Due to the nature of the inversion, a complete knowledge of the aircraft must be known for perfect command following. For many cases this accuracy is unobtainable given the higher order responses that are ignored and environmental disturbances. Secondly, it may prove computationally impossible to simulate the aircraft inversion and still have adequate actuator response. For this reason a second loop is added to guarantee robust stability and performance despite these errors in system modeling.

3.1.1. Classic Gain Control

Since the rate commands are needed for the dynamic inversion control law, a command transformation block is required to change commands into desired rate dynamics. This section describes how this transformation can be done using simple gains.

The first step to finishing the control laws depicted in equations 2-9 and 2-12 is to define the desired angular accelerations. This is achieved supplying a rate command and then developing an acceleration term by finding the difference of this rate command with the measured value.

$$\begin{aligned} \dot{p}_d &= \omega_p \left(p_{cmd} - p_m \right) \\ \dot{q}_d &= \omega_q \left(q_{cmd} - q_m \right) \\ \dot{r}_d &= \omega_r \left(r_{cmd} - r_m \right) \end{aligned} \qquad \textbf{Eqn. 3-1}$$

The gains – ω_p, ω_q, and ω_r – are given to limit the rate of the angular dynamics and prevent saturation of the control surface. Now an outer loop is created to solve for the command rates and ensure robust performance.

The simplest form is a first order feedback response or proportional control. This expression is similar to equation 3-1 where ω limits the bandwidth of the response, preventing saturation of the control surfaces. This gain value must be

selected such that performance requirements can be met and the response will not excite higher order structural modes or allow control surface saturation. In transfer function form this controller is essentially placing a pole at the location -ω.

$$\frac{u_m}{u_{cmd}} = \frac{\omega_x}{s + \omega_x} \qquad \textbf{Eqn. 3-2}$$

If the first order approximation proves insufficient to meet all the objectives listed above, the higher order controller can be implemented. Due to the fact that only the angular rates are applied to the feedback loop, this type of controller would only be suggested for piloted aircraft where stick commands are driven by rate command rather then command angle. The next simplest form would be a second order model composed of proportional and integral control, equation 3-3. This form has been used previously with dynamic inversion for fighter aircraft fly-by-wire systems. The desired dynamics are expressed by:

$$\dot{u}_d = \omega_x \left(\tfrac{1}{2} u_{cmd} - u_m\right) + \frac{\omega_x^2}{4s}\left(u_{cmd} - u_m\right) \qquad \textbf{Eqn. 3-3}$$

The unique form of equation 3-3 allows for the additional pole to be canceled by the one of the repeated zeros, which results in the following closed-loop transfer function form:

$$\frac{u_m}{u_{cmd}} = \frac{\tfrac{1}{2}\omega_x}{s + \tfrac{1}{2}\omega_x} \qquad \textbf{Eqn. 3-4}$$

Essentially this form compensates for the angular rate and acceleration. If using just a static gain for control the aircraft must be analyzed at varying trim conditions and a set of gains solved that stabilize the aircraft.

3.1.2. Dual Inversion: Separation of Dynamics

For aircraft applications there are times where orientation commands (ϕ, θ, ψ) are better than commanding the body axis rates (p, q, r), especially from an unmanned/autonomous point of view where the three basic commands are

altitude, heading, and velocity. This method requires a transformation block that can change angle commands to rate commands. The Euler angle equations for pith, roll, and yaw are used to form the transformation block.

$$\begin{aligned}\dot{\phi} &= p + \tan(\theta)[q\sin(\phi) + r\cos(\phi)] \\ \dot{\theta} &= q\cos(\phi) - r\sin(\phi) \\ \dot{\psi} &= \frac{[q\sin(\phi) + r\cos(\phi)]}{\theta}\end{aligned} \qquad \textbf{Eqn. 3-5}$$

By solving the above equations for the dynamic rates – p, q, and r – a relation can be made similar to the method used for the dynamic inversion equations.

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix}_{cmd} = \begin{bmatrix} 1 & 0 & -\sin(\phi) \\ 0 & \cos(\phi) & \cos(\theta)\sin(\phi) \\ 0 & -\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}_{d} \qquad \textbf{Eqn. 3-6}$$

Essentially this transformation block is a transformation of the slower aircraft dynamics, ϕ, θ, and ψ, to a set of faster dynamics, p, q, and r. One method employed in many dynamic inversion controllers is this separation of "fast" and "slow" dynamics to compensate for an ineffective control matrix[5]. By separating to multiple time scales, translational and rate dynamics, it is assumed that the rate dynamics evolve much faster than the translational allowing for separate inner loop control of the rates and outer loop control of the orientation. The Euler angles ϕ, θ, and ψ are deemed the "slow" dynamics because the control effectiveness on their response is quite low and the angular rates p, q, and r are the "fast" dynamics. This separation of dynamics seeks to reformulate the original function of the system, equation 2-1, into a set of two separate equations of fast and slow dynamics.

$$\begin{aligned}\dot{x} &= f(x) + g(x)y \\ \dot{y} &= h(x, y) + k(x, y)u\end{aligned} \qquad \textbf{Eqn. 3-7}$$

where x is the slow dynamics vector and y is the fast dynamics vector. The full state equations are found in h, and k is the control effectiveness function. Applying this approach to the aircraft equations of motion developed thus far,

substituting equation 3-6 in to 3-1 and substituting the result into equation 2-16, yields the following dynamic expression.

$$\begin{bmatrix} \delta e \\ \delta a \\ \delta r \end{bmatrix}_{cmd} = \begin{bmatrix} 0 & L_{\delta a} & L_{\delta r} \\ M_{\delta e} & 0 & 0 \\ 0 & N_{\delta a} & N_{\delta r} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \omega_p & 0 & 0 \\ 0 & \omega_q & 0 \\ 0 & 0 & \omega_r \end{bmatrix} \begin{bmatrix} 1 & 0 & -\sin(\phi) \\ 0 & \cos(\phi) & \cos(\theta)\sin(\phi) \\ 0 & -\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}_d - \begin{bmatrix} p \\ q \\ r \end{bmatrix}_m \right.$$

$$\left. - \begin{bmatrix} 0 & L_\beta & L_p & 0 & L_r \\ M_\alpha & 0 & 0 & M_q & 0 \\ 0 & N_\beta & N_p & 0 & N_r \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ p \\ q \\ r \end{bmatrix}_m - \frac{\begin{bmatrix} I_y I_z & 0 & I_y I_{xz} \\ 0 & I_x I_y - I_{xz}^2 & 0 \\ I_y I_{xz} & 0 & I_x I_z \end{bmatrix}}{\Gamma} \begin{bmatrix} pq \\ I_{xz}\left(r^2 - p^2\right) + \left(I_z - I_x\right)pr \\ qr \end{bmatrix} \right\}$$

Eqn. 3-8

Some have used the wind axis variables for referencing orientation as is shown above in the measured state vector. The downside to this approach is that the phugoid mode is unobservable with only the short period aerodynamic terms from the longitudinal axis present in equation 3-8. By substituting the relation for angle of attack in equation 2-6, this issue can be avoided.

Now a second set of desired dynamics must be applied for the angular rates. This can be done using the same conditions presented in the classic gain control section; the benefit of this approach is that only one set of gains are need for the entire flight envelope. Alternative forms of desired response can be solved using flying quality relations and ride qualities if designing for sensitive payloads using the military standards for stability and control, Mil-STD-1797A[6]. Flying quality specifications for different vehicle classes and mission types can be implemented based on desired response. One controller form is constructed by substituting the flying quality relation into the outer feedback path with the following transfer function:

$$\frac{u}{u_{cmd}} = \frac{K(s+a)}{s^3 + bs^2 + (c+K)s + Ka}$$

Eqn. 3-9

where K and a are real constants and $b = 2\zeta_{des}\omega_{n,des}$ and $c = \omega_{n,des}^2 - K$. The desired damping ratio, ζ_{des}, and desired natural frequency, $\omega_{n,des}$, are specified by handling levels for the separate longitudinal and lateral modes, dependent of vehicle classes.

3.1.3. H_∞ - Synthesis

One method of directly optimizing robust performance is through a design iteration of the structured singular values commonly known as μ – synthesis. When designing a controller structure to optimize a given plant's performance, the process is known as H_∞ – synthesis[7]. The main difference between this method and the ones previously explored is that all performance specifications are in the form of weighted transfer functions describing the magnitude and frequency content of the following aircraft parameters: control input, input disturbance, sensor noise, tracking errors, actuator limits, and flying qualities or desired dynamics. Proper selection of these weights depends a great deal upon the understanding of the model process and the physical system. The plant is modeled as a nominal plant coupled with a structured perturbation model, which is also weighted with a given transfer function. The entire structure is interconnected into a single standard structure whose form is presented in section 6.5.

This controller design is a frequency domain optimization for robust control. H_∞ is defined as the space of proper and stable transfer functions with the main objective of the optimization process being to minimize the H_∞ norm[8]. The advantages to minimizing this norm are that it is a measure of the energy gain for the system and the resultant controller has guaranteed stability margins. For a SISO case this process corresponds to minimizing the peak value of the Bode magnitude plot of the sensitivity function for the closed loop. For a MIMO system this is a minimization process of the peak singular value of the sensitivity function. Given the plant dynamics G(s) in the following block form:

$$G(s)=\begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ where } G(s)=C(sI-A)^{-1}B+D \qquad \textbf{Eqn. 3-10}$$

A set of controlled states and performance measured states can be separated to represent the plant dynamics by:

$$G(s)=\begin{bmatrix} A & B_1 & B_2 \\ C_1 & 0 & D_{12} \\ C_2 & D_{21} & 0 \end{bmatrix} \qquad \textbf{Eqn. 3-11}$$

The matrices B_1 and C_1 correspond to the controller input/output interface and the matrices B_2 and C_2 correspond to the performance measure input/output relations discussed further in sections 6.1–6.4. The following four conditions must be satisfied in order to proceed with the H_∞ controller design:

1) (A, B_1) is controllable and (C_1, A) is observable.

2) (A, B_2) is stabalizable and (C_2, A) is detectable.

3) $D_{12}^*\begin{bmatrix} C_1 & D_{12} \end{bmatrix}=\begin{bmatrix} 0 & I \end{bmatrix}$

4) $\begin{bmatrix} B_1 \\ D_{12} \end{bmatrix} D_{12}^* = \begin{bmatrix} 0 \\ I \end{bmatrix}$

The first condition was verified by solving the controllability and observability gramian. The remaining conditions are verifiable by manipulation of the matrix parts.

Given the two Hamiltonian matrices H and J, with the structure shown below, a stabilizing solution to the Riccati equation can be obtained by constructing a basis for the stable invariant subspace of H and J such that $\|T\|_\infty < \gamma$, where T is the complementary sensitivity function of the performance measure.

$$H = \begin{bmatrix} A & \gamma^{-2} B_1 B_1^* - B_2 B_2^* \\ -C_1^* C_1 & -A^* \end{bmatrix} \qquad J = \begin{bmatrix} A^* & \gamma^{-2} C_1^* C_1 - C_2^* C_2 \\ -B_1 B_1^* & -A \end{bmatrix} \quad \textbf{Eqn. 3-12}$$

These matrices are translated into the following Riccati equations:

$$X_\infty A + A^* X_\infty + X_\infty \left(B_1 B_1^* / \gamma^2 - B_2 B_2^* \right) X_\infty + C_1^* C_1 = 0$$

$$A Y_\infty + Y_\infty A^* + Y_\infty \left(C_1^* C_1 / \gamma^2 - C_2^* C_2 \right) Y_\infty + B_1 B_1^* = 0 \qquad \textbf{Eqn. 3-13}$$

where $\rho\left(X_\infty Y_\infty\right) < \gamma^2$

These equations are convex, and so a simple Newton iteration or any bifurcation method can be used to obtain the corresponding X_∞ and Y_∞ satisfying the above equations. With X_∞ and Y_∞ known, a suboptimal controller can be represented by the following system matrix:

$$K(s) = \begin{bmatrix} \hat{A}_\infty & -Z_\infty L_\infty \\ F_\infty & 0 \end{bmatrix} \qquad \textbf{Eqn. 3-14}$$

where

$$\hat{A}_\infty = A + \gamma^{-2} B_1 B_1^* X_\infty + B_2 F_\infty + Z_\infty L_\infty C_2$$

$$F_\infty = -B_2^* X_\infty \quad L_\infty = -Y_\infty C_2^* \quad Z_\infty = \left(I - \gamma^{-2} Y_\infty X_\infty \right)^{-1} \qquad \textbf{Eqn. 3-15}$$

While the first approach solves the two Ricatti equations directly, a second method solves for the basis of a linear matrix inequality (LMI). Given the plant form listed in equation 3-11 and the controller form as:

$$K = \begin{bmatrix} A_k & B_k \\ C_k & D_k \end{bmatrix} \qquad \textbf{Eqn. 3-16}$$

a closed loop system can be formed by:

$$G_{cl} = \begin{bmatrix} A + B_2 D_k C_2 & B_2 C_k & B_1 + B_2 D_k D_{21} \\ B_k C_2 & A_k & B_k D_{21} \\ C_1 + D_{12} D_k C_2 & D_{12} C_k & D_{11} + D_{12} D_k D_{21} \end{bmatrix} = \begin{bmatrix} A_{cl} & B_{cl} \\ C_{cl} & D_{cl} \end{bmatrix} \qquad \textbf{Eqn. 3-17}$$

Taking the Riccati equations listed in equation 3-13, a linear matrix inequality can be formed which guarantees A_{cl} is stable and that $\|T\|_\infty < \gamma$ if and only if $X_{cl} = X_{cl}^T$ and is positive definite.

$$\begin{bmatrix} XA_{cl} + A_{cl}^* X_{cl} & X_{cl} B_{cl} & C_{cl}^* \\ B_{cl}^* X_{cl} & -\gamma I & D_{cl}^* \\ C_{cl} & D_{cl} & -\gamma I \end{bmatrix} < 0 \qquad \textbf{Eqn. 3-18}$$

Re-writing the closed loop form presented in equation 3-17 and breaking up the parts as shown in equation 3-19 leads to an alternative inequality.

$$\begin{aligned} A_{cl} &= \overline{A} + \underline{B}K\underline{C} = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & B_2 \\ I & 0 \end{bmatrix} K \begin{bmatrix} 0 & I \\ C_2 & 0 \end{bmatrix} \\ B_{cl} &= \overline{B} + B\underline{B}K\underline{D}_{21} = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} + \underline{B}K \begin{bmatrix} 0 \\ D_{21} \end{bmatrix} \\ C_{cl} &= \overline{C} + \underline{D}_{12}K\underline{C} = \begin{bmatrix} C_1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & D_{12} \end{bmatrix} K\underline{C} \\ D_{cl} &= D_{11} + \underline{D}_{12}K\underline{D}_{21} \end{aligned} \qquad \textbf{Eqn. 3-19}$$

Now equation 3-18 can be re-written as:

$$HX_{cl} + Q^* J^* PX_{cl} + PX_{cl} JQ < 0 \qquad \textbf{Eqn. 3-20}$$

where

$$\begin{aligned} HX_{cl} &= \begin{bmatrix} \overline{A}^* X_{cl} + X_{cl}\overline{A} & X_{cl}\overline{B} & \overline{C}^* \\ \overline{B}^* X_{cl} & -\gamma I & D_{11}^* \\ \overline{C} & D_{11} & -\gamma I \end{bmatrix} \\ PX_{cl} &= \begin{bmatrix} \underline{B}^* X_{cl} & 0 & \underline{D}_{12}^* \end{bmatrix} \\ Q &= \begin{bmatrix} \underline{C} & \underline{D}_{21} & 0 \end{bmatrix} \end{aligned} \qquad \textbf{Eqn. 3-21}$$

By the elimination lemma the inequality in equation 3-20 is equivalent to the two inequalities given below:

$$N_{PX_{cl}}^* HX_{cl} N_{pX_{cl}} < 0 \ \textbf{and} \ N_Q^* HX_{cl} N_Q < 0 \qquad \textbf{Eqn. 3-22}$$

where

$$N_{PX_{cl}} = \ker(PX_{cl})$$
$$N_Q = \ker(Q)$$

Eqn. 3-23

given $\gamma = 1$, a vector X, such that $X^T = X > 0$ and the condition presented in equation 3-22 is satisfied, then the system is stable and has an H_∞ norm less than one. This is commonly referred to as the bounded real lemma. The system controller is optimized by finding a linear function of X and solving for the minimum value using Nesterov and Nemirovski's[9] projection method.

4. Robust Analysis

Classical root locus design methods allow the placement of a pair of complex conjugate poles to meet transient response specifications, although the designer has little control of the location of all other poles and zeros. The particular property that a control system must have to operate properly in the physical world is called robustness. A closed loop system that possesses both good disturbance rejection and low sensitivity is said to be robust. Disturbance rejection is the ability to possess minimal tracking error in the presence of disturbance. For an aircraft this disturbance can be generated from actuator error, excitation of structural modes, or wind gusts. Low sensitivity is the ability to maintain good tracking in the presence of changes in plant parameters, triggered by inaccuracies in the system model or noise sensor measurements. In summary, this means that a robust controller must operate satisfactorily for a family of plants and not just the nominal case.

