Linear Parameter-Varying Path Following Control of a Small Fixed Wing Unmanned Aerial Vehicle

Kyle T. Guthrie

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfilment of the requirements for the degree of

Master of Science
in
Aerospace Engineering

Mazen Farhood, Chair
Craig Woolsey
Cornel Sultan

July 31, 2013
Blacksburg, Virginia

Keywords: Path Following, LPV Control, Robust Control, Unmanned Aerial Vehicles

Linear Parameter-Varying Path Following Control of a Small Fixed Wing Unmanned Aerial Vehicle

Kyle T. Guthrie

(ABSTRACT)

A mathematical model of a small fixed-wing aircraft was developed through application of parameter estimation techniques to simulated flight test data. Multiple controllers were devised based on this model for path following, including a self-scheduled linear parameter-varying (LPV) controller with path curvature as a scheduling parameter. The robustness and performance of these controllers were tested in a rigorous MATLAB simulation environment that included steady winds and gusts, measurement noise, delays, and model uncertainties. The linear controllers designed within were found to be robust to the disturbances and uncertainties in the simulation environment, and had similar or better performance in comparison to a nonlinear control law operating in an inner-outer loop structure. Steps are being taken to implement the resulting controllers on the unmanned aerial vehicle (UAV) testbed in the Nonlinear Systems Laboratory at Virginia Tech.

To my family and friends

Acknowledgments

I'd first like to thank my advisor, Dr. Farhood, for his continued support and dedication over the last 3 years. Dr. Farhood always believed in my abilities, and always strived to bring out the best in me. I'd also like to thank my committee members, Dr. Woolsey and Dr. Sultan, for their support and guidance.

I'd like to thank my family for their immeasurable love and support during my time in school. You've all given me so much to strive for, and I'd like to think I'm one step closer to following in your footsteps.

I'd like to thank all my friends and peers who stood by me (and kept me sane) during my time here at Virginia Tech: Joseph Gooding, Brandon Hull, Jeff Garnand-Royo, Sebastian Fave, Zach Harlow, Mark Palframan, Dave Grymin, and Ankit Ganeriwal. I'd like to especially thank Mr. Ony Arifianto, for his incredible generosity and assistance with all my work, especially in the final few months.

To Mr. Roger Chang, I can't even imagine how different the past six years of my life would have been had I not had your mentorship and guidance. This achievement is as much yours as it is mine. I am truly honored to consider you a friend.

Last, but certainly not least, I'd like to thank Ms. Katie Prokop, for being a constant source of motivation and encouragement. I can't thank you enough for your kindness, but more importantly your patience.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The use of small-scale unmanned aerial vehicles (UAVs) in a variety of missions is becoming more common place in today's world of aviation. One issue that plagues many small aircraft is the presence of relatively significant environmental disturbances, namely steady winds and wind gusts. Owing to their size, small aircraft are affected much more dramatically than traditional aircraft in the presence of what may be considered small environmental disturbances. Traditional trajectory tracking methods with time-stamped inertial position feedback tend to produce poor results when relatively significant disturbances are present. Without inertial feedback, these methods may need to be supplemented with path planning algorithms to compensate for drifting from the reference trajectory. Path following control methods, on the other hand, aim to track a geometric reference without temporal specification, and can potentially handle stronger disturbances in general than trajectory tracking methods. In fact, it has been shown that path following control is often able to remove performance limitations present in traditional reference tracking control methods [1].

A natural benefit of path following control is the splitting of the control problem into two parts, the first being to track a geometric path parameterized by a path variable. The

secondary objective is to track a specified dynamic characterization of the path variable, traditionally chosen as a speed profile [2]. Path following control has been shown to have many useful applications to UAVs, primarily involving missions related to surveillance, imaging, formation flight, and station keeping [3, 4, 5].

Path following control has been applied in a variety of manners. Some notable approaches include standard waypoint guidance [6], using vector fields to generate course commands to drive the vehicle to the path [7], and virtual vehicle formulations [8, 9], where a fictitious vehicle that is constrained to the path is tracked. Waypoint guidance is advantageous in its simplicity of application, but is generally limited to simple paths. Vector field approaches allow for the inclusion of more interesting geometric paths, and the implementation is again simple, but the generation of vector fields can be complicated, and the paths are often restricted to be planar. Virtual vehicle approaches are the most flexible in terms of the geometric paths that can be followed, at the cost of introducing additional dynamics for the virtual vehicle system.

Much of the existing path following literature employs nonlinear control methods, such as backstepping, sliding mode, and adaptive control, for examples see [9, 10]. In this work, an H_∞ approach is employed to solve the path following control problem; specifically, linear time-invariant (LTI) and linear parameter-varying (LPV) controllers are designed with the H_∞ norm used as the performance measure. One of the advantages of formulating the problem as a standard H_∞ control problem is that it allows easy extension to tools from the vast literature on robust control, such as designing optimal control systems and analyzing the robustness of these systems.

The control design methodology used in this work is model based, and so it is important to obtain a reasonably accurate model of the UAV in order to be able to design high-performance control systems. As high fidelity UAV models go hand in hand with high costs, especially

in the case of wind tunnel testing, methods to derive aerodynamic models of UAVs from flight testing are very attractive. These methods are well studied, and have been applied to a wide variety of platforms [11, 12, 13, 14]. Of the most commonly applied approaches are the Ordinary Least Squares, Filter-Error, and Output-Error approaches.

An approach combining these areas of study in the design of controllers for a small fixed-wing UAV is presented within. The path following problem is presented in a virtual vehicle formulation adapted from the literature. As implementations of the developed control systems will be carried out in simulation only, we will assume that a United States Air Force Stability and Control Digital DATCOM [15] model of a six foot wingspan Telemaster UAV represents the physical system. Then, we will derive an aerodynamic model using the Ordinary Least Squares and Output Error parameter estimation techniques on simulated longitudinal and lateral-directional flight test maneuvers. Based on this aerodynamic model, four separate controllers will be designed for the path following problem; two LTI controllers and two LPV controllers. Extensive simulation in a rigorous simulation environment will be performed to test the path following performance of each controller, with the goal of eventually applying the controllers designed within on the physical UAV system.

The outline of this thesis is as follows: Chapter 2 will cover the background information of UAV dynamics, the path following problem formulation, and the control design methods used in this work. Some basic techniques commonly used in aerodynamic parameter estimation are covered in Chapter 3. The specifics of each controller synthesis are presented in Chapter 4. Flight simulation results of the various path following controllers are covered in Chapter 5. Finally, planned implementation on a real UAV system and possible topics of future work are covered in Chapters 6.

Chapter 2

Background

2.1 Definitions and Notation

Much of the notation contained within is adapted from [16]. $\mathcal{F}_I$ is the Earth-fixed inertial reference frame, whose axes are $\{\vec{x}_I, \vec{y}_I, \vec{z}_I\}$, which point in the North, East, and downward directions, respectively. $\mathcal{F}_b$ is the body fixed reference frame, with origin at the UAV center of mass. The axes of this frame are $\{\vec{x}_b, \vec{y}_b, \vec{z}_b\}$, which point along the fuselage reference line, towards the right wing tip, and downwards in the aircraft plane of symmetry, respectively. A depiction of these frames, along with the Euler angles that relate them, is shown in Figure 2.1. $\mathcal{F}_w$ is the wind axes reference frame, with origin at the UAV center of mass. The axes of this frame are $\{\vec{x}_w, \vec{y}_w, \vec{z}_w\}$. These axes are obtained by a left-hand rotation about the y_b axis by the aerodynamic angle $\alpha = \arctan(w_a/u_a)$ and a right-hand rotation about the resulting z axis by the aerodynamic angle $\beta = \arcsin(v_a/\sqrt{u_a^2 + v_a^2 + w_a^2})$, where $u_a = u - u_g$, $v_a = v - v_g$, and $w_a = w - w_g$ are the aircraft body-axis aircraft velocity components minus the body-axis wind gust components. The reference frame in which a vector is expressed is denoted by subscript, e.g. $\boldsymbol{v}_c$ is the vector $\boldsymbol{v}$ expressed in the $\mathcal{F}_c$ frame. A rotation matrix

$\boldsymbol{R}$ mapping from reference frame $\mathcal{F}_c$ into reference frame $\mathcal{F}_d$ is denoted by $\boldsymbol{R}_{dc}$.

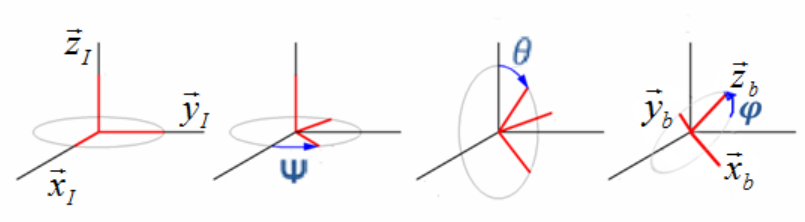


Figure 2.1: Aircraft reference frames and Euler angle rotations

The aircraft velocities and orientations are defined as follows: $\boldsymbol{w}_b^T = [p, q, r]$ is the vector of angular velocities about the body $\vec{x}_b$, $\vec{y}_b$, and $\vec{z}_b$ axes, respectively, $\boldsymbol{v}_b^T = [u, v, w]$ is the vector of translational velocities relative to the surrounding air, and $\boldsymbol{\Phi}^T = [\phi, \theta, \psi]$ is the vector of Euler angles, denoted bank angle, elevation angle, and heading angle, respectively. The magnitude of the velocity vector, also called the measured airspeed, is denoted $V_a = \sqrt{(u-u_g)^2 + (v-v_g)^2 + (w-w_g)^2}$. The Euler angles ϕ, θ, and ψ define the rotation $\boldsymbol{R}_{Ib}$: $\mathcal{F}_b \mapsto \mathcal{F}_I$. The velocity of surrounding air relative to the inertial frame is denoted $\boldsymbol{V}_{w_I}^T = [V_{wx}, V_{wy}, V_{wz}]$. The flight path angle of the aircraft is denoted $\gamma = \theta - \alpha$. The vector of control inputs is $\boldsymbol{\delta} = [\delta_e, \delta_a, \delta_r, \delta_T]^T$, consisting of elevator, aileron, rudder, and throttle deflection, respectively.

The aircraft forces and moments are defined as follows: $\boldsymbol{M}_b^T = [M_x, M_y, M_z]$ is the vector of total moments acting on the aircraft, including thrust effects. $\boldsymbol{M}_{A_b}^T = [\ell, m, n]$ is the vector of aerodynamic moments, denoted as roll, pitch, and yaw moments, respectively. $\boldsymbol{F}_b^T = [F_x, F_y, F_z]$ is the vector of total forces acting on the aircraft, including thrust effects. $\boldsymbol{F}_{A_b}^T = [X_A, Y_A, Z_A]$ is the vector of aerodynamic forces. In order to avoid confusion with aerodynamic pitching moment, $\|\boldsymbol{W}\|/g$ is used to represent the vehicle mass, where $\boldsymbol{W}_b^T = [W_x, W_y, W_z]$ represents the body-fixed components of the aircraft weight, and $\|\boldsymbol{W}\|$ represents the Euclidean norm of $\boldsymbol{W}$.

2.2 Aircraft Equations of Motion

Using Newton's second law for rigid-body dynamics as shown in Equation 2.1, where all vectors are expressed in the body-fixed frame, the equations of motion for the aircraft can be formulated. Note that symmetry about the $x_b - z_b$ plane is assumed, which simplifies the form of the inertia tensor.

$$\begin{aligned} \boldsymbol{F} + \boldsymbol{W} &= \frac{d}{dt}\left(\frac{\|\boldsymbol{W}\|}{g}\boldsymbol{V}\right) + \boldsymbol{\omega} \times \left(\frac{\|\boldsymbol{W}\|}{g}\boldsymbol{V}\right) \\ \boldsymbol{M} &= \frac{d}{dt}\left([\boldsymbol{I}]\,\boldsymbol{\omega}\right) + \boldsymbol{\omega} \times \left([\boldsymbol{I}]\,\boldsymbol{\omega}\right) \\ [\boldsymbol{I}] &= \begin{bmatrix} I_{xx} & 0 & -I_{xz} \\ 0 & I_{yy} & 0 \\ -I_{xz} & 0 & I_{zz} \end{bmatrix} \end{aligned} \tag{2.1}$$

where I_{xx}, I_{yy}, I_{zz} are the aircraft moments of inertia, and I_{xz} is an aircraft product of inertia. Making substitutions for previously defined quantities and defining the body-axis weight components using the Euler angles, the equations of motion for the velocity components associated with the six degrees of freedom of the aircraft become as given in Equations 2.2 and 2.3.

$$\begin{Bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{Bmatrix} = \frac{g}{\|\boldsymbol{W}\|} \begin{Bmatrix} F_x \\ F_y \\ F_z \end{Bmatrix} + g \begin{Bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{Bmatrix} + \begin{Bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{Bmatrix} \tag{2.2}$$

$$\begin{Bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{Bmatrix} = \begin{bmatrix} I_{xx} & 0 & -I_{xz} \\ 0 & I_{yy} & 0 \\ -I_{xz} & 0 & I_{zz} \end{bmatrix}^{-1} \begin{Bmatrix} \ell + (I_{yy} - I_{zz})qr + I_{xz}pq \\ m + (I_{zz} - I_{xx})pr + I_{xz}(r^2 - p^2) \\ n + (I_{xx} - I_{yy})pq - I_{xz}qr \end{Bmatrix} \tag{2.3}$$

The map from $\mathcal{F}_I$ to $\mathcal{F}_b$ is described by the following series of rotations:

1. Rotate the $\{\vec{x}_I, \vec{y}_I, \vec{z}_I\}$ axes about the $\vec{z}_I$ axis through an angle ψ to obtain the coordinate system $\{\vec{x}_1, \vec{y}_1, \vec{z}_1\}$.

2. Rotate the $\{\vec{x}_1, \vec{y}_1, \vec{z}_1\}$ axes about the $\vec{y}_1$ axis through an angle θ to obtain the coordinate system $\{\vec{x}_2, \vec{y}_2, \vec{z}_2\}$.

3. Rotate the $\{\vec{x}_2, \vec{y}_2, \vec{z}_2\}$ axes about the $\vec{x}_2$ axis through an angle ϕ to obtain the coordinate system $\{\vec{x}_b, \vec{y}_b, \vec{z}_b\}$.

Assembling the various rotation matrices, a rotation matrix $\boldsymbol{R}_{Ib}$ is obtained:

$$\boldsymbol{R}_{Ib} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{2.4}$$

It is then simple to obtain an expression for the time rate of change of the inertial position vector of the aircraft by rotating the body-fixed velocity components into the intertial frame, and accounting for velocity of the surrounding air relative to the Earth-fixed frame, which results in the Earth-fixed position dynamics given in Equation 2.5.

$$\begin{Bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{Bmatrix} = \boldsymbol{R}_{Ib} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} + \begin{Bmatrix} V_{wx} \\ V_{wy} \\ V_{wz} \end{Bmatrix} \tag{2.5}$$

The set of equations of motion is completed through use of the Euler kinematic equation (Equation 2.6), which results in the Euler angle dynamic equations given in Equation 2.7.

$$\dot{\mathbf{\Phi}} = \boldsymbol{S}(\mathbf{\Phi})^{-1}\boldsymbol{\omega}^b$$
$$\boldsymbol{S}(\mathbf{\Phi})^{-1} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \tag{2.6}$$

$$\begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} \tag{2.7}$$

This completes the equations of motion for the Euler angle formulation of the aircraft dynamics. It is important to note that a singularity exists at $\theta = \pm 90^\circ$ in 2.7, however for this work the elevation angle will not approach this singularity.

2.3 Telemaster UAV

The Telemaster UAV (Figure 2.2) is a 6 foot wingspan off-the-shelf radio control (RC) aircraft built, maintained, and operated in the Nonlinear Systems Lab at Virginia Tech [17, 18]. The on-board sensor package includes a 3DM-GX3 Inertial Measurement Unit (IMU) / Attitude and Heading Reference System (AHRS), an Antaris-4T GPS receiver from U-Blox, an SCP1000-D01 static pressure sensor from VTI Technologies, and an airdata probe developed in-house which uses five MPXV7002DP differential pressure sensors from Freescale Semiconductor. These sensors feed to Ardupilot, an open-source autopilot hardware system used as an on-board data acquisition system [18]. The basic geometry and mass parameters for the Telemaster are shown in Table 2.1.

For the purposes of control design and simulation, an aerodynamic model of the Telemaster was required. An aerodynamic model of the UAV was obtained using the USAF Stability and Control Digital DATCOM [15]. DATCOM takes aircraft geometry parameters as inputs, and returns aerodynamic coefficients and stability derivatives. In general, the aerodynamic coefficients are dependent on the aerodynamic angles α and β, as well as control surface deflections δ_e, δ_a, and δ_r. A full tabular listing of the DATCOM model is included in Appendix B.

Figure 2.2: Telemaster UAV platform

Table 2.1: Telemaster UAV configuration data

Mass	3.24 kg
I_x	0.22 kg-m^2
I_y	0.31 kg-m^2
I_z	0.45 kg-m^2
Wing area	0.56 m^2
Wing span	1.83 m
Wing mean aerodynamic chord	0.30 m

2.4 Path Following

Path following control refers to the design of control laws that force a vehicle to converge to and follow a geometric path in space that is specified without time parameterization [19]. The control strategy developed in this section utilizes the fact that control of the vehicle attitude can drive the vehicle towards a desired position, in this case along a geometric path in space. In addition, a remaining degree of freedom remains in tracking a desired dynamic behavior, typically a speed profile. In this work, a constant speed profile is tracked. Much of the background, definitions, and formulation of the path following problem is borrowed from [9], with some modifications made for simplification or preference.

To begin, the concept of a virtual vehicle is introduced. The virtual vehicle is an imaginary point that moves along the path at some prescribed rate. At every point on the path, the virtual vehicle has an associated reference frame. Let $\boldsymbol{Q}_I = [Q_x, Q_y, Q_z]^T$ be the location of the UAV center of mass, let $\boldsymbol{P}_I = [P_x, P_y, P_z]^T$ be the location of the virtual vehicle

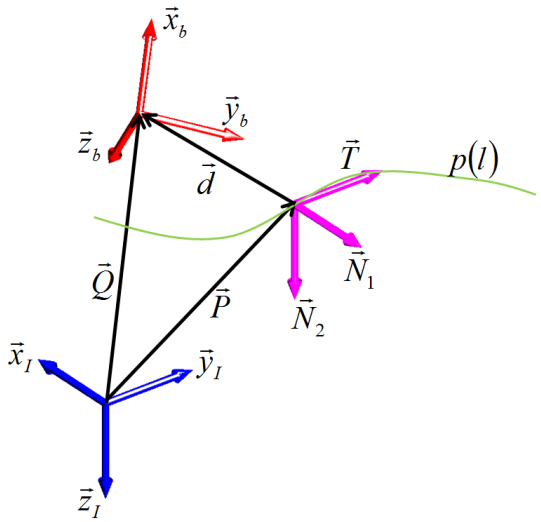


Figure 2.3: Parallel transport frame

center of mass, and let $\boldsymbol{p}(\ell)$ represent the path to be followed, parameterized by the path length ℓ. At each point on the path, a parallel transport frame (sometimes referred to as a "rotation minimizing frame") [20, 21], denoted $\mathcal{F}_p$, is affixed to the virtual vehicle. The three orthonormal basis vectors of $\mathcal{F}_p$, denoted $\vec{T}(\ell)$ (tangent vector), $\vec{N}_1(\ell)$ (first normal vector), and $\vec{N}_2(\ell)$ (second normal vector), satisfy the dynamics in Equation 2.8.

$$\begin{bmatrix} \frac{d\vec{T}(\ell)}{d\ell} \\ \frac{d\vec{N}_1(\ell)}{d\ell} \\ \frac{d\vec{N}_2(\ell)}{d\ell} \end{bmatrix} = \begin{bmatrix} 0 & k_1(\ell) & k_2(\ell) \\ -k_1(\ell) & 0 & 0 \\ -k_2(\ell) & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{T}(\ell) \\ \vec{N}_1(\ell) \\ \vec{N}_2(\ell) \end{bmatrix} \tag{2.8}$$

A visualization of the three frames in discussion is shown in Figure 2.3.

A parallel transport frame is chosen for this application over the more common Frenet-Serret frame because the Frenet-Serret frame becomes discontinuous for paths with locations of zero curvature [22]. As many desired flight paths contain locations of zero curvature (e.g. straight segments), the use of parallel transport frame is therefore advantageous. Also, unlike the

Frenet-Serret frame, the choice of the initial reference frame is not fixed for the parallel transport frame, but instead can be selected by the designer.

Using the three orthonormal basis vectors, the rotation between the path reference frame and inertial reference frames is $\boldsymbol{R}_{Ip} = [T(\ell), N_1(\ell), N_2(\ell)]$. Making an exception to the vector notation described previously, let ω^p_{pI} be the angular velocity of $\mathcal{F}_p$ with respect to $\mathcal{F}_I$, expressed in the $\mathcal{F}_I$ frame, which is equivalent to:

$$\begin{aligned}
\dot{\vec{e}}_i &= \vec{\omega}^p_{pI} \times \vec{e}_i \\
\dot{\vec{T}}(\ell) &= \frac{d\vec{T}(\ell)}{d\ell}\frac{d\ell}{dt} = \dot{\ell}k_1(\ell)\vec{N}_1(\ell) + \dot{\ell}k_2(\ell)\vec{N}_2(\ell) = \vec{\omega}^p_{pI} \times \vec{T}(\ell) \\
\dot{\vec{N}}_1(\ell) &= \frac{d\vec{N}_1(\ell)}{d\ell}\frac{d\ell}{dt} = -\dot{\ell}k_1(\ell)\vec{T}(\ell) = \vec{\omega}^p_{pI} \times \vec{N}_1(\ell) \\
\dot{\vec{N}}_2(\ell) &= \frac{d\vec{N}_2(\ell)}{d\ell}\frac{d\ell}{dt} = -\dot{\ell}k_2(\ell)\vec{T}(\ell) = \vec{\omega}^p_{pI} \times \vec{N}_2(\ell)
\end{aligned} \tag{2.9}$$

Expressing $\vec{\omega}^p_{pI}$ as $\vec{\omega}^p_{pI} = \omega_1\vec{T}(\ell) + \omega_2\vec{N}_1(\ell) + \omega_3\vec{N}_2(\ell)$, it can be seen that

$$\vec{\omega}^p_{pI} = [0, -\dot{\ell}k_2(\ell), \dot{\ell}k_1(\ell)]^T \tag{2.10}$$

Let $\boldsymbol{d}_p = \boldsymbol{P}_p - \boldsymbol{Q}_p = [d_x, d_y, d_z]^T$ be the difference of the UAV and virtual vehicle positions. Also, define a local UAV frame $\mathcal{W}'$, the projection of the UAV wind reference frame $\mathcal{W}$ onto the local level plane. This frames orientation can be described relative to the $\mathcal{F}_p$ frame defined earlier through a set of three Euler angles, $\Phi_{err} = [\phi_{err}, \theta_{err}, \psi_{err}]$. Through a small angle approximation, it is assumed that the UAV roll, pitch, and yaw rates (p, q, and r) in the $\mathcal{W}'$ frame are approximately equal to those in the body-fixed frame.

