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

February 2012

COMMITTEE MEMBERSHIP

TITLE:	Nonlinear UAV Flight Control Using Command Filtered Backstepping
AUTHOR:	Brian Borra
DATE SUBMITTED:	February 2012
COMMITTEE CHAIR:	Eric Mehiel, Ph.D.
COMMITTEE MEMBER:	Robert Miller, Ph.D.
COMMITTEE MEMBER:	Al Jiminéz, Ph.D.
COMMITTEE MEMBER:	Dan Biezad, Ph.D.?

Abstract

Nonlinear UAV Flight Control Using Command Filtered Backstepping

Brian Borra

RED TEXT ⇒ Comments Appreciated / Needs Attention

This effort is aimed to extend the state of the art in the areas of flight control, specifically flight path control via command filtered backstepping for use in AME UAS's Fury 1500 unmanned flying-wing aircraft. Backstepping is a non-linear systematic 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. Provisions for online parameter approximation and reconfigurable flight control can be made in order to add robustness to abnormally large disturbances.

The work herein is a simplified implementation based on publications by Farrell, Sharma, and Polycarpou: adaptation and dynamic control allocation are not applied, however command filtering is. The goal is to lay the ground work for physical implementation in the Fury air vehicle, therefore, to mitigate risk, advanced features will be added after this baseline case is verified. Command filtering assures that control inputs generated meet magnitude, rate, and bandwidth constraints for states and actuators.

State Results

Acknowledgements

Thank you...

Table of Contents

List of Tables

List of Figures

Nomenclature

α	Angle of Attack	deg
$\alpha(x)$	Stabilizing Feedback Control Law	BS
β	Angle of Sideslip	deg
γ	Flight-Path Angle	deg
ζ_x	Filtered State	
ζ	Damping Frequency	rad/s
μ	Bank Angle	deg
ξ	Virtual Control Law	BS
ϕ, θ, ψ	Roll, Pitch, and Yaw Angles	deg
χ	Heading Angle	deg
ω_n	Natural Frequency	rad/s
D	Drag	$\mathrm{lb_f}$
F	Known Sub-function	
L	Lift	$\mathrm{lb_f}$

V	Airspeed	ft/s
W	Positive (Semi-)Definite Function	BS
X	X-Coordinate for (α_c^o, μ_c^o) Solution	
Y	Sideforce	$\mathrm{lb_f}$
Y	Y-Coordinate for (α_c^o, μ_c^o) Solution	
$\bar{L}$, $\bar{M}$, $\bar{N}$	Roll, Pitch, and Yaw Moments	lb-ft
f	Nominal System	
f	Unknown or Partially Known Sub-function	
g	Perturbed System	
m	Mass	slug
u	Control Law	
u, v, w	Longitudinal, Lateral, and Normal Velocities	ft/s
z	Tracking Error	BS

Acronyms

AS	Asymptotically Stable
BS	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

S Stable

Subscript

c Commanded

des Desired

M Magnitude

R Rate

s Stability Axes

Superscript

$-$ Compensated Tracking Error

$\sim$ Tracking Error

o Unfiltered

Chapter 1

Introduction

> "Sometimes I tell people that I don't understand X. Sometimes they reply, 'Well, I clearly explained it on page 4.' That'd be the wrong answer. If I (or anyone else) tells you they don't understand X, they're saying you haven't explained X well enough. Rather than getting defensive, reexamine your explanation, see how you can rewrite to make it clearer. It's not the reader's fault if they don't understand what you're saying. Your job as a writer is to make it very clear where you're coming from. Telling me that I just missed the point won't score you any points. Take another look, see what's not obvious." `http://web.media.mit.edu/~intille/teaching/advising/draftAdvisorTips.htm`
>
> What's the best thesis? A done one.

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. As technology expands so does its complexity and the level of detail placed into the system model. Typically plants with non-linearities 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.

Focus

Implementation of a non-linear command filtered backstepping(CFBS) flight controller for use in an AME UAS program. Specifically flight path control tracking of ground track, flight path (climb rate), and airspeed commands [χ_c,γ_c,V_c] from mission planner.

The focus of this thesis is to explore backstepping, a relatively new control algorithm for non-linear systems. It's a subset of a popular modern adaptive control technique called Direct Adaptive Control (DAC) that utilizes Lyapunov synthesis to derive a stabilizing controller. "Backstepping 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 [1]

Need to turn these points into paragraphs...

Appeal

Nonlinear flight control methods offer

- increase in performance
- reduction in development time

Adaptive methods offer

- management of uncertain dynamics / less precise model knowledge required
- protection against mechanical failures / damage

Constrained Backstepping

- u(t) generated guaranteed implementable
- retains useful nonlinearities*
- Lyapunov stability results provable

Goals

Attain professional proficiency in topic.

Develop a simulink based backstepping control architecture with

- Command Filtering
- Control Allocation / Distribution**
- Reproduce Simulation Analysis in "Backstepping-Based Flight Control with Adaptive Function Approximation" by J. Farrell, M. Sharma, & M. Polycarpou, 2005

Thoroughly support simulation with

- Lyapunov Control Law Synthesis / Stability Proof

Backstepping has applications in a broad spectrum of engineering disciplines, from electrical motors to jet engines, ship control to flight control just to name a few. It offers a systematically methodology for developing control schemes for non-linear systems.

Figure 1.1: Fury 1500 UAS

http://www.prnewswire.com/news-releases/fury-1500-uas-achieves-14-hour-duration-flight-html

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 (UAS) 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.

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 AME UAS Engineering *Fury UAS* simulation model will be substituted with a publicly available one for security as well as legal reasons*. Proof of concept will be demonstrated by the dissimilarity in control effectiveness of these models. A concluding objective would include hardware implementation and flight test demonstration.

1.3 Thesis Outline

Review aircraft dynamics and supporting concepts essential to modeling and simulation. Next, results from Lyapunov theory then derive backstepping theory. Two cases were covered, a second order non-linear system and a higher order non-linear system. Each was selected to highlight a particular benefit of backstepping, however they repetitively demonstrate the steps required to manipulate the plant and develop a control law. A full state feedback, six degree of freedom, non-linear flight path control system will be derived for a UAV. To close the simulation results will be discussed with respect to feedback linearization.

Chapter 2

History

This section consists of thoughts, needs to be condensed into one flowing historical account...

Mission requirements drive vehicle requirements, and these drive control law requirements. See WPAFB Multivariate control methods

See "Constructive Nonlinear Control: Progress in the 90's"

Introduce the concept of control, lead into adaptive control section.

(1) Emergence of Adaptive Control - Krstic (2) WPAFB: MVAR CONTROL DESIGN GUIDELINES

FEEDBACK CONTROL! Feedback allows for large disturbances

"The main purpose of every feedback loop, created by nature of designed by engineers, is to reduce the effect of uncertainty on vital system functions. [5] Feedback can be used for stabilization, but inappropriately designed feedback controllers may reduce, rather than enlarge, regions of stability. "

Design Methods

Linear Quadratic Gaussian (LQG, H_2), Worst case L2 - induced norm problem (H∞), Worst case L∞ - induced norm problem (l1). "Introduction of these formal methods, together with use of state-space descriptions provided the first multivariable control design tools" (WL-TR-96-3099)

Table 1.1: Categorization of Design Methods

Formal Optimization Problems	
Basic Linear-Quadratic-Gaussian (LQG)	[Athans 1971]
LQG with Explicit/Implicit Model Following	[Asseo 1970, Tyler 1966]
Frequency Weighted LQG	[Gupta 1980]
Worst-case Induced L_2 Norm (H_∞)	[Zames 1983, Doyle 1989]
Worst-case Induced L_∞ Norm (ℓ_1)	[Dahleh 1987]
Mixed Criteria (H_2/H_∞)	[Zhou 1989, Rotea 1991, Yeh 1992]
Numerical Optimization Problems	
Fixed Structure LQ-Control	[Axsater 1966, Levine 1970, Stein 1971]
Fixed-Plus-Variable Gain LQ-Control	[VanDierendonck 1972]
Fixed Structure H_∞ Control	[Bernstein 1990]
Fixed Structure H_2/H_∞ Control	[Bernstein 1989, Ridgely 1992]
Q-Parameter Design (QDES)	[Boyd 1991]
μ-Synthesis via $D-K$-Iteration	[Doyle 1983, Stein 1991]
Frequency Domain Methods	
Diagonal Dominance/Inverse Nyquist Array	[Rosenbrock 1974]
Characteristic Loci	[Postlethwaite 1979]
Upper Triangular Structures	[Mayne 1973]
Singular-Value-Based Loop Shaping	
via Direct Inversion	[Hung 1982]
via LQG/LTR	[Stein 1987]
via H_∞	[McFarlane 1992]
via Dynamic Inversion	[Bugajski 1992b]
Quantitative Feedback Theory (QFT)	[Horowitz 1979]
Eigenstructure Assignment Methods	
via Full-State Feedback	[Andry 1983]
via Output Feedback	[Calvo-Ramon 1986, Sobel 1990]
via Quadratic Weights in LQ	[Harvey 1978]
via Numerical Optimization	[Garg 1989, Wilson 1990]
Fringe Methods	
Model Predictive Control (MPC, DMC, MAC)	[Morari 1989]
Covariance Control	[Skelton 1989]
Stochastic Parameters (maximum entropy)	[Hyland 1982]
Variable-Structure Control	[Utkin 1977]
Geometric Methods	[Wonham 1979]
Polynomial-Matrix Methods	[Peczkowski 1978, Wolovitch 1974]
Lyapunov-Based Methods	[Barmish 1985, Boyd 1989]

Figure 2.1: WPAFB: Design Methods

2.1 Emergence of Adaptive Control

Reference work from seminal papers on adaptive control: Bode lectures by Gunter Stein and Petar Kokotovic.

First implementations of feedback may be considered indirect attempts to control unknown plants

- "Even the most elementary feedback loops can tolerate significant uncertainties"

50's and 60's: Advances spawned more sophisticated systems

- Find Examples: Assume space-race drove this, Find aircraft that required adaptive control
- self-learning, self-optimizing, self-organizing, self-tuning, and adaptive control utilizing on-line identification or pattern recognition.
- Lyapunov techniques were not a major player in the control engineering until the early 1960's thanks to publications by Lur'e and a book by La Salle and Lefschetz* -Slotine and Lee

computers started to come into play

$>$ 60's: Theoretical: Stochastic and Dual Control

- gain-scheduling, fuzzy, neural, intelligent control

80's: Adaptive Linear Control or Traditional Adaptive Control

- Non-Linear Lyapunov-based control recently* achievable through recursive design procedures such as backstepping.
- "[Estimation-based designs are flexible and] achieved by treating the identifier as a separate module and guaranteeing its properties independent of the controller module." Usually referred to as modular designs; can use gradient and least squares algorithms for parameter update laws.

Traditional control relies on the certainty equivalence principal: ”controller first designed as if all the plant parameters are known.”

- ”... control parameters are calculated by solving design equation for model matching, pole-zero placement, or optimality.”
- For each adaptive control scheme it is up to the designer to choose:
  - Filters
  - Design Coefficients
  - Initialization Rules
  - etc? CONTROL ENGINEER JOB...
- Adaptive Controls Tradeoff: Transient Performance vs. Robustness

The appeal of non-linear control over linear is simple, performance. For systems which are highly structured or parametrically uncertain a nonlinear controller outperforms the best linear controller. [6]

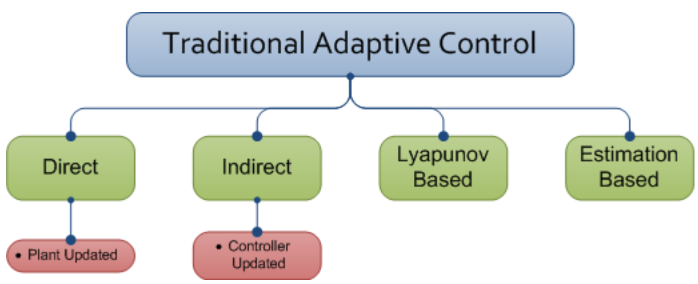


Figure 2.2: Traditional Adaptive Control Flowchart

“Progress for nonlinear systems was slow until the breakthrough Kanellakopoulos et al. [7] in adaptive non-linear control. The breakthrough was based on the nonlinear stabilization concept of “adding an integrator” as cited in **?** , Pg. 10]. The recursive application of this technique eventually led to the adaptive results of Kanellakopoulos et al. [7], is known and integrator backstepping or simply backstepping [8], [6], [9].

“backstepping led to the discovery of structural strict feedback condition (much weaker than the matching condition) under which the systematic construction of an RCLF is always possible. These first robust backstepping results appeared in... smoothness assumptions are common in recurisve backstepping designs becaue of the need to calculate derivates of various functions during the construction of the control law and Lyapunov function." [5]

2.2 Emergence of Backstepping

Development of backstepping, how it supports current requirements, and fits into current control law needs. Talk about how it is becoming a hot topic in the non-linear design world (an attempt to systematically approach non-linear control) and meets both aircraft and spacecraft control needs.

Adaptive Backstepping developed by Ioannis Kanellakopoulos [63] in collaboration with Petar Kokotovic, and Steve Morse "emerged as a confluence of the adaptive estimation idea on one side, and, on the other size, nonlinear control ideas expressed in works of [193], [12], [175], [85], [163]." Pg 11. Tuning functions were invented by Krstic [92,94] to reduce overparameterization.

- Development –¿ Kokotovic and Kanellakopoulos
  - Feedback linearization / NDI first
- Examples and reference papers used
  - Kokotovic Jet Engine Controller and other examples listed in Hrkegrd's dissertation
  - Hrkegrd's work with ADMIRE model (Canard Fighter)
  - Farrell's flying wing work which my controller is implementing

..

- Future of backstepping, ie. its role in future nonlinear control architecture. Do some research on alternative control techniques.

Backstepping has applications in a broad spectrum of engineering disciplines, from electrical motors to jet engines, ship control to flight control just to name a few. It offers a systematically methodology for developing control schemes for non-linear systems. The main appeal is that useful non-linearities do not have to be cancelled in the control law thereby increasing robustness to modeling errors and decreasing control effort. The alternative, feedback linearization, transforms a non-linear system into an equivalent linear system by cancelling all non-linearities via its control law.

Chapter 3

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 [10]

3.1 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 physical illustrations of key aspects to the derivation rather than pure mathematical formulation. The fallout allows the designer to both qualitatively *and* quantitatively evaluate the characteristic modes of motion, thereby providing an analytical playground for control design and performance evaluation via **handling qualities**[1]. 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

[1] Reference handling qualities for unmanned aerial vehicles, this is a hot topic; degree of instability depends on what autopilot can handle...

include necessary supporting concepts. The philosophy of this approach and procedure itself is credited to work by [11] and [12].

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 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." [12]

Assumption 3.1. 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." [10] 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 3.1. 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** [2] of this controller will consist of the first 9 variables.

Table 3.1: Choices for State Variables

		Body		Flight-Path / Wind	
x_1	Translational	Longitudinal Velocity	u	Heading Angle	χ
		Lateral Velocity	v	Flight-Path Angle	γ
		Normal Velocity	w	Velocity	V
x_2	Attitudes	Euler Roll	ϕ	Bank Angle	μ
		Euler Pitch	θ	Angle of Attack	α
		Euler Yaw	ψ	Sideslip Angle	β
x_3	Rotational	Roll Rate	P	Roll Rate	P
		Pitch Rate	Q	Pitch Rate	Q
		Yaw Rate	R	Yaw Rate	R
x_4	Navigation	North Position	ξ	North Position	ξ
		East Position	η	East Position	η
		Altitude	h	Altitude	h

Body

- 3 components of attitude to specify orientation relative to the gravity vector
- 3 components of velocity to specify translational kinetic energy
- 3 components of angular velocity to specify rotational kinetic energy
- 3 components of position to specify potential energy in earth's gravity field

Flight Path / Wind

- 3 components of attitude to specify orientation relative to the velocity vector
- 1 component of velocity magnitude, 2 components of velocity direction to specify translational kinetic energy
- 3 components of angular velocity to specify rotational kinetic energy
- 3 components of position to specify potential energy in earth's gravity field

[2] The minimal set of system variables necessary to indicate the energy of the system, potential and kinetic, and its distribution at any given time.

Again the flight path controller herein will only be using the first three subsets of Table 3.1:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \end{bmatrix} = \begin{bmatrix} (\chi\ \gamma\ V)^T \\ (\mu\ \alpha\ \beta)^T \\ (P\ Q\ R)^T \end{bmatrix} \begin{array}{r} \text{Translational} \\ \text{Attitude} \\ \text{Rotational} \end{array} \tag{3.1.1}$$

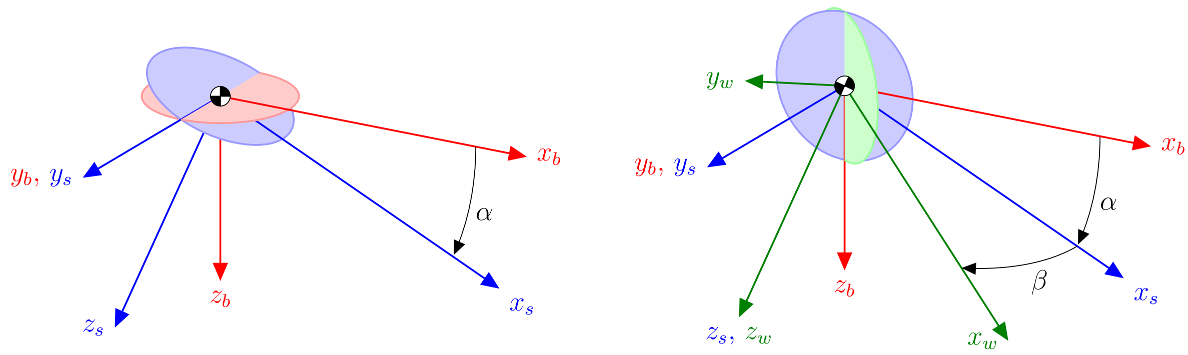
3.1.1 Reference Frames

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 3.2. 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 3.1. 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



(a) Body to Stability Axes

(b) Stability to Wind Axes

Figure 3.1: Air Vehicle Reference Frames

axes in the x_b-z_b plane by an angle-of-attack, α, that trims the air vehicle, ie. 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. Note that sideslip angle, β, is zero in stability axes. 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 3.2:

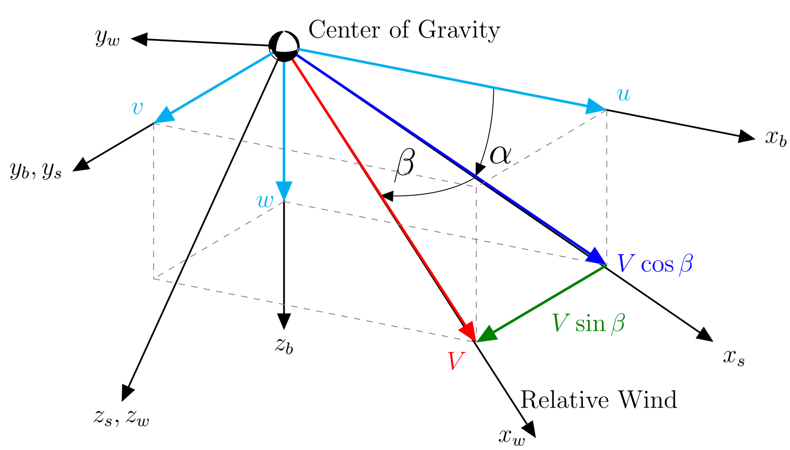


Figure 3.2: Axis Relationships: Body, Stability, and Wind Axes

The body state variables can be derived from the flight-path state variables as follows

[10], recall Table 3.1:

$$u = V\cos\beta\cos\alpha \tag{3.1.2}$$

$$v = V\sin\beta \tag{3.1.3}$$

$$w = V\cos\beta\sin\alpha \tag{3.1.4}$$

$$\phi = \tan^{-1}\left(\frac{\cos\gamma\sin\mu\cos\beta - \sin\gamma\sin\beta}{-\cos\gamma\sin\mu\sin\alpha\sin\beta + \cos\gamma\cos\alpha\cos\mu - \sin\gamma\sin\alpha\cos\beta}\right) \tag{3.1.5}$$

$$\theta = \sin^{-1}\left(\cos\gamma\sin\mu\cos\alpha\sin\beta + \cos\gamma\cos\mu\sin\alpha + \sin\gamma\cos\alpha\cos\beta\right) \tag{3.1.6}$$

$$\psi = \tan^{-1}\left\{\frac{(\sin\mu\sin\alpha - \cos\alpha\cos\mu\sin\beta)\cos\chi + [\cos\gamma\cos\alpha\cos\beta - \sin\gamma(\sin\alpha\cos\mu + \sin\beta\cos\alpha\sin\mu)]\sin\chi}{-(\sin\mu\sin\alpha - \cos\alpha\cos\mu\sin\beta)\sin\chi + [\cos\gamma\cos\alpha\cos\beta - \sin\gamma(\sin\alpha\cos\mu + \sin\beta\cos\alpha\sin\mu)]\cos\chi}\right\} \tag{3.1.7}$$

Flight-path variables ca be derived from the body state variables as follows [10], recall Table 3.1:

$$V = \sqrt{u^2 + v^2 + w^2} \tag{3.1.8}$$

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{3.1.9}$$

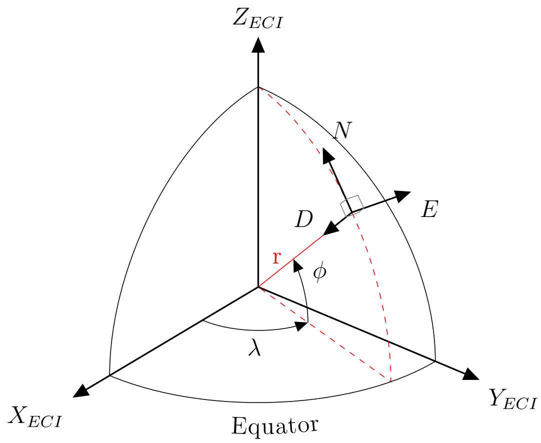
$$\beta = \sin^{-1}\left(\frac{v}{\sqrt{u^2 + v^2 + w^2}}\right) \tag{3.1.10}$$

$$\mu = \tan^{-1}\left[\frac{uv\sin\theta + (u^2 + w^2)\sin\phi\cos\theta - vw\cos\phi\cos\theta}{\sqrt{u^2 + v^2 + w^2}\,(w\sin\theta + u\cos\phi\cos\theta)}\right] \tag{3.1.11}$$

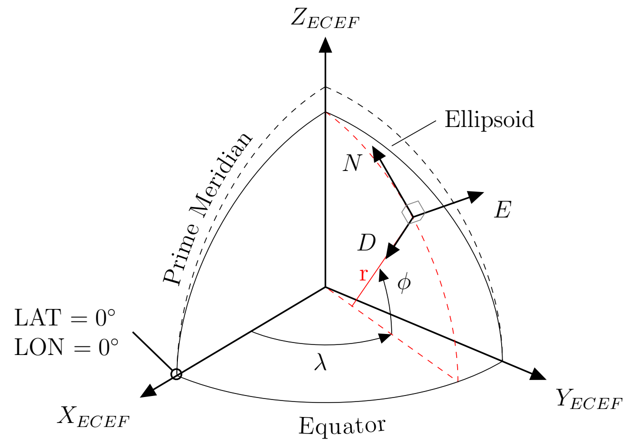
$$\gamma = \sin^{-1}\left(\frac{u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta}{\sqrt{\mathrm{u}^2 + v^2 + w^2}}\right) \tag{3.1.12}$$

$$\chi = \tan^{-1}\left[\frac{u\cos\theta\sin\psi + v(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) + w(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)}{u\cos\theta\cos\psi + v(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) + w(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)}\right] \tag{3.1.13}$$

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 (λ), east is parallel to lines of latitude (ϕ), 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.