Robustness is divided into to categories: robust stability and robust performance. Robust stability is the ability to maintain closed-loop stability in spite of parameter variations and un-modeled high-frequency dynamics. Robust performance is the ability to maintain acceptable performance despite system disturbances.

4.1. Time Domain Analysis

Time domain inputs, such as step or impulse inputs, can be used to evaluate system characteristics including damping, natural frequency, and peak overshoot. Initial conditions or impulse excitations are particularly useful in evaluating the damping of rate variables. Such time domain closed-loop response can be evaluated through simulation in MATLAB's Simulink environment.

4.2. Frequency Domain Analysis

The response of a linear system to sinusoidal input is referred to as the system's frequency response. Typical specifications of the frequency domain are gain

margin, phase margin, and bandwidth. Gain margin is the amount by which system gain can increase before the system becomes neutrally stable. Phase margin is the amount by which phase lag can increase before the system becomes neutrally stable. Bandwidth is defined as the maximum frequency at which system output will satisfactorily track a sinusoid input. The process of assessing robust performance and robust stability of a system using μ – synthesis relies on assessment of frequency domain requirements, which will be discussed further in the next sections.

4.2.1. Structured Singular Value Decomposition

For any given matrix a set of two orthogonal matrices, U and V, and a single diagonal matrix, Σ, can be formed such that

$$A = U\Sigma V^T \qquad \textbf{Eqn. 4-1}$$

such that

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_n \end{bmatrix} \quad \textbf{where } \boldsymbol{n = rank(A)} \qquad \textbf{Eqn. 4-2}$$

While the singular vectors are not unique, the singular values, σ, are unique and are the eigenvalues of $A^T A$. Singular values are a measure of the relative geometry of the system matrix. Specifically, the singular values tell how A distorts the unit sphere or, in other words, indicates the strong/weak input or output directions.

The main reason the structured singular values of the system are of concern is because the H_∞ norm of a given system is equal to the maximum singular value, $\sigma_1 = \overline{\sigma}$, over a given frequency range

$$\|G\|_\infty = \max_{1 \le i \le k} \overline{\sigma}\left[G\left(j\omega_i\right)\right] \qquad \textbf{Eqn. 4-3}$$

4.2.2. μ – Synthesis

The performance of a system is measured in terms of the behavior of the outputs, which may include desired dynamic response or error minimization. By applying disturbances to varying aspects of the plant dynamics and resulting output, one can characterize the robustness of the system, both in terms of stability and performance. For this type of analysis the system is constructed in the following form:

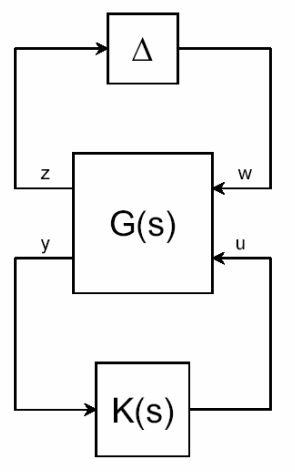

Figure 4-1: System interconnection with disturbance matrix

where all plant disturbances pass through the Δ matrix. μ–synthesis is a technique by which the smallest perturbation matrix is found and satisfies the equation:

$$\det(I - G\Delta) = 0 \qquad \textbf{Eqn. 4-4}$$

Additionally, we will assume that Δ is a diagonally structured matrix with a maximum singular value less than or equal to one. This perturbation can be rewritten as

$$\eta_{\min} = \inf\{\bar{\sigma}(\Delta) : \det(I - G\Delta) = 0\} \qquad \textbf{Eqn. 4-5}$$

Given the unique structure of Δ, the minimum perturbation can be related to the largest eigenvalue by

$$\eta_{\min} = \frac{1}{\max\limits_{\bar{\sigma}(\Delta)\le 1} \rho(G\Delta)} \qquad \textbf{Eqn. 4-6}$$

μ_Δ is defined as the reciprocal of the perturbation matrix. In order to find a tight upper and lower limit of μ_Δ, the system must be transformed such that μ_Δ(G) does not change but the maximum eigenvalue and maximum singular value are transformed. Therefore, a transformation matrix T is selected such that $T\Delta = \Delta T$ then form a unitary matrix which satisfies $T = UR$ where R is a Hermitian matrix, $0 < R = R^* \in D$, where D is a nonsingular diagonal matrix. The bounds can then be expressed in by equation 4-7. This bound is computed using the mu-toolbox[10] associated with MATLAB.

$$\max \rho(UG) \leq \mu_\Delta(G) \leq \inf \bar{\sigma}\left(DGD^{-1}\right) \qquad \textbf{Eqn. 4-7}$$

4.2.3. Model Uncertainty

Controllers are designed to maintain stability and performance requirements of the closed loop system over a set of transfer functions representing the range of plant dynamics and operating environment. Since it is impossible to empirically model with perfect accuracy a dynamic system and its operating environment, uncertainty modeling plays an important role in controller design and system analysis.

Even after applying uncertainty models to the nominal plant dynamics, there are still limitations to the controller based on hysteresis, backlash, deadbands, or other forms of inherently nonlinear dynamics. A second limitation commonly overlooked in the design phase is implementation of large order controllers resulting in computational delays. As the time to execute a segment of controller code increases, the frequency at which the controller is modeled increases. If this frequency approaches the Nyquist frequency of the system, then aliasing can occur.

Uncertainties are broadly classified by two categories, structured and unstructured, both of which are present in any given physical system. The key to establishing a good uncertainty model is to recognize to what category a particular uncertainty belongs and how to accurately characterize the uncertainty in the frequency domain for robust design and analysis.

4.2.3.1. Unstructured

Unstructured uncertainties are those forms for which little knowledge is possessed. These are usually not modeled or cannot be modeled at all. Examples of this type for an aircraft system would be un-modeled high frequency dynamics, structural vibration, measurement noise, computational round-off error, or sampling delays.

Since synthesis techniques of the full nonlinear dynamics are presently limited, linearization of nonlinear dynamics is required. The type of approximation methods used can introduce varying levels of uncertainty.

4.2.3.2. Structured

Structured uncertainty can be modeled with relatively well-known bounds and ranges. These types of uncertainty arise from accuracy of aerodynamics, mass, moments of inertia, and control effectiveness. These forms of uncertainty can be analyzed and modeled within the frequency domain and the perturbation block discussed earlier in section 4.2.2. Given a set of structured uncertainties Δ the nominal plant G can be varied by

$$G_{\Delta} = G + W_1 \Delta W_2 \qquad \textbf{Eqn. 4-8}$$

where W_1 and W_2 are stable transfer matrices that characterize the spatial and frequency structure of the uncertainty. Essentially, the weighting functions confine G_{Δ} to a normalized neighborhood of the nominal plant G.

5. Closed Loop Model Simulation

In this section the simulation component buildup for the Aerosonde UAV is presented for both a first order linear approximation and a complete nonlinear 6-degree-of-freedom (6DOF) model. The closed loop systems are compared using the linear model simulation results for the complete dynamic inversion controller with a nonlinear 6DOF model are also presented. The simulations are programmed within the Matlab Simulink environment, a time-based simulation model that is generated through block interconnection programming. Additionally, sections of code from the AeroSim block set are used in generating the nonlinear model.

5.1. Linear

Using small disturbance theory, a set of first order linear dynamic models is generated to simulate the aircraft dynamics. Given the geometry and weight distribution of the Aerosonde UAV, the longitudinal and lateral-directional dynamics can be separated due to lack of coupling due to the xz plane of symmetry[11]. The longitudinal dynamics can be represented by two force equations, F_x and F_z, and the equation for pitching moment, M.

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_w & 0 & -g \\ Z_u & Z_w & u_0 & 0 \\ M_u + M_{\dot{w}} Z_u & M_w + M_{\dot{w}} Z_w & M_q + M_{\dot{w}} u_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} X_{\delta e} \\ Z_{\delta e} \\ M_{\delta e} + M_{\dot{w}} Z_{\delta e} \\ 0 \end{bmatrix} \delta e$$

Eqn. 5-1

The lateral-directional dynamics are represented by the two moment equations, N and L, and the lateral force equation, F_y.

$$\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{Y_\beta}{U_0} & \frac{Y_p}{U_0} & \left(\frac{Y_r}{U_0} - 1\right) & \frac{g\cos\theta_0}{U_0} & 0 \\ L_\beta & L_p & L_r & 0 & 0 \\ N_\beta & N_p & N_r & 0 & 0 \\ 0 & 1 & \tan\theta_0 & 0 & 0 \\ 0 & 0 & \sec\theta_0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} 0 & \frac{Y_{\delta r}}{u_0} \\ L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta a \\ \delta r \end{bmatrix} \quad \textbf{Eqn. 5-2}$$

Each plant is then connected through a feedback loop with each controller structure.

5.2. Nonlinear 6-Degree-of-Freedom

The three force and moment equations presented in section 2.1 are used in the nonlinear model generating our six degrees of freedom. No longer applying small angle assumptions, the lateral and longitudinal dynamics may have coupling at large angles; therefore, they can no longer be applied separately. The equations of motion are also given in more complex form where each derivative is a function of dynamic pressure.

For the rapid development of nonlinear 6-degree-of-freedom aircraft dynamic models the AeroSim[12] aeronautical simulation block set is used. The AeroSim library includes complete aircraft models, which can be customized through parameter files as well as disturbance models for gust alleviation and wind shear. The internal layout of the complete aircraft model generated from the AeroSim library is shown in figure 5-1. The simplified diagram presented in figure 5-2 shows how the aerodynamics, propulsion, and inertia models are iterated to compute the airframe loads as functions of control input and environmental effects. The resulting directional and angular accelerations are then integrated to obtain all aircraft states. These state outputs update the environment model with each iteration, as well as provide sensor outputs. The discrete time model is run at a sampling frequency of 100 Hz while the controller is run at only 50 Hz.

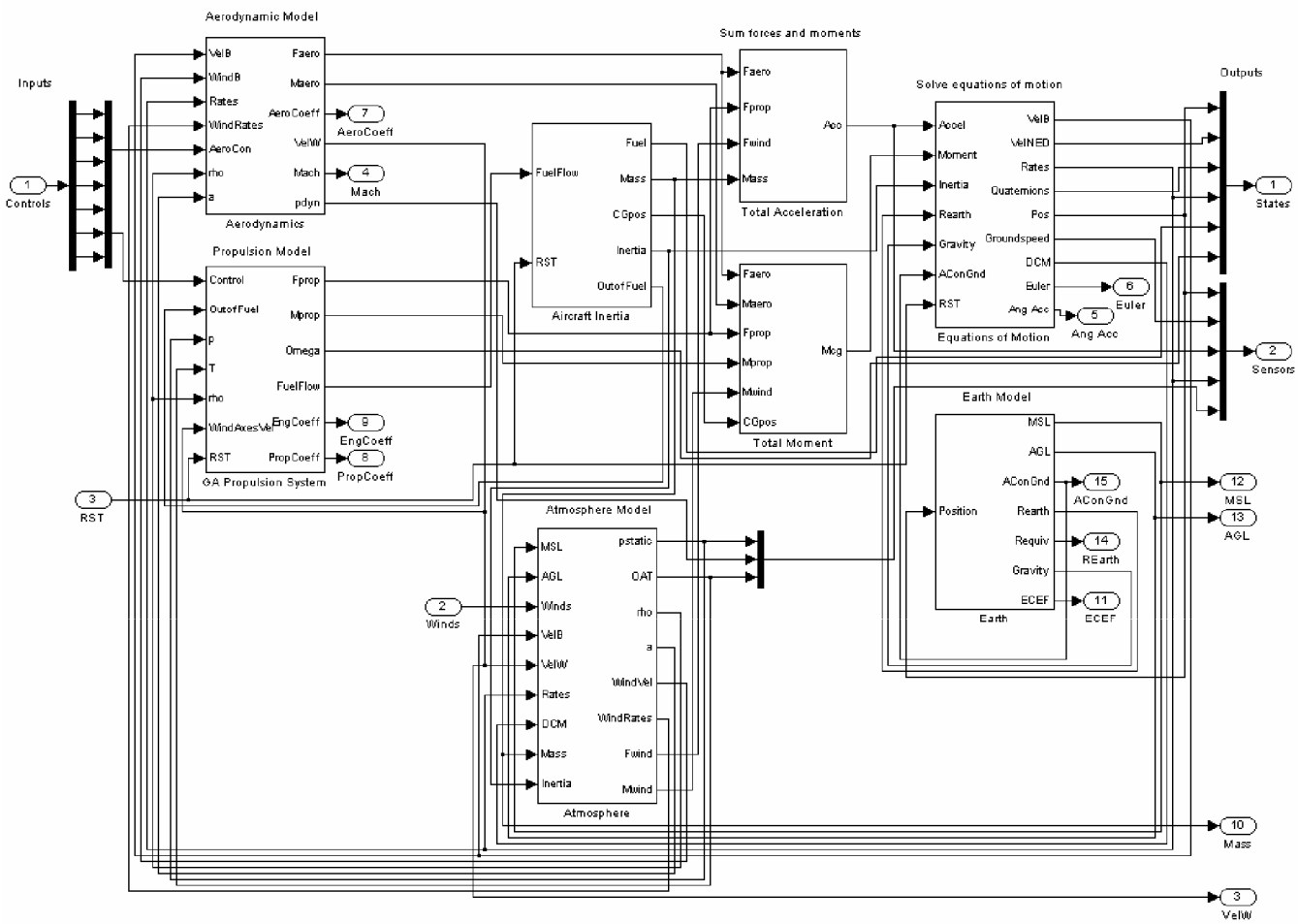

Figure 5-1: Internal structure of the plant dynamics[12]

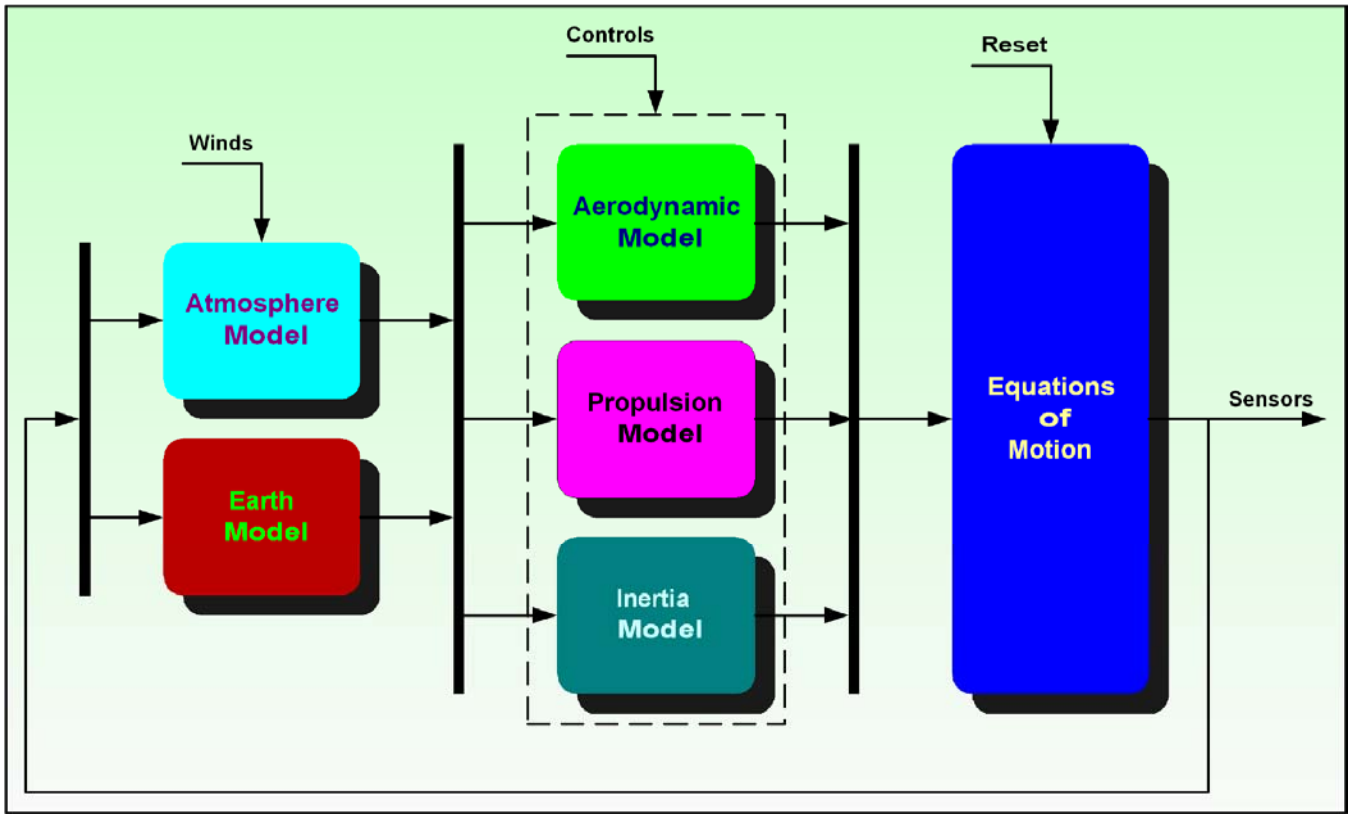

Figure 5-2: Simplified internal structure of the plant model[12]

5.3. Actuator Model

The actuator model is generated by using a second order approximation to the system dynamics.

$$G_{act} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad \textbf{Eqn. 5-3}$$

where ω_n is the natural frequency and ζ is the actuator's damping ratio. The function's input comes from the controller's command signal, and the output is the physical surface deflection given in units of radians and supplied to the plant. In addition to the second order model, nonlinear terms such as saturation and rate limit are added to the output signal. The max actuation rate defines the maximum speed at which the actuator can move. For the servos used in the simulation

model this rate is given as 11.35 rad/s. The second limit is the actuator range and saturates the signal when it has reached max deflection. For the elevator and rudder, this limit corresponds to ±0.44 rad and ±0.26 for the ailerons. All simulations ignore the use of flaps, so this has a constant zero value provided to the plant. All control surfaces are simulated with a HS-5925MG servo with a natural frequency of 52.3 rad/s and a dampening ratio of 1.4.

$$G_{act} = \frac{2735}{s^2 + 146s + 2735} \qquad \textbf{Eqn. 5-4}$$

The servos model dynamics were found through parameter identification of the frequency response. These functions were generated through parameter identification by Heinzen.