Re-writing the expression for $\boldsymbol{d}_p$, and taking the derivative:

$$\begin{aligned}
\boldsymbol{Q}_I &= \boldsymbol{P}_I + \boldsymbol{R}_{Ip}\boldsymbol{d}_p \\
\dot{\boldsymbol{Q}}_I &= \boldsymbol{R}_{Ip}\begin{bmatrix} \dot{\ell} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_{Ip}\begin{bmatrix} \dot{d}_x \\ \dot{d}_y \\ \dot{d}_z \end{bmatrix} + \boldsymbol{R}_{Ip}\left(\vec{\omega}_{pI}^{p} \times \begin{bmatrix} d_x \\ d_y \\ d_x \end{bmatrix}\right)
\end{aligned} \tag{2.11}$$

Recognizing that:

$$\boldsymbol{R}_{pI}\begin{bmatrix} \dot{Q}_x \\ \dot{Q}_y \\ \dot{Q}_z \end{bmatrix} = \boldsymbol{R}_{p\mathcal{W}'}\boldsymbol{R}_{\mathcal{W}'I}\begin{bmatrix} \dot{Q}_x \\ \dot{Q}_y \\ \dot{Q}_z \end{bmatrix} = \boldsymbol{R}_{p\mathcal{W}'}\begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix} \tag{2.12}$$

and substituting it back into Equation 2.11, after simplification it becomes:

$$\begin{cases}
\dot{d}_x &= -\dot{\ell}(1 - k_1(\ell)d_y - k_2(\ell)d_z) + V\cos\theta_{err}\cos\psi_{err} \\
\dot{d}_y &= -\dot{\ell}k_1(\ell)d_x + V\cos\theta_{err}\sin\psi_{err} \\
\dot{d}_z &= -\dot{\ell}k_2(\ell)d_x - V\sin\theta_{err}
\end{cases} \tag{2.13}$$

which represents the kinematic position error dynamics for the combined UAV and virtual vehicle system.

The attitude error dynamics can be derived in a similar fashion using the Euler kinematic equation (Equation 2.6). The angular velocity vector can be re-expressed as follows.

$$\begin{aligned}
\vec{\omega}_{\mathcal{W}'p}^{\mathcal{W}'} &= \vec{\omega}_{\mathcal{W}'I}^{\mathcal{W}'} - \vec{\omega}_{pI}^{\mathcal{W}'} \\
&= \vec{\omega}_{\mathcal{W}'I}^{\mathcal{W}'} - \boldsymbol{R}_{\mathcal{W}'p}\vec{\omega}_{pI}^{p}
\end{aligned} \tag{2.14}$$

Thus, Equation 2.6 becomes

$$\dot{\boldsymbol{\Phi}}_{err} = \boldsymbol{S}\left(\boldsymbol{\Phi}_{\boldsymbol{err}}\right)^{-1}\left(\vec{\omega}_{\mathcal{W}'I}^{\mathcal{W}'} - \boldsymbol{R}_{\mathcal{W}'p}\vec{\omega}_{pI}^{p}\right) \tag{2.15}$$

$$\begin{cases} \dot{\phi}_{err} = \dot{\ell} k_2(\ell) \sin\psi_{err} \sec\theta_{err} + p + q \sin\phi_{err} \tan\theta_{err} + r \cos\phi_{err} \tan\theta_{err} \\ \dot{\theta}_{err} = \dot{\ell} k_2(\ell) \cos\psi_{err} + q \cos\phi_{err} - r \sin\phi_{err} \\ \dot{\psi}_{err} = -\dot{\ell} \left(k_1(\ell) - k_2(\ell) \tan\theta_{err} \sin\psi_{err}\right) + q \sin\phi_{err} \sec\theta_{err} + r \cos\phi_{err} \sec\theta_{err} \end{cases} \tag{2.16}$$

Equations 2.13 and 2.16 describe the complete path following error system. Following the example set in [9], the error Euler angles θ_{err} and ψ_{err} are shaped using approach angle functions to produce better control performance. In a departure from the method used in [9], hyperbolic tangent functions are used as the approach angle functions for convenience. This transforms the path following state vector to

$$x = [d_x,\ d_y,\ d_z,\ \phi_{err},\ \theta_{err} - \delta_\theta,\ \psi_{err} - \delta_\psi]^T \tag{2.17}$$

where

$$\begin{aligned} \delta_\theta &= \theta_{app} \tanh\left(\frac{d_z}{C_1}\right) \\ \delta_\psi &= \psi_{app} \tanh\left(\frac{-d_y}{C_2}\right) \end{aligned} \tag{2.18}$$

with θ_{app} and ψ_{app} being the maximum desired approach angle, and C_1 and C_2 being scaling factors to determine at what magnitude of position error the maximum approach angle is attained. A sample depiction of one of these approach angles is shown in Figure 2.4.

The approach angles work to ensure that the vehicle is approaching the path at all times. Additionally, it gives an extra degree of freedom in the aggressiveness of the tracking behavior. High maximum angles and low scaling values result in aggressive corrections in terms of attitude to track the path, while low maximum angles and high scaling values result in passive corrections in terms of attitude. Over the course of this work, it was found that

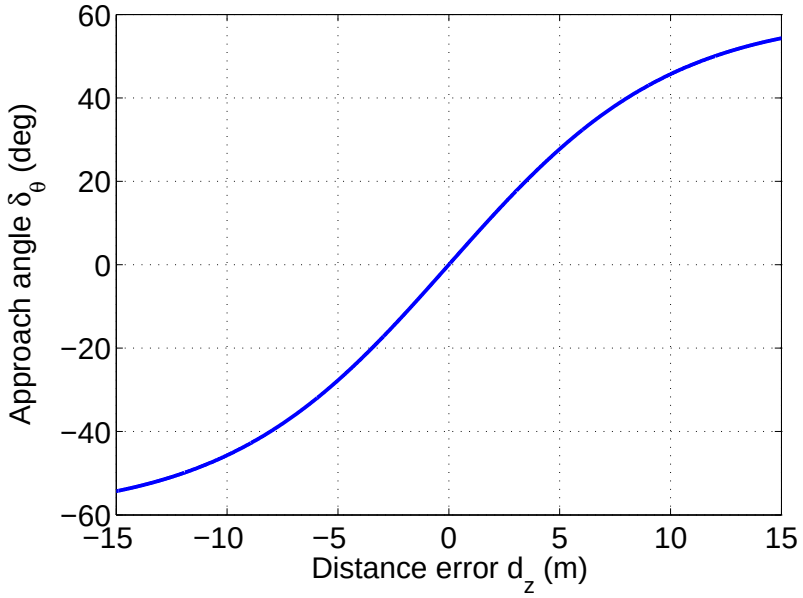


Figure 2.4: Sample approach angle function for $\theta_{app} = 60^\circ$ and $C_1 = 20$

the inclusion of these approach angles in the problem formulation resulted in increased path following performance in comparison to problem formulations that did not include these approach angles.

Finally, the dynamics of the virtual vehicle are chosen as

$$\dot{\ell}(t) = K_1 d_x(t) + V(t) \cos\theta_{err}(t) \cos\psi_{err}(t) \tag{2.19}$$

where K_1 is some positive constant.

The path following control objective then becomes: regulate the state vector 2.17 and hold a constant speed profile, subject to path following dynamics 2.13, 2.16, and 2.19, as well as the UAV flight dynamics 2.2, 2.3, 2.5, and 2.7 in the presence of disturbances.

In [9], a nonlinear control law is developed via backstepping, with q and r playing the role of virtual control inputs. The q and r commands generated by the nonlinear control law on the

outer loop are then tracked by a Piccolo auto pilot augmented by an $\mathcal{L}_1$ adaptive controller on the inner loop. In order to develop a rough point of reference, a similar approach will be taken in this work. The nonlinear control laws developed in [9] will be used to generate q and r commands, which will then be tracked by a standard $\mathcal{H}_\infty$ controller. The nonlinear control law developed in [9] to generate pitch and yaw rate commands q_c and r_c is given as follows.

$$\begin{aligned}
\begin{bmatrix} q_c \\ r_c \end{bmatrix} &= T^{-1}(\theta_{err}) \left(\begin{bmatrix} u_{\theta_c} \\ u_{\psi_c} \end{bmatrix} - D(\theta_{err}, \psi_{err} \right) \\
T(\theta_{err}) &= \begin{bmatrix} \cos\phi_{err} & -\sin\phi_{err} \\ \frac{\sin\phi_{err}}{\cos\theta_{err}} & \frac{\cos\phi_{err}}{\cos\theta_{err}} \end{bmatrix} \\
D(\theta_{err}, \psi_{err}) &= \begin{bmatrix} \dot{\ell} k_2(\ell) \cos\psi_{err} \\ -\dot{\ell}(k_1(\ell) - k_2(\ell)\tan\theta_{err}\sin\psi_{err} \end{bmatrix} \\
u_{\theta_c} &= -K_2(\theta_{err} - \delta_\theta) + \frac{c_2}{c_1} d_z V \frac{\sin\theta_{err} - \sin\delta_\theta}{\theta_{err} - \delta_\theta} + \dot{\delta}_\theta \\
u_{\psi_c} &= -K_3(\psi_{err} - \delta_\psi) - \frac{c_2}{c_1} d_y V \cos\theta_{err} \frac{\sin\psi_{err} - \sin\delta_\psi}{\psi_{err} - \delta_\psi} + \dot{\delta}_\psi
\end{aligned} \tag{2.20}$$

where $K_2 > 0$, $K_3 > 0$, and c_1/c_2 are positive constants that satisfy an inequality condition, the details of which can be found in [9, Equations 12 and 17].

2.5 H_∞ and LPV control

2.5.1 Preliminaries

In this section, the notation used for the control design techniques contained within will be defined. The notation contained within is borrowed from [23, 24]. The set of real numbers

is denoted by $\mathbb{R}$, the set of real $n \times m$ matrices is denoted by $\mathbb{R}^{n\times m}$, and the set of non-negative integers is denoted by $\mathbb{N}_0$. The $n \times n$ identity matrix is denoted I_n, and the $n \times m$ zero matrix is denoted $0_{n\times m}$. The transpose of a matrix X is written X^T. The kernel of a linear map L is denoted $\mathrm{Ker}(L)$, and likewise the image is denoted $\mathrm{Im}(L)$. A matrix $X \succ 0$ is said to be positive definite, while $X \prec 0$ is a negative definite matrix. The normed space of square summable vector-valued sequences is denoted by ℓ_2. It consists of elements $x = (x(0), x(1), x(2), \dots)$, with each $x(k) \in \mathbb{R}^n$ for some n, having a finite ℓ_2-norm $\|x\|_{\ell_2}^2 = \sum_{k=0}^{\infty} x(k)^T x(k) < \infty$.

2.5.2 Problem Formulation

This problem formulation was primarily adapated from [25].

Let G be an LPV discrete-time system defined by

$$\begin{bmatrix} x(k+1) \\ z(k) \\ y(k) \end{bmatrix} = \begin{bmatrix} A(p(k)) & B_1(p(k)) & B_2(p(k)) \\ C_1(p(k)) & D_{11}(p(k)) & D_{12}(p(k)) \\ C_2(p(k)) & D_{21}(p(k)) & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ w(k) \\ u(k) \end{bmatrix} \qquad (2.21)$$
$$x(0) = 0$$

for $w \in \ell_2$, where $k \in \mathbb{N}_0$ denotes the time instant. The signals $w(k)$, $z(k)$, $u(k)$, and $y(k)$ denote the exogenous disturbances, errors, applied control, and measurements, respectively. The system matrices are assumed to be continuous functions of the parameter vector p. The parameters $p(k) = (p_1(k), p_2(k), \dots, p_r(k))$ and their respective increments $dp(k) = p(k+1) - p(k)$ are assumed to satisfy $(p(k), dp(k)) \in \Gamma, \forall\ k \in \mathbb{N}_0$, where Γ is defined as

$$\Gamma = \left\{ (\delta, d\delta) \in \mathbb{R}^r \,|\, \underline{\delta} \le \delta \le \overline{\delta} \text{ and } \underline{d\delta} \le d\delta \le \overline{d\delta} \right\} \qquad (2.22)$$

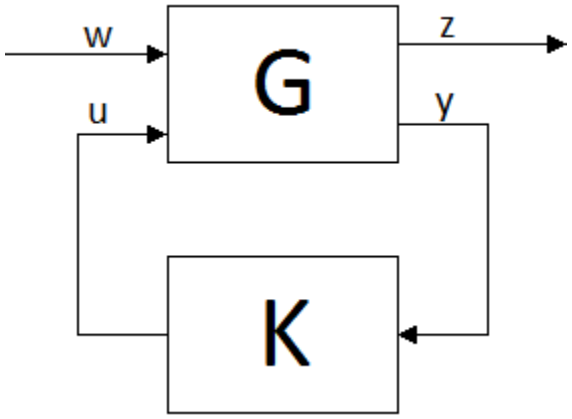


Figure 2.5: Closed-loop system L

where underline denotes the lower bound, overline denotes the upper bound, $\underline{d\delta} \leq 0$ and $\overline{d\delta} \geq 0$. Note that a hyper-rectangle is used in place of the more appropriate polytope (as used in [24, 25]) for simplicity. The allowable parameter-trajectories p can then be said to reside in the set

$$\Delta_\Gamma := \{p : \mathbb{N}_0 \to \mathbb{R}^r \mid (p(k), dp(k)) \in \Gamma, \ \forall \ k \in \mathbb{N}_0\} \tag{2.23}$$

Suppose that the plant G is controlled by a controller K whose state-space equation is

$$\begin{bmatrix} x_K(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} A_K(p(k), dp(k)) & B_K(p(k), dp(k)) \\ C_K(p(k), dp(k)) & D_K(p(k), dp(k)) \end{bmatrix} \begin{bmatrix} x_K(k) \\ y(k) \end{bmatrix} \tag{2.24}$$
$$x_K(0) = 0$$

where $x_K(k) \in \mathbb{R}^m$, $k \in \mathbb{N}_0$. The parameters $p(k)$ here are the same as those in the plant equations. Denoting the feedback interconnection of G and K, shown in Figure 2.5, as L, the closed-loop realization is expressed as

$$\begin{bmatrix} x_L(k+1) \\ z(k) \end{bmatrix} = \begin{bmatrix} A_L(p(k), dp(k)) & B_L(p(k), dp(k)) \\ C_L(p(k), dp(k)) & D_L(p(k), dp(k)) \end{bmatrix} \begin{bmatrix} x_L(k) \\ w(k) \end{bmatrix} \tag{2.25}$$

where $x_L(k) = [x(k)^T, x_K(k)^T]^T \in \mathbb{R}^{n+m}$, with the closed-loop state-space matrices defined in the obvious way.

The goal then becomes to find a γ-admissible controller K, defined as follows.

Definition 1. A controller K is a γ-admissible synthesis for the plant G if the closed-loop system in Fig. 2.5 is stable and the performance inequality $\|w \mapsto z\|_{\ell_2 \mapsto \ell_2} < \gamma$ is achieved for all $p \in \Delta_\Gamma$.

2.5.3 H_∞ Control Synthesis

In the case where the system matrices dependence on the parameters is dropped, the system becomes a standard LTI system. The control synthesis problem becomes the standard discrete-time H_∞ controller synthesis problem, which is given from [26] in the form of a semidefinite program (SDP) as follows.

Theorem 1 (Equations 6.5-6.7, [26])**.** *There exists a γ-admissible synthesis K to the plant G if and only if there exist symmetric matrices R, S satisfying the following linear matrix inequalities (LMIs):*

$$\left(\begin{array}{c|c} \mathcal{N}_R & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} ARA^T - R & ARC_1^T & B_1 \\ C_1RA^T & -\gamma I + C_1RC_1^T & D_{11} \\ \hline B_1^T & D_{11}^T & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_R & 0 \\ \hline 0 & I \end{array}\right) \prec 0 \tag{2.26}$$

$$\left(\begin{array}{c|c} \mathcal{N}_S & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A^TSA - S & A^TSB_1 & C_1^T \\ B_1^TSA & -\gamma I + B_1^TSB_1 & D_{11}^T \\ \hline C_1 & D_{11} & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_S & 0 \\ \hline 0 & I \end{array}\right) \prec 0 \tag{2.27}$$

$$\left(\begin{array}{cc} R & I \\ I & S \end{array}\right) \succeq 0 \tag{2.28}$$

where $\mathcal{N}_R$ and $\mathcal{N}_S$ denote bases of the null spaces of $\left(B_2^T, D_{12}^T\right)$ and (C_2, D_{21}), respectively.

2.5.4 LPV Synthesis

2.5.4.1 Parameter-Independent Lyapunov Approach

If it is possible to operate under the assumptions that:

- the state-space matrices depend affinely on the time-varying parameter p, and
- the measurements of p are available in real time at each discrete instant k, and
- the values of the parameter p reside in a polytope,

then an LPV controller K can be found, as detailed in [27]. The main results are included here for completeness.

First, we begin by defining the notion of a "polytopic" LPV system.

Definition 2 (*Polytopic LPV Systems*, Definition 2.1 [27])**.** An LPV system is called "polytopic" when it can be represented by state-space matrices $A(p)$, $B(p)$, $C(p)$, and $D(p)$, where

the parameter vector p ranges over a fixed polytope Θ of vertices $\omega_1, \omega_2, \ldots, \omega_q$; that is,

$$p \in \Theta := \mathrm{Co}\left\{\omega_1, \omega_2, \ldots, \omega_q\right\}. \tag{2.29}$$

where Co denotes the convex hull. Additionally, the dependence of $A(\cdot)$, $B(\cdot)$, $C(\cdot)$, and $D(\cdot)$ on p is affine.

Next, the notion of quadratic H_∞ performance is formally defined.

Definition 3 (*Quadratic H_∞ Performance*, Definition 3.2 [27])**.** A controller K in closed-loop system L has quadratic $\mathcal{H}_\infty$ performance γ if and only if there exists a single matrix $X \succ 0$ such that

$$\mathfrak{B}^{\sigma}_{[A(p),B(p),C(p),D(p)]}(X,\gamma) \prec 0 \tag{2.30}$$

for all admissible values of the parameter vector p.

$\mathfrak{B}^{\sigma}_{[A,B,C,D]}(X,\gamma)$ represents the so-called bounded real lemma (BRL) map, given by:

$$\mathfrak{B}^{\sigma}_{[A,B,C,D]}(X,\gamma) := \begin{pmatrix} -X^{-1} & A & B & 0 \\ A^T & -X & 0 & C^T \\ B^T & 0 & -\gamma I & D^T \\ 0 & C & D & -\gamma I \end{pmatrix} \tag{2.31}$$

It can be seen that 2.30 imposes an infinite number of constraints, due to the fact that it must hold for all values of $p \in \Theta$. It can be shown that 2.30 will hold for all values of $p \in \Theta$ if and only if it holds at the vertices $\omega_1, \omega_2, \ldots, \omega_q$. The following theorem from [27] formalizes that fact.

Theorem 2 (*Vertex Property*, Theorem 3.3 [27])**.** *Consider a polytopic LPV plant described*

by state-space equations

$$\begin{aligned} x(k+1) &= A(p(k))x(k) + B(p(k))u(k) \\ y(k) &= C(p(k))x(k) + D(p(k))u(k) \end{aligned} \tag{2.32}$$

with

$$\begin{pmatrix} A(p) & B(p) \\ C(p) & D(p) \end{pmatrix} \in \mathcal{P} := Co\left\{ \begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix}, i = 1, \ldots, q \right\} \tag{2.33}$$

where $A_i = A(w_i)$, $B_i = B(w_i)$, $C_i = C(w_i)$, *and* $D_i = D(w_i)$. *The following statements are equivalent:*

(i) this LPV system is stable with quadratic $\mathcal{H}_\infty$ *performance* γ*;*

(ii) there exists a single matrix $X \succ 0$ *such that, for all* $\begin{pmatrix} A(p) & B(p) \\ C(p) & D(p) \end{pmatrix} \in \mathcal{P}$,

$$\mathfrak{B}^{\sigma}_{[A(p),B(p),C(p),D(p)]}(X, \gamma) \prec 0 \tag{2.34}$$

(iii) there exists $X \succ 0$ *satisfying the system of LMIs*

$$\mathfrak{B}^{\sigma}_{[A_i,B_i,C_i,D_i]}(X, \gamma) \prec 0, i = 1, 2, \ldots, q. \tag{2.35}$$

Under the assumptions that:

A1. $D_{22}(p) = 0$, or equivalently, $D_{22i} = 0$ for $i = 1, 2, \ldots, q$;

A2. $B_2(p)$, $C_2(p)$, $D_{12}(p)$, and $D_{21}(p)$ are parameter independent, or equivalently

$$B_{2i} = B_2, C_{2i} = C_2, D_{12i} = D_{12}, D_{21i} = D_{21} \text{ for } i = 1, 2, \ldots, q; \tag{2.36}$$

A3. the pairs $(A(p), B_2)$ and $(A(p), C_2)$ are quadratically stabilizable and quadratically detectible over Θ respectively.

The control synthesis conditions are given in the following result.

Theorem 3 (*Convex Solvability Conditions*, Theorem 5.2 [27])**.** *Consider an LPV polytopic plant of the form 2.21, and assume (A1)-(A3). Let $\mathcal{N}_R$ and $\mathcal{N}_S$ denote bases of the null spaces of (B_2^T, D_{12}^T) and (C_2, D_{21}) respectively.*

There exists an LPV controller guaranteeing quadratic $\mathcal{H}_\infty$ performance γ along all parameter trajectories in the polytope

$$\Theta = \left\{ \sum_{i=1}^{q} \alpha_i \omega_i : \alpha_i \geq 0, \sum_{i=1}^{q} \alpha_i = 1 \right\}$$

if and only if there exist two symmetric matrices (R, S) in $\mathbb{R}^{n \times n}$ satisfying the system of $2q+1$ LMIs

$$\left(\begin{array}{c|c} \mathcal{N}_R & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i R A_i^T - R & A_i R C_{1i}^T & B_{1i} \\ C_{1i} R A_i^T & -\gamma I + C_{1i} R C_{1i}^T & D_{11i} \\ \hline B_{1i}^T & D_{11i}^T & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_R & 0 \\ \hline 0 & I \end{array}\right) \prec 0, \ i = 1, \ldots, q, \tag{2.37}$$

$$\left(\begin{array}{c|c} \mathcal{N}_S & 0 \\ \hline 0 & I \end{array}\right)^T \left(\begin{array}{cc|c} A_i^T S A_i - S & A_i^T S B_{1i} & C_{1i}^T \\ B_{1i}^T S A_i & -\gamma I + B_{1i}^T S B_{1i} & D_{11i}^T \\ \hline C_{1i} & D_{11i} & -\gamma I \end{array}\right) \left(\begin{array}{c|c} N_S & 0 \\ \hline 0 & I \end{array}\right) \prec 0, \ i = 1, \ldots, q, \tag{2.38}$$

$$\begin{pmatrix} R & I \\ I & S \end{pmatrix} \succeq 0 \tag{2.39}$$

Usually, we would like to find a $\gamma_{\min}$-admissible synthesis, where $\gamma_{\min}$ is the minimum achievable γ up to a certain tolerance. To do so, we solve the following convex optimization problem:

$$\begin{aligned} &\text{minimize } \gamma \\ &\text{subject to: 2.37 through 2.39} \end{aligned}$$

2.5.4.2 Parameter-Dependent Lyapunov Approach

We now consider the case where a parameter-dependent Lyapunov function is used to design to an LPV controller. Namely, the synthesis solutions R and S are parameter-dependent. We start by giving the following result adapted from [23, 25].