(a) Earth Centered Inertial (Fixed)



(b) Earth Centered Earth Fixed (Rotates)

Figure 3.3: 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 [13, 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_s 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 ellipsoid surface.

3.1.2 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[3] 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

[3] Definition

et al. [12] and Stevens and Lewis [14] 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, $\boldsymbol{F_B}$, to north-east-down frame, $\boldsymbol{F_{NED}}$, is illustrated by three plane rotations in Table 3.2.

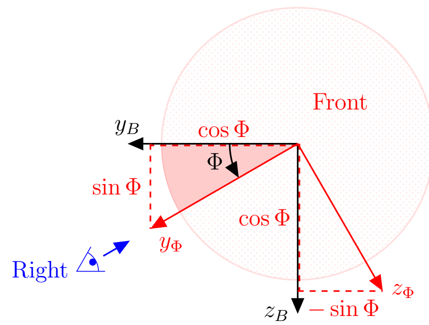
As an example, the array C_Φ from Table 3.2 may be read either left to right or down as $\boldsymbol{y_\Phi} = \boldsymbol{x_B}\ 0 + \boldsymbol{y_B}\ \cos\Phi - \boldsymbol{z_B}\ \sin\Phi$ and $\boldsymbol{y_B} = \boldsymbol{x_\Phi}\ 0 + \boldsymbol{y_\Phi}\ \cos\Phi + \boldsymbol{z_\Phi}\ \sin\Phi$ respectively. The transpose of C_Φ, ie. C_Φ^T allows us to get to $\boldsymbol{x_\Phi}$, $\boldsymbol{y_\Phi}$, $\boldsymbol{z_\Phi}$ from $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$, 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:

$$\begin{bmatrix} \boldsymbol{x_\Phi} \\ \boldsymbol{y_\Phi} \\ \boldsymbol{z_\Phi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Phi & \sin\Phi \\ 0 & -\sin\Phi & \cos\Phi \end{bmatrix} \begin{bmatrix} \boldsymbol{x_B} \\ \boldsymbol{y_B} \\ \boldsymbol{z_B} \end{bmatrix} \qquad \Longleftrightarrow \qquad \boldsymbol{F_\Phi} = C_\Phi \boldsymbol{F_B} \tag{3.1.14}$$

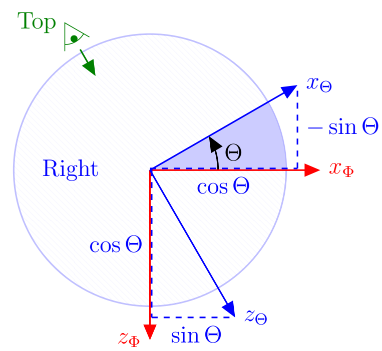
In this formulation C_Φ gets us to $\boldsymbol{x_B}$, $\boldsymbol{y_B}$, $\boldsymbol{z_B}$ from $\boldsymbol{x_\Phi}$, $\boldsymbol{y_\Phi}$, $\boldsymbol{z_\Phi}$. A variety of notations exist for direction cosine matrices, Stevens would write $C_{\boldsymbol{F_\Phi}/\boldsymbol{F_B}}$ instead of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

Table 3.2: Direction Cosine Matrices for Plane Rotations

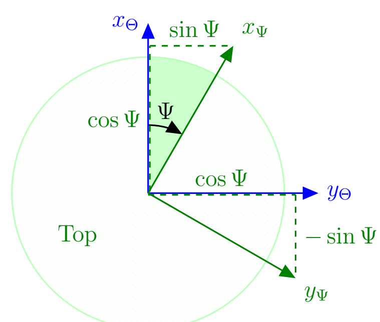
C_Φ	x_B	y_B	z_B
x_Φ	1	0	0
y_Φ	0	$\cos\Phi$	$\sin\Phi$
z_Φ	0	$-\sin\Phi$	$\cos\Phi$



C_Θ	x_Φ	y_Φ	z_Φ
x_Θ	$\cos\Theta$	0	$-\sin\Theta$
y_Θ	0	1	0
z_Θ	$\sin\Theta$	0	$\cos\Theta$



C_Ψ	x_Θ	y_Θ	z_Θ
x_Ψ	$\cos\Psi$	$\sin\Psi$	0
y_Ψ	$-\sin\Psi$	$\cos\Psi$	0
z_Ψ	0	0	1



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:

$$QQ^T = Q^TQ = I \tag{3.1.15}$$

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:

$$C_1 C_2 \neq C_2 C_1 \tag{3.1.16}$$

3. *Successive rotations may be described the 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 [14]; 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 3.4 depicts a complete coordinate transformation in a X-Y-Z (Roll-Pitch-Yaw) manner. This rotation sequence is suitable for calculating aircraft attitudes

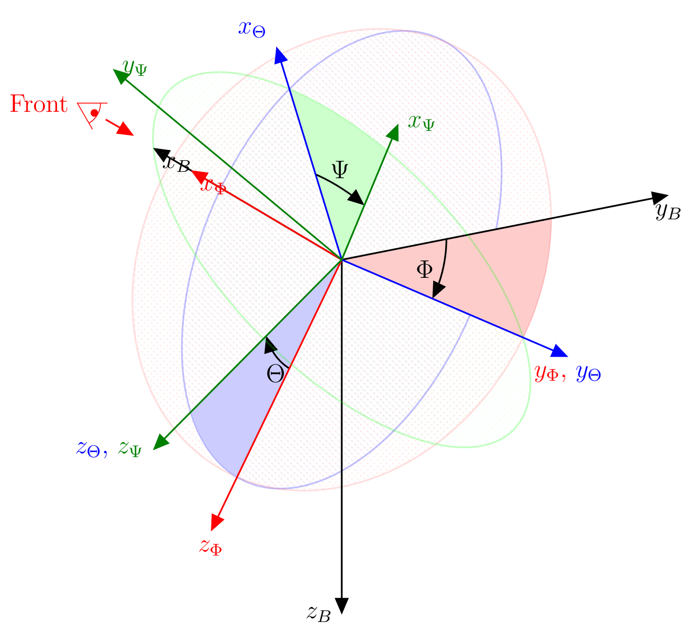


Figure 3.4: Direction Cosine Matrix Visual

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

In terms of coordinate transformations

$$\boldsymbol{F_B} = \left(C_\Phi C_\Theta C_\Psi \right) \boldsymbol{F_{NED}} \tag{3.1.17}$$

where Equation 3.1.18 is the complete coordinate transformation from north-east-down to the body frame, ie. $C_{\boldsymbol{F_{NED}}/\boldsymbol{F_B}} = C_\Phi C_\Theta C_\Psi$

$$C_\Phi C_\Theta C_\Psi = \begin{bmatrix} \cos\Theta\cos\Psi & \cos\Theta\sin\Psi & -\sin\Theta \\ -\cos\Phi\sin\Psi + \sin\Phi\sin\Theta\cos\Psi & \cos\Phi\cos\Psi + \sin\Phi\sin\Theta\sin\Psi & \sin\Phi\cos\Theta \\ \sin\Phi\sin\Psi + \cos\Phi\sin\Theta\cos\Psi & -\sin\Phi\cos\Psi + \cos\Phi\sin\Theta\sin\Psi & \cos\Phi\cos\Theta \end{bmatrix} \tag{3.1.18}$$

3.1.3 Aircraft Dynamics

With Assumptions 3.1 & 3.2 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 Newton's Laws. With this we can describe translational and rotational motion of the aircraft by its kinematic analogs: linear momentum, $\boldsymbol{p}$, and angular momentum, $\boldsymbol{H}$ respectively.

"By Newton's second law the time rate of change of linear momentum equals the sum of all *externally* applied forces, [$\boldsymbol{F}$].

$$\boldsymbol{F} = \frac{d}{dt}\left(\boldsymbol{p}\right) = \frac{d}{dt}\left(m\boldsymbol{V}\right) \tag{3.1.19}$$

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

$$\boldsymbol{M} = \frac{d}{dt}\left(\boldsymbol{H}\right) \tag{3.1.20}$$

These vector differential equations provide the starting point for a complete description of the rigid body motions of the vehicle." [12]

Assumption 3.3. 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].

3.1.3.1 Translational Acceleration

The goal is to derive equations for translation accelerations in the wind axis reference frame – $\dot{V}$, $\dot{\alpha}$, $\dot{\beta}$. Picking up where Equation 3.1.19 left off, along with Assumption 3.3, we may expand the expression to

$$\boldsymbol{F} = m\left[\frac{d}{dt}(\boldsymbol{V}) + \boldsymbol{\Omega} \times \boldsymbol{V}\right] \tag{3.1.21}$$

where $\boldsymbol{F}$ is the total force acing on the vehicle, m is the vehicle mass, $\boldsymbol{V}$ is the total vehicle velocity, and $\boldsymbol{\Omega}$ is the total vehicle angular velocity:

$$\boldsymbol{F} = \begin{bmatrix} \sum F_x \\ \sum F_y \\ \sum F_z \end{bmatrix} = \begin{bmatrix} F_{x_T} + F_{x_A} + F_{x_G} \\ F_{y_T} + F_{y_A} + F_{y_G} \\ F_{z_T} + F_{z_A} + F_{z_G} \end{bmatrix} \tag{3.1.22}$$

$$\boldsymbol{V} = [\,u,\, v,\, w\,]^T \tag{3.1.23}$$

$$\boldsymbol{\Omega} = [\,P,\, Q,\, R\,]^T \tag{3.1.24}$$

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

The body axis aerodynamic forces can be transformed to their equivalent stability axis components lift L, drag D, and side-force Y as Figure 3.5 indicates.

---

4 McRuer et al. [12] derives a modified extension of Equation 3.1.19 to take this into account.

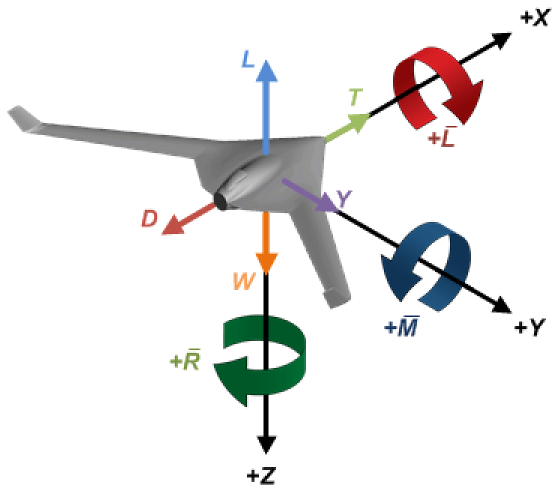


Figure 3.5: Body Axes, Forces, and Moments

$$F_{x_A} = -D\cos\alpha + L\sin\alpha \tag{3.1.25}$$

$$F_{y_A} = Y \tag{3.1.26}$$

$$F_{z_A} = -D\sin\alpha - L\cos\alpha \tag{3.1.27}$$

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

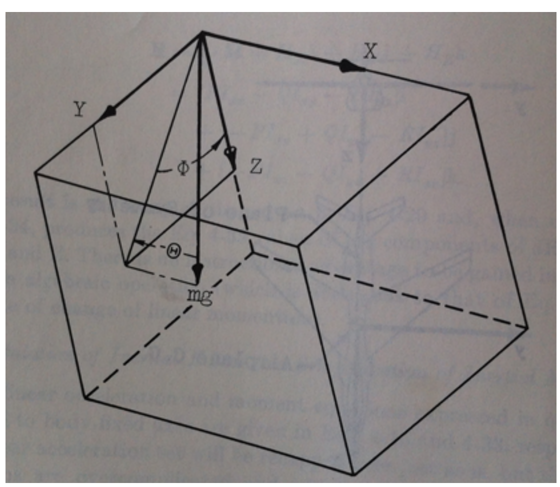


Figure 3.6: Orientation of Gravity Vector with Respect to Body Axes

$$F_{x_G} = -\, mg \sin\theta \tag{3.1.28}$$

$$F_{y_G} = \quad mg \sin\phi \cos\theta \tag{3.1.29}$$

$$F_{z_G} = \quad mg \cos\phi \cos\theta \tag{3.1.30}$$

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

$$\begin{bmatrix} \sum F_x \\ \sum F_y \\ \sum F_z \end{bmatrix} = \begin{bmatrix} F_{x_T} - D\cos\alpha + L\sin\alpha - mg\sin\theta \\ Y + mg\sin\phi\cos\theta \\ -D\sin\alpha - L\cos\alpha + mg\cos\phi\cos\theta \end{bmatrix} \tag{3.1.31}$$

By rearranging Equation 3.1.21 to solve for translational acceleration, $d\boldsymbol{V}/dt$, we can express body axis accelerations in terms of body axis forces, angular rates, and velocities:

$$\frac{d}{dt}(\boldsymbol{V}) = \frac{1}{m}\boldsymbol{F} - \boldsymbol{\Omega} \times \boldsymbol{V} \tag{3.1.32}$$

Substitution of Equations 3.1.23, 3.1.24, and 3.1.22 yields

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}(F_{x_T} + F_{x_A} + F_{x_G}) + Rv - Qw \\ \frac{1}{m}(F_{y_T} + F_{y_A} + F_{y_G}) + Pw - Ru \\ \frac{1}{m}(F_{z_T} + F_{z_A} + F_{z_G}) + Qu - Pv \end{bmatrix} \tag{3.1.33}$$

In order to express Equation 3.1.33 in the wind axis system, will need to use Equations 3.1.2, 3.1.3, and 3.1.4 as transforms

$$u = V\cos\beta\cos\alpha$$

$$v = V\sin\beta$$

$$w = V\cos\beta\sin\alpha$$

and Equations 3.1.8, 3.1.9, and 3.1.10

$$V = \sqrt{u^2 + v^2 + w^2}$$

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right)$$

$$\beta = \sin^{-1}\left(\frac{v}{\sqrt{u^2 + v^2 + w^2}}\right)$$

Leads to $\dot{V}$, $\dot{\alpha}$, and $\dot{\beta}$ equations

3.1.3.2 Rotational Acceleration

The goal is to derive rotational acceleration equations – $\dot{p}$, $\dot{q}$, $\dot{r}$. Picking up where Equation 3.1.20 left off and substituting total angular momentum for the product of the moment of inertia matrix and rotational velocity vector, $\boldsymbol{H} = \boldsymbol{I\Omega}$, we may expand the expression to

$$\boldsymbol{M} = \left[\frac{d}{dt}(\boldsymbol{I\Omega}) + \boldsymbol{\Omega} \times \boldsymbol{I\Omega}\right] \tag{3.1.34}$$

where $\boldsymbol{M}$ is the total moment acing on the vehicle, $\boldsymbol{I}$ is the inertia matrix (alternatively referred to as tensor or dyad), and $\boldsymbol{\Omega}$ is the total vehicle angular velocity:

$$\boldsymbol{M} = \begin{bmatrix} \sum L \\ \sum M \\ \sum N \end{bmatrix} = \begin{bmatrix} L + L_T \\ M + M_T \\ N + N_T \end{bmatrix} \tag{3.1.35}$$

L, M ,and 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.

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \tag{3.1.36}$$

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 3.4. 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 3.1.34 may be rewritten as

$$\frac{d}{dt}\boldsymbol{\Omega} = \boldsymbol{I}^{-1}\left(\boldsymbol{M} - \boldsymbol{\Omega} \times \boldsymbol{I}\boldsymbol{\Omega}\right) \tag{3.1.37}$$

$$\frac{d}{dt}\boldsymbol{\Omega} = [\,\dot{p},\, \dot{q},\, \dot{r}\,]^T \tag{3.1.38}$$

$$\boldsymbol{I}^{-1} = \frac{1}{\det I}\begin{bmatrix} I_1 & I_2 & I_3 \\ I_4 & I_5 & I_6 \\ I_7 & I_8 & I_9 \end{bmatrix} \tag{3.1.39}$$

Leads to $\dot{p}$, $\dot{q}$, and $\dot{r}$ equations

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \dots \tag{3.1.40}$$

3.1.3.3 Attitude Rates

Relative to velocity vector...

Leads to $\dot{\phi}$, $\dot{\theta}$, and $\dot{\psi}$ equations, but I need to derive in terms of flight path components $\dot{\mu}$, $\dot{\gamma}$, and $\dot{\chi}$

$$\begin{bmatrix} \dot{\mu} \\ \dot{\gamma} \\ \dot{\chi} \end{bmatrix} = \dots \tag{3.1.41}$$

3.1.3.4 Equations of Motion

Complete Set of Nonlinear EOM

$$\dot{\chi} = \frac{1}{mV\cos\gamma}\left[D\sin\beta\cos\mu + Y\cos\beta\cos\mu + L\sin\mu \right. \tag{3.1.42a}$$
$$\left. + T\left(\sin\alpha\sin\mu - \cos\alpha\sin\beta\cos\mu\right)\right]$$

$$\dot{\gamma} = \frac{1}{mV}\left[-D\sin\beta\sin\mu - Y\cos\beta\sin\mu + L\cos\mu \right. \tag{3.1.42b}$$
$$\left. + T\left(\sin\alpha\cos\mu + \cos\alpha\sin\beta\sin\mu\right)\right] - \frac{g}{V}\cos\gamma$$

$$\dot{V} = \frac{1}{m}\left(-D\cos\beta + Y\sin\beta + T\cos\alpha\cos\beta\right) - g\sin\gamma \tag{3.1.42c}$$

$$\dot{\mu} = \frac{1}{mV}\left[D\sin\beta\tan\gamma\cos\mu + Y\cos\beta\tan\gamma\cos\mu + L\left(\tan\beta + \tan\gamma\sin\mu\right)\right.$$
$$\left. + T\left(\sin\alpha\tan\gamma\sin\mu + \sin\alpha\tan\beta - \cos\alpha\sin\beta\tan\gamma\cos\mu\right)\right] \tag{3.1.42d}$$
$$- \frac{g\tan\beta\cos\gamma\cos\mu}{V} + \frac{P\cos\alpha + R\sin\alpha}{\cos\beta}$$

$$\dot{\alpha} = -\frac{1}{mV\cos\beta}\left(L + T\sin\alpha\right) + \frac{g\cos\gamma\cos\mu}{V\cos\beta} + Q \tag{3.1.42e}$$
$$- \tan\beta\left(P\cos\alpha + R\sin\alpha\right)$$

$$\dot{\beta} = \frac{1}{mV}\left(D\sin\beta + Y\cos\beta - T\sin\beta\cos\alpha\right) + \frac{g\cos\gamma\sin\mu}{V} \tag{3.1.42f}$$
$$P\sin\alpha - R\cos\alpha$$

$$\dot{P} = \left(c_1 R + c_2 P\right) Q + c_3 \bar{L} + c_4 \bar{N} \tag{3.1.42g}$$

$$\dot{Q} = c_5 P R - c_6\left(P^2 - R^2\right) + c_7 \bar{M} \tag{3.1.42h}$$

$$\dot{R} = \left(c_8 P - c_2 R\right) Q + c_4 \bar{L} + c_9 \bar{N} \tag{3.1.42i}$$

where [13] defines c terms as

$$\begin{aligned}
\Gamma &= I_{xx}I_{zz} - I_{xz}^2 \\
\Gamma c_1 &= \left(I_{yy} - I_{zz}\right) I_{zz} - I_{xz}^2 \\
\Gamma c_2 &= \left(I_{yy} - I_{zz}\right) I_{zz} - I_{xz}^2 \\
\Gamma c_3 &= I_{zz} \\
\Gamma c_4 &= I_{xz} \\
c_5 &= \frac{I_{zz} - I_{xx}}{I_{yy}} \\
c_6 &= \frac{I_{xz}}{I_{yy}} \\
c_7 &= \frac{1}{I_{yy}} \\
\Gamma c_8 &= I_{xx}\left(I_{xx} - I_{yy}\right) + I_{xz}^2 \\
\Gamma c_9 &= I_{xx}
\end{aligned} \tag{3.1.43}$$

3.1.4 System Observations

The nonlinear differential equations summarized in § 3.1.3.4 may be reduced to the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, t\right) \tag{3.1.44}$$

where $\dot{\boldsymbol{x}}$ is the $n \times 1$ derivative of the state vector with respect to time, $\boldsymbol{f}$ is an $n \times 1$ non-linear vector function expressing the six-degree of freedom rigid body equations, and $\boldsymbol{x}$ is the $n \times 1$ state-vector with respect to time. Additionally, the state-vector is defined as a set of real numbers, $\left(x_1, \ldots, x_n\right)^T$, 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).

[5] The highest derivative of the dependent variable with respect to the independent variable appearing in the equation.

Equation 3.1.44 represents the closed-loop time-variant dynamics of a feedback control system, even though it does not explicitly contain a control input vector $\boldsymbol{u}$. This is because the control input may be considered a function of state $\boldsymbol{x}$ and time t, therefore *disappearing* in the closed-loop dynamics. Showing this mathematically, if the plant dynamics are

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}, t\right) \tag{3.1.45}$$

and some control law $\boldsymbol{u}$ has been selected as

$$\boldsymbol{u} = \boldsymbol{g}\left(\boldsymbol{x}, t\right) \tag{3.1.46}$$

then the closed-loop dynamics are

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left[\boldsymbol{x}, \boldsymbol{g}\left(\boldsymbol{x}, t\right), t\right] \tag{3.1.47}$$

which can be rewritten in the form of Equation 3.1.44; since $\boldsymbol{g}(\boldsymbol{x}, t)$ is a function of the state $\boldsymbol{x}$, which is already represented in the expression, it may be discarded. Naturally, Equation 3.1.44 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 $\boldsymbol{f}$, ie. $\lim_{x \to a} \boldsymbol{f}(x) = \boldsymbol{f}(a)$, 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 $\boldsymbol{f}(x)$ having an infinite slope, eg. a discontinuity.

Definition 3.1.1 (Lipschitz Condition). Khalil [1, §2.2]

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

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

then the function $\boldsymbol{f}(\boldsymbol{x}, t)$ is said to be *Lipschitz in* $\boldsymbol{x}$ 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 $\boldsymbol{f}(\boldsymbol{x},t)$. A function is said to be *locally Lipschitz in* $\boldsymbol{x}$ if that for each point $\boldsymbol{x} \in \mathcal{D} \subset \mathbb{R}^n$ 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 3.1.1 (Global Existence and Uniqueness)**.** *Khalil [1, Thm 2.4]*
Let $\boldsymbol{f}(\boldsymbol{x},t)$ *be piecewise continuous in* t *and* ***locally Lipschitz in*** $\boldsymbol{x}$ *for all* $t \geq t_0$ *and all* $\boldsymbol{x}$ *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*

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

lies entirely in W*. Then there is a* ***unique solution*** *that is defined for all* $t \geq t_0$

Proof: Refer to Khalil [1, Pg. 77] □

"The trick in applying Theorem 3.1.1 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 3.1.2 (Autonomous and Non-Autonomous Systems)**.** Slotine and Weiping [2]
The non-linear system $\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x},t\right)$ is said to be **autonomous** if $\boldsymbol{f}$ does not depend explicitly on time, ie. if the system's state equation can be written

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \tag{3.1.48}$$

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."[2] Most importantly, Definition 3.1.2 implies that solutions, or system trajectories, of autonomous ODEs are independent of initial time, thereby greatly simplifying stability analysis. In other words, stability does not depend on initial conditions!

3.2 Stability

The goal is to familiarize the reader with concepts required to prove stability for the backstepping control architecture. Proofs will not be included[6], 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 [1, Chp. 2]. The following standard mathematical abbreviation symbols shall be used:

$\forall$	"for all"
$\exists$	"there exists"
$\ni$	"such that"
$\in$	"in" or "belongs to"
$\subset$	"a subset of"
$\Rightarrow$	"implies"

Backstepping control designs are constructed using Lyapunov stability theory. This is currently the most useful and general approach [2] to establishing stability for non-linear 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 § 3.3.3, 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

[6] Consider including in appendix?

analysis, indirect and direct. The first method[7], 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 non-linear only Lyapunov's direct method will be covered.

