NONLINEAR UAV FLIGHT CONTROL USING
COMMAND FILTERED BACKSTEPPING

A Thesis
Presented to
the Faculty of California Polytechnic State University
San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Aerospace Engineering

by
Brian Borra
March 2012
COMMITTEE MEMBERSHIP

TITLE: Nonlinear UAV Flight Control
Using Command Filtered Backstepping

AUTHOR: Brian Borra

DATE SUBMITTED: March 2012

COMMITTEE CHAIR: Eric Mehiel, Ph.D.

COMMITTEE MEMBER: Robert Miller, Ph.D.

COMMITTEE MEMBER: Al Jiminéz, Ph.D.

COMMITTEE MEMBER: Daniel Biezad, Ph.D.
Abstract

Nonlinear UAV Flight Control Using Command Filtered Backstepping

Brian Borra

The aim of this effort is to implement a nonlinear flight control architecture, specifically flight path control via command filtered backstepping, for use in AME UAS’s Fury® 1500 unmanned flying wing aircraft. Backstepping is a recursive, control-effort minimizing, constructive design procedure that interlaces the choice of a Lyapunov function with the design of feedback control. It allows the use of certain plant states to act as intermediate, virtual controls, for others breaking complex high order systems into a sequence of simpler lower-order design tasks.

Work herein is a simplified implementation based on publications by Farrell, Sharma, and Polycarpou. Online approximation is not applied, however command filtering along with two variants of control allocation is. This minimal approach was done to mitigate risk, as adaptation could be added in future work to this baseline. Command filtering assures that control inputs generated meet magnitude, rate, and bandwidth constraints for states and actuators as well as provides command derivatives that reduce computation. Two different forms of control allocation were implemented, the simplest a least-squares pseudo-inverse and the second an optimal quadratic programming method.

Two Simulink based simulations successfully flew AME’s Fury® 1500 UAS: a nominal case with fully operational actuators and a failure case with an actuator stuck at $-10^\circ$. Coordinated flight for both cases with outer loop tracking was achieved for a demanding autopilot task of simultaneously varying heading and flight-path angle commands, $\pm 60^\circ$ and $\pm 10^\circ$ respectively, for a constant airspeed command of 135 ft/s. Command signals were generated were achievable due to the command filter implementation.
Acknowledgements

I’d like to thank my parents first and foremost, for without their support I wouldn’t be who and where I am today. Thank you mom and dad!

My committee chair, Dr. Eric Mehiel, helped me most throughout the process with invaluable observations and always made time for discussion. Dr. Al Jiminéz and Dr. Daniel Biezad have been excellent resources throughout my academic career and I have much gratitude for the lessons I’ve learned through their courses. I’d also like to thank AME staff Dr. Robert Miller, Nick Brake, Alexander Bogdanov, and David Grosher who offered the topic suggestion and support.
# Table of Contents

List of Tables ix  
List of Figures x  
Nomenclature xiii  

1 Introduction  

1.1 Significance ........................................... 4  
1.2 Requirements and Outcomes ............................. 4  
1.3 Thesis Outline .......................................... 5  

2 Nonlinear Theory  

2.1 Modeling ............................................ 6  

2.1.1 Reference Frames .................................... 9  

2.1.2 Direction Cosine Matrices .......................... 12  

2.1.3 Aircraft Dynamics .................................... 16  

2.1.3.1 Translational Acceleration ....................... 17  

2.1.3.2 Rotational Acceleration .......................... 20  

2.1.3.3 Attitude Rates .................................... 21  

2.1.3.4 Equations of Motion .............................. 23  

2.1.4 System Observations .................................. 24  

2.2 Stability ............................................. 26  

2.2.1 Equilibrium and Operating Points .................. 28  

2.2.2 Lyapunov Stability Definitions ..................... 30  

2.2.2.1 Perturbed Systems ............................... 36  

2.2.3 Lyapunov Stability Theorems ......................... 37  

2.3 Control ............................................... 46  

2.3.1 Requirements ...................................... 46  

2.3.2 Objective and Methods .............................. 48  

2.3.3 Lyapunov Based Control Design ..................... 50
3 Derivation of UAV Flight Path Controller

3.1 Outer Loop

3.1.1 Flight Path Angle Control

3.1.1.1 Selection of $\alpha^c$ and $\mu^c$ Commands

3.1.2 Airspeed Control

3.2 Middle Loop

3.2.1 Wind-Axis Angle Control

3.3 Inner Loop

3.3.1 Body-Axis Angular Rate Control

3.4 Stability Proof

4 Application

4.1 Simulation Modeling

4.1.1 Fury 1500 UAS

4.1.2 Simulink Models

4.2 Simulation Analysis and Results

4.2.1 Nominal – PINV

4.2.2 Effector Failure – QCAT

5 Conclusion

Appendices

A Primary Subsystems Breakdown
A.1.2 Solve \( T_e^o \) ................................. 129
A.1.3 Solve \( P_e^o, Q_e^o, \) and \( R_e^o \) .......................... 130
A.1.4 Solve \( \delta_e^o \), Figure 2.24 ............................ 130
A.2 Command Filters (CF), Figure 4.4 .......................... 131
  A.2.1 Flight Path Angle & Airspeed Filters .......................... 131
  A.2.2 Wind Axis Angle Filters ................................. 131
  A.2.3 Body Axis Rate Filters ................................. 132
  A.2.4 Simple Deflection Filters ............................... 132
  A.2.5 Deflection Filters ...................................... 133
A.3 Virtual Control Laws (VCL), Figure 4.5 ...................... 134
  A.3.1 Loop Interactions ................................. 134
A.4 Vehicle Model & Dynamics (FURY), Figure 4.6 .............. 134
  A.4.1 System Dynamics ...................................... 135
    A.4.1.1 Flight Path Variables ............................. 136
    A.4.1.2 Body Axis Angular Rates .......................... 136
    A.4.1.3 Airspeed ......................................... 137
    A.4.1.4 Wind Axis Angles .................................. 137
    A.4.1.5 Control Laws ...................................... 138
    A.4.1.6 Filtered Control Laws ............................. 139
A.5 Scopes .............................................. 140
  A.5.1 Euler Scope Transformations ............................. 141
List of Tables

2.1 Choices for State Variables .................................................. 8
2.2 Direction Cosine Matrices for Plane Rotations ....................... 14
2.3 Stability Theorem Overview ................................................... 37
2.4 Nonlinear Control Method Overview ...................................... 49
2.5 Backstepping Key Terms ..................................................... 52

4.1 Simulation Tracking Error and Command Filter Gains ............. 116
List of Figures

1.1 Fury 1500 UAS on Launcher at Twilight [http://www.prnewswire.com/news-releases/] ................................. 2

2.1 Air Vehicle Reference Frames ................................................................. 10
2.2 Axis Relationships: Body, Stability, and Wind Axes ................................. 10
2.3 Earth Reference Frames ........................................................................... 12
2.4 Direction Cosine Matrix Visual ............................................................... 15
2.5 Body Axes, Forces, and Moments ............................................................. 18
2.6 Orientation of Gravity Vector with Respect to Body Axes [1] .................... 18
2.7 Fury 1500 UAS on Launcher [www.aviationweek.com/aw/blogsmain/] ........ 22
2.8 Operating Surface via Phase Portraits ....................................................... 29
2.9 First Order Unstable Systems Khalil [2, Online Lecture Notes] ................. 33
2.10 First Order Stable Systems Khalil [2, Online Lecture Notes] .................. 33
2.11 First Order Asymptotically Stable Systems Khalil [2, Online Lecture Notes] 34
2.12 Stability Concepts ................................................................................... 34
2.13 Equilibrium Point Classification for 2\textsuperscript{nd} Order Linear Systems [3] 35
2.15 Asymptotic Property 1: $\dot{f}(t) \to 0 \not\Rightarrow f \to \text{constant}$ .................. 44
2.16 Asymptotic Property 2: $f \to \text{constant} \not\Rightarrow \dot{f}(t) \to 0$ ....................... 44
2.17 Backstepping Block Diagram Representation: \textit{a)} Integral augmented cascade system \textit{b)} introducing $\pm \alpha(x)$ \textit{c)} “backstepping” $-\alpha(x)$ through the integrator 57
2.18 Composite Phase Portrait for System 2.3.25 with $u = 0$ ......................... 61
2.19 Equation 2.3.72 Low Pass Filter ........................................................... 72
2.20 Command Filter [5] ................................................................................. 73
2.21 Command Filter [6] ................................................................................. 76
2.22 Moment Control Derivative Dimensionalization & Control Effectiveness Matrix Concatenation in Simulink ................................................................. 77
2.23 Moment Control Derivative Look-up Tables for Control Effectiveness Matrix in Simulink ........................................... 78
2.24 Control Allocation Simulink Subsystem ......................................................... 79
2.25 PINV Simulink Subsystem within CA Block Fig 2.24 .......................... 80
2.26 QCAT Simulink Subsystem within CA Block Fig 2.24 ......................... 80

3.1 High Level Control Architecture Overview .............................................. 81
3.2 Low Level, Block Vector, Control Architecture Overview [5] .................... 82
3.3 Command Filtered Adaptive Backstepping, Sonneveldt et al. [7] ............. 83
3.4 Command Filtered Backstepping Simulink Model ....................................... 84
3.5 Two possible choices for $\mu^c_\alpha$ and $\alpha^c_\alpha$ to solve $(\chi, \gamma)$ control. [5] ......................................................... 90

4.2 Fury 1500 UAS - Specifications / Top-View [www.ameuas.com/uas_fury.html] 109
3.4 Command Filtered Backstepping Simulink Model ....................................... 111
4.3 Control Allocation (CA) Primary Subsystem .............................................. 112
4.4 Command Filter (CF) Primary Subsystem .................................................. 113
4.5 Virtual Control Law (VCL) Primary Subsystem ......................................... 114
4.6 Vehicle Model & System Dynamics (FURY) Primary Subsystem ................ 114
4.7 $\zeta_\chi$ Filter (LPF) Primary Subsystem ...................................................... 114
4.8 Desired Tracking Commands, Dissected from CA Primary Subsystem Block in Figure 4.3 ........................................................... 115
4.9 Nominal – Forces and Moments ................................................................. 117
4.10 Nominal – Body Axis Velocities and Angles ............................................. 117
4.11 Nominal – States ..................................................................................... 118
4.12 Nominal – Deflections ............................................................................ 118
4.13 Nominal – Outer Loop Tracking .............................................................. 119
4.14 Nominal – Middle Loop Tracking ............................................................ 119
4.15 Nominal – Inner Loop Tracking ............................................................... 120
4.16 Nominal – Flightpath Angle Pseudo-Controls, see Sec. 3.1.1.1 ................ 120
4.17 Nominal – Tracking Errors ................................................................. 121
4.18 Failure – Forces and Moments ............................................................. 122
4.19 Failure – Body Axis Velocities and Angles ............................................. 122
Nomenclature

\( \alpha \)  
Angle of Attack  
\( \alpha(x) \)  
Stabilizing Feedback Control Law  
\( \beta \)  
Angle of Sideslip  
\( \gamma \)  
Flight-Path Angle  
\( \zeta_x \)  
Filtered State  
\( \zeta \)  
Damping Frequency  
\( \mu \)  
Bank Angle  
\( \xi \)  
Virtual Control Law  
\( \phi, \theta, \psi \)  
Roll, Pitch, and Yaw Angles  
\( \chi \)  
Ground-Track Angle  
\( \omega_n \)  
Natural Frequency  
\( C \)  
Direction Cosine Matrix  
\( D \)  
Drag  
\( E \)  
Control Effectiveness Matrix  
\( F \)  
Known Sub-function  
\( I \)  
Moment or Product of Inertia
\(L\)  
Lift \(\text{lb}\)

\(V\)  
Airspeed \(\text{ft/s}\)

\(W\)  
Positive (Semi-)Definite Function 
BS

\(X\)  
X-Coordinate for \((\alpha^o_c, \mu^o_c)\) Solution 
BS

\(Y\)  
Sideforce \(\text{lb}\)

\(Y\)  
Y-Coordinate for \((\alpha^o_c, \mu^o_c)\) Solution 
BS

\(\bar{L}, \bar{M}, \bar{N}\)  
Roll, Pitch, and Yaw Moments 
\(\text{lb} \cdot \text{ft}\)

\(F\)  
Reference Frame; Force

\(f\)  
Nominal System 
DE

\(f\)  
Unknown or Partially Known Sub-function 
BS

\(g\)  
Perturbed System 
DE

\(m\)  
Mass 
slug

\(u\)  
Control Law

\(u, v, w\)  
Longitudinal, Lateral, and Normal Velocities 
\(\text{ft/s}\)

\(v\)  
Virtual Control Input Vector 
\(1/\text{s}^2\)

\(z\)  
Tracking Error 
BS

**Acronyms**

(O)DE  
(Ordinary) Differential Equation

AS  
Asymptotically Stable

BS  
Backstepping

xiv
CA  Control Allocation
CF  Command Filter
CFBS Command Filtered Backstepping
CLF  Control Lyapunov Function
CONV  Converges
ECEF  Earth Centered Earth Fixed
ECI  Earth Centered Inertial
GAS  Globally Asymptotically Stable
GUAS  Globally Uniformly Asymptotically Stable
LPF  Low Pass Filter
PINV  Pseudo Inverse
QCAT  Quadratic Programming Control Allocation Toolbox
S  Stable
UAS  Unmanned Aerial System
UAV  Unmanned Aerial Vehicle
VCL  Virtual Control Laws

Subscript

A  Aerodynamic
c  Commanded
des  Desired
$G$  Gravitational

$M$  Magnitude

$R$  Rate

$s$  Stability Axes

$T$  Thrust

**Superscript**

$-$  Compensated Tracking Error

$\sim$  Tracking Error

$o$  Unfiltered
Chapter 1

Introduction

In all control techniques the goal is the same: to achieve a desired response even in the presence of external disturbances and sensor noise. It is the control engineer’s responsibility to choose the appropriate design technique given the stability, performance, and cost requirements of the system; these requirements are dictated by vehicle requirements which are derived from a set of mission requirements. As technology expands so does its complexity and the level of detail placed into the system model; typically plants with nonlinearities and unknown or changing parameters are associated with this higher fidelity. These advances in control theory led to the development of adaptive control, the goal of which is to control plants with unknown or imperfect knowledge.

Backstepping has applications in a broad spectrum of engineering disciplines, from electrical motors \[8\] to jet engines \[9\], ship control \[10\] to flight control \[11\] just to name a few. It offers a systematic methodology for developing control schemes for nonlinear systems. The focus of this thesis is to implement a six degree of freedom, nonlinear, Command Filtered Backstepping (CFBS) flight controller for use in an unmanned aerial system (UAS). Specifically, flight path command tracking of ground-track, flight-path (climb rate), and airspeed \([\chi_c, \gamma_c, V_c]\) as generated by an external mission planner for AME UAS’s Fury® 1500 long-endurance, survivable UAS\(^1\).

---

\(^1\) Fury® Unmanned Aerial System, \url{http://www.chandlermay.com/uas_fury.html} (March 2012)
Before arriving at this level of implementation, backstepping will be introduced in its simplest possible form. Incremental development from this foundation will eventually lead to the desired flight path controller. Backstepping is a relatively new control algorithm for nonlinear systems that utilizes Lyapunov synthesis to derive a stabilizing controller. It “is a recursive procedure that interlaces the choice of a Lyapunov function with the design of feedback control. It breaks a design problem for the full system into a sequence of design problems for lower-order (even scalar) systems. By exploiting the extra flexibility that exists with lower order and scalar systems, backstepping can often solve stabilization, tracking, and robust control problems under conditions less restrictive than those encountered in other methods.” Khalil [2]

The appeal of a nonlinear flight control method is that it offers an increase in performance and potential reduction in development time. Nonlinear effects in aerodynamics and cross-
coupling between longitudinal and lateral motion may be modeled. Also, contrary to ones initial thoughts, there are potential time savings in analysis: linear systems are reduced to such around a particular operating point, and to completely guarantee stability all points over the entire flight envelope need to be analyzed. Nonlinear systems on the other hand, encompass a set of operating points inherently therefore less analysis is required. The price to pay is a lack of standardized evaluation tools in contrast to those available to linear, or classic, control theory.

Adaptive methods offer management of uncertain dynamics hence less precise model knowledge is required. Advanced control allocation techniques offer protection against mechanical failures or battle damage hence the system gains survivability. The weakness of backstepping is that it lacks support for dealing with actuator redundancy; control laws generate desired rates, but do not specify how much and in what combination to deflect surfaces to achieve them. Particular flavors of control allocation allow for the air vehicle to be controlled under situations where an actuator fails or is damaged, as will be shown. Furthermore, additional benefits of the control architecture include a guarantee that control inputs are implementable due to command filtering, less control effort as useful nonlinearities are retained in backstepping, and system stability is provable using Lyapunov synthesis.

The goal of this work is to attain a professional proficiency in the topic. In addition to this writing, the deliverable will be a Simulink based backstepping control architecture with command filtering and control allocation. Simulation development, in Chp. 3, will be thoroughly supported with Lyapunov control law synthesis including stability proof. Simulation analysis will be based on work by Farrell et al. [6], Backstepping-Based Flight Control with Adaptive Function Approximation.
1.1 Significance

Adaptive flight control has been viewed as an enabling technology to deal with plants with highly nonlinear, time-varying, and/or uncertain dynamical characteristics. As unmanned aerial systems become more interactive with humans, providing safety for flight crews, it must be assured that they offer a level of reliability comparable to manned systems; this project strives to accomplish this goal, hence fostering the advancement of autonomous systems and user safety.

This work stands out from referenced backstepping journal papers in that it is very thorough, ie. full of supporting theory and introduces backstepping by gradually increasing complexity. It also is more comprehensible given control architecture development/implementation through block diagrams with Simulink®; it’s at least a different way to present the problem, thereby providing the reader with an alternate means of learning. This graphical representation of underlying mathematics offers an intuitive visualization of the procedure and illustrates signal interdependencies that are not self-evident through pages and pages of equations.

1.2 Requirements and Outcomes

International Traffic in Arms Regulations (ITAR) requires that information and material pertaining to defense and military related technologies may only be shared with US persons. The proprietary, ITAR regulated AME UAS Fury® 1500 vehicle model and associated table data will not be publicly available for security, ownership, and legal reasons. All images, specifications, and verbiage used to present Fury® herein was found solely from online sources and cited immediately after presentation; it is therefore data available to the public domain.

Proof of concept will be demonstrated by the tracking of varying ground-track and
flight-path angle commands for a constant airspeed, ie. a coordinated turns. Additionally, an actuator will assume a failure in order to demonstrate the robustness of the controller paired with an advanced control allocation technique.

1.3 Thesis Outline

A short review of aircraft dynamics and supporting concepts in Sec. 2.1 Modeling essential to modeling and simulation. Next, preliminaries of and results for Lyapunov stability theory will be stated in Sec. 2.2 Stability. Following this will be a section on Control, Sec. 2.3 Control, where core backstepping theories will be derived along with brief overviews of command filtering in Sec. 2.3.5 and control allocation in Sec. 2.3.6. Three backstepping flavors are to be covered: a simple second order nonlinear system, a generic higher order nonlinear system, and another second order nonlinear system with command filtering. Each highlight a particular benefit of backstepping, however repetitively demonstrate the steps required to manipulate the plant and develop control laws. A full state feedback, six degree of freedom, nonlinear flight path control system will be derived for a UAV in Chp. 3. In closing, simulation block diagrams and tracking results will be presented and discussed in Chp. 4 Application.
Chapter 2

Nonlinear Theory

“The art of flight control design is to realize a solution that achieves an acceptable compromise among the evaluation criteria: [Stability, Performance, and Flying Qualities].”

Honeywell Inc. and Lockheed Martin Skunk Works [12]

2.1 Modeling

This section will introduce fundamental aircraft dynamics concepts and ultimately derive the equations of motion implemented in the backstepping control architecture. The aim is to establish a practical understanding of the equations of motion, reinforced by pure mathematical formulation\(^1\). The fallout allows the designer to quantitatively evaluate the characteristic modes of motion, thereby providing an analytical playground for control design and performance evaluation via handling qualities\(^2\). Assumptions, hence consequent limitations of the derived equations, will be clearly identified with discussion immediately succeeding. No stone will be left unturned; equations will be derived from Newton’s first principals and include necessary supporting concepts. The philosophy of this approach and procedure itself is credited to work by [13] and [1].

Establishing a way to mathematically describe the vehicle’s dynamics is a necessity for any flight control architecture. As with any dynamic system, a set of differential equations may

---

\(^1\) For physical illustrations of key aspects to the derivation refer to McRuer et al. [1, Chp. 4]  
\(^2\) Handling qualities for unmanned aerial vehicles is a topic in its infancy as standards for which are still being developed, the necessity of which is debatable. The degree of instability depends on what autopilot can handle.
be used to calculate an object’s position, velocity, and acceleration. Typically for complex systems – such as an airplane with flexible structure, rotating internal components, and variable mass – simplifying assumptions are applied in order to use Newton’s Laws to derive vehicle dynamics. These assumptions lead to a direct appreciation of important factors that govern the vehicle dynamics. This level of understanding is an “implicit requirement for effective and efficient flight control system design activities. It affords a basic understanding of the vehicle/control system interactions and of the flight controller possibilities which are most likely to succeed.” McRuer et al. [1]

**Assumption 2.1.** Airframe is a rigid body.

“Rigid body models are described by six degrees of freedom and include forces and moments caused by gravity, aerodynamics, and propulsion.” [12] The distances between any two points are fixed, hence forces acting between those points due to elastic deformation are absent. Consequently, the air vehicle may be modeled as an individual element of mass. In reality air vehicles diverge from the rigid body assumption in two ways: aeroelastic phenomena due to airframe structure deformation (such as wing bending due to air loads) and relative motion of components (engine, propeller, and control surfaces).

Under this assumption, the equations of motion may be decoupled into translational and rotational equations if the coordinate origin is chosen to coincide with the center of gravity. Two possible system descriptions are described in Table 2.1. It implies that there are 9 equations necessary for control, list items 1–3, and 3 for navigation, list item 4. The implementation herein is concerned with control, therefore the **state-vector** \(^3\) of this controller will consist of the first 9 variables.

---

\(^3\) The minimal set of system variables necessary to indicate the energy of the system, potential and kinetic, and its distribution at any given time.
<table>
<thead>
<tr>
<th>$x_1$</th>
<th>Translational</th>
<th>$x_2$</th>
<th>Attitudes</th>
<th>$x_3$</th>
<th>Rotational</th>
<th>$x_4$</th>
<th>Navigation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Longitudinal Vel.</td>
<td></td>
<td>Euler Roll</td>
<td>Roll Rate</td>
<td></td>
<td>North Position</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$u$</td>
<td></td>
<td>$\varphi$</td>
<td>$P$</td>
<td></td>
<td>$\xi$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lateral Velocity</td>
<td></td>
<td>Euler Pitch</td>
<td>Pitch Rate</td>
<td></td>
<td>East Position</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$v$</td>
<td></td>
<td>$\theta$</td>
<td>$Q$</td>
<td></td>
<td>$\eta$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Normal Velocity</td>
<td></td>
<td>Euler Yaw</td>
<td>Yaw Rate</td>
<td></td>
<td>Altitude</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$w$</td>
<td></td>
<td>$\psi$</td>
<td>$R$</td>
<td></td>
<td>$h$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Velocity</td>
<td></td>
<td>Bank Angle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V$</td>
<td></td>
<td>Angle of Attack</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Velocity</td>
<td></td>
<td>Sideslip Angle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V$</td>
<td></td>
<td>$\beta$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Body</th>
<th>Flight Path / Wind</th>
</tr>
</thead>
<tbody>
<tr>
<td>- 3 components of attitude to specify orientation relative to the</td>
<td>- 3 components of</td>
</tr>
<tr>
<td>gravity vector</td>
<td>attitude to specify</td>
</tr>
<tr>
<td>- 3 components of velocity to specify translational kinetic energy</td>
<td>orientation relative</td>
</tr>
<tr>
<td>- 3 components of angular velocity to specify rotational kinetic</td>
<td>to the velocity vector</td>
</tr>
<tr>
<td>energy</td>
<td>components of velocity</td>
</tr>
<tr>
<td>- 3 components of position to specify potential energy in earth’s</td>
<td>components of velocity</td>
</tr>
<tr>
<td>gravity field</td>
<td>direction to specify</td>
</tr>
<tr>
<td></td>
<td>translational kinetic</td>
</tr>
<tr>
<td></td>
<td>energy</td>
</tr>
<tr>
<td></td>
<td>- 3 components of angular velocity to specify rotational kinetic</td>
</tr>
<tr>
<td></td>
<td>energy</td>
</tr>
<tr>
<td></td>
<td>- 3 components of position to specify potential energy in earth’s</td>
</tr>
<tr>
<td></td>
<td>gravity field</td>
</tr>
</tbody>
</table>

Again the flight path controller herein will only be using the first three subsets of Table 2.1:
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix} = \begin{bmatrix}
  (\chi \gamma V)^T \\
  (\mu \alpha \beta)^T \\
  (P Q R)^T
\end{bmatrix}
\]

**2.1.1 Reference Frames**

Just as language is needed to express thoughts, a reference frame is necessary to convey motion. The relationship between an object and the space it resides in is relative; choosing a reference frame, or coordinate system, enables an observer to describe the motion of an object over time. Selecting an appropriate reference frame can greatly simplify the description of this relationship.

**Assumption 2.2.** Earth is an inertial reference frame.

When earth is considered as an inertial frame of reference, one that is fixed or moving at a constant velocity (non-rotating and non-accelerating) relative to earth, it permits accurate short-term control and navigation analysis. Conversely, an inertial frame of reference is unacceptable for air vehicles that require long-term navigation, especially for high-speed flight, or extra-atmospheric operation; for most UAVs this assumption is fairly accurate however. As this situation dictates, there are numerous reference systems in aerospace applications. The frames applicable to the equations of motion derivation herein are: body, stability, wind or flight-path, and earth-centered-inertial or earth-fixed. Additionally, north-east-down or local-tangent-plane, vehicle-carried-vertical, and earth-centered-earth-fixed frames will be covered. All coordinate systems follow the right hand rule and are orthogonal.
Body, stability, and wind axes are attached to the airframe at the center of gravity as depicted in Figure 2.1. By convention, body axis $x_b$ points out the nose, $y_b$ out the right wing, and $z_b$ down the bottom of the aircraft. Stability axes are defined by a rotation of the body axes in the $x_b$-$z_b$ plane by an angle-of-attack, $\alpha$, that trims the air vehicle, i.e. zero pitching moment; axis $x_s$ points into the direction of steady flow, $y_s = y_b$, and $z_s$ is perpendicular to the $x_s$-$y_s$ plane in the direction following the right handed sign convention. In wind, or flight-path, axes the $x_w$ axis always points into the relative wind. This is defined by a rotation of the body axes through angle-of-attack and sideslip angle with $z_w = z_s$ and $y_w$ following the right hand rule as shown in Figure 2.2:
The body state variables can be derived from the flight-path state variables as shown in Honeywell Inc. and Lockheed Martin Skunk Works [12], recall Table 2.1:

\[ u = V \cos \beta \cos \alpha \]  \hspace{1cm} (2.1.2)

\[ v = V \sin \beta \]  \hspace{1cm} (2.1.3)

\[ w = V \cos \beta \sin \alpha \]  \hspace{1cm} (2.1.4)

\[ \phi = \tan^{-1} \left( \frac{\cos \gamma \sin \mu \cos \beta - \sin \gamma \sin \beta}{- \cos \gamma \sin \mu \sin \alpha \sin \beta + \cos \gamma \cos \alpha \cos \mu - \sin \gamma \sin \alpha \cos \beta} \right) \]  \hspace{1cm} (2.1.5)

\[ \theta = \sin^{-1} (\cos \gamma \sin \mu \cos \alpha \sin \beta + \cos \gamma \cos \mu \sin \alpha + \sin \gamma \cos \alpha \cos \beta) \]  \hspace{1cm} (2.1.6)

\[ \psi = \tan^{-1} \left\{ \frac{(\sin \mu \sin \alpha - \cos \alpha \cos \mu \sin \beta) \cos \chi + [\cos \gamma \cos \alpha \cos \beta - \sin \gamma (\sin \alpha \cos \mu + \sin \beta \cos \alpha \sin \mu)] \sin \chi}{-(\sin \mu \sin \alpha - \cos \alpha \cos \mu \sin \beta) \sin \chi + [\cos \gamma \cos \alpha \cos \beta - \sin \gamma (\sin \alpha \cos \mu + \sin \beta \cos \alpha \sin \mu)] \cos \chi} \right\} \]  \hspace{1cm} (2.1.7)

Flight-path variables may be derived from the body state variables as follows, again refer to [12] and recall Table 2.1:

\[ V = \sqrt{u^2 + v^2 + w^2} \]  \hspace{1cm} (2.1.8)

\[ \alpha = \tan^{-1} \left( \frac{w}{u} \right) \]  \hspace{1cm} (2.1.9)

\[ \beta = \sin^{-1} \left( \frac{v}{\sqrt{u^2 + v^2 + w^2}} \right) \]  \hspace{1cm} (2.1.10)

\[ \mu = \tan^{-1} \left[ \frac{uv \sin \theta + (u^2 + w^2) \sin \phi \cos \theta - vw \cos \phi \cos \theta}{\sqrt{u^2 + v^2 + w^2} \left( w \sin \theta + u \cos \phi \cos \theta \right)} \right] \]  \hspace{1cm} (2.1.11)

\[ \gamma = \sin^{-1} \left( \frac{u \sin \theta - v \sin \phi \cos \theta - w \cos \phi \cos \theta}{\sqrt{u^2 + v^2 + w^2}} \right) \]  \hspace{1cm} (2.1.12)

\[ \chi = \tan^{-1} \left[ \frac{u \cos \theta \sin \psi + v \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi}{u \cos \theta \cos \psi + v \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi} \right] \]  \hspace{1cm} (2.1.13)

North East Down (NED), also known as Local Tangent Plane (LTP), is positioned on the surface of earth with it’s origin vertically aligned to the aircraft’s center of gravity. North is parallel to lines of longitude (\(\lambda\)), east is parallel to lines of latitude (\(\phi\)), and down completes the right hand rule pointing into earth. Vehicle Carried Vertical (VCV) shares the NED orientation definition, with the exception of a shift in origin from Earth’s surface to the
vehicle’s center of gravity, as the name suggests.