Theorem 4. *Given LPV plant G defined in 2.21 with $p \in \Delta_\Gamma$, suppose the matrices $[B_2^T, D_{12}^T]$ and $[C_2, D_{21}]$ have full row rank, and suppose assumption A2 holds. Then there exists a γ-admissible LPV synthesis K to G as defined in Definition 1 for some scalar γ if there exist matrix-valued functions $R(\delta) \succ 0$, $S(\delta) \succ 0$, continuous in δ, such that*

$$F^T(\delta)R(\delta)F(\delta) - V_1^T R(\delta + d\delta)V_1 + M(\delta)^T M(\delta) - \gamma^2 V_2^T V_2 \prec 0 \tag{2.40}$$

$$\begin{bmatrix} J(\delta)^T S(\delta + d\delta)J(\delta) - U_1^T S(\delta)U_1 - U_2^T U_2 & L(\delta)^T \\ L(\delta) & -\gamma^2 I \end{bmatrix} \prec 0 \tag{2.41}$$

$$\begin{bmatrix} R(\delta) & I \\ I & S(\delta) \end{bmatrix} \succeq 0 \tag{2.42}$$

for all $(\delta, d\delta) \in \Gamma$, as defined in Equation 2.22, where:

$$Im\left[V_1^T\ V_2^T\right]^T = Ker\left[B_2^T\ D_{12}^T\right], \qquad \left[V_1^T\ V_2^T\right]\left[V_1^T\ V_2^T\right]^T = I$$
$$Im\left[U_1^T\ U_2^T\right]^T = Ker\left[C_2^T\ D_{21}^T\right], \qquad \left[U_1^T\ U_2^T\right]\left[U_1^T\ U_2^T\right]^T = I$$
$$F(\delta) = A(\delta)^T V_1 + C_1(\delta)^T V_2, \quad M(\delta) = B_1(\delta)^T V_1 + D_{11}(\delta)^T V_2$$
$$J(\delta) = A(\delta)U_1 + B_1(\delta)U_2, \quad L(\delta) = C_1(\delta)U_1 + D_{11}(\delta)U_2$$

As B_2, D_{12}, C_2, and D_{21} are parameter independent, per assumption A2, the image and kernel are constant, and do not necessitate an application of Finsler's Lemma, as seen in [25]. It should also be noted that 2.40-2.42 impose an infinite number of LMI constraints, due to the fact that they must hold for all values of $(\delta, d\delta) \in \Gamma$. However, relaxation techniques exist to find solutions R and S that satisfy the LMIs above. Two of these techniques, among others, are the Sum of Squares (SOS) decomposition [28] and Multiconvexity relaxation [29]. To begin, note that the LMI constraint

$$X(\delta, d\delta) \in \mathbb{R}^{n\times n},\ X(\delta, d\delta) \prec 0,\ \forall\ (\delta, d\delta) \in \Gamma \tag{2.43}$$

can be equivalently stated as the inequality

$$f(\nu, X(\delta, d\delta)) = \nu^T X(\delta, d\delta)\nu < 0,\ \forall\ \nu \in \mathbb{R}^n/\{0\},\ (\delta, d\delta) \in \Gamma \tag{2.44}$$

Sum of Squares Relaxation A detailed analysis of the problem formulation and application of the SOS relaxation technique can be found in [28, 30, 31]. Due to the computational intensity involved with the SOS approach, it was found that model fidelity had to be sacrificed in order for the methods to be applied. It was reasoned that sacrificing model accuracy negates any benefit achieved by solving the parameter dependent solution. Thus, the method was abandoned in order to pursue other avenues.

Multiconvexity Relaxation The main development of the multiconvexity relaxation technique was given in [29] through use of the concept of directional convexity. The concept of multi-quasi-convexity, and the resulting concept of multiconvexity is included here for completeness.

Corollary 1 (Multi-quasi-convexity, Corollary 3.2 [29]). *Consider a polytope* Π *and the directions* $d_1, \ldots, d_q$*, determined by the edges of* Π*. Assume that for any* x *in* Π*, the function* f *is quasi-convex on the line segments* $L_{d_i}(x)$ *for* $i = 1, \ldots, q$*. Then,* f *has a maximum over* Π *at a vertex of* Π*.*

Corollary 2 (Multiconvexity, Corollary 3.4 [29]). *With the definitions in Corollary 1,* f *has a maximum over* Π *in* $\operatorname{vert}\Pi$ *whenver it holds that*

$$\left.\frac{\partial^2 f(x+\lambda d_i)}{\partial \lambda^2}\right|_{\lambda=0} \geq 0 \ \forall x \in \Pi, \ i = 1, \ldots, q. \tag{2.45}$$

Thus, if 2.44 is multiconvex along the directions of the edges of Δ_Γ, then the LMI constraint 2.43 holds for all $(p, dp) \in \Delta_\Gamma$ if and only if it holds for all $(p, dp) \in \operatorname{vert}\Delta_\Gamma$. The enforcement of multiconvexity comes through the inclusion of additional LMIs in the synthesis problem.

2.5.5 Controller Construction

The various solutions for R and S, whether they be parameter independent or parameter dependent, can be used to construct a controller K, using one of the methods shown in [24, 26, 27, 32]. Two of these approaches, one for standard LTI controllers, and one for LPV controllers, are included here for completeness.

For the standard LTI controller ([26], Section 7): Given solutions R and S to the control synthesis problem, compute the values:

$$M = (R - S^{-1})^{1/2}$$

$$X = \begin{bmatrix} S & -SM \\ -M^T S & I + M^T SM \end{bmatrix} \quad X^{-1} = \begin{bmatrix} R & M \\ M^T & I \end{bmatrix}$$

$$\tilde{C}_1 = \frac{1}{\gamma} C_1, \quad \tilde{D}_{11} = \frac{1}{\gamma} D_{11}, \quad \tilde{D}_{12} = \frac{1}{\gamma} D_{12}$$

$$H = \begin{bmatrix} -X^{-1} & \begin{matrix} A & 0 \\ 0 & 0 \end{matrix} & \begin{matrix} B_1 & 0 \\ 0 & 0 \end{matrix} \\ \begin{matrix} A^T & 0 \\ 0 & 0 \end{matrix} & -X & \begin{matrix} 0 & \tilde{C}_1^T \\ 0 & 0 \end{matrix} \\ \begin{matrix} B_1^T & 0 \\ 0 & 0 \end{matrix} & \begin{matrix} 0 & 0 \\ \tilde{C}_1 & 0 \end{matrix} & \begin{matrix} -I & \tilde{D}_{11}^T \\ \tilde{D}_{11} & -I \end{matrix} \end{bmatrix}$$

$$P = \begin{bmatrix} 0_n & I & 0 & 0 \\ B_2^T & 0 & 0 & \tilde{D}_{12}^T \end{bmatrix} \quad Q = \begin{bmatrix} 0 & 0 & I & 0 & 0 \\ 0 & C_2 & 0 & D_{21} & 0 \end{bmatrix}$$

then the controller can be found by solving the LMI:

$$H + P^T K Q + Q^T K^T P \prec 0 \tag{2.46}$$

where

$$K = \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} \tag{2.47}$$

This controller construction procedure works for both the standard H_∞ controller, and the LPV controller with parameter-independent Lyapunov function, which involves solving for an LTI controller at each vertex of the parameter hyper-rectangle. Then, as shown in [27], the LPV controller can be constructed as an interpolant of the vertex controllers

$$\begin{bmatrix} A_K(p(k)) & B_K(p(k)) \\ C_K(p(k)) & C_K(p(k)) \end{bmatrix} = \sum_{i=1}^{r} \alpha_i(k) \begin{bmatrix} A_{Ki} & B_{Ki} \\ C_{Ki} & D_{Ki} \end{bmatrix} \tag{2.48}$$

where α_i is as defined in Theorem 3, and A_{Ki}, B_{Ki}, C_{Ki}, and D_{Ki} are the controller matrices at each vertex w_i.

For the LPV controller with parameter-dependent Lyapunov function ([24], Section 4): Given solutions $R_k = R(p(k))$, $R_{k+1} = R(p(k) + dp(k))$ and $S_k = S(p(k))$, $S_{k+1} = S(p(k) + dp(k))$ to the control synthesis problem, we re-define the values M, X, and X^{-1} to reflect their new dependence on the time instant k:

$$M_k = (R_k - S_k^{-1})^{1/2}$$

$$X_k = \begin{bmatrix} S_k & -S_k M_k \\ -M_k^T S_k & I + M_k^T S_k M_k \end{bmatrix} \quad X_k^{-1} = \begin{bmatrix} R_k & M_k \\ M_k^T & I \end{bmatrix}$$

Next, we make the following definitions:

$$\overline{C} = \begin{bmatrix} C_1 & 0 \end{bmatrix}, \quad \underline{D_{12}} = \begin{bmatrix} 0 & D_{12} \end{bmatrix}$$
$$\overline{A} = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}, \quad \overline{B} = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \quad \underline{B} = \begin{bmatrix} 0 & B_2 \\ I & 0 \end{bmatrix}$$
$$\underline{C} = \begin{bmatrix} 0 & I \\ C_2 & 0 \end{bmatrix}, \quad \underline{D_{21}} = \begin{bmatrix} 0 \\ D_{21} \end{bmatrix}$$
$$V = \begin{bmatrix} \underline{C} X_k^{-1/2} & \frac{1}{\sqrt{\gamma}} \underline{D_{21}} \end{bmatrix}$$
$$W = \begin{bmatrix} X_{k+1}^{1/2} \overline{A} X_k^{-1/2} & \frac{1}{\sqrt{\gamma}} X_{k+1}^{1/2} \\ \frac{1}{\sqrt{\gamma}} \overline{C} X_k^{-1/2} & \frac{1}{\gamma} D_{11} \end{bmatrix}, \quad U = \begin{bmatrix} X_{k+1}^{1/2} \underline{B} \\ \frac{1}{\sqrt{\gamma}} \underline{D_{12}} \end{bmatrix}$$

If $U_\perp$ and $V_\perp$ are chosen such that $[U\ U_\perp]$ and $\begin{bmatrix} V \\ V_\perp \end{bmatrix}$ are invertible, as well as $U^T U_\perp = 0$ and $V V_\perp^T = 0$, then one choice for the controller is:

$$L = (U^T U)^{-1/2}, \quad E = (V V^T)^{-1/2}, \quad \tilde{U} = UL, \quad \tilde{V} = EV,$$
$$\hat{U} = (U_\perp^T U_\perp)^{-1/2} U_\perp^T, \quad \hat{V} = W V_\perp^T (V_\perp V_\perp^T)^{-1/2},$$
$$T_1 = \hat{U}\hat{V}, \quad T_2 = \hat{U}^T \hat{V}, \quad T_3 = \hat{U} W \hat{V}^T,$$
$$Q' = -T_2 (I - T_1^T T_1)^{-1/2} T_1^T (I - T_1 T_1^T)^{-1/2} T_3$$
$$\begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} = L(Q' - \hat{U}^T W \hat{V}^T) E$$

Chapter 3

Parameter Estimation and Modeling

As the controllers presented within are model based, it is necessary to develop beforehand a mathematical model of the system. In the case of UAVs, this is often a difficult challenge because the aerodynamic forces and moments depend heavily on the configuration of the vehicle being studied. The process of developing a high accuracy aerodynamic model has the potential to be both extremely time consuming and costly, if using traditional methods like wind tunnel testing.

Using the process of system identification, it is possible to identify a mathematical model of the vehicle using input-output data relationships. As the aerodynamic characteristics and dependencies of these types of vehicles are well studied, we can generally begin with an assumed model structure, and change the problem from system identification to one of parameter estimation.

There are a vast number of parameter estimation techniques, each with their benefits and pitfalls. The main division in practical parameter estimation techniques comes at the inclusion of process noise. Deterministic methods operate under the assumption of a lack

of process noise, while stochastic methods allow for the inclusion of these types of disturbances. For this reason, deterministic methods tend to be simple in their application, while stochastic methods introduce a further level of complexity. The most common example of a deterministic method is ordinary least squares (OLS). For stochastic systems, one of the most commonly applied techniques is the filter-error method.

For implementation on real UAV systems, modeling inaccuracies are always present. As the eventual goal is to implement the controllers contained within on a real UAV system, it is important to reflect these modeling inaccuracies in as accurate a fashion as possible. In order to introduce modeling inaccuracies into the control design process, traditional parameter estimation techniques, namely the OLS and output-error (OE) methods, were applied to simulated flight test data from the DATCOM model, which is assumed to represent the actual UAV system. The controllers are then synthesized using this estimated mathematical model of the UAV, and applied to the DATCOM model in simulation. In this way, it is possible to test the controller resilience to modeling inaccuracies, and provide confidence for eventual implementation on a real UAV system.

3.1 Common Modeling Methods

3.1.1 Ordinary Least Squares

Ordinary least squares parameter estimation falls under the umbrella of regression methods. The measurement equations for the system in consideration are modeled as a linear system of the form

$$z = X\theta + v \tag{3.1}$$

where

$$z = [z(1), z(2), \ldots, z(N)]^T = N \times 1 \text{ vector of measurements}$$

$$X = [1, \zeta_1, \ldots, \zeta_n] = N \times n_p \text{ matrix of regressors}$$

$$\theta = [\theta_0, \theta_1, \ldots, \theta_n]^T = n_p \times 1 \text{ vector of unknown parameters}$$

$$v = [v(1), v(2), \ldots, v(N)]^T = N \times 1 \text{ vector of measurement errors}$$

The measurement errors v are assumed to be zero mean and uncorrelated, with constant variance. The OLS regression solution seeks the best estimator θ that minimizes the sum of squared differences between the measurements and the model outputs,

$$J(\theta) = \frac{1}{2}(z - X\theta)^T(z - X\theta) \tag{3.2}$$

which, for linearly independent ζ (regressors), has the analytical solution

$$\hat{\theta} = (X^T X)^{-1} X^T z \tag{3.3}$$

The goodness of fit of the proposed model can be judged by the coefficient of determination, R^2, given by

$$R^2 = \frac{\hat{\theta} X^T z - N\bar{z}^2}{z^T z - N\bar{z}^2} \tag{3.4}$$

where $\bar{z}$ represents the mean value of z. The coefficient of determination varies in the range $[0, 1]$, where 1 represents a perfect fit to the measured data.

For aircraft, this method is most commonly applied to the aerodynamic coefficients. Using flight test data, the aerodynamic force and moment coefficients are estimated. Then, a model structure for each aerodynamic coefficient is assumed, and the OLS regression is performed. For three force and three moment coefficients, a total of six regressions are applied in order

to estimate all parameters.

3.1.2 Filter-Error

Filter-error methods combine a maximum likelihood estimator with an optimal filter. The filter is necessary due to the inclusion of process noise in the problem formulation. Assuming a measurement equation of the form

$$z = h(\theta) + v \tag{3.5}$$

the filter-error method aims to minimize the weighted least-squares difference between the measured outputs z and the model predicted outputs $\hat{y}$. The cost function used in filter-error estimation methods is the negative log of the likelihood function

$$\hat{\theta} = \max_{\theta} \sum_{i=1}^{N} - \ln \left\{ \mathbb{L} \left[z(i) | Z_{i-1}; \theta \right] \right\} \tag{3.6}$$

where $\mathbb{L}$ is the likelihood function, and Z_{i-1} is the series of measurements up to $z(i-1)$. Operating under the assumption that measurement and process noises are both independent and normally distributed, then the measurements $z(i)$ will also have these properties. Additionally, as sampling rate increases, the probability distribution function of the difference between the measurements $z(i)$ and estimates $\hat{y}$ (called the innovations) approaches a Gaussian distribution. After some manipulation [33, Equations 6.8 and 6.9], it is possible to express the negative log-likelihood function in the form:

$$- \ln[\mathbb{L}(Z_N; \theta)] = \frac{1}{2} \sum_{i=1}^{N} [v^T(i) \mathcal{B}^{-1}(i) v(i) + \ln |\mathcal{B}(i)|] \tag{3.7}$$

where $\mathcal{B}_i$ is the covariance of the innovations $v(i)$. This reduces the search of the negative log-likelihood function to finding the means and covariances of the innovations $v(i)$. These quantities are estimated using a Kalman filter, which recursively produces minimum variance estimates of the states and outputs. See [33, Equations 6.11] for formulaic implementation details. In order to minimize the cost function, and owing to the nonlinear form of the minimization problem, a Gauss-Newton iteration method is employed.

3.1.2.1 Simplifications

A useful simplification to the filter-error method can be obtained by neglecting the effects of process noise. In this special case, called the Output-Error method, the Kalman filter estimation is replaced by a direct integration of the state equations. In the course of this work, the filter-error method was found to be sensitive to the initial parameter estimates and Kalman filter tuning, and was not reliable in its resulting parameter estimates. For this reason, the output-error method was favored. While neglecting process noise in the problem formulation is often a non-realistic assumption, it is one that can often be met practically through careful choice of flight testing environments. For example, flying early in the mornings when winds are low will produce flight conditions with little to no process noise present. In addition, pre-filtering of flight test data can help mitigate the effects of process noise [34].

3.2 Application of Parameter Estimation

To begin the estimation procedure, the structure of the aerodynamic model must be chosen. This is an iterative process, where terms are added and removed with the aim of improving the fidelity of the model. After an iterative process, using candidate terms based on knowl-

edge of aerodynamic relationships, the following aerodynamic model structure was found to produce the best results,

$$
\begin{aligned}
C_D &= C_{D_0} + \frac{C_L^2}{\pi e AR} \\
C_L &= C_{L_0} + C_{L_\alpha}\alpha + C_{L_{\hat{q}}}\frac{q\bar{c}}{2V} \\
C_Y &= C_{Y_0} + C_{Y_\beta}\beta + C_{Y_{\hat{p}}}\frac{pb}{2V} + C_{Y_{\delta_r}}\delta_r \\
C_l &= C_{l_0} + C_{l_\beta}\beta + C_{l_{\hat{p}}}\frac{pb}{2V} + C_{l_{\hat{r}}}\frac{rb}{2V} + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r \\
C_m &= C_{m_0} + C_{m_\alpha}\alpha + C_{m_{\hat{q}}}\frac{q\bar{c}}{2V} + C_{m_{\delta_e}}\delta_e \\
C_n &= C_{n_0} + C_{n_\beta}\beta + C_{n_{\hat{p}}}\frac{pb}{2V} + C_{n_{\hat{r}}}\frac{rb}{2V} + C_{n_{\delta_r}}\delta_r
\end{aligned}
\tag{3.8}
$$

where the parameters to be estimated are in red.

3.2.1 Estimation Procedure

In order to estimate the 24 parameters contained within the aerodynamic force and moment coefficients, simulated flight testing was performed. A straight and level trim flight condition for the nonlinear DATCOM model was identified. The nonlinear simulation was set to an initial condition within approximately $\pm 15\%$ of all trim states. This departure from exact trim condition was included in order to reflect the difficulty of attaining exact trim conditions during system identification flight tests on a real system.

From this non-trim initial condition, various longitudinal and lateral-directional system identification inputs were applied to the system. These inputs included doublets, 3-2-1-1 inputs, and sweeps. In depth discussion on input design for parameter estimation can be found in [35]. Identification inputs varied in both control magnitude and signal duration. Examples

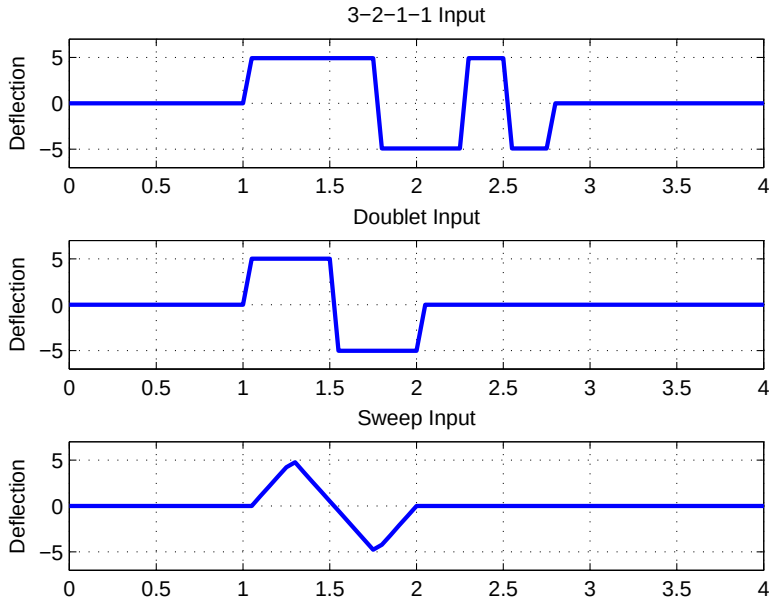


Figure 3.1: Various input signals for system identification

of the various input types is shown in Figure 3.1.

In addition to the non-trim initial condition, the simulation also includes light magnitude wind gusts, and measurement noise. The details of the implementation of these disturbances are in Section 5.1. The simulation is run for 10 seconds per input, and the noisy measurements are recorded. A sample measurement of longitudinal quantities from a 3-2-1-1 flight test maneuver is shown in Figure 3.2.

