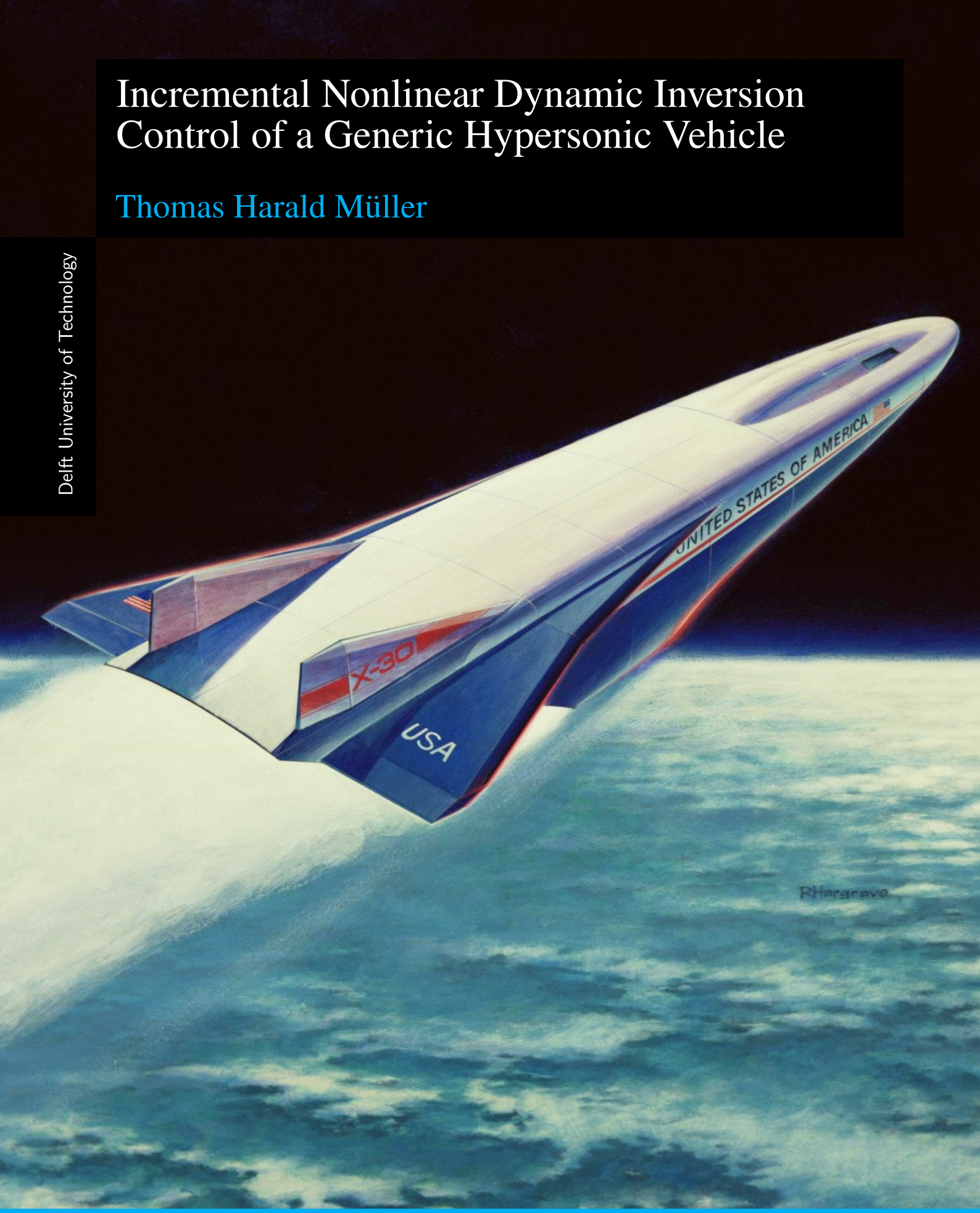
Incremental Nonlinear Dynamic Inversion Control of a Generic Hypersonic Vehicle

Thomas Harald Müller

Delft University of Technology



TU Delft Delft University of Technology

Challenge the future

Incremental Nonlinear Dynamic Inversion Control of a Generic Hypersonic Vehicle

by

Thomas Harald Müller

in partial fulfillment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on the 16th of June 2026.

Track:	Control & Operations (C&O)
Project duration:	October, 2024 – October, 2025

Supervisors:	Dr. Eng. Spilios Theodoulis
External:	Dr. Ioannis Sarras
Chairman:	Dr. Ir. Erik-jan van Kampen
Examiner:	Dr. Ir. Coen C. de Visser

Cover image: Rockwell X-30, from https://commons.wikimedia.org/wiki/File:X-30_NASP_3.jpg. Public domain. Accessed November 2025.

An electronic version of this document is available at https://repository.tudelft.nl/.

TU Delft Delft University of Technology

Abstract

The Generic Hypersonic Aerodynamics Model Example (GHAME) provides a practical benchmark for evaluating advanced control strategies for hypersonic vehicles. Its nonlinear dynamics and strong aero-propulsive coupling make it suitable for assessing nonlinear control methods. Although GHAME is the only publicly available hypersonic model derived from real flight data, Nonlinear Dynamic Inversion (NDI) has not previously been applied to it. This study develops a hierarchical control architecture based on time scale separation, combining NDI for attitude and position control with Incremental Nonlinear Dynamic Inversion (INDI) for angular-rate and velocity control. The implementation is carried out in MATLAB and Simulink, and the controller is tested under model uncertainty, atmospheric disturbances, and additionally tested under synchronized and desynchronized measurement time delays. The controller achieves accurate tracking when no delay is present. Under desynchronized delays, performance degrades rapidly and similarly in both axes. Under synchronized delays, degradation is more gradual: the lateral dynamics lose stability at roughly half of the allowable delay margin, whereas the longitudinal dynamics remain stable until the full margin is reached. This behavior reflects the very short natural time constants of the lateral subsystem in slender hypersonic configurations, which reduce the effectiveness of the assumed time scale separation. Overall, the study provides the first NDI application to the GHAME model and demonstrates that an NDI–INDI architecture can remain effective under uncertainty, disturbances, and realistic time delays when synchronization is maintained, although its application must be approached with caution due to the heightened sensitivity of the lateral dynamics. The findings naturally pave the way for future investigations, including flight envelope protection strategies, and integration with guidance and trajectory optimization methods for hypersonic missions.

Keywords: Nonlinear Dynamic Inversion, Incremental Nonlinear Dynamic Inversion, Hypersonic vehicle control, GHAME, Flight Dynamics, Robustness, Model uncertainty, MATLAB, Simulink.

Preface

To begin with, I want to express my deepest gratitude to my supervisor, Dr. Eng. Spilios Theodoulis, for his support and guidance throughout this work. I also wish to extend my heartfelt appreciation to my second supervisor, Dr. Ioannis Sarras, for his invaluable insights and uplifting encouragement. To both of you, thank you for always believing in me and for your patience in listening to me during every meeting. Your expertise and encouragement have been instrumental to this research. I am also deeply grateful to Johannes Autenrieb for the considerable time and effort he devoted to reviewing, discussing, and helping to improve my work.

Moreover, I am deeply grateful to my family. I want to thank my Mom and Dad, my sweet angel Luci, my granny, my sister Angela, and Cody for always being there for me and for their unwavering love and support throughout every step of this journey. To my friends in Delft, thank you for making me feel at home in the Netherlands and for being part of my everyday life. A very special thank you goes to Naishadha, the dearest and longest-lasting friend I made during my time at university. To my friends from the anime club Bahier, Baturay, Juan, Sasha, and Jelte, thank you for the time we spent together after uni and for making those moments genuinely fun, and especially for the amazing trip we took together to Japan. To the people from the aquarium, and especially Sahir and Bartosz, thank you for being there during lectures, gym sessions, and assignments, your presence made the work lighter and the long days easier to handle. To my friends from Hamburg, thank you for always making sure that I feel at home whenever I return.

Deutsche Version

Zu Beginn möchte ich meinen tiefsten Dank an meinen Betreuer, Dr. Spilios Theodoulis, aussprechen für seine Unterstützung und Begleitung während dieser Arbeit. Ebenso möchte ich Dr. Ioannis Sarras meinen herzlichen Dank für seine wertvollen Einsichten und seine aufmunternde Unterstützung aussprechen. Ihnen beiden danke ich dafür, dass Sie stets an mich geglaubt haben und mir in jedem Meeting mit Geduld zugehört haben. Ihre Expertise und Ermutigung waren von zentraler Bedeutung für diese Forschung. Mein besonderer Dank gilt außerdem Johannes Autenrieb für die beträchtliche Zeit und Mühe, die er in das Lesen, Diskutieren und Verbessern meiner Arbeit investiert hat.

Darüber hinaus gilt mein tiefster Dank meiner Familie. Ich möchte meiner Mutter und meinem Vater, meinem süßen Engel Luci, meiner Oma, meiner Schwester Angela und Cody danken, dass sie immer für mich da waren und mich auf jedem Schritt dieses Weges mit ihrer beständigen Liebe und Unterstützung begleitet haben. Meinen Freunden in Delft danke ich dafür, dass sie mir geholfen haben, mich in den Niederlanden zuhause zu fühlen, und dafür, dass sie Teil meines täglichen Lebens geworden sind. Ein ganz besonderer Dank gilt Naishadha, die liebste und langjährigste Freundin, die ich während meiner Zeit an der Universität gefunden habe. Meinen Freunden aus dem Anime-Club, Bahier, Baturay, Juan, Sasha und Jelte, danke ich für die gemeinsame Zeit nach der Uni und dafür, dass diese Momente wirklich Spaß gemacht haben, und ganz besonders für die großartige Reise, die wir zusammen nach Japan gemacht haben. Den Leuten aus dem Aquarium, und ganz besonders Sahir und Bartosz, möchte ich danken, dass sie während Vorlesungen, Gym Sessions und Hausarbeiten an meiner Seite waren; eure Anwesenheit hat die Arbeit leichter gemacht und die langen Tage erträglicher. Meinen Freunden aus Hamburg danke ich dafür, dass sie immer dafür sorgen, dass ich mich zuhause fühle, sobald ich zurückkomme.

Thomas Harald Müller
Delft, October 2025

Nomenclature

6DoF	Six-Degree-of-Freedom
ANDI	Actuator Nonlinear Dynamic Inversion
BIBS	Bounded-Input, Bounded-State
BLT	Bilinear Transform
CG	Center of Gravity
CGM	Classical Gain Margin
CM	Center of Mass
CT	Continuous-Time
CV	Control Variable
DCM	Direction Cosine Matrix
DGM	Disk Gain Margin
DT	Discrete-Time
EoM	Equations of Motion
FCC	Flight Control Computer
FCS	Flight Control System
FRD	Forward-Right-Down
GHAME	Generic Hypersonic Aerodynamic Model Example
HWM14	Horizontal Wind Model 14
IMU	Inertial Measurement Unit
INDI	Incremental Nonlinear Dynamic Inversion
INS	Inertial Navigation System
LQR	Linear-Quadratic Regulator
MAC	Mean Aerodynamic Chord
MIMO	Multiple-Input, Multiple-Output
NASA	National Aeronautics and Space Administration
NASP	National Aero-Space Plane
NDI	Nonlinear Dynamic Inversion
NED	North-East-Down
OBM	On-Board Model
PCH	Pseudo-Control Hedging
RMS	Root-Mean-Square
SBB	Sensor-Based Backstepping
SISO	Single-Input, Single-Output
SSTO	Single-Stage-To-Orbit
STOL	Short Take-Off and Landing
UAV	Unmanned Aerial Vehicle
UD	Unit Delay
USSA76	1976 U.S. Standard Atmosphere
WGS-84	World Geodetic System 1984
ZOH	Zero-Order Hold

Contents

1

Introduction

Hypersonic vehicles are at the forefront of future aerospace developments, both for space transportation concepts, such as Moon-to-Earth and orbit-to-Earth return, and for high-speed civil aviation envisioned as a successor to the Concorde [Hermeus, 2024]. While earlier hypersonic systems, including Single-Stage-To-Orbit (SSTO) concepts, re-entry capsules, and waveriders, have been studied extensively, modern air-breathing hypersonic vehicles introduce significantly greater complexity and must operate under more stringent requirements. These requirements extend beyond performance, with increasing emphasis on passenger comfort and safety.

Traditionally, flight control laws have been developed as linear controllers, with gain scheduling used to achieve the desired closed-loop behavior across the flight envelope. The gains are designed based on linearized models of the aircraft and its subsystems, including aerodynamics, actuators, sensors, and propulsion, at a range of operating conditions. For hypersonic vehicles, this approach becomes significantly more complex. Their dynamics differ fundamentally from those of conventional aircraft due to operation at extreme Mach numbers, where strong shock interactions, kinetic heating, and propulsion–airframe coupling introduce pronounced nonlinear and time-varying effects [McRuer, 1991]. In addition, these vehicles are subject to large uncertainties in aerodynamic coefficients, notable aeroelastic effects arising from slender flexible structures, and external disturbances such as gusts and turbulence [McRuer, 1991]. As a result, obtaining accurate models over the full flight envelope is challenging, and available aerodynamic data are often limited to specific operating points. Although linear control methods remain powerful, and modern approaches such as LPV can provide automated scheduling, they still rely on a collection of local models.

This motivates the exploration of nonlinear control strategies, which may offer the possibility of a single control law valid over a wide operating range and potentially reduce the need for complex gain scheduling. A widely studied nonlinear control strategy is Nonlinear Dynamic Inversion (NDI). Its objective is to transform the closed-loop input–output dynamics associated with the selected controlled variables into a chain of ideal integrators [Khalil, 2002]. This inversion is performed online through a combination of feedback and an On-Board Model (OBM) representation of the physical plant. This structure introduces a degree of modularity and transparency that is beneficial during control law development, as it largely decouples the airframe dynamics from the design of the flight qualities [Pollack, 2024]. Consequently, NDI-based design has become a widely used technique in aerospace applications and has demonstrated successful performance in numerous cases (e.g., [Harris and Stanford, 2018]).

However, this structure also introduces a key limitation in the hypersonic context. The effectiveness of NDI depends on the accuracy of the OBM used for inversion. In regimes with high modeling uncertainty and limited aerodynamic data, this requirement is difficult to satisfy.

This creates a mismatch between the need for nonlinear control and the availability of accurate models, which motivated the research for alternative approaches that reduce model dependence.

Incremental Nonlinear Dynamic Inversion (INDI) has emerged as one such approach [Sieberling et al., 2010]. In this formulation, a significant portion of the model is replaced by direct sensor feedback, reducing the reliance on high-fidelity aerodynamic models. Model information on the control effectiveness is still required for control allocation which may be complex to obtain but the overall dependence on the system model is reduced and it is assumed to have complete and accurate knowledge about the state of the system. Essentially, the burden has been changed from modeling to the availability of accurate sensors.

Furthermore, INDI can achieve improved low-frequency disturbance rejection compared to NDI at sufficiently high sampling rates, which is a notable advantage for hypersonic vehicles. However, this benefit comes at the cost of very limited delay margins in the sensor feedback path, making INDI significantly more sensitive to singular perturbations arising from unmodeled higher-order dynamics, such as actuator dynamics, sensor delays, and filtering effects. In particular, the improved low-frequency disturbance rejection is accompanied by increased sensitivity to high-frequency disturbances [Pollack and van Kampen, 2023]. As a result, low-pass filtering is typically required in practice, which introduces additional phase lag and further reduces the available delay margin [Sieberling et al., 2010].

Despite these limitations, the advantages of INDI make it a compelling candidate for hypersonic applications. Its reduced reliance on detailed modeling and its strong low-frequency disturbance rejection address two of the primary challenges in this regime. This motivates a systematic evaluation of INDI for air-breathing hypersonic vehicles, with particular attention to the interplay between sampling rate, actuator dynamics, and sensor-induced delays. Understanding whether the method can operate within practical limits, and to what extent its known weaknesses can be mitigated, is essential to assess its suitability for this class of vehicles.

An important reference for investigating such control strategies in a realistic setting is the Generic Hypersonic Aerodynamic Model Example (GHAME) vehicle, developed by the National Aeronautics and Space Administration (NASA) as a high-fidelity aerodynamic and propulsion dataset representative of air-breathing hypersonic flight up to Mach 24 [Zipfel, 2025]. The GHAME model has become a benchmark in hypersonic research, yet applications of nonlinear control methods to it remain limited compared to simpler analytical models. More recently, Goz [2024] implemented a six-degree-of-freedom flat-Earth, constant mass simulation of the GHAME vehicle, demonstrating how the dataset can be integrated into flight dynamics simulations and providing a basis for further control-oriented studies.

This thesis has two primary aims of equal importance. The first is to improve and extend the existing GHAME simulation framework introduced by Goz [2024]. The updated model features a rotating elliptical Earth, variable gravity, wind and turbulence models, a new atmospheric model valid up to an altitude of 1000 km, realistic fuel depletion, a variable center of gravity with corresponding changes in the moment of inertia, and second-order actuator dynamics. Together, these additions create a more complete and reliable simulation environment for investigating hypersonic flight dynamics. Verifying the model is a major part of the work, and the result is a simulation platform that can support future students and researchers working on hypersonic vehicle control.

The second aim is to design and evaluate an INDI control system using the improved GHAME model. The focus is on achieving robust stability and control performance in the presence of significant uncertainties and external disturbances. Particular attention is given to time delays, which constitute the fundamental limitation of INDI and therefore determine the achievable robustness margins. Through both analytical design and simulation-based testing, the thesis examines how INDI responds to these delays and assesses its strengths and limitations in this context. The two aims complement each other: the improved model provides the foundation for realistic control development, while the controller offers a meaningful demonstration of the model's capabilities.

2

Literature Review & Research Proposal

This chapter provides a detailed review of hypersonic flight control literature, with a focus on the nonlinear control methodologies of NDI and INDI. It begins by introducing hypersonic vehicles and their operational characteristics in Sec. 2.1. A discussion of the key technical challenges associated with controlling vehicles at hypersonic speeds is given in Sec. 2.2. Common flight-dynamics models used in hypersonic research are then presented in Sec. 2.3. Subsequent sections examine the historical development and current state-of-the-art applications of NDI and explores the emergence and recent advancements of INDI in Sec. 2.4, and explain how (I)NDI has been applied to hypersonic flight control Sec. 2.5. The literature study is summarized in Sec. 2.6 to identify the research gaps that motivate this work. These gaps form the foundation of the research proposal, from which the overall objective and corresponding research questions are formulated in Sec. 2.7.

2.1. Overview of Hypersonic Vehicles

Hypersonic vehicles are commonly classified by propulsion into three categories: rocket-propelled, boost–glide, and air-breathing systems [Besser et al., 2017]. Rocket-propelled vehicles carry both fuel and oxidizer, enabling very high speeds and altitudes. Examples include the X-15, which reached Mach 6.70, as well as the Space Shuttle and various missile systems [Mellinger, 1960]. Boost–glide vehicles use a solid rocket booster for ascent, then reenter the atmosphere and glide unpowered at hypersonic speeds, as seen in systems such as the U.S. Advanced Hypersonic Weapon, Russia's Yu–71, and China's DF–ZF [Besser et al., 2017]. While rocket-propelled and boost–glide systems are relatively mature, air-breathing configurations remain less developed and are the main focus of current research. Air-breathing hypersonic vehicles employ engines that ingest atmospheric air for combustion, eliminating the need to carry onboard oxidizers. This approach reduces vehicle mass and offers the potential for greater range and payload capacity compared to rocket propulsion. However, the associated challenges of sustaining stable combustion, managing extreme thermal loads, and maintaining efficient operation at hypersonic speeds have made air-breathing systems the least technologically mature class of hypersonic vehicles [Bruno, 2023].

(a) North American X-15 by NASA, licensed under PDM 1.0.

(b) AQM-60 "Kingfisher" by USAF, licensed under PDM 1.0.

Figure 2.1: The left image shows North American Aviation's X–15, while the right image shows the Lockheed AQM-60 Kingfisher drone.

The development of air-breathing hypersonic vehicles dates back to the 1940s with the emergence of the ramjet, which compresses incoming air through forward motion rather than rotating compressors. Combustion occurs in a subsonic chamber, producing thrust through exhaust acceleration. Research accelerated in the 1950s with the Mach 4.3 ramjet developed under the Lockheed X-7 program, later used in the AQM-60 Kingfisher drone between 1951 and 1959 [Jenkins et al., 2003]. The Kingfisher ramjet was subsequently adapted for the Lockheed D-21 reconnaissance drone, which completed four high-speed surveillance missions over China. These developments demonstrated sustained air-breathing propulsion at high supersonic speeds and laid the groundwork for later hypersonic research [Donald, 2002].

Figure 2.2: Lockheed SR-71 Blackbird by USAF / Judson Brohmer, licensed under CC BY-SA 4.0.

During the 1960s and 1970s, the limitations of ramjet propulsion became apparent. Ramjets cannot operate from rest and require a minimum flight speed of about Mach 2, necessitating auxiliary propulsion or external acceleration [Falempin, 2008]. This led to the development of the Pratt & Whitney J58, a hybrid turbojet–ramjet engine used in the SR-71 Blackbird. The J58 operated as a turbojet with afterburner at lower speeds and gradually transitioned toward ramjet-like behavior at higher Mach numbers. The SR-71 remains the fastest manned air-breathing aircraft, reaching Mach 3.2 in 1976 [Donald, 2004].

In the 1980s and 1990s attention shifted to another drawback of ramjet technology: its inability to operate efficiently at higher Mach numbers. Because a ramjet decelerates incoming supersonic air to subsonic speeds before combustion, excessive heating occurs at flight speeds above Mach 5 [Bruno, 2023]. This limitation motivated the development of the supersonic combustion ramjet, or scramjet, in which combustion occurs with supersonic airflow through the engine. The scramjet alleviates the thermal restrictions of the ramjet and enables sustained propulsion at far higher velocities. The Hypersonic Research Engine project by NASA was instrumental during this period, demonstrating key principles of hypersonic combustion in ground tests, while DARPA's Copper Canyon Program simultaneously addressed the integration of scramjets into vehicle designs and explored solutions for challenges such as supersonic ignition stability and thermal management [Bruno, 2023]. These advancements established scramjets as a viable successor to ramjets, unlocking the potential for sustained flight at speeds beyond Mach 5. The performance advantage of scramjets over other air-breathing and rocket systems is illustrated in Fig. 2.3, which compares the specific impulse of different propulsion concepts as a function of Mach number. Hydrogen emerged as the preferred scramjet fuel, offering superior specific impulse, efficient combustor cooling due to its cryogenic properties, and lower emissions relative to hydrocarbon fuels. Its rapid dispersion in the event of leaks or crashes also provides safety benefits for crewed vehicles, and it has been employed in experimental hypersonic programs such as NASA's X-43A and X-43D [Moses et al., 2004].

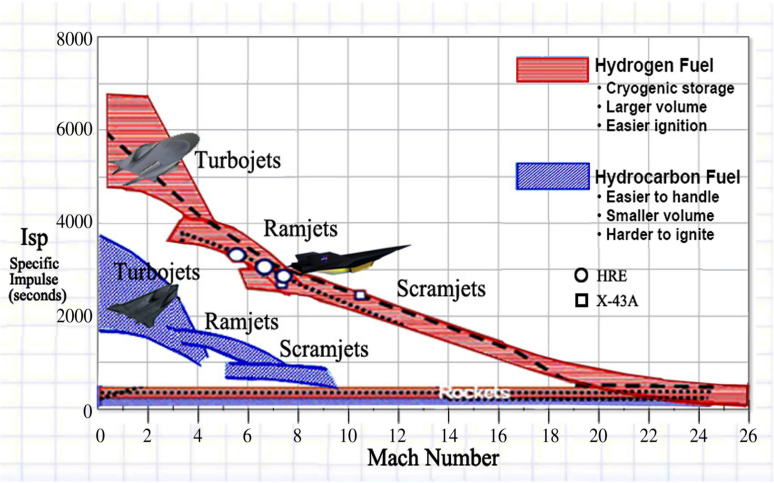


Figure 2.3: Specific impulse of different propulsion concepts as a function of Mach number, comparing turbojets, ramjets, scramjets, and rockets for hydrogen and hydrocarbon fuels [Bruno, 2023].

The 2000s marked key breakthroughs in scramjet technology. The first successful in-flight demonstration is generally attributed to the HyShot II program in 2002, which achieved a supersonic-combustion window of a few seconds during a ballistic re-entry trajectory [Bruno, 2023]. NASA's X-43A followed in 2004, reaching Mach 9.6 with a scramjet burn of roughly ten seconds [NASA, 2022]. In 2013, the X-51 Waverider demonstrated sustained scramjet operation at Mach 5.1 for 210 seconds until fuel exhaustion, the longest air-breathing hypersonic flight to date and a demonstration that longer-duration flight is feasible [Force, 2011].

In the 2010s, attention shifted to combined-cycle engines, which integrate a turbine with a dual-mode scramjet to cover a wider flight envelope [Thomas et al., 2009]. While the concept builds on earlier variable-cycle propulsion, such as the Pratt & Whitney J58, modern designs extend this idea into the hypersonic regime. By enabling operation from takeoff to beyond Mach 5 without external acceleration, these systems represent an important step toward practical hypersonic flight.

(a) NASA X-43A "Hyper-X" by Navneet Yadav, licensed under CC BY 2.0.

(b) Boeing X-51A "Waverider" by Mark Jones Jr., licensed under CC BY 2.0.

Figure 2.4: Side–by–side comparison of hypersonic vehicles that were successfully flight tested. The left image shows an artist's impression of NASA's X–43A, while the right image shows the X–51A Waverider mounted beneath its B–52 carrier aircraft.

In the 2020s, significant progress has been made in the development of air–breathing hypersonic vehicles for both military and civilian applications. On the defense side, Lockheed Martin's SR–72 is being designed as a reconnaissance and strike aircraft capable of Mach 6 flight using a combined–cycle engine [Hollings, 2024]. In parallel, commercial ventures are advancing reusable demonstrators. Hermeus is developing its Darkhorse hypersonic aircraft through the incremental Quarterhorse test series, which is progressively building toward sustained Mach 5+ flight, while Stratolaunch has flown its rocket–powered Talon–A vehicle at hypersonic speeds [Hermeus, 2024]. These initiatives illustrate the growing dual–use potential of hypersonic technology. To the author's knowledge, an overview of the most prominent publicly known vehicle concepts currently under development is provided in Table 2.1, based on open-source information.

Table 2.1: Overview of modern supersonic/hypersonic concept vehicles, based on open-source information.

Vehicle	**Origin**	**First flight**	**Max speed**	**Payload**
Lockheed SR-72	USA	~2030	Mach 6	No data
Venus Stargazer M4	USA	~2030	Mach 4	No data
Dawn Mk-II Aurora	NL/NZ	In flight test	Mach 3.7	Up to 15 kg
BrahMos-II	India/Russia	2028	Mach 6	300 kg
Hermeus Darkhorse	USA	2030	Mach 5+	No data
Stratolaunch Talon-A2	USA	2025	Mach 5+	Up to 450 kg
Invictus	ESA/UK	2031	Mach 5	No data
ASN4G	France	2035	Mach 5+	300 kg
HACM	USA/Australia	2027	Mach 8	Warhead (missile)

(a) The Stratolaunch Talon-A2 by Eva Folsom, licensed under CC BY-SA 4.0.

(b) Hermeus Darkhorse by Hermeus Corp. licensed under CC BY-SA 2.0.

2.2. Control Challenges

At hypersonic speeds, typically above Mach 5, vehicles face severe thermal, aerodynamic, and propulsion challenges that drive the need for specialized designs. Kinetic heating from air compression at the leading edges can produce stagnation temperatures above 1,800 K, which places strict limits on materials and internal systems. At the same time, aerodynamic loads become extreme, with surface pressures reaching several tons per square meter at 20 km altitude, leading to significant structural demands [Besser et al., 2017]. These combined effects require careful structural design to maintain integrity and performance. To cope with these conditions, hypersonic vehicles use highly streamlined shapes. A slender fuselage reduces frontal area, lowering drag and spreading thermal loads more evenly. It also helps manage shock interactions and stabilize the flow [Besser et al., 2017]. Large wings become impractical due to the associated thermal and structural penalties, so designs typically rely on lifting-body or compression-lift concepts, where the fuselage generates most of the lift. This reduces wing size, structural mass, drag, and surface heating while maintaining aerodynamic efficiency.

The high slenderness ratio and lightweight construction result in closely spaced elastic and rigid-body mode frequencies, with a separation of less than one decade [Gilbert et al., 1990]. These closely spaced modes produce significant aeroelastic interactions with the rigid-body response [Schmidt and Velapoldi, 1999]. This effect is increased by the nonuniform aerodynamic heating that reduces structural stiffness and introduces variability in vibration frequencies and mode shapes [Schmidt and Velapoldi, 1999].

Propulsion is tightly integrated into the airframe in hypersonic vehicle design. Ramjets and scramjets rely on the vehicle's forward motion to compress incoming air for combustion. To enhance this effect, the engines are placed along the underside, where the bow shock generated by the forebody provides additional precompression of the inflow, increasing inlet pressure and thrust. This configuration aligns the propulsion system with the aerodynamic shape, helping preserve shock structures and avoid adverse interactions. The aftbody is shaped to promote expansion of the exhaust flow, effectively acting as an extension of the nozzle. As a result, the lower surface of the vehicle contributes directly to thrust generation and becomes an integral part of the propulsion system [McRuer, 1991].

The integrated aerodynamic–propulsion design of hypersonic vehicles makes control particularly challenging, as aerodynamic, propulsion, and structural dynamics are strongly coupled and highly configuration dependent. The high slenderness ratio required for aerodynamic efficiency leads to highly nonlinear aerodynamics and varying stability characteristics [Schmidt and Velapoldi, 1999]. Studies on the X-30 showed that static stability margins decrease with increasing Mach number, along with significant changes in directional stability and dihedral effect [Penland et al., 1978]. Maintaining acceptable stability at hypersonic speeds requires larger margins at lower speeds, which increases control deflections and reduces maneuverability [Schmidt and Velapoldi, 1999].

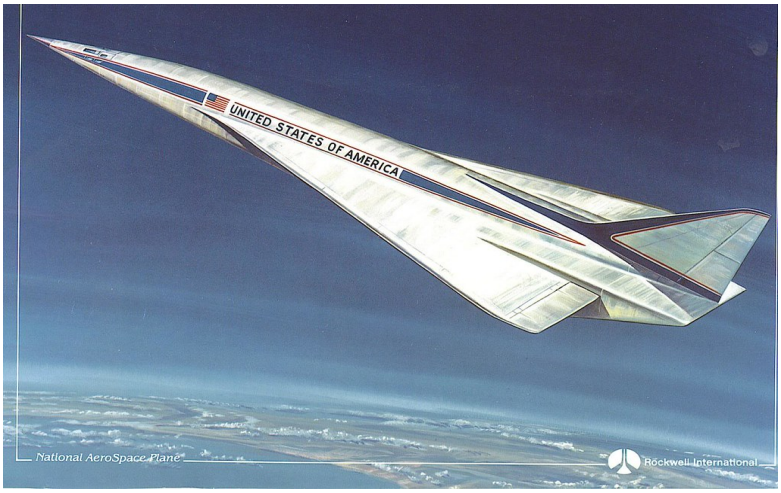


Figure 2.6: Rockwell X-30 "National AeroSpace Plane" by Jim Phillips, licensed under CC BY-SA 4.0.

At high angles of attack, the center of pressure shifts aft, reducing rudder effectiveness and complicating pitch trim, while directional stability may be lost in the yaw and roll planes [Schmidt and Velapoldi, 1999]. To manage pitch forces, the center of mass must lie between the extreme positions of the center of pressure, resulting in positive stability during descent and negative stability during ascent. Non-conventional control surfaces on the windward side are sometimes used to improve directional stability under these conditions [Schmidt and Velapoldi, 1999].

Moreover, the integration of scramjets into the forebody and underside of the vehicle creates bidirectional coupling between propulsion and aerodynamic forces [McRuer, 1991]. Increased forebody pressure during airflow compression generates lift and induces a nose-up pitching moment, while the external nozzle produces a counteracting nose-down moment [McRuer, 1991]. The propulsion system is therefore not only a source of thrust, but also contributes to lift and moments, directly affecting the thrust–drag and lift–moment balance. As a result, it becomes a major contributor to control authority alongside the aerodynamic effectors and has a strong influence on airframe stability [McRuer, 1991]. Additionally, the efficiency of scramjets depends on the capture of incoming mass flow, which is governed by the bow shock. The position and strength of the bow shock are highly sensitive to altitude, flight speed, and angle of attack [Schmidt et al., 1991].

A critical propulsion-related challenge with direct control implications is inlet unstart, a phenomenon in which the shock system within the scramjet isolator propagates upstream and exits the inlet. This event disrupts the internal flow to the combustor, drastically reducing total pressure recovery and leading to an immediate loss of thrust [Hall and Poggie, 2019]. At hypersonic speeds, scramjets operate with small net thrust margins, with gross thrust exceeding net thrust by an order of magnitude due to the substantial aerodynamic drag. As a result, even a modest reduction in thrust can rapidly destabilize flight and result in loss of control [Johnson et al., 1987]. Preventing unstart therefore becomes a control objective rather than solely a design constraint, and requires coordinated management of inlet geometry, fuel flow, and vehicle attitude to maintain a stable shock system and continuous combustion. An example of this phenomenon is illustrated in Fig. 2.7, where the unstarted flow condition (left) is compared to the stable started condition (right).

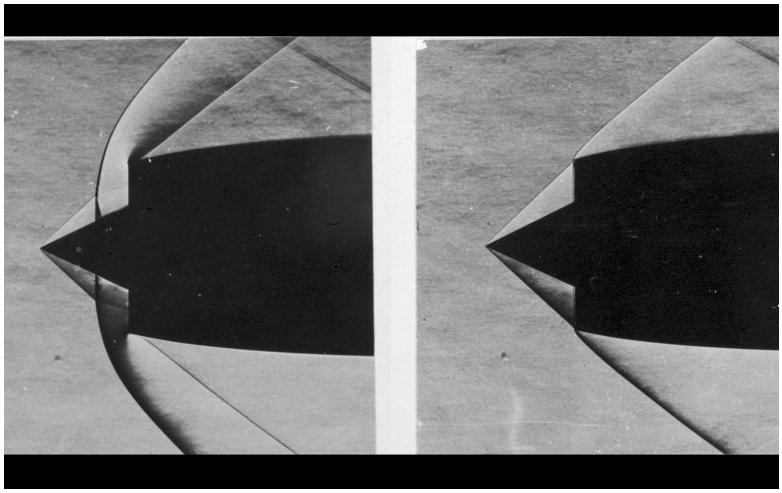


Figure 2.7: Schlieren visualization of shock wave structures in a supersonic axisymmetric cone inlet. The left image shows the unstarted flow condition, where the shock system is expelled upstream, while the right image shows the started flow condition with stable shock containment inside the inlet. Image from NASA document ID: C-1955-37520.

2.3. Public Hypersonic Flight Models

In 1989, NASA introduced the first publicly available mathematical model of a hypersonic aircraft, known as the GHAME [Bowers et al., 1989]. Developed at NASA Ames Research Center during the early phases of the National Aero-Space Plane (NASP) initiative, the GHAME model was intended to provide universities and U.S. government agencies with an unclassified, high-fidelity aerodynamic and geometric reference vehicle for research and simulation. Conceptually, it was based on the configuration proposed for the X-30 of the NASP program, which remained a conceptual design and was never constructed. As a result, the GHAME dataset combines flight-test data from the Space Shuttle and the X-24C with theoretical predictions derived from a swept double-delta configuration and a six-degree half-angle cone using the modified Newtonian impact-flow method. Covering flight conditions from Mach 0 to 25 and incorporating a turbo-ram-scramjet propulsion system, the GHAME model was released without further analysis to serve as a generic, physically representative benchmark for hypersonic research. Detailed aerodynamic and propulsion data are provided in White et al. [1992], while its implementation in simulation frameworks is described in Zipfel [2025].

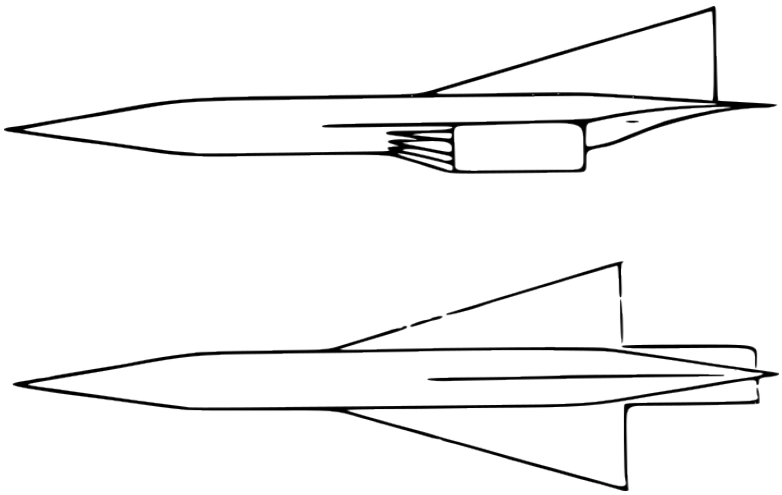


Figure 2.8: NASA's GHAME model, as shown in White et al. [1992].

The GHAME model was primarily employed at NASA's Dryden Flight Research Center (renamed to NASA Armstrong Flight Research Center in 2014) to evaluate the handling qualities of proposed controllers, with test pilots conducting simulations in flight simulators [Araki, 1992, Sachs, 1998, Sachs and Moravszki, 2006, Stich and Sachs, 1998, Vu and Biezad, 1993a,b]. Outside of NASA, research involving the GHAME model has been relatively sparse. Notable exceptions include the work of O. J. Murillo et al. in 2010, who used the model to test fast ascent trajectory optimization strategies [Murillo and Lu, 2010], and more recent contributions by Goz [2024], who developed a robust control framework for the GHAME model using modern $\mathcal{H}_\infty$ techniques.

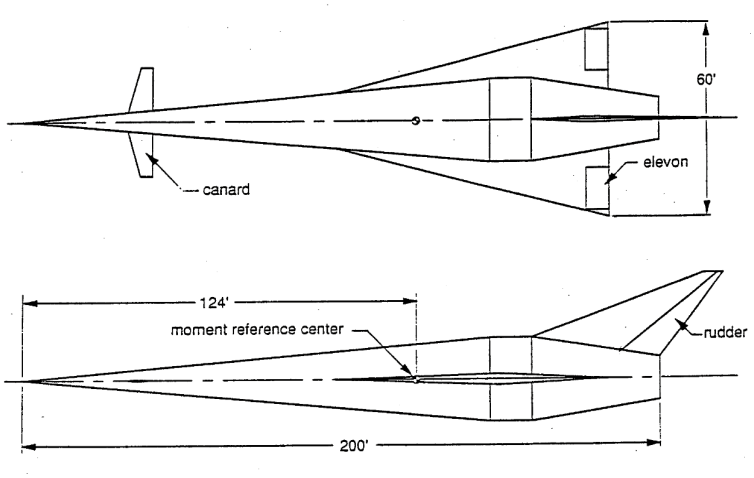


Figure 2.9: NASA's winged cone model, as shown in Shaughnessy et al. [1990]

In 1990, NASA introduced a second hypersonic aircraft model known as the Hypersonic Vehicle Simulation Model: Winged-Configuration. This model was also developed as part of the NASP program and can be considered a counterpart to the GHAME model, with its development taking place at the Langley Research Center. While the two models share many similarities, a key distinction lies in the inclusion of a canard in the winged-cone model. Additionally, unlike GHAME, which relied on mission data from various vehicles to estimate aerodynamic and force coefficients, the winged-cone model used subsonic/supersonic panel methods and hypersonic local surface inclination codes to obtain aerodynamic data [Shaughnessy et al., 1990]. The winged cone model garnered significantly more attention in research with far over 300 citations for the informal reason that the NASP program was significantly better funded and attracted more interest.

As part of the NASP program, further research focused on understanding the airframe–engine interactions from an integrated control perspective to develop improved hypersonic vehicle models McRuer [1991], Schmidt [1992], Schmidt et al. [1991]. All of this effort culminated in Chavez and Schmidt [1994] developing the first comprehensive analytical model of the longitudinal dynamics of a hypersonic vehicle, which incorporated both aeropropulsive and aeroelastic effects. In this model, Newtonian impact theory was employed to derive pressure distributions as functions of Mach number, freestream pressure, angle of attack, and geometry. These distributions were then used to compute the total aerodynamic forces, which were subsequently linearized to obtain analytical stability and control derivatives. The representative vehicle geometry used in this formulation is shown in Fig. 2.10.

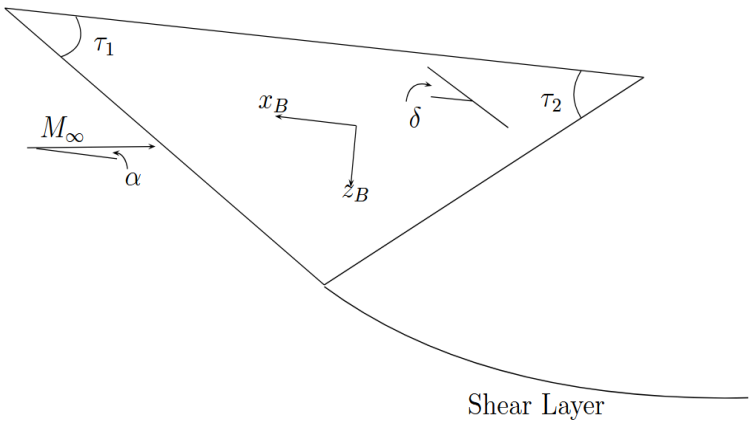


Figure 2.10: Hypersonic vehicle geometry used by Chavez and Schmidt [1994]. This early NASP–era model used Newtonian impact theory to approximate aerodynamic loads, offering a simple yet practical baseline for studying aeropropulsive coupling and longitudinal dynamics.

In 2005, Bolender and Doman [2005] extended the work by Chavez and Schmidt with several key modifications. First, oblique–shock and Prandtl–Meyer theory replaced Newtonian impact theory for calculating surface pressure, yielding more accurate aerodynamics. Second, their model introduced a translating cowl door to maintain a shock–on–lip condition during off–design flight, enhancing propulsion modeling fidelity. Third, and most significantly, Bolender employed a distributed double–cantilever beam model for structural dynamics in place of the lumped–mass model used by Chavez. This allowed the model to capture an unstable aeroelastic mode near 202 rad/s, which was absent in the earlier formulation. Although both models produce similar short–period dynamics, differences in the phugoid and altitude modes emerge due to variations in aerodynamic modeling, and the added aeroelastic mode in Bolender's work reflects the higher structural fidelity made possible by modeling a structure not constrained at any interior point. The vehicle geometry used for this study is shown in Fig. 2.11.

In subsequent studies, this configuration is often augmented with a canard in addition to the elevator. The canard was introduced to address controllability limitations and to mitigate non-minimum phase behavior that can arise when only the elevator is used as the primary longitudinal control effector. However, the concept of canards on hypersonic aircraft has largely fallen out of favor, as their forward placement exposes them to extreme thermal loads from aerodynamic heating, making their practical implementation highly questionable.

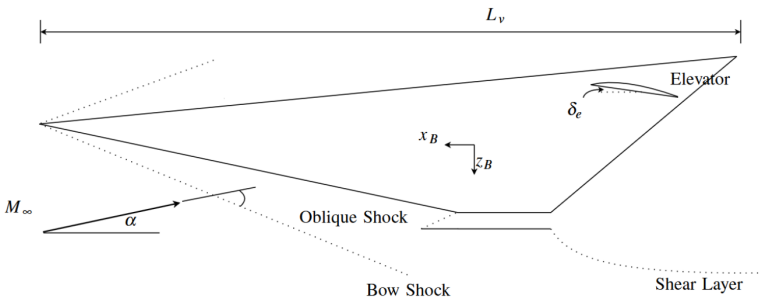


Figure 2.11: Hypersonic vehicle geometry used by Bolender and Doman [2005]. Compared to the earlier Chavez–Schmidt model, Bolender's configuration included a translating cowl door and refined structural representation, enabling improved aerodynamic and aeroelastic fidelity.

2.4. (I)NDI in General Flight Control

Nonlinear Dynamic Inversion (NDI) is the term commonly used in the aerospace community for feedback linearization [Enns et al., 1994]. More specifically, NDI is a form of feedback linearization that renders the input–output dynamics of the selected controlled variables linear through an inversion step applied to the corresponding input–output mapping of the system. It is applied to control-affine systems, where the outputs are differentiated until the inputs appear explicitly. If the system has a well-defined and constant relative degree, and if the associated decoupling matrix is nonsingular in the region of operation, state feedback can be used to transform the input–output dynamics into a chain of integrators [Khalil, 2002]. Once the input–output dynamics have been reduced to a set of integrator chains, a virtual control input is introduced to assign the desired closed-loop behavior, typically using linear control design. This formulation provides a relatively simple conceptual framework and introduces a level of modularity and transparency in the control law. NDI separates the flight qualities design from the airframe-dependent dynamics.

In practice, however, the inversion is computed online using an on-board model (OBM) representation of the airframe dynamics. Therefore, the performance of NDI depends critically on the availability of an accurate OBM representation of both the drift dynamics and the control effectiveness. Modeling errors or unmodeled dynamics lead to imperfect inversion, causing the actual closed-loop system to deviate from the ideal linear design. Consequently, this approach does not provide robustness guarantees with respect to modeling uncertainties or disturbances, and is therefore often combined with robust control techniques to achieve improved robustness [Balas et al., 1992].

Figure 2.12: NASA Buffalo Augmentor Wing Jet STOL Research Aircraft by NASA, licensed under Public Domain Mark 1.0.

The origins of NDI are often traced back to Meyer and Cicolani [1975], who, in their work on augmentor wing jet Short Take-Off and Landing (STOL) aircraft, claimed that the prevailing divide-and-conquer design strategy suffers from tractability issues as nonlinear effects become more pronounced. They also noted that it was not evident how reference trajectories should be selected for the transition and landing phases. To address these issues, they proposed a unified flight control methodology based on physical insights from flight dynamics, intended to provide a more general control structure for highly nonlinear flight regimes. They propose a flight control methodology featuring a trim map. The trim map calculates control settings for commanded forces and if the trim map accurately represents the vehicle's characteristics, the calculated control settings will generate trajectories that closely match the commanded ones. When the trim map is used in the control loop, the trim map and the aircraft together behave almost like an identity system, leading to nearly linear operational characteristics.

NDI was first demonstrated in flight in 1980 on both a STOL aircraft and a DHC-6 Twin Otter, confirming its practical viability for real-time control of nonlinear systems [Jr. and Meyer, 1980, Meyer and Cicolani, 1980]. The following year, Hunt and Su [1981] provided the necessary and sufficient conditions under which a class of time-varying nonlinear systems could be transformed into time-invariant, controllable linear systems through state feedback. In 1982, Su extended this framework to general multi-input systems, building on foundational work by Krener [1973], Brockett [1978], and Jakubczyk and Respondek [1980] on nonlinear system transformations. Around the same time, Meyer et al. [1984] applied these concepts to helicopter flight control, demonstrating their applicability beyond fixed-wing aircraft. These early theoretical and practical developments laid the foundation for more advanced formulations of NDI suited to the control of underactuated and multivariable systems.

In the late 1980s and early 1990s, researchers at the Honeywell Technology Center in Minneapolis made several key contributions to this effort. Dynamic inversion requires that the system have at least as many control inputs as states. This condition is often not satisfied in conventional aircraft, where the typical four control inputs (thrust, aileron, rudder, and elevator) are insufficient to control all six degrees of freedom. To address this, Elgersma [1988] developed the concept of partial dynamic inversion. This method focuses on directly controlling a subset of the system states, while the remaining states, known as zero dynamics or complementary dynamics, must be analyzed to ensure stability [Isidori, 1985]. Building on this, the same group at Honeywell including Elgersma [1988], Bugajski et al. [1990], Bugajski and Enns [1992], and Snell et al. [1992] introduced a time-scale separation framework for NDI [Chow and Kokotovic, 1978]. Their approach partitioned the system dynamics into fast and slow subsystems: the fast states, comprising the angular rates p, q, and r, were controlled using aerodynamic surfaces, while the slow states, such as the angle of attack α, sideslip angle β, and bank angle μ, were managed via an approximate inversion using the commanded angular rates as intermediate inputs. This decomposition relies on the assumption that the open-loop time scales of the fast and slow dynamics differ significantly, allowing for a hierarchical control architecture.

Figure 2.13: NASA X-38 by NASA, licensed under Public Domain Mark 1.0.

By the early 1990s, these developments positioned the Honeywell Technology Center as a central contributor to the maturation of NDI. In 1996, a joint effort involving Honeywell, Lockheed Martin Tactical Aircraft Systems, and Lockheed Martin Skunk Works produced a comprehensive set of design guidelines for multivariable flight control systems, including NDI-based architectures [Reiner et al., 1996]. In the early 2000s, Honeywell developed the MACH (Multi-Application Control for Hypersonics) design environment, a proprietary tool for systematically implementing NDI controllers in aerospace applications [Balas, 2003], which was applied to design the flight control system of the X-38. The

objective was to shorten the design cycle time for varying vehicle shapes, flight envelopes, and evolving aerodynamic databases [Wacker et al., 2001]. This methodology was later integrated into the STOVL variant of the X-35 demonstrator as part of the Joint Strike Fighter program, and eventually adopted in the production configuration of the Lockheed Martin F-35 Lightning II [Harris and Stanford, 2018].

Figure 2.14: Lockheed Martin F-35 "Lightning II" by Robert Sullivan, licensed under Public Domain Mark 1.0.

NDI might be a mature control technology now but it is far from perfect. Already in the 1990s, Enns et al. [1994] identified robustness as a critical challenge, particularly when the nominal aircraft model is replaced with a perturbed model or when variations in aerodynamic coefficients occur. Consequently, improving the robustness of NDI became a central focus in subsequent research. In 1992, Balas et al. [1992] introduced robust dynamic inversion control laws that combined μ-synthesis with dynamic inversion. In 1996, they extended this approach by incorporating time-scale separation for angle-of-attack control [Reiner et al., 1996]. Meanwhile, Snell [1992] pursued a different approach. He demonstrated that performance under aerodynamic coefficient perturbations could be improved by integrating terms within the control loops to eliminate steady-state errors. Initially, he applied this method to an exact aircraft model and later, in collaboration with Stout, extended it to a perturbed model [Snell and Stout, 1994]. Smit and Craig [1998] combined dynamic inversion with H_∞ loop-shaping, and Hyde and Papageorgiou [2001] analyzed the stability of NDI-based controllers using Linear Parameter-Varying methods, addressing stability concerns critical for robust operation. A good summary of robust methods was presented by Kolesnikov [2005].

As these studies highlighted the inherent model dependence of NDI and its sensitivity to uncertainties, efforts were made to reduce its reliance on high-fidelity models. This led to the development of Incremental Nonlinear Dynamic Inversion (INDI) Sieberling et al. [2010]. This approach reduces the required OBM representation by using direct sensor feedback of the control variable derivatives. Consequently, the reliance on model information is reduced, although not entirely eliminated. The remaining model dependence is the control effectiveness required for control allocation. The relative ease of implementation and the ability to maintain consistent flying qualities in the presence of linear and nonlinear variations in the airframe aerodynamics have contributed to the widespread interest INDI. However, compared to NDI, studies have shown that INDI exhibits a relatively limited stability robustness margin when subjected to unmodeled dynamics affecting the sensor feedback path [Pollack and van Kampen, 2023].

Figure 2.15: VAAC Harrier by Tony Hisgett, licensed under CC BY 2.0.

The INDI concept was likely first introduced in 1998, when a "simplified NDI" control approach for input–affine system dynamics was proposed and applied to the rate control of a simulation model of the McDonnell Douglas F/A-18 [Smith, 1998]. The resulting inversion law eliminated the need for state derivatives by expressing the dynamics in terms of measured signals, which structurally matches what later became known as INDI control laws. This approach was subsequently validated for a rate demand loop on the VAAC Harrier in flight tests with successful results [Smith and Berry, 2000]. In this work, the controllers were found to independent of aerodynamic coefficient and exhibit robustness under parametric variations of the inertias and control power coefficients. At the same time, it was observed that conditioning noisy feedback signals using filters could have an adverse effect on the closed-loop response. Rotational acceleration sensors were available, but the measurements were affected by noise generated by the aircraft structure and engines, which required filtering. Phase differences introduced by this filtering were found to result in oscillations in pitch rate following a step or disturbance, with amplitude, damping, and frequency depending on the magnitude of the phase shift. They highlight the sensitivity of the inversion loop to phase differences introduced in the signal path.

Building on this work, Bacon and Ostroff [2000] presented a more general formulation of the incremental inversion law, referred to as "modified NDI." Using a Taylor series approximation, the derivation was extended to a broader class of systems, including those with input non-affine control inputs. This formulation has since become the most commonly used approach for deriving INDI control laws. In their initial implementation, Bacon and Ostroff [2000] proposed a setup with multiple accelerometers to measure angular acceleration directly. However, this approach introduced practical challenges due to its complexity and the sensitivity of accelerometers to structural vibrations. To address this, Bacon et al. [2001] proposed estimating angular acceleration from gyroscope measurements. To mitigate the noise in these measurements, a second-order low-pass filter combined with discrete differentiation was introduced. This approach enables accurate estimation of the derivative at low frequencies while attenuating high-frequency sensor noise. However, the second-order filter differentiator introduces lag, resulting in a delayed estimate of the output derivative, which led to what they described as jitter in the control variable responses, similar to the oscillations previously reported by Smith and Berry [2000]. To mitigate this effect, Ostroff and Bacon [2002] proposed the use of a compensator placed around each actuator. It was found that a simple lag filter with a bandwidth similar to that of the washout filter provided the most effective reduction of jitter. Furthermore, this allowed the bandwidth of the washout filters used to generate angular accelerations to be reduced further while preserving the decoupling properties of the original design, resulting in improved control variable responses in the presence of sensor noise. This approach was referred to as "enhanced NDI."

In 2010, Sieberling et al. [2010] revisited the work of Bacon and Ostroff [2000] and, in doing so, introduced the term "Incremental Nonlinear Dynamic Inversion," arguing that the use of increments more accurately reflects the nature of the method than describing it as a simplification of NDI. Their work focused on two main aspects: the robustness of INDI to model uncertainty and the treatment of measurement delays. The robustness analysis was carried out using linear transfer functions derived from block diagrams, leading to the claim that, under sufficient time-scale separation and in the presence of ideal sensors and actuators, the linearizing control law obtained from INDI remains linearizing in the presence of model uncertainty, unlike NDI, where modeling errors directly affect the inversion [Juliana et al., 2004]. At the same time, the authors emphasized that ideal sensors do not exist and investigated in detail the effect of measurement delays arising from sensor dynamics or filtering, which had previously been observed by Smith and Berry [2000] and Bacon et al. [2001]. It was shown that such delays introduce an inversion residual, implying that in the presence of measurement delays the INDI control law no longer achieves exact linearization. To address this issue, a method was proposed that avoids delay by estimating angular acceleration through predictive filtering based on angular rate measurements and their reference trajectories. While this approach increased the performance of INDI in the presence of measurement delays, it requires additional modeling to predict the angular accelerations.

Subsequent research extended INDI to new application. Acquatella et al. [2012] was the first to successfully apply the method to spacecraft attitude tracking and disturbance rejection in the presence of parametric uncertainties. Simplício et al. [2013] applied INDI successfully to helicopter flight control and combined it with pseudo-control hedging (PCH) [Johnson and Calise, 2000]. PCH is used to account for actuator dynamics and limitations that are not included in the INDI formulation. Since INDI assumes that the commanded pseudo-control is always realized instantenously, actuator bandwidth limits and saturation can lead to a mismatch between commanded and achieved control. PCH compensates for this by feeding back the portion of the pseudo-control that is not achieved, thereby preventing the controller from reacting to unrealizable dynamics. In addition, angular accelerations were estimated using backward finite differences, while first-order low-pass filters were introduced before the actuators to attenuate high-frequency oscillations caused by numerical noise. The delay introduced by the filters degraded performance as no explicit synchronization between the measurement and actuator paths was applied. Acquatella et al. [2013] and Koschorke et al. [2013] combined incremental action with backstepping, demonstrating superior robustness under aerodynamic uncertainty compared to standard backstepping methods [Khalil, 2002].

The next major development occurred when researchers at the German Aerospace Center, Delft University of Technology, and the University of Minnesota addressed the challenges associated with measurement delays through signal synchronization. This was successfully demonstrated in a series of flight tests with the fixed-wing FASER UAV in 2013, although the concept was formally published only a few years later by Smeur et al. [2016a]. In this work, a synchronization strategy is proposed in which the actuator signal is passed through a matching filter, with a delay chosen such that the actuator response is aligned in time with the filtered estimate of the angular acceleration. This ensures consistent timing between the signals used in the incremental inversion loop and was validated in flight experiments using a Parrot Bebop quadrotor UAV [Smeur et al., 2016b].

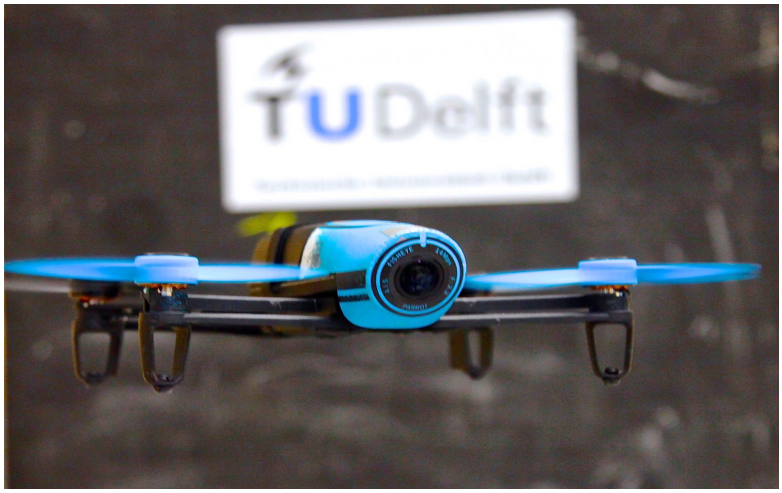


Figure 2.16: Parrot Bebop with credit to TU Delft.

Their approach can be interpreted as a formalization of a solution already reported by Ostroff and Bacon [2002], where a lag compensator around the actuator was used to reduce jitter caused by filtering effects, albeit without explicitly framing it as a synchronization problem. The method was later validated through flight tests on a PH-LAB Cessna Citation II by Grondman et al. [2018]. In van't Veld et al. [2018], the synchronization aspect was further investigated empirically, showing that INDI is more sensitive to delays in the output derivative feedback than to actuator delays.

Other approaches have been proposed to address delays in the measurement feedback path. de Angelis Cordeiro et al. [2021] introduced an input scaling gain to improve robustness to feedback delays when using a second-order filter differentiator. Kumtepe et al. [2022] proposed a complementary filtering approach, where the measured rate is low-pass filtered and differentiated, while the high-frequency component is obtained from a high-pass filtered model-based acceleration estimate. Furthermore, Liu et al. [2022] designed a non-delay differentiator based on a precise-delay differentiator to obtain output derivatives without introducing delay.

Another limitation of INDI is that, in most existing derivations, the control effector dynamics are assumed to be negligible, implying that commanded control inputs are applied instantaneously. While this assumption can be reasonable for systems with sufficiently fast actuators, it does not accurately capture the behavior of most real-world actuators. As a result, the actual control input builds up gradually, which leads to deviations in both the gain and phase of the pseudo-control signal. To address this limitation, Raab et al. [2019] propose a continuous reformulation of the INDI control law that explicitly incorporates actuator dynamics into the control allocation process, such that the control effectiveness also depends on the control effector speed. This formulation also allows different control effectors with different bandwidths to be taken into account explicitly. It shows significant improvements for systems with slow actuators, and the resulting approach is referred to as Extended INDI.

A related contribution is presented by Steffensen et al. [2023b], who derive an NDI control law that explicitly accounts for first-order actuator dynamics, in a manner similar to Raab et al. [2019]. The resulting approach, referred to as Actuator NDI (ANDI), retains the state-dependent terms and provides an exact inversion for systems with such actuator dynamics. In this formulation, the additional terms depend inversely on the actuator bandwidth, such that their influence diminishes as the actuator dynamics become faster. This also clarifies the relation between ANDI and INDI. In the limit of infinitely fast actuators, all bandwidth-dependent terms vanish, and the control law reduces to an INDI-like structure. Conversely, if a finite actuator bandwidth is retained but the state-dependent terms are neglected, the formulation reduces to the Extended INDI approach. This interpretation also makes explicit which terms are neglected in practical INDI implementations.

Figure 2.17: Cessna Citation II 'PH-LAB' by Alan Wilson, licensed under CC BY-SA 2.0.

Thus far, robustness to aerodynamic model uncertainties and external disturbances has primarily been demonstrated through simulation studies and in-flight experiments, while existing stability analyses rely on linear transfer function representations derived from block diagrams [Sieberling et al., 2010, Smeur et al., 2016a]. In other words, the stability and robustness properties of sensor-based INDI have lacked rigorous mathematical justification. To address this limitation, Wang et al. [2019] reformulate INDI for systems with arbitrary relative degree without neglecting the incremental perturbation term, and provide rigorous proofs of nominal stability and ultimate boundedness under external disturbances and regular perturbations[1] using nonlinear Lyapunov methods and system perturbation theory. It is shown that disturbances enter the external dynamics only through their increments, implying that their influence can be reduced by decreasing the sampling interval. Consequently, for sufficiently small sampling times, INDI exhibits improved disturbance rejection properties compared to NDI.

Furthermore, the authors show that INDI leads to smaller closed-loop perturbation terms than NDI in the presence of regular perturbations. In particular, the model uncertainty term in NDI, given by the difference between the true drift dynamics and their estimate, is replaced in INDI by an incremental term whose influence decreases with increasing sampling frequency. In addition, multiplicative uncertainty in the control effectiveness depends on the control increment rather than on the full control input, which further reduces its impact. As a result, for sufficiently small sampling intervals, the overall perturbation magnitude under INDI is reduced and can be further diminished by increasing the sampling frequency, leading to smaller ultimate bounds for the system states. Singular perturbations were briefly considered as well, but detailed analysis was left as future work.

Building directly on the work of Wang et al. [2019], Pollack and van Kampen [2023] extend the analysis from the point where it was left off to include both regular and singular perturbations. The authors first incorporate these perturbations into the NDI framework and show that, under bounded perturbations and bounded virtual control, the NDI inversion residual may become large but remains bounded. For INDI, the results differ. The upper bound of the INDI inversion residual converges to zero as the gain of the singular perturbations approaches zero. This behavior is independent of the regular perturbations. This confirms the results of Wang et al. [2019] and shows that the INDI closed-loop system is robust to regular perturbations in the output dynamics. However, certain combinations of singular perturbations cause the inversion residual to become unbounded and thus limit the admissible

[1] In system perturbation theory, regular perturbations are those that do not increase the order of the system, such as variations in model coefficients [Khalil, 2002]. Singular perturbations, on the other hand, change the system order and typically represent fast time-scale unmodeled dynamics.

perturbation dynamics, called the synchronization effect. This contrasts with NDI, where an upper bound on the inversion residual always exists under bounded perturbations.

The synchronization effect explains issues that have been observed in earlier studies, for example in Smeur et al. [2016a], van't Veld et al. [2018]. A matching strategy is proposed in which the known singular perturbations in the input and output channels are aligned such that their effects cancel within the incremental inversion loop. This approach is consistent with the synchronization design philosophy commonly adopted in INDI-based control before.

Furthermore, the structured singular value framework is used to analyze the robust stability and performance properties of the (I)NDI architecture, and Pollack and van Kampen [2023] show that the robustness characteristics of a fully sensor-based INDI control law are complementary to those of the classical model-based variant. Sensor-based inversion does not increase robustness uniformly, but redistributes it across frequency. It improves robustness to parametric uncertainty in the aerodynamic stability and control derivatives in the low-to-medium frequency range, but reduces robustness to high-frequency dynamic uncertainties associated with actuator and structural dynamics. As a result, it is not meaningful to state that INDI is more robust than NDI without specifying the type of uncertainty and the frequency range, since gains in one region come at the expense of losses in another.

This complementary behavior motivates the development of hybrid INDI, which aims to achieve a more balanced robustness profile. In hybrid INDI, model-based and sensor-based inversion signals are combined, typically through a complementary filter structure, to compensate for inversion errors within a desired bandwidth Kim et al. [2021], Kumtepe et al. [2022], Pollack et al. [2024]. This approach can be interpreted as an extension of model-based NDI with an additional inversion error compensation loop that corrects for model inaccuracies. By tuning this compensation loop, the inversion can be made predominantly model-based or sensor-based, allowing the designer to shape the robustness trade-offs across frequency. In this way, hybrid INDI allows the robustness characteristics of NDI and INDI to be combined and adjusted according to the design requirements.

Building on this concept, Pollack [2024] provide a quantitative assessment of the resulting robustness characteristics of NDI, INDI, and hybrid INDI. In addition, he introduces a systematic procedure for designing the synchronisation filter within a robust control framework based on μ-synthesis. The relative strengths and limitations of these approaches are further illustrated through a case study involving the design of a C^* controller for a Boeing 747-100/200 simulation model. In this study, a multi-objective structured $\mathcal{H}_\infty$ synthesis problem is formulated under mixed uncertainty to tune the control parameters of each architecture.

In a separate line of work, Pollack et al. [2024] investigate whether hybrid INDI provides improved robustness and performance compared to classical gain-scheduled PID control. They note that systematic studies on the robust design benefits of INDI are scarce. The authors formulate both control strategies within a unified robust control framework and impose identical design specifications. They show that hybrid INDI can be expressed as an equivalent regulation and loop-shaping problem. In this formulation, the compensation dynamics introduce proportional-, integral-, and derivative-like behavior and reveal a close structural correspondence to PID control. Based on $\mathcal{H}_\infty$ synthesis and μ-analysis, hybrid INDI and gain-scheduled PID achieve virtually equivalent robustness and performance under the same requirements and uncertainty descriptions. The main conclusion is therefore not an inherent robustness advantage of hybrid INDI, but its structural properties. In particular, hybrid INDI provides a modular architecture in which the equivalent regulation term, inversion error compensation, and plant dynamics are explicitly separated. This allows a transparent implementation of implicit model-following control and supports systematic robust design.

Systematic robust design methods for hybrid INDI control laws have so far remained limited. This gap is addressed by Pollack et al. [2025], who observe that INDI introduces parameter-varying dynamics through its dependence on the system state. Based on this, the nonlinear closed-loop dynamics of the INDI-controlled system are reformulated into a quasi-linear parameter-varying (q-LPV) representation that preserves local linear behavior. This representation is extended to include inversion error compensation, resulting in a unified description of plant dynamics, inversion, and compensation. Based on this formulation, both the compensation gains and filter dynamics can be scheduled with the operating condition and designed using $\mathcal{H}_\infty$ synthesis. This enables robustness requirements to be imposed explicitly, while the nominal closed-loop behavior remains defined by the virtual control design. The approach is demonstrated on a nonlinear aeroservoelastic system, where robust performance is achieved over a wide operating range without modifying the desired nominal dynamics.

The performance and robustness of (I)NDI-based control are analyzed within a q-LPV framework in Pollack et al. [2026]. By exploiting the duality between the inversion action of (I)NDI and state transformations in q-LPV modeling, a unified representation is developed that encompasses model-based, sensor-based, and hybrid INDI, including singular perturbations. This enables robust analysis and synthesis in an LPV setting for output-nonlinear systems. The framework is demonstrated on a nonlinear aeroservoelastic system, where INDI-based and direct q-LPV controllers are optimized and compared. The results show that the structural constraints of (I)NDI can limit achievable robustness and performance, with direct q-LPV control attaining higher robustness for the same controller order. Nevertheless, the transparency and modularity of the (I)NDI architecture remain practical advantages. The proposed framework allows these trade-offs to be addressed systematically and enables (I)NDI-based controllers to approach near-optimal q-LPV robustness levels.

For completeness and as a recommendation, Steffensen et al. [2023a] provides a comprehensive overview of methods for compensating and synchronizing measurement filtering and delays in standard INDI, INDI with derivative filtering, and hybrid INDI with complementary filters. In addition, detailed analyses of time-delay margin effects due to control effector mismatch, actuator dynamics, and sampling rates are presented in Huang et al. [2022] and Raab [2025]. A two-part review of INDI is provided in Steinert et al. [2024] and Steinert et al. [2025]. To conclude, INDI has been successfully flight-tested in a wide range of applications and has demonstrated strong practical performance, although it does not fully meet the expectations that were once initially attributed to it. A collection of flight-tested applications of INDI is given in Table 2.2.

Table 2.2: Flight-tested applications of INDI controllers

Plant	**Flight test references**
Fighter	Smith and Berry [2000]
Fixed wing	Grondman et al. [2018], van Ekeren et al. [2018], Pollack et al. [2019], Pfeifle and Fichter [2021], Steinleitner et al. [2022], Xin et al. [2023]
Airship	Azinheira et al. [2015]
Helicopter	Pavel et al. [2020]
MAV	Smeur et al. [2016a], Smeur et al. [2018], Akkinapalli and Holzapfel [2018], Wang et al. [2021]
Tilt-rotor & Tilt-wing	Raab et al. [2018], Bhardwaj et al. [2018], Milz and Looye [2022]
Tailsitter	Binz et al. [2019], Smeur et al. [2020], Tal and Karaman [2022], Lovell-Prescod et al. [2023]

2.5. (I)NDI in Hypersonic Flight Control

At the end of the 1990s, the first proposals for applying feedback linearization techniques to the specific problem of hypersonic aircraft control began to appear. In all subsequent studies of that period, the focus remained exclusively on longitudinal control. The first application of NDI to a hypersonic aircraft was presented by Wang and Stengel [2000], who combined nonlinear dynamic inversion with stochastic robust control to design a flight control law for the winged–cone hypersonic aircraft model [Shaughnessy et al., 1990]. In their approach, altitude and velocity were selected as the controlled variables, with the velocity differentiated three times and the altitude four times to obtain a decoupled, feedback–linearizable system. This work was later critiqued by Wang and Stengel [2001], who argued that the exact feedback–linearized form was only achievable because the influence of the elevator on the lift coefficient had been artificially removed. This assumption eliminated the nonminimum–phase behavior, leading them to conclude that only approximate feedback linearization is achievable when altitude or flight-path angle and velocity are chosen as the control variables.

A follow-up study by Parker et al. [2006] extended the use of NDI to a more advanced longitudinal model of a hypersonic aircraft, the Bolender model developed in 2004 to include flexible modes, and propulsion dynamics [Bolender and Doman, 2005]. They observed that when either altitude h or flight–path angle γ is chosen as an output alongside velocity, the internal dynamics that remain after enforcing the output constraints are the pitch–attitude and pitch–rate states, together with the flexible modes in the full aeroelastic model. In the rigid–body case, these internal dynamics reduce to the short–period (α, q) subsystem. Under feedback linearization, holding h or γ exactly at their commands requires fixing γ and its first two derivatives to zero. This consumes the elevator authority entirely in service of the altitude/flight–path channel, leaving no control freedom to damp the short–period motion. For a statically unstable hypersonic configuration the short–period mode has a real unstable root, so the Jacobian of the zero–dynamics at trim is a hyperbolic saddle with one positive and one negative real eigenvalue, rendering the internal dynamics exponentially unstable. This instability persists even when the flexible modes are removed, confirming the conclusion of Wang and Stengel [2001] that the pitch dynamics alone can render the zero–dynamics unstable when h or γ is directly regulated under exact feedback linearization. They then suggested an approach for approximate feedback linearization by removing the flexible states and the elevator coupling which reduced the system to a full relative degree simplified model. Subsequently they applied the same feedback linearization strategy as employed by Wang and Stengel [2000]. An inner-loop NDI controller was designed for this simplified system, while the outer loop employed an LQR-based design with integral and model reference augmentation. The resulting controller achieves comparable tracking performance over a wider operating range than an earlier LQR design with integral augmentation, as presented in Groves et al. [2005], which was based on a linearized version of the Bolender model. The authors state that the tighter integration with the nonlinear model reduces the need for extensive tuning to achieve accurate tracking. However, this reduced tuning effort comes at the cost of increased complexity in both the system identification process and the computational implementation of the controller.

Three years later, Fiorentini and Serrani [2009] suggested that by assuming the availability of a canard in addition to an elevator, the unstable internal dynamics caused by regulating altitude or flight–path angle directly could be effectively counteracted. With both surfaces contributing to the pitching moment, the control–effectiveness matrix gains an additional column. The canard can then be ganged with the elevator to cancel the elevator's lift coupling, the very coupling responsible for the right–half–plane zero, while the remaining independent control authority is directed toward actively stabilizing the short–period mode that forms the internal dynamics. Their method combined dynamic inversion with a backstepping–based design, in which virtual control inputs are introduced for intermediate subsystems and a Lyapunov function is constructed at each step to guarantee stability.

Although canards improve controllability, they complicate thermal protection since they must withstand significant aerodynamic heating. Consequently, for hypersonic vehicles, it is of interest to develop nonlinear control strategies under the assumption that the elevator is the only aerodynamic control surface available for longitudinal dynamics. Therefore, in the same year, Fiorentini et al. [2009] sought to design a control system that avoided the need for a canard to compensate for nonminimum–phase behavior. To achieve this, they reformulated the system outputs so as to remove the unstable short–period mode from the internal dynamics. Instead of using the flight–path angle γ directly as a regulated output alongside velocity, they selected the pitch angle θ as the second output. This change altered the system's relative degree structure, reducing the zero–dynamics to a stable one–dimensional subsystem in γ, thereby eliminating the hyperbolic saddle behavior that occurs when γ is chosen directly. The cost of this reformulation is that γ can no longer be commanded directly; instead, it must be achieved indirectly by generating a suitable θ command through a feedforward mapping based on the desired γ trajectory, while the feedback loop acts on θ to stabilize the short–period dynamics. The authors themselves expressed this as a time-scale separation method.

Around the same time, Poulain et al. [2009] tackled nonminimum phase dynamics by introducing a geometric reformulation of the dynamics. Their method is a reference–point relocation. Instead of evaluating the vertical motion at the center of mass, they shift the controlled point along the body to a location where elevator inputs no longer influence the vertical acceleration directly. This adjustment is based on a kinematic transformation that redefines the longitudinal dynamics relative to the new point. By carefully selecting the location of this point using aerodynamic and inertial parameters, they cancel the elevator's effect on the flight path angle dynamics. As a result, the flight path angle behaves as a minimum-phase output, enabling it to be shaped through a feedforward path, while the pitch dynamics are treated through feedback. This separation allows standard nonlinear control methods to be applied without encountering internal instability due to nonminimum-phase zeros.

In 2013, Serrani [2013] revisited the unstable internal–dynamics problem, beginning with the same observation that the zero–dynamics associated with regulating velocity and flight–path angle (or altitude) contain a hyperbolic–saddle short–period mode. Unlike Fiorentini et al. [2009], who eliminated this instability by redefining the outputs and replacing γ with θ, thereby removing the short–period from the internal dynamics altogether, Serrani retained the original outputs. His approach augments the plant with a nonlinear internal model that captures persistent disturbances and tracking–error dynamics, then applies feedback linearization to the augmented system. This yields a hierarchical structure with two distinct sets of zero–dynamics: the outer zero–dynamics, governing the augmented altitude and velocity errors, and the inner zero–dynamics, containing the pitch dynamics and internal–model states. The control design proceeds in two stages: the outer loop uses certainty–equivalence linearizing feedback to regulate the auxiliary altitude error and velocity, while the inner loop uses a high–gain zero–dynamics controller to actively stabilize the short–period mode. The result is a formal time–scale separation, where the fast inner loop rapidly stabilizes the pitch subsystem, enabling the slower outer loop to treat it as quasi–steady while regulating the primary outputs. In contrast to Fiorentini's avoidance strategy, Serrani's method tackles the unstable zero–dynamics directly, preserving the original output definitions while ensuring stability through active control of both the inner and outer dynamics.

Finally, Ye et al. [2018] introduced another output–redefinition–based dynamic inversion method for nonminimum–phase hypersonic vehicles. Building on Fiorentini et al. [2009], they replaced the original outputs with virtual control outputs, defined as linear combinations of the original outputs and their derivatives. The coefficients were chosen so that the redefined system had stable zero–dynamics, thereby removing the hyperbolic–saddle instability without additional actuators. This retained a more direct influence over the flight–path angle γ than Fiorentini's substitution of θ, and avoided the internal–model–based zero–dynamics stabilization of Serrani [2013]. The method yields a natural time–scale separation. A fast inner loop regulates the virtual outputs, linked to the short–period pitch

dynamics. A slower outer loop adjusts the virtual–output commands to track γ, altitude, and velocity. This structure is equivalent to a cascaded time–scale separated control system, where the inner loop's stability allows the outer loop to treat the inner dynamics as quasi–steady.

The case is different for INDI. The technology has matured significantly and found applications across many aerospace fields, yet to the author's knowledge it has not previously been applied to an air–breathing hypersonic vehicle. Two prior studies come close, each in a different way. The first is that of Li et al. [2016], who applied Sensor–Based Backstepping (SBB), also known as incremental backstepping. SBB employs angular accelerometers to determine the time derivatives of the body angular rates and requires almost no accurate model information, retaining the fundamental advantages of traditional INDI. Numerical simulations using the air–breathing hypersonic vehicle model from Wang and Stengel [2000], including sudden parameter uncertainties, compared SBB with a linear augmented LQR and a real–time model–based NDI controller. The results demonstrated superior tracking performance and strong robustness to model uncertainties, consistent with theoretical expectations. While SBB is an incremental, sensor–based method closely related to INDI, it is not INDI in its standard form, and the study was restricted to longitudinal control of an air–breathing configuration.

A second closely related contribution is that of Autenrieb [2023], who developed an incremental nonlinear model–following control architecture for the attitude control of a hypersonic glide vehicle, the DLR GHGV–2 waverider. The design combines a cascaded, time–scale–separated INDI feedback loop with a model–following feedforward path, and it addresses the measurement–path difficulties central to INDI through complementary–filter angular–acceleration estimation and a synchronization filter on the control–surface feedback, together with optimization–based control allocation for the over–actuated effector set. This is, to the author's knowledge, the only prior incremental–dynamic–inversion study applied to a hypersonic vehicle. It differs from the present work in two respects that matter for the research gap. First, the GHGV–2 is an unpropelled glide vehicle, as the work explicitly concerns the re–entry and atmospheric glide phases and therefore does not address the strong aero–propulsive coupling that characterizes air–breathing hypersonic flight. Second, although the INMFC architecture is formulated for full three–axis attitude control, the reported simulations are restricted to longitudinal α maneuvers, leaving the lateral–directional behavior under incremental control largely unexamined.

2.6. Discussion of Findings

The literature reviewed in this chapter reveals a clear evolution in both hypersonic vehicle design and the associated control methodologies. Early hypersonic concepts were predominantly rocket-propelled or glide-based, optimized for ballistic or skip reentry profiles. In contrast, modern research has shifted toward air-breathing configurations that promise extended range and operational flexibility. These new concepts, however, bring considerable design and control challenges. Strong coupling among aerodynamics, propulsion, and structural dynamics gives rise to a system that is highly nonlinear, exhibits pronounced aeroelastic interactions, and operates across a broad flight envelope where linear control assumptions often fail. These complexities have motivated the transition from conventional linear control strategies to nonlinear approaches capable of addressing the strongly coupled and rapidly varying dynamics of hypersonic flight.

The study of hypersonic flight dynamics has traditionally relied on analytical models that simplify the coupled aeroelastic and propulsive behavior to isolate dominant phenomena. Representative examples include the winged cone, Chavez, and Bolender configurations. The Chavez model introduced structural flexibility, allowing the study of aeroelastic effects, while Bolender's formulation incorporated additional flexible modes to capture richer coupled dynamics. These models have been invaluable for theoretical studies but lack complete aerodynamic and propulsion databases, limiting their fidelity for control validation. The winged cone model, although more simulation-oriented, remains based on synthetic aerodynamic data and employs a canard configuration that is not representative of modern hypersonic

designs. In contrast, the GHAME vehicle, developed at NASA Ames and Dryden, offers a unique dataset derived from real flight measurements, combining data from multiple supersonic and hypersonic vehicles to form a generic, high-fidelity reference model. Its design avoids canards and reflects current aerodynamic and thermal considerations. Despite these advantages, studies employing GHAME are rare, and the model has not yet been systematically utilized as a benchmark for modern nonlinear control development.

NDI has shaped much of the modern perspective on nonlinear flight control. Its central idea is straightforward but powerful. By inverting the nonlinear dynamics, the method cancels the system nonlinearities and, under ideal assumptions, yields an exact input–output linearization that enables the use of powerful linear control techniques to an otherwise nonlinear system. However, exact input–output linearization is only achieved under specific conditions. It relies on an accurate model, a well-defined relative degree, and full availability of the required state variables. When these conditions are not satisfied, the cancellation of the nonlinear dynamics is no longer exact. Instead, a residual term remains and in this case, the controller no longer linearizes the true system, but rather an approximation of it, and the closed-loop dynamics are influenced directly by this residual error. In hypersonic flight, this limitation becomes significant due to both model uncertainty and limited state availability. Aerodynamic and propulsion models are uncertain because of sparse data and strong sensitivity to flight conditions, while several required states are not directly measurable and must be reconstructed. As a result, both modeling and estimation errors enter the inversion directly, leading to incomplete cancellation and degraded performance. Since NDI does not explicitly account for these effects, its robustness is limited, motivating extensions such as robust outer-loop designs or incremental NDI.

Incremental NDI reformulates the inversion problem in terms of measured increments instead of a full-state model, using sensor feedback and control effectiveness directly. This reduces reliance on aerodynamic modeling, making INDI inherently robust to such uncertainty. Its incremental structure introduces an integrator-like effect, which improves rejection of external low frequency disturbances compared to NDI. This advantage, however, comes from shifting the design burden from model accuracy to measurement quality. Because INDI relies on measured state derivatives, any imperfections in the measurement chain directly affect the control input. The delay margin in its measurement feedback path is inherently small. As a result, the method is more sensitive to singular perturbations associated with unmodeled higher-order dynamics, such as flexible modes, actuator and sensor dynamics, and measurement delays compared to NDI. These effects reduce delay margins further and must be explicitly accounted for in the design.

This trade-off highlights that INDI and NDI possess complementary robustness properties, which has motivated the development of hybrid INDI formulations aimed at shaping and balancing these characteristics. These approaches introduce an additional degree of freedom to modify the open-loop gain response and thereby can improve robustness.

Nonetheless, INDI has shown strong performance in nonlinear control applications for conventional aircraft, rotorcraft, micro air vehicles, and spacecraft. To the author's knowledge, INDI has not previously been applied to an air–breathing hypersonic vehicle. The two most closely related studies illustrate the gap. One applies a related SBB method to an air–breathing configuration but is restricted to longitudinal control, while the other applies a full incremental model–following architecture to a hypersonic wave rider which is an unpropelled glide configuration rather than an air–breathing one. Neither therefore addresses incremental control of an air–breathing hypersonic vehicle across its coupled longitudinal and lateral–directional dynamics. This raises the question of how well the method carries over to such vehicles.

Taken together, the reviewed studies motivate two main research objectives. First, to develop a high-fidelity implementation of the GHAME vehicle for control design and analysis. Second, to evaluate the applicability of incremental nonlinear dynamic inversion to air-breathing hypersonic vehicles, including an assessment of its underlying assumptions under practical sampling and actuator conditions.

2.7. Research Formulation

From the identified research gaps, the research objective is defined. To achieve this objective, the work is structured around two main objectives. First, the development and verification of an improved GHAME model with higher fidelity for hypersonic flight simulation and second, the application and evaluation of advanced nonlinear control strategies, namely NDI and INDI, on this model.

The research objective is to enhance the fidelity of the GHAME simulation model in Simulink and to investigate, for the first time, the application of (I)NDI controllers on this dataset, with the aim of assessing their suitability for hypersonic flight control under realistic modeling assumptions.

To address this objective in a practical way, it is divided into several research questions that guide the development, implementation, and evaluation process.

Research Question 1

How can the challenges and limitations of GHAME's current nonlinear Simulink model be addressed, thereby creating a higher fidelity simulation?

1.1. What are the key limitations of the existing Simulink GHAME model implementation?

1.2. In what ways can the GHAME model be refined to achieve higher physical and dynamic fidelity for hypersonic flight simulation?

1.3. How was the trimming procedure redesigned to ensure reliable convergence and applicability across the supersonic and hypersonic regimes of the GHAME vehicle?

1.4. How can the correct implementation and expected behavior of the nonlinear GHAME model in Simulink be verified?

Research Question 2

How can NDI–INDI be applied to the GHAME vehicle to evaluate its suitability for hypersonic flight control?

2.1. How was the nonlinear control architecture of the GHAME vehicle designed to combine NDI and INDI, and how were implementation aspects such as incremental feedback and filtering addressed within this structure?

2.2. How were the linear controllers designed across the different control loops?

2.3. How were digital implementation effects incorporated into the controller design, and what impact did they have on closed-loop stability?

Research Question 3

To what extent does the digitally implemented NDI–INDI control system, integrated within the nonlinear simulation in Simulink, satisfy the specified performance requirements across two flight conditions when subject to sensor delay, measurement noise, wind, turbulence, and actuator perturbations?

3.1. How is the cascaded NDI–INDI FCS implemented in discrete form within Simulink?

3.2. How does the cascaded NDI–INDI FCS satisfy the specified performance requirements during altitude and heading maneuvers when subjected to wind, turbulence, sensor noise, and actuator perturbations?

3.3. How does the cascaded NDI–INDI FCS satisfy the specified performance requirements during altitude and heading maneuvers when subjected to continuous-time sensor delay?

3

Theoretical Background

The theoretical background chapter establishes the framework for the key concepts underlying this research. It begins with the tensor notation and mathematical conventions used throughout the thesis in Sec. 3.1 and Sec. 3.2. The next section introduces the relevant reference frames and their associated transformation matrices in Sec. 3.3, which together form the foundation for the development of the control law and simulation model. Subsequently, the principles of nonlinear dynamic inversion are introduced in Sec. 3.4, followed by its incremental formulation in Sec. 3.5, which also incorporates the concept of delay synchronization essential for practical implementation.

3.1. Notation

The notation adopted in this thesis follows the modified tensor convention described by Zipfel [2023]. In the Euclidean space of Newtonian mechanics, all physical quantities are categorized into three types. Scalars, which are rank-0 tensors, are written in italic font, such as b for span. Vectors, representing rank-1 tensors, are denoted by lowercase boldface letters, for example, $\boldsymbol{v}$. Rank-2 tensors, such as the inertia tensor, are written using uppercase boldface letters, e.g., $\boldsymbol{I}$ for the moment of inertia.

Subscripts are used to denote physical points, while superscripts on a tensor indicate the reference frame with respect to which the quantity is defined. When a tensor carries two such identifiers, whether as superscripts, subscripts, or both, they are always read from left to right: the first symbol refers to the object of interest, and the second to what it is defined relative to. For example, ω^{BI} denotes the angular velocity of frame B with respect to frame $\boldsymbol{I}$, and $\boldsymbol{s}_{AB}$ denotes the position of point A relative to point B. This reading order also applies to rotational derivatives and skew-symmetric angular velocity tensors, and ensures consistency across all multibody and flight dynamics expressions.

Coordinate representations of tensors are enclosed in square brackets, with a superscript indicating the coordinate system in which the components are expressed. Once enclosed in brackets, boldface is removed to emphasize that the object is no longer a geometric tensor. For example, $[\omega^{BI}]^B$ denotes the angular velocity of frame B with respect to frame I, expressed in body coordinates. Likewise, $[s_{AB}]^E$ denotes the displacement of point A with respect to point B, expressed in Earth coordinates.

An important exception to the left-to-right reading rule is the notation for transformation matrices. The matrix $[T]^{BA}$ transforms a tensor expressed in coordinate system A into its equivalent representation in coordinate system B. It does not alter the tensor's physical meaning or its reference frame, only the coordinate basis. For example, $[x]^B = [T]^{BA}[x]^A$ converts the same geometric vector from one coordinate system to another. The superscript ^{BA} must therefore be read as "from A to B" in terms of coordinates. This distinguishes transformation matrices from physical quantities like ω^{BI}, which describe motion or relative orientation between reference frames. The transpose of a transformation matrix is

denoted using an overline. For example, $[\overline{T}]^{AB}$ denotes the transpose of $[T]^{AB}$, and maps vectors from coordinate system A to coordinate system B. Although this operation is mathematically equivalent to $([T]^{AB})^\top$, the overline notation improves readability by eliminating the need for parentheses or stacked superscripts.

The operator D^A denotes the rotational time derivative of a tensor, taken with respect to reference frame A. This symbolic form is used instead of subscripts or overbars to clearly indicate the frame of differentiation. It applies to all tensors of rank one or higher and distinguishes geometric (frame-dependent) derivatives from purely component-wise rates. Coordinate expressions follow the standard bracketed form, e.g., $[D^A v]^B$. This convention avoids ambiguity in systems involving multiple rotating frames and is standard in flight dynamics. A hat symbol indicates an estimated quantity. For any tensor $\boldsymbol{x}$, its estimate is denoted $\hat{\boldsymbol{x}}$. The hat is applied before any coordinate transformation, so the coordinate representation of an estimate is written $[\hat{x}]^A$, not $\widehat{[x]^A}$. Hat notation is reserved exclusively for estimates and is not used to denote unit vectors or directions.

3.2. Tensors and Transformation Matrices

Tensors are geometric objects that exist independently of any coordinate system. They have an intrinsic meaning that is invariant under changes of basis. Coordinate systems are introduced solely for the purpose of numerical representation; they do not affect the physical definition of the tensor itself. In the context of this thesis, all tensors are assumed to reside in three-dimensional Euclidean space and are expressed in Cartesian coordinate systems. A geometric object qualifies as a tensor if its coordinate representations transform according to the standard tensor transformation laws. For a rank-one tensor $\boldsymbol{x}$ and a rank-two tensor $\boldsymbol{X}$, their components in two coordinate systems A and B are related by:

$$[x]^B = [T]^{BA}[x]^A, \tag{3.1}$$

$$[X]^B = [T]^{BA}[X]^A[\overline{T}]^{BA}, \tag{3.2}$$

where $[T]^{BA}$ is the transformation matrix from coordinate system A to coordinate system B, and $[\overline{T}]^{BA}$ denotes its transpose. These relations define the correct behavior of tensors under change of coordinates. If a quantity does not transform according to these rules, it is not a tensor. Any rank-one tensor $\boldsymbol{x}$ can also be associated with a skew-symmetric rank-two tensor $\boldsymbol{X}$, which is particularly useful for expressing cross products in matrix form. This tensorial identity is given in components as:

$$[x]^A = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^A \quad \Longleftrightarrow \quad [X]^A = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}^A . \tag{3.3}$$

This allows the cross product $\boldsymbol{x} \times \boldsymbol{y}$ to be written compactly as $\boldsymbol{X}\boldsymbol{y}$. This form appears frequently in the formulation of rotational kinematics and dynamics, particularly in the representation of angular velocity and torque. Rotational derivatives preserve the tensor character of quantities and play a central role in describing motion. If s_{BA} is the displacement of point B relative to a reference point on frame A, the velocity and acceleration of point B with respect to frame A are given by the first and second rotational derivatives of the displacement:

$$\boldsymbol{v}_B^A = D^A \boldsymbol{s}_{BA}, \tag{3.4}$$

$$\boldsymbol{a}_B^A = D^A D^A \boldsymbol{s}_{BA}. \tag{3.5}$$

Time derivatives of vectors with respect to different frames are related through the Euler transformation. For any vector $\boldsymbol{x}$, the relation between its time derivative in frames A and B is given by:

$$D^A \boldsymbol{x} = D^B \boldsymbol{x} + \boldsymbol{\Omega}^{BA} \boldsymbol{x}, \tag{3.6}$$

where $\boldsymbol{\Omega}^{BA}$ is the angular velocity matrix of frame B with respect to frame A. Tensors must follow appropriate index logic when combined. Displacements and angular velocities, for instance, follow consistent addition rules when expressed in a common coordinate system. Consider the vector addition of displacements. Given three points A, B, and C, the displacement from A to B can be decomposed as:

$$\boldsymbol{s}_{BA} = \boldsymbol{s}_{BC} + \boldsymbol{s}_{CA}. \tag{3.7}$$

This expression follows from the geometric interpretation of displacements and respects head-to-tail ordering of subscripts. Similarly, angular velocities add as follows:

$$\boldsymbol{\omega}^{BI} = \boldsymbol{\omega}^{BE} + \boldsymbol{\omega}^{EI}. \tag{3.8}$$

Transformation matrices are not tensors. They do not represent physical quantities and do not transform according to the tensor transformation laws. Instead, they are coordinate operators that act on the representation of tensors, enabling conversion between different coordinate systems while leaving the tensor itself and its reference frame unchanged. A transformation matrix $[T]^{BA}$ maps the coordinate representation of a tensor from coordinate system A to coordinate system B. It does not redefine the frame in which the tensor is observed or alter the geometric meaning of the tensor. All transformation matrices used in this thesis are orthogonal, which implies that their transpose equals their inverse:

$$[\overline{T}]^{BA} = ([T]^{BA})^{-1}. \tag{3.9}$$

This orthogonality ensures that the Euclidean norm of a tensor is preserved under coordinate transformations. Additionally, the determinant of a transformation matrix satisfies $\det([T]^{BA}) = \pm 1$, but only transformations with unit determinant are permitted to preserve right-handedness. That is, only matrices with $\det([T]^{BA}) = +1$ are used. The transpose of a transformation matrix reverses the direction of coordinate transformation without changing its superscript structure, meaning that $[\overline{T}]^{BA}$ maps from B back to A. Composition of transformations is associative: if $[T]^{BA}$ maps from A to B and $[T]^{CB}$ from B to C, then the combined transformation:

$$[T]^{CA} = [T]^{CB}[T]^{BA}, \tag{3.10}$$

maps from A to C. Transformation matrices do not, in general, commute.

3.3. Reference Frames, Coordinate Systems, and Transformations

Accurate modeling of vehicle motion requires a precise definition of reference frames and coordinate systems. This section introduces the four principal reference frames used throughout the simulation: the inertial frame I, the Earth-fixed frame E, the body-fixed frame B, and the air-relative frame A. These frames are physically meaningful constructs used to express positions, velocities, orientations, and forces. It is important to maintain a clear distinction between reference frames and coordinate systems. A reference frame defines an origin and orientation in space relative to other physical objects or frames, whereas a coordinate system provides a numerical representation of vectors and tensors within that frame.

Coordinate transformations are used to express geometric quantities in different coordinate systems associated with these frames. These are handled using transformation matrices, which preserve the underlying physical meaning while converting between coordinate representations. The following sections define the reference frames used, the coordinate systems in which quantities are expressed in, and the transformations that relate them.

Reference Frames and Coordinate Systems

A reference frame is defined by a point of origin, known as the frame center, and an orientation specified by three mutually orthogonal unit vectors $(\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3)$. The frame center establishes the location of the frame in space, while the ordered triad of base vectors defines its orientation. Together, they fully describe the frame A in both position and orientation. This is illustrated in Fig. 3.1.

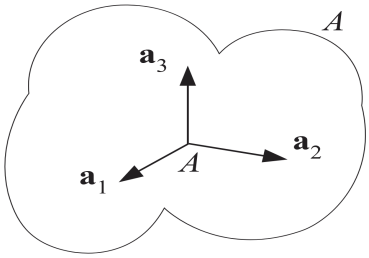
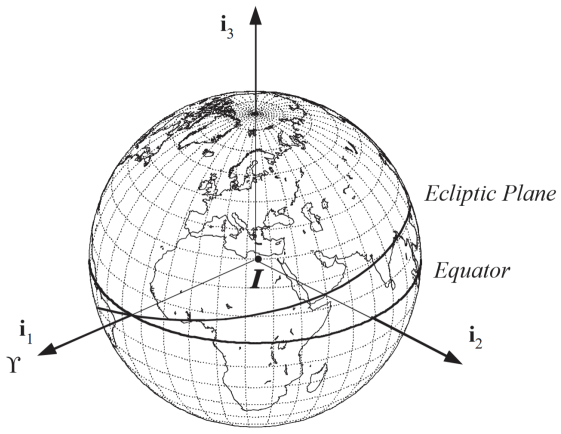


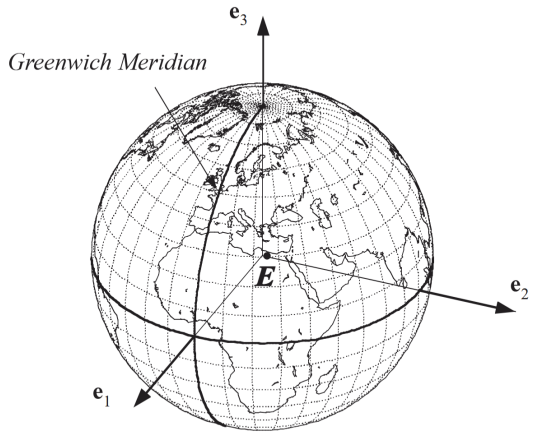
Figure 3.1: Reference frame A, consisting of a frame center and a right-handed orthonormal basis, taken from Zipfel [2025].

Inertial Frame and Coordinate System

The flat-Earth assumption is commonly adopted in low-altitude or short-range flight regimes where the motion occurs over distances small compared to Earth's radius, and where gravity can be considered uniform and constant in direction. However, this assumption becomes invalid at hypersonic speeds and high altitudes, where the vehicle traverses a significant portion of Earth's curvature within a short time. In such conditions, the curvature and rotation of Earth introduce non-negligible effects on gravity direction, inertial velocity, and trajectory propagation, making a flat-Earth approximation insufficient for accurate dynamics modeling. Therefore, the flat-Earth assumption is discarded in this model. As a result, the Earth-fixed coordinate system is non-inertial, and a truly inertial reference is required. The inertial coordinate system $]^I$ is taken to be the Earth-centered inertial system defined with respect to the J2000 epoch[1]. Its origin is located at the Earth's center, and its orientation is given by the orthonormal base vectors $\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3$.



(a) Inertial reference frame, taken from Zipfel [2025].



(b) Earth-fixed reference frame, taken from Zipfel [2025].

Figure 3.2: Reference frames used for vehicle modeling.

[1]The J2000 epoch corresponds to January 1, 2000, at 12:00 TT (Terrestrial Time), which serves as a standard reference for celestial coordinate systems and Earth-centered inertial frames.

The vector $\boldsymbol{i}_1$ points toward the vernal equinox, the intersection of the ecliptic and equatorial planes, and is fixed with respect to the distant stars. The vector $\boldsymbol{i}_3$ aligns with Earth's spin axis, pointing northward through the geographic poles. The third vector $\boldsymbol{i}_2$ completes the right-handed triad and lies in the equatorial plane, perpendicular to both $\boldsymbol{i}_1$ and $\boldsymbol{i}_3$. Because Earth's rotation axis, equatorial plane, and the ecliptic all undergo slow secular variations due to gravitational perturbations from other celestial bodies, a fixed epoch must be used to define an inertial frame. The J2000 system, based on the reference epoch J2000.0 (January 1, 2000, 12:00 TT), is commonly adopted for this purpose in aerospace applications [Zipfel, 2025].

The Inertial coordinate system $]^I$ is the preferred coordinate system of the Inertial reference frame I, which means that its coordinate axes $1^I, 2^I, 3^I$ are aligned with the frame basis $\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3$. Because of this alignment, the orientation of the coordinate system matches that of the reference frame: 1^I points toward the vernal equinox, 3^I points through the North Pole along Earth's rotation axis, and 2^I completes the right-handed coordinate system.

Earth-fixed Reference Frame and Coordinate System

The Earth-fixed frame, denoted by E, shares the same origin as the inertial frame and is initially aligned with it in both the equatorial plane and along Earth's spin axis. However, unlike the inertial frame, which remains fixed relative to the stars, the Earth-fixed frame rotates with the Earth at a constant angular velocity. Its orientation is defined by the orthonormal triad $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$. The vector $\boldsymbol{e}_1$ points toward the intersection of the Greenwich meridian with the equator. The Greenwich meridian, also known as the prime meridian, is the reference for zero longitude and passes through the Royal Observatory in Greenwich, England [Zipfel, 2025]. The vector $\boldsymbol{e}_3$ is aligned with Earth's rotation axis, directed northward through the geographic poles. The third vector $\boldsymbol{e}_2$ completes the right-handed system, lying in the equatorial plane and perpendicular to both $\boldsymbol{e}_1$ and $\boldsymbol{e}_3$.

Similarly, the Earth-fixed coordinate system $]^E$ is the preferred coordinate system of the Earth-fixed reference frame E, meaning that its coordinate axes $1^E, 2^E, 3^E$ are aligned with the basis $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$. Due to this alignment, the coordinate system shares the orientation of the reference frame: 1^E points along the intersection of the equator and the prime meridian, 3^E points through the North Pole along Earth's rotation axis, and 2^E completes the right-handed coordinate system.

Body-fixed Reference Frame and Coordinate System

The body-fixed frame, denoted by B, is defined with respect to a rigid body and remains fixed relative to its structure. Its origin is placed at the vehicle's center of mass (CM), and its orientation is described by the orthonormal basis $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$. The vector $\boldsymbol{b}_1$ lies in the symmetry plane and points forward toward the nose of the vehicle, $\boldsymbol{b}_2$ points laterally toward the right wing, and $\boldsymbol{b}_3$ points downward, forming a right-handed basis.

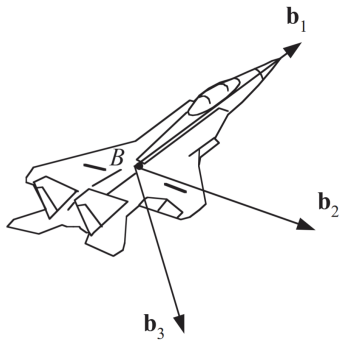


Figure 3.3: Body-fixed reference frame B with orthonormal basis $(\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3)$, taken from Zipfel [2025].