Ultimately, this section condenses the philosophy, definitions, and theorems of Slotine and Weiping [2, Chp. 3], Khalil [1, Chp. 3], Härkegå rd [15, Chp. 3], Krstic et al. [9, Chp. 2] and Farrell and Polycarpou [4, Appendix A] in a brief and clear manner.

3.2.1 Equilibrium and Operating Points

A system trajectory may correspond to only a single point $\boldsymbol{x}^*$, called an **equilibrium point**, if once $\boldsymbol{x}(t)$ is equal to $\boldsymbol{x}^*$ it remains equal to $\boldsymbol{x}^*$ for all time. For the non-autonomous system in Equation 3.1.44, the equilibrium points are the real roots (x-intercepts (for $n = 2$ systems)) of the differential equation, that is

$$\boldsymbol{f}\left(\boldsymbol{x}^*, t\right) = 0, \quad \forall t \geq 0 \tag{3.2.1}$$

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 $\boldsymbol{u}$ in Equation 3.1.46, the vectors $(\boldsymbol{x_0}, \boldsymbol{u_0})$ are operating points if

$$\boldsymbol{f}\left(\boldsymbol{x_0}, \boldsymbol{u_0}, t\right) = 0, \quad \forall t \geq 0 \tag{3.2.2}$$

[7] In my research I found conflicting evidence on what Lyapunov's first method actually was. [2] 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.

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**[8] as trajectories from $t = (0, \infty)$.

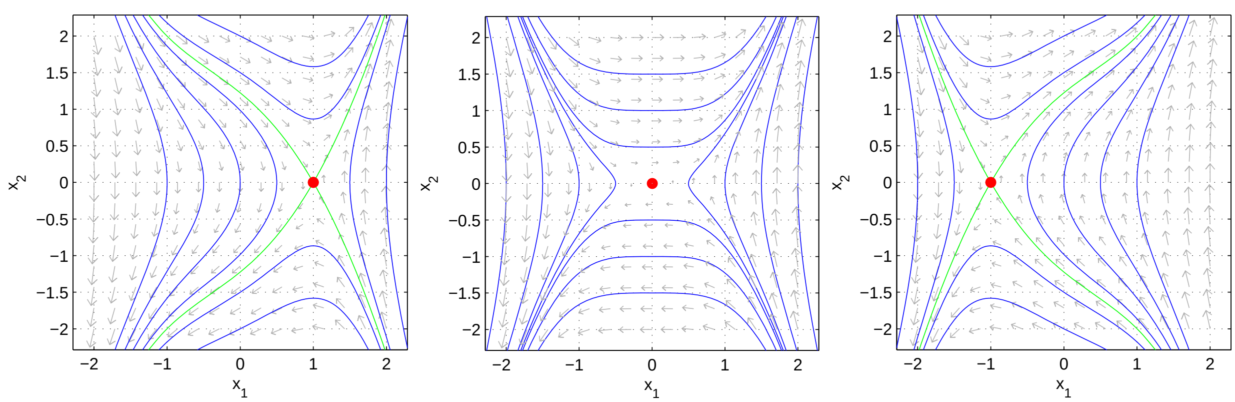
Example 3.2.1 (Operating Points)**.**

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 $\boldsymbol{u_0} = -1$, an operating point emerges at $\boldsymbol{x_0} = [1, 0]^T$. If the control input is varied, we can see the surface of operating points:



(a) $\boldsymbol{u_0} = -1, \quad \boldsymbol{x_0} = [1, 0]^T$ (b) $\boldsymbol{u_0} = 0, \quad \boldsymbol{x_0} = [0, 0]^T$ (c) $\boldsymbol{u_0} = 1, \quad \boldsymbol{x_0} = [-1, 0]^T$

Figure 3.7: Operating Surface via Phase Portraits

Visualization of trajectories, through **phase portraits**[9], provides an intuitive feel for

[8] 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 [9] Made using pplane8, http://math.rice.edu/~dfield/index.html

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 ($\rightarrow$) 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 $\boldsymbol{x_0} = [a, 0]^T$ with $\boldsymbol{u_0} = -a^3$. 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 3.7 shows, the operating point is not always coincident with the state-space origin, ie. $\boldsymbol{x} = \boldsymbol{0}$. 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 $\mathcal{O}^3$ and above. Using notation from [9, Pg. 23]:

$$\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{x}^* \tag{3.2.3}$$

To prove this statement, substitute a reformulation of previous expression, $\boldsymbol{x} = \boldsymbol{z} + \boldsymbol{x}^*$, into Equation 3.1.44 as shown:

$$\dot{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{z} + \boldsymbol{x}^*, t) \tag{3.2.4}$$

It is clear that by substituting Equation 3.2.3 into Equation 3.2.4 one would arrive at a system equivalent to the original, ie. $\dot{\boldsymbol{z}} = \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t)$. "In addition $\boldsymbol{z} = 0$, the solution corresponding to the original system's equilibrium point $\boldsymbol{x} = \boldsymbol{x}^*$, is an equilibrium point of Equation 3.2.4; [recall Equation 3.2.1]. Therefore, instead of studying the behavior of the original system, Equation 3.1.44, in the neighborhood of $\boldsymbol{x}^*$, one can equivalently study behavior of the redefined system, Equation 3.2.4, in the neighborhood of the origin"[2] ; 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.

3.2.2 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 non-linear 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 non-linear 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 ϵ denote a spherical region defined by $\|\boldsymbol{x}\| < \epsilon$ in state-space and δ denote a spherical region that is generally within ϵ; δ is called a domain of attraction. Recall that equilibrium points may be translated to the origin by redefining the state-vector as shown in Equation 3.2.3. The theorems in this section will be presented as if the equilibrium point was translated to the origin; note that $\boldsymbol{z}$ notation will be dropped, and the usual $\boldsymbol{x}$ representation will be used to represent a system with redefined equilibrium point.

Definition 3.2.1 (Stability in the Sense of Lypaunov). Khalil [1, Def 3.2]

For the non-autonomous system

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

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 $\boldsymbol{x}^* = 0$ is

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

$$\|\boldsymbol{x}(t_0)\| < \delta \quad \Rightarrow \quad \|\boldsymbol{x}(t)\| < \epsilon \quad , \qquad \forall\, t \geq t_0 \geq 0 \tag{3.2.5}$$

- **uniformly stable** if for each $\epsilon > 0$, there exists a variable $\delta = \delta(\epsilon) > 0$, *independent of* t_0 such that Equation 3.2.5 is satisfied
- **unstable** if not stable.
- **asymptotically stable** if it is stable and there exists a constant $c = c(t_0) > 0 \ni$

$$\|\boldsymbol{x}(t_0)\| < c \quad \Rightarrow \quad \lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{x}^* = 0 \tag{3.2.6}$$

- **uniformly asymptotically stable** if it is uniformly stable and there exists a constant $c > 0$ *independent of* t_0, such that Equation 3.2.6 is satisfied uniformly in t_0; that is, for each $\eta > 0$, there is $T(\eta) > 0$ such that

$$\|\boldsymbol{x}(t)\| < \eta \quad , \qquad \forall\, t \geq t_0 + T(\eta) \quad , \quad \forall\, \|\boldsymbol{x}(t_0)\| < c \tag{3.2.7}$$

- **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

$$\|\boldsymbol{x}(t)\| < \eta \quad , \qquad \forall\, t \geq t_0 + T(\eta, c) \quad , \quad \forall\, \|\boldsymbol{x}(t_0)\| < c \tag{3.2.8}$$

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

$$\|\boldsymbol{x}(t_0)\| < \delta \quad \Rightarrow \quad \|\boldsymbol{x}(t)\| < \epsilon e^{-\lambda(t-t_0)} \quad , \quad \forall\, t > t_0 \geq 0 \tag{3.2.9}$$

I think that asymptotic stability uses c–η instead of ϵ–δ because its already used in the stability def and asymptotic stability requires stability. Also, if you chose delta and epsilon large enough couldn't any system be stable?

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."[4].

With regards to *uniform* stability, the additional stipulation over ordinary stability is that δ is independent of t_0; all properties are uniform if the system is time-invariant. Important in the equilibrium points with forms of asymptotically stable is the additional stipulation on initial time or states, ie. $\|\boldsymbol{x}(t_0)\| = \|\boldsymbol{x_0}\| < c(t_0)$. 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 $\mathbb{R}^n$. I DONT SEE THIS EXPLICITLY IN KHALIL'S DEF. 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 3.2.1, the state vector of the exponentially stable system converges faster than an exponential function, where the positive number λ is the rate of exponential convergence. "By writing the positive constant ϵ 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 $\|\boldsymbol{x}(t)\|$ will be less than 5% ($\approx e^{-3}$) of $\|\boldsymbol{x}(0)\|$" [2]. 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 [1, Sec. 2.1]. It's also important to remember that these are merely definitions; positive constants δ, ϵ, c, and η are *arbitrarily fixed bounds* that are used to evaluate solution trajectories.

For systems up to second order ($n = 1, 2$), a two-dimensional[10] 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 $\dot{\boldsymbol{x}}$ at each $\boldsymbol{x}$. "The behavior of the solution in the neighborhood of the origin can be determined by examining the sign of $\boldsymbol{f}(x)$. The $\epsilon - \delta$ requirement for stability is violated if $\boldsymbol{x}(\boldsymbol{f}(x)) > 0$ on either side of the origin."

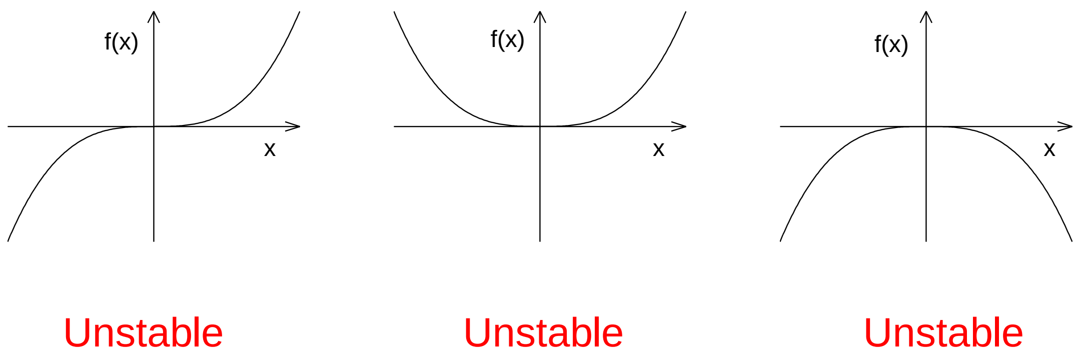


Figure 3.8: First Order Unstable Systems [1, Online Lecture Notes]

"The origin is stable if and only if $\boldsymbol{x}\boldsymbol{f}(x) \leq 0$ in some neighborhood of the origin"

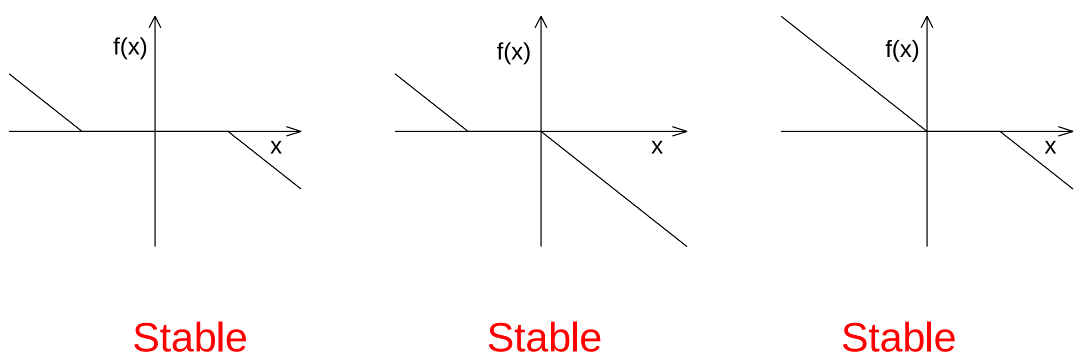
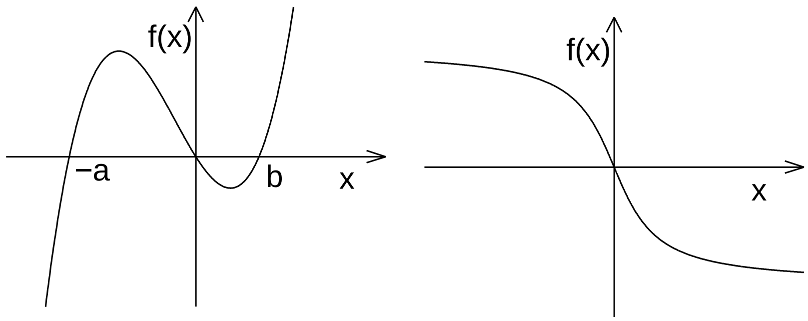


Figure 3.9: First Order Stable Systems [1, Online Lecture Notes]

"The origin is asymptotically stable if and only if $\boldsymbol{x}\boldsymbol{f}(x) < 0$ in some neighborhood of the origin"

[10] 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.



(a) Asymptotically Stable (b) Globally Asymptotically Stable

Figure 3.10: First Order Asymptotically Stable Systems [1, Online Lecture Notes]

In two-dimensions, Figure 3.11 depicts stable (S), unstable (US), and asymptotically stable (AS) trajectories in phase-space with respect to ϵ and δ contours in Definition 3.2.1; the initial condition is $x(0)$, solid line is the trajectory $x(t)$, and origin is the equilibrium point x^*.

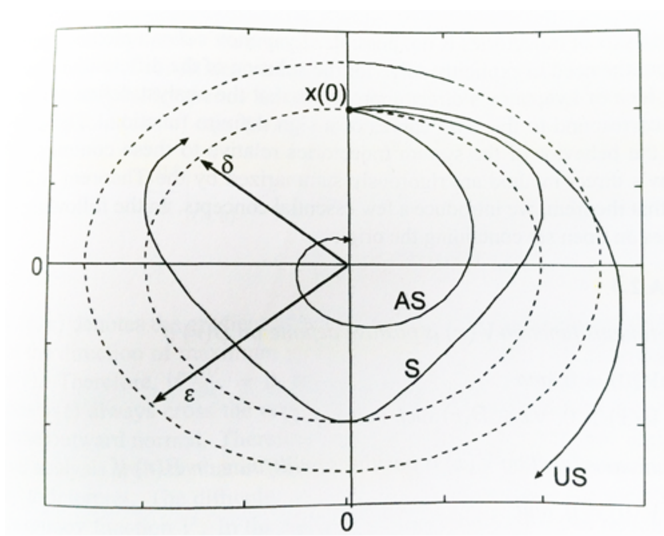


Figure 3.11: Concepts of Stability

This representation is a great conceptual tool, as it brings tangible meaning to the definitions. It may be used to analyze non-linear systems, especially useful for those with multiple equilibrium points, as around each point a linear system approximation is valid. In practice, many non-linear 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 3.12 illustrates possible equilibrium point classifications in equivalent root-loci and phase-portraits:

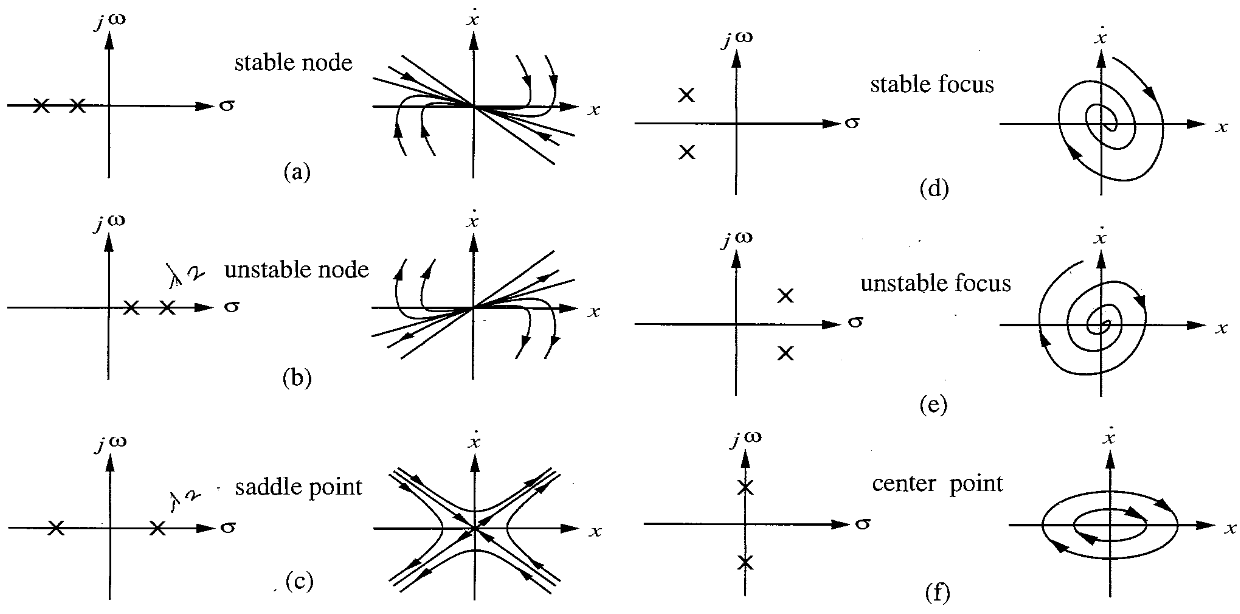


Figure 3.12: Equilibrium Point Classification for 2^{nd} Order Linear Systems [2]

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.

3.2.2.1 Perturbed Systems

These concepts evaluate the stability of an *ideal* system. Undoubtedly there are model uncertainties[11] and disturbances that effect the true stability of the air vehicle. Boundedness

[11] Errors in modeled parameters are called parametric uncertainties, while neglected or unknown parameters are non-parametric uncertainties.

concepts from stability analysis in the Lyapunov sense may be applied to a reformulated system if uncertainties do not effect system order, as:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t) + \boldsymbol{g}(\boldsymbol{x}, t) \tag{3.2.10}$$

where $\boldsymbol{f}(\boldsymbol{x}, t)$ is the nominal system and $\boldsymbol{g}(\boldsymbol{x}, t)$ is the perturbation term. Typically we don't know $\boldsymbol{g}(\boldsymbol{x}, t)$ but we know something about its bounds, for example an upper bound on $\|\boldsymbol{g}(\boldsymbol{x}, t)\|$. Note that if $\boldsymbol{g}(\boldsymbol{x}, t) \neq 0$ the origin is not necessarily an equilibrium point of Equation 3.2.10. 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 3.2.2 (Total Stability). Slotine and Weiping [2, Defn. 4.13]
The equilibrium point $\boldsymbol{x}^* = \boldsymbol{x}(0) = 0$ for the unperturbed system in Equation 3.1.48, $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$, is said to be totally stable if for every $\epsilon \geq 0$, two numbers exist δ_1 and δ_2 exist such that $\|\boldsymbol{x}(t_0)\| < \delta_1$ and $\|\boldsymbol{g}(\boldsymbol{x}, t)\| < \delta_2$ imply that every solution $\boldsymbol{x}(t)$ of the perturbed system Equation 3.2.10 satisfies the condition $\|\boldsymbol{x}(t_0)\| < \epsilon$

"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 § 3.2.3.

Theorem 3.2.1 (Asymptotic Stablity and Total Stability). *[2, Thm. 4.14]*
If the equilibrium point $\boldsymbol{x}^* = \boldsymbol{x}(0) = 0$ for the unperturbed system in Equation 3.1.48, $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$, is uniformly asymptotically stable, then it is totally stable.

Proof: Refer to Khalil [1, Chp. 5, Stability of Perturbed Systems] □

3.2.3 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 3.3 outlines the theorems that will be covered.

Table 3.3: Stability Theorem Overview

$\dot{\boldsymbol{x}} = \boldsymbol{f}(x)$	$V > 0$	$\dot{V} \leq 0$	S	Lyapunov's Direct Method	Thm. 3.2.2
	$V > 0$	$\dot{V} < 0$	AS		
	$V > 0$	$\dot{V} < 0$	GAS		
	$V \geq 0$	$\dot{V} \leq 0$	CONV	LaSalle	Thm. 3.2.3
	$V > 0$	$\dot{V} \leq 0$	GAS	Barbashin–Krasovskii	Cor. 3.2.1
$\dot{\boldsymbol{x}} = \boldsymbol{f}(x, t)$	$V \geq 0$	$\dot{V} \leq 0$	CONV	Barbalat's Lemma	Lem. 3.2.2
	$V > 0$	$\dot{V} \leq 0$	GUAS	LaSalle-Yoshizawa	Thm. 3.2.4

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

Definition 3.2.3. 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 $\|\boldsymbol{x}\| \to \infty$

Where $V(x)$ is assumed to be a scalar function on $\mathcal{D}$ into $\mathbb{R}$, ie. $V : \mathcal{D} \to \mathbb{R}$, is continuously differentiable, and is defined in a domain $\mathcal{D} \subset \mathbb{R}^n$ that contains the origin $\boldsymbol{x} = 0$. When a function is positive or negative (semi)definite 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 3.1.48 is obtained by the chain rule:

$$\dot{V}(\boldsymbol{x}) = \frac{d}{dt}V(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}}\frac{d\boldsymbol{x}}{dt} = \frac{\partial V}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \frac{\partial V}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \tag{3.2.11}$$

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

Theorem 3.2.2 (Lyapunov's Direct Method). *Khalil [1, Thm. 3.1]*
Let the origin be an equilibrium point, $\boldsymbol{x}^* = \boldsymbol{x}(0) = 0$, for an autonomous system, $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ and $V : \mathbb{R}^n \to \mathbb{R}$ a continuously differentiable, ***positive definite*** *function, then*

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

$$\dot{V}(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \leq 0 \quad , \quad \forall\, \boldsymbol{x} \in \mathcal{D} \tag{3.2.12}$$

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

$$\dot{V}(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) < 0 \quad , \quad \forall\, \{\boldsymbol{x} \in \mathcal{D}\,|\,\boldsymbol{x} \notin 0\} \tag{3.2.13}$$

Proof: Refer to Khalil [1, Pg. 101] □

Remark (Theorem 3.2.2) 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 $\boldsymbol{x}$ through a relationship between $\boldsymbol{f}$ and a positive definite function, $V(\boldsymbol{x})$. This theorem asserts that if the system loses energy over time it will eventually reach an equilibrium condition.

A function $V(\boldsymbol{x})$ *satisfying Theorem 3.2.2* 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[12] 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 3.13 demonstrates this analogy for decreasing values of c.

Formally, each contour, c, characterizes a level set[13] of the Lyapunov function $V(\boldsymbol{x}) = c$ 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. $V(\boldsymbol{x})$, 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.

I think I've cleared up my initial confusion. 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 3.2.11 may be considered as the gradient of V with respect to $\boldsymbol{x}$, representing a vector pointing in the direction of maximum increase of V. The vector $d\boldsymbol{x}/dt$ represents the system dynamics, which could've equivalently been labeled $\boldsymbol{f}(\boldsymbol{x})$, and is tangent to the solution $\boldsymbol{x}(t)$. The condition imposed by Lyapunov, ie. $\dot{V}(\boldsymbol{x}) = (\partial V/\partial \boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) \leq 0$, implies that solutions $\boldsymbol{x}(t)$ always cross the contours of V with an angle greater than or equal to 90° relative to the outward normal; if $d\boldsymbol{x}/dt$ points inward then system trajectories will always move to smaller and smaller values of V. $\dot{V}(\boldsymbol{x}) \leq 0$ does

[12] 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. [13] A level set is a collection of equilibrium points.