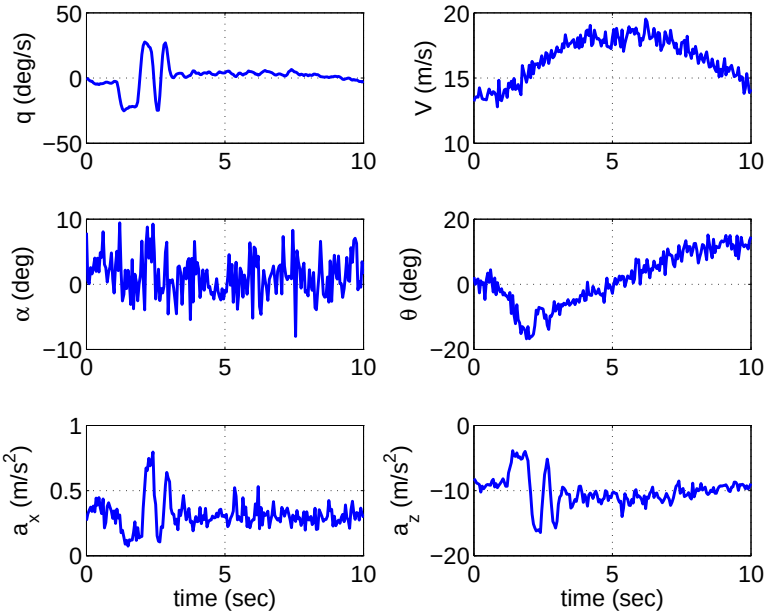


Figure 3.2: Measured longitudinal quantities during a 3-2-1-1 maneuver

Once the data has been collected, a flight path reconstruction process is implemented on the data, using a two-pass filter. The outputs of this flight path reconstruction process are smoothed estimates of the velocity components $u/v/w$ and Euler angles $\phi/\theta/\psi$. This has been shown in practice to produce improved parameter estimates [36, 37, 34].

First, the data is run through the forward pass portion of the two-pass smoother, an Extended Kalman Filter. Measured angular rates $p/q/r$ and accelerations $a_x/a_y/a_z$ are used as inputs to the system. This produces the state and covariance estimates. Next, the data is run through the backward pass portion, a Rauch-Tung-Striebel (RTS) smoother, which works backwards in time. This second pass computes smoothed estimates of the quantities from the forward pass. Figure 3.3 shows the good agreement between the resulting smoothed quantities and the exact values from the simulation for the same sample 3-2-1-1 maneuver in Figure 3.2.

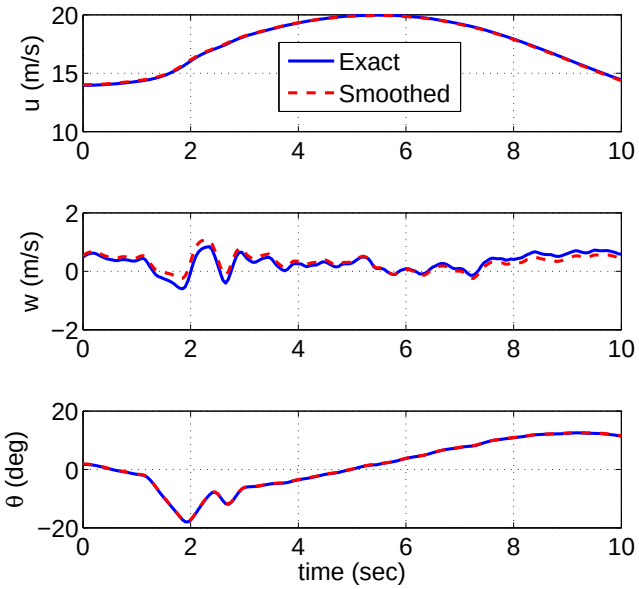


Figure 3.3: Smoothed longitudinal quantities for a 3-2-1-1 maneuver

Using this smoothed flight test data, the Maximum Likelihood Output-Error method discussed in Section 3.1.2.1 was implemented. The aircraft dynamics were split into two models, a longitudinal model and a lateral-directional model. For the longitudinal runs, the lateral directional quantities p, r, v, and ϕ were used as pseudo-inputs to the system in order to improve the performance of the estimation. A similar procedure was used in the lateral-directional runs using the longitudinal quantities as pseudo-inputs.

Owing to its ease of use and flexibility in model definition, the MATLAB estimation codes written by Jategaonkar [14] were used in the estimation procedure. Initial estimates of the parameters were identified using the OLS method. A collection of longitudinal flight tests, consisting of elevator doublets, sweeps, and 3-2-1-1 inputs was concatenated. The output-error method, with use of the Gauss-Newton iteration algorithm, was applied to this data series. All lateral directional parameters were fixed, and longitudinal parameters were

estimated.

Next, a collection of lateral-directional flight tests, consisting of rudder and aileron doublet sequences, was concatenated for use in the estimation of the lateral-directional parameters. The longitudinal parameters were set to the values estimated by the previous longitudinal estimation procedure, and the lateral-directional quantities were set to the values predicted by ordinary least squares. After each set of parameters had been estimated, the estimation procedure above was repeated, replacing the OLS parameter estimate initializations with their corresponding OE parameter estimate from the prior estimation run.

3.2.2 Estimation Results

Table 3.1 gives the resulting parameters and their estimated standard deviations, and the progression of the parameter estimates with 2σ error bars for longitudinal and lateral-directional parameter estimation is shown in Figures 3.4 and 3.5.

Table 3.1: Parameter estimates and standard deviations

Parameter	Estimate	Std. Dev.	Relative Std. Dev (%)
C_{D_0}	0.030	0.000	0.871
e	0.653	0.022	3.405
C_{L_0}	0.223	0.002	1.051
C_{L_α}	3.452	0.061	1.775
$C_{L_{\hat{q}}}$	11.012	0.465	4.226
C_{Y_0}	0.000	0.000	34.319
C_{Y_β}	-0.179	0.003	1.510
$C_{Y_{\hat{p}}}$	-0.029	0.010	35.210
$C_{Y_{\hat{r}}}$	0.154	0.004	2.846
$C_{Y_{\delta_r}}$	0.000	0.000	12.278
C_{l_0}	-0.102	0.001	1.444
C_{l_β}	-0.421	0.006	1.436
$C_{l_{\hat{p}}}$	0.107	0.001	1.340
$C_{l_{\hat{r}}}$	-0.194	0.003	1.526
$C_{l_{\delta_a}}$	0.023	0.001	6.265
$C_{l_{\delta_r}}$	-0.045	0.001	2.606
C_{m_0}	-0.756	0.017	2.297
C_{m_α}	-16.763	0.393	2.347
$C_{n_{\hat{q}}}$	-0.976	0.022	2.205
C_{n_0}	-0.000	0.000	13.256
C_{n_β}	0.030	0.000	0.522
$C_{n_{\hat{p}}}$	-0.058	0.001	1.086
$C_{n_{\hat{r}}}$	-0.056	0.000	0.688
$C_{n_{\delta_r}}$	-0.125	0.000	0.308

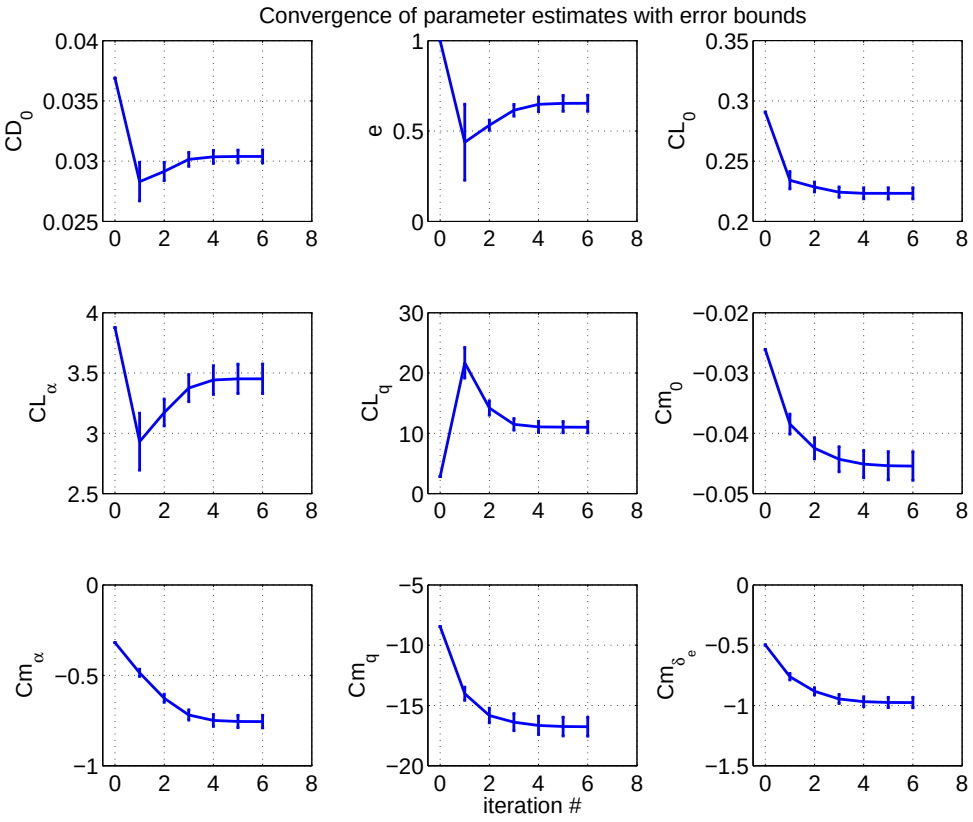


Figure 3.4: Longitudinal parameter estimate iterations

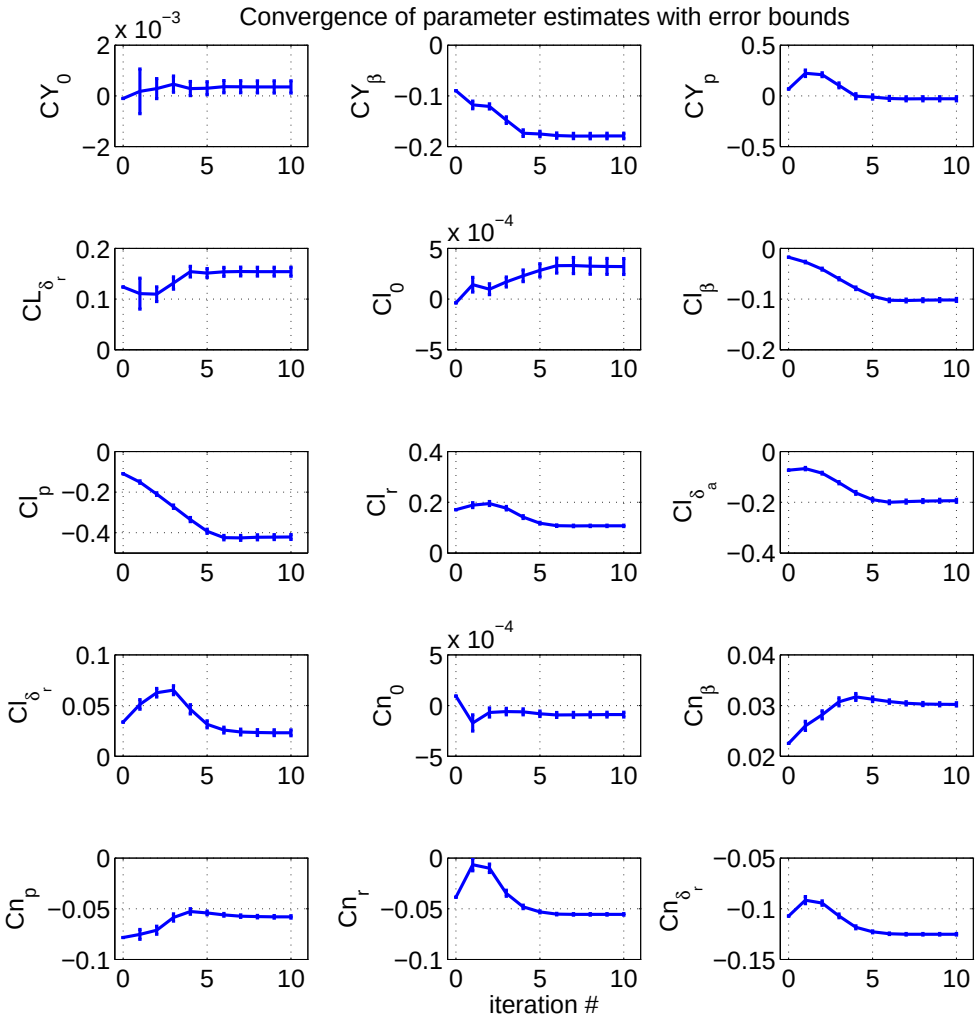


Figure 3.5: Lateral-directional parameter estimate iterations

In order to validate the resulting aerodynamic model, the parameters were inserted into a nonlinear simulation, and an identification flight test maneuver not used in modeling was applied to the system. The resulting outputs of the estimated model were compared to the outputs of the DATCOM simulation, shown in Figures 3.6 through 3.9. The model

prediction shows good agreement with the actual simulation, and so the model is deemed to be a reasonably accurate representation of the system for the purposes of control design.

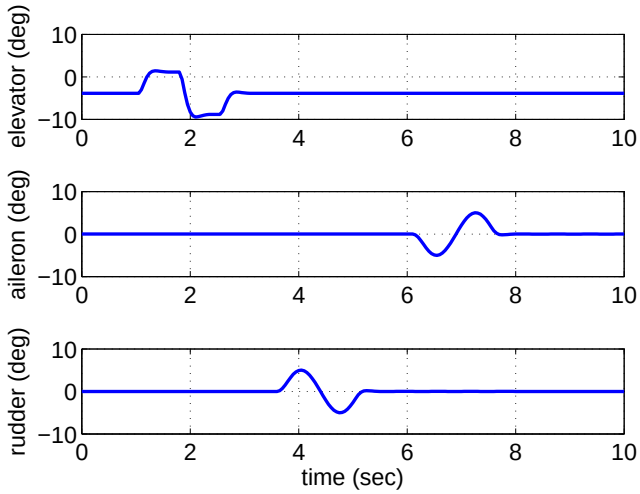


Figure 3.6: Control input for model validation maneuver

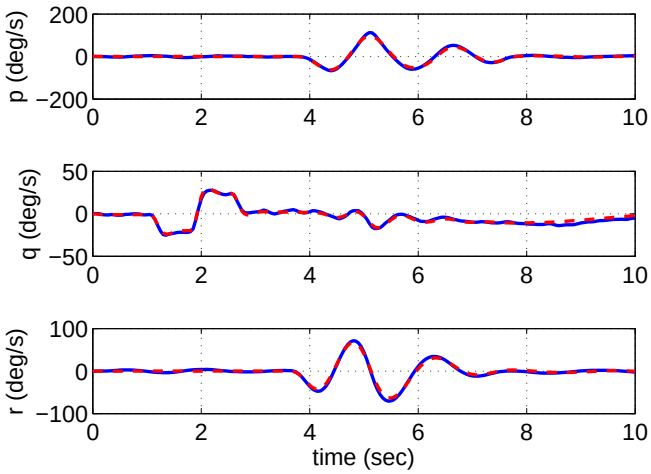


Figure 3.7: Angular rates for model validation maneuver

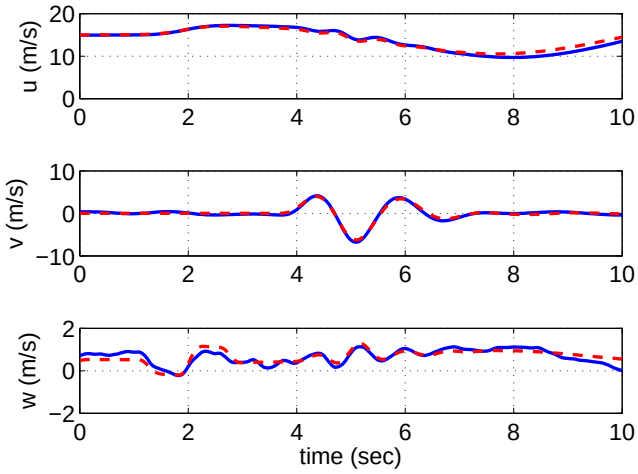


Figure 3.8: Velocity components for model validation maneuver

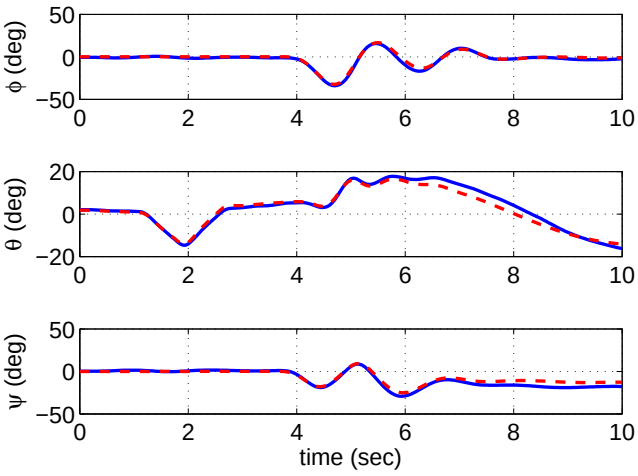


Figure 3.9: Euler angles for model validation maneuver

3.3 Actuator and Throttle Models

The control surface actuator dynamics relating the input commands to the control surface deflections are modeled by second order transfer functions, with natural frequency $\omega_n = 13.7$ rad/s, and damping ratio $\zeta = 0.67$.

$$G_{\delta_{(\cdot)}}(s) = \frac{187.69}{s^2 + 18.358s + 187.69} \tag{3.9}$$

The throttle dynamics relating commanded throttle to thrust are modeled by a fourth order transfer function of the form

$$G_{\delta_T}(s) = \frac{625}{s^4 + 20s^3 + 150s^2 + 500s + 625} \tag{3.10}$$

For reference, a step response of each system is given in Figure 3.10.

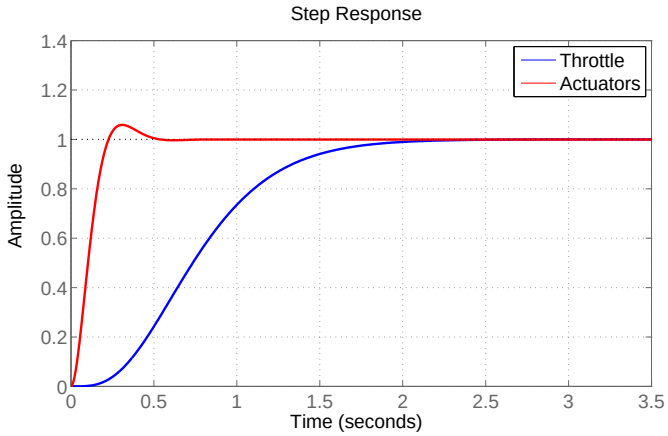


Figure 3.10: Step response of control surface actuator model and throttle model

Chapter 4

Controller Synthesis

The control systems discussed in this chapter are synthesized based on a linear plant model, which is obtained by linearizing the nonlinear equations of motion derived in Chapter 2 about a trim condition. For this work, the set of permissible geometric paths to be followed was restricted to 2D paths in the x/y plane. For this special case of paths, many simplifications can be made to the relevant system dynamics. The largest simplification is that the initial parallel transport frame for the virtual vehicle can be chosen such that the elevation and bank angles of the frame are zero and constant for all time. It then follows that the UAV elevation and bank angles can be considered identically equal to the previously defined error angles θ_{err} and ϕ_{err} (see Equation 2.16). Additionally, the $k_2(\ell)$ path parameter will be identically zero for all time, thus leaving $k_1(\ell)$ as the only relevant path parameter for the system. Taking these simplifications into consideration, the dynamics of the UAV(2.2, 2.3, 2.5, and 2.7) and the path following dynamics (2.13 and 2.16) can be combined into a 12 state system, given in Equation 4.1.

$$
\begin{aligned}
\dot{p} &= \frac{1}{\Gamma}\left(I_{xz}\left[I_x - I_y + I_z\right]pq - \left[I_z\left(I_z - I_y\right) + I_{xz}^2\right]qr + I_z\ell + I_{xz}n\right) \\
\dot{q} &= \frac{1}{I_y}\left[\left(I_z - I_x\right)pr - I_{xz}\left(p^2 - r^2\right) + m\right] \\
\dot{r} &= \frac{1}{\Gamma}\left(\left[\left(I_x - I_y\right)I_x + I_{xz}^2\right]pq - I_{xz}\left[I_x - I_y + I_z\right]qr + I_{xz}\ell + I_x n\right) \\
\dot{u} &= rv - qw - g\sin\theta + F_x\frac{g}{\|\boldsymbol{W}\|} \\
\dot{v} &= -ru + pw + g\sin\phi\cos\theta + F_y\frac{g}{\|\boldsymbol{W}\|} \\
\dot{w} &= qu - pv + g\cos\phi\cos\theta + F_z\frac{g}{\|\boldsymbol{W}\|} \\
\dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi}_{err} &= -(K_1 d_x + V\cos\theta\cos\psi_{err})k_1(\ell) + q\sin\phi\sec\theta + r\cos\phi\sec\theta \\
\dot{d}_x &= -(K_1 d_x + V\cos\theta\cos\psi_{err})(1 - k_1(\ell)d_y) + V\cos\theta\cos\psi_{err} \\
\dot{d}_y &= -(K_1 d_x + V\cos\theta\cos\psi_{err})k_1(\ell)d_x + V\cos\theta\sin\psi_{err} \\
\dot{d}_z &= -V\sin\theta
\end{aligned}
\tag{4.1}
$$

where $\Gamma = I_x I_z - I_{xz}^2$.

A steady straight-and-level flight trim condition for the nonlinear UAV model was found by minimizing a cost function penalizing the UAV dynamic equations (Equations 2.2, 2.3, 2.5, and 2.7) using the MATLAB function `fmincon`. The cost function is chosen as follows.

$$
\text{COST} = \dot{p}^2 + \dot{q}^2 + \dot{r}^2 + \dot{u}^2 + \dot{v}^2 + \dot{w}^2 + \dot{\phi}^2 + \dot{\theta}^2 + \dot{\psi}^2 + \dot{y_I}^2 + \dot{z_I}^2 + (\alpha - \theta)^2 + (15 - V)^2 \tag{4.2}
$$

In the minimization of the cost function, the values of $p/q/r/\phi/\psi/y_I/z_I$ are fixed at zero. The

values being chosen in the optimization procedure by `fmincon` are the inputs $\delta_e/\delta_a/\delta_r/\delta_T$, aerodynamic angles α/β, and elevation angle θ. The resulting minimum value of the cost function is zero. This results in the rates $p/q/r/u/v/w$ being constant, the flight path angle being zero, and the trim velocity being 15 m/s. This procedure results in the trim condition given in Table 4.1.

Table 4.1: Trim flight condition

$p/q/r$	0 deg/s
V	15 m/s
α/θ	2.14 deg
β/ϕ	0 deg
δ_a/δ_r	0 deg
δ_e	-4.14 deg
δ_T	0.041

The lumped UAV/path following state, input, and output vectors are defined as
$x^{\text{lump}} = [p,\, q,\, r,\, u,\, v,\, w,\, \phi,\, \theta,\, \psi_{err},\, d_x,\, d_x,\, d_z]^T$, $u = [\delta_e,\, \delta_a,\, \delta_r,\, \delta_T]^T$, and
$y^{\text{lump}} = [p,\, q,\, r,\, V_a,\, \alpha,\, \beta,\, \phi,\, \theta,\, \psi_{err},\, d_x,\, d_x,\, d_z]^T$.