5.4. Sensor Model

The interconnection of the plant with the controller requires the measurement of angular rates and translational accelerations as well as Euler angle measurement. This is all measured in the actual aircraft by the AHRS inertial measurement unit (IMU). For the simulation model the sensors performance are modeled as a first order system with output lag.

$$G_{sen} = \frac{1}{\tau s + 1} \qquad \textbf{Eqn. 5-5}$$

where τ is the time constant for the sensor dynamics. Included with the first order model are stochastic effects such as signal noise. Error associated with the integration of the rates is modeled as a Wiener process in discrete-time where $x_{t+k} = x_t + e_t$. The covariant error signal, *e(t)* for $t \geq 0$ with $e(0) = 0$, is such that the increment $e(t) - e(t+k)$ is Gaussian with mean 0 and variance of $t - (t+k)$ [13]. For each sensor measurement a noise signal is introduced whose variance is bounded by the resolution of the sensor. A random walk signal is also added to the sensor output. For rates and accelerations, this term is almost negligible; however, the Euler angles and body velocities, which are computed through integration, can generate large walks over time.

5.5. Gust Disturbance

Environmental disturbances are applied to the aircraft dynamics in two forms: turbulence and wind shear. The turbulence velocities and accelerations are computed using a von Karman turbulence model[14]. The model output is a function of background wind, altitude, airspeed, and three white noise signals. Wind shear is applied by calculating the angular rate variations due to background wind and turbulence velocities. The wind shear in pitch and yaw are calculated from the following equations:

$$q_{wind} = \frac{1}{u}\frac{dw_{wind}}{dt}$$
$$r_{wind} = \frac{1}{u}\frac{dv_{wind}}{dt} \qquad \textbf{Eqn. 5-6}$$

where w_{wind}, and v_{wind} are the vertical and side turbulence velocities, respectively and u is the aircraft's forward velocity. The background wind is simply a vector of the average wind velocity along a global x,y,z-axis.

5.6. Nominal and Perturbed System Response

Using a simple Simulink model the nominal closed loop response for each controller is simulated along with parametric disturbance models which vary equally between $\|\Delta\| = \pm 1$ or $\pm 100\%$ error for each variable. The plots illustrate the closed loop's ability to remain stable despite large uncertainty and to maintain some level of system performance. They also show that the system is less robust to negative error than it is to positive percent error.

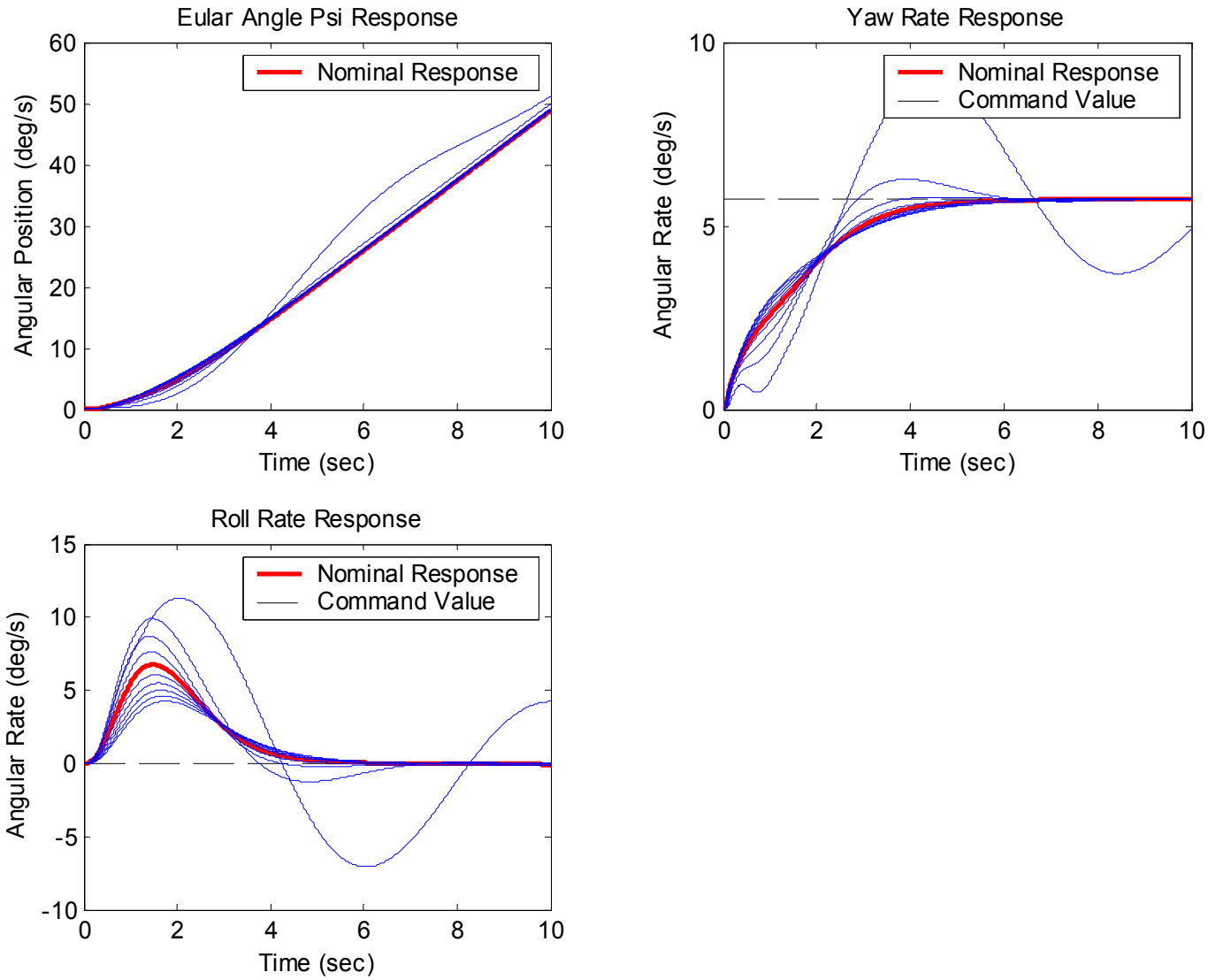

Figure 5-3: Full dynamic inversion response to a step command in yaw rate with parametric uncertainty

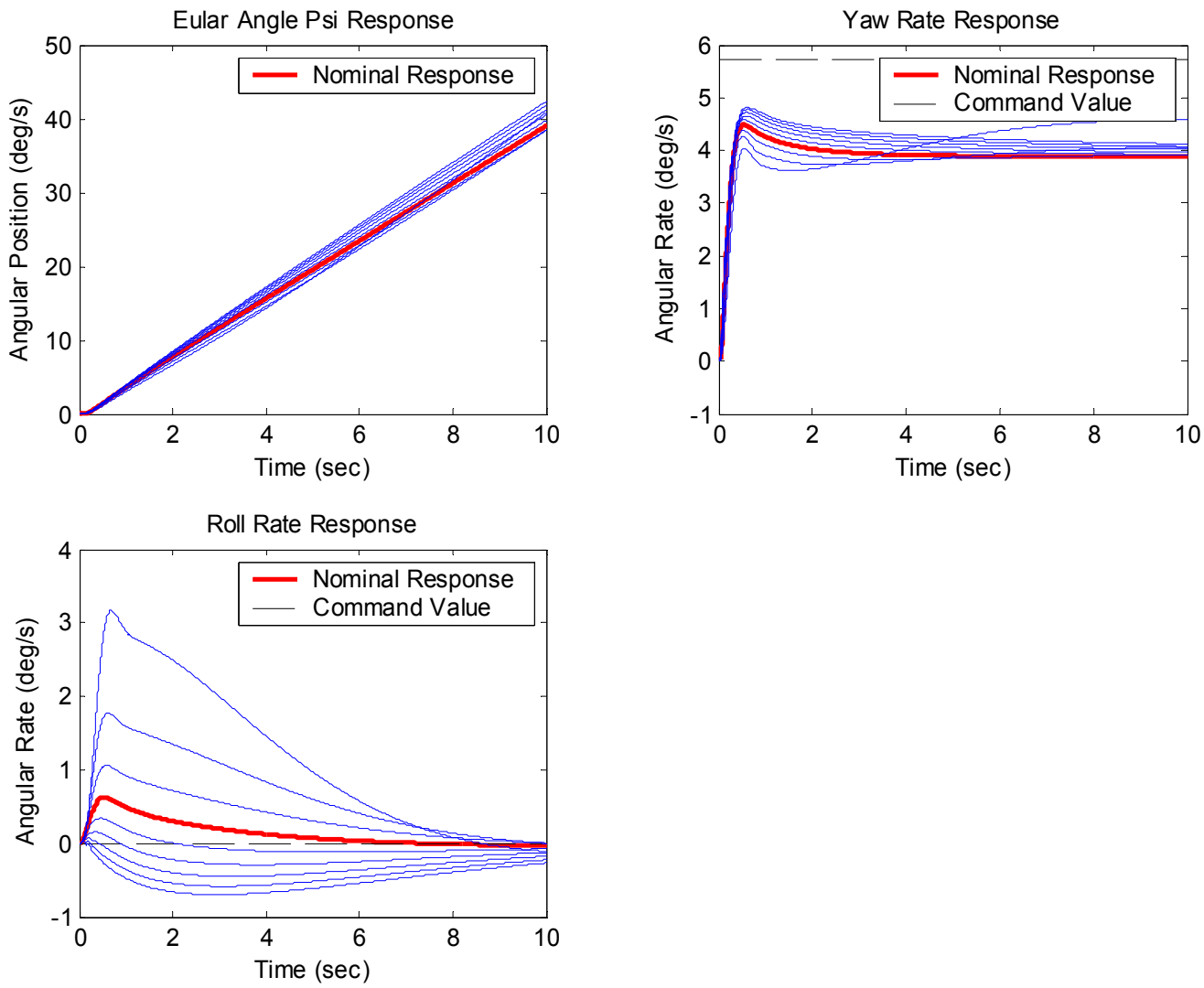

Figure 5-4: H_∞ response to a step command in yaw rate with parametric uncertainty

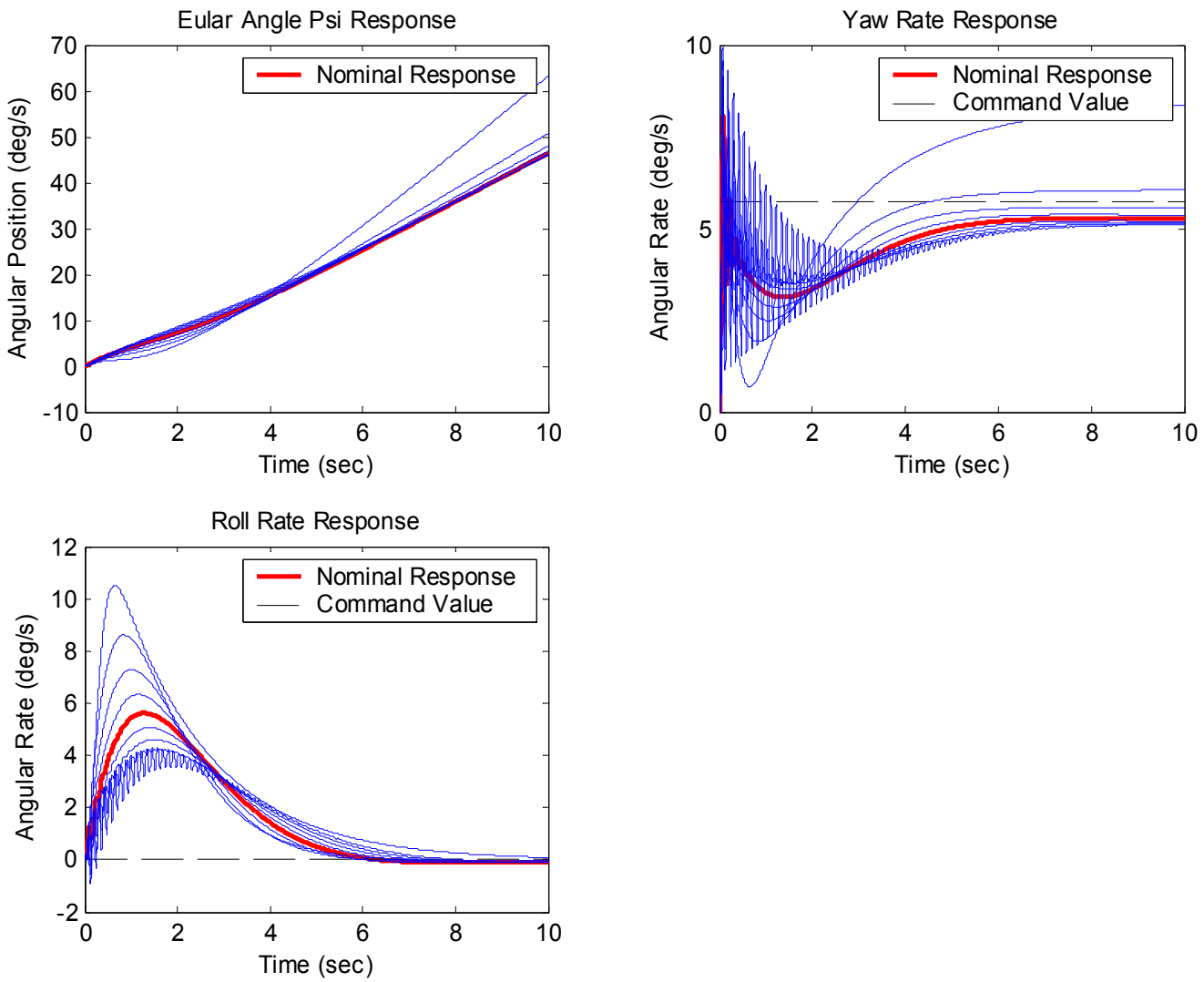

Figure 5-5: Dynamic inversion inner loop with PI outer loop response to a step command in yaw rate with parametric uncertainty