![Earth Centered Inertial (Fixed)](image1)

![Earth Centered Earth Fixed (Rotates)](image2)

**Figure 2.3: Earth Reference Frames**

The Earth Centered Inertial (ECI) frame is considered *fixed* in space with its origin at the center of Earth; it does not rotate with Earth and is oriented to suit the situation. Typically the $z_{ECI}$ axis is aligned along Earth’s spin axis pointing toward the North Pole. Consult Stevens and Lewis [14, Pg. 20], for alternative ECI orientations.

Earth Centered Earth Fixed (ECEF) is a non-inertial frame that rotates with earth. This reference system aligns $x_{ECEF}$ to the intersection of the zero-longitude prime meridian and zero-latitude equator. $y$, lies in the equatorial plane and $z_{ECEF}$ points toward the Earth’s North Pole. Note how the radius endpoint is not coincident with the center of the ellipsoid; this is because the radius emanates from a plane tangent to the ellipsoidal surface.

### 2.1.2 Direction Cosine Matrices

If reference frames were languages, direction cosine matrices would be interpreters. It allows a vector’s orientation to be expressed as components among relative coordinate systems. As the name suggests, rotations are achieved by defining a matrix of direction cosines that relate *unit vectors* in one axis system to those in another, preserving the length of the rotated
vector. The determination of matrix elements may be accomplished by inspection; McRuer et al. [1] and Stevens and Lewis [15] note several general properties for construction of these matrices:

- “The one is always associated with the axis about which rotation occurs.”
- “The remaining elements in the row and column containing the one are all zeros.”
- The remaining main diagonal terms are the cosine of the angle of rotation.
- The remaining matrix elements contain the sine of the angle of rotation and are always symmetrically placed relative to the cosine terms; this is done so that zero rotation produces an identity matrix.
- “In the direct right-handed rotation the negative sign always appears in the row above the one (this is to be interpreted as the third row if the one is in the first).”
- “Changing the sign of the rotation angle yields the matrix transpose.”

A coordinate rotation example from the body axis frame, $F_B$, to north-east-down frame, $F_{NED}$, is illustrated by three plane rotations in Table 2.2.

As an example, the array $C_\Phi$ from Table 2.2 may be read either left to right or down as $y_\Phi = x_B 0 + y_B \cos \Phi + z_B \sin \Phi$ and $y_B = x_\Phi 0 + y_\Phi \cos \Phi - z_\Phi \sin \Phi$ respectively. The transpose of $C_\Phi$, ie. $C_\Phi^T$ allows us to get to $x_\Phi, y_\Phi, z_\Phi$ from $x, y, z$, to be proven later. Doing the left to right read for the remaining rows and corresponding columns leads to a convenient matrix formulation of these equations:

$$
\begin{bmatrix}
x_\Phi \\
y_\Phi \\
z_\Phi
\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 \\0 & \cos \Phi & \sin \Phi \\0 & -\sin \Phi & \cos \Phi
\end{bmatrix} \begin{bmatrix}x_B \\
y_B \\
z_B
\end{bmatrix} \iff F_\Phi = C_\Phi F_B \tag{2.1.14}
$$

In this formulation $C_\Phi$ gets us to $x_B, y_B, z_B$ from $x_\Phi, y_\Phi, z_\Phi$. A variety of notations exist for direction cosine matrices, Stevens would write $C_{F_{\Phi}/F_B}$ instead of $C_\Phi$ which explicitly expresses the coordinate frame transformation in the subscript, ie. from $F_B$ to $F_{\Phi}$. Less trivial than notation are the properties which these rotation matrices possess:
Table 2.2: Direction Cosine Matrices for Plane Rotations

<table>
<thead>
<tr>
<th>$C_{\Phi}$</th>
<th>$x_B$</th>
<th>$y_B$</th>
<th>$z_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{\Phi}$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$y_{\Phi}$</td>
<td>0</td>
<td>$\cos \Phi$</td>
<td>$\sin \Phi$</td>
</tr>
<tr>
<td>$z_{\Phi}$</td>
<td>0</td>
<td>$-\sin \Phi$</td>
<td>$\cos \Phi$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C_{\Theta}$</th>
<th>$x_{\Phi}$</th>
<th>$y_{\Phi}$</th>
<th>$z_{\Phi}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{\Theta}$</td>
<td>$\cos \Theta$</td>
<td>0</td>
<td>$-\sin \Theta$</td>
</tr>
<tr>
<td>$y_{\Theta}$</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$z_{\Theta}$</td>
<td>$\sin \Theta$</td>
<td>0</td>
<td>$\cos \Theta$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$C_{\Psi}$</th>
<th>$x_{\Theta}$</th>
<th>$y_{\Theta}$</th>
<th>$z_{\Theta}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{\Psi}$</td>
<td>$\cos \Psi$</td>
<td>$\sin \Psi$</td>
<td>0</td>
</tr>
<tr>
<td>$y_{\Psi}$</td>
<td>$-\sin \Psi$</td>
<td>$\cos \Psi$</td>
<td>0</td>
</tr>
<tr>
<td>$z_{\Psi}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Orthogonality

If we let $Q$ be square, $n \times n$, matrix and suppose $Q^{-1} = Q^T$ then $Q$ is orthogonal if and only if:

$$QQ^T = Q^TQ = I$$  (2.1.15)

where $I$ is the identity matrix. This implies that the rotated vector length is unchanged. Alternatively, an orthogonal matrix may be defined as a square matrix with entries whose rows and columns are perpendicular and of unit length, ie. orthogonal unit vectors or orthonormal vectors.
2. *Non-Commutative*

Direction cosine matrices do not commute:

\[ C_1C_2 \neq C_2C_1 \]  

(2.1.16)

3. *Successive rotations may be described by product of individual plane rotation matrices.*

The orientation of a three-dimensional coordinate frame to another may be obtained by a sequence of three successive rotations. By tradition, aerospace applications perform these transformations through right handed rotations in each coordinate planes, referred to earlier as plane rotations, in the Z-Y-X order Stevens and Lewis [15]; alternate sign convention and planes of rotation exist in other fields, eg. 3D modeling in computer science. Right handed rotation about the z-axis is positive yaw, right handed rotation about the y-axis is positive pitch, and right handed rotation about the x-axis is positive roll. Order of rotation sequence is arbitrary, Figure 2.4 depicts a complete coordinate transformation in a X-Y-Z (Roll-Pitch-Yaw) manner. This rotation sequence is suitable for calculating aircraft attitudes

![Direction Cosine Matrix Visual](image)

**Figure 2.4: Direction Cosine Matrix Visual**

with respect to the north-east-down frame. These angles of rotation are called *Euler angles.*
In terms of coordinate transformations

\[ F_B = (C_\Phi C_\Theta C_\Psi) F_{NED} \]  

(2.1.17)

where Equation 2.1.18 is the complete coordinate transformation from north-east-down to the body frame, ie. \( C_{F_{NED}/F_B} = C_\Phi C_\Theta C_\Psi \)

\[
C_\Phi C_\Theta C_\Psi = \begin{bmatrix}
\cos \Theta \cos \Psi & \cos \Theta \sin \Psi & -\sin \Theta \\
-\cos \Phi \sin \Psi + \sin \Phi \sin \Theta \cos \Psi & \cos \Phi \cos \Psi + \sin \Phi \sin \Theta \sin \Psi & \sin \Phi \cos \Theta \\
\sin \Phi \sin \Psi + \cos \Phi \sin \Theta \cos \Psi & -\sin \Phi \cos \Psi + \cos \Phi \sin \Theta \sin \Psi & \cos \Phi \cos \Theta
\end{bmatrix}
\]  

(2.1.18)

2.1.3 Aircraft Dynamics

With Assumptions 2.1 & 2.2 in our front pocket, which is more accessible than the back pocket, we now have an idealized rigid body and live in a world suited for the application of Newton’s Laws. With this we can describe translational and rotational motion of the aircraft by its kinematic analogs: linear momentum, \( p \), and angular momentum, \( H \) respectively.

"By Newton’s second law the time rate of change of linear momentum equals the sum of all externally applied forces, \([ F ]\).

\[ F = \frac{d}{dt} (p) = \frac{d}{dt} (mV) \]  

(2.1.19)

and the rate of change of angular momentum is equal to the sum of all applied torques

\[ M = \frac{d}{dt} (H) \]  

(2.1.20)

These vector differential equations provide the starting point for a complete description of the rigid body motions of the vehicle.” McRuer et al. [1]

Assumption 2.3. Mass is considered constant

In most aerospace systems thrust is generated by an expenditure of vehicle mass; an exception being electric powered applications. Whether that trade in mass directly contributes to linear
momentum or not needs to be considered. In the present propulsion case a heavy fuel piston
engine turns a propeller, therefore the thrust generated may be considered an external force.
Alternatively, if a jet engine were used there would be a component of thrust due to expulsion
of vehicle mass\(^4\).

### 2.1.3.1 Translational Acceleration

The goal is to derive equations for translation accelerations in the wind axis reference frame,
eventually arriving at \(\dot{V}, \dot{\alpha}, \) and \(\dot{\beta}\). Picking up where Equation 2.1.19 left off, along with
Assumption 2.3, we may expand the expression to

\[
F = m \left[ \frac{d}{dt} (V) + \Omega \times V \right]
\]

where \(F\) is the total force acing on the vehicle, \(m\) is the vehicle mass, \(V\) is the total vehicle
velocity, and \(\Omega\) is the total vehicle angular velocity:

\[
F = \begin{bmatrix}
\sum F_x \\
\sum F_y \\
\sum F_z 
\end{bmatrix} = \begin{bmatrix}
F_{xT} + F_{xA} + F_{xG} \\
F_{yT} + F_{yA} + F_{yG} \\
F_{zT} + F_{zA} + F_{zG}
\end{bmatrix}
\]

\[
V = [u, v, w]^T
\]

\[
\Omega = [P, Q, R]^T
\]

The elements of \(F\) are summations of externally applied propulsive (T), aerodynamic (A),
and gravitational (G) forces respective to each body axis. It will be assumed that the engine
is mounted to align with body axes, therefore there are no thrust-angles or \(F_{yT} = F_{zT} = 0\).

The body axis aerodynamic forces can be transformed to their equivalent stability axis
components lift \(L\), drag \(D\), and side-force \(Y\) as Figure 2.5 indicates.

---

\(^4\) McRuer et al. [1] derives a modified extension of Equation 2.1.19 to take this into account.
Figure 2.5: Body Axes, Forces, and Moments

\[
F_{x_A} = - D \cos \alpha + L \sin \alpha \quad (2.1.25)
\]

\[
F_{y_A} = Y \quad (2.1.26)
\]

\[
F_{z_A} = - D \sin \alpha - L \cos \alpha \quad (2.1.27)
\]

The gravitational forces can be resolved into body axis components such that

Figure 2.6: Orientation of Gravity Vector with Respect to Body Axes [1]
\[ F_{xG} = -mg \sin \theta \quad (2.1.28) \]
\[ F_{yG} = mg \sin \phi \cos \theta \quad (2.1.29) \]
\[ F_{zG} = mg \cos \phi \cos \theta \quad (2.1.30) \]

Combining these we arrive at an expression that considers all external forces acting on the airframe.

\[
\begin{bmatrix}
\sum F_x \\
\sum F_y \\
\sum F_z
\end{bmatrix} =
\begin{bmatrix}
F_{xT} - D \cos \alpha + L \sin \alpha - mg \sin \theta \\
Y + mg \sin \phi \cos \theta \\
-D \sin \alpha - L \cos \alpha + mg \cos \phi \cos \theta
\end{bmatrix}
\quad (2.1.31)
\]

By rearranging Equation 2.1.21 to solve for translational acceleration, \( \frac{dv}{dt} \), we can express body axis accelerations in terms of body axis forces, angular rates, and velocities:

\[
\frac{d}{dt} (V) = \frac{1}{m} \mathbf{F} - \mathbf{\Omega} \times \mathbf{V}
\quad (2.1.32)
\]

Substitution of Equations 2.1.23, 2.1.24, and 2.1.22 yields

\[
\begin{bmatrix}
\dot{u} \\
\dot{v} \\
\dot{w}
\end{bmatrix} =
\begin{bmatrix}
\frac{1}{m} (F_{xT} + F_{xA} + F_{xG}) + Rv - Qw \\
\frac{1}{m} (F_{yT} + F_{yA} + F_{yG}) + Pw - Ru \\
\frac{1}{m} (F_{zT} + F_{zA} + F_{zG}) + Qu - Pv
\end{bmatrix}
\quad (2.1.33)
\]

In order to express Equation 2.1.33 in the wind axis system, \( \dot{\mathbf{V}} = [\dot{V}, \dot{\alpha}, \dot{\beta}]^T \), will need to use the aforementioned equations

\[
\begin{align*}
u &= V \cos \beta \cos \alpha \\
v &= V \sin \beta \\
w &= V \cos \beta \sin \alpha
\end{align*}
\quad (2.1.2, 2.1.3, 2.1.4)
\]

and transforms

\[
\begin{align*}
V &= \sqrt{u^2 + v^2 + w^2} \\
\alpha &= \tan^{-1} \left( \frac{w}{u} \right) \\
\beta &= \sin^{-1} \left( \frac{v}{\sqrt{u^2 + v^2 + w^2}} \right)
\end{align*}
\quad (2.1.8, 2.1.9, 2.1.10)
\]
Refer to Duke et al. [16, Appendix B.] to see the remainder of the derivation, ie. how to take derivatives and make the appropriate substitutions to arrive at $\dot{V}$, $\dot{\alpha}$, and $\dot{\beta}$ equations:

\[
\dot{V} = \frac{1}{m} \left( -D \cos \beta + Y \sin \beta + T \cos \alpha \cos \beta \right) - g \sin \gamma
\]

\[
\dot{\alpha} = -\frac{1}{mV \cos \beta} \left( L + T \sin \alpha \right) + \frac{g \cos \gamma \cos \mu}{V \cos \beta} + Q - \tan \beta \left( P \cos \alpha + R \sin \alpha \right)
\]

\[
\dot{\beta} = \frac{1}{mV} \left( D \sin \beta + Y \cos \beta - T \sin \beta \cos \alpha \right) + \frac{g \cos \gamma \sin \mu}{V} + P \sin \alpha - R \cos \alpha
\]

2.1.3.2 Rotational Acceleration

The goal is to derive rotational acceleration equations – $\dot{P}$, $\dot{Q}$, $\dot{R}$. Picking up where Equation 2.1.20 left off and substituting total angular momentum for the product of the moment of inertia matrix and rotational velocity vector, $H = I \Omega$, we may expand the expression to

\[
M = \left[ \frac{d}{dt} (I \Omega) + \Omega \times I \Omega \right]
\]  

where $M$ is the total moment acting on the vehicle, $I$ is the inertia matrix (alternatively referred to as tensor or dyad), and $\Omega = [P, Q, R]^T$ is the total vehicle angular velocity:

\[
M = \begin{bmatrix}
\sum \bar{L} \\
\sum \bar{M} \\
\sum \bar{N}
\end{bmatrix} = \begin{bmatrix}
\bar{L} + \bar{L}_T \\
\bar{M} + \bar{M}_T \\
\bar{N} + \bar{N}_T
\end{bmatrix}
\]

$\bar{L}$, $\bar{M}$ , and $\bar{N}$ are the total aerodynamic moments about the $x_B$, $y_B$, and $z_B$ body axes with T subscript indicates moments induced by the power-plant.

\[
I = \begin{bmatrix}
I_{xx} & -I_{xy} & -I_{xz} \\
-I_{xy} & I_{yy} & -I_{yz} \\
-I_{xz} & -I_{yz} & I_{zz}
\end{bmatrix}
\]

Elements along the main diagonal are called the **moments of inertia** with respect to the x, y, and z axes respectively and are always positive. The off diagonal terms are referred to as the **products of inertia** and may be positive, negative, or zero; they are measures of
the imbalance in mass distribution. Note that it is possible to orient the axes in such a way that the products of inertia are zero. In this case the diagonal terms are called the principal moments of inertia.

**Assumption 2.4.** The XZ plane is a plane of symmetry.

This is a very good approximation for most air vehicles, and leads to $I_{yz} = 0$ as well as $I_{xy} = 0$. If we assume that the inertia tensor is constant, as we did with mass in the translational acceleration derivation, then Equation 2.1.34 may be rewritten as

$$\frac{d}{dt} \Omega = I^{-1} (M - \Omega \times I \Omega) \quad (2.1.37)$$

where $d\Omega/dt = [\dot{P}, \dot{Q}, \dot{R}]^T$. Refer to Duke et al. [16, Sec. 1.2.1] to see the remainder of the derivation, i.e. how to make the appropriate substitutions to arrive at $\dot{P}$, $\dot{Q}$, and $\dot{R}$ equations.

This is further simplified by Stevens and Lewis [14, Pg. 80]:

$$\dot{P} = (c_1 R + c_2 P) Q + c_3 \bar{L} + c_4 \bar{N}$$

$$\dot{Q} = c_5 PR - c_6 (P^2 - R^2) + c_7 \bar{M}$$

$$\dot{R} = (c_8 P - c_2 R) Q + c_4 \bar{L} + c_9 \bar{N}$$

where [14] defines $c$ terms as

$$\Gamma = I_{xx} I_{zz} - I_{xz}^2$$

$$c_5 = \frac{I_{zz} - I_{xx}}{I_{yy}}$$

$$\Gamma c_1 = (I_{yy} - I_{zz}) I_{zz} - I_{xz}^2$$

$$c_6 = \frac{I_{xx}}{I_{yy}}$$

$$\Gamma c_2 = (I_{yy} - I_{zz}) I_{zz} - I_{xz}^2$$

$$c_7 = \frac{1}{I_{yy}}$$

$$\Gamma c_3 = I_{zz}$$

$$\Gamma c_4 = I_{xx}$$

$$\Gamma c_8 = I_{xx} (I_{xx} - I_{yy}) + I_{xz}^2$$

$$\Gamma c_9 = I_{xx}$$

(2.1.38)

### 2.1.3.3 Attitude Rates

Attitudes $\dot{\chi}$, $\dot{\gamma}$, and $\dot{\psi}$ may be derived by observing the velocity of the wind-axis system with respect to the gravity vector. Derivations in an Euler sense (φ, θ, ψ) are shown in
McRuer et al. [1, Pg. 222] and Duke et al. [16, Sec. 1.2.3]. Similarly to the wind-axis attitude derivation, the flight-path components end up being:

\[
\dot{\chi} = \frac{1}{mV \cos \gamma} \left[ D \sin \beta \cos \mu + Y \cos \beta \cos \mu + L \sin \mu \\
+ T \left( \sin \alpha \sin \mu - \cos \alpha \sin \beta \cos \mu \right) \right]
\]

\[
\dot{\gamma} = \frac{1}{mV} \left[ -D \sin \beta \sin \mu - Y \cos \beta \sin \mu + L \cos \mu \\
+ T \left( \sin \alpha \cos \mu + \cos \alpha \sin \beta \sin \mu \right) \right] - \frac{g}{V} \cos \gamma
\]

\[
\dot{\mu} = \frac{1}{mV} \left[ D \sin \beta \tan \gamma \cos \mu + Y \cos \beta \tan \gamma \cos \mu + L \left( \tan \beta + \tan \gamma \sin \mu \right) \right] \\
+ T \left( \sin \alpha \tan \gamma \sin \mu + \sin \alpha \tan \beta - \cos \alpha \sin \beta \tan \gamma \cos \mu \right) \\
- \frac{g \tan \beta \cos \gamma \cos \mu}{V} + \frac{P \cos \alpha + R \sin \alpha}{\cos \beta}
\]

Figure 2.7: Fury 1500 UAS on Launcher
[www.aviationweek.com/aw/blogsmain/]
2.1.3.4 Equations of Motion

Collecting equations from the previous sections, the complete set of nonlinear equations of motion turn out to be

\[
\dot{\chi} = \frac{1}{mV \cos \gamma} \left[ D \sin \beta \cos \mu + Y \cos \beta \cos \mu + L \sin \mu \right. \\
+ T \left( \sin \alpha \sin \mu - \cos \alpha \sin \beta \cos \mu \right] \\
\dot{\gamma} = \frac{1}{mV} \left[ -D \sin \beta \sin \mu - Y \cos \beta \sin \mu + L \cos \mu \right. \\
+ T \left( \sin \alpha \cos \mu + \cos \alpha \sin \beta \sin \mu \right] - \frac{g}{V} \cos \gamma \\
\dot{V} = \frac{1}{m} \left( -D \cos \beta + Y \sin \beta + T \cos \alpha \cos \beta \right) - g \sin \gamma \\
\dot{\mu} = \frac{1}{mV} \left[ D \sin \beta \tan \gamma \cos \mu + Y \cos \beta \tan \gamma \cos \mu + L \left( \tan \beta + \tan \gamma \sin \mu \right) \right. \\
+ T \left( \sin \alpha \tan \gamma \sin \mu + \sin \alpha \tan \beta - \cos \alpha \sin \beta \tan \gamma \cos \mu \right) \right. \\
- \frac{g \tan \beta \cos \gamma \cos \mu}{V} + \frac{P \cos \alpha + R \sin \alpha}{\cos \beta} \\
\dot{\alpha} = - \frac{1}{mV \cos \beta} \left( L + T \sin \alpha \right) + \frac{g \cos \gamma \cos \mu}{V \cos \beta} + Q \\
- \tan \beta \left( P \cos \alpha + R \sin \alpha \right) \\
\dot{\beta} = \frac{1}{mV} \left( D \sin \beta + Y \cos \beta - T \sin \beta \cos \alpha \right) + \frac{g \cos \gamma \sin \mu}{V} \\
+ P \sin \alpha - R \cos \alpha \\
\dot{P} = (c_1 R + c_2 P) Q + c_3 \bar{L} + c_4 \bar{N} \\
\dot{Q} = c_5 P R - c_6 \left( P^2 - R^2 \right) + c_7 \bar{M} \\
\dot{R} = (c_8 P - c_2 R) Q + c_4 \bar{L} + c_9 \bar{N} \\
\]

where \( c \) terms are defined in Equation 2.1.38.
2.1.4 System Observations

The nonlinear differential equations summarized in Sec. 2.1.3.4 may be reduced to the form

$$\dot{x} = f(x, t)$$ (2.1.40)

where $\dot{x}$ is the $n \times 1$ derivative of the state vector with respect to time, $f$ is an $n \times 1$ nonlinear vector function expressing the six-degree of freedom rigid body equations, and $x$ is the $n \times 1$ state-vector with respect to time. Additionally, the state-vector is defined as a set of real numbers, $(x_1, \ldots, x_n)^T$, contained in $n$-dimensional Euclidean space, denoted by the symbol $\mathbb{R}^n$, and is formally referred to as state-space. The parameter $n$ is the system order\(^5\) and refers to the number of first order differential equations required to represent an equivalent $n^{th}$ order ordinary differential equation (ODE).

Equation 2.1.40 represents the closed-loop time-variant dynamics of a feedback control system, even though it does not explicitly contain a control input vector $u$. This is because the control input may be considered a function of state $x$ and time $t$, therefore disappearing in the closed-loop dynamics. Showing this mathematically, if the plant dynamics are

$$\dot{x} = f(x, u, t)$$ (2.1.41)

and some control law $u$ has been selected as

$$u = g(x, t)$$ (2.1.42)

then the closed-loop dynamics are

$$\dot{x} = f [x, g(x, t), t]$$ (2.1.43)

which can be rewritten in the form of Equation 2.1.40; since $g(x, t)$ is a function of the state $x$, which is already represented in the expression, it may be discarded. Naturally,

\(^5\) The highest derivative of the dependent variable with respect to the independent variable appearing in the equation.
Equation 2.1.40 may also represent a system without control input, such as a freely moving spring-mass damper or pendulum. These examples, as with all physical systems, are time dependent.

Given a particular initial condition, an ODE may have several system trajectories. Continuity of $f$, ie. $\lim_{x \to a} f(x) = f(a)$, ensures that there is at least one solution but does not ensure uniqueness of the solution. A stronger and therefore more frequently used condition, that guarantees Lipschitz continuity, may be used to prove existence and uniqueness as well as protect against the possibility of $f(x)$ having an infinite slope, eg. a discontinuity.

**Definition 2.1.1 (Lipschitz Condition).**

Khalil [2, §2.2]

If there exists a strictly positive Lipschitz constant $L$ such that $f(x, t)$ satisfies the inequality,

$$\|f(x, t) - f(y, t)\| \leq L\|x - y\| \quad \forall x, y \in \mathcal{D}$$

then the function $f(x, t)$ is said to be **Lipschitz in** $x$ for all points in the domain $\mathcal{D}$.

Further specifying conditions on the domain $\mathcal{D}$, over which the Lipschitz condition holds, imposes restrictions on input values for the function $f(x, t)$. A function is said to be **locally Lipschitz in** $x$ if that for each point $x \in \mathcal{D} \subset \mathbb{R}^n$ there exists a finite neighborhood $\mathcal{D}_0 \in \mathcal{D}$ such that the Lipschitz condition holds for all points in $\mathcal{D}_0$ with a corresponding Lipschitz constant $L_0$.

**Theorem 2.1.1 (Global Existence and Uniqueness).**

Khalil [2, Thm 2.4]

Let $f(x, t)$ be piecewise continuous in $t$ and **locally Lipschitz in** $x$ for all $t \geq t_0$ and all $x$ in a domain $D \subset \mathbb{R}^n$ and let $W$ be a **compact (closed and bounded) subset** of $\mathcal{D}$. If for $x_0 \subset W$ it is known that every solution of

$$\dot{x} = f(x, t), \quad x(t_0) = x_0 \quad \forall t \geq t_0$$

lies entirely in $W$. Then there is a **unique solution** that is defined for all $t \geq t_0$
Proof: Refer to Khalil [2, Pg. 77]

“The trick in applying Theorem 2.1.1 is in checking the assumption that every solution lies in a compact set without actually solving the differential equation.” This concept is introduced in Lyapunov’s stability definitions to come.

**Definition 2.1.2** (Autonomous and Non-Autonomous Systems). Slotine and Weiping [3] The nonlinear system $\dot{x} = f(x, t)$ is said to be autonomous if $f$ does not depend explicitly on time, ie. if the system’s state equation can be written

$$\dot{x} = f(x) \quad (2.1.44)$$

otherwise, the system is called non-autonomous.

Again, all real-world systems are non-autonomous, but “in practice system properties often change very slowly, so we can neglect their time variation without causing any practically meaningful error.” [3] Most importantly, Definition 2.1.2 implies that solutions, or system trajectories, of autonomous ODEs are independent of initial time, thereby greatly simplifying stability analysis; In other words, stability for autonomous systems does not depend on initial conditions.