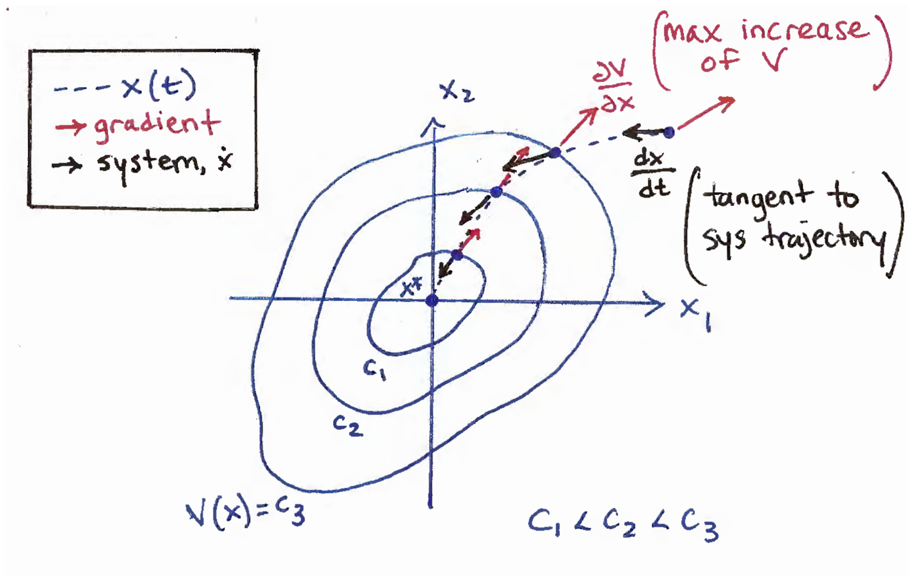


Figure 3.13: Geometric Interpretation of Lyapunov's Stability Theorem.

not ensure that a trajectory will get to the origin[14], but does imply the origin is stable, since by Definition 3.2.1: when a trajectory starts within δ (a level surface for this case), it will stay within ϵ.

When an equilibrium point is classified as asymptotically stable it requires the initial condition to be within some domain $\mathcal{D}$, 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 3.2.3.

Theorem 3.2.2 (continued) *Khalil [1, Thm. 3.2]*

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

$$\dot{V}(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) < 0 \quad , \quad \forall\, \boldsymbol{x} \neq 0 \tag{3.2.14}$$

then the origin is ***globally asymptotically stable (GAS)****.*

[14] LaSalle's Invariance Principle [1, 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 $\dot{V}(\boldsymbol{x}) = 0$.

Proof: Refer to Khalil [1, 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." [4] 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 [insert funny simile here]. 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*[15]. If a sub-system of a non-linear 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.

For cases where $\dot{V}$ is only negative semi-definite, ie. $\dot{V} \leq 0$, global asymptotic stability may still be established through LaSalle's[16] Invariance Principle, an invariant set theorem:

Lemma 3.2.1 (Fundamental Property of Limit Sets). *Khalil [1, Lem. 3.1]*

If a solution $\boldsymbol{x}(t)$ *to the autonomous system* $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ *is bounded and belongs to* $\mathcal{D}$ *for* $t \geq 0$,

[15] Converse Theorems, see Khalil [1, §3.6] [16] 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.

then its positive limit set L^+ is a nonempty, compact, invariant set. Moreover,

$$\lim_{t\to\infty} \boldsymbol{x}(t) = L^+ \tag{3.2.15}$$

Theorem 3.2.3 (LaSalle's Theorem)**.** *Khalil [1, Thm. 3.4]*

Let $\Omega \subset \mathcal{D}$ be a compact set that is positively invariant for an autonomous system, $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$. Let $V : \mathcal{D} \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(\boldsymbol{x}) \leq 0$ in Ω. Let $\mathcal{E}$ be the set of all points in Ω where $\dot{V}(\boldsymbol{x}) = 0$. Let $\mathcal{M}$ be the largest invariant set in $\mathcal{E}$. Then every solution starting in Ω approaches $\mathcal{M}$ as $t \to \infty$.

Proof: Refer to Khalil [1, 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 3.2.2 to be applied for cases where the system has an equilibrium set, ie. dynamic convergence or limit cycles, rather than a single equilibrium point.

Corollary 3.2.1 (Barbashin-Krasovskii)**.** *Khalil [1, Cor. 3.2]*

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

$$\dot{V}(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) \leq 0 \quad , \quad \forall\, \boldsymbol{x} \tag{3.2.16}$$

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

Remark (Corollary 3.2.1) When $\dot{V}(\boldsymbol{x}) \leq 0$ and $\mathcal{S} = \{0\}$, Corollary 3.2.1 coincides with Theorem 3.2.2. 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 $\boldsymbol{x}$ and t, ie. $V(\boldsymbol{x}, t)$, that has a negative-definite derivative. For tracking analysis, tools developed by LaSalle, Yoshizawa, and Barbalat are relied upon.

Theorem 3.2.4 (LaSalle–Yoshizawa). *Krstic et al. [9, Thm. 2.1/A.8]*
Let the origin be an equilibrium point, $\boldsymbol{x}^* = \boldsymbol{x}(0) = 0$, for a non-autonomous system, $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t)$ and suppose $\boldsymbol{f}$ is locally Lipschitz in $\boldsymbol{x}$ uniformly in t. Let $V : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}_+$ be a continuously differentiable, positive definite, and radially unbounded function $V = V(\boldsymbol{x}, t)$, then

$$\dot{V}(\boldsymbol{x}, t) = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, t) \leq -W(x) \leq 0 \quad , \quad \forall\, t \geq 0 \quad , \quad \forall\, \boldsymbol{x} \in \mathbb{R}^n \tag{3.2.17}$$

Where $W(x)$ is a continuous function. Then all solutions of [Equation 3.2.17] are globally uniformly bounded and satisfy

$$\lim_{t \to \infty} W(\boldsymbol{x}(t)) = 0 \tag{3.2.18}$$

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

Proof: Refer to Krstic et al. [9, Appendix A, Pg. 492] □

A technical lemma by Barbalat usually precedes Theorem 3.2.4 which is a purely mathematical result inferred by asymptotic properties of functions and their derivatives:

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

Example 3.2.2. As the derivative term converges to zero, the solution does not, for the system:

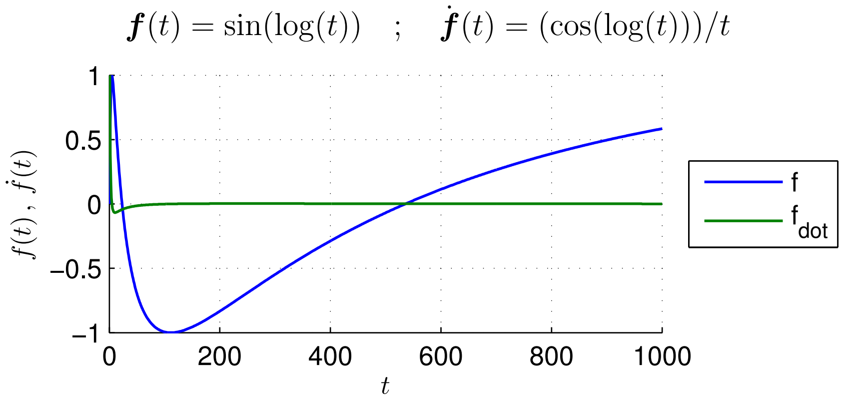


Figure 3.14: Asymptotic Property 1: $\dot{\boldsymbol{f}}(t) \to 0 \not\Rightarrow \boldsymbol{f} \to$ constant

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

  Example 3.2.3. As the solution tends to zero, the derivative is unbounded, for the system:

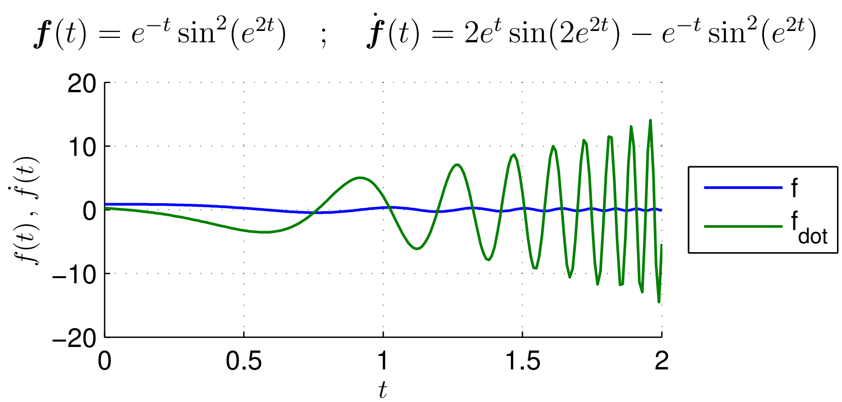


Figure 3.15: Asymptotic Property 2: $\boldsymbol{f} \to$ constant $\not\Rightarrow \dot{\boldsymbol{f}}(t) \to 0$

- If $\boldsymbol{f}(t)$ is lower bounded and decreasing, ie. $\dot{\boldsymbol{f}}(t) \leq 0$, then $\boldsymbol{f}(t)$ converges to a limit, but $\dot{\boldsymbol{f}}(t)$ **is not guaranteed to diminish if at all**.

An additional "smoothness" property on the Lyapunov derivative imposed by Barbalat's lemma guarantees that $\dot{\boldsymbol{f}}(t)$ actually converges to zero. This result ensures that a system will fulfill its tracking requirement.

Lemma 3.2.2 (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 [2]*

  "If the differentiable function $\boldsymbol{f}(t)$ has a finite limit as $t \to \infty$, and is such that $\ddot{\boldsymbol{f}}$ exists and is bounded, [ie. uniformly continuous], then:"

$$\lim_{t\to\infty} \dot{\boldsymbol{f}}(t) = 0$$

- ***Form 2:*** *"Lyapunov-Like Lemma"* *Slotine and Weiping [2]*

  If a scalar function $V(\boldsymbol{x}, t)$ satisfies the following conditions:

  - $V(\boldsymbol{x}, t)$ *is lower bounded*
  - $\dot{V}(\boldsymbol{x}, t)$ *is negative semi-definite*
  - $\dot{V}(\boldsymbol{x}, t)$ *is uniformly continuous in time (by proving $\ddot{V}$ is bounded)*

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

- ***Form 3:*** *$\mathcal{L}_p$ Space Representation* *Farrell and Polycarpou [4, 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 [1, 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 [2, 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 non-linearities and developing a stabilizing feedback control law.

3.3 Control

The aim of this section is to introduce control law design considerations and thoroughly present backstepping control theory. A simplified first order non-linear 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 non-linear equations of motion formulated in § 3.1.3.4: Equation 3.1.42.

3.3.1 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. [9]

Loosely speaking, the approach to non-linear 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 the term quantitative. For non-linear 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, non-linear 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 a analysis tools are limited for non-linear systems, each system is unique and exhibited motion is dependent on inputs. As a result, for non-linear 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 non-linear 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, non-linearities 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 3.1.3. Other benefits and disadvantages of backstepping will be covered, but first non-linear control design considerations and options are briefly introduced.

As outlined by Slotine and Weiping [2], 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 each of these characteristics to the 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 non-linear sense does not imply that a system is capable of handling *constant* disturbances; recall from Definitions 3.2.1 and 3.2.2 that stability is defined with respect to initial conditions. For example, a system is stable in the Lyapunov sense if a trajectory starts *within* δ and stays within ϵ; a persistent wind-shear – erratic disturbance due to thermals, downdraft, inversion layers, etc. – may shift the equilibrium point, therefore starting within the δ 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." [5]

3.3.2 Objective and Methods

Definition 3.3.1. 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 3.2.3, while tracking problems rely on Barbalat's Lemma 3.2.2.

One of the simplest and aging rivals of backstepping is **feedback linearization**, or nonlinear dynamic inversion. In backstepping literature this non-linear control architecture usually plays the victim; it is a point solution that renders the system linear by constructing control laws that cancel non-linear 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 3.4 covers features of a few non-linear control techniques.

Backstepping and feedback linearization cancel *known* nonlinearities... Farrell et al. [16]

Table 3.4: Nonlinear Control Method Overview

Method	Advantage	Disadvantage
Trial and Error	• Visual stability analysis via phase portraits	• Applicable only to simple systems up to second order
Small Singular Linearization / Gain Scheduling	• Good closed-loop performance for a equilibrium point (SSL). • Good closed-loop performance over many equilibrium points (GS).	• Accurate only in a neighborhood around operating point(s) • Controller parameters fixed online • A lot of offline validation required
Feedback Linearization	• Globally stable with exponential tracking error • Linear in modeled domain • Bandwidth theoretically infinite for input signal tracking	• Lack of controllability at singularities • Requires exact knowledge and special class of system • More control effort is required • Not robust to uncertainties
Backstepping / Robust / Adaptive	• Globally asymptotically stable • Model uncertainties well handled • Systematic procedures • Potential reduction in development time • Useful non-linearities retained	• Analytic derivative calculation • Feedback control algorithm complex, especially for high order systems

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 non-linear feedback are used, rather than the latter where linear *approximations* of the nonlinear system and linear feedback are used. The downside is that all non-linearities must be precisely known. Furthermore, even the "worst case performance attained by a non-linear controller coincides with the performance attained by the best linear design." Kokotovic [6, Linear Versus Nonlinear]

Brush up on FBL, Krstic et al. [9, Pg. 39]

"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."

3.3.3 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 [5] 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 non-linear system it's applied to has an explicit dependence on u, that is:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, u) \quad , \quad \boldsymbol{x} \in \mathbb{R}^n \quad , \quad \boldsymbol{u} \in \mathbb{R} \quad , \quad \boldsymbol{f}(0,0) = 0 \tag{3.3.1}$$

If a stabilizing feedback control law $\alpha(\boldsymbol{x})$ is chosen for control input u such that the inequality in Equation 3.3.2 holds, implying $\dot{\boldsymbol{x}} = \boldsymbol{f}(x, \alpha(\boldsymbol{x}))$ where $\boldsymbol{x} = 0$ is an equilibrium point, then the origin is globally asymptotically stable.

$$\dot{V}(\boldsymbol{x}, \alpha) = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \alpha(\boldsymbol{x})) \leq -W(\boldsymbol{x}) \tag{3.3.2}$$

where $W(x)$ is a positive definite function; see Krstic et al. [9, §2.1.2].

Definition 3.3.2 (Control Lyapunov Function (CLF))**.** Krstic et al. [9, Def. 2.4],[15] A positive definite, radially unbounded, smooth scalar function $V = V(\boldsymbol{x})$ is called a CLF for $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, u)$ if there exists a u such that:

$$\inf_{u \in \mathbb{R}} \left\{ \dot{V}(\boldsymbol{x}, u) = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, u) < 0 \right\} \quad , \quad \forall\, \boldsymbol{x} \neq 0 \tag{3.3.3}$$

where **inf** denotes infimum, the greatest lower bound. For systems affine in control, ie.

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x}) u \quad , \quad \boldsymbol{f}(0) = 0\,, \tag{3.3.4}$$

the CLF inequality in Equation 3.3.2 becomes

$$\dot{V}(\boldsymbol{x}, \alpha) = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) + \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{g}(\boldsymbol{x}) \alpha(\boldsymbol{x}) \leq -W(\boldsymbol{x}) \tag{3.3.5}$$

For this system the only way to satisfy Definition 3.3.2 is if:

$$\frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{g}(\boldsymbol{x}) = 0 \quad \Rightarrow \quad \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) < 0 \quad , \quad \forall\, \boldsymbol{x} \neq 0$$

The problem with the CLF concept is that for most non-linear 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. [9]"

3.3.4 Backstepping

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

Table 3.5: Backstepping Key Terms

Useful Nonlinearities	Less control effort and less precise model information required
Flexible	Less restrictive, more freedom in choosing stabilizing function $\alpha(x)$ and Lyapunov function $V(x)$
Recursive	System is augmentable by a chain of integrators, creating intermediate states called virtual control laws ξ_k that assist in control
Constructive	Systematic design procedure

Consider moving the applicable system types after command filtering

Backstepping offers a systematic design procedure via the addition of integrators, and can handle various classes of non-linear systems; two typical classes will be introduced based on presentation by Krstic et al. [9, §2.3.1] and Härkegå rd [15, §3.3.2].

Definition 3.3.3 (Strict Feedback System)**.**

The first is a *strict feedback* system and is of the form:

$$\begin{aligned}
\dot{x} &= f_0(x)g_0(x)\xi_1 \\
\dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2 \\
\dot{\xi}_2 &= f_2(x,\xi_1,\xi_2) + g_2(x,\xi_1,\xi_2)\xi_3 \\
&\vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x,\xi_1,\cdots,\xi_{k-1}) + g_{k-1}(x,\xi_1,\cdots,\xi_{k-1})\xi_k \\
\dot{\xi}_k &= f_k(x,\xi_1,\cdots,\xi_k) + g_k(x,\xi_1,\cdots,\xi_k)u
\end{aligned} \tag{3.3.6}$$

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 3.6. The

ξ-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 3.3.4 (Pure Feedback System)**.**

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

$$\begin{aligned}
\dot{x} &= f_0(x, \xi_1) \\
\dot{\xi}_1 &= f_1(x, \xi_1, \xi_2) \\
\dot{\xi}_2 &= f_2(x, \xi_1, \xi_2, \xi_3) \\
&\vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x, \xi_1, \cdots, \xi_k) \\
\dot{\xi}_k &= f_k(x, \xi_1, \cdots, \xi_k, u)
\end{aligned} \tag{3.3.7}$$

where $\xi_i \in \mathbb{R}^n$ and the x-subsystem again satisfies upcoming Assumption 3.6. 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 3.3.7 lacks the *affine*[17] appearance of variables ξ_k and u.

Krstic et al. [9, §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** ξ; to be demonstrated in the derivation of the flight-path controller in chapter 4.

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.

[17] "An affine function is just a linear function plus a translation." http://cfsv.synechism.org/c1/sec15.pdf

3.3.4.1 Second Order Systems

Typically in feedback control architectures the objective is to create a control law that cancels known dynamics and/or 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 *non-linear* 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 non-linear terms in the dynamics, if recognized, may be retained hence less precise modeling information and less control effort is necessary. Also, the inclusion of non-linearities improves transient performance.

To begin the backstepping[18] procedure, we'll select a general first order system ($n = 1$),

$$\dot{x} = f(x) + g(x)u \,, \tag{3.3.8}$$

and augment it with an integrator, thereby transforming it to a second order system:

$$\begin{aligned} \dot{x} &= f(x) + g(x)\xi && \text{(3.3.9a)} \\ \dot{\xi} &= u && \text{(3.3.9b)} \end{aligned}$$

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 ξ to perform either *1*) regulation by stabilizing the origin ($x = 0\,, \xi = 0$), ie. $x(t) \to 0$ as $t \to \infty$, or *2*) tracking by causing the x-portion of the state to track a reference signal, say y_d, ie. $x(t) \to y_d$ as $t \to \infty$; with respect to System 3.3.8, "this is equivalent to treating ξ as a *virtual control* input for the $\dot{x}$ equation," hence we call ξ a virtual control law. The same idea can be found

[18] Alternatively referred to as *integrator* backstepping.

in cascaded control design, as System 3.3.9 may be viewed as the cascade connection of $\dot{x}$ and $\dot{\xi}$ subsystems. This is shown in Figure 3.16 *a)*, where again the first equation treats ξ as a "control input," the second equation is the integrator, and the dashed box is the original system in Equation 3.3.8.

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 [4, §5.3.1]. The corresponding assumptions are:

Assumption 3.5 (*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 3.3.9a can be stabilized by a state feedback control $\alpha(x)$,

$$\xi = \alpha(x) \quad \Rightarrow \quad \dot{x} = f(x) + g(x)\alpha(x)$$

and further that a Lyapunov function is known that renders the equilibrium point, or origin in this case, asymptotically stable, $\alpha(0) = 0$:

$$\dot{V}(x) = \frac{\partial V}{\partial x}\dot{x} = \frac{\partial V}{\partial x}\left[f(x) + g(x)\alpha(x)\right] \leq -W(x) \quad , \quad \forall\, x \in \mathcal{D} \tag{3.3.10}$$

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 3.3.9, we obtain the equivalent system below, as shown in Figure 3.16 *b)*:

$$\dot{x} = [f(x) + g(x)\alpha(x)] + g(x)\left[\xi - \alpha(x)\right] \tag{3.3.11a}$$

$$\dot{\xi} = u \tag{3.3.11b}$$

If we let z be the **error state**, or deviation of actual from desired virtual control with

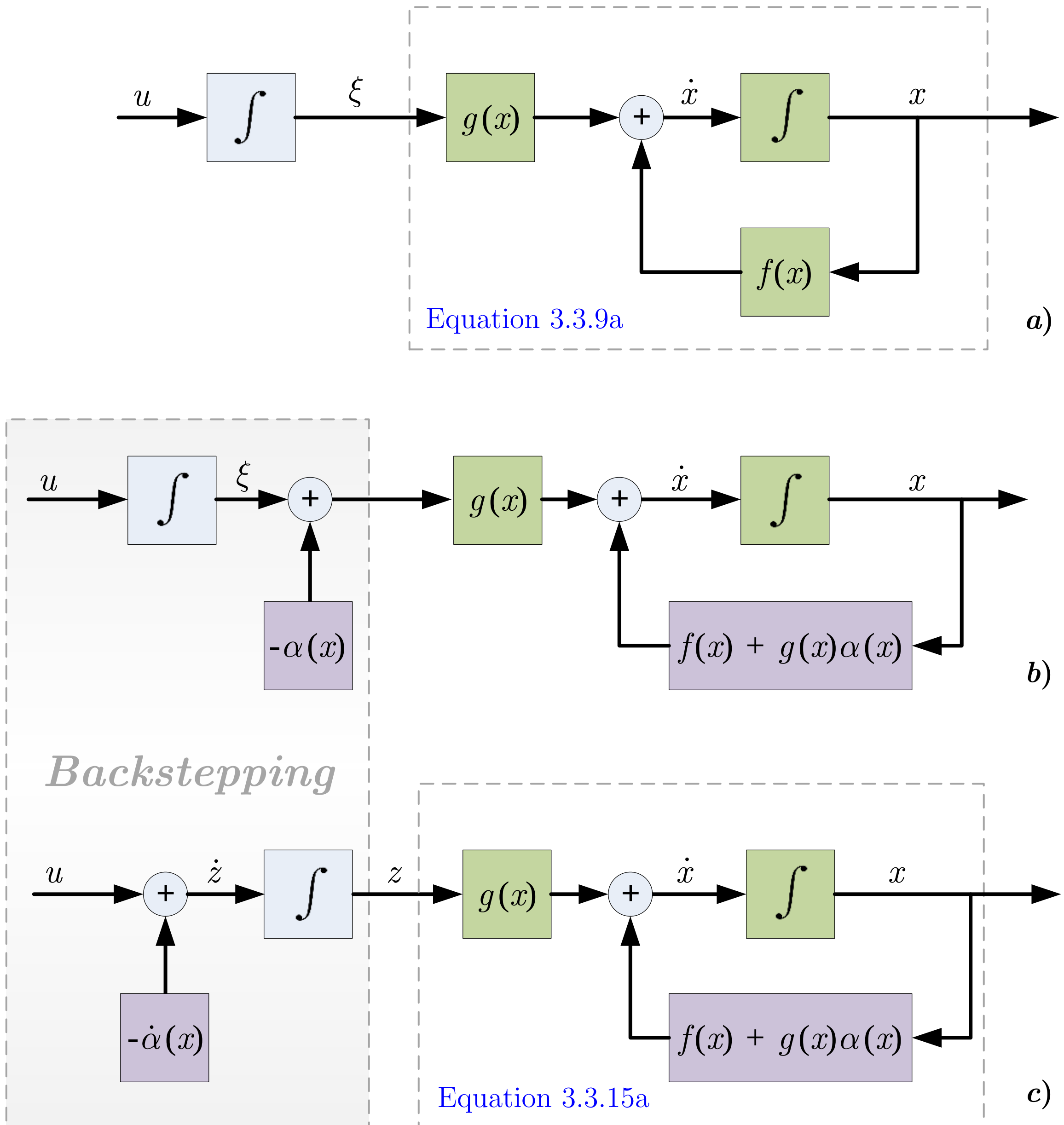


Figure 3.16: Backstepping Block Diagram Representation: *a*) Integral augmented cascade system *b*) introducing $\pm\alpha(x)$ *c*) "backstepping" $-\alpha(x)$ through the integrator

the latter achieved via a stabilizing function $\alpha(x)$,

$$z = \xi - \xi_{des} = \xi - \alpha(x) \tag{3.3.12}$$

then a transformed system, 3.3.13, is obtained by a) substitution of Equation 3.3.12 into Subsystem 3.3.11a and b) the derivative[19] of Equation 3.3.12 :

$$\dot{x} = [f(x) + g(x)\alpha(x)] + g(x)z \tag{3.3.13a}$$

$$\dot{z} = u - \dot{\alpha} \tag{3.3.13b}$$

as depicted in Figure 3.16 c). The term backstepping is derived from the procedure leading to System 3.3.13, whereby taking the derivative of z led to "backstepping" of $-\alpha(x)$ through the integrator; this defining feature is highlighted in Figures 3.16 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 α are known, the derivative $\dot{\alpha}$ may be computed offline, ie. analytically, without the need of a differentiator by using the chain rule:

$$\dot{\alpha} = \frac{\partial \alpha}{\partial x}\dot{x} = \frac{\partial \alpha}{\partial x}\left[f(x) + g(x)\xi\right] \tag{3.3.14}$$