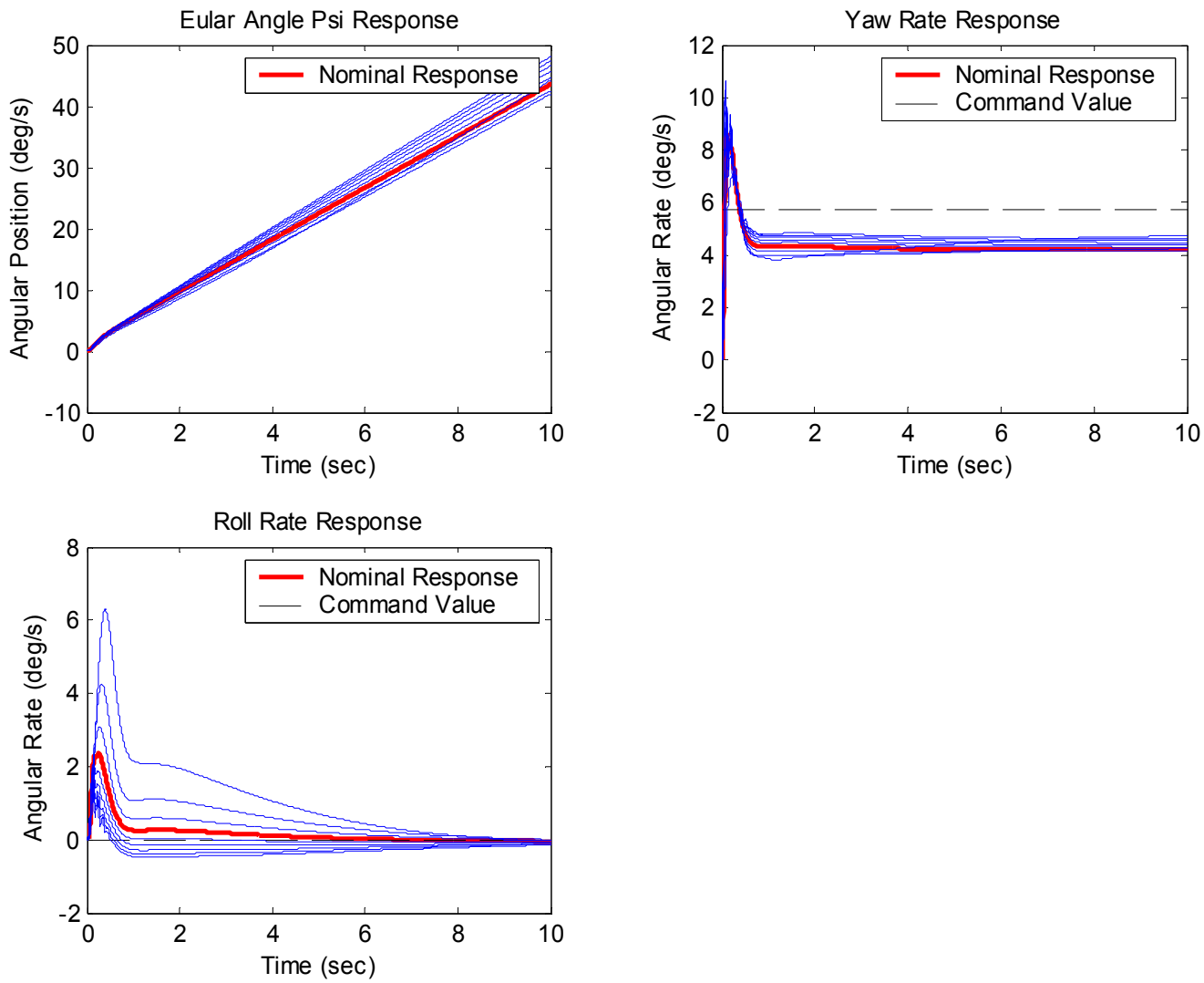

Figure 5-6: Dynamic inversion inner loop with H_∞ outer loop response to a step command in yaw rate with parametric uncertainty

6. Robust Stability and Robust Performance

To compute the robust stability of the closed-loop systems, a structured singular value analysis is conducted. The system's response is computed by forming a varying system matrix with complex frequency. This varying system response is then combined with a 4 x 4 complex perturbation block and the structured singular value is calculated by restructuring equation 4-5 to the following form:

$$\mu_{\Delta}(G) = \frac{1}{\min\{\overline{\sigma}(\Delta), \det(I - G\Delta) = 0\}} \qquad \textbf{Eqn. 6-1}$$

where Δ is the 4 x 4 complex perturbation block, $\Delta = diag(\Delta_1, \Delta_2, \Delta_3, ..., \Delta_9)$, and G is the closed loop system. Since this equation cannot be directly solved it is approached by an upper and lower bounds related by equation 6-2, see section 4.2.2 for more details.

$$\max_{\Delta \in B\Delta} \rho(\Delta M) \leq \mu_{\Delta}(M) \leq \inf_{D^{-1} exists} \overline{\sigma}(DMD^{-1}) \qquad \textbf{Eqn. 6-2}$$

The robust performance of a system looks at the effects of disturbance along with any parametric uncertainty. In most cases even before a system becomes unstable, the performance will degrade past acceptable limits. The same frequency varying system matrices used when calculating the robust stability are again applied for finding robust performance. The perturbation block is altered so that the four disturbances and the weighted input/output relations can be combined into a single Δ.

$$\Delta = \begin{bmatrix} \Delta_1 & 0 \\ 0 & \Delta_f \end{bmatrix} \text{ where } \Delta_1 \in C^{6x6}, \Delta_f \in C^{10x12} \text{ full block} \qquad \text{Eqn. 6-3}$$

The same μ_{Δ} relation shown previously was used but with the modified Δ structure.

6.1. Parametric Uncertainty

Parametric uncertainty is simply a well-defined form of structured uncertainty. The linearized lateral-directional dynamics presented in equation 6-4 are based

from unique trim values to represent the nominal plant at varying points within the flight envelope.

$$\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \frac{Y_\beta}{U_0} & \frac{Y_p}{U_0} & \left(\frac{Y_r}{U_0} - 1\right) & \frac{g\cos\theta_0}{U_0} \\ L_\beta & L_p & L_r & 0 \\ N_\beta & N_p & N_r & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \end{bmatrix} + \begin{bmatrix} 0 & \frac{Y_{\delta r}}{U_0} \\ L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta a \\ \delta r \end{bmatrix} \qquad \textbf{Eqn. 6-4}$$

All aerodynamic derivatives used by the dynamic inversion process will vary with known bounds represented mathematically by:

$$A_x^- \le A_x \le A_x^+ \qquad \textbf{Eqn. 6-5}$$

where A_x is the aerodynamic derivative and A_x^-, A_x^+ are the respective lower and upper limit of variation. All other derivatives will be assumed certain and fixed in value. Assuming the derivative value obtained for the given trim flight conditions is the nominal value, A_x^{nom}, then the expression for the varied parameter can be rewritten as

$$A_x = A_x^{nom} + \frac{k_1\delta}{1 - k_2\delta} \qquad \textbf{Eqn. 6-6}$$

where k_1 and k_2 are found by

$$\begin{aligned} k_1 &= \frac{2\left(A_x^+ - A_x^{nom}\right)\left(A_x^{nom} - A_x^-\right)}{A_x^+ - A_x^-} \\ k_2 &= \frac{\left(A_x^+ + A_x^-\right) - 2A_x^{nom}}{A_x^+ - A_x^-} \end{aligned} \qquad \textbf{Eqn. 6-7}$$

and $-1 \le \delta \le 1$. The perturbations for each aerodynamic derivative can be integrated into the plant model, equation 6-4, by introducing the terms z_x and w_x removing the variable δ from the system matrix.

$$\begin{bmatrix} \dot{x} \\ z_a \\ y \end{bmatrix} = \begin{bmatrix} A_x^{nom} & k_1 & B \\ 1 & k_2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ w_a \\ u \end{bmatrix} \qquad \textbf{Eqn. 6-8}$$

The above expression is simply a compact form of the state space model with the output function partitioned into the bottom portion of the matrix. Using the same format shown above, the variation in control derivatives, B, can be added to form the following system matrix

$$\begin{bmatrix} \dot{x} \\ z_a \\ z_b \\ y \end{bmatrix} = \begin{bmatrix} A_x^{nom} & k_{1a} & k_{1b} & b_x^{nom} \\ 1 & k_{2a} & 0 & 0 \\ 0 & 0 & k_{2b} & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ w_a \\ w_b \\ u \end{bmatrix} \qquad \textbf{Eqn. 6-9}$$

The delta structure can be represented by a single diagonal matrix

$$\Delta = \begin{bmatrix} \delta_a & 0 \\ 0 & \delta_b \end{bmatrix} \qquad \textbf{Eqn. 6-10}$$

and have the same interconnection structure with the plant as shown in figure 4-1 with the Δ structure shown in equation 6-10. The entire structure of the plant model with all parametric uncertainties for the lateral-directional equations of motion is given by equation 6-11.

$$
\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{Y_\beta}{U_0} & \frac{Y_p}{U_0} & \left(\frac{Y_r}{U_0} - 1\right) & \frac{g\cos\theta_0}{U_0} & 0 \\ L_\beta & L_p & L_r & 0 & 0 \\ N_\beta & N_p & N_r & 0 & 0 \\ 0 & 1 & \tan\theta_0 & 0 & 0 \\ 0 & 0 & \sec\theta_0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \frac{Y_{\delta r}}{U_0} \\ k_{1p} & k_{1\delta a} & 0 & 0 & L_{\delta a} & L_{\delta r} \\ 0 & 0 & k_{1r} & k_{1\delta r} & N_{\delta a} & N_{\delta r} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_p \\ w_{\delta a} \\ w_r \\ w_{\delta r} \\ \delta r \\ \delta a \end{bmatrix}
$$

$$
\begin{bmatrix} z_p \\ z_{\delta a} \\ z_r \\ z_{\delta r} \\ \beta \\ p \\ r \\ \phi \\ \psi \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} k_{2p} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{2\delta a} & 0 & 0 & 1 & 0 \\ 0 & 0 & k_{2r} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{2\delta r} & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_p \\ w_{\delta a} \\ w_r \\ w_{\delta r} \\ \delta r \\ \delta a \end{bmatrix}
$$

Eqn. 6-11

6.2. Disturbance Input

While parametric uncertainty is a large part of the plant model, it is not the only type of uncertainty considered in this study. As discussed previously, un-modeled dynamics or unstructured uncertainty can be simulated with a given bound by applying uncertainty at the system input. An input weighting function is generated such that there is a maximum modeling error of 1% at low frequency and that the uncertainty in the model grows up to 100% at high frequency. The weighting is in diagonal form such that the constraint applies equally on both the aileron and rudder control surfaces. Essentially this generates a sphere around the nominal plant known as the perturbation region.

$$
W_d = \begin{bmatrix} \frac{s+1}{s+100} & 0 \\ 0 & \frac{s+1}{s+100} \end{bmatrix}
$$

Eqn. 6-12

The interconnection of the weighting function is shown below where $-1 \le \Delta_{in} \le 1$.

6.3. Sensor Noise

A third type of uncertainty is uncertainty applied to the output, also known as sensor noise. For the Aerosonde model an output uncertainty weighting function is generated such that the angular rates, p and r, have a frequency measurement error of 0.0003 rad/sec at low frequency and an error of 0.015 rad/sec at high frequency. The Euler angles, which are measured through integration of the rates with Kalman filtering, are given a relatively large error in the mid-to-high frequency range. The interconnection of the weighting functions as well as the function itself are shown below.

$$W_n = \begin{bmatrix} 0.0003\dfrac{1+s/0.01}{1+s/0.5} & 0 & 0 & 0 \\ 0 & 0.0003\dfrac{1+s/0.01}{1+s/0.5} & 0 & 0 \\ 0 & 0 & 0.0007\dfrac{1+s/0.01}{1+s/2} & 0 \\ 0 & 0 & 0 & 0.0007\dfrac{1+s/0.01}{1+s/2} \end{bmatrix}$$

Eqn. 6-13

6.4. Performance Weightings

With the forms of structured and unstructured uncertainty that have been introduced into the model, a comprehensive analysis can be performed to measure the system's robust stability. However, we are also concerned with the robust performance of the model. To analyze this performance weighting functions must be introduced to measure the overall performance error of the system's response. For a dynamic inversion controller coupled with an H_∞ controller, the weighting function is selected such that the error of the roll and yaw rates have a tolerance of 0.1 rad/sec at low frequency and 1 rad/sec at high frequency. This alloys for

accurate command following at low frequencies and gradually degrades as frequency increases.

$$W_p = \begin{bmatrix} \frac{s+10}{s+1} & 0 \\ 0 & \frac{s+10}{s+1} \end{bmatrix} \qquad \textbf{Eqn. 6-14}$$

When analyzing the dual inversion controller, the performance weighting will measure the error associated with the Euler angles.

In addition to state variable response, a performance weighting will be applied to the system's actuators as well. This is done to eliminate both position and rate saturation of the actuators. The rudder is limited to a max deflection of 0.44 rad with a maximum actuation rate of 11.35 rad/sec. The ailerons are limited to a max deflection of 0.26 rad with a maximum deflection rate of 11.35 rad/sec.

$$W_a = \begin{bmatrix} 3.85 & 0 & 0 & 0 \\ 0 & 0.088 & 0 & 0 \\ 0 & 0 & 2.27 & 0 \\ 0 & 0 & 0 & 0.088 \end{bmatrix} \begin{bmatrix} \delta a \\ \delta \dot{a} \\ \delta r \\ \delta \dot{r} \end{bmatrix} \qquad \textbf{Eqn. 6-15}$$

6.5. Interconnection Structure

The uncertainties and performance weights are interconnected accordingly to form a complete open-loop structure as shown in the figures below. The first diagram is the control case for the study and contains no form of dynamic inversion. The two controller cases that will connect with this system are a robust controller using H_∞ – Synthesis and a proportional-integral (PI) controller.

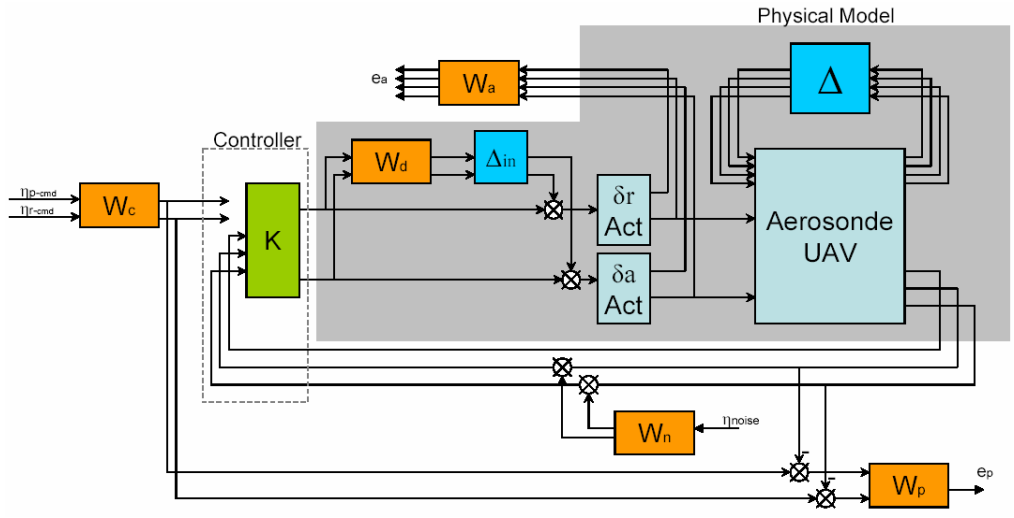

Figure 6-1: Open Loop Interconnection Structure

The second diagram has an inner loop dynamic inversion controller built into the system. A set of controllers will be constructed and compared against the ones acting directly on the plant.

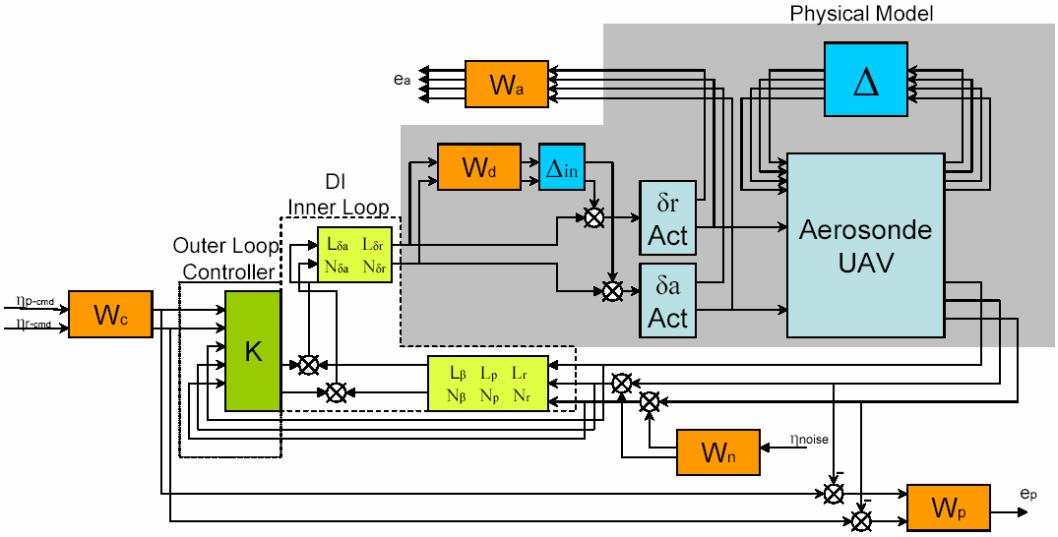

Figure 6-2: Open loop plant with dynamic inversion inner loop interconnection structure

6.6. System Response

Figures 6-3 through 6-6 shows the μ_Δ response for each closed-loop systems. The lower bound implies that there exists a perturbation size for each frequency response which will drive the system unstable and stems from the maximum modules theorem[15]. Conversely, the upper bound implies that a perturbation whose size is less than reciprocal of the value will still be a stable closed-loop system. The peak value of μ_Δ determines the maximum perturbation size for which the closed-loop is robustly stable. Note that for real perturbations, some of the system's response has zero for the lower bound. In actuality this is due to a convergence error where the perturbation size is extremely large for the lower bound. This convergence error only occurs for real perturbations, and typically only the upper bound is analyzed.