**2.2 Stability**

The goal is to familiarize the reader with concepts required to prove stability for the backstepping control architecture. Most proofs will not be included, but will be referenced immediately succeeding theorems. It is assumed that the reader has a good understanding of solution properties to ordinary differential equations such as existence, uniqueness, and continuity. Mathematical notation and terminology tied to these properties may be used in subsequent theorems and definitions without introduction. A great overview of mathematical preliminaries pertinent to nonlinear systems and backstepping control is included in Khalil
Backstepping control designs are constructed using Lyapunov stability theory. This is currently the “most useful and general approach,” Slotine and Weiping [3], to establishing stability for nonlinear systems and may also be extended to linear systems. It was published in 1892 by Russian mathematician Alexandr Lyapunov in *The General Problem of Motion Stability for systems without inputs*. As a consequence it has traditionally been applied to closed-loop control systems, ie. those in which a feedback control law has already been selected. Later, in Sec. 2.3.3, a method for designing a feedback control law in conjunction with Lyapunov theory will be introduced. Furthermore, Lyapunov stability theory provides two methods for stability analysis, indirect and direct. The first method\(^6\), indirect or linearization, determines *local* stability properties; eigenvalues of a linear (approximate, hence indirect) system reveals stability in the immediate vicinity of an equilibrium point. The second method, direct, determines *regional* stability properties; the aim is to make the system act like a function whose time derivative guarantees some form of stability. Since our system equations are nonlinear only Lyapunov’s direct method will be covered.

Ultimately, this section condenses the philosophy, definitions, and theorems of Slotine and Weiping [3, Chp. 3], Khalil [2, Chp. 3], Härkegård [17, Chp. 3], Krstic et al. [18, Chp. 2] and Farrell and Polycarpou [5, Appendix A] in a brief and clear manner.

\(^6\) In my research I found conflicting evidence on what Lyapunov’s first method actually was. [3] warns the reader that linearization is sometimes incorrectly referred to as the first method, which should be Lyapunov’s method of exponents. This was confirmed, as many other sources I found on the topic referred to the indirect method as the first method.
2.2.1 Equilibrium and Operating Points

A system trajectory may correspond to only a single point $x^*$, called an equilibrium point, if once $x(t)$ is equal to $x^*$ it remains equal to $x^*$ for all time. For the non-autonomous system in Equation 2.1.40, the equilibrium points are the real roots (x-intercepts (for $n = 2$ systems)) of the differential equation, that is

$$f(x^*, t) = 0, \quad \forall t \geq 0$$

(2.2.1)

An operating point is a region of stability formally defined as “any state space location at which the system can be forced into equilibrium by choice of control signal.” For the generalized system containing control input $u$ in Equation 2.1.42, the vectors $(x_0, u_0)$ are operating points if

$$f(x_0, u_0, t) = 0, \quad \forall t \geq 0$$

(2.2.2)

Varying the control input changes the operating point, implying that these points are not isolated. A collection of these points is called a surface of operating points, and is illustrated in the following example through multiple phase portraits.

For a second order system ($n = 2$), solutions of an ODE may be realized in phase-space\(^7\) as trajectories from $t = (0, \infty)$.

**Example 2.2.1** (Operating Points).

Given the system

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = x_1^3 + u$$

and applying the operating point definition for an arbitrary control signal $u_0 = -1$, an operating point emerges at $x_0 = [1, 0]^T$. If the control input is varied, we can see the surface of operating points:

\(^7\) A vector field of derivatives $[f_2(x_1, x_2), f_1(x_1, x_2)]$ at respective state variable $(x_2, x_1)$ locations that allows for visualization of the qualitative behavior of the system. Especially useful for classifying stability of equilibrium points.
Visualization of trajectories, through phase portraits\(^8\), provides an intuitive feel for the stability of an operating point. Red dots (•) are operating points, green lines (—) are stable or unstable manifolds, blue lines (—) are solution trajectories, and grey vectors (→) indicate solution direction tangent to trajectories. Qualitative graphical insight is often more clear, simple, and useful than an analytical approach. You can see that the operating point moves along the \(x_1\) axis as control input is varied and that trajectories flow away from these operating points, therefore they’re unstable. The surface of operating points may be expressed as \(x_0 = [a, 0]^T\) with \(u_0 = -a^3\). This example proves that operating points are not isolated. Most importantly, the system could be forced to operate at any point on the surface if a stabilizing controller, \(u = -x_1^3 - (x_1 - a) - x_2\), was selected.

As Figure 2.8 shows, the operating point is not always coincident with the state-space origin, ie. \(x = 0\). For the sake of notational and analytical simplicity one may translate the equilibrium point to the origin without loss of generality by redefining the state-vector. This allows one to analyze the local stability of the system, neglecting possible higher order terms \(O^3\) and above. Using notation from [18, Pg. 23]:

\[
z = x - x^* \tag{2.2.3}
\]

\(^8\) Made using pplane8, http://math.rice.edu/~dfield/index.html
To prove this statement, substitute a reformulation of previous expression, \( x = z + x^* \), into Equation 2.1.40 as shown:

\[
\dot{z} = f(z + x^*, t)
\]  

(2.2.4)

It is clear that by substituting Equation 2.2.3 into Equation 2.2.4 one would arrive at a system equivalent to the original, i.e. \( \dot{z} = \dot{x} = f(x, t) \). “In addition \( z = 0 \), the solution corresponding to the original system’s equilibrium point \( x = x^* \), is an equilibrium point of Equation 2.2.4; [recall Equation 2.2.1]. Therefore, instead of studying the behavior of the original system, Equation 2.1.40, in the neighborhood of \( x^* \), one can equivalently study behavior of the redefined system, Equation 2.2.4, in the neighborhood of the origin”[3] ; what was meant by, “without loss of generality.”

In cases like aircraft trajectory control, the concept of stability about a point is not what we’re concerned about. In the presence of atmospheric disturbances trajectory perturbations will arise, and it is the stability of motion that is important: Will the system remain on its desired trajectory if slightly perturbed away from it? Lyapunov synthesis will answer this question.

### 2.2.2 Lyapunov Stability Definitions

A stable system is one that starts near a desired operating point and stays within a bound of that point ever after; unstable otherwise. Linear stability is evaluated in terms of a single equilibrium point, while nonlinear stability is based on the idea of boundedness as these systems can have several isolated equilibrium points. This implies much more complex and unfamiliar behavior for nonlinear systems, therefore requiring more refined stability concepts. This section will formally introduce these concepts in the Lyapunov sense, thereby providing the tools necessary to prove stability for backstepping control architectures.

To begin, let \( \epsilon \) denote a spherical region defined by \( ||x|| < \epsilon \) in state-space and \( \delta \) denote a spherical region that is generally within \( \epsilon \); \( \delta \) is called a domain of attraction and \( \epsilon \) is a
function of $\delta$. Recall that equilibrium points may be translated to the origin by redefining the state-vector as shown in Equation 2.2.3. The theorems in this section will be presented as if the equilibrium point was translated to the origin; note that $z$ notation will be dropped, and the usual $x$ representation will be used to represent a system with redefined equilibrium point.

**Definition 2.2.1 (Stability in the Sense of Lyapunov).** Khalil [2, Def 3.2]

For the non-autonomous system

$$\dot{x} = f(x, t)$$

where $x \in \mathbb{R}^n$, and $f : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}^n$ is piecewise continuous in $t$ and locally Lipschitz in $x$, the equilibrium point $x^* = 0$ is

- **stable** if for each $\epsilon > 0$, there exists a variable $\delta = \delta(\epsilon, t_0) > 0$ such that

  $$\|x(t_0)\| < \delta \Rightarrow \|x(t)\| < \epsilon, \quad \forall t \geq t_0 \geq 0 \quad (2.2.5)$$

- **uniformly stable** if for each $\epsilon > 0$, there exists a variable $\delta = \delta(\epsilon) > 0$, independent of $t_0$ such that Equation 2.2.5 is satisfied

- **unstable** if not stable.

- **asymptotically stable** if it is stable and there exists a constant $c = c(t_0) > 0$ such that

  $$\|x(t_0)\| < c \Rightarrow \lim_{t \to \infty} x(t) = x^* = 0 \quad (2.2.6)$$

- **uniformly asymptotically stable** if it is uniformly stable and there exists a constant $c > 0$ independent of $t_0$, such that Equation 2.2.6 is satisfied uniformly in $t_0$; that is, for each $\eta > 0$, there is $T(\eta) > 0$ such that

  $$\|x(t)\| < \eta, \quad \forall t \geq t_0 + T(\eta), \quad \forall \|x(t_0)\| < c \quad (2.2.7)$$

- **globally uniformly asymptotically stable** if it is uniformly asymptotically stable and for each $\eta > 0$ and $c > 0$ there is $T(\eta, c) > 0$ such that
\[ \|x(t)\| < \eta \quad \forall t \geq t_0 + T(\eta, c) \quad \forall \|x(t_0)\| < c \quad (2.2.8) \]

- **exponentially stable** if for any \( \epsilon > 0 \) and some \( \lambda > 0 \) there exists \( \delta = \delta(\epsilon) > 0 \) such that

\[ \|x(t_0)\| < \delta \quad \Rightarrow \quad \|x(t)\| < \epsilon e^{-\lambda(t-t_0)} \quad \forall t > t_0 \geq 0 \quad (2.2.9) \]

Again these definitions are developed with respect to a non-autonomous, or time-variant, system therefore include initial time, \( t_0 \). It is presented in this form for the sake of generality, as these definitions are also applicable to autonomous systems by the substitution \( t_0 = 0 \). It is clear through these definitions that Lyapunov evaluated stability by ensuring that solutions were not only bounded, but also that “the bound on the solution [could] be made as small as desired by restriction of the size of the initial condition.”[5].

With regards to **uniform stability**, the additional stipulation over ordinary stability is that \( \delta \) is independent of \( t_0 \); all properties are uniform if the system is time-invariant. Important for asymptotically stable equilibrium points is the additional stipulation with initial time, ie. \( \|x(t_0)\| = \|x_0\| < c(t_0) \). This implies that an attractive region, \( c, \) for every initial time, \( c(t_0) \), exists: \( c(t_0) > 0 \). Stability properties are said to be **global** when the domain is equal to \( \mathbb{R}^n \). Terms like asymptotic and global stability are foreign in classic controls sense because all linear time-invariant (LTI) solutions are global and exponential. To conclude comments on **Definition 2.2.1**, the state vector of the exponentially stable system converges faster than an exponential function, where the positive number \( \lambda \) is the rate of exponential convergence. “By writing the positive constant \( \epsilon \) as \( \epsilon = e^{\lambda(t-t_0)} \) it is easy to see that after a time of \( \tau_0 + (1/\lambda) \), the magnitude of the state vector decreases to less than 35% \( (\approx e^{-1}) \) of its original value; this is similar to the notion of a time-constant in a linear system. After \( \tau_0 + (3/\lambda) \), the state magnitude \( \|x(t)\| \) will be less than 5% \( (\approx e^{-3}) \) of \( \|x(0)\| \)” [3]. Also, exponential stability implies asymptotic stability, but not the other way ‘round.

It turns out that operating points which are uniformly asymptotically stable are more
robust to disturbances than those that are merely stable, especially important for adaptive designs [2, Sec. 2.1]. It’s also important to remember that these are merely definitions; positive constants \( \delta, \epsilon, c, \) and \( \eta \) are arbitrarily fixed bounds that are used to evaluate solution trajectories.

For systems up to second order (\( n = 1, 2 \)), a two-dimensional\(^9\) graphical analysis via vector fields and solution trajectories, called phase portraits, may be utilized to determine stability. First order systems may be represented as a vector field on the “x-axis”: it dictates the velocity vector \( \dot{x} \) at each \( x \). “The behavior of the solution in the neighborhood of the origin can be determined by examining the sign of \( f(x) \). The \( \epsilon - \delta \) requirement for stability is violated if \( x(f(x)) > 0 \) on either side of the origin.”

\[\text{Figure 2.9: First Order Unstable Systems Khalil [2, Online Lecture Notes]}\]

“\( \text{The origin is stable if and only if } x f(x) \leq 0 \text{ in some neighborhood of the origin} \)”

\[\text{Figure 2.10: First Order Stable Systems Khalil [2, Online Lecture Notes]}\]

\(^9\) However, technically speaking phase portraits may be drawn in three dimensional space. This is not very common, and can be complicated to generate as well as interpret.
“The origin is asymptotically stable if and only if $x f(x) < 0$ in some neighborhood of the origin.”

Figure 2.11: First Order Asymptotically Stable Systems Khalil [2, Online Lecture Notes]

In two-dimensions, Figure 2.12 depicts stable (S), asymptotically stable (AS), and unstable (US) trajectories in phase-space with respect to $\epsilon$ and $\delta$ contours in Definition 2.2.1; the initial condition is $x(0)$, green line is the trajectory $x(t)$, and origin is the equilibrium point $x^*$.

Figure 2.12: Stability Concepts

This representation is a great conceptual tool, as it brings tangible meaning to the definitions. It may be used to analyze nonlinear systems, especially useful for those with
multiple equilibrium points, as around each point a linear system approximation is valid. In practice, many nonlinear aircraft control architectures are linearized about a trim condition in order to apply classical frequency domain control techniques such as Bode or Root-Locus analysis. These are formalized and well understood techniques which use performance metrics, such as gain and phase margin, to ensure a particular degree of stability.

Figure 2.13 illustrates possible equilibrium point classifications in equivalent root-loci and phase-portraits:

![Figure 2.13: Equilibrium Point Classification for 2nd Order Linear Systems][3]

Unfortunately however, most physical systems of interest are higher than second order and are therefore incapable of being displayed graphically. Lyapunov’s direct method provides an analytical tool for this case.
2.2.2.1 Perturbed Systems

These concepts evaluate the stability of an *ideal* system. Undoubtedly there are model uncertainties\(^{10}\) and disturbances that effect the true stability of the air vehicle. Boundedness concepts from stability analysis in the Lyapunov sense may be applied to a reformulated system if uncertainties do not effect system order, as:

\[
\dot{x} = f(x, t) + g(x, t) \tag{2.2.10}
\]

where \(f(x, t)\) is the nominal system and \(g(x, t)\) is the perturbation term. Typically we don’t know \(g(x, t)\) but we know something about its bounds, for example an upper bound on \(\|g(x, t)\|\). Note that if \(g(x, t) \neq 0\) the origin is not necessarily an equilibrium point of Equation 2.2.10. If a solution trajectory is kept arbitrarily close to an equilibrium point in the presence of *sufficiently small disturbances* then the system may be considered **totally stable**. The following definition formalizes this concept:

**Definition 2.2.2 (Total Stability).** Slotine and Weiping [3, Defn. 4.13]

The equilibrium point \(x^* = x(0) = 0\) for the unperturbed system in Equation 2.1.44, \(\dot{x} = f(x)\), is said to be totally stable if for every \(\epsilon \geq 0\), two numbers exist \(\delta_1\) and \(\delta_2\) exist such that \(\|x(t_0)\| < \delta_1\) and \(\|g(x, t)\| < \delta_2\) imply that every solution \(x(t)\) of the perturbed system Equation 2.2.10 satisfies the condition \(\|x(t_0)\| < \epsilon\). "Note that total stability is simply a local version (with small input) of BIBO (bounded-input bounded-output) stability.” Furthermore, equilibrium points that are uniformly asymptotically stable, therefore exponentially stable points as well, may be proven totally stable by use of converse Lyapunov theorems, mentioned in Sec. 2.2.3.

**Theorem 2.2.1 (Asymptotic Stability and Total Stability).** \([3, \text{Thm. 4.14}]\)

If the equilibrium point \(x^* = x(0) = 0\) for the unperturbed system in Equation 2.1.44, \(\dot{x} = f(x)\), is uniformly asymptotically stable, then it is totally stable.

\(^{10}\) Errors in modeled parameters are called parametric uncertainties, while neglected or unknown parameters are non-parametric uncertainties.
2.2.3 Lyapunov Stability Theorems

Aleksandr Lyapunov realized that stability of an equilibrium point may be established if one can make the system act like a scalar, energy-like function, $V(x)$, and examine its time derivative along trajectories of the system. If this function’s time derivative, $\dot{V}(x)$, is decreasing over time then it may be asserted that the system will eventually reach an equilibrium condition. Table 2.3 outlines the theorems that will be covered.

<table>
<thead>
<tr>
<th>$\dot{x} = f(x)$</th>
<th>$V &gt; 0$ $\dot{V} \leq 0$ S</th>
<th>Lyapunov’s Direct Method Thm. 2.2.2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$V &gt; 0$ $\dot{V} &lt; 0$ AS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V &gt; 0$ $\dot{V} &lt; 0$ GAS</td>
<td></td>
</tr>
<tr>
<td>$\dot{x} = f(x,t)$</td>
<td>$V \geq 0$ $\dot{V} \leq 0$ CONV LaSalle Thm. 2.2.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V &gt; 0$ $\dot{V} \leq 0$ GAS Barbashin–Krasovskii Cor. 2.2.1</td>
<td></td>
</tr>
<tr>
<td>$\dot{x} = f(x,t)$</td>
<td>$V \geq 0$ $\dot{V} \leq 0$ CONV Barbalat’s Lemma Lem. 2.2.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V &gt; 0$ $\dot{V} \leq 0$ GUAS LaSalle-Yoshizawa Thm. 2.2.4</td>
<td></td>
</tr>
</tbody>
</table>

Before presenting stability theorems, some function related terminology must be introduced:

**Definition 2.2.3.** A scalar function $V(x)$ is

- **positive definite** if $V(0) = 0$ and $V(x) > 0$, $x \neq 0$
- **negative definite** if $V(0) = 0$ and $V(x) < 0$, $x \neq 0$
- **positive semidefinite** if $V(0) = 0$ and $V(x) \geq 0$, $x \neq 0$
- **negative semidefinite** if $V(0) = 0$ and $V(x) \leq 0$, $x \neq 0$
- **sign indefinite** if it is not any of the above
- **radially unbounded** if $V(x) \to \infty$ as $\|x\| \to \infty$
Where \( V(x) \) is assumed to be a scalar function on \( D \) into \( \mathbb{R} \), ie. \( V : D \to \mathbb{R} \), is continuously differentiable, and is defined in a domain \( D \subset \mathbb{R}^n \) that contains the origin \( x = 0 \). When a function is positive or negative (semi)definite, ie. convex, then there is one and only one unique global minimum or maximum respectively; this is the mathematical justification for stability. The derivative of \( V(x) \) along the system trajectory for the autonomous system in Equation 2.1.44 is obtained by the chain rule:

\[
\dot{V}(x) = \frac{d}{dt} V(x) = \frac{\partial V}{\partial x} \frac{dx}{dt} = \frac{\partial V}{\partial x} \dot{x} = \frac{\partial V}{\partial x} f(x) \tag{2.2.11}
\]

The scalar function \( V(x) \) has an implicit dependence on time and its derivative is dependent on the system’s equation, therefore each system will have a different \( \dot{V}(x) \). Not to mention that the form of \( V(x) \) is anything but consistent, as will be covered later.

**Theorem 2.2.2** (Lyapunov’s Direct Method).

Khalil [2, Thm. 3.1]

Let the origin be an equilibrium point, \( x^* = x(0) = 0 \), for an autonomous system, \( \dot{x} = f(x) \) and \( V : \mathbb{R}^n \to \mathbb{R} \) a continuously differentiable, positive definite function, then

- the origin is **stable (S)** if

\[
\dot{V}(x) = \frac{\partial V}{\partial x} f(x) \leq 0 \quad , \quad \forall \ x \in D \tag{2.2.12}
\]

- the origin is **asymptotically stable (AS)** if

\[
\dot{V}(x) = \frac{\partial V}{\partial x} f(x) < 0 \quad , \quad \forall \ \{ x \in D \mid x \notin 0 \} \tag{2.2.13}
\]

**Proof:** Refer to Khalil [2, Pg. 101]

**Remark ( Theorem 2.2.2)** Because the system is time-invariant these equilibrium point properties are uniform, ie. uniformly stable and uniformly asymptotically stable. At its core, Lyapunov’s direct method determines stability properties of \( x \) through a relationship between \( f \) and a positive definite function, \( V(x) \). This theorem asserts that if the system loses energy over time it will eventually reach an equilibrium condition.
A function $V(x)$ satisfying Theorem 2.2.2 is called a Lyapunov function, otherwise a potential function. From a physics approach, it is essentially a model of system energy and therefore typically takes the form of a quadratic, kinetic-energy like term. It is most easily understood from a geometric perspective through a two-dimensional phase-portrait.

A Lyapunov function may be visualized as a collection of closed concentric contours in $\mathbb{R}^3$; in the simplest sense, imagine looking down a bowl-like shape with numerous rings drawn on the inside that specify height in that space. Figure 2.14 demonstrates this analogy for decreasing values of $c$.

Formally, each contour, $c$, characterizes a level set of the Lyapunov function $V(x) = c$ for some $c > 0$, and is called a Lyapunov or level surface. It’s very important to understand that a Lyapunov function is a projection of the state variables $x_1$ and $x_2$ onto a third dimension representing the energy of the system; in other words, the state solution is not fit to a Lyapunov function. It is directly dependent on system trajectories, ie. $V(x)$, which are solutions to the differential equations that govern system dynamics, thereby indirectly dependent on system dynamics. The take away is that $V$ is unique to every system and measures how far from equilibrium the system is.

A point of confusion for me was thinking that you specify a $V$ and the trajectory flows down whatever shape you have. This is wrong, a Lyapunov function is a projection of the state onto this 3rd dimension that characterizes energy of the system. You don’t pick an energy function you want and make the system fit it, you alter the system until you’re satisfied with its energy function.

The partial derivative term in Equation 2.2.11 may be considered as the gradient of $V$ with respect to $x$, representing a vector pointing in the direction of maximum increase of $V$. The vector $dx/dt$ represents the system dynamics, which could’ve equivalently been

---

11 Not in a circular sense by any means, the contours may comprise any closed shape. The main point is that they share a common center. 12 A level set is a collection of equilibrium points.
Figure 2.14: Geometric Interpretation of Lyapunov Stability Thm. -
Underlying image from [4]

labeled $f(x)$, and is tangent to the solution $x(t)$. The condition imposed by Lyapunov, ie.
$\dot{V}(x) = (\partial V/\partial x)f(x) \leq 0$, implies that solutions $x(t)$ always cross the contours of $V$ with
an angle greater than or equal to 90° relative to the outward normal; if $dx/dt$ points inward
then system trajectories will always move to smaller and smaller values of $V$. $\dot{V}(x) \leq 0$ does
not ensure that a trajectory will get to the origin\textsuperscript{13}, but does imply the origin is stable, since
by Definition 2.2.1: when a trajectory starts within $\delta$ (a level surface for this case), it will
stay within $\epsilon$.

When an equilibrium point is classified as asymptotically stable it requires the initial
condition to be within some domain $D$, but how big can this domain be? Establishing global
asymptotic stability expands the region of attraction to the whole space $\mathbb{R}^n$ by an extra
condition, radial unboundedness, as described previously in Definition 2.2.3.

\textsuperscript{13} LaSalle's Invariance Principle [2, Thm. 3.4] may be used to prove convergence of a solution to the largest
invariant set for all point within a region of attraction where $\dot{V}(x) = 0$. 
Theorem 2.2.2 (continued)\[Khalil \ [2, \ Thm. \ 3.2]\]

Let the origin be an equilibrium point, $x^* = x(0) = 0$, for an autonomous system, $\dot{x} = f(x)$ and $V : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable, positive definite, radially unbounded function such that

$$\dot{V}(x) = \frac{\partial V}{\partial x} f(x) < 0, \quad \forall \ x \neq 0$$ \quad (2.2.14)

then the origin is globally asymptotically stable (GAS).

**Proof:** Refer to Khalil [2, Pg. 111]

As shown, “the direct method of Lyapunov replaces [an] n-dimensional analysis problem that is difficult to visualize with a lower dimensional problem that is easy to interpret.” [5]

The large appeal of this method is that stability of an equilibrium point may be inferred without explicitly solving system equations. The difficulty is that finding a Lyapunov function for complex systems is like finding a needle in a haystack. Most significantly, this theorem is a) non-constructive – there is no systematic method for finding a $V$ to satisfy stability requirements – and b) only offers sufficient conditions for stability – it does not say whether the given conditions are also necessary. In other words, if a Lyapunov function does not satisfy the conditions for stability or asymptotic stability, then no conclusion can be made about the stability properties of the system.

However, there are tools that not only support the search for $V$, ie. addressing a), but also overturning b), referred to as Converse Lyapunov Theorems\[14\]. If a sub-system of a nonlinear system exhibits stability then converse theorems may be used to generate a Lyapunov function for the sub-system. This implies that a Lyapunov function for the whole system may exist, thereby making Lyapunov stability conditions necessary; it’s nice to know there’s hope! Unfortunately however, almost always they assume some knowledge of the solution to the differential equation.

\[14\] Converse Theorems, see Khalil [2, Sec. 3.6]
For cases where \( \dot{V} \) is only negative semi-definite, i.e. \( \dot{V} \leq 0 \), global asymptotic stability may still be established through LaSalle’s\(^{15}\) Invariance Principle, an invariant set theorem:

**Lemma 2.2.1** (Fundamental Property of Limit Sets). \(^{Khalil}[2, \text{Lem. 3.1}]\)
If a solution \( x(t) \) to the autonomous system \( \dot{x} = f(x) \) is bounded and belongs to \( D \) for \( t \geq 0 \), then its positive limit set \( L^+ \) is a nonempty, compact, invariant set. Moreover,

\[
\lim_{t \to \infty} x(t) = L^+
\]

**(2.2.15)**

**Theorem 2.2.3** (LaSalle’s Theorem). \(^{Khalil}[2, \text{Thm. 3.4}]\)
Let \( \Omega \subset D \) be a compact set that is positively invariant for an autonomous system, \( \dot{x} = f(x) \). Let \( V : D \to \mathbb{R} \) be a continuously differentiable function such that \( \dot{V}(x) \leq 0 \) in \( \Omega \). Let \( E \) be the set of all points in \( \Omega \) where \( \dot{V}(x) = 0 \). Let \( M \) be the largest invariant set in \( E \). Then every solution starting in \( \Omega \) approaches \( M \) as \( t \to \infty \).

**Proof:** Refer to Khalil [2, Pg. 115]

LaSalle’s theorem also extends Lyapunov’s theorem in two ways by \( a \) providing an estimate to the region of attraction specified as any compact positively invariant set and \( b \) allowing Theorem 2.2.2 to be applied for cases where the system has an equilibrium set, i.e. dynamic convergence or limit cycles, rather than a single equilibrium point.

**Corollary 2.2.1** (Barbashin-Krasovskii). \(^{Khalil}[2, \text{Cor. 3.2}]\)
Let the origin be an equilibrium point, \( x^* = x(0) = 0 \), for an autonomous system, \( \dot{x} = f(x) \) and \( V : \mathbb{R}^n \to \mathbb{R} \) be a continuously differentiable, positive definite, radially unbounded function such that

\[
\dot{V}(x) = \frac{\partial V}{\partial x} f(x) \leq 0 \quad \forall \ x
\]

**(2.2.16)**

Let \( S = \{ x \in \mathbb{R}^n \mid \dot{V}(x) = 0 \} \) and suppose that no solution can stay identically in \( S \), other than the trivial solution. Then the origin is **globally asymptotically stable (GAS)**.

\(^{15}\) Joseph P. LaSalle received a mathematics doctoral degree in 1941 from Caltech and worked alongside Lefschetz at Brown University in the 1960’s. Oddly there wasn’t much more information about him.
Remark (Corollary 2.2.1) When \( \dot{V}(x) \leq 0 \) and \( S = \{0\} \), Corollary 2.2.1 coincides with Theorem 2.2.2. It is also referred to as the Krasovskii-LaSalle method in some textbooks. LaSalle published this theorem in the west, unaware that it was earlier published in Russia; most likely attributed to a language barrier or lack of cooperation due to political tension of the 1950’s when this theorem was developed.

LaSalle’s invariant set based theorem is applicable to autonomous systems that desire state convergence to a constant, therefore time-invariant, reference signal. If the control objective is tracking of a time-varying reference signal then LaSalle’s theorem is insufficient because the system is then non-autonomous. Subsequently, stability analysis is more difficult when the system is time-variant because it’s harder to find a Lyapunov function, now dependent on both \( x \) and \( t \), ie. \( V(x,t) \), that has a negative-definite derivative. For tracking analysis, tools developed by LaSalle, Yoshizawa, and Barbalat are relied upon.