The state equation and output equation, respectively, can thus be defined as follows.

$$\dot{\boldsymbol{x}}^{\text{lump}} = \boldsymbol{f}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u}) \tag{4.3}$$

$$\boldsymbol{y}^{\text{lump}} = \boldsymbol{h}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u}) \tag{4.4}$$

where $\boldsymbol{f}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u})$ is defined as given in Equation 4.1, and $\boldsymbol{h}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u})$ is defined in the obvious way using the definitions of V, α, and β given in Chapter 2.

Next, the nonlinear equations $\boldsymbol{f}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u})$ and $\boldsymbol{h}(\boldsymbol{x}^{\text{lump}}, \boldsymbol{u})$ are linearized about this trim condition, and the zero error state, using the Jacobians:

$$\begin{aligned} A^{\text{lump}} &= \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big|_{(\boldsymbol{x}_{\text{trim}}^{\text{lump}}, \boldsymbol{u}_{\text{trim}})}, \quad B^{\text{lump}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\Big|_{(\boldsymbol{x}_{\text{trim}}^{\text{lump}}, \boldsymbol{u}_{\text{trim}})} \\ C^{\text{lump}} &= \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\Big|_{(\boldsymbol{x}_{\text{trim}}^{\text{lump}}, \boldsymbol{u}_{\text{trim}})}, \quad D^{\text{lump}} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{u}}\Big|_{(\boldsymbol{x}_{\text{trim}}^{\text{lump}}, \boldsymbol{u}_{\text{trim}})} \end{aligned} \tag{4.5}$$

Care is taken to leave the state-space matrix A as a function of the path parameter $k_1(\ell)$. For our specific application, no other system matrices besides the A matrix are parameter-dependent. Noting that the dependence on $k_1(\ell)$ is affine, A can be expressed as follows.

$$A^{\text{lump}}(k_1(\ell)) = A_0 + k_1(\ell)A_1 \tag{4.6}$$

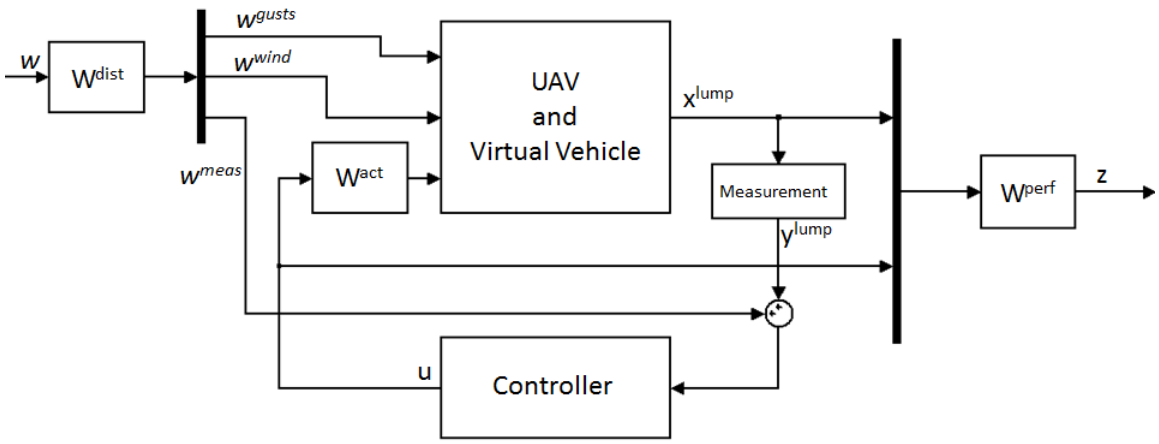
As the physical implementation of the system and control design will be in discrete-time, a simple Euler discretization is applied to the continuous-time state-space matrices:

$$A_d^{\text{lump}} \approx (I_n + TA^{\text{lump}}) \qquad B_d^{\text{lump}} \approx TB^{\text{lump}} \tag{4.7}$$

where T is the sampling time. In this work, the sampling time is T = 0.05 seconds. In all future notation, the subscript d is dropped, and all system matrices are assumed to be discrete time matrices.

After discretizing the state-space matrices, the various actuator and throttle models, disturbance weightings, and penalty weights were applied using the Robust Control Toolbox function `sysic`[38]. This approach was chosen because it allows for rapid changes in model formulations and penalty weights without slowing the iterative design process down prohibitively. A schematic representing the general interconnect formulation for control design purposes is shown in Figure 4.1.

Figure 4.1: System interconnection



The disturbance channel can be defined as $w = [(w^g)^T, (w^w)^T, (w^m)^T]^T$, representing gust, steady wind, and measurement noise disturbance signals, respectively; this signal is weighted by W_{dist}, a block diagonal matrix composed of gust weighting, steady wind weighting, and measurement noise weighting. Specifically, $W_{\text{dist}} = \text{diag}(3I_3, 5I_3, 3\sigma_{\text{meas}})$, where σ_{meas} is the diagonal matrix of measurement standard deviations, as defined in Section 5.1.3. The control input commands, $u(k) = [\delta_{e_{\text{cmd}}}, \delta_{a_{\text{cmd}}}, \delta_{r_{\text{cmd}}}, \delta_{T_{\text{cmd}}}]^T$, are subject to the dynamics of the transfer functions described in the previous section, namely $W_{\text{act}} = \text{diag}(G_{\delta_e}, G_{\delta_a}, G_{\delta_r}, G_{\delta_T})$. It is important to note that including actuator dynamics introduces additional states, which increases the dimension of the resulting controller. Including dynamic penalty weights also has the same effect. The performance weightings can be split into two categories, penalties on the measurements $y(k)$ denoted W_y, and penalties on the input commands $u(k)$ denoted W_u. These weightings are problem dependent, and are left to be described in the following sections.

The state vector is redefined as $x = [x^{\text{lump}}, x^{\text{act}}, x^{W_u}, x^{W_y}]^T$, where x^{act} are the 10 states introduced by the actuator dynamics, and x^{W_u} and x^{W_y} are states that may be added by

dynamic penalty weights chosen in each problem formulation in the following sections. In general, it is possible to express these actuator dynamics and possible dynamic input and output penalty weights in the state space form:

$$\begin{aligned} x^{(\cdot)}(k+1) &= \boldsymbol{A}^{(\cdot)}x^{(\cdot)}(k) + \boldsymbol{B}^{(\cdot)}u(k) \\ y^{(\cdot)}(k) &= \boldsymbol{C}^{(\cdot)}x^{(\cdot)}(k) + \boldsymbol{D}^{(\cdot)}u(k) \end{aligned} \tag{4.8}$$

where the superscript $(\cdot)$ will be W_u, W_y, or "act" in order to denote the system being represented.

The components of the system equations as represented in Equation 2.21 can thus be written as follows.

$$\begin{aligned} x(k+1) = &\begin{bmatrix} A^{\text{lump}}(k_1(\ell)) & B_2^{\text{lump}}C^{\text{act}} & 0 & 0 \\ 0 & A^{\text{act}} & 0 & 0 \\ 0 & 0 & A^{W_u} & 0 \\ B^{W_y}C_2^{\text{lump}} & 0 & 0 & A^{W_y} \end{bmatrix} x(k) \\ &+ \begin{bmatrix} B_1^{\text{lump}} \\ 0 \\ 0 \\ 0 \end{bmatrix} W^{\text{dist}}w(k) + \begin{bmatrix} B_2^{\text{lump}}D^{\text{act}} \\ B^{\text{act}} \\ B^{W_u} \\ 0 \end{bmatrix} u(k) \\ z(k) = &\begin{bmatrix} D^{W_y}C_2^{\text{lump}} & 0 & 0 & C^{W_y} \\ 0 & 0 & C^{W_u} & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0 \end{bmatrix} W^{\text{dist}}w(k) + \begin{bmatrix} 0 \\ D^{W_u} \end{bmatrix} u(k) \\ y(k) = &\begin{bmatrix} C_2^{\text{lump}} & 0 & 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 & 0 & D_{21}^{\text{lump}} \end{bmatrix} W^{\text{dist}}w(k) \end{aligned} \tag{4.9}$$

In order to find a γ_{min}-admissible controller, where γ_{min} is the minimum achievable γ to a certain tolerance, as discussed in Section 2.5, the following semi-definite program (SDP)

must be solved:

$$\text{minimize: } \gamma$$

$$\text{subject to: synthesis conditions}$$

where the synthesis conditions vary depending on the type of controller sought. In order to solve this SDP in MATLAB, the YALMIP toolbox was used, with SDPT3 as the chosen solver [39, 40]. All computations were carried out on a Dell Precision T3500 Desktop running 64-bit Windows 7, with an Intel Xeon W3550 Quad Core processor and 6 GB of RAM.

4.1 Standard H_∞ Controller

For the standard H_∞ controller, the parameter $k_1(\ell)$ is taken to be a constant value of zero, and thus the plant becomes an LTI system. The commanded inputs were weighted by high-pass filters, in an effort to penalize high-frequency control input commands. The chosen filters were second-order Butterworth filters with a cutoff frequency of 1 Hz, and high frequency gain of 0.3183 for elevator and aileron, and 0.6366 for rudder. The penalty weight for thrust command was a constant value of 50. These penalty weights can again be easily expressed in the state-space representation shown in Equation 4.8.

For output weighting, the measurements V, α, β, θ, ψ_{err}, d_y, and d_z were penalized. V was penalized by a constant value of 4.5. α, β, and ψ_{err} were penalized by second-order lowpass Butterworth filters, with a cutoff frequency of 2 Hz, and a low frequency gain of 1.9099, 4.7746, and 0.9549, respectively. θ was penalized by a constant value of 0.9549. d_x and d_y were each penalized by a constant value of 1/12. These weightings can be expressed in a state-space form in the same fashion as the weightings on commanded inputs, and are

collected in the system W_y.

To design the H_∞ controller, the results from Section 2.5.3 are applied. Specifically, the following semi-definite programming problem is solved:

$$\text{minimize } \gamma$$

$$\text{subject to: } 2.26 - 2.28$$

This SDP was solved in 15.72 seconds, with $\gamma_{min} = 0.9091$. This value of γ was relaxed by 50% to 1.3636, and the control synthesis problem was re-solved.

For the remainder of this thesis, this controller will be referred to as the H_∞ controller.

4.2 LPV Controller Based on a Parameter-Independent Lyapunov Function

For the LPV controller obtained using a parameter-independent Lyapunov function, the approach from Section 2.5.4.1 is applied. As there is only one parameter, the vertices w_1 and w_2 represent the endpoints of the parameter interval. The state-space matrices, namely the A matrix, from 4.10 are evaluated at each of these interval endpoints, resulting in two state-space systems.

All penalty weights for this LPV controller were chosen identical to those of the previous standard H_∞ controller, with one exception. It was found that this controller behaved more erratically, producing high values of bank angle to the point of loss of control of the aircraft. To mitigate this problem, a constant penalty weight of 0.1910 was applied to the bank angle ϕ. This is added to the system W_y.

To design the LPV controller with parameter-independent Lyapunov function, the following semi-definite programming problem is solved:

$$\text{minimize } \gamma$$

$$\text{subject to: } 2.37 - 2.39$$

This SDP was solved in 39.04 seconds, with $\gamma_{min} = 0.9085$. This value of γ was relaxed 50% to 1.3628, and the control synthesis problem was resolved.

For the remainder of this thesis, this controller will be referred to as the PI-LPV controller.

4.3 LPV Controller Based on a Parameter-Dependent Lyapunov Function

The LPV controller obtained using a parameter-dependent Lyapunov function was solved for two solution forms:

$$\text{Solution 1: } R(p) = R_0 + pR_1,\ S(p) = S_0 + pS_1$$

$$\text{Solution 2: } R(p) = R_0 + pR_1 + p^2R_2,\ S(p) = S_0 + pS_1 + p^2S_2$$

Multiconvexity relaxation was applied as described in 2.5.4.2. A full listing of the problem formulation and resulting LMIs for both solution cases can be found in Appendix C. Penalty weightings were chosen identically to the previous PI-LPV controller.

To design the LPV controller with parameter-dependent Lyapunov function, the following semi-definite programming problem is solved:

$$\text{minimize } \gamma^2$$

$$\text{subject to: } 2.40 - 2.42$$

For both solution cases, γ_{min} was found to be the same as that of the parameter-independent case. Thus, Solution 1 was chosen due to the noticeable decrease in the number of LMIs required.

For the remainder of this thesis, this controller will be referred to as the PD-LPV controller.

4.4 H_∞ Rate Tracking Controller

Due to the rate tracking controller's inner-outer loop structure, the combined UAV/path following linearized system matrices are not used in the control design. Instead, only the UAV equations of motion are linearized for the formulation of the plant. These system matrices are denoted with the superscript "UAV". An H_∞ controller is then designed to track the pitch and yaw rate commands generated by Equation 2.20. A diagram of the high-level control structure is shown in Figure 4.2.

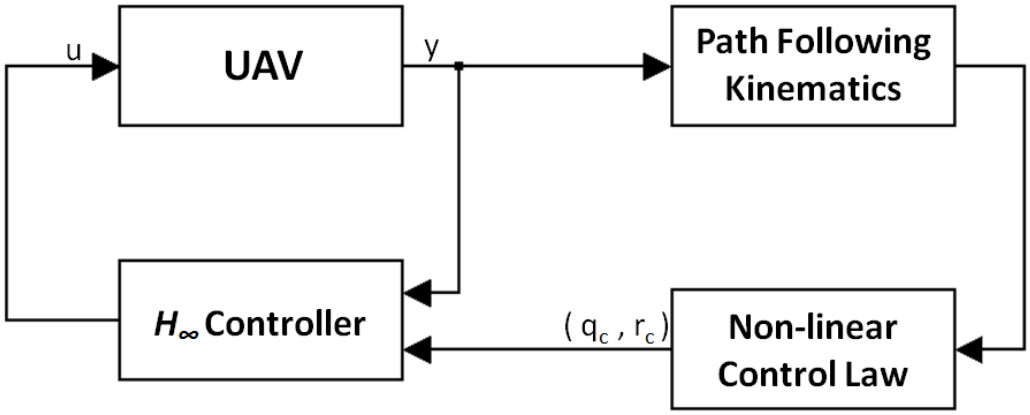


Figure 4.2: Control diagram for rate tracking H_∞ controller

To begin, the ideal response for the rate tracking objective was defined as that of a second-order system with $\omega_n = 12$ and $\zeta = 0.8$. An example of using an ideal response model in a command tracking problem in literature can be found in [41]. This command signal, namely, $w^{\text{cmd}} = [q^{\text{cmd}},\ r^{\text{cmd}}]^T$ is augmented to the disturbance vector $w(k)$, and the respective component of W^{dist} is set to the identity matrix. In a similar fashion to the previous controllers, adding a second-order system weighting the command signals adds states to the overall system, denoted x^{cmd}. The state matrix for this control design case becomes $x = [x^{\text{UAV}}, x^{\text{act}}, x^{\text{cmd}}, x^{W_u}, x^{W_y}]^T$. As with the previously discussed controllers, these dynamic weightings can be expressed in the state space representation given in Equation 4.8. The command system is represented by the superscript "cmd". The state-space representation of the plant can then be expressed as given below.

$$
\begin{aligned}
x(k+1) = &\begin{bmatrix} A^{\mathrm{UAV}}(k_1(\ell)) & B_2^{\mathrm{UAV}} C^{\mathrm{act}} & 0 & 0 & 0 \\ 0 & A^{\mathrm{act}} & 0 & 0 & 0 \\ 0 & 0 & A^{\mathrm{cmd}} & 0 & 0 \\ 0 & 0 & 0 & A^{W_u} & 0 \\ B^{W_y} C_2^{\mathrm{UAV}} & 0 & 0 & 0 & A^{W_y} \end{bmatrix} x(k) \\
&+ \begin{bmatrix} B_1^{\mathrm{UAV}} \\ 0 \\ B^{\mathrm{cmd}} \\ 0 \\ 0 \end{bmatrix} W^{\mathrm{dist}} w(k) + \begin{bmatrix} B_2^{\mathrm{UAV}} D^{\mathrm{act}} \\ B^{\mathrm{act}} \\ 0 \\ B^{W_u} \\ 0 \end{bmatrix} u(k) \\
z(k) = &\begin{bmatrix} D^{W_y} C_2^{\mathrm{UAV}} & 0 & -C^{\mathrm{cmd}} & 0 & C^{W_y} \\ 0 & 0 & 0 & C^{W_u} & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 & -D^{\mathrm{cmd}} \\ 0 & 0 \end{bmatrix} W^{\mathrm{dist}} w(k) + \begin{bmatrix} 0 \\ D^{W_u} \end{bmatrix} u(k) \\
y(k) = &\begin{bmatrix} C_2^{\mathrm{UAV}} & 0 & -C^{\mathrm{cmd}} & 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 & 0 & D_{21}^{\mathrm{UAV}} & -D^{\mathrm{cmd}} \end{bmatrix} W^{\mathrm{dist}} w(k)
\end{aligned}
\tag{4.10}
$$

The difference between the vehicle pitch and yaw rate from the ideal response signals was weighted by a constant value of 1. The control surface deflection commands and throttle command were weighted by second-order high-pass Butterworth filters with a cutoff frequency of 2 Hz, and unity gain. A high-pass filter weighting was chosen in order to penalize primarily high frequency control commands. Velocity was penalized by a constant value of 0.02, and α and β were penalized by a constant value of 0.6.

To design the H_∞ rate tracking controller, the results from Section 2.5.3 are applied. Specifically, the following semi-definite programming problem is solved:

$$\text{minimize } \gamma$$

$$\text{subject to: } 2.26 - 2.28$$

This SDP was solved in 8.26 seconds, with $\gamma_{min} = 1.00$. This value of γ was relaxed by 50% to 1.50, and the control synthesis problem was re-solved.

For the remainder of this thesis, this controller will be referred to as the Rate Tracking controller, or RT controller.

Chapter 5

Results and Discussion

5.1 Simulation Environment

To test the resulting controllers, a simulation environment was developed in MATLAB that was designed to analyze controller performance under realistic operating conditions. Steady wind disturbances, wind gusts, measurement noise and delay, and actuator model uncertainty are included in the simulation environment.

5.1.1 Steady Wind

Steady wind is applied as in Equation 2.5. North/South, East/West, and up/down velocity components of the steady wind are supplied to the system dynamics. These velocities can be chosen as constants for a given run, or as time-varying quantities with desired time dependence to produce changes in magnitude or direction. For all simulations in this work, steady wind was chosen as a constant 5 m/s North wind.

5.1.2 Wind Gusts

Wind gusts were modeled using the Dryden Wind Turbulence model (MIL-F-8785C)[42]. The Telemaster aircraft operates at altitudes of less than 1000 feet, so the low altitude model of turbulence scaling, intensity, and orientation was used [43]. The velocity turbulence components are generated by the transfer functions:

$$\begin{aligned} H_u(s) &= \sigma_u \sqrt{\frac{2L_u}{\pi V_a}} \cdot \frac{1}{1 + \frac{L_u}{V_a} s} \\ H_v(s) &= \sigma_v \sqrt{\frac{L_v}{\pi V_a}} \cdot \frac{1}{\left(1 + \frac{L_v}{V_a} s\right)^2} \\ H_w(s) &= \sigma_w \sqrt{\frac{L_w}{\pi V_a}} \cdot \frac{1}{\left(1 + \frac{L_w}{V_a} s\right)^2} \end{aligned} \tag{5.1}$$

where $\sigma_{(\cdot)}$ is the turbulence intensity for each direction, $L_{(\cdot)}$ represents the turbulence scale for each direction, and V_a is the vehicle airspeed. For the low altitude model, the turbulence scale lengths and intensities are:

$$\begin{aligned} L_w &= h \\ L_u = L_v &= \frac{h}{(0.177 + 0.000823h)^{1.2}} \\ \sigma_w &= 0.1 u_{20} \\ \frac{\sigma_u}{\sigma_w} = \frac{\sigma_v}{\sigma_w} &= \frac{1}{(0.177 + 0.000823h)^{0.4}} \end{aligned} \tag{5.2}$$

where h is the altitude of the vehicle above ground level in feet, and u_{20} is the wind speed at 20 feet above the ground. Typically, this value is 15 knots for light turbulence, 30 knots for moderate turbulence, and 45 knots for severe turbulence. The u gust direction for low altitude model is defined as along the horizontal relative mean wind vector, and the w gust direction is aligned with the vertical. The inputs to the transfer functions 5.1 are unit

variance band-limited white noise signals, and the outputs are the u, v, and w wind gust components, denoted u_g, v_g, and w_g, respectively. A sample of a typical gust field used in simulation is shown in Figure 5.1.

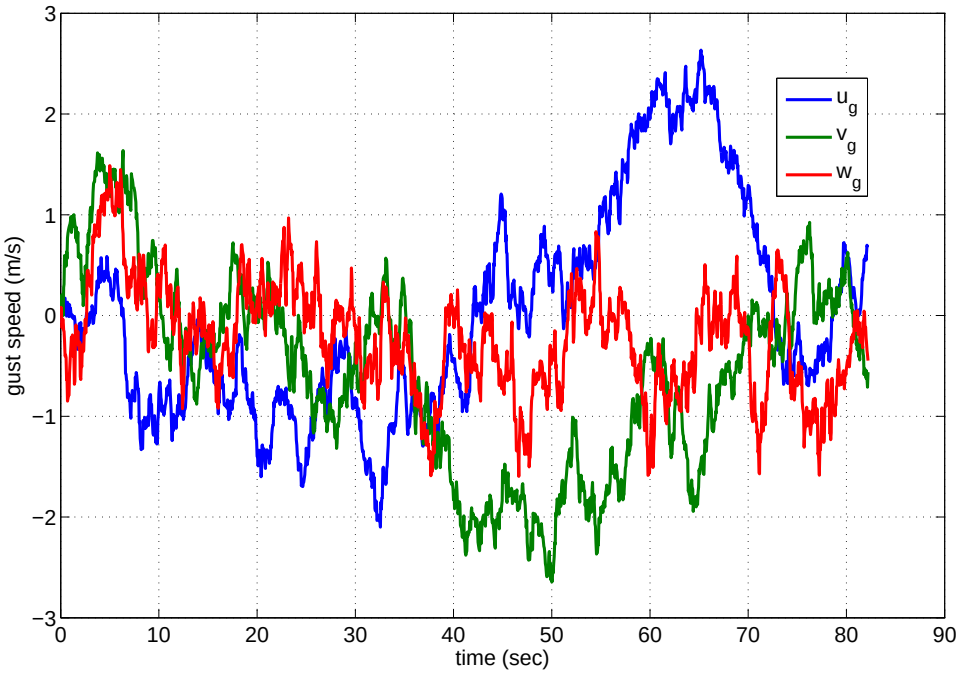


Figure 5.1: Sample Dryden wind gust field in simulation

5.1.3 Measurement Noise and Delay

The measurement noise included in the simulation environment attempts to emulate the actual performance of the sensors on the Telemaster platform (Section 2.3). The measurement systems operate at 20 Hz, with the exception of GPS, which operates at 4 Hz. However, a Kalman filter is used in flight to estimate position data between GPS updates, and thus the GPS is assumed to operate at 20 Hz as well. All measurement noise is modeled using the `randn` function in MATLAB. Standard deviations of the measurement noise are given in

Table 5.1. Table 5.1: Measurement noise standard deviations

$p/q/r$	0.2 deg/s
V	0.5 m/s
α	2.75 deg
β	1.3 deg
$\phi/\theta/\psi$	2 deg
North/East position	0.833 m
Altitude	1.33 m
$a_x/a_y/a_z$	0.005 g

Testing of the hardware systems on the UAV identified a delay from the sensors to the controller (measurement delay) of approximately 15 milliseconds (ms). This delay is incorporated into the simulation environment by partitioning the 50 ms integration steps into two parts: a 15 ms integration following a system measurement using the control input calculated at the previous measurement instant, and a 35 ms integration following a control input update. It is important to note that although delay is included in the simulation environment, it is not incorporated into the control design process.