If we introduce a *modified control input* by defining $v = u - \dot{\alpha}$ then the System 3.3.13 may be reduced to

$$\dot{x} = [f(x) + g(x)\alpha(x)] + g(x)z \tag{3.3.15a}$$

$$\dot{z} = v \tag{3.3.15b}$$

This is nearly the same as System 3.3.9, except that the inclusion of a stabilizing function α 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

$$V(x, z) = V(x) + \frac{1}{2}z^2 \tag{3.3.16}$$

[19] z is a function of x only, hence we may use the sum rule for differentiation: $\dot{z} = d\xi/dx - d\alpha/dx = u - \dot{\alpha}$

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

$$\begin{aligned}\dot{V} &= \frac{\partial V}{\partial x}\frac{dx}{dt} + \frac{\partial V}{\partial z}\frac{dz}{dt} \\ &= \frac{\partial V}{\partial x}\left\{[f(x) + g(x)\alpha(x)] + g(x)z\right\} + \frac{\partial V}{\partial z}v \\ &\leq -W(x) + \frac{\partial V}{\partial x}g(x)z + zv\end{aligned} \tag{3.3.17}$$

where Equation 3.3.10 was utilized to make the $-W(x)$ substitution, then choosing a modified control input v

$$v = -\frac{\partial V}{\partial x}g(x) - kz \quad , \quad k > 0 \tag{3.3.18}$$

and substituting this back into Equation 3.3.17 implies

$$\dot{V} \leq -W(x) - kz^2 \tag{3.3.19}$$

$\dot{V}$ 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 3.3.12 the origin $(x = 0\,, \alpha = 0)$ is also asymptotically stable. The final step is substituting $\dot{z} = v$ and $\dot{\alpha}$ into Subsystem 3.3.13b and rearranging in order to obtain the state feedback control law, u, for the second order system we started from:

$$u = \dot{\alpha} + \dot{z} = \frac{\partial \alpha}{\partial x}\left[f(x) + g(x)\xi\right] - \frac{\partial V}{\partial x}g(x) - kz \tag{3.3.20}$$

Further, if all items in Assumption 3.5 hold and $V(x)$ is radially unbounded then the equilibrium point is globally asymptotically stable. This process is summarized formally by Assumption 3.6 and Lemma 3.3.1:

Assumption 3.6 (Backstepping Stabilizing Function). Krstic et al. [9, Asmp. 2.7]
For the general first order system

$$\dot{x} = f(x) + g(x)u \quad , \quad f(0) = 0 \tag{3.3.21}$$

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 $\alpha(x)$

$$u = \alpha(x) \quad , \quad \alpha(0) = 0$$

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

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

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

With Theorem 3.2.4 LaSalle-Yoshizawa the control law in Assumption 3.6 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 3.2.3 LaSalle is satisfied with $W(x)$ is positive definite, the control renders $x = 0$ the GAS equilibrium of Equation 3.3.21.

Lemma 3.3.1 (Backstepping). *Krstic et al. [9, Lem. 2.8]*

Let Equation 3.3.21 be augmented by an integrator:

$$\dot{x} = f(x) + g(x)\xi \tag{3.3.22a}$$

$$\dot{\xi} = u\,, \tag{3.3.22b}$$

and suppose that Subsystem 3.3.22a satisfies Assumption 3.6 with $\xi = \mathbb{R}$ as its control.

*i) If $W(x)$ is **positive definite**, then*

$$V(x,\xi) = V(x) + \frac{1}{2}[\xi - \alpha(x)]^2 \tag{3.3.23}$$

*is a CLF for System 3.3.22, that is, there exists a feedback control $u = \alpha(x,\xi)$ which renders **the origin** $(\mathbf{x} = \mathbf{0}, \xi = \mathbf{0})$ the GAS equilibrium point. One such control is*

$$u = \frac{\partial \alpha}{\partial x}\left[f(x) + g(x)\xi\right] - \frac{\partial V}{\partial x} g(x) - k[\xi - \alpha(x)] \quad , \quad k > 0 \tag{3.3.24}$$

*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 $\mathcal{E} = \left\{[x,\xi]^T \in \mathbb{R}^{n+1} \quad | \quad W(x) = 0\,,\, \xi = \alpha(x)\right\}$*

"The **main result of backstepping** is not the specific form of the control law, Equation 3.3.24, 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. [9].

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. [3]

Definition 3.3.5 (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 3.3.1, 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 3.3.2, backstepping will be applied to the same system to demonstrate the technique and show the flexibility in this procedure.

Example 3.3.1 (Useful Nonlinearities)**.** Härkegå rd [15, Ex. 3.1]

Consider the system

$$\dot{x} = -x^3 + x + u \tag{3.3.25}$$

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 3.17.

Note that this is a 1D phase portrait, unlike the 2D phase portraits illustrated in subsection 3.2.1; 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

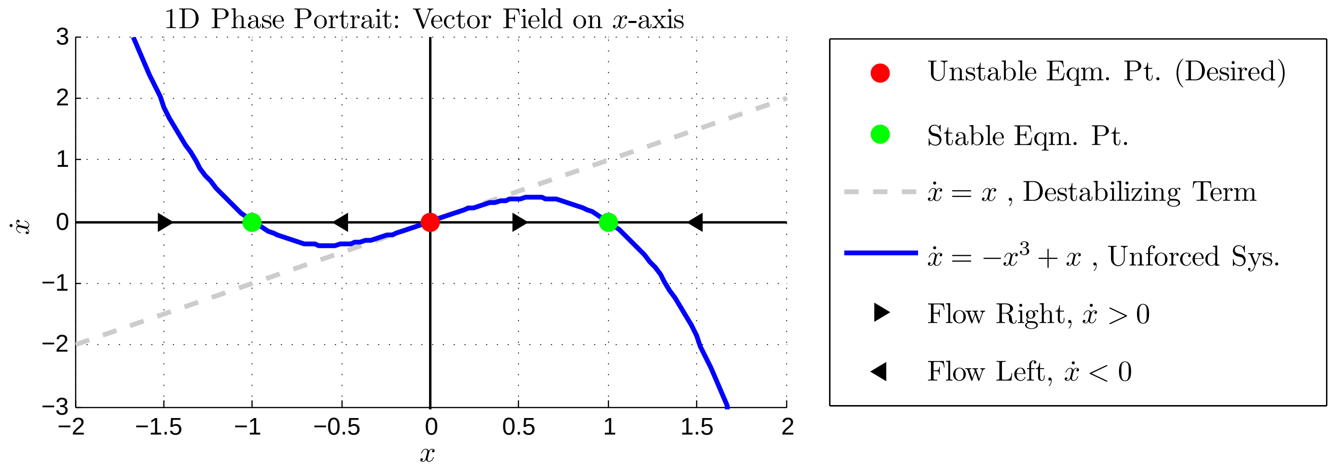


Figure 3.17: Composite Phase Portrait for System 3.3.25 with $u = 0$

flowing along the x-axis with a velocity that varies according to System 3.3.25; Strogatz [17]. 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 3.2.1 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 3.17 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 ± 1. Remember that these observations are for the unforced form of System 3.3.25. Developing a stabilizing control law, ie. 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

$$u = -x \tag{3.3.26}$$

and a CLF given by

$$V(x) = \frac{1}{2}x^2 \tag{3.3.27}$$

which yields

$$\dot{V}(x) = \frac{\partial V}{\partial x}\dot{x} = x(-x^3 + x + u) = -x^4 \tag{3.3.28}$$

proving the origin is GAS according to Cor. 3.2.1 Barbashin-Krasovskii."

Alternatively, applying **feedback linearization** requires that the control law counteracts all non-linear dynamics:

$$u = x^3 - kx \quad , \quad k > 1 \tag{3.3.29}$$

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 3.3.2 (Flexibility in Backstepping). Härkegå rd [15, Ex. 3.1]

Consider System 3.3.25 in the previous example augmented by an integrator where $x = x_1$ and $\xi = x_2$.

$$\dot{x}_1 = -x_1^3 + x_1 + x_2 \tag{3.3.30a}$$

$$\dot{x}_2 = u \tag{3.3.30b}$$

The first task is to stabilize the equilibrium point of Subsystem 3.3.30a by treating x_2 as a virtual control input. Since we know from phase portrait observations in Figure 3.17 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 3.3.26 did:

$$x_{2des} \equiv \alpha(x_1) = -x_1 \tag{3.3.31}$$

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

$$z = x_2 - \alpha(x_1) = x_2 + x_1 \tag{3.3.32}$$

Now we can rewrite the system in xz coordinates with substitution of 3.3.32 into 3.3.30a and the analytic derivative of z, ie. $\dot{z} = \dot{x}_2 + \dot{x}_1$ with substitution of 3.3.30b:

$$\dot{x}_1 = -x_1^3 + z \tag{3.3.33a}$$

$$\dot{z} = u - x_1^3 + z \tag{3.3.33b}$$

Following Backstepping Lemma 3.3.1, a CLF is now constructed for System 3.3.33. Our initial choice will reuse the positive definite quadratic postulated in Equation 3.3.27 for the first term of CLF 3.3.23 in the lemma.

$$V(x_1) = \frac{1}{2}x_1^2 \tag{3.3.34}$$

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

$$V(x_1, x_2) = V(x_1) + \frac{1}{2}(x_2 - \alpha(x_1))^2 = \frac{1}{2}x_1^2 + \frac{1}{2}z^2 \tag{3.3.35}$$

Differentiating with respect to time will allow us to evaluate which choices for u satisfy Lyapunov stability theorems

$$\dot{V} = x_1(-x_1^3 + z) + z(u - x_1^3 + z) = -x_1^4 + z(x_1 + u - x_1^3 + z) \tag{3.3.36}$$

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"

$$u = x_1^3 - x_1 - 3z \tag{3.3.37}$$

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 3.3.23 again

$$V(x_1, x_2) = V(x_1) + \frac{1}{2}(z)^2 \tag{3.3.38}$$

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

$$\begin{aligned}\dot{V} &= \frac{\partial V}{\partial x_1}\frac{dx_1}{dt} + \frac{\partial V}{\partial x_2}\frac{dx_2}{dt} \\ &= \dot{V}(-x_1^3 + z) + z(u - x_1^3 + z) \\ &= -\dot{V}x_1^3 + z(\dot{V} + u - x_1^3 + z)\end{aligned} \tag{3.3.39}$$

Now we are able to choose a $V(x_1)$ such that $\dot{V}$ in 3.3.39 cancels the indefinite mixed terms. This is achieved by selecting

$$\dot{V}(x_1) = x_1^3 \qquad \ni \qquad V(x_1) = \frac{1}{4}x_1^4 \tag{3.3.40}$$

With this CLF, 3.3.39 is now

$$\dot{V} = -x_1^6 + z(u + z) \tag{3.3.41}$$

In contrast to Equation 3.3.36 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

$$u = -3z = -3x_1 - 3x_2 \tag{3.3.42}$$

which renders $\dot{V} = -x_1^6 - 2z_2^2$ negative definite, thereby making the origin GAS. This technique for choosing $V(x_1)$ was published in Krstic et al. [18].

3.3.4.2 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 which applicable to any systems over an order of two.

Example 3.3.3 (Recusrive Nature of Backstepping). Khalil [1, Ex. 13.7]

Consider the third-order system,

$$\dot{x}_1 = x_1^2 - x_1^3 + x_2 \tag{3.3.43a}$$

$$\dot{x}_2 = x_3 \tag{3.3.43b}$$

$$\dot{x}_3 = u \tag{3.3.43c}$$

which is nearly identical to System 3.3.30 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 3.3.2, we begin by developing a stabilizing feedback control $x_2 \equiv \alpha(x_1)$ for Subsystems 3.3.43a and 3.3.43b. We may consider x_3 as the control input "u" that stabilizes the origin $x_1 = 0$; necessary in order to draw from Assumption 3.6 and Lemma 3.3.1.

$$\dot{x}_1 = x_1^2 - x_1^3 + x_2 \tag{3.3.44a}$$

$$\dot{x}_2 = x_3 \tag{3.3.44b}$$

Choosing a stabilizing function, again that recognizes $-x_1^3$ as stabilizing,

$$\alpha(x_1) = -x_1^2 - x_1 \tag{3.3.45}$$

leads to the reformulated system

$$\dot{x}_1 = -x_1 - x_1^3 \tag{3.3.46}$$

Note that the error state is defined as

$$z = x_2 - \alpha(x_1) = x_2 - x_1^2 - x_1 \tag{3.3.47}$$

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

$$\dot{V} = \frac{\partial V}{\partial x_1}\dot{x}_1 = -x_1^2 - x_1^4 \leq -x_1^2 \quad , \quad \forall\, x_1 \in \mathbb{R} \tag{3.3.48}$$

then apply u and V from Lemma 3.3.1 to build the control x_3, with $f = x_1^2 - x_1^3$, $g = 1$, and $k = 1$,

$$\begin{aligned} x_3 &= \frac{\partial \alpha}{\partial x_1}\left(x_1^2 - x_1^3 + x_2\right) - \frac{\partial V}{\partial x_1} - [x_2 - \alpha(x_1)] \\ &= -(2x_1+1)(x_1^2 - x_1^3 + x_2) - x_1 - (x_2 + x_1^2 + x_1) \end{aligned} \tag{3.3.49}$$

that stabilizes the origin $x = 0$ globally to form the composite Lyapunov function

$$V(x_1, x_2) = \frac{1}{2}x_1^2 + \frac{1}{2}(x_2 - x_1^2 - x_1)^2 \tag{3.3.50}$$

Step 2: We now consider the next subsystem, 3.3.43c, by viewing the third order system as a special case of System 3.3.11

$$\begin{aligned} \dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= u \end{aligned}$$

with the system in Step 1 condensed into

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad , \quad f = \begin{bmatrix} x_1^2 - x_1^3 + x_2 \\ 0 \end{bmatrix} \quad , \quad g = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad , \quad \xi = x_3 \tag{3.3.51}$$

We know from Step 1 that the following feedback control, $x_3 \equiv \alpha(x_1, x_2)$, and CLF stabilizes System 3.3.44

$$\begin{aligned} \alpha(x_1, x_2) &= -(2x_1+1)(x_1^2 - x_1^3 + x_2) - x_1 - (x_2 + x_1^2 + x_1) \\ V(x_1, x_2) &= \frac{1}{2}x_1^2 + \frac{1}{2}(x_2 - x_1^2 - x_1)^2 \end{aligned}$$

which we define as the virtual control and portion of the CLF for 3.3.51. Next, Backstepping Lemma 3.3.1 is applied in order to obtain the globally stabilizing feedback control

$$u = \frac{\partial \alpha}{\partial x_1}(x_1^2 - x_1^3 + x_2) + \frac{\partial \alpha}{\partial x_2}(x_3) - \frac{\partial V}{\partial x_2} - [x_3 - \alpha(x_1, x_2)] \tag{3.3.52}$$

and corresponding Lyapunov function

$$V(x) = V(x_1, x_2) + \frac{1}{2}(x_3 - \alpha(x_1, x_2))^2 \tag{3.3.53}$$

This example is now **summarized for a more general form** of System 3.3.9

$$\dot{x} = f(x) + g(x)\xi \tag{3.3.54a}$$

$$\dot{\xi} = f_a(x,\xi) + g_a(x,\xi)u \tag{3.3.54b}$$

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

$$u = \frac{1}{g_a(x,\xi)}\left[u_a - f_a(x,\xi)\right] \tag{3.3.55}$$

which reduces Subsystem 3.3.54b to the "integrator form" $\dot{\xi} = u_a$. If Backstepping Lemma 3.3.1 conditions are satisfied by a stabilizing function $\alpha(x)$ and Lyapunov function $V(x)$ then the Lemma combined with Equation 3.3.55 yields the stabilizing state feedback control

$$u = \alpha_a(x,\xi) = \frac{1}{g_a(x,\xi)}\left\{\frac{\partial\alpha}{\partial x}\left[f(x) + g(x)\xi\right] - \frac{\partial V}{\partial x}g(x) - k[\xi - \alpha(x)] - f_a(x,\xi)\right\} \tag{3.3.56}$$

for some $k > 0$ and the Lyapunov function

$$V_a(x,\xi) = V(x) + \frac{1}{2}\left[\xi - \alpha(x)\right]^2 \tag{3.3.57}$$

for System 3.3.54. By recursive application of backstepping, we can stabilize strict feedback systems as shown in Definition 3.3.3:

$$\begin{aligned}
\dot{x} &= f_0(x) + g_0(x)\xi_1 \\
\dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2 \\
\dot{\xi}_2 &= f_2(x,\xi_1,\xi_2) + g_2(x,\xi_1,\xi_2)\xi_3 \\
&\vdots \\
\dot{\xi}_{k-1} &= f_{k-1}(x,\xi_1,\cdots,\xi_{k-1}) + g_{k-1}(x,\xi_1,\cdots,\xi_{k-1})\xi_k \\
\dot{\xi}_k &= f_k(x,\xi_1,\cdots,\xi_k) + g_k(x,\xi_1,\cdots,\xi_k)u
\end{aligned} \tag{3.3.6}$$

Step 1: The recursive procedure begins with the first subsystem of 3.3.6

$$\dot{x} = f_0(x) + g_0(x)\xi_1 \tag{3.3.58}$$

where ξ_1 is considered the control input. Just as in Section 3.3.4.1 and Assumption 3.6 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,

$$\dot{V}_0 = \frac{\partial V_0}{\partial x}\left[f_0(x) + g_0(x)\alpha_0(x)\right] \leq -W(x) \tag{3.3.59}$$

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 3.3.6 combined with the subsystem in Step 1

$$\begin{aligned} \dot{x} &= f_0(x) + g_0(x)\xi_1 \\ \dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2 \end{aligned} \tag{3.3.60}$$

which is a special case of System 3.3.54,

$$\begin{aligned} \dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= f_a(x,\xi) + g_a(x,\xi)u \end{aligned} \tag{3.3.54}$$

with

$$x = x \quad , \quad \xi = \xi_1 \quad , \quad u = \xi_2 \quad , \quad f = f_0 \quad , \quad g = g_0 \quad , \quad f_a = f_1 \quad , \quad g_a = g_1$$

We may reuse 3.3.56 and 3.3.57 to obtain the stabilizing state feedback control and Lyapunov function for System 3.3.60 as

$$\alpha_1(x,\xi_1) = \frac{1}{g_1}\left[\frac{\partial \alpha_0}{\partial x}(f_0 + g_0\xi_1) - \frac{\partial V_0}{\partial x}g_0 - k_1(\xi_1 - \alpha) - f_1\right] \quad , \quad k_1 > 0 \tag{3.3.61}$$

$$V_1(x,\xi_1) = V_0(x) + \frac{1}{2}\left[\xi_1 - \alpha(x)\right]^2 \tag{3.3.62}$$

Step 3: Now, consider the next subsystem of System 3.3.6 combined with the subsystems in Step 2

$$\begin{aligned} \dot{x} &= f_0(x) + g_0(x)\xi_1 \\ \dot{\xi}_1 &= f_1(x,\xi_1) + g_1(x,\xi_1)\xi_2 \\ \dot{\xi}_2 &= f_2(x,\xi_1,\xi_2) + g_2(x,\xi_1,\xi_2)\xi_3 \end{aligned} \tag{3.3.63}$$

which is again a special case of System 3.3.54,

$$\begin{aligned}\dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= f_a(x,\xi) + g_a(x,\xi)u\end{aligned} \tag{3.3.54}$$

with

$$x = \begin{bmatrix} x \\ \xi_1 \end{bmatrix}, \; \xi = \xi_2 \;, \; u = \xi_3 \;, \; f = \begin{bmatrix} f_0 + g_0 z_1 \\ f_1 \end{bmatrix}, \; g = \begin{bmatrix} 0 \\ g_1 \end{bmatrix}, \; f_a = f_2 \;, \; g_a = g_2$$

We may again reuse 3.3.56 and 3.3.57 to obtain the stabilizing state feedback control and Lyapunov function for System 3.3.60 as

$$\alpha_2(x,\xi_1,\xi_2) = \frac{1}{g_2}\left[\frac{\partial\alpha_1}{\partial x}(f_0 + g_0\xi_1) + \frac{\partial\alpha_1}{\partial\xi_1}(f_1 + g_1\xi_2) - \frac{\partial V_1}{\partial\xi_1}g_1 - k_2(\xi_2 - \alpha_1) - f_2\right] \tag{3.3.64}$$

for some $k_2 > 0$ and

$$V_2(x,\xi_1,\xi_2) = V_1(x,\alpha_1) + \frac{1}{2}\left[\xi_2 - \alpha_2(x,\xi_1)\right]^2 \tag{3.3.65}$$

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 3.3.6.

3.3.4.3 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 3.3.5; for now, we'll assume that we have command filtered signals and derivatives available.[20]

[20] This section follows work in Farrell and Polycarpou [4][§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.

We will introduce the concept with a simple(r) second order system [21]

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \tag{3.3.66a}$$

$$\dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)u \tag{3.3.66b}$$

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 $\mathcal{D}$ with f_i and g_i for $i = 1, 2$ known and locally Lipschitz in x. Furthermore, $g_i \neq 0$ for all $x \in \mathcal{D}$ 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**

$$\tilde{x}_1 = x_1 - x_{1c} \tag{3.3.67a}$$

$$\tilde{x}_2 = x_2 - x_{2c} \tag{3.3.67b}$$

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 α_1 to cancel dynamics in the first subsystem of 3.3.66, ie. $x_2 \equiv \alpha_1$,

$$\alpha_1(x_1, \tilde{x}_1, \dot{x}_{1c}) = \frac{1}{g_1}[-f_1 - k_1\tilde{x}_1 + \dot{x}_{1c}] \tag{3.3.68}$$

with $k_1 > 0$, assuming it's a smooth feedback control. Note that this choice of stabilizing function reduces Subsystem 3.3.66a to a tracking problem

$$\dot{x}_1 - \dot{x}_{1c} = -k_1(x_1 - x_{1c}) \tag{3.3.69}$$

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

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

$$\dot{V} = \frac{\partial V_1}{\partial \tilde{x}_1}[f_1 + g_1\alpha_1 - \dot{x}_{1c}] = -W(\tilde{x}_1) \tag{3.3.70}$$

[21] 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.