**Theorem 2.2.4 (LaSalle–Yoshizawa).** *Krstic et al. [18, Thm. 2.1/A.8]*

Let the origin be an equilibrium point, \( x^* = x(0) = 0 \), for a non-autonomous system, \( \dot{x} = f(x,t) \) and suppose \( f \) is locally Lipschitz in \( x \) uniformly in \( t \). Let \( V : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}_+ \) be a continuously differentiable, positive definite, and radially unbounded function \( V = V(x,t) \), then

\[
\dot{V}(x,t) = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x,t) \leq -W(x) \leq 0, \quad \forall t \geq 0, \quad \forall x \in \mathbb{R}^n \tag{2.2.17}
\]

Where \( W(x) \) is a continuous function. Then all solutions of [Equation 2.2.17] are globally uniformly bounded and satisfy

\[
\lim_{t \to \infty} W(x(t)) = 0 \tag{2.2.18}
\]

In addition, if \( W(x) > 0 \) ie. positive definite, then the equilibrium \( x^* = x(0) = 0 \) is globally uniformly asymptotically stable (GUAS).

**Proof:** Refer to Krstic et al. [18, Appendix A, Pg. 492]
A technical lemma by Barbalat usually precedes Theorem 2.2.4 which is a purely mathematical result inferred by asymptotic properties of functions and their derivatives:

- If $\dot{f}(t) \to 0$ it does not imply that $f(t)$ converges

**Example 2.2.2.** As the derivative term converges to zero, the solution does not, for the system:

$$
f(t) = \sin(\log(t)) \quad ; \quad \dot{f}(t) = (\cos(\log(t)))/t$$

![Figure 2.15: Asymptotic Property 1: $\dot{f}(t) \to 0 \not\Rightarrow f \to \text{constant}$](image)

- If $f(t)$ converges it does not imply that $\dot{f}(t) \to 0$

**Example 2.2.3.** As the solution tends to zero, the derivative is unbounded, for the system:

$$
f(t) = e^{-t} \sin^2(e^{2t}) \quad ; \quad \dot{f}(t) = 2e^t \sin(2e^{2t}) - e^{-t} \sin^2(e^{2t})$$

![Figure 2.16: Asymptotic Property 2: $f \to \text{constant} \not\Rightarrow \dot{f}(t) \to 0$](image)

- If $f(t)$ is lower bounded and decreasing, ie. $\dot{f}(t) \leq 0$, then $f(t)$ converges to a limit, but $\dot{f}(t)$ is not guaranteed to diminish if at all.
An additional “smoothness” property on the Lyapunov derivative imposed by Barbalat’s lemma guarantees that \( \dot{f}(t) \) actually converges to zero. This result ensures that a system will fulfill its tracking requirement.

**Lemma 2.2.2** (Barbalat’s Lemma).

*If it can be shown that the differentiable function is bounded, then it may be considered uniformly continuous and convergence may be established.*

- **Form 1:** Examine the Function’s Derivative
  
  Slotine and Weiping [3]
  
  “If the differentiable function \( f(t) \) has a finite limit as \( t \to \infty \), and is such that \( \ddot{f} \) exists and is bounded, [ie. uniformly continuous], then:"

  \[
  \lim_{t \to \infty} \dot{f}(t) = 0
  \]

- **Form 2:** “Lyapunov-Like Lemma”
  
  Slotine and Weiping [3]
  
  If a scalar function \( V(x, t) \) satisfies the following conditions:
  
  - \( V(x, t) \) is lower bounded
  - \( \dot{V}(x, t) \) is negative semi-definite
  - \( \dot{V}(x, t) \) is uniformly continuous in time (by proving \( \ddot{V} \) is bounded)

  then \( \dot{V}(x, t) \to 0 \) as \( t \to \infty \).

- **Form 3:** \( \mathcal{L}_p \) Space Representation
  
  Farrell and Polycarpou [5, A.2.2.3]
  
  Consider the function \( \phi(t) : \mathbb{R}_+ \to \mathbb{R} \) be in \( \mathcal{L}_\infty \), \( d\phi/dt \in \mathcal{L}_\infty \), and \( d\phi/dt \in \mathcal{L}_2 \), then

  \[
  \lim_{t \to \infty} \phi(t) = 0
  \]

- **Form 4:** Initial Version, Barbalat 1959
  
  Khalil [2, Lemma 4.2]
  
  Let \( \phi(t) : \mathbb{R}_+ \to \mathbb{R} \) be a uniformly continuous function. Suppose that \( \lim_{t \to \infty} \int_0^t \phi(\tau) d\tau \) exists and is finite, then:

  \[
  \lim_{t \to \infty} \phi(t) = 0
  \]

**Proof:** Refer to Slotine and Weiping [3, Sec. 4.5.2, Pg. 124]
In the next sections you will see how backstepping may be applied to control of nonlinear systems. The procedure involves choosing a $V(x)$ that retains useful nonlinearities and developing a stabilizing feedback control law.

## 2.3 Control

The aim of this section is to introduce control law design considerations and thoroughly present backstepping control theory. A simplified first order nonlinear system will serve as the basis for development, in order to clearly conceptualize motivating factors. Key features of the method will be cited along the way, eventually leading to a command filtered backstepping technique for the ninth order nonlinear equations of motion formulated in Sec. 2.1.3.4: Equation 2.1.39.

### 2.3.1 Requirements

“The most important design specification is to achieve asymptotic tracking of a known reference trajectory with the strongest possible form of stability. The designed controller should provide effective means for shaping the transient performance and thus allow different performance-robustness trade-offs.”

–Krstic et al. [18]

Loosely speaking, the approach to nonlinear control design is qualitative, while linear control is quantitative. Numerous analysis tools are available for linear control system design that have very specific metrics common in the controls community; in the time domain step response analysis yields terms like rise time; overshoot; and settling time, in the frequency-domain Bode analysis yields terms like phase margin; gain margin; and bandwidth, and root-locus / Nichols analysis represents a mixture of aforementioned methods. Through use of tools like these, control system requirements may be imposed *systematically* on closed-loop controllers for linear systems, hence quantitatively. For nonlinear systems a frequency-domain description is not possible, therefore requirements are typically evaluated using time-domain analyses of transient system response for a given control input. Furthermore, nonlinear
systems often act in peculiar ways, expressing erratic behavior sensitive to even the smallest changes in initial condition and system parameters, or commands, (think chaotic systems, ie. Lorenz Attractor). This demonstrated inconsistency is a reason why analysis tools are limited for nonlinear systems, each system is unique and exhibited motion is dependent on inputs. As a result, for nonlinear systems, qualitative requirements for desired behavior are specified within the operating region of interest.

The implications are that on the designers end a much deeper understanding of vehicle dynamics are necessary for nonlinear system control development; one can’t simply set and forget system equations as with linear systems and then rely on analysis tools to help tune a controller. In fact in backstepping designs if useful, ie. stabilizing, nonlinearities are recognized then they may be retained, thereby reducing a) the overall control effort needed and b) the level of modeling fidelity. The problem is that experience drives these considerations, hence the motivation for including the equations of motion derivation in subsection 2.1.3. Other benefits and disadvantages of backstepping will be covered, but first nonlinear control design considerations and options are briefly introduced.

As outlined by Slotine and Weiping [3], the following characteristics are considered by designers when evaluating nonlinear control system requirements and behavior.

- **Stability** must be guaranteed for the nominal model, either in a local or global sense. The region of stability and convergence are also of interest.

- **Accuracy and Speed of Response** may be considered for some motion trajectories, typically derived from mission requirements, in the region of operation.

- **Robustness** is the sensitivity to effects which are not considered in the design, such as disturbances, measurement noise, unmodeled dynamics, etc. Leads to robust or adaptive control problem formulations.

- **Cost** of a control system is determined by the number and type of actuators, sensors, computers, and time necessary to implement it.
These metrics are in direct competition with one another, as control systems cannot exhibit all of these characteristics to their full extent. The designer is responsible for making effective trade-offs for conflicting requirements and most importantly recognizing when to freeze design iterations.

Stability in the nonlinear sense does not imply that a system is capable of handling constant disturbances; recall from Definitions 2.2.1 and 2.2.2 that stability is defined with respect to initial conditions. For example, a system is stable in the Lyapunov sense if a trajectory starts within $\delta$ and stays within $\epsilon$; a persistent wind-shear – erratic disturbance due to thermals, downdraft, inversion layers, etc. – may shift the equilibrium point, therefore starting within the $\delta$ region does not mean you’re starting within some bounds of the true equilibrium point. The effects of persistent disturbances are resolved through robustness techniques. “Robustness is a property which guarantees that essential functions of the designed system are maintained under adverse conditions in which the model no longer accurately reflects reality.” [19]

2.3.2 Objective and Methods

Definition 2.3.1. Control Tasks

- **Regulation**: Reference signal is constant.
- **Tracking**: Reference signal is time-varying.

In general, there are two tasks for any flight control system: regulation, sometimes referred to as stabilization, aims to hold a particular state to a time-independent reference value, common examples being temperature control and aircraft altitude control. If the objective of the controller is tracking, commonly called a tracker, then the aim is to make a particular state to follow a time-dependent reference signal, examples include making a robot hand draw circles or aircraft flight path following. The goal in both these cases is to drive the deviation from a desired reference value/signal to zero. Convergence for regulation problems
may be achieved using LaSalle’s Theorem 2.2.3, while tracking problems rely on Barbalat’s Lemma 2.2.2.

One of the simplest and aging rivals of backstepping is feedback linearization, or nonlinear dynamic inversion. In backstepping literature this nonlinear control architecture usually plays the victim; it is a point solution that renders the system linear by constructing control laws that cancel nonlinear plant dynamics with feedback, just as the term reads. Consequently it’s considered wasteful and inflexible, requiring more control effort and exhibiting a lack of robustness to uncertainties. It can get especially complicated for high order systems, over an order of two. Table 2.4 covers features of a few nonlinear control techniques.

Table 2.4: Nonlinear Control Method Overview

<table>
<thead>
<tr>
<th>Method</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial and Error</td>
<td>• Visual stability analysis via phase portraits</td>
<td>• Applicable only to simple systems up to second order</td>
</tr>
<tr>
<td>Small Singular Linearization</td>
<td>• Good closed-loop performance for a equilibrium point (SSL).</td>
<td>• Accurate only in a neighborhood around operating point(s)</td>
</tr>
<tr>
<td>Gain Scheduling</td>
<td>• Good closed-loop performance over many equilibrium points (GS).</td>
<td>• Controller parameters fixed online</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A lot of offline validation required</td>
</tr>
<tr>
<td>Feedback Linearization</td>
<td>• Globally stable with exponential tracking error</td>
<td>• Lack of controllability at singularities</td>
</tr>
<tr>
<td></td>
<td>• Linear in modeled domain</td>
<td>• Requires exact knowledge and special class of system</td>
</tr>
<tr>
<td></td>
<td>• Bandwidth theoretically infinite for input signal tracking</td>
<td>• More control effort is required</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Not robust to uncertainties</td>
</tr>
<tr>
<td>Backstepping / Robust /</td>
<td>• Globally asymptotically stable</td>
<td>• Analytic derivative calculation</td>
</tr>
<tr>
<td>Adaptive</td>
<td>• Model uncertainties well handled</td>
<td>• Feedback control algorithm complex, especially for high order systems</td>
</tr>
<tr>
<td></td>
<td>• Systematic procedures</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Potential reduction in development time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Useful nonlinearities retained</td>
<td></td>
</tr>
</tbody>
</table>
But how is feedback linearization better than small singular linearization for the same equilibrium point, especially since both systems are linear? In the former, exact state transformations and nonlinear feedback are used, rather than the latter where linear approximations of the nonlinear system and linear feedback are used. The downside is that all nonlinearities must be precisely known. Furthermore, even the “worst case performance attained by a nonlinear controller coincides with the performance attained by the best linear design.” Kokotovic [20, Linear Versus Nonlinear]

“One of the main problems with applying feedback linearization techniques is that the process produces a system with the same relative degree as the original system, but usually with an order that is less. This process results in zero or internal dynamics, which are modes that are effectively rendered unobservable by the linearization process. If the system is non-minimum phase, then the zero dynamics are unstable. The analogy with linear systems is that a zero-pole system is linearized into an all-pole system by selecting the pole-zero excess as the order of the approximating system. In order to produce linearized systems that have no internal dynamics, techniques which preserve the dynamic order of the system are needed.” Tsourdos and White [21]

2.3.3 Lyapunov Based Control Design

There are two ways to apply Lyapunov’s direct method based on where you begin: 1) If a control law is hypothesized then a valid Lyapunov function needs to be found to justify the choice. 2) If a Lyapunov function is hypothesized then a control law needs to be found to make the Lyapunov function valid. This section will apply the latter technique using a control Lyapunov function (CLF).

Up to this point Lyapunov theorems have been used to prove stability of a given system, however the main objective is to design closed-loop systems with desirable stability properties. The following point by Freeman and Kokotovic [19] clearly summarizes the parallelism
between Lyapunov functions and CLFs: “Just as the existence of a Lyapunov function is necessary and sufficient for the stability of a system without inputs [closed-loop], the existence of a CLF is necessary and sufficient for the stabilizability of a system with a control input.”

The “control” prefix implies that the nonlinear system it’s applied to has an explicit dependence on $u$, that is:

$$\dot{x} = f(x, u), \quad x \in \mathbb{R}^n, \quad u \in \mathbb{R}, \quad f(0, 0) = 0 \quad (2.3.1)$$

If a stabilizing feedback control law $\alpha(x)$ is chosen for control input $u$ such that the inequality in Equation 2.3.2 holds, implying $\dot{x} = f(x, \alpha(x))$ where $x = 0$ is an equilibrium point, then the origin is globally asymptotically stable.

$$\dot{V}(x, \alpha) = \frac{\partial V}{\partial x} f(x, \alpha(x)) \leq -W(x) \quad (2.3.2)$$

where $W(x)$ is a positive definite function; see Krstic et al. [18, Sec. 2.1.2].

**Definition 2.3.2 (Control Lyapunov Function (CLF)).** Krstic et al. [18, Def. 2.4],[17]

A positive definite, radially unbounded, smooth scalar function $V = V(x)$ is called a CLF for $\dot{x} = f(x, u)$ if there exists a $u$ such that:

$$\inf_{u \in \mathbb{R}} \left\{ \dot{V}(x, u) = \frac{\partial V}{\partial x} f(x, u) < 0 \right\}, \quad \forall x \neq 0 \quad (2.3.3)$$

where $\inf$ denotes infimum, the greatest lower bound. For systems affine in control, ie. $\dot{x} = f(x) + g(x)u$, $f(0) = 0$,

$$\dot{x} = f(x) + g(x)u, \quad f(0) = 0, \quad (2.3.4)$$

the CLF inequality in Equation 2.3.2 becomes

$$\dot{V}(x, \alpha) = \frac{\partial V}{\partial x} f(x) + \frac{\partial V}{\partial x} g(x)\alpha(x) \leq -W(x) \quad (2.3.5)$$

For this system the only way to satisfy Definition 2.3.2 is if:

$$\frac{\partial V}{\partial x} g(x) = 0 \Rightarrow \frac{\partial V}{\partial x} f(x) < 0, \quad \forall x \neq 0$$
The problem with the CLF concept is that for most nonlinear systems it is unknown. “The task of finding an appropriate CLF may be as complex as that of designing a stabilizing feedback law. For several important classes of nonlinear systems, we will solve these two tasks simultaneously using a backstepping procedure.” Krstic et al. [18]

2.3.4 Backstepping

Key features of backstepping will be verified alongside implementation of the control architecture. Defining attributes and commonly mentioned benefits of backstepping are:

<table>
<thead>
<tr>
<th>Table 2.5: Backstepping Key Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Useful Nonlinearities</strong></td>
</tr>
<tr>
<td><strong>Flexible</strong></td>
</tr>
<tr>
<td><strong>Recursive</strong></td>
</tr>
<tr>
<td><strong>Constructive</strong></td>
</tr>
</tbody>
</table>

Backstepping offers a systematic design procedure via the addition of integrators, and can handle various classes of nonlinear systems; two typical classes will be introduced based on presentation by Krstic et al. [18, Sec. 2.3.1] and Härkégård [17, Sec. 3.3.2].

**Definition 2.3.3** (Strict Feedback System).
The first is a strict feedback system and is of the form:

\[
\begin{align*}
\dot{x} &= f_0(x) + g_0(x)\xi_1 \\
\dot{\xi}_1 &= f_1(x, \xi_1) + g_1(x, \xi_1)\xi_2 \\
\dot{\xi}_2 &= f_2(x, \xi_1, \xi_2) + g_2(x, \xi_1, \xi_2)\xi_3 \\
&\quad \vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x, \xi_1, \cdots, \xi_{k-1}) + g_{k-1}(x, \xi_1, \cdots, \xi_{k-1})\xi_k \\
\dot{\xi}_k &= f_k(x, \xi_1, \cdots, \xi_k) + g_k(x, \xi_1, \cdots, \xi_k)u
\end{align*}
\]  

(2.3.6)

where \( x \in \mathbb{R}^n \), \( \xi_1, \cdots, \xi_k \) are scalar virtual control laws, and the \( x \)-subsystem satisfies assumptions necessary to apply backstepping, to be introduced in Assumption 2.6. The \( \xi \)-system is referred to as “strict-feedback” because “nonlinearities \( f_i \) and \( g_i \) in the \( \dot{\xi}_i \)-equation \((i = 1, \cdots, k)\) depend only on \( x, \xi_1, \cdots, \xi_i \) that is, on state variables that are fed back.”

**Definition 2.3.4 (Pure Feedback System).**

The second is a pure feedback system and is of the form:

\[
\begin{align*}
\dot{x} &= f_0(x, \xi_1) \\
\dot{\xi}_1 &= f_1(x, \xi_1, \xi_2) \\
\dot{\xi}_2 &= f_2(x, \xi_1, \xi_2, \xi_3) \\
&\quad \vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x, \xi_1, \cdots, \xi_k) \\
\dot{\xi}_k &= f_k(x, \xi_1, \cdots, \xi_k, u)
\end{align*}
\]  

(2.3.7)

where \( \xi_i \in \mathbb{R}^n \) and the \( x \)-subsystem again satisfies upcoming Assumption 2.6. The form of this system represents a more general class of “triangular” systems, specifically a lower triangular system. In comparison to strict-feedback systems, System 2.3.7 lacks the affine\(^{16} \) appearance of variables \( \xi_k \) and \( u \).

\(^{16}\) An affine function is just a linear function plus a translation term.
Krstic et al. [18, Sec. 2.3] explicitly shows the recursive design procedures in which a stabilizing control law $\alpha(x)$ is generated from a Lyapunov function $V$ for each intermediate virtual control law $\xi$; to be demonstrated in the derivation of the higher order backstepping controller in Sec. 2.3.4.2.

The control architecture will first be applied to a general second order system, so that the reader may develop a sense of procedure and terminology. An extension of this special case to a higher order system will follow, where clear steps that characterize recursive application of the procedure will be defined.

### 2.3.4.1 Second Order Systems

Typically in feedback control architectures the objective is to create a control law that cancels known dynamics and impose variables that transform the system into a tracking problem. The key term here is known, implying that complete model information is available. Feedback linearization is one such case, where the exact knowledge of nonlinear system dynamics is required; if one of the functions is uncertain then cancellation is not possible. The key idea here is tracking, with the goal of driving the error between an actual and desired value to zero.

Backstepping synthesis efficiently handles these two critical objectives. Stabilizing nonlinear terms in the dynamics, if recognized, may be retained hence less precise modeling information and less control effort is necessary. Also, the inclusion of nonlinearities improves transient performance.

To begin the backstepping procedure, we’ll select a general first order system ($n = 1$),

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

(2.3.8)

---

17 Alternatively referred to as integrator backstepping.
and augment it with an integrator, thereby transforming it to a second order system:

\[
\dot{x} = f(x) + g(x)\xi \\
\dot{\xi} = u
\]  

(2.3.9a)  

(2.3.9b)

where \([x, \xi]^T \in \mathbb{R}^{n+1}\) is the state and \(u \in \mathbb{R}\) is the control input. The function \(f : \mathcal{D} \to \mathbb{R}^n\) and \(g : \mathcal{D} \to \mathbb{R}^n\) are smooth in a domain \(\mathcal{D} \subset \mathbb{R}^n\) that contains \(x = 0\) and \(f(0) = 0\). The main objective is to design a state feedback control law \(u\) to force \(\xi\) to perform either 1) regulation by stabilizing the origin \((x = 0, \xi = 0)\), i.e. \(x(t) \to 0\) as \(t \to \infty\), or 2) tracking by causing the \(x\)-portion of the state to track a reference signal, say \(y_d\), i.e. \(x(t) \to y_d\) as \(t \to \infty\); with respect to System 2.3.8, “this is equivalent to treating \(\xi\) as a virtual control input for the \(\dot{x}\) equation,” hence we call \(\xi\) a virtual control law. The same idea can be found in cascaded control design, as System 2.3.9 may be viewed as the cascade connection of \(\dot{x}\) and \(\dot{\xi}\) subsystems. This is shown in Figure 2.17 a), where again the first equation treats \(\xi\) as a “control input,” the second equation is the integrator, and the dashed box is the original system in Equation 2.3.8.

For simplification, hence comprehensibility, backstepping for the regulation task will be derived herein. If the objective is tracking an exceptional derivation is provided in Farrell and Polycarpou [5, Sec. 5.3.1]. The corresponding assumptions are:

**Assumption 2.5 (Integrator Backstepping).**

- Full state feedback
- System in Lower-Triangular form*
- System parameters \(f\) and \(g\) known*
- Smooth, positive definite CLF known*
- Smooth state feedback control \(u\)

*specific to this flavor of backstepping

Suppose that Subsystem 2.3.9a can be stabilized by a state feedback control \(\alpha(x)\),

\[\xi = \alpha(x) \implies \dot{x} = f(x) + g(x)\alpha(x)\]
and further that a Lyapunov function is known that renders the equilibrium point, or origin in this case, asymptotically stable, \( \alpha(0) = 0 \):

\[
\dot{V}(x) = \frac{\partial V}{\partial x} \dot{x} = \frac{\partial V}{\partial x} [f(x) + g(x)\alpha(x)] \leq -W(x), \quad \forall x \in \mathcal{D}
\]  

(2.3.10)

where \( W(x) \) is a positive definite function. By adding and subtracting the term involving the stabilizing function, \( \pm g(x)\alpha(x) \), from the first equation in System 2.3.9, we obtain the equivalent system below, as shown in Figure 2.17 b):

\[
\begin{align*}
\dot{x} &= [f(x) + g(x)\alpha(x)] + g(x) [\xi - \alpha(x)] \quad (2.3.11a) \\
\dot{\xi} &= u \quad (2.3.11b)
\end{align*}
\]

If we let \( z \) be the error state, or deviation of actual from desired virtual control with the latter achieved via a stabilizing function \( \alpha(x) \),

\[
z = \xi - \xi_{des} = \xi - \alpha(x)
\]  

(2.3.12)

then a transformed system, 2.3.13, is obtained by a) substitution of Equation 2.3.12 into Subsystem 2.3.11a and b) the derivative\(^{18} \) of Equation 2.3.12:

\[
\begin{align*}
\dot{x} &= [f(x) + g(x)\alpha(x)] + g(x)z \quad (2.3.13a) \\
\dot{z} &= u - \dot{\alpha} \quad (2.3.13b)
\end{align*}
\]

as depicted in Figure 2.17 c). The term backstepping is derived from the procedure leading to System 2.3.13, whereby taking the derivative of \( z \) led to “backstepping” of \( -\alpha(x) \) through the integrator; this defining feature is highlighted in Figures 2.17 b) and c). A key feature of backstepping is that we don’t use a differentiator to implement the time derivative \( \dot{\alpha} \); since \( f, g, \) and \( \alpha \) are known, the derivative \( \dot{\alpha} \) may be computed offline, ie. analytically, without the need of a differentiator by using the chain rule:

\[
\dot{\alpha} = \frac{\partial \alpha}{\partial x} \dot{x} = \frac{\partial \alpha}{\partial x} [f(x) + g(x)\xi]
\]  

(2.3.14)

\(^{18}z\) is a function of \( x \) only, hence we may use the sum rule for differentiation: \( \dot{z} = d\xi/dx - d\alpha/dx = u - \dot{\alpha} \)
Equation 2.3.9a

Equation 2.3.15a

Figure 2.17: Backstepping Block Diagram Representation: a) Integral augmented cascade system b) introducing $\pm \alpha(x)$ c) “backstepping” $-\alpha(x)$ through the integrator
If we introduce a modified control input by defining \( v = u - \dot{\alpha} \) then the System 2.3.13 may be reduced to

\[
\begin{align*}
\dot{x} & = [f(x) + g(x)\alpha(x)] + g(x)z \quad (2.3.15a) \\
\dot{z} & = v \quad (2.3.15b)
\end{align*}
\]

This is nearly the same as System 2.3.9, except that the inclusion of a stabilizing function \( \alpha \) and change of variables to \( xz \) causes the first equation in this cascade system to have an asymptotically stable origin when the input is zero. If we use

\[ V(x, z) = V(x) + \frac{1}{2} z^2 \quad (2.3.16) \]

as a control Lyapunov function (CLF) and find the total derivative,

\[
\dot{V} = \frac{\partial V}{\partial x} \frac{dx}{dt} + \frac{\partial V}{\partial z} \frac{dz}{dt} \\
= \frac{\partial V}{\partial x} \{[f(x) + g(x)\alpha(x)] + g(x)z\} + \frac{\partial V}{\partial z} v \quad (2.3.17)
\]

where Equation 2.3.10 was utilized to make the \( -W(x) \) substitution, then choosing a modified control input \( v \)

\[ v = -\frac{\partial V}{\partial x} g(x) - kz \quad , \quad k > 0 \quad (2.3.18) \]

and substituting this back into Equation 2.3.17 implies

\[ \dot{V} \leq -W(x) - kz^2 \quad (2.3.19) \]

\( \dot{V} \) proves that in the \( (x, z) \) coordinates the equilibrium point, or origin in this case, \( (x = 0, z = 0) \) is asymptotically stable. Since we chose \( \alpha(0) = 0 \), we may conclude that from Equation 2.3.12 the origin \( (x = 0, \alpha = 0) \) is also asymptotically stable. The final step is substituting \( \dot{z} = v \) and \( \dot{\alpha} \) into Subsystem 2.3.13b and rearranging in order to obtain the state feedback control law, \( u \), for the second order system we started from:

\[
u = \dot{\alpha} + \dot{z} = \frac{\partial \alpha}{\partial x} [f(x) + g(x)\xi] - \frac{\partial V}{\partial x} g(x) - kz \quad (2.3.20)
\]
Further, if all items in Assumption 2.5 hold and \( V(x) \) is radially unbounded then the equilibrium point is globally asymptotically stable. This process is summarized formally by Assumption 2.6 and Lemma 2.3.1:

**Assumption 2.6 (Backstepping Stabilizing Function).** Krstic et al. [18, Asmp. 2.7]

For the general first order system

\[
\dot{x} = f(x) + g(x)u , \quad f(0) = 0 \tag{2.3.21}
\]

where \( x \in \mathbb{R}^n \) is the state and \( u \in \mathbb{R} \) is the control input. There exists a continuously differentiable feedback control law, called a stabilizing function \( \alpha(x) \)

\[
u = \alpha(x) , \quad \alpha(0) = 0
\]

and a smooth, positive definite, radially unbounded function \( V : \mathbb{R}^n \to \mathbb{R} \) such that

\[
\dot{V} = \frac{\partial V}{\partial x} [f(x) + g(x)\alpha(x)] \leq -W(x) \leq 0 \quad , \quad \forall x \in \mathbb{R}^n
\]

where \( W : \mathbb{R}^n \to \mathbb{R} \) is positive semidefinite.

With Theorem 2.2.4 LaSalle-Yoshizawa the control law in Assumption 2.6 guarantees global boundedness of \( x(t) \) via the regulation of \( W(x(t)) \):

\[
\lim_{t \to \infty} W(x(t)) = 0
\]

Furthermore, a stronger convergence result is achievable if Theorem 2.2.3 LaSalle is satisfied with \( W(x) \) is positive definite, the control renders \( x = 0 \) the GAS equilibrium of Equation 2.3.21.