5.1.4 Actuator Model Uncertainty

As the actuator models used in control design are based upon exact knowledge of the models used in simulation, it is sensible to add uncertainty to the actuator models used during simulation in order to reflect real world modeling uncertainty. This is achieved by perturbing the natural frequencies and damping ratios of each actuator model by $\pm 5\%$. This is implemented in simulation by scaling each natural frequency and damping ratio for each actuator model by a normally distributed random number with mean one and standard deviation of approximately 0.0167. Thus, the actuator model used in control design is at best a close approximation to the real actuator dynamics, as would be expected in real world modeling and control design.

5.2 Geometric Path

The geometric path to be followed by the UAV is shown in Figure 5.2.

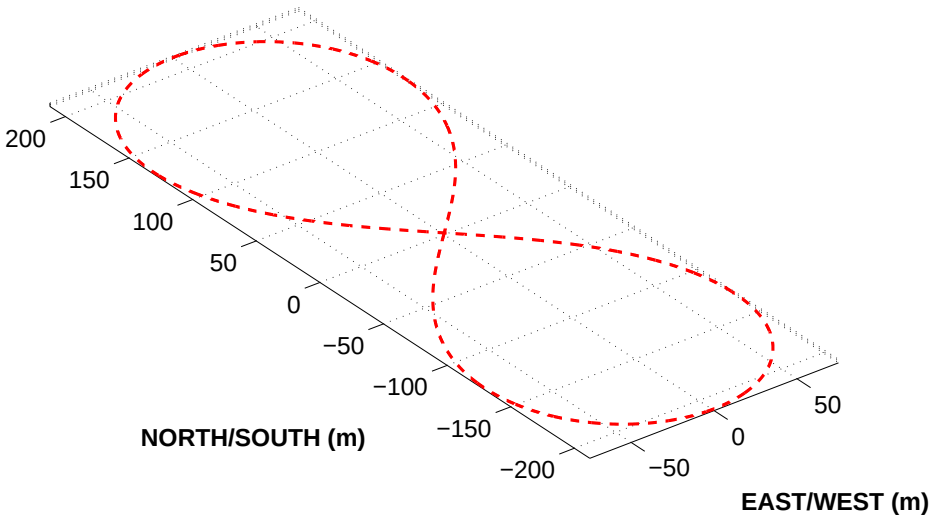


Figure 5.2: Figure-8 reference path

This path is generated by the functions

$$\begin{aligned} x(\xi) &= \frac{150}{\sqrt{2}} \frac{\cos(\xi)}{1+\sin(\xi)^2} \\ y(\xi) &= \frac{150}{\sqrt{2}} \frac{\sin(\xi)\cos(\xi)}{1+\sin(\xi)^2} \\ \xi &\in [\pi/2, 9\pi/2] \end{aligned} \tag{5.3}$$

Note that due to the method of construction, the path is not guaranteed to be dynamically feasible, and as such it is chosen somewhat conservatively in terms of radius. If the path were followed perfectly at 15 m/s, it would take 74 seconds to traverse the path. A plot of the path parameter $k_1(\ell)$ versus path length for this figure-8 path is shown in Figure 5.3.

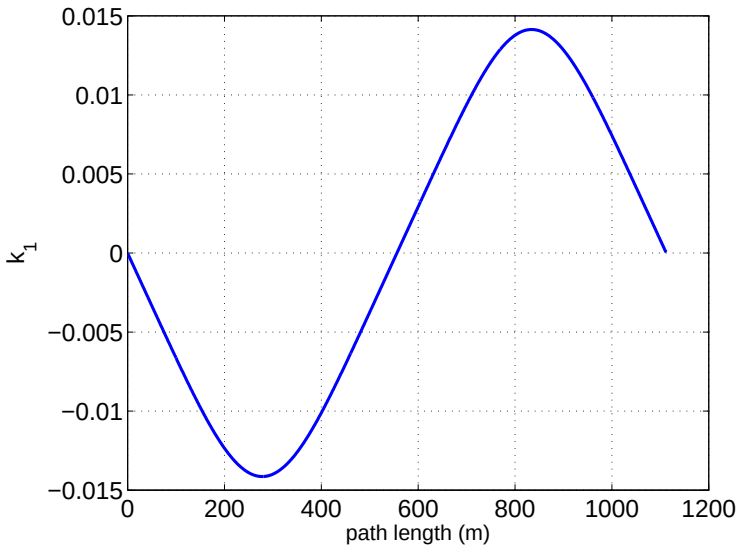


Figure 5.3: $k_1(\ell)$ for reference path

For the purpose of control design, the parameter $k_1(\ell)$ varies in the interval $[-0.0141, 0.0141]$. Its increment, $dk_1(\ell)$, varies in the interval $[-1 \times 10^{-3}, 1 \times 10^{-3}]$.

5.3 Performance Comparison Metrics

All controllers are tested at "worst case" expected flight conditions during the flight objective, with steady wind disturbance set to a 5 m/s steady North wind, and gusts at severe intensity. In this case, the flight objective is to track a figure-8 path. For the results to be statistically meaningful, it is prudent to use each controller to force the UAV to track the figure-8 path 1000 times consecutively. In order to compare controller performance, a metric is defined to quantitatively compare the path following performance. For this work, this metric was chosen to be the average error between the UAV and the closest point on the path over each completion of the figure-8 path. In order to be able to easily compute the distance to

the path, the path is "discretized" in the sense that only a finite set of points on the path are considered in the computation of the distance. That is, the parametric variable ξ in Equation 5.3 is sampled at a uniform interval, $\delta\xi$, resulting in N_ξ uniformly spaced points along the path. Note that as $N_\xi \to \infty$, $\delta\xi \to 0$. In our case, $\delta\xi$ was chosen sufficiently small such that the distance between the discretized points was on the order of centimeters. This has the effect of also discretizing the path length, by which we consider our path to be parameterized. Thus the set of points on the path, P, can be given as follows.

$$P = \{p(\ell_i) \text{ for } i = 1,\, 2,\, \ldots,\, N_\xi\} \tag{5.4}$$

Next, we define the function "dist" as

$$\text{dist}(a, P) = \inf\{\|a - b\| \,|\, b \in P\} \tag{5.5}$$

That is, $\text{dist}(a, P)$ gives the minimum distance between the point a and the set P. Given the above, we define the path error as

$$\text{Path Error} = \frac{\sum_{k=t_1}^{t_N} \text{dist}(Q_I(k), P)}{N} \tag{5.6}$$

where t_1 denotes the initial time of the figure-8 circuit, t_N the final time of the figure-8 circuit, N equals the number of measurements over the circuit of the figure-8, and $Q_I(k)$ is the position of the UAV at time k. The path error corresponding to circuit i will be denoted by d_i.

For each completion (or circuit) of the figure-8 path, we compute the path error. After the UAV traverses all 1000 circuits completely, we obtain the sequence of path errors, d_i, for $i = 1, 2, \ldots, 1000$. This sequence can then be regarded as a distribution of path errors, which serves as a point of comparison between the controllers. Specifically, the path error will be used as a metric to assess the path following performance of the various designed controllers.

Other factors considered in assessing controller performance include norms of the feedback control input histories (to examine total control effort):

$$\text{Control Effort} = \text{CE}_{\delta_{(\cdot)}} = \left(\sum_{k=0}^{N_T - 1} \| \delta_{(\cdot)}(k) - \delta_{(\cdot)_{\text{trim}}} \|^2 \right)^{\frac{1}{2}} \tag{5.7}$$

where N_T is the total number of time-instants over the 1000 circuits of the figure-8, and $\delta_{(\cdot)}$ is the input signal. Additionally, maximum and minimum excursion from trim aerodynamic angles, and velocity tracking performance will be examined.

5.4 Rate Tracking Controller

The rate tracking controller completed all 1000 circuits without failure. The mean and median of the previously defined "path error" sequence are 3.32 m and 3.29 m, respectively. Figure 5.4 gives the histogram of this sequence. Notice that the distribution has a slight positive skew. All of the data is contained within a 1.89 m interval, with a minimum attained path error of 2.58 m. On average, a circuit of the figure-8 took 82.48 seconds to complete. A summary of overall control performance is given in Table 5.2, and the worst-case circuit completion, as judged by path error, is shown in Figure 5.5.

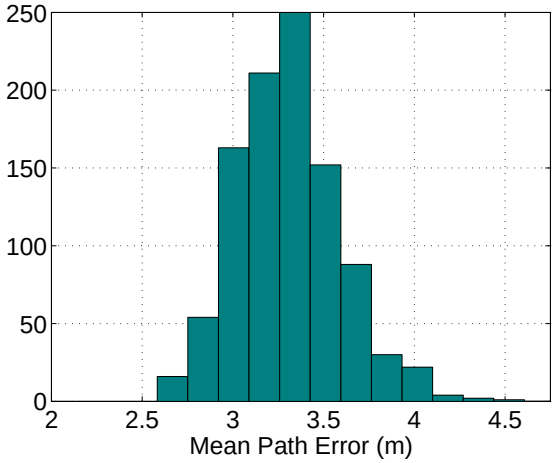


Figure 5.4: Mean path error for rate tracking H_∞ controller

Table 5.2: Rate tracking H_∞ performance metrics

Mean path error	3.32 m
Median path error	3.29 m
Min/Max/σ V	$[10.28, 20.15]$ m/s, $\sigma_V = 1.39$ m/s
Min/Max/σ α	$[-11.94, 18.04]$ deg, $\sigma_\alpha = 3.31$ deg
Min/Max/σ β	$[-17.53, 18.03]$ deg, $\sigma_\beta = 4.45$ deg
CE_{δ_e}	56.20
CE_{δ_a}	55.22
CE_{δ_r}	47.75
CE_{δ_T}	23.40

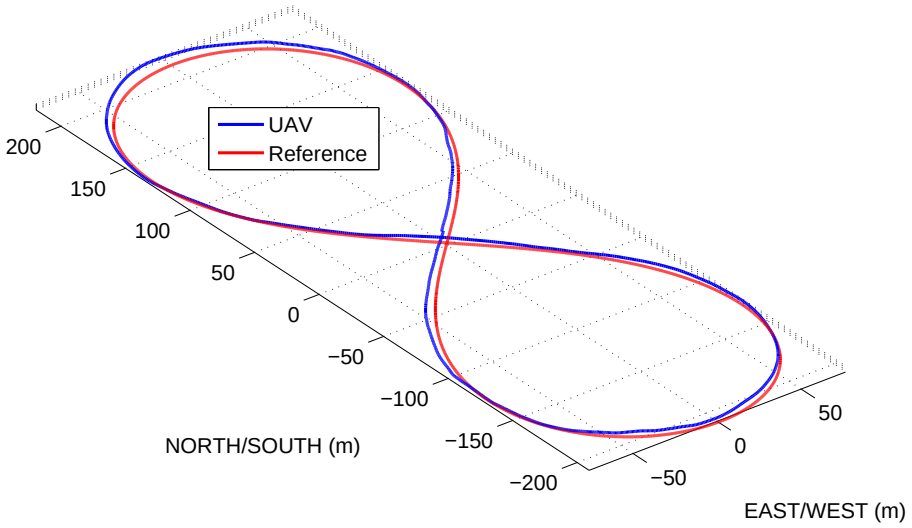


Figure 5.5: Figure-8 tracking for worst-case performance circuit of rate tracking controller

Plots of the state and control histories for the worst-case circuit of the figure-8 path can be found in Appendix A.

5.5 Standard H_∞ Controller

The standard H_∞ controller completed all 1000 circuits without failure. The mean and median of the previously defined "path error" sequence are 2.13 m and 2.11 m, respectively. In terms of the mean, this represents a 35.8% improvement over the rate tracking controller. Figure 5.6 gives the histogram of this sequence. The histogram can be seen to be bimodal, which was not present in the rate tracking controller. The distribution of path errors is much narrower for the H_∞ controller, with all the data falling in a 0.54 m wide interval,

with a minimum attained path error of 1.88 m. On average, a circuit of the figure-8 took 84.28 seconds to complete, which is slightly slower than that for the rate tracking controller. A summary of overall control performance is given in Table 5.3, and the worst-case circuit completion, as judged by path error, is shown in Figure 5.7.

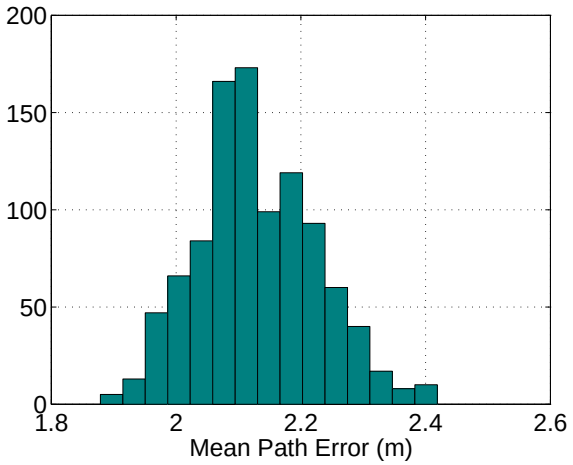


Figure 5.6: Mean path error for standard H_∞ controller

Table 5.3: Standard H_∞ performance metrics

Mean path error	2.13 m
Median path error	2.11 m
Min/Max/σ V	$[11.31, 18.03]$ m/s, $\sigma_V = 0.53$ m/s
Min/Max/σ α	$[-10.43, 19.44]$ deg, $\sigma_\alpha = 3.20$ deg
Min/Max/σ β	$[-10.81, 10.26]$ deg, $\sigma_\beta = 1.95$ deg
CE_{δ_e}	72.07
CE_{δ_a}	101.47
CE_{δ_r}	42.05
CE_{δ_T}	24.60

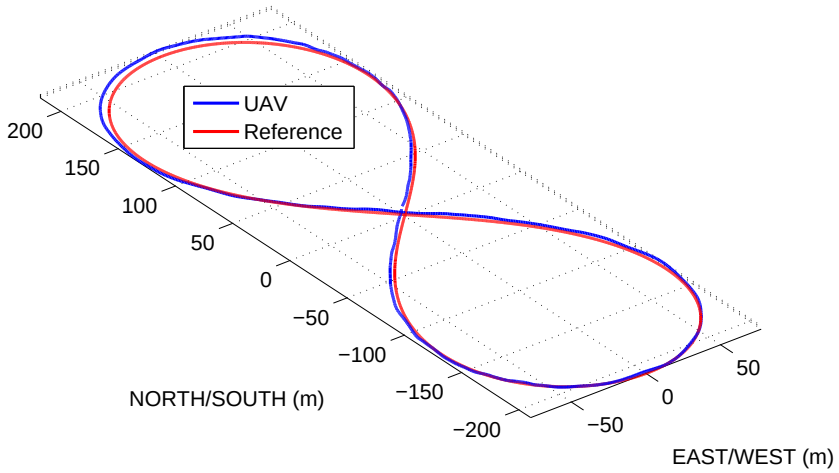


Figure 5.7: Figure-8 tracking for worst-case performance circuit of H_∞ controller

From Table 5.3, a drop in maximum velocity as well as a tighter velocity and β range relative to the rate tracking controller is noticed. However, this comes at the expense of a wider range of α. Additionally, the magnitude of the applied control input is significantly higher than in the rate tracking controller case, especially in the case of elevator and aileron. Thus, it may be concluded that the increase in performance seen in the mean path error is due to more aggressive behavior from the controller.

Plots of the state and control histories for a sample circuit of the figure-8 path can be found in Appendix A.

5.6 PI-LPV Controller

The PI-LPV controller completed all 1000 circuits without failure. The mean and median of the previously defined "path error" sequence are 2.031 and 2.027, respectively. In terms of the mean, this represents a 4.64% improvement over the standard H_∞ controller. Figure 5.8 gives the histogram of this sequence. Similar to the rate tracking controller, the histogram for the PI-LPV controller exhibits positive skew. The data falls in an interval narrower than that of the rate tracking controller, but wider than that of the H_∞ controller, with all the data falling in an interval 0.69 m wide. The minimum path error attained was 1.66 m, which is lower than the previous two controllers. On average, a circuit of the figure-8 took 83.08 seconds to complete, which is slightly closer to the ideal time to circuit than the H_∞ controller, but farther from the ideal time than the rate tracking controller. A summary of overall control performance is given in Table 5.4, and the worst-case circuit completion, as judged by path error, is shown in Figure 5.9.

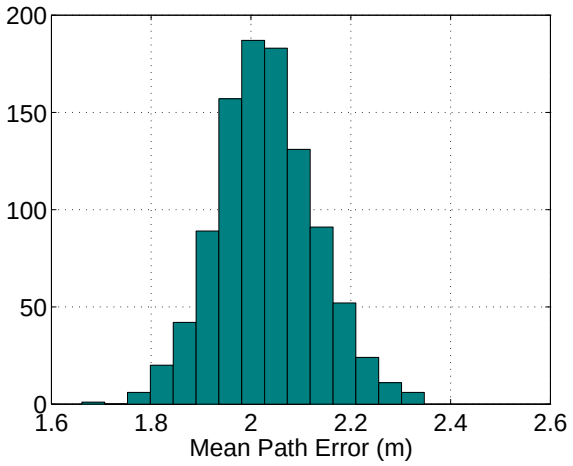


Figure 5.8: Mean path error for PI-LPV controller

Table 5.4: PI-LPV performance metrics

Mean path error	2.031 m
Median path error	2.027 m
Min/Max/σ V	$[10.14, 17.96]$ m/s, $\sigma_V = 0.56$ m/s
Min/Max/σ α	$[-10.03, 21.26]$ deg, $\sigma_\alpha = 3.15$ deg
Min/Max/σ β	$[-11.26, 12.24]$ deg, $\sigma_\beta = 1.86$ deg
CE_{δ_e}	67.47
CE_{δ_a}	116.81
CE_{δ_r}	46.39
CE_{δ_T}	23.95

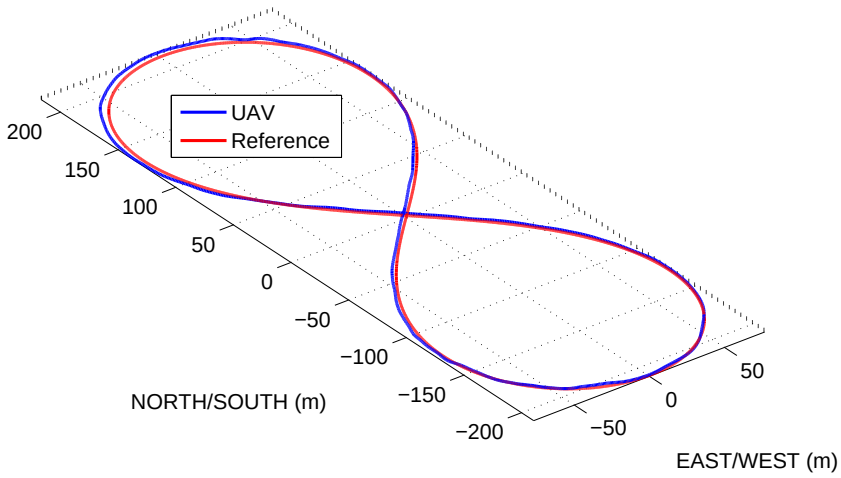


Figure 5.9: Figure-8 tracking for worst-case performance circuit of PI-LPV controller

From Table 5.4, the results can be seen to be comparable to those of the standard H_∞

controller. Again, the range in α has increased, as well as β, giving a possible explanation to the slight decrease in mean path error. Overall, the total control input, with the exception of an increase in aileron control effort, is comparable to that of the standard H_∞ results.

Plots of the state and control histories for a sample circuit of the figure-8 path can be found in Appendix A.

5.7 PD-LPV Controller

The PD-LPV controller completed all 1000 circuits without failure. The mean and median of the previously defined "path error" sequence are each 2.57 m. In terms of the mean, this represents a 22.6% increase over the rate tracking controller, but still slightly worse than the H_∞ or PI-LPV controllers. Figure 5.10 gives the histogram of this sequence. The distribution of the path errors has no discernable skew. All of the data falls within an interval 1.43 m wide, which again puts it between the rate tracking and H_∞/PI-LPV controllers. The minimum path error attained was 1.87 m, which is similar to that of the H_∞ controller. On average, a circuit of the figure-8 took 80.61 seconds to complete, which puts it the closest to the ideal time of all the controllers. A summary of overall control performance is given in Table 5.5, and the worst-case circuit completion, as judged by path error, is shown in Figure 5.11.