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**[22] in System 3.3.66.

Step 1: Define the unfiltered state command and the filter

$$x_{2c}^o = \alpha_1 - \zeta_2 \tag{3.3.71}$$

$$\dot{\zeta}_1 = -k_1\zeta_1 + g_1(x_{2c} - x_{2c}^o) \tag{3.3.72}$$

where ζ_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 3.3.5.

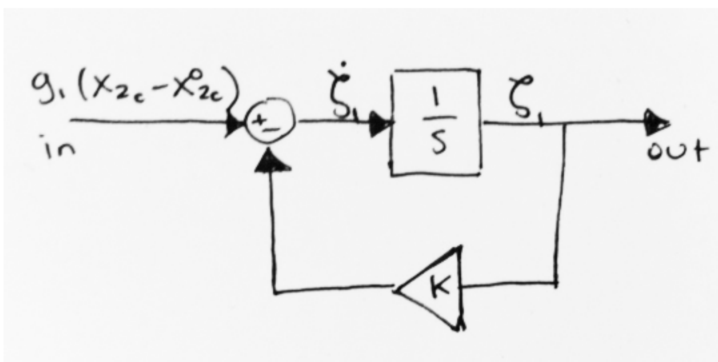


Figure 3.18: Equation 3.3.72 Low Pass Filter

By design of Equation 3.3.72, 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 ζ_1 will be bounded; a stable linear filter with a bounded input and output.

Step 2: Define compensated tracking errors as

$$\bar{x}_i = \tilde{x}_i - \zeta_i \quad , \quad \text{for } i = 1, 2 \tag{3.3.73}$$

22 Note that Backstepping sections before this one performed regulation control, ie. the desired equilibrium point was the system origin.

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

$$u_c^o = \frac{1}{g_2}[-k_2\tilde{x}_2 + \dot{x}_{2c} - f_2 - \bar{x}_1^T g_1] \quad , \quad \text{with } k_2 > 0 \tag{3.3.74}$$

$$\dot{\zeta}_2 = -k_2\zeta_2 + g_2(u_c - u_c^o) \tag{3.3.75}$$

"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 3.3.75 is bounded and small, therefore if g_2 is bounded then the output ζ_2 is bounded; another stable linear filter with a bounded input and output. Note, if $u_c^o = u_c = u$ then $\zeta_2 = 0$.

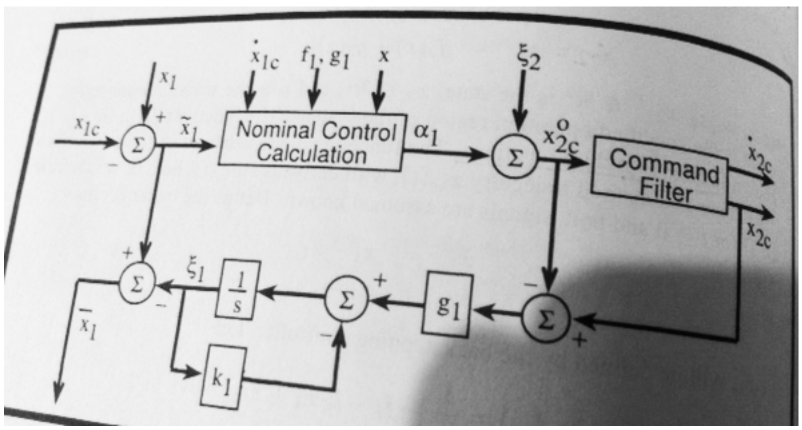


Figure 3.19: Command Filter

Figure 3.19 displays a block diagram representation of Steps 1-3 collectively. Farrell and Polycarpou [4] explicitly note that u_c^o is computed using $\dot{x}_{2c}$ and not $\dot{x}_{2c}^o$, where $\dot{x}_{2c}$ is the output of the command filter for Equation 3.3.71.

Now, to analyze the stability of the control law we need to derive the tracking error dynamics. Begin by taking the derivative of Equation 3.3.67a, then substituting *a*) Equation 3.3.66a and *b*) $\dot{x}_{1c}$ by solving for this term with nested substitution of Equation 3.3.68 into Equation 3.3.71 and re-arranging. Lastly, include $\pm g_1 x_{2c}$ at the point shown in the derivation

$$
\begin{aligned}
\dot{\tilde{x}}_1 &= \dot{x}_1 - \dot{x}_{1c} \\
&= [f_1 + g_1 x_2] - [g_1(x_{2c}^o + \zeta_2) + f_1 + k_1\tilde{x}_1] \\
&= \cancel{f_1} + g_1 x_2 - g_1 x_{2c}^o - g_1\zeta_2 - \cancel{f_1} - k_1\tilde{x}_1 \pm g_1 x_{2c} \\
&= -k_1\tilde{x}_1 + g_1(x_2 - x_{2c}) - g_1\zeta_2 + g_1(x_{2c} - x_{2c}^o) \\
&= -k_1\tilde{x}_1 + g_1\bar{x}_2 + g_1(x_{2c} - x_{2c}^o) \qquad (3.3.76)
\end{aligned}
$$

Taking the derivative of Equation 3.3.67b and performing similar substitutions with Equation 3.3.66a, recall $u = u_c$, and Equation 3.3.74 yields

$$
\begin{aligned}
\dot{\tilde{x}}_2 &= \dot{x}_2 - \dot{x}_{2c} \\
&= [f_2 + g_2 u] - [g_2 u_c^o + k_2\tilde{x}_2 + f_2 + \bar{x}_1^T g_1] \\
&= \cancel{f_2} + g_2 u - g_2 u_c^o - k_2\tilde{x}_2 - \cancel{f_2} - \bar{x}_1^T g_1 \pm g_2 u_c \\
&= -k_2\tilde{x}_2 - g_1^T\bar{x}_1 + g_2(u - u_c) + g_2(u_c - u_c^o) \\
&= -k_2\tilde{x}_2 - g_1^T\bar{x}_1 + g_2(u_c - u_c^o) \qquad (3.3.77)
\end{aligned}
$$

"As defined by 3.3.72 and 3.3.75, the variables ζ_1 and ζ_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"

$$
\begin{aligned}
\dot{\bar{x}}_1 &= -k_1\bar{x}_1 + g_1\bar{x}_2 \qquad (3.3.78) \\
\dot{\bar{x}}_2 &= -k_2\bar{x}_2 - g_1^T\bar{x}_1 \qquad (3.3.79)
\end{aligned}
$$

With the following Lyapunov function

$$
V = \sum_{i=1}^{2} \frac{1}{2}\bar{x}_i^T\bar{x}_i \qquad (3.3.80)
$$

and the corresponding derivative of V along the solution of Equation 3.3.78 and 3.3.79 is

$$\begin{aligned} \dot{V} &= \dot{V}_1 + \dot{V}_2 \\ &= -k_1 \bar{x}_1^T \bar{x}_1 - k_2 \bar{x}_2^2 \leq -\lambda V \end{aligned} \tag{3.3.81}$$

where $\lambda = 2min(k_1, k_2) > 0$. 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.

Consider using Barbalat's Lemma as in Farrell et al. [16]...

Lemma 3.3.2 (Command Filtered Backstepping). *Farrell and Polycarpou [4, Lem. 5.3.2] Let the control law α_1 solve the tracking problem for the system*

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)\alpha_1 \quad \textit{with} \quad x_1 \in \mathbb{R}^{n-1}$$

with Lyapunov function V_1 satisfying Equation 3.3.80. Then the controller of Equation 3.3.71 to 3.3.75 solves the tracking problem (ie., guarantees that $x_1(t)$ converges to $y_d(t)$) for the system described by Equation 3.3.66.

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

3.3.5 Command Filter

This filter was developed in Farrell et al. [19, Appendix A] and provides bounded and continuous command and command-derivative signals. There are two significance 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 its up to the designer whether this filter should be used for actuator commands or not. The non-linear state space representation of this filter is as follows: Do I need q_1 and q_2, couldn't I just use x_c and $\dot{x}_c$?

$$\begin{bmatrix} x_c \\ \dot{x}_c \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{3.3.82}$$

$$\begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \end{bmatrix} = \begin{bmatrix} q_2 \\ 2\zeta\omega_n \left(S_R \left\{ \frac{\omega_n^2}{2\zeta\omega_n} \left[S_M(x_c^o) - q_1 \right] \right\} - q_2 \right) \end{bmatrix} \tag{3.3.83}$$

where S_M and S_R are the magnitude and rate limit functions:

$$S_M = \begin{cases} M & \text{if} \quad x \geq M \\ x & \text{if} \quad \|x\| < M \\ -M & \text{if} \quad x \leq -M \end{cases} \qquad S_R = \begin{cases} R & \text{if} \quad x \geq R \\ x & \text{if} \quad \|x\| < R \\ -R & \text{if} \quad x \leq -R \end{cases} \tag{3.3.84}$$

Equation 3.3.82 may be represented in block diagram form as:

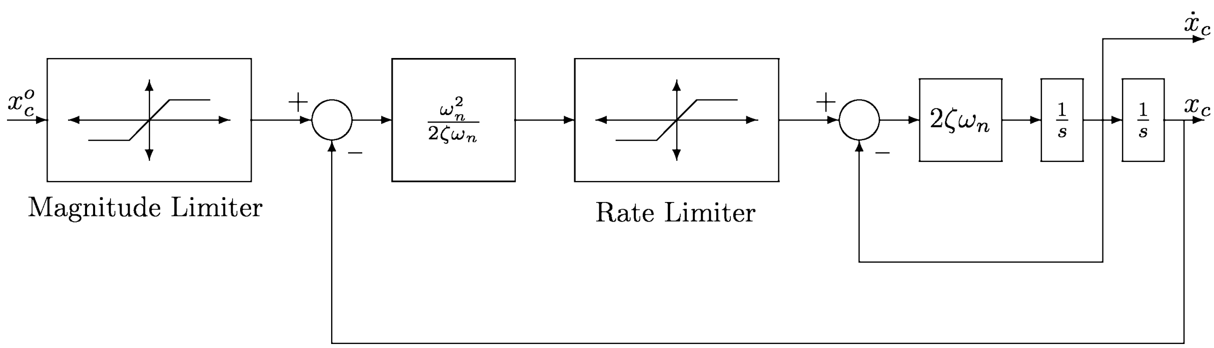


Figure 3.20: Command Filter

As shown in Figure 3.20, this filter generates command (x_c) and command derivative ($\dot{x}_c$) from an unfiltered command (x_c^o) while enforcing magnitude, rate, and bandwidth constraints.

3.3.6 Control Allocation

v = Bu

pinv(B)*v = u

Least squares minimization via pseudo inverse, simplest technique.

Crucial to the robustness of the design, and attempted to implement simplest form of control allocation: not the focus of my thesis, and all the papers I read didn't fully address it, with the exception of Harkegard's dissertation.

http://research.harkegard.se/qcat/

Chapter 4

Derivation of UAV Flight Path Controller

This derivation is based on work by Farrell et al. [16], [19], [4] and Sonneveldt et al. [3]. 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 3.1 and Equation 3.1.1 are employed in that order, ie. the state-vector for the flight path controller is:

$$\boldsymbol{x} = [\chi \; \gamma \; V \; \mu \; \alpha \; \beta \; P \; Q \; R]^T \tag{4.0.1}$$

The inputs to the controller are commanded heading χ_c, climb rate or glide-path angle γ_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 μ_c and angle-of-attack α_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 β_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.

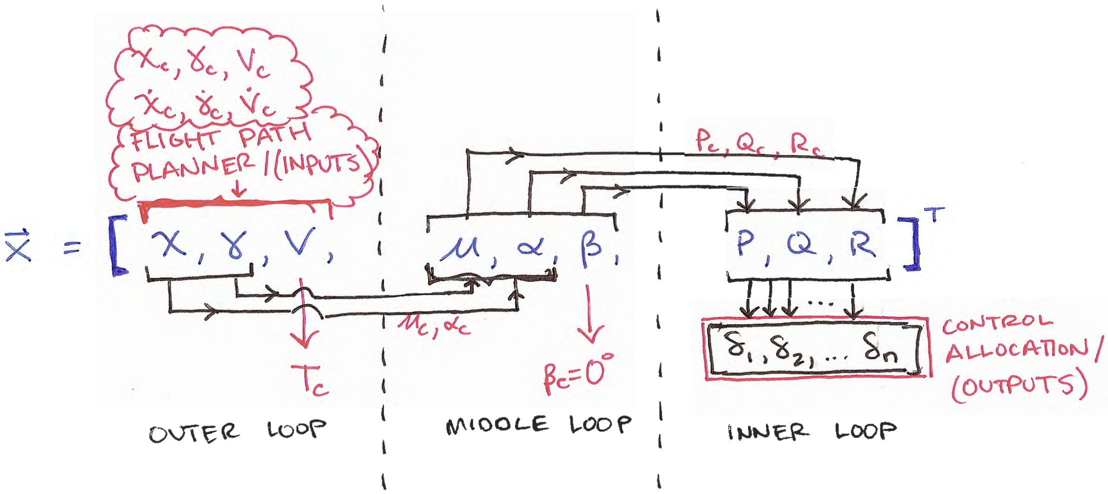


Figure 4.1: Loop Interactions: Flight-Path & Airspeed, Wind, and Body

Design of the flight-path controller herein compliments a publication by Farrell et al. [19], 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 4.1 (*Command Filtered* Backstepping)**.**

- Full State Feedback
- System may be Non-Triangular*
- Lyapunov Function Known*
- System Dynamics Known*
- Smooth Feedback Control
- Actuator Dynamics Neglected

A major contribution of this thesis is the comprehensible, Simulink[2] driven, graphical

[1] However, adaptive parameter estimation is not independent of the stability analysis. [2] Simulink - Model Based Design - http://www.mathworks.com/products/simulink/

block diagram implementation of the control architecture. It offers an intuitive visualization of the procedure and illustrates signal interdependencies that are not self-evident through pages and pages of equations; especially useful for higher order systems. A journal publication by Sonneveldt et al. [3, Figure 1] contains one of the clearest block diagram representation of a backstepping flight controller I've found.

His...

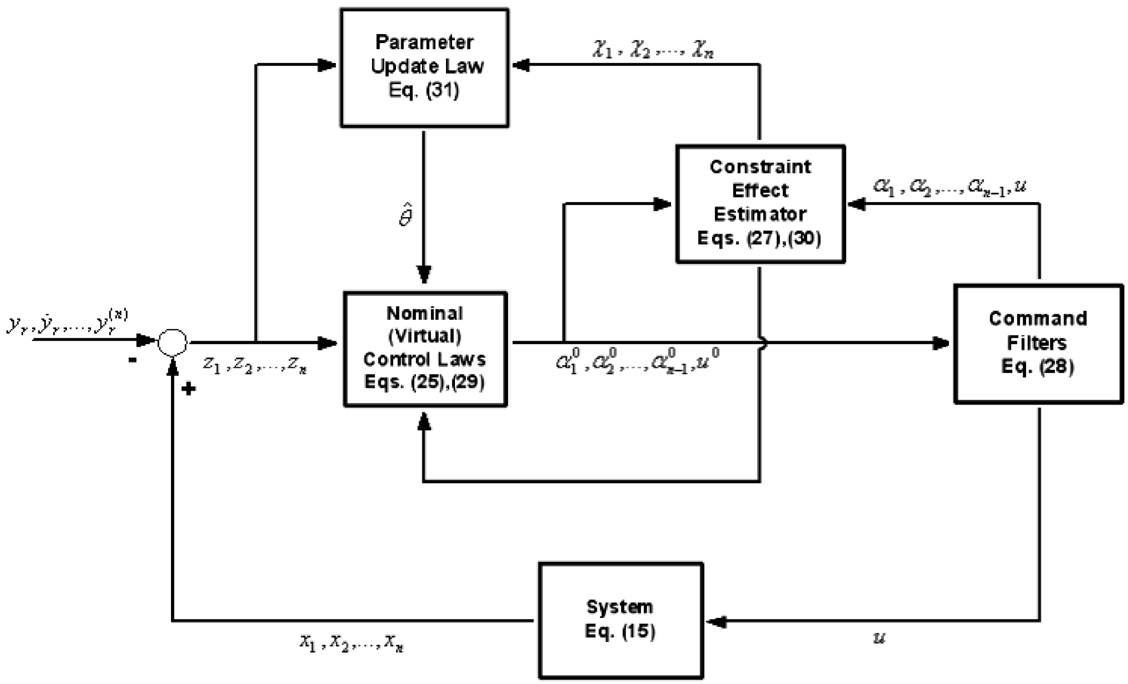


Figure 4.2: Command filtered adaptive backstepping control scheme [3]

Mine...

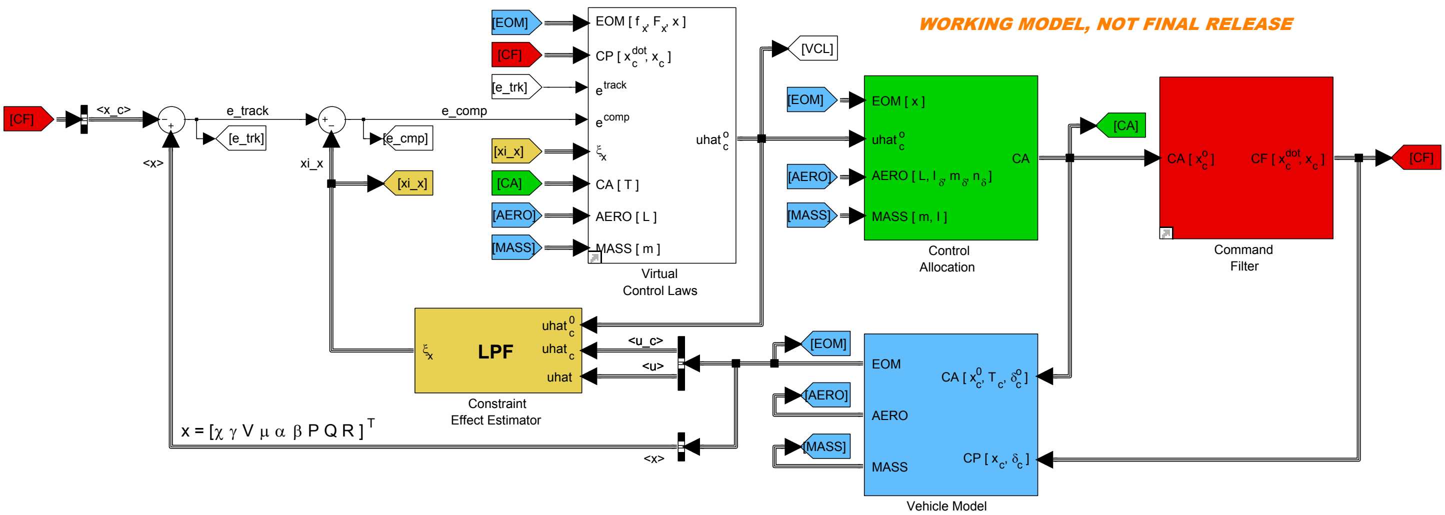


Figure 4.3: High Level Block Diagram

As cited within Farrell et al. [19], "the main advantages of the approach include ... 2) Lyapunov stability results are provable and 3) state and control constraints can be enforced while maintaining Lyapunov stability.[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, *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. [16], formalized in a textbook by Farrell and Polycarpou [4, §5.3.3], and summarized here in § 3.3.4.3, Lemma 3.3.2. **The block vector representation of equations within Farrell et al.**

[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"

[19] **are expanded with respect to each state to make the control law derivation easier to understand. Dynamics will be grouped into sub-functions consisting of known F_x and partially or completely unknown variables f_x, 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.**

4.1 Outer Loop

To begin, flight-path and airspeed **dynamics for the outer loop**, Equations 3.1.42a, 3.1.42b, and 3.1.42c respectively, are grouped into sub-functions of their partially or completely unknown f, known F, and control signals u:

Heading Angle Dynamics

$$\dot{\chi} = f_\chi + F_\chi + u_\chi \tag{4.1.1}$$

$$f_\chi = \frac{\cos\mu}{mV\cos\gamma}\left(D\sin\beta + Y\cos\beta\right) \tag{4.1.2}$$

$$F_\chi = \frac{1}{mV\cos\gamma}\left(-T\cos\alpha\sin\beta\cos\mu\right) \tag{4.1.3}$$

$$u_\chi = \frac{\sin\mu}{mV\cos\gamma}\left(L + T\sin\alpha\right) \tag{4.1.4}$$

Flight-Path Angle Dynamics

$$\dot{\gamma} = f_\gamma + F_\gamma + u_\gamma \tag{4.1.5}$$

$$f_\gamma = \frac{\sin\mu}{mV}\left(-D\sin\beta - Y\cos\beta\right) \tag{4.1.6}$$

$$F_\gamma = \frac{1}{mV}\left(T\cos\alpha\sin\beta\sin\mu - mg\cos\gamma\right) \tag{4.1.7}$$

$$u_\gamma = \frac{\cos\mu}{mV}\left(L + T\sin\alpha\right) \tag{4.1.8}$$

$$\dot{V} = f_V + F_V + u_V \tag{4.1.9}$$

$$f_V = \frac{1}{m}\left(-D\cos\beta + Y\sin\beta\right) \tag{4.1.10}$$

$$F_V = -\,g\sin\gamma \tag{4.1.11}$$

$$u_V = \frac{1}{m}\left(T\cos\alpha\cos\beta\right) \tag{4.1.12}$$

In block vector notation Equations 4.1.1, 4.1.5, and 4.1.9 may be combined, as defined in Farrell et al. [19] and Farrell and Polycarpou [4][8.3.1]:

$$\dot{z}_1 = \bar{A}_1 f_1 + F_1 + G_1(\mu_1, x) \tag{4.1.13}$$

with $z_1 = [\chi\,,\gamma\,,V]^T$, $f_1 = [D\,,Y\,,L]^T$ and

$$\bar{A}_1 = \begin{bmatrix} \sin\beta\cos\mu/\cos\gamma & \cos\beta\cos\mu/\cos\gamma & 0 \\ -\sin\beta\sin\mu & -\cos\beta\sin\mu & 0 \\ -V\cos\beta & V\sin\beta & 0 \end{bmatrix} \tag{4.1.14}$$

$$F_1 = \begin{bmatrix} (-T\cos\alpha\sin\beta\cos\mu)/(mV\cos\gamma) \\ (T\cos\alpha\sin\beta\sin\mu - mg\cos\gamma)/(mV) \\ -g\sin\gamma \end{bmatrix} \tag{4.1.15}$$

$$G_1(\mu_1, x) = \begin{bmatrix} u_\chi \\ u_\gamma \\ u_V \end{bmatrix} = \begin{bmatrix} (g(\alpha,x)\sin\mu)/(mV\cos\gamma) \\ (g(\alpha,x)\cos\mu)/(mV) \\ (T\cos\beta\cos\mu)/(m) \end{bmatrix} \tag{4.1.16}$$

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

4.1.1 Flight Path Angle Control

Wind-axis command signals μ_c and α_c are generated internally in the outer loop of the controller, as formed in this sub-section; β_c is set to zero, as we wish to fly coordinated.