**Lemma 2.3.1 (Backstepping).** Krstic et al. [18, Lem. 2.8]

Let Equation 2.3.21 be augmented by an integrator:

\[
\begin{align*}
\dot{x} &= f(x) + g(x)\xi \\
\dot{\xi} &= u, 
\end{align*} \tag{2.3.22a}
\]

and suppose that Subsystem 2.3.22a satisfies Assumption 2.6 with \( \xi = \mathbb{R} \) as its control.
i) If $W(x)$ is positive definite, then

$$V(x, \xi) = V(x) + \frac{1}{2}[\xi - \alpha(x)]^2$$

is a CLF for System 2.3.22, that is, there exists a feedback control $u = \alpha(x, \xi)$ which renders the origin $(x = 0, \xi = 0)$ the GAS equilibrium point. One such control is

$$u = \frac{\partial \alpha}{\partial x} [f(x) + g(x)\xi] - \frac{\partial V}{\partial x}g(x) - k[\xi - \alpha(x)] , \quad k > 0$$

ii) If $W(x)$ is positive semi-definite, then there exists a feedback control which renders

$$\dot{V} \leq -W(x, \xi) \leq 0,$$

such that $W(x, \xi) > 0$ whenever $W(x) > 0$ or $\xi \neq \alpha(x)$. This guarantees global boundedness and convergence of $[x(t), \xi(t)]^T$ to the largest invariant set $\mathcal{M}$ contained in the set $\mathcal{E} = \{[x, \xi]^T \in \mathbb{R}^{n+1} \mid W(x) = 0, \xi = \alpha(x)\}$

“The main result of backstepping is not the specific form of the control law, Equation 2.3.24, but rather the construction of a Lyapunov function whose derivative can be made negative definite by a wide variety of control laws.” Krstic et al. [18].

This result may be extended to systems with a chain of integrators in a systematic fashion. “The only difference is that there will be more virtual states to “backstep” through. Starting with the “farthest” from the actual control, each step of the backstepping technique can be broken up into three parts;” Sonneveldt et al. [7]

**Definition 2.3.5** (Constructive Nature of Backstepping - Design Procedure).

1. Introduce a virtual control $\xi = \alpha(x)$, an error state $z$, and rewrite the current state equation in terms of these.

2. Choose a CLF for the system, treating it as a final stage.

3. Choose an equation for the virtual control that makes the CLF stabilizable.

The following examples will show how control law and control Lyapunov function (CLF) selection effects control design. In Example 2.3.1, it will shown how recognition of useful
nonlinearities in Lyapunov based control design leads to a more efficient control law than one designed via feedback linearization. In Example 2.3.2, backstepping will be applied to the same system to demonstrate the technique and show the flexibility in this procedure.

**Example 2.3.1 (Useful Nonlinearities).**

Consider the system

\[ \dot{x} = -x^3 + x + u \tag{2.3.25} \]

and let \( x = 0 \) be the desired equilibrium point. The unforced dynamics, \( u = 0 \), \( \therefore \) \( \dot{x} = -x^3 + x \), are depicted via the phase portrait in Figure 2.18.

Note that this is a 1D phase portrait, unlike the 2D phase portraits illustrated in subsection 2.2.1; the differential equation is represented as a vector field on the x-axis, determined by the velocity \( \dot{x} \) at each \( x \). A physical way to understand this is to imagine fluid flowing along the x-axis with a velocity that varies according to System 2.3.25; Strogatz [22]. If \( \dot{x} < 0 \) then the flow is to the left and to the right if \( \dot{x} > 0 \). Equilibrium points coincide with x-axis intersections, as application of Equation 2.2.1 dictates, and depending on the surrounding flow will be stable or unstable.

For the desired equilibrium point, ie. the origin, “to be asymptotically stable, the sign of
\( \dot{x} \) should be opposite that of \( x \) for all \( x \).” Figure 2.18 shows the linear term, \(+x\), dominates and de-stabilizes near the origin (dashed gray line), while the cubic term, \(-x^3\), dominates and stabilizes for values of \( x \) outside of \( \pm 1 \). Remember that these observations are for the unforced form of System 2.3.25. Developing a stabilizing control law, i.e. working with the forced form of the system, will transform the dynamics to make a stable operating point coincident with the origin; in the phase portrait we would see a single, stable equilibrium point at the origin and system trajectory only occupying quadrants II and IV.

We begin Lyapunov based control law development by recognizing that “to make the origin GAS only the linear dynamics need to be counteracted by the control input. This can be achieved by selecting

\[
u = -x
\] (2.3.26)

and a CLF given by

\[
V(x) = \frac{1}{2}x^2
\] (2.3.27)

which yields

\[
\dot{V}(x) = \frac{\partial V}{\partial x} \dot{x} = x(-x^3 + x + u) = -x^4
\] (2.3.28)

proving the origin is GAS according to Cor. 2.2.1 Barbashin-Krasovskii.”

Alternatively, applying feedback linearization requires that the control law counteracts all nonlinear dynamics:

\[
u = x^3 - kx \quad , \quad k > 1
\] (2.3.29)

As mentioned previously, this control law does not recognize the naturally stabilizing nonlinear term, \(-x^3\), in fact it counteracts it thus requiring more control effort in contrast to \( u = -x \).

Example 2.3.2 (Flexibility in Backstepping). Härkegård [17, Ex. 3.1]

Consider System 2.3.25 in the previous example augmented by an integrator where \( x = x_1 \)
and \( \xi = x_2 \).

\[
\begin{align*}
\dot{x}_1 & = -x_1^3 + x_1 + x_2 \quad (2.3.30a) \\
\dot{x}_2 & = u \quad (2.3.30b)
\end{align*}
\]

The first task is to stabilize the equilibrium point of Subsystem 2.3.30a by treating \( x_2 \) as a virtual control input. Since we know from phase portrait observations in Figure 2.18 that \(-x_1^3\) is a useful nonlinearity we set out to preserve this term. The objective is to choose a stabilizing function \( \alpha(x_1) \) that cancels the de-stabilizing linear term \( x_1 \), just as the control law \( u \) in Equation 2.3.26 did:

\[
x_{2des} \equiv \alpha(x_1) = -x_1 \quad (2.3.31)
\]

Next, the error variable \( z \) is introduced, which is defined as the difference between actual and desired virtual control.

\[
z = x_2 - \alpha(x_1) = x_2 + x_1 \quad (2.3.32)
\]

Now we can rewrite the system in \( xz \) coordinates with substitution of 2.3.32 into 2.3.30a and the analytic derivative of \( z \), i.e. \( \dot{z} = \dot{x}_2 + \dot{x}_1 \) with substitution of 2.3.30b:

\[
\begin{align*}
\dot{x}_1 & = -x_1^3 + z \quad (2.3.33a) \\
\dot{z} & = u - x_1^3 + z \quad (2.3.33b)
\end{align*}
\]

Following Backstepping Lemma 2.3.1, a CLF is now constructed for System 2.3.33. Our initial choice will reuse the positive definite quadratic postulated in Equation 2.3.27 for the first term of CLF 2.3.23 in the lemma.

\[
V(x_1) = \frac{1}{2}x_1^2 \quad (2.3.34)
\]

Including the penalizing term for the deviation from the stabilizing function yields

\[
V(x_1, x_2) = V(x_1) + \frac{1}{2}(x_2 - \alpha(x_1))^2 = \frac{1}{2}x_1^2 + \frac{1}{2}z^2 \quad (2.3.35)
\]
Differentiating with respect to time will allow us to evaluate which choices for \( u \) satisfy Lyapunov stability theorems

\[
\dot{V} = x_1(-x_1^3 + z) + z(u - x_1^3 + z) = -x_1^4 + z(x_1 + u - x_1^3 + z) \tag{2.3.36}
\]

In order to “render \( \dot{V} \) negative definite, \( u \) must dominate the \( z \) term using a control input of, eg. \(-3z\). In addition, since the mixed terms between \( x_1 \) and \( z \) are indefinite, there seems to be no other choice than to cancel them using the control law

\[
u = x_1^3 - x_1 - 3z \tag{2.3.37}
\]

This choice however, does not recognize the fact that the \(-x_1^3\) term in subsystem is naturally stabilized outside of \( x_1 = \pm 1 \). We can do better by choosing a different \( V(x_1) \); instead of specifying a CLF beforehand, we will leave \( V(x_1) \) alone and let its formulation fall out of the backstepping design process. This is the reason as to why backstepping is considered a **flexible design procedure**. Consider CLF 2.3.23 again

\[
V(x_1, x_2) = V(x_1) + \frac{1}{2}(z)^2 \tag{2.3.38}
\]

and compute its derivative without presuming \( V(x_1) \) known

\[
\dot{V} = \frac{\partial V}{\partial x_1} \frac{dx_1}{dt} + \frac{\partial V}{\partial x_2} \frac{dx_2}{dt} \\
= \dot{V}(-x_1^3 + z) + z(u - x_1^3 + z) \tag{2.3.39}
\]

\[
= -\dot{V}x_1^3 + z(\dot{V} + u - x_1^3 + z)
\]

Now we are able to choose a \( V(x_1) \) such that \( \dot{V} \) in 2.3.39 cancels the indefinite mixed terms. This is achieved by selecting

\[
\dot{V}(x_1) = x_1^3 \quad \ni \quad V(x_1) = \frac{1}{4}x_1^4 \tag{2.3.40}
\]

With this CLF, 2.3.39 is now

\[
\dot{V} = -x_1^6 + z(u + z) \tag{2.3.41}
\]

64
In contrast to Equation 2.3.36 it is clear that the control law $u$ no longer needs to unnecessarily cancel the $-x_1^3$ term, in fact now we can build a linear control law in $x_1$ and $x_2$

$$u = -3z = -3x_1 - 3x_2$$

(2.3.42)

which renders $\dot{V} = -x_1^6 - 2x_2^2$ negative definite, thereby making the origin GAS. This technique for choosing $V(x_1)$ was published in Krstic et al. [9].

### 2.3.4.2 Higher Order Systems

Lower order system backstepping concepts may be extended to higher order systems, for which simplicity is maintained via recursive application of integrator backstepping. The aim of this section is to exemplify this procedure for a third order, $n = 3$, system applicable to any systems over an order of two.

**Example 2.3.3 (Recursive Nature of Backstepping).** Khalil [2, Ex. 13.7]

Consider the third-order system,

$$\dot{x}_1 = x_2^2 - x_1^3 + x_2$$

(2.3.43a)

$$\dot{x}_2 = x_3$$

(2.3.43b)

$$\dot{x}_3 = u$$

(2.3.43c)

which is nearly identical to System 2.3.30 however it’s further altered by an additional integrator and $x_1$ is now squared. Each step below divides the procedure into a series of virtual control law solutions, with the final step defining the true control input $u$.

**Step 1:** As with Example 2.3.2, we begin by developing a stabilizing feedback control $x_2 \equiv \alpha(x_1)$ for Subsystems 2.3.43a and 2.3.43b. We may consider $x_3$ as the control input “$u$” that stabilizes the origin $x_1 = 0$; necessary in order to draw from Assumption 2.6 and Lemma 2.3.1.

$$\dot{x}_1 = x_2^2 - x_1^3 + x_2$$

(2.3.44a)

$$\dot{x}_2 = x_3$$

(2.3.44b)
Choosing a stabilizing function, again that recognizes \(-x_1^3\) as stabilizing,

\[
\alpha(x_1) = -x_1^2 - x_1 \tag{2.3.45}
\]

leads to the reformulated system

\[
\dot{x}_1 = -x_1 - x_1^3 \tag{2.3.46}
\]

Note that the error state is defined as

\[
z = x_2 - \alpha(x_1) = x_2 - x_1^2 - x_1 \tag{2.3.47}
\]

If we chose the CLF \(V(x_1) = \frac{1}{2}x_1^2\) and take the derivative

\[
\dot{V} = \frac{\partial V}{\partial x_1} \dot{x}_1 = -x_1^2 - x_1^4 \leq -x_1^2, \quad \forall x_1 \in \mathbb{R} \tag{2.3.48}
\]

then apply \(u\) and \(V\) from Lemma 2.3.1 to build the control \(x_3\), with \(f = x_1^2 - x_1^3\), \(g = 1\), and \(k = 1\),

\[
x_3 = \frac{\partial \alpha}{\partial x_1}(x_1^2 - x_1^3 + x_2) - \frac{\partial V}{\partial x_1} - [x_2 - \alpha(x_1)] \tag{2.3.49}
\]

that stabilizes the origin \(x = 0\) globally to form the composite Lyapunov function

\[
V(x_1, x_2) = \frac{1}{2}x_1^2 + \frac{1}{2}(x_2 - x_1^2 - x_1)^2 \tag{2.3.50}
\]

**Step 2:** We now consider the next subsystem, 2.3.43c, by viewing the third order system as a special case of System 2.3.11

\[
\dot{x} = f(x) + g(x)\xi
\]

\[
\dot{\xi} = u
\]

with the system in Step 1 condensed into

\[
x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad f = \begin{bmatrix} x_1^2 - x_1^3 + x_2 \\ 0 \end{bmatrix}, \quad g = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \xi = x_3 \tag{2.3.51}
\]
We know from Step 1 that the following feedback control, \( x_3 \equiv \alpha(x_1, x_2) \), and CLF stabilizes System 2.3.44

\[
\alpha(x_1, x_2) = -(2x_1 + 1)(x_1^2 - x_3 + x_2) - x_1 - (x_2 + x_1^2 + x_1)
\]

\[
V(x_1, x_2) = \frac{1}{2}x_1^2 + \frac{1}{2}(x_2 - x_1^2 - x_1)^2
\]

which we define as the virtual control and portion of the CLF for 2.3.51. Next, Backstepping Lemma 2.3.1 is applied in order to obtain the globally stabilizing feedback control

\[
u = \frac{\partial \alpha}{\partial x_1}(x_1^2 - x_3 + x_2) + \frac{\partial \alpha}{\partial x_2}(x_3) - \frac{\partial V}{\partial x_2} - [x_3 - \alpha(x_1, x_2)] \tag{2.3.52}
\]

and corresponding Lyapunov function

\[
V(x) = V(x_1, x_2) + \frac{1}{2}(x_3 - \alpha(x_1, x_2))^2 \tag{2.3.53}
\]

This example is now summarized for a more general form of System 2.3.9

\[
\dot{x} = f(x) + g(x)\xi \tag{2.3.54a}
\]

\[
\dot{\xi} = f_a(x, \xi) + g_a(x, \xi)u \tag{2.3.54b}
\]

where \( f_a \) and \( g_a \) are smooth, and \( g_a(x, \xi) \neq 0 \) over the domain of interest. We may use the control input

\[
u = \frac{1}{g_a(x, \xi)}[u_a - f_a(x, \xi)] \tag{2.3.55}
\]

which reduces Subsystem 2.3.54b to the “integrator form” \( \dot{\xi} = u_a \). If Backstepping Lemma 2.3.1 conditions are satisfied by a stabilizing function \( \alpha(x) \) and Lyapunov function \( V(x) \) then the Lemma combined with Equation 2.3.55 yields the stabilizing state feedback control

\[
u = \alpha_a(x, \xi) = \frac{1}{g_a(x, \xi)} \left\{ \frac{\partial \alpha}{\partial x} [f(x) + g(x)\xi] - \frac{\partial V}{\partial x} g(x) - k[\xi - \alpha(x)] - f_a(x, \xi) \right\} \tag{2.3.56}
\]

for some \( k > 0 \) and the Lyapunov function

\[
V_a(x, \xi) = V(x) + \frac{1}{2}[\xi - \alpha(x)]^2 \tag{2.3.57}
\]
for System 2.3.54. By recursive application of backstepping, we can stabilize strict feedback systems as shown in Definition 2.3.3:

\[
\begin{align*}
\dot{x} &= f_0(x) + g_0(x)\xi_1 \\
\dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2 \\
\dot{\xi}_2 &= f_2(x,\xi_1,\xi_2) + g_2(x,\xi_1,\xi_2)\xi_3 \\
&\vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x,\xi_1,\cdots,\xi_{k-1}) + g_{k-1}(x,\xi_1,\cdots,\xi_{k-1})\xi_k \\
\dot{\xi}_k &= f_k(x,\xi_1,\cdots,\xi_k) + g_k(x,\xi_1,\cdots,\xi_k)u
\end{align*}
\] (2.3.6)

**Step 1:** The recursive procedure begins with the first subsystem of 2.3.6

\[
\dot{x} = f_0(x) + g_0(x)\xi_1 
\] (2.3.58)

where \(\xi_1\) is considered the control input. Just as in Section 2.3.4.1 and Assumption 2.6 if we assume that a stabilizing state feedback control exists \(\xi_1 = \alpha_0(x)\) with \(\alpha_0(0) = 0\), and a Lyapunov function \(V_0(x)\) such that,

\[
\dot{V}_0 = \frac{\partial V_0}{\partial x} [f_0(x) + g_0(x)\alpha_0(x)] \leq -W(x) 
\] (2.3.59)

where \(W(x)\) is some positive definite function, then this subsystem is considered stabilized.

**Step 2:** We may now consider the next subsystem of System 2.3.6 combined with the subsystem in Step 1

\[
\begin{align*}
\dot{x} &= f_0(x) + g_0(x)\xi_1 \\
\dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2
\end{align*}
\] (2.3.60)

which is a special case of System 2.3.54,

\[
\begin{align*}
\dot{x} &= f(x) + g(x)\xi \\
\dot{\xi} &= f_a(x,\xi) + g_a(x,\xi)u
\end{align*}
\] (2.3.54)

with

\[
x = x, \quad \xi = \xi_1, \quad u = \xi_2, \quad f = f_0, \quad g = g_0, \quad f_a = f_1, \quad g_a = g_1
\]
We may reuse 2.3.56 and 2.3.57 to obtain the stabilizing state feedback control and Lyapunov function for System 2.3.60 as

\[
\alpha_1(x, \xi_1) = \frac{1}{g_1} \left[ \frac{\partial \alpha_0}{\partial x} (f_0 + g_0 \xi_1) - \frac{\partial V_0}{\partial x} g_0 - k_1 (\xi_1 - \alpha) - f_1 \right], \quad k_1 > 0 \tag{2.3.61}
\]

\[
V_1(x, \xi_1) = V_0(x) + \frac{1}{2} [\xi_1 - \alpha(x)]^2 \tag{2.3.62}
\]

**Step 3:** Now, consider the next subsystem of System 2.3.6 combined with the subsystems in Step 2

\[
\begin{align*}
\dot{x} &= f_0(x) + g_0(x) \xi_1 \\
\dot{\xi}_1 &= f_1(x, \xi_1) + g_1(x, \xi_1) \xi_2 \\
\dot{\xi}_2 &= f_2(x, \xi_1, \xi_2) + g_2(x, \xi_1, \xi_2) \xi_3
\end{align*} \tag{2.3.63}
\]

which is again a special case of System 2.3.54,

\[
\begin{align*}
\dot{x} &= f(x) + g(x) \xi \\
\dot{\xi} &= f_a(x, \xi) + g_a(x, \xi) u
\end{align*} \tag{2.3.54}
\]

with

\[
x = \begin{bmatrix} x \\ \xi_1 \end{bmatrix}, \quad \xi = \xi_2, \quad u = \xi_3, \quad f = \begin{bmatrix} f_0 + g_0 \xi_1 \\ f_1 \end{bmatrix}, \quad g = \begin{bmatrix} 0 \\ g_1 \end{bmatrix}, \quad f_a = f_2, \quad g_a = g_2
\]

We may again reuse 2.3.56 and 2.3.57 to obtain the stabilizing state feedback control and Lyapunov function for System 2.3.60 as

\[
\alpha_2(x, \xi_1, \xi_2) = \frac{1}{g_2} \left[ \frac{\partial \alpha_1}{\partial x} (f_0 + g_0 \xi_1) + \frac{\partial \alpha_1}{\partial \xi_1} (f_1 + g_1 \xi_2) - \frac{\partial V_1}{\partial \xi_1} g_1 - k_2 (\xi_2 - \alpha_1) - f_2 \right] \tag{2.3.64}
\]

for some \( k_2 > 0 \) and

\[
V_2(x, \xi_1, \xi_2) = V_1(x, \alpha_1) + \frac{1}{2} [\xi_2 - \alpha_2(x, \xi_1)]^2 \tag{2.3.65}
\]

**Step k:** This process may be repeated \( k \) times, hence systematically, to obtain the overall stabilizing state virtual control \( u = \alpha_k(x, \xi_1, \cdots, \xi_k) \) and the Lyapunov function \( V_k(x, \xi_1, \cdots, \xi_k) \) for Strict Feedback System 2.3.6.
2.3.4.3 Command Filtering

As the backstepping procedure for higher order systems shows, the time derivative of the virtual control variables $\alpha_k(x, \xi_1, \cdots, \xi_k)$ may be quite complex, and especially straining computationally when $f$ and $g$ are approximated online. Analytic computation of the virtual control derivative may be avoided by use of the command filter introduced later in Section 2.3.5; for now, we’ll assume that we have command filtered signals and derivatives available.\(^{19}\)

We will introduce the concept with a simple(r) second order system\(^{20}\)

\[
\begin{align*}
\dot{x}_1 &= f_1(x_1) + g_1(x_1)x_2 \\ 
\dot{x}_2 &= f_2(x_1, x_2) + g_2(x_1, x_2)u
\end{align*}
\] (2.3.66a)

where $x = [x_1, x_2]^T \in \mathbb{R}^2$ is the state, $x_2 \in \mathbb{R}^1$, and $u$ is the scalar control signal. The system operates in a region $D$ with $f_i$ and $g_i$ for $i = 1, 2$ known and locally Lipschitz in $x$. Furthermore, $g_i \neq 0$ for all $x \in D$ and again it’s assumed that there is a known command (or desired) trajectory $x_{1c}(t)$, with derivative $\dot{x}_{1c}(t)$, both of which lie in the operating region for $t \geq 0$.

Command filtering introduces tracking errors

\[
\begin{align*}
\bar{x}_1 &= x_1 - x_{1c} \\ 
\bar{x}_2 &= x_2 - x_{2c}
\end{align*}
\] (2.3.67a)

where $x_{2c}$ will be defined by the backstepping controller. This residual is between actual states $x_i$ and compensated (or filtered) states $x_{ic}$. Choose a stabilizing function $\alpha_1$ to cancel

---

\(^{19}\) This section follows work in Farrell and Polycarpou [5][Sec. 5.3.3], and is merely a summary; I strongly urge the reader to read this book if you wish to learn and apply command filtered backstepping flight path control.\(^{20}\) I apologize for the change in system variables. The variables that Farrell chooses $x_1, \cdots, x_k$ are better suited, in my opinion, for full-state flight control design, where system variables up to this point $x, \xi_1, \cdots, \xi_k$ stress concepts crucial to understanding the backstepping procedure.
dynamics in the first subsystem of 2.3.66, i.e. \( x_2 \equiv \alpha_1 \),

\[
\alpha_1(x_1, \tilde{x}_1, \dot{x}_{1c}) = \frac{1}{g_1} [-f_1 - k_1 \tilde{x}_1 + \dot{x}_{1c}]
\]

(2.3.68)

with \( k_1 > 0 \), assuming it’s a smooth feedback control. Note that this choice of stabilizing function reduces Subsystem 2.3.66a to a tracking problem

\[
\dot{x}_1 - \dot{x}_{1c} = -k_1 (x_1 - x_{1c})
\]

(2.3.69)

which implies \( x_1(t) \) converges to \( x_{1c}(t) \) when \( x_1 = x_{1c} \).

Next, define a smooth positive definite function \( V_1(\tilde{x}_1) = \frac{1}{2} \tilde{x}_1^T \tilde{x} \) such that

\[
\dot{V} = \frac{\partial V_1}{\partial \tilde{x}_1} [f_1 + g_1 \alpha_1 - \dot{x}_{1c}] = -W(\tilde{x}_1)
\]

(2.3.70)

where \( W(\tilde{x}_1) = k_1 \tilde{x}_1^T \tilde{x}_1 \) is positive definite in \( \tilde{x}_1 \).

The following procedure may be used to solve the tracking control problem\(^{21} \) in System 2.3.66.

**Step 1**: Define the unfiltered state command and the filter

\[
x_{2c}^o = \alpha_1 - \zeta_2
\]

(2.3.71)

\[
\dot{\zeta}_1 = -k_1 \zeta_1 + g_1 (x_{2c} - x_{2c}^o)
\]

(2.3.72)

where \( \zeta_2 \) will be defined in Step 3. The signal \( x_{2c}^o \) is command filtered to produce the command signal \( x_{2c} \) and its derivative \( \dot{x}_{2c} \); as mentioned in this section’s introduction, the command filter that produces these signals is defined in Section 2.3.5.

\(^{21} \) Note that Backstepping sections before this one performed regulation control, i.e. the desired equilibrium point was the system origin.
Figure 2.19: Equation 2.3.72 Low Pass Filter

By design of Equation 2.3.72, the residual term in this filter \((x_{2c} - x_{2c}^o)\) is bounded and small, therefore as long as \(g_1\) is bounded, then the output \(\zeta_1\) will be bounded; a stable linear filter with a bounded input and output.

**Step 2**: Define compensated tracking errors as

\[
\bar{x}_i = \hat{x}_i - \zeta_i, \quad \text{for } i = 1, 2 \tag{2.3.73}
\]

**Step 3**: Define the unfiltered control input and low pass filter

\[
u_c^o = \frac{1}{g_2} \left[-k_2 \hat{x}_2 + \hat{x}_{2c} - f_2 - x_1^T g_1\right], \quad \text{with } k_2 > 0 \tag{2.3.74}
\]

\[
\dot{\zeta}_2 = -k_2 \zeta_2 + g_2 (u_c - u_c^o) \tag{2.3.75}
\]

"where \(u_c^o\) is filtered to produce \(u_c\) and \(\dot{u}_c\) where \(u = u_c\) is the control signal applied to the actual system.” As mentioned in Step 1, the signal \((u_c - u_c^o)\) in Equation 2.3.75 is bounded and small, therefore if \(g_2\) is bounded then the output \(\zeta_2\) is bounded; another stable linear filter with a bounded input and output. Note, if \(u_c^o = u_c = u\) then \(\zeta_2 = 0\).
Figure 2.20: Command Filter [5]

Figure 2.20 displays a block diagram representation of Steps 1-3 collectively. Farrell and Polycarpou [5] explicitly note that \( u^o_c \) is computed using \( \dot{x}_{2c} \) and not \( \dot{x}^o_{2c} \), where \( \dot{x}_{2c} \) is the output of the command filter for Equation 2.3.71.

Now, to analyze the stability of the control law we need to derive the tracking error dynamics. Begin by taking the derivative of Equation 2.3.67a, then substituting a) Equation 2.3.66a and b) \( \dot{x}_{1c} \) by solving for this term with nested substitution of Equation 2.3.68 into Equation 2.3.71 and re-arranging. Lastly, include \( \pm g_1 x_{2c} \) at the point shown in the derivation

\[
\begin{align*}
\dot{x}_1 &= [\dot{x}_1] - [\dot{x}_{1c}] \\
&= [f_1 + g_1 x_2] - [g_1(x^o_{2c} + \zeta_2) + f_1 + k_1 \ddot{x}_1] \\
&= f' + g_1 x_2 - g_1 x^o_{2c} - g_1 \zeta_2 - f' - k_1 \ddot{x}_1 \mp g_1 x_{2c} \\
&= -k_1 \ddot{x}_1 + g_1 (x_2 - x_{2c}) - g_1 \zeta_2 + g_1 (x_{2c} - x^o_{2c}) \\
&= -k_1 \ddot{x}_1 + g_1 \dddot{x}_2 + g_1 (x_{2c} - x^o_{2c})
\end{align*}
\]

Taking the derivative of Equation 2.3.67b and performing similar substitutions with
Equation 2.3.66a, recall \( u = u_c \), and Equation 2.3.74 yields

\[
\dot{x}_2 = [\dot{x}_2] - [\dot{x}_2c] \\
= [f_2 + g_2u] - [g_2u_c + k_2\dot{x}_2 + f_2 + \bar{x}_1^T g_1] \\
= f_2' + g_2u - g_2u_c - k_2\dot{x}_2 - \bar{x}_1^T g_1 \pm g_2u_c \\
= -k_2\dot{x}_2 - g_1^T \bar{x}_1 + g_2(u - u_c) + g_2(u_c - u_c) \\
= -k_2\dot{x}_2 - g_1^T \bar{x}_1 + g_2(u_c - u_c) \tag{2.3.77}
\]

“As defined by 2.3.72 and 2.3.75, the variables \( \zeta_1 \) and \( \zeta_2 \) represent the filtered effect of the errors \( (x_{2c} - x_{2c}^o) \) and \( (u_c - u_c^o) \) respectively. The variables \( \bar{x}_i \) represent the compensated tracking errors, obtained after removing the corresponding unachieved portion of \( x_{2c}^o \) and \( u_c^o \). After some algebraic manipulation, the dynamics of the tracking errors are described by”

\[
\dot{x}_1 = -k_1\bar{x}_1 + g_1\bar{x}_2 \tag{2.3.78}
\]

\[
\dot{x}_2 = -k_2\bar{x}_2 - g_1^T \bar{x}_1 \tag{2.3.79}
\]

With the following Lyapunov function

\[
V = \sum_{i=1}^{2} \frac{1}{2} \bar{x}_i^T \bar{x}_i \tag{2.3.80}
\]

and the corresponding derivative of \( V \) along the solution of Equation 2.3.78 and 2.3.79 is

\[
\dot{V} = \dot{V}_1 + \dot{V}_2 \\
= -k_1\bar{x}_1^T \bar{x}_1 - k_2\bar{x}_2^2 \leq -\lambda V \tag{2.3.81}
\]

where \( \lambda = 2\min(k_1, k_2) > 0 \). If \( \dot{V} \leq -\lambda V \) is satisfied then the origin \( (\bar{x}_1, \bar{x}_2) \) is exponentially stable. The following lemma summarizes the results developed up to this point.