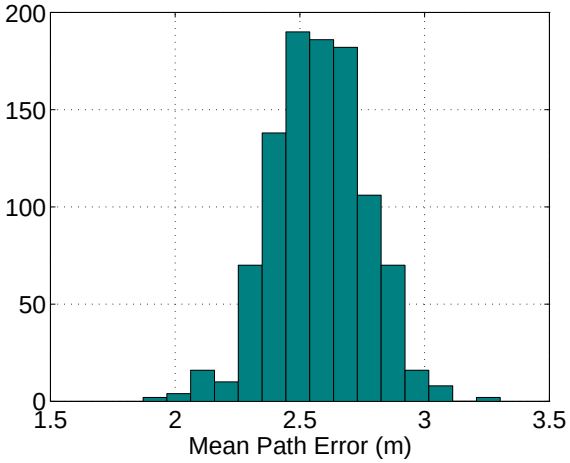


Figure 5.10: Mean path error for PD-LPV controller

Table 5.5: PD-LPV performance metrics

Mean path error	2.57 m
Median path error	2.57 m
Min/Max/σ V	$[12.13, 19.90]$ m/s, $\sigma_V = 0.69$ m/s
Min/Max/σ α	$[-10.87, 20.35]$ deg, $\sigma_\alpha = 2.89$ deg
Min/Max/σ β	$[-22.50, 22.02]$ deg, $\sigma_\beta = 5.54$ deg
CE_{δ_e}	44.44
CE_{δ_a}	155.39
CE_{δ_r}	57.96
CE_{δ_T}	17.31

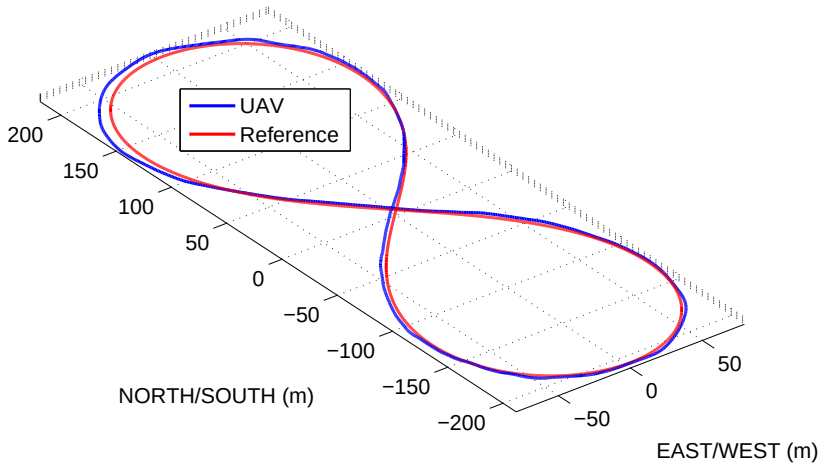


Figure 5.11: Figure-8 tracking for worst-case performance circuit of PD-LPV controller

From Table 5.5, the results can be seen to be worse than the previous two controllers, with the most noticeable decrease in performance coming in the form of the path error and wide range of β. However, this controller did track the path faster on average (i.e. closer to the ideal time) in comparison to the previous controllers.

It is important to note that this result is not indicative of the efficacy of the parameter-dependent control or multiconvexity relaxation techniques in general. Rather, it only shows that for the current system under investigation and our specific problem formulation, these methods did not result in an increase in performance.

Plots of the state and control histories for a sample circuit of the figure-8 path can be found in Appendix A.

5.8 Summary of Results

It is apparent that the rate tracking, H_∞, PI-LPV, and PD-LPV controllers are comparable in their performance. It is difficult, however, to draw direct and definitive comparisons, and as such, the conclusions are based on the specific problem formulations.

However, noting that the performance for these four controllers was comparable, there is a clear argument to be made for the combined system control methods presented in this work. A major drawback of a control method like the rate tracking controller is the sheer number of tunable parameters. In the nonlinear control law formulation from [9], there are 7 distinct tunable constants that affect the resulting rate commands, some in non-intuitive ways. The addition of the H_∞ controller only compounds on this issue. By combining the UAV and path following systems into one system, the amount of tuning needed in the control design process becomes more manageable.

Additionally, the application of H_∞ control allows the control designer to easily tune controller performance. For example, the control designer can deal with the trade off between path following performance and input costs directly and judiciously through careful selection of the penalty weightings. For some specific nonlinear controllers, for example the nonlinear control law from [9] contained within, the method to tune the balance of this trade-off is not always as clear and intuitive. Moving forward, these results are encouraging for the application of the designed controllers in real flight testing scenarios.

Chapter 6

Conclusions and Future Work

The use of linear parameter varying control as applied to a path following problem was shown to be comparable in performance to a method existing in the literature. While a direct comparison is difficult to make, the ease of implementation of some of these methods in comparison to the reference method makes them potentially valuable in application. In addition, the controllers designed within were shown to be sufficiently robust to noise, disturbances, delays, and modeling inaccuracies, with 62.5 hours of failure free simulated flight time.

The results presented within are in no way intended to be general, and are highly dependent on the system being analyzed. They are even more highly dependent on the control designer, and other choices of outputs to penalize or penalty weight formulations may prove to produce better results for one or several of the controllers or methods presented within.

Current efforts are focused on implementing the previously described LPV path following control methods on the actual Telemaster UAV system. An aerodynamic model for the Telemaster UAV has been identified using parameter estimation techniques similar to those

outlined in Chapter 3. Using this new model, the control synthesis procedure, as discussed in Chapter 4, is re-applied to obtain new controller matrices. Look-up tables of relevant pre-computed path information for use in the virtual vehicle dynamics and on-line controller determination are saved to the on-board computing hardware. Initial hardware in the loop testing has shown promising results.

Areas of future theoretical work include extending the combined path following system to 3-dimensional paths, which possibly involves the inclusion of $k_2(\ell)$ and ϕ_{err} as system parameters, if LPV control is pursued. Additionally, scheduling of the aircraft dynamics may be a topic of future work that shows promise for increasing path following performance. One final area of future work that may show promising results when applied to the path following problem is applying the concept of energy height to the scheduling of the speed profile. This may result in a more uniform tracking performance than the constant speed profile shown within.

Bibliography

[1] A. P. Aguiar, J. P. Hespanha, and P. Kokotovic, "Performance Limitations in Reference-Tracking and Path-Following for Nonlinear Systems," *Automatica*, vol. 44(3), no. March, pp. 598–610, 2008.

[2] R. Skjetne, *The Maneuvering Problem*. PhD thesis, 2005.

[3] R. Rysdyk, "Unmanned Aerial Vehicle Path Following for Target Observation in Wind," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 5, pp. 1092–1100, 2006.

[4] R. A. Wise and R. T. Rysdyk, "UAV Coordination for Autonomous Target Tracking," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pp. 3210–3231, 2006.

[5] I. Kaminer, O. Yakimenko, V. Dobrokhodov, A. Pascoal, N. Hovakimyan, C. Cao, A. Young, and V. Patel, "Coordinated Path Following for Time-critical Missions of Multiple UAVs via $\mathcal{L}_1$ Adaptive Output Feedback Controllers," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007.

[6] J. Osborne and R. Rysdyk, "Waypoint Guidance for Small UAVs in Wind," in *AIAA Infotech@ Aerospace*, (Reston, VA), pp. 459–470, 2005.

[7] D. R. Nelson, D. B. Barber, T. W. McLain, and R. W. Beard, "Vector Field Path Following for Miniature Air Vehicles," *Robotics, IEEE Transactions on*, vol. 23, no. 3, pp. 519–529, 2007.

[8] "Adaptive, Non-singular Path-Following Control of Dynamic Wheeled Robots, author=Soetanto, D and Lapierre, L and Pascoal, A, booktitle=Decision and Control, 2003. Proceedings. 42nd IEEE Conference on, volume=2, pages=1765–1770, year=2003, organization=IEEE,"

[9] I. Kaminer, A. Pascoal, E. Xargay, N. Hovakimyan, C. Cao, and V. Dobrokhodov, "Path Following for Unmanned Aerial Vehicles Using $\mathcal{L}_1$ Adaptive Augmentation of Commercial Autopilots," *Journal of Guidance, Control, and Dynamics*, vol. 33, pp. 550–564, Mar. 2010.

[10] J.-M. Yang and J.-H. Kim, "Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 3, pp. 578–587, 1999.

[11] K. Illif, "Parameter Estimation for Flight Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 12, no. 5, 1989.

[12] V. Klein, "Estimation of Aircraft Aerodynamic Parameters from Flight Data," *Progress in Aerospace Sciences*, vol. 26, pp. 1–77, 1989.

[13] R. Jategaonkar, "Algorithms For Aircraft Parameter Estimation Accounting for Process and Measurement Noise," *Journal of Aircraft*, vol. 26, no. 4, pp. 360–372, 1989.

[14] R. V. Jategaonkar, *Flight Vehicle System Identification: A Time Domain Methodology*. AIAA, 2006.

[15] J. E. Williams and S. R. Vukelich, "The USAF Stability and Control Digital DATCOM Volume I, Users Manual," tech. rep., McDonnell Douglas Astronautics Company, St. Louis, MO, 1979.

[16] W. F. Phillips, *Mechanics of Flight*. John Wiley & Sons, Inc., 2 ed., 2010.

[17] NSL@VT, "Nonlinear Systems Lab at Virginia Tech - Platforms." `http://www.nsl.aoe.vt.edu/Telemaster.html`.

[18] O. Arifianto and M. Farhood, "Optimal Control of Fixed-Wing UAVs Along Real-Time Trajectories," in *ASME Dynamic Systems and Control Conference*, 2012.

[19] D. Dacic, *Path-following: an Alternative to Reference Tracking*. PhD thesis, University of California, Santa Barbara, 2005.

[20] R. Bishop, "There is More than One Way to Frame a Curve," *American Mathematical Monthly*, vol. 82, no. 3, pp. 246–251, 1975.

[21] A. Hanson and H. Ma, "Parallel Transport Approach to Curve Framing," *Indiana University, Techreports-TR425*, pp. 1–20, 1995.

[22] F. Klok, "Two Moving Coordinate Frames for Sweeping Along a 3D Trajectory," *Computer Aided Geometric Design*, vol. 3, no. 3, pp. 217–229, 1986.

[23] M. Farhood and G. E. Dullerud, "Duality and Eventually Periodic Systems," *International Journal of Robust and Nonlinear Control*, vol. 15, pp. 575–599, Sept. 2005.

[24] M. Farhood, "LPV Control of Nonstationary Systems : A Parameter-Dependent Lyapunov Approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 212–218, 2012.

[25] M. Farhood and E. Feron, "Obstacle-Sensitive Trajectory Regulation via Gain Scheduling and Semidefinite Programming," *IEEE Transactions on Control Systems Technology*, vol. 20, pp. 1107–1115, July 2012.

[26] P. Gahinet and P. Apkarian, "A Linear Matrix Inequality Approach to H_∞ Control," *International Journal of Robust and Nonlinear Control*, vol. 4, no. 4, pp. 421–448, 1994.

[27] P. Apkarian, P. Gahinet, and G. Becker, "Self-scheduled H_∞ Control of Linear Parameter-varying Systems : a Design Example," *Automatica*, vol. 31, no. 9, pp. 1251–1261, 1995.

[28] P. A. Parrilo, *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Inst. Technol.,, May 2000.

[29] P. Apkarian and H. D. Tuan, "Parameterized LMIs in Control Theory," *SIAM Journal on Control and Optimization*, vol. 38, no. 4, pp. 1241–1264, 2000.

[30] A. Papachristodoulou and S. Prajna, "A Tutorial on Sum of Squares Techniques for Systems Analysis," *Proceedings of the 2005, American Control Conference, 2005.*, pp. 2686–2700, 2005.

[31] J. Mohammadpour and C. W. Scherer, *Control of Linear Parameter Varying Systems with Applications*. Boston, MA: Springer US, 2012.

[32] P. Gahinet, "Explicit Controller Formulas for LMI-based H_∞ Synthesis," *Automatica*, vol. 32, no. 7, pp. 1007–1014, 1996.

[33] V. Klein and E. A. Morelli, *Aircraft System Identification: Theory and Practice*. American Institute of Aeronautics and Astronautics Reston, VA, USA, 2006.

[34] W.-L. Chan and F.-B. Hsiao, "Implementation of the Rauch-Tung-Striebel Smoother for Sensor Compatibility Correction of a Fixed-Wing Unmanned Air Vehicle.," *Sensors*, vol. 11, pp. 3738–3764, Jan. 2011.

[35] R. Mehra, "Optimal Input Signals for Parameter Estimation in Dynamic Systems–Survey and New Results," *Automatic Control, IEEE Transactions on*, vol. 19, no. 6, pp. 753–768, 1974.

[36] J. A. Mulder, Q. P. Chu, J. K. Sridhar, J. H. Breeman, and M. Laban, "Nonlinear Aircraft Flight Path Reconstruction Review and New Advances," *Progress in Aerospace Sciences*, vol. 35, pp. 673–726, 1999.

[37] H. E. Rauch, C. T. Striebel, and F. Tung, "Maximum Likelihood Estimates of Linear Dynamic Systems," *AIAA Journal*, vol. 3, pp. 1445–1450, Aug. 1965.

[38] G. Balas, R. Chiang, A. Packard, and M. Safonov, "Robust Control Toolbox 3," 2005.

[39] J. Lofberg, "YALMIP: A Toolbox for Modeling and Optimization in MATLAB," in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pp. 284–289, IEEE, 2004.

[40] R. Tütüncü, K. Toh, and M. Todd, "SDPT3: A MATLAB Software Package For Semidefinite-quadratic-linear Programming, Version 3.0," *Web page http://www. math. nus. edu. sg/˜ mattohkc/sdpt3. html*, 2001.

[41] J. Doyle, K. Lenz, and A. Packard, "Design Examples Using μ-synthesis: Space Shuttle Lateral Axis FCS During Re-entry," in *Decision and Control, 1986 25th IEEE Conference on*, vol. 25, pp. 2218–2223, IEEE, 1986.

[42] D. J. Moorhouse and R. J. Woodcock, "Background Information and User Guide for MIL-F-8785C, Military Specification-Flying Qualities of Piloted Airplanes," tech. rep., DTIC Document, 1982.

[43] S. Gage, "Creating a Unified Graphical Wind Turbulence Model from Multiple Specifications," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, 2003.

Appendix A

Simulation Plots

Below are plots of the state and control histories for a sample circuit of the figure-8 path for each of the 4 controllers.

A.1 Rate Tracking Controller

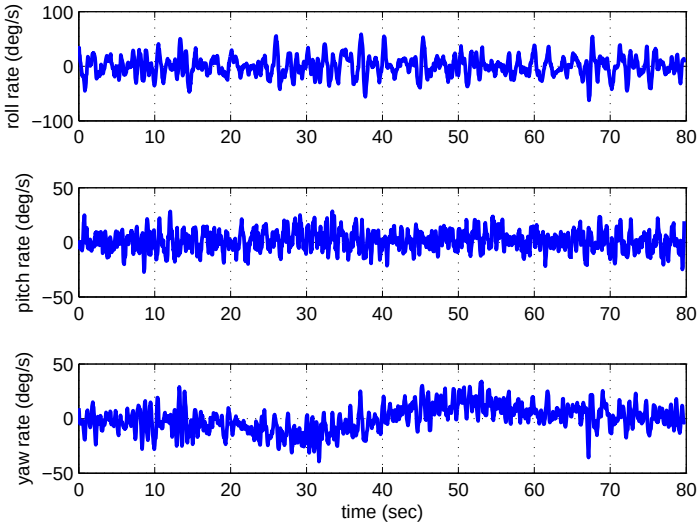


Figure A.1: p/q/r time history for rate tracking controller

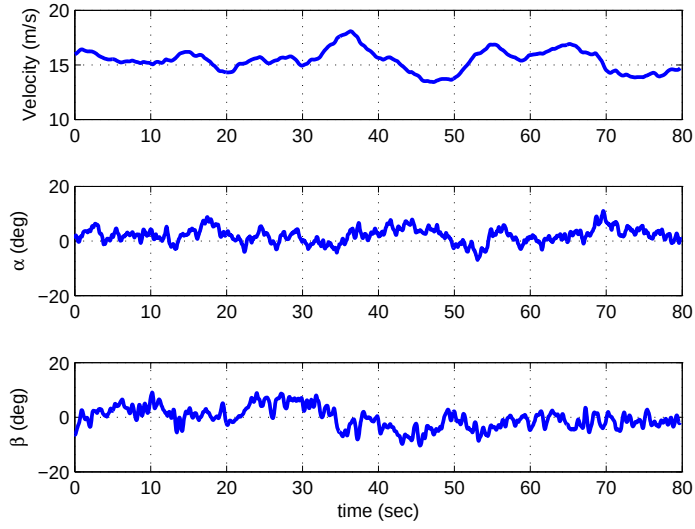


Figure A.2: $V/\alpha/\beta$ time history for rate tracking controller

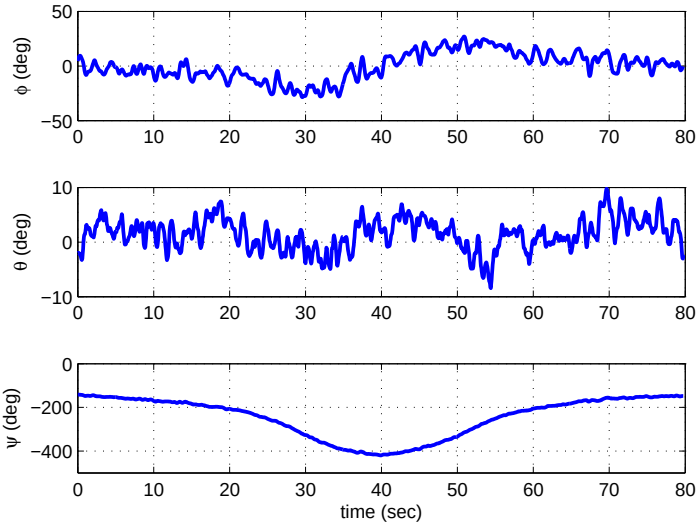


Figure A.3: $\phi/\theta/\psi$ time history for rate tracking controller

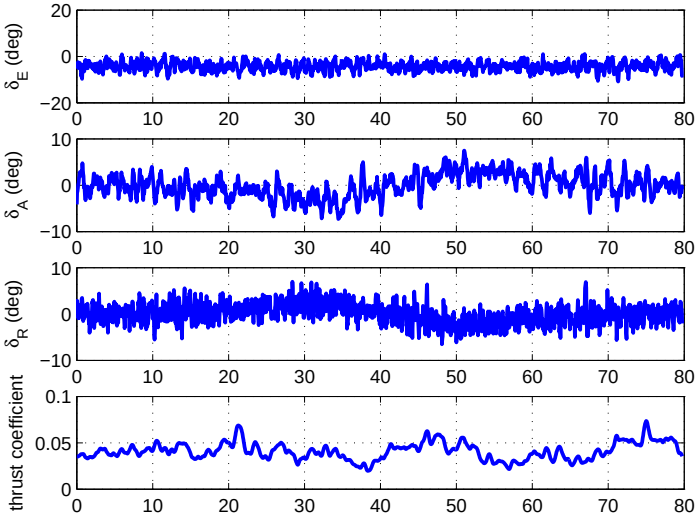


Figure A.4: Elevator/aileron/rudder/throttle time history for rate tracking controller

A.2 H_∞ Controller

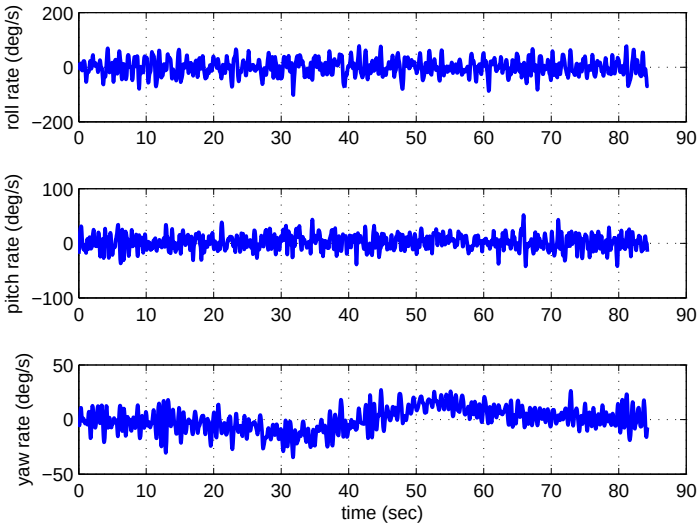


Figure A.5: p/q/r time history for rate tracking controller

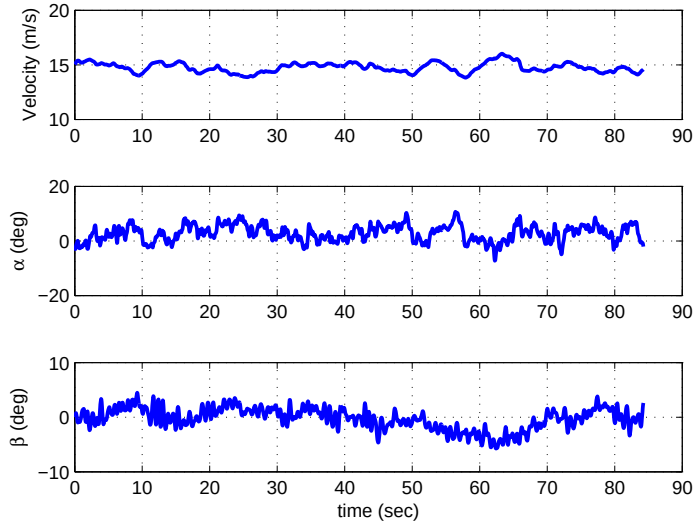


Figure A.6: $V/\alpha/\beta$ time history for rate tracking controller

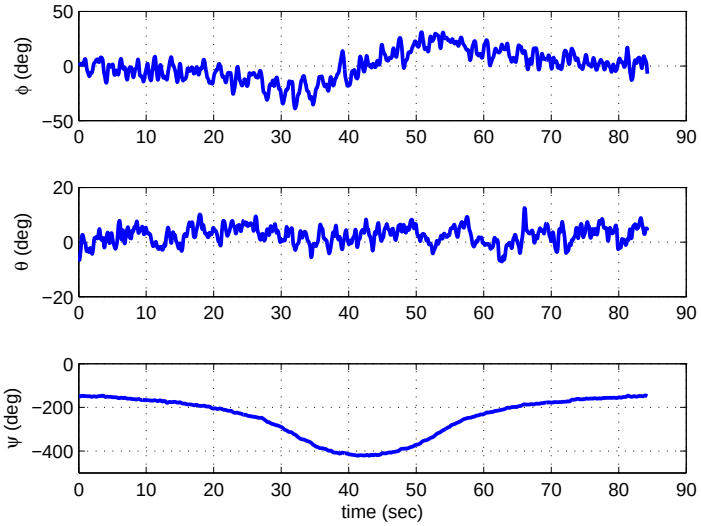


Figure A.7: $\phi/\theta/\psi$ time history for rate tracking controller

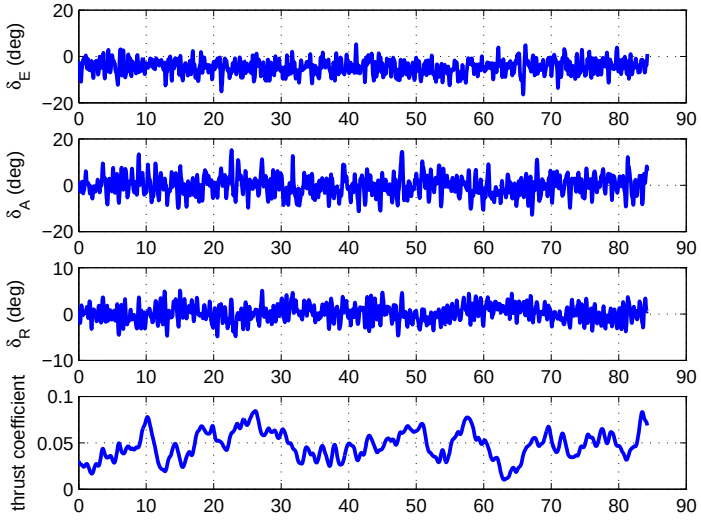


Figure A.8: Elevator/aileron/rudder/throttle time history for rate tracking controller