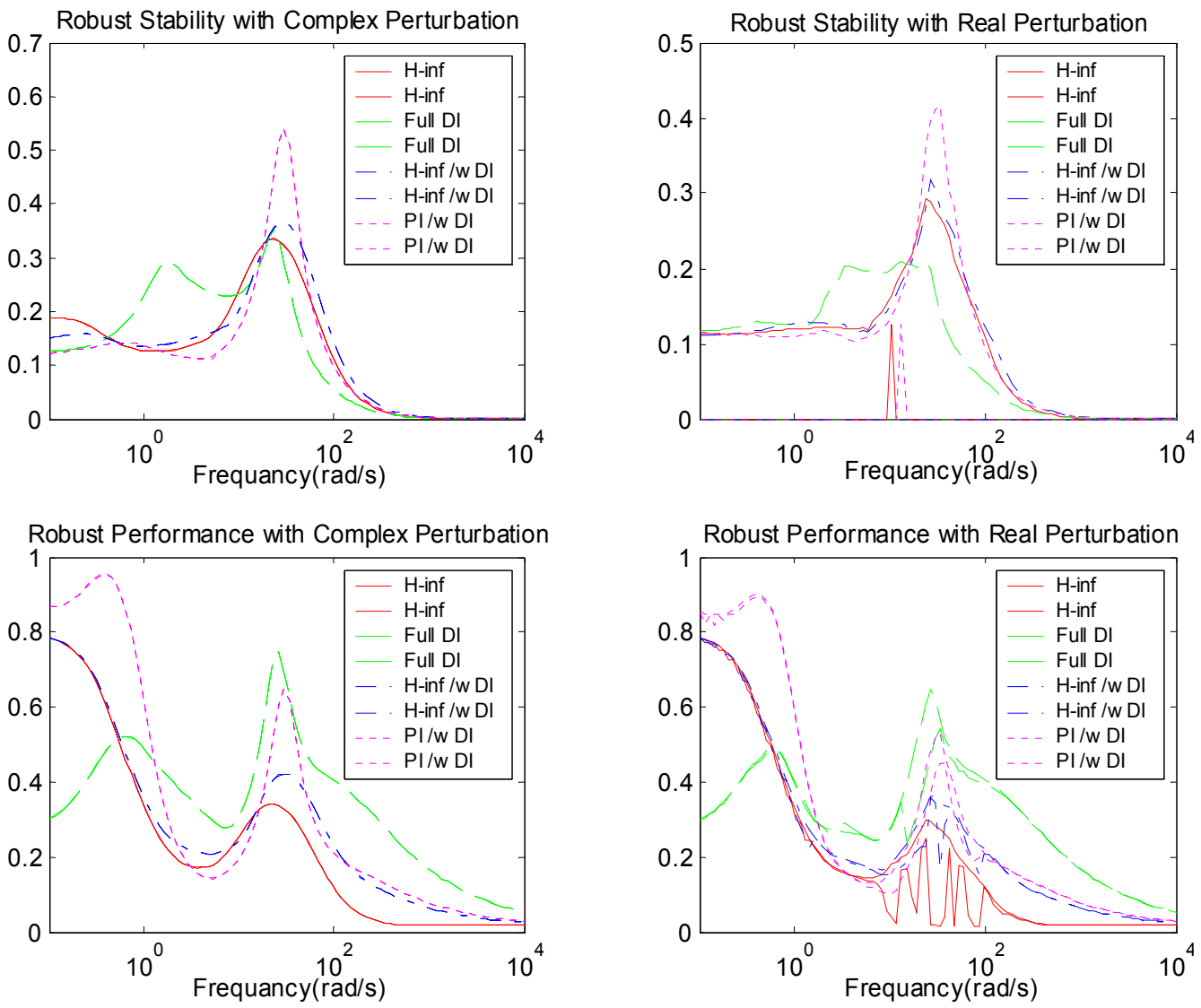

Figure 6-3: Closed loop system stability and performance with a maximum parametric uncertainty error of 10%

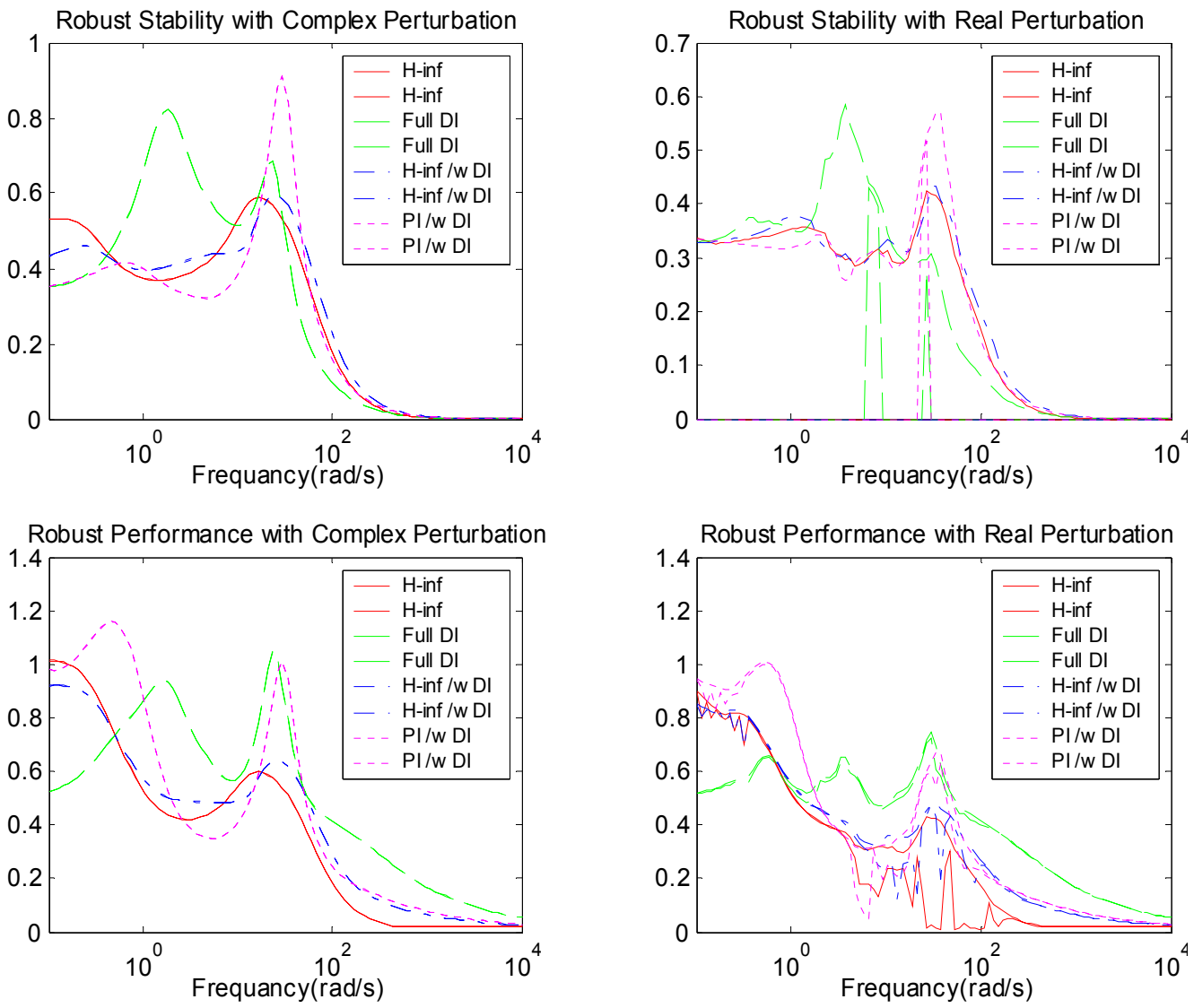

Figure 6-4: Closed loop system stability and performance with a maximum parametric uncertainty error of 30%

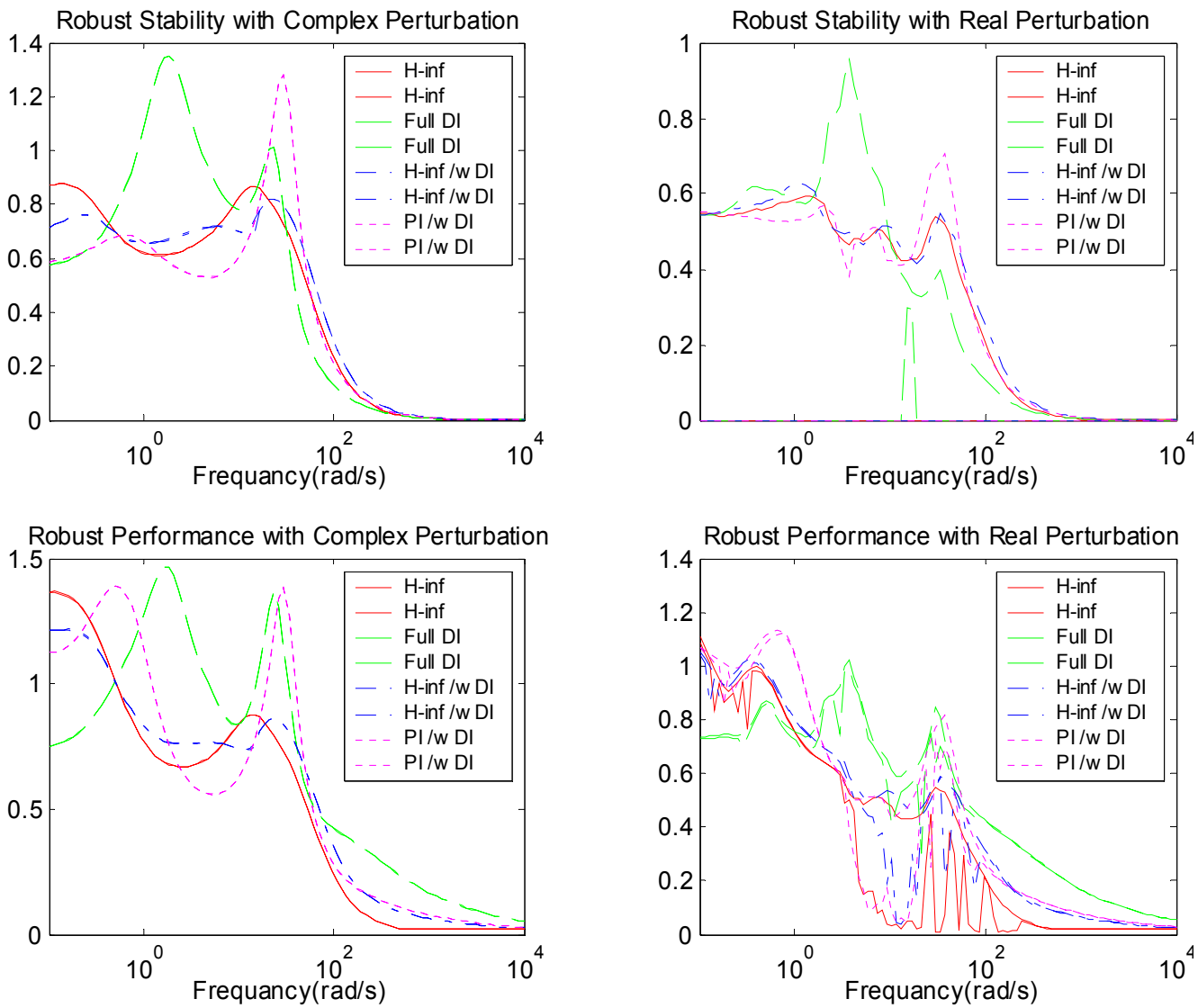

Figure 6-5: Closed loop system stability and performance with a maximum parametric uncertainty error of 50%

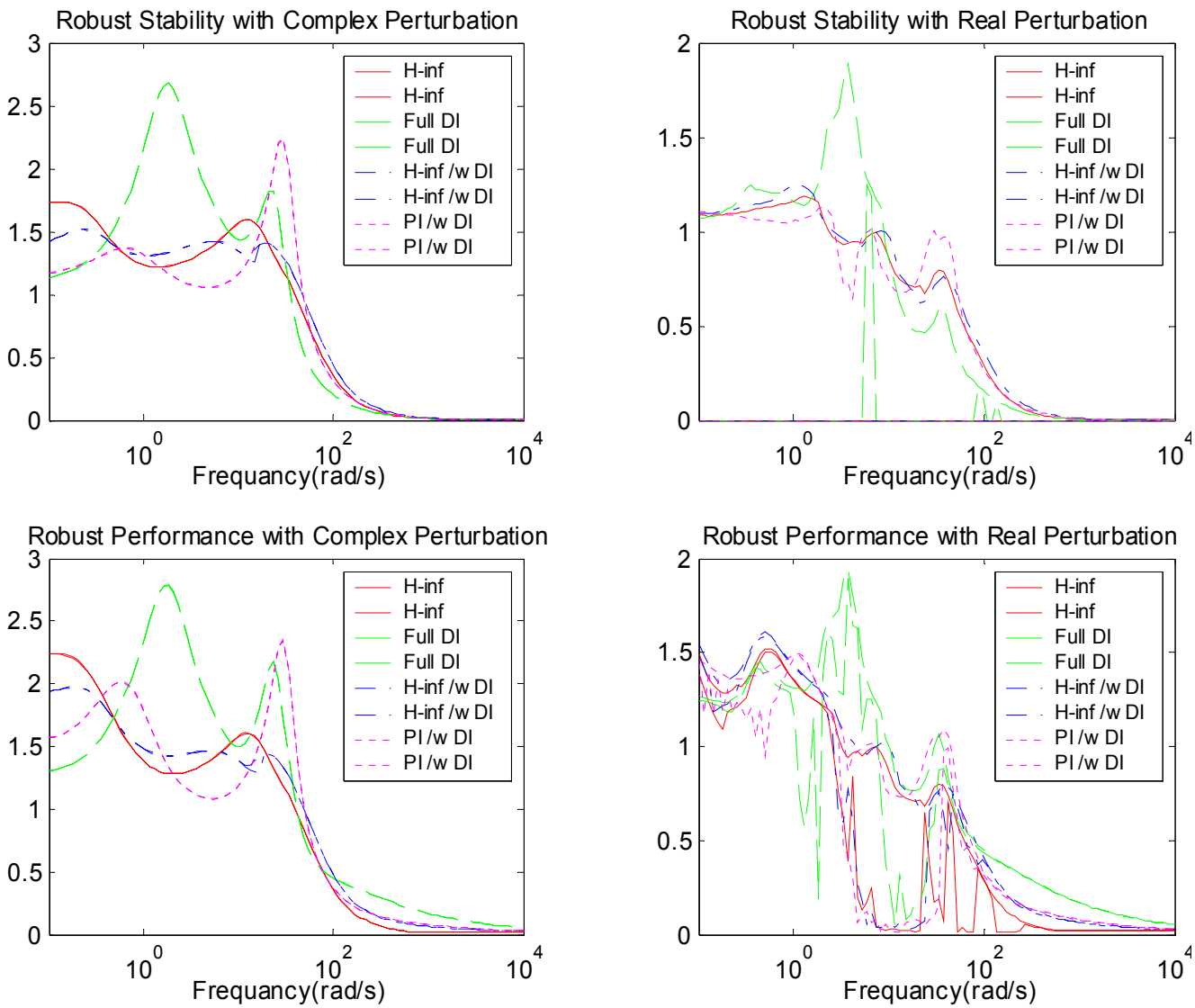

Figure 6-6: Closed loop system stability and performance with a maximum parametric uncertainty error of 100%

From the figures above we see that all systems perform better with real perturbations than complex uncertainty. Both the full dynamic inversion and the dynamic inversion loop coupled with a simple PI control show a quicker degradation in stability and performance with increasing uncertainty; however, the full DI controller has better nominal or near nominal performance. A better way of visualizing this is by looking at the worst case uncertainty matrix and the performance of the system with increasing uncertainty magnitude, figure 6-7. By

using robust techniques to compute the worst case uncertainty matrix large samplings of system uncertainty commonly used in Monte Carlo analysis can be avoided.