**Lemma 2.3.2** (Command Filtered Backstepping). Farrell and Polycarpou [5, Lem. 5.3.2]

Let the control law \( \alpha_1 \) solve the tracking problem for the system

\[
\dot{x}_1 = f_1(x_1) + g_1(x_1)\alpha_1 \quad \text{with} \quad x_1 \in \mathbb{R}^{n-1}
\]
with Lyapunov function $V_1$ satisfying Equation 2.3.80. Then the controller of Equation 2.3.71 to 2.3.75 solves the tracking problem (i.e., guarantees that $x_1(t)$ converges to $y_d(t)$) for the system described by Equation 2.3.66.

This lemma may be applied recursively $n - 1$ times to address a system with $n$ states.

### 2.3.5 Command Filter

This filter was developed in Farrell et al. [6, Appendix A] and provides bounded and continuous command and command-derivative signals. There are two significant features of this filter: the first is the ability to generate derivatives of intermediate control signals, alleviating the need for analytic derivative calculation as mentioned in the preceding sections; the second is magnitude, rate, and bandwidth limiting of state and actuator commands, ensuring that control signals generated are implementable. Some control allocation procedures make provisions for rate and magnitude limiting, so it’s up to the designer whether this filter should be used for actuator commands or not. The nonlinear state space representation of this filter is as follows:

$$
\begin{bmatrix}
  x_c \\
  \dot{x}_c \\
\end{bmatrix}
= \begin{bmatrix}
  q_1 \\
  q_2 \\
\end{bmatrix}
$$

(2.3.82)

$$
\begin{bmatrix}
  \dot{q}_1(t) \\
  \dot{q}_2(t) \\
\end{bmatrix}
= \begin{bmatrix}
  2\zeta\omega_n \left( S_R \left\{ \frac{\omega_n^2}{2\zeta\omega_n} \left[ S_M(x_c^o) - q_1 \right] \right\} - q_2 \right) \\
\end{bmatrix}
$$

(2.3.83)

If $q_1$ and $q_2$ were not used, i.e. using $x_c$ and $\dot{x}_c$ directly, then that the matrix equation would involve the second derivative. With the q variables, one ends up with a first order system which widely supported by mathematical tools. $S_M$ and $S_R$ are the magnitude and rate limit
functions:

\[
S_M = \begin{cases} 
M & \text{if } x \geq M \\
M & \text{if } \|x\| < M \\
-M & \text{if } x \leq -M 
\end{cases}
\]

\[
S_R = \begin{cases} 
R & \text{if } x \geq R \\
R & \text{if } \|x\| < R \\
-R & \text{if } x \leq -R 
\end{cases}
\]

Equation 2.3.82 may be represented in block diagram form as:

As shown in Figure 2.21, this filter generates command \((x_c)\) and command derivative \((\dot{x}_c)\) from an unfiltered command \((x^o_c)\) while enforcing magnitude, rate, and bandwidth constraints. Because we don’t have to calculate analytic derivatives with command filters, the system to be controlled no longer has to be in lower triangular form, but it still must be affine in the control variables, \(\delta_c\) in Chp. 3.

### 2.3.6 Control Allocation

Actuator distribution is crucial to the robustness of the design. The goal is to distribute total control effort amongst a redundant set of actuators; the problem may be formulated as

\[
v(t) = E u(t) \tag{2.3.85}
\]

where \(v\) is a \(k \times 1\) vector of desired values, i.e. virtual control inputs

\[
v = [u^o_{P_c}, u^o_{Q_c}, u^o_{R_c}]^T,
\]
$u$ is an $m \times 1$ vector of true control inputs, i.e. surface deflections

$$u = [\delta_1, \delta_2, \ldots, \delta_6]^T,$$

and $E$ is an $k \times m$ control effectiveness matrix

$$E = B_3 G_3 = \begin{bmatrix} c_3 & 0 & c_4 \\ 0 & c_7 & 0 \\ c_4 & 0 & c_9 \end{bmatrix} \begin{bmatrix} \bar{L}_{\delta_1}, \bar{L}_{\delta_2}, \ldots, \bar{L}_{\delta_6} \\ \bar{M}_{\delta_1}, \bar{M}_{\delta_2}, \ldots, \bar{M}_{\delta_6} \\ \bar{N}_{\delta_1}, \bar{N}_{\delta_2}, \ldots, \bar{N}_{\delta_6} \end{bmatrix},$$

where $c$ terms are defined in Equation 2.1.38 and the $G_3$ matrix terms are dimensional moment control derivatives calculated via look-up tables of aerodynamic data in Figures 2.22 and 2.23.

---

Figure 2.22: Moment Control Derivative Dimensionalization & Control Effectiveness Matrix Concatenation in Simulink

Figure 2.23: Moment Control Deriv LUT
These Lookup Tables are for deflections averaged over \([-6,0) \cup (0,6]\).  

Figure 2.23: Moment Control Derivative Look-up Tables for Control Effectiveness Matrix in Simulink

As these look-up tables imply $G_3$, therefore the control effectiveness matrix $E$, is dynamically updated by simulation-time $\alpha$ and $\beta$ values. It is important to note that deflections are static for look-up table data, otherwise the problem would be much more complex to solve.

Least squares minimization via pseudo inverse of $E$ is the simplest way to solve for $u$

$$u = \text{pinv}(E) v$$  \hspace{1cm} (2.3.86)

If the air vehicle weren’t over-actuated, ie. aileron, elevator, and rudder control surfaces were implemented for roll, pitch, and yaw control respectively, then $E$ would be a $3 \times 3$ square matrix, hence invertible and the pseudo inverse unnecessary.

To jump ahead, for the sake of understanding, here’s what the control allocation subsystem looks like for the ensuing full state flight path controller in Chp. 3:
Figure 2.24: Control Allocation Simulink Subsystem

As this switch scenario in Figure 2.24 depicts, the PINV method is implemented for the case where all actuators are fully operational. In the case where an actuator is intentionally stuck, an optimal control allocation solution via Ola Härkegård’s QCAT toolbox\textsuperscript{22,23} was used, as results in Sec. ?? will show. Since control distribution is not the focus of my thesis and it’s a topic which is thesis worthy on its own, as divided in Härkegård’s doctoral dissertation \cite{17}, advanced methods will not be covered.

\textsuperscript{22} http://research.harkegard.se/qcat/ \quad \textsuperscript{23} http://www.mathworks.com/matlabcentral/fileexchange/4609
**PINV - Simple Control Allocation**

$$\delta_0_c = \text{pseudo_inverse}(B_3 \times G_3) \times v$$

Figure 2.25: PINV Simulink Subsystem within CA Block  **Fig 2.24**

**QCAT - Control Allocation Toolbox**

http://www.mathworks.com/matlabcentral/fileexchange/4609
http://research.harkegard.se/

bborra 3/10/2012: To use Ola Harkegard's Toolbox we need to breakout the mask he has and use the qp_ca_sl.m function. Our control effectiveness matrix, $E = [B_3 \times G_3]$, is a function of $\alpha$ and $\beta$; therefore not entirely static; for now using the $E$ generated by control_effectiveness_matrix.m

Figure 2.26: QCAT Simulink Subsystem within CA Block  **Fig 2.24**
Chapter 3

Derivation of UAV Flight Path Controller

This derivation is based on work by Farrell et al. [23], [6], [5] and Sonneveldt et al. [7]. In the Farrell et al. papers online approximation based, command filtered, backstepping flight path control is developed alongside control laws for three feedback loops. The translational, attitude, and rotational subsets of the state-vector described in Table 2.1 and Equation 2.1.1 are employed in that order, i.e. the state-vector for the flight path controller is:

\[ \mathbf{x} = [\chi \gamma V \mu \alpha \beta P Q R]^T \]  

(3.0.1)

The inputs to the controller are commanded heading \( \chi_c \), climb rate or glide-path angle \( \gamma_c \), airspeed \( V_c \), and the bounded first derivatives of these signals; the subscripts \( c \) here mean commanded. The outer loop controller generates roll angle \( \mu_c \) and angle-of-attack \( \alpha_c \) commands for the middle loop along with a thrust command \( T_c \) that is not passed to succeeding loops. The objective is coordinated flight, hence angle-of-sideslip command \( \beta_c \) is always set to zero. The middle loop generates roll rate \( P_c \), pitch rate \( Q_c \), and yaw rate \( R_c \) commands that serve as inputs to a control allocator which produces actuator deflection.
commands $\delta_1, \delta_2, \cdots, \delta_n$, with $n$ being the number of available actuators.

Design of the flight-path controller herein compliments a publication by Farrell et al. [6], with the exception of adaptive approximation for aerodynamic force and moment coefficients and dynamic control allocation. For the air vehicle this control architecture was closed around an exhaustive set of wind tunnel data was available. The aerodynamics model was also validated through full-scale flight testing with a static control allocation method utilized for initial development. Even for a well characterized model, online approximation would still be advantageous as aerodynamic parameters always contain some degree of uncertainty. The motive of these choices was simplification, the goal being a baseline backstepping controller; these features are independent of the control law design procedure\(^1\) and may be implemented in future phases of development.

**Assumption 3.1 (Command Filtered Backstepping).**

- Full State Feedback  \hspace{2cm} System Dynamics Known\(^*\)
- System may be Non-Triangular\(^*\)  \hspace{1cm} Smooth Feedback Control
- Lyapunov Function Known\(^*\)  \hspace{1cm} Actuator Dynamics Neglected

A major contribution of this thesis is the comprehensible, Simulink\(^2\) driven, graphical block diagram implementation of the control architecture. It offers an intuitive visualization

---

\(^1\) However, adaptive parameter estimation is not independent of the stability analysis.  \(^2\) Simulink - Model Based Design - [http://www.mathworks.com/products/simulink/](http://www.mathworks.com/products/simulink/)
of the procedure and illustrates signal interdependencies that are not self-evident through pages and pages of equations; especially useful for higher order systems. The most succinct block diagram of a backstepping flight controller, with respect to the scope of my literature review, resides in a journal publication by Sonneveldt et al. [7, Figure 1]. This served as a starting point for Simulink modeling:

![Block Diagram of Backstepping Flight Controller](image)

**Figure 3.3: Command Filtered Adaptive Backstepping, Sonneveldt et al. [7]**

After months of initial development and many iterations thereafter, I settled on the block diagram presented in Figure 3.4, which clearly identifies core functions of the controller as well as the two error terms at the summers: tracking and compensated tracking errors. Each subsystem under Figure 3.4 will be revealed and discussed in Chp. 4 along with simulation results; until then we will continue with the derivation at hand.

As cited within Farrell et al. [6], “the main advantages of the approach include ... 2) Lyapunov stability results are provable and 3) state and control constraints can be enforced
Virtual Control Laws

Control Allocation

Vehicle Mdl & Sys Dynamics

zeta-filter / Constraint Effect

Command Filtered Backstepping - Brian Borra

03/28/11 - Control Law Subsystem Development
04/12/11 - Control Allocation and Aero Development
04/27/11 - Aero Block Control Derivative: Force / Mom
05/04/11 - $z$ and $r$ to $z_s$ and $r_s$
05/25/11 - Control Allocation Block - pinv
05/31/11 - Changed the way $l,m,n$ delta were calculated
06/01/11 - Algebraic loop solved with memory block
06/07/11 - migration to library blocks for unit testing
08/26/11 - Implemented QCAT control allocation function
08/30/11 - Changed the way $l,m,n$ delta were calculated
09/02/11 - Mixing matrix integrated into CA block, works well
09/04/11 - Fixed force and moment generation, stability coef to stability forces (D,Y,L)
09/07/11 - Made $z$ gains the same as $k$ gains
09/10/11 - Updated trim state, using FuryB -52, mass now in slugs, removed alpha and delta dimensionalization from respective coefs
09/13/11 - $c$ values fixed, based on Stevens and Lewis Aircraft Dynamics 1992 Pg 80
09/21/11 - Fixed force and moment generation, stability coef to stability forces (D,Y,L)
09/22/11 - $l,m,n$ were wrong, $E$ matrix is now correct and control allocation somewhat working...
10/18/11 - Fixed constraint calculation in CA>angles
11/25/11 - Validated EOM, re-organized block diagram structure
12/02/11 - Mixing matrix integrated into CA block, works well
12/04/11 - Gains updated, following outer loop chi/gam cmds... inner loop gains must be high however
12/17/11 - $l,m,n$ dimensionalized according to McRuer 294-295...
02/03/12 - Fixed $\alpha_o$ calculation in CA>angles
03/11/12 - Model matches all equations in thesis writing, correctly implemented moment control derivatives for $\delta = [8,0,0]$ U (0,0), and simple pinv control allocation working!

Figure 3.4: Command Filtered Backstepping Simulink Model
while maintaining Lyapunov stability.\textsuperscript{3} Robustness to large disturbances, extreme cases being airframe battle damage or actuator failure, could be ensured by implementing an adaptive online approximation and dynamic control allocation. If this were the case, then conceivably the control architecture could be applied to variants of the base vehicle configuration without controller redesign or tuning. This implies that the control designer would need less input from the aerodynamicist, hence less analysis or wind-tunnel testing resulting in a lower-fidelity aero model. Consequently this could increase cost savings and decrease development time, \textit{if} the overall investment in adaptive backstepping development was cheaper than an exhaustive aerodynamic evaluation.

The following sections use a stability result initially developed in Farrell et al. [23], formalized in a textbook by Farrell and Polycarpou [5, Sec. 5.3.3], and summarized here in Sec. 2.3.4.3, Lemma 2.3.2. The block vector representation of equations within Farrell et al. [6] are expanded with respect to each state to make the control law derivation easier to understand. \textbf{Dynamics will be grouped into sub-functions consisting of known variables, }$F_x$, \textbf{and partially or completely unknown variables, }$f_x$, \textbf{even though online approximation is not use; this was done for the sake of scalability and it’s cleaner to write in this fashion.} It is assumed that there are no parametric uncertainties in the aerodynamic coefficients.

\subsection{3.1 Outer Loop}
To begin, flight-path and airspeed \textbf{dynamics for the outer loop,} Equations 2.1.39a, 2.1.39b, and 2.1.39c respectively, are grouped into sub-functions of their partially or completely unknown $f$, known $F$, and control signals $u$:

\textsuperscript{3} If online approximation were implemented then: “1) the aerodynamic force and moment models are automatically adjusted to accommodate changes to the aerodynamic properties of the vehicle”
\[ \dot{x} = f_x + F_x + u_x \] (3.1.1)

\begin{align*}
    f_x &= \frac{\cos \mu}{mV \cos \gamma} (D \sin \beta + Y \cos \beta) \\
    F_x &= \frac{1}{mV \cos \gamma} (-T \cos \alpha \sin \beta \cos \mu) \\
    u_x &= \frac{\sin \mu}{mV \cos \gamma} (L + T \sin \alpha)
\end{align*}

\[ \dot{\gamma} = f_\gamma + F_\gamma + u_\gamma \] (3.1.5)

\begin{align*}
    f_\gamma &= \frac{\sin \mu}{mV} (-D \sin \beta - Y \cos \beta) \\
    F_\gamma &= \frac{1}{mV} (T \cos \alpha \sin \beta \sin \mu - mg \cos \gamma) \\
    u_\gamma &= \frac{\cos \mu}{mV} (L + T \sin \alpha)
\end{align*}

\[ \dot{V} = f_V + F_V + u_V \] (3.1.9)

\begin{align*}
    f_V &= \frac{1}{m} (-D \cos \beta + Y \sin \beta) \\
    F_V &= -g \sin \gamma \\
    u_V &= \frac{1}{m} (T \cos \alpha \cos \beta)
\end{align*}

In block vector notation, Equations 3.1.1, 3.1.5, and 3.1.9 may be combined, as defined in Farrell et al. [6] and Farrell and Polycarpou [5][Sec. 8.3.1]:

\[ \dot{z}_1 = \bar{A}_1 f_1 + F_1 + G_1(\mu_1, x) \] (3.1.13)
with \( z_1 = [\chi, \gamma, V]^T \), \( f_1 = [D, Y, L]^T \) and

\[
\tilde{A}_1 = \begin{bmatrix}
\sin \beta \cos \gamma & \cos \beta \cos \mu / \cos \gamma & 0 \\
- \sin \beta \sin \mu & - \cos \beta \sin \mu & 0 \\
-V \cos \beta & V \sin \beta & 0
\end{bmatrix}
\] (3.1.14)

\[
\tilde{F}_1 = \begin{bmatrix}
(-T \cos \alpha \sin \beta \cos \mu) / (mV \cos \gamma) \\
(T \cos \alpha \sin \beta \sin \mu - mg \cos \gamma) / (mV) \\
-g \sin \gamma
\end{bmatrix}
\] (3.1.15)

\[
G_1(\mu_1, x) = \begin{bmatrix}
u_\chi \\
u_\gamma \\
u_V
\end{bmatrix} = \begin{bmatrix}
(g(\alpha, x) \sin \mu) / (mV \cos \gamma) \\
(g(\alpha, x) \cos \mu) / (mV) \\
(T \cos \beta \cos \mu) / (m)
\end{bmatrix}
\] (3.1.16)

where \( g(\alpha, x) = L(\alpha, x) + T \sin \alpha \) and \( \mu_1 = [\mu, \alpha, T]^T \).

### 3.1.1 Flight Path Angle Control

Wind-axis command signals \( \mu_c \) and \( \alpha_c \) are generated internally in the outer loop of the controller, as formed in this sub-section; \( \beta_c \) is set to zero, as we wish to fly coordinated. These and derivatives of these signals drive middle loop virtual control laws. Remember that this is a feedback control architecture, do not confuse state **commands** with state **dynamics**: state commands, \( x_c \), are fed top-down in the controller, as Figure 3.1 illustrates, while state dynamics, \( x \), are fed bottom-up, or back with respect to System 2.1.39. **Tracking errors** are defined as

\[
\tilde{\chi} = \chi - x_c \quad , \quad \tilde{\gamma} = \gamma - \gamma_c
\] (3.1.17)

and additionally, **compensated tracking errors** defined as

\[
\bar{\chi} = \tilde{\chi} - \zeta_\chi \quad , \quad \bar{\gamma} = \tilde{\gamma} - \zeta_\gamma
\] (3.1.18)

Tracking errors \((\tilde{\chi}, \tilde{\gamma})\) account for deviation away from the commanded and actual state signals and compensated tracking errors \((\bar{\chi}, \bar{\gamma})\) are tracking errors that compensate for a
state or actuator signal that is limited by rate, magnitude, and/or bandwidth constraints; \( \zeta \) terms will be defined shortly.

The immediate goal is to compute unfiltered wind-axis angle commands \( \mu^o_c \) and \( \alpha^o_c \). We begin by developing stabilizing functions\(^4\) that cancel undesirable system dynamics and incorporate tracking terms; we choose the following **virtual control laws**, which are placed in the ‘virtual control law’ simulink subsystem,

\[
\begin{align*}
\dot{u}_x^o &= -f_x - F_x + \dot{\chi}_c - k_x \ddot{x} \\
\dot{u}_\gamma^o &= -f_\gamma - F_\gamma + \dot{\gamma}_c - k_\gamma \ddot{\gamma}
\end{align*}
\]  

(3.1.19)

(3.1.20)

where the \( k \) terms are gains proportional to the tracking error. Note that these virtual control laws transform their respective systems, 3.1.1 and 3.1.5, into a tracking problem, recall Equation 2.3.69.

**Unfiltered control laws** are equivalent to their counterparts in Equations 3.1.4 and 3.1.8, placed in the ‘control allocation’ simulink subsystem, and defined as

\[
\begin{align*}
\dot{u}_x^o &= \frac{g(\alpha^o_c)}{mV \cos \gamma} \\
\dot{u}_\gamma^o &= \frac{g(\alpha^o_c)}{mV}
\end{align*}
\]  

(3.1.21)

(3.1.22)

where

\[
g(\alpha^o_c) = L(\alpha^o_c) + T \sin \alpha^o_c
\]  

(3.1.23)

Critical in the definition of \( g(\alpha^o_c) \) is recalling/recognizing that lift, not just the thrust term, is dependent on angle-of-attack; this interdependency was not transparent in Equations 3.1.4 and 3.1.8.

Since terms on the right hand side of Equations 3.1.19 – 3.1.20, hence \( \dot{u}_x^o \) and \( \dot{u}_\gamma^o \), are known, unfiltered wind axis commands \( \alpha^o_c \) and \( \mu^o_c \) may be solved for in Equations 3.1.21 – 3.1.22 with a Cartesian based solution technique discussed in Section 3.1.1.1.

\(^4\) \( u^o_x \) is substituted for the initial variable choice \( \alpha^o_{x_c} \) in Backstepping Section 2.3.4 to avoid confusion with angle-of-attack.
Once $\mu^o_c$ and $\alpha^o_c$ are calculated, they are each command filtered, as introduced in Sec. 2.3.5, to produce “magnitude, rate, and bandwidth-limited signals” $\mu_c, \dot{\mu}_c, \alpha_c, \dot{\alpha}_c$. These filtered commands will then be used to generate achievable control laws

\[
\begin{align*}
  u_{\chi_c} & \equiv \frac{g(\alpha_c) \sin \mu_c}{mV \cos \gamma} \\
  u_{\gamma_c} & \equiv \frac{g(\alpha_c) \cos \mu_c}{mV}
\end{align*}
\]  

(3.1.24)

(3.1.25)

which are fed into the $\zeta_x$–filters

\[
\begin{align*}
  \dot{\zeta}_\chi &= -k_\chi \zeta_\chi + (u_{\chi_c} - u^o_{\chi_c}) \\
  \dot{\zeta}_\gamma &= -k_\gamma \zeta_\gamma + (u_{\gamma_c} - u^o_{\gamma_c})
\end{align*}
\]  

(3.1.26)

(3.1.27)

that compensate for the tracking errors $\tilde{\chi}$ and $\tilde{\gamma}$ for the effect of any differences between the desired $(u^o_{\chi_c}, u^o_{\gamma_c})$ and achievable $(u_{\chi_c}, u_{\gamma_c})$ control signals.

### 3.1.1.1 Selection of $\alpha^o_c$ and $\mu^o_c$ Commands

Defining $X$ and $Y$ such that Equations 3.1.24 and 3.1.27 may be written as

\[
\begin{align*}
  X & \equiv mV \cos \gamma u_\chi = g(\alpha^o_c) \sin(\mu^o_c) \\
  Y & \equiv mV u_\gamma = g(\alpha^o_c) \cos(\mu^o_c)
\end{align*}
\]  

(3.1.28)

(3.1.29)

allows us to interpret a point $(X, Y)$ in a Cartesian, or rectangular, coordinate system with positive or negative radius $g(\alpha^o_c)$, corresponding to positive and negative lift, and an angle $\mu^o_c$ relative to the positive Y-axis.

Since there are two possible solutions for $g(\alpha^o_c)$, as Figure 3.5 depicts, safeguards must be implemented to prevent switching among either option outside of a small range near 0° of bank angle $\mu^o_c$; otherwise, a sign change – $\mu^o_c$ changing by 180° – would cause the air vehicle to make an aggressive diving turn, potentially performing the maneuver inverted with bank angle greater than 90°. To reiterate, “when choosing $(\mu^o_c, \alpha^o_c)$ to satisfy Equation 3.1.21 and 3.1.22, the designer should only allow $g(\alpha^o_c)$ to reverse its sign when [the bank angle virtual
control law] $u_{x_c}$ is near zero.” [5].

$$g(X, Y) = \sqrt{X^2 + Y^2} = \sqrt{(mV \cos \gamma u_\chi)^2 + (mV u_r)^2}$$  \hspace{0.5cm} (3.1.30)

$$\mu^o_c = \text{atan2}(Y, X) = \text{atan2}[(mV u_r), (mV \cos \gamma u_\chi)]$$  \hspace{0.5cm} (3.1.31)

$$\alpha^o_c = \text{interp1}(g(\alpha), \alpha, g(X, Y))$$  \hspace{0.5cm} (3.1.32)

where ‘atan2’ is the four quadrant inverse tangent and ‘interp1’ is 1-D linear interpolation, both of which are inherent to Matlab. Vectors $\alpha$ and $g(\alpha)$ are angles-of-attack and pre-calculated signed radii corresponding each $\alpha$ in Figure 3.5, which is characterized by Equation 3.1.23; given a $g(X, Y)$ we may interpolate using the aforementioned vectors to solve for $\alpha^o_c$. Once $\mu^o_c$ and $\alpha^o_c$ have been specified, then $T^o_c$ may be directly solved for, upcoming in Equation 3.1.37.

\footnote{atan2(Y/X) is intentional, not atan2(X/Y) because we defined 0° to align with the positive Y-axis.}
3.1.2 Airspeed Control

$T_c$ is generated internally by the controller, as derived in this sub-section; airspeed, $V$, in the outer-loop is managed via thrust, $T_c$. Airspeed tracking error is defined as

$$\dot{V} = V - V_c$$ \hspace{1cm} (3.1.33)

and additionally, compensated tracking error defined as

$$\dot{V} = \dot{V} - \zeta_V$$ \hspace{1cm} (3.1.34)

Tracking error ($\dot{V}$) accounts for deviation away from the commanded and actual state signal and compensated tracking error ($\dot{V}$) is a tracking error that compensates for a state or actuator signal that is limited by rate, magnitude, and/or bandwidth constraints; $\zeta_V$ terms will be defined shortly.

The immediate goal is to compute unfiltered thrust command $T_o$. We begin by developing a stabilizing function that cancels undesirable system dynamics and incorporates tracking terms; we choose the following virtual control law, placed in the ‘virtual control law’ simulink subsystem,

$$u_{V^c} = -f_V - F_V + \dot{V}_c - k_V \dot{V}$$ \hspace{1cm} (3.1.35)

where the $k_V$ term is a gain proportional to the tracking error. Note that this virtual control law transforms its respective system, 3.1.9, into a tracking problem, recall Equation 2.3.69.

The unfiltered control law is equivalent to its counterpart in Equation 3.1.12, placed in the ‘control allocation’ simulink subsystem, and defined as

$$u_{V^c} = \frac{T_o \cos \beta \cos \alpha^o_c}{m}$$ \hspace{1cm} (3.1.36)

Since terms on the right hand side of Equation 3.1.35, hence $u_{V^c}$, are/are known, the unfiltered thrust command $T_o$ may be solved for in Equation 3.1.36 and filtered by a simple
saturation and rate limiter directly after calculation. Recall that by the previous Section 3.1.1, \( \alpha^o_c \) is available.

In the current implementation \( T^o_c \) is not passed through a command filter. If one were to choose to use the command filter, \( T^o_c \) would be passed through the filter introduced in Sec. 2.3.5, to produce “magnitude, rate, and bandwidth-limited signals” \( T_c \) and \( \dot{T}_c \). The derivative control signal \( \dot{T}_c \) is not used, hence the reason why a command filter is not necessary for this loop. This filtered command could then be used to generate the achievable control law

\[
u_{V_c} \equiv \frac{T_c \cos \beta \cos \alpha_c}{m}
\]

which is fed into the \( \zeta_V \)-filter

\[
\dot{\zeta}_V = -k_V \zeta_V + (u_{V_c} - u_{V_c}^o)
\]

that compensates for the tracking error \( \tilde{V} \) for the effect of any differences between the desired \( u_{V_c}^o \) and achievable \( u_{V_c} \) control signals. If this is bypassed, by avoiding the command filter for this loop, the error will be zero, i.e. \( u_{V_c} = u_{V_c}^o \).