These and derivatives of the wind axis command 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, $\boldsymbol{x}_c$, are fed top-down in the controller, as Figure 4.1 illustrates, while state dynamics, $\boldsymbol{x}$, are fed bottom-up, or back with respect to System 3.1.42. **Tracking errors** are defined as

$$\tilde{\chi} = \chi - \chi_c \quad , \quad \tilde{\gamma} = \gamma - \gamma_c \tag{4.1.17}$$

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

$$\bar{\chi} = \tilde{\chi} - \zeta_\chi \quad , \quad \bar{\gamma} = \tilde{\gamma} - \zeta_\gamma \tag{4.1.18}$$

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; ζ_x terms will be defined shortly.

The immediate goal is to compute unfiltered wind-axis angle commands μ_c^o and α_c^o. 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,

$$u_{\chi_c}^o = -\, f_\chi - F_\chi + \dot{\chi}_c - k_\chi \tilde{\chi} \tag{4.1.19}$$

$$u_{\gamma_c}^o = -\, f_\gamma - F_\gamma + \dot{\gamma}_c - k_\gamma \tilde{\gamma} \tag{4.1.20}$$

where the k terms are gains proportional to the tracking error. Note that these virtual control laws transform their respective systems, 4.1.1 and 4.1.5, into a tracking problem, recall Equation 3.3.69.

Unfiltered control laws are equivalent to their counterparts in Equations 4.1.4 and

[4] $u_{x_c}^o$ is substituted for the initial variable choice $\alpha_{x_c}^o$ in Backstepping Section 3.3.4 to avoid confusion with angle-of-attack

4.1.8, placed in the 'control allocation' simulink subsystem, and defined as

$$u^o_{\chi_c} \equiv \frac{g(\alpha^o_c)\sin\mu^o_c}{mV\cos\gamma} \tag{4.1.21}$$

$$u^o_{\gamma_c} \equiv \frac{g(\alpha^o_c)\cos\mu^o_c}{mV} \tag{4.1.22}$$

where

$$g(\alpha^o_c) = L(\alpha^o_c) + T\sin\alpha^o_c \tag{4.1.23}$$

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 4.1.4 and 4.1.8.

Since terms on the right hand side of Equations 4.1.19– 4.1.20, hence $u^o_{\chi_c}$ and $u^o_{\gamma_c}$, are known, unfiltered wind axis commands α^o_c and μ^o_c may be solved for in Equations 4.1.21–4.1.22 with a Cartesian based solution technique discussed in Section 4.1.1.1.

Once μ^o_c and α^o_c are calculated, they are each command filtered, as introduced in § 3.3.5, 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**

$$u_{\chi_c} \equiv \frac{g(\alpha^o_c)\sin\mu^o_c}{mV\cos\gamma} \tag{4.1.24}$$

$$u_{\gamma_c} \equiv \frac{g(\alpha^o_c)\cos\mu^o_c}{mV} \tag{4.1.25}$$

which are fed into the $\zeta_{\mathbf{x}}$–**filters**

$$\dot{\zeta}_\chi = -k_\chi\zeta_\chi + (u_{\chi_c} - u^o_{\chi_c}) \tag{4.1.26}$$

$$\dot{\zeta}_\gamma = -k_\gamma\zeta_\gamma + (u_{\gamma_c} - u^o_{\gamma_c}) \tag{4.1.27}$$

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.

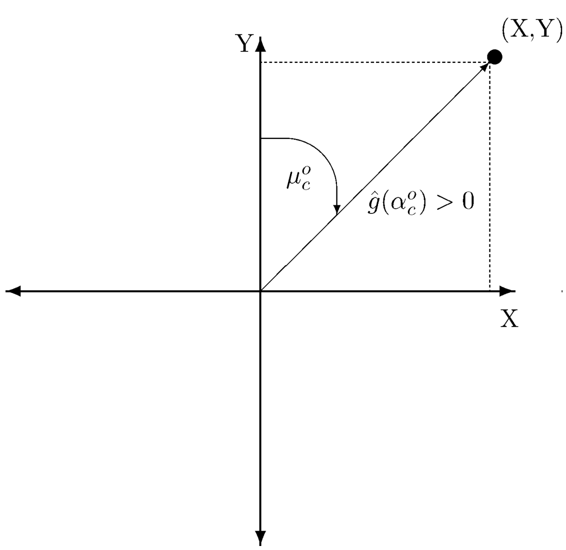
4.1.1.1 Selection of α_c^o and μ_c^o Commands

Defining X and Y such that Equations 4.1.24 and 4.1.27 may be written as

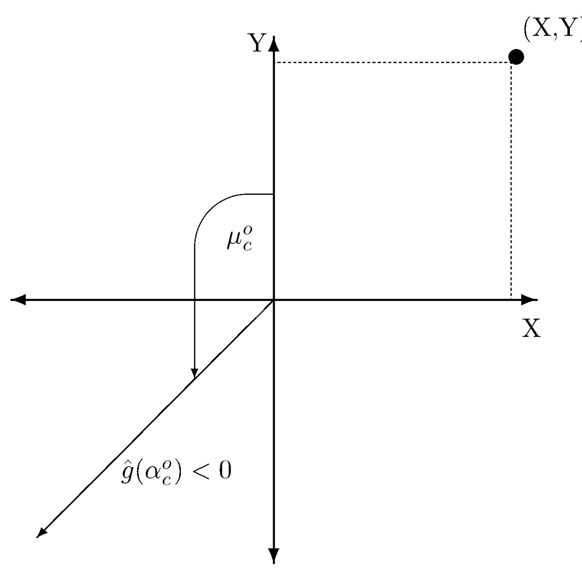
$$X \equiv mV\cos\gamma\, u_\chi = g(\alpha_c^o)\sin(\mu_c^o) \tag{4.1.28}$$

$$Y \equiv mVu_\gamma = g(\alpha_c^o)\cos(\mu_c^o) \tag{4.1.29}$$

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



(a) Positive lift case



(b) Negative lift case

Figure 4.4: Two possible choices for μ_c^o and α_c^o to solve (χ, γ) control. [4]

Since there are two possible solutions for $g(\alpha_c^o)$, as Figure 4.4 depicts, safeguards must be implemented to prevent switching among either option outside of a small range near 0° of bank angle μ_c^o; otherwise, a sign change – μ_c^o 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 (μ_c^o, α_c^o) to satisfy Equation 4.1.21 and 4.1.22, the designer should only allow $g(\alpha_c^o)$ to reverse its sign when [the bank angle virtual control law] u_{χ_c} is near zero." [4].

$$g(X,Y) = \sqrt{X^2+Y^2} = \sqrt{(mV\cos\gamma\, u_\chi)^2 + (mVu_\gamma)^2} \tag{4.1.30}$$

$$\mu_c^o = \text{atan2}(Y,X) = \text{atan2}[(mVu_\gamma),(mV\cos\gamma\, u_\chi)] \tag{4.1.31}$$

$$\alpha_c^o = \text{interp1}\left(g(\boldsymbol{\alpha}),\, \boldsymbol{\alpha},\, g(X,Y)\right) \tag{4.1.32}$$

where 'atan2' is the four quadrant inverse tangent[5] and 'interp1' is 1-D linear interpolation, both of which are inherent to Matlab. Vectors $\boldsymbol{\alpha}$ and $g(\boldsymbol{\alpha})$ are angles-of-attack and pre-calculated signed radii corresponding each $\boldsymbol{\alpha}$ in Figure 4.4, which is characterized by Equation 4.1.23; given a $g(X,Y)$ we may interpolate using the aforementioned vectors to solve for α_c^o.

Once μ_c^o and α_c^o have been specified, then T_c^o may be directly solved for, upcoming in Equation 4.1.37.

4.1.2 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

$$\tilde{V} = V - V_c \tag{4.1.33}$$

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

$$\bar{V} = \tilde{V} - \zeta_V \tag{4.1.34}$$

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

[5] atan2(Y/X) is intentional, not atan2(X/Y) because we defined $0°$ to align with the positive Y-axis.

The immediate goal is to compute unfiltered thrust command T_c^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,

$$u_{V_c}^o = -f_V - F_V + \dot{V}_c - k_V \tilde{V} \tag{4.1.35}$$

where the k_V term is a gain proportional to the tracking error. Note that this virtual control law transforms its respective system, 4.1.9, into a tracking problem, recall Equation 3.3.69.

The **unfiltered control law** is equivalent to its counterpart in Equation 4.1.12, placed in the 'control allocation' simulink subsystem, and defined as

$$u_{V_c}^o \equiv \frac{T_c^o \cos\beta \cos\alpha_c^o}{m} \tag{4.1.36}$$

Since terms on the right hand side of Equation 4.1.35, hence $u_{V_c}^o$, are/is known, the unfiltered thrust command T_c^o may be solved for in Equation 4.1.36 and filtered by a simple saturation and rate limiter directly after calculation. Recall that by the previous Section 4.1.1, α_c^o is available.

In the current implementation T_c^o is not passed through a command filter. If one were to choose to use the command filter, T_c^o would be passed through the filter introduced in § 3.3.5, 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**

$$u_{V_c} \equiv \frac{T_c \cos\beta \cos\alpha_c}{m} \tag{4.1.37}$$

which is fed into the $\zeta_{\mathbf{V}}$**–filter**

$$\dot{\zeta}_V = -k_V \zeta_V + (u_{V_c} - u_{V_c}^o) \tag{4.1.38}$$

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, ie. $u_{V_c} = u_{V_c}^o$.

As previously handled with system dynamics, the equations used to solve for μ_c^o, α_c^o, and T_c^o in Sections 4.1.1 and 4.1.2 may be condensed into block vector notation, as defined in Farrell et al. [19] and Farrell and Polycarpou [4][8.3.1]:

$$G_1(\mu_{1_c}^o, x) = -\bar{A}_1 f_1 - F_1 - K_1 \tilde{z}_1 + \dot{z}_{1_c} \tag{4.1.39}$$

with $G_1 = [u_{\chi_c}^o,\ u_{\gamma_c}^o,\ u_{V_c}^o]^T \equiv [4.1.21,\ 4.1.22,\ 4.1.36]^T$, $\mu_{1_c}^o = [\mu_c^o,\ \alpha_c^o,\ T_c^o]^T$, $\bar{A}_1 =$ Equation 4.1.14, $f_1 = [D,\ Y,\ L]^T$, $F_1 =$ Equation 4.1.15, K_1 is a positive definite diagonal 3x3 matrix, diag$(k_\chi,\ k_\gamma,\ k_V)$, $\tilde{z}_1 = z_1 - z_{1_c}$, and $z_1 = [\chi,\ \gamma,\ V]^T$. The goal is to select $\mu_{1_c}^o$ so that Equation 4.1.39 holds, then command filter $z_{2_c}^o$ to produce the signals z_{2_c} and $\dot{z}_{2_c}$.

Assuming that the solution $\mu_{1_c}^o$ to Equation 4.1.39 has been found, **define**

$$z_{2_c}^o = \mu_{1_c}^o = [\mu_c^o,\ \alpha_c^o,\ \beta_c^o] \tag{4.1.40}$$

and the ζ_1–filter in block vector notation be defined as

$$\dot{\zeta}_1 = -K_1 \zeta_1 + (G(z_2, x) - G(z_{2_c}^o, x)) \tag{4.1.41}$$

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

4.2 Middle Loop

Wind-axis **dynamics for the middle loop**, Equations 3.1.42d, 3.1.42e, and 3.1.42f respectively, are now grouped into sub-functions of their partially or completely unknown f, known F, and control signals u:

Bank Angle Dynamics

$$\dot{\mu} = f_\mu + F_\mu + u_\mu \tag{4.2.1}$$

$$
\begin{aligned}
f_\mu &= \frac{1}{mV}\left[D\sin\beta\tan\gamma\cos\mu + Y\cos\beta\tan\gamma\cos\mu + L\left(\tan\beta+\tan\gamma\sin\mu\right)\right] \quad (4.2.2)\\
F_\mu &= \frac{1}{mV}\left[T\left(\sin\alpha\tan\gamma\sin\mu + \sin\alpha\tan\beta - \cos\alpha\sin\beta\tan\gamma\cos\mu\right)\right.\\
&\quad \left. - mg\tan\beta\cos\gamma\cos\mu\right] \quad (4.2.3)\\
u_\mu &= \frac{P\cos\alpha + R\sin\alpha}{\cos\beta} \equiv \frac{P_s}{\cos\beta} \quad (4.2.4)
\end{aligned}
$$

Angle-of-Attack Dynamics

$$
\dot{\alpha} = f_\alpha + F_\alpha + u_\alpha \quad (4.2.5)
$$

$$
\begin{aligned}
f_\alpha &= -\frac{L}{mV\cos\beta} \quad (4.2.6)\\
F_\alpha &= \frac{1}{mV\cos\beta}\left[-T\sin\alpha + mg\cos\gamma\cos\mu\right] \quad (4.2.7)\\
u_\alpha &= Q - \tan\beta\left(P\cos\alpha + R\sin\alpha\right) \equiv Q - P_s\tan\beta \quad (4.2.8)
\end{aligned}
$$

Angle-of-Sideslip Dynamics

$$
\dot{\beta} = f_\beta + F_\beta + u_\beta \quad (4.2.9)
$$

$$
\begin{aligned}
f_\beta &= \frac{1}{mV}\left(D\sin\beta + Y\cos\beta\right) \quad (4.2.10)\\
F_\beta &= \frac{1}{mV} - T\sin\beta\cos\alpha + mg\cos\gamma\sin\mu \quad (4.2.11)\\
u_\beta &= P\sin\alpha - R\cos\alpha \equiv -R_s \quad (4.2.12)
\end{aligned}
$$

In block vector notation Equations 4.2.1, 4.2.5, and 4.2.9 may be combined, as defined in Farrell et al. [19] and Farrell and Polycarpou [4][8.3.2]:

$$
\dot{z}_2 = A_2 f_1 + F_2 + B_2\mu_2 \quad (4.2.13)
$$

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

$$A_2 = \frac{1}{mV}\begin{bmatrix} \sin\beta\cos\mu\tan\gamma & \cos\beta\cos\mu\tan\gamma & \tan\beta + \tan\gamma\sin\mu \\ 0 & 0 & -1/\cos\beta \\ \sin\beta & \cos\beta & 0 \end{bmatrix} \tag{4.2.14}$$

$$F_2 = \frac{1}{mV}\begin{bmatrix} T(\sin\alpha\tan\gamma\sin\mu + \sin\alpha\tan\beta - \cos\alpha\tan\gamma\cos\mu\sin\beta) - mg\cos\gamma\cos\mu\tan\beta \\ (-T\sin\alpha + mg\cos\gamma\cos\mu)/(\cos\beta) \\ -T\sin\beta\cos\alpha + mg\cos\gamma\sin\mu \end{bmatrix} \tag{4.2.15}$$

$$B_2 = \begin{bmatrix} \cos\alpha/\cos\beta & 0 & \sin\alpha/\cos\beta \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix} \tag{4.2.16}$$

where $\mu_2 = [P\,, Q\,, R]^T$.

4.2.1 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

$$\tilde{\mu} = \mu - \mu_c\,, \qquad \tilde{\alpha} = \alpha - \alpha_c\,, \qquad \tilde{\beta} = \beta - \beta_c \tag{4.2.17}$$

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

$$\bar{\mu} = \tilde{\mu} - \zeta_\mu\,, \qquad \bar{\alpha} = \tilde{\alpha} - \zeta_\alpha\,, \qquad \bar{\beta} = \tilde{\beta} - \zeta_\beta \tag{4.2.18}$$

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; ζ_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,

$$u_{\mu_c}^o = -f_\mu - F_\mu + \dot{\mu}_c - k_\mu \tilde{\mu} \tag{4.2.19}$$

$$u_{\alpha_c}^o = -f_\alpha - F_\alpha + \dot{\alpha}_c - k_\alpha \tilde{\alpha} \tag{4.2.20}$$

$$u_{\beta_c}^o = -f_\beta - F_\beta + \dot{\beta}_c - k_\beta \tilde{\beta} \tag{4.2.21}$$

where the k terms are gains proportional to the tracking error. Note that these virtual control laws transform their respective systems, 4.2.1, 4.2.5 and 4.2.9, into a tracking problem, recall Equation 3.3.69.

Unfiltered control laws are equivalent to their counterparts in Equations 4.2.4, 4.2.8, and 4.2.12, placed in the 'control allocation' simulink subsystem, and defined as

$$u_{\mu_c}^o \equiv P_c^o\left(\frac{\cos\alpha}{\cos\beta}\right) + R_c^o\left(\frac{\sin\alpha}{\cos\beta}\right) = \frac{P_{sc}^o}{\cos\beta} \tag{4.2.22}$$

$$u_{\alpha_c}^o \equiv Q_c^o - P_c^o(\cos\alpha\tan\beta) - R_c^o(\sin\alpha\tan\beta) = Q_c^o - P_{sc}^o\tan\beta \tag{4.2.23}$$

$$u_{\beta_c}^o \equiv P_c^o\sin\alpha - R_c^o\cos\alpha = -R_{sc}^o \tag{4.2.24}$$

Since terms on the right hand side of Equations 4.2.19– 4.2.21, hence $u_{\mu_c}^o$, $u_{\alpha_c}^o$ and $u_{\beta_c}^o$, are known, unfiltered wind axis commands P_c^o, Q_c^o, and R_c^o may be solved for in Equations 4.2.22–4.2.24 directly: First transform unfiltered control law equations to a matrix representation

$$\begin{bmatrix} u_{\mu_c}^o \\ u_{\alpha_c}^o \\ u_{\beta_c}^o \end{bmatrix} \equiv \begin{bmatrix} \cos\alpha/\cos\beta & 0 & \sin\alpha\cos\beta \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix} \begin{bmatrix} P_c^o \\ Q_c^o \\ R_c^o \end{bmatrix} = B_2\mu_{2_c}^o \tag{4.2.25}$$

and solve for $\mu_{2_c}^o = [P_c^o, Q_c^o, R_c^o]^T$ by taking the inverse of B_2 and pre-multiplying by the known virtual control law vector

$$\begin{bmatrix} P_c^o \\ Q_c^o \\ R_c^o \end{bmatrix} = \begin{bmatrix} \cos\alpha\cos\beta & 0 & \sin\alpha \\ \sin\beta & 1 & 0 \\ \cos\beta\sin\alpha & 0 & -\cos\alpha \end{bmatrix} \begin{bmatrix} u_{\mu_c}^o \\ u_{\alpha_c}^o \\ u_{\beta_c}^o \end{bmatrix} \tag{4.2.26}$$

Once P_c^o , Q_c^o, and R_c^o are calculated, they are each command filtered, as introduced in § 3.3.5, 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**

$$u_{\mu_c} \equiv P_c\left(\frac{\cos\alpha}{\cos\beta}\right) + R_c\left(\frac{\sin\alpha}{\cos\beta}\right) = \frac{P_{sc}}{\cos\beta} \tag{4.2.27}$$

$$u_{\alpha_c} \equiv Q_c - P_c(\cos\alpha\tan\beta) - R_c(\sin\alpha\tan\beta) = Q_c - P_{sc}\tan\beta \tag{4.2.28}$$

$$u_{\beta_c} \equiv P_c\sin\alpha - R_c\cos\alpha = -R_{sc} \tag{4.2.29}$$

which are fed into the $\zeta_{\mathbf{x}}$**–filters**

$$\dot{\zeta}_\mu = -k_\mu\zeta_\mu + (u_{\mu_c} - u_{\mu_c}^o) \tag{4.2.30}$$

$$\dot{\zeta}_\alpha = -k_\alpha\zeta_\alpha + (u_{\alpha_c} - u_{\alpha_c}^o) \tag{4.2.31}$$

$$\dot{\zeta}_\beta = -k_\beta\zeta_\beta + (u_{\beta_c} - u_{\beta_c}^o) \tag{4.2.32}$$

that compensate for the tracking errors $\tilde{\mu}$, $\tilde{\alpha}$, and $\tilde{\beta}$ for the effect of any differences between the desired $(u_{\mu_c}^o\,, u_{\alpha_c}^o\,, u_{\beta_c}^o)$ 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. [19] and Farrell and Polycarpou [4][8.3.2]:

$$B_2\mu_{2_c}^o = -A_2 f_1 - F_2 - K_2\tilde{z}_2 + \dot{z}_{2_c} \tag{4.2.33}$$

with $B_2\mu_{2_c}^o = [u_{\mu_c}^o,\, u_{\alpha_c}^o,\, u_{\beta_c}^o]^T \equiv [4.2.22,\, 4.2.23,\, 4.2.24]^T$, $\mu_{2_c}^o = [P_c^o,\, Q_c^o,\, R_c^o]^T$, $A_2 =$ Equation 4.2.14, $f_1 = [D,\, Y,\, L]^T$, $F_2 =$ Equation 4.2.15, K_2 is a positive definite diagonal 3x3 matrix, $\mathrm{diag}(k_\mu,\, k_\alpha,\, k_\beta)$, $\tilde{z}_2 = z_2 - z_{2_c}$, and $z_2 = [\mu,\, \alpha,\, \beta]^T$. The goal is to select $\mu_{2_c}^o$ so that Equation 4.2.33 holds, then command filter $z_{3_c}^o$ to produce the signals z_{3_c} and $\dot{z}_{3_c}$.

Assuming that the solution $\mu_{2_c}^o$ to Equation 4.2.33 has been found, **define**

$$z_{3_c}^o = \mu_{2_c}^o - \zeta_3 \tag{4.2.34}$$

where ζ_3 will be defined in the next section, and the ζ_2–filter in block vector notation be defined as

$$\dot{\zeta}_2 = -K_2\zeta_2 + B_2(z_{3_c} - z_{3_c}^o) \tag{4.2.35}$$

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

4.3 Inner Loop

Body-axis **dynamics for the inner loop**, Equations 3.1.42g, 3.1.42h, and 3.1.42i respectively, are now grouped into sub-functions of their partially or completely unknown f, known F, and control signals u:

Roll Rate Dynamics

$$\dot{P} = f_P + F_P + u_P \tag{4.3.1}$$

$$\begin{aligned} f_P &= c_3\bar{L}' + c_4\bar{N}' && (4.3.2)\\ F_P &= (c_1R + c_2P)\,Q && (4.3.3)\\ u_P &= c_3\sum_{i=1}^{n}\bar{L}_{\delta_i}\delta_i + c_4\sum_{i=1}^{n}\bar{N}_{\delta_i}\delta_i && (4.3.4)\end{aligned}$$

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

Pitch Rate Dynamics

$$\dot{Q} = f_Q + F_Q + u_Q \tag{4.3.5}$$

$$f_Q = c_7 \bar{M}' \tag{4.3.6}$$

$$F_Q = c_5 PR - c_6 \left(P^2 - R^2\right) \tag{4.3.7}$$

$$u_Q = c_7 \sum_{i=1}^{n} \bar{M}_{\delta_i} \delta_i \tag{4.3.8}$$

where the pitching moment was decomposed into moment due to aerodynamics and deflections as $\bar{M} = \bar{M}' + \sum_{i=1}^{n} \bar{M}_{\delta_i} \delta_i$

Yaw Rate Dynamics

$$\dot{R} = f_R + F_R + u_R \tag{4.3.9}$$

$$f_R = c_4 \bar{L}' + c_9 \bar{N}' \tag{4.3.10}$$

$$F_R = \left(c_8 P - c_2 R\right) Q \tag{4.3.11}$$

$$u_R = c_4 \sum_{i=1}^{n} \bar{L}_{\delta_i} \delta_i + c_9 \sum_{i=1}^{n} \bar{N}_{\delta_i} \delta_i \tag{4.3.12}$$

where the rolling and yawing moments were decomposed as in Equation 4.3.2.