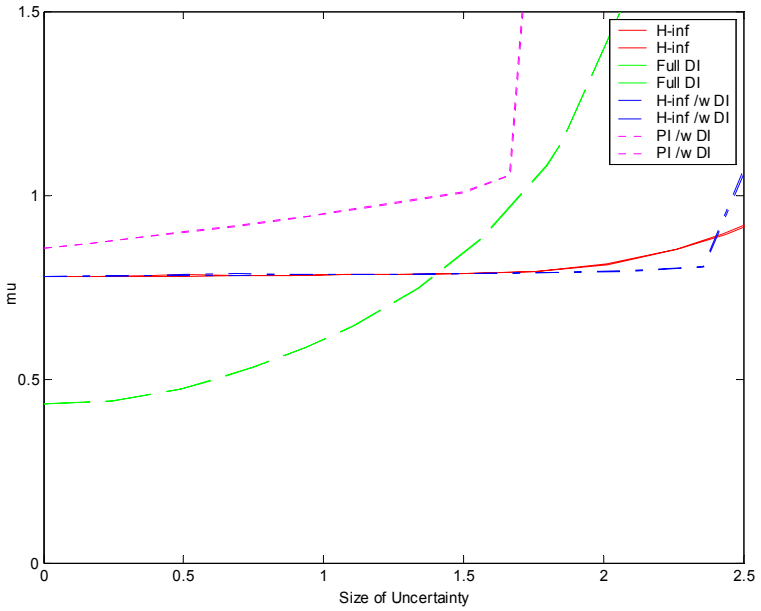

Figure 6-7: Worst case performance with varying uncertainty

7. Controller Model Reduction

Once a set of controllers has been found which satisfy given robust stability and performance requirements, it is necessary to simplify the controllers' structure for easier integration into the actual system. This simplified controller design is found through balanced realization and error bounded truncation. Given a minimal controller realization, a balanced realization is obtained by computing the controllability and observability gramians: $P > 0, Q > 0$ respectively. Next a matrix R is found such that:

$$P = R^* R \qquad \textbf{Eqn. 7-1}$$

We then diagonalize Q by R and obtain the following relation:

$$RQR^* = U\Sigma^2 U^* \qquad \textbf{Eqn. 7-2}$$

A system transformation matrix T can then be found and used to compute the balanced realization and the Hankel singular values, Σ.

$$T^{-1} = R^* U \Sigma^{-1/2} \qquad \textbf{Eqn. 7-3}$$

$$TPT^* = \left(T^*\right)^{-1} QT^{-1} = \Sigma \ \textbf{and} \ G_{bal} = \begin{bmatrix} TAT^{-1} & TB \\ CT^{-1} & D \end{bmatrix}$$

This process can be done in Matlab using the *sysbal* command. With the balanced realization obtained, one can examine the Hankel singular values and find a reduced order controller such that the following norm relation is maintained.

$$\left\| G(s) - G_r(s) \right\|_\infty \le 2\left(\sigma_{r+1} + \sigma_{r+2} + \ldots + \sigma_N\right) \le 0.1 \qquad \textbf{Eqn. 7-4}$$

where $G_r(s)$ is the reduced controller and $\Sigma = diag(\sigma_1, \ldots, \sigma_N)$ with $\sigma_i \le \sigma_{i+1} \ i \in 1, \ldots, N$

This relation will make sure the reduced controller will still satisfy the robust requirements. The H_∞ controller acting solely on the open loop was reduced from a 19th order to only a 3rd order system and still satisfied the norm relation. The

second controller, coupling the H_∞ controller with the DI inner loop, was reduced from a 19th order to only a 14th order. Figure 7-1 shows the change in robust stability and robust performance for each controller with reduction in system order. In all systems there is an increased degradation at lower frequencies with reduced order models and at the peak μ value.

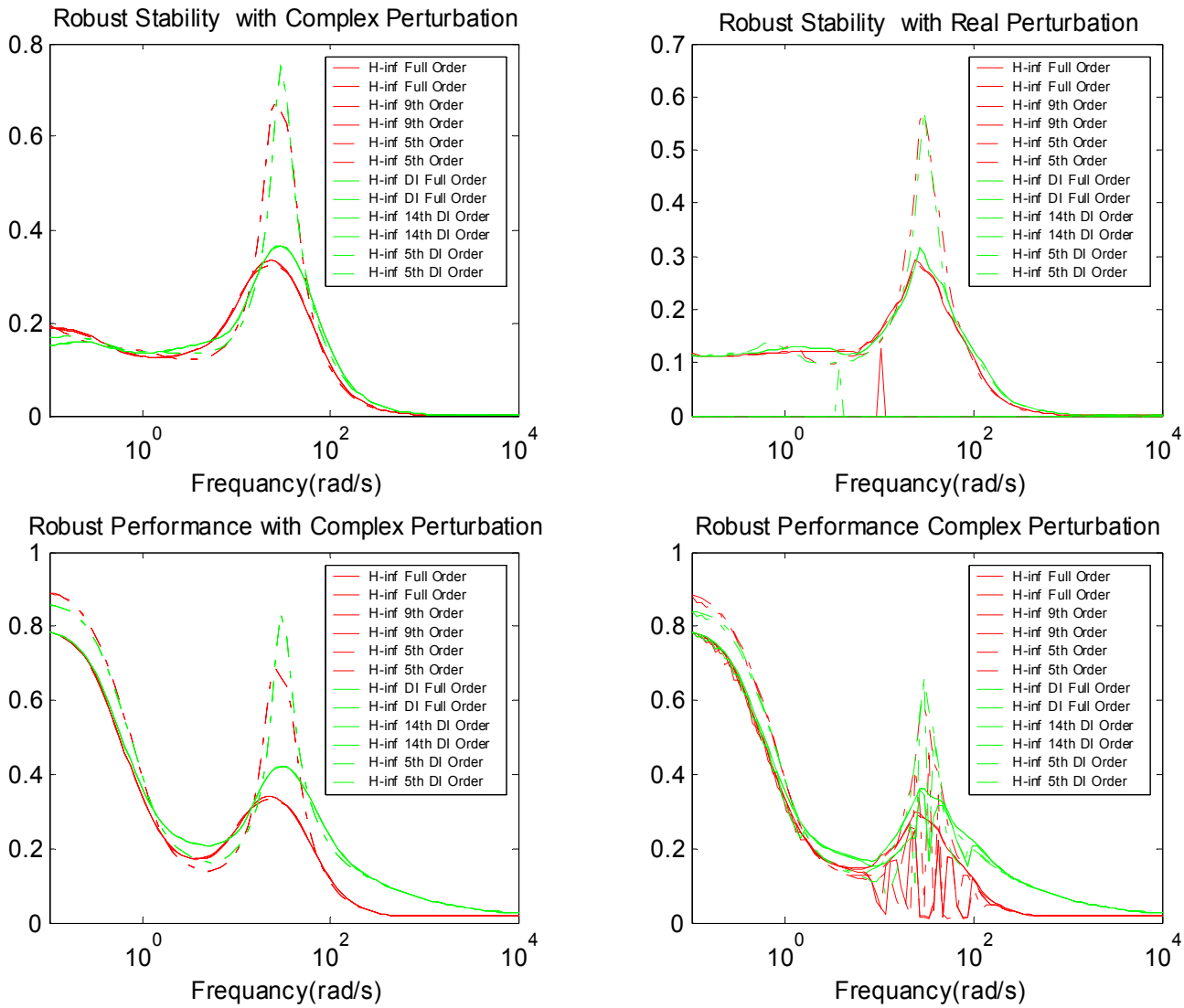

Figure 7-1: Robust stability and robust performance of the H_∞ controllers with varying system order

Similarly the degradation can be observed from the worst case performance graph with varying uncertainty, see figure 7-2. Notice how the performance of the lower order controllers is very similar to the PI controller.

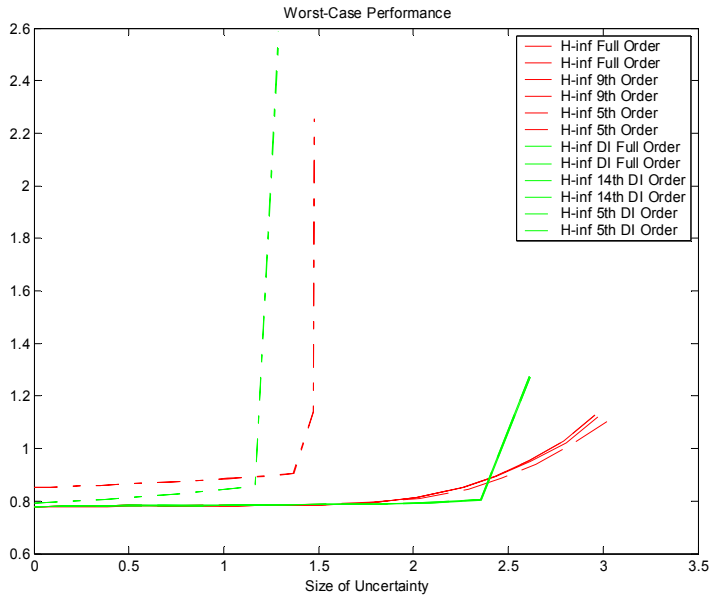

Figure 7-2: worst case performance with varying uncertainty of the H_∞ controllers

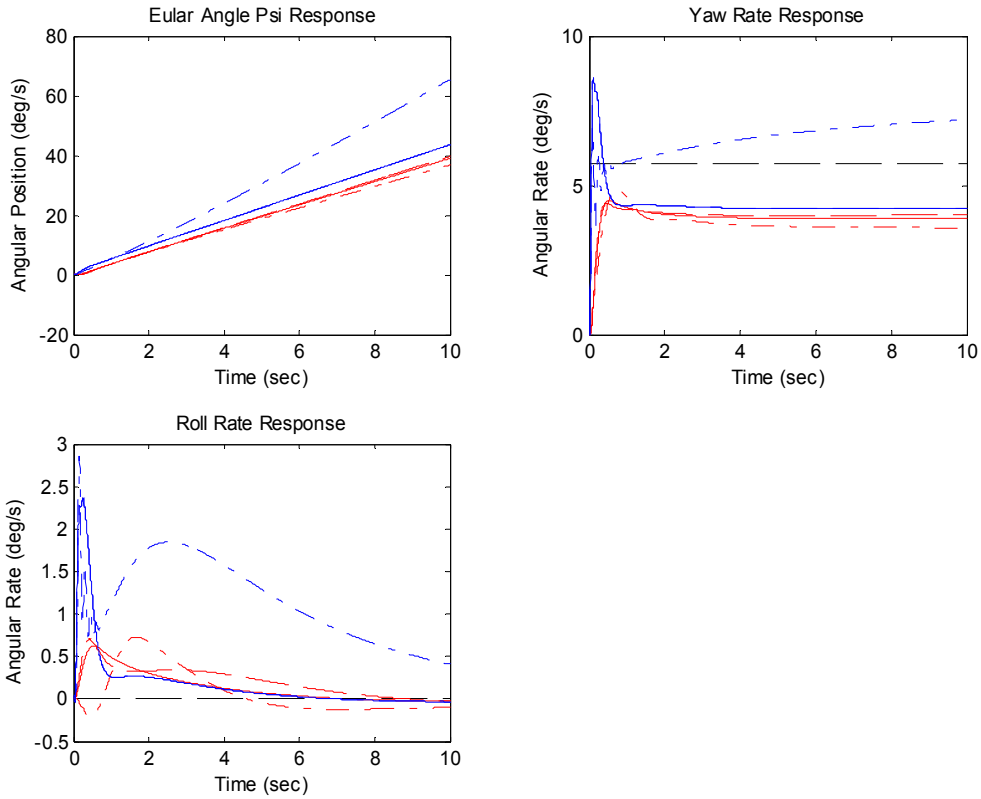

Figure 7-3: Time response of the H_∞ controller (red) and the H_∞ controller coupled with the DI inner loop (blue) with varying order: Full order —, .1 error norm reduction — —, 5th order — · —

8. Conclusions

Each of the four UAV controller designs developed and illustrated in the previous sections have shown to give the closed loop system robust stability and robust performance. However, the margin of robustness varies between the controllers, and the complexity of execution also delineates one from the other. The H_∞ controller acting solely on the open loop system has the best mu response, both in stability and performance, with lower peak values. The downside to this controller can be seen in the time simulations where the steady state error is larger than other system responses. Secondly, a single controller cannot provide desired performance throughout the entire flight envelope, requiring the use of gain scheduling.

The H_∞ controller coupled with an inner dynamic inversion loop does allow for a single controller to stabilize the entire flight envelope; however, the system order of the outer loop cannot be reduced as much and still maintain performance. This controller provides a robustness margin comparable to the single H_∞ controller and, due to the difference in reduced system orders, requires similar computational power.

The dynamic inversion loop coupled with a proportional integral control is one of the simpler controller forms; however, the cost of reduction in controller size can easily be seen in the loss of robustness. Looking at the 5th order reductions of the H_∞ controllers, we see the mu bounds approach the mu bounds of the PI controller. This is an expected response since the PI is simply a second order for of the H_∞ controller.

Finally the full dynamic inversion loop offers the best response computationally; however, the robustness of the system is much smaller than the three previous designs discussed. While this design is the easiest to implement with little to no computation required, there is a detrimental cost to the system's robust performance and stability, which can only be overcome with highly accurate model dynamics.