The body-fixed coordinate system $]^B$ is the preferred coordinate system of the body-fixed reference frame and its coordinate axis $1^B, 2^B, 3^B$ are aligned with the triad of the body reference frame. The 1^B-axis points forward through the nose of the vehicle, the 2^B-axis points out the right wing, and the 3^B-axis points downward, perpendicular to the symmetry plane. It is therefore often referred to as a Forward-Right-Down (FRD) coordinate system.

Atmosphere-fixed Reference Frame

The atmosphere-fixed reference frame, denoted by A, is a non-inertial frame attached to the surrounding air mass. Its motion is defined by the local wind field and is therefore independent of the vehicle motion. However, if the air mass is in uniform rectilinear motion with respect to an inertial frame, then the atmosphere-fixed frame becomes inertial.

It is important to note that this is not a different coordinate system of the body frame, but a distinct reference frame. A coordinate transformation (e.g., from body to inertial) only changes the representation of a physical vector while preserving its magnitude. By contrast, introducing the atmosphere-fixed frame changes the underlying physical quantity itself: the presence of wind alters the vehicle's velocity, so that its magnitude differs from the Earth-relative velocity. If the atmosphere is at rest, the atmosphere-fixed frame coincides with the Earth-fixed reference frame. In this case, the velocity of the vehicle relative to the atmosphere reduces to its velocity relative to the Earth.

Geocentric and Geodetic Coordinate System

Different coordinate systems are used to describe the position of an aircraft or any object on Earth, depending on the assumed Earth model and the required level of accuracy. The simplest model represents Earth as a perfect sphere, characterized by a constant radius in all directions. While this approximation is mathematically convenient, it neglects the fact that Earth's rotation causes a bulging at the equator and a flattening at the poles. As a result, the actual shape of Earth is better represented by an oblate spheroid with a larger equatorial radius than polar radius. When Earth is approximated as a sphere, the geographical coordinate system is commonly used. If an oblate spheroid model is adopted to account for Earth's equatorial bulge and polar flattening, the geographical system is renamed the geocentric coordinate system, and the geodetic coordinate system is introduced to distinguish between directions defined relative to Earth's center and those defined relative to the surface normal of the ellipsoid.

The geocentric coordinate system $]^G$ represents position relative to Earth's center. Geocentric latitude is defined as the angle between the equatorial plane and the position vector from the center of the Earth. The system follows the North-East-Down (NED) convention: the first axis 1^G points toward geographic North along the meridian, the second axis 2^G points East along the parallel, and the third axis 3^G points directly toward Earth's center. The coordinate axes rotate with Earth.

In contrast, the geodetic coordinate system $]^D$ accounts for the ellipsoidal shape of Earth. Here, the third axis 3^D aligns with the local surface normal of the reference ellipsoid, not with the Earth's center. Geodetic latitude is measured as the angle between the equatorial plane and this local normal. The system also follows the NED convention: 1^D points North, 2^D points East, and 3^D points downward along the ellipsoid normal. Unlike 3^G, the axis 3^D does not generally intersect Earth's center, except at the equator and the poles. These coordinate axes likewise rotate with Earth.

Flight-path Coordinate System

The flight-path coordinate system is defined using the direction of the velocity vector $\boldsymbol{v}$, which can refer either to the ground-relative velocity $\boldsymbol{v}_B^E$ or the air-relative velocity $\boldsymbol{v}_B^A$, depending on the context. The first axis, 1^V, is aligned with $\boldsymbol{v}$. The second axis, 2^V, lies in the local horizontal plane and points to the right of the velocity vector, and the third axis, 3^V, completes the right-handed triad by pointing downward. This coordinate system captures the vehicle's motion direction and is very useful when formulating kinematic or navigation equations.

Wind Coordinate System

The wind coordinate system shares its first axis 1^A with the flight-path coordinate system, aligning with the direction of the velocity vector $\boldsymbol{v}$. However, unlike the flight-path coordinates system, the orientation of the remaining axes is chosen to reflect the symmetry of the aircraft. The third axis 3^A lies in the symmetry plane of the vehicle and points downward relative to the body, while the second axis 2^A points to the right, completing a right-handed triad. This choice makes the wind coordinate system better suited for expressing forces such as lift and drag, which are defined relative to the aircraft's geometry.

Table 3.1: Summary of reference frames

Frame	**Base point**	**Base vectors**	**First direction**	**Third direction**
Inertial	CM of Earth, I	$\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3$	$\boldsymbol{i}_1$: Vernal Equinox	$\boldsymbol{i}_3$: Earth's spin axis
Earth-fixed	CM of Earth, E	$\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$	$\boldsymbol{e}_1$: Greenwich meridian	$\boldsymbol{e}_3$: Earth's spin axis
Body-fixed	CM of body, B	$\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$	$\boldsymbol{b}_1$: along nose direction	$\boldsymbol{b}_3$: downward through fuselage

Coordinate Transformations

Accurately describing the vehicle's motion requires transformations between different coordinate systems. A coordinate transformation changes only the representation of a vector, while the underlying reference frame remains fixed. In flight dynamics, different physical quantities are naturally expressed in different coordinate systems. For example, position is described in geocentric or geodetic coordinates, while aerodynamic quantities are defined in body- or velocity-aligned systems. To ensure consistency in the equations of motion, these quantities must be transformed between coordinate systems. This section introduces the transformation matrices required to relate the coordinate systems used in the model.

Unit Axis Transformation Matrices

Transformation matrices are used to convert the coordinate representation of a tensor between Cartesian coordinate systems. Their mathematical properties, notation, and composition rules are defined in Sec. 3.2. In the following, the matrix forms used to construct rotations in three-dimensional space are introduced.

In three-dimensional space, coordinate transformations between Cartesian systems can be constructed from *unit axis transformation matrices*, which correspond to rotations about the principal axes of a coordinate system. These matrices serve as the fundamental building blocks for all transformations used in this work. A rotation by angle φ_1 about the 1-axis, by angle φ_2 about the 2-axis, and by angle φ_3 about the 3-axis are respectively given by:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_1 & \sin\varphi_1 \\ 0 & -\sin\varphi_1 & \cos\varphi_1 \end{bmatrix}, \quad \begin{bmatrix} \cos\varphi_2 & 0 & -\sin\varphi_2 \\ 0 & 1 & 0 \\ \sin\varphi_2 & 0 & \cos\varphi_2 \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} \cos\varphi_3 & \sin\varphi_3 & 0 \\ -\sin\varphi_3 & \cos\varphi_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.11}$$

General transformations are formed by multiplying these elementary rotations in a prescribed sequence. In three-dimensional space, at most three independent rotations are required to describe any relative orientation between Cartesian coordinate systems. The order of multiplication must be preserved, since transformation matrices do not commute, as discussed in Sec. 3.2. Each transformation introduced in this chapter is therefore defined by a specific rotation sequence.

Inertial to Earth-fixed Coordinate System $[T]^{EI}$

The transformation between these coordinate systems accounts for Earth's rotation, which continuously alters the orientation of the Earth-fixed coordinate system relative to the Inertial coordinate system. Since Earth completes one full rotation relative to the stars in a sidereal day (approximately 23 hours, 56 minutes, and 4 seconds), the Earth-fixed frame realigns with the inertial frame once per sidereal day. The accumulated rotation is quantified by the Greenwich hour angle Ξ, which measures the angular displacement of the Greenwich meridian relative to the vernal equinox. A visual representation is provided in Fig. 3.4.

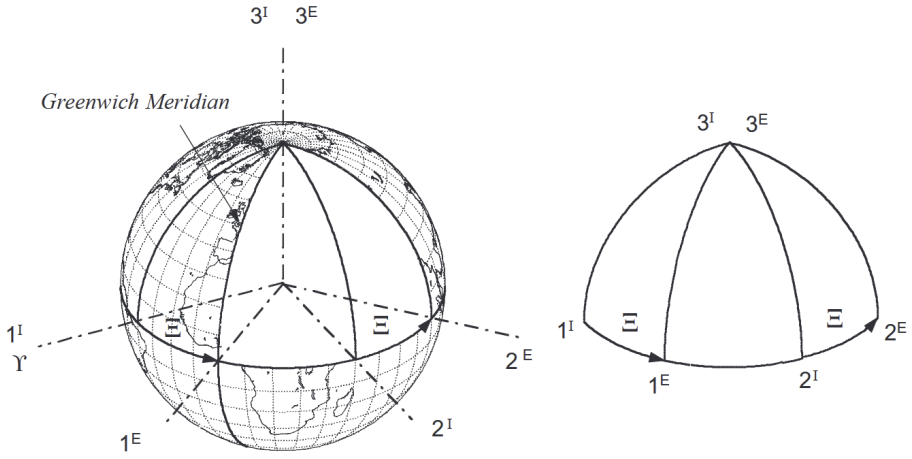


Figure 3.4: Inertial and Earth coordinate systems, taken from Zipfel [2025].

By inspection, this rotation is a constant counterclockwise rotation by angle Ξ around the third inertial axis, 3^I, which corresponds to the third-axis unit rotation matrix. The transformation from $]^I$ to $]^E$ is therefore given by:

$$[T]^{EI} = \begin{bmatrix} \cos\Xi & \sin\Xi & 0 \\ -\sin\Xi & \cos\Xi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.12}$$

The angle Ξ is initialized based on the date and time at the start of the simulation. As the simulation progresses, it evolves due to Earth's uniform rotation and must be continuously updated to reflect the accumulated angular displacement over time. With Earth's angular velocity $\omega_\oplus$, the Greenwich hour angle at time t is given by:

$$\Xi(t) = \Xi_0 + \omega_\oplus t, \tag{3.13}$$

where Ξ_0 is the initial Greenwich hour angle at the simulation start time, and t is the elapsed simulation time. This formulation ensures that the transformation matrix $[T]^{EI}$ correctly tracks the Earth-fixed frame's orientation with respect to the inertial frame throughout the simulation.

Geocentric to Geodetic Coordinate System $[T]^{DG}$

The difference between geocentric and geodetic latitudes is quantified by the deflection angle δ, which arises due to Earth's oblateness. This angle accounts for Earth's flattening f and can be computed using the relation from Britting [2010]:

$$\delta = f \sin(2\lambda_d) \left(1 - \frac{f}{2} - \frac{h}{R_0}\right), \tag{3.14}$$

where λ_d is the geodetic latitude, h is the altitude of the body B above the spheroidal surface, and R_0 denotes the distance from the reference point B_0 on the surface to Earth's center. The quantity R_0 can be approximated as:

$$R_0 = a \left\{1 - \frac{f}{2}\left[1 - \cos(2\lambda_d)\right] + \frac{5f^2}{16}\left[1 - \cos(4\lambda_d)\right]\right\}. \tag{3.15}$$

A graphical depiction of the deflection angle is provided in Fig. 3.5, illustrating the geometric relationship between the geocentric and geodetic latitude angles.

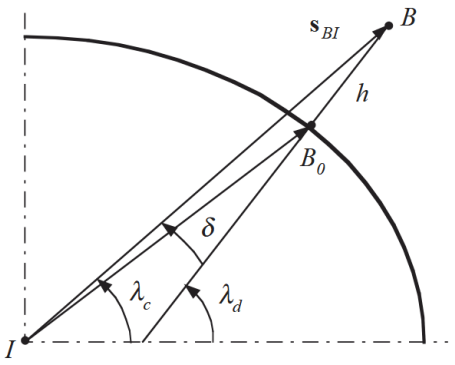


Figure 3.5: Illustration of the deflection angle δ, adapted from Zipfel [2025].

The angle δ represents the rotation necessary to transform vectors between the geodetic and geocentric frames. Since both frames share the same local meridional plane, this transformation is a right-handed rotation about the common second axis $2^G = 2^D$, which points East. A positive deflection angle corresponds to a rotation from the geodetic to the geocentric frame. The transformation is performed using the second-axis unit rotation matrix as defined in Eq. (3.11):

$$[T]^{GD} = \begin{bmatrix} \cos\delta & 0 & -\sin\delta \\ 0 & 1 & 0 \\ \sin\delta & 0 & \cos\delta \end{bmatrix}. \tag{3.16}$$

In the translational dynamics model, gravitational acceleration must be transformed from geocentric to geodetic coordinates. In that case, the direction is reversed, and the rotation angle becomes $-\delta$, corresponding to the transpose of the above matrix:

$$[T]^{DG} = [\overline{T}]^{GD} = \begin{bmatrix} \cos\delta & 0 & \sin\delta \\ 0 & 1 & 0 \\ -\sin\delta & 0 & \cos\delta \end{bmatrix}. \tag{3.17}$$

For a perfectly spherical Earth, $\delta = 0$, meaning the geocentric and geodetic coordinate systems coincide. Since the geodetic coordinate system is locally normal to Earth's surface, any point on the surface can be described using geodetic latitude and longitude.

Earth-fixed to Geodetic Coordinate System $[T]^{DE}$

The transformation from the Earth-fixed coordinate system $]^E$ to the geodetic coordinate system $]^D$ is carried out using two consecutive rotations. Therefore, one intermediate frame is introduced. The first step is to rotate about the 3^E-axis the longitude l, bringing the 2^E-axis to the 2^D-axis. This corresponds to the third unit axis transformation matrix around l:

$$[T]^{XE} = \begin{bmatrix} \cos l & \sin l & 0 \\ -\sin l & \cos l & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.18}$$

The second stage of the transformation is performed about the east-pointing axis 2^X and is described in two consecutive rotations. First, a right-handed rotation of angle $\frac{\pi}{2} - \lambda_d$ is applied about the axis 2^X, transforming the intermediate coordinate system $]^X$ into $]^Y$. This rotation aligns the third axis with the outward normal to the ellipsoidal surface. At this stage, the coordinate system has the correct geometric alignment with respect to the local vertical direction. However, the orientation of the axes does not yet match the geodetic coordinate system: the axis 1^Y points toward the South instead of North, and the axis 3^Y points upward instead of downward.

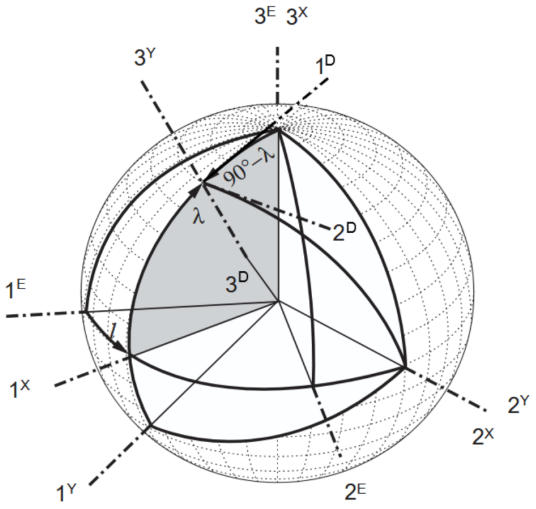


Figure 3.6: Relationship between the Earth-fixed and geodetic coordinate systems, adapted from Zipfel [2025].

To correct this orientation, a second rotation of $-180°$ (i.e., $-\pi$) is applied about the same axis $2^Y = 2^D$. This rotation leaves the east axis unchanged but reverses the directions of the other two axes. As a result, 1^Y is mapped to 1^D, now pointing North, and 3^Y is mapped to 3^D, now pointing downward. This completes the alignment with the geodetic coordinate system.

Since both rotations are performed about the same axis, they can be combined into a single equivalent rotation about 2^X. The total rotation angle is therefore:

$$\left(\frac{\pi}{2} - \lambda_d\right) - \pi = -\lambda_d - \frac{\pi}{2}, \tag{3.19}$$

which is implemented using the second-axis unit transformation matrix:

$$[T]^{DX} = \begin{bmatrix} \cos\left(-\lambda_d - \frac{\pi}{2}\right) & 0 & -\sin\left(-\lambda_d - \frac{\pi}{2}\right) \\ 0 & 1 & 0 \\ \sin\left(-\lambda_d - \frac{\pi}{2}\right) & 0 & \cos\left(-\lambda_d - \frac{\pi}{2}\right) \end{bmatrix} = \begin{bmatrix} -\sin\lambda_d & 0 & \cos\lambda_d \\ 0 & 1 & 0 \\ -\cos\lambda_d & 0 & -\sin\lambda_d \end{bmatrix}. \tag{3.20}$$

Combining this with the transformation matrix from Eq. (3.18) and leads to the transformation chain:

$$E \xrightarrow{l} X \xrightarrow{\frac{\pi}{2}-\lambda_d} Y \xrightarrow{-180^\circ} D.$$

The resulting coordinate system is oriented such that the first axis 1^D points North, the second axis 2^D points East, and the third axis 3^D points downward, normal to the ellipsoidal surface. The full transformation matrix from Earth-fixed to geodetic coordinates is:

$$[T]^{DE} = [T]^{DX}[T]^{XE} = \begin{bmatrix} -\sin\lambda_d \cos l & -\sin\lambda_d \sin l & \cos\lambda_d \\ -\sin l & \cos l & 0 \\ -\cos\lambda_d \cos l & -\cos\lambda_d \sin l & -\sin\lambda_d \end{bmatrix}. \tag{3.21}$$

This process is valid for both the geocentric and geodetic coordinate system because the only difference is the used latitude angle. The geometry of this process is illustrated in Fig. 3.6, which shows a general latitude λ and the relationship between the Earth-fixed, intermediate, and geographic coordinate systems.

Inertial to Geodetic Coordinate System $[T]^{DI}$

The transformation from the inertial coordinate system $]^I$ to the geodetic coordinate system $]^D$ is obtained by extending the Earth-fixed to geodetic transformation to account for Earth's rotation. This is achieved by replacing the geodetic longitude l with the celestial longitude l_i, which describes the inertial orientation of the local meridian. The celestial longitude is defined as:

$$l_i = \Xi_0 + \omega_\oplus t + l, \tag{3.22}$$

where Ξ_0 is the Greenwich hour angle at the reference epoch t_0, and $\omega_\oplus$ is the Earth's angular velocity. Substituting l_i into the Earth-fixed to geodetic transformation yields the inertial to geodetic transformation matrix:

$$[T]^{DI} = \begin{bmatrix} -\sin\lambda_d \cos l_i & -\sin\lambda_d \sin l_i & \cos\lambda_d \\ -\sin l_i & \cos l_i & 0 \\ -\cos\lambda_d \cos l_i & -\cos\lambda_d \sin l_i & -\sin\lambda_d \end{bmatrix}. \tag{3.23}$$

This transformation is also used to construct the inertial-to-geocentric transformation matrix $[T]^{GI}$, which is required to express gravitational acceleration in inertial coordinates. It is obtained by combining the geodetic-to-geocentric transformation $[\overline{T}]^{DG}$ from Eq. (3.17) with $[T]^{DI}$:

$$[T]^{GI} = [\overline{T}]^{DG}[T]^{DI}. \tag{3.24}$$

Geodetic to Body-fixed Coordinate System $[T]^{BD}$

By convention, the orientation of an aircraft is defined as the relative orientation of the body coordinate system $]^B$ with respect to the geodetic coordinate system $]^D$, describing how the vehicle is rotated relative to the local-level reference. This relative orientation is expressed by the three Euler angles θ_{bd}, ϕ_{bd}, and ψ_{bd}, where the subscript $_{bd}$ denotes the rotation of the body with respect to the geodetic coordinate system. For readability, the subscript is omitted in the following expressions, but it should not be confused with the inertial Euler angles, which describe orientation with respect to the inertial coordinate system $]^I$. The transformation matrix from $]^D$ to $]^B$ is constructed from a sequence of three right-handed rotations, most commonly in the *yaw–pitch–roll* order. To represent this sequence, two intermediate coordinate systems, X and Y, are introduced, leading to the transformation chain:

$$D \xrightarrow{\psi} X \xrightarrow{\theta} Y \xrightarrow{\phi} B.$$

The transformation begins with a rotation about the vertical axis 3^D by the yaw angle ψ, aligning the intermediate axis 1^X with the vertical plane of the vehicle's nose, i.e., the direction of 1^B. Next, a

rotation about the lateral axis 2^X by the pitch angle θ adjusts the elevation so that 1^Y aligns with 1^B. Finally, a rotation about the longitudinal axis 1^Y by the roll angle ϕ completes the transformation, placing 2^Y and 3^Y into their final body-aligned positions. Each of these rotations is represented by a unit axis transformation matrix. To sum it up, the yaw rotation is given by a transformation about the z-axis, the pitch rotation about the y-axis, and the roll rotation about the x-axis. Combining these gives the full transformation from geodetic to body coordinates:

$$[T]^{BD} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.25}$$

The sequence of rotations is important because rotation matrices do not commute, so changing the order generally leads to a different final orientation. Once the sequence is fixed, the matrix product can be evaluated with any grouping, since matrix multiplication is associative. Multiplying the three unit axis transformation matrices leads to the resulting matrix $[T]^{BD}$:

$$[T]^{BD} = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \cos\theta\sin\phi \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{3.26}$$

This matrix serves two purposes. It provides the elements from which the Euler angles are extracted from and initializes the transformation matrix from inertial to body coordinates, given by:

$$[T]^{BI} = [T]^{BD}[T]^{DI}. \tag{3.27}$$

Geodetic to Velocity Coordinates $[T]^{VD}$

The orientation of the flight-path coordinate system with respect to the local-level geodetic coordinate system $]^D$ is defined by two angles, the flight path angle γ and the heading angle χ. The heading angle is defined as the azimuth of the velocity vector projected onto the local horizontal plane, measured from geographic north toward east. The flight path angle gives the angle between the velocity vector and the local horizontal plane. When the velocity is ground-relative, these angles are written as γ_K and χ_K. If the velocity is air-relative, they are denoted γ_A and χ_A. Their angle range is given by [Zipfel, 2025]:

$$-\frac{\pi}{2} \leq \gamma \leq \frac{\pi}{2}, \quad 0 \leq \chi \leq 2\pi.$$

To construct the transformation from $]^D$ to $]^V$, the coordinate system is rotated such that its 1-axis becomes aligned with the velocity vector. First, a rotation by χ about the vertical axis 3^D aligns the intermediate axis 1^X with the projection of the velocity vector onto the local horizontal plane. Next, a rotation by γ about the intermediate axis 2^X tilts the system so that 1^V coincides with the full velocity vector. This leads to the transformation chain:

$$D \xrightarrow{\chi} X \xrightarrow{\gamma} V.$$

Substituting the corresponding elementary rotation matrices and applying them from right to left yields:

$$[T]^{VD} = \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix} \begin{bmatrix} \cos\chi & \sin\chi & 0 \\ -\sin\chi & \cos\chi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.28}$$

The complete transformation matrix from the geodetic coordinate system to the flight-path coordinate system is then given by:

$$[T]^{VD} = \begin{bmatrix} \cos\gamma\cos\chi & \cos\gamma\sin\chi & -\sin\gamma \\ -\sin\chi & \cos\chi & 0 \\ \sin\gamma\cos\chi & \sin\gamma\sin\chi & \cos\gamma \end{bmatrix}. \tag{3.29}$$

Geodetic to Wind Coordinates $[T]^{WD}$

The wind coordinate system $]^W$ is obtained by applying an additional rotation to the flight-path coordinate system $]^V$, which serves as an intermediate coordinate system. The bank angle μ is defined as the rotation about the velocity direction 1^V that aligns the flight-path coordinate system with the aircraft symmetry plane. More precisely, it is the angle between the plane spanned by 1^V and 3^V and the body symmetry plane spanned by 1^B and 3^B, measured about the axis 1^V. The bank angle is typically defined in the range [Boiffier, 2000]:

$$-\pi < \mu \leq \pi.$$

Using this definition, the wind coordinate system is obtained by rotating $]^V$ about 1^V by the angle μ. This leads to the transformation chain:

$$D \xrightarrow{\chi} X \xrightarrow{\gamma} V \xrightarrow{\mu} W.$$

Substituting the corresponding elementary rotation matrices and applying them from right to left yields:

$$[T]^{WD} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\mu & \sin\mu \\ 0 & -\sin\mu & \cos\mu \end{bmatrix} \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix} \begin{bmatrix} \cos\chi & \sin\chi & 0 \\ -\sin\chi & \cos\chi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.30}$$

The complete transformation matrix from the geodetic coordinate system to the wind coordinate system is then given by:

$$[T]^{WD} = \begin{bmatrix} \cos\chi\cos\gamma & \sin\chi\cos\gamma & -\sin\gamma \\ \cos\chi\sin\gamma\sin\mu - \sin\chi\cos\mu & \sin\chi\sin\gamma\sin\mu + \cos\chi\cos\mu & \cos\gamma\sin\mu \\ \cos\chi\sin\gamma\cos\mu + \sin\chi\sin\mu & \sin\chi\sin\gamma\cos\mu - \cos\chi\sin\mu & \cos\gamma\cos\mu \end{bmatrix}. \tag{3.31}$$

Body to Wind Coordinates $[T]^{WB}$

The orientation of the body coordinate system $]^B$ with respect to the wind coordinate system $]^W$ is described by two angles: the angle of attack α and the sideslip angle β. The angle of attack α is defined as the angle between the body axis 1^B and the projection of the velocity vector onto the 1^B–3^B plane, measured about the body axis 2^B. The sideslip angle β is defined as the angle between the velocity vector and its projection onto the 1^B–3^B plane, measured about the body axis 3^B. Their ranges are given by convention as [Boiffier, 2000]:

$$-\pi < \alpha \leq \pi, \quad -\frac{\pi}{2} \leq \beta \leq \frac{\pi}{2}.$$

To construct the transformation matrix, an intermediate stability coordinate system $]^S$ is introduced. The transformation proceeds in two steps. First, a rotation by $-\alpha$ about the body 2^B-axis aligns the 1^S-axis with the projection of the velocity vector in the symmetry plane. The negative sign follows from the definition of the angle of attack α, which is measured in the 1^B–3^B plane about the axis 2^B. Since the velocity vector is inclined by a positive angle α with respect to 1^B, the coordinate system must be rotated in the opposite direction to align the axis with this vector.

Then, a rotation by β about the stability 3^S-axis completes the alignment with the velocity direction, resulting in the wind coordinate system. This sequence can be written as:

$$B \xrightarrow{-\alpha} S \xrightarrow{\beta} W.$$

Substituting the corresponding elementary rotation matrices and applying them from right to left yields:

$$[T]^{WB} = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{3.32}$$

Multiplying the two transformation matrices gives the full transformation from body to wind coordinates:

$$[T]^{WB} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{3.33}$$

Body to Flight-Path Coordinates $[T]^{VB}$

The transformation from the body coordinate system $]^B$ to the flight-path coordinate system $]^V$ is completed by a rotation about the velocity direction. Starting from the wind coordinate system $]^W$, a rotation by the bank angle μ is applied about the axis 1^W. This rotation defines the orientation of the flight-path coordinate system about the velocity vector and determines the direction of the lateral axis 2^V relative to the local horizontal plane. The transformation sequence is:

$$B \xrightarrow{-\alpha} S \xrightarrow{\beta} W \xrightarrow{-\mu} V.$$

The complete transformation is then given by:

$$[T]^{VB} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\mu & -\sin\mu \\ 0 & \sin\mu & \cos\mu \end{bmatrix} \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{3.34}$$

Multiplying the three transformation matrices yields the transformation matrix from body to flight-path coordinates:

$$[T]^{VB} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta\cos\mu + \sin\alpha\sin\mu & \cos\beta\cos\mu & -\sin\alpha\sin\beta\cos\mu - \cos\alpha\sin\mu \\ -\cos\alpha\sin\beta\sin\mu - \sin\alpha\cos\mu & \cos\beta\ \sin\mu & -\sin\alpha\sin\beta\sin\mu + \cos\alpha\cos\mu \end{bmatrix}. \tag{3.35}$$

This completes the set of transformations between the relevant coordinate systems. Whether it is more appropriate to transform from the geodetic coordinate system to the wind coordinate system directly, or instead through the body coordinate system, depends on the specific application. The choice is guided by which states or measurements are available and which coordinate system provides the most meaningful physical interpretation for the task at hand.

Coordinate System Transformation Diagram

The coordinate systems used in flight dynamics are related through a sequence of passive, right-handed rotations. These transformations do not alter the physical vectors themselves but instead express them in different coordinate systems attached to various reference frames. Each rotation captures the relative orientation between frames such as inertial, Earth-fixed, geocentric, geodetic, and body-fixed.

Given the number and complexity of these transformations, they can be difficult to visualize intuitively. To aid understanding, a schematic diagram is provided in Fig. 3.7, which illustrates the relationships and transformation paths between the relevant coordinate systems used throughout this work. The direction of each arrow indicates a transformation from the coordinate system at the tail of the arrow to the coordinate system at its head, with the corresponding transformation matrix shown along the arrow. The inverse transformation is obtained by reversing the arrow direction, which corresponds to taking the transpose of the associated transformation matrix.

3.4. Nonlinear Dynamic Inversion (NDI)

The fundamental idea of NDI is to transform selected input–output channels into chains of integrators through model-based state transformation and feedback. This results in a decoupled, linearized system.

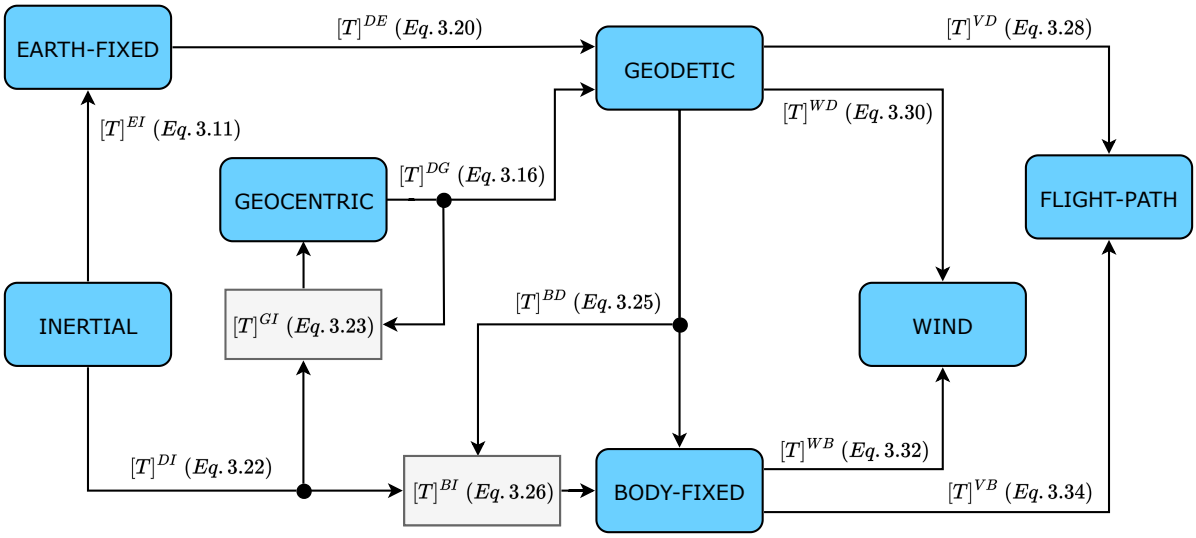


Figure 3.7: Relationships between coordinate systems used in the flight dynamics model.

While not a control technique in itself, NDI provides a framework on which linear control methods can be applied to shape the closed-loop behavior, including tracking performance, disturbance rejection, and noise attenuation [Slotine and Li, 1991].

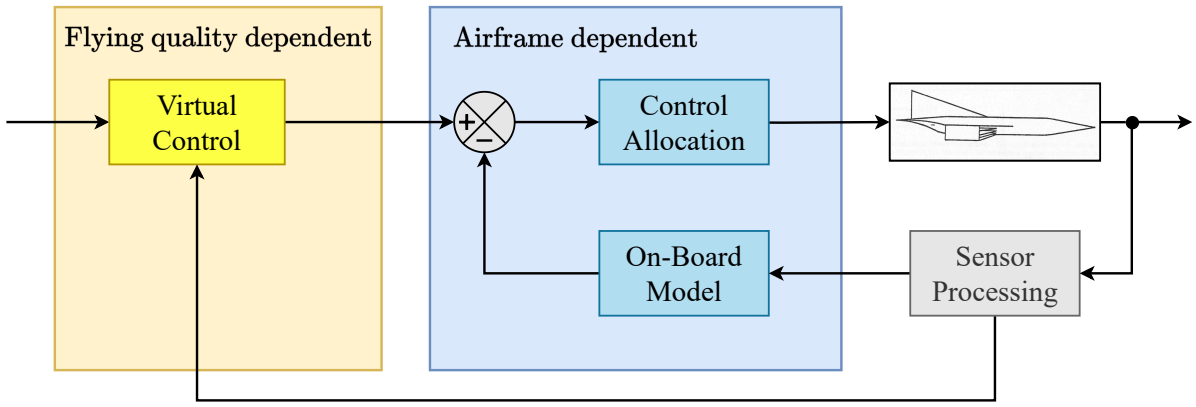


Figure 3.8: Basic NDI control structure adapted from [Pollack, 2024].

The basic NDI structure consists of an outer loop that computes the virtual control to achieve the desired flying qualities and an inner loop that executes this command by cancelling the airframe nonlinearities. This modular separation allows the outer loop to remain largely independent of the specific airframe dynamics, while the inner loop uses an On-Board Model (OBM) together with sensor feedback to perform the inversion and generate actuator commands via control allocation, as shown in Fig. 3.8. This structure enables the flying qualities design to be carried out in a largely decoupled manner, which is a key advantage of NDI and has supported its use in practical control system design [Pollack, 2024].

MIMO Input-Output Linearization

For the derivation of the NDI control law, the concept of the Lie derivative will be introduced first. The Lie derivative describes the rate of change of a smooth real-valued function along the flow of a smooth vector field. It provides a formal way to describe how the output of a nonlinear system evolves along the

system dynamics. Consider a nonlinear system of the form:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} \tag{3.36}$$

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}), \tag{3.37}$$

where $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^p$ are smooth vector fields. The matrix $\boldsymbol{G} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ is a smooth function mapping whose columns $\boldsymbol{g}_j$ are smooth vector fields.

The relationship between the number of outputs p and the number of inputs m determines the feasibility of exact input–output linearization. If $p < m$, the problem is overdetermined and requires a control allocation method. If $p > m$, the problem is underdetermined and exact tracking cannot be achieved. In the following derivations, the case $p = m$ is assumed so that an exact nonlinear inversion is possible.

If the elements of $\boldsymbol{h}$ are denoted as h_i, $i = 1, 2, \ldots, m$, and the column vectors of the matrix $\boldsymbol{G}$ are denoted as $\boldsymbol{g}_j$, $j = 1, 2, \ldots, m$, then the Lie derivatives ([Slotine and Li, 1991]) of the scalar functions h_i with respect to the vector fields $\boldsymbol{f}$ and $\boldsymbol{g}_j$ are defined as follows:

$$L_{\boldsymbol{f}} h_i(\boldsymbol{x}) = \frac{\partial h_i}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}), \qquad L_{\boldsymbol{g}_j} h_i(\boldsymbol{x}) = \frac{\partial h_i}{\partial \boldsymbol{x}} \boldsymbol{g}_j(\boldsymbol{x}). \tag{3.38}$$

Higher-order Lie derivatives along $\boldsymbol{f}$ are defined recursively as:

$$L_{\boldsymbol{f}}^k h_i(\boldsymbol{x}) = L_{\boldsymbol{f}} \left(L_{\boldsymbol{f}}^{k-1} h_i(\boldsymbol{x}) \right) = \nabla \left(L_{\boldsymbol{f}}^{k-1} h_i(\boldsymbol{x}) \right) \boldsymbol{f}(\boldsymbol{x}), \quad L_{\boldsymbol{f}}^0 h_i(\boldsymbol{x}) = h_i(\boldsymbol{x}), \tag{3.39}$$

where ∇ denotes the gradient with respect to the state vector $\boldsymbol{x}$. Finally, the Lie derivative of the k^{th} Lie derivative along $\boldsymbol{f}$, taken along $\boldsymbol{g}_j$, is:

$$L_{\boldsymbol{g}_j} L_{\boldsymbol{f}}^k h_i(\boldsymbol{x}) = \nabla \left(L_{\boldsymbol{f}}^k h_i(\boldsymbol{x}) \right) \cdot \boldsymbol{g}_j(\boldsymbol{x}). \tag{3.40}$$

A feedback linearization law can then be constructed for the system in Eq. (3.37) by taking repeated Lie derivatives of the controlled variable $\boldsymbol{y} \in \mathbb{R}^p$ until the input vector $\boldsymbol{u} \in \mathbb{R}^m$ appears explicitly in the dynamics. The number of differentiations required for each output y_i, $i = 1, \ldots, p$, is determined by its relative degree ρ_i.

Formally, the relative degree ρ_i of the i-th output channel $y_i = h_i(\boldsymbol{x})$ is the smallest integer ρ_i such that:

$$L_{\boldsymbol{g}_j} L_{\boldsymbol{f}}^k h_i(\boldsymbol{x}) = 0, \quad 0 \le k \le \rho_i - 1, \quad j = 1, \ldots, m, \tag{3.41}$$

and for $k = \rho_i$ there exists at least one j such that:

$$L_{\boldsymbol{g}_j} L_{\boldsymbol{f}}^{\rho_i - 1} h_i(\boldsymbol{x}) \neq 0. \tag{3.42}$$

In other words, ρ_i counts the number of successive differentiations of the output y_i along the vector field $\boldsymbol{f}$ before the input $\boldsymbol{u}$ appears explicitly through one of the input vector fields $\boldsymbol{g}_j$. Intuitively, the relative degree describes how "deeply" the control input is embedded in the system dynamics for each output channel. Collecting the relative degrees for all outputs yields the vector relative degree:

$$\boldsymbol{\rho} = [\rho_1 \;\; \rho_2 \;\; \ldots \;\; \rho_m]^\top, \tag{3.43}$$

and the total relative degree of the system is defined as:

$$\rho = \|\boldsymbol{\rho}\|_1 = \sum_{i=1}^{m} \rho_i \le n. \tag{3.44}$$

The inequality expresses that the number of independent output derivatives cannot exceed the order of the system n. The system is said to possess a *well-defined vector relative degree* on a region $\mathcal{D}_0$ when each ρ_i exists in the sense of Eqs. (3.41)–(3.42) and the control effectiveness matrix introduced below has full rank throughout $\mathcal{D}_0$. This is a purely local structural condition and constitutes the minimal requirement for the inversion of the dynamics to be well defined.

If the conditions for the existence of a vector relative degree $\boldsymbol{\rho}$ are satisfied and the control effectiveness matrix $\mathcal{B}(\boldsymbol{x})$ is nonsingular in a region of interest, then the external output dynamics of the system can be expressed as [Khalil, 2002]:

$$\begin{bmatrix} y_1^{(\rho_1)} \\ \vdots \\ y_m^{(\rho_m)} \end{bmatrix} = \underbrace{\begin{bmatrix} L_f^{\rho_1} h_1(\boldsymbol{x}) \\ \vdots \\ L_f^{\rho_m} h_m(\boldsymbol{x}) \end{bmatrix}}_{\boldsymbol{\alpha}(\boldsymbol{x})} + \underbrace{\begin{bmatrix} L_{\boldsymbol{g}_1} L_f^{\rho_1 - 1} h_1(\boldsymbol{x}) & \cdots & L_{\boldsymbol{g}_m} L_f^{\rho_1 - 1} h_1(\boldsymbol{x}) \\ \vdots & \ddots & \vdots \\ L_{\boldsymbol{g}_1} L_f^{\rho_m - 1} h_m(\boldsymbol{x}) & \cdots & L_{\boldsymbol{g}_m} L_f^{\rho_m - 1} h_m(\boldsymbol{x}) \end{bmatrix}}_{\mathcal{B}(\boldsymbol{x})} \cdot \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix}, \tag{3.45}$$

or in compact form:

$$\boldsymbol{y}^{(\rho)} = \boldsymbol{\alpha}(\boldsymbol{x}) + \mathcal{B}(\boldsymbol{x})\boldsymbol{u}, \tag{3.46}$$

with $\mathcal{B}(\boldsymbol{x}) \in \mathbb{R}^{p \times m}$ being the control effectiveness matrix and $\boldsymbol{\alpha}(\boldsymbol{x}) \in \mathbb{R}^p$ collecting the nonlinear dynamics not directly attributable to the control inputs. In the general case, the number of outputs p and the number of inputs m need not be equal. If $p < m$, the system is over-actuated: the matrix $\mathcal{B}(\boldsymbol{x})$ has more columns than rows, and a right pseudo-inverse or a control allocation scheme is typically used to distribute control effort among the actuators. If $p > m$, the system is under-actuated: there are more outputs than inputs, and it is not possible to simultaneously linearize all channels, so one must select a subset of outputs or use approximate methods.

The standard nonlinear dynamic inversion formulation assumes $p = m$, such that $\mathcal{B}(\boldsymbol{x}) \in \mathbb{R}^{m \times m}$ is square and nonsingular. Under this assumption, a control law can be established that reduces the nonlinear dynamics to a set of integrators:

$$\boldsymbol{u} = \hat{\mathcal{B}}^{-1}(\boldsymbol{x})\big[\boldsymbol{\nu} - \hat{\boldsymbol{\alpha}}(\boldsymbol{x})\big], \tag{3.47}$$

where $\hat{\mathcal{B}}^{-1}(\boldsymbol{x})$ and $\hat{\boldsymbol{\alpha}}(\boldsymbol{x})$ are OBM representations of $\mathcal{B}^{-1}(\boldsymbol{x})$ and $\boldsymbol{\alpha}(\boldsymbol{x})$, respectively. The control effectiveness matrix $\mathcal{B}(\boldsymbol{x})$ is necessarily assumed to be invertible. Moreover, $\boldsymbol{\nu} \in \mathbb{R}^m$ represents the virtual control signal generated by an outer-loop controller. This term ensures stability of the external dynamics and leads to improved performance and robustness properties. Returning the input as defined in Eq. (3.47) to Eq. (3.46) yields:

$$\boldsymbol{y}^{(\rho)} = \boldsymbol{\alpha}(\boldsymbol{x}) + \mathcal{B}(\boldsymbol{x})\hat{\mathcal{B}}^{-1}(\boldsymbol{x})\big[\boldsymbol{\nu} - \hat{\boldsymbol{\alpha}}(\boldsymbol{x})\big]. \tag{3.48}$$

In the absence of modeling uncertainty and disturbances, the static feedback law leads to the linearized map:

$$\boldsymbol{y}^{(\rho)} = \boldsymbol{\nu}. \tag{3.49}$$

Whenever a vector relative degree exists, this construction renders the input–output map linear; the system is then said to be *input–output linearizable*. Note that this property concerns only the map from $\boldsymbol{\nu}$ to $\boldsymbol{y}$: if $\rho < n$, the inversion leaves $n - \rho$ states unaccounted for, a point examined in the following subsection.

By considering the transformation of the nonlinear system Eq. (3.49) in the frequency domain, the relationship between the pseudo-controls $\boldsymbol{\nu}$ and the system outputs $\boldsymbol{y}$ can be expressed as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \frac{1}{s^{\rho_1}} & 0 & \cdots & 0 \\ 0 & \frac{1}{s^{\rho_2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{s^{\rho_m}} \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_m \end{bmatrix}, \tag{3.50}$$

The linearized system exhibits a decoupled structure in which each output component y_i evolves as a ρ_i-fold integral of its corresponding pseudo-control input ν_i. This chain of integrators separates the pseudo-controls $\boldsymbol{\nu}$ from the outputs $\boldsymbol{y}$ in each input/output channel, allowing the designer to directly specify the desired tracking dynamics through the selection of $\boldsymbol{\nu}$.

Figure 3.9 shows the control architecture for a Single-Input, Single-Output (SISO) NDI system. The flying-quality-dependent linear controller generates the virtual control ν from the tracking error, while the airframe-dependent inversion block computes the commanded input u^{cmd} using the model-based control law in Eq. (3.47). The commanded signal passes through the actuator dynamics $A(s)$ before reaching the plant. When the actuator dynamics are sufficiently fast, such that $A(s) \approx 1$, the overall behavior approximates an ideal integrator with $y = \frac{1}{s}\nu$.

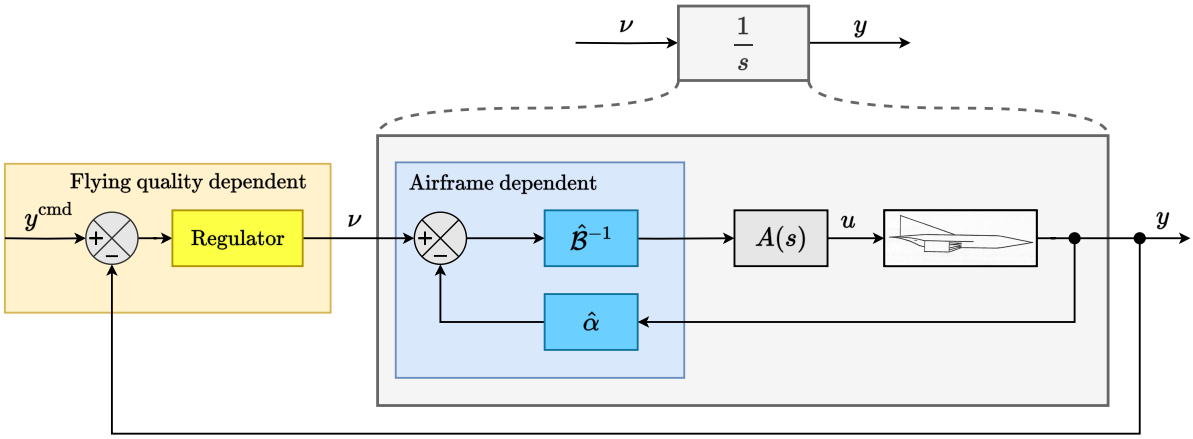


Figure 3.9: Ideal SISO NDI control structure.

Internal Dynamics

The strongest of these structural properties is *full-state feedback linearizability*, which holds when the total relative degree $\rho = \sum_{i=1}^{p} \rho_i$ equals the dimension of the state vector n. In this case the input–output dynamics fully capture the system behavior, and the feedback linearization law derived above reduces the nonlinear dynamics to a set of integrator chains without leaving any unmodeled dynamics. If, however, $\rho < n$, then not all states are reflected in the output derivatives. The remaining $n - \rho$ states evolve according to their own dynamics, which are not directly influenced by the input–output linearization. These are referred to as the internal dynamics of the system.

To characterize the internal and external dynamics, input–output linearization introduces a coordinate transformation $\boldsymbol{z} = \mathcal{T}(\boldsymbol{x}) = [\boldsymbol{\xi}\ \boldsymbol{\eta}]^\top$, where the external states $\boldsymbol{\xi}$ are formed from the outputs and their

successive derivatives up to order $(\rho_i - 1)$:

$$\begin{aligned}
\xi_1^1 &:= h_1(\boldsymbol{x}), \quad \xi_2^1 := L_f h_1(\boldsymbol{x}), \quad \cdots, \quad \xi_{\rho_1}^1 := L_f^{\rho_1 - 1} h_1(\boldsymbol{x}), \\
\xi_1^2 &:= h_2(\boldsymbol{x}), \quad \xi_2^2 := L_f h_2(\boldsymbol{x}), \quad \cdots, \quad \xi_{\rho_2}^2 := L_f^{\rho_2 - 1} h_2(\boldsymbol{x}), \\
&\vdots \qquad\qquad \vdots \qquad\qquad \ddots \qquad\qquad \vdots \\
\xi_1^m &:= h_m(\boldsymbol{x}), \quad \xi_2^m := L_f h_m(\boldsymbol{x}), \quad \cdots, \quad \xi_{\rho_m}^m := L_f^{\rho_m - 1} h_m(\boldsymbol{x}).
\end{aligned} \tag{3.51}$$

In parallel, the internal states are defined as smooth functions $\boldsymbol{\eta} = \boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^{n-\rho}$ that satisfy:

$$\frac{\partial \phi_i}{\partial \boldsymbol{x}} \boldsymbol{g}_j(\boldsymbol{x}) = 0, \quad 1 \le i \le n - \rho, \quad 1 \le j \le m, \quad \forall \boldsymbol{x} \in \mathcal{D}_0, \tag{3.52}$$

which ensures that the internal dynamics are unaffected by the control input. Such functions are guaranteed to exist only locally, and the resulting coordinate transformation $\mathcal{T}$ is a valid diffeomorphism on a neighborhood $\mathcal{D}_0$ of the operating point [Isidori, 1985].

This transformation maps the system into the normal form [Isidori, 1985]:

$$\begin{aligned}
\dot{\boldsymbol{\xi}} &= \boldsymbol{A}_c \boldsymbol{\xi} + \boldsymbol{B}_c \left[\boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x}) \boldsymbol{u} \right] \Big|_{\boldsymbol{x} = \mathcal{T}^{-1}(\boldsymbol{z})}, \\
\dot{\boldsymbol{\eta}} &= \boldsymbol{f}_c(\boldsymbol{\xi}, \boldsymbol{\eta}) = \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) \Big|_{\boldsymbol{x} = \mathcal{T}^{-1}(\boldsymbol{z})}, \\
\boldsymbol{y} &= \boldsymbol{C}_c \boldsymbol{\xi}.
\end{aligned} \tag{3.53}$$

where the triplet (A_c, B_c, C_c) is in Brunovsky block canonical form:

$$\boldsymbol{A}_c = \operatorname{diag}\{A_0^i\}, \quad \boldsymbol{B}_c = \operatorname{diag}\{B_0^i\}, \quad \boldsymbol{C}_c = \operatorname{diag}\{C_0^i\}, \quad i = 1, \ldots, m, \tag{3.54}$$

and $(\boldsymbol{A}_0^i, \boldsymbol{B}_0^i, \boldsymbol{C}_0^i)$ is a canonical form representation of a chain of ρ_i integrators:

$$\boldsymbol{A}_0^i := \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 1 \\ 0 & \cdots & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{B}_0^i := \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \quad \boldsymbol{C}_0^i := \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}, \quad i = 1, \ldots, m. \tag{3.55}$$

For each input–output channel i, a feedback law for ν_i can be designed to drive its associated linearized chain of integrators to zero. In other words, the external dynamics can be stabilized through appropriate feedback design. However, since the internal dynamics are not directly influenced by the control input, they cannot be stabilized by the control law in Eq. (3.47). Their stability must therefore be analyzed separately.

To guarantee the stability of the overall closed-loop system, the remaining internal dynamics must also be stable when the external coordinates are constrained to zero [Khalil, 2002, p. 517]. This condition corresponds to the so-called *zero dynamics*, obtained by enforcing $\boldsymbol{\xi} = 0$ in Eq. (3.53), which yields:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{f}_c(0, \boldsymbol{\eta}). \tag{3.56}$$

The zero dynamics describe the evolution of the internal states that are unobservable from the output when the output and all its time derivatives are identically zero for $t \ge 0$. In the linear case, the eigenvalues of $\boldsymbol{f}_c(0, \boldsymbol{\eta})$ coincide with the transmission zeros of the open-loop transfer function. If these zero dynamics are asymptotically stable within the operating region, the system is said to be

minimum phase. Conversely, if any zero lies in the open right-half of the complex plane, the system is *non-minimum phase* and possesses unstable zero dynamics.

This instability can be understood by observing that feedback linearization effectively performs a mathematical inversion of the system's input–output mapping. The linear case offers a useful illustration: inverting a transfer function $G(s) = N(s)/D(s)$ yields $G^{-1}(s) = D(s)/N(s)$, which exchanges its poles and zeros, so that a zero in the right-half plane becomes a pole of the inverted system and directly causes internal instability. In the nonlinear case the corresponding role is played by the zero dynamics of Eq. (3.56). If these zero dynamics are unstable, any attempt to enforce perfect output tracking through inversion excites divergent internal modes. Consequently, the feedback-linearized model is valid and stabilizing only when the underlying system is minimum phase [Khalil, 2002].

One can therefore summarize that for standard NDI to be applicable, the system must satisfy the following conditions:

1. The system must have a well-defined vector relative degree, i.e. the control effectiveness matrix $\mathcal{B}(\boldsymbol{x})$ formed from Lie derivatives must be nonsingular in the region of interest.

2. The zero dynamics must be globally asymptotically stable with respect to the origin, or at least bounded-input bounded-state stable.

Linear Error Feedback Control Design

The feedback control law defined in Eq. (3.47) can be inserted into the nonlinear system representation in Eq. (3.37), which yields the input–output dynamics given by Eq. (3.49). In this formulation, the ρ_i-th derivative of each output y_i becomes equal to the corresponding pseudo-control variable ν_i. Consequently, the pseudo-controls ν_i can be designed as functions of the tracking error to ensure that the actual outputs y_i follow the desired reference trajectories $y_{r,i}$. The control objective is therefore to minimize the error:

$$e_i = y_{r,i} - y_i. \tag{3.57}$$

A common and straightforward way to specify the pseudo-control ν_i is by applying a linear feedback law. In this case, the pseudo-control is defined as a weighted combination of the error and its derivatives up to order $\rho_i - 1$:

$$\nu_i = (y_{r,i})^{(\rho_i)} + k_{i,0}(y_{r,i} - y_i) + k_{i,1}(\dot{y}_{r,i} - \dot{y}_i) + \ldots + k_{i,\rho_i-1}(y_{r,i}^{(\rho_i-1)} - y_i^{(\rho_i-1)}), \tag{3.58}$$

or, equivalently,

$$\nu_i = y_{r,i}^{(\rho_i)} + \sum_{j=0}^{\rho_i-1} k_{i,j} e_i^{(j)}. \tag{3.59}$$

If the feedback gains $k_{i,j}$ are chosen to be positive, the closed-loop error dynamics become:

$$e_i^{(\rho_i)} = -k_{i,0}e_i - k_{i,1}\dot{e}_i - \ldots - k_{i,\rho_i-1}e_i^{(\rho_i-1)}, \tag{3.60}$$

which represents a linear differential equation with constant coefficients. The characteristic polynomial associated with this equation can be selected to ensure that all roots have negative real parts, leading to exponential convergence of the tracking error. Substituting the feedback law Eq. (3.59) into the input–output linearized model produces the final NDI-based control law:

$$\boldsymbol{u} = \hat{\mathcal{B}}^{-1}(\boldsymbol{x}) \begin{bmatrix} y_{r,1}^{(\rho_1)} + \sum_{j=0}^{\rho_1-1} k_{1,j} e_1^{(j)} \\ \vdots \\ y_{r,m}^{(\rho_m)} + \sum_{j=0}^{\rho_m-1} k_{m,j} e_m^{(j)} \end{bmatrix} - \hat{\boldsymbol{\alpha}}(\boldsymbol{x}). \tag{3.61}$$

Robustness Properties

The major dynamic inversion robustness issues are exhibited by replacing the nominal aircraft model in Eq. (3.37) with a perturbed model:

$$[\boldsymbol{y}^{(\rho)}] = [\hat{\boldsymbol{\alpha}}(\boldsymbol{x}) + \boldsymbol{\xi}(\boldsymbol{x})] + \left[\hat{\mathcal{B}}(\boldsymbol{x}) + \boldsymbol{\Xi}(\boldsymbol{x})\right] \boldsymbol{u}, \tag{3.62}$$

where $\hat{\boldsymbol{\alpha}}(\boldsymbol{x})$ and $\hat{\mathcal{B}}(\boldsymbol{x})$ are the nominal OBM drift and control effectiveness terms, and $\boldsymbol{\xi}(\boldsymbol{x})$ and $\boldsymbol{\Xi}(\boldsymbol{x})$ represent the corresponding perturbations in the drift and input terms, respectively. These are assumed to be regular (i.e. state-dependent but free of fast parasitic dynamics) and bounded on the region of interest. In addition, the ideal control effector position in Eq. (3.47) is replaced by a perturbed value obtained by passing the ideal position through actuator dynamics, flexible structural elements, and other high-frequency uncertainties:

$$\boldsymbol{u} = (\boldsymbol{I} + \boldsymbol{\Delta})\, \hat{\mathcal{B}}^{-1}(\boldsymbol{x})\, [\boldsymbol{\nu} - \hat{\boldsymbol{\alpha}}(\boldsymbol{x})]\,, \tag{3.63}$$

where $\boldsymbol{\Delta}(s)$ is an arbitrary stable dynamic perturbation, small for low-frequency signals but increasing in magnitude towards unity and beyond as frequency increases. It is likewise assumed to be bounded at every frequency. A short derivation shows that the resulting dynamic model for $\boldsymbol{y}$ is then given by [Enns et al., 1994]:

$$\boldsymbol{y}^{(\rho)} = [\boldsymbol{\xi}(\boldsymbol{x}) - \mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})\hat{\boldsymbol{\alpha}}(\boldsymbol{x})] + [\boldsymbol{I} + \mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})]\, \boldsymbol{\nu}, \tag{3.64}$$

where the term $\mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})$ is given by:

$$\mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta}) = \boldsymbol{\Xi}(\boldsymbol{x})\hat{\mathcal{B}}^{-1}(\boldsymbol{x}) + \hat{\mathcal{B}}(\boldsymbol{x})\boldsymbol{\Delta}\hat{\mathcal{B}}^{-1}(\boldsymbol{x}) + \boldsymbol{\Xi}(\boldsymbol{x})\boldsymbol{\Delta}\hat{\mathcal{B}}^{-1}(\boldsymbol{x}). \tag{3.65}$$

These equations replace the integrators in Eq. (3.49) as a new dynamic model for the controlled variables. Note that there are two major uncertainty terms. The first term, $[\boldsymbol{\xi}(\boldsymbol{x}) - \mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})\hat{\boldsymbol{\alpha}}(\boldsymbol{x})]$, is a direct disturbance input to the integrators, while the second term, $[\boldsymbol{I} + \mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})]\, \boldsymbol{\nu}$, is a multiplicative perturbation on the control inputs of the integrators. Clearly the new system is not necessarily a linear system anymore. Furthermore, a different insight can be reached by writing the closed loop as [Pollack and van Kampen, 2023]:

$$\boldsymbol{y}^{(\rho)} = \boldsymbol{\nu} + [\boldsymbol{\xi}(\boldsymbol{x}) + \mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})\, (\boldsymbol{\nu} - \hat{\boldsymbol{\alpha}}(\boldsymbol{x}))] \triangleq \boldsymbol{\nu} + \boldsymbol{\epsilon}_{\text{NDI}}(\boldsymbol{x}, \boldsymbol{\nu}, \boldsymbol{\Delta}). \tag{3.66}$$

Because $\boldsymbol{\xi}$, $\boldsymbol{\Xi}$, and $\boldsymbol{\Delta}$ are all bounded as assumed above and $\hat{\mathcal{B}}^{-1}(\boldsymbol{x})$ is well defined on the region of interest, the term $\mathcal{D}(\boldsymbol{x}, \boldsymbol{\Delta})$ is itself bounded, and consequently $\boldsymbol{\epsilon}_{\text{NDI}}$ admits an upper bound at a given state $\boldsymbol{x}$ under bounded virtual control $\boldsymbol{\nu}$. This means the inversion error does not grow without bound. However, both regular perturbations, such as modeling errors, and singular perturbations, such as actuator dynamics and delays, directly increase this residual. In practice, this can lead to noticeable errors and reflects limited robustness, even though the error remains bounded under the given assumptions.

To sum it up, although this control law has appealing analytical properties, it depends on the structural assumptions introduced in the previous section. Exact inversion requires a well-defined relative degree and stable zero dynamics. When these conditions are not met, for instance in non-minimum phase systems, exact inversion is no longer possible. In practice, these requirements are not always satisfied, and the problem must be addressed through a different choice of outputs or an alternative control design. Furthermore, the implementation relies on an accurate OBM and access to the required system states. This is expensive and not always possible. Model inaccuracies lead to incomplete cancellation of the nonlinear dynamics, resulting in residual terms that affect the desired behavior. When full state measurements are not available, observers must be used, and their dynamics can further influence the closed-loop response.

To address these limitations, NDI is often combined with additional robustness measures in aerospace applications, for example through $\mathcal{H}_\infty$ loop shaping Franco et al. [2006], Papageorgiou and Polansky [2009]. Another approach is incremental NDI, which replaces explicit model cancellation with a local, measurement-based update using control effectiveness and measured state derivatives, thereby reducing reliance on the nonlinear model [Sieberling et al., 2010].

3.5. Incremental Nonlinear Dynamic Inversion (INDI)

In NDI control, the goal is to cancel the nonlinear terms $\boldsymbol{\alpha}(\boldsymbol{x})$ and $\boldsymbol{\mathcal{B}}(\boldsymbol{x})$ exactly through the control input. In practice, achieving such exact cancellation is nearly impossible because of model simplifications, numerical computation errors, and parametric uncertainty. INDI control addresses this issue by reducing the dependence of the control law on the system model and replacing part of the model information with additional sensor measurements.

INDI is derived by taking the first-order Taylor expansion of the output dynamics in Eq. (3.46) around the previous state $\boldsymbol{x}_0$ and control input $\boldsymbol{u}_0$ at time $t - \Delta t$, denoted by the subscript 0, where Δt is the sampling interval:

$$\begin{aligned}\boldsymbol{y}^{(\rho)} &= \boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u} \\ &= \boldsymbol{y}_0^{(\rho)} + \left.\frac{\partial\left[\boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u}\right]}{\partial \boldsymbol{x}}\right|_0 (\boldsymbol{x} - \boldsymbol{x}_0) + \left.\frac{\partial\left[\boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u}\right]}{\partial \boldsymbol{u}}\right|_0 (\boldsymbol{u} - \boldsymbol{u}_0) + O(\Delta \boldsymbol{x}^2) \\ &= \boldsymbol{y}_0^{(\rho)} + \left.\frac{\partial\left[\boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u}\right]}{\partial \boldsymbol{x}}\right|_0 \Delta \boldsymbol{x} + \boldsymbol{\mathcal{B}}(\boldsymbol{x}_0)\Delta \boldsymbol{u} + O(\Delta \boldsymbol{x}^2),\end{aligned} \tag{3.67}$$

where $\Delta \boldsymbol{x}$ and $\Delta \boldsymbol{u}$ denote the state and control increments over one sampling interval Δt, and $O(\Delta \boldsymbol{x}^2)$ represents higher-order terms. INDI employs a time-scale separation assumption in the design of the control input $\Delta \boldsymbol{u}$. Specifically, it is assumed that over one sampling interval the state increment $\Delta \boldsymbol{x}$ produces a negligible change in the dynamics compared with that produced by the control increment $\Delta \boldsymbol{u}$:

$$\left\| \left.\frac{\partial\left[\boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\mathcal{B}}(\boldsymbol{x})\boldsymbol{u}\right]}{\partial \boldsymbol{x}}\right|_0 \Delta \boldsymbol{x} \right\| \ll \left\| \boldsymbol{\mathcal{B}}(\boldsymbol{x}_0)\, \Delta \boldsymbol{u} \right\|. \tag{3.68}$$

This is usually justified by the cascaded structure of the dynamics. With an instantaneous actuator, a deflection produces the aerodynamic moments algebraically, and these enter the angular accelerations $\dot{p}, \dot{q}, \dot{r}$ directly. The rates p, q, r lie one integration downstream, so over a sampling interval they change only by $\Delta \boldsymbol{x} \approx \dot{\boldsymbol{x}}\, \Delta t$. The control increment thus acts on the output at full magnitude immediately, whereas the induced state increment is first-order small in Δt. For a sufficiently small sampling interval it can therefore be disregarded together with the higher-order terms [Bacon and Ostroff, 2000], leaving:

$$\boldsymbol{y}^{(\rho)} = \boldsymbol{y}_0^{(\rho)} + \boldsymbol{\mathcal{B}}(\boldsymbol{x}_0)\Delta \boldsymbol{u}. \tag{3.69}$$

Shorter sampling intervals only improve this separation, and in the limit $\Delta t \to 0$ the neglected term in Eq. (3.68) vanishes entirely. It weakens, however, when the controlled states are themselves fast: a low-inertia subsystem produces large angular accelerations, so the rate can drift substantially within one interval and the left-hand side in Eq. (3.68) is no longer small relative to the control increment. The separation then breaks down and the simplification introduces large residuals, a point that becomes important for the lateral dynamics examined later.

The control law is completed by adding the control vector $\boldsymbol{u}_0$ to the incremental input term:

$$\boldsymbol{u} = \boldsymbol{u}_0 + \hat{\boldsymbol{\mathcal{B}}}^{-1}(\boldsymbol{x}_0)[\boldsymbol{\nu} - \boldsymbol{y}_0^{(\rho)}]. \tag{3.70}$$

Figure 3.10 shows the complete block diagram of an INDI controller for a SISO system. The inner loop computes the incremental control action using delayed measurements of the input and output, while the outer loop determines the virtual control according to the desired flying qualities[2].

[2]Although the derivation above is expressed in continuous time for clarity, INDI is fundamentally a discrete-time control law. The quantities $\boldsymbol{u}_0$ and $\boldsymbol{y}_0$ correspond to the control input and output measured at the previous sampling instant $t - \Delta t$. This introduces an inherent one-sample delay, represented as a unit delay z^{-1} in the discrete domain. Continuous-time diagrams often omit this block to highlight the control structure, but when modeling a digital implementation, these delays must be included explicitly.

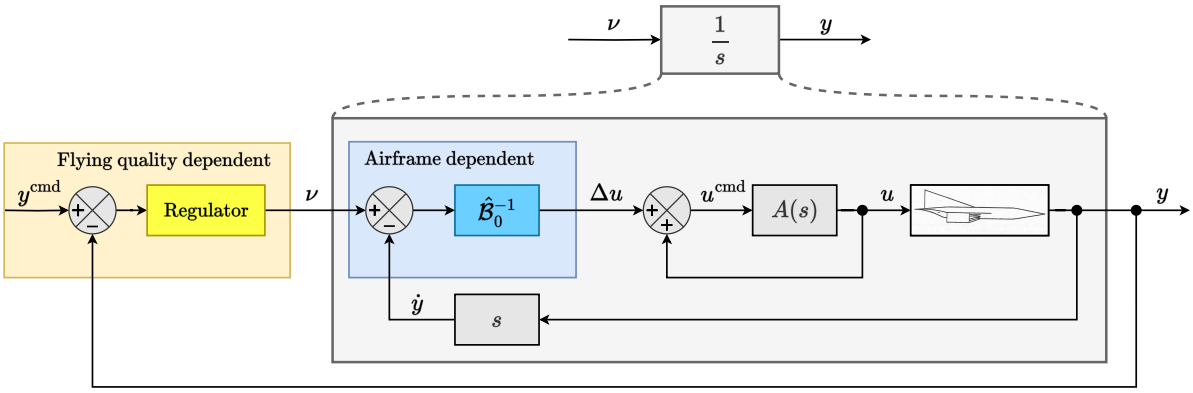


Figure 3.10: Ideal SISO INDI control structure.

Compared to the non-incremental formulation in Eq. (3.47), INDI replaces the model-dependent term $\boldsymbol{\alpha}(\boldsymbol{x})$ with direct sensor measurements of the previous control input and the derivative of the controlled variable. This shift explains why INDI is often referred to as a sensor-based (SB) control method. By inspection the comparison can be made that:

$$\boldsymbol{\alpha}(\boldsymbol{x}) = \boldsymbol{y}_0^{(\rho)} - \boldsymbol{\mathcal{B}}(\boldsymbol{x}_0)\boldsymbol{u}_0. \tag{3.71}$$

However, this equality only holds in the absence of model uncertainty and external disturbances, which introduces a fundamental distinction between model-based and SB inversion schemes. This difference leads to distinct robustness characteristics [Pollack and van Kampen, 2023]. In particular, SB INDI implementations tend to exhibit elevated open-loop gains at low frequencies, making them more effective at rejecting slowly varying uncertainties such as aerodynamic modeling errors. This helps explain why INDI is often considered more robust than conventional NDI. However, the associated high crossover frequencies can degrade robustness at higher frequencies unless proper roll-off is enforced.

Another important advantage is that the INDI approach is not limited to input-affine systems of the form Eq. (3.37). By leveraging a Taylor expansion around recent measurements, the method can also be extended to handle non-input-affine dynamics [Bacon and Ostroff, 2000].

Obtaining u_0 in INDI

In SISO INDI, the incremental control command is defined in as:

$$u^{\text{cmd}} = u_0 + \Delta u, \tag{3.72}$$

where u_0 denotes the actuator deflection from the previous timestep. Since the accuracy of u_0 directly influences the control law, several approaches exist to obtain this signal.

1. **Delayed command approximation.** When actuator position measurements are unavailable, u_0 can be approximated by delaying the commanded input by one sample, z^{-1}. This approach assumes that within one sample time the commanded deflection can be achieved. It is efficient and removes the need for any actuator modeling or sensors but is suitable only for very fast actuators [Azinheira et al., 2015].

2. **Direct measurement.** This method relies on dedicated sensors to measure actuator deflections directly, such that u_0 reflects the actual actuator state rather than a model-based approximation. While this can provide high accuracy, it depends on measurement quality. In practice, noise and structural vibrations are fed directly into the control loop, often requiring filtering which adds delay [Smith and Berry, 2000].

3. **Actuator model.** A more accurate alternative is to use a dynamic or predictive actuator model to estimate u_0. This approach captures bandwidth limitations and provides a realistic representation of actuator behavior, even without direct measurements. However, differences between the model and the actual actuator must be evaluated carefully to avoid errors such as integrator wind-up [Smeur et al., 2016a].

4. **Model–measurement combination.** A hybrid approach combines actuator measurements with model predictions. The model provides a smooth and continuous estimate of actuator dynamics, while the measurements correct for drift and modeling errors. This method improves robustness in the presence of noise and uncertainties but introduces higher implementation complexity [Myschik et al., 2022].

In this work, $\boldsymbol{u}_0$ is obtained from a dynamic actuator model (method 3), which provides a continuous estimate of the actuator state and captures its bandwidth limitations without requiring dedicated deflection sensors.

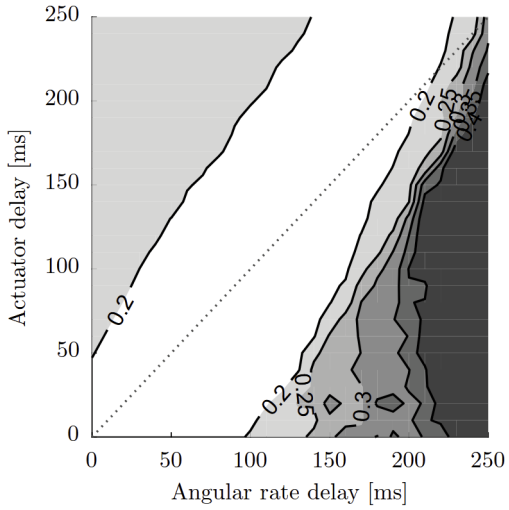


Figure 3.11: RMS tracking error for INDI as a function of actuator and angular rate delays [van't Veld et al., 2018].

Filter and Sensor Delay Synchronization

A distinctive feature of INDI is that angular acceleration is used as feedback. Since it cannot usually be measured directly, it must be estimated from angular rate measurements. These measurements are affected by sensor dynamics, and because angular acceleration must be estimated by differentiating noisy rate signals, filtering is applied to suppress noise, at the cost of introducing additional delay. At the same time, actuator commands are also subject to dynamics. If these two signal paths are not synchronized, the controller may respond too early or too late, leading to oscillations or even instability.

Stability analyses have shown that the system achieves its largest stability margin when actuator signals and angular acceleration feedback are subjected to the same delay [van't Veld et al., 2018]. Synchronization therefore improves stability, but only up to a limit: if the common delay becomes too large, the system will still become unstable. This limitation can be understood from fundamental constraints on feedback systems. As shown by Stein [2003], all control performance must be achieved within a finite available bandwidth determined by sensing, actuation, and structural dynamics. A time delay introduces phase lag that effectively reduces this available bandwidth. For unstable systems, stabilization requires that the available bandwidth be sufficiently large compared to the unstable pole. As the delay increases, the available bandwidth decreases until it becomes comparable to the unstable dynamics. At this point, the constraints imposed by Bode's integral prevent any controller from maintaining sufficient stability margins, and the closed-loop system becomes unstable. Stein [2003] further emphasizes that no controller can overcome these fundamental limitations, as they arise from the inherent physics of feedback systems rather than from the design method itself. Figure 3.11, adapted from van't Veld et al. [2018], illustrates that the best performance occurs along the diagonal where both delays are equal, while the Root-Mean-Square (RMS) tracking error grows more rapidly with angular rate delay. This highlights the importance of synchronizing delays and minimizing sensor-path latency in practical INDI implementations.

To address the delay introduced by filtering, Smeur et al. [2016a] proposed a synchronization method in which the same filter $H(s)$ used to estimate angular acceleration from the angular rate is also applied to the actuator feedback signal. This ensures that both signals entering the INDI control law experience identical delay, preserving their relative timing and maintaining stability. Figure 3.12 illustrates this concept: the angular rate is filtered by $H(s)$ and differentiated with s, while the actuator command u is passed through an equivalent filter $\bar{H}(s)$, which may differ in dimension from $H(s)$. Under these synchronized conditions, the closed-loop transfer function from the commanded pseudo-control input ν to the system output matches the actuator dynamics $A(s)/s$, restoring the intended INDI behavior.

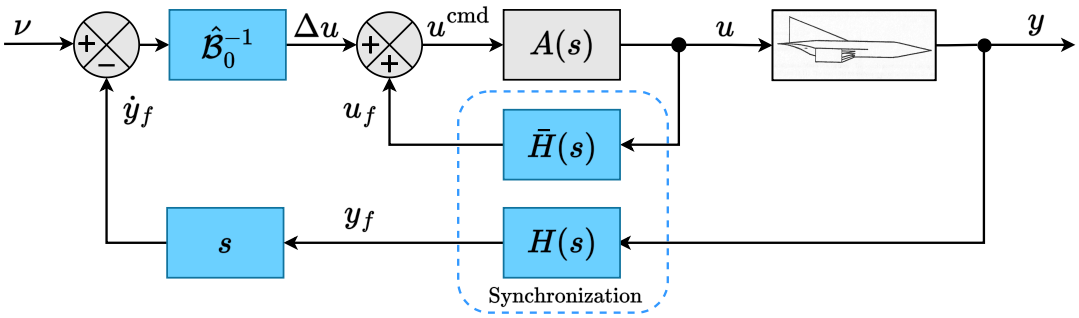


Figure 3.12: INDI control loop with synchronized filtering for delay compensation [Lu et al., 2016].

A limitation with synchronized filtering in the actuator feedback path is that in the MIMO case, either the condition $\bar{H}(s) = H(s)$ must be satisfied or all entries of $\bar{H}(s)$ and $H(s)$ must be identical, so that $\bar{H}(s)$ matches $H(s)$ elementwise, even with different dimensions. This implies that in the presence of a dimension mismatch, different filters cannot be applied independently to each axis [Steffensen et al., 2023a]. To avoid this restriction, a reformulated structure removes the filter from the actuator feedback path while preserving the same closed-loop behavior. This alternative formulation is shown in Fig. 3.13.

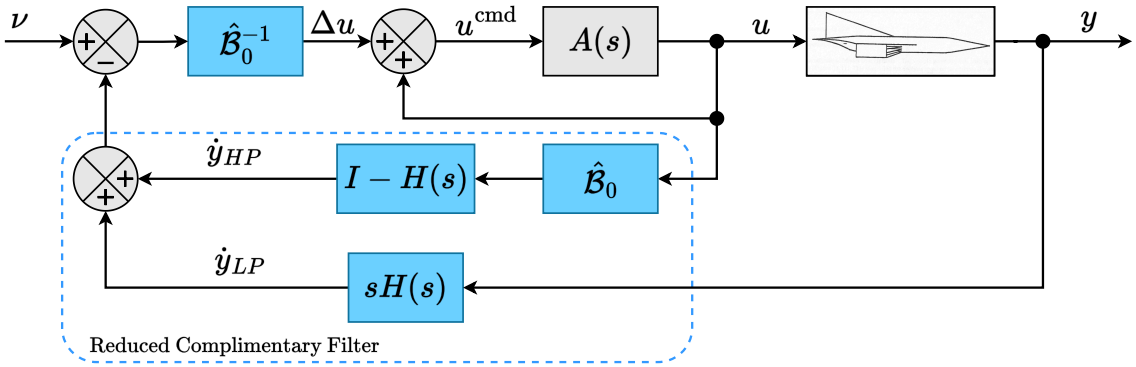


Figure 3.13: Alternative INDI control loop with synchronized filtering for delay compensation [Steffensen et al., 2023a].

This version also corresponds to INDI with a complementary filter, where only the high-frequency content of $\dot{y}$ is used to estimate $\dot{y}_{\text{OBM}} = \mathcal{B}u$, allowing per-axis filtering in multi-channel control systems.

Sensor Dynamics

The problem of delay synchronization becomes more involved once the dynamics of the sensors themselves are considered. Rate gyros and other sensors introduce dynamics into the measurement path. These effects act as an extra source of delay that must be accounted for in the INDI design.

A pragmatic solution, proposed by Grondman et al. [2018], is to experimentally determine the effective delay induced by the sensors and then compensate for it by inserting an equivalent pure delay in the actuator path. Since INDI tolerates delays on the actuator side better than on the sensor side, van't Veld et al. [2018] recommend adding a slightly larger delay than measured to ensure safe synchronization. This method is simple to implement and does not require detailed sensor models, but it can be conservative and may unnecessarily reduce performance. More rigorous approaches have been introduced by Kumtepe et al. [2022] and Steffensen et al. [2023a], where the sensor path is explicitly modeled as a chain of filters. In the latter work, the sensor dynamics are represented as:

$$F_{cy}(s) = F(s)D_y(s)S(s), \tag{3.73}$$

with three distinct contributions. The term $F(s)$ represents a potential roll-off or notch filters used to attenuate noise, engine vibrations, and structural mode excitations. The term $D_y(s)$ captures sensor measurement delays. Finally, $S(s)$ models the intrinsic sensor dynamics, which almost always include an anti-aliasing filter. This anti-aliasing is required because sensor measurements are sampled digitally in the flight control computer. Without such pre-filtering, high-frequency components above the Nyquist frequency would fold back into the measured bandwidth, corrupting the control signals with spurious low-frequency artifacts. All these filters unavoidably introduce additional phase lag, which must be synchronized with the actuator commands to preserve stability and performance. To achieve this, Steffensen et al. [2023a] applied the same chain of filters to the actuator feedback path, in the same way that noise filtering is handled. Figure 3.14 illustrates the latter architecture, where the complementary filter $H(s)$ and synchronization through $F_{cy}(s)$ align all control paths and preserve closed-loop stability and performance.

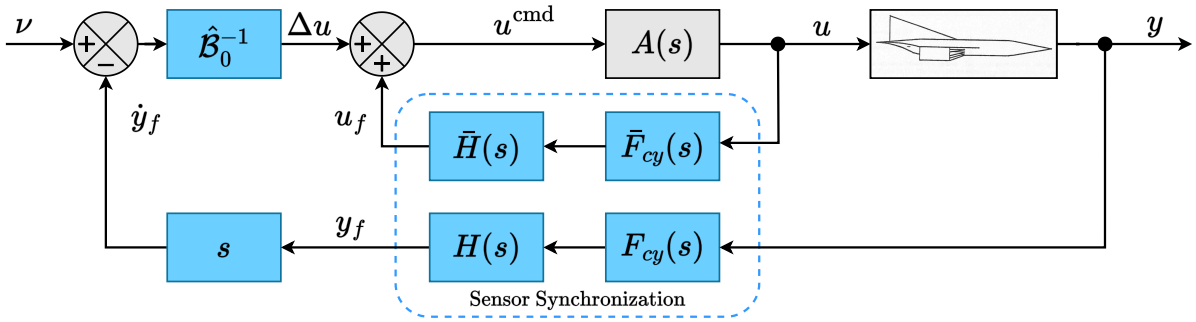


Figure 3.14: INDI control architecture with explicit sensor dynamics and synchronization [Steffensen et al., 2023a].

In this work, the analytical strategy of Fig. 3.14 is adopted, in which the sensor dynamics are modeled explicitly and the same filter chain $F_{cy}(s)$ is applied to the actuator feedback path to keep the actuator and measurement paths synchronized. Since the applied sensor dynamics are known, modeling them explicitly is the more natural choice than approximating them with a single compensating delay. Applying the filters to the actuator feedback path imposes the restriction that the same filter must act on every channel, which rules out independent per-axis filtering. This is not a limitation here, as a common filter chain is applied across all axes, so the reduced complementary filter of Fig. 3.13 is not required. It is precisely this synchronization that the delay experiments later probe, by comparing the controller's behavior under matched and mismatched actuator and sensor delays.

4

GHAME Vehicle Model Implementation

This chapter outlines the implementation of the GHAME vehicle model used in this research. The work is organized into seven sections, each addressing a key component of the modeling effort. Section 4.1 presents the GHAME configuration and mass properties, describing the geometry, propulsion layout, and overall vehicle structure. Section 4.2 formulates the 6 DoF equations of motion that govern the vehicle dynamics. Section 4.3 introduces the kinematic relationships and the environmental modeling framework, including the atmosphere, wind, and turbulence representations. Section 4.4 defines the modeling of aerodynamic and propulsive forces as well as actuator dynamics. Section 4.5 describes the sensor models used for feedback and state estimation within the control system. Section 4.6 details the initialization and trimming procedures required to establish a steady-flight condition for simulation. Finally, Sec. 4.7 presents the model verification process, in which the Simulink implementation is validated against the original NASA Fortran version[1] to ensure consistency and physical accuracy.

4.1. Model Description

The GHAME is a high-fidelity aerodynamic and geometric model developed at the NASA Ames Research Center and extensively tested at the Dryden Flight Research Center [Bowers et al., 1989]. It was conceived during the early phases of the NASP initiative to provide accurate and physically representative aerodynamic data for hypersonic vehicle research. GHAME was designed as a generic hypersonic configuration capable of performing a SSTO mission, involving horizontal takeoff using air-breathing propulsion, acceleration to orbital velocity, orbital insertion, atmospheric reentry, and unpowered gliding recovery. The aerodynamic database spans the complete mission envelope, including subsonic, transonic, hypersonic, and reentry flight regimes [White et al., 1992].

The model enables high-fidelity Six-Degree-of-Freedom (6DoF) simulations and produces a dynamic response representative of hypersonic flight. Consequently, GHAME has become a benchmark for evaluating control architectures, developing guidance strategies, and conducting trajectory optimization studies. Multiple versions of the model have been implemented in Fortran by P. Zipfel, using the original NASA Ames aerodynamic dataset as documented in White et al. [1992] and detailed in his textbook [Zipfel, 2025]. More recently, Goz [2024] developed a Simulink-based implementation of a simplified GHAME model assuming a nonrotating, flat Earth, constant mass and first-order actuator dynamics.

This thesis builds on that foundation by implementing Zipfel's highest-fidelity GHAME model in Simulink and extending it to include additional physical effects. In contrast to the flat-Earth formulation, the model developed here incorporates Earth's rotation, ellipticity, and a full gravity model. It also

[1]Available at: https://arc.aiaa.org/doi/suppl/10.2514/4.107535

includes an atmosphere extending to 1000 km, wind and turbulence effects, variable mass and inertia due to fuel consumption, sensor models, and second-order actuator dynamics. The objective is to reproduce the most complete and physically representative version of the GHAME model available while maintaining consistency with the original aerodynamic and dynamic formulations.

The GHAME vehicle geometry consists of analytically defined surfaces that capture the essential aerodynamic characteristics of a hypersonic lifting body. The fuselage is modeled as a 120 ft-long cylinder with a diameter of 20 ft, terminated at the front and aft ends by 10° half-angle conical nose and boat-tail sections. The wings and vertical tail are represented as thin, flat, triangular plates with no dihedral, arranged in a mid-wing configuration. Strakes extend rearward from the wing trailing edges to enhance directional stability at hypersonic speeds. Primary control is provided by all-moving elevons mounted on the trailing edges of the wing panels, performing the combined functions of elevators and ailerons. Symmetric deflection produces pitch control, differential deflection produces roll control, and yaw control is achieved by a rudder mounted on the trailing edge of the vertical tail [White et al., 1992].

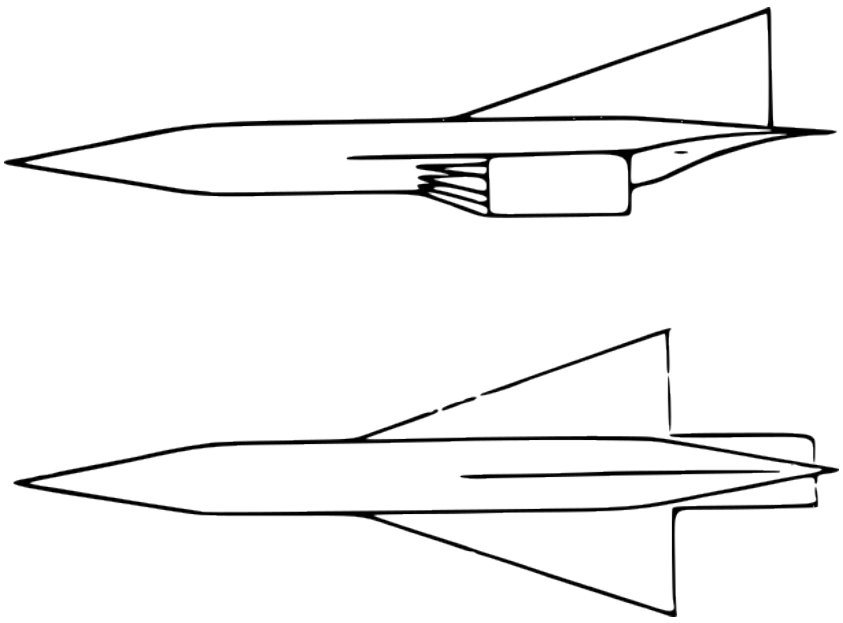


Figure 4.1: Schematic configuration of the GHAME vehicle model, adapted from White et al. [1992].

The GHAME model uses mass properties derived from the North American XB-70 Valkyrie. The takeoff gross weight is 300,000 lb, with 60 percent allocated to liquid-hydrogen fuel. The fuel-burnout weight is 120,000 lb. All liquid-hydrogen fuel is stored within the fuselage. The propulsion system is integrated into the lower fuselage between the wing roots, featuring a forward-facing intake on the underside and an aft-mounted nozzle extending beneath the tail section [White et al., 1992]. Moments of inertia are specified for both takeoff and burnout conditions, with mass and inertia values varying linearly with fuel consumption during simulation. The aerodynamic reference area and inertia data used for nondimensionalization are summarized in Appendix A.1. A schematic of the configuration is shown in Fig. 4.1.

The wing planform consists of a triangular delta wing with a root chord $c_r = 120$ ft and a span of $b_{\text{ref}} = 80$ ft, combined with a 30 ft span rectangular trailing surface of length 40 ft. The combined lifting surface yields a reference area of $S = 6000$ ft^2, and the reference chord is $c_{\text{ref}} = 75$ ft. The Mean Aerodynamic Chord (MAC) and its leading-edge location are MAC = 72 ft and $x_{\text{LEMAC}} = 116.5$ ft [White et al., 1992].

Model Assumptions

The GHAME model used in this work is the highest-fidelity version publicly available. Nevertheless, several simplifying assumptions were necessary to enable practical simulation. The following list summarizes the major assumptions adopted during the development and implementation of the model:

1. The vehicle is assumed to be a rigid body, neglecting structural deformation as well as aeroelastic and thermoelastic effects. Although these phenomena are known to occur in hypersonic flight, accurately modeling their influence remains an active area of research. The aerodynamic–structural–thermal coupling involved is complex, and defining a reliable model would constitute a research effort in its own right. Moreover, high-fidelity simulation models that capture such effects are not publicly available, making independent implementation and validation infeasible. For these reasons, such effects are excluded from the present simulation.

2. The total mass $m(t)$ and moment of inertia matrix $[I_B^B]^B(t)$ are assumed to vary linearly with fuel consumption, using interpolation between values at takeoff and burnout. This simplifies the underlying mass distribution, which in reality may vary nonlinearly depending on tank geometry and fuel routing. The linear assumption is made due to the absence of detailed internal configuration data and is consistent with prior implementations of the GHAME model.

3. The Center of Gravity (CG) is modeled and its position is computed from the mass distribution. However, its influence on aerodynamic characteristics and stability derivatives is not considered, as such data are not available. The CG position is used only for IMU correction calculations.

4. The wind velocity field is assumed to be uniform, and although it is not strictly constant in time, it is sufficiently constant within the framework of flight dynamics modeling. As a result, the atmosphere is considered stationary with respect to the Earth-fixed frame implying no angular velocity of the atmosphere relative to Earth.

5. Atmospheric turbulence is assumed to be accurately represented by the Dryden turbulence model, and steady horizontal wind profiles are modeled using the Horizontal Wind Model (HWM14).

6. Thrust is assumed to be aligned with the velocity vector. In the present formulation, this is approximated by aligning the thrust force with the 1^B-axis and neglecting any thrust incidence angle. As a result, the thrust vector does not generate pitching or yawing moments. This simplification neglects possible off-axis thrust components that can arise in high-speed propulsion systems, such as afterbody expansion or pressure effects. Such effects can generate additional normal forces and pitching moments, which would alter the trim condition and may increase trim drag [Calise and Flandro, 1988]. Such contributions are not included here and go beyond the scope of this thesis.

7. A key assumption in the aerodynamic model is that sideslip angle is not included as a gridded variable in the aerodynamic database. All aerodynamic lookup tables are defined only over Mach number and angle of attack. Sideslip-dependent effects are retained, such as $C_{Y_\beta}\beta$, but the sideslip angle itself is not used in the interpolation process.

4.2. Equations of Motion

The equations of motion describe the translational and attitude dynamics of a rigid body. Translational dynamics govern the motion of the vehicle's center of mass under the influence of external forces, while attitude dynamics describe the evolution of the body's orientation due to external moments. Together, these six coupled equations capture the full 6DoF of vehicle motion in space.

Translational Dynamics

The translational motion of a rigid body is governed by Newton's second law, which states that the net external force equals the time derivative of linear momentum. In an inertial reference frame I, this gives:

$$D^I(m\boldsymbol{v}_B^I) = \boldsymbol{f}_{\text{ext}}, \tag{4.1}$$

where D^I denotes the time derivative in the Inertial frame, $\boldsymbol{v}_B^I$ is the velocity of the vehicle center of mass B relative to the inertial frame I, m is the mass of the vehicle, and $\boldsymbol{f}_{\text{ext}}$ denotes the sum of all external forces acting on the body. This expression is valid in an inertial coordinate system and does not depend on the particular orientation of that system. When the mass m is constant, it can be taken outside the derivative, and Newton's law becomes:

$$mD^I\boldsymbol{v}_B^I = \boldsymbol{f}_{\text{ext}}. \tag{4.2}$$

Although Newton's law is valid only in inertial frames, it is often convenient to express the equations of motion in a rotating frame. By applying Euler's transformation of frames, also referred to as the Coriolis transformation, the time derivative can be shifted from the inertial frame I to a rotating frame R. The most common choice for R is the Earth-fixed frame E, which is non inertial due to the Earth's rotation.

The advantage of using the Earth-fixed frame is that the atmosphere rotates with the Earth, so aerodynamic forces depend on the vehicle's velocity relative to it. Moreover, trajectories are typically referenced to Earth's surface, making the Earth-fixed frame a natural choice. However, when transforming to a rotating reference frame with constant angular velocity, two correction terms appear in the equations of motion: the Coriolis and centrifugal accelerations. Thus, assuming constant mass and a uniformly rotating Earth, i.e. $\dot{\boldsymbol{\omega}}^{EI} = \mathbf{0}$, the translational dynamics become::

$$mD^E\boldsymbol{v}_B^E = \boldsymbol{f}_{\text{ext}} - m\left(\underbrace{2\boldsymbol{\omega}^{EI}\times\boldsymbol{v}_B^E}_{\text{Coriolis acceleration}} + \underbrace{\boldsymbol{\omega}^{EI}\times(\boldsymbol{\omega}^{EI}\times\boldsymbol{s}_{BE})}_{\text{Centrifugal acceleration}}\right). \tag{4.3}$$

Here, D^E denotes the time derivative in the Earth-fixed frame, and $\boldsymbol{\omega}^{EI}$ is the angular velocity vector of the Earth with respect to the inertial frame, and $\boldsymbol{s}_{BE}$ is the displacement vector from the center of the Earth to the center of mass of the vehicle. The added complexity is the reason that this model was decided to be implemented with the inertial reference frame as the reference for Newton's law without the transformation.

For a variable-mass system, the change in momentum must account for the momentum carried by the expelled mass. Consider the vehicle over a small time interval Δt. At time t, the vehicle has mass m and velocity $\boldsymbol{v}_B^I$. At time $t+\Delta t$, the vehicle mass has changed to $m+\dot{m}\Delta t$, and a mass $-\dot{m}\Delta t$ has been expelled.

Let $\boldsymbol{v}_{\text{rel}}^I$ denote the exhaust velocity relative to the vehicle, expressed in inertial coordinates. The inertial velocity of the expelled mass is therefore defined as:

$$\boldsymbol{v}_{\text{exh}}^I = \boldsymbol{v}_B^I - \boldsymbol{v}_{\text{rel}}^I. \tag{4.4}$$

The total momentum at time $t+\Delta t$ is the sum of the momentum of the remaining vehicle and the expelled mass:

$$\boldsymbol{p}(t+\Delta t) = (m+\dot{m}\Delta t)\left(\boldsymbol{v}_B^I + D^I\boldsymbol{v}_B^I\Delta t\right) + (-\dot{m}\Delta t)\left(\boldsymbol{v}_B^I - \boldsymbol{v}_{\text{rel}}^I\right). \tag{4.5}$$

Expanding and retaining only first-order terms in Δt yields:

$$\boldsymbol{p}(t+\Delta t) = m\boldsymbol{v}_B^I + mD^I\boldsymbol{v}_B^I\Delta t + \dot{m}\boldsymbol{v}_B^I\Delta t - \dot{m}\boldsymbol{v}_B^I\Delta t + \dot{m}\boldsymbol{v}_{\text{rel}}^I\Delta t. \tag{4.6}$$

The terms $+\dot{m}\boldsymbol{v}_B^I\Delta t$ and $-\dot{m}\boldsymbol{v}_B^I\Delta t$ cancel, giving:

$$\boldsymbol{p}(t+\Delta t) = m\boldsymbol{v}_B^I + mD^I\boldsymbol{v}_B^I\Delta t + \dot{m}\boldsymbol{v}_{\text{rel}}^I\Delta t. \tag{4.7}$$

Subtracting $\boldsymbol{p}(t) = m\boldsymbol{v}_B^I$ and dividing by Δt, the rate of change of momentum becomes:

$$D^I\left(m\boldsymbol{v}_B^I\right) = mD^I\boldsymbol{v}_B^I + \dot{m}\boldsymbol{v}_{\text{rel}}^I. \tag{4.8}$$

Applying Newton's second law (Eq. (4.1)) yields:

$$mD^I\boldsymbol{v}_B^I = \boldsymbol{f}_{\text{ext}} - \dot{m}\boldsymbol{v}_{\text{rel}}^I. \tag{4.9}$$

The second term represents the momentum flux associated with mass ejection and gives rise to the thrust force, defined as $\boldsymbol{f}_{\text{thrust}} = -\dot{m}\boldsymbol{v}_{\text{rel}}^I$. The translational dynamics can therefore be written as:

$$mD^I\boldsymbol{v}_B^I = \boldsymbol{f}_{\text{ext}} + \boldsymbol{f}_{\text{thrust}}. \tag{4.10}$$

The external forces are decomposed as $\boldsymbol{f}_{\text{ext}} = \boldsymbol{f}_a + m\boldsymbol{g}$, while the aero-propulsive forces are defined as $\boldsymbol{f}_{a,p} = \boldsymbol{f}_a + \boldsymbol{f}_{\text{thrust}}$. Substituting these definitions yields the final expression:

$$mD^I\boldsymbol{v}_B^I = \boldsymbol{f}_{a,p} + m\boldsymbol{g}. \tag{4.11}$$

Here, the aero-propulsive forces $\boldsymbol{f}_{a,p}$ are expressed in body coordinates $]^B$, while the gravitational force $m\boldsymbol{g}$ is expressed in geocentric coordinates $]^G$. Since neither are expressed in inertial coordinates, they are transformed into inertial form using the transformation matrices $[\overline{T}]^{BI}$, defined in Eq. (3.27) and $[\overline{T}]^{GI}$, defined in Eq. (3.24). The final expression for the translational dynamics is therefore:

$$m\left[\frac{dv_B^I}{dt}\right]^I = [\overline{T}]^{BI}\left[f_{a,p}\right]^B + m[\overline{T}]^{GI}[g]^G. \tag{4.12}$$

To calculate the actual trajectory, another integration is necessary for obtaining the displacement vector $\boldsymbol{s}_{BI}$ of the center of mass B relative to the center of the inertial reference frame I. The inertial position vector is updated by integrating the body velocity which is in inertial coordinates:

$$\left[\frac{ds_{BI}}{dt}\right]^I = [v_B^I]^I. \tag{4.13}$$

Equation (4.12) and Eq. (4.13) are the six differential equations that must be solved starting from initial conditions to solve the trajectory traces. From the inertial velocity $\boldsymbol{v}_B^I$ and inertial position $\boldsymbol{s}_{BI}$ it is now very simple to obtain the kinematic (or geographic) velocity $\boldsymbol{v}_B^E$ defined as the velocity of the aircraft with respect to the Earth, which is used to describe the trajectory of the vehicle. The geographic velocity is obtained by correcting the inertial velocity $\boldsymbol{v}_B^I$ for Earth's rotation:

$$[v_B^E]^I = [v_B^I]^I - [\Omega^{EI}]^I[s_{BI}]^I. \tag{4.14}$$

For convenience, this velocity is subsequently expressed in the NED coordinate system by applying the transformation matrix $[T]^{DI}$ from Eq. (3.23):

$$[v_B^E]^D = [T]^{DI}\left([v_B^I]^I - [\Omega^{EI}]^I\,[s_{BI}]^I\right). \tag{4.15}$$

In the presence of wind, two additional velocity vectors are introduced. The first is the aerodynamic velocity $\boldsymbol{v}_B^A$, which defines the motion of the aircraft relative to the surrounding air mass and governs both aerodynamic forces and propulsion effects. The second is the wind velocity $\boldsymbol{v}_A^E$, which represents the velocity of the atmosphere relative to the Earth. The aerodynamic velocity is obtained by subtracting the wind velocity from the aircraft's velocity relative to Earth:

$$[v_B^A]^D = [v_B^E]^D - [v_A^E]^D, \tag{4.16}$$

where the wind velocity $[v_A^E]^D$ is provided by the wind module described in Sec. 4.3. The aerodynamic velocity in geodetic coordinates can be used to determine the aerodynamic heading χ_A and flight path angle γ_A. However, evaluation of the aerodynamic angle of attack α_A and aerodynamic sideslip angle β_A requires the aerodynamic velocity expressed in body coordinates. This is obtained by applying the transformation matrix $[T]^{BD}$ from Eq. (3.26):

$$[v_B^A]^B = [T]^{BD}\left([v_B^E]^D - [v_A^E]^D\right). \tag{4.17}$$

Finally, the magnitude of the aerodynamic velocity is the true air speed of the vehicle:

$$V_{\text{TAS}} = \|\boldsymbol{v}_B^A\|_2 \tag{4.18}$$

Attitude Dynamics

The attitude dynamics of a rigid body are governed by Euler's law. It states that the time derivative of the angular momentum vector $\boldsymbol{l}_B^{BI}$ of a rigid body, taken relative to the inertial frame and expressed about the body's center of mass, is equal to the net external moment $\boldsymbol{m}_B$ applied about that same point:

$$D^I \boldsymbol{l}_B^{BI} = \boldsymbol{m}_B. \tag{4.19}$$

The angular momentum $\boldsymbol{l}_R^{BR}$ of a rigid body B with relation to any reference frame R and referred to a reference point R can be calculated from two additive terms:

$$\boldsymbol{l}_R^{BR} = \boldsymbol{I}_B^B \boldsymbol{\omega}^{BI} + m^B(\boldsymbol{r}_{BR} \times \boldsymbol{v}_B^R), \tag{4.20}$$

where $\boldsymbol{I}_B^B$ is the inertia tensor about the center of mass, $\boldsymbol{\omega}^{BI}$ is the angular velocity of B relative to I, $\boldsymbol{r}_{BR}$ is the vector from R to the center of mass, and $\boldsymbol{v}_B^R$ is the linear velocity of that center with respect to R. The first term describes the angular momentum about its center of mass and the second term corrects it in case that the reference point lies somewhere else. If the reference point is chosen as the center of mass itself ($R = B$), the second term vanishes and the angular momentum simplifies to:

$$\boldsymbol{l}_B^{BI} = \boldsymbol{I}_B^B \boldsymbol{\omega}^{BI}. \tag{4.21}$$

Substituting this into Eq. (4.19) gives:

$$D^I\left(\boldsymbol{I}_B^B \boldsymbol{\omega}^{BI}\right) = \boldsymbol{m}_B. \tag{4.22}$$

The inertia tensor and angular velocity are most naturally expressed in the body-fixed frame, so the time derivative is shifted from the inertial frame to the body frame using the rotational transport theorem:

$$D^I(\cdot) = D^B(\cdot) + \boldsymbol{\Omega}^{BI}(\cdot)\,. \tag{4.23}$$

Here D^I and D^B denote time derivatives taken in the inertial and body frames, respectively, and $\boldsymbol{\Omega}^{BI}$ is the skew-symmetric matrix corresponding to the body's angular velocity relative to inertial. Physically, the extra term $\boldsymbol{\Omega}^{BI}(\cdot)$ accounts for the apparent change in a vector simply because the coordinate axes themselves are rotating. Applying this to the angular momentum term yields:

$$D^B\left(\boldsymbol{I}_B^B\boldsymbol{\omega}^{BI}\right)+\boldsymbol{\Omega}^{BI}\boldsymbol{I}_B^B\boldsymbol{\omega}^{BI}=\boldsymbol{m}_B. \tag{4.24}$$

To solve the angular momentum rate in the rotating frame, the product rule must be applied to $D^B\left(\boldsymbol{I}_B^B\boldsymbol{\omega}^{BI}\right)$:

$$D^B\left(\boldsymbol{I}_B^B\,\boldsymbol{\omega}^{BI}\right)=\left(D^B\boldsymbol{I}_B^B\right)\boldsymbol{\omega}^{BI}+\boldsymbol{I}_B^B\left(D^B\boldsymbol{\omega}^{BI}\right). \tag{4.25}$$

For a constant inertia matrix, the derivative $D^B\boldsymbol{I}_B^B$ vanishes, otherwise, the additional term must be retained. Substituting this expansion into Eq. (4.24) yields:

$$\left(D^B\boldsymbol{I}_B^B\right)\boldsymbol{\omega}^{BI}+\boldsymbol{I}_B^B\left(D^B\boldsymbol{\omega}^{BI}\right)+\boldsymbol{\Omega}^{BI}\,\boldsymbol{I}_B^B\,\boldsymbol{\omega}^{BI}=\boldsymbol{m}_B. \tag{4.26}$$

Finally, expressing each tensor in body-fixed coordinates $\cdot]^B$ yields:

$$\left[\frac{dI_B^B}{dt}\right]^B[\omega^{BI}]^B+[I_B^B]^B\left[\frac{d\omega^{BI}}{dt}\right]^B+[\Omega^{BI}]^B\,[I_B^B]^B\,[\omega^{BI}]^B=[m_B]^B. \tag{4.27}$$

This is the full expression for Euler's law in the body frame with a time-varying inertia tensor. It accounts for three terms on the left-hand side: the contribution due to the change in inertia, the angular acceleration term, and the gyroscopic coupling due to rotation of the body frame. To isolate the angular acceleration $D^B[\omega^{BI}]^B$, the equation is rearranged:

$$\left[\frac{d\omega^{BI}}{dt}\right]^B=\left(\left[I_B^B\right]^B\right)^{-1}\left([m_B]^B-\left[\frac{dI_B^B}{dt}\right]^B[\omega^{BI}]^B-[\Omega^{BI}]^B\left[I_B^B\right]^B[\omega^{BI}]^B\right). \tag{4.28}$$

This is the rotational equation of motion given in the body frame. The translational plus attitude equations together form nine first-order differential equations and therefore nine state variables are needed for initialization. It is general in the sense that it does not assume constant inertia, and can therefore accommodate time-dependent mass distributions. For aerodynamic calculations, the angular velocity of the body relative to the rotating Earth, expressed in body coordinates $[\omega^{BE}]^B$, is required. It is related to the inertial angular velocity $[\omega^{BI}]^B$ via:

$$[\omega^{BE}]^B=[\omega^{BI}]^B-[T]^{BI}[\omega^{EI}]^I. \tag{4.29}$$

Here, $[\omega^{EI}]^I$ is the angular velocity of the Earth frame with respect to the inertial frame, expressed in inertial coordinates. For a rotating Earth model, this vector is defined as:

$$[\omega^{EI}]^I=\begin{bmatrix}0\\0\\\omega_\oplus\end{bmatrix}^\top, \tag{4.30}$$

where $\omega_\oplus$ is the Earth's angular velocity. The components of $[\omega^{BE}]^B$ are used in the aerodynamic moment calculation.