A.3 PI-LPV Controller

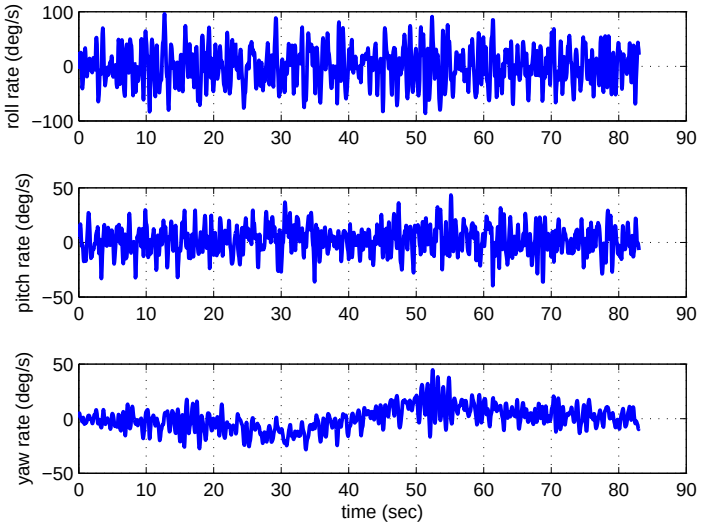


Figure A.9: p/q/r time history for rate tracking controller

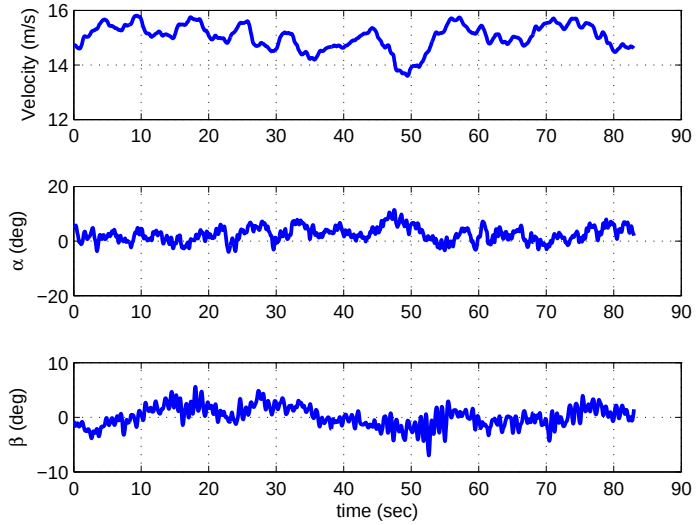


Figure A.10: $V/\alpha/\beta$ time history for rate tracking controller

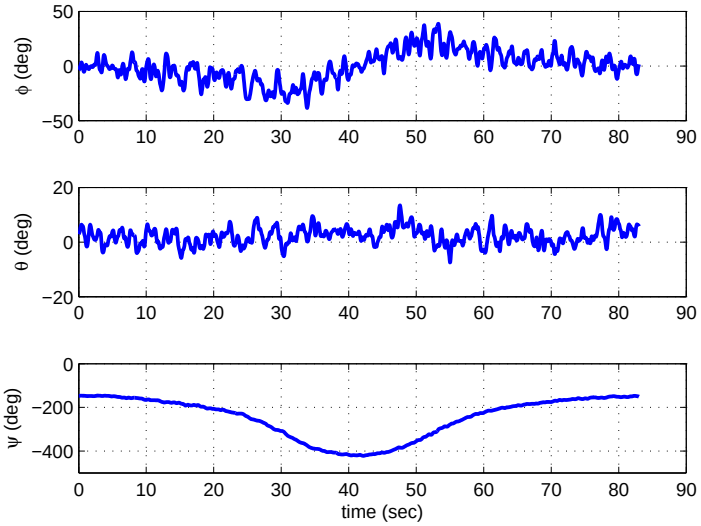


Figure A.11: $\phi/\theta/\psi$ time history for rate tracking controller

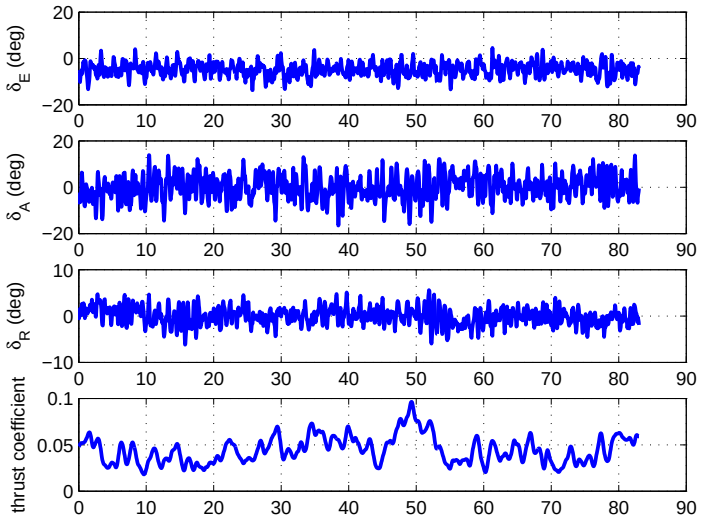


Figure A.12: Elevator/aileron/rudder/throttle time history for rate tracking controller

A.4 PD-LPV Controller

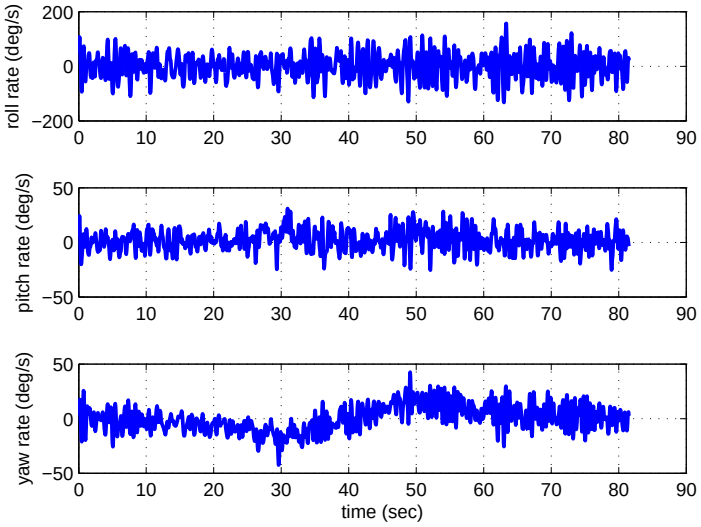


Figure A.13: p/q/r time history for rate tracking controller

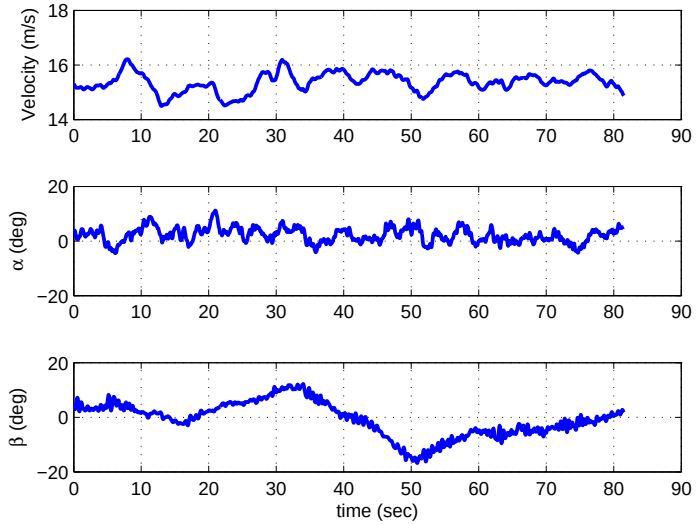


Figure A.14: $V/\alpha/\beta$ time history for rate tracking controller

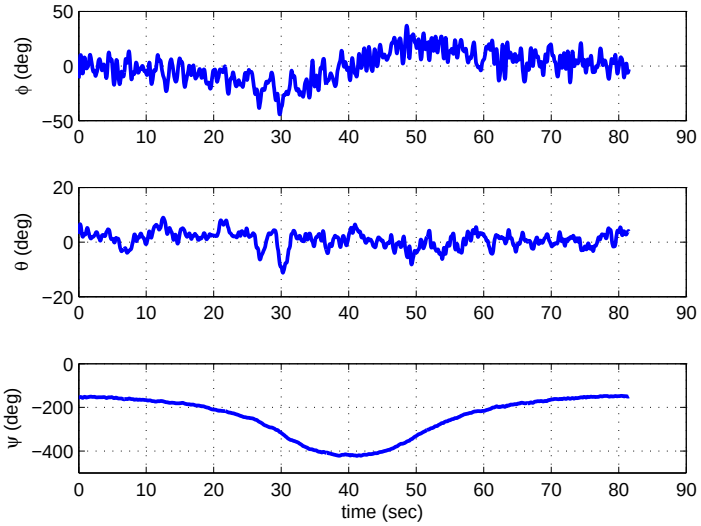


Figure A.15: $\phi/\theta/\psi$ time history for rate tracking controller

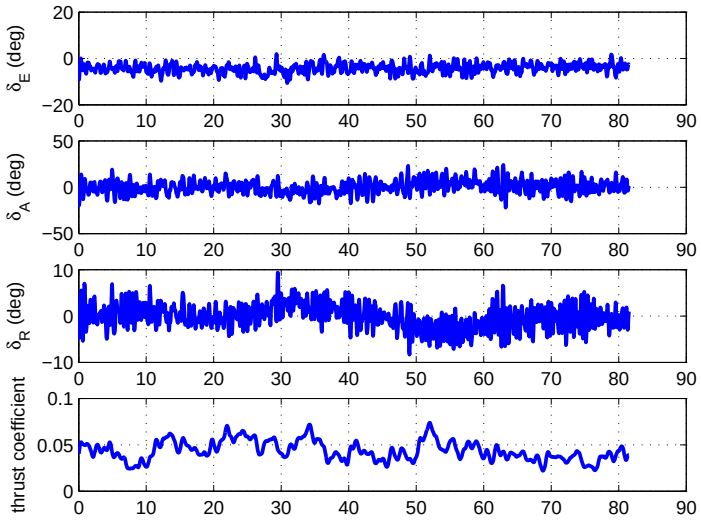


Figure A.16: Elevator/aileron/rudder/throttle time history for rate tracking controller

Appendix B

DATCOM Model

Table B.1: Static aerodynamic coefficients

α	C_D	C_L	C_m	C_Y	C_n	C_l
-10	0.052	-0.595	0.157	-0.177	0.034	-0.018
-8.	0.042	-0.417	0.125	-0.177	0.034	-0.033
-6	0.035	-0.245	0.080	-0.177	0.034	-0.047
-4	0.032	-0.080	0.032	-0.177	0.034	-0.061
-2	0.031	0.080	-0.013	-0.177	0.034	-0.074
0	0.034	0.246	-0.049	-0.177	0.034	-0.088
2	0.040	0.421	-0.081	-0.177	0.034	-0.102
4	0.051	0.605	-0.117	-0.177	0.034	-0.117
6	0.067	0.797	-0.166	-0.177	0.034	-0.133
8	0.087	0.993	-0.216	-0.177	0.034	-0.149
10	0.111	1.194	-0.253	-0.177	0.034	-0.165
12	0.136	1.367	-0.316	-0.177	0.034	-0.178
13	0.148	1.441	-0.378	-0.177	0.034	-0.183
14	0.160	1.508	-0.449	-0.177	0.034	-0.187
15	0.171	1.567	-0.527	-0.177	0.034	-0.191
16	0.181	1.618	-0.592	-0.177	0.034	-0.194
17	0.190	1.659	-0.646	-0.177	0.034	-0.196
18	0.198	1.690	-0.702	-0.177	0.034	-0.197

Table B.2: Dynamic aerodynamic coefficients

α	$C_{L_{\hat{q}}}$	$C_{m_{\hat{q}}}$	$C_{L_{\dot{\alpha}}}$	$C_{l_{\hat{p}}}$	$C_{Y_{\hat{p}}}$	$C_{n_{\hat{p}}}$	$C_{n_{\hat{r}}}$	$C_{l_{\hat{r}}}$
-10	6.764	-13.960	2.356	-6.782	-0.464	-0.106	0.045	-0.046
-8	6.764	-13.960	2.355	-6.779	-0.443	-0.093	0.031	-0.044
-6	6.764	-13.960	2.343	-6.745	-0.422	-0.081	0.018	-0.043
-4	6.764	-13.960	2.252	-6.483	-0.403	-0.070	0.006	-0.043
-2	6.764	-13.960	2.085	-6.002	-0.408	-0.059	-0.006	-0.044
0	6.764	-13.960	1.983	-5.709	-0.432	-0.047	-0.019	-0.045
2	6.764	-13.960	2.046	-5.891	-0.454	-0.035	-0.032	-0.048
4	6.764	-13.960	2.205	-6.347	-0.472	-0.022	-0.046	-0.052
6	6.764	-13.960	2.285	-6.577	-0.487	-0.009	-0.060	-0.057
8	6.764	-13.960	2.189	-6.302	-0.497	0.004	-0.074	-0.063
10	6.764	-13.960	1.941	-5.587	-0.458	0.018	-0.090	-0.071
12	6.764	-13.960	1.522	-4.382	-0.364	0.030	-0.107	-0.078
13	6.764	-13.960	1.270	-3.657	-0.319	0.036	-0.115	-0.081
14	6.764	-13.960	1.071	-3.083	-0.279	0.041	-0.121	-0.084
15	6.764	-13.960	0.897	-2.584	-0.235	0.046	-0.128	-0.086
16	6.764	-13.960	0.704	-2.027	-0.189	0.050	-0.134	-0.088
17	6.764	-13.960	0.526	-1.515	-0.137	0.054	-0.139	-0.090
18	6.764	-13.960	0.445	-1.281	-0.082	0.057	-0.142	-0.091

Table B.3: Elevator aerodynamic coefficients

δ_e	ΔC_L	ΔC_m	ΔC_D
-30	-0.126	0.397	0.017
-20	-0.110	0.344	0.008
-10	-0.067	0.208	0.002
0	0.000	0.000	0.000
10	0.067	-0.208	0.002
20	0.110	-0.344	0.008
30	0.126	-0.398	0.017

Table B.4: Aileron aerodynamic coefficients

δ_a	ΔC_l
30	-0.074
25	-0.069
20	-0.064
10	-0.036
0	0.000
-10	0.036
-20	0.064
-25	0.069
-30	0.074

Table B.5: Rudder aerodynamic coefficients

δ_r	ΔC_l	ΔC_Y	ΔC_N	ΔC_D
-30	-0.003	-0.074	0.006	0.011
-25	-0.003	-0.068	0.005	0.008
-20	-0.003	-0.065	0.005	0.005
-10	-0.002	-0.040	0.003	0.001
0	0.000	0.000	0.000	0.000
10	0.002	0.040	-0.003	0.001
20	0.003	0.065	-0.005	0.005
25	0.003	0.068	-0.005	0.008
30	0.003	0.074	-0.006	0.011

Appendix C

Multiconvexity

Given the LMIs 2.40 thru 2.42, which will be referred to as $\mathcal{F}_1(p, dp)$, $\mathcal{F}_2(p, dp)$, and $\mathcal{F}_3(p)$, two solution forms are assumed:

$$\text{Solution 1: } R(p) = R_0 + pR_1,\ S(p) = S_0 + pS_1$$

$$\text{Solution 2: } R(p) = R_0 + pR_1 + p^2R_2,\ S(p) = S_0 + pS_1 + p^2S_2$$

The parameter value p and its increment are assumed to vary in a hyper-rectangle, and so the directions d_1 and d_2 are $[1, 0]$ and $[0, 1]$ respectively. Using a hyper-rectangle instead of a polytope adds some conservatism to the solution in a trade-off for ease of implementation. Additional polynomial terms are added to $\mathcal{F}_1$ and $\mathcal{F}_2$ in an effort to both strengthen the resulting LMI, while at the same time relaxing the multiconvexity conditions. These terms are, for solution case 1:

$$F^T RF - V_1^T R_+ V_1 + H + (\nu_0 + p^2\nu_1 + dp^2\nu_2)I \prec 0 \tag{C.1}$$

$$J^T S_+ J - U_1^T S U_1 + W + (\eta_0 + p^2\eta_1 + dp^2\eta_2)I \prec 0 \tag{C.2}$$

where the constants ν and η are assumed positive, and the rest of the variables are as defined in Section 2.5.4. The multiconvexity constraints are enforced by:

$$\begin{gathered}
\left.\frac{\partial^2 \mathcal{F}_1(p+\lambda,dp)}{\partial\lambda^2}\right|_{\lambda=0} \geq 0, \qquad \left.\frac{\partial^2 \mathcal{F}_1(p,dp+\lambda)}{\partial\lambda^2}\right|_{\lambda=0} \geq 0 \\
\left.\frac{\partial^2 \mathcal{F}_2(p+\lambda,dp)}{\partial\lambda^2}\right|_{\lambda=0} \geq 0, \qquad \left.\frac{\partial^2 \mathcal{F}_2(p,dp+\lambda)}{\partial\lambda^2}\right|_{\lambda=0} \geq 0 \\
\left.\frac{\partial^2 \mathcal{F}_3(p+\lambda,dp)}{\partial\lambda^2}\right|_{\lambda=0} \geq 0
\end{gathered} \tag{C.3}$$

Note that if terms of order p^2 or dp^2 or higher remain after an application of the multiconvexity relexation, a reapplication of the constraints is necessary. For example, if a problem resulted in the multiconvexity constraint:

$$f(X,p) = X_0 + pX_1 + p^2X_2 + p^3X_3 \succeq 0$$

then a reapplication of the multiconvexity constraints would be applied.

$$\left.\frac{\partial^2(-f(X,p+\lambda p))}{\partial\lambda^2}\right|_{\lambda=0} \geq 0$$

The resulting LMI would have linear dependence on p, and thus the multiconvexity relaxation process could be stopped.

C.1 Solution Case 1

Noting that $A(p) = A_0 + pA_1$, $R(p) = R_0 + pR_1$, and $S(p) = S_0 + pS_1$, the LMIs that result after application of multiconvexity relaxation are:

$$\begin{aligned} &6pV_1^T A_1 R_1 A_1^T V_1 + 2\left[V_1^T A_1 R_1 A_0^* V_1 + V_1^T A_1 R_1 C_1^T V_2 + V_1^T A_1 R_0 A_1^T V_1 \ldots\right. \\ &\left. + V_1^T A_0 R_1 A_1^T V_1 + V_2^T C_1 R_1 A_1^T V_1 + \nu_1 I\right] \succeq 0 \end{aligned}$$

$$\begin{aligned} &6pU_1^T A_1^T S_1 A_1 U_1 + 2\left[U_1^T A_1^T S_1 A_0 U_1 + U_1^T A_1^T S_1 B_1 U_2 + U_1^T A_1^T S_0 A_1 U_1 \ldots\right. \\ &\left. + U_1^T A_0^T S_1 A_1 U_1 + U_2^T B_1^T S_1 A_1 U_1 + \nu_1 I\right] + 2dpU_1^T A_1^T S_1 A_1 U_1 \succeq 0 \end{aligned}$$

which must hold for all $(p, dp) \in \operatorname{vert} \Delta_\Gamma$.

C.2 Solution Case 2

Noting that $A(p) = A_0 + pA_1$, $R(p) = R_0 + pR_1 + p^2 R_2$, and $S(p) = S_0 + pS_1 + p^2 S_2$, the LMIs that result after application of multiconvexity relaxation are:

$$12p^2\left[V_1^TA_1R_2A_1^TV_1\right]+6p\left[V_1^TA_1R_2A_0^TV_1+V_1^TA_1R_2C_1^TV_2+V_1^TA_1R_1A_1^TV_1\ldots\right.$$
$$\left.+V_1^TA_0R_2A_1^TV_1+V_2^TC_1R_2A_1^TV_1\right]+2\left[V_1^TA_0R_2A_0^TV_1+V_1^TA_0R_2C_1^TV_2+V_2^TC_1R_2A_0^TV_1\ldots\right.$$
$$\left.+V_2^TC_1R_2C_1^TV_2+V_1^TA_1R_1A_0^TV_1+V_1^TA_1R_1C_1^TV_2+V_1^TA_1R_0A_1^TV_1+V_1^TA_0R_1A_1^TV_1\ldots\right.$$
$$\left.+V_2^TC_1R_1A_1^TV_1-V_1^TR_2V_1+\nu_1I\right]\succeq 0$$

$$2\left[-V_1^TR_2V_1+\nu_2I\right]\succeq 0$$

$$-24V_1^TA_1R_2A_1^TV_1\succeq 0$$

$$12p^2U_1^TA_1^TS_2A_1U_1+12pdpU_1^TA_1^TS_2A_1U_1+6p\left[U_1^TA_1^TS_2A_0U_1+U_1^TA_1^TS_2B_1U_2+\ldots\right.$$
$$\left.+U_1^TA_1^TS_1A_1U_1+U_1^TA_0^TS_2A_1U_1+U_2^TB_1^TS_2A_1U_1\right]+2dp\left[2U_1^TA_1^TS_2A_0U_1+2U_1^TA_1^TS_2B_1U_2\ldots\right.$$
$$\left.+2U_1^TA_0^TS_2A_1U_1+2U_2^TB_1^TS_2A_1U_1+U_1^TA_1^TS_1A_1U_1\right]+2\left[U_1^TA_0^TS_2A_0U_1+U_1^TA_0^TS_2B_1U_2\ldots\right.$$
$$\left.+U_2^TB_1^TS_2A_0U_1+U_2^TB_1^TS_2B_1U_2+U_1^TA_1^TS_0A_1U_1+U_1^TA_1^TS_1A_0U_1+U_1^TA_1^TS_1B_1U_2\ldots\right.$$
$$\left.+U_1^TA_0^TS_1A_1U_1+U_2^TB_1^TS_1A_1U_1-U_1^TS_2U_1+\eta_1I\right]\succeq 0$$

$$-24U_1^TA_1^TS_2A_1U_1\succeq 0$$

$$2p\left[U_1^TA_0^TS_2A_0U_1+U_1^TA_1^TS_2B_1U_2+U_1^TA_0^TS_2A_1U_1+U_2^TB_1^TS_2A_1U_1\right]\succeq 0$$

$$2\left[U_1^TA_0^TS_2A_0U_1+U_1^TA_0^TS_2B_1U_2+U_2^TB_1^TS_2A_0U_1+U_2^TB_1^TS_2B_1U_2+\eta_2I\right]\succeq 0$$

$$-\begin{bmatrix}R_2 & 0\\ 0 & S_2\end{bmatrix}\succeq 0$$

which must hold for all $(p, dp) \in \operatorname{vert} \Delta_\Gamma$.