Appendix I: mkACLat

```
% Program Name:          mkACLat
% Description:           Performs robust stability and robust
%                        performance analysis of the closed
%                        loop lateral dynamics
% Previously Modified: 10/03/05
% Last Modified:         10/16/05
% Programmer:            Joseph Peterson


% Compute Parametric uncertainty
%       The uncertainty is given by the general form
%       a=a_nom+k*delta where k=a_nom*percent_error
%-----------------------------------------------------------------------
error=1; % Percent error
k1p=Lp*error;
k1da=Lda*error;
k1r=Nr*error;
k1dr=Ndr*error;
k2p=0;
k2da=0;
k2r=0;
k2dr=0;


% Form Plant with parametric uncertainty
%       LatDyn:  5 states, 9 outputs, 6 input
%       INPUTS:                 OUTPUTS:
%               1) wp                   1) zp
%               2) Wda                  2) zda
%               3) wr                   3) zr
%               4) wdr                  4) zdr
%               5) da                   5) beta
%               6) dr                   6) p
%                                       7) r
%                                       8) phi
%                                       9) psi
%-----------------------------------------------------------------------
GA=[Ybeta/V Yp/V -(1.0-(Yr/V)) g*cos(thetaO)/V  0.0
        Lbeta     Lp    Lr              0.0                   0.0
        Nbeta     Np    Nr              0.0                   0.0
        0.0       1.0   tan(thetaO)     0.0                   0.0
        0.0       0.0   sec(thetaO)     0.0                   0.0];

GB=[0.0  0.0    0.0  0.0    0.0   Ydr/V
```

```
        k1p  k1da 0.0  0.0    Lda  Ldr
        0.0  0.0    k1r  k1dr Nda  Ndr
        0.0  0.0    0.0  0.0    0.0    0.0
        0.0  0.0    0.0  0.0    0.0    0.0];

GC=[0.0  1.0  0.0  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
        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];

GD=[k2p  0.0  0.0  0.0   0.0  0.0
        0.0  k2da 0.0  0.0   1.0  0.0
        0.0  0.0    k2r  0.0   0.0  0.0
        0.0  0.0    0.0  k2dr 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  0.0
        0.0  0.0    0.0  0.0   0.0  0.0
        0.0  0.0    0.0  0.0   0.0  0.0
        0.0  0.0    0.0  0.0   0.0  0.0];

LatDyn=pck(GA,GB,GC,GD);

% Weighting Functions
%-----------------------------------------------------------------------
wd=nd2sys([1 1],[1 100]);
wd=daug(wd,wd);                                   % disturbance
wn_rate=nd2sys([.003 .0003],[2 1]);
wn_nois=nd2sys([.07 .0007],[0.5 1]);
wn=daug(wn_rate,wn_rate,wn_nois,wn_nois);   % noise
wp=nd2sys([1 10],[1 1]);
wp=daug(wp,wp);                                   % performance
wa=[3.85 0.0     0.0  0.0                         % actuator limits
        0.0   0.088 0.0  0.0
        0.0   0.0    1.91 0.0
        0.0   0.0    0.0   0.088];
wc=nd2sys([0.04 0.08],[2 1]);
wc=daug(wc,wc);                                   % rate command
```

```
% Second Order Actuator Model of The HS-5925MG Servo
%    The actuator models each have 1 input
%    and 2 outputs.  They are listed below:
%
%    ACTRUD:  4 states, 2 outputs, 1 input
%            OUTPUTS:                        INPUTS:
%                    1) position             1) rudder_cmd
%                    2) rate
%
%    ACTAIL:  4 states, 2 outputs, 1 input
%            OUTPUTS:                        INPUTS:
%                    1) position          1) elevon_cmd
%                    2) rate
%-----------------------------------------------------------------------
wrud = 52.3;
zetarud = 1.40;
wail = 52.3;
zetaail = 1.40;

int1 = nd2sys(1,[1 0]);
int2 = nd2sys(1,[1 0]);
c1 = wrud^2;
c2 = wrud^2;
c3 = 2*zetarud*wrud;
systemnames = 'c1 c2 c3 int1 int2';
inputvar = '[u]';
outputvar = '[int2;int1]';
input_to_c1 = '[u]';
input_to_c2 = '[int2]';
input_to_c3 = '[int1]';
input_to_int1 = '[c1-c2-c3]';
input_to_int2 = '[int1]';
sysoutname = 'actrud';
cleanupsysic = 'yes';
sysic

int1 = nd2sys(1,[1 0]);
int2 = nd2sys(1,[1 0]);
c1 = wail^2;
c2 = wail^2;
c3 = 2*zetaail*wail;
systemnames = 'c1 c2 c3 int1 int2';
```

```
inputvar = '[u]';
outputvar = '[int2;int1]';
input_to_c1 = '[u]';
input_to_c2 = '[int2]';
input_to_c3 = '[int1]';
input_to_int1 = '[c1-c2-c3]';
input_to_int2 = '[int1]';
sysoutname = 'actail';
cleanupsysic = 'yes';
sysic

% Full Dynamic Inversion (DI) Controller
%
%       Kdi: 2 outputs, 4 input
%       INPUTS:                 OUTPUTS:
%               1) p                    1) da
%               2) r                    2) dr
%               3) phi
%               4) psi
%-----------------------------------------------------------------------------
kp= 10; kr=30; kphi=1; kpsi=1;
Kdi=inv([Lda Ldr; Nda Ndr])*…
        [kp*kphi 0 -kp-Lp -kp*tan(thetaO)-Lr -kp*kphi 0;
         0 kr*kpsi/sec(thetaO) -Np -kr-Nr 0 -kr*kpsi/sec(thetaO)];

% Dynamic Inversion (DI) Controller Separated Parts
%       Kdii1: 2 outputs, 4 inputs
%       INPUTS:                 OUTPUTS:
%               1) pdotdes                      1) eda
%               2) rdotdes                      2) edr
%               3) p
%               4) r
%
%       Kdii2: 2 outputs, 2 inputs
%       INPUTS:                 OUTPUTS:
%               1) eda                  1) da
%               2) edr                  2) dr
%
%       desdynP: 2 outputs, 4 inputs
%       INPUTS:                 OUTPUTS:
%               1) pdes                 1) pdotdes
%               2) rdes                 2) rdotdes
%               3) pmes
%               4) rmes
%-----------------------------------------------------------------------------
Kdii1=[1 0 -Lp -Lr; 0 1 -Np -Nr];
```

```
Kdii2=inv([Lda Ldr; Nda Ndr]);
kp_p= 10; kr_p=30;
desdynP=[kp_p 0 -kp_p 0; 0 kr_p 0 -kr_p];

% Proportional Integral (PI) Controller Outer Loop
%-----------------------------------------------------------------------
kp_pi=10; kr_pi=10;
desdynPI=[(35) 0 -(35) 0 (3)^2/4 0; 0 (45) 0 -(45) 0 (1)^2/4];
int=daug(int1,int1);

% Create Open-Loop System Interconnection for the
% Aerosonde Lateral Dynamics
%
%       LatSys: 20 outputs, 14 inputs
%-----------------------------------------------------------------------
systemnames=' LatDyn wd wn wp wa wc actrud actail ';
inputvar='[pertin{4}; dist{2}; noise{4}; cmd{2}; ailcmd; rudcmd]';
outputvar='[LatDyn(1:4); wd; wp; wa; wc; LatDyn(6:9)+wn]';
input_to_LatDyn='[pertin; actail(1); actrud(1)]';
input_to_wd='[ailcmd; rudcmd]';
input_to_wn='[noise]';
input_to_wp='[wc(1)-LatDyn(6); wc(2)-LatDyn(7)]';
input_to_wa='[actail; actrud]';
input_to_wc='[cmd]';
input_to_actrud='[rudcmd+dist(1)]';
input_to_actail='[ailcmd+dist(2)]';
sysoutname='LatSys';
cleanupsysic='yes';
sysic;
fprintf('Plant Info \n');
minfo(LatSys)

% Creat Closed-Loop System Interconnection with Full DI Control
%
%       CL_di: 18 outputs, 12 inputs
%-----------------------------------------------------------------------
systemnames=' LatDyn wd wn wp wa wc actrud actail Kdi ';
inputvar='[pertin{4}; dist{2}; noise{4}; cmd{2}]';
outputvar='[LatDyn(1:4); wd; wp; wa(1); wa(2); wa(3); wa(4)]';
input_to_LatDyn='[pertin; actail(1); actrud(1)]';
input_to_wd='[Kdi]';
input_to_wn='[noise]';
input_to_wp='[0*wc(1); wc(2)-LatDyn(9)]';
input_to_wa='[actail; actrud]';
input_to_wc='[cmd]';
input_to_actrud='[Kdi(2)+dist(1)]';
```

```
input_to_actail='[Kdi(1)+dist(2)]';
input_to_Kdi='[wc; LatDyn(6:9)+wn]';
sysoutname='CL_di';
cleanupsysic='yes';
sysic;
fprintf('Plant Info \n');
minfo(LatSys)

% Creat Closed-Loop System Interconnection with P Inner DI Control
%
%       LatSysDI: 20 outputs, 14 inputs
%-----------------------------------------------------------------------------
systemnames=' LatDyn wd wn wp wa wc actrud actail Kdii1 Kdii2 desdynP ';
inputvar='[pertin{4}; dist{2}; noise{4}; cmd{2}; ailcmd; rudcmd]';
outputvar='[LatDyn(1:4); wd; wp; wa; wc; LatDyn(6:9)+wn]';
input_to_LatDyn='[pertin; actail(1); actrud(1)]';
input_to_wd='[Kdii2]';
input_to_wn='[noise]';
input_to_wp='[wc(1)-LatDyn(6); wc(2)-LatDyn(7)]';
input_to_wa='[actail; actrud]';
input_to_wc='[cmd]';
input_to_actrud='[Kdii2(2)+dist(1)]';
input_to_actail='[Kdii2(1)+dist(2)]';
input_to_Kdii1='[desdynP; LatDyn(6:7)+wn(1:2)]';
input_to_Kdii2='[Kdii1(1)+ailcmd; Kdii1(2)+rudcmd]';
input_to_desdynP='[wc; LatDyn(6:7)+wn(1:2)]';
sysoutname='LatSysDI';
cleanupsysic='yes';
sysic;
fprintf('Plant Info \n');
minfo(LatSys)

% Creat Closed-Loop System Interconection with PI Inner DI Control
%
%       LatSysDIPI: 12 outputs, 12 inputs
%-----------------------------------------------------------------------------
systemnames=' LatDyn wd wn wp wa wc actrud actail Kdii1 Kdii2 desdynPI int ';
inputvar='[pertin{4}; dist{2}; noise{4}; cmd{2}]';
outputvar='[LatDyn(1:4); wd; wp; wa]';
input_to_LatDyn='[pertin; actail(1); actrud(1)]';
input_to_wd='[Kdii2]';
input_to_wn='[noise]';
input_to_wp='[wc(1)-LatDyn(6); wc(2)-LatDyn(7)]';
input_to_wa='[actail; actrud]';
input_to_wc='[cmd]';
input_to_actrud='[Kdii2(2)+dist(1)]';
```

```
input_to_actail='[Kdii2(1)+dist(2)]';
input_to_Kdii1='[desdynPI; LatDyn(6:7)+wn(1:2)]';
input_to_Kdii2='[Kdii1(1); Kdii1(2)]';
input_to_desdynPI='[wc; LatDyn(6:7)+wn(1:2); int]';
input_to_int='[wc(1)-LatDyn(6); wc(2)-LatDyn(7)]';
sysoutname='LatSysDIPI';
cleanupsysic='yes';
sysic;
fprintf('Plant Info \n');
minfo(LatSys)

% Construct H-inf Controller Using hinflmi Command
%-----------------------------------------------------------------------
LatSys_temp=sel(LatSys,[7:18],[7:14]);
[opt,K_hinfl]=hinflmi(LatSys_temp,[6,2],0,0.001);

LatSysDI_temp=sel(LatSysDI,[7:18],[7:14]);
[opt,Kdi_hinfl]=hinflmi(LatSysDI_temp,[6,2],0,0.001);

% Form Closed-Loop System Using Star Product
%-----------------------------------------------------------------------
CL_hinfl=starp(LatSys,K_hinfl,6,2);
CLdi_hinfl=starp(LatSysDI,Kdi_hinfl,6,2);

% Compute Frequency Response Model
%-----------------------------------------------------------------------
omega=logspace(-1,4,100);
CL_hinfl_f=frsp(CL_hinfl,omega);
CL_di_f=frsp(CL_di,omega);
CLdi_hinfl_f=frsp(CLdi_hinfl,omega);
CLdipi_hinfl_f=frsp(LatSysDIPI,omega);

% Compute Nominal System Performance and Plot Results
%-----------------------------------------------------------------------
vplot('liv,m',vnorm(sel(CL_hinfl_f,7:12,7:12)),…
        '-r',vnorm(sel(CL_di_f,7:12,7:12)),'--g',…
        vnorm(sel(CLdi_hinfl_f,7:12,7:12)),'-.b',…
        vnorm(sel(CLdipi_hinfl_f,7:12,7:12)),':m');
title('Nominal Performance Of The Closed-Loop System')
xlabel('Frequancy(rad/s)')
legend('H-inf /w OL','Full DI','H-inf /w DI','PI /w DI')
temp=input('Hit any key to continue... \n\n');
```

```
% Compute Systems Robust Stability with Complex Perturbation
%-----------------------------------------------------------------------
delta_rs_c=ones(6,2);
CL_hinfl_frs=sel(CL_hinfl_f,1:6,1:6);
CL_di_frs=sel(CL_di_f,1:6,1:6);
CLdi_hinfl_frs=sel(CLdi_hinfl_f,1:6,1:6);
CLdipi_hinfl_frs=sel(CLdipi_hinfl_f,1:6,1:6);
[bnds_hinfl,dvec,sens_hinfl,pvec_hinfl]=…
        mu(CL_hinfl_frs,delta_rs_c,'c');
[bnds_di,dvec,sens_di,pvec_di]=…
        mu(CL_di_frs,delta_rs_c,'c');
[bnds_hinfldi,dvec,sens_hinfldi,pvec_hinfldi]=…
        mu(CLdi_hinfl_frs,delta_rs_c,'c');
[bnds_hinfldipi,dvec,sens_hinfldipi,pvec_hinfldipi]=…
        mu(CLdipi_hinfl_frs,delta_rs_c,'c');

% Generate first subplot
subplot(221)
vplot('liv,d',bnds_hinfl,'-r',bnds_di,'--g',bnds_hinfldi,…
        '-.b',bnds_hinfldipi,':m');
title('Robust Stability of the Controller with Complex Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf /w OL','H-inf /w OL','Full DI','Full DI',…
        'H-inf /w DI','H-inf /w DI','PI /w DI','PI /w DI')

% Compute Systems Robust Stability with Real Perturbation
%-----------------------------------------------------------------------
delta_rs_r=[-ones(6,1) zeros(6,1)];
CL_hinfl_frs=sel(CL_hinfl_f,1:6,1:6);
CL_di_frs=sel(CL_di_f,1:6,1:6);
CLdi_hinfl_frs=sel(CLdi_hinfl_f,1:6,1:6);
CLdipi_hinfl_frs=sel(CLdipi_hinfl_f,1:6,1:6);
[bnds_hinfl,dvec,sens_hinfl,pvec_hinfl]=…
        mu(CL_hinfl_frs,delta_rs_r,'c');
[bnds_di,dvec,sens_di,pvec_di]=mu(CL_di_frs,delta_rs_r,'c');
[bnds_hinfldi,dvec,sens_hinfldi,pvec_hinfldi]=…
        mu(CLdi_hinfl_frs,delta_rs_r,'c');
[bnds_hinfldipi,dvec,sens_hinfldipi,pvec_hinfldipi]=…
        mu(CLdipi_hinfl_frs,delta_rs_r,'c');

% Generate second subplot
subplot(222)
vplot('liv,d',bnds_hinfl,'-r',bnds_di,'--g',bnds_hinfldi,…
        '-.b',bnds_hinfldipi,':m');
title('Robust Stability of the Controller with Real Perturbation')
xlabel('Frequancy(rad/s)')
```

```
legend('H-inf /w OL','H-inf /w OL','Full DI','Full DI',…
        'H-inf /w DI','H-inf /w DI','PI /w DI','PI /w DI')

% Compute Systems Robust Performance with Complex Perturbation
%-----------------------------------------------------------------------------
delta_rp=[delta_rs_c; [6 6]];
[rbnds_hinfl, ph]=mu(CL_hinfl_f,delta_rp,'c');
[rbnds_di, ph]=mu(CL_di_f,delta_rp,'c');
[rbnds_hinfldi, ph]=mu(CLdi_hinfl_f,delta_rp,'c');
[rbnds_hinfldipi, ph]=mu(CLdipi_hinfl_f,delta_rp,'c');

% Generate third subplot
subplot(223)
vplot('liv,d',rbnds_hinfl,'-r',rbnds_di,'--g',rbnds_hinfldi,…
        '-.b',rbnds_hinfldipi,':m');
title('Robust Performance of the Controller with Complex Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf /w OL','H-inf /w OL','Full DI','Full DI',…
        'H-inf /w DI','H-inf /w DI','PI /w DI','PI /w DI')


% Compute Systems Robust Performance with Real Perturbation
%-----------------------------------------------------------------------------
delta_rp=[delta_rs_r; [6 6]];
[rbnds_hinfl, ph]=mu(CL_hinfl_f,delta_rp,'c');
[rbnds_di, ph]=mu(CL_di_f,delta_rp,'c');
[rbnds_hinfldi, ph]=mu(CLdi_hinfl_f,delta_rp,'c');
[rbnds_hinfldipi, ph]=mu(CLdipi_hinfl_f,delta_rp,'c');

% Generate fourth subplot
subplot(224)
vplot('liv,d',rbnds_hinfl,'-r',rbnds_di,'--g',rbnds_hinfldi,…
        '-.b',rbnds_hinfldipi,':m');
title('Robust Performance of the Controller with Real Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf /w OL','H-inf /w OL','Full DI','Full DI',…
        'H-inf /w DI','H-inf /w DI','PI /w DI','PI /w DI')
temp=input('Hit any key to continue... \n\n');
```

```
% Compute the Worst Case Perturbation with Varying Uncertainty
%-----------------------------------------------------------------------
[deltabad_hinfl,low_hinfl,upp_hinfl]=…
        wcperf(CL_hinfl_f,delta_rs_c,0.5,10);
[deltabad_di,low_di,upp_di]=…
        wcperf(CL_di_f,delta_rs_c,0.5,10);
[deltabad_hinfldi,low_hinfldi,upp_hinfldi]=…
        wcperf(CLdi_hinfl_f,delta_rs_c,0.5,10);
[deltabad_hinfldipi,low_hinfldipi,upp_hinfldipi]=…
        wcperf(CLdipi_hinfl_f,delta_rs_c,0.5,10);

% Generate Plot
vplot(low_hinfl,'-r',upp_hinfl,'-r',low_di,'--g',upp_di,…
        '--g',low_hinfldi,'-.b',upp_hinfldi,'-.b',…
        low_hinfldipi,':m',upp_hinfldipi,':m');
title('Worst-Case Performance')
xlabel('Size of Uncertainty')
legend('H-inf /w OL','H-inf /w OL','Full DI','Full DI'…
        ,'H-inf /w DI','H-inf /w DI','PI /w DI','PI /w DI')
temp=input('Hit any key to continue... \n\n');

% Balanced Realization of the H-inf Controller
%-----------------------------------------------------------------------
[K_bal,hanksv1]=sysbal(K_hinfl,0);
[Kdi_bal,hanksv2]=sysbal(Kdi_hinfl,0);
% Reduction of the h-inf controller
K_balred1=strunc(K_bal,9);
K_balred2=strunc(K_bal,5);
Kdi_balred1=strunc(Kdi_bal,14);
Kdi_balred2=strunc(Kdi_bal,5);

% Test Error Associated with Reduction of H-inf Controller
%-----------------------------------------------------------------------
test1=hinfnorm(msub(K_hinfl,K_balred1))
test2=hinfnorm(msub(K_hinfl,K_balred2))
test3=hinfnorm(msub(Kdi_hinfl,Kdi_balred1))
test4=hinfnorm(msub(Kdi_hinfl,Kdi_balred2))

% Form Closed-Loop System with Reduced Order Controller
%-----------------------------------------------------------------------
CL_hinfl_balred1=starp(LatSys,K_balred1,6,2);
CL_hinfl_balred2=starp(LatSys,K_balred2,6,2);
CLdi_hinfl_balred1=starp(LatSysDI,Kdi_balred1,6,2);
CLdi_hinfl_balred2=starp(LatSysDI,Kdi_balred2,6,2);
```

```
% Compute Closed-Loop System Frequency Response
%-----------------------------------------------------------------------
CL_hinfl_balred1_f=frsp(CL_hinfl_balred1,omega);
CL_hinfl_balred2_f=frsp(CL_hinfl_balred2,omega);
CLdi_hinfl_balred1_f=frsp(CLdi_hinfl_balred1,omega);
CLdi_hinfl_balred2_f=frsp(CLdi_hinfl_balred2,omega);

CL_hinfl_balred1_frs=sel(CL_hinfl_balred1_f,1:6,1:6);
CL_hinfl_balred2_frs=sel(CL_hinfl_balred2_f,1:6,1:6);
CLdi_hinfl_balred1_frs=sel(CLdi_hinfl_balred1_f,1:6,1:6);
CLdi_hinfl_balred2_frs=sel(CLdi_hinfl_balred2_f,1:6,1:6);

% Compute robust stability and robust performance
%-----------------------------------------------------------------------
[bnds_hinfl,dvec,sens_hinfl,pvec_hinfl]=…
        mu(CL_hinfl_frs,delta_rs_c,'c');
[bnds_hinf2,dvec,sens_hinf2,pvec_hinf2]=…
        mu(CL_hinfl_balred1_frs,delta_rs_c,'c');
[bnds_hinf3,dvec,sens_hinf3,pvec_hinf3]=…
        mu(CL_hinfl_balred2_frs,delta_rs_c,'c');
[bnds_hinf4,dvec,sens_hinf4,pvec_hinf4]=…
        mu(CLdi_hinfl_frs,delta_rs_c,'c');
[bnds_hinf5,dvec,sens_hinf5,pvec_hinf5]=…
        mu(CLdi_hinfl_balred1_frs,delta_rs_c,'c');
[bnds_hinf6,dvec,sens_hinf6,pvec_hinf6]=…
        mu(CLdi_hinfl_balred2_frs,delta_rs_c,'c');

subplot(221)
vplot('liv,d',bnds_hinfl,'-r',bnds_hinf2,'--r',bnds_hinf3,…
        '-.r',bnds_hinf4,'-g',bnds_hinf5,'--g',bnds_hinf6,'-.g');
title('Robust Stability of the H-inf Controller of Varying…
         Order with Complex Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf Full Order','H-inf Full Order','H-inf 9th Order',…
        'H-inf 9th Order','H-inf 5th Order',…
        'H-inf 5th Order','H-inf DI Full Order',…
        'H-inf DI Full Order','H-inf 14th DI Order',…
        'H-inf 14th DI Order','H-inf 5th DI Order',…
        'H-inf 5th DI Order')

[bnds_hinfl,dvec,sens_hinfl,pvec_hinfl]=…
        mu(CL_hinfl_frs,delta_rs_r,'c');
[bnds_hinf2,dvec,sens_hinf2,pvec_hinf2]=…
        mu(CL_hinfl_balred1_frs,delta_rs_r,'c');
[bnds_hinf3,dvec,sens_hinf3,pvec_hinf3]=…
        mu(CL_hinfl_balred2_frs,delta_rs_r,'c');
```

```
[bnds_hinf4,dvec,sens_hinf4,pvec_hinf4]=…
        mu(CLdi_hinfl_frs,delta_rs_r,'c');
[bnds_hinf5,dvec,sens_hinf5,pvec_hinf5]=…
        mu(CLdi_hinfl_balred1_frs,delta_rs_r,'c');
[bnds_hinf6,dvec,sens_hinf6,pvec_hinf6]=…
        mu(CLdi_hinfl_balred2_frs,delta_rs_r,'c');

subplot(222)
vplot('liv,d',bnds_hinfl,'-r',bnds_hinf2,'--r',bnds_hinf3,…
        '-.r',bnds_hinf4,'-g',bnds_hinf5,'--g',bnds_hinf6,'-.g');
title('Robust Stability of the H-inf Controller of Varying…
         Order with Real Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf Full Order','H-inf Full Order',…
        'H-inf 9th Order','H-inf 9th Order',…
        'H-inf 5th Order','H-inf 5th Order',…
        'H-inf DI Full Order','H-inf DI Full Order',…
        'H-inf 14th DI Order','H-inf 14th DI Order',…
        'H-inf 5th DI Order','H-inf 5th DI Order')

delta_rp=[delta_rs_c; [6 6]];
[rbnds_hinfl, ph]=mu(CL_hinfl_f,delta_rp,'c');
[rbnds_hinf2, ph]=mu(CL_hinfl_balred1_f,delta_rp,'c');
[rbnds_hinf3, ph]=mu(CL_hinfl_balred2_f,delta_rp,'c');
[rbnds_hinf4, ph]=mu(CLdi_hinfl_f,delta_rp,'c');
[rbnds_hinf5, ph]=mu(CLdi_hinfl_balred1_f,delta_rp,'c');
[rbnds_hinf6, ph]=mu(CLdi_hinfl_balred2_f,delta_rp,'c');

subplot(223)
vplot('liv,d',rbnds_hinfl,'-r',rbnds_hinf2,'--r',rbnds_hinf3,…
        '-.r',rbnds_hinf4,'-g',rbnds_hinf5,'--g',rbnds_hinf6,'-.g');
title('Robust Performance of the H-inf Controller of Varying…
         Order with Complex Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf Full Order','H-inf Full Order',…
        'H-inf 9th Order','H-inf 9th Order',…
        'H-inf 5th Order','H-inf 5th Order',…
        'H-inf DI Full Order','H-inf DI Full Order',…
        'H-inf 14th DI Order','H-inf 14th DI Order',…
        'H-inf 5th DI Order','H-inf 5th DI Order')

delta_rp=[delta_rs_r; [6 6]];
[rbnds_hinfl, ph]=mu(CL_hinfl_f,delta_rp,'c');
[rbnds_hinf2, ph]=mu(CL_hinfl_balred1_f,delta_rp,'c');
[rbnds_hinf3, ph]=mu(CL_hinfl_balred2_f,delta_rp,'c');
[rbnds_hinf4, ph]=mu(CLdi_hinfl_f,delta_rp,'c');
```

```
[rbnds_hinf5, ph]=mu(CLdi_hinfl_balred1_f,delta_rp,'c');
[rbnds_hinf6, ph]=mu(CLdi_hinfl_balred2_f,delta_rp,'c');

subplot(224)
vplot('liv,d',rbnds_hinfl,'-r',rbnds_hinf2,'--r',rbnds_hinf3,…
        '-.r',rbnds_hinf4,'-g',rbnds_hinf5,'--g',rbnds_hinf6,'-.g');
title('Robust Performance of the H-inf Controller of Varying…
         Order with Complex Perturbation')
xlabel('Frequancy(rad/s)')
legend('H-inf Full Order','H-inf Full Order',…
        'H-inf 9th Order','H-inf 9th Order',…
        'H-inf 5th Order','H-inf 5th Order',…
        'H-inf DI Full Order','H-inf DI Full Order',…
        'H-inf 14th DI Order','H-inf 14th DI Order',…
        'H-inf 5th DI Order','H-inf 5th DI Order')
temp=input('Hit any key to continue... \n\n');

% Compute the Worst Case Perturbation
%-----------------------------------------------------------------------
[deltabad_hinfl,low_hinfl,upp_hinfl]=…
        wcperf(CL_hinfl_f,delta_rs_c,0.5,10);
[deltabad_hinf2,low_hinf2,upp_hinf2]=…
        wcperf(CL_hinfl_balred1_f,delta_rs_c,0.5,10);
[deltabad_hinf3,low_hinf3,upp_hinf3]=…
        wcperf(CL_hinfl_balred2_f,delta_rs_c,0.5,10);
[deltabad_hinf4,low_hinf4,upp_hinf4]=…
        wcperf(CLdi_hinfl_f,delta_rs_c,0.5,10);
[deltabad_hinf5,low_hinf5,upp_hinf5]=…
        wcperf(CLdi_hinfl_balred1_f,delta_rs_c,0.5,10);
[deltabad_hinf6,low_hinf6,upp_hinf6]=…
        wcperf(CLdi_hinfl_balred2_f,delta_rs_c,0.5,10);
vplot(low_hinfl,'-r',upp_hinfl,'-r',low_hinf2,'--r',upp_hinfl,…
        '--r',low_hinf3,'-.r',upp_hinf3,'-.r',low_hinf4,'-g',…
        upp_hinf4,'-g',low_hinf5,'--g',upp_hinf5,'--g',…
        low_hinf6,'-.g',upp_hinf6,'-.g');
title('Worst-Case Performance')
xlabel('Size of Uncertainty')
legend('H-inf Full Order','H-inf Full Order',…
        'H-inf 9th Order','H-inf 9th Order',…
        'H-inf 5th Order','H-inf 5th Order',…
        'H-inf DI Full Order','H-inf DI Full Order',…
        'H-inf 14th DI Order','H-inf 14th DI Order',…
        'H-inf 5th DI Order','H-inf 5th DI Order')
temp=input('Hit any key to continue... \n\n');
```