4.3. Kinematics and Environment

The orientation of the vehicle can be represented using either quaternions, Direction Cosine Matrix (DCM), or Euler angles. Quaternions are efficient for integration and avoid singularities, but they represent orientation implicitly and complicate initialization when multiple reference frames are involved. By contrast, DCMs express orientation explicitly as a sequence of rotations between coordinate systems, making them straightforward to interpret and manipulate in a multi-frame formulation. Both are valid options but for this reason, the present simulation employs DCMs rather than quaternions.

Attitude is defined using yaw, pitch, and roll angles which give the orientation of the body-fixed coordinates relative to the geodetic coordinate system. These Euler angles are updated indirectly through the body-to-inertial transformation matrix, $[T]^{BI}$ whose time evolution is governed by the angular velocity of the body relative to the inertial frame:

$$\left[\frac{dT}{dt}\right]^{BI} = [\bar{\Omega}^{BI}]^{B}[T]^{BI}, \tag{4.31}$$

where $[\bar{\Omega}^{BI}]^{B}$ is the transpose of the skew-symmetric matrix of $[\omega^{BI}]^{B}$. The transformation matrix from inertial to body coordinates $[T]^{BI}$ is initialized as shown in Eq. (4.86). Therefore, after initialization the DCM only depends on the inertial rotational rates.

Numerical integration of Eq. (4.31) introduces small errors that accumulate over time and gradually degrade the orthogonality of $[T]^{BI}$. To preserve the rotation matrix properties, $[T]^{BI}$ is re-orthogonalized at each time step using the algorithm of Savage [1984]:

$$[T(n+1)]^{BI} = [T(n)]^{BI} + \frac{1}{2}\left([E][T(n)]^{BI}\right), \quad \text{where} \quad [E] = [I] - [T]^{BI}[\bar{T}]^{BI}. \tag{4.32}$$

Here $[T(n)]^{BI}$ is the transformation matrix after integration, $[T(n+1)]^{BI}$ is the re-orthogonalized matrix, $[E]$ denotes the error matrix, and $[I]$ the identity. The error $[E]$ vanishes when $[T(n)]^{BI}$ is perfectly orthogonal. The corrected matrix is then used to update the geodetic–to–body transformation:

$$[T]^{BD} = [T]^{BI}\,[\bar{T}]^{DI}. \tag{4.33}$$

At each time step, the updated Euler angles are extracted from the transformation matrix $[T]^{BD}$. Two methods can be used for this purpose. The first is the approach implemented by Zipfel [2025], which computes the pitch angle first and then recovers yaw and roll using arccos and sign disambiguation. This method includes explicit safeguards against domain errors and singularities. The second method makes use of the arctan 2 function to compute yaw and roll directly, avoiding the need to divide by $\cos\theta_{bd}$. While both approaches are equivalent away from singularities, the arctan 2-based method was implemented here due to its cleaner structure, reduced numerical sensitivity, and better behavior near gimbal lock.

Starting with the method by Zipfel [2025], the pitch angle θ_{bd} is computed first, along with its cosine, using the matrix elements t_{ij} of $[T]^{BD}$ as defined in Eq. (3.26):

$$\theta_{bd} = \begin{cases} -\arcsin(t_{13}), & |t_{13}| < 1, \\ \dfrac{\pi}{2}\,\mathrm{sign}(-t_{13}), & |t_{13}| \geq 1, \end{cases} \qquad \cos\theta_{bd} = \begin{cases} \sqrt{1-t_{13}^2}, & |t_{13}| < 1, \\ \mathrm{EPS}, & |t_{13}| \geq 1, \end{cases}$$

where $\mathrm{EPS} \ll 1$ prevents division by zero when $|t_{13}| \approx 1$. This guarded formulation serves two purposes. First, it accounts for slight numerical errors that may push $|t_{13}|$ just outside the valid domain $[-1, 1]$ of arcsin, avoiding NaN results. Second, as $|t_{13}| \to 1$ (i.e., $\theta_{bd} \to \pm\frac{\pi}{2}$), $\cos\theta_{bd}$ approaches zero, making any subsequent division by $\cos\theta_{bd}$ (required for yaw and roll extraction) ill-conditioned. By clamping θ_{bd} to $\pm\frac{\pi}{2}$ and enforcing a minimum value $\cos\theta_{bd} = \mathrm{EPS}$, the algorithm maintains stable, finite intermediate values and ensures reliable recovery of all three Euler angles even near the gimbal-lock condition. With $\cos\theta_{bd}$ thus guaranteed nonzero, the cosine of the yaw angle is then computed using the $(1,1)$ and $(1,2)$ element:

$$\cos\psi_{bd} = \min\left(1,\ \left|\frac{t_{11}}{\cos\theta_{bd}}\right|\right) \cdot \mathrm{sign}(\cos\psi_{bd}), \quad \psi_{bd} = \arccos\left(\cos\psi_{bd}\right) \cdot \mathrm{sign}(t_{12}).$$

Similarly, the cosine of the roll angle and the roll angle itself are computed using the $(3,3)$ and $(2,3)$ element:

$$\cos\phi_{bd} = \min\left(1,\ \left|\frac{t_{33}}{\cos\theta_{bd}}\right|\right)\cdot \operatorname{sign}(\cos\phi_{bd}), \qquad \phi_{bd} = \arccos\left(\cos\phi_{bd}\right)\cdot\operatorname{sign}(t_{23}).$$

This formulation ensures all three Euler angles remain well-defined and numerically stable even when $\theta_{bd} \to \pm\frac{\pi}{2}$, where the Euler angle extraction would become ill-conditioned. The second formulation makes use of arctan2 to compute the ϕ_{bd} and ψ_{bd} angles in which case special logic for $\cos\theta$ is not needed anymore:

$$\theta_{bd} = \begin{cases} \arcsin(-t_{13}), & |t_{13}| < 1, \\ \dfrac{\pi}{2}\cdot\operatorname{sign}(-t_{13}), & |t_{13}| \geq 1, \end{cases}$$

$$\psi_{bd} = \arctan 2(t_{12}, t_{11}), \qquad \phi_{bd} = \arctan 2(t_{23}, t_{33}).$$

This formulation avoids discontinuities and domain errors by using the two-argument arctangent, which preserves sign information and avoids division by $\cos\theta_{bd}$. The pitch angle is clamped to $\pm\frac{\pi}{2}$ when $|t_{13}| \geq 1$, ensuring robustness near the gimbal-lock condition.

Angle Computation from $\boldsymbol{v}_B^E$ and $\boldsymbol{v}_B^A$

Using the components of the air-relative velocity vector expressed in the body-fixed coordinates, $[v_B^A]^B = [u_a\ v_a\ w_a]^\top$, the aerodynamic sideslip angle β_A is given by:

$$\beta_A = \arcsin\left(\frac{v_a}{V_{\text{TAS}}}\right), \qquad -\frac{\pi}{2} < \beta_A \leq \frac{\pi}{2}, \tag{4.34}$$

where $V_{\text{TAS}} = \sqrt{u_a^2 + v_a^2 + w_a^2}$ is the magnitude of the aerodynamic velocity vector. Using this result, the aerodynamic angle of attack α_A is defined by

$$\cos\alpha_A = \frac{u_a}{V_{\text{TAS}}\cos\beta_A}, \quad \text{and} \quad \sin\alpha_A = \frac{w_a}{V_{\text{TAS}}\cos\beta_A}. \tag{4.35}$$

Alternatively, a more compact and robust expression is given by the two-argument arctangent function:

$$\alpha_A = \arctan 2(w_a, u_a), \qquad -\pi < \alpha_A < \pi. \tag{4.36}$$

The aerodynamic bank angle μ_A requires additional algebraic manipulation to express in closed form. The resulting relation is:

$$\mu_A = \arctan 2\left(\tan\phi_{bd} + \tan\beta_A\left(\tan\theta_{bd}\frac{\cos\alpha_A}{\cos\phi_{bd}} - \frac{\sin\alpha_A}{\cos\theta_{bd}}\right), 1 + \tan\theta_{bd}\frac{\sin\alpha_A}{\cos\beta_A\cos\phi_{bd}}\right). \tag{4.37}$$

Kinematic Flight Path Angle and Heading Angle γ_K, χ_K

The kinematic heading angle χ_K and kinematic flight path angle γ_K are computed from the Earth-relative velocity vector $[v_B^E]^D = [v_N,\ v_E,\ v_D]^\top$, expressed in the geodetic coordinate system. The ground-relative velocity magnitude is:

$$V_K = ||\boldsymbol{v}_B^E||. \tag{4.38}$$

The velocity components relate to the kinematic angles by:

$$v_N = V_K\cos\chi_K\cos\gamma_K, \tag{4.39}$$

$$v_E = V_K\sin\chi_K\cos\gamma_K, \tag{4.40}$$

$$v_D = -V_K\sin\gamma_K. \tag{4.41}$$

The heading angle is defined as the angle between the horizontal projection of the velocity vector and the north direction:

$$\chi_K = \arctan 2(v_N,\ v_E), \qquad -\pi < \chi_K < \pi. \tag{4.42}$$

The flight path angle is defined as the angle between the velocity vector and the local horizontal plane:

$$\gamma_K = \arcsin\left(-\frac{v_D}{V_K}\right), \qquad -\frac{\pi}{2} < \gamma_K \leq \frac{\pi}{2}. \tag{4.43}$$

The use of γ_K and χ_K in this context provides a natural parameterization of the trajectory based on Earth-referenced velocity components. These angles differ from the aerodynamic flight path, γ_A, and heading angle, χ_A, when wind is present.

Gravity Computation in Geocentric $]^G$ Coordinate Axis

Gravitational acceleration is required for the vehicle's translational dynamics. For an elliptical-Earth simulation, the standard model is the World Geodetic System 84 (WGS-84) [WGS, 1997], which represents gravity using a zonal harmonic expansion truncated after the dominant J_2 term. In this formulation, the gravitational acceleration expressed in geocentric $]^G$ coordinates is given by

$$[g]^G = \frac{GM}{d_{BI}^2}\begin{bmatrix} -3\sqrt{5}\,C_{2,0}\left(\frac{a}{d_{BI}}\right)^2 \sin\lambda_c \cos\lambda_c \\ 0 \\ 1 + \frac{3}{2}\sqrt{5}\,C_{2,0}\left(\frac{a}{d_{BI}}\right)^2 \left(3\sin^2\lambda_c - 1\right) \end{bmatrix}, \qquad J_2 = -\sqrt{5}\,C_{2,0}. \tag{4.44}$$

Here, GM is the Earth's gravitational parameter and λ_c is the geocentric latitude. The parameter a denotes the Earth's reference equatorial radius. The scalar d_{BI} is the distance between the vehicle's center of mass and the Earth's center. The coefficient $C_{2,0}$ is the normalized degree-2, order-0 Stokes coefficient representing the leading-order deviation from spherical symmetry due to the Earth's oblateness.

The third component of the gravity vector contains both the dominant radial $1/d_{BI}^2$ term and a latitude-dependent correction from the J_2 perturbation. This correction increases gravity near the poles and reduces it near the equator. While the equatorial bulge contains more mass than a perfect sphere, this mass is distributed farther from the Earth's center and therefore contributes less effectively to the gravitational attraction at a given altitude. The inverse-square nature of gravity amplifies this effect, resulting in a net reduction in gravitational acceleration close to the equator. The first component introduces a horizontal acceleration directed toward the equator, positive in the Northern Hemisphere and negative in the Southern, caused by the off-axis gravity induced by the oblate shape. The second component is zero due to the axial symmetry of the pure zonal harmonic field.

When $C_{2,0} = 0$, the oblateness terms vanish, and the acceleration reduces to the classical spherical-Earth result. To evaluate Eq. (4.44), use the inertial position vector $\left[s_{BI}\right]^I = [s_1\ s_2\ s_3]^\top$ to compute the distance d_{BI} and the geocentric latitude λ_c:

$$d_{BI} = \|s_{BI}\| = \sqrt{s_1^2 + s_2^2 + s_3^2}, \qquad \lambda_c = \arcsin\left(\frac{s_3}{d_{BI}}\right).$$

Gravity must be expressed in geocentric coordinates because gravitational force acts toward the Earth's center of mass, not perpendicular to its surface. Geodetic coordinates, which define "down" as normal to the reference ellipsoid, misalign with the actual gravity vector, particularly at high latitudes where the ellipsoid surface deviates most from spherical. Expressing $[g]^G$ in geocentric coordinates avoids this geometric inconsistency and ensures that the gravity vector is modeled correctly in both direction and magnitude. The constants used in this formulation are summarized in Appendix A.

Wind and Turbulence

To enhance atmospheric fidelity, this simulation incorporates horizontal wind and turbulence models. Wind and turbulence change the incidence angle of the aircraft and thus change the aerodynamic moment and forces. These two are superimposed to find the wind velocity. The horizontal wind is obtained from the HWM14. For a given geodetic latitude, geo-potential altitude, longitude, time of day, and day of year it outputs the zonal and meridional components of wind (east and north), corresponding to the first and second components of a velocity vector in geodetic coordinates. No vertical wind component is modeled, so the third element of the wind vector is set to zero. Accordingly, this wind vector in geodetic coordinates can be expressed as:

$$[v_A^E]_{\text{HW}}^D = \begin{bmatrix} v_{\text{Wind},N} \\ v_{\text{Wind},E} \\ 0 \end{bmatrix}. \tag{4.45}$$

To account for random turbulence, a modified Dryden model is implemented. The turbulence is first computed in body axes using a second-order linear filter shaped by the Dryden spectral distribution, then transformed to geodetic coordinates. The stochastic disturbance is modeled as a scalar turbulence amplitude, filtered using the following differential equations:

$$\dot{\tau}_1 = \tau_2, \qquad \dot{\tau}_2 = -v_\ell^2 \tau_1 - 2v_\ell \tau_2 + v_\ell^2 w(t), \tag{4.46}$$

where $w(t)$ is zero-mean white Gaussian noise scaled for discrete-time simulation, are the gust variables and $v_\ell = v_{\text{TAS}}/L$ is the turbulence frequency bandwidth based on true airspeed and turbulence scale length L. For high-altitude flight, a typical value is $L = 150\,\text{m}$ [Zipfel, 2025]. The resulting turbulence amplitude is computed as:

$$\tau(t) = \sigma \sqrt{\frac{1}{2\pi v_\ell}} \left(\tau_1 + \frac{\sqrt{3}}{v_\ell} \tau_2 \right). \tag{4.47}$$

This model assumes isotropic turbulence aligned with the freestream direction. While this simplification neglects cross-axis coupling and vertical shear, it is appropriate for evaluating control performance under moderate atmospheric disturbance. The turbulence τ is then projected into body axes using the angle of attack α and roll angle ϕ, forming a three-dimensional turbulence velocity vector:

$$[v_T^A]^B = \begin{bmatrix} -\tau \sin\alpha \\ \tau \sin\phi \cos\alpha \\ \tau \cos\phi \cos\alpha \end{bmatrix}. \tag{4.48}$$

This projection approximates the local direction of atmospheric gusts relative to the body, it assumes negligible sideslip and small yaw excursions. The turbulence vector is transformed to geodetic coordinates using the body-to-geodetic transformation matrix $[T]^{DB}$ which is given in Eq. (3.26):

$$[v_T^A]^D = [T]^{DB} [v_T^A]^B. \tag{4.49}$$

The total wind velocity in geodetic coordinates is defined as the sum of the horizontal wind and the turbulence velocity vector:

$$[v_A^E]^D = [v_A^E]_{\text{HW}}^D + [v_T^A]^D. \tag{4.50}$$

To prevent discontinuities, a first-order low-pass filter is applied to the wind velocity with a time constant $t_{\text{wind}} = 0.1\,\text{s}$, yielding the smoothed wind velocity $[v_A^E]_{\text{S}}^D$ and its time derivative:

$$\left[\frac{dv_A^E}{dt} \right]_{\text{S}}^D = \frac{[v_A^E]^D - [v_A^E]_{\text{S}}^D}{t_{\text{wind}}}, \qquad [v_A^E]^D := [v_A^E]_{\text{S}}^D. \tag{4.51}$$

The chosen time constant balances responsiveness with robustness to high-frequency noise and ensures numerical stability in the downstream computation of air-relative velocity. After filtering, the smoothed velocity $[v_A^E]_S^D$ replaces the original $[v_A^E]^D$ in all subsequent computations. The wind model is subsequently integrated into the environment and kinematics subsystems to determine air-relative velocity and dynamic pressure.

Atmospheric Model

The atmospheric model used in this work is the 1976 U.S. Standard Atmosphere (USSA76), implemented following the NASA Marshall Space Flight Center realization [NASA, 1976]. For a given geopotential altitude Z_g^{wgs84}, the model provides local air density ρ, pressure p, temperature T, and speed of sound a over the full range from sea level to 1000 km. The implementation combines two regimes. Below 86 km, the standard stratified-layer structure of the USSA76 is applied, using geopotential altitude correction, piecewise-linear temperature gradients, and hydrostatic pressure relations defined by the specified lapse rates or isothermal layers. In this region, density is computed from the ideal gas law with a constant mean molecular weight.

Above 86 km, the model transitions into the thermosphere and exosphere, where molecular diffusion dominates and the atmosphere becomes compositionally stratified. Between 86 km and 91 km, temperature is obtained by direct lookup from USSA76 tabulated values, and pressure is interpolated in logarithmic space. From 91 km to 110 km, temperature follows a nonlinear empirical fit to capture the curvature observed in measured thermospheric profiles, with pressure and molecular weight interpolated quadratically. Between 110 km and 120 km, temperature increases linearly with altitude. From 120 km to 500 km, an exponential formulation drives the temperature asymptotically toward 1000 K, while the molecular weight decreases with altitude due to the growing abundance of lighter atomic species such as helium and hydrogen. In the exosphere (500–1000 km), temperature is held constant at 1000 K, and pressure continues to be interpolated in log-space. To ensure smooth transitions across all layers above 86 km, interpolated quantities are averaged at the regime boundaries.

Most atmospheric and wind models, including the USSA76 and the HWM14, use geopotential altitude above the WGS-84 ellipsoid as input. This differs from the geodetic altitude, which measures the vertical distance above the reference ellipsoid. Geopotential altitude accounts for the reduction in gravitational acceleration with height and represents the altitude at which a parcel of air has the same potential energy in a constant-gravity field. The conversion from geodetic altitude h (in meters) at geodetic latitude λ_d to geopotential altitude Z_g^{wgs84} is given by [Scherllin-Pirscher et al., 2017]:

$$Z_g^{\text{wgs84}}(\lambda_d, h) = \frac{\gamma(\lambda_d)}{g_0} h \left[1 - \frac{h}{R_e}\left(1 + f + m - 2f \sin^2 \lambda_d\right) + \frac{h^2}{R_e^2}\right], \tag{4.52}$$

where g_0 is the nominal gravitational acceleration at sea level, R_e the equatorial radius of the WGS-84 ellipsoid, f the Earth's flattening, and m the geodynamical constant. The latitude-dependent normal gravity $\gamma(\lambda_d)$ is computed as [Scherllin-Pirscher et al., 2017]:

$$\gamma(\lambda_d) = \frac{R_e \gamma_e \cos^2 \lambda_d + R_p \gamma_p \sin^2 \lambda_d}{\sqrt{R_e^2 \cos^2 \lambda_d + R_p^2 \sin^2 \lambda_d}}, \tag{4.53}$$

where γ_e and γ_p are the gravitational accelerations at the equator and pole, respectively, and R_p is the polar radius. This formulation accounts for the variation of gravitational acceleration with both latitude and altitude and ensures consistency with the hydrostatic assumption underlying standard atmospheric models. The constants used are listed in Appendix A.

4.4. Aeropropulsive Forces and Actuators

The GHAME aerodynamic database was developed by combining theoretical flow models and empirical data from multiple legacy hypersonic vehicle configurations. Longitudinal aerodynamic coefficients were constructed using blended data from the Space Shuttle Orbiter, lifting-body vehicles, a swept double-delta wing platform, and a 60° half-angle cone. For lateral-directional coefficients, Space Shuttle and double-delta data were used at Mach numbers below 8. Above this threshold, only Space Shuttle data were retained. Drag coefficients were scaled to match lift-to-drag ratios consistent with measured Space Shuttle performance. All coefficients were normalized using a fixed aerodynamic reference area and span. To extend the aerodynamic model into the hypersonic regime, a modified Newtonian impact theory was applied. This combination of analytical and empirical methods yields an aerodynamic model capable of capturing the nonlinearities, cross-coupling, and regime transitions critical for high-fidelity hypersonic flight simulation [White et al., 1992].

The aerodynamic force and moment coefficients are expressed as linear combinations of the control surface deflections, angular rates, sideslip angle β, and angle of attack α. For readability, the aerodynamic subscripts have been omitted, but it is important to note that in simulations with wind, the aerodynamic angles of attack α_A and sideslip β_A must be used, whereas in wind-free conditions, the corresponding kinematic angles α_K and β_K are applied. All aerodynamic coefficients are tabulated as functions of Mach number M and angle of attack α and were derived under the assumption of zero sideslip angle ($\beta = 0$), as specified in the original GHAME reference model, even though β may vary during simulation. Additionally, the coefficients C_{L_q}, C_{Y_0}, C_{Y_p}, C_{Y_r}, C_{l_0}, and C_{n_0} are negligible across the evaluated flight envelope [White et al., 1992]. The force coefficients are defined as follows:

$$\begin{aligned} C_L &= C_{L_0} + C_{L_\alpha}\,\alpha_A + C_{L_{\delta_e}}\,\delta_e, \\ C_Y &= C_{Y_\beta}\,\beta + C_{Y_{\delta_a}}\,\delta_a + C_{Y_{\delta_r}}\,\delta_r, \\ C_D &= C_{D_0} + C_{D_\alpha}\,\alpha_A, \end{aligned} \tag{4.54}$$

where δ_e is the elevator deflection, δ_a the aileron deflection, and δ_r the rudder deflection. The moment coefficients are calculated as follows:

$$\begin{aligned} C_m &= C_{m_0} + C_{m_\alpha}\,\alpha_A + C_{m_{\delta_e}}\,\delta_e + C_{m_q}\,\frac{q c_{\text{ref}}}{2V_{\text{TAS}}}, \\ C_l &= C_{l_\beta}\,\beta_A + C_{l_{\delta_a}}\,\delta_a + C_{l_{\delta_r}}\,\delta_r + C_{l_p}\,\frac{p b_{\text{ref}}}{2V_{\text{TAS}}} + C_{l_r}\,\frac{r b_{\text{ref}}}{2V_{\text{TAS}}}, \\ C_n &= C_{n_\beta}\,\beta_A + C_{n_{\delta_r}}\,\delta_r + C_{n_{\delta_a}}\,\delta_a + C_{n_p}\,\frac{p b_{\text{ref}}}{2V_{\text{TAS}}} + C_{n_r}\,\frac{r b_{\text{ref}}}{2V_{\text{TAS}}}, \end{aligned} \tag{4.55}$$

where p, q, and r denote the roll, pitch, and yaw rates of the vehicle with respect to Earth in body axes, respectively. Then c_{ref} is the reference chord, b_{ref} the reference span of the aircraft and V_{TAS} is the true airspeed which is defined as the magnitude of the aerodynamic velocity $\boldsymbol{v}_B^A$. All partial derivatives with respect to angles have units of $1/^\circ$, while those with respect to angular rates have units of $1/\text{rad}$.

The lift coefficient C_L is defined as positive upwards and perpendicular to the velocity vector, and the drag coefficient C_D is positive in the direction opposite to flight. The side force coefficient C_Y is positive to the right (starboard), the rolling moment coefficient C_l is positive for right-wing-down roll, the pitching moment coefficient C_m is positive for nose-up rotation, and the yawing moment coefficient C_n is positive for nose-right yaw. Since the aerodynamic force coefficients are expressed in stability coordinates, they must be transformed into body coordinates using the angle of attack α_A:

$$\begin{aligned} C_X &= -C_D \cos\alpha_A + C_L \sin\alpha_A, \\ C_Z &= -C_D \sin\alpha_A - C_L \cos\alpha_A. \end{aligned} \tag{4.56}$$

The total force vector $[f_{a,p}]^B$ is computed by combining the aerodynamic forces with the propulsive force. The total force vector in body coordinates is given by:

$$[f_{a,p}]^B \equiv \begin{bmatrix} f_{a,p_1} \\ f_{a,p_2} \\ f_{a,p_3} \end{bmatrix} = \begin{bmatrix} \bar{q}SC_X + f_P \\ \bar{q}SC_Y \\ \bar{q}SC_Z \end{bmatrix}, \tag{4.57}$$

where $\bar{q} = \frac{1}{2}\rho V_{\text{TAS}}^2$ is the dynamic pressure, in which ρ is the air density, S is the wing reference area and f_P is the propulsive force. The aerodynamic moment vector in the body frame is computed as:

$$[m_B]^B \equiv \begin{bmatrix} m_{B_1} \\ m_{B_2} \\ m_{B_3} \end{bmatrix} = \begin{bmatrix} \bar{q}Sb_{\text{ref}}C_l \\ \bar{q}Sc_{\text{ref}}C_m \\ \bar{q}Sb_{\text{ref}}C_n \end{bmatrix}, \tag{4.58}$$

where C_l, C_m, C_n are the aerodynamic moment coefficients. These force and moment vectors are inputs to the translational and rotational dynamics equations. Each coefficient is represented in a tabulated format over a fixed grid of nine angle-of-attack rows and thirteen Mach number columns, as shown below.

$$\begin{aligned} \alpha &= [-3 \quad 0 \quad 3 \quad 6 \quad 9 \quad 12 \quad 15 \quad 18 \quad 21], \\ M &= [0.4 \quad 0.6 \quad 0.8 \quad 0.9 \quad 0.95 \quad 1.05 \quad 1.2 \quad 1.5 \quad 2.0 \quad 3.0 \quad 6.0 \quad 12.0 \quad 24.0]. \end{aligned} \tag{4.59}$$

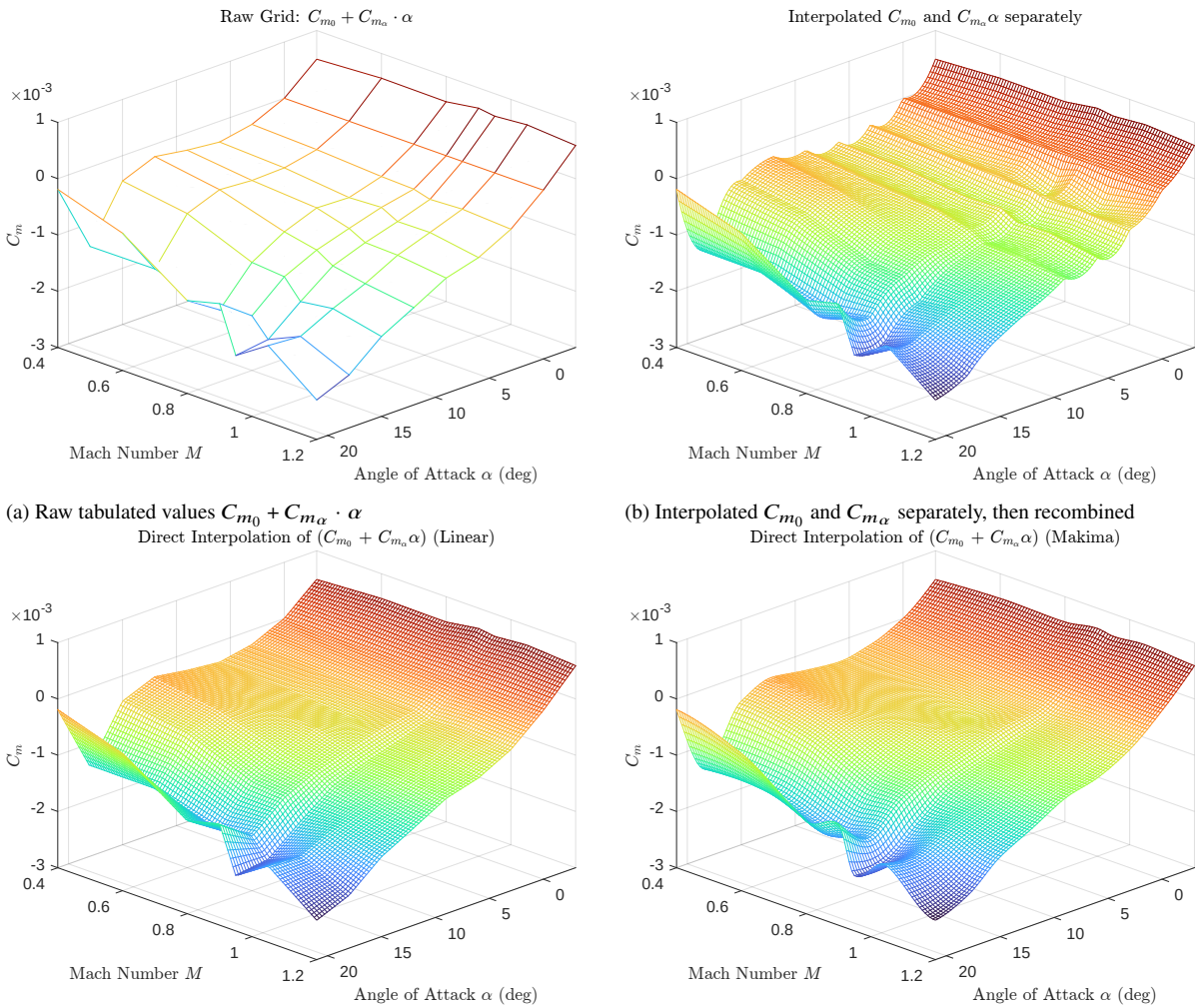
To be used in simulation, the aerodynamic dataset must be interpolated so that coefficient values are available throughout the flight envelope. The tabulated aerodynamic coefficients are defined at discrete combinations of angle of attack α and Mach number M, which requires constructing a two-dimensional interpolation surface. A typical approach is to interpolate each partial derivative term (e.g., C_{m_0}, C_{m_α}) independently and then reconstruct the total coefficient using a linear combination such as $C_m = C_{m_0} + C_{m_\alpha} \cdot \alpha$.

While this method may appear modular, it introduces structural inconsistencies. The partial derivatives are derived from local linearizations valid only at specific grid points. Interpolating them separately and combining them at arbitrary off-grid locations can distort the intended behavior of the aerodynamic model. This is especially evident in regions with strong curvature or cross-coupling between α and M, where the recombined surfaces exhibit slope inversions and local extrema that contradict the original data.

This issue is illustrated in Fig. 4.2. Subfigure 4.2a shows the raw tabulated surface for $C_{m_0} + C_{m_\alpha} \cdot \alpha$, which correctly captures the negative slope expected in the subsonic regime. Subfigure 4.2b, however, shows the effect of interpolating C_{m_0} and C_{m_α} independently and recombining them afterward. This surface contains artificial undulations and local slope reversals, even though the original partials are strictly monotonic. These artifacts arise because the interpolated partials no longer retain the internal consistency of the linear model structure. This problem can be overcome by first computing the total coefficient expressions $C = C_0 + C_\alpha \cdot \alpha$ at each discrete (α, M) grid point. These precomputed values were then interpolated directly using standard surface interpolation. This eliminated the inconsistencies caused by independent interpolation of partials, as shown in Subfig. 4.2c. In the current implementation, shown in Subfig. 4.2d, the total coefficient $C = C_0 + C_\alpha \cdot \alpha$ is interpolated directly using the modified Akima method (Makima). This method preserves the shape of the surface and provides smooth first derivatives. This approach maintains consistency with the tabulated aerodynamic model while providing smoothness for use in a simulation environment.

Propulsion System Model

The engine approximates a generic combined-cycle propulsion system. It transitions automatically between turbojet (Mach 0–2), ramjet (Mach 2–6), and scramjet (Mach > 6) operation based on Mach



(a) Raw tabulated values $C_{m_0} + C_{m_\alpha} \cdot \alpha$

(b) Interpolated C_{m_0} and C_{m_α} separately, then recombined

(c) Linear interpolation of the full C_m surface

(d) Makima interpolation of the full C_m surface

Figure 4.2: Comparison of interpolation methods for the pitch moment coefficient C_m.

number. The inlet is modeled as a variable-geometry system whose capture area coefficient C_a reflects the effective shockwave structure under the vehicle forebody and varies with both Mach number and angle of attack. The full thrust expression is given by Eq. (4.60):

$$f_P = 0.029\,\delta_t\, I_{sp}(M, \delta_t)\, g_0\, \rho\, Ma\, C_a(M, \alpha_A)\, A_c. \quad \delta_t \in [0.05, 2] \tag{4.60}$$

The thrust model in Eq. (4.60) expresses the net axial force as a product of multiple factors. The commanded throttle δ_t, with idle at 0.05, is scaled by 0.029 to yield a stoichiometrically normalized mass flow rate. This is multiplied by the specific impulse I_{sp}, the standard gravitational acceleration g_0, and the true airspeed Ma, where M is the Mach number and a is the local speed of sound. The capture-area coefficient C_a accounts for inlet shock losses, and A_c is the fixed engine cowl area. The resulting force acts along the 1^B-axis and is applied through the vehicle's center of gravity.

The capture-area coefficient C_a is defined over a two-dimensional grid in angle of attack and Mach number, as given in Eq. (4.59), and is interpolated using the same method employed for the aerodynamic coefficients. The specific impulse I_{sp} is obtained from a separate Mach–throttle grid and uses the same interpolation scheme. The throttle grid is defined as:

$$\delta_t = \big[\,0,\ 0.25,\ 0.5,\ 0.75,\ 1.0,\ 1.25,\ 1.5,\ 1.75,\ 2.0\,\big]. \tag{4.61}$$

In addition to force generation, the propulsion system governs the evolution of the vehicle's mass and inertia properties. The fuel mass is updated dynamically based on the instantaneous thrust output and the specific impulse I_{sp}, according to:

$$\dot{m}_f = \frac{f_P}{I_{sp}\, g_0}. \tag{4.62}$$

The fuel mass flow rate $\dot{m}_f$ is integrated to obtain the fuel consumption m_f. The total mass m decreases linearly with fuel consumption:

$$m(t) = m_0 - m_f(t), \tag{4.63}$$

where m_0 is the initial mass and $m_f(t)$ is the cumulative fuel mass burned. As the fuel depletes, the vehicle's moment of inertia matrix $[I_B^B]^B$ evolves as a function of fuel mass fraction. This is modeled by linear interpolation between the inertia matrix at full fuel load, denoted $[I_{B,0}^B]^B$, and at dry mass $[I_{B,1}^B]^B$:

$$[I_B^B]^B(t) = [I_{B,0}^B]^B + \left(\frac{m_f(t)}{m_{f,0}}\right)\left([I_{B,1}^B]^B - [I_{B,0}^B]^B\right). \tag{4.64}$$

A simulation stop condition is triggered automatically when the remaining fuel falls below zero.

Actuator Model

The vehicle contains one rudder and two elevons located at the trailing edge of the wing. The elevons function simultaneously as elevator and aileron control surfaces. Their symmetric and antisymmetric combinations define the elevator and aileron control inputs, respectively, according to:

$$\delta_e = \frac{\delta_{vl} + \delta_{vr}}{2}, \quad \delta_a = \frac{\delta_{vl} - \delta_{vr}}{2}, \tag{4.65}$$

where δ_{vl} and δ_{vr} are the left and right elevon deflections, both defined as positive in the upward direction. The rudder deflection δ_r is defined to be positive for a trailing-edge right deflection. Each control surface is subject to deflection limits of $\pm 20°$ and rate limits of $\pm 150°/\mathrm{s}$. To realistically capture actuator dynamics, all surfaces are modeled using a second-order system with rate and position limits. The actuator transfer function from commanded deflection δ_c to realized deflection δ is given by:

$$A(s) = \frac{\delta(s)}{\delta_c(s)} = \frac{\omega_{act}^2}{s^2 + 2\zeta_a \omega_{act} s + \omega_{act}^2}, \tag{4.66}$$

where $\omega_{act} = 50\,\mathrm{rad/s}$ is the natural frequency and $\zeta_a = 0.707$ is the damping ratio. To enforce physical constraints, the implementation includes:

- A saturation block limiting the actuator deflection δ to $\pm 20°$.
- A rate limiter applied to the actuator input signal $(\delta^{\mathrm{cmd}} - \delta)$, limiting the rate at which the actuator dynamics are driven.
- The second-order actuator is realized using two integrators, where the position integrator is constrained by the deflection limit and the rate integrator is constrained by the rate limit.

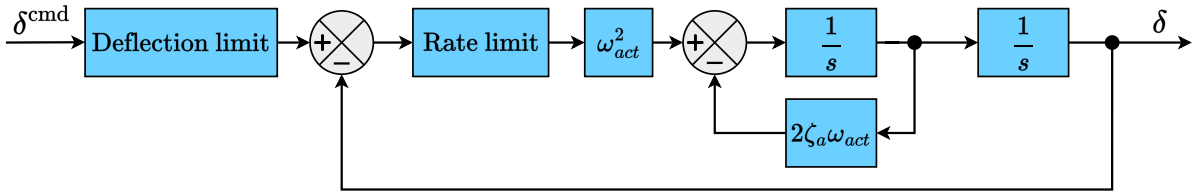


Figure 4.3: Actuator model with deflection and rate limits, damping feedback, and position feedback.

4.5. Sensors

An Inertial Measurement Unit (IMU) provides measurements of angular rates and linear accelerations by means of gyroscopes and accelerometers integrated in a single device. Once processed into navigation information, it becomes an Inertial Navigation System (INS). Errors in the measurements and computations corrupt the navigation solution. The GHAME model includes a strapdown INS, as implemented in the reference code provided by Zipfel [2025]. Unlike gimbaled systems, the strapdown INS mounts its inertial instruments rigidly on the vehicle body axes and numerically integrates the equations of motion to determine navigation states. The accelerometer cluster consists of three instruments aligned nominally with the body axes to sense the specific force, while the gyro cluster consists of three rotary devices measuring the inertial angular velocity about the same axes.

IMU Location and the Grubin Transformation

In this simulation all kinematic and dynamic quantities are computed directly at the CG, and the IMU is therefore assumed to be located at the CG. Under this assumption, no correction for displacement is required. In practice, however, the IMU is rarely mounted exactly at the CG, and the measured accelerations then include additional angular acceleration and centrifugal contributions. The Grubin transformation provides the relation between the acceleration at the CG and the acceleration at any displaced point in the body [Zipfel, 2025, p. 152]. This makes it possible, for example, to compute the accelerations experienced by the pilots in the cockpit or by an IMU mounted off-center.

The relation between the acceleration at the IMU and at the center of gravity is:

$$\boldsymbol{a}^I_{B_{\text{IMU}}} = \boldsymbol{a}^I_B - \left(D^I \boldsymbol{\omega}^{BI} \times \boldsymbol{s}_{BB_{\text{IMU}}} + \boldsymbol{\omega}^{BI} \times \left(\boldsymbol{\omega}^{BI} \times \boldsymbol{s}_{BB_{\text{IMU}}} \right) \right), \tag{4.67}$$

where $\boldsymbol{s}_{BB_{\text{IMU}}}$ denotes the displacement from the CG to the IMU (or, more generally, to any point of interest). Equation (4.67) indicates that the acceleration at a displaced point equals the CG acceleration corrected by angular acceleration and centrifugal contributions due to the vehicle's rotation.

Center of Gravity

The CG calculation is necessarily approximate, since the exact dry CG of the GHAME is not reported in the literature. A reasonable estimate can be obtained using the information provided in White et al. [1992]. Two aspects are important for this estimate. First, all fuel is located exclusively in the cylindrical fuselage of $L_{\text{cyl}} = 120$ ft length and $D = 20$ ft diameter, beginning immediately aft of the conical nose with half-angle $\theta = 10°$. The centroid of this uniform tank is therefore:

$$x_f = \frac{D/2}{\tan\theta} + \tfrac{1}{2} L_{\text{cyl}} = \frac{10}{\tan 10°} + 60 \approx 116.7 \text{ ft}. \tag{4.68}$$

The second aspect is that the GHAME mass distribution is known to follow that of the XB-70. The dry CG of the XB-70 is $x_d = 123.17$ ft for a total length of $L = 196$ ft, corresponding to a nondimensional fraction $x_d/L = 0.628$. This fraction can be applied as a scaling factor to the GHAME configuration. For a total length of $L = 233.4$ ft this gives a dry centroid of:

$$x_d = 0.628 \cdot 233.4 \approx 146.7 \text{ ft}. \tag{4.69}$$

With the fuel and dry mass centroids established, the takeoff mass centroid can be computed as a weighted average. With a fuel mass of $m_f = 180{,}000$ lb and a dry mass of $m_d = 120{,}000$ lb, the takeoff mass is $m_{\text{TO}} = 300{,}000$ lb. The combined CG at takeoff is given by:

$$x_{\text{MTOW}} = \frac{m_d x_d + m_f x_f}{m_d + m_f} = \frac{120{,}000 \cdot 146.7 + 180{,}000 \cdot 116.7}{300{,}000} \approx 128.7 \text{ ft}. \tag{4.70}$$

At burnout, only the dry mass remains and the CG coincides with $x_d = 146.7$ ft. The results are summarized in Table 4.1.

Table 4.1: Center of gravity locations for the GHAME aircraft

Condition	**Mass, lb**	**CG, ft**	**CG, %MAC**
Takeoff (fuel + dry)	300,000	128.7	16.9%
Burnout (dry only)	120,000	146.7	41.9%
Nominal reference CG	–	140.3	33.0%

Thus, the center of gravity migrates aft from 128.7 ft (16.9% MAC) at takeoff to 146.7 ft (41.9% MAC) at burnout, corresponding to a total shift of 18.0 ft, or approximately 25% MAC. The nominal reference CG at 0.33 MAC (140.3 ft) lies between these values.

IMU Error Modeling

The accelerometer error is modeled as:

$$[\varepsilon f_{sp}]^B = [\varepsilon b_a]^B + [\varepsilon n_a]^B + \left([S_a]^B + [M_a]^B\right)[f_{sp}]^B, \tag{4.71}$$

where $[\varepsilon b_a]^B$ denotes the bias instability and $[\varepsilon n_a]^B$ velocity random walk, $[S_a]^B$ is the diagonal scale factor error matrix, and $[M_a]^B$ is the skew-symmetric misalignment matrix. The specific force output is then simply the sum of the true specific force and the instrument error:

$$[\hat{f}_{sp}]^B = [f_{sp}]^B + [\varepsilon f_{sp}]^B. \tag{4.72}$$

The angular velocity error is modeled as:

$$[\varepsilon \omega^{BI}]^B = [\varepsilon b_g]^B + [\varepsilon n_g]^B + \left([S_g]^B + [M_g]^B\right)[\omega^{BI}]^B + [U_g]^B[f_{sp}]^B, \tag{4.73}$$

with $[\varepsilon b_g]^B$ the bias instability, $[\varepsilon n_g]^B$ the angular random walk, $[S_g]^B$ the diagonal scale factor error matrix, $[M_g]^B$ the skew-symmetric misalignment matrix, and $[U_g]^B$ the diagonal imbalance matrix coupling linear acceleration into the angular rate measurement. The misalignment reflects only the cluster error of the otherwise orthogonal gyro triad. The final gyro output is therefore:

$$[\hat{\omega}^{BI}]^B = [\omega^{BI}]^B + [\varepsilon \omega^{BI}]^B. \tag{4.74}$$

All errors are modeled as zero-mean Gaussian processes. The parameters adopted here are representative of a tactical-grade IMU suitable for hypersonic flight [Zipfel, 2025].

Table 4.2: Gyroscope and accelerometer error parameters (1σ values) used in the strapdown INS

Symbol	Description	Distribution	Units
$[M_g]^B$, $[M_a]^B$	Gyro and accel. misalignment	$\mathcal{N}(0,\ 0.11)$	mrad
$[S_g]^B$	Gyro scale factor	$\mathcal{N}(0,\ 20)$	ppm
$[S_a]^B$	Accel. scale factor	$\mathcal{N}(0,\ 500)$	ppm
$[\varepsilon b_g]^B$	Gyro bias	$\mathcal{N}(0,\ 0.206)$	deg/hr
$[\varepsilon b_a]^B$	Accel. bias	$\mathcal{N}(0,\ 3.56 \times 10^{-3})$	m/s^2
$[\varepsilon n_g]^B$	Gyro noise	$\mathcal{N}(0,\ 0.15)$	deg/$\sqrt{\text{hr}}$
$[\varepsilon n_a]^B$	Accel. noise	$\mathcal{N}(0,\ 2.5 \times 10^{-3})$	m/s^2

The specific error magnitudes adopted in this study are summarized in Table 4.2. In the simulation, these errors are added to the sensor outputs after passing through the respective sensor dynamics.

Sensor Dynamics

In the present simulation only the rate gyroscopes and accelerometers are modeled with explicit sensor dynamics. These devices are subject to two main effects: filtering to prevent aliasing and computation delays introduced by the sensor processor.

The anti–aliasing filter is required because the IMU signals are sampled at a fixed interval $\Delta t = 0.01$ s. The corresponding sampling frequency and Nyquist frequency are:

$$\omega_s = \frac{2\pi}{\Delta t} = 628.32 \text{ rad/s}, \qquad \omega_N = \frac{\omega_s}{2} = 314.16 \text{ rad/s}. \tag{4.75}$$

Any signal content above ω_N will fold into the lower frequency band, producing aliasing in the sampled data. Since sensor noise is broadband and not naturally band–limited, a low–pass filter is required before sampling. To guarantee sufficient attenuation of out–of–band noise, the cutoff is conservatively placed at half the Nyquist frequency, $\omega_a = \frac{\omega_N}{2} = 157.08$ rad/s. The anti–aliasing dynamics are represented as a first–order low–pass transfer function:

$$G_a(s) = \frac{\omega_a}{s + \omega_a}. \tag{4.76}$$

In addition to filtering, the IMU introduces a finite computation delay. This effect can be represented exactly by a pure time delay which is given by the nonrational transfer function:

$$G_{\text{SD}}(s) = e^{-\tau_{SD} s}, \tag{4.77}$$

which is referred to as the sensor delay transfer function. The exact duration of this sensor delay τ_{SD} is uncertain and therefore not modeled as a fixed constant, but it is expected to be on the order of one to several sampling intervals. Since its precise value cannot be specified a priori, the sensor delay is not fixed in the nominal simulation model. Instead, it is varied parametrically in the high-fidelity nonlinear simulation studies to evaluate how well the controller can tolerate such effects.

The overall sensor model therefore consists of a first-order low-pass filter for anti-aliasing, followed by the optional application of the sensor delay transfer function when robustness to measurement delays is investigated.

4.6. Initialization and Trimming

Before any dynamic simulation can be performed, the vehicle must be placed in a steady-flight condition that satisfies all physical and dynamic equilibrium constraints. This is achieved through two coupled

procedures, initialization and trimming. Initialization specifies the vehicle's geometric and kinematic configuration from known parameters, providing the foundation for the trimming routine. Trimming then computes the equilibrium states and corresponding control inputs that bring aerodynamic, gravitational, and propulsive forces into balance for the prescribed flight condition.

The model requires initialization of four key quantities: the inertial body rate $[\omega^{BI}]^B$, the inertial velocity $[v_B^I]^I$, the inertial position $[s_{BI}]^I$, and the inertial-to-body transformation matrix $[T]^{BI}$. These quantities are derived from known inputs such as the vehicle's longitude l, geodetic latitude λ_d, altitude h, and Greenwich longitude l_G, together with the geodetic Euler angles ψ_{bd}, θ_{bd}, and ϕ_{bd}. The initial geographic speed $|v_B^E|$ must also be specified, along with either the kinematic flight-path and heading angles (γ_K, χ_K) or the kinematic angle of attack and sideslip angle (α_K, β_K).

1. Inertial Position in Inertial Coordinates

The inertial position vector $[s_{BI}]^I$ is computed using the longitude l, geodetic latitude λ_d, geodetic altitude h, and the Greenwich hour angle Ξ_0. The deflection of the normal δ and the Earth's radius to the ellipsoidal surface R_0 at the given latitude are defined in Eqs. (3.14)–(3.15). With these quantities, the displacement vector from the Earth's center to the vehicle, expressed in geodetic coordinates, is:

$$[s_{BI}]^D = \begin{bmatrix} -(R_0 + h)\sin\delta \\ 0 \\ -(R_0 + h)\cos\delta \end{bmatrix}. \tag{4.78}$$

At initialization time $t = 0$, the celestial longitude of the vehicle is:

$$l_i = \Xi_0 + l. \tag{4.79}$$

The inertial position vector is then obtained by applying the transpose of the transformation matrix from geodetic to inertial coordinates, $[\overline{T}]^{DI}$, evaluated at λ_d and l_i, as defined in Eq. (3.23):

$$[s_{BI}]^I = [\overline{T}]^{DI}[s_{BI}]^D. \tag{4.80}$$

2. Inertial Velocity in Inertial Coordinates

The inertial velocity can be initialized in two ways depending on the available data. In both cases, the geographic speed $|v_B^E|$ is assumed known. When the kinematic heading χ_K and flight path angle γ_K are specified, the velocity vector in geodetic coordinates is:

$$[v_B^E]^D = |v_B^E| \begin{bmatrix} \cos\gamma_K \cos\chi_K \\ \cos\gamma_K \sin\chi_K \\ -\sin\gamma_K \end{bmatrix}. \tag{4.81}$$

Alternatively, when the kinematic angle of attack α_K, sideslip angle β_K, and Euler angles $(\phi_{bd}, \theta_{bd}, \psi_{bd})$ are known, the velocity vector in body coordinates is:

$$[v_B^E]^B = |v_B^E| \begin{bmatrix} \cos\alpha_K \cos\beta_K \\ \sin\beta_K \\ \sin\alpha_K \cos\beta_K \end{bmatrix}, \tag{4.82}$$

which is then transformed into geodetic coordinates via:

$$[v_B^E]^D = [\overline{T}]^{BD}[v_B^E]^B, \tag{4.83}$$

where $[T]^{BD}$ is the geodetic-to-body transformation matrix defined in Eq. (3.26). Once $[v_B^E]^D$ is obtained, the inertial velocity in inertial coordinates follows as:

$$[v_B^I]^I = [\overline{T}]^{DI}[v_B^E]^D + [\Omega^{EI}]^I[s_{BI}]^I, \tag{4.84}$$

where $[\Omega^{EI}]^I$ is the skew-symmetric matrix of the Earth's rotation rate:

$$[\Omega^{EI}]^I = \begin{bmatrix} 0 & -\omega_\oplus & 0 \\ \omega_\oplus & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{4.85}$$

3. Transformation Matrix from Inertial to Body Coordinates

The initial transformation matrix from inertial to body-fixed coordinates is constructed as:

$$[T]^{BI} = [T]^{BD}[T]^{DI}, \tag{4.86}$$

with both components initialized in the preceding steps.

4. Angular Velocity in Body Coordinates

The angular velocity of the body with respect to the inertial frame, expressed in body coordinates, is:

$$[\omega^{BI}]^B = [\omega^{BE}]^B + [T]^{BI}[\omega^{EI}]^I, \tag{4.87}$$

where $[\omega^{BE}]^B$ denotes the angular velocity of the body relative to the Earth-fixed frame, expressed in body coordinates. For initialization, these rates are set to zero. The Earth's rotation vector in inertial coordinates is given by $[\omega^{EI}]^I = [0\ 0\ \omega_\oplus]^\top$, and the transformation matrix $[T]^{BI}$ maps this vector into the body coordinate system.

Trimming Methodology

After initialization, the trimming routine computes the states and control inputs that satisfy the steady-state solution of the nonlinear equations of motion for the prescribed Mach number and altitude. A steady-flight condition is required to define a physically consistent operating point about which the aircraft dynamics can be analyzed. In particular, trim conditions provide the equilibrium states used for performance analysis, control design, and linearization of the nonlinear equations of motion. The initialization procedure is then repeated using the computed trim conditions to ensure that the simulation begins from a consistent steady-flight equilibrium suitable for subsequent time-domain analyses and controller evaluations.

A trim condition consists of two components: the prescribed flight condition and the imposed steady-state constraints. The prescribed flight condition is defined by the freestream Mach number, geodetic altitude, and fuel fraction,

$$\boldsymbol{\rho} = \begin{bmatrix} M & h & f_{\text{fuel}} \end{bmatrix}. \tag{4.88}$$

The steady-state constraints require all states to remain constant in time except for the inertial z_I-coordinate, as the aircraft is in steady forward flight. All trim solutions are computed assuming a stationary atmosphere with no wind.

The trim problem is initialized by setting the angular rates p, q, r, roll angle ϕ_{bd}, yaw angle ψ_{bd}, and sideslip angle β_K to zero. The sideslip and yaw angle are free to choose. The angular rates are fixed, and the constraints $\dot{p} = \dot{q} = \dot{r} = 0$ ensure a non-rotating steady state, while $\dot{\beta}_K = 0$ enforces steady sideslip. The free trim variables are the angle of attack α_K, pitch attitude θ_{bd}, and the control inputs $\delta_a, \delta_e, \delta_r, \delta_T$. This gives six trim variables for six imposed derivative constraints.

Table 4.3: Trim variables and prescribed quantities

States	Steady State	Known
p	✓	✓
q	✓	✓
r	✓	✓
ϕ	✓	✓
θ	✓	×
ψ	✓	✓
V_K	✓	✓
α_K	✓	×
β_K	✓	×
x_I	✓	✓
y_I	✓	✓
z_I	×	✓

Inputs	Known
δ_T	×
δ_a	×
δ_e	×
δ_r	×

The throttle is bounded between 0.05 (idle) and 2 (maximum), while the aileron, elevator, and rudder deflections are each limited to ±20°. The angle of attack is additionally restricted to the aerodynamic range of $-3° \leq \alpha_K \leq 21°$ to prevent going outside the given aerodataset.

The trim condition is computed using the function `jj_trim` from the open source TrimMod package [Buchholz, 2026][2]. The trim problem is formulated as a square nonlinear system, meaning that the number of unknowns matches the number of constraints, so that a unique solution can be determined. At each iteration, the nonlinear model is evaluated at the current estimate of the trim variables to obtain the corresponding state derivatives. The residual vector is defined as the deviation of the selected derivatives from their desired values. The objective is to drive this residual vector to zero. A local linear approximation of the mapping from variables to residuals is constructed by computing the Jacobian matrix numerically using finite differences. Each trim variable is perturbed individually, and the resulting change in the residual vector is used to estimate the sensitivities. The correction step is obtained by solving the linear system:

$$\boldsymbol{J}\,\Delta z = \boldsymbol{r}, \tag{4.89}$$

where $\boldsymbol{J}$ is the Jacobian, Δz is the increment in the trim variables, and $\boldsymbol{r}$ is the residual vector. The trim variables are updated as:

$$z_{k+1} = z_k + \Delta z. \tag{4.90}$$

If the Jacobian becomes ill-conditioned or the residual increases, the algorithm applies step-size reduction through bisection. Additional safeguards enforce maximum step sizes and hard bounds on states and inputs, ensuring that all iterates remain physically feasible. The procedure is repeated until the maximum absolute residual falls below the prescribed tolerance. Convergence depends on the initial guess, and accurate initial values for α_K and θ_{bd} significantly improve the speed.

The resulting trim routine yields equilibrium states and control inputs that satisfy the steady-flight condition defined in Eq. (4.88). Three representative flight conditions within the considered envelope are analyzed to characterize the trimmed response. The selected cases span Mach numbers from 3 to 10 and altitudes from 60,000 ft to 110,000 ft, thereby covering a broad range of hypersonic operating conditions. These trim points are chosen to represent distinct regions of the flight regime and provide a basis for assessing controller performance under different operating conditions. The resulting trimmed states serve as reference points for the subsequent simulations and analyses presented throughout this work.

[2]The MATLAB `TrimMod` package is available through MATLAB File Exchange: https://nl.mathworks.com/matlabcentral/fileexchange/268-trimmod. The primary motivation for using this package is that it does not require the Simulink Control Design Toolbox.

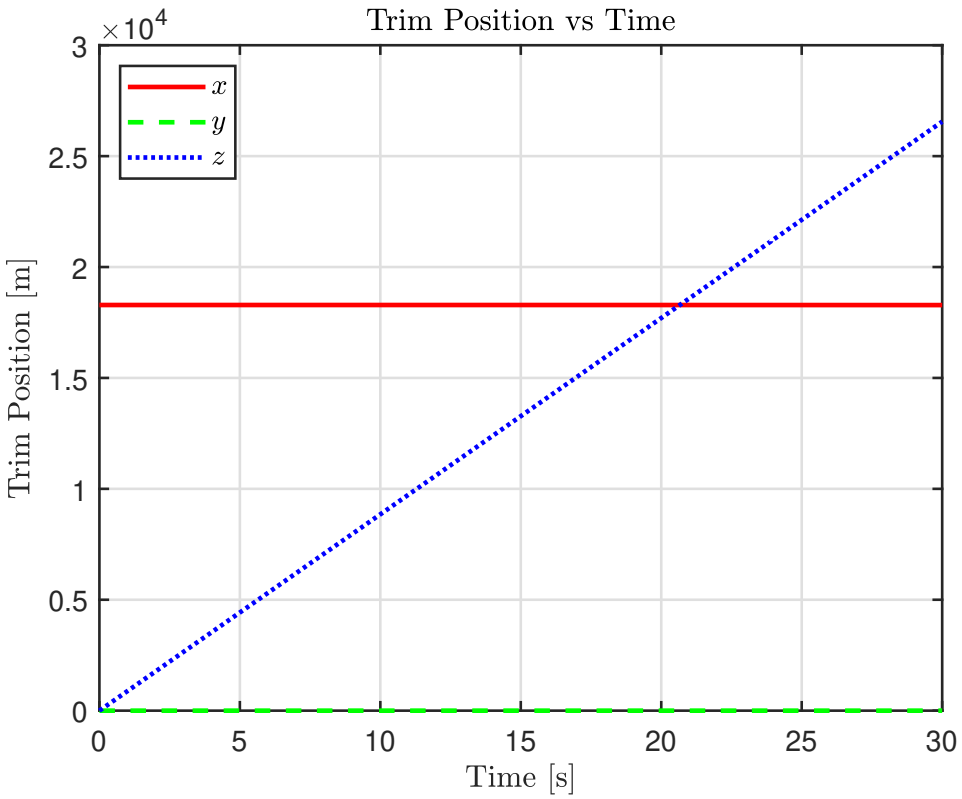


Figure 4.4: Trimmed inertial position response for the Mach 3 case at 60,000 ft.

The inertial position response for the Mach 3 case at an altitude of 60,000 ft is depicted in Fig. 4.4. The plot shows the time histories of the position vector components x, y, and z. For clarity, the mean Earth radius has been subtracted from the position magnitude so that the figure emphasizes the relative displacement about the trimmed operating point rather than the absolute geocentric distance. This representation highlights how the vehicle maintains its equilibrium state in the inertial coordinate system.

A broader summary of the trim settings for all three flight conditions is provided in Table 4.4. The table lists the throttle setting, aileron, elevator, and rudder deflections, along with the trimmed angle of attack α_K and pitch angle θ_{bd}. For level flight α_{trim} and θ_{trim} coincide.

Table 4.4: Trim settings for representative flight conditions

Condition	Throttle	δ_{trim}, deg	α_{trim}, deg	θ_{trim}, deg
Mach 3, 60,000 ft	0.8894	[0.0, -5.1039, 0.0]	3.45	3.45
Mach 6, 90,000 ft	0.7468	[0.0, -6.5197, 0.0]	8.10	8.10
Mach 10, 110,000 ft	0.9080	[0.0, -8.0474, 0.0]	9.40	9.40

4.7. Model Verification

Before the simulation can be used for analysis or control design, the model must be verified to ensure correct implementation of the underlying equations and algorithms. The development of the present model followed the methodology and equations described in Zipfel [2025]. To verify that these formulations were implemented correctly, the Simulink model is compared against the GHAME flight dynamics simulations independently developed by Zipfel himself, first in Fortran and later in C++. For verification in this project, the most complete available version of the C++ simulation is used[3].

A common approach is to run both implementations with the same initial conditions and compare their output time histories. If no errors are present they should naturally be the same. However, while this will reveal discrepancies, it does not show where they originate. In such a large nonlinear system, small mismatches can cause rapid errors, making it difficult to trace errors to a specific subsystem. For this reason, verification should be integrated into the development process from the start. A more effective strategy is to test smaller components individually, using unit tests to confirm that elements such as transformation matrices, aerodynamic models, actuator dynamics, and integration schemes produce correct results for known inputs. This avoids building the full system on top of faulty subsystems. Therefore, the Simulink model was developed incrementally, with each module locally verified against the C++ implementation before evaluating the overall simulation.

To ensure meaningful comparisons between Simulink and C++, the simulation conditions were matched exactly. This included setting the integration step to 0.01 s and hard-coding all physical constants, unit conversions, and numerical tolerances to the same values as in the C++ code. Examples include fixed definitions for π, radian-to-degree conversions, and other constants which are all provided in Appendix A. The used integration scheme in the C++ code is the modified Euler method which was replicated in Simulink by selecting Heun's method:

$$y_{n+1} = y_n + \tfrac{1}{2}(f_{n+1} + f_n) \cdot \Delta t, \tag{4.91}$$

The methodology for all verification tests is to apply identical inputs to both implementations, log the outputs, and compare them element by element. The expected result is agreement within a predefined tolerance ε, while the actual result is given as the maximum absolute error. When the expected result matches the actual result, the test is marked as **Pass**. If a test does not pass, the corresponding subsystem is inspected until the error is found and corrected, after which the verification is repeated until it is done.

As an example, the environment module was verified by isolating the original C++ code and supplying predefined test arrays for time, inertial position, velocity, and geodetic altitude. The outputs, gravitational acceleration, air density, pressure, and temperature, were logged and compared element by element with those from the Simulink model. A wide range of geodetic altitudes was included to confirm that the atmospheric model produced correct results across all layers of the atmosphere. A difference between the Simulink model is that wind is not constant but modeled using the `HWM14` block which means that cannot be verified. The results are summarized in Table 4.5.

[3]Simulation files are available at: https://arc.aiaa.org/doi/suppl/10.2514/4.107535

Table 4.5: Environment module verification tests

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T1.1	Computation of gravitational acceleration g	Inertial position $[s_{BI}]^I$	Matches reference within $\varepsilon = 10^{-6}\,\mathrm{m/s^2}$	Max. absolute error $< 10^{-6}\,\mathrm{m/s^2}$	Pass
T1.2	Computation of pressure p and speed of sound a	Geodetic altitude h	Matches reference within $\varepsilon = 10^{-6}$ Pa and $10^{-6}\,\mathrm{m/s}$	Max. absolute error $< 10^{-6}$ Pa and $10^{-6}\,\mathrm{m/s}$	Pass

The verification philosophy followed a bottom–up approach. Once a module was verified against reference data, it was regarded as a trusted component and subsequently used as the foundation for testing higher-level functionality. After the environment module had been verified, the same procedure was applied to the kinematics module.

Table 4.6: Verification tests for the kinematics module

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T2.1	Computation of DCM $[T]^{DI}$	Geodetic latitude, longitude, geodetic altitude, time	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$ for each entry	Pass
T2.2	Computation of DCM $[T]^{BD}$	Geodetic yaw, pitch, and roll angles	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$ for each entry	Pass
T2.3	Computation of DCM $[T]^{BI}$	Time, inertial body rate $[\omega^{BI}]^B$, DCMs $[T]^{BD}$ and $[T]^{DI}$	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$ for each entry	Pass
T2.4	Computation of airspeed relative velocity $[v_B^A]^B$	Geographic velocity $[v_B^E]^D$, wind velocity $[v_A^E]^D$, DCM $[T]^{BD}$	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$ m/s	Pass
T2.5	Computation of aerodynamic angles α_A, β_A	Airspeed relative velocity $[v_B^A]^B$	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$ rad	Pass
T2.6	Computation of Euler angles $\psi_{\mathrm{bd}}, \theta_{\mathrm{bd}}, \phi_{\mathrm{bd}}$	DCM $[T]^{BD}$	Matches reference within $\varepsilon = 10^{-6}$; numerically stable when $\theta_{\mathrm{bd}} \to \pm 90°$	Max. absolute error $< 10^{-6}$ rad; no instability or NaN behavior detected	Pass

With the environment and kinematics modules verified, these components were integrated with the propulsion module. Verification of the propulsion module focused on several key aspects: the interpolation of the specific impulse, the interpolation of the capture area coefficient, the correctness of the thrust calculation itself, and the resulting variations in vehicle mass and moment of inertia due to fuel depletion.

Table 4.7: Verification tests for the propulsion module

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T3.1	Interpolation of specific impulse and capture area	Mach number, throttle, angle of attack	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error < 10^{-6}	Pass
T3.2	Computation of thrust	Air density, throttle, airspeed $\lvert v_B^A \rvert$, SPI and C_a	Matches reference within $\varepsilon = 10^{-6}$ N	Max. absolute error < 10^{-6} N	Pass
T3.3	Computation of expended fuel mass	Time series of thrust values and specific impulse	Matches reference within $\varepsilon = 10^{-6}$ kg	Max. absolute error < 10^{-6} kg	Pass
T3.4	Interpolation of moment of inertia tensor	Initial and burn-out MOI, total and expended fuel mass	Matches reference within $\varepsilon = 10^{-6}$ kg·m^2	Max. absolute error < 10^{-6} kg·m^2	Pass

Table 4.8: Verification tests for the aerodynamics module

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T4.1	Interpolation behavior	Inputs between table grid points	No discontinuities or undefined values	All output values finite and continuous	Pass
T4.2	Clamping behavior at table limits	Inputs beyond upper/lower bounds of table grid points	Coefficients equal to values at nearest table edge; no extrapolation occurs	$c_y, c_{ll}, c_{lm}, c_{ln}$, c_l, c_d remain finite and constant	Pass
T4.3	Force coefficient computation	Angle of attack, Mach number, control deflections	Matches reference output within tolerance	Max. absolute error < 10^{-6}	Pass
T4.6	Force transformation to body-fixed coordinates	c_d, c_l, angle of attack	c_x and c_z match reference force transformation	Max. absolute error < 10^{-6}	Pass
T4.7	Moment coefficient computation	Angle of attack, Mach number, control inputs, angular rates	c_{ll}, c_{lm}, and c_{ln} match reference output within tolerance	Max. absolute error < 10^{-6}	Pass

The next module to be verified is the aerodynamics module, which computes the aerodynamic force and moment coefficients. Particular attention was given to verifying that the interpolation and clamping behavior is correct, especially in edge cases. This verification relied on the previously verified atmospheric and kinematic variables, together with the body rates with respect to the Earth, and control deflections provided as a time series from the Zipfel reference implementation.

Table 4.9: Verification tests for the Euler module

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T5.1	Integration of inertial body rate $[\omega^{BI}]^B$	Aero. moments $\boldsymbol{m}_B$, moment of inertia $\boldsymbol{I}_B^B$	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$	Pass
T5.2	Computation of body rate wrt Earth $[\omega^{BE}]^B$	$[\omega^{BI}]^B$, DCM $[T]^{BI}$, Earth rotation	Matches reference within $\varepsilon = 10^{-6}$	Max. absolute error $< 10^{-6}$	Pass

The next module to be verified is the Euler module, which has two primary tasks: the integration of the inertial body rates from the aerodynamic and control moments, and the computation of the body rates with respect to the Earth, which are required for the aerodynamic moment calculations.

Table 4.10: Verification tests for the Newton module

Test ID	Purpose	Inputs	Expected Result	Actual Result	Status
T6.1	Computation of DCM $[T]^{GI}$	Geod. latitude and altitude, DCM $[T]^{DI}$	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$	Pass
T6.2	Integration of inertial acceleration	Initial $[a_B^I]^I$, specific forces, gravity force, DCMs $[T]^{BI}$ and $[T]^{GI}$	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$ m/s^2	Pass
T6.3	Integration of inertial velocity	Initial $[v_B^I]^I$, inertial acceleration $[a_B^I]^I$	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$ m/s	Pass
T6.4	Integration of inertial position	Initial $[s_B^I]^I$, inertial velocity $[v_B^I]^I$	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$ m	Pass
T6.5	Computation of geodetic position	Inertial position $[s_B^I]^I$, time	Matches reference within $\varepsilon < 10^{-6}$	Max. abs. error $< 10^{-6}$ rad or m	Pass
T6.6	Computation of geographic velocity $[v_B^E]^D$	Inertial velocity and position, DCM $[T]^{DI}$, Earth rate	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$ m/s	Pass
T6.7	Computation of geographic angles	Geographic velocity $[v_B^E]^D$	Matches reference within $\varepsilon < 10^{-6}$	Max. absolute error $< 10^{-6}$ rad	Pass

Finally, the Newton module was verified by assessing the integration of inertial acceleration, velocity, and position, as well as the updates of geographic position and velocity, together with the computation of the geographic angles.

Discussion on unit tests

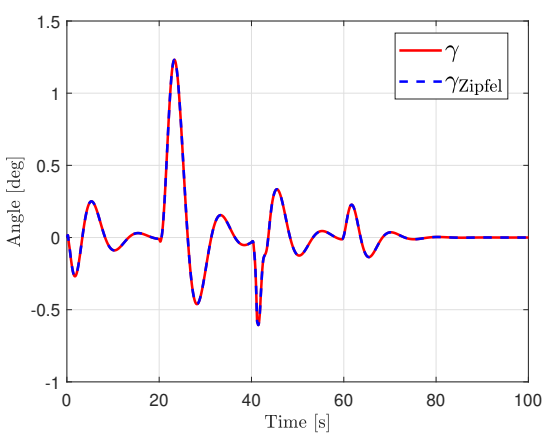
Across the verification of all modules, unit testing was especially helpful for uncovering many small mistakes that would otherwise have been difficult to trace. These included the use of inconsistent constants, such as Zipfel's different value of gravitational acceleration, as well as minor indexing and parameter errors. Catching such issues early saved considerable time, since they would have been far more difficult to isolate once the modules were integrated. At the same time, the approach also ensured that larger mistakes were identified quickly, providing confidence in the correctness of the overall implementation.

System-level verification using logged inputs

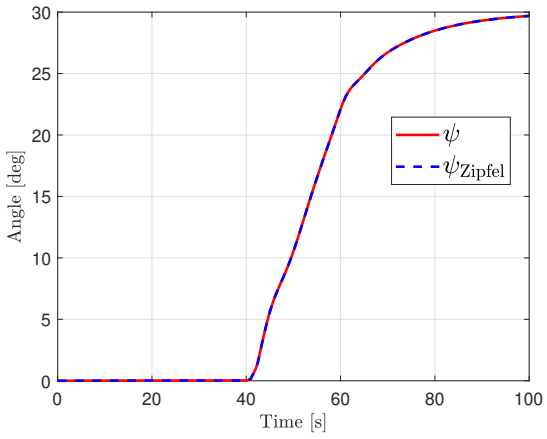
After verifying all individual building blocks in isolation, the final step was to connect the full simulation chain and test whether the integrated Simulink implementation could faithfully reproduce the behavior of the C++ reference model from A to Z. This was done by executing mission files in the reference model, which computed the actuator and throttle commands necessary to perform specific maneuvers. These inputs, aileron, elevator, rudder, and throttle, were then saved to file and fed, without modification, into the Simulink implementation under open-loop conditions. Because both systems started from identical initial states and shared the same environmental settings and vehicle parameters, any discrepancies in the resulting trajectories would indicate a modeling error. Zipfel does not trim his model so verification is performed without the trim routine.

The criterion for a successful test is strict: for identical input histories, the Simulink model must produce the same outputs, position, velocity, orientation, and mass, as the reference model. This mission scenario is a short 100s simulation to verify correct turn and climb behavior. The reference mission began in level northbound flight at Mach 3.3 and 10 km altitude. The autopilot (`maut = 45`) executed the following maneuvers:

1. $0\,\text{s} \leq t < 20\,\text{s}$: altitude and heading hold.
2. $20\,\text{s} \leq t < 40\,\text{s}$: elevator command initiates a 50 m climb (`altcom` raised from 10 000 m to 10 050 m).
3. $40\,\text{s} \leq t \leq 100\,\text{s}$: coordinated heading change of 30°.



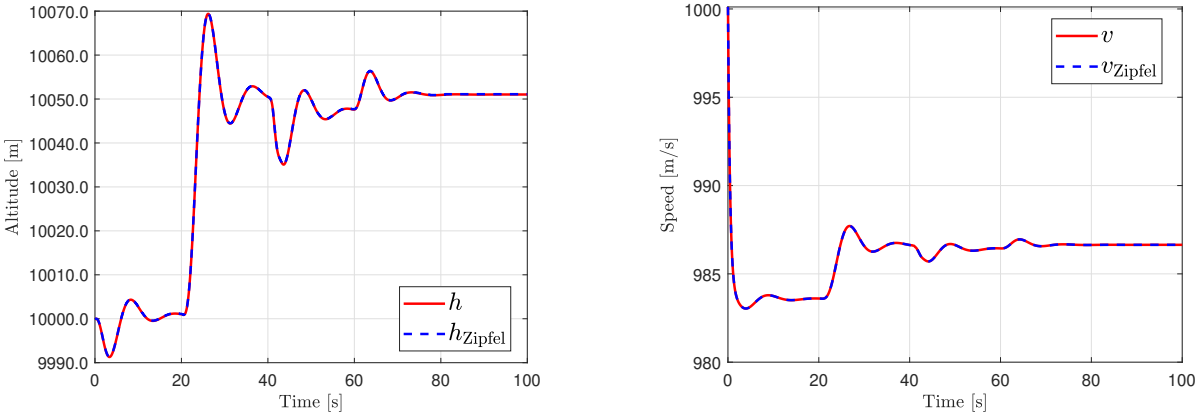
(a)



(b)

Figure 4.5: Comparison of the simulated flight path and heading responses during verification. Subfigure (a) shows the flight path response, and subfigure (b) shows the corresponding heading response.

All actuator and throttle inputs were logged and subsequently applied to the Simulink model. Verification was based on output agreement across all integrated states. The final response, after correcting the timing-related issues, is shown in Fig. 4.5 and Fig. 4.6. The Simulink model response is plotted in red, while the C++ Zipfel model response is shown in blue. The two responses are essentially identical. After 100 seconds, the altitude difference is approximately 4.0×10^{-7} m, which is sufficiently small to be attributed to rounding errors.



(a) (b)

Figure 4.6: Comparison of the simulated altitude and speed responses during verification. Subfigure (a) shows the altitude response over time, and subfigure (b) shows the corresponding speed response.

Discussion

At the system level, the main challenge was ensuring correct timing across modules. The Simulink implementation tended to update integrated states before dependent modules provided their new inputs, which led to inconsistencies in the data flow. Execution order timing in Simulink is difficult to control and not entirely transparent, so it is often unclear how the software determines the sequence of updates. In contrast, the Zipfel implementation, written in Fortran and C++, avoids this issue because the execution order is explicitly coded and therefore fixed. He enforces a fixed evaluation order that begins with the environment and kinematics modules and continues sequentially through the Newton and Euler modules. Replicating this ordering in Simulink was necessary to resolve the timing problem and ensure consistent results. This required explicitly reordering the execution sequence within the Simulink model to match the Zipfel implementation.

5

Flight Control System Design

This chapter presents the design of the Flight Control System (FCS) for the GHAME vehicle. The process begins with an overview of the control requirements, including the target maneuvers, performance bounds, and the rationale for the chosen control-law structure, as detailed in Sec. 5.1. The subsequent section justifies the selected architecture with reference to established results in hypersonic vehicle control and the assumptions used for controller synthesis. The derivation of the control laws for each loop is then given in Sec. 5.2. The design and tuning of the linear controllers embedded within the nonlinear inversion framework are described in Sec. 5.3. Finally, Sec. 5.4 addresses the digital control design, incorporating sampling and discretization effects into the synthesis.

5.1. Control System Selection

The FCS developed for the GHAME vehicle simulation were evaluated on three representative tasks:

(1) heading control (2) altitude hold (3) airspeed trimming

These tasks were conducted across multiple Mach numbers and altitudes, ensuring that the control architecture was tested under varying aerodynamic, propulsion, and stability conditions. Performance was evaluated using two benchmarks: adequate values, defining the minimum acceptable accuracy, and desired values, representing the target tracking precision. The corresponding specifications for the three tasks, originally defined as SR-71 design requirements are summarized in Table 5.1 and are adopted here as the relevant performance criteria for GHAME [Vu and Biezad, 1994].

Table 5.1: Performance specifications, adapted from Vu and Biezad [1994]

Flight condition	**Controlled parameter**	**Adequate performance**	**Desired performance**
Mach 3, altitude 60,000 ft	Target heading (8 deg)	±2 deg	±1 deg
	Target altitude	±400 ft	±200 ft
	Trim KEAS	±20 knots	±10 knots
Mach 6, altitude 90,000 ft	Target heading (4 deg)	±1 deg	±0.5 deg
	Target altitude	±600 ft	±200 ft
	Trim KEAS	±20 knots	±10 knots
Mach 10, altitude 110,000 ft	Target heading (2 deg)	±1 deg	±0.5 deg
	Target altitude	±1000 ft	±300 ft
	Trim KEAS	±20 knots	±10 knots

Abbreviation: KEAS, knots equivalent air speed.

The inclusion of both altitude-tracking and heading-tracking tasks requires regulation of both the longitudinal and the lateral directional dynamics. A controller acting only in the longitudinal axis would therefore be insufficient, and an architecture limited to inner-loop rate and attitude stabilization would also fall short, since outer-loop control of flight path and altitude is required.

The altitude–change tasks for the pilot evaluation are summarized in Table 5.2. Each maneuver was executed at a specified Mach number and initial altitude, with the pilot commanding a fixed climb or descent rate until the required altitude change was achieved. At Mach 3 and 60,000 ft, the pilot performed a 5,000 ft climb at a constant rate of 5,000 ft/min. At Mach 6 and 90,000 ft, two separate maneuvers were executed: a 10,000 ft descent at −10,000 ft/min and a 10,000 ft climb at 10,000 ft/min. Finally, at Mach 10 and 110,000 ft, the pilot performed a 20,000 ft climb at 10,000 ft/min. These altitude–change profiles were designed to evaluate the FCSs ability to maintain smooth vertical transitions under high–speed flight conditions.

Table 5.2: Altitude–change maneuvers, adapted from Vu and Biezad [1994]

Mach number	**Initial altitude, ft**	**Altitude change, ft**	**Vertical speed, fpm**
3	60,000	5,000	5,000
6	90,000	−10,000	−10,000
6	90,000	10,000	10,000
10	110,000	20,000	10,000

The purpose of this section is to justify the selection of the control architecture for the hypersonic vehicle studied in this work. The choice is guided directly by insights from the literature on NDI for hypersonic flight [Ye et al., 2018], with particular attention to how recent advances have addressed the well–known problem of unstable internal dynamics.

As outlined in Sec. 2.5, NDI for hypersonic vehicles has almost exclusively been applied to the longitudinal dynamics, with the objective of controlling velocity and either altitude or flight–path angle. This is typically achieved by differentiating altitude four times and velocity three times to obtain a decoupled, feedback–linearizable system. However, this formulation introduces unstable internal dynamics in the pitch channel, often taking the form of a hyperbolic–saddle short–period mode [Serrani, 2013]. To address this, various strategies have been explored: adding a canard to provide additional control authority, neglecting certain coupling effects, or redefining outputs to remove the elevator's influence on the flight–path angle dynamics. Since the output–redefinition method of Fiorentini et al. [2009], the most effective solutions, whether by output redefinition, internal–model augmentation, or generalized virtual–output design, have all relied on time–scale separation. In such designs, a fast inner loop stabilizes the short–period or virtual–output dynamics, while slower outer loops regulate velocity, flight–path angle, and altitude. This arrangement allows each loop to treat the faster dynamics as quasi–steady, thereby avoiding the destabilizing effects of unstable internal modes.

As a direct consequence of these findings, the control architecture in this work adopts the principle demonstrated by Ye et al. [2018] and other recent designs: a cascaded system with explicit time–scale separation between loops. Ye et al. implemented a two–loop structure combining velocity and flight–path angle control with attitude stabilization. In contrast, the present design extends this concept to a four–loop hierarchy, with nested loops for angular rate, attitude, velocity, and position control. Each loop has full relative degree and is differentiated only once, eliminating internal dynamics while keeping the command laws straightforward. Time–scale separation ensures that faster subsystems appear quasi–steady to slower ones, enabling stable inversion at each stage and simplifying tuning.

In this framework, the main control approach is INDI. To the author's knowledge, INDI has not previously been applied to an airbreathing hypersonic vehicle control. It is well suited to this problem because hypersonic flight is characterized by large variations in aerodynamic coefficients, strong state coupling, and significant uncertainty in aerodynamic models. NDI relies heavily on accurate modeling, which can limit performance under such conditions. INDI, on the other hand, updates the control effectiveness in real time using measured accelerations and angular rates, making it more robust to modeling errors and rapid aerodynamic changes. In the proposed four–loop architecture, INDI is used in the inner angular–rate loop, where fast and reliable response is essential for stabilizing the outer loops. For relationships that are purely kinematics, NDI is applied to achieve exact inversion without relying on aerodynamic models.

The cascaded FCS is organized into four nested loops, each operating at a distinct dynamic level. The innermost rate loop stabilizes the angular rates, while the successive outer loops regulate attitude, velocity, and finally position. This hierarchical structure, illustrated in Fig. 5.1, ensures that the fast inner–loop dynamics provide stability for the slower outer–loop objectives, thereby achieving a natural separation of time scales and facilitating controller design. The subscript K is used throughout to denote quantities derived from Earth–relative velocities rather than air–relative ones. This distinction is crucial because the onboard INS, as described in Zipfel [2025], provides only the Earth–relative velocity vector $[\boldsymbol{v}_B^E]^D$ and without direct wind measurements, it cannot compute the air–relative velocity $[\boldsymbol{v}_B^A]^D$. Consequently, all kinematic quantities used in the control system are expressed relative to Earth motion.

Within this framework, each control loop acts on a specific subset of system states. The indices 1 through 4 are used solely to distinguish the different control loops and do not refer to points. The corresponding state vectors are defined as:

$$\boldsymbol{x}_1 = \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \quad \boldsymbol{x}_2 = \begin{bmatrix} \mu_K \\ \alpha_K \\ \beta_K \end{bmatrix}, \quad \boldsymbol{x}_3 = \begin{bmatrix} V_K \\ \gamma_K \\ \chi_K \end{bmatrix}, \quad \boldsymbol{x}_4 = \begin{bmatrix} l \\ \lambda \\ h \end{bmatrix}. \tag{5.1}$$

Here, p, q, r are the measured inertial body rates. The quantities μ_K, α_K, β_K denote the kinematic bank angle, angle of attack, and sideslip angle, while V_K, γ_K, χ_K represent the geographic speed, flight–path, and heading angle. Finally, l, λ, h denote the geocentric longitude, latitude, and altitude. These state groups correspond directly to the four nested control loops:

1. **Angular rate control loop**
   Drives the angular rate state $\boldsymbol{x}_1$ to match the command $\boldsymbol{x}_1^{\text{cmd}}$. It produces actuator commands $\boldsymbol{u}_1^{\text{cmd}}$ that are sent to the actuators. Uncertainty in the control effectiveness and aerodynamic moments is handled using INDI.

2. **Attitude control loop**
   Regulates the attitude state $\boldsymbol{x}_2$ to track the commands from the velocity loop. The resulting body rate command vector is $\boldsymbol{x}_1^{\text{cmd}}$. Since this mapping is kinematic, it is implemented using NDI.

3. **Velocity control loop**
   Steers the velocity state $\boldsymbol{x}_3$ to follow $\boldsymbol{x}_3^{\text{cmd}}$. It determines incremental attitude angles and throttle input for the attitude loop. Model uncertainty arises here from the aerodynamic and propulsive forces and is compensated using INDI.

4. **Position control loop**
   Tracks the commanded position vector $\boldsymbol{x}_4^{\text{cmd}}$. It provides the velocity command $\boldsymbol{x}_3^{\text{cmd}}$ for the velocity loop. Since this mapping is purely kinematic, it is implemented using NDI.

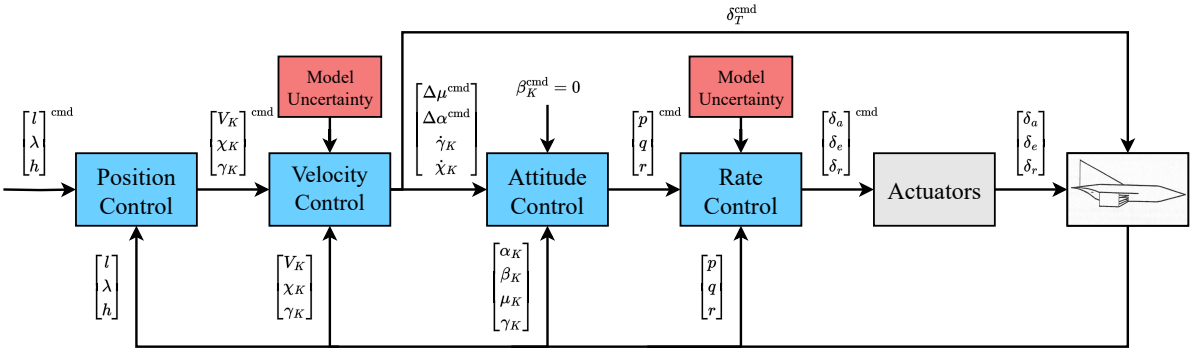


Figure 5.1: Cascaded flight control architecture used in this work.

Model Assumptions for Control Law Development

The complete flight dynamics model presented in Chapter 4 captures all major physical effects relevant to the vehicle's motion. Such fidelity is essential for accurate simulation and reliable performance evaluation. However, formulating a control law directly from this high-fidelity model would introduce unnecessary complexity, complicating analytical derivations, controller tuning, and practical implementation.

A simplified dynamic model is therefore used for control law development. Only effects whose omission has a negligible impact on the vehicle's response are removed, resulting in a reduced-order model that preserves the dominant control-relevant dynamics. This simplification makes the analytical derivation of the control law feasible while maintaining sufficient accuracy within the intended flight envelope. The final control law is then validated using the complete high-fidelity model to confirm its performance under all relevant physical effects and interactions.

1. Non-rotating Earth

   The rotational velocity of the Earth introduces transport and Coriolis accelerations in the equations of motion. For sub-orbital flight in the sensible atmosphere, the transport acceleration is very small compared to aerodynamic and gravitational forces. Its maximum value, occurring at the equator, is about 0.034 m/s^2 ($3.5\times10^{-3}\,g_0$), and it decreases with latitude to zero at the poles. The Coriolis acceleration magnitude is bounded by $2\omega_\oplus V$; at $V = 300$ m/s it is 0.044 m/s^2 $\approx 4.5\times10^{-3}\,g_0$, at $V = 1500$ m/s it is 0.22 m/s^2 $\approx 2.2\times10^{-2}\,g_0$, and it only becomes significant ($\approx 0.116\,g_0$) at orbital velocities. For the control loops designed here, these values are smaller than typical modeling and actuation uncertainties, so omitting them simplifies the derivation without altering closed-loop behavior. All rotational effects are restored for the full nonlinear model validation.

2. Spherical Earth Model

   The high-fidelity model represents the Earth as an ellipsoid, but for the reduced-order control model it is assumed spherical, as the difference between an ellipsoidal and spherical representation is negligible for controller design. This simplification removes the distinction between geodetic and geocentric latitude and makes the Earth's radius constant, rather than dependent on latitude. Quantitatively, the difference between the equatorial and polar radii in WGS-84 is about 21.4 km out of 6378 km, i.e. less than 0.34%. The maximum discrepancy between geodetic and geocentric latitude is approximately 0.19°, which has little influence on the closed-loop dynamics considered here.

   A further simplification to a flat Earth, however, would remove curvature-related terms from the equations of motion. In particular, omitting the term $V^2/(gr)$ eliminates the centrifugal force component that contributes to the lift vector. As orbital velocity is approached, the lift required to

balance this term reduces to zero, and consequently the induced drag is also reduced to zero. This is a significant physical effect that directly influences aerodynamic force balance and performance predictions. For this reason, the flat-Earth assumption is not adopted, and the reduced-order control model retains the curvature terms in the equations of motion.

3. Gravity field proportional to the inverse of the radius squared

   In the reduced-order control model, the latitude-dependent variation in gravitational acceleration is neglected, but the change in gravity with altitude is retained. For flight within the atmosphere, the variation with latitude is small compared to other modeling uncertainties and has little influence on the short time scales relevant for control. The altitude-dependent variation, on the other hand, is very straightforward to include and becomes more relevant at higher altitudes, so it is kept in the model.

4. Stationary Atmosphere Model

   The reduced-order control model assumes steady atmospheric conditions and ignores changes in wind over time or location. At hypersonic speeds, the contribution of wind to the total airspeed is negligible compared to the vehicle's forward velocity, and its rotational influence on the body dynamics is effectively zero [Autenrieb, 2023]. At subsonic speeds, Pfeifle and Fichter [2021] have shown that INDI exhibits excellent disturbance rejection properties against constant wind and wind gusts, further reducing the need to model wind explicitly during control synthesis. Moreover, the onboard INS has no direct access to wind information, making it unrealistic to incorporate wind into the controller as a measured quantity. It is therefore more appropriate to treat wind as a disturbance that must be rejected rather than as a modeled input. This assumption simplifies the controller and allows the design process to focus on the core vehicle dynamics. Wind and turbulence effects are, however, retained in the high-fidelity model to quantify performance in realistic atmospheric conditions.

5. Coordinated-Flight and Constant-Speed Assumption

   The attitude control loop regulates the sideslip angle β_K to zero, so both the velocity and position control loops are formulated under the assumption $\beta_K = 0$. Similarly, the velocity control loop ensures that the commanded speed is reached and maintained, allowing the position control loop to be formulated under the additional assumption $V_K = V_K^{\text{cmd}}$. Together, these assumptions reduce the number of states that must be considered in the outer loops and provide the additional benefit of filtering out small-scale fluctuations in the computed velocity and sideslip angle, since both quantities are treated as constant.

6. Small-Displacement Assumption

   The change in longitude l and latitude λ is negligible over the short time scales relevant to the velocity dynamics. As a result, $\dot{l}$ and $\dot{\lambda}$ are omitted from the attitude control loop design [Juliana et al., 2004]. Numerical evaluation confirms that their effect on the velocity response is insignificant, with differences in the computed states below 10^{-6}.

7. Accurate Control Effectiveness and Actuator Dynamics

   Throughout the control-law derivation, it is assumed that the estimated control effectiveness matrix matches the true system behavior and that the modeled actuator dynamics accurately represent the physical actuator response. These assumptions ensure that the nonlinear inversion remains valid and that the incremental control updates behave as predicted by the reduced-order model.

5.2. Control Law Design

This section presents the design of the dynamic inversion control laws for each loop in the cascaded architecture. It begins with the angular–rate control loop, where both NDI and INDI formulations are developed, including the method for reconstructing previous control inputs and the selection criteria for the low–pass filter used in the INDI implementation. The next part addresses the attitude–control loop, describing the derivation of the nonlinear inversion law used to track commanded attitudes. The velocity–control loop is then introduced, outlining its inversion law and its role in regulating airspeed and flight–path dynamics. Finally, the position–control loop is discussed, including its two functional modes for altitude and heading tracking.

Angular Rate Control Loop

The angular rate loop forms the innermost part of the FCS. It is responsible for tracking angular velocity commands generated by the attitude controller. The system is described using the following state, output, and input vectors:

$$\begin{aligned} \boldsymbol{x}_1 &= \begin{bmatrix} p & q & r \end{bmatrix}^\top = \boldsymbol{y}_1, \\ \boldsymbol{u}_1 &= \begin{bmatrix} \delta_a & \delta_e & \delta_r \end{bmatrix}^\top, \end{aligned} \tag{5.2}$$

where p, q, r are the inertial body rates, and the control inputs $\delta_a, \delta_e, \delta_r$ are the aileron, elevator, and rudder deflections, respectively. The output function $\boldsymbol{h}_1(\boldsymbol{x}_1)$ implies that state and output are the same. The variable-mass attitude equations of motion for a symmetric aircraft are given in Eq. (4.26) but are repeated here for reference:

$$\left(D^B \boldsymbol{I}_B^B\right) \boldsymbol{x}_1 + \boldsymbol{I}_B^B \left(D^B \boldsymbol{x}_1\right) + \boldsymbol{\Omega}^{BI} \boldsymbol{I}_B^B \boldsymbol{x}_1 = \boldsymbol{m}_B. \tag{5.3}$$

The external moments in $\boldsymbol{m}_B$ are considered as the sum of moments generated by the airframe aerodynamics $\boldsymbol{m}_a$ and moments generated by control surface deflections $\boldsymbol{m}_c$. Assuming the control derivatives to be linear as $(\boldsymbol{m}_c)_\delta = \frac{\partial}{\partial \delta} \boldsymbol{m}_c$, then $\boldsymbol{m}_B$ is linear in the deflection angles:

$$\boldsymbol{m}_B = \boldsymbol{m}_a + (\boldsymbol{m}_c)_\delta \, \boldsymbol{u}_1. \tag{5.4}$$

Substituting Eq. (5.4) into the rotational equation of motion yields:

$$\left(D^B \boldsymbol{I}_B^B\right) \boldsymbol{x}_1 + \boldsymbol{I}_B^B \left(D^B \boldsymbol{x}_1\right) + \boldsymbol{\Omega}^{BI} \boldsymbol{I}_B^B \boldsymbol{x}_1 = \boldsymbol{m}_a + (\boldsymbol{m}_c)_\delta \, \boldsymbol{u}_1. \tag{5.5}$$

Solving for the body angular velocity derivative yields the control-affine form:

$$\left(D^B \boldsymbol{x}_1\right) = \left(\boldsymbol{I}_B^B\right)^{-1} \left[-\left(D^B \boldsymbol{I}_B^B\right) \boldsymbol{x}_1 - \boldsymbol{\Omega}^{BI} \boldsymbol{I}_B^B \boldsymbol{x}_1 + \boldsymbol{m}_a\right] + \left(\boldsymbol{I}_B^B\right)^{-1} (\boldsymbol{m}_c)_\delta \, \boldsymbol{u}_1. \tag{5.6}$$

By introducing the virtual control input $\boldsymbol{\nu}_1 = D^B \boldsymbol{x}_1^{\text{cmd}}$, the rate–loop control law becomes:

$$\boldsymbol{u}_1 = (\boldsymbol{m}_c)_\delta^{-1} \left[\boldsymbol{I}_B^B \boldsymbol{\nu}_1 + \left(\left(D^B \boldsymbol{I}_B^B\right) \boldsymbol{x}_1 + \boldsymbol{\Omega}^{BI} \boldsymbol{I}_B^B \boldsymbol{x}_1 - \boldsymbol{m}_a\right)\right]. \tag{5.7}$$

For clarity, the same control law can be expressed in NDI form as:

$$\boldsymbol{u}_1 = \boldsymbol{G}_1^{-1}(\boldsymbol{x}_1) \left(\boldsymbol{\nu}_1 - \boldsymbol{f}_1(\boldsymbol{x}_1)\right). \tag{5.8}$$

The control effectiveness matrix and plant dynamics are consequently:

$$\begin{aligned} \boldsymbol{G}_1 \boldsymbol{x}_1) &= \left(\boldsymbol{I}_B^B\right)^{-1} (\boldsymbol{m}_c)_\delta, \\ \boldsymbol{f}_1(\boldsymbol{x}_1) &= \left(\boldsymbol{I}_B^B\right)^{-1} \left\{-\left(D^B \boldsymbol{I}_B^B + \boldsymbol{\Omega}^{BI} \boldsymbol{I}_B^B\right) \boldsymbol{x}_1 + \boldsymbol{m}_a\right\}. \end{aligned} \tag{5.9}$$

All quantities are expressed in the body-axis coordinate system:

$$[I_B^B]^B = \begin{bmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{zx} & 0 & I_{zz} \end{bmatrix}, \quad [(m_c)_\delta]^B = \bar{q}\, S \begin{bmatrix} b\, C_{l_{\delta_a}} & 0 & b\, C_{l_{\delta_r}} \\ 0 & \bar{c}\, C_{m_{\delta_e}} & 0 \\ b\, C_{n_{\delta_a}} & 0 & C_{n_{\delta_r}} \end{bmatrix}, \quad [m_a]^B = \bar{q}\, S \begin{bmatrix} b\, C_{l_a} \\ \bar{c}\, C_{m_a} \\ b\, C_{n_a} \end{bmatrix}.$$

This NDI control law depends on accurate knowledge of both aerodynamic and inertial parameters. It is assumed that the control effectiveness matrix $\boldsymbol{G}_1(\boldsymbol{x}_1)$ is invertible over the entire domain of operation, and that the number of control inputs equals the number of controlled outputs, i.e., $\dim(\boldsymbol{u}_1) = \dim(\boldsymbol{y}_1)$. This ensures that the inverse $\boldsymbol{G}_1^{-1}(\boldsymbol{x}_1)$ is well-defined and that the system is fully actuated. In situations where $\dim(\boldsymbol{u}_1) > \dim(\boldsymbol{y}_1)$, the system is overactuated and control allocation methods must be used to distribute the virtual control command $\boldsymbol{\nu}_1$ among redundant actuators. Conversely, when $\dim(\boldsymbol{u}_1) < \dim(\boldsymbol{y}_1)$, the system is underactuated and internal dynamics may emerge that must be analyzed for stability.

Furthermore, the effectiveness of NDI is contingent on accurate modeling of the aerodynamic moment coefficients and the mass/inertia properties, which enter explicitly in both $\boldsymbol{f}_1(\boldsymbol{x}_1)$ and $\boldsymbol{G}_1(\boldsymbol{x}_1)$. As such, NDI offers precise inner-loop linearization and decoupling, but its robustness is inherently limited by the fidelity of the plant model. To reduce this sensitivity in the presence of modeling errors, an alternative formulation is adopted that preserves the structure of the control effectiveness matrix $\boldsymbol{G}_1(\boldsymbol{x}_1)$ but avoids explicit dependence on the plant dynamics $\boldsymbol{f}_1(\boldsymbol{x}_1)$. This leads to the INDI version.

Incremental Nonlinear Dynamic Inversion (INDI)

To reduce the dependency of the control law on the full nonlinear model, the angular–rate loop employs an incremental NDI formulation. The key idea is to replace part of the model information by locally measured state–rate data, thereby improving robustness to modeling uncertainty. Starting from the angular–rate dynamics in Eq. (5.6):

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{G}_1(\boldsymbol{x}_1)\boldsymbol{u}_1. \tag{5.10}$$

A standard Taylor series expansion is taken about the values at the previous sampling instant, where $\boldsymbol{x}_{1,0} = \boldsymbol{x}_1(t - \Delta t)$ and $\boldsymbol{u}_{1,0} = \boldsymbol{u}_1(t - \Delta t)$. This yields:

$$\begin{aligned} \dot{\boldsymbol{x}}_1 &= \dot{\boldsymbol{x}}_{1,0} + \left.\frac{\partial\left[\boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{G}_1(\boldsymbol{x}_1)\boldsymbol{u}_1\right]}{\partial \boldsymbol{x}_1}\right|_0 \Delta\boldsymbol{x}_1 + \left.\frac{\partial\left[\boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{G}_1(\boldsymbol{x}_1)\boldsymbol{u}_1\right]}{\partial \boldsymbol{u}_1}\right|_0 \Delta\boldsymbol{u}_1 + O(\Delta\boldsymbol{x}_1^2), \\ &= \dot{\boldsymbol{x}}_{1,0} + \left.\frac{\partial\left[\boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{G}_1(\boldsymbol{x}_1)\boldsymbol{u}_1\right]}{\partial \boldsymbol{x}_1}\right|_0 \Delta\boldsymbol{x}_1 + \boldsymbol{G}_1(\boldsymbol{x}_{1,0})\,\Delta\boldsymbol{u}_1 + O(\Delta\boldsymbol{x}_1^2). \end{aligned} \tag{5.11}$$

Here $\Delta\boldsymbol{x}_1 = \boldsymbol{x}_1 - \boldsymbol{x}_{1,0}$, $\Delta\boldsymbol{u}_1 = \boldsymbol{u}_1 - \boldsymbol{u}_{1,0}$, and the term $O(\Delta\boldsymbol{x}_1^2)$ collects all higher–order contributions.