As previously handled with system dynamics, the equations used to solve for \( \mu_c^o, \alpha_c^o, \) and \( T_c^o \) in Sections 3.1.1 and 3.1.2 may be condensed into block vector notation, as defined in Farrell et al. [6] and Farrell and Polycarpou [5][8.3.1]:

\[
G_1(\mu_1^o, x) = -\bar{A}_1 f_1 - F_1 - K_1 \tilde{z}_1 + \dot{z}_1
\]

with \( G_1 = [u_{\chi c}^o, u_{\gamma c}^o, u_{V c}^o]^T \equiv [3.1.21, 3.1.22, 3.1.36]^T, \mu_1^o = [\mu_c^o, \alpha_c^o, T_c^o]^T, \bar{A}_1 = \text{Equation 3.1.14}, f_1 = [D, Y, L]^T, F_1 = \text{Equation 3.1.15}, K_1 \) is a positive definite diagonal 3x3 matrix, \( \text{diag}(k_\chi, k_\gamma, k_V) \), \( \tilde{z}_1 = z_1 - z_{1c} \), and \( z_1 = [\chi, \gamma, V]^T \). The goal is to select \( \mu_1^o \) so that Equation 3.1.39 holds, then command filter \( z_{2c}^o \) to produce the signals \( z_{2c} \) and \( \dot{z}_{2c} \).

Assuming that the solution \( \mu_1^o \) to Equation 3.1.39 has been found, define

\[
z_{2c}^o = \mu_1^o = [\mu_c^o, \alpha_c^o, \beta_c^o]
\]
and the $\zeta_1$-filter in block vector notation be defined as

$$\dot{\zeta}_1 = -K_1 \zeta_1 + (G(z_2, x) - G(z_2^c, x)) \quad (3.1.41)$$

where $z_2 = [\mu, \alpha, \beta]^T$.

### 3.2 Middle Loop

Wind-axis dynamics for the middle loop, Equations 2.1.39d, 2.1.39e, and 2.1.39f respectively, are now grouped into sub-functions of their partially or completely unknown $f$, known $F$, and control signals $u$:

**Bank Angle Dynamics**

$$\dot{\mu} = f_\mu + F_\mu + u_\mu \quad (3.2.1)$$

$$f_\mu = \frac{1}{mV} \left[D \sin \beta \tan \gamma \cos \mu + Y \cos \beta \tan \gamma \cos \mu + L (\tan \beta + \tan \gamma \sin \mu)\right] \quad (3.2.2)$$

$$F_\mu = \frac{1}{mV} \left[T (\sin \alpha \tan \gamma \sin \mu + \sin \alpha \tan \beta - \cos \alpha \sin \beta \tan \gamma \cos \mu) - mg \tan \beta \cos \gamma \cos \mu\right] \quad (3.2.3)$$

$$u_\mu = \frac{P \cos \alpha + R \sin \alpha}{\cos \beta} \equiv \frac{P_s}{\cos \beta} \quad (3.2.4)$$

**Angle-of-Attack Dynamics**

$$\dot{\alpha} = f_\alpha + F_\alpha + u_\alpha \quad (3.2.5)$$

$$f_\alpha = -\frac{L}{mV \cos \beta} \quad (3.2.6)$$

$$F_\alpha = \frac{1}{mV \cos \beta} \left[-T \sin \alpha + mg \cos \gamma \cos \mu\right] \quad (3.2.7)$$

$$u_\alpha = Q - \tan \beta (P \cos \alpha + R \sin \alpha) \equiv Q - P_s \tan \beta \quad (3.2.8)$$
\[ \dot{\beta} = f_\beta + F_\beta + u_\beta \] (3.2.9)

\[ f_\beta = \frac{1}{mV} (D \sin \beta + Y \cos \beta) \] (3.2.10)

\[ F_\beta = \frac{1}{mV} - T \sin \beta \cos \alpha + mg \cos \gamma \sin \mu \] (3.2.11)

\[ u_\beta = P \sin \alpha - R \cos \alpha \equiv -R_s \] (3.2.12)

In block vector notation Equations 3.2.1, 3.2.5, and 3.2.9 may be combined, as defined in Farrell et al. [6] and Farrell and Polycarpou [5][8.3.2]:

\[ \dot{z}_2 = A_2 f_1 + F_2 + B_2 \mu_2 \] (3.2.13)

with \( z_2 = [\mu, \alpha, \beta]^T \), \( f_1 = [D, Y, L]^T \) and

\[
A_2 = \frac{1}{mV} \begin{bmatrix}
\sin \beta \cos \mu \tan \gamma & \cos \beta \cos \mu \tan \gamma & \tan \beta + \tan \gamma \sin \mu \\
0 & 0 & -1/ \cos \beta \\
\sin \beta & \cos \beta & 0
\end{bmatrix}
\] (3.2.14)

\[
F_2 = \frac{1}{mV} \begin{bmatrix}
T(\sin \alpha \tan \gamma \sin \mu + \sin \alpha \tan \beta - \cos \alpha \tan \gamma \cos \mu \sin \beta) - mg \cos \gamma \cos \mu \tan \beta \\
(-T \sin \alpha + mg \cos \gamma \cos \mu)/ (\cos \beta) \\
-T \sin \beta \cos \alpha + mg \cos \gamma \sin \mu
\end{bmatrix}
\] (3.2.15)

\[
B_2 = \begin{bmatrix}
\cos \alpha/ \cos \beta & 0 & \sin \alpha/ \cos \beta \\
-\cos \alpha \tan \beta & 1 & -\sin \alpha \tan \beta \\
\sin \alpha & 0 & -\cos \alpha
\end{bmatrix}
\] (3.2.16)

where \( \mu_2 = [P, Q, R]^T \).
3.2.1 Wind-Axis Angle Control

Body-axis command signals $P_c$, $Q_c$ and $R_c$ are generated internally in the middle loop of the controller, as formed in this sub-section. These and derivatives of the body-axis command signals drive inner loop virtual control laws. Tracking errors are defined as

$$\tilde{\mu} = \mu - \mu_c, \quad \tilde{\alpha} = \alpha - \alpha_c, \quad \tilde{\beta} = \beta - \beta_c \quad (3.2.17)$$

and additionally, compensated tracking errors defined as

$$\bar{\mu} = \tilde{\mu} - \zeta_{\mu}, \quad \bar{\alpha} = \tilde{\alpha} - \zeta_{\alpha}, \quad \bar{\beta} = \tilde{\beta} - \zeta_{\beta} \quad (3.2.18)$$

Tracking errors ($\tilde{\mu}, \tilde{\alpha},$ and $\tilde{\beta}$) account for deviation away from the commanded and actual state signals and compensated tracking errors ($\bar{\mu}, \bar{\alpha},$ and $\bar{\beta}$) are tracking errors that compensate for a state or actuator signal that is limited by rate, magnitude, and/or bandwidth constraints; $\zeta_x$ terms will be defined shortly.

The immediate goal is to compute unfiltered body-axis rate commands, $P_c^o$, $Q_c^o$, and $R_c^o$. We begin by developing stabilizing functions that cancel undesirable system dynamics and incorporate tracking terms; we choose the following virtual control laws, which are placed in the ‘virtual control law’ simulink subsystem,

$$u_{\mu_c}^o = -f_\mu - F_\mu + \dot{\mu}_c - k_\mu \tilde{\mu} \quad (3.2.19)$$
$$u_{\alpha_c}^o = -f_\alpha - F_\alpha + \dot{\alpha}_c - k_\alpha \tilde{\alpha} \quad (3.2.20)$$
$$u_{\beta_c}^o = -f_\beta - F_\beta + \dot{\beta}_c - k_\beta \tilde{\beta} \quad (3.2.21)$$

where the $k$ terms are gains proportional to the tracking error. Note that these virtual control laws transform their respective systems, 3.2.1, 3.2.5 and 3.2.9, into a tracking problem, recall Equation 2.3.69.

Unfiltered control laws are equivalent to their counterparts in Equations 3.2.4, 3.2.8,
and 3.2.12, placed in the ‘control allocation’ simulink subsystem, and defined as

\[ u_{\mu c}^o \equiv P_c^o \left( \frac{\cos \alpha}{\cos \beta} \right) + R_c^o \left( \frac{\sin \alpha}{\cos \beta} \right) = \frac{P_{sc}}{\cos \beta} \]  
\(3.2.22\)

\[ u_{\alpha c}^o \equiv Q_c^o - P_c^o (\cos \alpha \tan \beta) - R_c^o (\sin \alpha \tan \beta) = Q_{sc}^o - P_{sc}^o \tan \beta \]  
\(3.2.23\)

\[ u_{\beta c}^o \equiv P_c^o \sin \alpha - R_c^o \cos \alpha = -R_{sc}^o \]  
\(3.2.24\)

Since terms on the right hand side of Equations 3.2.19–3.2.21, hence \( u_{\mu c}^o, u_{\alpha c}^o \) and \( u_{\beta c}^o \), are known, unfiltered wind axis commands \( P_{sc}^o, Q_{sc}^o, \) and \( R_{sc}^o \) may be solved for in Equations 3.2.22–3.2.24 directly: First transform unfiltered control law equations to a matrix representation

\[
\begin{bmatrix}
  u_{\mu c}^o \\
  u_{\alpha c}^o \\
  u_{\beta c}^o
\end{bmatrix}
\equiv
\begin{bmatrix}
  \frac{\cos \alpha}{\cos \beta} & 0 & \frac{\sin \alpha}{\cos \beta} \\
  -\cos \alpha \tan \beta & 1 & -\sin \alpha \tan \beta \\
  \sin \alpha & 0 & -\cos \alpha
\end{bmatrix}
\begin{bmatrix}
  P_c^o \\
  Q_c^o \\
  R_c^o
\end{bmatrix}
= B_2 \mu_{2c}^o
\]  
\(3.2.25\)

and solve for \( \mu_{2c}^o = [P_{sc}^o, Q_{sc}^o, R_{sc}^o]^T \) by taking the inverse of \( B_2 \) and pre-multiplying by the known virtual control law vector

\[
\begin{bmatrix}
  P_c^o \\
  Q_c^o \\
  R_c^o
\end{bmatrix}
= \begin{bmatrix}
  \cos \alpha \cos \beta & 0 & \sin \alpha \\
  \sin \beta & 1 & 0 \\
  \cos \beta \sin \alpha & 0 & -\cos \alpha
\end{bmatrix}
\begin{bmatrix}
  u_{\mu c}^o \\
  u_{\alpha c}^o \\
  u_{\beta c}^o
\end{bmatrix}
\]  
\(3.2.26\)

Once \( P_{sc}^o, Q_{sc}^o, \) and \( R_{sc}^o \) are calculated, they are each command filtered, as introduced in Sec. 2.3.5, to produce “magnitude, rate, and bandwidth-limited signals” \( P_{c}, \dot{P}_{c}, Q_{c}, \dot{Q}_{c} \) and \( R_{c}, \dot{R}_{c} \). These filtered commands will then be used to generate achievable control laws

\[ u_{\mu c} \equiv P_{c} \left( \frac{\cos \alpha}{\cos \beta} \right) + R_{c} \left( \frac{\sin \alpha}{\cos \beta} \right) = \frac{P_{sc}}{\cos \beta} \]  
\(3.2.27\)

\[ u_{\alpha c} \equiv Q_{c} - P_{c} (\cos \alpha \tan \beta) - R_{c} (\sin \alpha \tan \beta) = Q_{sc} - P_{sc} \tan \beta \]  
\(3.2.28\)

\[ u_{\beta c} \equiv P_{c} \sin \alpha - R_{c} \cos \alpha = -R_{sc} \]  
\(3.2.29\)
which are fed into the $\zeta_x$–filters

\[
\dot{\zeta}_\mu = -k_\mu \zeta_\mu + (u_{\mu c} - \bar{u}_{\mu c})
\] (3.2.30)

\[
\dot{\zeta}_\alpha = -k_\alpha \zeta_\alpha + (u_{\alpha c} - \bar{u}_{\alpha c})
\] (3.2.31)

\[
\dot{\zeta}_\beta = -k_\beta \zeta_\beta + (u_{\beta c} - \bar{u}_{\beta c})
\] (3.2.32)

that compensate for the tracking errors $\bar{\mu}$, $\bar{\alpha}$, and $\bar{\beta}$ for the effect of any differences between the desired $(u_{\mu c}^0, u_{\alpha c}^0, u_{\beta c}^0)$ and achievable $(u_{\mu c}, u_{\alpha c}, u_{\beta c})$ control signals.

As previously handled with system dynamics, the equations used to solve for $P_c^o$, $Q_c^o$, and $R_c^o$ in this section may be condensed into block vector notation, as defined by Farrell et al. [6] and Farrell and Polycarpou [5][8.3.2]:

\[
B_{2\mu_2} = -A_{2f} - F_2 - K_2 \tilde{z}_2 + \dot{z}_2
\] (3.2.33)

with $B_{2\mu_2} = [u_{\mu c}, u_{\alpha c}^0, u_{\beta c}^0]^T \equiv [3.2.22, 3.2.23, 3.2.24]^T$, $\mu_2^0 = [P_c^o, Q_c^o, R_c^o]^T$, $A_2 = \text{Equation 3.2.14}$, $f_1 = [D, Y, L]^T$, $F_2 = \text{Equation 3.2.15}$, $K_2$ is a positive definite diagonal 3x3 matrix, $\text{diag}(k_\mu, k_\alpha, k_\beta)$, $\tilde{z}_2 = z_2 - z_{2c}$, and $z_2 = [\mu, \alpha, \beta]^T$. The goal is to select $\mu_2^0$ so that Equation 3.2.33 holds, then command filter $z_3^o$ to produce the signals $z_{3c}$ and $\dot{z}_{3c}$.

Assuming that the solution $\mu_2^0$ to Equation 3.2.33 has been found, define

\[
z_3^o = \mu_2^0 - \zeta_3
\] (3.2.34)

where $\zeta_3$ will be defined in the next section, and the $\zeta_2$–filter in block vector notation be defined as

\[
\dot{\zeta}_2 = -K_2 \zeta_2 + B_2 (z_{3c} - z_3^o)
\] (3.2.35)

where $z_{3c} = [P_c, Q_c, R_c]^T$.

### 3.3 Inner Loop

Body-axis dynamics for the inner loop, Equations 2.1.39g, 2.1.39h, and 2.1.39i respectively, are now grouped into sub-functions of their partially or completely unknown $f$, known
\( F \), and control signals \( u \):

\[ \dot{P} = f_P + F_P + u_P \quad (3.3.1) \]

\[ f_P = c_3 \bar{L}' + c_4 \bar{N}' \quad (3.3.2) \]

\[ F_P = (c_1 R + c_2 P) Q \quad (3.3.3) \]

\[ u_P = c_3 \sum_{i=1}^{6} \bar{L}_\delta \delta_i + c_4 \sum_{i=1}^{6} \bar{N}_\delta \delta_i \quad (3.3.4) \]

where the rolling and yawing moments were decomposed into moments due to aerodynamics and deflections as \( \bar{L} = \bar{L}' + \sum_{i=1}^{6} \bar{L}_\delta \delta_i \) and \( \bar{N} = \bar{N}' + \sum_{i=1}^{6} \bar{N}_\delta \delta_i \).

\[ \dot{Q} = f_Q + F_Q + u_Q \quad (3.3.5) \]

\[ f_Q = c_7 \bar{M}' \quad (3.3.6) \]

\[ F_Q = c_5 PR - c_6 \left( P^2 - R^2 \right) \quad (3.3.7) \]

\[ u_Q = c_7 \sum_{i=1}^{6} \bar{M}_\delta \delta_i \quad (3.3.8) \]

where the pitching moment was decomposed into moment due to aerodynamics and deflections as \( \bar{M} = \bar{M}' + \sum_{i=1}^{6} \bar{M}_\delta \delta_i \).

\[ \dot{R} = f_R + F_R + u_R \quad (3.3.9) \]

\[ f_R = c_4 \bar{L}' + c_9 \bar{N}' \quad (3.3.10) \]

\[ F_R = (c_8 P - c_2 R) Q \quad (3.3.11) \]

\[ u_R = c_4 \sum_{i=1}^{6} \bar{L}_\delta \delta_i + c_9 \sum_{i=1}^{6} \bar{N}_\delta \delta_i \quad (3.3.12) \]
where the rolling and yawing moments were decomposed as in Equation 3.3.2.

In block vector notation Equations 3.3.1, 3.3.5, and 3.3.9 may be combined, as defined in Farrell et al. [6] and Farrell and Polycarpou [5][8.3.3]:

\[
\dot{z}_3 = A_3 f_3 + F_3 + B_3 G_3 \delta 
\]

(3.3.13)

with \( z_3 = [P, Q, R]^T \), \( f_3 = [\bar{L}', \bar{M}', \bar{N}']^T \),

\[
A_3 = B_3 = \begin{bmatrix}
c_3 & 0 & c_4 \\
0 & c_7 & 0 \\
c_4 & 0 & c_9
\end{bmatrix}
\]

(3.3.14)

where \( c \) terms are not elements of the moment of inertia matrix, recall Equation 2.1.38 which defines \( c \)-values as a combination of moment of inertia matrix elements,

\[
F_3 = \begin{bmatrix}
(c_1 R + c_2 P)Q \\
c_5 P R - c_6 (P^2 - R^2) \\
(c_8 P - c_2 R)Q
\end{bmatrix}
\]

(3.3.15)

and

\[
G_3 = \begin{bmatrix}
\bar{L}_{\delta_1}, \cdots, \bar{L}_{\delta_6} \\
\bar{M}_{\delta_1}, \cdots, \bar{M}_{\delta_6} \\
\bar{N}_{\delta_1}, \cdots, \bar{N}_{\delta_6}
\end{bmatrix}
\]

(3.3.16)

### 3.3.1 Body-Axis Angular Rate Control

Commanded deflection signals \( \delta_{1c}, \cdots, \delta_{6c} \) are generated internally in the inner loop of the controller, as formed in this sub-section. Tracking errors are defined as

\[
\tilde{P} = P - P_c, \quad \tilde{Q} = Q - Q_c, \quad \tilde{R} = R - R_c
\]

(3.3.17)

and additionally, compensated tracking errors defined as

\[
P = \tilde{P} - \zeta_P, \quad Q = \tilde{Q} - \zeta_Q, \quad R = \tilde{R} - \zeta_R
\]

(3.3.18)
Tracking errors \((\tilde{P}, \tilde{Q}, \text{and } \tilde{R})\) account for deviation away from the commanded and actual state signals and compensated tracking errors \((\bar{P}, \bar{Q}, \text{and } \bar{R})\) are tracking errors that \emph{compensate} for a state or actuator signal that is limited by rate, magnitude, and/or bandwidth constraints; \(\zeta_x\) terms will be defined shortly.

The immediate goal is to compute unfiltered deflection commands, \(\delta_{1c}, \cdots, \delta_{6c}\). We begin by developing stabilizing functions that cancel undesirable system dynamics and incorporate tracking terms; we choose the following \textbf{virtual control laws}, which are placed in the ‘virtual control law’ simulink subsystem,

\begin{align*}
    u_{Pc}^o &= -f_P - F_P + \dot{P}_c - k_P \tilde{P} - x_P \quad (3.3.19) \\
    u_{Qc}^o &= -f_Q - F_Q + \dot{Q}_c - k_Q \tilde{Q} - x_Q \quad (3.3.20) \\
    u_{Rc}^o &= -f_R - F_R + \dot{R}_c - k_R \tilde{R} - x_R \quad (3.3.21)
\end{align*}

where the \(k\) terms are gains proportional to the tracking error and \(x\) terms are

\begin{align*}
    x_P &= \mu \frac{\cos \alpha}{\cos \beta} - \bar{\alpha} \cos \alpha \tan \beta - \bar{\beta} \sin \alpha \quad (3.3.22) \\
    x_Q &= \bar{\alpha} \quad (3.3.23) \\
    x_R &= \mu \frac{\sin \alpha}{\cos \beta} - \bar{\alpha} \sin \alpha \tan \beta - \bar{\beta} \cos \alpha \quad (3.3.24)
\end{align*}

which allows the subsequent Lyapunov stability analysis in \textbf{section 3.4} to stabilize the system for the desired trajectory. Note that these virtual control laws transform their respective systems, 3.3.1, 3.3.5 and 3.3.9, into a tracking problem, recall Equation 2.3.69.

\textbf{Unfiltered control laws} are equivalent to their counterparts in Equations 3.3.4, 3.3.8, and 3.3.12, placed in the ‘control allocation’ simulink subsystem, and defined as

\begin{align*}
    u_{Pc}^o &\equiv c_3 \sum_{i=1}^{6} \bar{L}_i \delta_{ic}^o + c_4 \sum_{i=1}^{6} \bar{N}_i \delta_{ic}^o \quad (3.3.25) \\
    u_{Qc}^o &\equiv c_7 \sum_{i=1}^{6} \bar{M}_i \delta_{ic}^o \quad (3.3.26) \\
    u_{Rc}^o &\equiv c_4 \sum_{i=1}^{6} \bar{L}_i \delta_{ic}^o + c_9 \sum_{i=1}^{6} \bar{N}_i \delta_{ic}^o \quad (3.3.27)
\end{align*}
Since terms on the right hand side of Equations 3.3.19–3.3.21, hence $u^o_P$, $u^o_Q$, and $u^o_R$, are known, unfiltered deflection commands $\delta^o_1, \ldots, \delta^o_6$ may be solved for in Equations 3.3.25–3.3.27 directly with some form of actuator distribution.

Once $\delta^o_1, \ldots, \delta^o_6$ are calculated, they are each command filtered, as introduced in Sec. 2.3.5, to produce “magnitude, rate, and bandwidth-limited signals” $\delta_1, \ldots, \delta_6$. These filtered commands will then be used to generate achievable control laws

\[
\begin{align*}
 u_P & \equiv c_3 \sum_{i=1}^{6} \bar{L}_i \delta_i + c_4 \sum_{i=1}^{6} \bar{N}_i \delta_i \\
 u_Q & \equiv c_7 \sum_{i=1}^{6} \bar{M}_i \delta_i \\
 u_R & \equiv c_3 \sum_{i=1}^{6} \bar{L}_i \delta_i + c_9 \sum_{i=1}^{6} \bar{N}_i \delta_i
\end{align*}
\]  

(3.3.28)  

(3.3.29)  

(3.3.30)

which are fed into the $\zeta$–filters

\[
\begin{align*}
 \dot{\zeta}_P & = -k_P \zeta_P + (u_P - u^o_P) \\
 \dot{\zeta}_Q & = -k_Q \zeta_Q + (u_Q - u^o_Q) \\
 \dot{\zeta}_R & = -k_R \zeta_R + (u_R - u^o_R)
\end{align*}
\]  

(3.3.31)  

(3.3.32)  

(3.3.33)

that compensate for the tracking errors $\tilde{P}, \tilde{Q}$, and $\tilde{R}$ for the effect of any differences between the desired $(u^o_P, u^o_Q, u^o_R)$ and achievable $(u_P, u_Q, u_R)$ control signals.

As previously handled with system dynamics, the equations used to solve for $\delta_1, \ldots, \delta_6$ in this section may be condensed into block vector notation, as defined by Farrell et al. [6] and Farrell and Polycarpou [5][8.3.3]:

\[
B_3G_3\delta^o = -A_3f_3 - F_3 - K_3\tilde{z}_3 + \dot{z}_3 = B_2^T\tilde{z}_2
\]  

(3.3.34)

with $B_3G_3\delta^o = [u^o_P, u^o_Q, u^o_R]^T \equiv [3.3.25, 3.3.26, 3.3.27]^T$, $\delta^o = [\delta^o_1, \ldots, \delta^o_6]^T$, $A_3 = \text{Equation 3.3.14}$, $f_3 = [\bar{L}', \bar{M}', \bar{N}']^T$, $F_3 = \text{Equation 3.3.15}$, $K_3$ is a positive definite diagonal $3\times3$ matrix, diag($k_P, k_Q, k_R$), and $z_3 = [P, Q, R]^T$; the goal is to select $\delta^o$ such that Equation 3.3.34 holds. If there are more columns than rows in $B_3G_3$ the aircraft is over-actuated,
ie. multiple solutions to Equation 3.3.34 exist and some form of actuator distribution is required to select $\delta^o_c$.

Assume that the deflection solution to Equation 3.3.34 has been found, by some form of actuator distribution, and the $\zeta_3$–filter in block vector notation be defined as

$$\dot{\zeta}_3 = -K_3\zeta_3 + B_3G_3(\delta - \delta^o_c) \quad (3.3.35)$$

### 3.4 Stability Proof

The proof will be done in block vector notation to cut down on the number of equations. To summarize the flight-path backstepping development up to this point:

---

**Tracking & Compensated Tracking Errors**

$$\begin{align*}
\tilde{z}_1 &= z_1 - z_{1_c}, \\
\tilde{z}_2 &= z_2 - z_{2_c}, \\
\tilde{z}_3 &= z_3 - z_{3_c} \\
\bar{z}_1 &= \tilde{z}_1 - \zeta_1, \\
\bar{z}_2 &= \tilde{z}_2 - \zeta_2, \\
\bar{z}_3 &= \tilde{z}_3 - \zeta_3
\end{align*} \quad (3.4.1)$$

---

**System Dynamics**

$$\begin{align*}
\dot{z}_1 &= \bar{A}_1f_1 + F_1 + G_1(z_2) \\
\dot{z}_2 &= \bar{A}_2f_1 + F_2 + B_2\bar{z}_3 \\
\dot{z}_3 &= \bar{A}_3f_3 + F_3 + B_3G_3\delta
\end{align*} \quad (3.1.13, 3.2.13, 3.3.13)$$

---

**Control Allocation / Virtual Control Laws**

$$\begin{align*}
G_1(\mu_1^o, x) &= -\bar{A}_1f_1 - F_1 - K_1\bar{z}_1 + \dot{z}_{1_c} \\
B_2\mu_2^o &= -\bar{A}_2f_1 - F_2 - K_2\bar{z}_2 + \dot{z}_{2_c} \\
B_3G_3\delta^o_c &= -\bar{A}_3f_3 - F_3 - K_3\bar{z}_3 + \dot{z}_{3_c} - B_2^T\bar{z}_2
\end{align*} \quad (3.1.39, 3.2.33, 3.3.34)$$

with definitions $z_{2_c}^o = \mu_1^o$ and $z_{3_c}^o = \mu_2^o - \zeta_3$ for 3.1.39 and 3.2.33 respectively.
3.4.1 Tracking Error Dynamics

The goal is to compute compensated tracking error dynamics and use them to prove stability and convergence of system trajectories for the command filtered backstepping technique developed in this chapter. Analogous to the tracking error development in Section 2.3.4.3, we begin by taking the derivative of the outer loop tracking error equation, the first term in Equation 3.4.1, and then substitute the appropriate terms for this expression.

\[
\dot{\tilde{z}}_1 = \left[ \dot{z}_1 \right] - \left[ \dot{z}_{1c} \right] = \left[ \tilde{A}_1 f_1 + \tilde{F}_1 + G(z_2) \right] - \left[ G(z_{2c}) + \tilde{A}_1 f_1 + \tilde{F}_1 + K_1 \tilde{z}_1 \right] \pm G(z_{2c})
\]

\[
= -K_1 \tilde{z}_1 + G(z_2 - z_{2c}) + G(z_{2c} - z_{2c}^o) \quad (3.4.3)
\]

Recall that \( \dot{z}_1 \) is provided by 3.1.13 in ‘System Dynamics’ and \( \dot{z}_{1c} \) is calculated by algebraic manipulation of 3.1.39 in ‘Control Allocation / Virtual Control Laws’. Critical to the process evident in Equation 3.4.3 is the inclusion of a \( \pm \) virtual control law expression with filtered, feed-forward, command terms\(^6\), and the compensated tracking error substitution between the second to last and last step. This procedure for the outer loop is repeated for the middle loop

\[
\dot{\tilde{z}}_2 = \left[ \dot{z}_2 \right] - \left[ \dot{z}_{2c} \right] = \left[ \tilde{A}_2 f_1 + \tilde{F}_2 + B_2 z_3 \right] - \left[ B_2(z_{3c} - \zeta_3) + \tilde{A}_2 f_2 + \tilde{F}_2 + K_2 \tilde{z}_2 \right] \pm B_2 z_{3c}
\]

\[
= -K_2 \tilde{z}_2 + B_2(z_3 - z_{3c}) + B_2(z_{3c} - z_{3c}^o) \quad (3.4.4)
\]

\(^6\) Remember unfiltered is indicated with super-script “o” and commanded with sub-script “c”
and again repeated for the inner loop

\[
\dot{\hat{z}}_3 = [\hat{z}_3] - [\hat{\zeta}_3]
\]

\[
= [A_3 \bar{f}_3 + P_3 + B_3 G_3 \delta] - [B_3 G_3 \delta_c^o + A_3 \bar{f}_3' + P_3' + K_3 \bar{z}_3 + B_2^T \bar{z}_2] \pm B_3 G_3 \delta_c
\]

\[
= -K_3 \bar{z}_3 + B_3 G_3 (\delta - \delta_c) + B_3 G_3 (\delta_c - \delta_c^o) - B_2^T \bar{z}_2
\]

\[
= -K_3 \bar{z}_3 - B_2^T \bar{z}_2 + B_3 G_3 (\delta - \delta_c) + B_3 G_3 (\delta_c - \delta_c^o)
\] (3.4.5)

Now that we have expressions for tracking error, we may develop compensated tracking error dynamics, which requires Equations 3.4.3 – 3.4.5. As with the preceding development, we begin by taking the derivative of the outer loop compensated tracking error equation, the first term in Equation 3.4.2, and then substitute the appropriate terms for this expression.

\[
\dot{\hat{z}}_1 = [\hat{z}_1] - [\hat{\zeta}_1]
\]

\[
= [-K_1 \bar{z}_1 + G(z_2 - z_2^o_c) + G(z_2^o_c - z_2^o_c)] - [-K_1 \zeta_1 + G(z_2 - z_2^o_c)]
\]

\[
= -K_1 \bar{z}_1 + K_1 \zeta_1 + G(\bar{z}_2 - \bar{z}_2^o_c) + G(\bar{z}_2^o_c - \bar{z}_2^o_c) + G(\bar{z}_2^o_c - \bar{z}_2)
\]

\[
= -K_1 (\zeta_1 - \bar{z}_1)
\]

\[
= -K_1 \bar{z}_1
\] (3.4.6)

\[\hat{z}_1\] is provided by Equation 3.4.3 and \[\hat{\zeta}_1\] is provided by Equation 3.1.41 in ‘\(\zeta\)-Filters’. The \(K_1\) term is a positive definite diagonal matrix, as required by succeeding Lyapunov theory. The derivation for the outer loop compensated tracking error is repeated for the middle loop

\[
\dot{\hat{z}}_2 = [\hat{z}_2] - [\hat{\zeta}_2]
\]

\[
= [-K_2 \bar{z}_2 + B_2 \bar{z}_3 + B_2 (z_3 - z_3^o_c)] - [-K_2 \zeta_2 + B_2 (z_3_c - z_3^o_c)]
\]

\[
= -K_2 \bar{z}_2 + K_2 \zeta_2 + B_2 \bar{z}_3 + B_2 (z_3 - z_3^o_c) + B_2 (z_3^o_c - z_3^o_c)
\]

\[
= -K_2 (\bar{z}_2 - \zeta_2) + B_2 \bar{z}_3
\]

\[
= -K_2 \bar{z}_2 - B_2 \bar{z}_3
\] (3.4.7)
and again repeated for the inner loop

\[
\ddot{z}_3 = [\ddot{z}_3] - [\dot{\zeta}_3] = [-K_3 \dot{z}_3 - B_2^T \ddot{z}_2 + B_3 G_3 (\delta - \delta_c) + B_3 G_3 (\dot{\delta}_c - \dot{\delta}_c^o)] - [-K_3 \dot{\zeta}_3 + B_3 G_3 (\delta - \delta_c^o)] = -K_3 \ddot{z}_3 + K_3 \dot{\zeta}_3 + B_3 G_3 (\delta - \dot{\delta}_c^o) + B_3 G_3 (\dot{\delta}_c - \delta) + \dot{z}_3^T \dot{z}_2 = -K_3(z_3 - \zeta_3) - B_2^T \ddot{z}_2 = -K_3 \ddot{z}_3 - B_2^T \ddot{z}_2 \tag{3.4.8}
\]

Voila, one part tracking error derivative and one part compensated tracking error derivative yields system dynamics in terms of the control task at hand! Now to prove that the virtual control laws selected are stabilizing for a bounded flight path command.