Appendix II: linac

```
% Program Name:        linac
% Description:         Formulates the Longitudinal and
%                      Lateral sate and control Matrixes
% Previously Modified: 10/18/04
% Last Modified:       11/04/04
% Programmer:          Joseph Peterson

%INPUTS:    Mach number
%           Density (kg/m^3)
%           Ref. Drag
%           Ref. Lift
%           Ref. angle (rad)
% Compute Aircraft Parameters and Assign Constants
%-----------------------------------------------------------------------------
M=.0958;
rho=1.1674;
CDo=0.0449;
CLo=0.3043;
thetaO=0.0116;
CDa=0.4;
g=9.81;
V=M*331.5;
CDu=CDM*M;
Cmu=CmM*M;
CLu=CLM*M;
Q=0.5*rho*V^2;
C=[1 0 0 0 ; 0 1 0 0 ; 0 0 1 0 ; 0 0 0 1];
D=[0 ; 0 ; 0 ; 0 ];
D1=[0 0; 0 0; 0 0; 0 0];

% Longitudinal Dynamic Stability Derivatives
%-----------------------------------------------------------------------------
Xu=-(CDu+2*CDo)*Q*S/(mgross*V);
Xw=-(CDa-CLo)*Q*S/(mgross*V);
Xde=-CDde*Q*S;
Zu=-(CLu+2*CLo)*Q*S/(mgross*V);
Zw=-(CLa+CDo)*Q*S/(mgross*V);
Za=V*Zw;
Zde=-CLde*Q*S/mgross; % Czde=-CLde
Mu=Cmu*Q*S*MAC/(V*Jgross(2)); % CMu=(dCm/dM)*Mo
Mw=Cma*Q*S*MAC/(V*Jgross(2));
Mwdot=Cmalphadot*MAC^2*Q*S/(2*V^2*Jgross(2));
Mq=Cmq*MAC^2*Q*S/(2*V*Jgross(2));
```

```
Ma=V*Mw;
Malphadot=V*Mwdot;
Mde=Cmde*Q*S*MAC/Jgross(2);

% Lateral-Directional Stability Derivatives
%-----------------------------------------------------------------------
Nbeta=Cnbeta*Q*S*b/Jgross(3);
Np=Cnp*Q*S*b^2/(2*Jgross(3)*V);
Nr=Cnr*Q*S*b^2/(2*Jgross(3)*V);
Ndr=Cndr*Q*S*b/Jgross(3);
Nda=Cnda*Q*S*b/Jgross(3);
Lbeta=Clbeta*Q*S*b/Jgross(3);
Lp=Clp*Q*S*b^2/(2*Jgross(3)*V);
Lr=Clr*Q*S*b^2/(2*Jgross(3)*V);
Ldr=Cldr*Q*S*b/Jgross(3);
Lda=Clda*Q*S*b/Jgross(3);
Ybeta=CYbeta*Q*S/mgross;
Yp=CYp*Q*S*b/(2*mgross*V);
Yr=CYr*Q*S*b/(2*mgross*V);
Ydr=CYdr*Q*S/mgross;

% Longitudinal Matrix Builder
%-----------------------------------------------------------------------
LonA=[Xu                Xw                 0                -g
      Zu                Zw                 V                0.0
      Mu+Mwdot*Zu       Mw+Mwdot*Zw        Mq+Mwdot*V       0.0
      0.0               0.0                1.0              0.0];

LonB=[Xde
      Zde
      Mde+Mwdot*Zde
      0.0];

% Construct Mu-Toolbox Compatible Matrix
%-----------------------------------------------------------------------
LonDynF=pck(LonA,LonB,C,D);
LonDynP=pck(LonA,LonB,[0 0 1 0; 0 0 0 1],[0; 0]);

% Construct LTI Toolbox Compatible System
%-----------------------------------------------------------------------
Lonstates = {'Forward Velocity' 'Vertical Speed' 'Pitchrate' 'Theta'};
Loninputs = {'elevator'};
Lonoutputs ={'Forward Velocity' 'Vertical Speed' 'Pitchrate' 'Theta'};
Lonsys=ss(LonA,LonB,C,D,'statename',Lonstates,…
      'inputname',Loninputs,'outputname',Lonoutputs);
```

```
% Lateral Matrix Builder
%-----------------------------------------------------------------------
LatA=[Ybeta/V Yp/V -(1.0-(Yr/V))  g*cos(thetaO)/V        0.0
      Lbeta    Lp     Lr           0.0                   0.0
      Nbeta    Np     Nr           0.0                   0.0
      0.0      1.0    tan(thetaO)  0.0                   0.0
      0.0      0.0    sec(thetaO)  0.0                   0.0];

LatB=[0.0 Ydr/V
      Lda Ldr
      Nda Ndr
      0.0   0.0
      0.0   0.0];

% Construct LTI Toolbox Compatible System
%-----------------------------------------------------------------------
Latstates = {'beta' 'rollrate' 'yawrate' 'phi' 'psi'};
Latinputs = {'rudder' 'aileron'};
Latoutputs ={'beta' 'rollrate' 'yawrate' 'phi' 'psi'};
C=[1 0 0 0 0; 0 1 0 0 0; 0 0 1 0 0; 0 0 0 1 0; 0 0 0 0 1];
D1=[0 0; 0 0; 0 0; 0 0; 0 0];
Latsys=ss(LatA,LatB,C,D1,'statename',Latstates,…
       'inputname',Latinputs,'outputname',Latoutputs);
```

Appendix III: Linearsim

```
% Program Name:          linearsim
% Description:           runs the Simulink model Linear-Solution03
%                        for varying levels of disturbance
% Previously Modified: 09/27/05
% Last Modified:         10/03/05
% Programmer:            Joseph Peterson


% Assign Initial Parametric Disturbance
%-----------------------------------------------------------------------
delta=zeros(4,4);

% Assign Controller and Command Values
%-----------------------------------------------------------------------
[Ka,Kb,Kc,Kd]=unpck(K_hinfl);
[Ka1,Kb1,Kc1,Kd1]=unpck(Kdi_hinfl);
pcmd=0.0;
rcmd=0.1;
numsim=10;
del=.2;

% Open Simulink Model and Run Nominal Solution
%-----------------------------------------------------------------------
open_system('Linear_Solution03');
sim('Linear_Solution03');
nominal=simout;

% Iterate Increasing Magnitude of Parametric
% Disturbance and Simulate
%-----------------------------------------------------------------------
for i=1:numsim/2
   d=del*i;
   delta=[d 0 0 0; 0 d 0 0; 0 0 d 0; 0 0 0 d];
   sim('Linear_Solution03');
   sys(i)=simout;
end
```

```
% Iterate Decreasing Magnitude of Parametric
% Disturbance and Simulate
%-----------------------------------------------------------------------
for i=1:numsim/2
   d=-del*i;
   delta=[d 0 0 0; 0 d 0 0; 0 0 d 0; 0 0 0 d];
   sim('Linear_Solution03');
   sys(i+numsim/2)=simout;
end

% Close Simulation Model and Clear Command Screen
%-----------------------------------------------------------------------
close_system('linear_Solution03');
clc
```

References

[1] Ledger, David, "Electronic Warfare Capabilities of Mini UAVs," Proceeding of the Electronic Warfare Conference (Kuala Lumpur), 2002.

[2] Reiner, Jacob, Balas, Gary J. and, Garrard, William L., "Robust Dynamic Inversion for Control of Highly Maneuverable Aircraft," Journal of Guidance, Control, and Dynamics, Vol. 18, No. 1, 1995, p.18-24.

[3] Snell, Antony S., and Stout, Perry W., "Robust Longitudinal Control Design Using Dynamic Inversion and Quantitative Feedback Theory," Journal of Guidance, Control, and Dynamics, Vol. 20, No. 5, 1997, p.933-940.

[4] Snell, Antony S., Enns, Dale F., and Garrard, William L., "Nonlinear Inversion Flight Control for a Supermaneuverable Aircraft," Journal of Guidance, Control, and Dynamics, Vol. 15, No. 4., 1992, p.916-984.

[5] Brinker, Joseph S., and Wise, Kevin A., "Stability and Flying Qualities Robustness of a Dynamic Inversion Aircraft Control Law," Journal of Guidance, Control, and Dynamics, Vol. 19, No. 6, 1996, p.1270-1277.

[6] Anon., "Military Standard Flying Qualities of Piloted Aircraft," Mil-STD-1797A, Jan 30, 1990.

[7] Doman, David B., and Ngo, Anhtuan D., "Dynamic Inversion-Based Adaptive/Reconfigurable Control of the X-33 on Ascent," Journal of Guidance, Control, and Dynamics, Vol. 25, No. 2, 2002, p. 275-284.

[8] Zhou, Kemin, and Doyle, John C., Essentials of Robust Control, Prentice Hall, New Jersey, 1998, p.269-294.

[9] Nesterov, Yu, and Nemirovski, A., *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*, SIAM Books, Philadelphia, 1994.

[10] Balas, G. J., Doyle, J. C., Glover, K., Packard, A., and Smith, R., *μ-Analysis and Synthesis TOOLBOX for use with MATLAB*, The Mathworks Inc., First Edition, July 1993.

[11] Etkin, Bernard and Reid, Lloyd D., Dynamics of Flight Stability and Control, Third Edition, John Wily & Sons, Inc, 1996, p.161-203.

[12] *User's Guide: AeroSim Aeronautical Simulation Blockset*, Unmanned Dynamics, LLC, Version 1.1, 2004.

[13] Papoulis, A. "Wiener-Lévy Process." §15-3 in *Probability, Random Variables, and Stochastic Processes, 2nd ed.* New York: McGraw-Hill, pp. 292-293, 1984.

[14] "U.S. Military Handbook," MIL-HDBK-1797, Dec 19, 1997.

[15] Brown, James W. and Churchill, Ruel V., Complex Variables and Applications: Seventh Edition, McGraw-Hill, Inc. 2004, p 169-171.