As in the standard INDI argument, a time–scale separation assumption is introduced: over one sampling step the state increment is much smaller than the control increment, $\Delta\boldsymbol{x}_1 \ll \Delta\boldsymbol{u}_1$. Actuator dynamics typically evolve on a much faster time scale than the rotational states, so the term involving $\Delta\boldsymbol{x}_1$ and the higher–order contributions can be neglected [Bacon and Ostroff, 2000]. This reduces Eq. (5.49) to the incremental approximation:

$$\dot{\boldsymbol{x}}_1 \approx \dot{\boldsymbol{x}}_{1,0} + \boldsymbol{G}_1(\boldsymbol{x}_{1,0})\,\Delta\boldsymbol{u}_1. \tag{5.12}$$

The matrix $\boldsymbol{G}_{1,0}$ is treated as constant over this interval and may be estimated offline or updated online. To enforce desired angular acceleration dynamics, a virtual control input $\boldsymbol{\nu}_1$ is introduced. Substituting this into the approximation Eq. (5.12) and solving for $\boldsymbol{u}_1$ yields the INDI control law:

$$\boldsymbol{u}_1 = \boldsymbol{u}_{1,0} + \boldsymbol{G}_{1,0}^{-1}\left(\boldsymbol{\nu}_1 - \dot{\boldsymbol{x}}_{1,0}\right). \tag{5.13}$$

This control law preserves the input-affine structure of feedback linearization while avoiding explicit dependence on the full nonlinear model. Provided that measurements of $\dot{\boldsymbol{x}}_1(t-\Delta t)$ and $\boldsymbol{u}_1(t-\Delta t)$ are accurate and the delay Δt is sufficiently small, the INDI formulation achieves robust inner-loop angular rate tracking. In the specific case of angular rate control using the variables shown in Fig. 5.2, the generic INDI law becomes:

$$\boldsymbol{\delta}^{\text{cmd}} = \boldsymbol{\delta}_0 + \hat{\boldsymbol{G}}_{1,0}^{-1}\left(\boldsymbol{\nu}_1 - \dot{\boldsymbol{\omega}}_0\right), \tag{5.14}$$

where $\boldsymbol{\delta}_0 = \boldsymbol{\delta}(t-\delta t)$ is the previously applied control input, $\dot{\boldsymbol{\omega}}_0 = \dot{\boldsymbol{\omega}}(t-\delta t)$ is the previous angular acceleration measurement, and $\boldsymbol{\nu}_1$ is the virtual control input derived from the desired angular rates.

State estimation, filtering and synchronisation

This subsection explains how the delayed control input δ_0 and the delayed state derivative $\dot{\omega}_0$ are obtained for the SISO case. Since neither quantity is directly measurable, both must be estimated. The delayed control input is generated by passing the commanded signal $\delta^{\text{cmd}}(t)$ through a second–order actuator model with a fixed delay Δt. Using the actuator dynamics defined in Eq. (4.66), the resulting delayed actuator deflection is:

$$\delta_0 = \delta(t-\Delta t) = \mathcal{L}^{-1}\left\{e^{-s\Delta t} A(s)\, \delta^{\text{cmd}}(s)\right\}. \tag{5.15}$$

The presence of the fixed delay Δt arises from the local linearization of the system dynamics about a previously measured operating point. The control law is defined relative to this delayed point, not the current state, which is why both δ_0 and $\dot{\omega}_0$ appear with the same delay. In the discrete time implementation the delay Δt corresponds to the sample time of the FCC and is implemented using a single unit delay. The delayed angular acceleration $\dot{\omega}_0 = \dot{\omega}(t-\Delta t)$ is obtained by differentiating the measured angular rate signal $\omega(t)$ in the frequency domain, followed by the same fixed delay:

$$\dot{\omega}_0 = \dot{\omega}(t-\Delta t) = \mathcal{L}^{-1}\left\{s e^{-s\Delta t}\, \omega(s)\right\}. \tag{5.16}$$

In practice, angular acceleration is obtained by numerically differentiating the measured angular rate at the sampling rate of the FCC. Since angular rate sensors inevitably introduce measurement noise, this differentiation process amplifies high-frequency components that will be passed to the actuators, wearing them down and potentially degrading control performance [Grondman et al., 2018]. To reduce this effect, the measured angular rate is passed through a second-order low-pass filter:

$$H(s) = \frac{\omega_H^2}{s^2 + 2\zeta_H \omega_H s + \omega_H^2}. \tag{5.17}$$

A lower cut-off frequency attenuates noise more effectively but also degrades the disturbance–rejection capability of the rate–control loop. The cut-off frequency is therefore chosen as a compromise between noise suppression and maintaining good disturbance rejection. Applying the filter yields the filtered angular–acceleration estimate:

$$\dot{\omega}_{f_0} = \dot{\omega}_f(t-\Delta t) = \mathcal{L}^{-1}\left\{sH(s) e^{-s\Delta t}\, \omega(s)\right\}. \tag{5.18}$$

The low-pass filter introduces a phase delay in the angular rate estimate, causing it to become desynchronized from the unfiltered actuator signal δ_0. Since the Taylor series approximation in the INDI control law is only valid when the angular acceleration and actuator signal are delayed by the same amount of time, temporal alignment between the two must be preserved [Smeur et al., 2016a]. To achieve this, the same low-pass filter $H(s)$ is applied to the actuator path:

$$\delta_{f_0} = \delta_f(t-\Delta t) = \mathcal{L}^{-1}\left\{H(s) e^{-s\Delta t} A(s) \delta^{\text{cmd}}(s)\right\}. \tag{5.19}$$

The final filtered control law then becomes:

$$\delta^{\text{cmd}}(t) = \delta_{f_0} + \hat{\boldsymbol{G}}_{1,0}^{-1}\left(\nu_1(t) - \dot{\omega}_{f_0}\right). \tag{5.20}$$

The full structure of this implementation, including the filtering and reconstruction paths, is illustrated in Fig. 5.2. The diagram highlights the role of the filter $H(s)$, actuator model $A(s)$, and fixed delay Δt in generating the desired actuator deflection under the INDI framework.

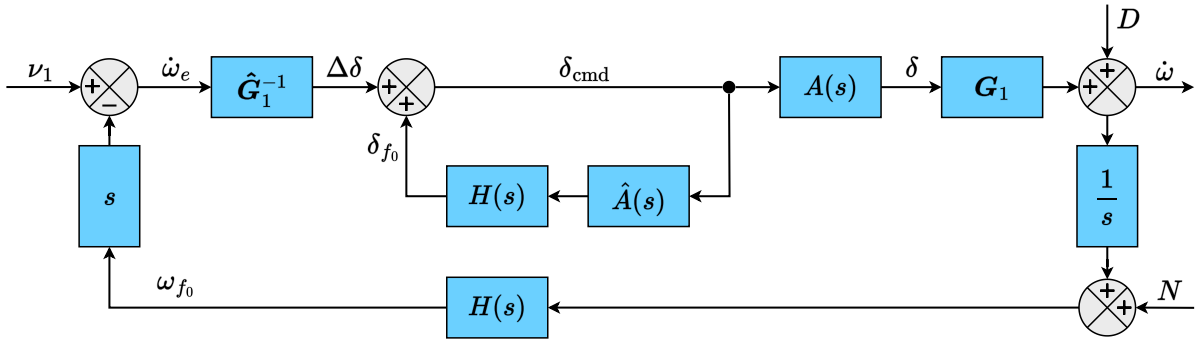


Figure 5.2: Frequency-domain implementation of the INDI control loop for angular rate tracking. The loop includes angular acceleration estimation, actuator modeling, and consistent filtering of both the feedback and control branches.

The following transfer functions characterize the frequency-domain behavior of the INDI inner loop. Each one describes how the angular acceleration $\dot{\boldsymbol{\omega}}(s)$ responds to a different input, whether it be a commanded signal, disturbance, or sensor noise, under the influence of actuator dynamics, sensor filtering, and delay. This analysis relies on the previously stated assumption that the control effectiveness matrix and actuator dynamics are accurately represented in the reduced-order model, i.e., $\hat{\boldsymbol{G}}_1 = \boldsymbol{G}_1$ and $\hat{A}(s) = A(s)$.

1. **Open-Loop Transfer Function:** The expression below represents the open-loop transfer function from the angular rate error $\dot{\boldsymbol{\omega}}_e(s) = \boldsymbol{\nu}_1(s) - \dot{\boldsymbol{\omega}}_f(s)$ to the resulting angular acceleration $\dot{\boldsymbol{\omega}}(s)$. This expression captures how the control law propagates the effect of an angular acceleration command through the actuator and filter dynamics. The expression may also be written in fraction form as:

$$\begin{aligned}\frac{\dot{\boldsymbol{\omega}}(s)}{\dot{\boldsymbol{\omega}}_e(s)} = \boldsymbol{G}_{\text{ol}}(s) &= \boldsymbol{G}_1 \left(\boldsymbol{I}_3(1 - A(s)H(s))^{-1} A(s)\right)\boldsymbol{G}_1^{-1} \\ &= \boldsymbol{I}_3 \left(1 - A(s)H(s)\right)^{-1} A(s).\end{aligned} \tag{5.21}$$

2. **Closed-Loop Transfer Function:** The following expression defines the closed-loop transfer function from the commanded angular acceleration $\boldsymbol{\nu}_1(s)$ to the actual angular acceleration $\dot{\boldsymbol{\omega}}(s)$, assuming zero sensor noise and no external disturbances. The expression accounts for the feedback path through the filtered measurement channel $H(s)e^{-s\Delta t}$.

$$\begin{aligned}\frac{\dot{\boldsymbol{\omega}}(s)}{\boldsymbol{\nu}_1(s)} = \boldsymbol{G}_{\text{cl}}(s) &= \left(\boldsymbol{I}_3 + \boldsymbol{G}_{\text{ol}}(s)H(s)\right)^{-1} \boldsymbol{G}_{\text{ol}}(s) \\ &= \boldsymbol{I}_3 \cdot \frac{\left(1 - A(s)H(s)\right)^{-1} A(s)}{1 + \left(1 - A(s)H(s)\right)^{-1} A(s)H(s)} \\ &= \boldsymbol{I}_3 A(s).\end{aligned} \tag{5.22}$$

   The closed-loop transfer function shows that, without external input, the aircraft dynamics and the delay-compensated feedback cancel out, leaving the actuator dynamics $A(s)$ as the exact input–output behavior from the virtual control input to the angular acceleration. This means that, from the perspective of the outer loops, the angular acceleration loop can be modeled by the actuator dynamics [Smeur et al., 2016a]. In practice, this implies that an ideal integrator behavior

is not achieved, as the closed-loop response is shaped by the finite actuator bandwidth. This is not necessarily detrimental. While the limited bandwidth constrains the achievable response speed, it can also increase the delay margin of the closed-loop system and thereby improve robustness to time delays [Raab, 2025].

3. **Disturbance Transfer Function:** Unlike NDI, INDI does not achieve exact linearization of the system dynamics. Instead, it provides an approximate linearization around a previous operating point. As a result, the nonlinearities in the angular acceleration dynamics are not fully canceled and persist as residual terms. These residual angular acceleration nonlinearities, which include unmodeled dynamics and higher-order effects, are modeled as an additive disturbance $\boldsymbol{D}(s)$ acting on the angular acceleration. This disturbance is present even under ideal measurement conditions, because the linearization achieved by INDI is inherently local and only approximate. Following the formulation first derived by Smeur et al. [2016a], the resulting transfer function from the disturbance $\boldsymbol{D}(s)$ to the angular acceleration $\dot{\boldsymbol{\omega}}(s)$ quantifies the system's disturbance rejection capability:

$$\begin{aligned}\frac{\dot{\boldsymbol{\omega}}(s)}{\boldsymbol{D}(s)} = \boldsymbol{G}_d(s) &= \left(\boldsymbol{I}_3 + \boldsymbol{G}_{\text{ol}}(s)H(s)\right)^{-1}\boldsymbol{I}_3 \\ &= \boldsymbol{I}_3 \cdot \frac{1}{1 + \left(1 - A(s)H(s)\right)^{-1} A(s)H(s)} \\ &= \boldsymbol{I}_3 \left(1 - A(s)H(s)\right).\end{aligned} \tag{5.23}$$

This expression shows that the disturbance rejection capability is determined by the product $A(s)H(s)$. The faster the actuator dynamics $A(s)$ and the filter $H(s)$, the more closely this product approaches unity across the frequency range of interest. As a result, the disturbance transfer function $\boldsymbol{G}_d(s)$ becomes smaller, leading to faster and more effective disturbance rejection by the INDI loop.

4. **Noise Transfer Function:** This transfer function quantifies the sensitivity of the system to sensor noise $\boldsymbol{N}(s)$, which enters through the feedback path. Due to the numerical differentiation of measured angular rates, this path can amplify high-frequency components:

$$\begin{aligned}\frac{\dot{\boldsymbol{\omega}}(s)}{\boldsymbol{N}(s)} = \boldsymbol{G}_n(s) &= -\boldsymbol{G}_{\text{ol}}(s)\, sH(s)\, \left(\boldsymbol{I}_3 + \boldsymbol{G}_{\text{ol}}(s)H(s)\right)^{-1} \\ &= -\boldsymbol{I}_3 \cdot sH(s) \cdot \frac{\left(1 - A(s)H(s)\right)^{-1} A(s)}{1 + \left(1 - A(s)H(s)\right)^{-1} A(s)H(s)} \\ &= -\boldsymbol{I}_3 \cdot sH(s) \cdot A(s).\end{aligned} \tag{5.24}$$

This transfer function shows that sensor noise is shaped by the term $sH(s)$, which reflects the numerical differentiation of the measured angular rates. Without filtering, high-frequency noise would be strongly amplified.

Together, these transfer functions provide a complete frequency-domain characterization of the INDI inner-loop behavior, including performance, disturbance rejection, and robustness to sensor noise and actuator lag. They also clarify the effect of filtering and delay on the dynamic response of the angular rate control loop. The total system output $\dot{\boldsymbol{\omega}}(s)$ is the result of superposition of the desired signal, the disturbance, and sensor noise:

$$\dot{\boldsymbol{\omega}}(s) = \boldsymbol{G}_{\text{cl}}(s)\, \boldsymbol{\nu}_1(s) + \boldsymbol{G}_d(s)\, \boldsymbol{D}(s) + \boldsymbol{G}_n(s)\, \boldsymbol{N}(s). \tag{5.25}$$

Both the low-pass filter $H(s)$ and the actuator model $A(s)$ shape how the inner control loop responds to disturbances and sensor noise. In most implementations, $A(s)$ is determined by physical actuator

dynamics, leaving $H(s)$ as the only tuning mechanism. For disturbance rejection, the closed-loop sensitivity function includes the factor:

$$S(s) \propto (1 - A(s)H(s)) .$$

Increasing the bandwidth of $H(s)$ (and where possible, $A(s)$) raises the magnitude of $A(s)H(s)$ over a wider frequency range, reducing $S(s)$, and thus improving disturbance rejection. This leads to a faster and smaller-amplitude transient in response to external disturbances. In contrast, the response to sensor noise follows a different transfer function,

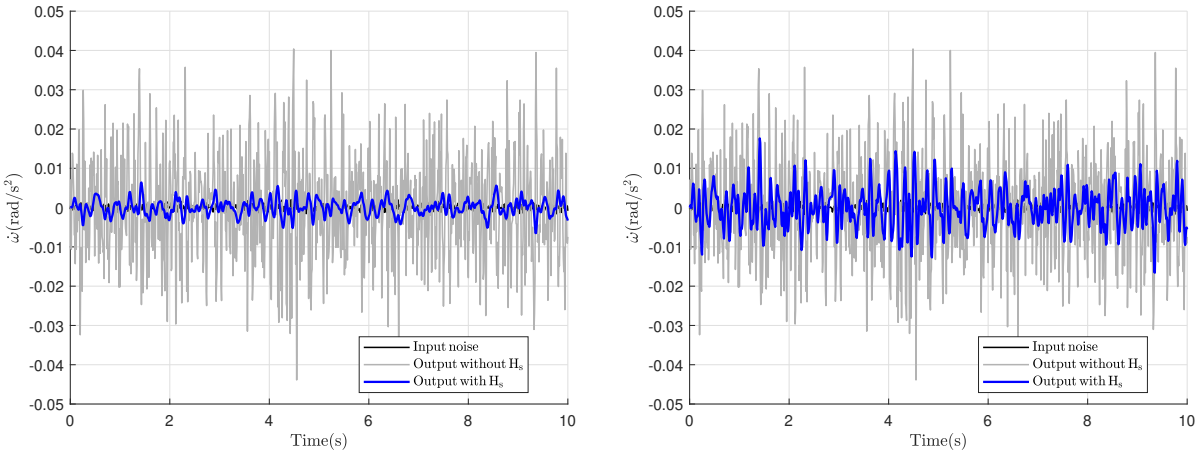
$$N(s) \propto s\, H(s)\, A(s),$$

which increases with frequency. As a result, increasing the bandwidth of $H(s)$ allows more high-frequency measurement noise to propagate to the output. The filter bandwidth must therefore balance disturbance rejection against noise amplification. This trade-off is evident in flight-tested INDI controllers: reported corner frequencies range from 5 rad/s for a tailsitter Unmanned Aerial Vehicle (UAV), to 20 rad/s for the Cessna Citation, and up to 45 rad/s for a fixed-wing UAV, all with unity damping [Grondman et al., 2018, Pfeifle and Fichter, 2021, Smeur et al., 2020].

To evaluate the noise response, the measurement noise is modeled as a zero-mean Gaussian process with variance $\sigma^2 = 4 \times 10^{-7}$ rad^2/s^2, corresponding to the INS measurement noise level specified by Zipfel [2025]. Each sample is drawn as $n_k \sim \mathcal{N}(0, \sigma^2)$, yielding a white sequence band-limited by the sampling interval Δt with Nyquist frequency:

$$\omega_{\text{Nyq}} = \frac{\pi}{\Delta t}. \tag{5.26}$$

Fig. 5.3 shows identical noise inputs passed through $G_n(s)$ with corner frequencies of 20 rad/s and 45 rad/s. The lower bandwidth suppresses high-frequency content more effectively, producing significantly less amplification in the angular-acceleration output. Increasing the bandwidth to 45 rad/s allows a larger portion of the noise spectrum to pass, which yields a visibly higher output variance.



(a) ω_H = 20 rad/s and ζ_H = 1

(b) ω_H = 45 rad/s and ζ_H = 1

Figure 5.3: Simulated noise-to-$\dot{\omega}$ responses for two shaping filter bandwidths. The large-amplitude gray trace shows the raw amplified noise, the blue curve shows the response with $H(s)$, and the low-amplitude black trace is the input noise.

The effect on disturbance rejection is shown in Fig. 5.4. The step responses show that the 20 rad/s filter rejects the disturbance in roughly twice the time required by the 45 rad/s filter. Both filters reduce the residual disturbance to essentially zero in steady state, so the difference is limited to the transient response. In the simulations, the 45 rad/s filter shows visible noise during the transient, whereas this effect is much less noticeable for the 20 rad/s filter.

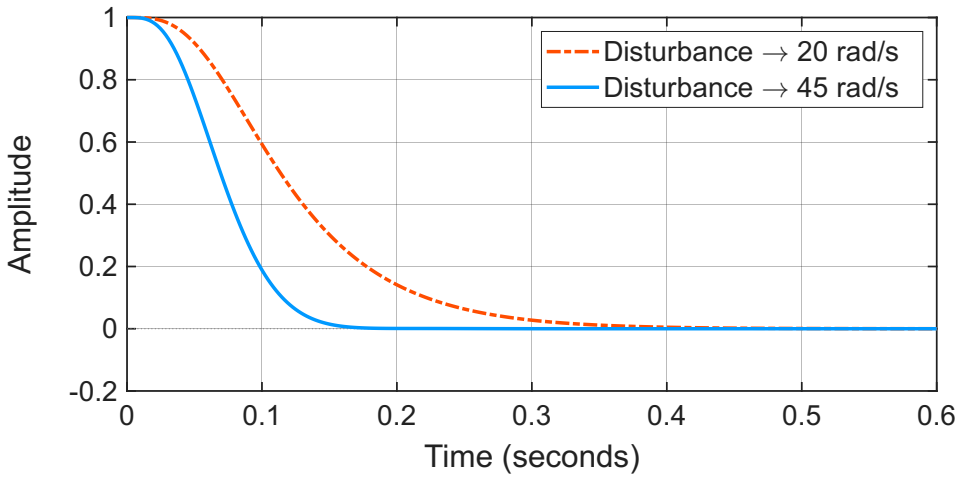


Figure 5.4: Simulated disturbance rejection for two shaping filter bandwidths. A higher corner frequency yields a faster transient, while both cases achieve near-zero steady-state error.

To assess the influence of the low-pass filter $H(s)$ on the closed-loop stability of the SISO continuous INDI system in Figure 5.2, the feedback loop is opened at the location indicated in Figure 5.5, namely in the sensor feedback path. This is necessary because, under ideal inversion, the closed-loop input–output dynamics reduce to the actuator dynamics, and the effect of the low-pass filter does not explicitly appear.

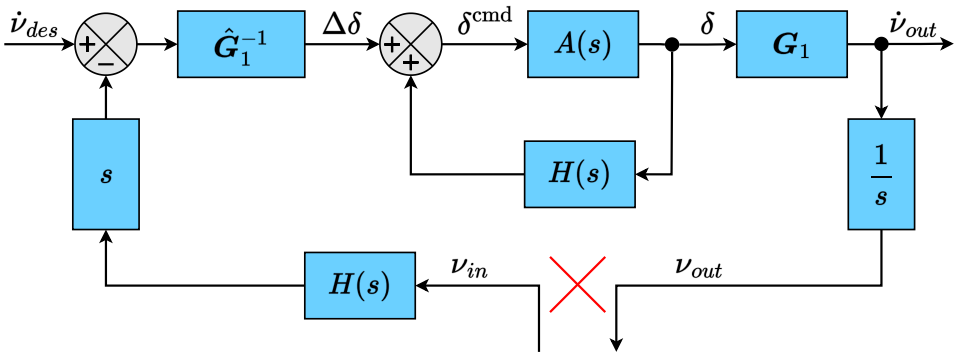


Figure 5.5: Broken-loop configuration of the inner INDI control loop used for stability margin analysis.

Opening the loop at this location exposes the internal feedback structure in which the filter introduces phase lag. This path is particularly critical for delay margin analysis, as the sensor feedback path is known to possess low delay margins [Pollack and van Kampen, 2023]. The resulting diagram is equivalent to the original system with non-essential elements omitted, while retaining the dynamics relevant for robustness analysis.

The output at the break point, denoted by v_{out}, follows directly from the system dynamics:

$$v_{\text{out}} = \frac{1}{s}\left[G(\boldsymbol{x})\frac{A(s)}{1-H(s)A(s)}\hat{G}^{-1}(\boldsymbol{x})\left[\dot{v}_{\text{des}} - sH(s)v_{\text{in}}\right]\right]. \tag{5.27}$$

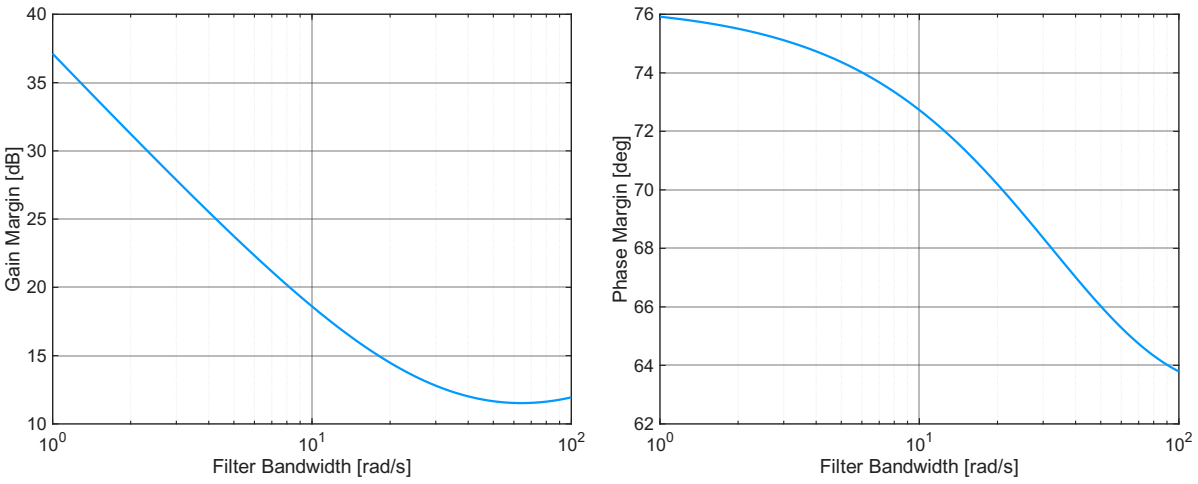
The control effectiveness is assumed to be known exactly, such that $G(\boldsymbol{x})\hat{G}^{-1}(\boldsymbol{x}) = 1$. For stability margin analysis, only the transfer function from v_{in} to v_{out} is required. Rearranging the terms and extracting gives for the synchronized configuration the relation:

$$G_{v_{\text{in}} \to v_{\text{out}}}(s) = -\frac{H(s)A(s)}{1 - H(s)A(s)}. \tag{5.28}$$

The classical gain and phase margins can then be evaluated using the margin command in MATLAB:

$$\text{margin}\left(-G_{v_{\text{in}} \to v_{\text{out}}}(s)\right)^{1}. \tag{5.29}$$

First, consider the case in which no filter is present. In this case, $H(s) = 1$ in Eq. (5.28). The gain margin then becomes infinite, the phase margin is 65.5°, and the delay margin is 0.036 s, corresponding to approximately three sample periods.



(a) Gain margin

(b) Phase margin

Figure 5.6: Classical stability margins for the broken loop in Figure 5.5 as a function of the low-pass filter bandwidth, for fixed actuator dynamics.

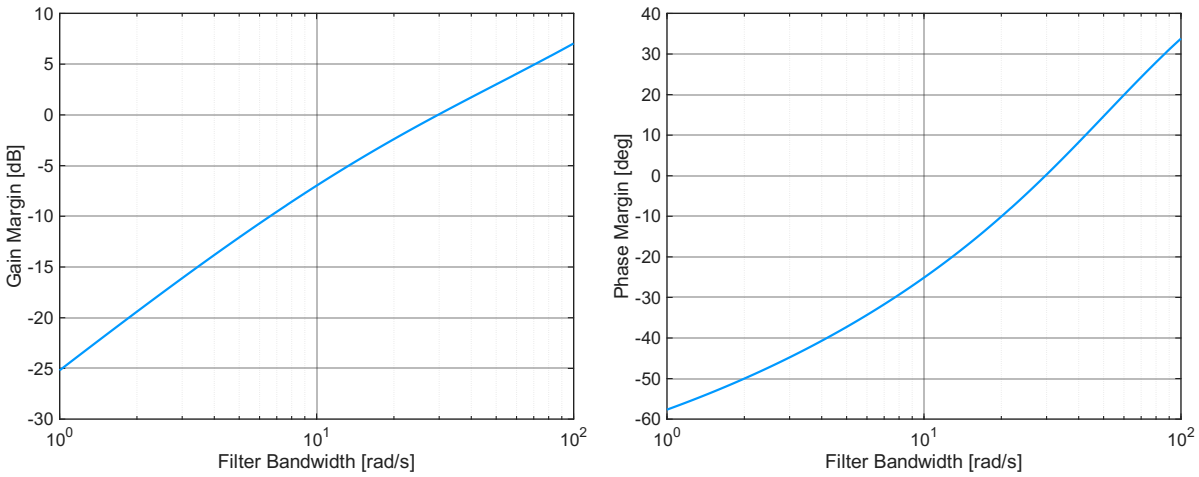
The classical stability margins as a function of the low-pass filter bandwidth are shown in Fig. 5.6. The gain and phase margins remain high over the full bandwidth range. The gain margin decreases from infinity to a minimum of approximately 11 dB near 70 rad/s and increases again thereafter, while the phase margin rises from 63° to 76°. Reducing the bandwidth increases the delay margin approximately logarithmically, whereas increasing the bandwidth causes the delay margin to asymptotically approach 0.035 s, which corresponds to the delay margin without a low-pass filter. These results are in close agreement with Raab [2025]. For the synchronized configuration with a low-pass filter bandwidth of 20 rad/s, the delay margin is 0.158 s, corresponding to approximately 15 sample periods for a sampling frequency of 100 Hz.

In the case that no synchronization is applied between the actuator and low-pass filter dynamics, the additional phase lag introduced by the low-pass filter is no longer compensated. The resulting transfer function from v_{in} to v_{out} is given by:

$$G_{v_{\text{in}} \to v_{\text{out}}}(s) = -\frac{H(s)A(s)}{1 - A(s)}. \tag{5.30}$$

[1]The negative sign is required because the extracted transfer function does not follow the standard negative feedback convention. In its current form, the closed-loop condition is $1 - G_{v_{\text{in}} \to v_{\text{out}}}(s) = 0$, whereas classical margin analysis assumes $1 + L(s) = 0$. Therefore, the loop transfer must be defined as $L(s) = -G_{v_{\text{in}} \to v_{\text{out}}}(s)$ to apply the margin definition correctly

The corresponding classical stability margins are shown in Fig. 5.7. As the filter bandwidth decreases, the additional phase lag introduced by the low-pass filter reduces the delay margin and can eventually lead to instability. For the selected low-pass filter with a bandwidth of 20 rad/s and unit damping, combined with the chosen actuator dynamics, the system is unstable.



(a) Gain margin

(b) Phase margin

Figure 5.7: Classical stability margins for the broken loop in Fig. 5.5 without synchronization in the actuator path, shown as a function of the low-pass filter bandwidth ω_H, for fixed actuator dynamics and $\zeta_H = 1$.

The preceding analysis establishes the basis for selecting the filter configuration. As shown, the bandwidth of $H(s)$ trades disturbance rejection against noise amplification. A corner frequency of 20 rad/s with unit damping is selected as the compromise: it suppresses noise sufficiently while retaining acceptable disturbance-rejection speed, and it is consistent with the bandwidths reported for flight-tested INDI controllers, in particular the 20 rad/s used on the Cessna Citation.

The stability analysis confirms that this choice is only viable with synchronization between the actuator and filter dynamics. The synchronized configuration retains large classical gain and phase margins together with the delay margin reported above, whereas without synchronization the uncompensated phase lag of the same filter drives the system unstable, so unsynchronized operation is not feasible at this bandwidth. The synchronized 20 rad/s, unit-damping configuration is therefore adopted, as it jointly satisfies the noise, disturbance-rejection, and robustness requirements of the angular rate control loop.

Attitude Control Loop

The attitude control loop governs the vehicle's orientation relative to the velocity direction and is responsible for tracking commanded attitude angles. In this formulation, the attitude is described by the bank angle μ_K, angle of attack α_K, and sideslip angle β_K, rather than by the Euler angles ϕ, θ, and ψ. The velocity control loop generates commands directly in terms of μ_K and α_K, so introducing an Euler angle representation would require an unnecessary transformation. It is assumed that the INS provides estimates of μ_K, α_K, and β_K[2]. The system is defined using the following state, output, and input variables:

$$\begin{aligned} \boldsymbol{x}_2 &= \begin{bmatrix} \mu_K & \alpha_K & \beta_K \end{bmatrix}^\top = \boldsymbol{y}_2, \\ \boldsymbol{u}_2 &= \begin{bmatrix} p & q & r \end{bmatrix}^\top. \end{aligned} \tag{5.31}$$

Here, $[p \quad q \quad r]$ are the inertial angular rates. Unlike the angular rate control loop, where the dynamics depend on aerodynamic moments and control effectiveness, the attitude loop is governed solely by kinematic relations between angular rates and the attitude angles. As a result, the system dynamics are known exactly and do not depend on uncertain aerodynamic parameters. NDI is therefore appropriate for this loop. The goal is now to express the rate of change of the aerodynamic angles as a nonlinear function of angular rates and flight path dynamics that can be written in the control-affine form as:

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(\boldsymbol{x}_2) + \boldsymbol{G}_2(\boldsymbol{x}_2)\,\boldsymbol{u}_2. \tag{5.32}$$

A set of dynamic attitude equations suitable for expressing this form was derived by Mooij [1997]. The dynamics of the aerodynamic angles, expressed with respect to the vehicle's geographic velocity, in the notation of this thesis, are given by[3]:

$$\begin{aligned} \dot{\mu} &= -\dot{\alpha}\sin\beta + \dot{\chi}\sin\gamma + \dot{\lambda}_d \sin\chi\cos\gamma - (\dot{l}+\omega_\oplus)(\cos\lambda_d\cos\chi\cos\gamma + \sin\lambda_d\sin\gamma) \\ &\quad + p\cos\alpha\sin\beta + q\sin\beta + r\sin\alpha\cos\beta, \\ \dot{\alpha} &= -\sin\mu/\cos\beta\big[\,\dot{\chi}\cos\gamma - \dot{\lambda}_d\sin\chi\sin\gamma + (\dot{l}+\omega_\oplus)(\cos\lambda_d\cos\chi\sin\gamma - \sin\lambda_d\cos\gamma)\big] \\ &\quad - \cos\mu/\cos\beta\big[\,\dot{\gamma} - \dot{\lambda}_d\cos\chi - (\dot{l}+\omega_\oplus)\cos\lambda_d\sin\chi\big] - p\cos\alpha\tan\beta + q - r\sin\alpha\tan\beta, \\ \dot{\beta} &= -\sin\mu\big[\,\dot{\gamma} - \dot{\lambda}_d\cos\chi - (\dot{l}+\omega_\oplus)\cos\lambda_d\sin\chi\big] + \cos\mu\big[\,\dot{\chi}\cos\gamma - \dot{\lambda}_d\sin\chi\sin\gamma\big] \\ &\quad + \cos\mu\big[(\dot{l}+\omega_\oplus)(\cos\lambda_d\cos\chi\sin\gamma - \sin\lambda_d\cos\gamma)\big] + p\sin\alpha - r\cos\alpha, \end{aligned} \tag{5.33}$$

where γ and χ are the Earth-relative flight-path and heading angles, l is the longitude, λ_d is the geodetic latitude, and $\omega_\oplus$ is Earth's angular velocity.

The original equation is unnecessarily complex for control law derivation. It can therefore be simplified by retaining only the dominant physical contributions. The equation is primarily influenced by the body rates, which must always be retained. The flight path rate terms $[\dot{\gamma} \quad \dot{\chi}]$ are also significant and are therefore included. In accordance with the *Small-Displacement Assumption*, variations in λ_d and l, and consequently their rates $\dot{\lambda}_d$ and $\dot{l}$, are considered negligible and omitted from the formulation [Juliana et al., 2004]. Consistent with the *Non-rotating Earth Assumption*, Earth's angular velocity is set to zero, $\omega_\oplus = 0$. The resulting reduced formulation captures the essential dynamics while preserving sufficient accuracy for control law development.

[2]The subscript K denotes kinematic angles, defined with respect to the velocity direction rather than the air-relative flow. As such, these quantities can be obtained from inertial velocity estimates without requiring direct air data measurements.

[3]For improved readability of the equations, the subscript K is omitted.

For the purposes of dynamic inversion, the term $\dot{\alpha}$ appearing in the $\dot{\mu}$ equation should be replaced by the expression for $\dot{\alpha}$ from Eq. (5.33). This substitution yields a form in which all derivatives of the aerodynamic angles are expressed explicitly in terms of measurable quantities and retained rates. The new set of equations is then:

$$\begin{aligned}
\dot{\mu} &= \dot{\chi}(\sin\gamma + \sin\mu\tan\beta\cos\gamma) + \dot{\gamma}\cos\mu\tan\beta + p\cos\alpha/\cos\beta + r\sin\alpha/\cos\beta, \\
\dot{\alpha} &= -\dot{\chi}\sin\mu\cos\gamma/\cos\beta - \dot{\gamma}\cos\mu/\cos\beta - p\cos\alpha\tan\beta + q - r\sin\alpha\tan\beta, \\
\dot{\beta} &= \dot{\chi}\cos\mu\cos\gamma - \dot{\gamma}\sin\mu + p\sin\alpha - r\cos\alpha.
\end{aligned} \tag{5.34}$$

This can be readily separated into the body rate contribution and the flight path dynamics. Therefore, if Earth rotation is neglected and changes in geodetic position are slow, the aerodynamic angle dynamics in control-affine form take the form:

$$\begin{bmatrix} \dot{\mu} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} \sin\gamma + \sin\mu\tan\beta\cos\gamma & \cos\mu_K\tan\beta \\ -\sin\mu\cos\gamma/\cos\beta & -\cos\mu/\cos\beta \\ \cos\mu\cos\gamma & -\sin\mu \end{bmatrix} \begin{bmatrix} \dot{\chi} \\ \dot{\gamma} \end{bmatrix} + \begin{bmatrix} \cos\alpha/\cos\beta & 0 & \sin\alpha/\cos\beta \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{5.35}$$

A virtual control input $\boldsymbol{\nu}_2$ is defined to represent the desired attitude angle rates. Enforcing $\dot{\boldsymbol{x}}_2 = \boldsymbol{\nu}_2$ in Eq. (5.35) and writing $\boldsymbol{u}_2 = \boldsymbol{\omega}$ gives:

$$\boldsymbol{\nu}_2 = \boldsymbol{f}_2(\boldsymbol{x}_2, \boldsymbol{x}_3) + \boldsymbol{G}_2(\boldsymbol{x}_2)\,\boldsymbol{\omega}. \tag{5.36}$$

In this expression, $\boldsymbol{G}_2$ maps the body angular rates to the attitude angle rates, while $\boldsymbol{f}_2$ represents the flight path dynamics[4]. The required body angular rates then follow by taking the inverse of Eq. (5.36):

$$\boldsymbol{\omega}^{\text{cmd}} = \boldsymbol{G}_2^{-1}(\boldsymbol{x}_2)\,(\boldsymbol{\nu}_2 - \boldsymbol{f}_2(\boldsymbol{x}_2, \boldsymbol{x}_3))\,. \tag{5.37}$$

Substituting $\boldsymbol{f}_2$ and $\boldsymbol{G}_2$ with the expressions given in Eq. (5.35), and subsequently inverting $\boldsymbol{G}_2$, yields the commanded body angular rates:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix}^{\text{cmd}} = \begin{bmatrix} \cos\alpha\cos\beta & 0 & \sin\alpha \\ \sin\beta & 1 & 0 \\ \sin\alpha\cos\beta & 0 & -\cos\alpha \end{bmatrix} \left(\begin{bmatrix} \nu_\mu \\ \nu_\alpha \\ \nu_\beta \end{bmatrix} - \begin{bmatrix} \sin\gamma + \sin\mu\tan\beta\cos\gamma & \cos\mu\tan\beta \\ -\sin\mu\cos\gamma/\cos\beta & -\cos\mu/\cos\beta \\ \cos\mu\cos\gamma & -\sin\mu \end{bmatrix} \begin{bmatrix} \dot{\chi} \\ \dot{\gamma} \end{bmatrix} \right). \tag{5.38}$$

The matrix $\boldsymbol{G}_2$ is invertible for all α in the operating regime of interest, provided the sideslip angle β remains small such that $\cos\beta \neq 0$. The singularity at $\cos\beta = 0$ lies outside the considered flight envelope. This inversion yields the commanded body rates required to track the desired changes in attitude angles.

The velocity control loop does not generate a sideslip command for the attitude control loop. Instead, the commanded sideslip angle β^{cmd} is set to zero by default.

[4]The influence of $\dot{\gamma}$ and $\dot{\chi}$ on the prior equation can alternatively be expressed as functions of the externally acting weight force, aerodynamic lift force, and aerodynamic side force [Snell et al., 1992]

Velocity Control Loop

The velocity control loop governs the translational dynamics of the vehicle and is responsible for flight–path tracking. It generates angle of attack and bank angle commands for the inner attitude control loop, and throttle commands for the propulsion system. The state vector consists of the vehicle speed V_K, heading angle χ_K, and flight-path angle γ_K, which fully describe the motion of the vehicle along its trajectory. The control inputs act on these states through thrust and orientation, where the throttle primarily influences the speed, while the bank angle and angle of attack affect the direction of the velocity vector. The system is defined using the following state, output, and input variables:

$$\begin{aligned} \boldsymbol{x}_3 &= \begin{bmatrix} V_K & \chi_K & \gamma_K \end{bmatrix}^\top = \boldsymbol{y}_3, \\ \boldsymbol{u}_3 &= \begin{bmatrix} \delta_T & \mu_K & \alpha_K \end{bmatrix}^\top. \end{aligned} \tag{5.39}$$

In contrast to the attitude loop, the translational dynamics depend directly on uncertain aerodynamic forces. This makes INDI suitable, as it compensates for modeling errors and neglected effects that act as disturbances on the slower velocity dynamics. The governing dynamics are formulated by expressing Newton's Second Law in the air-path coordinate system. The resulting time derivatives of the velocity states are given by[5]:

$$\dot{\boldsymbol{x}}_3 = \boldsymbol{M}^{-1}\left([T]^{VB}[f_{a,p}]^B + [T]^{VG}[f_g]^G + m\boldsymbol{a}_{\text{corr}}\right), \text{ where } \boldsymbol{M} = \begin{bmatrix} m & 0 & 0 \\ 0 & mV\cos\gamma & 0 \\ 0 & 0 & -mV \end{bmatrix}. \tag{5.40}$$

Here, $[f]^B$ is the aero-propulsive force vector expressed in body coordinates, $[f_g]^G$ is the gravitational force vector in the geocentric coordinates $[T]^{VG}$ and $[T]^{VB}$ are transformation matrices from geocentric and body coordinates to the velocity coordinates, which are given by Eq. (3.29)[6] and Eq. (3.35), respectively, and $\boldsymbol{a}_{\text{corr}}$ contains rotation and curvature corrections [Etkin, 2005]. The spherical Earth assumption implies that all latitude angles are identical, $\lambda_c = \lambda_d = \lambda$, and that the Earth radius function $R_0(\lambda_d)$ reduces to the constant mean spherical Earth radius R_0. In this case, the corrections are:

$$\boldsymbol{a}_{\text{corr}} = \begin{bmatrix} \omega_\oplus^2 R_0 \cos\lambda\,(\sin\gamma\cos\lambda - \cos\gamma\sin\lambda) \\ 2\omega_\oplus V(\sin\lambda\cos\gamma - \cos\lambda\sin\gamma\cos\chi) + \sin\chi\left(\dfrac{V^2}{R_0}\cos^2\gamma\tan\lambda + \omega_\oplus^2 R_0\cos\lambda\sin\lambda\right) \\ -2\omega_\oplus V(\cos\lambda\sin\chi) - \dfrac{V^2}{R_0}\cos\gamma - \omega_\oplus^2 R_0\cos\lambda(\cos\lambda\cos\gamma + \sin\gamma\sin\lambda\cos\chi) \end{bmatrix}. \tag{5.41}$$

As before this needs to be simplified, with the non-rotation Earth assumption the terms containing, $\omega_\oplus$ fall away. Together this yields the simplified form of the velocity dynamics for a spherical Earth[7]:

$$\begin{bmatrix} \dot{V} \\ \dot{\chi} \\ \dot{\gamma} \end{bmatrix} = \begin{bmatrix} m & 0 & 0 \\ 0 & mV\cos\gamma & 0 \\ 0 & 0 & -mV \end{bmatrix}^{-1} \left([T]^{VB}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + [T]^{VG}\begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + m\begin{bmatrix} 0 \\ \frac{V^2}{R_0}\cos^2\gamma\tan\lambda\sin\chi \\ -\frac{V^2}{R_0}\cos\gamma \end{bmatrix} \right). \tag{5.42}$$

[5] For improved readability of the equations, the subscript K is omitted.
[6] Due to the spherical Earth assumption, $[T]^{VD} = [T]^{VG}$.
[7] Under the spherical-Earth assumption, the gravitational acceleration is modeled using Newton's law of gravitation as $g = \frac{GM}{(R_0+h)^2}$, where G is the universal gravitational constant, M is the mass of the Earth, R_0 is the mean Earth radius, and h is the altitude above the Earth's surface.

All Earth rotation effects are ignored. Earth curvature effects are retained, and the gravity vector is assumed to follow an inverse square law and act vertically. It is also assumed that the sideslip angle β is kept at approximately zero by the attitude control loop. With $\beta = 0$, the transformation matrix $[T]^{VB}$ becomes:

$$[T]^{VB} = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ \sin\alpha\sin\mu & \cos\mu & -\cos\alpha\sin\mu \\ -\sin\alpha\cos\mu & \sin\mu & \cos\alpha\cos\mu \end{bmatrix}. \tag{5.43}$$

Next, it is noted that μ, α, and δ_T do not appear in an affine form in Eq. (5.40). The following step is to express the nonlinear dynamics in the standard input-affine form:

$$\dot{\boldsymbol{x}}_3 = \boldsymbol{f}_3(\boldsymbol{x}_3) + \boldsymbol{G}_3(\boldsymbol{x}_3)\,\boldsymbol{u}_3. \tag{5.44}$$

This can be partially accomplished by separating the force contributions to isolate the effects of bank angle, angle of attack, and thrust setting on the dynamics, thereby attempting to put the equations into a control-affine structure. The forces in body coordinates are modeled as:

$$\begin{aligned} X &= \bar{q}SC_X = \bar{q}S\left(C_{X_0} + C_{X_\alpha}\alpha_A + C_{X_q}\frac{q\bar{c}}{2V} + C_{X_{\delta_e}}\delta_e + C_{X_T}\delta_T\right), \\ Y &= \bar{q}SC_Y = \bar{q}S\left(C_{Y_0} + C_{Y_\beta}\beta_A + C_{Y_r}\frac{r\bar{b}}{2V} + C_{Y_p}\frac{p\bar{b}}{2V} + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r + C_{Y_T}\delta_T\right), \\ Z &= \bar{q}SC_Z = \bar{q}S\left(C_{Z_0} + C_{Z_\alpha}\alpha_A + C_{Z_q}\frac{q\bar{c}}{2V} + C_{Z_{\delta_e}}\delta_e + C_{Z_T}\delta_T\right). \end{aligned} \tag{5.45}$$

Here the angles α_A and β_A are used, rather than α_K and β_K, because the aerodynamic coefficients depend on the air–relative angles. This expression can be simplified by noting three things; first, the thrust vector is aligned with the 1^B-axis, which implies $C_{Z_T} = 0$ and $C_{Y_T} = 0$. Secondly, in the GHAME aerodynamic dataset, the coefficients C_{Y_0}, C_{Y_r}, C_{Y_p}, C_{X_q}, and C_{Z_q} are zero or not modeled. Finally, the coordinated flight assumption eliminates the contribution from C_{Y_β}. Together, this yields:

$$\begin{aligned} X &= \bar{q}S\left(C_{X_0} + C_{X_\alpha}\alpha_A + C_{X_{\delta_e}}\delta_e + C_{X_T}\delta_T\right), \\ Y &= \bar{q}S\left(C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right), \\ Z &= \bar{q}S\left(C_{Z_0} + C_{Z_\alpha}\alpha_A + C_{Z_{\delta_e}}\delta_e\right). \end{aligned} \tag{5.46}$$

It is now possible to solve Eq. (5.42) for each of its components and express the results in terms of lift and drag. The body-axis force coefficients are first transformed to the stability axis using the relations:

$$\begin{aligned} C_{D_i} &= -\left(\cos\alpha_A\, C_{X_i} + \sin\alpha_A\, C_{Z_i}\right), \\ C_{L_i} &= \sin\alpha_A\, C_{X_i} - \cos\alpha_A\, C_{Z_i}, \end{aligned} \tag{5.47}$$

where the subscript i denotes the contribution from a specific aerodynamic or propulsive effect. Another important information is that the controller only has access to the kinematic values of angle of attack, α_K, from the INS. So the on board-model introduces an error if wind is present. Substituting these relations into the velocity dynamics of Eq. (5.42) and retaining curvature and gravity variation effects yields:

$$\begin{aligned} \dot{V} &= \frac{\bar{q}S}{m}\left[C_{X_T}\delta_T\cos\alpha - C_{D_0} - C_{D_\alpha}\alpha\right] - g\sin\gamma, \\ \dot{\chi} &= \frac{\bar{q}S}{mV\cos\gamma}\left[\left(C_{X_T}\delta_T\sin\alpha + C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha\right)\sin\mu + C_Y\cos\mu\right] + \frac{V}{R_0}\cos\gamma\tan\lambda\sin\chi, \\ \dot{\gamma} &= \frac{\bar{q}S}{mV}\left[\left(C_{X_T}\delta_T\sin\alpha + C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha\right)\cos\mu - C_Y\sin\mu\right] + \cos\gamma\left(\frac{V}{R_0} - \frac{g}{V}\right). \end{aligned} \tag{5.48}$$

Looking at Eq. (5.48), the bank angle μ enters the dynamics only through the nonlinear terms $\sin\mu$ and $\cos\mu$. As a result, the velocity subsystem is not control-affine in μ, meaning it cannot be written in the form $f(x) + g_\mu(x)\,\mu$. This violates the standard requirement for applying NDI directly with μ treated as a control input.

INDI does not have this limitation [Bacon et al., 2001]. By using a first-order Taylor expansion, it can produce an inversion law that works locally around the operating point. A standard Taylor series expansion is taken about the values at the previous sampling instant, where $\boldsymbol{x}_{3,0} = \boldsymbol{x}_3(t-\Delta t)$ and $\boldsymbol{u}_{3,0} = \boldsymbol{u}_3(t-\Delta t)$. This yields for the usual INDI assumptions:

$$\begin{aligned}\dot{\boldsymbol{x}}_3 &= \dot{\boldsymbol{x}}_{3,0} + \left.\frac{\partial \dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial \boldsymbol{x}_3}\right|_0 \Delta\boldsymbol{x}_3 + \left.\frac{\partial \dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial \boldsymbol{u}_3}\right|_0 \Delta\boldsymbol{u}_3 + O(\Delta\boldsymbol{x}_3^2),\\ &\approx \dot{\boldsymbol{x}}_{3,0} + \left.\frac{\partial \dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial \boldsymbol{u}_3}\right|_0 \Delta\boldsymbol{u}_3,\end{aligned} \tag{5.49}$$

here $\Delta\boldsymbol{x}_3 = \boldsymbol{x}_3 - \boldsymbol{x}_{3,0}$, $\Delta\boldsymbol{u}_3 = \boldsymbol{u}_3 - \boldsymbol{u}_{3,0}$, and the term $O(\Delta\boldsymbol{x}_3^2)$ collects all higher–order contributions.

As in the rate control loop, a time–scale separation assumption is introduced: over one sampling step the state increment is much smaller than the control increment, $\Delta\boldsymbol{x}_3 \ll \Delta\boldsymbol{u}_3$. Attitude dynamics typically evolve on a much faster time scale than the flight path states, so the term involving $\Delta\boldsymbol{x}_3$ and the higher–order contributions can be neglected [Bacon and Ostroff, 2000]. From Eq. (5.48) and by using $C_L = C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha$, the control effectiveness matrix is found by taking the partial derivatives of each state rate $\dot{\boldsymbol{x}}_3$ with respect to each control input $\boldsymbol{u}_3$:

$$\frac{\partial \dot{\boldsymbol{x}}_3}{\partial \boldsymbol{u}_3} = \frac{\bar{q}S}{mV}\begin{bmatrix} C_{X_T}\cos\alpha V & 0 & -(C_{X_T}\delta_T\sin\alpha + C_{D_\alpha})V \\ \dfrac{C_{X_T}\sin\alpha\sin\mu}{\cos\gamma} & \dfrac{(C_{X_T}\delta_T\sin\alpha + C_L)\cos\mu - C_Y\sin\mu}{\cos\gamma} & \dfrac{(C_{X_T}\delta_T\cos\alpha + C_{L_\alpha})\sin\mu}{\cos\gamma} \\ C_{X_T}\sin\alpha\cos\mu & -(C_{X_T}\delta_T\sin\alpha + C_L)\sin\mu - C_Y\cos\mu & (C_{X_T}\delta_T\cos\alpha + C_{L_\alpha})\cos\mu \end{bmatrix}. \tag{5.50}$$

To evaluate these partial derivatives, several simplifying assumptions are introduced at the linearization point. Vertical force equilibrium is expressed in a form consistent with the spherical Earth model assumption:

$$\bar{q}S\left(C_{X_T}\delta_T\sin\alpha + C_L\right)\cos\mu = m\cos\gamma\left(g - \frac{V^2}{R_0}\right), \tag{5.51}$$

ensuring that aerodynamic lift and the vertical component of thrust balance both gravity and the curvature-induced centrifugal term. Additionally, the influence of thrust on lift and drag derivatives is neglected, as aerodynamic forces dominate in magnitude:

$$C_{L_\alpha}\bar{q}S \gg \bar{q}SC_{X_T}\delta_T\cos\alpha, \qquad C_{D_\alpha}\bar{q}S \gg \bar{q}SC_{X_T}\delta_T\sin\alpha.$$

The lateral coefficient C_Y is also neglected. With $\beta = 0$ its magnitude is much smaller than the lift- and thrust-related terms, and because the remaining C_Y signal contains small-magnitude high-frequency fluctuations that lead to noisy bank angle commands. Substituting Eq. (5.51) into Eq. (5.50) and removing negligible terms yields the following expression for the control effectiveness matrix:

$$\boldsymbol{G}_3(\boldsymbol{x}_3,\boldsymbol{u}_3) \approx \frac{\bar{q}S}{mV}\begin{bmatrix} C_{X_T}\cos\alpha V & 0 & -C_{D_\alpha}V \\ \dfrac{C_{X_T}\sin\alpha\sin\mu}{\cos\gamma} & \dfrac{m}{\bar{q}S}\left(g - \dfrac{V^2}{R_0}\right) & \dfrac{C_{L_\alpha}\sin\mu}{\cos\gamma} \\ C_{X_T}\sin\alpha\cos\mu & -\dfrac{m\tan\mu\cos\gamma}{\bar{q}S}\left(g - \dfrac{V^2}{R_0}\right) & C_{L_\alpha}\cos\mu \end{bmatrix}. \tag{5.52}$$

A virtual input $\boldsymbol{\nu}_3 = [\nu_V \quad \nu_\chi \quad \nu_\gamma]^\top$ is now defined to represent the desired rates of the velocity states. Linearizing the dynamics around the previously measured operating point yields the incremental inversion law:

$$\begin{bmatrix} \delta_T \\ \mu \\ \alpha \end{bmatrix}^{\text{cmd}} = \begin{bmatrix} \delta_T \\ \mu \\ \alpha \end{bmatrix}_0 + \boldsymbol{G}_3^{-1} \left(\begin{bmatrix} \nu_V \\ \nu_\chi \\ \nu_\gamma \end{bmatrix} - \begin{bmatrix} \dot{V} \\ \dot{\chi} \\ \dot{\gamma} \end{bmatrix}_0 \right). \tag{5.53}$$

The matrix $\boldsymbol{G}_3$ becomes non-invertible only in vertical flight ($\gamma = \pm 90°$), knife-edge flight ($\mu = \pm 90°$), or in the limiting case $g = V^2/R_0$, where centrifugal force cancels gravity and lift can no longer steer the velocity vector.

The inversion relies on the aerodynamic force coefficients, which may contain significant uncertainty. In addition, wind introduces errors because the onboard model uses kinematic angles from the INS rather than the true air–relative angles. These effects appear as modeling errors and external disturbances that vary slowly, allowing the INDI controller to reject them effectively.

Position Control Loop

The outermost loop governs the vehicle's position by generating reference commands for velocity, heading, and flight–path angle, which are tracked by the velocity–control loop. Its purpose is twofold. First, it enables trajectory tracking, allowing the vehicle to follow prescribed references in longitude, latitude, and altitude. Second, it enables test–case generation, where the performance of heading and altitude tracking is evaluated. The system is formulated using the state vector, output vector, and control input shown below, where l denotes the vehicle's longitude, λ its latitude, and h its altitude above a spherical Earth:

$$\begin{aligned} \boldsymbol{x}_4 &= \begin{bmatrix} l & \lambda & h \end{bmatrix}^\top = \boldsymbol{y}_4 \\ \boldsymbol{u}_4 &= \begin{bmatrix} u_E & u_N & u_U \end{bmatrix}^\top. \end{aligned} \tag{5.54}$$

In this context the heading angle χ_K is defined in the standard navigation convention: $\chi_K = 0°$ corresponds to motion due north, $\chi_K = 90°$ to motion due east, $\chi_K = 180°$ to motion due south, and $\chi_K = 270°$ to motion due west. A positive latitude rate corresponds to northward motion, a positive longitude rate corresponds to eastward motion, and a positive altitude rate corresponds to an increase in altitude.

With this convention, the translational kinematics for a point–mass model on a spherical Earth are[8]:

$$\dot{l} = \frac{V^{\text{ref}} \sin\chi \cos\gamma}{(R_0 + h)\cos\lambda}, \quad \dot{\lambda} = \frac{V^{\text{ref}} \cos\chi \cos\gamma}{R_0 + h}, \quad \dot{h} = V^{\text{ref}} \sin\gamma, \tag{5.55}$$

with R_0 the mean Earth radius and V^{ref} the prescribed geographic speed[9]. The velocity loop is assumed to be able to maintain the reference geographic speed V^{ref}, which is therefore treated as constant.

For nonlinear dynamic inversion it is convenient to define an intermediate input vector:

$$\boldsymbol{u} = \begin{bmatrix} u_E \\ u_N \\ u_U \end{bmatrix} = \begin{bmatrix} V^{\text{ref}} \sin\chi \cos\gamma \\ V^{\text{ref}} \cos\chi \cos\gamma \\ V^{\text{ref}} \sin\gamma \end{bmatrix}, \tag{5.56}$$

where u_E, u_N, and u_U denote the linear velocity components of the vehicle in the geographic East–North–Up coordinate system. These components serve as intermediate control variables: when

[8] For improved readability of the equations, the subscript K is omitted.

[9] The superscript "ref" denotes an externally specified reference command provided by the guidance system or designer, whereas the superscript "cmd" denotes an internally generated command signal computed within the control loop and passed to the subsequent inner loop.

mapped through the spherical–dependent matrix $\boldsymbol{G}_4(\boldsymbol{x})$, they yield the correct time derivatives of (l, λ, h). The dynamics of the output vector can then be expressed in control–affine form as:

$$\dot{\boldsymbol{y}}_4 = \boldsymbol{f}_4(\boldsymbol{x}) + \boldsymbol{G}_4(\boldsymbol{x})\,\boldsymbol{u}_4. \tag{5.57}$$

From the control–affine representation the output dynamics can be written explicitly as:

$$\begin{bmatrix} \dot{l} \\ \dot{\lambda} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{(R_0+h)\cos\lambda} & 0 & 0 \\ 0 & \dfrac{1}{R_0+h} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_E \\ u_N \\ u_U \end{bmatrix}. \tag{5.58}$$

In nonlinear dynamic inversion, the virtual control input $\boldsymbol{\nu}_4$ is applied by enforcing the output dynamics such that $\dot{\boldsymbol{y}}_4 = \boldsymbol{\nu}_4$, which gives $\boldsymbol{\nu}_4 = \boldsymbol{G}_4(\boldsymbol{x})\,\boldsymbol{u}_4$ so that the commanded inputs are:

$$\boldsymbol{u}_4^{\text{cmd}} = \boldsymbol{G}_4^{-1}(\boldsymbol{x})\,\boldsymbol{\nu}_4, \tag{5.59}$$

which yields the explicit inversion law:

$$\begin{bmatrix} u_E \\ u_N \\ u_U \end{bmatrix}^{\text{cmd}} = \begin{bmatrix} (R_0+h)\cos\lambda & 0 & 0 \\ 0 & (R_0+h) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \nu_l \\ \nu_\lambda \\ \nu_h \end{bmatrix}, \tag{5.60}$$

where ν_l, ν_λ, and ν_h denote the virtual control inputs for the longitudinal, latitudinal, and vertical channels, respectively. The commanded flight–path and heading angles are obtained directly by inverting the kinematic relations:

$$\gamma^{\text{cmd}} = \arcsin\left(\frac{u_U}{V^{\text{ref}}}\right), \qquad \chi^{\text{cmd}} = \operatorname{atan2}\left(u_E, u_N\right). \tag{5.61}$$

These angles, together with the reference speed, form the commanded inputs to the velocity control loop. The required virtual control inputs ν_l, ν_λ, and ν_h are generated from reference trajectories defined using the reference altitude h^{ref}, latitude l^{ref}, and longitude λ^{ref}. These references may either correspond to constant setpoints or to waypoint-based trajectories generated by the guidance system.

For control–system testing, however, the objective is not to follow arbitrary reference trajectories, but to assess how well the system can maintain a prescribed heading or climb at a fixed climb rate. Therefore, two test modes are implemented to facilitate these specific scenarios.

1. **Mode 1: Constant–rate climb to altitude:**

   In this mode, the vehicle is commanded to reach a target altitude h^{cmd} while maintaining a specified vertical rate $\dot{h}^{\text{ref}}$. The commanded up–velocity component is therefore imposed directly as:

   $$u_U = \dot{h}^{\text{ref}}. \tag{5.62}$$

   This value is substituted into Eq. (5.61) to compute the commanded flight–path angle γ^{cmd}. Once the altitude error is eliminated (i.e. $h \approx h^{\text{ref}}$), the climb rate command is set to zero. This mode can also be combined with Mode 2.

2. **Mode 2: Heading–hold:**

   In this mode, when heading–hold is enabled (e.g. `mheading = 1`), any heading value calculated from Eq. (5.61) is overridden and the commanded heading is specified directly as:

   $$\chi^{\text{cmd}} = \chi^{\text{ref}}, \tag{5.63}$$

   while the commanded flight–path angle γ^{cmd} is still obtained from Eq. (5.61) to ensure that the altitude h tracks its reference h^{ref}.

This completes the design of the position–control loop. Its feasibility depends on several conditions that ensure the inversion law is well–defined and that the commanded angles remain physically meaningful. Specifically, the input–gain matrix $\boldsymbol{G}_4(\boldsymbol{x})$ must be invertible, which requires $\cos\lambda \neq 0$. The inner velocity loop must accurately track the commanded reference speed. The east and north velocity components (u_E, u_N) cannot both be zero, since in that case the commanded heading χ^{cmd} becomes undefined. Finally, the longitude l is ill–defined at the poles ($\lambda = \pm 90°$), where the geographic equations lose validity.

5.3. Linear Controller Design

Nonlinear dynamic inversion reshapes the system dynamics into approximate chains of integrators, which then serve as the foundation for feedback design. To reduce the design burden, a time–scale separation assumption is adopted, so that the faster inner loops appear instantaneous from the perspective of the slower outer loops. This hierarchical structure allows each loop to be designed independently.

With the dynamics reduced to integrator–like behavior and the control architecture organized hierarchically, the remaining task is to specify the desired transient and steady–state characteristics. This is accomplished by placing linear controllers on top of the inverted dynamics. The following subsections describe the design of these controllers at each level of the hierarchy. The hierarchical control architecture follows a set of design requirements:

1. **Rate Requirement:** The rate loop requires a 0% overshoot at the lowest settling time for good performance.
2. **Damping requirements:** The outer loops are designed to be well damped, with
$$\zeta_\theta = 0.7, \qquad \zeta_V = 0.9, \qquad \zeta_X = 0.7,$$
ensuring fast responses with little overshoot.
3. **Time–scale separation:** Each loop is designed to be significantly slower than the loop beneath it. A practical and commonly used choice is a bandwidth ratio of approximately four between adjacent loops, leading to:
$$\text{BW}_\theta \leq \frac{\text{BW}_\omega}{4}, \qquad \text{BW}_V \leq \frac{\text{BW}_\theta}{4}, \qquad \text{BW}_X \leq \frac{\text{BW}_V}{4}.$$
4. **Robustness margins:** All loops must achieve a classical phase margin of at least 40° and a gain margin of at least 6 dB. In addition, disk margins must guarantee robustness to simultaneous gain and phase variations of at least ±3 dB and ±20°.

Rate Loop

The design of the angular–rate controller starts from the observation that, under ideal INDI and in the absence of disturbances and sensor noise, the airframe dynamics are cancelled by the inversion, leaving only the actuator dynamics $A(s)$ in the loop. This follows directly from Eq. (5.25). The assumption of rapid disturbance rejection and effective noise suppression is justified for a suitably chosen filter $H(s)$, as established in the earlier analysis of the INDI structure.

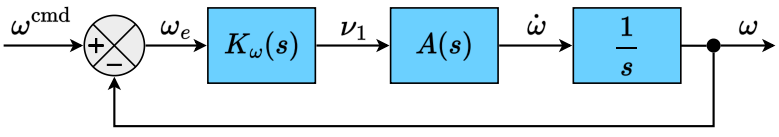
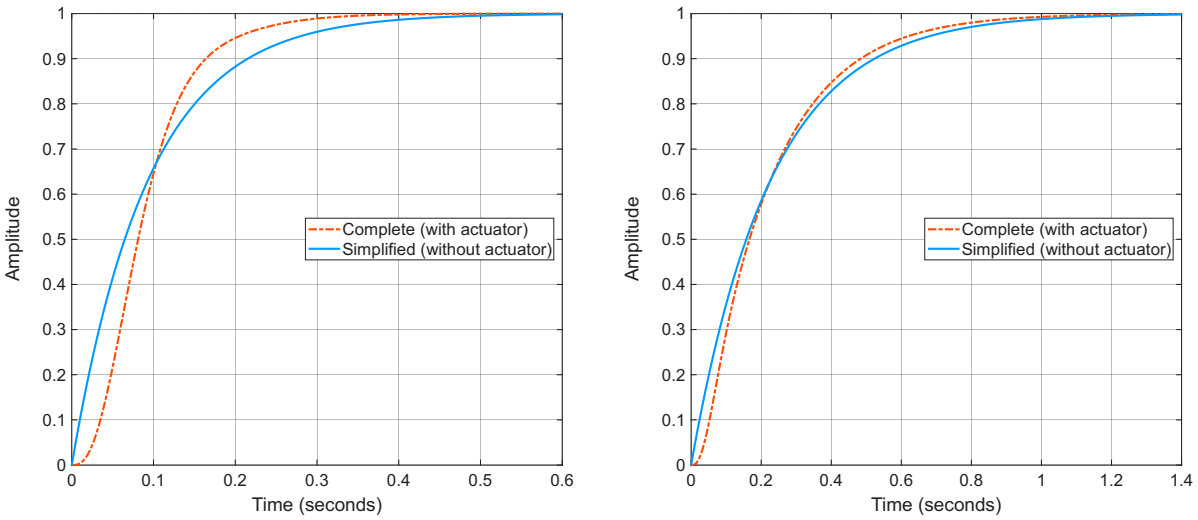


Figure 5.8: Idealized closed–loop structure of the angular–rate control loop.

Figure 5.8 illustrates the corresponding idealized closed–loop structure of the angular–rate controller, where the linear controller $K_\omega(s)$ generates the virtual input ν_1 from the rate error, and the integrator yields the angular rate $\boldsymbol{\omega}$, which corresponds to the state vector $\boldsymbol{x}_1$. Thus, in this simplified setting the closed–loop $\boldsymbol{\omega}^{\text{cmd}}(s)$ to $\boldsymbol{\omega}(s)$ is:

$$H_\omega(s) = \frac{A(s)K_\omega(s)}{s + A(s)K_\omega(s)}. \tag{5.64}$$

The actuator dynamics are often assumed to be sufficiently fast so that $A(s) \approx 1$ over the frequencies of interest [Smeur et al., 2016a]. This is reasonable for systems with very fast actuators, such as small UAVs, where bandwidths can reach several hundred rad/s. When actuator bandwidth is limited, this assumption breaks down, and the predicted loop performance no longer reflects the true limitations, which affect stability and achievable bandwidth. Two approaches can address this. The first approach is to reduce the rate–loop bandwidth to a fraction of the actuator bandwidth so that $A(s) \approx 1$ holds, at the cost of slower response. The other is to include the actuator dynamics $A(s)$ explicitly in the controller design, avoiding this limitation.



(a) Bandwidth ratio 3:1.

(b) Bandwidth ratio 10:1.

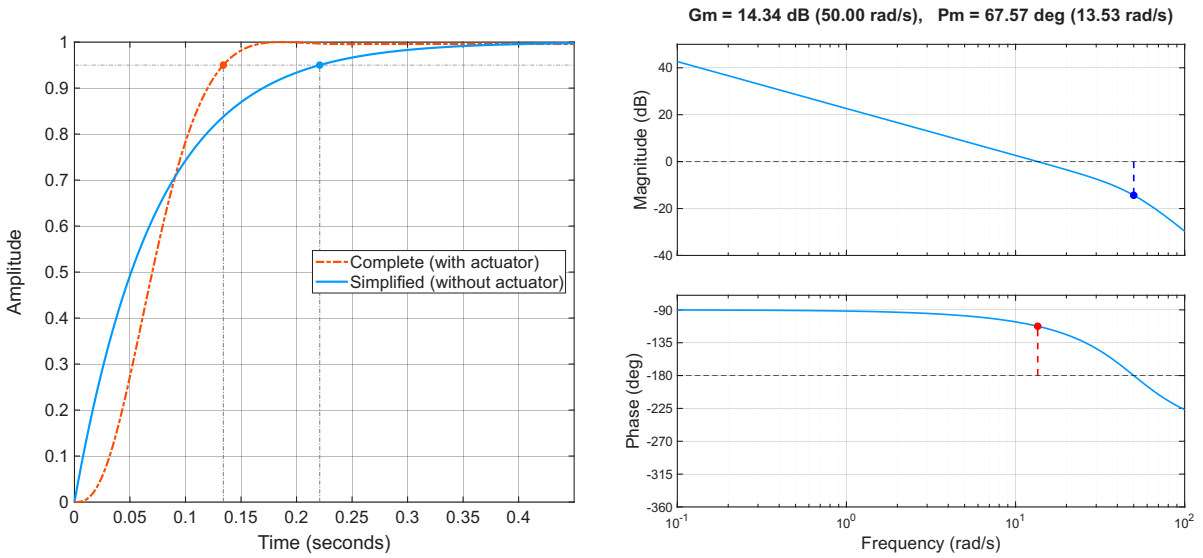
Figure 5.9: Rate–loop step responses for different actuator bandwidth separation ratios.

For the present aircraft, an actuator bandwidth of 50 rad/s clearly illustrates this limitation. Subfigure 5.9a shows that with a bandwidth ratio of three between the actuator and the rate loop, actuator dynamics still noticeably affect the response. Only at a ratio of ten, as shown in Subfig. 5.9b, does this influence become negligible. Achieving such a ratio would require reducing the rate–loop bandwidth to about 5 rad/s, which is not acceptable for a high-performance controller. For this reason, the controller developed in this thesis adopts the second approach and explicitly includes the actuator dynamics $A(s)$ in both the design and stability analysis. This enables a practically relevant rate–loop bandwidth while accurately capturing actuator effects.

The controller is designed for the third–order plant $P(s) = A(s)\frac{1}{s}$ formed by the second–order actuator dynamics $A(s)$ from Eq. (4.66) in series with the integrator as can be seen in Fig. 5.8. For this plant, a proportional control strategy was adopted, using proportional control, $K_\omega(s) = K_p$. For the system under consideration, the maximum admissible gain is determined from the root locus of $L(s) = K_p P(s)$. The gain is increased until the overshoot constraint is just satisfied, yielding a 0% overshoot and a 5% settling time of approximately 0.13 s:

$$K_p = 13.45. \tag{5.65}$$

The proportional controller yields a fast-settling step response with zero overshoot, as shown in Subfig. 5.10a. The controller was designed with the actuator dynamics included in the loop. For comparison, neglecting the actuator dynamics increases the settling time by approximately 0.08 s and the rise time by approximately 0.06 s. The zero-overshoot response is consistent with the large phase and gain margins observed in the open-loop response in Subfig. 5.10b, indicating a well-damped closed-loop system. A summary of the classical and disk margins is provided in Table. 5.3.



(a) (b)

Figure 5.10: Closed-loop performance and open-loop stability characteristics of the ideal angular rate controller. Subfig. (a) shows the closed loop step response, and Subfig. (b) shows the open-loop gain and phase margins.

The large stability margins are a consequence of the strict 5% overshoot requirement. Reducing the phase margin further would increase the closed-loop bandwidth and crossover frequency, but would also reduce the damping of the dominant complex poles and therefore increase the overshoot beyond the specified limit. Consequently, the proportional controller is already operating near the achievable performance limit for the imposed transient-response constraint.[10] The closed-loop pole locations are shown in Fig. 5.11. The system contains a real pole at -24.96 and a complex conjugate pair at $-22.87 \pm 28.90i$ with damping ratio $\zeta = 0.62$ and natural frequency of approximately 36 rad/s.

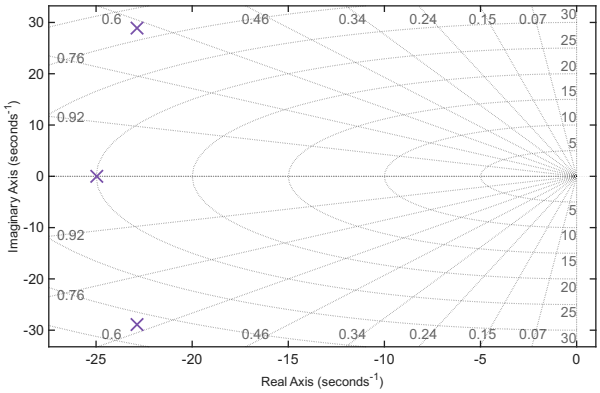


Figure 5.11: Closed–loop stability characteristics of the rate controller including the actuator dynamics.

To evaluate the steady-state tracking performance of a proportional controller applied to the plant, $P(s) = A(s)\frac{1}{s}$, the error dynamics are analyzed explicitly. With a proportional controller $K_\omega(s) = K_p$ the tracking error is defined in the Laplace domain as:

$$E(s) = \frac{1}{1 + L(s)} \cdot R(s), \quad \text{with} \quad L(s) = K_p \cdot P(s), \tag{5.66}$$

where $R(s)$ is the Laplace transform of the reference input. For a unit step input, $R(s) = 1/s$, the steady-state error is obtained using the final value theorem:

$$e_\infty = \lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} \left(\frac{s}{1 + K_p P(s)} \cdot \frac{1}{s} \right) = \lim_{s\to 0} \frac{1}{1 + K_p P(s)}. \tag{5.67}$$

Noting that $P(s) = A(s)/s$, the plant contains an integrator through the $1/s$ term. Furthermore, since $A(0) = 1$, the remaining dynamics have a finite nonzero value at zero frequency. Consequently, as s approaches zero, the proportional open-loop transfer function $K_p P(s)$ becomes unbounded because the integrator causes the low-frequency gain to increase without limit. Therefore, the system has infinite DC gain, which results in zero steady-state error for a step reference input under ideal conditions. Therefore,

$$e_\infty = \lim_{s\to 0} \frac{1}{1 + K_p P(s)} = 0. \tag{5.68}$$

The presence of an integrator in the plant guarantees that the error dynamics asymptotically drive the tracking error to zero under proportional feedback, provided the reference is a step input. This confirms that the system achieves zero steady-state error despite the simplicity of the controller.

[10] A two-degree-of-freedom architecture with a reference prefilter could be used to reduce overshoot without altering the feedback-loop stability margins.

Attitude Loop

For the linear controller in the attitude loop, the first step is to obtain the closed–loop expression of the system, shown schematically in Fig. 5.12. Here, NDI is used rather than the incremental form. Unlike INDI, where additional dynamics appeared as disturbances to reject, the NDI formulation retains these explicitly. In this case it is the term $\boldsymbol{f}_2$, representing the flight–path dynamics given in Eq. (5.35):

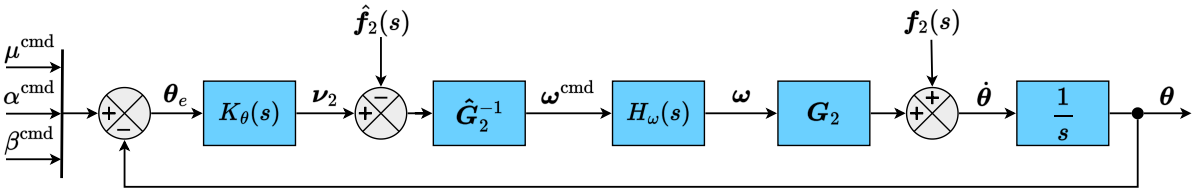


Figure 5.12: Closed-loop architecture for attitude control using model-based inversion and linear compensation.

The block diagram represents a cascaded control structure. The commands for μ^{cmd} and α^{cmd} are supplied by the outer velocity loop, while the sideslip reference is fixed at $\beta^{\text{cmd}} = 0$ to enforce coordinated flight. A linear controller $K_\theta(s)$ regulates the commanded attitude error $\boldsymbol{\theta}_e$ to produce the virtual control vector $\boldsymbol{\nu}_2(s)$, which defines the desired attitude rate. The angular rate command is obtained by subtracting the estimated kinematic term $\hat{\boldsymbol{f}}_2(s)$ from $\boldsymbol{\nu}_2(s)$ and mapping the result through the inverse of the kinematic matrix as given in Eq. (5.38).

The matrices $\boldsymbol{G}_2(s)$ and $\hat{\boldsymbol{G}}_2(s)$ map angular rates to attitude rates. Since this relation is known analytically and the angle of attack and sideslip are assumed to be estimated with sufficient accuracy, the two matrices are considered equal. The resulting closed-loop relation for $\boldsymbol{\theta}(s)$ is:

$$\boldsymbol{\theta}(s) = \frac{H_\omega(s)\,K_\theta(s)}{s + H_\omega(s)\,K_\theta(s)}\,\boldsymbol{\theta}^{\text{cmd}}(s) - \frac{H_\omega(s)}{s + H_\omega(s)\,K_\theta(s)}\,\hat{\boldsymbol{f}}_2(s) + \frac{1}{s + H_\omega(s)\,K_\theta(s)}\,\boldsymbol{f}_2(s). \tag{5.69}$$

At this stage, the bandwidth separation assumption becomes relevant. The standard approach is to design the inner rate loop to operate on a sufficiently faster timescale than the outer attitude loop, such that it can be approximated as ideal, i.e. $\omega(s) \approx \omega^{\text{cmd}}(s)$, which implies $H_\omega(s) \approx 1$. With this approximation, the attitude dynamics reduce to:

$$\boldsymbol{\theta}(s) = \frac{K_\theta(s)}{s + K_\theta(s)}\,\boldsymbol{\theta}^{\text{cmd}}(s) + \frac{1}{s + K_\theta(s)}\left(\boldsymbol{f}_2(s) - \hat{\boldsymbol{f}}_2(s)\right), \tag{5.70}$$

where $\boldsymbol{f}_2(s)$ represents the true flight-path kinematics and $\hat{\boldsymbol{f}}_2(s)$ denotes its model estimate used in the controller. Due to the assumptions in the attitude control loop design, $\boldsymbol{f}_2(s) \neq \hat{\boldsymbol{f}}_2(s)$, but the mismatch is extremely small in magnitude and varies slowly with time [Juliana et al., 2004]. As a result, the induced tracking error remains bounded and does not compromise overall system performance. The closed loop from $\boldsymbol{\theta}^{\text{cmd}}(s)$ to $\boldsymbol{\theta}(s)$ is therefore:

$$\lim_{H_\omega(s)\to 1} H_\theta(s) = \frac{K_\theta(s)}{s + K_\theta(s)}. \tag{5.71}$$

This assumption effectively decouples the rotational axes in the outer loop, allowing each attitude channel to be treated independently. While this simplification introduces a potential risk of instability, exponential stability of the outer-loop states around their commanded values is still guaranteed if the inner-loop bandwidth is sufficiently high [Schumacher and Khargonekar, 1998]. In practice, it is recommended that the rate loop bandwidth exceeds the attitude loop bandwidth by at least a factor of three to ensure that the approximation remains accurate [Naidu and Calise, 2001].

For this single–integrator loop structure, exact second–order closed-loop dynamics with the desired natural frequency ω_θ and damping ratio ζ_θ can be obtained through pole placement. A purely proportional controller is not sufficient for this purpose. Substituting $K_\theta(s) = K_\theta$ into Eq. (5.71) yields a first-order system. As a result, only a single pole can be assigned, and independent specification of ω_θ and ζ_θ is not possible. To obtain second-order dynamics, a proportional-with-roll-off controller is introduced:

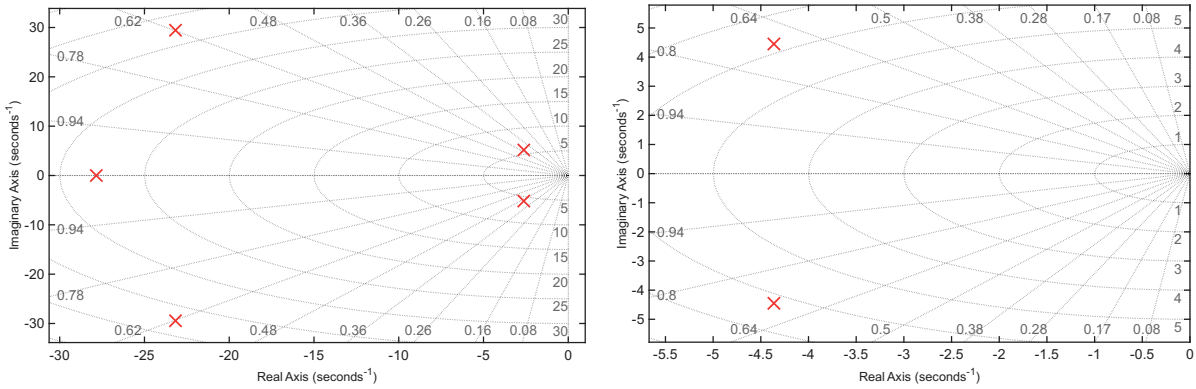
$$K_\theta(s) = \frac{K_\theta \omega_f}{s + \omega_f} = \frac{\omega_\theta^2}{s + 2\zeta_\theta \omega_\theta}, \tag{5.72}$$

where the second equality follows from coefficient matching using $\omega_f = 2\zeta_\theta\omega_\theta$ and $K_\theta = \dfrac{\omega_\theta}{2\zeta_\theta}$.

These expressions directly link the controller parameters to the desired dynamic characteristics. Using the design values specified in the control requirements yields the second order closed-loop system:

$$H_\theta(s) = \frac{38.84}{s^2 + 11.22s + 38.84}, \quad \text{using} \quad K_\theta(s) = \frac{38.84}{s + 11.22}. \tag{5.73}$$

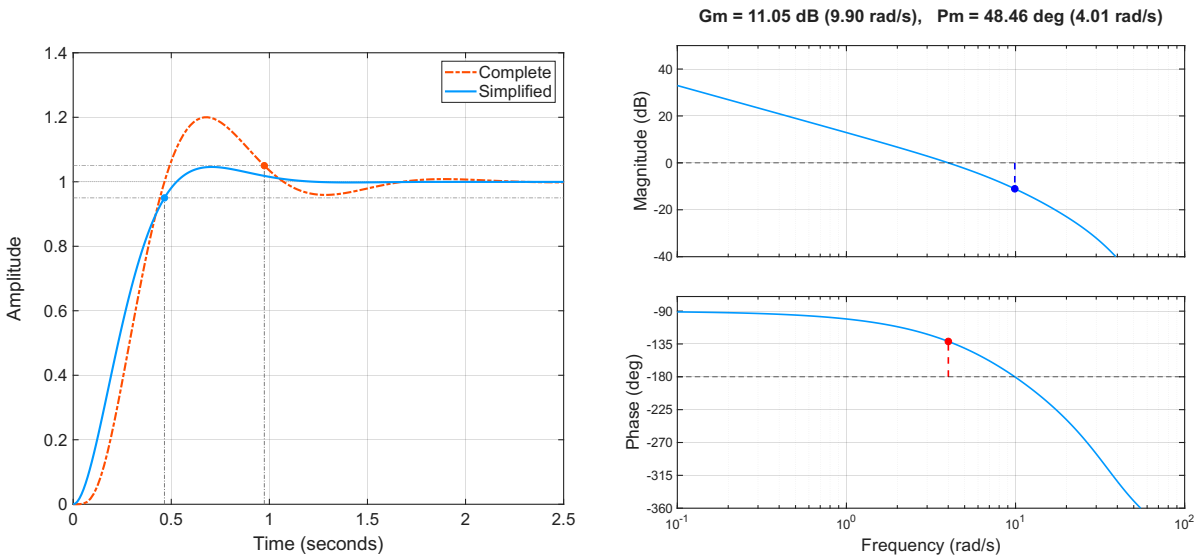
The difference in closed-loop pole locations is shown in Fig. 5.13. The reduced system places the poles exactly at the designed second-order locations, whereas the full system also contains contributions from the actuator and rate loop dynamics. Consequently, unlike the rate control loop, the desired damping ratio is not fully achieved in the complete closed-loop system. Increasing the time-scale separation would cause the damping ratio to approach the desired value of 0.9, but this would also slow down the overall system response. Therefore, a trade-off exists between achieving the exact damping ratio and maintaining a faster closed-loop response. Nevertheless, the discrepancy is considered acceptable for the controller design, since the dominant pole behavior is still captured sufficiently accurately.



(a) (b)

Figure 5.13: Closed–loop stability characteristics of the attitude controller. Subfig. (a) shows the closed–loop poles including the inner–loop dynamics, and Subfig. (b) shows the poles assuming ideal inner loops.

Moreover, since the gain and phase margins in Subfig. 5.14b are already large, further increasing the time–scale separation would only enlarge them unnecessarily and make the system more sluggish. The corresponding effect on the step response is shown in Subfig. 5.14a. The “Simplified” curve assumes instantaneous inner–loop dynamics, resulting in a monotonic rise without overshoot, whereas the “Complete” response includes the inner rate–loop dynamics, producing a slightly faster rise accompanied by a small overshoot. As the bandwidth separation increases, the difference between the idealized and full–system responses becomes negligible.



(a) (b)

Figure 5.14: Closed–loop performance and open–loop stability characteristics of the full and reduced attitude control loops. Subfig. (a) compares the closed–loop step responses, while Subfig. (b) shows the open–loop gain and phase margins of the full system.

To show that the controller achieves zero steady–state error when tracking a step command, the error dynamics can be expressed in the Laplace domain as:

$$E(s) = \frac{R(s)}{1 + L(s)}, \qquad L(s) = \frac{H_\omega(s)}{s} K_\theta(s), \tag{5.74}$$

where $L(s)$ is the open–loop transfer function composed of the inner angular–rate loop $H_\omega(s)$, the integrator $1/s$, and the outer attitude controller $K_\theta(s)$. For a unit–step input $R(s) = 1/s$, applying the Final Value Theorem gives:

$$e_\infty = \lim_{s \to 0} sE(s) = \lim_{s \to 0} \frac{s}{s + H_\omega(s) K_\theta(s)}. \tag{5.75}$$

At low frequency, the inner loop behaves as unity gain ($H_\omega(s) \to 1$) because the actuator has a finite, nonzero DC gain ($A(0) \neq 0$), allowing the rate loop to track constant commands without steady–state error. The attitude controller approaches a finite gain ($K_\theta(s) \to \omega_\theta / (2\zeta_\theta)$). Substituting these limits into Eq. (5.75) yields:

$$e_\infty = \lim_{s \to 0} \frac{s}{s + 1 \cdot \frac{\omega_\theta}{2\zeta_\theta}} = 0, \tag{5.76}$$

confirming that the attitude loop eliminates steady–state error for a step command.

Velocity Loop

The outer velocity loop regulates the geographic airspeed V_K, the heading angle χ_K, and the flight–path angle γ_K. This regulation is achieved indirectly by generating small changes in attitude and thrust, which the inner attitude and rate loops convert into linear accelerations.

The cascaded control structure used for velocity regulation is illustrated in Fig. 5.15, where the velocity state vector $\boldsymbol{V}$ corresponds to $\boldsymbol{x}_3$. Commands originate from the position control loop, and the resulting command error is processed by the linear controller K_V to produce the desired attitude and thrust setting rates. The error in the commanded derivatives is mapped through the control effectiveness matrix from Eq. (5.48), generating incremental attitude commands. These increments are added to the

previous values and forwarded to the inner loops for bank angle and angle of attack, while the throttle command is sent directly to the engine.

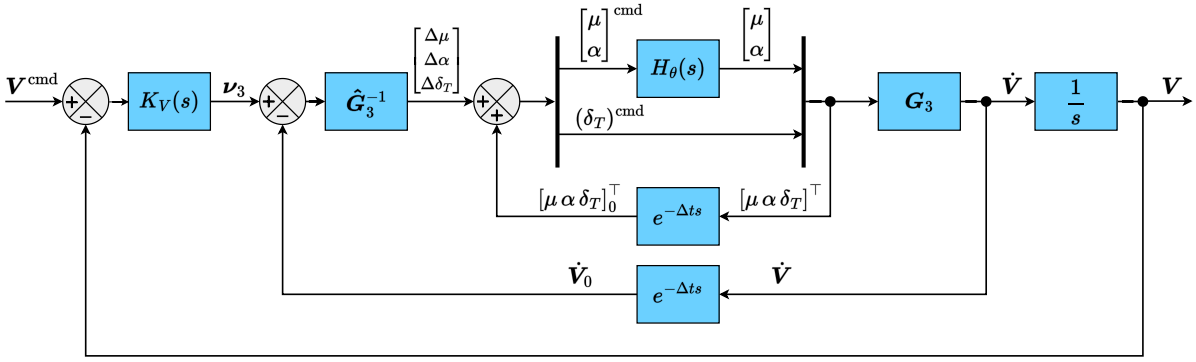


Figure 5.15: Closed-loop block diagram of the velocity loop, assuming perfect attitude and angular rate tracking.

For the linear controller design, the closed–loop expression plays a central role. Under the assumption that the estimated control effectiveness matrix $\hat{\boldsymbol{G}}_3$ is exact and that disturbances are rejected sufficiently fast, the closed–loop relation is given by:

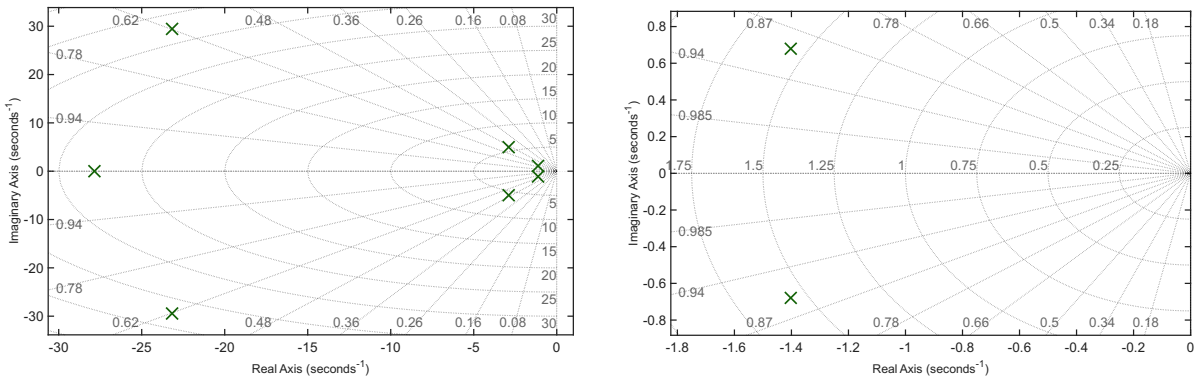
$$\boldsymbol{V}(s) = \frac{H_\theta(s)K_V(s)}{s + H_\theta(s)K_V(s)}\,\boldsymbol{V}^{\text{cmd}}(s), \qquad \lim_{H_\theta(s)\to 1} H_V(s) = \frac{K_V(s)}{s + K_V(s)}. \tag{5.77}$$

This relation applies to the angle-of-attack and bank-angle channels, where the inversion-based commands pass through inner dynamics represented by $H_\theta(s)$. The throttle channel does not include this term, since it acts directly on the propulsion system. When the inner loops are sufficiently fast such that $H_\theta(s) \approx 1$, the closed-loop map reduces to the expression shown on the right.

The velocity controller is obtained using the same second–order pole–matching approach applied in the attitude-loop design together with the bandwidth-separation and damping requirements, resulting in the closed-loop relation:

$$H_V(s) = \frac{2.428}{s^2 + 2.181s + 2.428}, \quad \text{using} \quad K_V(s) = \frac{2.428}{s + 2.181}. \tag{5.78}$$

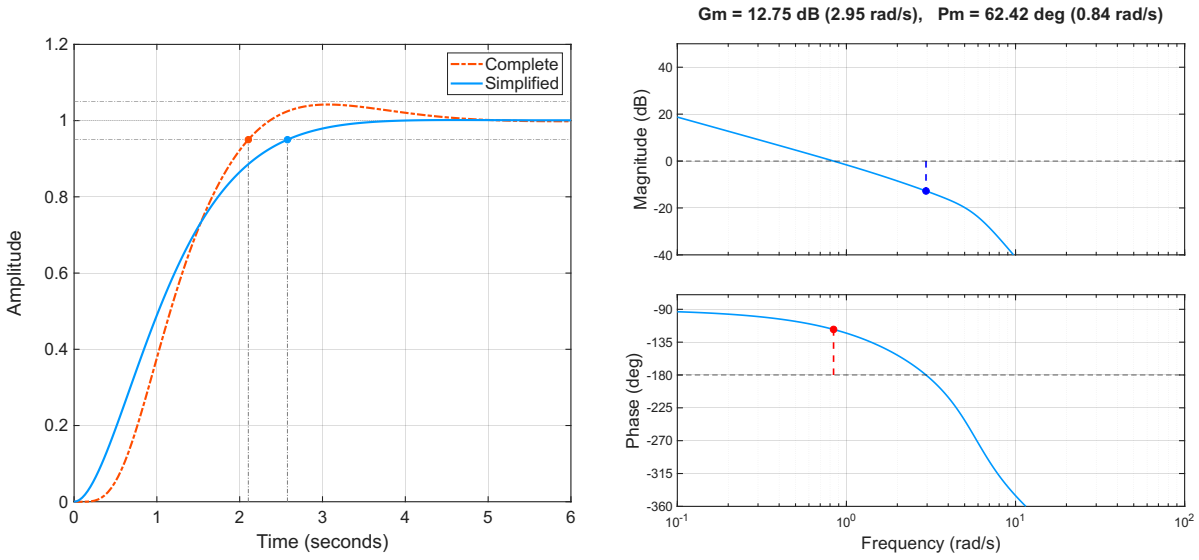
The closed-loop pole locations of the reduced and complete transfer functions are illustrated in Fig. 5.16. The figure compares the closed-loop dynamics obtained with the full controller structure, including the inner-loop dynamics, against the reduced formulation that assumes ideal instantaneous inner loops.



(a) (b)

Figure 5.16: Closed-loop stability characteristics of the velocity controller. Subfig. (a) shows the closed-loop poles including the inner-loop dynamics, and Subfig. (b) shows the poles assuming ideal inner loops.

The effect of these differences on system performance is further illustrated in Fig. 5.17. Specifically, Subfig. 5.17a compares the step responses of the ideal and non-ideal closed-loop systems. The inclusion of inner-loop dynamics introduces slight coupling effects that reduce the overall damping; however, the system remains stable and converges to the commanded velocity. Subfigure 5.17b shows the open-loop gain and phase margins of the complete system, which confirm adequate stability margins.



(a) (b)

Figure 5.17: Closed–loop performance and open–loop stability characteristics of the full and reduced velocity control loops. Subfig. (a) compares the closed–loop step responses, while Subfig. (b) shows the open–loop gain and phase margins of the full system.

Position Loop

The position control loop determines the required heading χ_K and flight-path angle γ_K commands to track the commanded waypoints in latitude, longitude, and altitude. As shown in Fig. 5.18, these waypoint commands originate from the onboard guidance system, and the resulting command error is processed by the linear controller $K_X(s)$ to produce the desired changes in heading and flight-path angle. In the figure, the position state vector $\boldsymbol{X}$ corresponds to $\boldsymbol{x}_4$. The errors in the commanded derivatives are then mapped through the inverse of the matrix $\boldsymbol{G}_4$ to account for spherical Earth effects, yielding intermediate commanded inputs. These are used in the inverse of Eq. (5.56) to generate the corresponding commands passed to the inner loops, which realize the actual heading and flight-path angle. The resulting control inputs are subsequently remapped through $\boldsymbol{G}_4$ to obtain the geodetic rate derivatives, which are integrated to produce the achieved tracking vector.

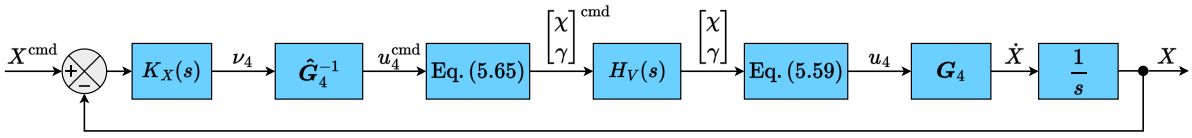


Figure 5.18: Closed loop block diagram of the position control loop [Stevens et al., 2016].

Analogous to the attitude loop, and assuming the estimated mapping $\hat{\boldsymbol{G}}_4$ is exact, the position closed–loop transfer function is given by the expression on the left below. This formulation incorporates the inner velocity dynamics through $H_V(s)$. When the velocity loop is sufficiently fast such that

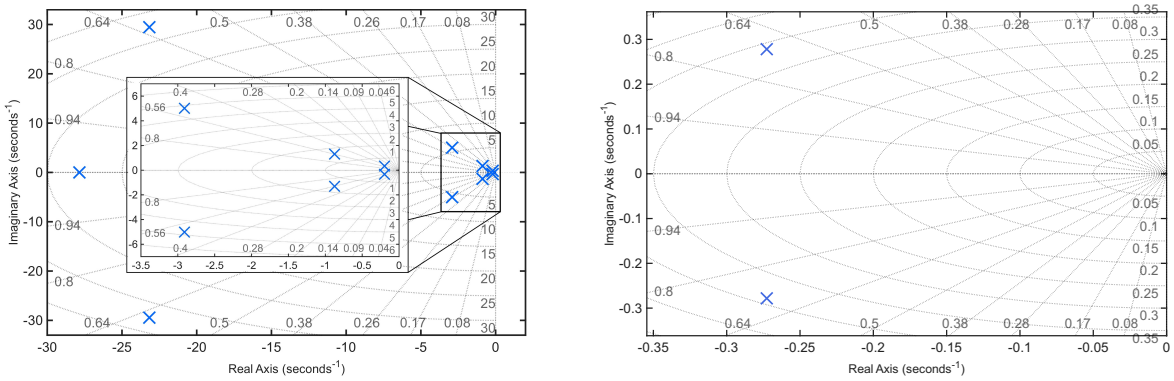
$H_V(s) \approx 1$, the closed–loop map reduces to the simplified first–order form shown on the right:

$$\boldsymbol{X}(s) = \frac{H_V(s)\,K_X(s)}{s + H_V(s)\,K_X(s)}\,\boldsymbol{X}^{\text{cmd}}(s), \qquad \lim_{H_V(s)\to 1} H_X(s) = \frac{K_X(s)}{s + K_X(s)}. \tag{5.79}$$

The position control loop follows the same design principles as the inner loops. The loop bandwidth is selected to preserve the hierarchical time–scale structure. It uses the same controller as designed in Eq. (5.72) which yields the second order closed-loop system:

$$H_X(s) = \frac{0.1517}{s^2 + 0.7012s + 0.1517}, \quad \text{using} \quad K_X(s) = \frac{0.1517}{s + 0.7012}. \tag{5.80}$$

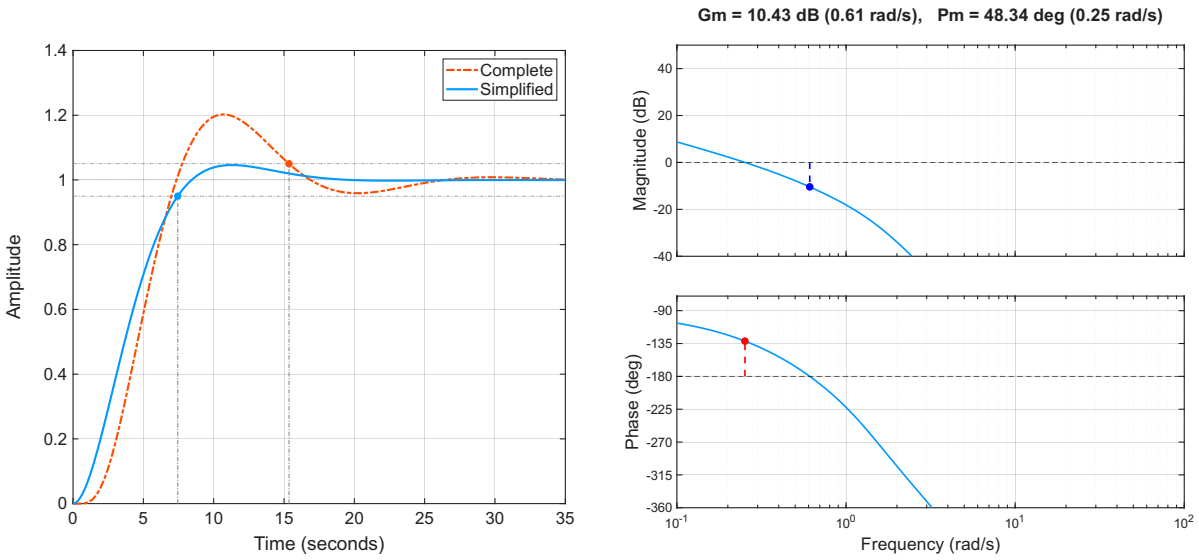
The pole locations of the reduced transfer function and the complete transfer function, which includes the inner control loops, are illustrated in Fig. 5.19.