In block vector notation Equations 4.3.1, 4.3.5, and 4.3.9 may be combined, as defined in Farrell et al. [19] and Farrell and Polycarpou [4][8.3.3]:

$$\dot{z}_3 = A_3 f_3 + F_3 + B_3 G_3 \boldsymbol{\delta} \tag{4.3.13}$$

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

$$A_3 = B_3 = \begin{bmatrix} c_3 & 0 & c_4 \\ 0 & c_7 & 0 \\ c_4 & 0 & c_9 \end{bmatrix} \tag{4.3.14}$$

where c terms are not elements of the moment of inertia matrix, recall Equation 3.1.43 which defines c-values as a combination of moment of inertia matrix elements,

$$F_3 = \begin{bmatrix} (c_1 R + c_2 P) Q \\ c_5 PR - c_6 (P^2 - R^2) \\ (c_8 P - c_2 R) Q \end{bmatrix} \tag{4.3.15}$$

and

$$G_3 = \begin{bmatrix} \bar{L}_{\delta_1}, & \cdots, & \bar{L}_{\delta_6} \\ \bar{M}_{\delta_1} & \cdots & \bar{M}_{\delta_6} \\ \bar{N}_{\delta_1}, & \cdots, & \bar{N}_{\delta_6} \end{bmatrix} \tag{4.3.16}$$

4.3.1 Body-Axis Angular Rate Control

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

$$\tilde{P} = P - P_c\,, \qquad \tilde{Q} = Q - Q_c\,, \qquad \tilde{R} = R - R_c \tag{4.3.17}$$

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

$$\bar{P} = \tilde{P} - \zeta_P\,, \qquad \bar{Q} = \tilde{Q} - \zeta_Q\,, \qquad \bar{R} = \tilde{R} - \zeta_R \tag{4.3.18}$$

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

The immediate goal is to compute unfiltered deflection commands, $\delta^o_{1_c}, \cdots, \delta^o_{6_c}$. 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,

$$u^o_{P_c} = -f_P - F_P + \dot{P}_c - k_P\tilde{P} - x_P \tag{4.3.19}$$

$$u^o_{Q_c} = -f_Q - F_Q + \dot{Q}_c - k_Q\tilde{Q} - x_Q \tag{4.3.20}$$

$$u^o_{R_c} = -f_R - F_R + \dot{R}_c - k_R\tilde{R} - x_R \tag{4.3.21}$$

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

$$x_P = \bar{\mu}\frac{\cos\alpha}{\cos\beta} - \bar{\alpha}\cos\alpha\tan\beta + \bar{\beta}\sin\alpha \tag{4.3.22}$$

$$x_Q = \bar{\alpha} \tag{4.3.23}$$

$$x_R = \bar{\mu}\frac{\sin\alpha}{\cos\beta} - \bar{\alpha}\sin\alpha\tan\beta - \bar{\beta}\cos\alpha \tag{4.3.24}$$

which allows the subsequent Lyapunov stability analysis in section 4.4 to stabilize the system for the desired trajectory.

Note that these virtual control laws transform their respective systems, 4.3.1, 4.3.5 and 4.3.9, into a tracking problem, recall Equation 3.3.69.

Unfiltered control laws are equivalent to their counterparts in Equations 4.3.4, 4.3.8, and 4.3.12, placed in the 'control allocation' simulink subsystem, and defined as

$$u^o_{P_c} \equiv c_3\sum_{i=1}^{n}\bar{L}_{\delta_i}\delta^o_{i_c} + c_4\sum_{i=1}^{n}\bar{N}_{\delta_i}\delta^o_{i_c} \tag{4.3.25}$$

$$u^o_{Q_c} \equiv c_7\sum_{i=1}^{n}\bar{M}_{\delta_i}\delta^o_{i_c} \tag{4.3.26}$$

$$u^o_{R_c} \equiv c_4\sum_{i=1}^{n}\bar{L}_{\delta_i}\delta^o_{i_c} + c_9\sum_{i=1}^{n}\bar{N}_{\delta_i}\delta^o_{i_c} \tag{4.3.27}$$

Since terms on the right hand side of Equations 4.3.19– 4.3.21, hence $u^o_{P_c}$, $u^o_{Q_c}$ and $u^o_{R_c}$, are known, unfiltered deflection commands $\delta^o_{1_c}, \cdots, \delta^o_{6_c}$ may be solved for in Equations 4.3.25–4.3.27 directly with some form of **actuator distribution**.

Once $\delta^o_{1_c}, \cdots, \delta^o_{6_c}$ are calculated, they are each command filtered, as introduced in § 3.3.5, to produce "magnitude, rate, and bandwidth-limited signals" $\delta_{1_c}, \cdots, \delta_{6_c}$. These filtered commands will then be used to generate **achievable control laws**

$$u_{P_c} \equiv c_3 \sum_{i=1}^{n} \bar{L}_{\delta_i} \delta_{i_c} + c_4 \sum_{i=1}^{n} \bar{N}_{\delta_i} \delta_{i_c} \tag{4.3.28}$$

$$u_{Q_c} \equiv c_7 \sum_{i=1}^{n} \bar{M}_{\delta_i} \delta_{i_c} \tag{4.3.29}$$

$$u_{R_c} \equiv c_4 \sum_{i=1}^{n} \bar{L}_{\delta_i} \delta_{i_c} + c_9 \sum_{i=1}^{n} \bar{N}_{\delta_i} \delta_{i_c} \tag{4.3.30}$$

which are fed into the $\zeta_{\mathbf{x}}$–**filters**

$$\dot{\zeta}_P = -k_P \zeta_P + (u_{P_c} - u^o_{P_c}) \tag{4.3.31}$$

$$\dot{\zeta}_Q = -k_Q \zeta_Q + (u_{Q_c} - u^o_{Q_c}) \tag{4.3.32}$$

$$\dot{\zeta}_R = -k_R \zeta_R + (u_{R_c} - u^o_{R_c}) \tag{4.3.33}$$

that compensate for the tracking errors $\tilde{P}$, $\tilde{Q}$, and $\tilde{R}$ for the effect of any differences between the desired $(u^o_{P_c},\, u^o_{Q_c},\, u^o_{R_c})$ and achievable $(u_{P_c},\, u_{Q_c},\, u_{R_c})$ control signals.

As previously handled with system dynamics, the equations used to solve for $\delta_{1_c}, \cdots, \delta_{6_c}$ in this section may be condensed into block vector notation, as defined by Farrell et al. [19] and Farrell and Polycarpou [4][8.3.3]:

$$B_3 G_3 \boldsymbol{\delta}^o_c = -A_3 f_3 - F_3 - K_3 \tilde{z}_3 + \dot{z}_{3_c} - B_2^T \bar{z}_2 \tag{4.3.34}$$

with $B_3 G_3 \boldsymbol{\delta}^o_c = [u^o_{P_c},\, u^o_{Q_c},\, u^o_{R_c}]^T \equiv [4.3.25,\, 4.3.26,\, 4.3.27]^T$, $\boldsymbol{\delta}^o_c = [\delta^o_{1_c}, \cdots, \delta^o_{6_c}]^T$, $A_3 =$ Equation 4.3.14, $f_3 = [\bar{L}',\, \bar{M}',\, \bar{N}']^T$, $F_3 =$ Equation 4.3.15, K_3 is a positive definite diagonal 3x3 matrix, $\mathrm{diag}(k_P,\, k_Q,\, k_R)$, and $z_3 = [P,\, Q,\, R]^T$; the goal is to select $\boldsymbol{\delta}^o_c$ such that Equation 4.3.34 holds. If there are more columns than rows in $B_3 G_3$ the aircraft is over-actuated, ie. multiple solutions to Equation 4.3.34 exist and some form of actuator distribution is required to select $\boldsymbol{\delta}^o_c$.

Assume that the deflection solution to Equation 4.3.34 has been found, by some form of actuator distribution, and the ζ_3–filter in block vector notation be defined as

$$\dot{\zeta}_3 = -K_3 \zeta_3 + B_3 G_3 (\delta - \delta^o_c) \tag{4.3.35}$$

4.4 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:

System Dynamics

$$\dot{z}_1 = \bar{A}_1 f_1 + F_1 + G_1(z_2) \tag{4.1.13}$$

$$\dot{z}_2 = A_2 f_1 + F_2 + B_2 z_3 \tag{4.2.13}$$

$$\dot{z}_3 = A_3 f_3 + F_3 + B_3 G_3 \boldsymbol{\delta} \tag{4.3.13}$$

Control Allocation / Virtual Control Laws

$$G_1(\mu^o_{1_c}, x) = -\bar{A}_1 f_1 - F_1 - K_1 \tilde{z}_1 + \dot{z}_{1_c} \tag{4.1.39}$$

$$B_2 \mu^o_{2_c} = -A_2 f_1 - F_2 - K_2 \tilde{z}_2 + \dot{z}_{2_c} \tag{4.2.33}$$

$$B_3 G_3 \boldsymbol{\delta}^o_c = -A_3 f_3 - F_3 - K_3 \tilde{z}_3 + \dot{z}_{3_c} - B_2^T \bar{z}_2 \tag{4.3.34}$$

with definitions $z^o_{2_c} = \mu^o_{1_c}$ and $z^o_{3_c} = \mu^o_{2_c} - \zeta_3$ for 4.1.39 and 4.2.33 respectively.

ζ–Filters

$$\dot{\zeta}_1 = -K_1 \zeta_1 + G(z_2 - z^o_{2_c}) \tag{4.1.41}$$

$$\dot{\zeta}_2 = -K_2 \zeta_2 + B_2(z_{3_c} - z^o_{3_c}) \tag{4.2.35}$$

$$\dot{\zeta}_3 = -K_3 \zeta_3 + B_3 G_3(\delta - \delta^o_c) \tag{4.3.35}$$

Tracking & Compensated Tracking Errors

$$\tilde{z}_1 = z_1 - z_{1_c}, \qquad \tilde{z}_2 = z_2 - z_{2_c}, \qquad \tilde{z}_3 = z_3 - z_{3_c} \tag{4.4.1}$$

$$\bar{z}_1 = \tilde{z}_1 - \zeta_1, \qquad \bar{z}_2 = \tilde{z}_2 - \zeta_2, \qquad \bar{z}_3 = \tilde{z}_3 - \zeta_3 \tag{4.4.2}$$

4.4.1 Tracking Error Dynamics

$$\begin{aligned}
\dot{\tilde{z}}_1 &= [\dot{z}_1] - [\dot{z}_{1_c}] \\
&= [\cancel{\bar{A}_1 f_1} + \cancel{F_1} + G(z_2)] - [G(z^o_{2_c}) + \cancel{\bar{A}_1 f_1} + \cancel{F_1} + K_1\tilde{z}_1] \pm G(z_{2_c}) \\
&= -K_1\tilde{z}_1 + G(z_2 - z_{2_c}) + G(z_{2_c} - z^o_{2_c})
\end{aligned} \tag{4.4.3}$$

$$\begin{aligned}
\dot{\tilde{z}}_2 &= [\dot{z}_2] - [\dot{z}_{2_c}] \\
&= [\cancel{A_2 f_1} + \cancel{F_2} + B_2 z_3] - [B_2(z^o_{3_c} + \zeta_3) + \cancel{A_2 f_1} + \cancel{F_2} + K_2\tilde{z}_2] \pm B_2 z_{3_c} \\
&= -K_2\tilde{z}_2 + B_2(z_3 - z_{3_c}) + B_2(z_{3_c} - z^o_{3_c}) - B_2\zeta_3 \\
&= -K_2\tilde{z}_2 + B_2\bar{z}_3 + B_2(z_{3_c} - z^o_{3_c})
\end{aligned} \tag{4.4.4}$$

$$\begin{aligned}
\dot{\tilde{z}}_3 &= [\dot{z}_3] - [\dot{z}_{3_c}] \\
&= [\cancel{A_3 f_3} + \cancel{F_3} + B_3 G_3 \delta] - [B_3 G_3 \delta^o_c + \cancel{A_3 f_3} + \cancel{F_3} + K_3\tilde{z}_3 + B_2^T\bar{z}_2] \pm B_3 G_3 \delta_c \\
&= -K_3\tilde{z}_3 + B_3 G_3(\delta - \delta_c) + B_3 G_3(\delta_c - \delta^o_c) - B_2^T\bar{z}_2 \\
&= -K_3\tilde{z}_3 - B_2^T\bar{z}_2 + B_3 G_3(\delta - \delta_c) + B_3 G_3(\delta_c - \delta^o_c)
\end{aligned} \tag{4.4.5}$$

$$\begin{aligned}
\dot{\bar{z}}_1 &= [\dot{\tilde{z}}_1] - [\dot{\zeta}_1] \\
&= [-K_1\tilde{z}_1 + G(z_2 - z_{2_c}) + G(z_{2_c} - z^o_{2_c})] - [-K_1\zeta_1 + G(z_2 - z^o_{2_c})] \\
&= -K_1\tilde{z}_1 + K_1\zeta_1 + G(\cancel{z_2} - \cancel{z_{2_c}}) + G(\cancel{z_{2_c}} - \cancel{z^o_{2_c}}) + G(\cancel{z^o_{2_c}} - \cancel{z_2}) \\
&= -K_1(\zeta_1 - \tilde{z}_1) \\
&= -K_1\bar{z}_1
\end{aligned} \tag{4.4.6}$$

$$
\begin{aligned}
\dot{\bar{z}}_2 &= [\dot{\tilde{z}}_2] - [\dot{\zeta}_2] \\
&= [-K_2\tilde{z}_2 + B_2\bar{z}_3 + B_2(z_{3_c} - z^o_{3_c})] - [-K_2\zeta_2 + B_2(z_{3_c} - z^o_{3_c})] \\
&= -K_2\tilde{z}_2 + K_2\zeta_2 + B_2\bar{z}_3 + B_2(\cancel{z_{3_c}} - \cancel{z^o_{3_c}}) + B_2(\cancel{z^o_{3_c}} - \cancel{z^o_{3_c}}) \\
&= -K_2(\tilde{z}_2 - \zeta_2) + B_2\bar{z}_3 \\
&= -K_2\bar{z}_2 - B_2\bar{z}_3
\end{aligned}
\tag{4.4.7}
$$

$$
\begin{aligned}
\dot{\bar{z}}_3 &= [\dot{\tilde{z}}_3] - [\dot{\zeta}_3] \\
&= [-K_3\tilde{z}_3 - B_2^T\bar{z}_2 + B_3G_3(\delta - \delta_c) + B_3G_3(\delta_c - \delta_c^o)] - [-K_3\zeta_3 + B_3G_3(\delta - \delta_c^o)] \\
&= -K_3\tilde{z}_3 + K_3\zeta_3 + B_3G_3(\cancel{\delta} - \cancel{\delta_c}) + B_3G_3(\cancel{\delta_c} - \cancel{\delta_c^o}) + B_3G_3(\cancel{\delta_c^o} - \cancel{\delta}) - B_2^T\bar{z}_2 \\
&= -K_3(\tilde{z}_3 - \zeta_3) - B_2^T\bar{z}_2 \\
&= -K_3\bar{z}_3 - B_2^T\bar{z}_2
\end{aligned}
\tag{4.4.8}
$$

4.4.2 Lyapunov Theorem Application

Akin to Section 3.3.4.3, Equation 3.3.80 define the Lyapunov function for this system to be

$$
V = \frac{1}{2}\left[\sum_{i=1}^{3} \bar{z}_i^T \bar{z}_i\right] \tag{4.4.9}
$$

The time derivative of the Lyapunov function along solutions from Equation 4.4.1 to 4.4.1 is given by

$$
\begin{aligned}
\dot{V} &= \bar{z}_1^T\dot{\bar{z}}_1 + \bar{z}_2^T\dot{\bar{z}}_2 + \bar{z}_3^T\dot{\bar{z}}_3 \\
&= -\bar{z}_1^T K_1\bar{z}_1 - \bar{z}_2^T K_2\bar{z}_2 + \cancel{\bar{z}_2^T B_2\bar{z}_3} - \bar{z}_3^T K_3\bar{z}_3 - \cancel{\bar{z}_3^T B_2^T\bar{z}_2} \\
&= -\bar{z}_1^T K_1\bar{z}_1 - \bar{z}_2^T K_2\bar{z}_2 - \bar{z}_3^T K_3\bar{z}_3
\end{aligned}
\tag{4.4.10}
$$

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 $-B_2^T\bar{z}_2$ in the virtual control law developed for the inner loop, Equation 4.3.34, was implemented: to cancel the $\bar{z}_2^T B_2 \bar{z}_3$ term in Equation 4.4.10! This is absolutely incredible, as we now have a positive definite Lyapunov function with a derivative that guarantees stability for the system's operating point.

Apply Barbalat's Theorem...

Chapter 5

Application

Will eventually test in hardware... In the meanwhile and for the thesis I will be running a purely computer simulation.

5.1 Modeling

Describe simulink environment and any models referenced from AME (i.e the aerodynamics model for Fury based on windtunnel data / CFD analysis)

Introduce 3 view of air vehicle

5.2 Simulation

List simulation stipulations: ie. back up assumptions (constant mass, deflections assumed to be measured, ideal actuators, constant density therefore altitude, what have you)

Possibly discuss simulation configuration –¿ ODE solver, considerations or lessons learned for those who follow. ie. dave suggested typecasting vars for future encapsulation into hardward (FCAS), library block usage, unit testing, initialization w/ memory block (avoid algebraic loops), how to send items to command workspae for plotting nicely, ...

5.2.1 Nominal

Plot time histories between command and response $[\gamma \mu V]$

5.2.2 Turbulence

Throw in atmospheric model to test turbulence? Currently model has constant rho... hmmm

5.2.3 Modeling Error / Unfamiliar Plant

Won't work if I don't have aero function approximations

5.2.4 Effector Failure

Won't work unless control allocation robust

5.3 Results

Chapter 6

Conclusion

Could've covered...

1. adaptive aero coefficient modeling
2. linearization around operating points to evaluate PM/GM in classic controls sense
3. New(er) derivation for backstepping that uses tikanov's theorem to guaranteed GES.
4. hardware in the loop testing

GES stability vis Tikhonov's Theorem: Farrell et al. [20], Farrell et al. [21]

Bibliography

[1] Hassan K. Khalil. *Nonlinear Systems*. Prentice-Hall, Upper Saddle River, New Jersey, 2nd edition, 1996. ISBN 0132280248. URL http://www.worldcat.org/oclc/757428832.

[2] Jean-Jacques E Slotine and Li Weiping. *Applied Nonlinear Control*. Prentice-Hall, Englewood Cliffs, NJ, 1st edition, 1991. ISBN 0130408905. URL http://www.worldcat.org/oclc/636122876.

[3] L. Sonneveldt, Q. P. Chu, and J. A. Mulder. Nonlinear Flight Control Design Using Constrained Adaptive Backstepping. *Journal of Guidance, Control, and Dynamics*, 30 (2):322–336, March 2007. ISSN 0731-5090. doi: 10.2514/1.25834. URL http://doi.aiaa.org/10.2514/1.25834.

[4] Jay A. Farrell and Marios M. Polycarpou. *Adaptive Approximation Based Control*. Wiley-Interscience, Hoboken, NJ, 1st edition, 2006. ISBN 0471727881. URL http://www.worldcat.org/oclc/61169767.

[5] Randy A. Freeman and Petar V. Kokotovic. *Robust Nonlinear Control Design*. Birkhäuser Boston, Boston, MA, 2008 edition, 1996. ISBN 978-0-8176-4758-2. doi: 10.1007/978-0-8176-4759-9. URL http://www.springerlink.com/index/10.1007/978-0-8176-4759-9.

[6] Petar V. Kokotovic. The joy of feedback: nonlinear and adaptive. *IEEE Control*

Systems Magazine, 12(3):7–17, June 1992. ISSN 02721708. doi: 10.1109/37.165507. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=165507.

[7] Ioannis Kanellakopoulos, Petar V. Kokotovic, and A.S. Morse. Systematic design of adaptive controllers for feedback linearizable systems. *IEEE Transactions on Automatic Control*, 36(11):1241–1253, 1991. ISSN 00189286. doi: 10.1109/9.100933. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=100933.

[8] Ioannis Kanellakopoulos. A toolkit for nonlinear feedback design. *Systems & Control Letters*, 18(2):83–92, February 1992. ISSN 01676911. doi: 10.1016/0167-6911(92)90012-H. URL http://linkinghub.elsevier.com/retrieve/pii/016769119290012H.

[9] Miroslav Krstic, Ioannis Kanellakopoulos, and Petar V. Kokotovic. *Nonlinear and adaptive control design*. John Wiley & Sons, Inc., New York, 1995. ISBN 0471127329. URL http://www.worldcat.org/title/nonlinear-and-adaptive-control-design/oclc/32166728?referer=di&ht=edition.

[10] Honeywell Inc. and Lockheed Martin Skunk Works. Application of Multivariable Control Theory to Aircraft Control Laws. Final Report: Multivariable Control Design Guideline. Technical Report March 1993, Wright Laboratory, Palmdale, CA, 1996. URL http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA315259.

[11] Courtland D. Perkins and Robert E. Hage. *Airplane Performance, Stability, and Control*. Wiley, New York, 1949. ISBN 047168046X. URL http://www.worldcat.org/oclc/1650766.

[12] Duane T. McRuer, Irving Ashkenas, and Dunstan Graham. *Aircraft Dynamics and Automatic Control*. Princeton University Press, New Jersey, 1973. ISBN 0691080836. URL http://www.worldcat.org/oclc/900904.

[13] Brian Stevens and Frank Lewis. *Aircraft Control and Simulation*. Wiley, Hoboken, NJ, 1st edition, 1992. ISBN 0471613975. URL http://www.worldcat.org/oclc/23356757.

[14] Brian Stevens and Frank Lewis. *Aircraft Control and Simulation*. Wiley, Hoboken, NJ, 2nd edition, 2003. ISBN 978-0-471-37145-8. URL http://www.worldcat.org/oclc/51751879.

[15] Ola Härkegå rd. *Backstepping and Control Allocation with Applications to Flight Control*. Dissertation, Linköping University, 2003. URL http://www.control.isy.liu.se/research/reports/Ph.D.Thesis/PhD820.pdf.

[16] Jay A. Farrell, Marios M. Polycarpou, and Manu Sharma. On-Line Approximation Based Control of Uncertain Nonlinear Systems with Magnitude , Rate and Bandwidth Constraints on the States and Actuators. In *Proc. of the American Control Conference*, pages 2557–2562, Boston, MA, 2004. IEEE. doi: 0.1109/ACC.2004.182491.

[17] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications To Physics, Biology, Chemistry, And Engineering*. Perseus Books, Westview Press, Cambridge, MA, 1st edition, 1994. ISBN 0738204536. URL http://www.worldcat.org/title/nonlinear-dynamics-and-chaos-with-applications-to-physics-biology-chemistry-and-eng oclc/42140115.

[18] Miroslav Krstic, D. Fontaine, Petar V. Kokotovic, and J.D. Paduano. Useful nonlinearities and global stabilization of bifurcations in a model of jet engine surge and stall. *IEEE Transactions on Automatic Control*, 43(12):1739–1745, 1998. ISSN 00189286. doi: 10.1109/9.736075. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=736075.

[19] Jay A. Farrell, Marios M. Polycarpou, and Manu Sharma. Backstepping-Based Flight Control with Adaptive Function Approximation. *Journal of Guidance, Control, and*

Dynamics, 28(6):1089–1102, November 2005. ISSN 0731-5090. doi: 10.2514/1.13030. URL http://doi.aiaa.org/10.2514/1.13030.

[20] Jay A. Farrell, Marios M. Polycarpou, Manu Sharma, and Wenjie Dong. Command Filtered Backstepping. *2008 American Control Conference*, pages 1923–1928, June 2008. doi: 10.1109/ACC.2008.4586773. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4586773.

[21] Jay A. Farrell, Marios M. Polycarpou, and Manu Sharma. Command Filtered Backstepping. *IEEE Transactions on Automatic Control*, 54(6):1391–1395, June 2009. ISSN 0018-9286. doi: 10.1109/TAC.2009.2015562. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4982643.