### 3.4.2 Stability and Convergence

Akin to Equation 2.3.80 in Command Filtered Backstepping 2.3.4.3, define the control Lyapunov function for this system to be

\[
V = \frac{1}{2} \left[ \sum_{i=1}^{3} \ddot{z}_i^T \ddot{z}_i \right] \tag{3.4.9}
\]

The time derivative of the Lyapunov function along solutions from Equation 3.4.6 to 3.4.8 is given by

\[
\dot{V} = \ddot{z}_1^T \dot{z}_1 + \ddot{z}_2^T \dot{z}_2 + \ddot{z}_3^T \dot{z}_3 = -\ddot{z}_1^T K_1 \ddot{z}_1 - \ddot{z}_2^T K_2 \ddot{z}_2 + \ddot{z}_3^T B_2 \ddot{z}_2 - \ddot{z}_3^T K_3 \ddot{z}_3 - \ddot{z}_2^T B_2^T \ddot{z}_2 \\
= -\ddot{z}_1^T K_1 \ddot{z}_1 - \ddot{z}_2^T K_2 \ddot{z}_2 - \ddot{z}_3^T K_3 \ddot{z}_3 \leq 0 \tag{3.4.10}
\]

This true power of this technique is the flexibility that we have in choosing the virtual control laws, as the \(\dot{V}\) equation justifies. We now understand why the term \(-B_2^T \ddot{z}_2\) in the virtual control law developed for the inner loop, Equation 3.3.34, was implemented: to cancel the sign indefinite \(\ddot{z}_2^T B_2 \ddot{z}_3\) term in Equation 3.4.10! This is absolutely incredible, as we now
have a positive definite CLF with a negative semi-definite time derivative that guarantees
stability for the system’s operating point.

The $K_i$ terms are required to be positive definite in order to ensure that $\dot{V}$ in Equation 3.4.10
stays negative semi-definite. Convergence to a desired reference signal is guaranteed
through Barbalat’s Lemma, as the proof of the following theorem will show.

**Theorem 3.4.1 (CFBS Stability and Convergence).** Farrell et al. [6, Thm. 2]
The controller summarized in the introduction to Section 3.4 has the following properties:

1. The compensated tracking errors $\bar{z}_i$ for $i = 1, 2, 3$ are bounded
2. $\|\bar{z}_i(t)\| \to 0$ as $t \to \infty$ for $i = 1, 2, 3$
3. $\bar{z}_i(t) \in L_2$ for $i = 1, 2, 3$

**Proof:** Boundedness of the compensated tracking errors is guaranteed by $V$ being positive
definite in those errors and $\dot{V}$ being negative semidefinite, see Equation 3.4.9 and 3.4.10
respectively. Therefore ,

$$V(t) \leq V(0)$$

and

$$\dot{V}(t) \leq V(0) < \infty$$

both for $t > 0$; this completes the proof of item 1. The second derivative of the Lyapunov function

$$\ddot{V} = - \bar{z}_1^T (K_1 + K_1^T) (-K_1 \bar{z}_1 - A_1 f_1)$$

$$- \bar{z}_2^T (K_2 + K_2^T) (-K_2 \bar{z}_2 - A_2 f_1 + B_2 \bar{z}_3 - K_3 \bar{z}_3)$$

$$- \bar{z}_3^T (K_3 + K_3^T) (-K_3 \bar{z}_3 - A_3 f_3 + B_3 G_3 \delta - B_2^T \bar{z}_2)$$

is bounded, therefore $\dot{V}$ is uniformly continuous. This parallels Barbalat’s Lemma 2.2.2 ,
Form 2, implying that

$$\dot{V} \to 0 \quad \text{as} \quad t \to \infty$$
This requires that $\bar{z}_i^T K_i \bar{z}_i \to 0$ for $i = 1, 2, 3$ as $t \to \infty$, and because $\bar{z}_i^T K_i \bar{z}_i \geq \lambda K_i \|\bar{z}_i\|^2$, where $\lambda K_i$ is the minimum eigenvalue of the positive definite matrix $K_i$, we see that $\|\bar{z}_i\|^2 \to 0$ as $t \to \infty$ for $i = 1, 2, 3$; this completes the proof of item 2. Integrating both sides of Equation 3.4.10 yields

$$V(t) - V(0) \leq \int_0^t \bar{z}_i^T(\tau) K_i \bar{z}_i(\tau) d\tau$$

$$-V(0) \leq -\int_0^\infty \bar{z}_i^T(\tau) K_i \bar{z}_i(\tau) d\tau$$

$$V(0) \geq \int_0^\infty \bar{z}_i^T(\tau) K_i \bar{z}_i(\tau) d\tau$$

where $0 \leq V(t) \leq V(0)$ for all $t \geq 0$ and $\dot{V} \leq 0$ implies that $\lim_{t \to \infty} V(t)$ is well defined; this completes proof of item 3.

Remarks (Theorem 3.4.1)

1. The second property in Theorem 3.4.1 states that compensated tracking errors, $\bar{z}_i$, approach zero as time approaches infinity, regardless of the input signals ($\chi_c, \gamma_c, V_c$); this is not the case for tracking errors, $\tilde{z}_i$. When inputs are outside the magnitude, rate, or bandwidth limits\(^7\) imposed by the command filter “it is not possible to track the input because the desired control signals are not able to be implemented” thereby causing $\tilde{z}_i$ to become non-zero. When constraints are in effect this also causes the $\zeta_i$ signals to become non-zero, however remaining bounded; when constraints are not in effect $\zeta_i$ signals converge to zero, and $\tilde{z}_i$ converges toward $\bar{z}_i$.

2. Since we have a nominal design model available that defines function $f_1$, $f_2$, and $f_3$ (partially or completely unknown terms in adaptive backstepping), “the stability and tracking performance [is] affected by the errors between the design model and the actual system as indicated in the tracking error Equations 3.4.6 – 3.4.8. Using command filters ensures that intermediate commands remain in the specified operating envelope.”

\(^7\) collectively referred to as constraints
Chapter 4

Application

The simulation is designed without consideration for hardware implementation, ie. steps necessary to compile the model into C or C++ code; the assumption from the beginning is that it would be tested solely in a software environment.

4.1 Simulation Modeling

A quick background of the air vehicle’s role, along with supporting specifications, will be provided in this section. Again, as mentioned in Sec. 1.2, “All images, specifications, and verbiage used to present Fury® herein was found solely from online sources and cited immediately after presentation; it is data available to the public domain.”

High level Simulink models are shown within this section as well, with lower level subsystems included in Appendix A. As displayed atop the mass, aerodynamics, and thrust filter subsystem blocks, the text “AME*” designates proprietary information and will not be revealed in the Appendix.

4.1.1 Fury 1500 UAS

This controller is applied to AME UAS’s Fury long-endurance, runway-independent, unmanned aircraft. It “provides the longest endurance and largest payload capability of any runway independent small or small-tactical UAS available today... in an easily deployable, heavy-fuel capable and affordable package.” The following specifications, as cited by the Fury 1500
product card\(^1\), provide the reader with a sense of the size and performance of the air vehicle.

![Fury 1500 UAS - Rear Isometric View](www.defense-update.com/20110621_fury-1500-uav.html)

<table>
<thead>
<tr>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>MGTOW</td>
</tr>
<tr>
<td>Empty Weight</td>
</tr>
<tr>
<td>Max Payload</td>
</tr>
<tr>
<td>Fuel Load</td>
</tr>
<tr>
<td>Wingspan</td>
</tr>
<tr>
<td>Length</td>
</tr>
<tr>
<td>Max Range</td>
</tr>
<tr>
<td>Control Surfaces</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Endurance</td>
</tr>
<tr>
<td>Cruise Speed</td>
</tr>
<tr>
<td>Dash Speed</td>
</tr>
<tr>
<td>Max Altitude</td>
</tr>
<tr>
<td>Payload Power</td>
</tr>
</tbody>
</table>

![Fury 1500 UAS - Specifications / Top-View](www.ameuas.com/uas_fury.html)

Pertinent to the development herein is the flying wing configuration and six trailing edge control surfaces available, evident in Figure 4.1. Aerodynamic data was developed from wind-tunnel testing and condensed into tables for simulink interpolation via look-up tables.

\(^1\) http://www.ameuas.com/PDF/Fury1500ProductCard.pdf
It is recommended that the reader review the aerodynamic coefficient buildup in Duke et al. [16, Appendix A.], Stevens and Lewis [15, Sec. 2.3], or Klein and Morelli [24, Chp. 3]; if adaptive function approximation were implemented, as Farrell et al. [6, VII. Approximator Definition] shows, then fewer aerodynamic coefficients and analysis is necessary.

4.1.2 Simulink Models

Generic servo saturation and rate limits are specified on the surface position signal bus, as shown in Appendix A.2.4; in other words servos are assumed ideal. The same limits are used within QCAT’s subsystem mask properties. Provisions are made to model servo dynamics with the second order filter in Appendix A.2.5, however they are not used in ensuing simulation cases.

Furthermore, wind turbulence is not modeled, constant sea-level density is assumed where required, mass may be considered constant given the short simulation time span, and surface deflection positions are assumed to be measured and available. In this section only primary subsystems, as dictated by subsequent figures and corresponding captions, will be introduced with respect to Figure 3.4, repeated in this sub-section for convenience; note that sub-subsystem backgrounds, goto/from, and inport/outport colors match their respective primary subsystem’s block color, also repeated in a text based legend in the figure.
Figure 3.4: Command Filtered Backstepping Simulink Model
This is really the only block with control allocation, as all other subsystems here are directly solvable.

Figure 4.3: Control Allocation (CA) Primary Subsystem
Figure 4.4: Command Filter (CF) Primary Subsystem
Figure 4.5: Virtual Control Law (VCL) Primary Subsystem

Figure 4.6: Vehicle Model & System Dynamics (FURY) Primary Subsystem

Figure 4.7: $\zeta_x$ Filter (LPF) Primary Subsystem
4.2 Simulation Analysis and Results

Two different simulation scenarios are examined, nominal and effector failed tracking of outer loop commands $\chi_c$, $\gamma_c$, and $V_c$. Time series analysis will illustrate the performance of the control laws. The simulation spans 120 seconds, and follows the test case setup in Farrell et al. [6], where “both flight-path and heading angles [are varied] simultaneously while holding constant airspeed and regulating sideslip to zero, ie. coordinated turns. This set of commands is relatively challenging for the autopilot because it induces significant amounts of coupling between all three channels [(control loops)] and requires flight at high roll angles.”

Bounded unfiltered commands $\chi^o_c$, $\gamma^o_c$, and $V^o_c$ are specified at magnitudes of $\pm 60^\circ$, $\pm 10^\circ$, and 135 ft/s respectively. The frequency of the sinusoid generator for $\chi^o_c$ is 0.3 rad/s with 0.3 rad phase bias. The frequency of the sinusoid generator for $\gamma^o_c$ is 0.15 rad/s with 0.15 rad phase bias. The commands are shown in Figure 4.8, which for future reference are also available in tracking Figures 4.13, 4.14, and 4.15.

![Figure 4.8: Desired Tracking Commands, Dissected from CA Primary Subsystem Block in Figure 4.3](image)

These are generated for all intents and purposes “outside” the controller, meaning the commands are isolated from controller function. The implementation assumes an external command generation system, ie. flight path planner, does not generate bounded command signals and their derivatives. These signals are filtered in the command filter block and passed to succeeding loops where the controller calculates deflections that force the air vehicle to track the aforementioned inputs.
The air vehicle’s $t = 0$ conditions are initialized to a trimmed flight condition and no delay from simulation start is allowed to let the air vehicle wring out. The gains for both simulation cases were settled based on a brute force gains tuning script, then manual tweaking:

**Table 4.1: Simulation Tracking Error and Command Filter Gains**

<table>
<thead>
<tr>
<th>$\zeta = 0.7$</th>
<th>$\chi$</th>
<th>$\gamma$</th>
<th>$V$</th>
<th>$\mu$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$P$</th>
<th>$Q$</th>
<th>$R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega_n$</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>30</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>$k$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>20</td>
<td>20</td>
<td>10</td>
</tr>
</tbody>
</table>

As Sonneveldt et al. [7, Part VI.A] says, unfortunately “there are no real guidelines for tuning a backstepping controller.” The only requirement, as evident by the block vector formulated stability proof in Sec. 3.4, is that the $K_i$-gain matrices are positive definite so that $\hat{V}$ in Equation 3.4.10 stays negative semi-definite\(^2\). Lyapunov stability technically only requires the gains to be larger than zero for stability to be guaranteed. However, typically they are selected such that $K_3 > K_2 > K_1 > 0$. As Farrell et al. [6, Sec. IX.] notes, the $K_i$ terms do not determine the trajectory tracking bandwidth, they determine “the rate of decay of transient errors in $\tilde{z}_i$ caused, for example, by initial condition errors or disturbances.” This is the crux of the backstepping method, and what keeps controls engineers in business.

The recommended gain relationships are reflected in Table 4.1, as the inner loop gains are larger than middle loop gains, which are larger than outer loop gains. Given the choice of damping ratio, $\zeta = 0.7$, filtered bandwidth terms ($\omega_n$) had to be larger than it’s associated $k$ term so that the filtered command term wouldn’t lag the unfiltered command, designated by blue and red dashed lines respectively in tracking figures below. The bandwidth terms of the command filter will influence the state trajectory, as the command derivatives generated by this subsystem are fed into virtual control laws. “If, for example $\gamma_c^\phi$ were a step command, then as the bandwidth of the $\gamma$-command filter is increased, the magnitude of $\dot{\gamma}_c$ will increase.” Farrell and Polycarpou [5]

\(^2\) Recall $K_i$ matrices are diagonal matrices, which are grouped into subsets of the control loops; diagonal terms are listed in Table 4.1 as $k$’s.
This gain set is by no means the optimal gain set. A trial-and-error script that ran the simulation with sweeps of gains, under the premise just established, was the only tool that was used to pick gains. After running hundreds of sets of gains, one of the better sets were picked and then manually tuned to the parameters shown. If more time were available a test procedure for narrowing in on optimal gain-sets could’ve been created.

4.2.1 Nominal – PINV

The nominal configuration implies that all actuators are working normally and the PINV control allocation technique is used with simple deflection filters, refer to Figure 2.25 and Appendix A.2.4 respectively. Note that the first 10 seconds for the plots in this sub-section have been dropped; within this time window tracking errors were much larger, see Figure 4.17, and took away from the true tracking region (plot limits much tighter).

The most meaningful figures in the set – 4.13, 4.14, and 4.15 – of the tracking performance, depict ideal (unfiltered) commands ($x_{oc}$), actual commands ($x_c$), and states ($x$). The inner and middle loops track extremely well, as Figure 4.17 confirms, however in the outer loop, glide-path angle tracking specifically, has an average tracking error of $-3^\circ$. Another suspicious observation in the outer loop tracking figure is the filtered ground-track command ($\chi_c$), which does not track the unfiltered signal’s amplitude or phase whatsoever; the odd part though is that the tracking error for this variable, $\tilde{\chi}$, is nearly zero, leading me to believe this is a signal routing issue.
Figure 4.11: Nominal – States

Figure 4.12: Nominal – Deflections
Figure 4.13: Nominal – Outer Loop Tracking

Figure 4.14: Nominal – Middle Loop Tracking
Roll Rate Command Tracking
Roll Rate, [deg/s]
CA: P
CF: P
EOM: P

Pitch Rate Command Tracking
Pitch Rate, [deg/s]
CA: Q
CF: Q
EOM: Q

Yaw Rate Command Tracking
Yaw Rate, [deg/s]
CA: R
CF: R
EOM: R

Figure 4.15: Nominal – Inner Loop Tracking

Figure 4.16: Nominal – Flightpath Angle

Pseudo-Controls, see Sec. 3.1.1.1
Figure 4.17: Nominal – Tracking Errors
4.2.2 Effector Failure – QCAT

As shown in Sec. 2.3.6, Figure 2.26, optimal actuator distribution via QCAT is implemented for this scenario: effector 2, left wing midboard, stuck at $-10^\circ$, trailing edge up, 60 seconds into simulation. This toolbox’s subsystem requires magnitude and rate limits for operation, therefore deflection filters are manually bypasses in this case. The control effectiveness matrix is also a requirement, however in its current implementation this matrix is static, ie. does not update with $\alpha$ and $\beta$ values due to the nature of the subsystem mask. If more time were available breaking this system out by developing a custom subsystem with Ola Härkegård’s toolbox scripted functions should significantly improve performance and robustness, hopefully reducing much of the oscillations.

Nevertheless, the utility of optimal control allocation is definitely evident, and provides an extra margin of safety for UAV operators; an area of growing concern, as Sec. 1.1 pointed out. Even with a static control effectiveness margin and a sub-optimal gain-set the air vehicle still remains airworthy!
Figure 4.20: Failure - States

Figure 4.21: Failure - Deflections
Figure 4.22: Failure – Outer Loop Tracking

Figure 4.23: Failure – Middle Loop Tracking

Figure 4.24: Failure – Outer Loop Tracking
Roll Rate Command Tracking

Roll Rate, [deg/s]

CA: P
CF: P
EOM: P

Pitch Rate Command Tracking

Pitch Rate, [deg/s]

CA: Q
CF: Q
EOM: Q

Yaw Rate Command Tracking

Yaw Rate, [deg/s]

CA: R
CF: R
EOM: R

Figure 4.24: Failure – Inner Loop Tracking

Figure 4.25: Failure – Flightpath Angle

Pseudo-Controls

Figure 4.24: Failure – Inner Loop Tracking

Pseudo-Controls
Chapter 5

Conclusion

A short background of aircraft dynamics and supporting concepts in Sec. 2.1 Modeling led to a deep understanding of the nonlinear equation of motion set essential to the modeling and simulation effort herein. Preliminary results of Lyapunov stability theory were developed in Sec. 2.2 Stability. Following this was a thorough section on Control, Sec. 2.3 Control, where core backstepping theories were derived along with brief overviews of command filtering in Sec. 2.3.5 and control allocation in Sec. 2.3.6. Three backstepping flavors were covered: a simple second order nonlinear system, a generic higher order nonlinear system, and another second order nonlinear system with command filtering. Each was selected to highlight a particular benefit of backstepping, and repetitively demonstrated steps required to manipulate the plant and develop control laws. A full state feedback, six degree of freedom, nonlinear flight path control system was derived for a UAV in Chp. 3. Finally, simulation block diagrams and tracking results were presented and discussed in Chp. 4 Application.

A Simulink driven simulation implementing the control architecture outlined in this thesis successfully flew AME’s Fury 1500 UAS. Coordinated flight with outer loop tracking was achieved for a case that was demanding of the autopilot, with heading and flight-path angle commands varying from ±60° and ±10° respectively at a constant airspeed command. These signals were guaranteed achievable due to a command filter implementation, which imposed magnitude, rate, and bandwidth constraints on external and internally generated command signals. Two different forms of control allocation were implemented, the first a very simple least-squares pseudo-inverse and the second an optimal quadratic programming based toolbox.
Future Recommended Work

1. Adaptive aero coefficient approximation for unfamiliar plants and modeling error

2. Gains optimization
   (a) Linearize system around particular operating point to analyze with classic control techniques
   (b) Control effort minimization algorithm

3. Evaluate robustness to external disturbances with turbulence simulation

4. Flight path controller derivation for backstepping that uses Tikhonov’s theorem to guaranteed global exponential stability GES Farrell et al. [25] and Farrell et al. [11]

5. Hardware in the loop testing and preceding simulink model preparation

6. Work on initialization for all buses / signals in model to remove the 10 sec tracking error transient observed
Appendix A

Primary Subsystems Breakdown

Primary subsystems are those depicted in Figure 3.4\textsuperscript{1}. Each section herein steps into its respective figure in Sec. 4.1. Recall, all models with masks that read, “AME*” are omitted.

A.1 Control Allocation (CA), Figure 4.3

A.1.1 Solve – $\alpha_c^o$ and $\mu_c^o$

\textsuperscript{1} Quick Tip: In some PDF browsers, “ALT+←” acts like a back button, therefore after you click a hyperlink in this document you may easily go back to the original location with this command.
A.1.1.1  X and Y

A.1.2  Solve – $T_c^{\circ}$
A.1.3 Solve – $P_c^o, Q_c^o, \text{ and } R_c^o$

Refer to figures introduced in previous section for Pseudo Inverse (PINV) Figures 2.25 and Quadratic Programming Control Allocation Toolbox (QCAT) Figure 2.26 control allocation methods.

A.1.4 Solve – $\delta_c^o$, Figure 2.24
A.2 Command Filters (CF), Figure 4.4

A.2.1 Flight Path Angle & Airspeed Filters

A.2.2 Wind Axis Angle Filters
A.2.3 Body Axis Rate Filters

A.2.4 Simple Deflection Filters
A.2.5 Deflection Filters

\[ \delta_c = \frac{1}{2\zeta\omega_n d} \]

\[ \omega_n^2 = \frac{1}{2\zeta\omega_n d} \]

\[ \delta_0_c \]

\[ \delta_c \]

\[ d_1 \]

\[ d_2 \]

\[ d_3 \]
A.3 Virtual Control Laws (VCL), Figure 4.5

A.3.1 Loop Interactions

\[ x = [\chi \gamma V \mu \alpha \beta P Q R]^T \]

A.4 Vehicle Model & Dynamics (FURY), Figure 4.6
A.4.1 System Dynamics

\[ x = [\chi \gamma \mu \alpha \beta \ P \ Q \ R]^T \]

Control Laws

Filtered Control Laws

1. Flight Path Variables
   \[ x = [D, Y, mass]^T \]
   - \[ f_{\chi} \]
   - \[ F_{\chi} \]

2. Wind Axis Angles
   \[ x = [D, Y, T, mass]^T \]
   - \[ f_{\mu} \]
   - \[ F_{\mu} \]
   - \[ f_a \]
   - \[ F_a \]
   - \[ f_b \]
   - \[ F_b \]

3. Body Axis Angular Rates
   \[ \{L, M, N\}_aero \]
   - \[ f_p \]
   - \[ F_p \]
   - \[ f_q \]
   - \[ F_q \]
   - \[ f_r \]
   - \[ F_r \]

- \[ u_{\chi} \]
- \[ u_{\gam} \]
- \[ u_{\mu} \]
- \[ u_{a} \]
- \[ u_{b} \]
- \[ u_{p} \]
- \[ u_{q} \]
- \[ u_{r} \]
- \[ u_{\alpha} \]
- \[ u_{\beta} \]
- \[ u_{\mu} \]
- \[ u_{\beta} \]
- \[ u_{\mu} \]

\[ xdot = f(x, u) \]
\[ \text{EOM} \]
\[ x(t) = x(0) + \int_0^t f(x(s), u(s)) \, ds \]
A.4.1.1 Flight Path Variables

\[ (T \cos A \sin B \sin P - mg \cos G) \cdot \frac{1}{mV} \]
\[ (-T \cos A \sin B \cos P) \cdot \frac{1}{mV \cos G} \]
\[ -(D \sin B + Y \cos B) \cdot \frac{\sin P}{mV} \]
\[ (D \sin B + Y \cos B) \cdot \frac{\cos (\mu)}{mV \cos G} \]

\[ F_{\text{gam}} \]
\[ f_{\text{gam}} \]
\[ F_{\text{chi}} \]
\[ f_{\text{chi}} \]

\[ \text{[m]} \]
\[ \text{[T]} \]
\[ \text{[T]} \]
\[ \text{[T]} \]
\[ \text{[sB]} \]
\[ \text{[sB]} \]
\[ \text{[sM]} \]
\[ \text{[sB]} \]
\[ \text{[sM]} \]
\[ \text{[sB]} \]

A.4.1.2 Body Axis Angular Rates

\[ F_r \]
\[ f_r \]
\[ F_q \]
\[ f_q \]
\[ F_p \]
\[ f_p \]

\[ \text{[c9]} \]
\[ \text{[c7]} \]
\[ \text{[c5]} \]
\[ \text{[c8]} \]
\[ \text{[c6]} \]
\[ \text{[c3]} \]
\[ \text{[c1]} \]
\[ \text{[c4]} \]
\[ \text{[c2]} \]

\[ \text{[P]} \]
\[ \text{[R]} \]
\[ \text{[P]} \]
\[ \text{[Q]} \]
\[ \text{[R]} \]

\[ \text{[n_aero]} \]
\[ \text{[m_aero]} \]
\[ \text{[l_aero]} \]

\[ \text{[P]} \]
\[ \text{[n_aero]} \]
\[ \text{[l_aero]} \]
\[ \text{[m_aero]} \]
\[ \text{[n_aero]} \]

\[ \text{[c9]} \]
\[ \text{[c7]} \]
\[ \text{[c5]} \]
\[ \text{[c8]} \]
\[ \text{[c6]} \]
\[ \text{[c3]} \]
\[ \text{[c1]} \]
\[ \text{[c4]} \]
\[ \text{[c2]} \]
A.4.1.3 Airspeed

\[ \frac{1}{m} \cdot (-D \cos B + Y \sin B) \]

A.4.1.4 Wind Axis Angles
A.4.1.5 Control Laws
A.4.1.6 Filtered Control Laws
$x = [\chi \gamma V u \alpha \beta P Q R]^T$
A.5.1 Euler Scope Transformations
Bibliography