(a) (b)

Figure 5.19: Closed–loop stability characteristics of the position controller. Subfig. (a) shows the closed-loop poles including the inner-loop dynamics, and Subfig. (b) shows the poles assuming ideal inner loops.

The effect of these differences on system performance is further illustrated in Fig. 5.20. Specifically, Subfig. 5.20a compares the step responses of the ideal and non-ideal closed-loop systems.



(a) (b)

Figure 5.20: Closed–loop performance and open–loop stability characteristics of the full and reduced position control loops. Subfig. (a) compares the closed–loop step responses, while Subfig. (b) shows the open–loop gain and phase margins of the full system.

The inclusion of inner-loop dynamics introduces slight coupling effects that reduce the overall damping; however, the system remains stable and converges to the commanded velocity. Subfigure 5.20b shows the open-loop gain and phase margins of the complete system, which confirm adequate stability margins. Figure 5.21 shows the open-loop Bode plots of all four control loops. The figure clearly illustrates the intended bandwidth separation between the rate, attitude, velocity, and position loops.

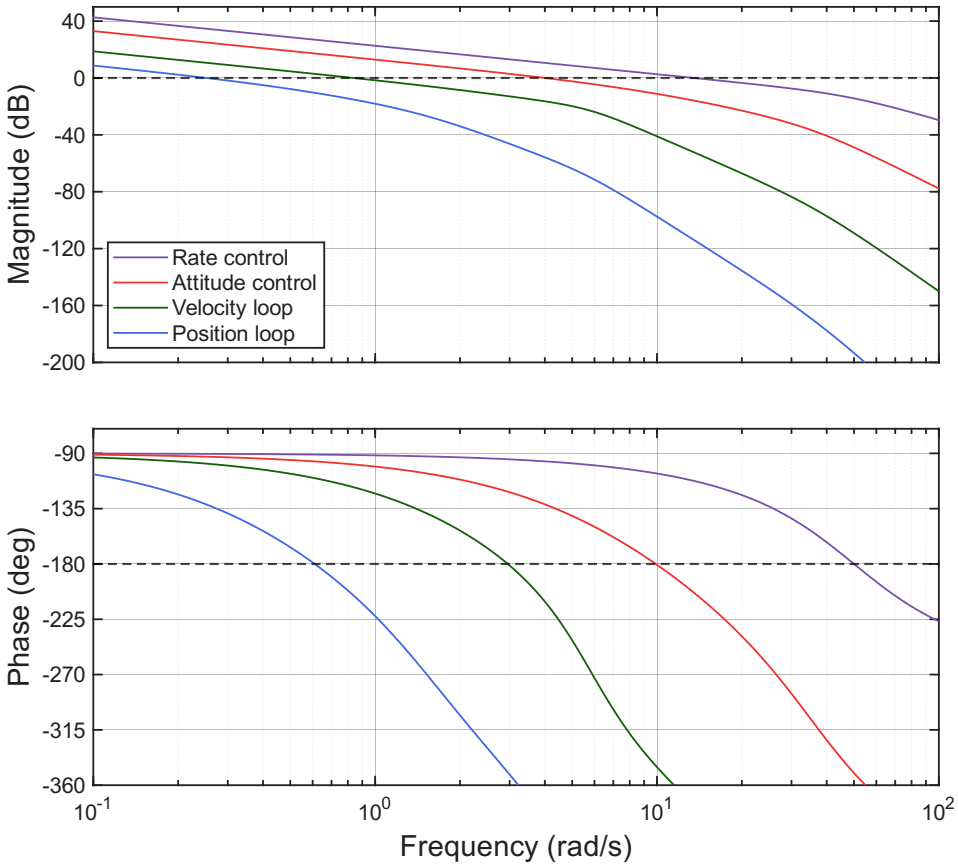


Figure 5.21: Bode plots of all control loops showing bandwidth separation across the cascaded NDI–INDI architecture.

Table 5.3 summarizes the classical stability margins of the ideal control loops, including the gain margin, phase margin, delay margin, and crossover frequency for each loop. The results show that all control loops maintain sufficient stability margins. As expected, the rate loop has the highest crossover frequency and smallest delay margin, while the outer loops exhibit lower bandwidths and larger delay margins due to the desired time-scale separation.

Table 5.3: Stability margins summary for the ideal control loops

	Classical			Disk	
Loop	CGM, dB	CPM, deg	CDM, s	DGM, dB	DPM, deg
Rate	14.35	67.55	0.087	10.0	54.85
Attitude	11.05	48.45	0.211	6.70	40.35
Velocity	10.75	51.40	0.871	7.13	42.50
Position	11.15	51.60	3.464	7.20	42.85

5.4. Digital Control Design

So far, the control design has been carried out in continuous time (CT). In practice, however, FCSs are implemented on a digital flight control computer operating in discrete time (DT). A common approach is therefore to first design the controller in CT and then convert it to a DT implementation using methods such as the bilinear transform (BLT) or the matched pole-zero method. However, this approach neglects the inherent properties of digital implementation. The controller interacts with sampled signals, Zero-Order Hold (ZOH) devices, computation delays, and on-board filtering, none of which are captured in a purely CT design [Stevens et al., 2016]. Only for very small sampling intervals Δt does a directly converted DT controller reproduce the performance of its CT counterpart.

A more sophisticated approach is the modified CT controller design for discretization. Here, the DT effects of sampling, hold operations, computation delay, and aliasing are modeled explicitly as CT transfer functions and incorporated directly into the synthesis. The controller is then designed with these effects included, such that the DT effects are accounted for from the start. Only then is the BLT method applied to obtain the digital controller for implementation. This approach ensures that the implemented digital controller preserves the intended performance characteristics, even for practical sampling rates. The relevant digital effects are modeled as follows:

1. **Anti-aliasing filter:** Before sampling, all sensor signals must be band-limited to avoid aliasing. Aliasing occurs when frequency components above the Nyquist frequency are folded back into the sampled spectrum, corrupting the measured signal with low-frequency artifacts that cannot be distinguished from the true dynamics [Stevens et al., 2016]. To prevent this, an analog low-pass filter is applied to the sensor signals prior to sampling. The filter has already been defined in Eq. (4.76), but for completeness it is repeated here with cutoff frequency $\omega_{AA} = \omega_{\text{Nyq}}/2 = 157.08$ rad/s:

$$G_{AA}(s) = \frac{\omega_{AA}}{s + \omega_{AA}}, \quad \angle G_{AA}(j\omega) = -\tan^{-1}\left(\frac{\omega}{\omega_{AA}}\right). \tag{5.81}$$

   This anti-aliasing filter introduces its own phase lag and attenuation, which must be included in the plant model $G(s)$ for realistic analysis and ideally the controller must be designed against it.

2. **Sampler and ZOH.** After sampling, the control input is only updated at DT intervals of Δt. Between these updates, the actuator input is held constant by a ZOH. This piecewise-constant behavior attenuates high-frequency components and introduces additional phase lag, which can be approximated as an effective delay of about $\Delta t/2$. The combined effect of the sampler and ZOH can be represented in CT by the transfer function and frequency response [Stevens et al., 2016]:

$$G_{\text{ZOH}}(s) = \frac{1 - e^{-s\Delta t}}{s\,\Delta t}, \quad \angle G_{\text{ZOH}}(j\omega) = -\frac{\omega\Delta t}{2}. \tag{5.82}$$

3. **Computation delay.** The flight control computer requires time to process the sampled measurements and compute the actuator command. This introduces a delay that depends on the processor load and scheduling. Although the actual delay is typically smaller than one sampling period, a conservative worst-case assumption is to model it as a full-sample delay [Stevens et al., 2016]. In this case, the control command is delayed by a full sampling period Δt, which in CT can be represented by the transfer function and frequency response:

$$G_{CD}(s) = e^{-s\Delta t}, \quad \angle G_{CD}(j\omega) = -\omega\Delta t. \tag{5.83}$$

In addition to these three modeled effects, real sensors may introduce additional sensor delays $G_{SD}(s)$, as discussed in Eq. (4.77). Since such delays are generally unknown during the design stage, they are not included in the modified CT controller synthesis.

The overall control structure, shown in Fig. 5.22, includes the key effects introduced by digital implementation. The controller $K_c(s)$ operates together with the sampler and ZOH element $G_{ZOH}(s)$, computation delay $G_{CD}(s)$, and sensor anti-aliasing filter $G_{AA}(s)$. Together with the plant dynamics $G(s)$, these components form a continuous-time representation of how the digital controller interacts with the aircraft.

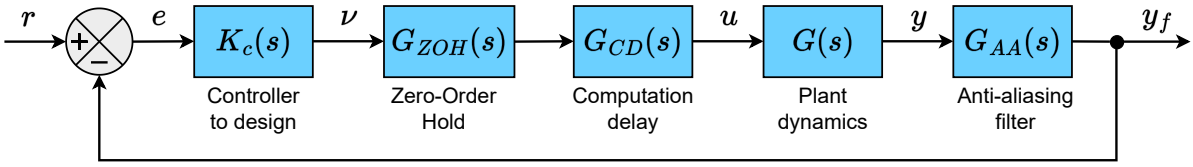


Figure 5.22: Continuous-time control structure with digital implementation effects [Stevens et al., 2016].

The transfer functions $G_{ZOH}(s)$ and $G_{CD}(s)$ contain exponential terms, making them irrational functions and therefore unsuitable for controller synthesis. To obtain rational transfer functions, these exponentials are approximated using Padé approximations. A Padé approximation is a rational function whose Taylor series expansion matches that of the original function up to a specified order [Stevens et al., 2016]. This makes Padé approximations well suited for delay modeling, since they capture both the initial slope and the phase lag using relatively low-order rational functions. The third-order Padé forms used here follow the formulations presented in [Stevens et al., 2016]. For the sampler and ZOH transfer function, the third-order Padé approximation is:

$$G_{ZOH}(s) \approx \frac{1 - (s\Delta t)/14 + 23(s\Delta t)^2/840 - (s\Delta t)^3/840}{1 + 3(s\Delta t)/7 + (s\Delta t)^2/14 + (s\Delta t)^3/120}. \tag{5.84}$$

For the case where the computation delay is equal to one sample time, the third-order Padé approximation of the computation delay transfer function becomes:

$$G_{CD}(s) \approx \frac{1 - (s\Delta t)/4}{1 + 3(s\Delta t)/4 + (s\Delta t)^2/4 + (s\Delta t)^3/24}. \tag{5.85}$$

A representative block diagram illustrating how the system is implemented in Simulink, including the digital effects (shown in purple) and a sensor delay for testing, is presented in Fig. 5.23[11]. Continuous-time blocks are shown in blue, while discrete-time blocks are shown in orange.[12]

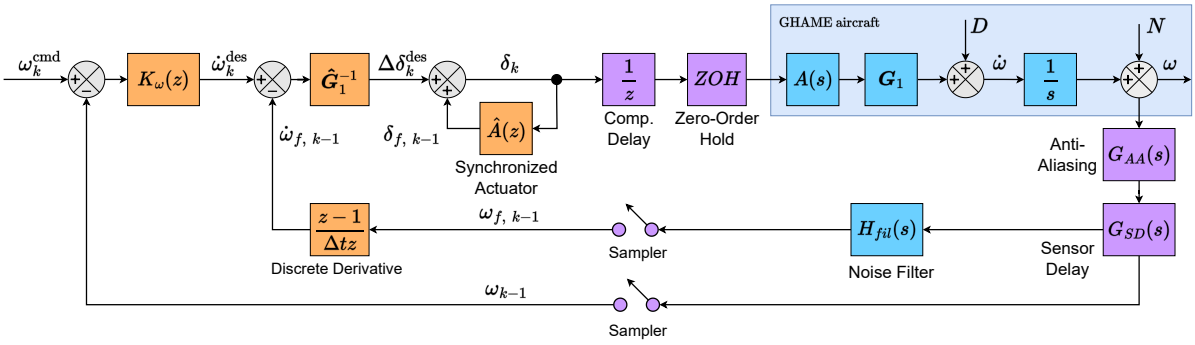


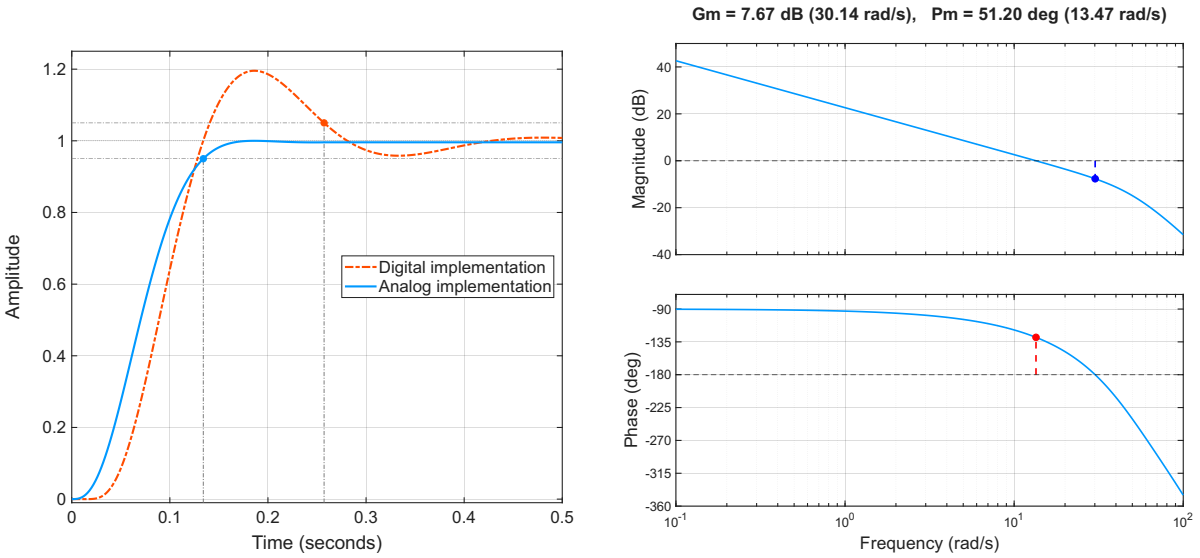
Figure 5.23: Discrete-time INDI rate control loop including actuator dynamics, sensor dynamics, sampling, and noise filtering.

[11] In Simulink, the computation delay is modeled using a unit delay block and is therefore represented as $1/z$. The ZOH operation is handled automatically by Simulink and does not require an explicit block implementation. The ZOH block is nevertheless shown to indicate the effect of the hold operation within the digital control loop.

[12] Although G_1 and G_1^{-1} are technically constant matrix gains, they are colored for consistency with the surrounding block structure.

The digital effects on the rate control loop are now considered. Since digital effects become more significant for faster dynamics, their influence is most pronounced in the innermost rate loop. The rate loop therefore requires particular attention. When the control structure from Fig. 5.22 is applied to the angular rate loop described in Sec. 5.3, the additional digital dynamics invalidate the previously calculated open-loop gain and phase margins.

The crossover frequencies used for evaluating these effects are given in Table 5.3. Using the frequency responses defined in Eqs. (5.81), (5.82), and (5.83), the additional phase lag can be evaluated at the phase crossover frequency of 13.5 rad/s. For a sampling period of $\Delta t = 0.01$ s, the computation delay contributes approximately $-7.7°$, the anti-aliasing filter contributes approximately $-4.9°$, and the ZOH contributes approximately $-3.9°$. Together, these effects introduce about $-16.5°$ of additional phase lag. As a result, the original continuous-time phase margin of $67.6°$ is reduced to approximately $51.1°$ in the digital implementation. Since the phase response is shifted downward, the $-180°$ crossover frequency also shifts to lower frequencies, thereby reducing the gain margin. This reduction is more significant than for the phase margin and is illustrated in the open-loop Bode plot of Fig. 5.24.



(a) (b)

Figure 5.24: Closed–loop performance and open–loop stability characteristics of the digital and analog rate control loops. Subfig. (a) compares the closed–loop step responses, while Subfig. (b) shows the open–loop gain and phase margins of the rate control loop with digital implementation effects included.

As shown in Fig. 5.24b, the classical stability margins are reduced to a gain margin of 7.67 dB and a phase margin of $51.20°$ at 13.47 rad/s, giving a delay margin of 0.066 s.Furthermore, the closed-loop step response in Fig. 5.24a exhibits an overshoot of 19.5% and a settling time of 0.257 s, meaning that the original zero-overshoot requirement is no longer satisfied. Although the reduction in stability margins is not critical, the increase in overshoot significantly changes the system behavior and should therefore be considered during controller design.

The influence of digital effects diminishes as the system dynamics become slower and is thus significant only in the rate and attitude control loops. The gain and phase margins, with a comparison between the ideal continuous-time and digital implementations per loop, are summarized in Table 5.4.

Table 5.4: Comparison of stability margins for each control loop without and with digital effects (D)

	Classical			Disk	
Loop	CGM, dB	CPM, deg	CDM, s	DGM, dB	DPM, deg
Rate	14.34	67.57	0.087	9.98	54.83
Rate (D)	7.67	51.20	0.066	6.10	37.3
Attitude	11.05	48.45	0.211	6.70	40.35
Attitude (D)	7.60	43.40	0.301	5.43	33.70
Velocity	10.75	51.40	0.871	7.13	42.50
Velocity (D)	10.50	51.30	0.816	7.08	42.30
Position	11.15	51.60	3.464	7.20	42.85
Position (D)	11.15	51.60	5.19	7.20	42.85

The table illustrates why the modified CT analysis is valuable. With the original proportional gain from Eq. (5.65), two design paths can now be considered. The first option is to accept the reduced stability margins and the small overshoot as sufficient for operation, since the system remains stable with acceptable stability margins. The second option is to redesign the linear controllers with the digital effects explicitly included, thereby recovering the intended performance targets. This approach is adopted here to restore the zero-overshoot behavior of the rate loop and to improve the limited delay margins observed under digital implementation.

The resulting redesigned linear controllers for the rate, attitude, velocity, and position loops are given by:

$$K_\omega = 7.97, \quad K_\theta(s) = \frac{13.96}{s+6.73}, \quad K_V(s) = \frac{0.87}{s+1.31}, \quad \text{and} \quad K_X(s) = \frac{0.055}{s+0.42}. \tag{5.86}$$

The corresponding stability margins obtained with these redesigned controllers are summarized in Table 5.5.

Table 5.5: Stability margins obtained with the redesigned controllers from Eq. (5.86), based on the modified continuous–time control design

	Classical			Disk	
Loop	CGM, dB	CPM, deg	CDM, s	DGM, dB	DPM, deg
Rate	12.3	67.3	0.15	9.91	54.55
Attitude	10.8	48.1	0.35	6.59	39.80
Velocity	10.7	51.4	1.45	7.12	42.45
Position	11.2	51.6	6.00	7.20	42.85

To summarize the key characteristics of the developed cascaded controller, the control architecture consists of four nested loops: rate, attitude, velocity, and position. Each outer loop is tuned to operate at a lower bandwidth than the one inside it, ensuring that the inner dynamics are already attenuated within its operating range. This arrangement minimizes coupling between loops and simplifies both tuning and analysis. The corresponding loop transfer functions are shown in Fig. 5.25. The magnitude plot illustrates clear bandwidth separation among the loops. For clarity, reference lines at 0 dB and -180° are added to visualize crossover frequencies and phase margins. All results presented here use the final tuned gains and filters listed in Table 5.6.

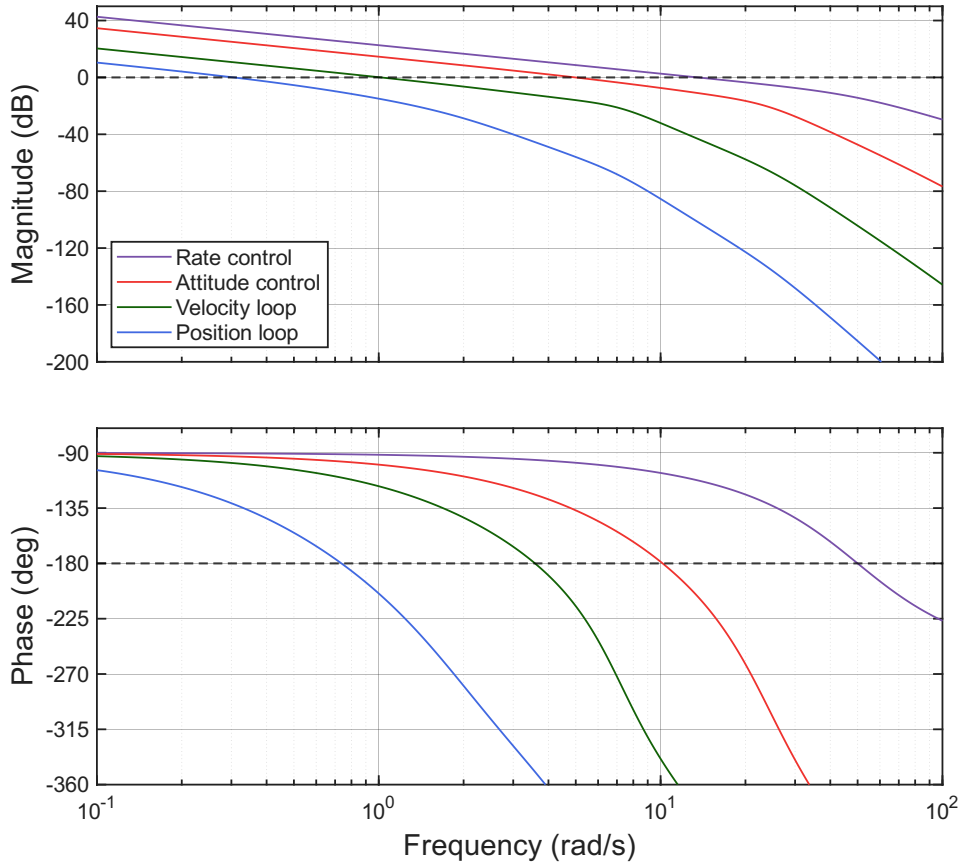


Figure 5.25: Open loop Bode magnitude and phase for the rate, attitude, velocity, and position loops with the redesigned controllers.

Table 5.6: Design parameters used for control law development using the modified continuous control design

Parameter	Symbol	Value	Unit	Parameter	Symbol	Value	Unit
Separation factor	k	4	-	Velocity bandwidth	BW_V	1.60	rad/s
Filter corner freq.	ω_H	25	rad/s	Velocity nat. freq.	ω_V	1.56	rad/s
Filter damp. ratio	ζ_H	1	-	Velocity des. damp.	ζ_V	0.9	-
Actuator nat. freq.	ω_{act}	50.0	rad/s	Position bandwidth	BW_X	0.49	rad/s
Actuator damp. ratio	ζ_{act}	0.707	-	Position nat. freq.	ω_X	0.39	rad/s
Rate bandwidth	BW_ω	24.93	rad/s	Position des. damp.	ζ_X	0.7	-
Attitude bandwidth	BW_θ	7.58	rad/s	Anti-aliasing freq.	ω_{AA}	157.07	rad/s
Attitude nat. freq.	ω_θ	6.23	rad/s	Comp. delay	t_d	0.01	s
Attitude des. damp.	ζ_θ	0.7	-	Sampling time	Δt	0.01	s

6

Nonlinear Simulation Results

In the previous chapter, the controllers for the FCS were designed. This chapter focuses on their implementation in a digital environment using Simulink. Once implemented, the controllers are integrated with the nonlinear simulation model to evaluate overall control system performance through a set of well-defined test cases. The digital implementation, including the discretization of the controllers and their integration within the FCS, is presented in Sec. 6.1, while the resulting non-linear simulations are presented in Sec 6.2.

6.1. Digital Implementation

Digital implementation is required because the FCS developed in Chapter 6 and the hierarchical linear controllers described in Eq. (5.86) are formulated in continuous time. In practice, however, a flight control computer operates discretely with a finite sampling time, meaning that the FCS implemented within it must also be discrete. These discrete effects influence system performance and motivate the digital control design presented in Sec. 5.4. The next logical step is therefore to discretize all continuous-time elements of the FCS that directly affect the control law implementation. In particular, the continuous derivative blocks and the hierarchical INDI controllers must be expressed in discrete form to ensure proper operation on the flight control computer. The aircraft dynamics, actuator models, sensor dynamics, and noise filters, on the other hand, operate in continuous time. As a result, all input signals to the flight control computer must be sampled before use by the discrete controllers. Likewise, the outputs of the flight control computer must interface with continuous subsystems. In theory, this would require converting the discrete outputs back to continuous signals through a ZOH with a sample time equal to that of the computer. In practice, Simulink manages rate transitions between discrete and continuous domains automatically, eliminating the need for explicit conversion blocks.

The final layout of the hierarchical control system is shown in Fig. 6.1. The diagram presents all five main components: the sampling of input signals into the flight control computer, followed by the four control loops described previously. This configuration represents the discrete-time Simulink implementation of the architecture shown in Fig. 5.1.

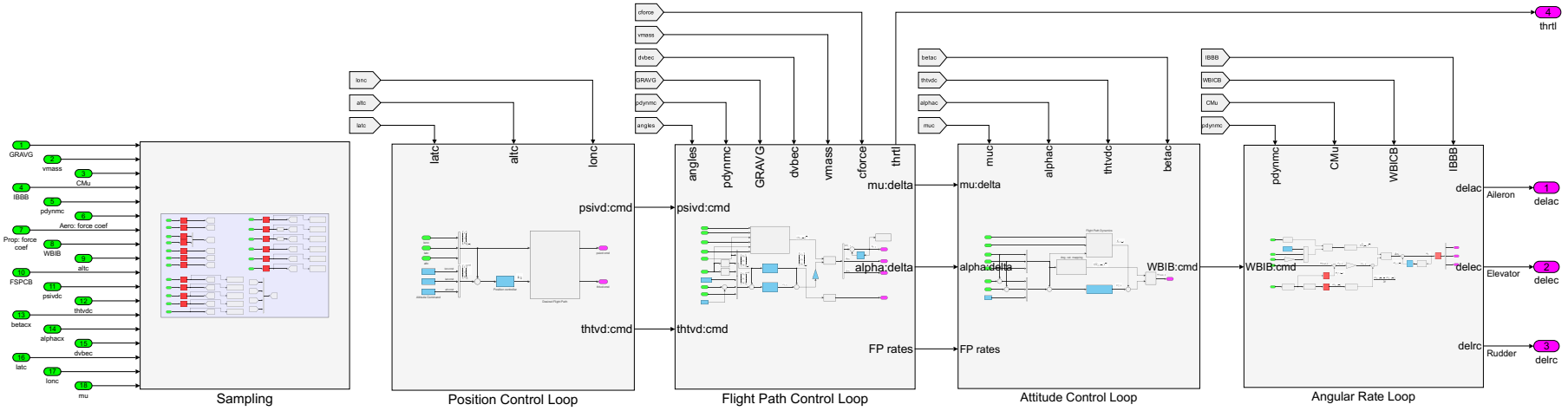


Figure 6.1: Discrete-time hierarchical INDI control architecture implemented in Simulink, including the sampling subsystem.

All controllers are executed at a baseline sampling frequency of 100 Hz. Simulations are performed using a fixed-step solver to capture discretization effects. In addition to the defined sampling times for each discrete element, the solver step size is set to $\Delta t/25$, corresponding to 4×10^{-4} s for a computer sampling period of $\Delta t = 1/100$ s.

The sampling subsystem, illustrated in Fig. 6.2, contains a set of samplers that process nearly all variables obtained from the INS. For convenience, all sampled signals are routed through `Goto` blocks, which simplifies their use in the downstream control subsystems, as shown in Fig. 6.1. The angular rates, however, are not sampled within this subsystem because they must first pass through the sensor dynamics in the rate control loop before sampling. Consequently, dedicated samplers for the angular rate signals are included within the rate control loop itself.

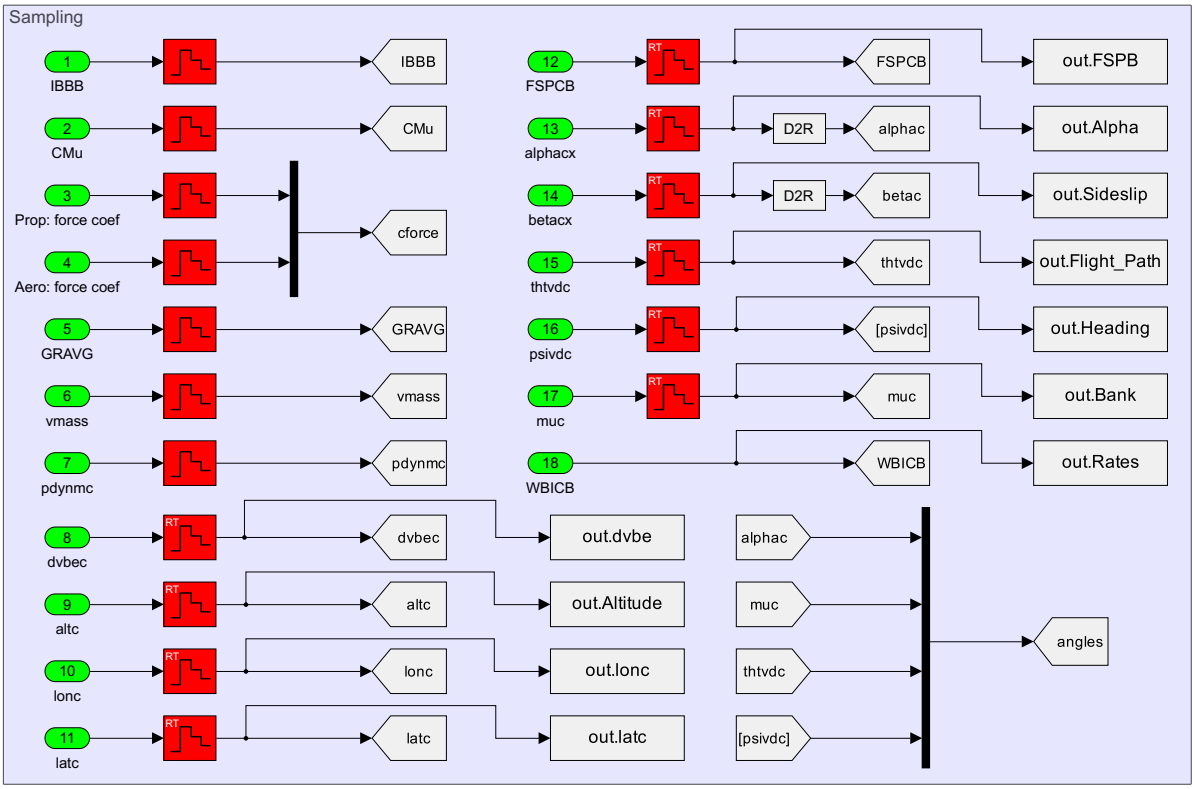


Figure 6.2: Structure of the sampling subsystem within the control system.

Position Control

The Simulink representation of the position control loop is shown in Fig. 6.3. The desired flight path function implements the equations in Eq. (5.60) and Eq. (5.61). The green inputs represent sampled variables from the sampling subsystem, while the blue blocks correspond to the commanded signals and the discrete position controller. The blue color indicates that these elements operate at a defined sampling time. The pink blocks denote the outputs, which are the commanded flight path and heading signals passed to the next control loop.

The position controller is the discretized version of the continuous-time controller defined in Eq. (5.86). The discretization is performed using the bilinear (Tustin) transformation, which maps the continuous s-domain to the discrete z-domain according to:

$$s = \frac{2}{\Delta t} \frac{1 - z^{-1}}{1 + z^{-1}}, \tag{6.1}$$

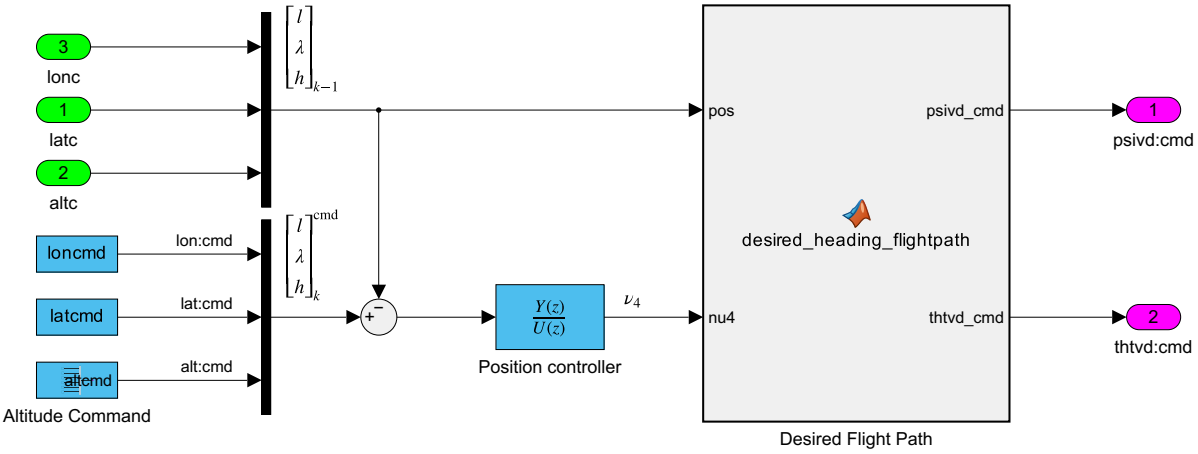


Figure 6.3: Simulink implementation of the discrete-time position control loop, showing sampled inputs, discrete controller, and commanded flight path outputs.

where Δt is the sampling period. This transformation maintains the stability properties of the continuous system while providing a consistent mapping of its low-frequency dynamics to the discrete domain. Applying it to the continuous controller in Eq. (5.86) with $\Delta t = 0.01$ s yields the following discrete-time transfer function:

$$K_X(z) = \frac{4.0319 \times 10^{-5} + 4.0319 \times 10^{-5} z^{-1}}{1 - 0.99838\, z^{-1}}. \tag{6.2}$$

The resulting discrete controller preserves the dominant low-frequency dynamics of the continuous design, ensuring similar closed-loop behavior within the controller's operating bandwidth. Deviations are primarily expected at higher frequencies due to frequency warping and the finite sampling period.

Velocity and Flight Path Control

The velocity control loop is shown in Fig. 6.5. The `compute_inv_G3` function determines the inverse of Eq. (5.50), which is required for the control allocation within this loop. It is assumed that the throttle actuator can achieve the commanded change within a single sampling period. Under this assumption, a Unit Delay (UD) block provides a sufficient approximation to generate the incremental input required for speed control. A potential extension of this work would be to investigate how incorporating a throttle delay model influences the performance of the velocity control loop.

The remaining incremental inputs, corresponding to the commanded bank angle and angle of attack, are forwarded to the attitude control loop, where they are combined with the corresponding values from the previous sampling instant. The computed derivatives are also passed to the attitude control loop for its control law. The speed command is provided through a constant block that defines the trimmed velocity, while a step block is used to introduce heading changes when the hold–heading mode is enabled in the simulation.

A key feature of this loop is that the derivatives required for the INDI control law are computed using the backward difference numerical differentiation method, as shown in Fig. 6.4.

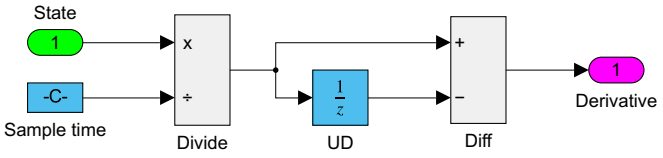


Figure 6.4: Discrete differentiation algorithm used for computing derivatives in the velocity control loop.

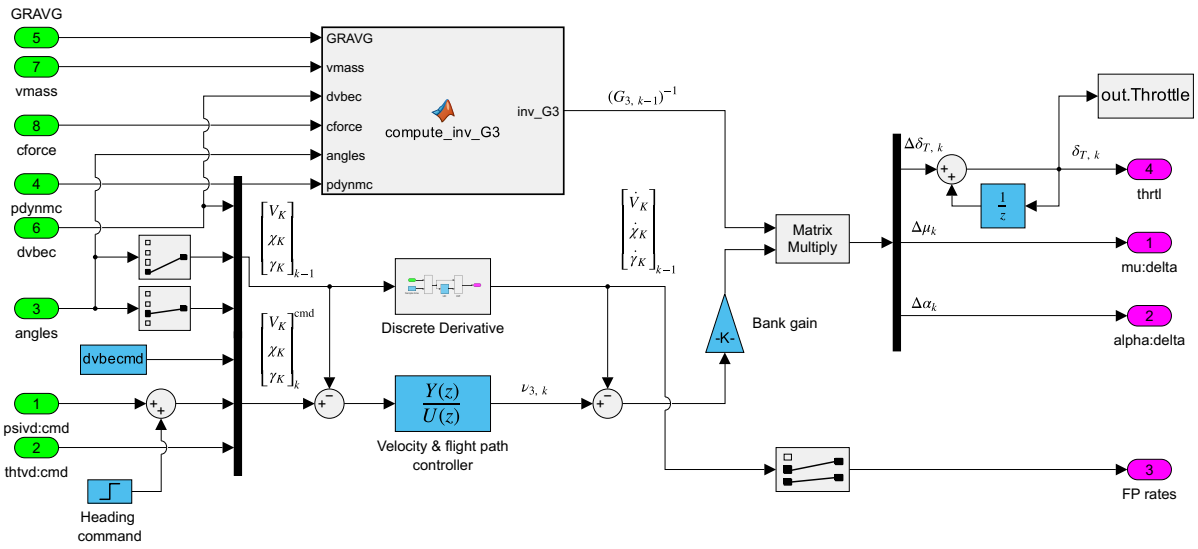


Figure 6.5: Simulink implementation of the velocity control loop with the discrete controller $K_V(z)$.

Unlike the rate control loop, a second–order noise filter is not applied before differentiation. Strictly speaking, one would need to verify whether the measured airspeed, flight path angle, and heading signals are sufficiently noise-free to allow omission of a noise filter. In this thesis, the INS model includes errors only in the angular velocity measurements and does not account for tilt, velocity, or position errors. Consequently, the airspeed, flight path, and heading angles are assumed noise–free, and filtering is unnecessary.

The velocity controller is obtained by discretizing the continuous-time design in Eq. (5.86) using the Tustin (bilinear) transformation. With a sampling period of $\Delta t = 0.01$ s and normalized coefficients, the resulting discrete transfer function is:

$$K_V(z) = \frac{0.0003297 + 0.0006595\, z^{-1} + 0.0003297\, z^{-2}}{1 - 1.9220\, z^{-1} + 0.9249\, z^{-2}}. \tag{6.3}$$

The discretization approach and its implications are identical to those discussed for the position controller.

Attitude Control

The attitude control loop is implemented using NDI, which eliminates the need for discrete derivative approximations and thereby simplifies implementation in Simulink. The overall structure of the loop is shown in Fig. 6.6. At each integration step, the terms $\boldsymbol{G}_2$ and $\boldsymbol{f}_2$ are computed according to Eq. (5.38) to perform the dynamic inversion. The attitude command is formed by combining the incremental control inputs from the preceding velocity loop with the sampled states provided by the INS. The sideslip command is fixed at zero to maintain coordinated flight, and the resulting rate command is passed to the rate control loop.

The attitude controller is obtained by discretizing the continuous-time design in Eq. (5.86) using the Tustin (bilinear) transformation. With a sampling period of $\Delta t = 0.01$ s and normalized coefficients, the resulting discrete transfer function is:

$$K_\theta(z) = \frac{0.1271 + 0.1271\, z^{-1}}{1 - 0.9113\, z^{-1}}. \tag{6.4}$$

The discretization procedure follows the same approach as for the position and velocity controllers. The resulting discrete controller is implemented within the blue discrete transfer function block of the attitude control loop, as shown in Fig. 6.6.

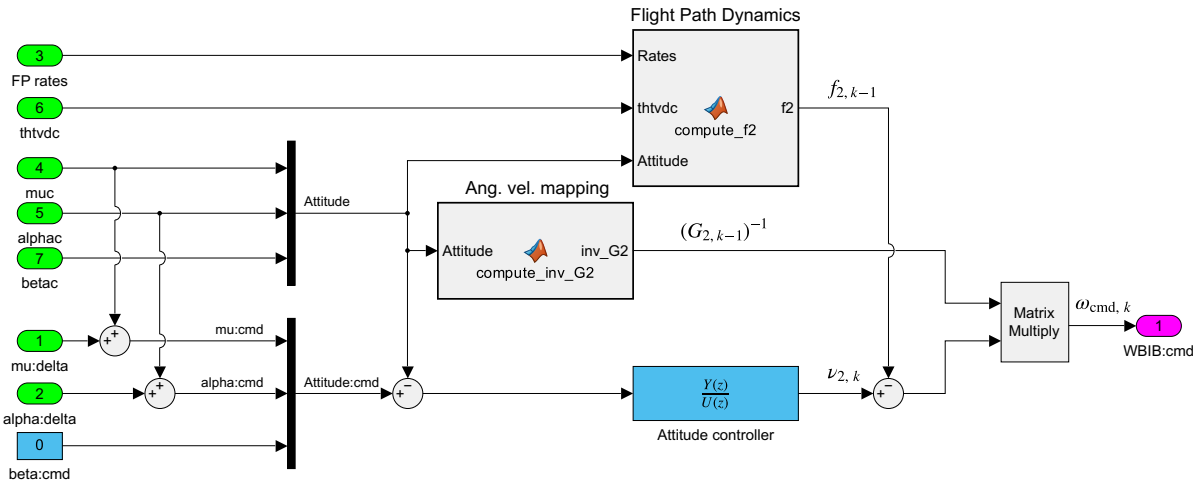


Figure 6.6: Simulink implementation of the attitude control loop. The discrete controller $K_\theta(z)$, obtained via Tustin discretization, is located in the linear control subsystem.

Rate Control:

The rate control loop is implemented using INDI and therefore requires discrete derivatives. Unlike the velocity control loop, noise filtering is applied in this case and sensor dynamics are present, which introduces the need for actuator synchronization. The discrete–time block diagram of the INDI rate controller, including sensor, noise, and actuator dynamics, is shown in Fig. 5.23 it is essentially the Simulink implementation of Fig. 5.23.

In the upper path, the control effectiveness matrix $\boldsymbol{G}_1$ is computed according to Eq. (5.9). The lower path contains the continuous implementation of the sensor dynamics, representing anti-aliasing effects plus an unknown delay. The INS state is subsequently processed through a continuous noise filter before being sampled to obtain the discrete derivatives, which are identical to those presented in Fig. 6.4. The angular rate state that is not differentiated bypasses the noise filter and is directly sampled. Another important aspect is the worst-case computational delay, which occurs when the flight control computer fails to compute the command within one sampling period, resulting in a one-step delay. In the simulation, this effect is modeled using a UD block placed immediately before the control signal is sent to the actuators. Although this delay is not required for the control law to function, it provides a more realistic representation of digital implementation effects. For the rate control loop, the controller reduces to a constant linear gain, which requires no translation into discrete time.

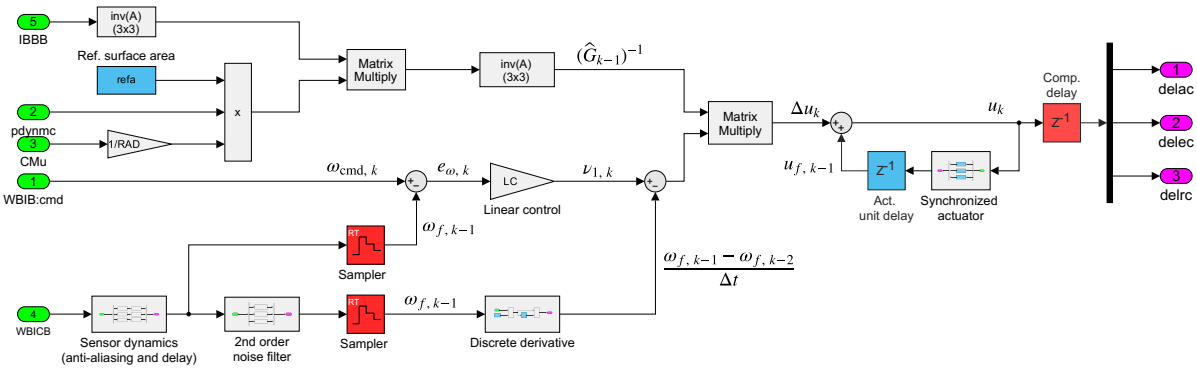


Figure 6.7: Simulink implementation of the INDI subsystem for angular rate control, including sensor, noise, and actuator dynamics.

The key aspect here is the discretization of the actuator path, which includes the synchronization transfer functions. The corresponding discrete formulation for implementation within the digital control loop is presented next.

Discrete Synchronized Actuators

The physical actuators operate in continuous time, while their implementation in the flight control computer is discrete. To ensure consistent timing between the actuation and measurement paths, the actuator dynamics are augmented with synchronization and filtering dynamics. Based on the sensor synchronization method defined in Eq. (3.73), the combined continuous-time representation is given by

$$\hat{A}(s) = A(s)\, H_{\text{sync}}(s)\, H_{\text{fil}}(s), \tag{6.5}$$

where $A(s)$ denotes the actuator dynamics and $H_{\text{fil}}(s)$ represents the second order low-pass filter designed in Section 5.2. The term $H_{\text{sync}}(s)$ captures additional synchronization dynamics that introduce phase or timing mismatches between the actuator and measurement paths. In this work, these effects arise from the anti-aliasing filter $G_a(s)$ and the sensor delay $G_{SD}(s)$. In the first set of simulations, $H_{\text{sync}}(s)$ includes only the anti-aliasing filter $G_a(s)$. For the fully synchronized case, the sensor delay $G_{SD}(s)$ is incorporated as well.

All components except the delay are combined in the continuous–time domain prior to discretization, because the transformation from the s–domain to the z–domain is nonlinear and must, in general, be applied to the complete transfer function [Nise, 2015]:

$$z\{S(s)G(s)\} \neq z\{S(s)\}\, z\{G(s)\}. \tag{6.6}$$

Here, $z\{\cdot\}$ denotes the z–transform, and discretization is performed using the ZOH method rather than the Tustin method, since it reproduces the sampled-data behavior of actuators with constant input during each sampling period. By separating the delay component from Eq. (6.5), the non-delayed portion of the system is defined as:

$$H_c(s) = A(s)\, H_{\text{sync}}(s)\, H_{\text{fil}}(s). \tag{6.7}$$

Equation (6.7) is now discretized in a single step using ZOH discretization with sampling period Δt:

$$\hat{A}(z) = z\left\{\frac{1 - e^{-s\Delta t}}{s} H_c(s)\right\}. \tag{6.8}$$

The resulting discretized synchronized actuator dynamics can be expressed as follows:

$$\hat{A}(z) = \frac{8.503 \cdot 10^{-5}\, z^{-1} + 1.46 \cdot 10^{-3}\, z^{-2} + 0.0024\, z^{-3} + 6.024 \cdot 10^{-4}\, z^{-4} + 1.43 \cdot 10^{-5}\, z^{-5}}{1 - 3.163\, z^{-1} + 3.935\, z^{-2} - 2.381\, z^{-3} + 0.682\, z^{-4} - 0.0687\, z^{-5}}. \tag{6.9}$$

The final step is the initialization of the discrete synchronized actuator. To avoid spurious transients at $t = 0$, the actuator must start at the calculated trim input; otherwise, a mismatch between the trim condition and the actuator state would occur. In a discrete-time transfer function block in Simulink, however, the trim deflection cannot be assigned directly as the initial output. The parameter *Initial states* instead specifies the contents of the internal delay registers of the filter. Consequently, the internal states must be computed such that the filter output equals the trim deflection u_{trim} at the start of the simulation.

Simulink realizes a discrete-time transfer function in state-space form as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\,\boldsymbol{u}_k, \\ y_k &= \boldsymbol{C}\boldsymbol{x}_k + D\,\boldsymbol{u}_k, \end{aligned} \tag{6.10}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ are the filter states, $\boldsymbol{u}_k$ is the vector of control inputs, and y_k is the output. The matrices $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, D)$ are obtained directly from the actuator numerator and denominator coefficients using the

MATLAB command `tf2ss`. At trim, the input is constant, $\boldsymbol{u}_k = \boldsymbol{u}_{\text{trim}}$. Requiring the state to remain constant, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, leads to the steady-state condition:

$$\boldsymbol{x}_0 = \left(\boldsymbol{I} - \boldsymbol{A}\right)^{-1} \boldsymbol{B}\, \boldsymbol{u}_{\text{trim}}, \tag{6.11}$$

with $\boldsymbol{I}$ the identity matrix of the same size as $\boldsymbol{A}$. Assigning the vector $\boldsymbol{x}_0$ to the actuator block as its initial state ensures that, at $t = 0$, the actuator outputs match the trimmed control deflections $\boldsymbol{u}_{\text{trim}}$. The corresponding initial states associated with the equilibrium control inputs listed in Table 4.4 are summarized in Table 6.1.

Table 6.1: Initial condition for discrete actuators

Condition	Initial state $\boldsymbol{x}_0$	δ_{trim}, deg	α_{trim}, deg	θ_{trim}, deg
Mach 3, 60,000 ft	[0, -12.21, 0]	[0.0, -5.1039, 0.0]	3.45	3.45

6.2. Simulation

The simulation integrates the formulated control law, the designed controller, and the improved, trimmed, and verified nonlinear model to create a representative test environment for performance evaluation. The objective is to assess the effectiveness of the INDI controller under realistic operating conditions, including horizontal wind, atmospheric turbulence, IMU noise, sensor delays, multiplicative parameter uncertainties, and actuator perturbations.

The flight conditions used for testing, together with the corresponding performance specifications, are listed in Table 5.1. The specific parameters of the altitude-change maneuvers are given in Table 5.2. At the start of each simulation, the aircraft is trimmed at zero longitude and zero latitude, and initially commanded to hold attitude and airspeed while maintaining a zero heading (northward flight). Depending on the test case, the aircraft then performs altitude or heading step commands. Throughout these simulations, the aircraft is subjected to several environmental and system-level effects that must be properly accounted for.

The controller and the model are tested according to the test cases specified in the table reproduced here for convenience. The model is first trimmed for level flight, as described in the trim section, using the values provided in Table 4.4. Target heading and altitude commands are then applied, and performance is evaluated based on whether these targets are achieved within the desired tolerances. Results are first presented for the nominal case, after which the tests are repeated with the inclusion of real-world effects to examine the controller's robustness. The trimmed airspeed is also monitored to ensure that it remains stable and does not deviate from its reference value.

Altitude Change Maneuvers

The first case tests the vehicle response during altitude climb and sink maneuvers. The test parameters are summarized below.

Free-stream Mach number: $M_\infty = 3$ Altitude: $h = 60{,}000$ ft Vertical speed: ±5,000 fpm
Altitude change: ±5,000 ft

The altitude reference trajectory is generated from the vertical speed requirement. A constant climb or sink rate of ±5,000 fpm (≈ ±25.4 m/s) produces the black dashed reference line. The desired and adequate performance bounds prescribed in Table 5.1 correspond to altitude deviations of ±200 ft and ±400 ft, respectively, and are shown as green and magenta shaded regions in Fig. 6.8. The red line represents the nominal response without uncertainty.

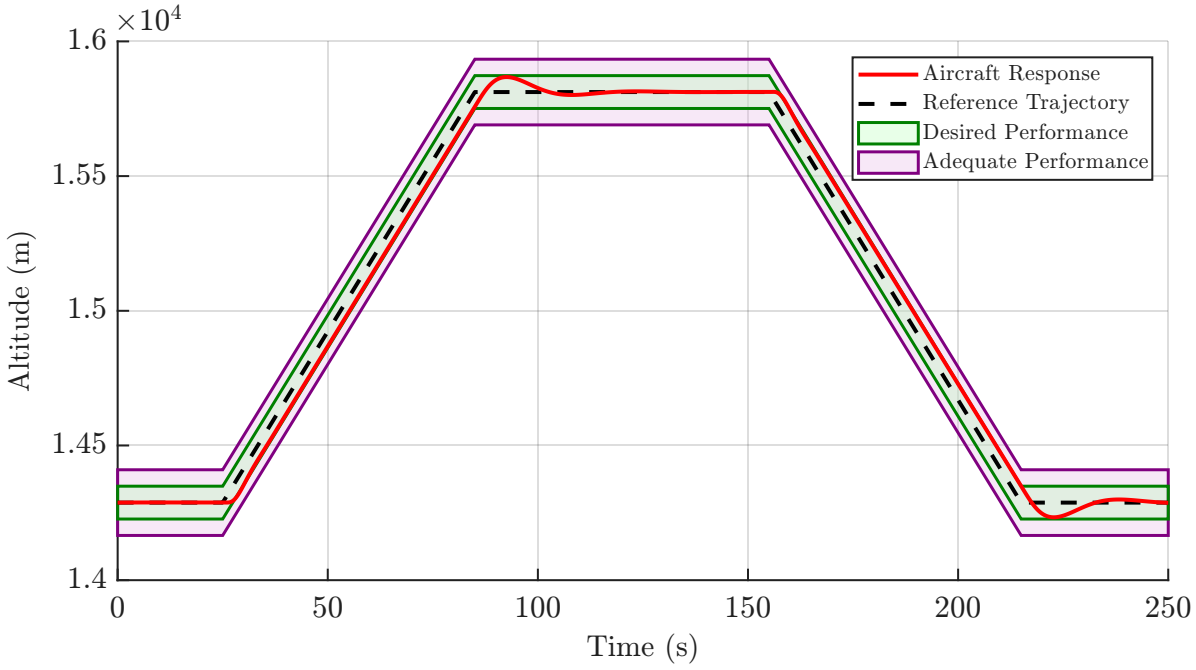


Figure 6.8: Altitude and load factor response for the climb–sink maneuver at 60,000 ft and free-stream Mach number of 3. The maneuver consists of a 5,000 ft rise and sink at a prescribed vertical speed of ±5,000 fpm.

The nominal model successfully follows the reference closely with little overshoot with a lag of almost exactly 2.0 s.[1] The response remains within the desired performance bounds during all phases, demonstrating that the cascaded INDI-based control method is capable of achieving accurate altitude tracking while satisfying the prescribed performance requirements in the nominal case. The corresponding load factor response is shown in Fig. 6.9.

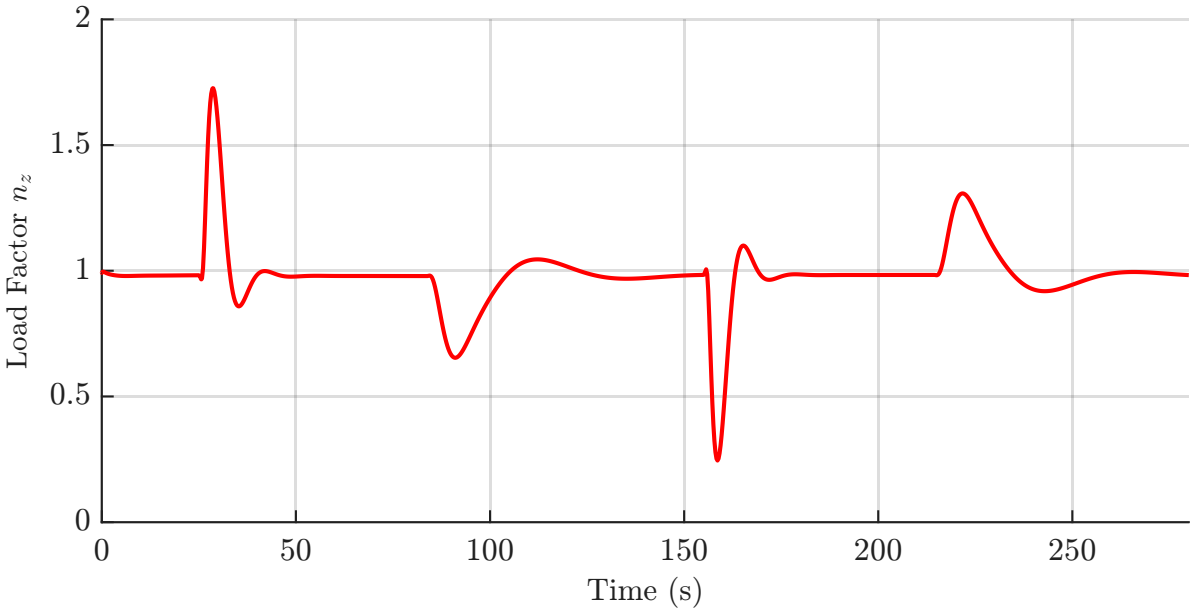


Figure 6.9: Load factor response using the INDI controller.

[1]This lag is expected, since the closed-loop flight path angle dynamics exhibit a rise time on the order of 2 s.

Figure 6.10 shows the flight path angle tracking response using the INDI controller. The tracking performance is very good throughout the maneuver, with the maximum flight path angle remaining below 2° due to the fixed vertical speed. The small overshoots are consistent with the designed closed-loop dynamics and arise from the selected damping characteristics of the controller. The larger transient excursions occur once the commanded altitude has already been reached and the flight path angle command returns toward zero. At this stage, the aircraft still possesses vertical momentum associated with the climb or descent maneuver, causing the altitude to overshoot the target value. This altitude overshoot then forces the controller to command a corrective flight path angle of opposite sign to remove the altitude error and return the aircraft to the reference trajectory. Consequently, the flight path angle excursions become noticeably larger than those associated with the nominal closed-loop response alone.

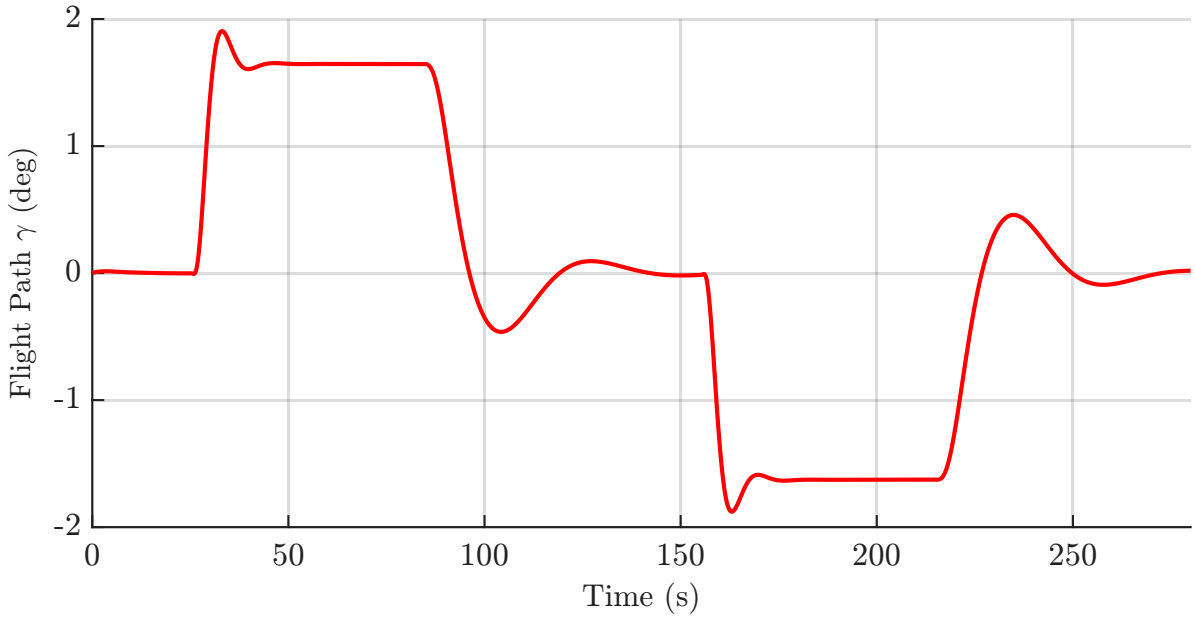


Figure 6.10: Flight path tracking responses using the INDI controller.

The sideslip angle response during the maneuver is shown in Fig. 6.11. The sideslip angle remains essentially zero throughout the simulation, which is expected since no wind disturbances are present and the maneuver does not require any lateral motion.

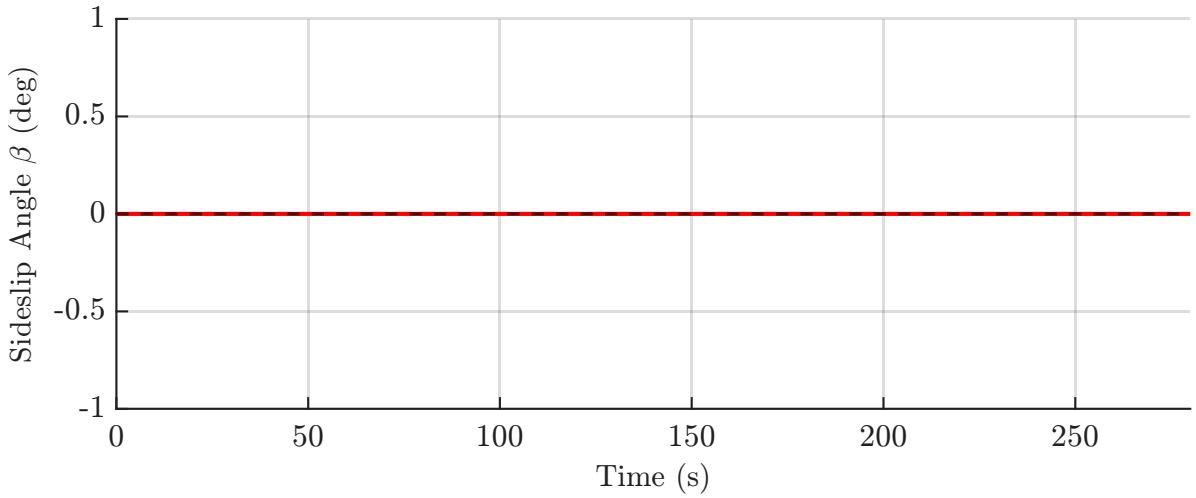


Figure 6.11: Sideslip angle tracking responses using the INDI controller.

The angle of attack response is shown in Fig. 6.12. The gradual variation in angle of attack throughout the climb and descent phases is caused by two primary effects. First, the angle of attack required to maintain a given flight path angle depends on the aircraft mass. As fuel is consumed during flight, the aircraft mass decreases, reducing the lift required for equilibrium flight. Consequently, a smaller angle of attack is sufficient to maintain the same flight path angle later in the maneuver. The second effect comes from the altitude change itself. As the aircraft climbs to higher altitude, the atmospheric density decreases, requiring a higher angle of attack to generate the lift needed to sustain the maneuver. During the descent phase, the increasing atmospheric density allows the required lift to be generated at a lower angle of attack.

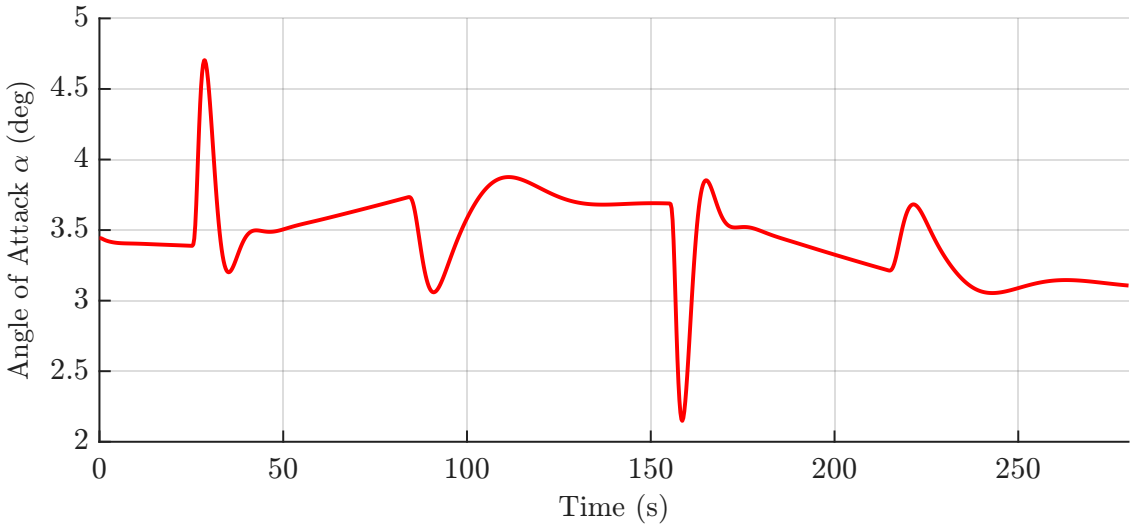


Figure 6.12: Angle of attack tracking responses using the INDI controller.

Good airspeed tracking is also achieved, as shown in Fig. 6.13. The deviation remains below 1 m/s and appears only during maneuvers. This high performance results from the immediate response of the INDI throttle loop, which excludes throttle dynamics and therefore corrects airspeed deviations almost instantaneously.

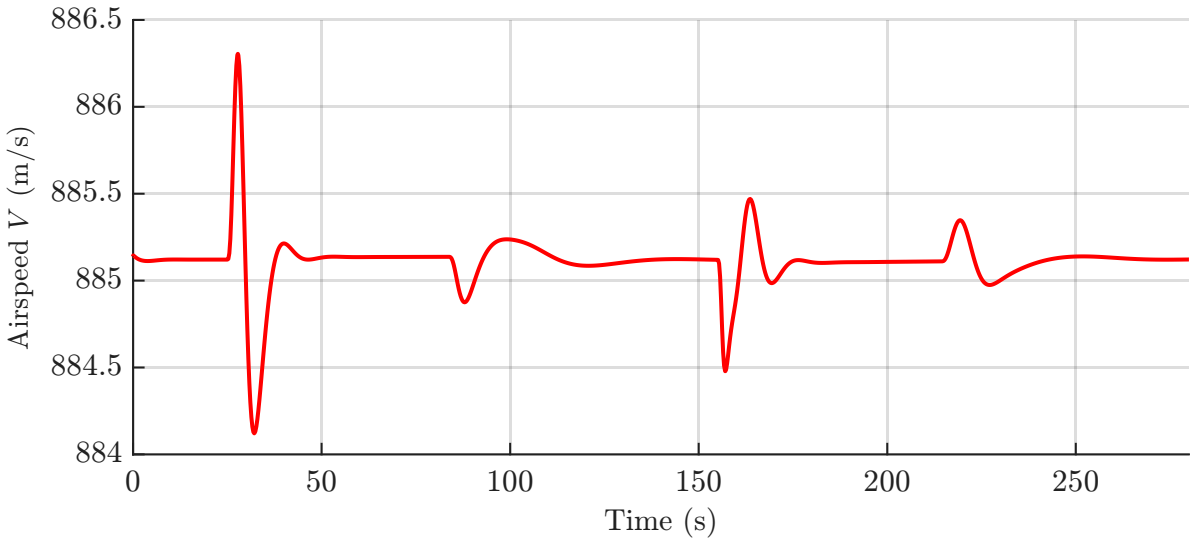


Figure 6.13: Airspeed tracking responses using the INDI controller.

However, the absence of throttle dynamics also causes the sharp transients observed in the throttle command, since the thrust reacts much faster than the flight path angle and angle of attack. Including realistic throttle or engine dynamics would significantly smooth this behavior.

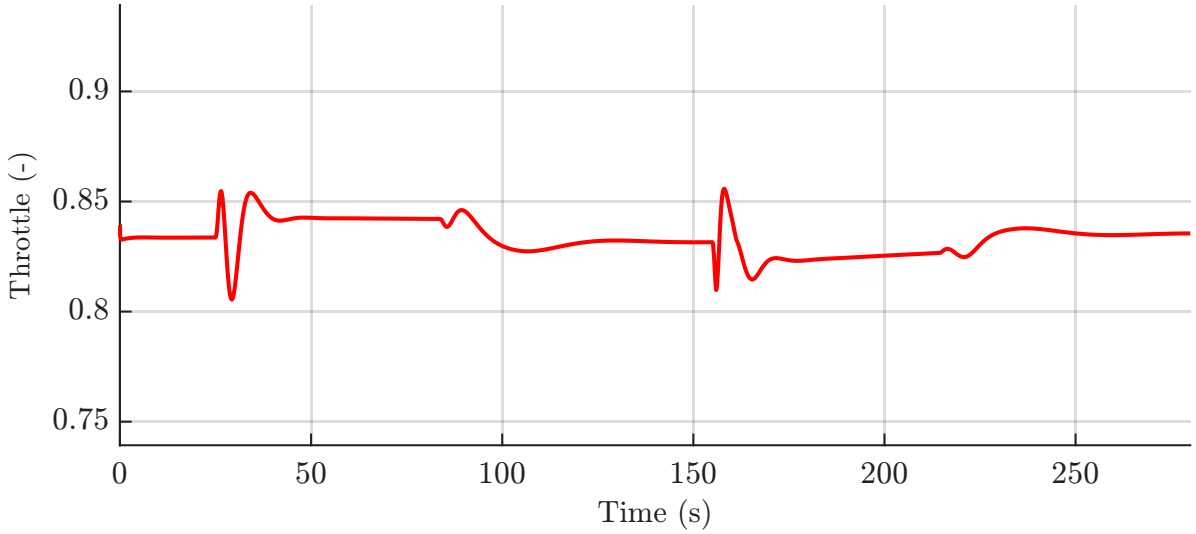


Figure 6.14: Throttle tracking responses using the INDI controller.

The pitch rate in Fig. 6.15 shows brief spikes during each commanded maneuver, caused by the virtual control signal reacting strongly to the initial attitude error. Once the command is reached, the response settles rapidly back to zero, showing that the controller damps the motion effectively.

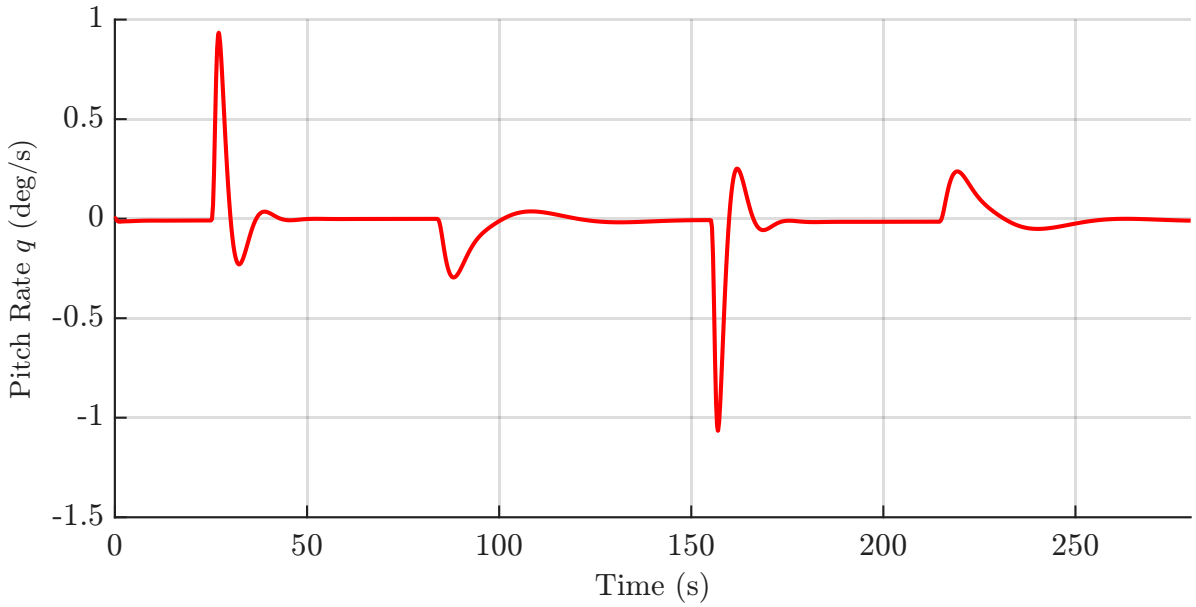


Figure 6.15: Pitch rate response using the INDI controller in the nominal case.

Figure 6.16 shows the control deflections during the climb and sink phases. Both elevons move symmetrically to produce an effective elevator input and stay close to their trim values throughout. A sharp spike appears at the start of the maneuver, where the large initial command error produces a correspondingly large virtual control output. When the command returns to level flight, the state error is small and the deflections are much smoother. The elevons stay well below saturation, showing that the

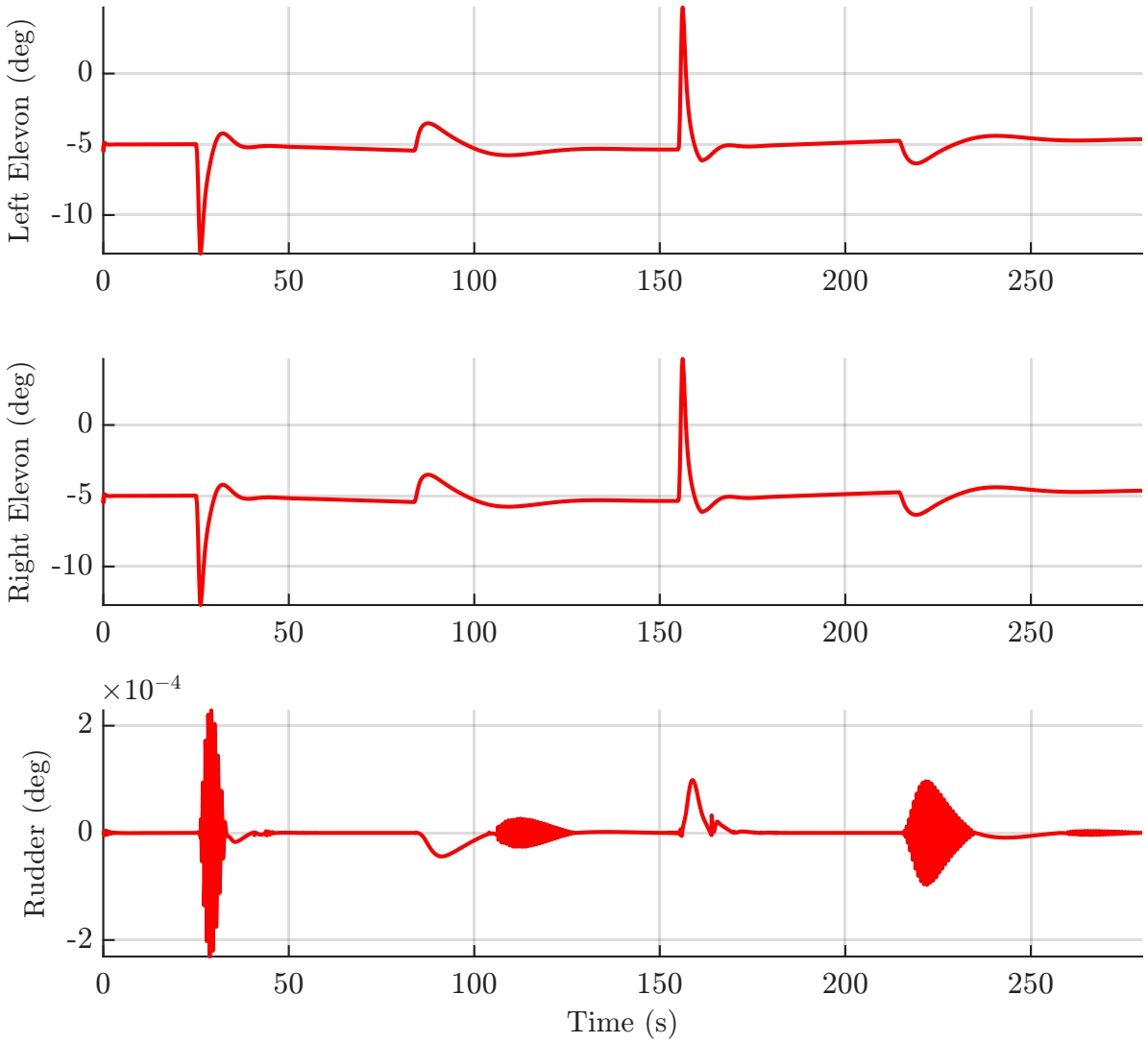


Figure 6.16: Left and right elevon and rudder deflection responses using the INDI controller.

climb requires no excessive control effort, and the rudder remains essentially at zero, as expected for a purely longitudinal maneuver without wind or turbulence.

Heading Change Maneuvers

The next objective is to verify whether the GHAME vehicle can track a commanded heading change within the specified performance requirements. In this case, a maneuver of 8° in heading is performed at a freestream Mach number of 3 over a simulation duration of 60 s.

Free-stream Mach number: $M_\infty = 3$ Altitude: $h = 60{,}000$ ft Heading change: 8 deg

The altitude response behaves as expected. The altitude-hold algorithm successfully maintains constant altitude throughout the maneuver, as shown in Fig. 6.17. However, the load factor response in Fig. 6.18 requires further discussion. The steady-state load factor settles slightly below unity, at $n_z \approx 0.982$. The same effect is also present in the climb maneuver of Fig. 6.8, although it is less visible. The load factor being less than one arises from gravity contributing to the required centripetal acceleration during high-speed flight over a spherical Earth.

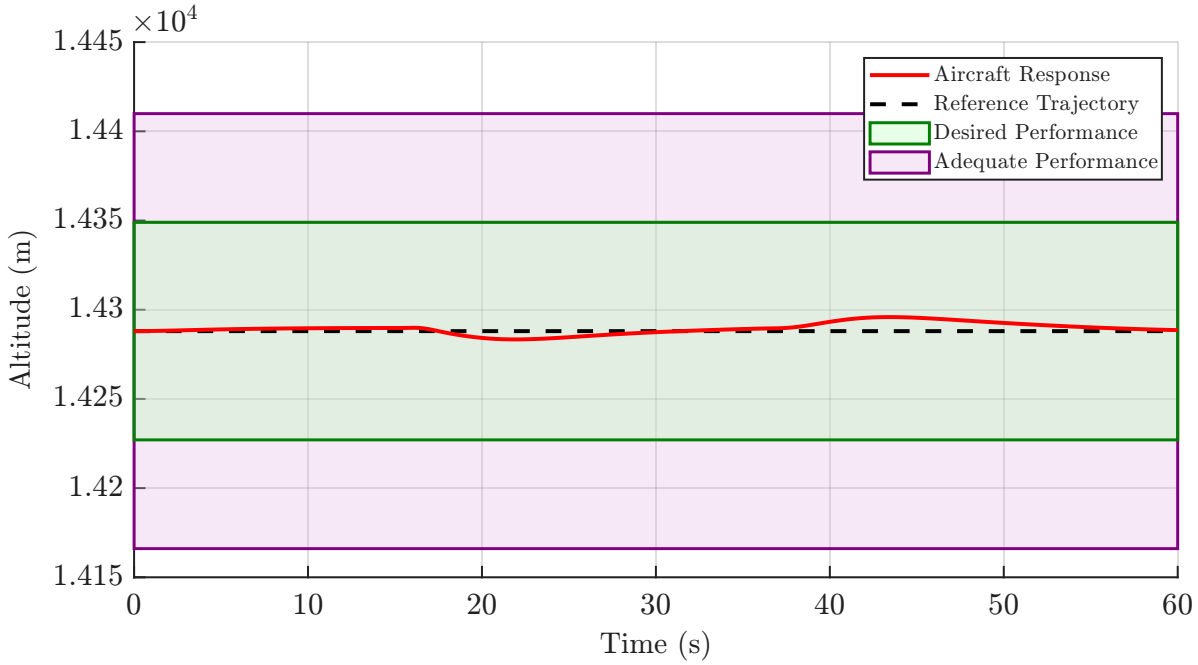


Figure 6.17: Altitude response during heading command maneuver.

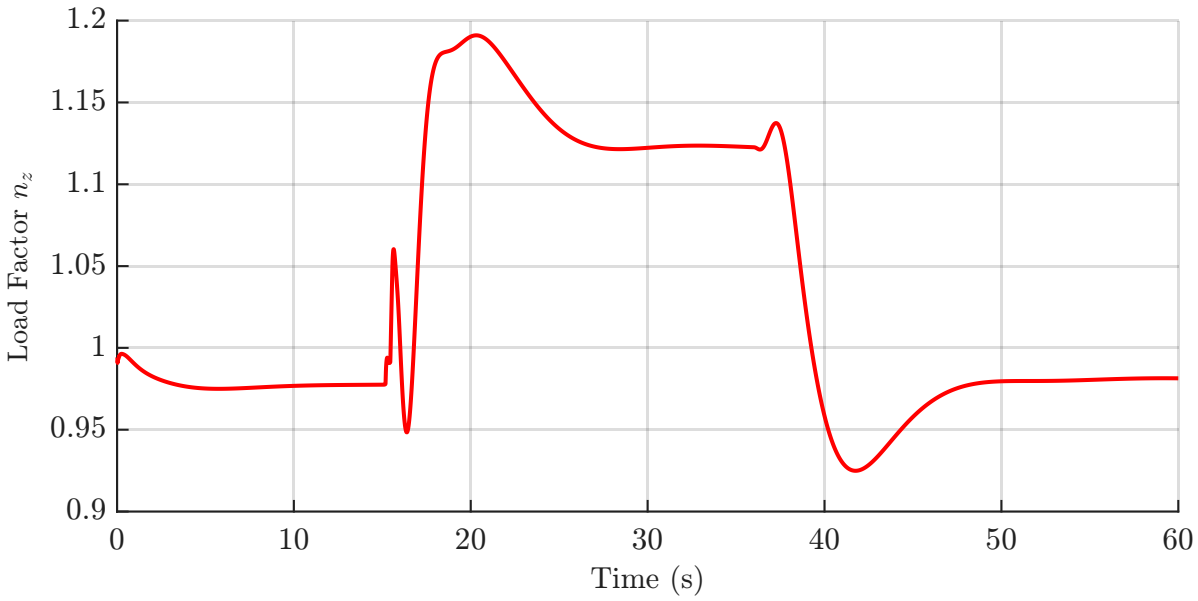


Figure 6.18: Load factor response during heading command maneuver.

In a flat-Earth formulation, a velocity vector that is initially tangent to the surface remains tangent indefinitely, so maintaining constant altitude requires a load factor of exactly $n_z = 1$. In contrast, in a spherical and rotating Earth model, the aircraft moves in an inertial coordinate system while the Earth rotates beneath it. For eastward flight at the equator, the total inertial velocity about Earth's centre is the sum of the aircraft's Earth-relative velocity V_K and the Earth's surface velocity $\omega_\oplus R_0$. To remain at constant altitude, the aircraft must generate a centripetal acceleration equal to

$$a_c = \frac{(V_K + \omega_\oplus R_0)^2}{R_0}.$$

Since gravity already provides most of this inward acceleration, the aerodynamic lift required to maintain altitude is reduced accordingly, resulting in a steady load factor of

$$n_z = 1 - \frac{(V_K + \omega_\oplus R_0)^2}{R_0\, g}.$$

Substituting the simulated values $V_K = 885.15$ m/s, $R_0 = 6.3853 \times 10^6$ m, $\omega_\oplus = 7.292115 \times 10^{-5}$ rad/s, and $g = 9.79$ m/s^2 yields $n_z \approx 0.9821$, which agrees well with the observed steady-state value. The same steady-state load factor is also obtained from the HYPER6 simulation of Zipfel [2025].

During the heading maneuver, the aircraft performs a coordinated bank to generate the horizontal component of lift required for turning. In such a coordinated, level turn, the lift vector is inclined by the bank angle ϕ, so its vertical component must increase to balance the weight. The total lift, and thus the load factor, scale with $1/\cos(\phi)$. With a bank angle of $\phi_{\max} = 30°$, the instantaneous load factor becomes:

$$n_{z,\text{turn}} = \frac{n_z}{\cos(\phi_{\max})} = \frac{0.982}{\cos(30°)} = 1.134,$$

which matches the increase in n_z observed during the turn. After the bank is removed, the load factor returns to its steady-state value of approximately 0.982.

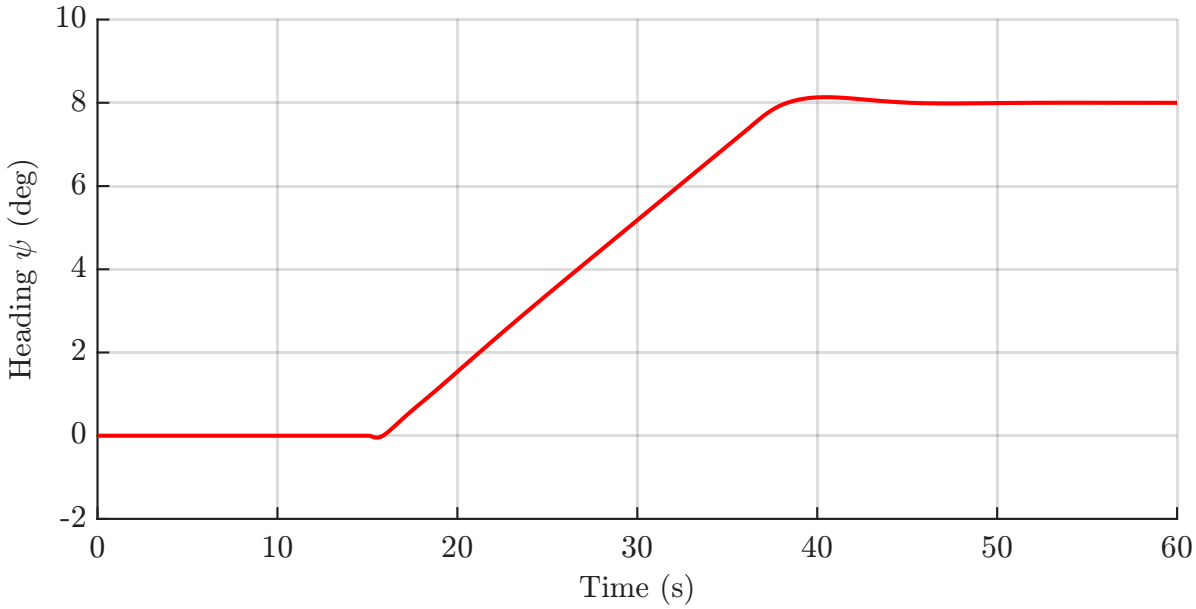


Figure 6.19: Heading response during the heading-command maneuver.

The heading response in Fig. 6.19 shows that the command is tracked accurately, with only a minor overshoot well below the specified 1° limit. The heading holds at its trimmed value until the command is applied at $t \approx 15$ s, after which the aircraft banks and turns. The change proceeds almost linearly because the turn rate is constrained by the maximum bank angle $\phi_{\max} = 30°$ and is low due to the high flight speed, giving a large turn radius. The heading settles onto the commanded value at approximately 40 s, so the turn itself lasts about 22 s. The response is stable, well damped, and meets the performance requirement.

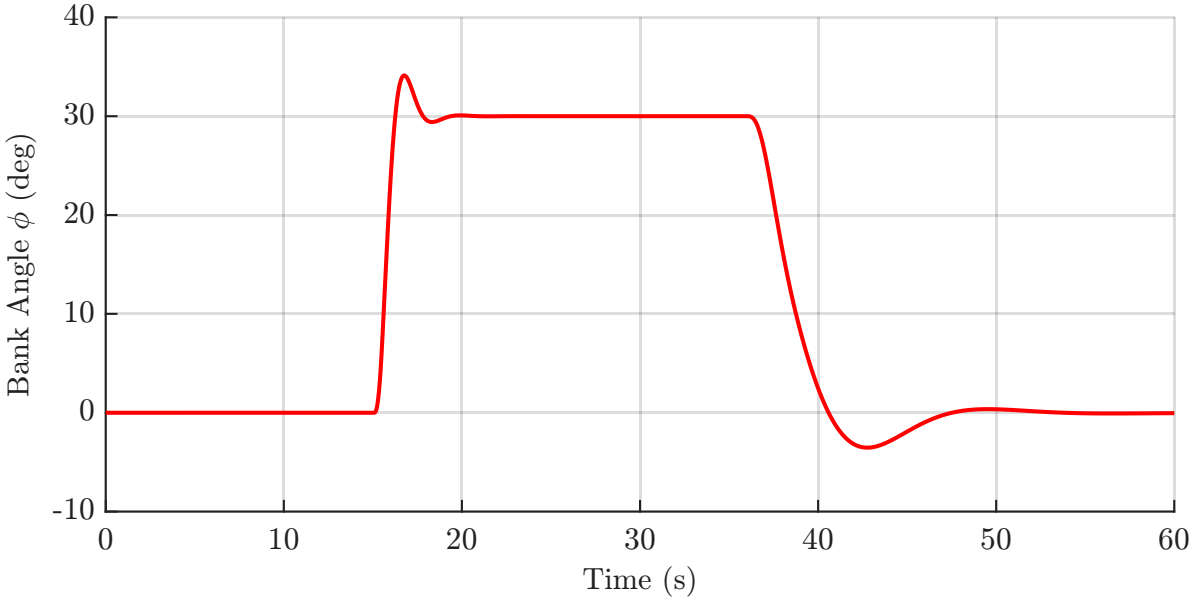


Figure 6.20: Bank angle response during the heading-command maneuver.

The bank angle response in Fig. 6.20 shows how the aircraft rolls to initiate and sustain the commanded turn. The bank angle holds at zero until the command is applied, after which the aircraft rolls rapidly to establish the turn. A strict bank angle limit of 30° is enforced, and the response saturates at this value during the turn, apart from a small initial overshoot of a few degrees before it settles onto the limit. The bank angle is held at 30° throughout the turn and is then commanded back to zero once the target heading is approached, with a small undershoot before the aircraft rolls level and returns to zero with negligible steady-state error.

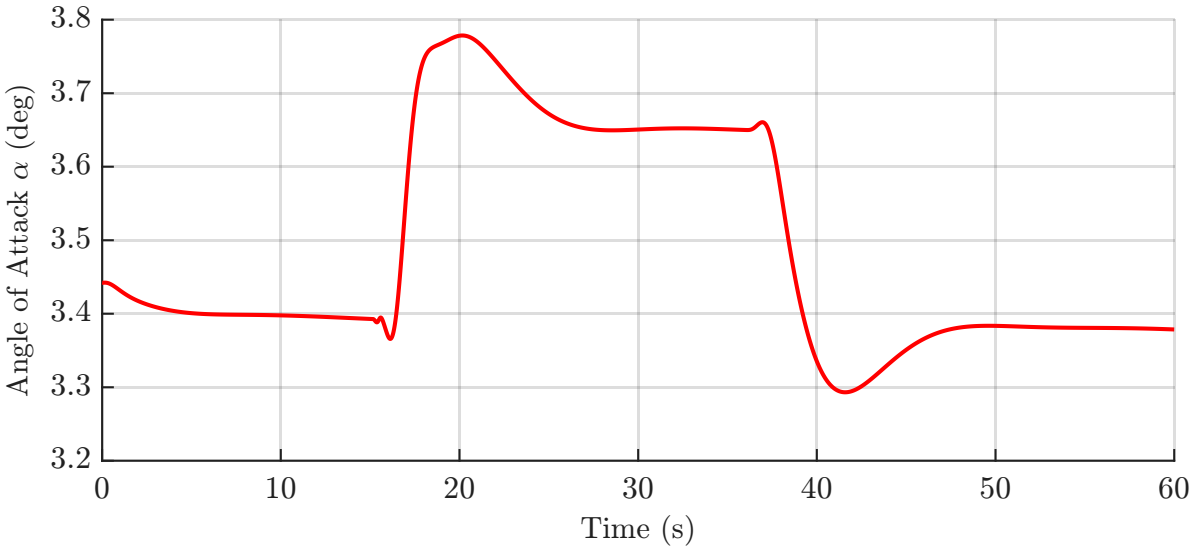


Figure 6.21: Angle of attack response during the heading-command maneuver.

The angle of attack shown in Fig. 6.21 is not commanded directly but is obtained as part of the trimmed solution, set by the lift required to balance the aircraft weight at the given flight condition. Initially, α decreases slightly to reduce lift and achieve the steady load factor $n_z \approx 0.982$. It then continues to decline gradually over time as fuel is consumed: as the aircraft loses mass, less lift is

required to maintain level flight, and a correspondingly smaller angle of attack is needed. When the heading command is applied, α increases sharply. This occurs because the aircraft must maintain constant altitude while the airspeed is held fixed by the FCS. During the banked turn, the vertical component of lift decreases with $\cos(\phi)$, but the total lift must still balance the aircraft weight, so with the dynamic pressure essentially constant the lost lift can only be recovered by increasing the angle of attack. Once the aircraft rolls level, the additional angle of attack is no longer required and α drops back, after which it resumes its slow decline as fuel continues to burn.

The sideslip angle in Fig. 6.22 is held very close to zero throughout the maneuver, with only small transients of a few hundredths of a degree at the roll-in and roll-out, confirming that coordinated flight is maintained at all times.

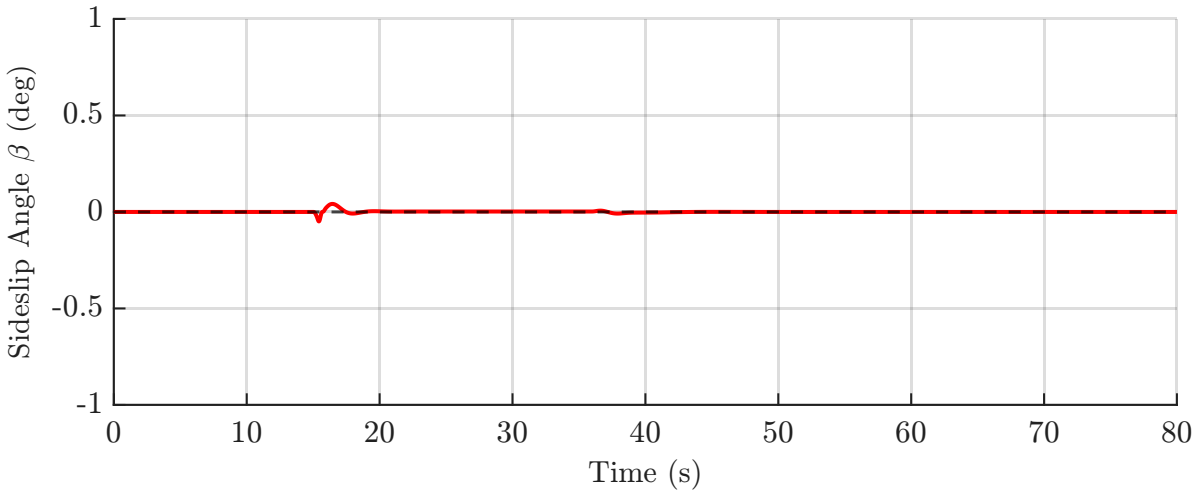


Figure 6.22: Sideslip angle response during the heading-command maneuver.

The airspeed and throttle are essentially unaffected by the roll maneuver, as shown in Fig. 6.23 and Fig. 6.24. The FCS holds the airspeed at its commanded value throughout, with only negligible variation in the throttle command during the turn.

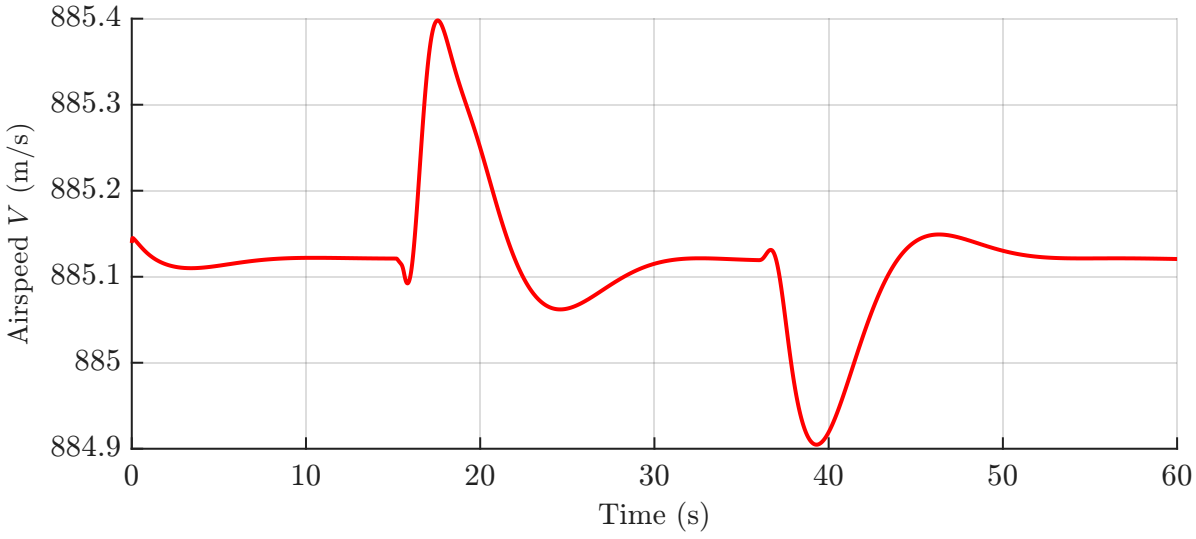


Figure 6.23: Airspeed during heading-command maneuver.

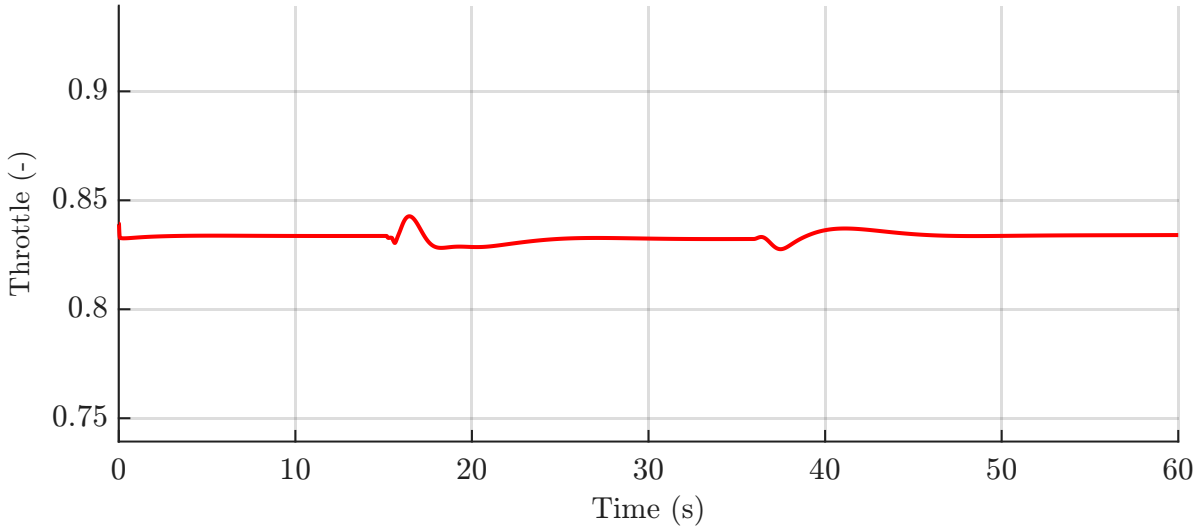


Figure 6.24: Throttle during heading-command maneuver.

The roll and yaw rate responses in Fig. 6.25 reflect the two phases of the turn, the roll-in near 15 s and the roll-out near 37 s. At roll-in the aircraft rotates to the commanded bank angle, producing a sharp positive roll rate that peaks at about 38°/s before returning to zero. A mirror-image negative peak appears at roll-out as the aircraft returns to wings-level. Between these events the roll rate is essentially zero, consistent with the constant 30° bank held during the steady turn.

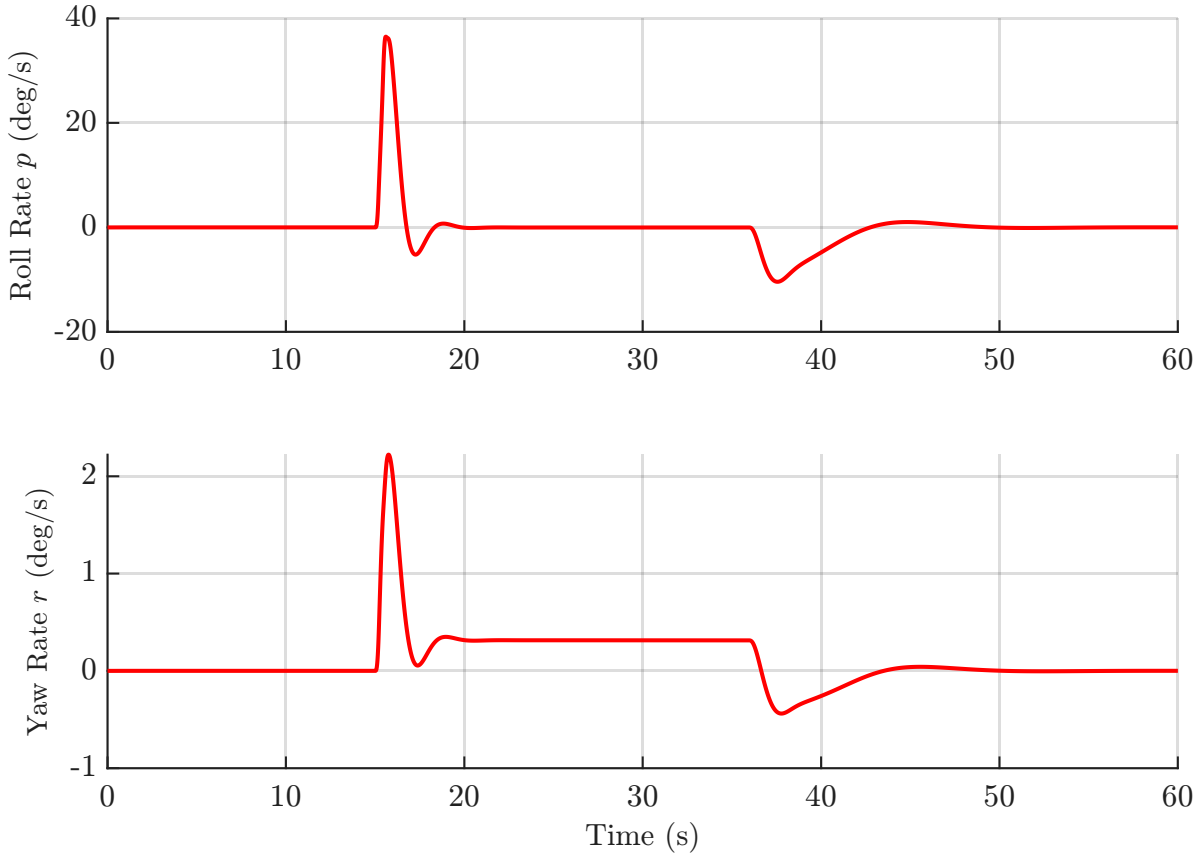


Figure 6.25: Roll rate p and yaw rate r during the heading-command maneuver.

The yaw rate has two contributions. The sharp transients at roll-in and roll-out coincide with the roll rate peaks and stem from the yawing moment produced during the rapid roll, including adverse yaw and rotational coupling. More significant is the small but sustained yaw rate of roughly $0.3^\circ/\mathrm{s}$ held throughout the steady turn. This is the kinematic yaw rate of a coordinated turn. As the aircraft banks at constant altitude, the velocity vector rotates in the horizontal plane, so the nose must yaw steadily to stay aligned with the flight path and keep the sideslip at zero.

This value can be verified analytically. For a coordinated level turn, the turn rate is given by:

$$\dot{\psi} = \frac{g \tan(\phi_{\max})}{V_K}, \tag{6.12}$$

At the flight condition considered, $V_K \approx 885$ m/s (Mach 3 at 60,000 ft) and $\phi_{\max} = 30^\circ$, giving $\dot{\psi} \approx 0.37^\circ/\mathrm{s}$. This agrees closely with the sustained yaw rate in Fig. 6.25 and confirms a coordinated turn. The corresponding turn radius is about 138 km, and a full 360° turn would take roughly 16 minutes, reflecting the very wide turning circle of high-speed flight.

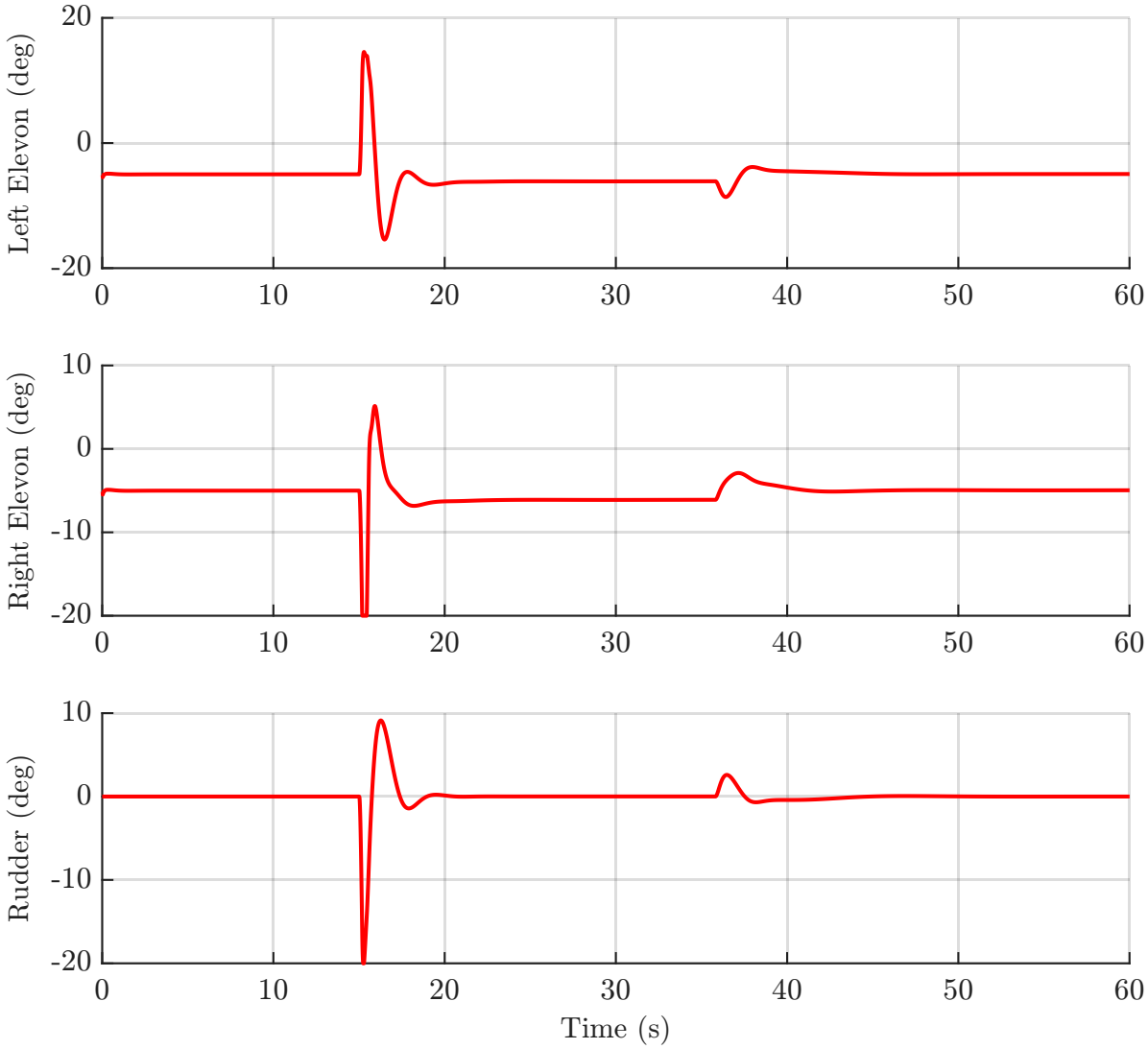


Figure 6.26: Left and right elevon and rudder deflections during the heading-command maneuver.

The control surface deflections remain close to their trim values and deviate only briefly during the banking maneuver, as shown in Fig. 6.26. At the onset of the heading command, the elevons deflect differentially to generate the rolling moment that initiates the bank, while the rudder deflects momentarily to maintain coordinated flight and suppress sideslip. The rudder transient is the largest of the three surfaces and briefly approaches saturation during roll-in. Once the target bank angle is reached, the elevons settle close to their trimmed value of about $-5°$, with the slightly more negative deflection providing the extra lift required during the turn, and the rudder returns near zero. During roll-out near 37 s, the elevons and rudder deflect in the opposite sense to roll the wings level, after which they return to their trim positions.

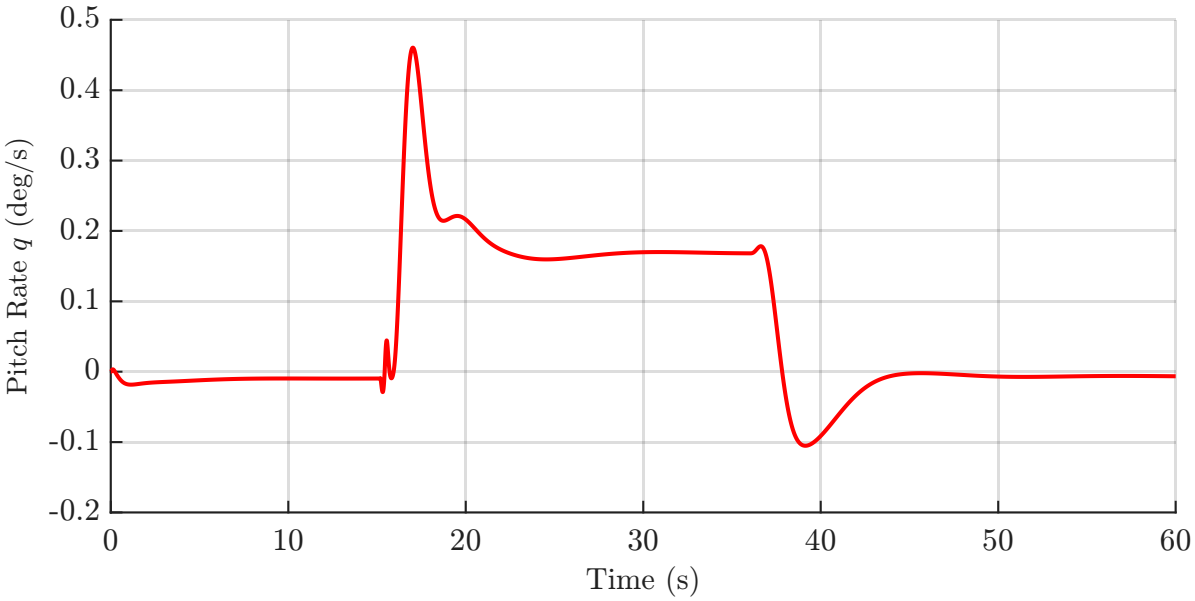


Figure 6.27: Pitch rate q during the heading-command maneuver.

The pitch rate in Fig. 6.27 shows a positive pulse at roll-in, which raises the angle of attack to the higher value needed to sustain lift during the turn. It then returns near zero while the bank is held, and a negative pulse at roll-out lowers the angle of attack back to its level-flight value.

In summary, the nominal simulations confirm that the cascaded INDI controller performs well in both test cases. In the altitude maneuver, the aircraft tracks the climb and sink reference within the desired performance bounds, with airspeed deviations kept below 1 m/s. In the heading maneuver, the $8°$ command is tracked accurately with an overshoot well below the $1°$ limit, while altitude, airspeed, and sideslip are held essentially constant throughout the turn. In both cases the responses are stable and well damped, the control surfaces stay within their authority, and the observed behaviour agrees with the analytical predictions for load factor and turn rate. These results establish the nominal baseline against which the controller's robustness to real-world effects is assessed in the following sections.

Effect of Perturbations

To assess the controller's performance under realistic flight conditions, the same nonlinear simulations for the altitude and heading commands are repeated with selected physical and environmental disturbances enabled. These perturbations do not represent model uncertainty, which will be analyzed separately, but instead simulate measurable disturbances that occur during actual flight. The objective is to verify that the control system maintains satisfactory tracking and stability when exposed to wind, turbulence, sensor noise, and actuator disturbances. These effects are modeled as follows:

1. **IMU errors:** sensor bias, zero-mean Gaussian noise and misalignment errors are added to the accelerometer and gyroscope outputs within the INS to emulate realistic measurement errors. The specific parameters are listed in Table 4.2, and the implementation is discussed in Sec. 4.5.

2. **Wind and turbulence:** atmospheric wind in geodetic coordinates $[v_A^E]_{\text{HW}}^D$ is computed using the `HWM14` block in Simulink, while stochastic turbulence is modeled using the Dryden formulation with intensity $\sigma = 2$ m/s and correlation length $L = 150$ m. The implementation of both effects is discussed in Sec. 4.3.

3. **Actuator perturbations:** two elevon perturbation types are applied: short Gaussian impulses at selected times and constant step biases to represent actuator offsets. The impulses test the controller's response to rapid deflection changes, while the steps evaluate its ability to reject longer-term biases. No constant bias is applied initially.

These effects have negligible influence on the command tracking shown in Fig. 6.28, as the altitude command remains accurately followed throughout the entire maneuver. High-frequency fluctuations appear in the load factor response but do not affect the overall tracking performance.

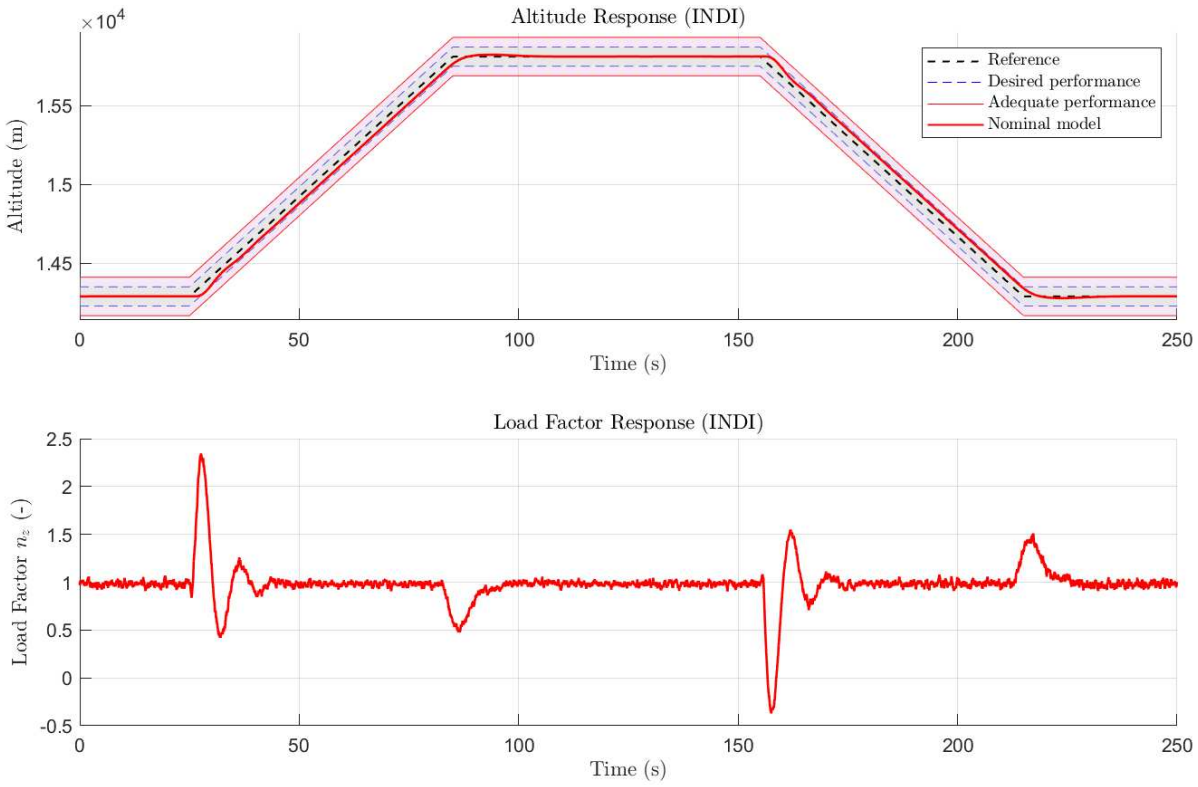


Figure 6.28: Altitude and load factor response during altitude command maneuver under perturbations, noise, wind and turbulence.

The actuator perturbations follow a predefined sequence of time intervals and amplitudes, which are summarized in Table 6.2. The chosen schedule introduces control input variations throughout the entire altitude command maneuver, and allows a consistent evaluation of the controller's tracking performance under different actuation conditions.

Table 6.2: Elevator perturbations applied in the altitude command test case.

Type	**Time interval, s**	**Amplitude, deg**
Gaussian impulse	$t = [40,\ 140,\ 210]$	$[11.5,\ -13.75,\ -11.5]$
Constant step	$20 \leq t < 100,\ 100 \leq t < 200,\ t \geq 200$	$-3,\ 7,\ -4$

The effect of the perturbations can be clearly observed in the control surface deflection plot shown in Fig. 6.29. The perturbation signal is applied to the right elevon deflection after the actuator dynamics, and thus, its influence is directly visible in the right elevon response. In the figure, the green line represents the injected perturbation signal, the blue line corresponds to the commanded elevon deflection computed by the INDI-based FCS, and the red line shows the final deflection applied to the airframe.

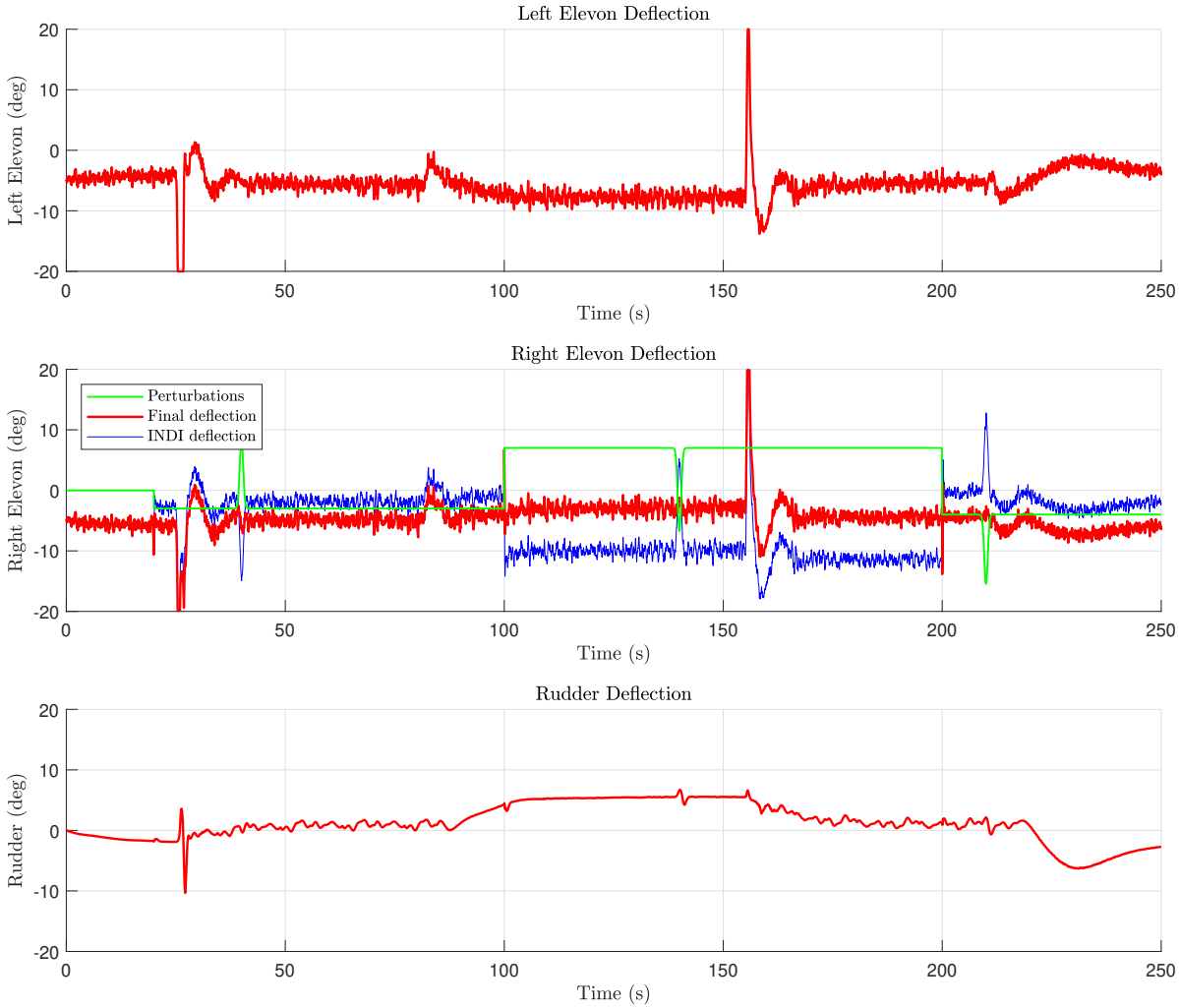


Figure 6.29: Left and right elevon and rudder deflection during altitude command maneuver under perturbations, noise, wind, and turbulence.

Compared with Fig. 6.16, the combined disturbances introduce small high-frequency variations in the control surface deflections. These oscillations are caused by the imposed turbulence and measurement noise, with turbulence having the larger influence. Most of the sensor noise is successfully suppressed by the implemented noise filter, leaving only minor residual fluctuations in the deflection signals. The wind field produces a slight rudder offset to maintain coordinated flight, while the injected perturbations on the right elevon (green line) are rapidly counteracted by the controller. The commanded response (blue line) adjusts almost instantaneously, resulting in an overall deflection (red line) that closely resembles the undisturbed case. It is important to note that the FCS has no direct knowledge of the applied actuator perturbations, as it relies on its internal on-board model of the actuator, which is unaffected by the disturbances. A good demonstration of the strong disturbance rejection capability of the INDI architecture, even when operating with a nominal actuator OBM.

During the start of the climb and sink phases, small sideslip deviations appear due to the presence of wind. This occurs because, at the onset of the maneuver, the control surfaces are primarily engaged in generating the pitching moment required to establish the commanded climb or sink rate. As a result, the available control authority for counteracting wind-induced roll moments is temporarily reduced, allowing a transient roll and sideslip response to develop. The actuator perturbations also contribute to sideslip deviations, as they generate roll moments that the control system must counteract, as shown in Fig. 6.30.

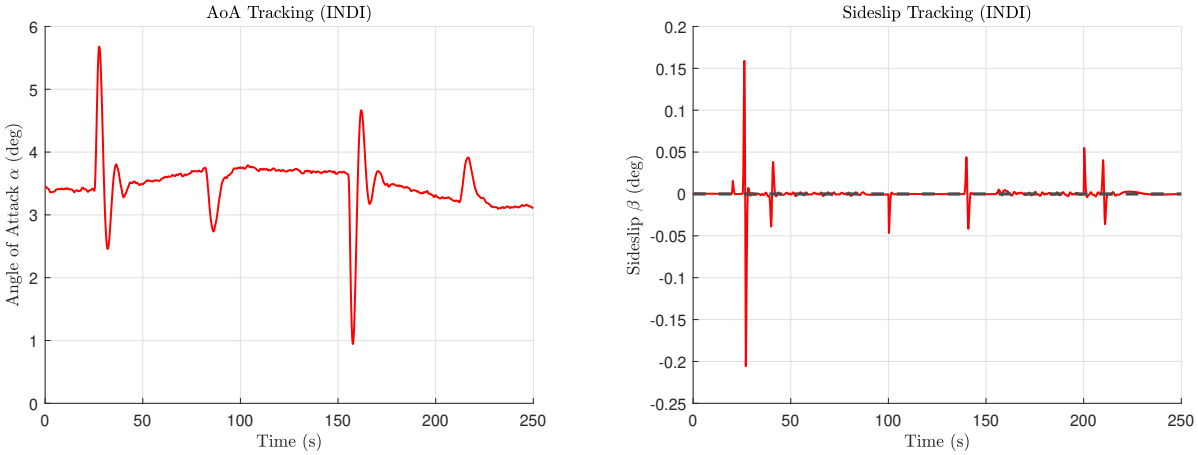


Figure 6.30: Angle of attack and sidedslip angle response during altitude command maneuver under perturbations, noise, wind and turbulence.

The fluctuations in sideslip and angle of attack are less pronounced than those observed in the control deflections, since the vehicle's aerodynamic response and inertia naturally smooth out the high-frequency control activity. Once the maneuver stabilizes, the INDI controller compensates for these effects, restoring the bank and sideslip to their nominal values, while the flight path and angle of attack remain closely aligned with the nominal trajectory.

Heading maneuver with perturbations:

For the heading maneuver, the perturbations applied to the heading command are slightly adjusted, as this scenario corresponds to a shorter simulation lasting only 80 seconds. The modified perturbation schedule, including the timing and amplitudes of the applied inputs, is summarized in Table 6.3.

Table 6.3: Elevator perturbations applied in the heading command test case.

Type	**Time interval, s**	**Amplitude, deg**
Gaussian impulse	$t = [20,\ 40]$	$[11.5,\ -13.75]$
Constant step	$20 \leq t < 60,\ t \geq 60$	$-3,\ 7$

Compared with Fig. 6.26, the effects are similar to those observed in the altitude maneuver case, including small fluctuations caused by turbulence, and the disturbances are effectively rejected. As shown in Fig. 6.31, a new observation is the increased oscillatory behavior during the heading hold phase, which is not present in longitudinal flight and gradually subsides once the maneuver concludes.

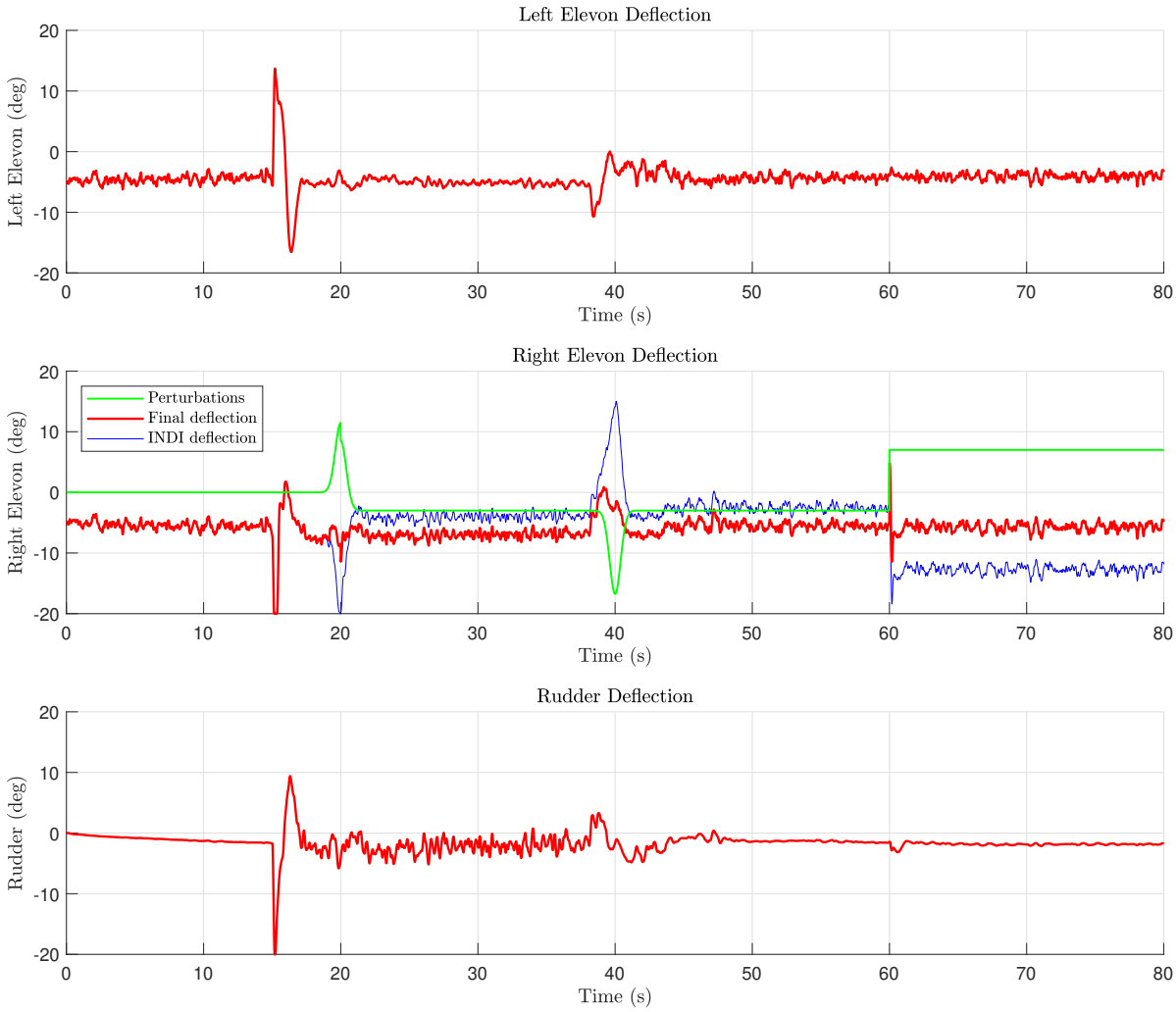


Figure 6.31: Left and right elevon and rudder deflection during heading command maneuver under perturbations, noise, wind, and turbulence.

The rapid disturbance rejection of the INDI controller ensures that the heading command remains accurately tracked, as shown in Fig. 6.32. Despite the presence of perturbations, noise, wind, and turbulence, the controller maintains the commanded heading without noticeable degradation. The actuator disturbances are slightly visible in the bank angle response, but no significant fluctuations appear here. The effect of wind is seen as a small steady-state offset in the bank angle, as a slight roll is required to maintain the desired heading against the crosswind component.

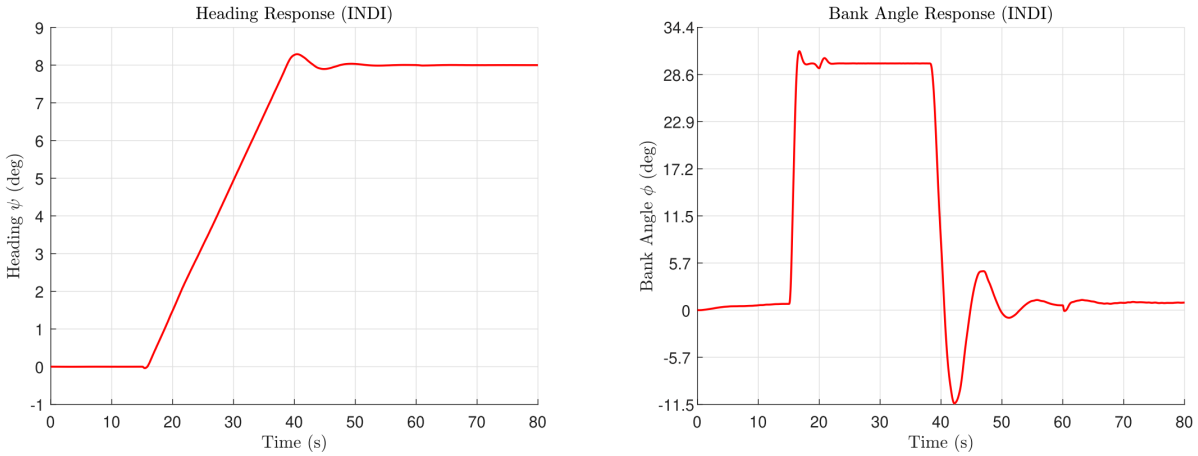


Figure 6.32: Heading and flight-path response during the heading-command maneuver under perturbations, noise, wind and turbulence.

The angle of attack response closely follows the unperturbed case, with only small fluctuations introduced by the turbulence model. The right elevon perturbations excite the sideslip angle, as shown in Fig. 6.33, which exhibits brief spikes of up to 0.1° that are rapidly corrected by the controller. These deviations are minor and do not pose a problem for stability or performance. The wind itself has a negligible effect at a flight speed of Mach 3, although its influence can be expected to become more pronounced at lower speeds.

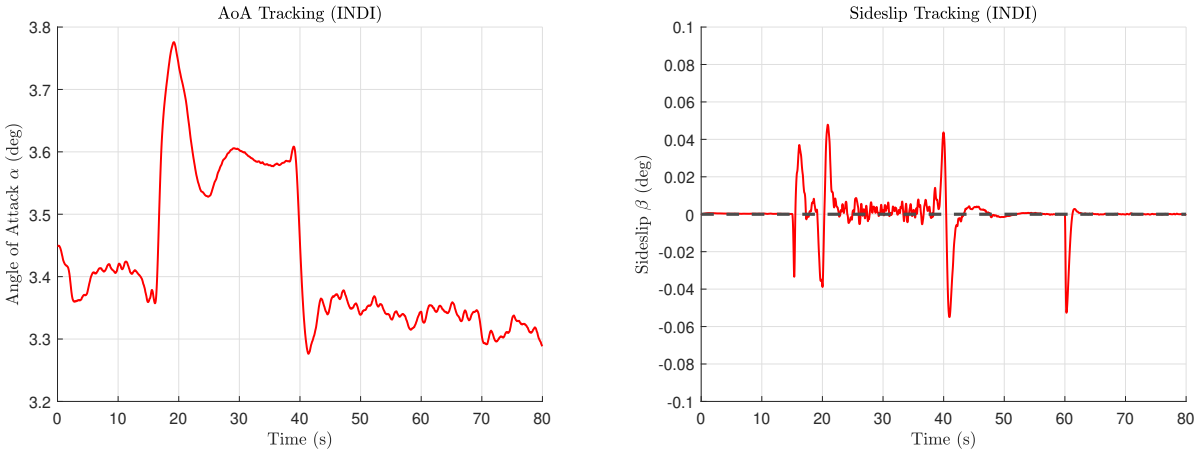


Figure 6.33: Angle of attack and sideslip angle during heading command maneuver under perturbations, noise, wind and turbulence.

Effect of Uncertainty

The following simulations investigate the effect of aerodynamic coefficient uncertainty on the closed-loop performance. Although INDI is generally effective in rejecting disturbances and parameter variations, its robustness is verified through a Monte Carlo analysis. Each aerodynamic coefficient is modeled using the parametric uncertainty formulation [Skogestad and Postlethwaite, 2005]:

$$\alpha_p = \bar{\alpha}\,(1 + r_\alpha \Delta)\,, \quad |\Delta| \leq 1, \tag{6.13}$$

where $\bar{\alpha}$ denotes the nominal value, r_α is the relative uncertainty, and Δ is a normalized uncertainty variable. In this simulation $r_\alpha = 0.3$, resulting in parameter variations within the interval $\alpha_p \in \bar{\alpha}\,[0.7,\ 1.3]$. This corresponds to a maximum deviation of ±30%, which reflects the significant modeling uncertainty associated with hypersonic flight conditions. The FCS uses the nominal aerodynamic model for control law computation, while the nonlinear simulation evaluates performance using the perturbed coefficients defined in Eq. (6.13). Consequently, the controller is required to maintain acceptable performance despite discrepancies between the assumed and actual aerodynamic parameters.

The simulations were performed with 100 different uncertainty realizations to provide sufficient data. The altitude command response under aerodynamic uncertainty for all realizations is shown in Fig. 6.34. The blue traces correspond to the uncertain realizations, while the red line represents the nominal response. As expected for INDI, the controller effectively rejects disturbances even under this level of aerodynamic uncertainty. The 30% variation in control effectiveness does not significantly degrade command tracking performance.

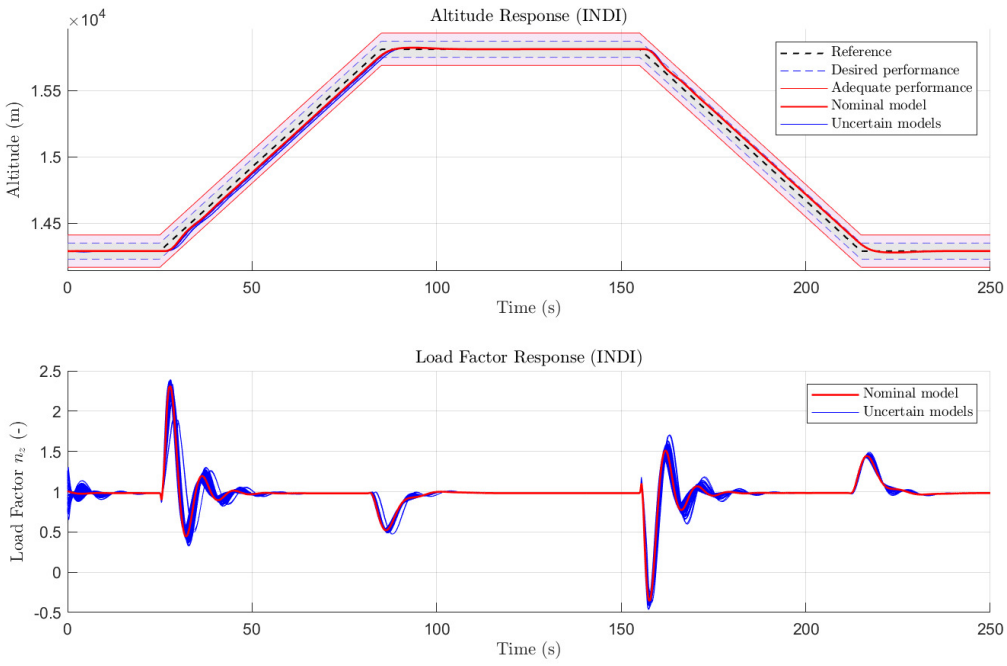


Figure 6.34: Altitude and load factor for nominal and uncertain models during the altitude-command maneuver.

The heading command response under aerodynamic uncertainty is shown in Fig. 6.35. The figure includes one hundred uncertainty realizations, illustrating the variation in response due to aerodynamic parameter deviations. The heading response shows only minor deviation from the nominal case, as the heading dynamics evolve more slowly than the effects introduced by the aerodynamic uncertainty. The controller maintains accurate heading tracking throughout the maneuver, and the steady-state error remains negligible even under large parameter variations.

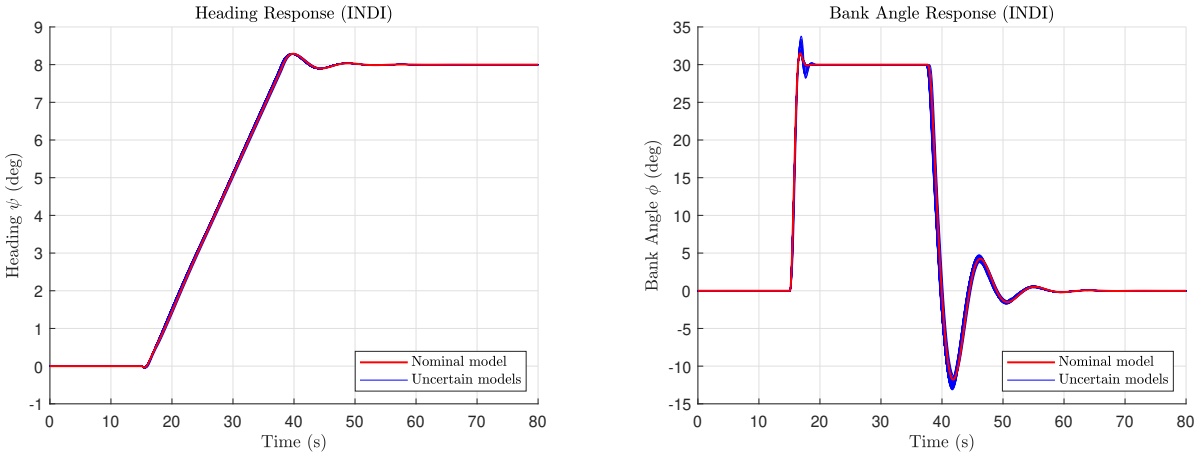


Figure 6.35: Heading and bank angle response for nominal and 100 uncertain models during the heading-command maneuver.

The right plot in Fig. 6.35 shows the bank angle response, where the effects of aerodynamic uncertainty are slightly more evident. Up to the onset of the maneuver, the response remains nearly identical across all models. During the maneuver, variations in aerodynamic coefficients cause small deviations in the instantaneous aerodynamic balance and dynamic response.

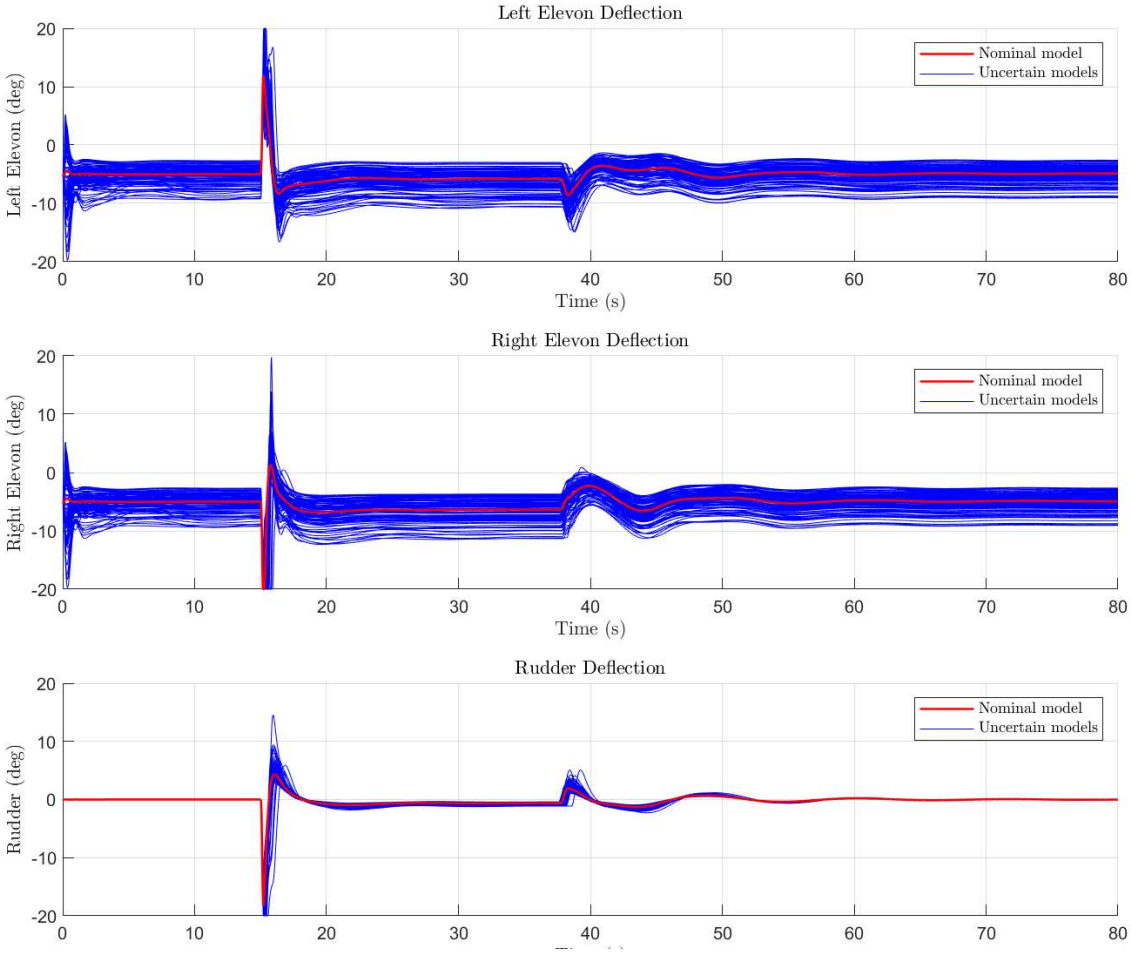


Figure 6.36: Elevons and rudder deflection for the nominal and 100 uncertain models during the heading-command maneuver.

These deviations remain minor and decay rapidly once the maneuver ends, demonstrating that the INDI controller effectively rejects the induced disturbances and restores the nominal bank trajectory. While the overall attitude response appears largely unaffected by uncertainty, its influence becomes evident in the control surface deflections shown in Fig. 6.36. Each uncertain airframe model requires slightly different control deflections to maintain force and moment equilibrium and to achieve the commanded motion. The INDI controller adapts to these variations almost instantaneously, successfully compensating for aerodynamic uncertainty and preserving the desired response.

Effect of Sensor Delay

The effect of unsynchronized sensor delay is analyzed by progressively increasing the measurement delay while keeping the actuator dynamics and filters unchanged. The sensor delay is increased by 0.01 s for each simulation run. The corresponding altitude, flight path, and heading responses are shown in Figs. 6.37, 6.38, and 6.39.

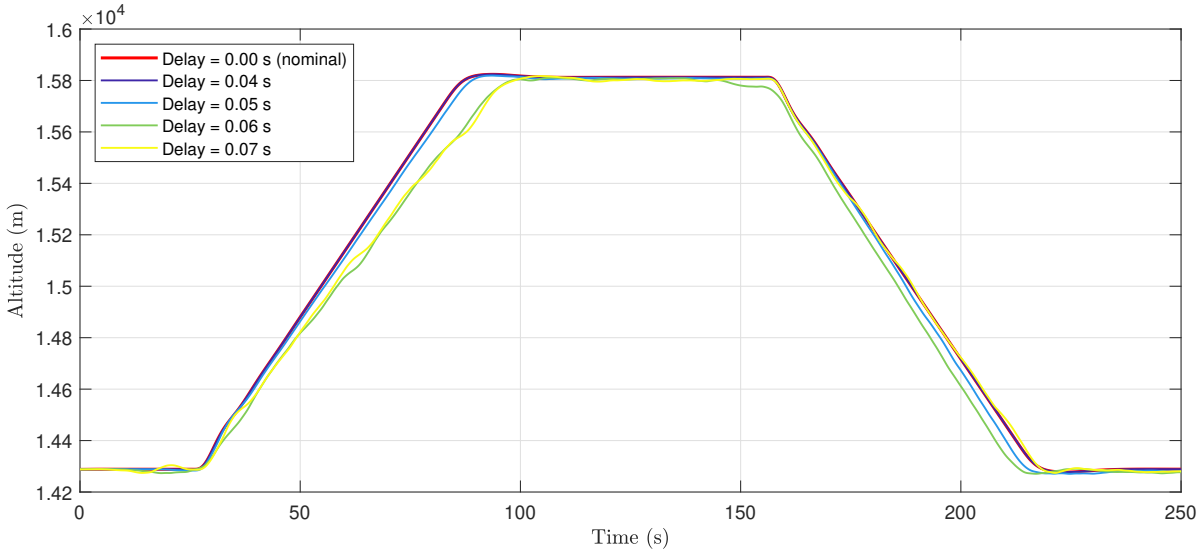


Figure 6.37: Effect of increasing sensor delay on altitude tracking without sensor delay synchronization.

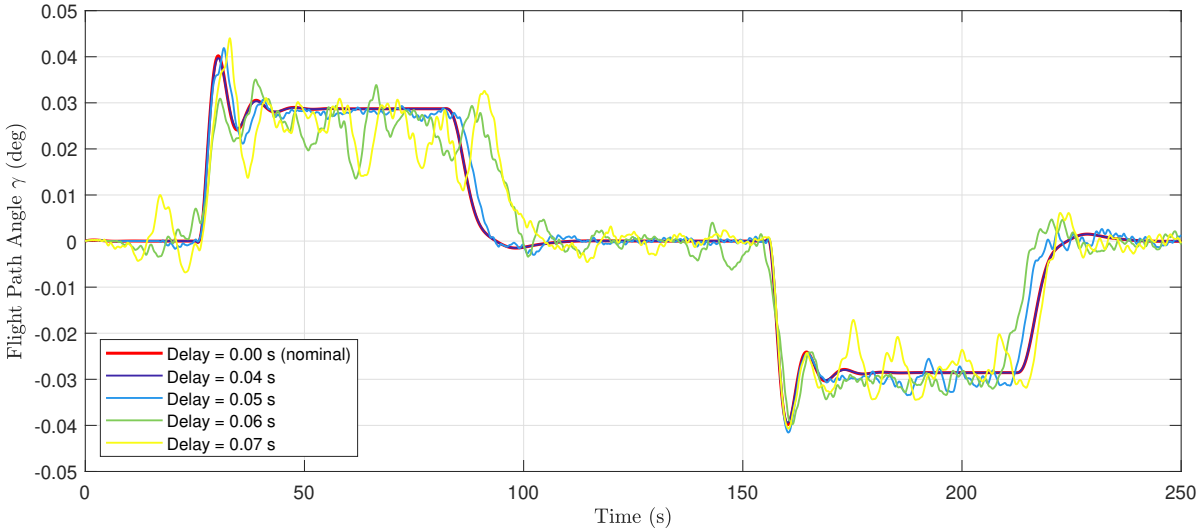


Figure 6.38: Effect of increasing sensor delay on flight-path angle without sensor delay synchronization.

The effect of sensor delay is twofold. First, it causes a desynchronization between the angular acceleration and actuator paths, which directly disturbs the incremental feedback mechanism. Second, it introduces a pure delay in the angular rate feedback loop, reducing both the gain and phase margins. While proper synchronization can mitigate the first issue by aligning the actuator and measurement paths, it cannot eliminate the second effect.

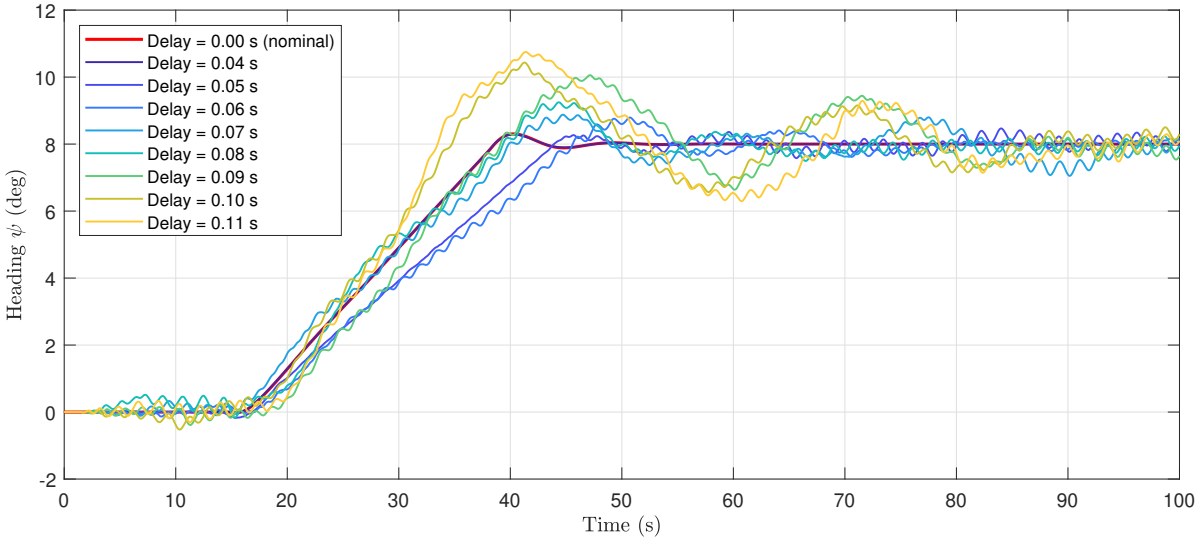


Figure 6.39: Effect of increasing sensor delay on heading response without sensor delay synchronization.

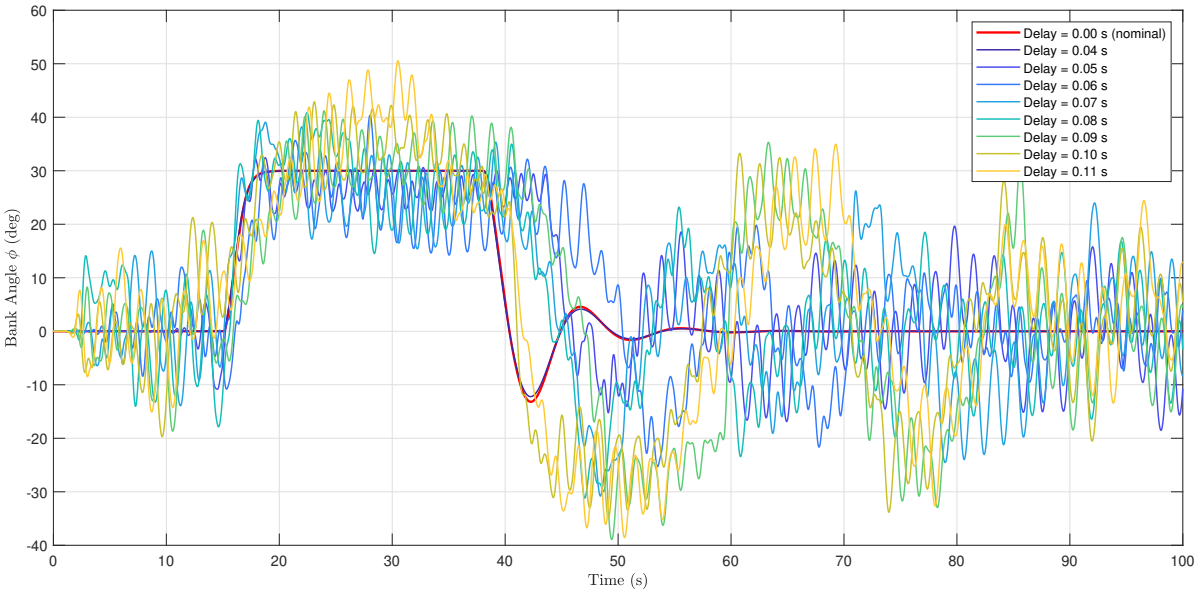


Figure 6.40: Effect of increasing sensor delay on bank angle response without sensor delay synchronization.

The controller maintains accurate tracking up to a sensor delay of approximately 0.04 s. Beyond this point, the response begins to deteriorate, as evident from Fig. 6.37. Although the altitude error remains moderate, the inner control loops exhibit growing oscillations that indicate the onset of instability, as shown in Fig. 6.38. Once the delay exceeds 0.04 s, the closed-loop dynamics degrade rapidly and eventually the simulation automatically terminates at a delay of 0.08 s when the angle of attack reaches 21°, which lies outside the aerodynamic data set.

A similar trend appears in the heading and bank angle responses shown in Fig. 6.39 and Fig. 6.40. Oscillations develop once the sensor delay exceeds approximately 0.04 s, matching the onset observed in the altitude and flight path responses. However, the simulation remains stable for a longer duration and only terminates at 0.12 s. This delayed onset of instability is likely caused by the nonlinear effect introduced by the bank angle limitation, which constrains the commanded bank to within ±30°. Simulations performed without this constraint diverge earlier, at a delay of approximately 0.07 s. INDI shows its major limitation when not designing for delay margins.

Synchronized Time Delay

When sensor delay synchronization is not applied, the effective delay margin is approximately 0.04 s, which is considerably lower than the values reported in Table 5.5. In practical implementations, sensor delay synchronization is therefore essential to compensate for known timing offsets between the measurement and actuation paths. The principle is to introduce an equivalent delay in the actuator or reference signal so that both signals are temporally aligned when processed by the controller. This alignment ensures that the control law operates on consistent information despite the presence of sensor latency. The following analysis presents the results obtained with synchronized sensor delay.

The synchronized case for the altitude command is shown in Fig. 6.41 and Fig. 6.42. With synchronization enabled, the controller tolerates sensor delays perfectly up to approximately 0.13 s, which is more than twice the limit observed without synchronization, before any degradation occurs. At a delay of 0.14 s, sustained oscillations appear that drive the angle of attack beyond the valid aerodynamic range, terminating the simulation. The system is therefore close to divergence. This behavior occurs because the pure delay continues to erode the gain and phase margins in the angular rate loop. As expected, the critical delay of 0.14 s aligns closely with the delay margin calculated for the rate loop in Table 5.5. Synchronization therefore extends the stable delay range to the theoretical limit given by the delay margins for the longitudinal case.

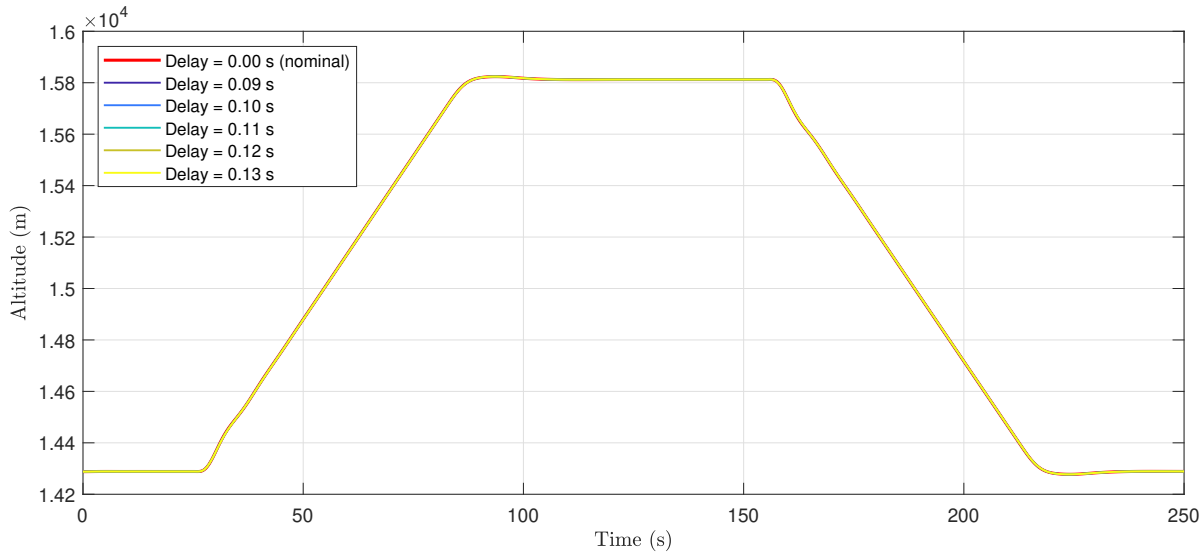


Figure 6.41: Effect of increasing sensor delay on altitude tracking with sensor delay synchronization.

For the heading command case shown in Fig. 6.43 and Fig. 6.44, the response differs notably from the longitudinal case. Oscillations appear at a sensor delay of about 0.08 s, much earlier than predicted by the calculated delay margin. This indicates that the lateral–directional loop is considerably more sensitive to delay. Beyond this point, the system still tracks the heading command but with large oscillations. The apparent stability up to around 0.16 s results mainly from nonlinear effects, particularly the roll

command limit of ±30°, which prevents full divergence. Without this limit, the simulation would fail at smaller delays.

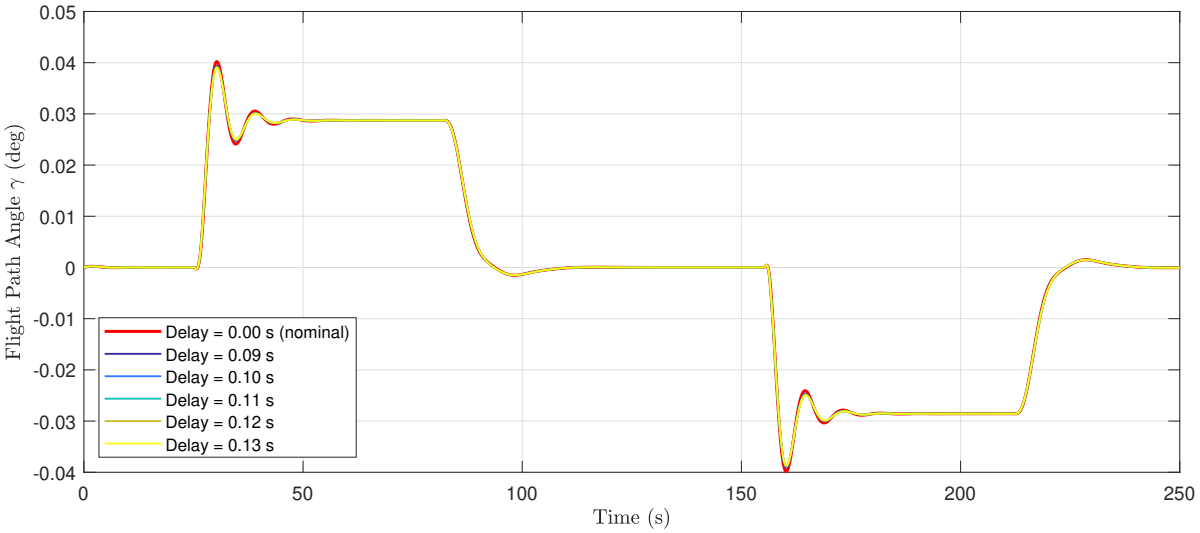


Figure 6.42: Effect of increasing sensor delay on flight-path angle tracking with sensor delay synchronization.

The early onset of oscillations indicates that a core assumption of the INDI formulation may not be fully satisfied for the lateral–directional dynamics of the GHAME vehicle. Examination of the airframe provides some clues. The GHAME's slender body, low–aspect–ratio wings, and tight wing–body integration yield an unusually low roll moment of inertia, producing exceptionally fast roll dynamics. As a result, the lateral states evolve on time scales close to those of the control inputs rather than being clearly separated as assumed in the incremental formulation. Under these conditions, the system departs from the ideal chain–of–integrators behavior on which INDI relies. The measured acceleration increment is then affected not only by the control input but also by the natural evolution of the states, reducing the validity of the quasi–static mapping between input and acceleration. This coupling introduces additional phase lag in the feedback path and plausibly explains the premature oscillations observed in simulation.

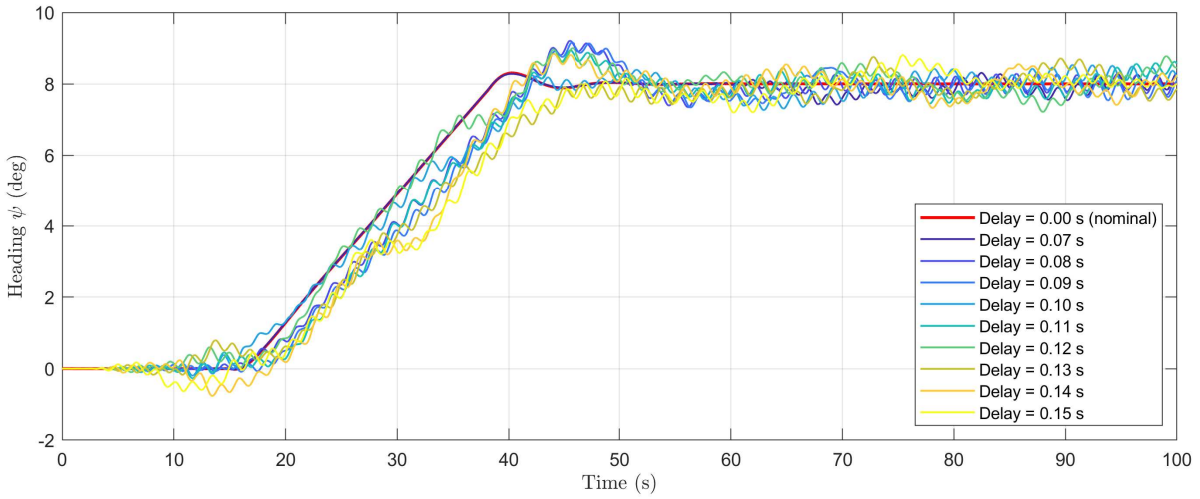


Figure 6.43: Effect of increasing sensor delay on heading response with sensor delay synchronization.

Because the integrator assumption is not fully satisfied, the linear model used for design no longer provides an accurate representation of the system. Consequently, gain and phase margins derived from the linearized model are not predictive of the actual closed–loop behavior. Retuning the controller gains can slow the response and give the impression of increased robustness, yet instability still occurs at delay values inconsistent with linear predictions. The discrepancy is therefore structural rather than tuning related.

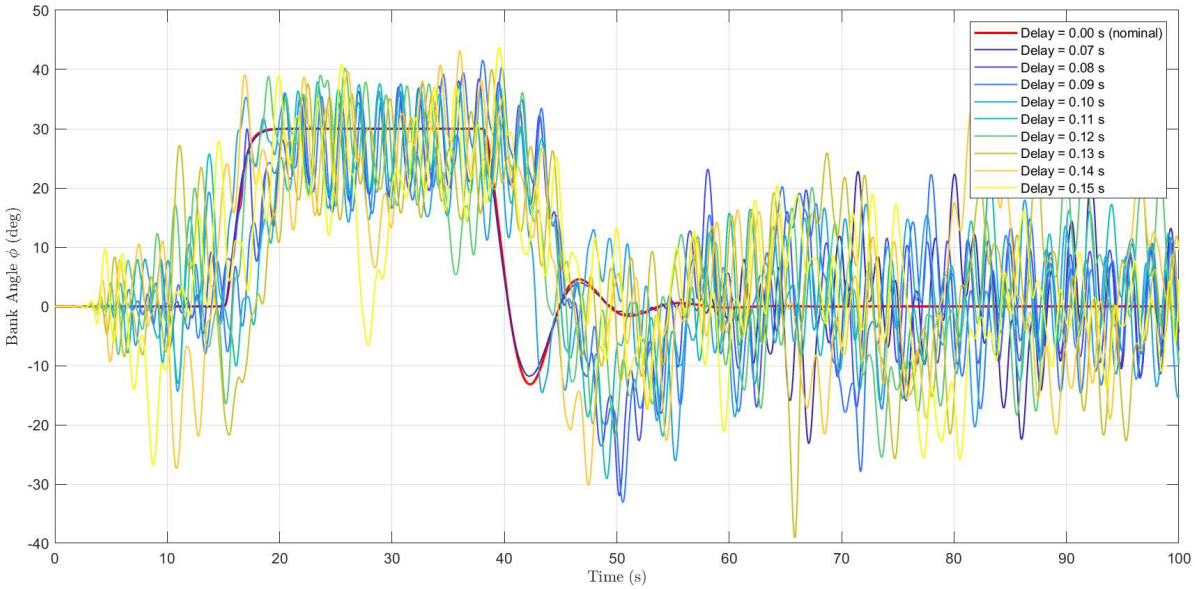


Figure 6.44: Effect of increasing sensor delay on bank angle response with sensor delay synchronization.

Nonlinear effects such as the roll–command limit can temporarily slow the lateral dynamics and delay the onset of oscillations. However, this behavior arises from nonlinear saturation rather than from a restoration of the quasi–integrator structure assumed by INDI. As such, the apparent stability does not imply that the underlying assumptions of the incremental formulation are recovered. A more consistent improvement would be expected from physical or implementation-level changes that increase the separation between control and airframe dynamics. Increasing actuator bandwidth or sampling frequency strengthens the influence of the control input within each update interval, thereby helping to reestablish the quasi-static relationship between control input and acceleration increment on which INDI depends.

Taken together, these findings indicate that the longitudinal and lateral dynamics of the GHAME vehicle behave fundamentally differently with respect to the INDI assumptions. The longitudinal motion satisfies the time-scale separation assumed by INDI, allowing accurate inversion and stable control performance. In contrast, the lateral–directional motion does not. The combination of low roll inertia and fast lateral response prevents the system from behaving as an ideal chain of integrators, making the incremental inversion only approximate. As a result, linear control design and classical stability margins lose validity for this axis. At present, it is therefore not recommended to apply INDI in this form to control the lateral–directional dynamics. Only physical or hardware-level modifications, such as faster actuators, higher sampling rates, or an altered vehicle design that slows the lateral dynamics, could restore the conditions required for accurate inversion.

7

Simulink Details

This chapter documents the simulation framework developed for this thesis, with the aim of providing sufficient detail for future users to understand, modify, and extend the models. Section 7.1 describes the script files required to run the simulations, while Sec. 7.2 presents the structure of the nonlinear Simulink model, explaining the purpose and implementation of each module.

7.1. Scripts

The GHAME model implementation in MATLAB and Simulink is supported by a set of scripts and functions that organize, initialize, and execute the simulations. To help future users understand the workflow, this section provides an overview of the main files contained in the simulation directory. The directory consists of ten primary scripts together with one supporting folder, summarized below:

1. `Main.m`: the master script that organizes and runs the full simulation workflow. It sets up the environment, adds the required helper paths, loads the aeropropulsive data through `Load_Aero_Data`, and defines the function handles used throughout the model. It computes the initial condition with `Initialize_Simulation` and prints the mission description for verification. The steady-state trim is then obtained through `Trim_Run_jj`, after which `Reinitialize_Simulation` updates the initial states from the trimmed angle of attack and pitch angle. Command inputs are built with `Build_Command_Profile`, the controllers are synthesized through `ComputeControllers`, and the actuator and filter models are constructed with `Build_Actuator_Model`. If enabled, parameter uncertainty is generated through `Create_Uncertainty`. The nonlinear Simulink model is then executed through `Run_Simulations`. Finally, the outputs are collected, the results directory and filename suffix are set according to the active modeling options, and the responses are plotted through `Plot_Heading` or `Plot_Responses` depending on the maneuver. All helper functions in the `Helper_Functions` folder are required for execution.

2. `Initialize_Simulation.m`: defines the initial conditions and configuration settings for a simulation run. It assigns all relevant constants, parameters, and toggles directly into the base workspace for use in Simulink and supporting scripts.

3. `Initial_Condition.m`: computes the initial state variables of the nonlinear aircraft model from the specified inputs (Earth model, flight condition, attitude, position, and mass properties) and returns them in the structure `IC`.

4. `Load_Aero_Data.m`: loads aerodynamic coefficient tables and engine performance data into the structured output `Data`. It provides lookup tables for lift, drag, side force, and moment coefficients

as functions of angle of attack, Mach number, and control deflections, as well as lookup tables for engine capture area and specific impulse data.

5. `Trim_Run_jj.m`: performs the trim computation for the Simulink model using the trimming function. It sets the state, input, and output constraints, runs the optimizer, and extracts the trimmed quantities together with the operating point report. The function also includes options to linearize the trimmed model for subsequent analysis.

6. `Reinitialize_Simulation.m`: updates the initial conditions based on the trimmed angle of attack and pitch angle. After the trim solution is obtained, it recomputes the dependent initial states so that the simulation starts from the correct trimmed operating point.

7. `Helper_Functions`: this folder contains the key functions required to initialize and run the simulation, together with several minor routines used internally. Important functions include `InitializeSimulation.m`, `InitialCondition.m`, and `loadAeroData.m`, which set simulation parameters, compute initial states, and provide aerodynamic and propulsion data, respectively. In addition, the folder contains utility functions such as coordinate transformations (`cad_tdi84.m`, `cad_tgi84.m`, `cad_in_geo84.m`), and inertial navigation initialization (`init_ins.m`).

8. `ComputeControllers.m`: the main controller script for controller synthesis files and their `.mat` outputs: the Simulink file used by the script above containing the FCS layout and the linear models. a partial function of the optimization process to compute optimal ω and ζ for reference tracking. It calculates the weighted sum of the squared errors.

9. `Build_Actuator_Model.m`: constructs the actuator and filter models for the simulation. It returns two outputs: a struct `Act` containing the discrete-time transfer function coefficients used in the discrete simulation, and a struct `Filters` containing the continuous-time transfer functions (sensor dynamics, noise filter, and sensor delays) required for the simulation.

10. `GHAME_Trim.slx`: used to trim and linearize the airframe. It is a copy of the nonlinear model with the actuator, control, and INS subsystems removed. The kinematics subsystem is also modified to use Euler integration instead of DCM integration.

11. `GHAME_Control.slx`: the full nonlinear GHAME model, including the control, actuator, and INS subsystems, together with the disturbance, noise, and perturbation models.

To summarize, files 1–7 are required to run the trimmed GHAME model. Files 8–9 are used for flight control system design and controller synthesis, while files 10–11 are the Simulink model implementations.

7.2. Modeling

The trim model `Trim_Version.slx` is a simplified Simulink representation of the full non-linear model, used exclusively for computing steady-state operating points. Unlike the full nonlinear model, it contains only the bare airframe dynamics needed for equilibrium calculations. All actuator dynamics, the INS, and the flight control system are omitted, since they play no role in determining the steady trim condition and would only add unnecessary complexity.

Figure 7.1 shows the layout of the trim model, which consists only of the bare airframe dynamics without actuators, INS, or controller. On the left, the control inputs, aileron (δ_a), elevator (δ_e), rudder (δ_r), and throttle, are bundled into an input bus and passed to the nonlinear airframe block (center). The airframe computes the resulting angular rates, accelerations, and flight path quantities, which are collected in the output bus (right). These outputs are then provided to the MATLAB trimming routine, which uses them to solve for the control inputs that yield an equilibrium state.

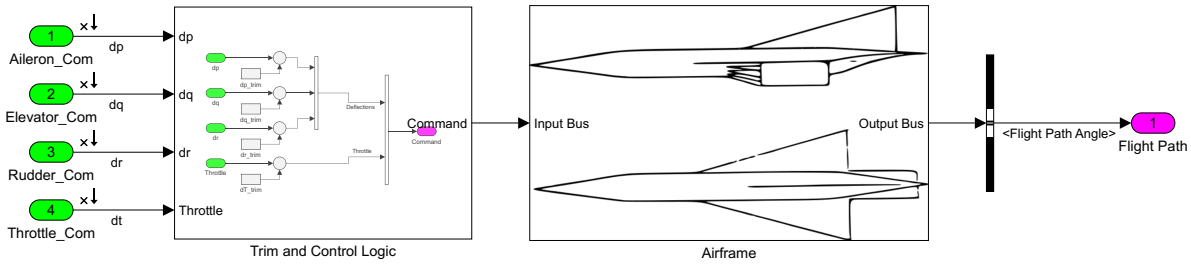


Figure 7.1: Block diagram of the trim model.

Full non-linear model

The overall layout of the GHAME nonlinear Simulink model, illustrated in Fig. 7.2, follows the standard structure of a closed-loop flight dynamics simulation. The Control block generates pilot or guidance commands based on information from the INS and the Force Bus. These commands are passed to the Actuator block, which models the control effector dynamics and outputs the corresponding surface deflections. The deflections and throttle setting are then applied to the GHAME Airframe block, which implements the vehicle's rigid-body dynamics as described in the previous sections.

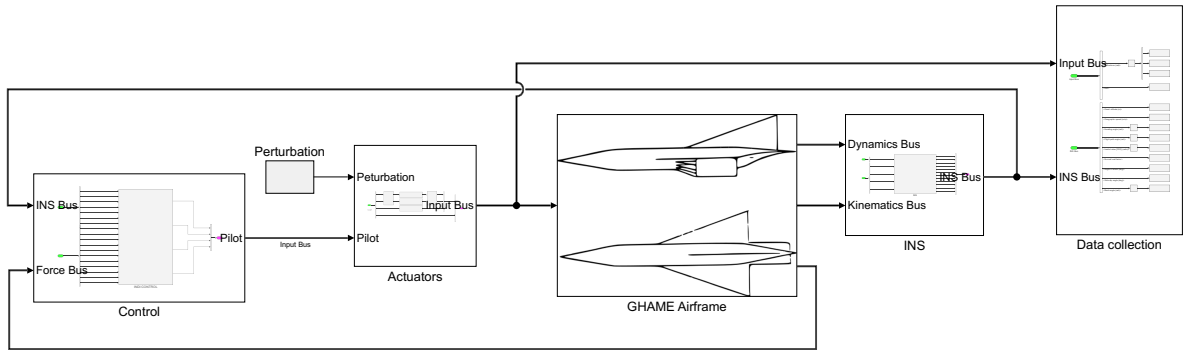


Figure 7.2: Overall layout of the GHAME nonlinear Simulink model. Includes the control, actuator, GHAME airframe and the INS subsystems.

The airframe outputs its updated state through the Dynamics, Kinematics, and Force buses. These signals are processed by the INS block to provide navigation measurements, thereby closing the feedback loop with the control system. The Force Bus additionally supplies the aerodynamic and propulsive force coefficients required by the controller for nonlinear dynamic inversion.

Airframe System:

The overall GHAME Airframe subsystem, illustrated in Fig. 7.3, represents the rigid-body dynamics of the vehicle. It is organized into four main components. The Environment block provides gravitational, atmospheric, and wind quantities based on the current state. The Kinematics block computes the vehicle orientation and transformation matrices. The Forces and Moments block evaluates aerodynamic and propulsive forces and moments as functions of the vehicle state, control inputs, and environmental conditions. Finally, the Dynamics block integrates the Newton–Euler equations of motion to update the vehicle state. Data exchange between subsystems is managed through the Input, Dynamics, Environment, Force, and Kinematics buses.

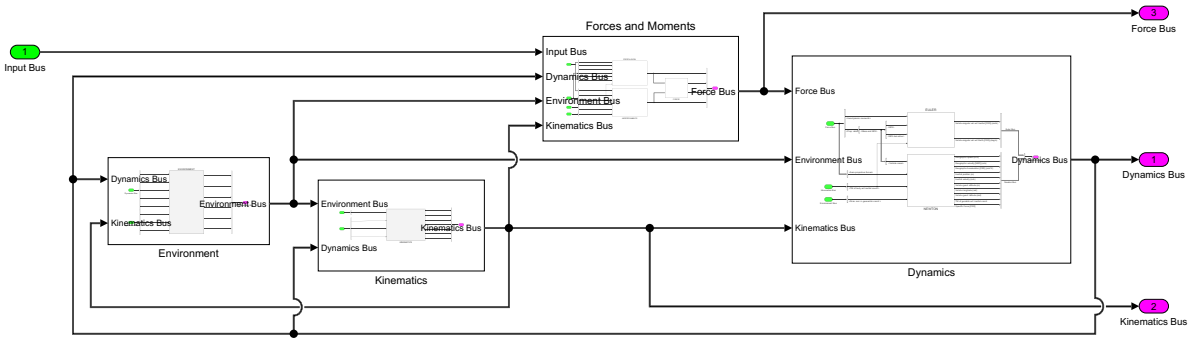


Figure 7.3: Top-level structure of the GHAME Airframe subsystem. It contains the environment, kinematics, forces & moments and the dynamics subsystems.

Kinematics Module:

The subsystem shown in Fig. 7.4 updates the transformation matrix $[T]^{BI}$, which defines the vehicle's orientation relative to the inertial coordinate system and serves as the DCM. The matrix is propagated using the body angular velocity $[\omega^{BI}]^B$ and orthonormalized at each integration step to prevent numerical drift and maintain a valid rotation matrix.

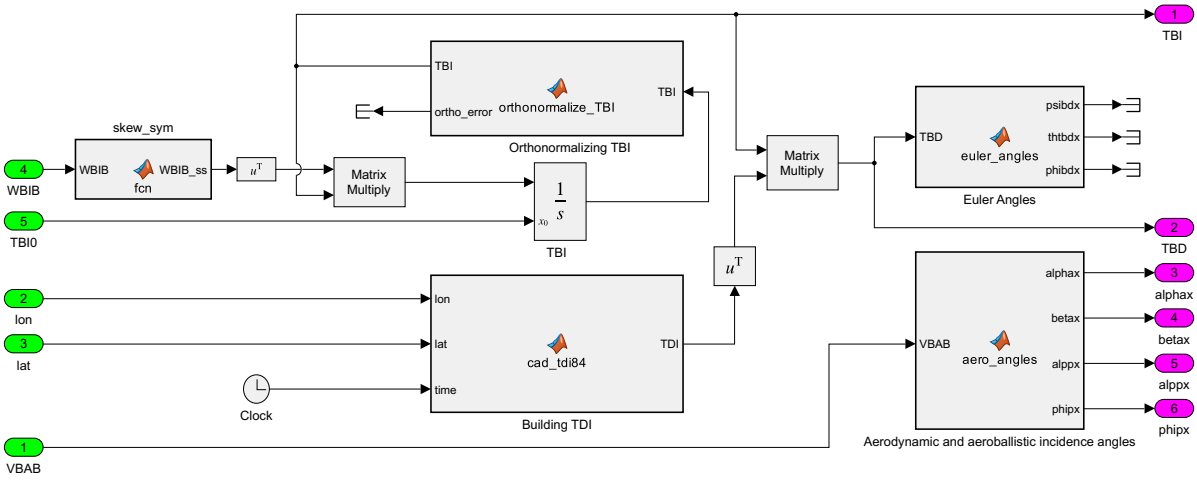


Figure 7.4: Kinematics subsystem responsible for computing the body-to-inertial transformation matrix $[T]^{BI}$.

Along with the update of $[T]^{BI}$, the subsystem provides three sets of angles. The first set consists of the geodetic Euler angles ψ_{bd}, θ_{bd}, and ϕ_{bd}, obtained from $[T]^{BD}$. The second set contains the aerodynamic incidence angles: the angle of attack α_A and the sideslip angle β_A. The third set contains the aero-ballistic incidence angles: the total angle of attack α_p and the aerodynamic roll angle ϕ_p. These quantities are derived from the velocity of the body relative to the air, $[v_B^A]^B$.

Dynamics Module:

The subsystem shown in Fig. 7.5 represents the core of the vehicle dynamics model. It consists of the Euler subsystem, which integrates the rotational dynamics, and the Newton subsystem, which integrates the translational dynamics. Together, these components solve the full 6DoF equations of motion and provide the body angular rates, attitude, position, and velocity to the other modules through the Dynamics Bus.

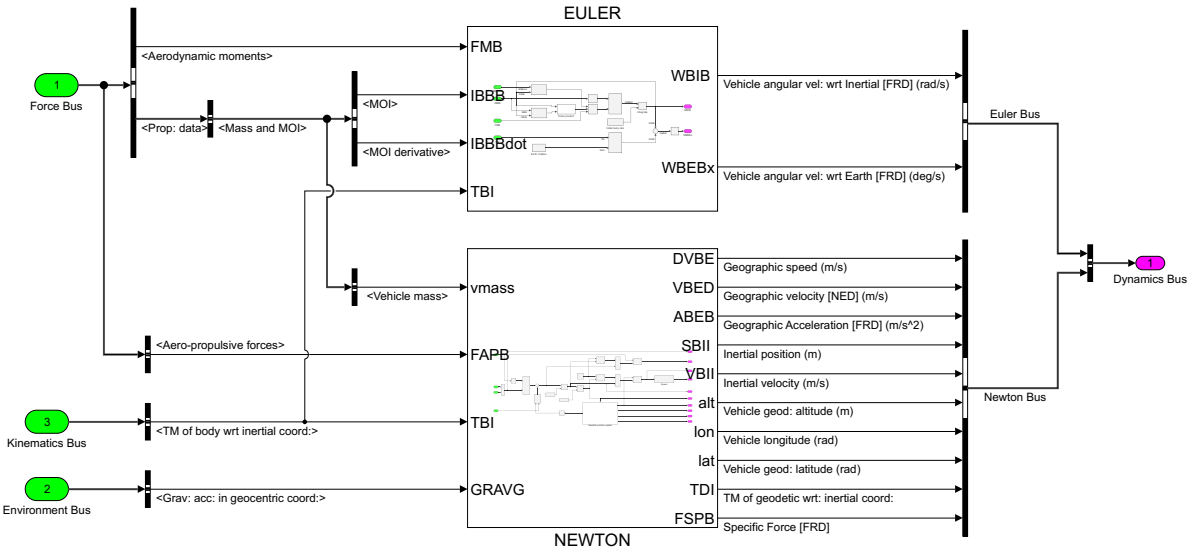


Figure 7.5: Internal structure of the Dynamics subsystem. It contains the Euler dynamics and the Newton dynamics as separate subsystem.

The Newton subsystem shown in Fig. 7.6 integrates the translational dynamics of the vehicle. Using the current mass, the external forces expressed in the body coordinate system, and the gravitational acceleration in geocentric coordinates, it computes the inertial position $[s_B^I]^I$ and inertial velocity $[v_B^I]^I$. From these, altitude, longitude, latitude, and geographic velocity are derived. Additional navigation quantities, including geographic speed, flight path angle, heading angle, and load factors[1], are also determined.

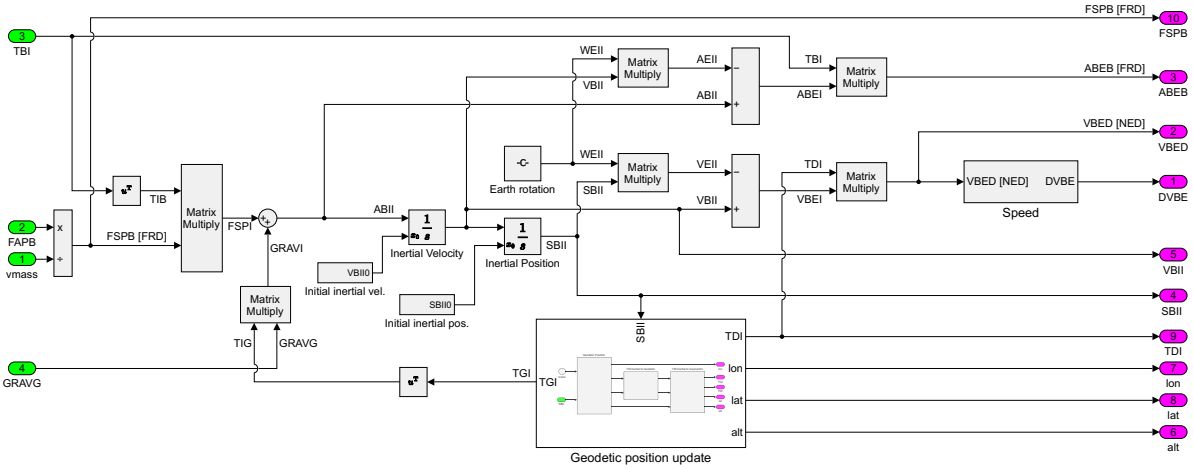


Figure 7.6: Newton subsystem for translational dynamics. It computes accelerations from the applied forces, integrates inertial velocity and position, and updates the geodetic position, flight path angles, and load factors.

[1]Alternative coordinate systems are explicitly distinguished with bracketed notation. The body coordinate system is defined as FRD, but the Forward–Right–Up (FRU) convention is used for load factors so that n_z is positive under upward acceleration. Similarly, the geodetic coordinate system is defined as NED, whereas the North–East–Up (NEU) system is used for position so that altitude remains positive.

A dedicated sub-block within the Newton subsystem updates the geodetic position, as shown in Fig. 7.7. This subsystem takes the inertial position vector $[s_B^I]^I$ and the simulation time as inputs and computes the geodetic latitude, geodetic altitude, the longitude, and the celestial longitude. It also evaluates the transformation matrices $[T]^{DI}$ and $[T]^{GI}$, which are required for converting between inertial, geodetic and geocentric coordinates.

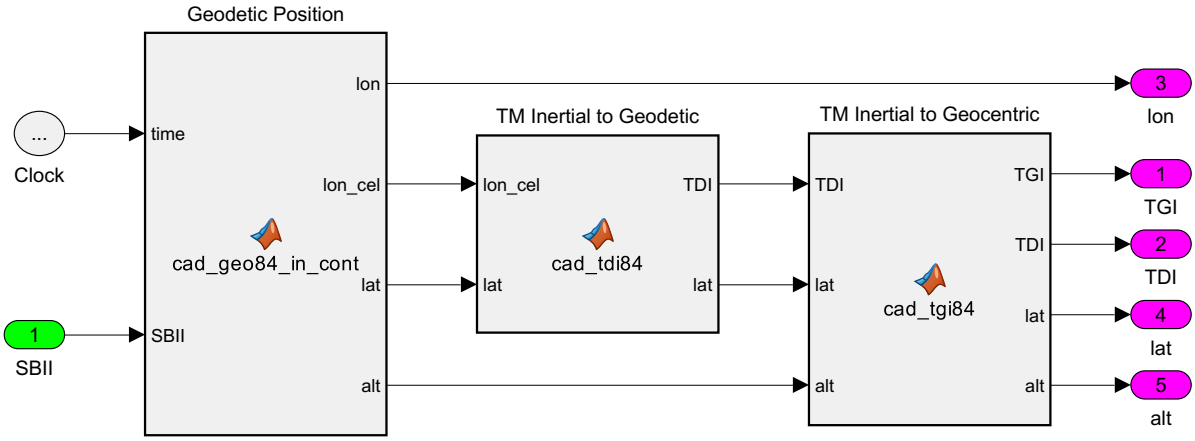


Figure 7.7: Geodetic position update sub-block.

The Euler subsystem shown in Fig. 7.8 integrates the vehicle's rotational dynamics. It receives the aerodynamic and propulsive moments, the inertia tensor, and its time derivatives, and uses these inputs to compute the body angular acceleration, which is integrated to obtain the body angular velocity $[\omega^{BI}]^B$. The subsystem also receives the transformation matrix $[T]^{BI}$, which is required to convert the Earth rotation rate $[\omega^{EI}]^I$ into body coordinates. The transformed quantity $[\omega^{EI}]^B$ is then subtracted from $[\omega^{BI}]^B$ to obtain the angular velocity relative to the rotating Earth $[\omega^{BE}]^B$ for use by other modules.

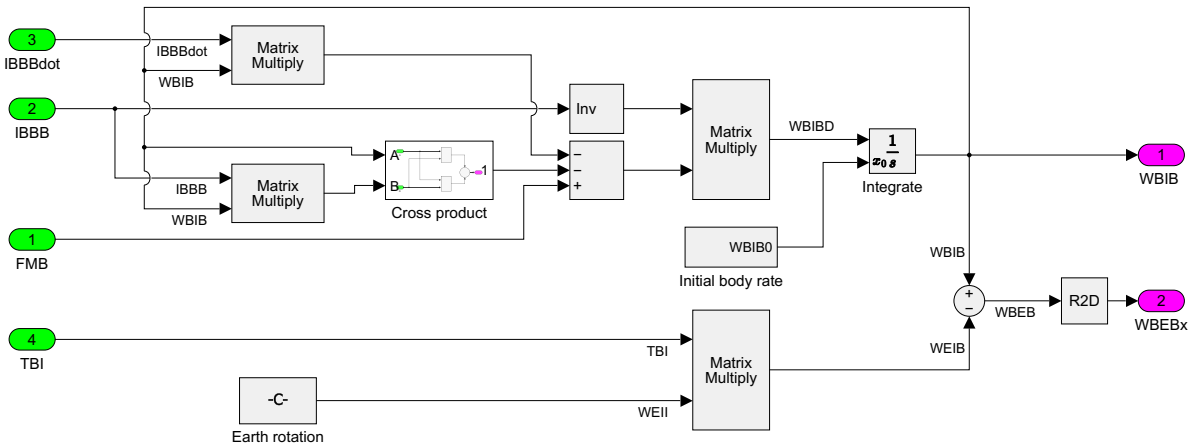


Figure 7.8: Euler subsystem for rotational dynamics.

Environment Module:

The subsystem shown in Fig. 7.9 defines the external conditions acting on the vehicle. It consists of three main components: the gravity model, the atmospheric model, and the wind and turbulence block. The gravity model computes the geocentric gravity vector $[g]^G$ from the inertial position $[s_{BI}]^I$. The atmospheric model determines altitude-dependent properties, including air density ρ, static pressure, temperature, and the local speed of sound. From these quantities, the dynamic pressure $\bar{q}$ and Mach number M are obtained, which are subsequently used by the aerodynamic and propulsion models. The wind and turbulence block introduces environmental disturbances by superimposing horizontal wind and stochastic turbulence on the geographic velocity to obtain the air-relative velocity vector.

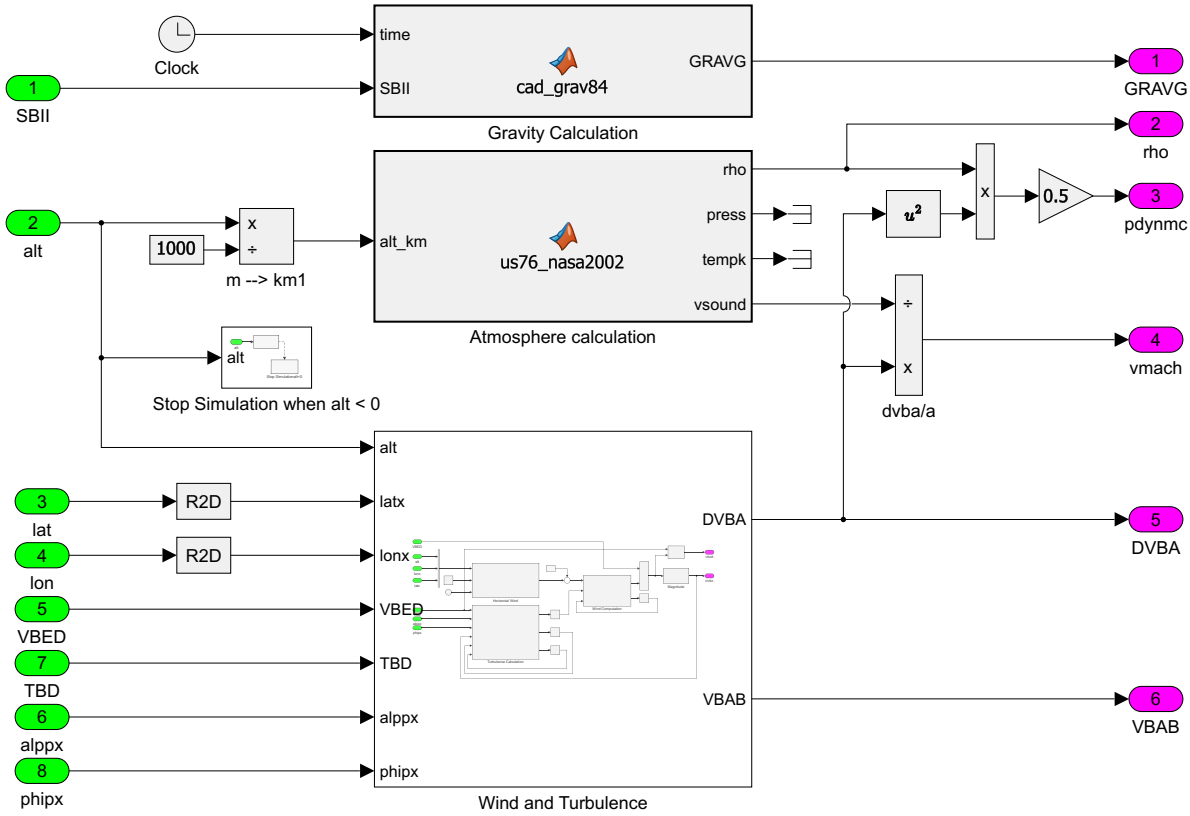


Figure 7.9: Environment subsystem responsible for gravity, atmospheric properties, and wind and turbulence modeling.

Finally, the wind and turbulence block in Fig. 7.10 models the wind velocity in geodetic coordinates $[v_A^E]^D$ and combines it with the vehicle velocity in geodetic coordinates $[v_B^E]^D$. The HWM14 model provides the mean wind components and the Dryden model adds turbulence, which together yield the wind velocity $[v_A^E]^D$. This is then used to compute the true airspeed DVBA and the vehicle velocity relative to air $[v_B^A]^B$.

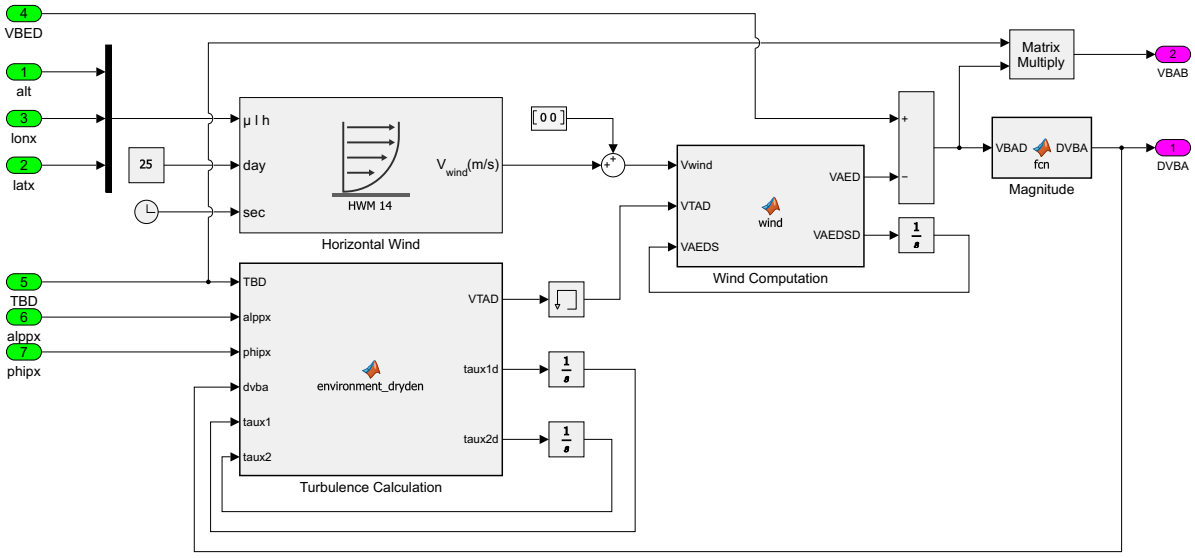


Figure 7.10: Wind and turbulence modeling within the environment subsystem.

Forces Module:

The subsystem shown in Fig. 7.11 combines the aerodynamic and propulsion models to compute the total forces and moments acting on the vehicle. It determines the aerodynamic and propulsive force and moment coefficients, assembles the corresponding vectors, and provides these together with supporting aerodynamic and propulsion data. The resulting outputs supply both the equations of motion and the control system with the required aero-propulsive information.

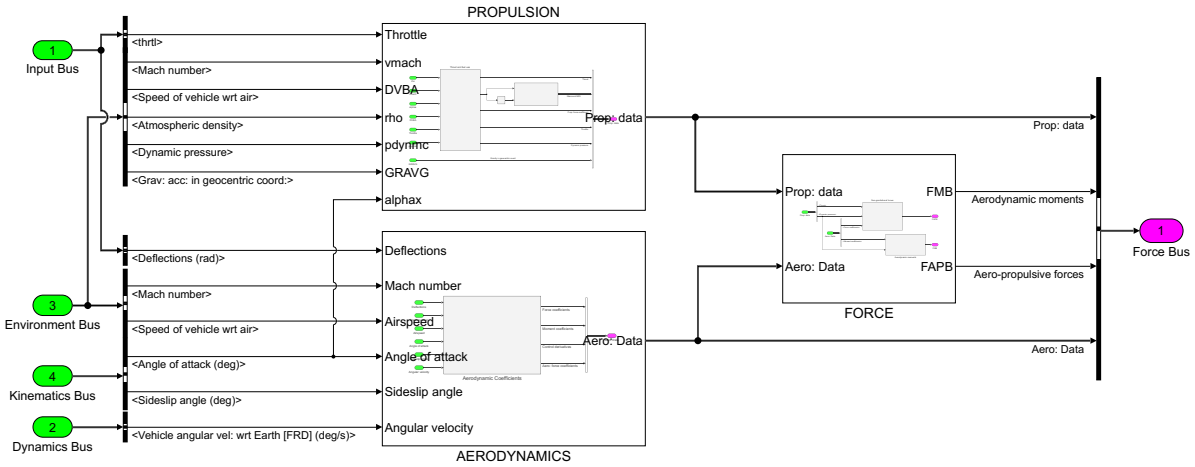


Figure 7.11: Forces module, including propulsion, aerodynamic coefficients, and force and moment calculation.

Within the Forces subsystem, the propulsion block shown in Fig. 7.12 models thrust generation, fuel consumption, and the resulting updates to vehicle mass and inertia. The thrust model evaluates the propulsive force as a function of Mach number, angle of attack, throttle setting, and atmospheric conditions. From the computed thrust, the fuel mass flow rate is determined and integrated to track the remaining fuel. The mass and inertia update block adjusts the total vehicle mass and interpolates the inertia tensor between the full- and empty-fuel conditions. It also provides the time derivative of the inertia tensor, which is required in the rotational dynamics. A stop flag is raised once the fuel is depleted so that the simulation terminates when no fuel remains.

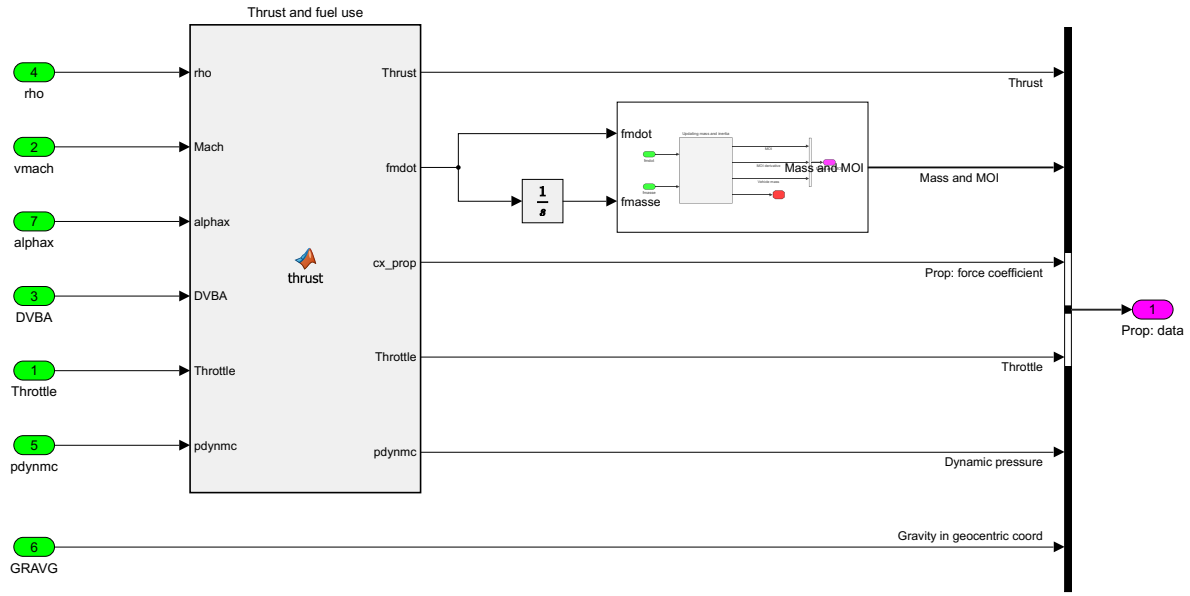


Figure 7.12: Propulsion subsystem, including thrust and thrust-coefficient calculation, fuel-mass depletion, mass and inertia updates, and the stop flag.

The assembly of forces and moments is handled in a dedicated block, shown in Fig. 7.13. Aerodynamic and propulsive forces are computed from aerodynamic coefficients, dynamic pressure, and thrust, while aerodynamic moment coefficients yield rolling, pitching, and yawing moments using the same parameters and reference lengths. These contributions combine into the total aero-propulsive force vector $[f_{a,p}]^B$ and moment vector $[m_B]^B$, expressed in the body coordinate system, and are passed to the dynamics subsystem through the force bus for integration of the equations of motion.

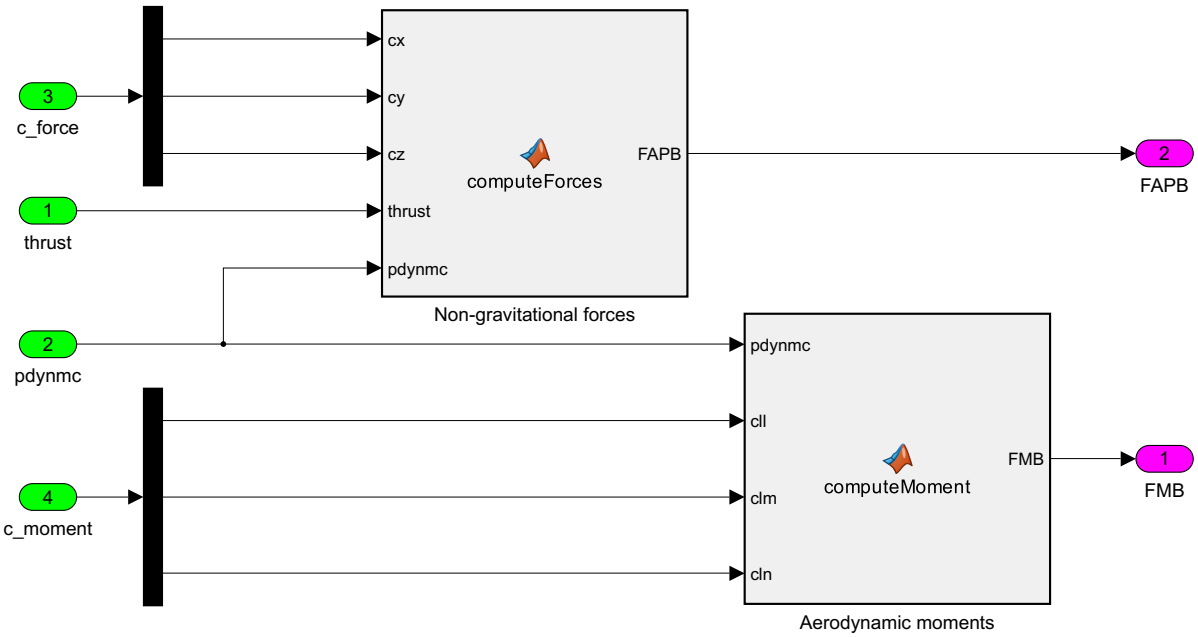


Figure 7.13: Force and moments assembly subsystem, computing the aero-propulsive forces $[f_{a,p}]^B$ and aerodynamic moments $[m_B]^B$.

Actuator Module:

The subsystem shown in Fig. 7.14 maps pilot or control system inputs to the control surface deflections used by the airframe. It splits aileron and elevator commands into left and right elevon deflections to distribute roll and pitch inputs across the control surfaces. Each elevon channel, together with the rudder command, passes through a second-order nonlinear actuator model that accounts for rate and position limits. After the actuator dynamics, the elevon deflections recombine to yield effective elevator and aileron deflections, while the rudder remains a separate channel. The throttle command passes directly through the subsystem, and perturbations are superimposed on the right elevon deflection to enable control system testing in the nonlinear simulation.

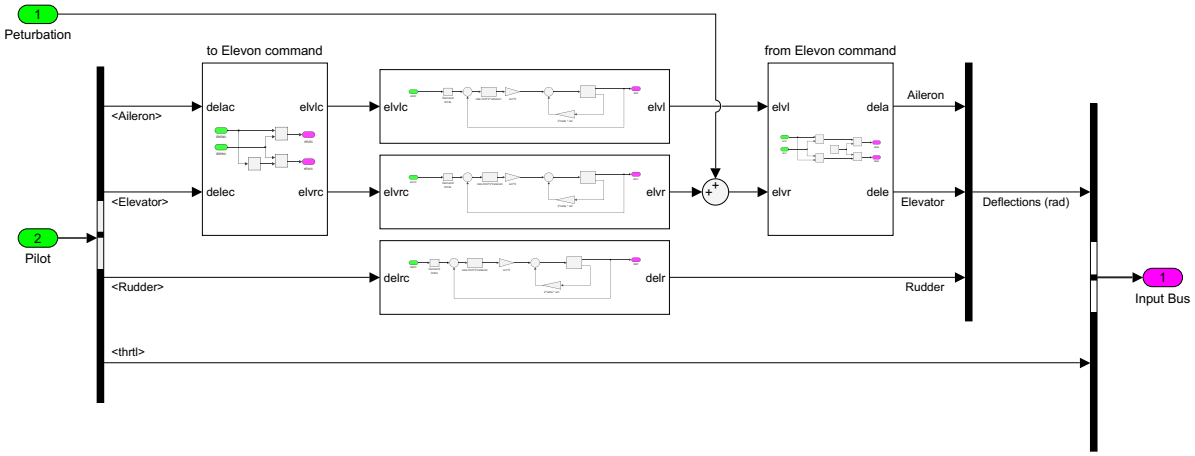


Figure 7.14: Actuator subsystem, including elevon control allocation, second-order nonlinear actuator dynamics, recombination of effective control deflections and actuator perturbations.

8

Conclusions

This research was carried out using the GHAME vehicle as a case study, which is of particular interest because its aerodynamic dataset is derived from flight test data. A central part of the work was the redevelopment and extension of the GHAME simulation model to increase its fidelity by incorporating several physical effects that had not been included previously. On this foundation, controllers based on NDI and INDI were designed and implemented at different levels of the control architecture. Their performance was assessed through nonlinear time–domain simulations, with emphasis on stability and tracking capability across hypersonic flight conditions. The main findings are discussed in Sec. 8.1, where the research questions are revisited in light of the results, and Sec. 8.2 reflects on the overall objectives of the study.

8.1. Revisiting the Research Questions

1. **How can the challenges and limitations of GHAME's current nonlinear Simulink model be addressed, thereby creating a higher fidelity simulation?**

   - **What are the key limitations of the existing Simulink GHAME model implementation?**

     The model from Goz [2024] provided a useful basis but relied on simplifications that were acceptable in the subsonic regime yet break down at hypersonic speeds: first-order actuators, constant gravity, and a flat, non-rotating Earth, which neglect variations in apparent gravity, inertial acceleration, and trajectory curvature. Mass variation was omitted, despite hypersonic vehicles losing more than half their mass in flight, and the atmosphere excluded wind and turbulence. Sensor dynamics and measurement errors were also absent, preventing realistic navigation behavior. In addition, only longitudinal motion was verified, leaving full 6DOF behavior and cross-coupling untested. Finally, the trimming algorithm did not converge reliably beyond Mach 1, restricting the model to subsonic conditions and preventing its use as a baseline for hypersonic FCS development.

   - **In what ways can the GHAME model be refined to achieve higher physical and dynamic fidelity for hypersonic flight simulation?**

     The model was rebuilt on an elliptical, rotating Earth with a variable gravity field, time-varying mass and inertia from fuel consumption, second-order actuators with rate limits, and an atmosphere extended to 1000 km including wind and turbulence. An inertial navigation system with IMU error models was integrated to assess sensor-induced state deviations, with additional blocks for time delays and filtering. The resulting Simulink architecture, organised into four main systems coordinated by a centralised initialisation script, provides

a high-fidelity environment for evaluating nonlinear inversion–based control under coupled aerodynamic and environmental effects.

- **How was the trimming procedure redesigned to ensure reliable convergence and applicability across the supersonic and hypersonic regimes of the GHAME vehicle?**

  The original two–step approach, with separate airframe and engine models, often failed to converge at higher Mach numbers. The redesigned procedure combines airframe and propulsion dynamics into a single framework and formulates the trim as a one–step steady-state solution, solving simultaneously for throttle and control deflections. This removes model switching and provides reliable convergence across the supersonic and hypersonic regimes. The trim enforces force and moment equilibrium at a specified speed V, altitude h and mass ratio, with ϕ and the angular rates p, q, r set to zero for symmetric flight and the yaw angle left free. The unknowns are the throttle, elevator, rudder, and aileron deflections together with α, β, and θ; force and moment balance provide six equations, closed by the level-flight condition $\gamma = \theta - \alpha = 0$ (see Section 4.6).

- **How can the correct implementation and expected behavior of the nonlinear GHAME model in Simulink be verified?**

  Verification followed a structured bottom-up process (Sec. 4.7). Each subsystem was tested in isolation against the reference C++ implementation, with outputs compared element by element within strict tolerances. To ensure a fair comparison, the integration scheme (Heun's method reproducing the modified Euler method), fixed time step, and hard-coded constants were matched exactly. Once the individual modules were confirmed error-free, the full model was verified by replaying logged actuator and throttle inputs under identical conditions. The resulting trajectories matched the reference to near machine precision, with altitude differences after 100 s on the order of 10^{-7} m, giving strong confidence that the model is correctly implemented and suitable for control design.

2. **How can INDI be applied to the GHAME vehicle to evaluate its suitability for hypersonic flight control?**

   - **How was the nonlinear control architecture of the GHAME vehicle designed to combine NDI and INDI, and how were implementation aspects such as incremental feedback and filtering addressed within this structure?**

     The architecture is a cascaded system with four loops, rate, attitude, velocity/trajectory, and position, using time-scale separation to avoid full inversion (Sec. 5.1). Each loop uses NDI where the dynamics are well modeled and invertible, and INDI in regions with significant aerodynamic uncertainty, where reliance on measured state increments preserves performance as the aerodynamic coefficients vary. Splitting the system into cascaded subsystems, each with direct control authority and full relative degree, limits inversion to controllable dynamics and avoids the unstable zero dynamics of full nonlinear inversion. The control laws are derived directly from the GHAME dynamics (Sec. 5.2), capturing the actual aerodynamic behaviour and coupling rather than simplified formulations. Since the incremental formulation relies on real-time angular-acceleration estimates, a filter is required; a cutoff of 20 rad/s was chosen to balance noise reduction against disturbance rejection.

   - **How were the linear controllers designed across the different control loops?**

     Building on the inversion that transforms the dynamics into a chain of integrators, the linear controllers shape the bandwidth, damping, and steady-state performance of each loop (Sec. 5.3). Each loop is designed at least four times slower than the one below it, ensuring approximate decoupling. Actuator dynamics are included explicitly in the rate

loop; neglecting them would force a lower rate-loop bandwidth and slow the entire cascade. With this design, the idealised attitude, velocity, and position controllers achieve near-critical damping, smooth transients, and zero steady-state error. In the full cascade with a separation of four, however, the inner loops are not truly instantaneous, so the outer loops show slightly lower effective damping than designed while steady-state tracking remains exact, illustrating the trade-off between ideal behaviour and overall responsiveness.

- **How were digital implementation effects incorporated into the controller design, and what impact did they have on closed-loop stability?**

  Rather than discretising a continuous design afterward, the discrete effects were included directly in the synthesis through continuous-time approximations (Sec. 5.4). The anti-aliasing filter $G_a(s)$, sample-and-hold $G_{SH}(s)$, and computation delay $G_D(s)$ were combined with the plant, with third-order Padé approximations used for the exponential terms. Applied to the rate loop, these effects added about 16.5° of phase lag, reducing the phase margin from 67.6° to 51.1° and raising the overshoot to 5.2%, violating the zero-overshoot requirement. The controllers were therefore reoptimised within the modified continuous-time framework, restoring the desired margins and ensuring the intended performance in discrete time.

3. **To what extent does the digitally implemented NDI–INDI control system meet the specified performance requirements across two flight conditions under sensor delay, measurement noise, wind, turbulence, and actuator perturbations?**

   - **How is the cascaded NDI–INDI FCS implemented in discrete form within Simulink?**

     The flight control computer operates in discrete time on sampled sensor data, while the aircraft, aerodynamics, environment, and filters remain continuous, reflecting a real onboard system (Sec. 6.1). All linear controllers are discretised using the Tustin transformation to preserve frequency and phase behaviour near the controller bandwidth. Angular rates pass through a first-order anti-aliasing filter and a second-order noise filter before sampling. Actuator dynamics are combined with the synchronisation dynamics $H_{\text{sync}}(s)$ and noise filter $H_{\text{fil}}(s)$ in the continuous domain and discretised using ZOH to preserve correct timing. The worst-case computational delay is modeled by a unit delay z^{-1} on the control inputs, with a second z^{-1} on the throttle command. All controllers run at 100 Hz with a fixed step of 1×10^{-3} s.

   - **How does the cascaded NDI–INDI FCS satisfy the specified performance requirements during altitude and heading maneuvers when subjected to wind, turbulence, sensor noise, and actuator perturbations?**

     Evaluated at Mach 3 and 60,000 ft (Sec. 6.2), the system maintains precise altitude and heading tracking under nominal conditions, with altitude following the command within limits, heading tracked with less than 1° overshoot, and airspeed deviations below 1 m/s. The internal dynamics remain physically consistent, including a sub-unity load factor ($n_z \approx 0.982$) and a small steady roll rate from Earth-rotation coupling. Under measurement noise, turbulence, and wind, tracking remains good: the noise filter suppresses IMU noise and turbulence produces only minor control-surface motion. A brief heading deviation during climb and descent in wind is quickly compensated, and the steady rudder offset under crosswind matches the expected coordinated-flight correction. The INDI controller also rejects both impulsive and step actuator perturbations rapidly while keeping control motion within limits, and the response stays close to nominal under multiplicative aerodynamic uncertainty. Although limited to a single flight condition, the results show that the digital NDI–INDI controller meets the performance requirements and remains stable under realistic disturbances.

- **How does the cascaded NDI–INDI FCS satisfy the specified performance requirements during altitude and heading maneuvers when subjected to continuous-time sensor delay?**

  A continuous-time delay introduced after the anti-aliasing filter adds phase lag to the angular-rate feedback path. In the unsynchronised case, degradation begins beyond about 0.04 s, with the rate loop developing oscillations that lead to instability. When the delay is included in the synchronisation filter H_{sync}, the longitudinal axis remains stable up to about 0.15 s, consistent with the linear delay margin, but the lateral–directional axis improves far less, becoming oscillatory at about 0.08 s. This early onset indicates that the linearised model does not fully capture the nonlinear stability characteristics. The lateral–directional dynamics appear to violate the time-scale separation implicit in INDI: the combination of low roll inertia and strong lateral control effectiveness causes the airframe states to evolve on time scales similar to the actuators, making the incremental control-to-derivative mapping dynamic rather than quasi-static and adding phase lag that reduces the effective delay margin. The GHAME lateral–directional dynamics therefore only partially satisfy the conditions for ideal incremental inversion, so linear stability predictions are overly optimistic. Increasing actuator bandwidth or sampling frequency, or raising roll inertia, would restore the quasi-static relationship on which INDI relies and extend the range of stable operation without modifying the control law.

8.2. Research Objective

With the research questions answered, attention can now turn to the overarching objective to determine the extent to which it has been achieved. The objective is stated as follows:

The research objective is to enhance the fidelity of the GHAME simulation model in Simulink and to investigate, for the first time, the application of (I)NDI controllers on this dataset, with the aim of assessing their suitability for hypersonic flight control under realistic modeling assumptions.

The work presented in this study successfully achieved both aspects of this objective. The fidelity of the GHAME model has been substantially improved through the incorporation of high-resolution aerodynamic data, an updated propulsion and mass-inertia model, and a complete six-degree-of-freedom environment including geocentric gravity, rotating Earth effects, and dynamic atmospheric and wind conditions. The model has been thoroughly verified through extensive simulation testing and demonstrated numerical consistency with existing hypersonic simulation models. This verified platform now provides a good foundation for further research in hypersonic guidance and control development, both for linear and nonlinear controller.

The INDI control framework was successfully implemented for the GHAME vehicle and evaluated under two specific flight conditions and perturbations. For the longitudinal axis, INDI achieved stable and accurate tracking despite the presence of wind, turbulence, measurement noise, time delay, and significant aerodynamic uncertainty. The controller exhibited strong robustness to regular perturbations and maintained precise airspeed and altitude control across all tested conditions. Similar robustness was demonstrated for the lateral-directional axis; however, analysis revealed that the time-scale separation between realistic roll control inputs and the airframe dynamics of the GHAME configuration is insufficient to achieve ideal inversion. As a result, the system departs from the perfect integrator behavior assumed by INDI, making purely linear outer-loop design less effective for the lateral channel. Consequently, the method cannot be universally recommended for all control axes without further adaptation to account for the rapid roll dynamics characteristic of hypersonic vehicles.

Recommendations

More work can always be done to improve the results of this thesis, and several recommendations for future work follow naturally from the current research. The following is therefore addressed specifically to the next student who continues this work, with the aim of making their start easier. Some of these recommendations are outlined below.

- **Feasibility of ideal inversion:** Future work should examine under which sampling times ideal nonlinear inversion for lateral–directional control is recovered. This dependency on the sampling time is particularly interesting to analyze in conjunction with the actuators, since the interaction between sampling time and actuator dynamics directly affects inversion accuracy and performance. In this context, it should also be investigated how changing the actuators alters this interaction, so that the combined effect of sampling time and actuator characteristics can be properly characterized.

- **Advanced control methods:** A more rigorous tuning procedure should be applied to the outer control loops. The current use of time-scale separation combined with pole placement provides a straightforward design approach, but it does not fully exploit the achievable performance of the system. More systematic and advanced techniques, such as loop shaping or structured (robust) control, should be considered to improve disturbance rejection and robustness; the design need not be strictly robust, but a more advanced control system would be preferable. In addition, when the system is simplified as a chain of integrators, it should be verified that the system actually behaves in this manner. Otherwise, the controller is effectively designed for a different system than the one being controlled.

- **NDI versus INDI:** The current design combines NDI with INDI. Future work should examine whether a pure NDI design with robust outer loops might be a better option overall for this type of vehicle, as it could perform better due to the fast lateral dynamics.

- **Engine model and dynamics:** The current simulation employs a simplified combined–cycle engine model in which many effects critical to hypersonic flight are neglected. As a result, phenomena such as choking and inlet unstart are not represented. Furthermore, the model does not include any engine dynamics, which is not realistic. Future work should address both aspects: incorporating a more advanced engine model, if available, to capture these effects and provide a more realistic assessment of vehicle performance and controller behavior, and adding engine dynamics, for which a first–order transfer function should be sufficient to simulate the response of the engine.

- **Engine placement in the control structure:** The thrust setting is currently computed in the trajectory/velocity loop. This is not an ideal placement, since velocity controlled through the thrust setting acts on a faster time scale than the trajectory. It is therefore recommended to move the computation of the thrust setting out of the trajectory/velocity loop and place it one loop ahead, at the level of the attitude loop.

- **Aerodynamic data:** The aerodynamic dataset for the GHAME vehicle is fixed and cannot be altered, but the way it is modeled can be refined. In this work, a simple linear approximation was employed. In theory, higher–order polynomial representations could improve accuracy, particularly for NDI, while the effect on INDI would be less critical.

- **Elastic effects:** The current simulation assumes a rigid body, which neglects the flexible modes of the vehicle. Including these elastic effects is a monstrous task, well beyond the scope of this work and arguably a PhD–level undertaking in its own right. Nevertheless, it would improve the analysis significantly, as these effects are very important for the simulation of hypersonic vehicles due to the high dynamic excitation encountered in flight. Accounting for them would align the GHAME model more closely with modern research models, such as the Bolender model, which explicitly captures aeroelastic coupling.

- **Aerothermodynamic heating:** The current model neglects aerothermodynamic heating effects, which are critical in hypersonic flight. For simulations beyond Mach 5, such effects must be included to ensure physical realism. Aerothermodynamic data for the GHAME vehicle are available in White et al. [1992], and incorporating these data would represent an important step toward a more comprehensive and accurate simulation.

- **Sensor fusion and state estimation:** The present work assumes that all states required for control, such as angle of attack, sideslip angle, bank angle, flight path angle and heading angle, are always provided accurately by the IMU. This is not realistic, since many of these quantities are not measured directly and all measurements are subject to noise and bias. Future work should therefore incorporate state estimation techniques such as Kalman filtering and sensor fusion to reconstruct these states from realistic sensor data. This would improve the realism of the simulation and could additionally reduce the impact of delays introduced by noise filtering.

- **Flight protection:** The present work does not include flight protection mechanisms. One possibility for future work is the use of control barrier functions (CBFs) or barrier Lyapunov functions (BLFs) to directly enforce safety limits. For instance, such functions could be applied to limit both the magnitude of the angle of attack and its rate of change in order to prevent inlet unstart. They could similarly be used to constrain the magnitude and rate of change of the bank angle. The approach could also be extended to structural limitations or other safety constraints, which are not modeled in the current study.

Bibliography

(1997). Department of defense, world geodetic system 1984, its definition and relationships with local geodetic systems. Technical Report NIMA TR 8350.2, National Imagery and Mapping Agency (NIMA), Bethesda, MD.

Acquatella, P., Falkena, W., van Kampen, E., and Chu, Q. (2012). Robust nonlinear spacecraft attitude control using incremental nonlinear dynamic inversion. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.2012-4623.

Acquatella, P., van Kampen, E., and Chu, Q. P. (2013). Incremental backstepping for robust nonlinear flight control. In *Proceedings of the 2nd CEAS Specialist Conference on Guidance, Navigation & Control (EuroGNC)*, pages 1–10.

Akkinapalli, V. S. and Holzapfel, F. (2018). Incremental dynamic inversion based velocity tracking controller for a multicopter system. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.2018-1345.

Araki, J. J. (1992). Reentry dynamics and handling qualities of a generic hypersonic vehicle. Master's thesis, Massachusetts Institute of Technology, URL: https://dspace.mit.edu/bitstream/handle/1721.1/42532/26912515-MIT.pdf.

Autenrieb, J. (2023). Data fusion-based incremental nonlinear model following control design for a hypersonic waverider configuration. In *AIAA SCITECH 2023 Forum*. DOI: https://doi.org/10.2514/6.2023-1997.

Azinheira, J. R., Moutinho, A., and Carvalho, J. (2015). Lateral control of airship with uncertain dynamics using incremental nonlinear dynamic inversion. *IFAC-PapersOnLine*, volume 48, number 19, pages 69–74, DOI: https://doi.org/10.1016/j.ifacol.2015.12.012.

Bacon, B. and Ostroff, A. (2000). Reconfigurable flight control using nonlinear dynamic inversion with a special accelerometer implementation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.2000-4565.

Bacon, B., Ostroff, A., and Joshi, S. (2001). Reconfigurable ndi controller using inertial sensor failure detection and isolation. *IEEE Transactions on Aerospace and Electronic Systems*, volume 37, number 4, pages 1373–1383, DOI: https://doi.org/10.1109/7.976972.

Balas, G. J. (2003). Flight control law design: An industry perspective. *European Journal of Control*, volume 9, number 2, pages 207–226, DOI: https://doi.org/10.3166/ejc.9.207-226.

Balas, G. J., Garrard, W. L., and Reiner, J. (1992). Robust dynamic inversion control laws for aircraft control. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.1992-4329.

Besser, H., Zimper, D., Huggins, M., Göge, D., and Shaffer, A. (2017). Hypersonic vehicles - game changers for future warfare? *The Journal of the JAPCC*, volume 24, pages 11–27, DOI: https://doi.org/10.13140/RG.2.1.3360.8566.

Bhardwaj, P., Raab, S., Zhang, J., and Holzapfel, F. (2018). Integrated reference model for a tilt-rotor vertical take-off and landing transition uav. In *Proceedings of the 36th AIAA Applied Aerodynamics Conference*. DOI: https://doi.org/10.2514/6.2018-3479.

Binz, F., Islam, T., and Moormann, D. (2019). Attitude control of tiltwing aircraft using a wing-fixed coordinate system and incremental nonlinear dynamic inversion. *International Journal of Micro Air Vehicles*, volume 11, DOI: https://doi.org/10.1177/1756829319861370.

Boiffier, J.-L. (2000). *The Dynamics of Flight: The Equations*. Wiley & Sons, ISBN: 978-0470860526.

Bolender, M. and Doman, D. (2005). A non-linear model for the longitudinal dynamics of a hypersonic air-breathing vehicle. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.2005-6255.

Bowers, A. H. et al. (1989). A generic hypersonic aerodynamic model example (ghame). Technical report, NASA Dryden Flight Research Facility.

Britting, K. R. (2010). *Inertial Navigation System Analysis*. Artech House Inc, Norwood, Massachusetts, ISBN: 978-1608070787.

Brockett, R. (1978). Feedback invariants for nonlinear systems. *IFAC Proceedings Volumes*, volume 11, number 1, pages 1115–1120, DOI: https://doi.org/10.1016/S1474-6670(17)66062-2.

Bruno, C. (2023). *Airbreathing Hypersonic Propulsion: An Introduction*. Space Propulsion. Springer Singapore, DOI: https://doi.org/10.1007/978-981-19-7927-9.

Buchholz, J. (2026). Trimmod. https://www.mathworks.com/matlabcentral/fileexchange/268-trimmod. MATLAB Central File Exchange, Retrieved April 25, 2026.

Bugajski, D. J. and Enns, D. F. (1992). Nonlinear control law with application to high angle-of-attack flight. *Journal of Guidance, Control, and Dynamics*, volume 15, number 3, pages 761–767, DOI: https://doi.org/10.2514/3.20902.

Bugajski, D. J., Enns, D. F., and Elgersma, M. (1990). A dynamic inversion based control law with application to the high angle-of-attack research vehicle. In *AIAA Guidance, Navigation and Control Conference*. DOI: https://doi.org/10.2514/6.1990-3407.

Calise, A. J. and Flandro, G. A. (1988). Trajectory optimization and guidance law development for national aerospace plane applications. Final Report NASA-CR-182211, NASA Langley Research Center and Georgia Institute of Technology, URL: https://ntrs.nasa.gov/citations/19890003167.

Chavez, F. and Schmidt, D. (1994). Analytical aeropropulsive-aeroelastic hypersonic-vehicle model with dynamic analysis. *Journal of Guidance, Control, and Dynamics*, volume 17, number 6, pages 1308–1319, DOI: https://doi.org/10.2514/3.21349.

Chow, J. and Kokotovic, P. (1978). Two-time-scale feedback design of a class of nonlinear systems. *IEEE Transactions on Automatic Control*, volume 23, number 3, pages 438–443, DOI: https://doi.org/10.1109/TAC.1978.1101736.

de Angelis Cordeiro, R., Marton, A. S., Azinheira, J. R., Carvalho, J. R., and Moutinho, A. (2021). Increased roooustness to delay in incremental controllers using input scaling gain. *IEEE Transactions on Aerospace and Electronic Systems*, volume 58, number 2, pages 1199–1210, DOI: https://doi.org/10.1109/TAES.2021.3123215.

Donald, D. (2002). *International Air Power Review: 3*. AIRtime Publishing, Norwalk, Connecticut.

Donald, D. (2004). *Black Jets: The Development and Operation of America's Most Secret Warplane*. AIRtime Publishing, Norwalk, Connecticut.

Elgersma, M. R. (1988). *Control of Nonlinear Systems Using Partial Dynamic Inversion*. PhD thesis, University of Minnesota.

Enns, D., Bugajski, D., Hendrick, R., and Stein, G. (1994). Dynamic inversion: an evolving methodology for flight control design. *International Journal of Control*, volume 59, number 1, pages 71–91, DOI: https://doi.org/10.1080/00207179408923070.

Etkin, B. (2005). *Dynamics of Atmospheric Flight*. Dover Publications Inc., ISBN: 978-0471246206.

Falempin, F. (2008). Ramjet and dual mode operation. *Advances on Propulsion Technology for High-Speed Aircraft*, , pages 1–36.

Fiorentini, L. and Serrani, A. (2009). Nonlinear robust adaptive control of flexible air-breathing hypersonic vehicles. *Journal of Guidance, Control, and Dynamics*, volume 32, number 2, pages 402–417, DOI: https://doi.org/10.2514/1.39210.

Fiorentini, L., Serrani, A., Bolender, M. A., and Doman, D. B. (2009). Nonlinear control of non-minimum phase hypersonic vehicle models. In *Proceedings of the American Control Conference*, pages 1585–1590, St. Louis, MO. DOI: https://doi.org/10.1109/ACC.2009.5160211.

Force, U. A. (2011). X-51a waverider. URL: https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104467/x-51a-waverider/. Accessed: 2024-11-18.

Franco, A., Bourlès, H., De Pieri, E., and Guillard, H. (2006). Robust nonlinear control associating robust feedback linearization and h_∞ control. *IEEE Transactions on Automatic Control*, volume 51, number 7, pages 1200–1207, DOI: 10.1109/TAC.2006.878782.

Gilbert, M. G., Heeg, J., and Pototzky, A. S. (1990). The application of active controls technology to a generic hypersonic aircraft configuration. Technical report, NASA, URL: https://ntrs.nasa.gov/citations/19900010755.

Goz, E. (2024). Robust Multi-Objective H-Infinity Control of a Generic Hypersonic Vehicle. Master's thesis, Delft University of Technology.

Grondman, F., Looye, G., Kuchar, R. O., Chu, Q. P., and van Kampen, E. (2018). Design and flight testing of incremental nonlinear dynamic inversion-based control laws for a passenger aircraft. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.2018-0619.

Groves, K. P., Sigthorsson, D. O., Serrani, A., Yurkovich, S., Bolender, M. A., and Doman, D. B. (2005). Reference command tracking for a linearized model of an air-breathing hypersonic vehicle. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.2005-6144.

Hall, I. A. and Poggie, J. (2019). Simulation of unstart in hypersonic flow with a dual-mode scramjet model. In *AIAA Scitech 2019 Forum*. DOI: https://doi.org/10.2514/6.2019-0946.

Harris, J. J. and Stanford, J. R. (2018). F-35 flight control law design, development and verification. In *2018 Aviation Technology, Integration, and Operations Conference*. AIAA, DOI: https://doi.org/10.2514/6.2018-3516.

Hermeus (2024). Hermeus Celebrates Groundbreaking For Hypersonic Engine And Flight Test Facility In Jacksonville, Florida. URL: https://www.hermeus.com/press-release-heat. Accessed: 2024-11-18.

Hollings, A. (2024). Evidence is Mounting That Lockheed Martin's SR-72 Could Be in Production. URL: https://www.sandboxx.us/news/evidence-is-mounting-that-lockheed-martins-sr-72-could-be-in-production/. Accessed: 2024-11-18.

Huang, Y., Zhang, Y., Pool, D., Stroosma, O., and Chu, Q. (2022). Time-delay margin and robustness of incremental nonlinear dynamic inversion control. *Journal of Guidance, Control, and Dynamics*, volume 45, number 2, pages 394–404, DOI: https://doi.org/10.2514/1.G006024.

Hunt, L. R. and Su, R. (1981). Control of nonlinear time-varying systems. In *1981 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pages 558–563. DOI: https://doi.org/10.1109/CDC.1981.269267.

Hyde, R. and Papageorgiou, G. (2001). Analysing the stability of ndi-based flight controllers with lpv methods. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.2001-4039.

Isidori, A. (1985). *Nonlinear Control Systems: An Introduction*, volume 72 of *Lecture Notes in Control and Information Sciences*. Springer, Berlin, Heidelberg, DOI: https://doi.org/10.1007/BFb0006368.

Jakubczyk, B. and Respondek, W. (1980). On Linearization of Control Systems. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, volume 28, number 9-10, pages 517–522, URL: https://heuklyd.github.io/papers/pdf/Jakubczyk-Respondek-1980.pdf.

Jenkins, D. R., Landis, T., and Miller, J. (2003). *American X-Vehicles: An Inventory—X-1 to X-50*. Number 31 in Monographs in Aerospace History. NASA History Office, URL: https://www.nasa.gov/wp-content/uploads/2023/04/sp-4531.pdf. NASA SP-2003-4531.

Johnson, E. N. and Calise, A. J. (2000). Pseudo-control hedging: A new method for adaptive control. In *Proceedings of the Advances in Navigation and Control Technology Workshop*, Redstone Arsenal, Alabama.

Johnson, P. J., Whitehead, A. H., J., and Chapman, G. T. (1987). Fitting aerodynamics and propulsion into the puzzle. *AIAA Aerospace America*, , pages 32–34.

Jr., W. R. W. and Meyer, G. (1980). Flight tests of the total automatic flight control system (Tafcos) concept on a DHC-6 Twin Otter aircraft. Technical Report NASA-TP-1513, NASA, URL: https://ntrs.nasa.gov/citations/19800008821.

Juliana, S., Chu, Q. P., Mulder, J. A., and van Baten, T. J. (2004). Flight control of atmospheric re-entry vehicle with non-linear dynamic inversion. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.2004-5330.

Khalil, H. (2002). *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ, 3rd edition, ISBN: 978-0130673893.

Kim, C.-S., Ji, C.-H., Koh, G.-O., and Kim, B. S. (2021). Stability margin and structural coupling analysis of a hybrid indi control for the fighter aircraft. *International Journal of Aeronautical and Space Sciences*, volume 22, number 5, pages 1154–1169, DOI: https://doi.org/10.1007/s42405-021-00394-8.

Kolesnikov, E. (2005). Ndi-based flight control law design. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.2005-5977.

Koschorke, J., Falkena, W., Kampen, E.-J. V., and Chu, Q. P. (2013). Time delayed incremental nonlinear control. In *AIAA Guidance, Navigation, and Control (GNC) Conference*. AIAA, DOI: https://doi.org/10.2514/6.2013-4929.

Krener, A. J. (1973). On the Equivalence of Control Systems and the Linearization of Nonlinear Systems. *SIAM Journal on Control*, volume 11, number 4, pages 670–676, DOI: https://doi.org/10.1137/0311051.

Kumtepe, Y., Pollack, T., and Van Kampen, E. J. (2022). Flight control law design using hybrid incremental nonlinear dynamic inversion. In *AIAA SCITECH 2022 Forum*. AIAA, DOI: https://doi.org/10.2514/6.2022-1597.

Li, Y., Sun, L., Qu, X., and Tan, W. (2016). Acceleration measurement-based incremental nonlinear flight control for air-breathing hypersonic vehicles. *Aerospace Science and Technology*, volume 58, pages 235–247, DOI: https://doi.org/10.1016/j.ast.2016.08.022.

Liu, Z., Zhang, Y., Liang, J., and Chen, H. (2022). Application of the improved incremental nonlinear dynamic inversion in fixed-wing uav flight tests. *Journal of Aerospace Engineering*, volume 35, number 6, DOI: https://doi.org/10.1061/(ASCE)AS.1943-5525.0001495.

Lovell-Prescod, G. H. L. H., Ma, Z., and Smeur, E. J. J. (2023). Attitude control of a tilt-rotor tailsitter micro air vehicle using incremental control. In *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: https://doi.org/10.1109/ICUAS57906.2023.10156272.

Lu, P., van Kampen, E.-J., de Visser, C., and Chu, Q. (2016). Aircraft fault-tolerant trajectory control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, volume 57, pages 126–141, DOI: https://doi.org/10.1016/j.conengprac.2016.09.010.

McRuer, D. (1991). Design and modeling issues for integrated airframe/propulsion control of hypersonic flight vehicles. In *1991 American Control Conference*, pages 729–734. DOI: https://doi.org/10.23919/ACC.1991.4791471.

Mellinger, G. R. (1960). Design and Operation of the X-15 Hypersonic Research Airplane. Technical report, Advisory Group for Aeronautical Research and Development, URL: https://apps.dtic.mil/sti/tr/pdf/AD0279830.pdf.

Meyer, G. and Cicolani, L. S. (1975). A Formal Structure for Advanced Automatic Flight-Control Systems. Technical Report NASA-TN-D-7940, NASA, URL: https://ntrs.nasa.gov/citations/19750015534.

Meyer, G. and Cicolani, L. S. (1980). Application of Nonlinear Systems Inverses to Automatic Flight Control Design: System Concepts and Flight Evaluations. Technical report, NASA, URL: https://ntrs.nasa.gov/citations/19820003210.

Meyer, G., Su, R., and Hunt, L. (1984). Application of nonlinear transformations to automatic flight control. *Automatica*, volume 20, number 1, pages 103–107, DOI: https://doi.org/10.1016/0005-1098(84)90069-4.

Milz, D. and Looye, G. (2022). Tilt-wing control design for a unified control concept. In *AIAA Scitech 2022 Forum*. DOI: https://doi.org/10.2514/6.2022-1084.

Mooij, E. (1997). *The Motion of a Vehicle in a Planetary Atmosphere*. Delft University Press, The Netherlands, ISBN: 978-9040715952.

Moses, P. L., Rausch, V. L., Nguyen, L. T., and Hill, J. R. (2004). Nasa hypersonic flight demonstrators; overview, status, and future plans. *Acta Astronautica*, volume 55, number 3–9, pages 619–630, DOI: https://doi.org/10.1016/j.actaastro.2004.05.045.

Murillo, O. J. and Lu, P. (2010). Fast ascent trajectory optimization for hypersonic air-breathing vehicles. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.2010-8173.

Myschik, S., Kinast, L., Huemer, M., et al. (2022). Development of a flight control system for a cyclocopter uav demonstrator. In *AIAA AVIATION 2022 Forum*. DOI: https://doi.org/10.2514/6.2022-3282.

Naidu, D. S. and Calise, A. J. (2001). Singular perturbations and time scales in guidance and control of aerospace systems: A survey. *Journal of Guidance, Control, and Dynamics*, volume 24, number 6, pages 1057–1078, DOI: https://doi.org/10.2514/2.4830.

NASA (1976). U.s. standard atmosphere. 1976. Technical Memorandum NASA-TM-X-7433, NASA, Washington, D.C., URL: https://ntrs.nasa.gov/api/citations/19770009539/downloads/19770009539.pdf. Work of the U.S. Government. Public use permitted.

NASA (2022). X-43A Hyper-X. URL: https://www.nasa.gov/reference/x-43a/. Accessed: 2024-11-18.

Nise, N. S. (2015). *Control Systems Engineering*. Wiley & Sons, Hoboken, NJ, 7th edition, ISBN: 978-1118170519.

Ostroff, A. J. and Bacon, B. J. (2002). Enhanced ndi strategies for reconfigurable flight control. In *Proceedings of the American Control Conference*, pages 3631–3636, Piscataway, NJ, USA. IEEE, DOI: https://doi.org/10.1109/ACC.2002.1024492.

Papageorgiou, G. and Polansky, M. (2009). Tuning a dynamic inversion pitch axis autopilot using mcfarlane–glover loop shaping. *Optimal Control Applications and Methods*, volume 30, number 3, pages 287–308, DOI: https://doi.org/10.1002/oca.864.

Parker, J. T., Serrani, A., Yurkovich, S., Bolender, M. A., and Doman, D. B. (2006). Approximate feedback linearization of an air-breathing hypersonic vehicle. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. AIAA, DOI: https://doi.org/10.2514/6.2006-6556.

Pavel, M., Shanthakumaran, P., Chu, Q., Stroosma, O., Wolfe, M., and Cazemier, H. (2020). Incremental nonlinear dynamic inversion for the apache ah-64 helicopter control. *Journal of the American Helicopter Society*, volume 65, number 2, pages 1–16, DOI: https://doi.org/10.4050/JAHS.65.022006.

Penland, J. A., Dillon, J. L., and Pittman, J. L. (1978). An aerodynamic analysis of several hypersonic research airplane concepts from m= 0.2 to 6.0. *Journal of Aircraft*, volume 15, number 11, pages 716–723, DOI: https://doi.org/10.2514/3.58437.

Pfeifle, O. and Fichter, W. (2021). Cascaded incremental nonlinear dynamic inversion for three-dimensional spline-tracking with wind compensation. *Journal of Guidance, Control, and Dynamics*, volume 44, number 8, pages 1559–1571, DOI: https://doi.org/10.2514/1.G005785.

Pollack, T. (2024). *Advances in Dynamic Inversion-based Flight Control Law Design: Multivariable Analysis and Synthesis of Robust and Multi-Objective Design Solutions*. PhD thesis, Delft University of Technology, DOI: https://doi.org/10.4233/uuid:28617ba0-461d-48ef-8437-de2aa41034ea.

Pollack, T., Looye, G., and der Linden, F. V. (2019). Design and flight testing of flight control laws integrating incremental nonlinear dynamic inversion and servo current control. In *AIAA Scitech 2019 Forum*. DOI: https://doi.org/10.2514/6.2019-0130.

Pollack, T., Theodoulis, S., and Kampen, E.-J. V. (2024). Commonalities between robust hybrid incremental nonlinear dynamic inversion and proportional-integral- derivative flight control law design. *Aerospace science and technology*, DOI: https://doi.org/10.1016/J.AST.2024.109377.

Pollack, T., Theodoulis, S., and Wang, X. (2025). Quasi-lpv transformations for robust gain scheduling of incremental nonlinear dynamic inversion-based controllers. *IFAC-PapersOnLine*, volume 59, number 15, pages 97–102, DOI: https://doi.org/10.1016/j.ifacol.2025.10.064. 6th IFAC Workshop on Linear Parameter Varying Systems LPVS 2025.

Pollack, T., Theodoulis, S., and Wang, X. (2026). Duality between incremental nonlinear dynamic inversion and quasi-linear parameter-varying control. *Journal of Guidance, Control, and Dynamics*, volume 0, number 0, pages 1–17, DOI: https://doi.org/10.2514/1.G009559.

Pollack, T. and van Kampen, E.-J. (2023). Robust stability and performance analysis of incremental dynamic-inversion-based flight control laws. *Journal of Guidance, Control, and Dynamics*, volume 46, number 9, pages 1785–1798, DOI: https://doi.org/10.2514/1.G006576.

Poulain, F., Piet-Lahanier, H., and Serre, L. (2009). Nonlinear control of a airbreathing hypersonic vehicle. In *16th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. DOI: https://doi.org/10.2514/6.2009-7290.

Raab, S., Zhang, J., Bhardwaj, P., and Holzapfel, F. (2018). Proposal of a unified control strategy for vertical take-off and landing transition aircraft configurations. In *36th AIAA Applied Aerodynamics Conference*. DOI: https://doi.org/10.2514/6.2018-3478.

Raab, S. A. (2025). *Contributions to Incremental Nonlinear Dynamic Inversion for Flight Control*. PhD thesis, Technische Universität München, DOI: https://doi.org/10.14459/2026md1785447, URL: https://mediatum.ub.tum.de/1785447.

Raab, S. A., Zhang, J. N., Bhardwaj, P., et al. (2019). Consideration of control effector dynamics and saturations in an extended indi approach. In *AIAA Aviation 2019 Forum*. DOI: https://doi.org/10.2514/6.2019-3267.

Reiner, J., Balas, G. J., and Garrard, W. L. (1996). Flight control design using robust dynamic inversion and time-scale separation. *Automatica*, volume 32, number 11, pages 1493–1504, DOI: https://doi.org/10.1016/S0005-1098(96)00101-X.

Sachs, G. (1998). Path-attitude decoupling and flying qualities implications in hypersonic flight. *Aerospace Science and Technology*, volume 2, number 1, pages 49–59, DOI: https://doi.org/10.1016/S0034-1223(98)80005-5.

Sachs, G. and Moravszki, C. (2006). Predictive tunnel display for hypersonic flight path control. In *AIAA Guidance, Navigation, and Control Conference*, pages 3731–3744. DOI: https://doi.org/10.2514/6.2006-6562.

Savage, P. G. (1984). Strapdown system algorithms. In *AGARD Lecture Series*, number 133, page 379.

Scherllin-Pirscher, B., Steiner, A. K., Kirchengast, G., Schwärz, M., and Leroy, S. S. (2017). The power of vertical geolocation of atmospheric profiles from gnss radio occultation. *Journal of Geophysical Research: Atmospheres*, volume 122, pages 1595–1616, DOI: https://doi.org/10.1002/2016JD025902.

Schmidt, D. (1992). Dynamics and control of hypersonic aeropropulsive/aeroelastic vehicles. In *AIAA Guidance, Navigation and Control Conference*. DOI: https://doi.org/10.2514/6.1992-4326.

Schmidt, D., Mamich, H., and Chavez, F. (1991). Dynamics and control of hypersonic vehicles - the integration challenge for the 1990's. In *3rd International Aerospace Planes Conference*. DOI: https://doi.org/10.2514/6.1991-5057.

Schmidt, D. and Velapoldi, J. (1999). Flight dynamics and feedback guidance issues for hypersonic air-breathing vehicles. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.1999-4122.

Schumacher, C. and Khargonekar, P. P. (1998). Stability analysis of a missile control system with a dynamic inversion controller. *Journal of Guidance, Control, and Dynamics*, volume 21, number 3, pages 508–515, DOI: https://doi.org/10.1109/ACC.1998.707303.

Serrani, A. (2013). Nested zero-dynamics redesign for a non-minimum phase longitudinal model of a hypersonic vehicle. In *52nd IEEE Conference on Decision and Control*, pages 4833–4838. DOI: https://doi.org/10.1109/CDC.2013.6760647.

Shaughnessy, J. D., Pinckney, S. Z., McMinn, J. D., Cruz, C. I., and Kelley, M.-L. (1990). Hypersonic vehicle simulation model: Winged-cone configuration. Technical Memorandum NASA TM 102610, NASA, URL: https://ntrs.nasa.gov/citations/19910003392.

Sieberling, S., Chu, Q., and Mulder, J. (2010). Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction. *Journal of Guidance, Control, and Dynamics*, volume 33, number 6, pages 1732–1742, DOI: https://doi.org/10.2514/1.49978.

Simplício, P., Pavel, M., van Kampen, E., and Chu, Q. (2013). An acceleration measurements-based approach for helicopter nonlinear flight control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, volume 21, number 8, pages 1065–1077, DOI: https://doi.org/10.1016/j.conengprac.2013.03.009.

Skogestad, S. and Postlethwaite, I. (2005). *Multivariable Feedback Control: Analysis and Design*. Wiley & Sons, Hoboken, NJ, USA, 2nd edition, ISBN: 978-0470011683.

Slotine, J. and Li, W. (1991). *Applied Nonlinear Control*. Prentice Hall, ISBN: 978-0130408907.

Smeur, E. J., Bronz, M., and de Croon, G. C. H. E. (2020). Incremental control and guidance of hybrid aircraft applied to a tailsitter unmanned air vehicle. *Journal of Guidance, Control, and Dynamics*, volume 43, number 2, pages 274–287, DOI: https://doi.org/10.2514/1.G004520.

Smeur, E. J. J., Chu, Q. P., and de Croon, G. C. H. E. (2016a). Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *Journal of Guidance, Control, and Dynamics*, volume 39, number 3, pages 450–461, DOI: https://doi.org/10.2514/1.G001490.

Smeur, E. J. J., de Croon, G. C. H. E., and Chu, Q. (2018). Cascaded incremental nonlinear dynamic inversion for mav disturbance rejection. *Control Engineering Practice*, volume 73, pages 79–90, DOI: https://doi.org/10.1016/j.conengprac.2018.01.003.

Smeur, E. J. J., de Croon, G. C. H. E., and Chu, Q. P. (2016b). Gust disturbance alleviation with incremental nonlinear dynamic inversion. *IEEE Transactions on Aerospace and Electronic Systems*, volume 52, number 6, pages 2591–2603, DOI: https://doi.org/10.1109/TAES.2016.160106.

Smit, M. and Craig, I. (1998). Robust flight controller design using h-infinity loop-shaping and dynamic inversion techniques. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: https://doi.org/10.2514/6.1998-4132.

Smith, P. (1998). A simplified approach to nonlinear dynamic inversion based flight control. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, pages 762–770. DOI: https://doi.org/10.2514/6.1998-4461.

Smith, P. and Berry, A. (2000). Flight test experience of a non-linear dynamic inversion control law on the vaac harrier. In *AIAA Atmospheric Flight Mechanics Conference*. DOI: https://doi.org/10.2514/6.2000-3914.

Snell, S. (1992). Preliminary assessment of the robustness of dynamic inversion based flight control laws. In *AIAA Guidance, Navigation, and Control Conference*. DOI: https://doi.org/10.2514/6.1992-4330.

Snell, S. A., Enns, D. F., and Garrard, W. L. (1992). Nonlinear inversion flight control for a supermaneuverable aircraft. *Journal of Guidance, Control, and Dynamics*, volume 15, number 4, pages 976–984, DOI: https://doi.org/10.2514/3.20932.

Snell, S. A. and Stout, P. W. (1994). Improved robustness for dynamic inversion based nonlinear flight control laws. *IFAC Proceedings Volumes*, volume 27, number 13, pages 261–266, DOI: https://doi.org/10.1016/S1474-6670(17)45810-1.

Steffensen, R., Steinert, A., Mbikayi, Z., Raab, S., Angelov, J., and Holzapfel, F. (2023a). Filter and sensor delay synchronization in incremental flight control laws. *Aerospace Systems*, volume 6, pages 285–304, DOI: https://doi.org/10.1007/s42401-022-00186-2.

Steffensen, R., Steinert, A., and Smeur, E. J. J. (2023b). Nonlinear dynamic inversion with actuator dynamics: An incremental control perspective. *Journal of Guidance, Control, and Dynamics*, volume 46, number 4, pages 709–717, DOI: https://doi.org/10.2514/1.G007079.

Stein, G. (2003). Respect the unstable. *IEEE Control Systems Magazine*, volume 23, number 4, pages 12–25, DOI: https://doi.org/10.1109/MCS.2003.1213600.

Steinert, A., Raab, S., Hafner, S., Holzapfel, F., and Hong, H. (2024). From fundamentals to applications of incremental nonlinear dynamic inversion: A survey on indi – part i. *Chinese Journal of Aeronautics*, DOI: https://doi.org/10.1016/j.cja.2025.103553.

Steinert, A., Raab, S., Hafner, S., Holzapfel, F., and Hong, H. (2025). Advancements in incremental nonlinear dynamic inversion and its components: A survey on indi – part ii. *Chinese Journal of Aeronautics*, volume 38, number 11, DOI: https://doi.org/10.1016/j.cja.2025.103591.

Steinleitner, A., Frenzel, V., Pfeifle, O., Denzel, J., and Fichter, W. (2022). Automatic takeoff and landing of tailwheel aircraft with incremental nonlinear dynamic inversion. In *AIAA Scitech 2022 Forum*. DOI: https://doi.org/10.2514/6.2022-1228.

Stevens, B. L., Lewis, F. L., and Johnson, E. N. (2016). *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. Wiley & Sons, Hoboken, NJ, third edition, ISBN: 978-1118870983.

Stich, R. and Sachs, G. (1998). Simulation tests for investigating flying qualities of aerospace planes. In *8th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. DOI: https://doi.org/10.2514/6.1998-1520.

Tal, E. and Karaman, S. (2022). Global incremental flight control for agile maneuvering of a tailsitter flying wing. *Journal of Guidance, Control, and Dynamics*, volume 45, number 12, pages 2332–2349, DOI: https://doi.org/10.2514/1.G006645.

Thomas, S. R., Walker, J. F., and Pittman, J. L. (2009). Overview of the turbine based combined cycle discipline. Technical Report E-17652, NASA, URL: https://ntrs.nasa.gov/api/citations/20110012002/downloads/20110012002.pdf.

van Ekeren, W., Looye, G., Kuchar, R. O., Chu, Q. P., and van Kampen, E. (2018). Design, implementation and flight-tests of incremental nonlinear flight control methods. In *2018 AIAA Guidance, Navigation, and Control Conference*, Kissimmee, FL, USA. American Institute of Aeronautics and Astronautics, DOI: https://doi.org/10.2514/6.2018-0384.

van't Veld, R., Van Kampen, E. J., and Chu, Q. (2018). Stability and robustness analysis and improvements for incremental nonlinear dynamic inversion control. In *2018 AIAA Guidance, Navigation, and Control Conference*. AIAA, DOI: https://doi.org/10.2514/6.2018-1127.

Vu, P. and Biezad, D. J. (1993a). Longitudinal control of hypersonic aircraft: An alpha follow-up scheme. In *Proceedings. The First IEEE Regional Conference on Aerospace Control Systems*, pages 440–444. DOI: https://doi.org/10.1109/AEROCS.1993.720973.

Vu, P. and Biezad, D. J. (1993b). A pseudo-loop design strategy for the longitudinal control of hypersonic aircraft. In *AIAA Guidance, Navigation and Control Conference*. DOI: https://doi.org/10.2514/6.1993-3814.

Vu, P. and Biezad, D. J. (1994). Direct-lift design strategy for longitudinal control of hypersonic aircraft. *Journal of Guidance, Control, and Dynamics*, volume 17, number 6, pages 1260–1266, DOI: https://doi.org/10.2514/3.21342.

Wacker, Munday, S., and Merkle, S. (2001). X-38 application of dynamic inversion flight control. In *24th Annual Guidance and Control Conference*, Breckenridge, CO, USA. American Astronautical Society.

Wang, Q. and Stengel, R. F. (2000). Robust nonlinear control of a hypersonic aircraft. *Journal of Guidance, Control, and Dynamics*, volume 23, number 4, pages 577–585, DOI: https://doi.org/10.2514/2.4580.

Wang, Q. and Stengel, R. F. (2001). Comment on "robust nonlinear control of a hypersonic aircraft". *Journal of Guidance, Control, and Dynamics*, volume 24, number 1, pages 143–144, DOI: https://doi.org/10.2514/2.G6088-R.

Wang, X., van Kampen, E.-J., Chu, Q., and Lu, P. (2019). Stability analysis for incremental nonlinear dynamic inversion control. *Journal of Guidance, Control, and Dynamics*, volume 42, number 5, pages 1116–1129, DOI: https://doi.org/10.2514/1.G003791.

Wang, Z., Zhao, J., Cai, Z., Wang, Y., and Liu, N. (2021). Onboard actuator model-based incremental nonlinear dynamic inversion for quadrotor attitude control: Method and application. *Chinese Journal of Aeronautics*, volume 34, number 1, pages 27–37, DOI: https://doi.org/10.1016/j.cja.2021.03.018.

White, D. A., Bowers, A., Iliff, K., and Menousek, J. (1992). *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*. Multiscience Press, Inc., New York, NY, ISBN: 978-0442308575.

Xin, H. B., Chen, Q. Y., Wang, P., et al. (2023). The control performance analysis of the incremental nonlinear dynamic inverse method and flight test. In *Proceedings of the 2022 International Conference on Autonomous Unmanned Systems*, pages 3609–3619, Singapore. Springer Nature Singapore, DOI: https://doi.org/10.1007/978-981-99-0479-2_333.

Ye, L., Zong, Q., Crassidis, J. L., and Tian, B. (2018). Output-redefinition-based dynamic inversion control for a nonminimum phase hypersonic vehicle. *IEEE Transactions on Industrial Electronics*, volume 65, number 4, pages 3447–3457, DOI: https://doi.org/10.1109/TIE.2017.2760246.

Zipfel, P. H. (2023). *Introduction to Tensor Flight Dynamics*. Independently published, ISBN: 979-8396149229.

Zipfel, P. H. (2025). *Modeling and Simulation of Aerospace Vehicle Dynamics*. AIAA, Reston, Virginia, 4th edition, ISBN: 978-1624107542.

A

Physical and Mathematical Constants

Table A.1: GHAME geometric and mass properties, together with WGS-84 physical constants (SI units). Values correspond exactly to constants used by the Hyper6 simulation of Zipfel [2025]

Parameter	**Symbol**	**Constant**	**Takeoff**	**Fuel Burnout**	**Unit**
Fuselage length	l	71.1	–	–	m
Reference area	S	557.42	–	–	m^2
Reference chord	c	22.86	–	–	m
Reference span	b	24.38	–	–	m
Inlet cowl area	A_c	27.87	–	–	m^2
Weight	W	–	1.334×10^6	5.337×10^5	N
Mass	m	–	136,080	54,432	kg
Moment of inertia	I_{xx}	–	1.573×10^6	1.180×10^6	kg·m^2
Moment of inertia	I_{yy}	–	31.60×10^6	19.25×10^6	kg·m^2
Moment of inertia	I_{zz}	–	32.54×10^6	20.20×10^6	kg·m^2
Product of inertia	I_{xz}	–	0.380×10^6	0.240×10^6	kg·m^2
Earth gravitational const.	GM	3.986005×10^{14}	–	–	m^3/s^2
2^{nd} deg. zonal grav. coef.	$\bar{C}_{2,0}$	$-4.8416685 \times 10^{-4}$	–	–	–
Flattening	f	1/298.257223563	–	–	–
Equatorial radius	R_e	6.378137×10^6	–	–	m
Mean Earth radius	R_0	6.370987308×10^6	–	–	m
Earth rotation rate	$\omega_\oplus$	7.292115×10^{-5}	–	–	rad/s
Stand. gravity at sea level	g_0	9.80675445	–	–	m/s^2
Gravity at equator	γ_e	9.7803253359	–	–	m/s^2
Gravity at pole	γ_p	9.8321849378	–	–	m/s^2
Geodynamical constant	m	$3.449786506 \times 10^{-3}$	–	–	–
Conversion deg->rad	RAD	0.0174532925199432	–	–	rad
Conversion rad->deg	DEG	57.2957795130823	–	–	deg
Number of PI	π	3.1415926536	–	–	–

B

SciTech 2026 Paper Submission

A condensed version of this thesis, presenting some of its main results, was submitted as a paper to SciTech Forum 2026. The paper places particular focus on the time-delay aspect of the measurement feedback path and on the derivation of the control laws.

AIAA SciTech Forum
12-16 January 2026, Orlando, FL
AIAA SCITECH 2026 Forum

10.2514/6.2026-0548

Incremental Nonlinear Dynamic Inversion based Control of a Generic Hypersonic Vehicle (GHAME)

T. H. Mueller* and S. Theodoulis †
Delft University of Technology, Delft, 2628CD, Netherlands

I. Sarras‡
ONERA - The French Aerospace Lab, Palaiseau, 91123, France

The Generic Hypersonic Aerodynamics Model Example (GHAME) provides a practical benchmark for evaluating advanced control strategies for hypersonic vehicles. Its nonlinear dynamics and strong aero–propulsive coupling create challenges well suited to nonlinear inversion methods. This work develops a hierarchical control architecture based on time–scale separation, combining NDI for attitude and position control with Incremental Nonlinear Dynamic Inversion (INDI) for angular–rate and velocity control. The controller is implemented in MATLAB and Simulink and evaluated under synchronized and desynchronized sensor delays. The results show that delay synchronization markedly increases the admissible delay margin. The study also reveals a fundamental limitation in the lateral axis: the lateral-directional dynamics of GHAME are too fast to satisfy the time-scale separation assumption required by INDI, leading to unreliable linear stability predictions. In contrast, the longitudinal dynamics do satisfy this assumption and remain well suited to inversion-based control. Overall, the NDI–INDI structure is effective for the longitudinal motion when delays are synchronized, but the intrinsic speed of the lateral dynamics imposes a major constraint on its applicability for lateral control.

Nomenclature

$\alpha,\ \mu,\ \beta$	=	AoA, bank, sideslip angles	C_L, C_Y, C_D	=	Stability-axis force coeffs.
C_X, C_Y, C_Z	=	Body-axis force coeffs.	C_l, C_m, C_n	=	Moment coeffs.
$c_{\text{ref}},\ b_{\text{ref}}$	=	Ref. chord and span	$\omega_{act},\ \zeta_{act}$	=	Act. natural freq. and damp.
p, q, r	=	Inertial angular rates	$\delta_a, \delta_e, \delta_r$	=	Aileron, elevator, rudder defl.
δ_{vl}, δ_{vr}	=	Left/right elevon	$u_N,\ u_E,\ u_U$	=	North, East, Up velocity comp.
f_P	=	Propulsive force	$\omega_H,\ \zeta_H$	=	Noise filter natural freq. and damp.
ρ	=	Air density	$\bar{q}$	=	Dynamic pressure
M	=	Mach number	ω_a	=	Anti-aliasing natural freq.
g_0	=	Standard gravity	R_0	=	Standard radius Earth
γ	=	Flightpath angle	χ	=	Heading angle
$\boldsymbol{f}_g$	=	Grav. force	$\boldsymbol{f}_{a,p}$	=	Aero-prop. force
V_K	=	Groundspeed	δ_t	=	Throttle setting
$(\boldsymbol{m}_c)_\delta$	=	Control moment coefs.	S	=	Ref. surface area
$\boldsymbol{m}_a$	=	Aero. moment coefs.	Δt	=	Sample time
$\boldsymbol{m}_B$	=	External moments	τ_{SD}	=	Sensor delay
BW	=	Bandwidth	τ_{CD}	=	Computational delay
$\omega_\oplus$	=	Earth angular rate	$[T]^{VG}$	=	Geoc.-to-vel. transform
ω	=	Inertial angular rate	$[T]^{VB}$	=	Body.-to-vel. transform
λ, l, h	=	Lat., long., altitude	$\boldsymbol{I}$	=	Moment of inertia matrix

*MSc, student, Control & Simulation division, Faculty of Aerospace Engineering, P.O. Box 5058, 2600GB Delft, Netherland

†Associate Professor, Control & Simulation division, Faculty of Aerospace Engineering, P.O. Box 5058, 2600GB Delft, Netherlands; Associate Fellow AIAA.

‡Research engineer, Information Processing and Systems Department, ONERA, Univ. Paris-Saclay, F-91123 Palaiseau, France;

I. Introduction

Recent interest in applying Incremental Nonlinear Dynamic Inversion (INDI) to hypersonic vehicles is driven by the extreme control challenges posed by these platforms. Air-breathing hypersonic vehicles exhibit strong propulsion–airframe coupling, highly nonlinear and speed-dependent stability characteristics, and structural flexibility arising from slender geometries and thermal loading [1, 2]. These effects produce rapid variations in aerodynamic forces and moments that are difficult to model accurately in real time. The reduced reliance of INDI on precise aerodynamic models therefore makes it an appealing candidate for such environments.

INDI achieves this reduced model dependence by using measurements of the controlled-variable derivatives to construct an incremental inversion of the dynamics. In an aircraft rate-control study, Smith [3] demonstrated that the resulting control law becomes largely insensitive to uncertainties in aerodynamic coefficients, centre-of-gravity position, and inertial properties, since these effects are already embedded in the measured angular accelerations. The same study, however, showed that this benefit comes at the cost of increased sensitivity to sensor delay.

To assess the applicability of INDI in the hypersonic regime, this study uses the Generic Hypersonic Aerodynamic Model Example (GHAME), a publicly released NASA model of a single-stage-to-orbit vehicle. GHAME combines aerodynamic and propulsive characteristics of turbojet, ramjet, and scramjet propulsion across Mach 0–24 and captures the dominant nonlinearities governing hypersonic flight [4]. Using this model, the paper develops and evaluates a cascaded INDI controller and examines its robustness under synchronized and desynchronized sensor delays to determine whether the method's inherent robustness extends to flight conditions characterized by strong nonlinearities and rapid variations in aerodynamic behavior. The main contribution is the first systematic application and assessment of an INDI-based architecture on a high-fidelity hypersonic model.

II. Modelling

A. GHAME Vehicle Model

The GHAME is a high-fidelity aerodynamic and geometric model developed at the NASA Ames Research Center and extensively tested at the Dryden Flight Research Center [4]. It was conceived during the early phases of the NASP initiative to provide accurate and physically representative aerodynamic data for hypersonic vehicle research. GHAME was designed as a generic hypersonic configuration capable of performing a SSTO mission, involving horizontal takeoff using air-breathing propulsion, acceleration to orbital velocity, orbital insertion, atmospheric reentry, and unpowered gliding recovery. The aerodynamic database spans the complete mission envelope, including subsonic, transonic, hypersonic, and reentry flight regimes [5].

The model enables high-fidelity Six-Degree-of-Freedom (6 DOF) simulations and produces a dynamic response representative of hypersonic flight. Consequently, GHAME has become a benchmark for evaluating control architectures, developing guidance strategies, and conducting trajectory optimization studies. Multiple versions of the model have been implemented in Fortran by P. Zipfel, using the original NASA Ames aerodynamic dataset as documented in White et al. [5] and detailed in his textbook [6]. More recently, Goz [7] developed a Simulink-based implementation of a simplified GHAME model assuming a nonrotating, flat Earth, constant mass and first-order actuator dynamics.

B. Dynamics

The simulation in this work is a Simulink implementation of the full six-degree-of-freedom Hyper6 C++ flight-dynamics model described by Zipfel [6].* Hyper6 is the most complete and rigorously tested version of the GHAME model. It incorporates a rotating elliptical Earth, propellant depletion with the associated variation in mass and moment of inertia, and the coupled translational and rotational equations of motion. The Simulink implementation used here is a direct port of these dynamics and has been extensively verified, providing a high-fidelity environment for evaluating the proposed control architecture.

*Simulations can be downloaded from: https://arc.aiaa.org/doi/suppl/10.2514/4.107535

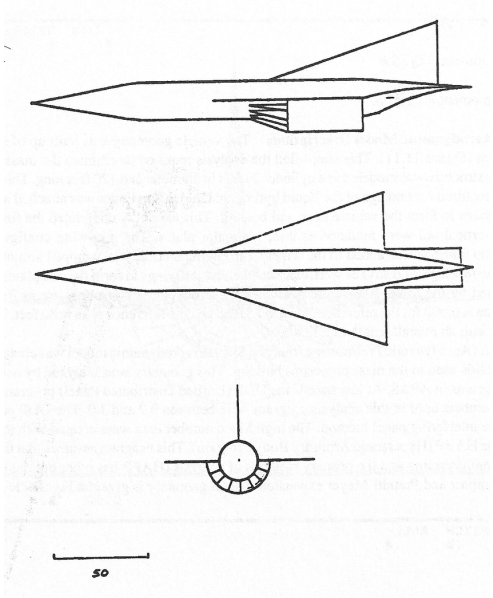


Fig. 1 Schematic configuration of the GHAME vehicle model

C. Aerodynamics and Forces

The GHAME aerodynamic database was developed by combining theoretical flow models and empirical data from multiple legacy hypersonic vehicle configurations. Longitudinal aerodynamic coefficients were constructed using blended data from the Space Shuttle Orbiter, lifting-body vehicles, a swept double-delta wing platform, and a 60° half-angle cone. For lateral-directional coefficients, Space Shuttle and double-delta data were used at Mach numbers below 8. Above this threshold, only Space Shuttle data were retained. Drag coefficients were scaled to match lift-to-drag ratios consistent with measured Space Shuttle performance. All coefficients were normalized using a fixed aerodynamic reference area and span. To extend the aerodynamic model into the hypersonic regime, a modified Newtonian impact theory was applied. This combination of analytical and empirical methods yields an aerodynamic model capable of capturing the nonlinearities, cross-coupling, and regime transitions critical for high-fidelity hypersonic flight simulation [5].

The aerodynamic force and moment coefficients are expressed as linear combinations of the control surface deflections, angular rates, sideslip angle β, and angle of attack α. In simulations with wind, the aerodynamic angles are used directly, whereas in wind-free conditions the corresponding kinematic angles apply. All aerodynamic data follow the GHAME reference model and were generated under the assumption of zero sideslip, although β may vary during simulation. Additionally, the coefficients C_{L_q}, C_{Y_0}, C_{Y_p}, C_{Y_r}, C_{l_0}, and C_{n_0} are negligible across the evaluated flight envelope [5]. The force coefficients are defined as follows:

$$\begin{aligned} C_L &= C_{L_0} + C_{L_\alpha}\,\alpha + C_{L_{\delta_e}}\,\delta_e, \\ C_Y &= C_{Y_\beta}\,\beta + C_{Y_{\delta_a}}\,\delta_a + C_{Y_{\delta_r}}\,\delta_r, \\ C_D &= C_{D_0} + C_{D_\alpha}\,\alpha, \end{aligned} \tag{1}$$

where δ_e is the elevator deflection, δ_a the aileron deflection, and δ_r the rudder deflection. The moment coefficients are calculated as follows:

$$\begin{aligned} C_m &= C_{m_0} + C_{m_\alpha}\,\alpha + C_{m_{\delta_e}}\,\delta_e + C_{m_q}\,\frac{q\,c_{\text{ref}}}{2V_{\text{TAS}}}, \\ C_l &= C_{l_\beta}\,\beta + C_{l_{\delta_a}}\,\delta_a + C_{l_{\delta_r}}\,\delta_r + C_{l_p}\,\frac{p\,b_{\text{ref}}}{2V_{\text{TAS}}} + C_{l_r}\,\frac{r\,b_{\text{ref}}}{2V_{\text{TAS}}}, \\ C_n &= C_{n_\beta}\,\beta + C_{n_{\delta_r}}\,\delta_r + C_{n_{\delta_a}}\,\delta_a + C_{n_p}\,\frac{p\,b_{\text{ref}}}{2V_{\text{TAS}}} + C_{n_r}\,\frac{r\,b_{\text{ref}}}{2V_{\text{TAS}}}. \end{aligned} \tag{2}$$

where p, q, and r denote the roll, pitch, and yaw rates of the vehicle with respect to Earth in body axes, respectively. Then c_{ref} is the reference chord, b_{ref} the reference span of the aircraft and V_{TAS} is the true airspeed. All partial derivatives with respect to angles have units of $1/^\circ$, while those with respect to angular rates have units of $1/\text{rad}$.

The lift coefficient C_L is defined as positive upwards and perpendicular to the velocity vector, the drag coefficient C_D is positive in the direction opposite to flight and the side force coefficient C_Y is positive to the right (starboard). Rolling moment C_l is positive for right–wing–down roll, pitching moment C_m for nose-up rotation, and yawing moment C_n for nose-right yaw. Since the aerodynamic force coefficients are expressed in stability coordinates, they must be transformed into body coordinates using α:

$$\begin{aligned} C_X &= -C_D \cos\alpha + C_L \sin\alpha, \\ C_Z &= -C_D \sin\alpha - C_L \cos\alpha. \end{aligned} \tag{3}$$

The total force vector $[f_{a,p}]^B$ is computed by combining the aerodynamic forces with the propulsive force. The total force vector in body coordinates is given by:

$$[f_{a,p}]^B \equiv \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^B = \begin{bmatrix} \bar{q}SC_X + f_P \\ \bar{q}SC_Y \\ \bar{q}SC_Z \end{bmatrix}, \tag{4}$$

where $\bar{q} = \frac{1}{2}\rho V_{\text{TAS}}^2$ is the dynamic pressure, in which ρ is the air density, S is the wing reference area and f_P is the propulsive force. The aerodynamic moment vector in the body frame is computed as:

$$[m_B]^B \equiv \begin{bmatrix} L \\ M \\ N \end{bmatrix}^B = \bar{q}S \begin{bmatrix} C_l b_{\text{ref}} \\ C_m c_{\text{ref}} \\ C_n b_{\text{ref}} \end{bmatrix}, \tag{5}$$

The aerodynamic coefficients are tabulated over a two–dimensional grid in angle of attack and Mach number, spanning $\alpha \in [-3^\circ, 21^\circ]$ and $M \in [0.4, 24]$. For simulation, most coefficients are interpolated independently on this grid. The only exceptions are the grouped forms $C_D = C_{D_0} + C_{D_\alpha}\alpha$, $C_M = C_{M_0} + C_{M_\alpha}\alpha$, and $C_L = C_{L_0} + C_{L_\alpha}\alpha$, which are combined first and then interpolated as single variables. This follows the implementation approach described in [7].

D. Actuator Model

The vehicle contains one rudder and two elevons located at the trailing edge of the wing. The elevons function simultaneously as elevator and aileron control surfaces. Their symmetric and antisymmetric combinations define the elevator and aileron control inputs, respectively, according to:

$$\delta_e = \frac{\delta_{vl} + \delta_{vr}}{2}, \quad \delta_a = \frac{\delta_{vl} - \delta_{vr}}{2}, \tag{6}$$

where δ_{vl} and δ_{vr} are the left and right elevon deflections, both defined as positive in the upward direction. The rudder deflection δ_r is defined to be positive for a trailing-edge right deflection. Each control surface is subject to deflection limits of $\pm 20^\circ$ and rate limits of $\pm 150^\circ/\text{s}$. To realistically capture actuator dynamics, all surfaces are modeled using a second-order system with rate and position limits. The actuator transfer function from commanded deflection δ_c to realized deflection δ is given by:

$$A(s) = \frac{\delta(s)}{\delta_c(s)} = \frac{\omega_{act}^2}{s^2 + 2\zeta_{act}\omega_{act}s + \omega_{act}^2}, \tag{7}$$

where $\omega_{act} = 50\,\text{rad/s}$ is the natural frequency and $\zeta_{act} = 0.707$ is the damping ratio. To enforce physical constraints, the implementation includes a saturation block limiting the input command δ_c to the allowable deflection range $\pm 20^\circ$ and a rate limiter which is calculated as $\frac{150\pi}{180}\frac{2\zeta_{act}}{\omega_{act}}$ applied to the difference in deflection , ensuring the actuator slew rate does not exceed prescribed limits.

E. Propulsion System Model

The engine models a generic combined-cycle propulsion system that switches automatically between turbojet, ramjet, and scramjet modes based on Mach number. The inlet is represented as a variable-geometry system whose effective capture area depends on Mach number and angle of attack, reflecting the changing shock structure on the forebody. The thrust is computed as a product of throttle setting, specific impulse, atmospheric density, true airspeed, and the inlet capture–area coefficient. Implementation details follow the formulation in [6]. The commanded throttle ranges from idle at 0.05 to a maximum value of 2.

Both the capture–area coefficient and the specific impulse are stored on lookup tables in Mach–angle–of–attack and Mach–throttle space, respectively, and are interpolated using the same method applied to the aerodynamic coefficients. The propulsion model also includes fuel consumption. The instantaneous thrust determines the fuel mass flow rate, which is integrated to update the remaining fuel. The total vehicle mass decreases accordingly. As fuel is depleted, the moment of inertia matrix is updated by interpolating linearly between the full–fuel and dry–mass inertia tensors. The simulation terminates automatically once the fuel level drops below zero.

F. Sensor Dynamics

The rate gyroscopes and accelerometers are modeled with explicit sensor dynamics. These devices are subject to two main effects: filtering to prevent aliasing and computation delays introduced by the sensor processor. The anti–aliasing filter is required because the Inertial Measurement Unit (IMU) signals are sampled at a fixed interval $\Delta t = 0.01$ s. The corresponding sampling frequency and Nyquist frequency are:

$$\omega_s = \frac{2\pi}{\Delta t} = 628.32 \text{ rad/s}, \qquad \omega_N = \frac{\omega_s}{2} = 314.16 \text{ rad/s}. \tag{8}$$

Any signal content above ω_N will fold into the lower frequency band, producing aliasing in the sampled data. Since sensor noise is broadband and not naturally band–limited, a low–pass filter is required before sampling. To guarantee sufficient attenuation of out–of–band noise, the cutoff is conservatively placed at half the Nyquist frequency, $\omega_a = \frac{\omega_N}{2} = 157.08$ rad/s. The anti–aliasing dynamics are represented as a first–order low–pass transfer function:

$$G_a(s) = \frac{\omega_a}{s + \omega_a}. \tag{9}$$

In addition to filtering, the IMU introduces a finite computation delay. This effect can be represented exactly by a pure time delay which is given by the nonrational transfer function:

$$G_{\text{SD}}(s) = e^{-\tau_{SD} s}, \tag{10}$$

which is referred to as the sensor delay transfer function. The exact duration of this sensor delay τ_{SD} is uncertain and therefore not modeled as a fixed constant, but it is expected to be on the order of one to several sampling intervals. Since its precise value cannot be specified a priori, the sensor delay is not fixed in the nominal simulation model. Instead, it is varied parametrically in the high-fidelity nonlinear simulation studies to evaluate how well the controller can tolerate such effects.

The overall sensor model therefore consists of a first-order low-pass filter for anti-aliasing, followed by the optional application of the sensor delay transfer function when robustness to measurement delays is investigated.

III. Flight Control System

In this framework, the main control approach is INDI. To the author's knowledge, INDI has not previously been applied to hypersonic vehicle control. It is well suited to this problem because hypersonic flight is characterized by large variations in aerodynamic coefficients, strong state coupling, and significant uncertainty in aerodynamic models. Classical NDI relies heavily on accurate modeling, which can limit performance under such conditions. INDI, on the other hand, updates the control effectiveness in real time using measured accelerations and angular rates, making it more robust to modeling errors and rapid aerodynamic changes. In the proposed four–loop architecture, INDI is used in the inner angular–rate loop, where fast and reliable response is essential for stabilizing the outer loops. For relationships that are purely kinematics, standard NDI is applied to achieve exact inversion without relying on aerodynamic models.

The cascaded Flight Control System (FCS) is organized into four nested loops, each operating at a distinct dynamic level. The innermost rate loop stabilizes the angular rates, while the successive outer loops regulate attitude, velocity, and finally position. This hierarchical structure, illustrated in Fig. 2, ensures that the fast inner–loop dynamics provide stability for the slower outer–loop objectives, thereby achieving a natural separation of time scales and facilitating controller design. The subscript K is used throughout to denote quantities derived from Earth–relative velocities rather than air–relative ones. This distinction is crucial because the onboard Inertial Navigation System (INS) provides only the Earth–relative velocity vector without direct wind measurements, and thus, it cannot compute the air–relative velocity. Consequently, all kinematic quantities used in the control system are expressed relative to Earth motion.

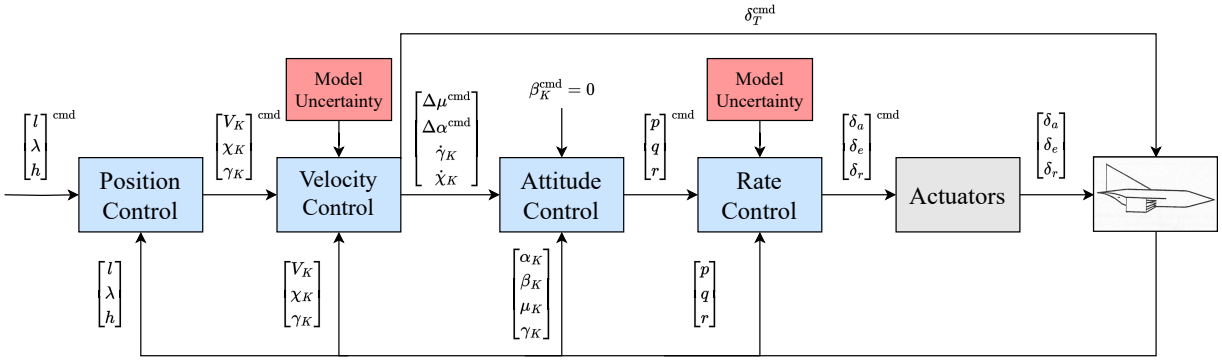


Fig. 2 Cascaded flight control architecture used in this work.

Model Assumptions for Control Law Development

The complete flight dynamics model captures all major physical effects relevant to the vehicle's motion. Such fidelity is essential for accurate simulation and reliable performance evaluation. However, formulating a control law directly from this high-fidelity model would introduce unnecessary complexity, complicating analytical derivations, controller tuning, and practical implementation.

A simplified dynamic model is therefore used for control law development. Only effects whose omission has a negligible impact on the vehicle's response are removed, resulting in a reduced-order model that preserves the dominant control-relevant dynamics. This simplification makes the analytical derivation of the control law feasible while maintaining sufficient accuracy within the intended flight envelope. The final control law is then validated using the complete high-fidelity model to confirm its performance under all relevant physical effects and interactions.

1) *Non-Rotating Earth*
The angular rate of Earth is assumed to be zero, $\omega_\oplus = 0$. The rotational velocity of the Earth introduces transport and Coriolis accelerations in the equations of motion. For sub-orbital flight in the sensible atmosphere, the transport acceleration is very small compared to aerodynamic and gravitational forces. Its maximum value, occurring at the equator, is about 0.034 m/s^2 ($3.5 \times 10^{-3}\, g_0$), and it decreases with latitude to zero at the poles. The Coriolis acceleration magnitude is bounded by $2\omega_\oplus V$; at $V = 300$ m/s it is 0.044 m/s^2 $\approx 4.5 \times 10^{-3}\, g_0$, at $V = 1500$ m/s it is 0.22 m/s^2 $\approx 2.2 \times 10^{-2}\, g_0$, and it only becomes significant ($\approx 0.116\, g_0$) at orbital velocities. For the control loops designed here, these values are smaller than typical modeling and actuation uncertainties, so omitting them simplifies the derivation without altering closed-loop behavior.

2) *Spherical Earth Model*
The high-fidelity model represents the Earth as an ellipsoid, but the reduced-order control model assumes a spherical Earth, because the difference between an ellipsoidal and spherical representation is negligible for the purposes of controller design. This simplification removes the distinction between geodetic and geocentric latitude and makes the Earth's radius constant rather than latitude dependent. Quantitatively, the WGS–84 equatorial and polar radii differ by approximately 21.4 km. At a flight altitude of 91 km, the equatorial and polar distances from Earth's center are 6469.137 km and 6447.752 km respectively, based on the WGS–84 radii. The

corresponding gravitational accelerations are $g_{\text{eq}}(91\text{ km}) \approx 9.516\text{ m s}^{-2}$ and $g_{\text{pol}}(91\text{ km}) \approx 9.547\text{ m s}^{-2}$, so the maximum latitude-dependent difference $\Delta g \approx 0.031\text{ m s}^{-2}$ corresponds to roughly 0.33%. This variation is small and has negligible influence on the closed-loop dynamics considered here.

A further simplification to a flat Earth would eliminate the curvature terms from the equations of motion. In particular, omitting the term $V^2/(gR)$ removes the centrifugal contribution that reduces the net force which the aerodynamic lift must balance. As the vehicle's speed approaches the orbital condition $V^2 = gR$, this centrifugal contribution becomes large enough to offset gravity, so the aerodynamic lift required to maintain altitude decreases toward zero. Because induced drag depends directly on lift, it also decreases toward zero in this regime. This reduction is a real physical effect that influences high-speed aerodynamic behavior. For this reason, the flat-Earth assumption is not adopted and the reduced-order control model retains the curvature terms.

3) *Central Gravity Field*
In the reduced-order control model, the gravitational field is taken to be purely central, meaning that zonal harmonics such as J_2 are omitted. Combined with the spherical Earth assumption, which already removes geometric latitude dependence, the only remaining variation in g is its inverse-square dependence on the distance from Earth's center.

Assuming constant gravity introduces a larger error when the $1/r^2$ decrease with altitude is ignored: evaluating $g(h) = g_0(R_0/(R_0 + h))^2$ at $h = 91$ km yields $g(91\text{ km}) \approx 9.530\text{ m s}^{-2}$, so the constant-gravity approximation overestimates gravitational acceleration by about 0.277 m s^{-2}, or approximately 2.83%. Although this altitude-induced variation is comparable to the error introduced by neglecting Coriolis effects at 1500 m/s, it is retained because it is straightforward to include and does not introduce any additional complexity into the reduced-order model.

4) *Stationary Atmosphere Model*
The reduced-order control model assumes steady atmospheric conditions and ignores changes in wind over time or location. At hypersonic speeds, the contribution of wind to the total airspeed is negligible compared to the vehicle's forward velocity, and its rotational influence on the body dynamics is effectively zero [8]. At subsonic speeds, [9] have shown that INDI exhibits excellent disturbance rejection properties against constant wind and wind gusts, further reducing the need to model wind explicitly during control synthesis. Moreover, the onboard INS has no direct access to wind information, making it unrealistic to incorporate wind into the controller as a measured quantity. It is therefore more appropriate to treat wind as a disturbance that must be rejected rather than as a modeled input. This assumption simplifies the controller and allows the design process to focus on the core vehicle dynamics. Wind and turbulence effects are, however, retained in the high-fidelity model to quantify performance in realistic atmospheric conditions.

5) *Coordinated Flight and Constant Speed*
The attitude control loop regulates the sideslip angle β_K to zero, so both the velocity and position control loops are formulated under the assumption $\beta_K = 0$. Similarly, the velocity control loop ensures that the commanded speed is reached and maintained, allowing the position control loop to be formulated under the additional assumption $V_K = V_K^{\text{cmd}}$. Together, these assumptions reduce the number of states that must be considered in the outer loops and provide the additional benefit of filtering out small-scale fluctuations in the computed velocity and sideslip angle, since both quantities are treated as constant.

6) *Small-Displacements*
The change in longitude l and latitude λ is negligible over the short time scales relevant to the velocity dynamics. As a result, $\dot{l}$ and $\dot{\lambda}$ are omitted from the attitude control loop design [10]. Numerical evaluation confirms that their effect on the velocity response is insignificant, with differences in the computed states below 10^{-6}.

7) *Accurate Control Effectiveness and Actuator Dynamics*
Throughout the control-law derivation, it is assumed that the estimated control effectiveness matrix matches the true system behavior and that the modeled actuator dynamics accurately represent the physical actuator response.

These assumptions ensure that the nonlinear inversion remains valid and that the incremental control updates behave as predicted by the reduced-order model.

IV. Control Law Design

This section presents the design of the dynamic inversion control laws for each loop in the cascaded architecture. It begins with the angular–rate control loop, where both NDI and INDI formulations are developed, including the method for reconstructing previous control inputs and the selection criteria for the low–pass filter used in the INDI implementation. The next part addresses the attitude–control loop, describing the derivation of the nonlinear inversion law used to track commanded attitudes. The velocity–control loop is then introduced, outlining its inversion law and its role in regulating airspeed and flight–path dynamics. Finally, the position–control loop is discussed, including its two functional modes for altitude and heading tracking.

A. Position Control Loop

The outermost loop governs the vehicle's position by generating reference commands for velocity, heading, and flight–path angle, which are tracked by the velocity–control loop. Its purpose is twofold. First, it enables trajectory tracking, allowing the vehicle to follow prescribed references in longitude, latitude, and altitude. Second, it enables test–case generation, where the performance of heading and altitude tracking is evaluated. The system is formulated using the state vector, output vector, and control input shown below, where λ denotes the vehicle's latitude, l its longitude, and h its altitude above a spherical Earth[†]:

$$\sum \begin{cases} \boldsymbol{x}_4 = [\lambda \;\; l \;\; h]^\top, \\ \boldsymbol{y}_4 = \boldsymbol{h}_4(\boldsymbol{x}_4) = \boldsymbol{x}_4, \\ \boldsymbol{u}_4 = [u_N \;\; u_E \;\; u_U]^\top. \end{cases} \tag{11}$$

In this context the heading angle χ_K is defined in the standard navigation convention: $\chi_K = 0^\circ$ corresponds to motion due north, $\chi_K = 90^\circ$ to motion due east and so forth. A positive latitude rate corresponds to northward motion, a positive longitude rate corresponds to eastward motion, and a positive altitude rate corresponds to an increase in altitude.

With this convention, the translational kinematics for a point–mass model on a spherical Earth are:

$$\dot{\lambda} = \frac{V_K^{\text{ref}} \cos\chi_K \cos\gamma_K}{R_0 + h}, \quad \dot{l} = \frac{V_K^{\text{ref}} \sin\chi_K \cos\gamma_K}{(R_0 + h)\cos\lambda}, \quad \dot{h} = V_K^{\text{ref}} \sin\gamma_K, \tag{12}$$

with R_0 the mean Earth radius, V_K^{ref} the commanded geographic speed, χ_K heading, and γ_K the flight–path angle. Due to the constant speed assumption V_K^{ref} is treated as constant.

For nonlinear dynamic inversion it is convenient to define an intermediate input vector:

$$\boldsymbol{u} = \begin{bmatrix} u_N \\ u_E \\ u_U \end{bmatrix} = \begin{bmatrix} V_K^{\text{ref}} \cos\chi_K \cos\gamma_K \\ V_K^{\text{ref}} \sin\chi_K \cos\gamma_K \\ V_K^{\text{ref}} \sin\gamma_K \end{bmatrix}, \tag{13}$$

where u_N, u_E, and u_U denote the linear velocity components of the vehicle in the geographic North–East-Up coordinate system. These components serve as intermediate control variables: when mapped through the spherical–dependent matrix $\boldsymbol{G}_4(\boldsymbol{x})$, they yield the correct time derivatives of (λ, l, h). The dynamics of the output vector can then be expressed in control–affine form as:

$$\dot{\boldsymbol{y}}_4 = \boldsymbol{f}_4(\boldsymbol{x}) + \boldsymbol{G}_4(\boldsymbol{x})\,\boldsymbol{u}_4. \tag{14}$$

From the control–affine representation the output dynamics can be written explicitly as:

$$\begin{bmatrix} \dot{\lambda} \\ \dot{l} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{R_0 + h} & 0 & 0 \\ 0 & \dfrac{1}{(R_0 + h)\cos\lambda} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_N \\ u_E \\ u_U \end{bmatrix}. \tag{15}$$

[†]For a spherical Earth the distinction between geocentric and geodetic disappears, so that $\lambda = \lambda_c = \lambda_d$ and $h = h_c = h_d$.

In nonlinear dynamic inversion, the control objective is expressed in terms of virtual controls, defined as the desired geographic rates $\boldsymbol{\nu}_4 = [\dot{\lambda} \;\; \dot{l} \;\; \dot{h}]^\top_{\text{cmd}}$ which are obtained from the reference trajectory $\boldsymbol{y}_4^{\text{cmd}} = [\lambda \;\; l \;\; h]^\top_{\text{cmd}}$ by suitable tracking laws. Enforcing $\dot{\boldsymbol{y}}_4 = \boldsymbol{\nu}_4$ gives:

$$\boldsymbol{\nu}_4 = \boldsymbol{G}_4(\boldsymbol{x})\, \boldsymbol{u}_4, \tag{16}$$

so that the required linear velocities are:

$$\boldsymbol{u}_4^{\text{cmd}} = \boldsymbol{G}_4^{-1}(\boldsymbol{x})\, \boldsymbol{\nu}_4, \tag{17}$$

which yields the explicit inversion law:

$$\begin{bmatrix} u_N \\ u_E \\ u_U \end{bmatrix}_{\text{cmd}} = \begin{bmatrix} (R_0 + h) & 0 & 0 \\ 0 & (R_0 + h)\cos\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \nu_\lambda \\ \nu_l \\ \nu_h \end{bmatrix}, \tag{18}$$

where ν_λ, ν_l, and ν_h denote the virtual control inputs for the longitudinal, latitudinal, and vertical channels, respectively. The commanded flight–path and heading angles are obtained directly by inverting the kinematic relations:

$$\gamma_K^{\text{cmd}} = \arcsin\left(\frac{u_U}{V_K^{\text{cmd}}}\right), \qquad \chi_K^{\text{cmd}} = \operatorname{atan2}\left(u_N, u_E\right). \tag{19}$$

These angles, together with the reference speed, form the commanded inputs to the velocity control loop.

For control–system testing, however, the objective is not to follow arbitrary reference trajectories, but to assess how well the system can maintain a prescribed heading or climb at a fixed climb rate. Therefore, two additional test modes are implemented to facilitate these specific scenarios.

1) **Mode 1: Constant–rate climb to altitude:**
   In this mode, the vehicle is commanded to reach a target altitude h^{ref} while maintaining a specified vertical rate $\dot{h}^{\text{ref}}$. The up–velocity component is therefore imposed directly as:

$$u_U^{\text{cmd}} = \dot{h}^{\text{ref}}. \tag{20}$$

   This value is substituted into Eq. (19) to compute the commanded flight–path angle γ_K^{cmd}. Once the altitude error is eliminated (i.e. $h = h^{\text{ref}}$), the climb rate command is set to zero. This mode can be combined with Mode 2.
2) **Mode 2: Heading–hold:**
   In this mode, when heading–hold is enabled (e.g. `mheading = 1`), any heading value calculated from Eq. (19) is overridden and the commanded heading is specified directly by the reference heading:

$$\chi_K^{\text{cmd}} = \chi_K^{\text{ref}}, \tag{21}$$

   while the commanded flight–path angle γ_K^{cmd} is still obtained from Eq. (19) to ensure that the altitude h tracks its reference h^{cmd}.

This completes the design of the position–control loop. Its feasibility depends on several conditions that ensure the inversion law is well defined and that the commanded angles remain physically meaningful. The input–gain matrix $\boldsymbol{G}_4(\boldsymbol{x})$ must be invertible, which requires $\cos\lambda \neq 0$ and therefore excludes the poles where the geographic formulation is ill defined. The inner velocity loop must accurately track the commanded reference speed, and the north and east velocity components (u_N, u_E) cannot both be zero, because the commanded heading χ_K^{cmd} is otherwise undefined.

B. Velocity Control Loop

The velocity control loop governs the translational dynamics of the vehicle and is responsible for tracking the flight-path command $\boldsymbol{u}_4^{\text{cmd}}$. It generates the throttle, δ_T, bank-angle, μ_K, and angle-of-attack, α_K, commands $\boldsymbol{u}_3^{\text{cmd}}$ for the inner attitude control loop. As in the previous section, the controller has access only to the kinematic, groundspeed-relative values α_K and μ_K provided by the INS, which means that the on-board model incurs an error when wind is present.

In contrast to the attitude and position loop, where the dynamics are purely kinematic, the translational dynamics depend directly on uncertain aerodynamic forces. This makes INDI particularly suitable for this loop. Its disturbance–rejection mechanism allows the controller to compensate for modeling errors and neglected effects, which appear

as disturbances on the slower time scale of the velocity dynamics [11]. The system is defined using the following state, output, and input variables:

$$\sum \begin{cases} \boldsymbol{x}_3 = [V_K \;\; \chi_K \;\; \gamma_K]^\top, \\ \boldsymbol{y}_3 = \boldsymbol{h}_3(\boldsymbol{x}_3) = \boldsymbol{x}_3, \\ \boldsymbol{u}_3 = [\delta_T \;\; \mu_K \;\; \alpha_K]^\top. \end{cases} \tag{22}$$

Here, V_K denotes the groundspeed. The angle of attack α_K is positive for nose–up motion relative to the groundspeed velocity vector, and the bank angle μ_K is positive for a right–wing–down rotation about the groundspeed velocity vector. The throttle command δ_T denotes the throttle setting and takes values in the interval $[0.05, 2]$. The governing dynamics are formulated by expressing Newton's Second Law in the air-path coordinate system. The resulting time derivatives of the velocity states are given by [12]:

$$\dot{\boldsymbol{x}}_3 = \boldsymbol{M}^{-1}\left([T]^{VB}[f_{a,p}]^B + [T]^{VG}[f_g]^G + m\boldsymbol{a}_{\text{corr}}\right), \text{ where } \boldsymbol{M} = \begin{bmatrix} m & 0 & 0 \\ 0 & mV_K\cos\gamma_K & 0 \\ 0 & 0 & -mV_K \end{bmatrix}. \tag{23}$$

Here, $[f]^B$ is the aero-propulsive force vector expressed in body coordinates, $[f_g]^G$ is the gravitational force vector in expressed in geocentric coordinates, $[T]^{VG}$ and $[T]^{VB}$ are transformation matrices from geocentric and body coordinates to the velocity coordinates, which are given by Eq. (26) and $\boldsymbol{a}_{\text{corr}}$ contains rotation and curvature corrections [12]. The spherical Earth assumption implies that the Earth radius function $R_0(\lambda_d)$ reduces to the constant mean spherical Earth radius R_0. In this case, the corrections are:

$$\boldsymbol{a}_{\text{corr}} = \begin{bmatrix} \omega_\oplus^2 R_0\cos\lambda\,(\sin\gamma_K\cos\lambda - \cos\gamma_K\sin\lambda\cos\chi_K) \\ 2\omega_\oplus V_K(\sin\lambda\cos\gamma_K - \cos\lambda\sin\gamma_K\cos\chi_K) + \dfrac{V_K^2}{R_0}\cos^2\gamma_K\tan\lambda\sin\chi_K + \omega_\oplus^2 R_0\cos\lambda\sin\lambda\sin\chi_K \\ -2\omega_\oplus V_K\cos\lambda\sin\chi_K - \dfrac{V_K^2}{R_0}\cos\gamma_K - \omega_\oplus^2 R_0\cos\lambda(\cos\lambda\cos\gamma_K + \sin\gamma_K\sin\lambda\cos\chi_K) \end{bmatrix}. \tag{24}$$

With the non-rotating Earth assumption, all terms containing $\omega_\oplus$ vanish. Together, this yields the simplified spherical-Earth form of the velocity dynamics:

$$\begin{bmatrix} \dot{V}_K \\ \dot{\chi}_K \\ \dot{\gamma}_K \end{bmatrix} = \begin{bmatrix} m & 0 & 0 \\ 0 & mV_K\cos\gamma_K & 0 \\ 0 & 0 & -mV_K \end{bmatrix}^{-1}\left([T]^{VB}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^B + m[T]^{VG}\begin{bmatrix} 0 \\ 0 \\ \frac{GM}{(R_0+h)^2} \end{bmatrix}^G + m\begin{bmatrix} 0 \\ \frac{V_K^2}{R_0}\cos^2\gamma_K\tan\lambda\sin\chi_K \\ -\frac{V_K^2}{R_0}\cos\gamma_K \end{bmatrix}\right). \tag{25}$$

In the gravitational force term, GM denotes the Earth's gravitational parameter. Although $[T]^{VB}$ is normally written in its full three–angle form, the condition $\beta_K = 0$ reduces it to the simplified expression shown below. The corresponding transformation matrix $[T]^{VG}$ taken from the same reference is placed alongside it [13].

$$[T]^{VB} = \begin{bmatrix} \cos\alpha_K & 0 & \sin\alpha_K \\ -\sin\alpha_K\sin\mu_K & -\cos\mu_K & \cos\alpha_K\sin\mu_K \\ -\sin\alpha_K\cos\mu_K & \sin\mu_K & \cos\alpha_K\cos\mu_K \end{bmatrix}, \qquad [T]^{VG} = \begin{bmatrix} \cos\gamma\cos\chi & \cos\gamma\sin\chi & -\sin\gamma \\ -\sin\chi & \cos\chi & 0 \\ \sin\gamma\cos\chi & \sin\gamma\sin\chi & \cos\gamma \end{bmatrix}. \tag{26}$$

Next, it is noted that μ_K, α_K, and δ_T do not appear in an affine form in Eq. (25) because the aerodynamic forces contain nonlinear trigonometric couplings with these inputs. The following step is to express the nonlinear dynamics in the standard input-affine form:

$$\dot{\boldsymbol{x}}_3 = \boldsymbol{f}_3(\boldsymbol{x}_3) + \boldsymbol{G}_3(\boldsymbol{x}_3)\,\boldsymbol{u}_3. \tag{27}$$

This can be partially accomplished by separating the force contributions to isolate the effects of bank angle, angle of attack, and thrust setting on the dynamics. The propulsive force is modeled as $f_p = \bar{q}SC_{X_T}\,\delta_T$, and, using the axial force definition in Eq. (4), the body–axis force relation becomes $[X]^B = \bar{q}S\left(C_X + C_{X_T}\delta_T\right)$, showing that thrust enters the dynamics in the same nondimensional manner as the aerodynamic coefficients. Since the velocity dynamics are formulated in the air–path coordinate system, the body–axis force vector is first transformed using $[T]^{VB}$. The resulting

expressions contain products of $\cos\alpha_K$, $\sin\alpha_K$, and the body–axis coefficients C_X and C_Z. To simplify these terms, the standard relations between body–axis and stability–axis coefficients are used:

$$C_D = -\left(\cos\alpha_K\, C_X + \sin\alpha_K\, C_Z\right), \qquad C_L = \sin\alpha_K\, C_X - \cos\alpha_K\, C_Z,$$

allowing the trigonometric combinations in the velocity–axis forces to be replaced by the physically meaningful drag and lift coefficients:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^V = [T]^{VB} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^B = \bar{q}S \begin{bmatrix} \cos\alpha_K\, C_{X_T}\delta_T - C_D \\ -\sin\mu_K \left(C_{X_T}\delta_T \sin\alpha_K + C_L\right) - \cos\mu_K\, C_Y \\ -\cos\mu_K \left(C_{X_T}\delta_T \sin\alpha_K + C_L\right) + \sin\mu_K\, C_Y \end{bmatrix}. \tag{28}$$

Finally, C_L and C_D are decomposed using the definitions in Eq. (1) so that the α_K-dependent terms required by the controller appear explicitly. Substituting the velocity–axis force vector from Eq. (28) into the translational dynamics of Eq. (25), and explicitly evaluating the resulting expressions, gives the nonlinear velocity equations shown in Eq. (29):

$$\begin{aligned}
\dot{V}_K &= \frac{\bar{q}S}{m}\left[C_{X_T}\delta_T\,\cos\alpha_K - C_{D_0} - C_{D_\alpha}\alpha_K\right] - g\sin\gamma_K, \\
\dot{\chi}_K &= \frac{\bar{q}S}{mV_K\cos\gamma_K}\left[-\sin\mu_K\left(C_{X_T}\delta_T\sin\alpha_K + C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha_K\right) - C_Y\cos\mu_K\right] + \frac{V_K}{R_0}\cos\gamma_K\tan\lambda\sin\chi_K, \\
\dot{\gamma}_K &= \frac{\bar{q}S}{mV_K}\left[\cos\mu_K\left(C_{X_T}\delta_T\sin\alpha_K + C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha_K\right) - C_Y\sin\mu_K\right] + \cos\gamma_K\left(\frac{V_K}{R_0} - \frac{g}{V_K}\right).
\end{aligned} \tag{29}$$

Wind effects are not included but could be incorporated, in which case all subscripts $_K$ change to the wind case $_A$. Looking at Eq. (29), the bank angle μ_K enters the dynamics only through the nonlinear terms $\sin\mu_K$ and $\cos\mu_K$. As a result, the velocity subsystem is not control-affine in μ_K, meaning it cannot be written in the form $f(x) + g_\mu(x)\,\mu_K$. This violates the standard requirement for applying NDI directly with μ_K treated as a control input.

INDI does not have this limitation [14]. By using a first-order Taylor expansion, it can produce an inversion law that works locally around the operating point. A standard Taylor series expansion is taken about the values at the previous sampling instant, where $\boldsymbol{x}_{3,0} = \boldsymbol{x}_3(t-\Delta t)$ and $\boldsymbol{u}_{3,0} = \boldsymbol{u}_3(t-\Delta t)$. This yields:

$$\begin{aligned}
\dot{\boldsymbol{x}}_3 &= \dot{\boldsymbol{x}}_{3,0} + \left.\frac{\partial\dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial\boldsymbol{x}_3}\right|_0 \Delta\boldsymbol{x}_3 + \left.\frac{\partial\dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial\boldsymbol{u}_3}\right|_0 \Delta\boldsymbol{u}_3 + O(\Delta\boldsymbol{x}_3^2), \\
&\approx \dot{\boldsymbol{x}}_{3,0} + \left.\frac{\partial\dot{\boldsymbol{x}}_3(\boldsymbol{x}_3,\,\boldsymbol{u}_3)}{\partial\boldsymbol{u}_3}\right|_0 \Delta\boldsymbol{u}_3,
\end{aligned} \tag{30}$$

here $\Delta\boldsymbol{x}_3 = \boldsymbol{x}_3 - \boldsymbol{x}_{3,0}$, $\Delta\boldsymbol{u}_3 = \boldsymbol{u}_3 - \boldsymbol{u}_{3,0}$, and the term $O(\Delta\boldsymbol{x}_3^2)$ collects all higher–order contributions.

As in the standard INDI argument, a time–scale separation assumption is introduced: over one sampling step the state increment is much smaller than the control increment, $\Delta\boldsymbol{x}_3 \ll \Delta\boldsymbol{u}_3$. Attitude dynamics typically evolve on a much faster time scale than the flight path states, so the term involving $\Delta\boldsymbol{x}_3$ and the higher–order contributions can be neglected [15]. From Eq. (29) and by seeing that $C_L = C_{L_0} + C_{L_{\delta_e}}\delta_e + C_{L_\alpha}\alpha_K$, the control effectiveness matrix is found by taking the partial derivatives of each state rate $\dot{\boldsymbol{x}}_3$ with respect to each control input $\boldsymbol{u}_3$, which yields:

$$\frac{\partial\dot{\boldsymbol{x}}_3}{\partial\boldsymbol{u}_3} = \frac{\bar{q}S}{mV_K}\begin{bmatrix} C_{X_T}\cos\alpha_K V_K & 0 & -V_K(C_{X_T}\delta_T\sin\alpha_K + C_{D_\alpha}) \\ \dfrac{-C_{X_T}\sin\mu_K\sin\alpha_K}{\cos\gamma_K} & \dfrac{-\cos\mu_K(C_L + C_{X_T}\delta_T\sin\alpha_K) + C_Y\sin\mu_K}{\cos\gamma_K} & \dfrac{-\sin\mu_K(C_{X_T}\delta_T\cos\alpha_K + C_{L_\alpha})}{\cos\gamma_K} \\ C_{X_T}\cos\mu_K\sin\alpha_K & -\sin\mu_K(C_L + C_{X_T}\delta_T\sin\alpha_K) - C_Y\cos\mu_K & \cos\mu_K(C_{X_T}\delta_T\cos\alpha_K + C_{L_\alpha}) \end{bmatrix}. \tag{31}$$

To evaluate these partial derivatives, the vehicle is assumed to be in vertical force equilibrium, expressed in a form consistent with the spherical Earth model:

$$\bar{q}S\cos\mu_K(C_L + C_{X_T}\delta_T\sin\alpha_K) = m\cos\gamma_K\left(g - \frac{V_K^2}{R_0}\right). \tag{32}$$

This condition makes it possible to replace the aerodynamic term on the left in Eq. (32) by the gravitational–curvature term on the right. Additionally, the influence of thrust on lift and drag derivatives is neglected, as aerodynamic forces dominate in magnitude:

$$C_{L_\alpha}\bar{q}S \gg \bar{q}SC_{X_T}\delta_T\cos\alpha_K, \qquad C_{D_\alpha}\bar{q}S \gg \bar{q}SC_{X_T}\delta_T\sin\alpha_K.$$

The lateral coefficient C_Y is also neglected. With $\beta_K = 0$ its magnitude is much smaller than the lift- and thrust-related terms, making its contribution to the velocity dynamics negligible. Substituting Eq. (32) into Eq. (31) and removing negligible terms yields the following expression for the control effectiveness matrix:

$$\boldsymbol{G}_3(\boldsymbol{x}_3,\, u_3) \approx \frac{\bar{q}S}{mV_K}\begin{bmatrix} C_{X_T}\cos\alpha_K V_K & 0 & -C_{D_\alpha}V_K \\ -\dfrac{C_{X_T}\sin\mu_K\sin\alpha_K}{\cos\gamma_K} & -\dfrac{m}{\bar{q}S}\left(g-\dfrac{V_K^2}{R_0}\right) & -\dfrac{C_{L_\alpha}\sin\mu_K}{\cos\gamma_K} \\ C_{X_T}\cos\mu_K\sin\alpha_K & -\dfrac{m\tan\mu_K\cos\gamma_K}{\bar{q}S}\left(g-\dfrac{V_K^2}{R_0}\right) & C_{L_\alpha}\cos\mu_K \end{bmatrix}. \tag{33}$$

A virtual input $\boldsymbol{\nu}_3 = [\nu_V \;\; \nu_\chi \;\; \nu_\gamma]^\top$ is now defined to represent the desired rates of the velocity states. Linearizing the dynamics around the previously measured operating point yields the incremental inversion law:

$$\boldsymbol{u}_3^{\text{cmd}} = \boldsymbol{u}_{3,0} + \boldsymbol{G}_3^{-1}(\boldsymbol{x}_{3,0},\, \boldsymbol{u}_{3,0})\,\left(\boldsymbol{\nu}_3 - \dot{\boldsymbol{x}}_{3,0}\right). \tag{34}$$

The subscript "0" refers to measurements from the previous timestep. The matrix $\boldsymbol{G}_3$ becomes non-invertible only in vertical flight ($\gamma_K = \pm 90°$), knife-edge flight ($\mu_K = \pm 90°$), or in the limiting case $g = V_K^2/R_0$, where centrifugal force cancels gravity and lift can no longer steer the velocity vector.

C. Attitude Control Loop

The objective of the attitude loop is to follow the commands $\boldsymbol{u}_3^{\text{cmd}}$ generated by the velocity control loop. It does so by producing the angular-rate commands $\boldsymbol{u}_2^{\text{cmd}}$ for the inner rate loop. The loop regulates the aerodynamic angles μ_K, α_K, and β_K, which are used instead of Euler angles because they are directly provided by the INS and match the commanded quantities from the velocity loop. For the control design, the attitude subsystem is written as:

$$\sum \begin{cases} \boldsymbol{x}_2 = [\mu_K \;\; \alpha_K \;\; \beta_K]^\top, \\ \boldsymbol{y}_2 = \boldsymbol{h}_2(\boldsymbol{x}_2) = \boldsymbol{x}_2, \\ \boldsymbol{u}_2 = [p \;\; q \;\; r]^\top. \end{cases} \tag{35}$$

Here, $[p \;\; q \;\; r]$ are the inertial angular rates. For the NDI design, the attitude kinematics are written in the control-affine form:

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(\boldsymbol{x}_2) + \boldsymbol{G}_2(\boldsymbol{x}_2)\,\boldsymbol{u}_2. \tag{36}$$

A set of dynamic attitude equations suitable for expressing this form was derived by Mooij [16]. The general dynamics of the aerodynamic angles for a rotating, geodetic Earth are given by:

$$\begin{aligned}
\dot{\mu}_K &= -\dot{\alpha}_K\sin\beta_K + \dot{\chi}_K\sin\gamma_K + \dot{\lambda}\sin\chi_K\cos\gamma_K - (\dot{l}+\omega_\oplus)(\cos\lambda\cos\chi_K\cos\gamma_K + \sin\lambda\sin\gamma_K) \\
&\quad + p\cos\alpha_K\sin\beta_K + q\sin\beta_K + r\sin\alpha_K\cos\beta_K, \\
\dot{\alpha}_K &= -\frac{\sin\mu_K}{\cos\beta_K}\left[\,\dot{\chi}_K\cos\gamma_K - \dot{\lambda}\sin\chi_K\sin\gamma_K + (\dot{l}+\omega_\oplus)(\cos\lambda\,\cos\chi_K\sin\gamma_K - \sin\lambda\cos\gamma_K)\right] \\
&\quad - \frac{\cos\mu_K}{\cos\beta_K}\left[\,\dot{\gamma}_K - \dot{\lambda}\cos\chi_K - (\dot{l}+\omega_\oplus)\cos\lambda\sin\chi_K\right] - p\cos\alpha_K\tan\beta_K + q - r\sin\alpha_K\tan\beta_K, \\
\dot{\beta}_K &= -\sin\mu_K\left[\,\dot{\gamma}_K - \dot{\lambda}\,\cos\chi_K - (\dot{l}+\omega_\oplus)\cos\lambda\sin\chi_K\right] + \cos\mu_K\left[\,\dot{\chi}_K\cos\gamma_K - \dot{\lambda}\sin\chi_K\sin\gamma_K\right] \\
&\quad + \cos\mu_K\left[(\dot{l}+\omega_\oplus)(\cos\lambda\cos\chi_K\sin\gamma_K - \sin\lambda\cos\gamma_K)\right] + p\sin\alpha_K - r\cos\alpha_K,
\end{aligned} \tag{37}$$

where l is the longitude, λ is the geodetic latitude, $\dot{l}$ and $\dot{\lambda}$ are their time derivatives, and $\omega_\oplus$ is Earth's angular rate.

The full expression in Eq. (37) contains several terms that have negligible influence on the control law. The body–rate contributions must be retained, and the flight–path rates $[\dot{\gamma}_K \;\; \dot{\chi}_K]$ remain significant for maneuvering. Under the *Small-Displacement Assumption*, variations in l and λ_d and their rates $(\dot{l}, \dot{\lambda}_d)$ are neglected [10]. Consistent with the *Non-rotating Earth Assumption*, the Earth's rotation rate is set to $\omega_\oplus = 0$ for control design.

For dynamic inversion, each aerodynamic–angle derivative must be expressed explicitly in terms of measurable quantities rather than in terms of other angle derivatives. In Mooij's formulation, the equation for $\dot{\mu}_K$ contains the term $\dot{\alpha}_K$, whose value is already given by the second line of Eq. (37). Substituting this expression into the $\dot{\mu}_K$ equation removes the derivative coupling and yields a set of equations in which $\dot{\mu}_K$, $\dot{\alpha}_K$, and $\dot{\beta}_K$ depend only on INS–measurable states, flight–path rates, and body angular rates. Because the resulting dynamics contain only kinematic terms, they admit a direct nonlinear dynamic inversion without the need for an incremental control law. The resulting equations are:

$$\begin{aligned}
\dot{\mu}_K &= \dot{\chi}_K(\sin\gamma_K + \sin\mu_K \tan\beta_K \cos\gamma_K) + \dot{\gamma}_K \cos\mu_K \tan\beta_K + p\frac{\cos\alpha_K}{\cos\beta_K} + r\frac{\sin\alpha_K}{\cos\beta_K},\\
\dot{\alpha}_K &= -\dot{\chi}_K \frac{\sin\mu_K \cos\gamma_K}{\cos\beta_K} - \dot{\gamma}_K \frac{\cos\mu_K}{\cos\beta_K} - p\cos\alpha_K \tan\beta_K + q - r\sin\alpha_K \tan\beta_K,\\
\dot{\beta}_K &= \dot{\chi}_K \cos\mu_K \cos\gamma_K - \dot{\gamma}_K \sin\mu_K + p\sin\alpha_K - r\cos\alpha_K.
\end{aligned} \tag{38}$$

Equation 38 can be separated into the contribution from the body angular rates and the terms arising from the flight–path kinematics which yields the following control-affine form from Eq. 36:

$$\begin{bmatrix}\dot{\mu}_K\\ \dot{\alpha}_K\\ \dot{\beta}_K\end{bmatrix} = \underbrace{\begin{bmatrix}\sin\gamma_K + \sin\mu_K\tan\beta_K\cos\gamma_K & \cos\mu_K\tan\beta_K\\ -\frac{\sin\mu_K\cos\gamma_K}{\cos\beta_K} & -\frac{\cos\mu_K}{\cos\beta_K}\\ \cos\mu_K\cos\gamma_K & -\sin\mu_K\end{bmatrix}\begin{bmatrix}\dot{\chi}_K\\ \dot{\gamma}_K\end{bmatrix}}_{\boldsymbol{f}_2(\boldsymbol{x}_2)} + \underbrace{\begin{bmatrix}\frac{\cos\alpha_K}{\cos\beta_K} & 0 & \frac{\sin\alpha_K}{\cos\beta_K}\\ -\cos\alpha_K\tan\beta_K & 1 & -\sin\alpha_K\tan\beta_K\\ \sin\alpha_K & 0 & -\cos\alpha_K\end{bmatrix}}_{\boldsymbol{G}_2(\boldsymbol{x}_2)}\underbrace{\begin{bmatrix}p\\ q\\ r\end{bmatrix}}_{\boldsymbol{u}_2}. \tag{39}$$

The influence of $\dot{\gamma}_K$ and $\dot{\chi}_K$ can alternatively be expressed using the specific forces measured by the onboard accelerometers, from which the contributions of weight, lift, and side force to the flight–path angles are reconstructed [17].

A virtual input $\boldsymbol{\nu}_2 = [\nu_\mu \;\; \nu_\alpha \;\; \nu_\beta]^\top$ is defined to represent the desired attitude angle rates. By enforcing $\dot{\boldsymbol{x}}_2 = \boldsymbol{\nu}_2$ and inverting Eq. (36), the required body rates become:

$$\boldsymbol{u}_2^{\text{cmd}} = \boldsymbol{G}_2^{-1}(\boldsymbol{x}_2)\,(\boldsymbol{\nu}_2 - \boldsymbol{f}_2(\boldsymbol{x}_2))\,. \tag{40}$$

The matrix $\boldsymbol{G}_2$ is invertible for all α_K provided $\cos\beta_K \neq 0$, i.e. $\beta_K \neq \frac{\pi}{2} + n\pi,\ n \in \mathbb{Z}$. This inversion yields the commanded body rates required to track the desired changes in attitude angles.

D. Angular Rate Control Loop

The angular rate loop forms the innermost part of the FCS. It is responsible for tracking angular rate commands $\boldsymbol{u}_2^{\text{cmd}}$ generated by the attitude controller. It calculates the actuator commands The system is described using the following state, output, and input vectors:

$$\sum \quad \begin{cases}\boldsymbol{x}_1 = [p \;\; q \;\; r]^\top,\\ \boldsymbol{y}_1 = \boldsymbol{h}_1(\boldsymbol{x}_1) = \boldsymbol{x}_1,\\ \boldsymbol{u}_1 = [\delta_a \;\; \delta_e \;\; \delta_r]^\top,\end{cases} \tag{41}$$

where p, q, r are the inertial body rates, and the control inputs $\delta_a, \delta_e, \delta_r$ are the aileron, elevator, and rudder deflections, respectively. The variable–mass attitude equations of motion for a symmetric aircraft, written in the body coordinate system with superscripts omitted for clarity, are:

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{I}^{-1}\left(\boldsymbol{m}_B - \dot{\boldsymbol{I}}\boldsymbol{x}_1 - \boldsymbol{x}_1 \times I\,\boldsymbol{x}_1\right). \tag{42}$$

where I are the vehicle's moment of inertia and m_B are the external moments from Eq. (5). These moments are the sum of the aerodynamic moments $\boldsymbol{m}_a$ and the control–generated moments $\boldsymbol{m}_c$. Assuming linear control derivatives $(\boldsymbol{m}_c)_\delta = \frac{\partial}{\partial\delta}\boldsymbol{m}_c$, the total moment becomes linear in the deflection inputs:

$$\boldsymbol{m}_B = \boldsymbol{m}_a + (\boldsymbol{m}_c)_\delta\,\boldsymbol{u}_1. \tag{43}$$

Substituting Eq. (43) into the rotational equation of motion and solving for the input-affine form yields:

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{I}^{-1}\left[-\dot{\boldsymbol{I}}\boldsymbol{x}_1 - \boldsymbol{x}_1 \times \boldsymbol{I}\,\boldsymbol{x}_1 + \boldsymbol{m}_a\right] + \boldsymbol{I}^{-1}(\boldsymbol{m}_c)_\delta\,\boldsymbol{u}_1. \tag{44}$$

The control effectiveness matrix and plant dynamics are consequently:

$$\boldsymbol{G}_1 = \boldsymbol{I}^{-1}(\boldsymbol{m}_c)_\delta, \quad \text{and} \quad \boldsymbol{f}_1(\boldsymbol{x}_1) = \boldsymbol{I}^{-1}\left(-\dot{\boldsymbol{I}}\,\boldsymbol{x}_1 - \boldsymbol{x}_1 \times \boldsymbol{I}\,\boldsymbol{x}_1 + \boldsymbol{m}_a\right). \tag{45}$$

To reduce the dependency of the control law on the full nonlinear model, the angular–rate loop also employs an INDI formulation. The key idea here is to replace part of the model information by locally measured state–rate data, thereby improving robustness to modeling uncertainty. Take the local Taylor series expansion on the control-affine Eq. (44) and remove state dependent contributions by using the time scale separation assumption. This reduces Eq. (44) to the incremental approximation:

$$\dot{\boldsymbol{x}}_1 \approx \dot{\boldsymbol{x}}_{1,0} + \boldsymbol{G}_1\,\Delta\boldsymbol{u}_1. \tag{46}$$

The matrix $\boldsymbol{G}_1$ is treated as constant over the interval and is obtained from the OBM, which updates it online. To enforce desired angular acceleration dynamics, a virtual control input $\boldsymbol{\nu}_1$ is introduced. Substituting this into the approximation Eq. (46) and solving for $\boldsymbol{u}_1$ yields the INDI control law:

$$\boldsymbol{u}_1^{\text{cmd}} = \boldsymbol{u}_{1,0} + \boldsymbol{G}_1^{-1}\left(\boldsymbol{\nu}_1 - \dot{\boldsymbol{x}}_{1,0}\right). \tag{47}$$

This control law preserves the input-affine structure of feedback linearization while avoiding explicit dependence on the full nonlinear model. Provided that measurements of $\dot{\boldsymbol{x}}_1(t-\Delta t)$ and $\boldsymbol{u}_1(t-\Delta t)$ are accurate and the delay Δt is sufficiently small, the INDI formulation achieves robust inner-loop angular rate tracking. All quantities are expressed in the body-axis coordinate system:

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{zx} & 0 & I_{zz} \end{bmatrix}, \quad (\boldsymbol{m}_c)_\delta = \bar{q}\,S \begin{bmatrix} b\,C_{l_{\delta_a}} & 0 & b\,C_{l_{\delta_r}} \\ 0 & \bar{c}\,C_{m_{\delta_e}} & 0 \\ b\,C_{n_{\delta_a}} & 0 & b\,C_{n_{\delta_r}} \end{bmatrix}, \quad \boldsymbol{m}_a = \bar{q}\,S \begin{bmatrix} b\,C_{l_a} \\ \bar{c}\,C_{m_a} \\ b\,C_{n_a} \end{bmatrix}.$$

E. State Estimation, Filtering, and Synchronization

In the SISO implementation, both the delayed control input δ_0 and the delayed angular acceleration $\dot{\omega}_0$ must be reconstructed, since neither quantity is directly measurable. The delayed actuator deflection is obtained by passing the commanded signal $\delta_{\text{cmd}}(t)$ through the second–order actuator model in Eq. (7) together with the fixed delay Δt:

$$\delta_0 = \delta(t-\Delta t) = \mathcal{L}^{-1}\left\{e^{-\Delta t s} A(s)\,\delta_{\text{cmd}}(s)\right\}. \tag{48}$$

The same delay appears in the angular–acceleration term because the INDI control law is linearized about a previously measured operating point. In the Discrete Time (DT) implementation, this delay coincides with the Flight Control Computer (FCC) sampling time and is represented by a single Unit Delay (UD) block. The delayed angular acceleration follows from differentiation of the measured angular rate:

$$\dot{\omega}_0 = \dot{\omega}(t-\Delta t) = \mathcal{L}^{-1}\left\{s\,e^{-\Delta t s}\,\omega(s)\right\}. \tag{49}$$

Direct numerical differentiation amplifies sensor noise, so the angular–rate measurement is first passed through the second–order low–pass filter:

$$H(s) = \frac{\omega_H^2}{s^2 + 2\zeta_H\omega_H s + \omega_H^2}, \qquad \text{using} \quad \omega_H = 25\,\text{rad/s},\ \zeta_H = 1, \tag{50}$$

This filter produces the angular–acceleration estimate:

$$\dot{\omega}_{f_0} = \mathcal{L}^{-1}\left\{s\,H(s)\,e^{-\Delta t s}\,\omega(s)\right\}. \tag{51}$$

The filter introduces a phase lag that would desynchronize $\dot{\omega}_{f_0}$ from the unfiltered actuator signal δ_0. Because INDI relies on a first–order Taylor expansion evaluated at a common delayed operating point, both paths must experience the same temporal shift. The actuator path is therefore filtered identically:

$$\delta_{f_0} = \mathcal{L}^{-1}\left\{H(s)\,e^{-\Delta t s}\,A(s)\,\delta_{\text{cmd}}(s)\right\}. \tag{52}$$

With the two signals synchronized, the filtered INDI control law becomes:

$$\delta_{\text{cmd}} = \delta_{f_0} + G_{1,0}^{-1}\left(\nu_1 - \dot{\omega}_{f_0}\right). \tag{53}$$

The complete structure, including the filtering and reconstruction paths, is illustrated in Fig. 3. The diagram summarizes the roles of the actuator dynamics $A(s)$, the noise–attenuating filter $H(s)$, and the fixed delay Δt within the INDI formulation.

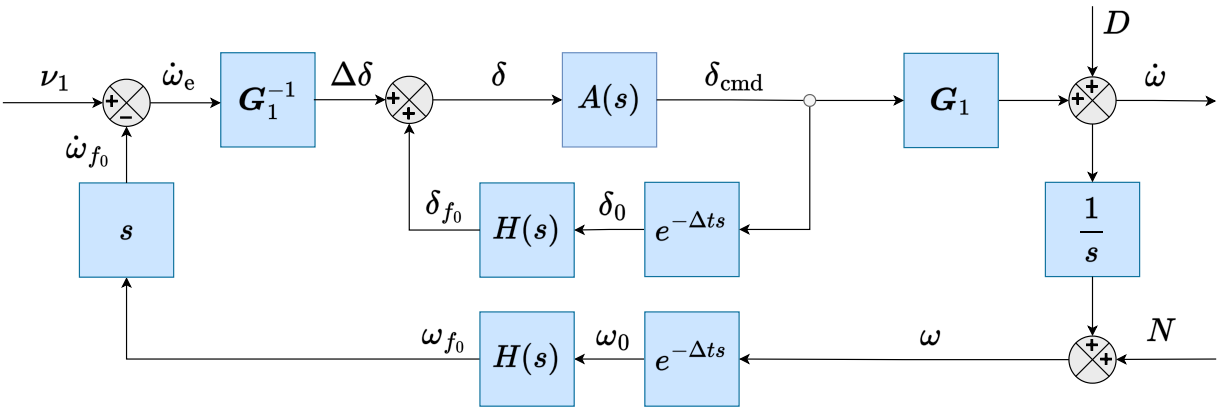


Fig. 3 Frequency-domain implementation of the INDI control loop for angular rate tracking. The loop includes angular acceleration estimation, actuator modeling, and consistent filtering of both the feedback and control branches.

V. Linear Controller Design

Nonlinear dynamic inversion reshapes the system into approximate chains of integrators, forming the basis for hierarchical feedback design. A time–scale separation assumption is adopted so that the faster inner loops appear instantaneous from the perspective of the slower outer loops, allowing each loop to be designed independently.

With the dynamics rendered integrator–like and the hierarchy established, the remaining task is to specify the desired transient and steady–state characteristics through linear controllers placed on top of the inverted dynamics. The hierarchical control architecture follows a set of design requirements:

1) **Damping requirements:** The rate loop is designed to be critically damped ($\zeta_\omega = 1$) with no overshoot in the angular rate. The outer loops are designed to be well damped, with

$$\zeta_\theta = 0.9, \qquad \zeta_V = 0.7, \qquad \zeta_X = 0.9,$$

ensuring fast responses with minimal overshoot.
2) **Rate Requirement:** The rate loop bandwidth BW_ω is selected as high as possible subject to the critical–damping requirement, as a faster inner loop eases the design constraints imposed on the slower outer loops.
3) **Time–scale separation:** Each loop is designed to be significantly slower than the loop beneath it. A practical and commonly used choice is a bandwidth ratio of approximately four between adjacent loops, leading to:

$$\text{BW}_\theta \leq \frac{\text{BW}_\omega}{4}, \qquad \text{BW}_V \leq \frac{\text{BW}_\theta}{4}, \qquad \text{BW}_X \leq \frac{\text{BW}_V}{4}.$$

4) **Robustness margins:** All loops must achieve a phase margin of at least 30° and a gain margin of at least 6 dB.

Rate Loop

The design of the angular–rate controller follows from the observation that, under ideal INDI and in the absence of disturbances and sensor noise, the airframe dynamics are cancelled by the inversion, leaving only the actuator dynamics

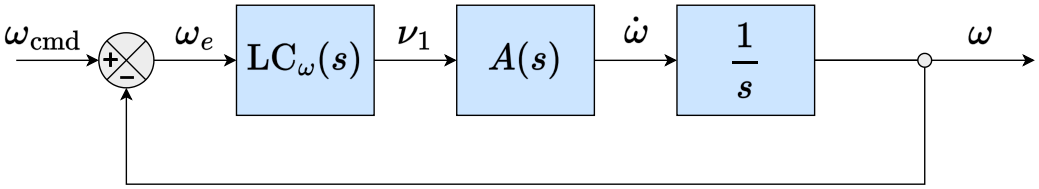


Fig. 4 Idealized closed–loop structure of the angular–rate control loop.

$A(s)$ in the loop, and that rapid disturbance rejection together with effective noise suppression can be ensured by a suitable choice of the noise filter $H(s)$ [18].

Figure 4 illustrates the corresponding idealized closed–loop structure of the angular–rate controller, where the linear controller $LC_\omega(s)$ generates the virtual input ν_1 from the rate error, and the integrator yields the angular rate ω. Thus, in this simplified setting the closed–loop $\omega_{\text{cmd}}(s)$ to $\omega(s)$ is:

$$H_\omega(s) = \frac{A(s)LC_\omega(s)}{s + A(s)LC_\omega(s)}. \tag{54}$$

The actuator dynamics are often assumed to be sufficiently fast to justify the approximation $A(s) \approx 1$ over the relevant frequency range [18]. This assumption holds for systems with very high actuator bandwidths, but becomes inaccurate when the actuator bandwidth is more moderate. In such cases, neglecting $A(s)$ leads to optimistic predictions of loop performance, since the actuator still limits stability and achievable bandwidth. For the present aircraft, the actuator bandwidth of approximately 50 rad/s is not high enough for $A(s) \approx 1$ to be a valid approximation within the desired control bandwidth. A substantially lower rate-loop bandwidth would be required to make actuator dynamics negligible, which is undesirable for performance. Therefore, the actuator model $A(s)$ is included explicitly in both the controller design and the stability analysis, ensuring that achievable bandwidth and robustness are evaluated with the true actuator limitations in mind.

The controller is designed for the third-order plant $P(s) = A(s)\,\frac{1}{s}$, where the actuator dynamics $A(s)$ from Eq. (7) act in series with the integrator, as shown in Fig. 4. For this plant, a proportional control strategy is adopted, using the control law:

$$LC_\omega(s) = K_p.$$

From the requirements it follows that the proportional gain K_p must be chosen to maximize the closed–loop bandwidth while ensuring negligible overshoot and sufficient robustness. To this end, a bisection search is employed to determine the largest feasible gain subject to the constraints of at most 0.1% overshoot and a minimum phase margin of 30°. The search begins with a broad interval $[K_{p,\min}, K_{p,\max}]$. At each iteration, the midpoint value of K_p is evaluated by forming the closed-loop transfer function:

$$T(s) = \frac{K_p P(s)}{1 + K_p P(s)}. \tag{55}$$

From this closed–loop system, the overshoot is extracted from the time–domain step response, while the phase margin is obtained from the frequency response of the open–loop transfer function $L(s) = K_p P(s)$. If the constraints are satisfied, the gain is deemed feasible and the lower bound of the search interval is raised, otherwise, the upper bound is reduced. This procedure continues until the interval width falls below the prescribed tolerance ε, guaranteeing convergence to the largest feasible proportional gain. The result is expressed as:

$$K_p^\star = \max\left\{K_p \;\middle|\; \text{OS} \le 0.1\%,\ \text{PM} \ge 30^\circ\right\}, \tag{56}$$

which yields the fastest critically damped response consistent with the actuator dynamics described in Eq. (7). The bisection search is characterized by the iteration bound:

$$N = \left\lceil \log_2\left(\frac{K_{p,\max} - K_{p,\min}}{\varepsilon}\right)\right\rceil, \tag{57}$$

where $K_{p,\min}$ and $K_{p,\max}$ represent the initial search interval. With $K_{p,\min} = 10^{-3}$, $K_{p,\max} = 10^{2}$, and a convergence tolerance of $\varepsilon = 10^{-6}$, the method is guaranteed to converge, if a solution exists, within $N = 27$ iterations. For the plant under consideration, the optimized gain and maximum achievable bandwidth and phase margin are:

$$LC_\omega(s) = 13.5625, \qquad \mathrm{BW}_\omega = 24.96 \text{ rad/s} \qquad \mathrm{PM} = 67.57 \text{ deg.} \tag{58}$$

This yields the following closed loop relation:

$$H_\omega(s) = \frac{33906}{(s + 24.96)(s^2 + 45.74s + 1358)}. \tag{59}$$

Allowing a small overshoot would enable a higher crossover frequency and bandwidth at the expense of reduced phase margin[‡]. It should also be noted that digital implementation effects, such as sampling and computation delays, introduce extra phase lag. For this reason, retaining sufficient phase and gain margins is essential to ensure robustness against these delays.

Table 1 Pole locations, natural frequencies, and damping ratios of the angular rate dynamics

Real ± Imag	ω_n, rad/s	ζ	Type
$-22.87 \pm 28.90i$	36.85	0.62	Actuator poles
$-24.96 \pm 0.00i$	24.96	1.00	Rate pole

Attitude Loop

For the linear controller in the attitude loop, the first step is to obtain the closed–loop expression of the system, shown schematically in Fig. 5. Here, standard NDI is used rather than the incremental form. Unlike INDI, where additional dynamics appeared as disturbances to reject, the NDI formulation retains these explicitly. In this case it is the term $\boldsymbol{f}_2$, representing the flight–path dynamics given in Eq. (39):

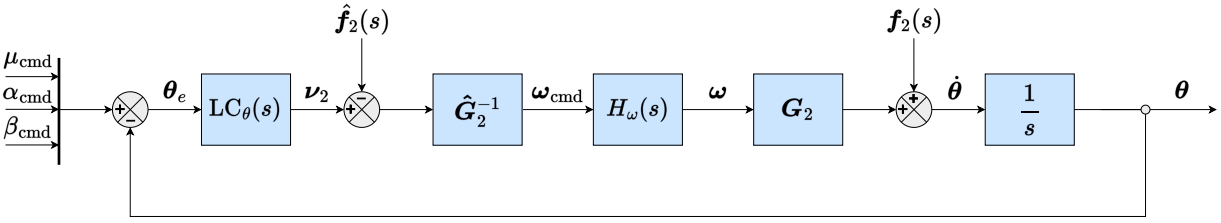


Fig. 5 Closed-loop architecture for attitude control using model-based inversion and linear compensation.

The commands for μ_{cmd} and α_{cmd} are supplied by the outer velocity loop, while the sideslip reference is fixed at $\beta_{\mathrm{cmd}} = 0$ to enforce coordinated flight. A linear controller $LC_\theta(s)$ regulates the commanded attitude error $\boldsymbol{\theta}_e$ to produce the virtual control vector $\boldsymbol{\nu}_2(s)$, which defines the desired attitude rate.

The angular rate command is obtained by subtracting the estimated kinematic term $\hat{\boldsymbol{f}}_2(s)$ from $\boldsymbol{\nu}_2(s)$ and mapping the result through the inverse of the kinematic matrix. The matrices $\boldsymbol{G}_2(s)$ and $\hat{\boldsymbol{G}}_2(s)$ map angular rates to attitude rates, and since this relation is known analytically, they are exactly equal. The resulting closed-loop relation for $\boldsymbol{\theta}(s)$ is:

$$\boldsymbol{\theta}(s) = \frac{H_\omega(s)\, LC_\theta(s)}{s + H_\omega(s)\, LC_\theta(s)}\, \boldsymbol{\theta}_{\mathrm{cmd}}(s) - \frac{H_\omega(s)}{s + H_\omega(s)\, LC_\theta(s)}\, \hat{\boldsymbol{f}}_2(s) + \frac{1}{s + H_\omega(s)\, LC_\theta(s)}\, \boldsymbol{f}_2(s). \tag{60}$$

At this stage, the bandwidth separation assumption becomes relevant. Including the full inner-loop dynamics makes the system effectively fourth order, complicating controller design. To simplify the problem, the standard approach in the literature is to assume that the inner rate loop operates on a sufficiently faster timescale than the outer attitude loop.

[‡]This trade-off is intrinsic to the structure of the system and the absence of any compensatory phase lead in the proportional controller. Achieving higher closed-loop bandwidths would require a lead or PD compensator to provide additional phase margin.

If the inner-loop bandwidth is sufficiently higher than that of the attitude loop, it is reasonable to approximate the rate loop as ideal, i.e. $\omega(s) \approx \omega_{\text{cmd}}(s)$, which implies $H_\omega(s) \approx 1$. With this approximation, the attitude dynamics reduce to:

$$\boldsymbol{\theta}(s) = \frac{LC_\theta(s)}{s + LC_\theta(s)}\,\boldsymbol{\theta}_{\text{cmd}}(s) + \frac{1}{s + LC_\theta(s)}\left(\boldsymbol{f}_2(s) - \hat{\boldsymbol{f}}_2(s)\right), \tag{61}$$

where $\boldsymbol{f}_2(s)$ represents the true flight-path kinematics and $\hat{\boldsymbol{f}}_2(s)$ denotes its model estimate used in the controller. Due to the assumptions in the attitude control loop design, $\boldsymbol{f}_2(s) \neq \hat{\boldsymbol{f}}_2(s)$, but the mismatch is extremely small in magnitude and varies slowly with time. As a result, the induced tracking error remains bounded and does not compromise overall system performance. The closed loop from $\boldsymbol{\theta}_{\text{cmd}}(s)$ to $\boldsymbol{\theta}(s)$ is therefore:

$$\lim_{H_\omega(s)\to 1} H_\theta(s) = \frac{LC_\theta(s)}{s + LC_\theta(s)}. \tag{62}$$

The bandwidth separation assumption effectively decouples the rotational axes in the outer loop, allowing each attitude channel to be treated independently. While this simplification introduces a potential risk of instability, theoretical results show that exponential stability of the outer-loop states around their commanded values is still guaranteed if the inner-loop bandwidth is sufficiently high [19]. In practice, it is recommended that the rate loop bandwidth exceeds the attitude loop bandwidth by at least a factor of four to ensure that the approximation remains accurate [20]. For this single–integrator loop structure, exact second–order closed–loop dynamics with the desired natural frequency ω_θ and damping ratio ζ_θ can be obtained through pole placement. This is achieved using a proportional–with–roll–off controller of the form:

$$LC_\theta(s) = \frac{K_\theta}{1 + s/\omega_f} = \frac{K_\theta \omega_f}{s + \omega_f}.$$

Substituting the controller into the attitude dynamics yields the second–order pitch response:

$$H_\theta(s) = \frac{K_\theta \omega_f}{s^2 + \omega_f s + K_\theta \omega_f}, \qquad \omega_f = 2\zeta_\theta \omega_\theta, \qquad K_\theta = \frac{\omega_\theta}{2\zeta_\theta}.$$

These expressions directly link the controller parameters to the desired dynamic characteristics. Using the design values specified in the control requirements yields the second order closed-loop system:

$$H_\theta(s) = \frac{38.84}{s^2 + 11.22s + 38.84}, \quad \text{using} \quad LC_\theta(s) = \frac{38.84}{s + 11.22}. \tag{63}$$

The corresponding pole locations are summarized in Table 2. Unlike the rate control loop, however, the desired damping ratio is not fully achieved in the full closed loop. Increasing the time-scale separation would cause this value to approach 0.9, but such an adjustment would also slow down the overall system response. Hence, the trade-off lies between achieving the exact damping ratio or maintaining a faster response.

Table 2 Pole locations, natural frequency and damping for the full and the reduced attitude closed-loop system

System	Real ± Imag	ω_n, rad/s	ζ	Type
$H_{\theta_{\text{full}}}$	$-23.2 \pm 29.4i$	37.5	0.619	Actuator dynamics
	$-28.2 \pm 0.00i$	28.2	1.00	Rate pole
	$-3.66 \pm 4.45i$	5.77	0.635	Attitude poles
$H_{\theta_{\text{red}}}$	$-5.61 \pm 2.72i$	6.23	0.900	Attitude poles

Velocity Loop

The outer velocity loop regulates the geographic airspeed V_K, the heading angle χ_K, and the flight–path angle γ_K. This regulation is achieved indirectly by generating small changes in attitude and thrust, which the inner attitude and rate loops convert into linear accelerations.

The cascaded control structure used for velocity regulation is illustrated in Fig. 6. Commands originate from the position control loop, and the resulting command error is processed by the linear controller LC_V to produce the desired attitude and thrust setting rates. The error in the commanded derivatives is mapped through the control effectiveness matrix from Eq. (29), generating incremental attitude commands. These increments are added to the previous values and forwarded to the inner loops for bank angle and angle of attack, while the throttle command is sent directly to the engine.

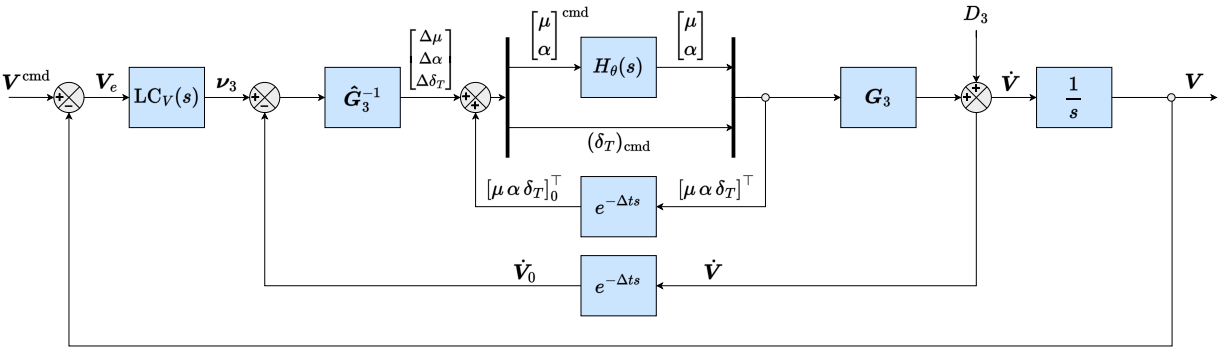


Fig. 6 Closed-loop block diagram of the velocity loop, assuming perfect attitude and angular rate tracking.

For the linear controller design, the closed–loop expression plays a central role. Under the assumption that the estimated control effectiveness matrix $\hat{\boldsymbol{G}}_3$ is exact and that disturbances are rejected sufficiently fast, the closed–loop relation is:

$$\boldsymbol{V}(s) = \frac{H_\theta(s)LC_V(s)}{s + H_\theta(s)LC_V(s)}\,\boldsymbol{V}_{\text{cmd}}(s), \qquad \lim_{H_\theta(s)\to 1} H_V(s) = \frac{LC_V(s)}{s + LC_V(s)}. \tag{64}$$

This relation applies to the angle-of-attack and bank-angle channels, where the inversion-based commands pass through inner dynamics represented by $H_\theta(s)$. The throttle channel does not include this term, since it acts directly on the propulsion system. When the inner loops are sufficiently fast such that $H_\theta(s) \approx 1$, the closed-loop map reduces to the expression shown on the right.

The velocity controller is obtained using the same second–order pole–matching approach applied in the attitude-loop design together with the bandwidth-separation and damping requirements, resulting in the closed-loop relation:

$$H_V(s) = \frac{2.428}{s^2 + 2.181s + 2.428}, \quad \text{using} \quad LC_V(s) = \frac{2.428}{s + 2.181}. \tag{65}$$

The poles of the reduced and complete transfer function, which includes the inner control loops, are given in Table 3:

Table 3 Pole locations, natural frequency and damping for the full and the reduced velocity closed-loop system

System	Real ± Imag	ω_n [rad/s]	ζ [–]	Type
$H_{V_{\text{full}}}$	$-23.2 \pm 29.4i$	37.5	0.62	Actuator poles
	$-28.2 + 0.00i$	28.2	1.00	Rate pole
	$-4.08 \pm 4.43i$	6.0	0.68	Attitude poles
	$-0.67 \pm 1.33i$	1.5	0.45	Velocity poles
$H_{V_{\text{red}}}$	$-1.09 \pm 1.11i$	1.6	0.70	

Position Loop

The position control loop determines the required heading χ_K and flight-path angle γ_K commands to track the commanded waypoints in latitude, longitude, and altitude. As shown in Fig. 7, these waypoint commands originate from the onboard guidance system, and the resulting command error is processed by the linear controller $LC_X(s)$ to

produce the desired changes in heading and flight-path angle. The errors in the commanded derivatives are then mapped through the inverse of the matrix $\boldsymbol{G}_4$ to account for spherical Earth effects, yielding intermediate commanded inputs. These are used in the inverse of Eq. (13) to generate the corresponding commands passed to the inner loops, which realize the actual heading and flight-path angle. The resulting control inputs are subsequently remapped through $\boldsymbol{G}_4$ to obtain the geodetic rate derivatives, which are integrated to produce the achieved tracking vector.

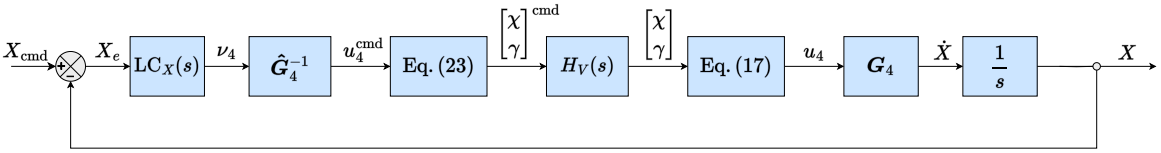


Fig. 7 Closed loop block diagram of the position control loop.

Analogous to the attitude loop, and assuming the estimated mapping $\hat{\boldsymbol{G}}_4$ is exact, the position closed–loop transfer function is given by the expression on the left below. This formulation incorporates the inner velocity dynamics through $H_V(s)$. When the velocity loop is sufficiently fast such that $H_V(s) \approx 1$, the closed–loop map reduces to the simplified first–order form shown on the right:

$$\boldsymbol{X}(s) = \frac{H_V(s)\, LC_X(s)}{s + H_V(s)\, LC_X(s)}\, \boldsymbol{X}_{\text{cmd}}(s), \qquad \lim_{H_V(s)\to 1} H_X(s) = \frac{LC_X(s)}{s + LC_X(s)}. \tag{66}$$

As with the attitude and velocity loops, the controller is derived using the same second-order pole-matching procedure applied to the inverted dynamics, together with the bandwidth-separation and damping requirements, producing the second-order closed-loop system:

$$H_X(s) = \frac{0.1517}{s^2 + 0.7012s + 0.1517}, \quad \text{using} \quad LC_X(s) = \frac{0.1517}{s + 0.7012}. \tag{67}$$

The poles corresponding to the reduced transfer function and the complete transfer function, which includes the inner control loops, are given in Table 4:

Table 4 Pole locations, natural frequency and damping for the full and the reduced position closed-loop system

System	Real ± Imag	ω_n, rad/s	ζ	Type
$H_{X_{\text{full}}}$	$-23.2 \pm 29.4i$	37.5	0.619	Actuator poles
	$-28.2 \pm 0.00i$	28.2	1.000	Rate pole
	$-4.08 \pm 4.43i$	6.02	0.677	Attitude poles
	$-0.74 \pm 1.28i$	1.48	0.502	Velocity poles
	$-0.28 \pm 0.27i$	0.39	0.716	Position poles
$H_{X_{\text{red}}}$	$-0.35 \pm 0.17i$	0.39	0.900	

The stability margins in Table 5 show that all control loops maintain sufficient gain, phase, and delay margins, consistent with well-damped and stable behavior. As expected, the rate loop has the highest crossover frequency and smallest delay margin, while the outer loops exhibit lower bandwidths and larger delay margins, preserving the desired time-scale separation. Overall, the ideal Continuous Time (CT) control structure achieves zero steady-state error and robust stability across all levels of the cascaded NDI–INDI architecture.

VI. Digital Control Design

In practice, FCSs must be implemented on digital FCCs, which operate in DT. A straightforward approach is to design a controller in CT and then transform it into a discrete controller using, for example, the bilinear transform or the matched pole–zero method. This approach, however, neglects the inherent properties of digital implementation: the

controller interacts with sampled signals, Zero-Order Hold (ZOH) devices, computation delays, and on-board filtering, none of which are captured in a purely CT design [12]. Only when the sampling interval Δt is very small can one argue that the discrete controller obtained from such a transformation accurately reproduces the performance of its CT counterpart.

A more sophisticated approach is the so-called modified CT controller design. Here, the discrete effects of sampling, holding, and computation are modeled explicitly as CT transfer functions and included in the synthesis process. The controller is then designed in this modified CT framework, so that the delay and hold dynamics are taken into account from the outset. Finally, the bilinear transform or a similar mapping is applied to obtain the actual digital controller for implementation. This approach ensures that the implemented digital controller preserves the intended performance characteristics even for practical sampling rates. Within this framework, the additional dynamics introduced by digital implementation must be modeled explicitly, as they influence both stability margins and transient response. The considered effects are:

1) **Sensor anti-aliasing filter.** Before sampling, sensor signals are band-limited to prevent aliasing, which folds high-frequency content above the Nyquist frequency into the measured bandwidth [12]. The anti-aliasing low-pass filter defined in Eq. (9) attenuates these components but introduces phase lag and gain reduction, both of which must be included in the plant model $G(s)$ for accurate analysis.
2) **Sampler and ZOH.** After sampling, the controller updates commands only at intervals of Δt, while the ZOH holds the actuator input constant between updates. This piecewise-constant reconstruction attenuates high-frequency content and introduces an effective delay of roughly $\Delta t/2$. The corresponding CT transfer functions are:

$$G_{ZOH}(s) = \frac{1 - e^{-\Delta t s}}{s}, \qquad G_{SH}(s) = \frac{1 - e^{-\Delta t s}}{s\,\Delta t}. \tag{68}$$

   The second expression includes the sampling action through normalization by the sampling interval.
3) **Computation delay.** The FCC requires finite time to read measurements, compute the control input, and send the command to the actuator. A conservative model assumes the worst case which is a full-sample computation delay of Δt, represented in CT as:

$$G_D(s) = e^{-\Delta t s}. \tag{69}$$

In addition to the modeled effects, real sensors may introduce further delays, as noted in Eq. (10). Because these delays are typically unknown at the design stage, they are not included in the modified CT controller synthesis. Instead, potential sensor delays are introduced in the high-fidelity simulation to evaluate how much delay the controller can tolerate.

The overall control structure, shown in Fig. 8, includes the key effects introduced by digital implementation. The controller $LC(s)$ operates together with the sample-and-hold element $G_{SH}(s)$, computation delay $G_D(s)$, and sensor anti-aliasing filter $G_a(s)$. Combined with the plant dynamics $G(s)$, these components form a CT representation of how the digital controller interacts with the aircraft.

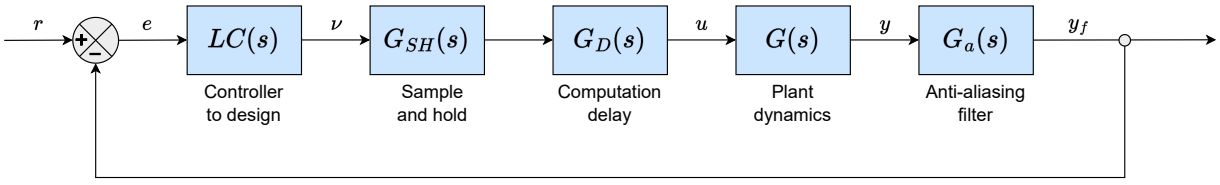


Fig. 8 CT control structure with digital implementation effects [12].

The transfer functions $G_{SH}(s)$ and $G_D(s)$ contain exponential terms and are therefore irrational, which is inconvenient for controller synthesis where rational transfer functions are preferred. A standard remedy is to approximate these exponentials with Padé expansions. The (m, n) Padé approximation represents a function by a ratio of two polynomials whose Taylor series matches the first $m + n$ terms of the original function [12]. Padé forms capture the key delay behavior, including phase lag, with low-order rational models suitable for analysis and simulation. The third-order approximations used here follow [12].

For the sample-and-hold and the computation delay transfer function, a third-order Padé approximation of the exponential yields:

$$G_{SH}(s) \approx \frac{1 - \frac{1}{14}(s\Delta t) + \frac{23}{840}(s\Delta t)^2 - \frac{1}{840}(s\Delta t)^3}{1 + \frac{3}{7}(s\Delta t) + \frac{1}{14}(s\Delta t)^2 + \frac{1}{120}(s\Delta t)^3}, \qquad G_D(s) \approx \frac{1 - \frac{1}{2}(s\tau_{CD}) + \frac{1}{10}(s\tau_{CD})^2 - \frac{1}{120}(s\tau_{CD})^3}{1 + \frac{1}{2}(s\tau_{CD}) + \frac{1}{10}(s\tau_{CD})^2 + \frac{1}{120}(s\tau_{CD})^3}. \tag{70}$$

It is important to note that the influence of digital implementation effects strongly depends on the sampling interval. For small sampling times, the impact of sample–and–hold behavior, computation delay, and sensor filtering is minor and can often be neglected. However, as the sampling interval increases, these effects introduce additional phase lag and attenuation that can noticeably degrade closed–loop performance. Consequently, explicitly accounting for these effects in the controller design is advisable to preserve the intended performance.

A representative block diagram illustrating the inclusion of discrete effects (shown in pink), as well as sensor delay and anti-aliasing effects, is presented in Fig. 9.

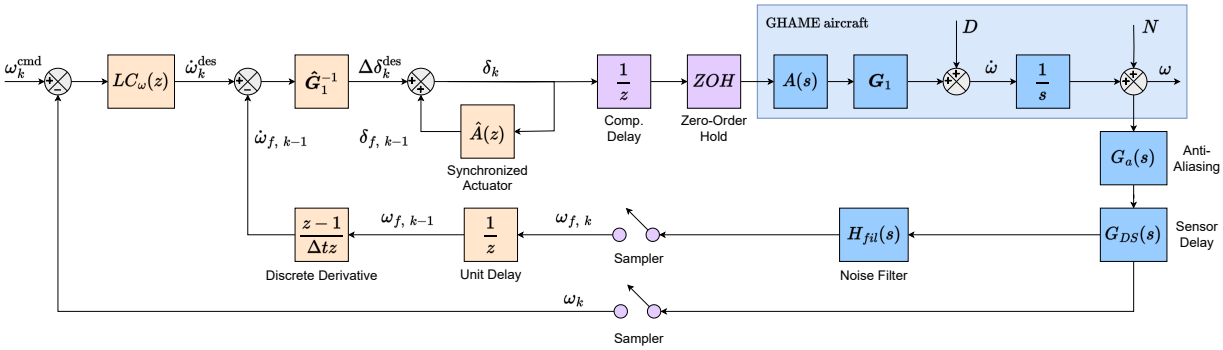


Fig. 9 DT INDI rate control loop including actuator dynamics, sensor dynamics, sampling, and noise filtering.

The digital effects on the rate control loop are now considered. When the control structure from Fig. 8 is applied to the angular rate loop described in Sec. V, the additional dynamics introduced by the sample-and-hold, computation delay, and anti-aliasing filter invalidate the previously calculated open-loop gain and phase margins.

The phase effects of the digital elements can be assessed directly at the crossover frequencies. A one–sample computational delay with $\Delta t = 0.01$ s introduces about $-7.7°$ of lag at the phase crossover frequency of 13.5 rad/s. The anti–aliasing filter contributes roughly $-4.9°$, and the ZOH adds about $-3.9°$, giving a total of approximately $-16.5°$. This reduces the CT phase margin from 67.6° to about 51.1° in the digital implementation. At the gain crossover frequency of 50 rad/s, the same mechanisms generate nearly $-60°$ of lag. Although the exact gain–margin reduction cannot be computed analytically, the additional phase clearly lowers the achievable margin. These reductions confirm that a sampling time of $\Delta t = 0.01$ s is not small enough for digital effects to be ignored.

Table 5 Comparison of stability margins for each control loop without and with digital effects (D)

Loop	Gain margin, dB	Phase margin, deg	Delay margin, s	Crossover, rad/s
Rate	14.3	67.6	0.0872	13.5
Rate (D)	7.67	51.2	0.0664	13.5
Attitude	13.5	59.7	0.315	3.3
Attitude (D)	10.9	57.9	0.301	3.35
Velocity	10.0	48.1	0.831	1.01
Velocity (D)	9.78	47.7	0.816	1.02
Position	12.8	62.4	5.19	0.21
Position (D)	12.7	62.4	5.19	0.21

The impact of discretization becomes smaller as the dynamics slow down and is therefore most relevant in the rate and attitude loops. Table 5 summarizes the resulting gain and phase margins for the continuous and digital cases. With the nominal CT gain from Eq. (58), the digital implementation yields a gain margin of 7.67 dB, a phase margin of 51.20°, and a delay margin of 0.066 s. The corresponding step response shows an overshoot of 5.2%, meaning the zero–overshoot requirement is no longer met. The reduction in stability is not severe, but it is significant enough that it should be considered when choosing controller parameters.

Table 6 Final stability margins obtained with the redesigned controllers from Eq. (71), based on the modified CT control design

Loop	Gain margin, dB	Phase margin, deg	Delay margin, s	Crossover, rad/s
Rate	12.3	67.3	0.148	7.95
Attitude	12.9	58.7	0.52	2.0
Velocity	9.83	47.8	1.37	0.61
Position	12.8	62.4	8.64	0.125

Two design paths can now be considered. The first option is to accept the reduced stability margins and the small overshoot as sufficient for operation, since the system remains stable with acceptable stability margin. The second option is to redesign the linear controllers with the digital effects explicitly included, thereby recovering the intended performance targets. This approach is adopted here to restore the zero-overshoot behavior of the rate loop and to improve the limited delay margins observed under digital implementation. The resulting four linear controllers obtained from this reoptimization are presented in Eq. (71), and their corresponding stability margins are summarized in Table 6:

$$LC_\omega = 7.9663, \quad LC_\theta(s) = \frac{13.96}{s+6.726}, \quad LC_V(s) = \frac{0.8726}{s+1.308}, \quad \text{and} \quad LC_X(s) = \frac{0.05454}{s+0.4204}. \tag{71}$$

To summarize the key characteristics of the developed cascaded controller, the control architecture consists of four nested loops: rate, attitude, velocity, and position. Each outer loop is tuned to operate at a lower bandwidth than the one inside it, ensuring that the inner dynamics are already attenuated within its operating range. This arrangement minimizes coupling between loops and simplifies both tuning and analysis. The corresponding loop transfer functions are shown in Fig. 10. The magnitude plot illustrates clear bandwidth separation among the loops. For clarity, reference lines at 0 dB and −180° are added to visualize crossover frequencies and phase margins.

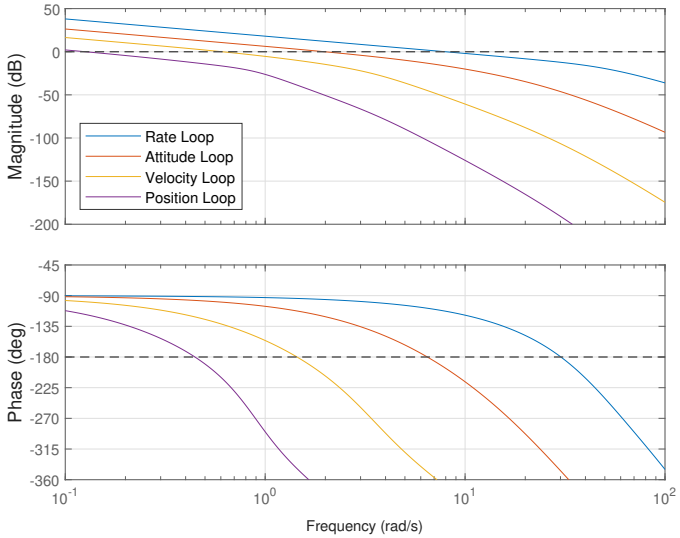


Fig. 10 Open loop Bode magnitude and phase for the rate, attitude, velocity, and position loops.

The parameters used in the modified CT control design are listed in Table 7. They include the selected bandwidths, damping ratios, filter and actuator characteristics, and timing values that define the dynamic properties of each loop. These parameters produce the frequency responses shown in Fig. 10.

Table 7 Design parameters used for control law development using the modified continuous control design.

Parameter	Symbol	Value	Unit
Filter corner freq.	ω_H	25	rad/s
Filter damping	ζ_H	1	-
Actuator nat. freq.	ω_{act}	50.0	rad/s
Actuator damping	ζ_{act}	0.707	-
Rate bandwidth	BW_ω	15.0	rad/s
Attitude damping	ζ_θ	0.9	-
Attitude bandwidth	BW_θ	3.75	rad/s

Parameter	Symbol	Value	Unit
Velocity nat. freq.	BW_V	0.9375	rad/s
Velocity damping	ζ_V	0.7	-
Position nat. freq.	BW_X	0.234375	rad/s
Position damping	ζ_X	0.9	-
Anti-aliasing freq.	ω_a	157.07	rad/s
Comp. delay	τ_{CD}	0.01	s
Sampling time	Δt	0.01	s

VII. Digital Implementation

Digital implementation is required because the FCS and the hierarchical controllers in Eq. (71) were developed in CT, whereas the FCC operates with a finite sampling interval. The effects of sampling and delays have already been incorporated in the digital control design of Sec. VI. The objective here is therefore to discretize the CT components that must execute on the FCC, including the derivative blocks and the hierarchical INDI controllers. All signals entering the FCC must be sampled accordingly. The aircraft dynamics, actuator models, sensor dynamics, and noise filters remain in CT, requiring the FCC outputs to interface with continuous-time subsystems. Although this interaction would nominally require a zero-order hold, Simulink manages the discrete–continuous rate transitions automatically, so no explicit conversion blocks are needed.

The final layout of the hierarchical control system is shown in Fig. 11. The diagram presents all five main components: the sampling of input signals into the FCC, followed by the four control loops described previously. This configuration represents the DT Simulink implementation of the architecture shown in Fig. 2.

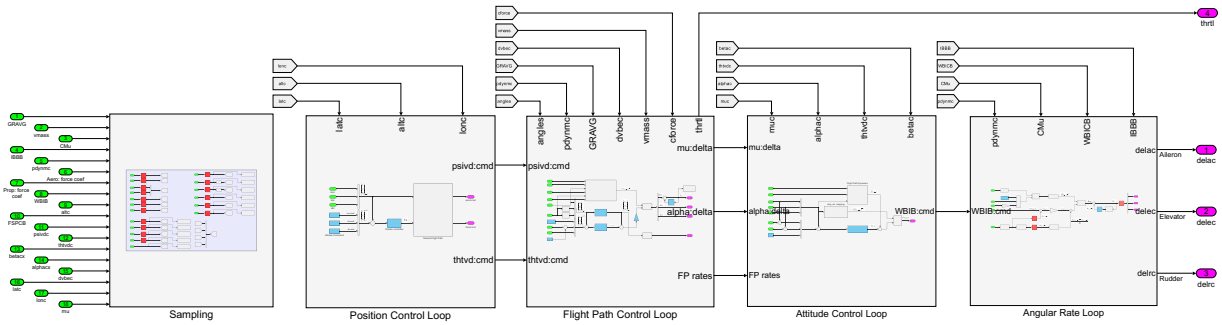


Fig. 11 DT hierarchical INDI control architecture implemented in Simulink, including the sampling subsystem.

All controllers are executed with a sampling time of $\Delta t = 0.01$ s. Simulations use a fixed-step solver to capture discretization effects, with the step size set to $\Delta t/25$, which corresponds to 4×10^{-4} s.

Position Control

The Simulink representation of the DT position control loop is shown in Fig. 12. The function implements the equations in Eq. (18) and Eq. (19). The green inputs represent sampled variables from the sampling subsystem, while the blue blocks correspond to the commanded way points and the DT position controller. The blue color indicates that

these elements operate at a defined sampling time. The pink blocks denote the outputs, which are the commanded flight path and heading signals passed to the next control loop.

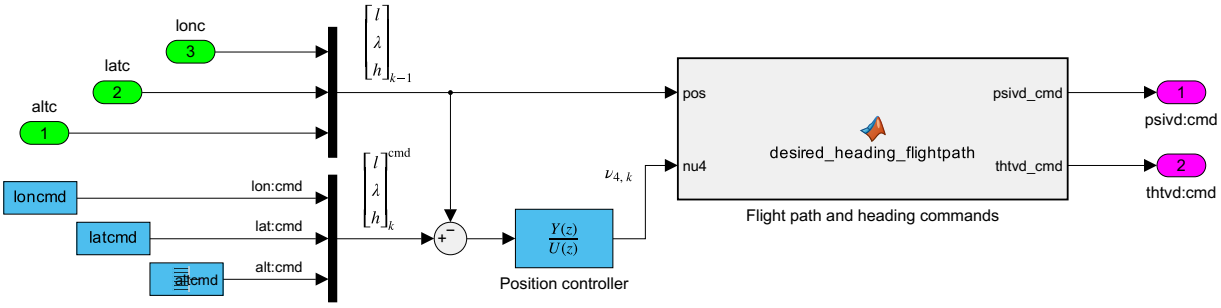


Fig. 12 Simulink implementation of the DT position control loop.

The DT position controller is obtained by discretizing the CT controller in Eq. (71). The Tustin transformation is used for this purpose, mapping the s-domain to the z-domain while preserving stability and accurately capturing the low-frequency dynamics. With a sampling period of $\Delta t = 0.01$ s, the resulting DT controller is:

$$LC_X(z) = \frac{4.0319 \times 10^{-5} + 4.0319 \times 10^{-5} z^{-1}}{1 - 0.99838\, z^{-1}}. \tag{72}$$

The DT controller maintains the dominant low-frequency behavior of the CT design, yielding comparable closed-loop performance within the operating bandwidth. Differences appear mainly at higher frequencies due to frequency warping and the finite sampling period.

Velocity and Flight Path Control

The DT velocity control loop is shown in Fig. 13. The inverse of Eq. (31) is computed for control allocation within this loop. The throttle actuator is assumed to track the commanded change within one sampling period, so a UD block provides a sufficient approximation for generating the incremental input for speed control. The remaining incremental inputs for commanded bank angle and angle of attack are passed to the attitude loop and combined with values from the previous sampling instant. The heading and flight-path derivatives are likewise provided to that loop for its control law. The speed command is set by a constant block defining the trimmed velocity, and a step block introduces heading changes when the hold-heading mode is active.

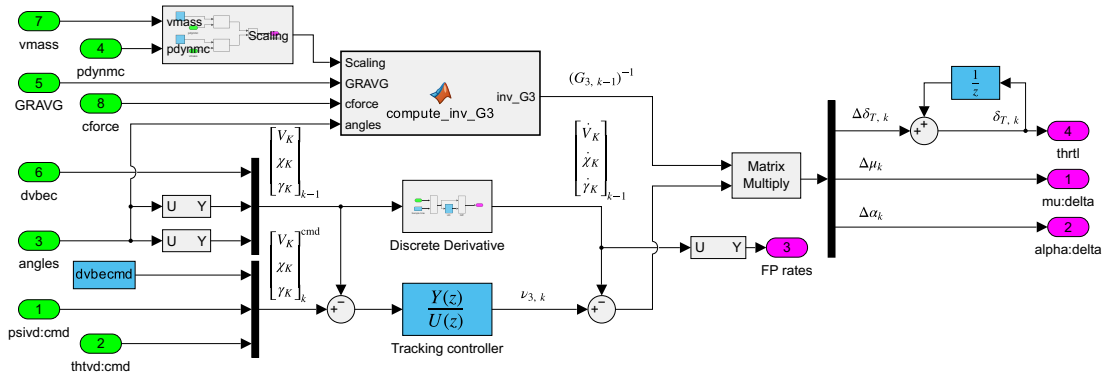


Fig. 13 Simulink implementation of the DT tracking control loop.

A key feature of this loop is that the derivatives required for the INDI control law are computed using the backward difference numerical differentiation method, as shown in Fig. 14.

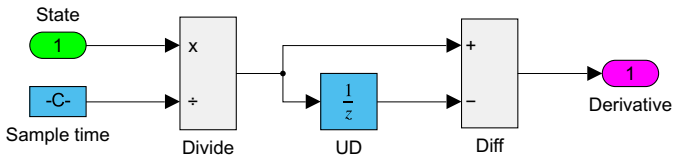


Fig. 14 Discrete differentiation algorithm used for computing derivatives.

Unlike the rate loop, no second–order noise filter is applied before differentiation. The INS model used here introduces errors only in angular rate and excludes tilt, velocity, and position errors. The resulting airspeed, flight path, and heading signals are therefore treated as noise-free, and additional filtering is not required.

The DT velocity controller is again obtained by discretizing the CT design in Eq. (71) using the Tustin transformation. With a sampling period of $\Delta t = 0.01$ s and normalized coefficients, the resulting discrete transfer function is:

$$LC_V(z) = \frac{0.0003297 + 0.0006595\, z^{-1} + 0.0003297\, z^{-2}}{1 - 1.9220\, z^{-1} + 0.9249\, z^{-2}}. \tag{73}$$

Attitude Control:

The attitude control loop is implemented using NDI, which eliminates the need for discrete derivative approximations and thereby simplifies implementation in Simulink. The overall structure of the loop is shown in Fig. 15. At each integration step, the terms $\boldsymbol{G}_2$ and $\boldsymbol{f}_2$ are computed according to Eq. (39) to perform the dynamic inversion. The attitude command is formed by combining the incremental control inputs from the preceding velocity loop with the sampled states provided by the INS. The sideslip command is fixed at zero to maintain coordinated flight, and the resulting rate command is passed to the rate control loop.

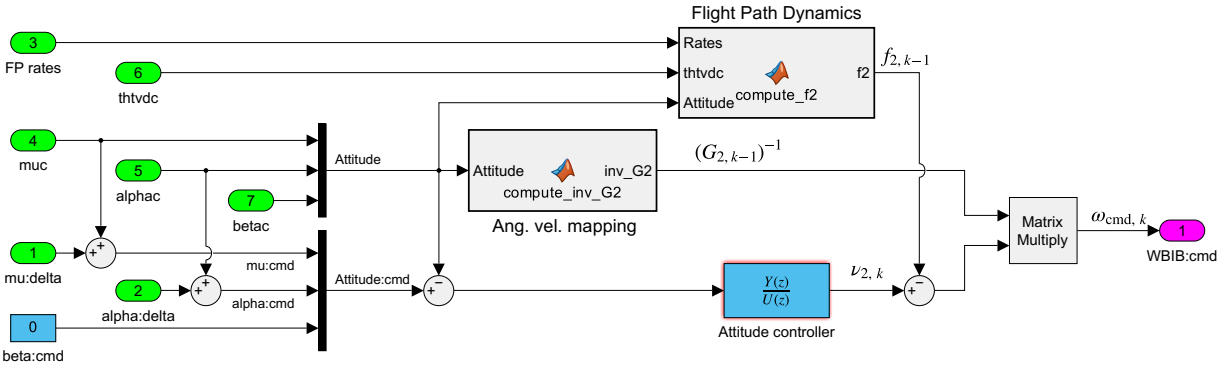


Fig. 15 Simulink implementation of the attitude control loop. The discrete controller $LC_\theta(z)$, obtained via Tustin discretization, is located in the linear control subsystem.

The attitude controller is obtained by discretizing the CT design in Eq. (71) using the Tustin transformation. With a sampling period of $\Delta t = 0.01$ s and normalized coefficients, the resulting discrete transfer function is:

$$LC_\theta(z) = \frac{0.1271 + 0.1271\, z^{-1}}{1 - 0.9113\, z^{-1}}. \tag{74}$$

Rate Control:

The rate control loop is implemented using INDI and therefore requires discrete derivatives. Unlike the velocity control loop, noise filtering is applied in this case and sensor dynamics are present, which introduces the need for actuator synchronization. The DT block diagram of the INDI rate controller, including sensor, noise, and actuator dynamics, is shown in Fig. 16 it is essentially the Simulink implementation of Fig. 9.

In the upper path, the control effectiveness matrix $\boldsymbol{G}_1$ is computed according to Eq. (45). The lower path contains the CT implementation of the sensor dynamics, representing anti-aliasing effects plus an unknown delay. The INS state is subsequently processed through a CT noise filter before being sampled to obtain the discrete derivatives. The angular rate state that is not differentiated bypasses the noise filter and is directly sampled. Another important aspect is the worst-case computational delay in the simulation, this effect is modeled using a UD block placed immediately before the control signal is sent to the actuators. For the rate control loop, the controller reduces to a constant linear gain, which requires no translation into DT.

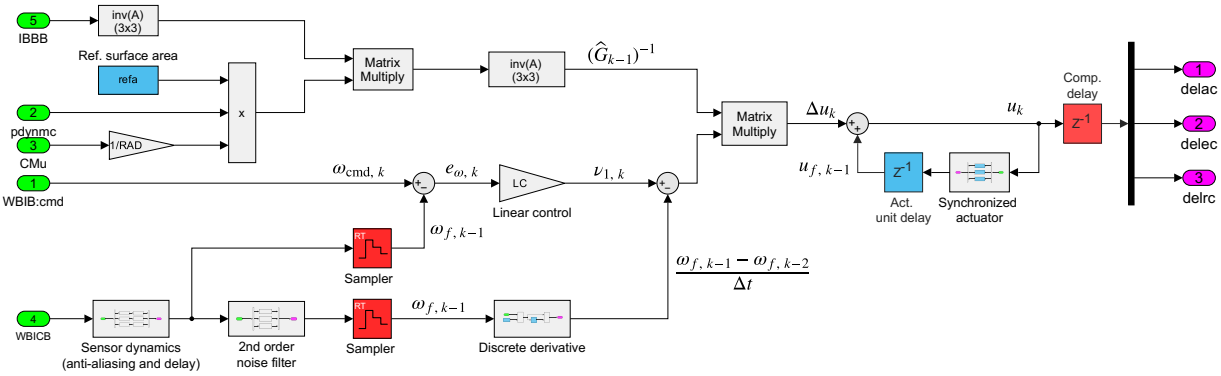


Fig. 16 Simulink implementation of the INDI subsystem for angular rate control, including sensor, noise, and actuator dynamics.

The key aspect here is the discretization of the actuator path, which includes the synchronization transfer functions. The corresponding discrete formulation for implementation within the digital control loop is presented next.

Discrete Synchronized Actuators:

The physical actuators operate in CT, but their representation within the FCC must be discrete. The CT formulation of the combined actuator, synchronization dynamics, and filtering is based on the sensor synchronization method defined in [21] and is expressed as:

$$\hat{A}(s) = A(s)\, H_{\mathrm{sync}}(s)\, H_{\mathrm{fil}}(s)\, e^{-\Delta t s}, \tag{75}$$

where $A(s)$ denotes the actuator dynamics, $H_{\mathrm{fil}}(s)$ represents the noise filter from Eq. (50), and $e^{-\Delta t s}$ models a one–sample delay aligning the actuator output with the previous control step. The term $H_{\mathrm{sync}}(s)$ represents the synchronization dynamics, capturing all phase and timing mismatches between the actuator and measurement paths. In this system, these correspond to $G_a(s)$ and $G_{SD}(s)$ from Eqs. (9) and (10), respectively. The synchronized simulation includes both. In the unsynchronized case, only $G_a(s)$ is applied, and the sensor delay is modeled as an uncertainty.

All components except the delay are combined in the CT domain prior to discretization, because the transformation from the s–domain to the z–domain is nonlinear and must, in general, be applied to the complete transfer function [22]:

$$z\{S(s)G(s)\} \neq z\{S(s)\}\, z\{G(s)\}. \tag{76}$$

Here, $z\{\cdot\}$ denotes the z–transform, and discretization is performed using the ZOH method rather than the Tustin method, since it reproduces the sampled-data behavior of actuators with constant input during each sampling period.

By separating the non delayed portion of Eq. (75), obtain $A(s)\, H_{\mathrm{sync}}(s)\, H_{\mathrm{fil}}(s)$. Using this expression directly inside the ZOH discretization gives:

$$H_c(z) = z\left\{\frac{1-e^{-\Delta t s}}{s}\, A(s)\, H_{\mathrm{sync}}(s)\, H_{\mathrm{fil}}(s)\right\}. \tag{77}$$

The exponential term $e^{-\Delta t s}$ corresponds directly to a UD in the z–domain, i.e. multiplication by z^{-1}. The final DT synchronized actuator is therefore:

$$\hat{A}(z) = H_c(z)\, z^{-1}. \tag{78}$$

The resulting discretized synchronized actuator dynamics can be expressed as follows:

$$\hat{A}(z) = \frac{z^{-2}\left(1.3389 \cdot 10^{-4} + 0.0023z^{-1} + 0.0039z^{-2} + 9.7198 \cdot 10^{-4}z^{-3} + 2.3291 \cdot 10^{-5}z^{-4}\right)}{1 - 3.1777z^{-1} + 3.9921z^{-2} - 2.4448z^{-3} + 0.7099z^{-4} - 0.0722z^{-5}}. \tag{79}$$

The final step is the initialization of the discrete synchronized actuator. To avoid spurious transients at $t = 0$, the actuator must start at the calculated trim input; otherwise, a mismatch between the trim condition and the actuator state would occur. In a DT transfer function block in Simulink, however, the trim deflection cannot be assigned directly as the initial output. The parameter *Initial states* instead specifies the contents of the internal delay registers of the filter. Consequently, the internal states must be computed such that the filter output equals the trim deflection u_{trim} at the start of the simulation.

Simulink realizes a DT transfer function in state-space form as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\,\boldsymbol{u}_k, \\ y_k &= \boldsymbol{C}\boldsymbol{x}_k + D\,\boldsymbol{u}_k, \end{aligned} \tag{80}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ are the filter states, $\boldsymbol{u}_k$ is the vector of control inputs, and y_k is the output. The matrices $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, D)$ are obtained from the actuator numerator and denominator coefficients using the MATLAB command `tf2ss`. At trim, the input is constant, $\boldsymbol{u}_k = \boldsymbol{u}_{\text{trim}}$. Requiring the state to remain constant, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, leads to the steady-state condition:

$$\boldsymbol{x}_0 = \left(\boldsymbol{I} - \boldsymbol{A}\right)^{-1}\boldsymbol{B}\,\boldsymbol{u}_{\text{trim}}, \tag{81}$$

with $\boldsymbol{I}$ the identity matrix of the same size as $\boldsymbol{A}$. Assigning the vector $\boldsymbol{x}_0$ to the actuator block as its initial state ensures that, at $t = 0$, the actuator outputs match the trimmed control deflections $\boldsymbol{u}_{\text{trim}}$. The corresponding initial states associated with the trimmed states are summarized in Table 8.

Table 8 Initial condition for discrete actuators

Condition	Initial state $\boldsymbol{x}_0$	δ_{trim}, deg	α_{trim}, rad	θ_{trim}, rad
Mach 3, 60,000 ft	[0, -12.21, 0]	[0.0, -5.1039, 0.0]	0.0602	0.0602

VIII. Simulation Results

Effect of Sensor Delay:

The effect of unsynchronized sensor delay is analyzed by progressively increasing the measurement delay while keeping the actuator dynamics and filters unchanged. The sensor delay is increased by 0.01 s for each simulation run. The corresponding altitude, flight path, and heading responses are shown in Figs. 17, 18, and 19. The effect of sensor delay is twofold. First, it causes a desynchronization between the angular acceleration and actuator paths, which directly disturbs the incremental feedback mechanism. Second, it introduces a pure delay in the angular rate feedback loop, reducing both the gain and phase margins. While proper synchronization can mitigate the first issue by aligning the actuator and measurement paths, it cannot eliminate the second effect.

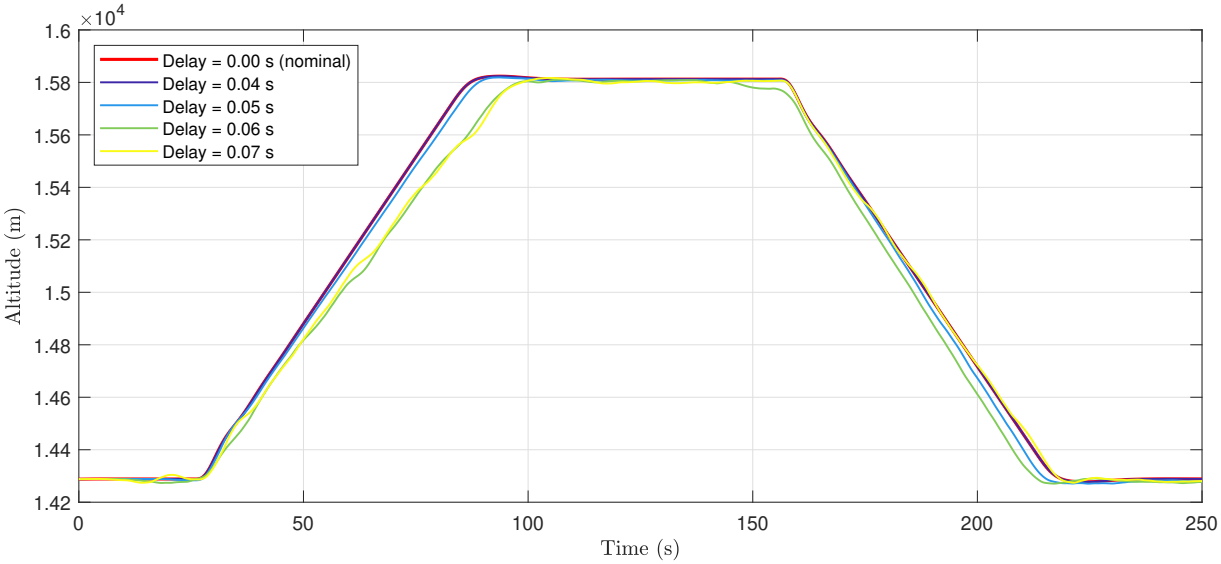


Fig. 17 Effect of increasing sensor delay on altitude tracking without sensor delay synchronization.

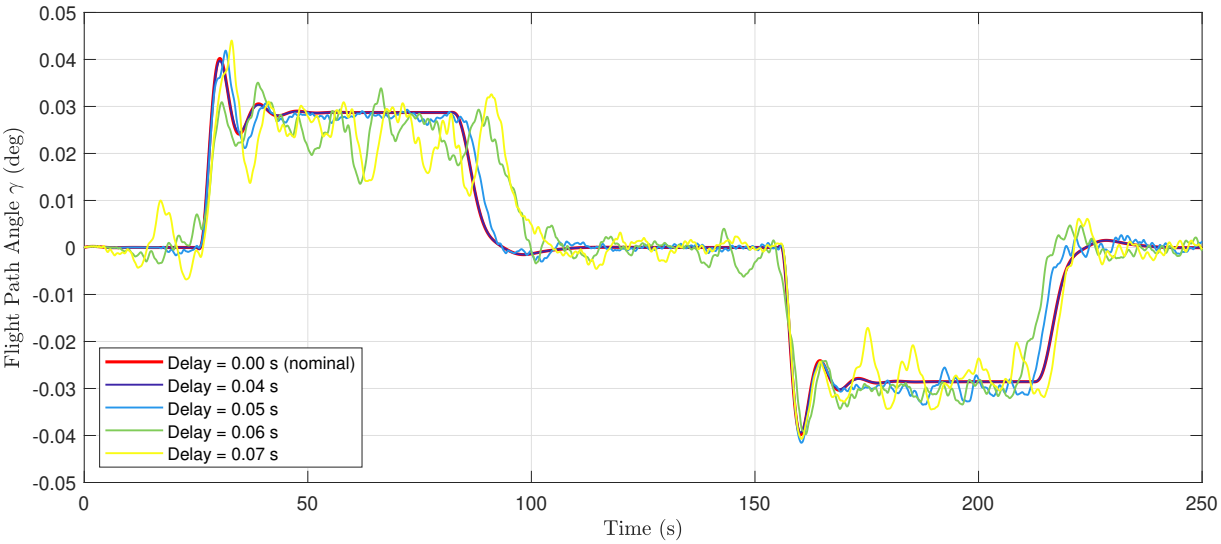


Fig. 18 Effect of increasing sensor delay on flight-path angle without sensor delay synchronization.

The controller maintains accurate tracking up to a sensor delay of approximately 0.04 s. Beyond this point, the response begins to deteriorate, as evident from Fig. 17. Although the altitude error remains moderate, the inner control loops exhibit growing oscillations that indicate the onset of instability, as shown in Fig. 18. Once the delay exceeds 0.04 s, the closed-loop dynamics degrade rapidly and eventually the simulation automatically terminates at a delay of 0.08 s when the angle of attack reaches 21°, which lies outside the aerodynamic data set. A similar trend appears in the heading and bank angle responses shown in Fig. 19 and Fig. 20. Oscillations develop once the sensor delay exceeds approximately 0.04 s, matching the onset observed in the altitude and flight path responses. However, the simulation remains stable for a longer duration and only terminates at 0.12 s. This delayed onset of instability is likely caused by the nonlinear effect introduced by the bank angle limitation, which constrains the commanded bank to within ±30°. Simulations performed without this constraint diverge earlier, at a delay of approximately 0.07 s.

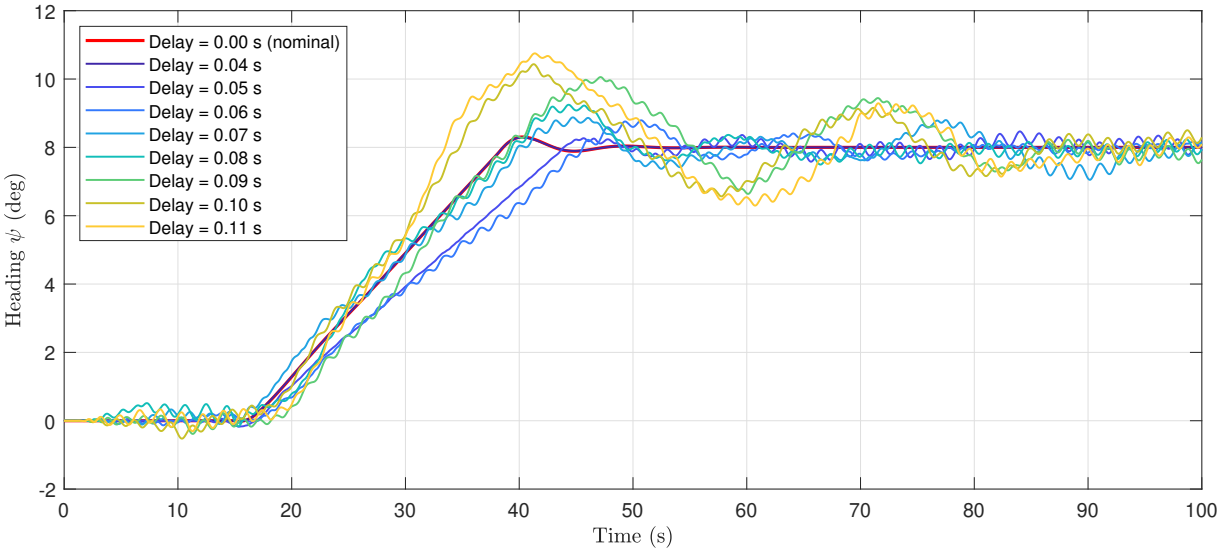


Fig. 19 Effect of increasing sensor delay on heading response without sensor delay synchronization.

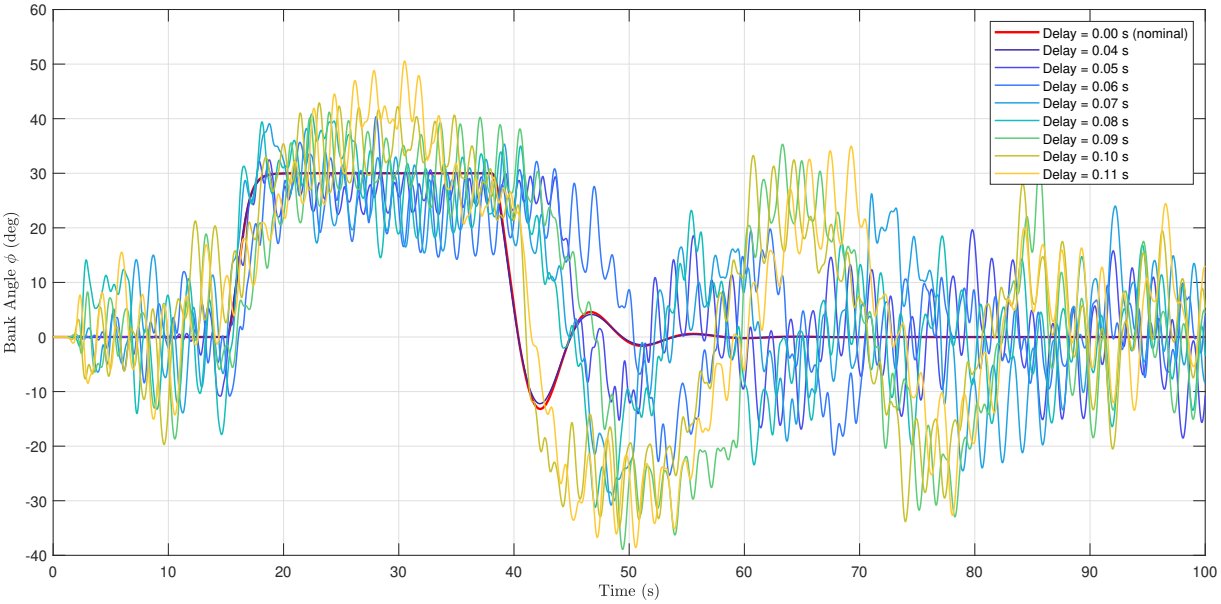


Fig. 20 Effect of increasing sensor delay on bank angle response without sensor delay synchronization.

Synchronized Time Delay

When sensor delay synchronization is not applied, the effective delay margin is only 0.04 s, far below the values in Table 6. Practical implementations therefore require sensor delay synchronization to compensate for timing offsets between the measurement and actuation paths. The idea is to introduce an equivalent delay in the actuator or reference signal so that both signals are aligned when processed by the controller, ensuring consistent information despite sensor latency. The following analysis presents results obtained with synchronized delays.

The synchronized altitude-response results are shown in Fig. 21 and Fig. 22. With synchronization enabled, the controller tolerates delays up to approximately 0.13 s before degradation occurs, more than twice the unsynchronized limit. At 0.14 s, sustained oscillations drive the angle of attack outside its valid aerodynamic range, causing the

simulation to terminate; the system is close to divergence. This follows from the continued erosion of gain and phase margins in the rate loop as the pure delay increases. The critical delay of 0.14 s matches the rate-loop delay margin in Table 6, showing that synchronization extends the stable delay range to the theoretical limit for the longitudinal case.

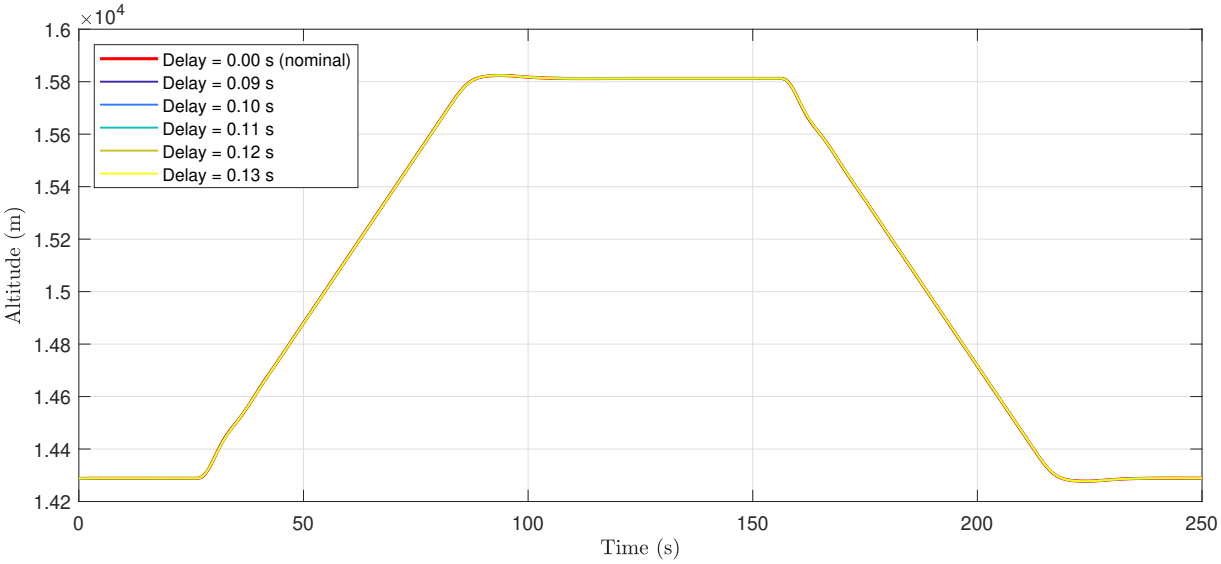


Fig. 21 Effect of increasing sensor delay on altitude tracking with sensor delay synchronization.

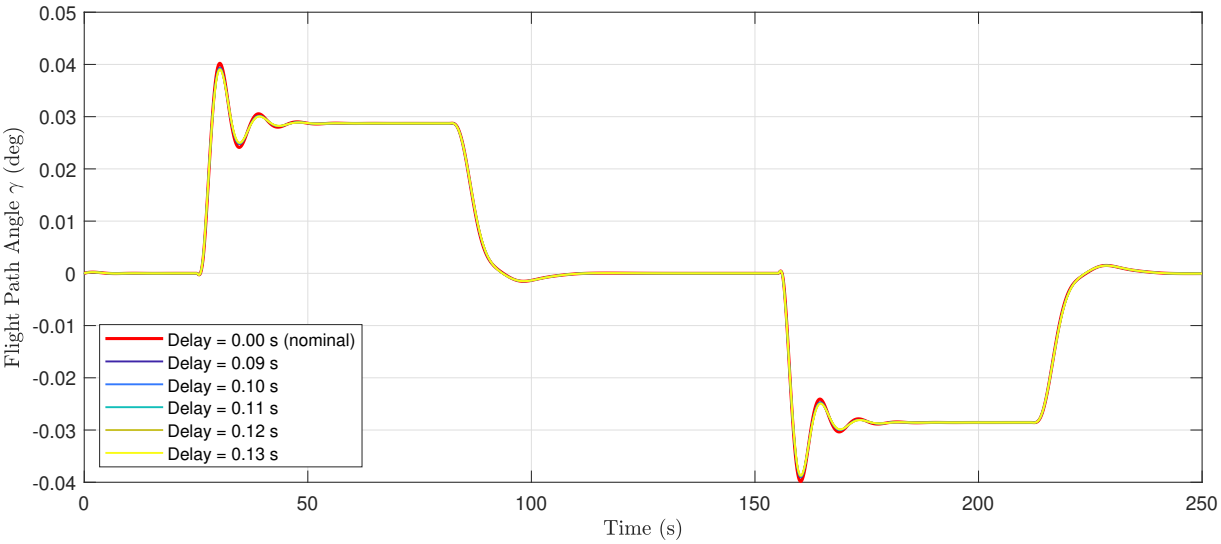


Fig. 22 Effect of increasing sensor delay on flight-path angle tracking with sensor delay synchronization.

For the heading command case shown in Fig. 23 and Fig. 24, the response differs notably from the longitudinal case. Oscillations appear at a sensor delay of about 0.08 s, much earlier than predicted by the calculated delay margin. This indicates that the lateral–directional loop is considerably more sensitive to delay. Beyond this point, the system still tracks the heading command but with large oscillations. The apparent stability up to around 0.16 s results mainly from nonlinear effects, particularly the roll command limit of ±30°, which prevents full divergence. Without this limit, the simulation would fail at smaller delays.

The early onset of oscillations indicates that a core assumption of the INDI formulation may not be fully satisfied for the lateral–directional dynamics of the GHAME vehicle. Examination of the airframe provides some clues. The

GHAME's slender body, low–aspect–ratio wings, and tight wing–body integration yield an unusually low roll moment of inertia, producing exceptionally fast roll dynamics. As a result, the lateral states evolve on time scales close to those of the control inputs rather than being clearly separated as assumed in the incremental formulation. Under these conditions, the system departs from the ideal chain–of–integrators behavior on which INDI relies. The measured acceleration increment is then affected not only by the control input but also by the natural evolution of the states, reducing the validity of the quasi–static mapping between input and acceleration. This coupling introduces additional phase lag in the feedback path and plausibly explains the premature oscillations observed in simulation.

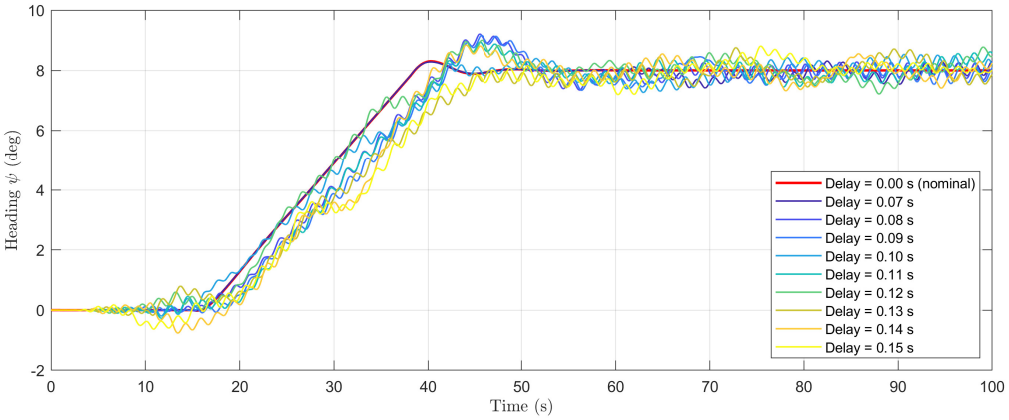


Fig. 23 Effect of increasing sensor delay on heading response with sensor delay synchronization.

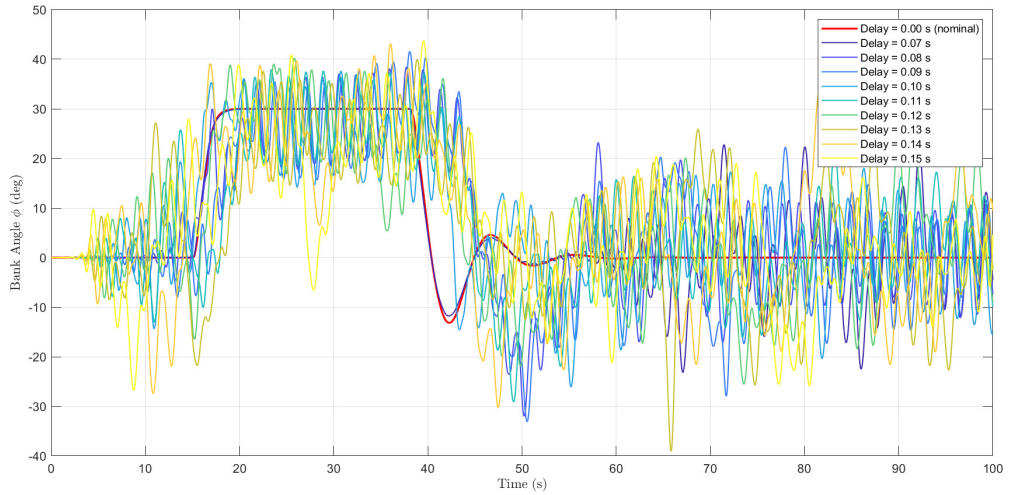


Fig. 24 Effect of increasing sensor delay on bank angle response with sensor delay synchronization.

If that is the case, the linear model used for design no longer provides an accurate representation of the system. Consequently, gain and phase margins derived from the linearized model are not predictive of the actual closed–loop behavior. Retuning the controller gains can slow the response and give the impression of increased robustness, yet instability still occurs at delay values inconsistent with linear predictions. The discrepancy is therefore structural rather than tuning related.

Conclusion

The analysis demonstrates that the NDI–INDI control architecture performs well for GHAME when no sensor delay is present. Introducing unsynchronized delay causes the system to fail rapidly, confirming that misaligned measurement and actuation paths severely degrades the incremental control law formulation. Synchronizing the delay substantially increases the admissible delay margin and is therefore essential for reliable performance.

The delay study also exposed a deeper structural limitation in the lateral–directional axis. Even with synchronized delays, the lateral subsystem remains far more sensitive than the longitudinal subsystem. This arises from the intrinsic characteristics of GHAME: low roll inertia and very fast lateral dynamics leave insufficient separation between the control input and the airframe response. As a result, the system cannot approximate the chain of integrators required by INDI. Incremental inversion becomes only approximate, and linear stability margins lose predictive value. In contrast, the longitudinal axis exhibits adequate separation between control and dynamics, satisfies the INDI assumptions, and supports accurate inversion and stable control. Overall, the study shows that while NDI–INDI performs effectively for the longitudinal motion, particularly when sensor delays are synchronized and the lateral axis suffers from a fundamental structural limitation. In its current form, INDI should only be applied to GHAME's lateral–directional dynamics with caution. Only physical or implementation-level modifications, such as higher actuator bandwidth, faster sampling, or increased roll inertia, could establish the conditions required for reliable inversion-based control.

Appendix

Table 9 GHAME Geometric, Mass Properties, and WGS-84 Physical Constants (SI Units)

Parameter	Symbol	Constant	Takeoff	Fuel Burnout	Unit
Reference area	S	557.42	–	–	m^2
Reference chord	c	22.86	–	–	m
Reference span	b	24.38	–	–	m
Mass	m	–	136,080	54,432	kg
Moment of inertia	I_{xx}	–	1.573×10^6	1.180×10^6	$kg{\cdot}m^2$
Moment of inertia	I_{yy}	–	31.60×10^6	19.25×10^6	$kg{\cdot}m^2$
Moment of inertia	I_{zz}	–	32.54×10^6	20.20×10^6	$kg{\cdot}m^2$
Product of inertia	I_{xz}	–	0.380×10^6	0.240×10^6	$kg{\cdot}m^2$
Earth gravitational const.	GM	3.986005×10^{14}	–	–	m^3/s^2
Mean Earth radius	R_0	6.370987308×10^6	–	–	m
Earth rotation rate	$\omega_\oplus$	7.292115×10^{-5}	–	–	rad/s
Stand. gravity at sea level	g_0	9.80675445	–	–	m/s^2

References

[1] Schmidt, D., and Velapoldi, J., *Flight Dynamics and Feedback Guidance Issues for Hypersonic Air-Breathing Vehicles*, American Institute of Aeronautics and Astronautics Inc., 1999. https://doi.org/10.2514/6.1999-4122.

[2] Gilbert, M. G., Heeg, J., and Pototzky, A. S., "The application of active controls technology to a generic hypersonic aircraft configuration," Tech. Rep. NASA-TM-101689, NASA, 1990. URL https://ntrs.nasa.gov/citations/19900010755.

[3] Smith, P., "A Simplified Approach to Nonlinear Dynamic Inversion Based Flight Control," *23rd Atmospheric Flight Mechanics Conference*, AIAA, 1998, pp. 762–770. https://doi.org/10.2514/6.1998-4461, aIAA Paper 1998-4461.

[4] Bowers, A. H., et al., "A Generic Hypersonic Aerodynamic Model Example (GHAME)," Technical report, NASA Dryden Flight Research Facility, 1989.

[5] White, D. A., Bowers, A., Iliff, K., and Menousek, J., "Flight, Propulsion, and Thermal Control of Advanced Aircraft and Hypersonic Vehicles," *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, Multiscience Press, Inc., New York, NY, 1992, pp. 357–465.

[6] Zipfel, P. H., *Modeling and Simulation of Aerospace Vehicle Dynamics*, 3rd ed., AIAA, Reston, Virginia, 2014. https://doi.org/https://doi.org/10.2514/4.102509.

[7] Goz, E., "Robust Multi-Objective H-Infinity Control of a Generic Hypersonic Vehicle," Master's thesis, Delft University of Technology, 2024.

[8] Autenrieb, J., "Data Fusion-Based Incremental Nonlinear Model Following Control Design for a Hypersonic Waverider Configuration," *AIAA SciTech 2023 Forum*, American Institute of Aeronautics and Astronautics, 2023. https://doi.org/10.2514/6.2023-1997.

[9] Pfeifle, O., and Fichter, W., "Cascaded Incremental Nonlinear Dynamic Inversion for Three-Dimensional Spline-Tracking with Wind Compensation," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 8, 2021, pp. 1559–1571.

[10] Juliana, S., Chu, Q. P., Mulder, J. A., and van Baten, T. J., "Flight Control of Atmospheric Re-entry Vehicle with Nonlinear Dynamic Inversion," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004. https://doi.org/https://doi.org/10.2514/6.2004-5330.

[11] Smeur, E. J. J., de Croon, G. C. H. E., and Chu, Q., "Cascaded Incremental Nonlinear Dynamic Inversion for MAV Disturbance Rejection," *Control Engineering Practice*, Vol. 73, 2018, pp. 79–90. https://doi.org/10.1016/j.conengprac.2018.01.003.

[12] Stevens, B. L., Lewis, F. L., and Johnson, E. N., *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*, 3rd ed., John Wiley & Sons, Hoboken, NJ, 2016.

[13] Mooij, E., *Re-entry Systems*, Springer, 2024. https://doi.org/10.1007/978-3-031-62174-1.

[14] Bacon, B., Ostroff, A., and Joshi, S., "Reconfigurable NDI controller using inertial sensor failure detection and isolation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 37, No. 4, 2001, pp. 1373–1383. https://doi.org/https://doi.org/10.1109/7.976972.

[15] Bacon, B., Ostroff, A., and Joshi, S., "Nonlinear dynamic inversion reconfigurable controller utilizing a fault tolerant accelerometer," *19th Digital Avionics Systems Conference*, Vol. 2, 2000. https://doi.org/https://doi.org/10.1109/DASC.2000.884920.

[16] Mooij, E., *The Motion of a Vehicle in a Planetary Atmosphere*, Delft University Press, The Netherlands, 1997.

[17] Snell, S. A., Enns, D. F., and Garrard, W. L., "Nonlinear Inversion Flight Control for a Supermaneuverable Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 976–984. https://doi.org/https://doi.org/10.2514/3.20932.

[18] Smeur, E. J. J., Chu, Q. P., and de Croon, G. C. H. E., "Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Air Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 3, 2016, pp. 450–461. https://doi.org/https://doi.org/10.2514/1.G001490.

[19] Schumacher, C., and Khargonekar, P. P., "Stability Analysis of a Missile Control System with a Dynamic Inversion Controller," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 3, 1998, pp. 508–515.

[20] Naidu, D. S., and Calise, A. J., "Singular Perturbations and Time Scales in Guidance and Control of Aerospace Systems: A Survey," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 6, 2001, pp. 1057–1078.

[21] Steffensen, R., Steinert, A., and Smeur, E. J. J., "Nonlinear Dynamic Inversion with Actuator Dynamics: An Incremental Control Perspective," *Journal of Guidance, Control, and Dynamics*, Vol. 46, No. 4, 2022, pp. 709–717. https://doi.org/https://doi.org/10.2514/1.G006813.

[22] Nise, N. S., *Control Systems Engineering*, 7th ed., Wiley, Hoboken, NJ, 2015.